A Novel Service Discovery Model for Decentralised Online Social Networks

Bo Yuan

Doctor of Philosophy 2017

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

List of Abbreviations and Acronyms

AQ	Advertisement Query
BFS	Breadth-First Search
DFS	Depth-First Search
DHT	Distributed Hash Table
DOSN	Decentralised Online Social Network
F2F	Friend to Friend
GQ	General Query
HQ	Heartbeat Query
LKI	Local Knowledge Index
LSI	Local Service Index
NFH	Node Fingerprint Hash
NSI	Network Shortcut Index
OSN	Online Social Network
P2P	Peer-to-Peer
RQ	Report Query
SDOSN	Self-organised Decentralised Online Social Network
SH	Status Homophily
TTL	Time to Live
URI	Uniform Resource Identifier
VH	Value Homophily
VQ	Virgin Query
VSM	Vector Space Model

DECLARATION

The research works in the dissertation were carried out in the College of Engineering and Technology at University of Derby, under supervision of Professor Lu Liu and Professor Nick Antonopoulos. This is to declare that the work stated in this dissertation was done by the author, and no part of the dissertation has been submitted in a dissertation form to any other university or similar institution. No human or animal participation have been included in this research and the research presented in this dissertation has been ethically approved. The author confirms that appropriate credit has been given within the thesis where reference has been made to the work of others. Parts of this dissertation have previously appeared in the papers listed in the primary list of publications.

Bo Yuan

University of Derby

2017

LIST OF PUBLICATIONS

Journals

- **B. Yuan**, L. Liu and N. Antonopoulos, "Efficient Service Discovery in Decentralised Online Social Networks" 2017 Future Generation Computer Systems [J] http://dx.doi.org/10.1016/j.future.2017.04.022. (*Impact Factor 3.99*).
- M. Ahmed, L. Liu, J. Hardy, **B. Yuan**, and N. Antonopoulos, "An efficient algorithm for partially matched services in internet of services," Personal and Ubiquitous Computing, vol. 20, pp. 283-293, 2016 Springer.(*Impact Factor 2.39*)
- J. Panneerselvam; J. Hardy; L. Liu; **B. Yuan**; N. Antonopoulos, "Mobilouds: An Energy Efficient MCC Collaborative Framework with Extended Mobile Participation for Next Generation Networks," in IEEE Access, vol.PP, no.99, pp.1-1 doi: 10.1109/ACCESS.2016.2602321. (*Impact Factor 3.24*)

Conferences

- W. Xu, Y. Guo, L. Shi, L. Liu, **B. Yuan**, "A Search Strategy for Social Resource in Decentralized Social Networks", in the Fifth International Conference on Advanced Cloud and Big Data (CBD), 2017 IEEE.
- **B. Yuan**, L. Liu, and N. Antonopoulos, "Efficient service discovery in decentralized online social networks," in Proceedings of the 3rd IEEE/ACM International Conference on Big Data Computing, Applications and Technologies, 2016, pp. 73-78.
- **B. Yuan**, L. Liu and N. Antonopoulos, "A Self-organised Architecture for Efficient Service Discovery in Future Peer-to-Peer Online Social Networks," 2016 IEEE Symposium on Service-Oriented System Engineering (SOSE), Oxford, United Kingdom, 2016, pp. 415-422.
- N. Whittington, L. Liu, **B. Yuan**, and M. Trovati, "Investigation of energy efficiency on cloud computing," in Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on, 2015, pp. 2080-2087.

- M. Ahmed, L. Liu, **B. Yuan**, M. Trovati, and J. Hardy, "Context-Aware Service Discovery and Selection in Decentralized Environments," in Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), 2015 IEEE International Conference on, 2015, pp. 2224-2231.(*Best paper award*)
- J. Panneerselvam, L. Liu, N. Antonopoulos, and **B. Yuan**, "Workload analysis for the scope of user demand prediction model evaluations in cloud environments," in Proceedings of the 2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing, 2014, pp. 883-889.
- M. Ahmed, L. Liu, J. Hardy, and **B. Yuan**, "An Efficient Algorithm for Partially Matched Web Services Based on Consumer's QoS Requirements," in Utility and Cloud Computing (UCC), 2014 IEEE/ACM 7th International Conference on, 2014, pp. 859-864.
- Paradowski, L. Liu, and **B. Yuan**, "Benchmarking the performance of openstack and cloudstack," in Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC), 2014 IEEE 17th International Symposium on, 2014, pp. 405-412.

ABSTRACT

Online social networks (OSNs) have become the most popular Internet application that attracts billions of users to share information, disseminate opinions and interact with others in the online society. The unprecedented growing popularity of OSNs naturally makes using social network services as a pervasive phenomenon in our daily life. The majority of OSNs service providers adopts a centralised architecture because of its management simplicity and content controllability. However, the centralised architecture for large-scale OSNs applications incurs costly deployment of computing infrastructures and suffers performance bottleneck. Moreover, the centralised architecture has two major shortcomings: the single point failure problem and the lack of privacy, which challenges the uninterrupted service provision and raises serious privacy concerns.

This thesis proposes a decentralised approach based on peer-to-peer (P2P) networks as an alternative to the traditional centralised architecture. Firstly, a self-organised architecture with self-sustaining social network adaptation has been designed to support decentralised topology maintenance. This self-organised architecture exhibits small-world characteristics with short average path length and large average clustering coefficient to support efficient information exchange. Based on this self-organised architecture, a novel decentralised service discovery model has been developed to achieve a semantic-aware and interest-aware query routing in the P2P social network. The proposed model encompasses a service matchmaking module to capture the hidden semantic information for query-service matching and a homophily-based query processing module to characterise user's common social status and interests for personalised query routing. Furthermore, in order to optimise the efficiency of service discovery, a swarm intelligence inspired algorithm has been designed to reduce the query routing overhead. This algorithm employs an adaptive forwarding strategy that can adapt to various social network structures and achieves promising search performance with low redundant query overhead in dynamic environments. Finally, a configurable software simulator is implemented to simulate complex networks and to evaluate the proposed service discovery model. Extensive experiments have been conducted through simulations, and the obtained results have demonstrated the efficiency and effectiveness of the proposed model.

ACKNOWLEDGEMENTS

First of all, I would like to acknowledge the valuable guidance of my supervisor Professor Lu Liu. He is one of the most supportive and helpful supervisors who I have ever come across in my academic life. Under his tutelage, I have learned the value of research and importance of collaboration. His abundant expertise and insightful guidance served as a fundamental pillar of my work. His encouragement and patience always kept me in a positive upward research trajectory. I feel a great sense of pride and honour for being his student. Further, I would like to thank Professor Nick Antonopoulos who is my second supervisor. He provided much helpful advice for my work and steered me to meet the highest expectations of scholarly research.

I am also very thankful to Mr James Hardy and Mr John Panneerselvam for their helpful feedback that allowed me to polish the language of this dissertation. I also want to extend my gratitude to all colleagues for being such great friends and collaborators in the department and the University of Derby. Furthermore, thanks for all the administrative staff in the College of Engineering and Technology for their advice and practical support

Finally, I would like to make a special acknowledgement to my family, specifically my wife Wenxia Yan, my father Hongqi Yuan, my mother Shufen Li and my sister Yuan Yuan. Thanks for their constant love and unwavering belief in me at all times.

1 INTRODUCTION

The drastic development of the Internet from its original communication purpose and content provision to a collective user-created intelligent data platform has availed provisions and supplements for increased and intensive computing and storage applications, which has facilitated users with constant access and engagement to computation resources. Online social networks (OSNs) connect millions of Internet users and have played a principal role in changing the world into a new era of globalisation, i.e. by making the way for online society. OSNs websites, such as Facebook, Twitter, Google Plus, Myspace and Flickr have become immensely 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 the future years [1]. Most OSNs sites provide free storage provisions for users to share their social content. The data warehouse of Facebook, the largest social website, reported a storage of around 300 PB of Hive data using a cloud system, with an incoming daily rate of about 600 TB user-generated content [2]. In simple words, social networks are generating the data at a rate and volume that any IT-centric organisations have never faced before.

To add to this challenge, the generated data are extremely heterogeneous and produced in various formats, including plain text, tags, image, video and geographic location etc. Moreover, the widespread diffusion of smart mobile devices such as smart phones, personal digital assistant (PDA), tablets are boosting the pervasive use of OSNs services in our daily lives. The user's mobility has become one of the key features in the design of OSNs applications. Owing to the affordability of the mobile devices and the growing number of daily users staying connected, OSNs are witnessing increased amounts of data from these hand-held devices. Clearly, OSNs are dealing with data in the range of Big Data which urges the development of novel computing models and new types of architecture in order to manage the sheer volume of data as well as the user's mobility.

1.1 Background and Motivations

The majority of the mainstream OSNs is designed in a logically centralised architecture and uses the traditional client-server model due to its design simplicity and controllability. In the centralised architecture, social network users communicate through a central entity that provides the necessary services and resources for computation, storage and communication. The main advantages of this architecture include higher availability of user profiles, ease of global data management and the higher efficiency of establishing online interconnectivity among users. However, centralised social networking applications incur costly deployments of computing infrastructures in order to provide the uninterrupted and acceptable range of services to millions of users. Cloud computing is an effective platform 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 a decade of the development phase, cloud computing is maturing and effectively addressing the requirement of big data analytics challenges. Therefore, OSNs service providers host their applications on cloud platforms to effectively exploit the scalable computing and storage services to process their data. Though cloud computing has managed to reduce the underlying infrastructure costs by facilitating the OSNs service providers with the provisions of on-demand rapid scaling of computing resources upon necessary, it still incurs significant costs to support millions of user requests in a centralised fashion. The centralised architecture also suffers two long-recognised shortcomings such as the lack of privacy and the single point of failure.

1.1.1 Privacy Issue

Centralised OSNs provide the platform for varied social interactions and facilitate the sharing of information with friends using cloud-based databases under the control of a single authority, namely the social networking service provider. In this sense, all personal data are usually stored in the provider's repository and users are susceptible to loss of control over their private data. It is obvious that users often want to restrict the access privileges to their personal information for third parties. However, as per the terms and conditions of using OSNs services, users must waive ownership of their sensitive information and shared resources to the service providers. For example, Facebook, the most popular OSNs website, allows user to maintain ownership of their data, but its terms of service state that users should grant Facebook "*a non-exclusive, transferable, sub-licensable, royalty-free, worldwide license to use any IP content that you post on or in connection with Facebook*" [4]. Such a kind of agreement is common with most of the centralised OSNs websites. OSNs providers store extensive amounts of user-created data and often claim full ownership all the data that include identity, personal photos and videos, social relationships and contact information and even private messages to friends etc. In addition, user's browsing history, social behaviours and daily activities are also recorded. Nowadays, with increasing popularity of big data analytics, the social network data are highly valuable to many stakeholders such as advertisement marketers, governments, researchers and data analytics companies [5-7]. Although the data may be anonymised by removing user's identities prior to transferring data for third-parties, it has been reported [8-10] that some techniques can be used to re-identify user' sensitive attributes. OSNs providers reserve the right of how to use the collected user data at their own discretion to generate revenue through data mining or targeted advertisement [11]. The unspecified use of user's sensitive data may violate the trust that users attribute to these providers [12].

This situation introduces a long-recognised dilemma between privacy advocates and OSNs service providers. On the one hand, users desire an integrative social networking service that can provide rich social information with easy access and little management of a large amount of personal data. However, this requires personal information to be stored in service provider's cloud infrastructures for centralised management. On the other hand, in order to maintain such a large-scale infrastructure and to improve the Quality of Service (QoS), service providers rely on the utilisation of the treasurable social data to generate revenue through targeted advertising or through other profitable means.

Therefore, this privacy predicament has motivated researchers to re-examine the centralised OSNs for the purpose of developing decentralised alternatives [13] that can provide a certain quality of social networking service and give user better privacy control.

1.1.2 Single Point of Failure Problem

Apart from the privacy issue faced by current OSNs, their vulnerability to the single point of failure is another prominent disadvantage of the centralised architecture. The single point of failure problem refers to that if a part of a system fails, it will stop the entire system from functioning properly. When a centrally controlled server is under the distributed denial of service (DDoS) attacks or in a malfunctioned state, the social networking services are usually completely affected. Such events have been widely reported to display the vulnerability of centralised OSNs [14-17].

Additionally, all user data stored in the centralised repository may also face the high risk of being compromised. Any vulnerability in the system could be exploited by a malicious hacker to obtain access to all the user's data at once. It is worthy of note that hackers claimed to have breached hundreds of millions of Apple accounts and threatened to remotely erase user's account [18]. At Yahoo, 1 billion user accounts were compromised by hackers in the year 2013 and 2014 [19]. In the year 2012, LinkedIn has been the victim of an unauthorised access which led to the disclosure of 117 million emails and passwords and the leaked information had been posted online later in the year 2016 [20]. There are numerous other major data breach events [21-24] on OSNs such as Facebook, Google, Twitter and Myspace. Due to the centralised architecture of current OSNs, where all the data are stored by a service provider, such threats usually have a critical impact on the privacy issue since a single successful attack can breach millions of user's information at once. Moreover, with the increasing amounts of user data being accumulated, the OSNs have become huge risk-takers to be targeted by malicious hackers. Such issues necessitate the need for new research on novel decentralised infrastructures to overcome the shortcomings of centralised approach such as the single point of failure problem and data breaches of private user information.

1.1.3 Decentralised Architecture

The architecture of decentralised OSNs (DOSNs) is generally created through the participation of a set of autonomous OSNs users collaborating with each other without a centralised repository. Social data are stored in the local devices and controlled by the end users instead of the OSNs 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, DOSNs can alleviate the privacy issues, as well as provide adequate flexibility and capability to deal with big data problems.

P2P networks [13, 25-29] have been proposed to support the decentralised architecture of OSNs in existing research works. The similarity between P2P networks and social networks, in such a way that peers can be considered as Internet users and connections reflect social relationships, leads us to believe the principles of P2P networks can suit and scale-well for the design of DOSNs. The traditional centralised architecture resembles a monolithic top-down structure, but the bottom-to-top structure of the P2P networks resembles a real-life social graph, thus provides people with more control over the contents they share. Moreover, it possesses many favourable topological features such as self-organisation, strong adaptability and fine scalability for large-scale networking applications. Furthermore, P2P networks can gather and harness tremendous computation and storage across the Internet. Advances in P2P technologies, smart devices, vehicles, and other objects embedded with sensing, computing and communication capabilities can participate in the OSNs with a dual role of computational processing as well as the content provision.

In summary, the service providers of centralised OSNs are potentially capable of exploiting the user's data at their own discretion that can violate user's privacy. Additionally, as the single point of failure problem is a major setback for the centralised architectures, which opens the gate for malicious hackers. Moreover, the increasing amounts of user requests and social data creates more vulnerability for the centralised architecture towards performance bottleneck is a critical problem in real-time systems directly affecting smooth continuation and acceptable performance. Research into the mechanisms of DOSNs based on P2P networks, which is still at an early stage, has identified many problems and challenges yet to be solved [28, 29]. These issues have provided the momentum to the development of open and effective alternatives.

1.2 Research Problems

This thesis examines three major research problems that arise in the process of designing the service discovery model for DOSNs.

1. How to design a self-organised architecture that has strong adaptability and high scalability?

2. How to support efficient information dissemination and service discovery in the designed system?
3. How to balance the trade-off between search performance and network cost during service discovery?

This research defines and characterises a general architecture design for DOSNs in a P2P network. The social network system is fully decentralised and the participants can choose the users to connect to and can form a self-organised social network. This design is named as *Self-organised Decentralised Online Social Network* (SDOSN). The users are organised in a social overlay network, where the connection between users are formed and maintained locally without a centralised control. Inexpensive network adaption algorithms are proposed to bootstrap social network and to handle network dynamisms.

Additionally, users store data locally and have full control over their shared data within the social network. The information dissemination and service discovery between users are realised through query messages. In the proposed model, users can issue different types of query messages to adapt their connections, to update personal data and to conduct service discovery. During the service discovery process, the network topology and the location of data are unknown. The communication is only enabled between users when there is a connection in the social overlay network. In order to process query messages over a decentralised network, users utilise a very limited local knowledge to forward query messages to suitable neighbours (i.e. directly connected users), then onto neighbours of neighbours. In this way, the query messages are disseminated in widely after each forwarding between neighbours until successful discovery or time out.

The dissemination of query messages may become prohibitively expensive in the decentralised network, which may impose challenges whilst balancing the trade-off between search performance and network cost. This problem triggered the research on a swarm intelligence algorithm that has the scope for deploying an adaptive forwarding degree to facilitate service discovery.

1.3 Aim and Objectives

The aim of this thesis is to design a novel self-organised architecture and adaptive mechanisms to support efficient service discovery in decentralised online social networks, which can enable equal participation, social collaboration and censorship resistance for information sharing and discovery without centralised control.

This aim is fulfilled by completing the following objectives:

- Conduct detailed research review and critical analysis on state-of-the-art methods on the areas of P2P architecture design, decentralised social systems and service discovery mechanisms.
- Design a self-organised architecture using P2P technologies and social network theories, which should provide self-sustaining mechanisms to maintain the network structure and support social communication without a dedicated central server.
- Develop a socially aware search approach to utilise social network knowledge for effective query routing and service discovery in dynamic environments.
- Evaluate the efficiency and effectiveness of the proposed model by comparing its performance against the state-of-the-art methods through simulations.

1.4 Potential Contributions

The primary contribution of this thesis is a self-organised architecture that provides adaptive topology and efficient service discovery in decentralised social networks. This is accomplished by incorporating the below novel capabilities for decentralised social networks:

- A self-organised architecture for decentralised online social networks with strong adaptability and high scalability to support decentralised topology maintenance and information exchange. This architecture defines user profile and communication mechanisms based on a friend-to-friend approach and provides distributed topology adaptation to enable network self-organisation.

- A homophily-based service discovery approach has been developed to achieve the semantic-aware and social-aware search in the social networks. The approach encompasses a service matchmaking module to capture the hidden semantic information for complex query matching and a homophily-based query processing module to characterise user's common social status and interests to conduct efficient query routing.

- A novel swarm intelligence inspired algorithm has been proposed to reduce the query routing overhead and to improve search efficiency. This algorithm characterises the service discovery as the food foraging process of swarms and employs an adaptive forwarding strategy that can adapt to various network structures and achieves good search performance and low overheads in dynamic environments.
- A software simulator has been designed and implemented with social network elements, configurable routing protocols and evaluation metrics. The efficiency and effectiveness of the proposed self-organised architecture, decentralised service discovery and adaptive routing algorithm have been systematically simulated and evaluated in dynamic complex networks.

1.5 Organisation of the Thesis

The rest of this thesis is organised as follows:

Chapter 2 presents the related works on the techniques that handle topology management and social service discovery in OSNs and P2P networks. It also discusses the existing state-of-the-art models in decentralised OSNs. Chapter 3 introduces the detailed design of the SDOSN in terms of for constructing the social overlay network and to achieve distributed topology adaptation. The correctness and effectiveness are evaluated through simulations. Chapter 4 describes the details the design of homophily-based service discovery model which covers the semantic matchmaking, query routing and social knowledge maintenance. The performance of the model has been evaluated and compared with other models through simulations. Chapter 5 details the implementation of is the swarm intelligence inspired algorithm called olfactory sensitive search to conduct an adaptive forwarding in the dynamic environment. The conclusion of the main results, limitations and future works are presented in chapter 6.

2 STATE OF THE ART

2.1 Overview

Despite the tremendous success of OSNs, centrally controlled OSNs have inherent issues related to the lack of user privacy and single point of failure. These limitations have motivated the research community to shift the computing paradigm from a centralised architecture to decentralised alternatives using P2P networks. Advantaged properties, such as decentralisation, self-organisation and fault tolerance, make P2P networks naturally scalable and attractive for designing large-scale DOSNs applications. The philosophy of the P2P design relies on the end-user's devices that can provide CPU cycles, communication and storage capacity in order to self-organise a large-scale social network. DOSNs based on P2P networks do not need (or have limited dependency upon) dedicated servers or centralised control, which gives users more autonomy without losing control over their personal data. The OSNs instances running on each peer is equivalent in functionality, so peers can manage social resources locally and act as a dual role of both service provider and service requester, rather than being confined as only the role of client in centralised OSNs applications. DOSNs can exploit the content resources that are available at the end-users and can harness a significant amount of potential computation from their devices.

This chapter presents and classifies the types of socially aware P2P networks with the focus on the topology design and service discovery mechanisms. The state-of-the-art literature are systematically reviewed to understand the way of exploiting social information to facilitate service discovery and the challenges of applying these techniques to DOSNs. The related works are summarised with their salient benefits and shortcomings to provide the context and basis for the comparison to our proposed approach.

2.2 P2P Networks

The P2P networks have been proposed as an alternative computing paradigm to the traditional client-server architecture. In a general P2P network, nodes are called peers playing equivalent roles and are managed by independent participants without dedicated servers or central authority. During last decades, the proliferation of the Internet applications and advances in communication technologies have enabled the development of a great number of research on decentralised applications powered by P2P technologies. Napster [30], Gnutella [31], BitComet [32], eMule [33], KaZaA [34], Skype [35] and Tribler [36] were among the most popular applications in file sharing, though some of them stopped services due to legal reasons. P2P applications have been estimated to collectively account for approximately 43% of Internet traffic [37]. The low cost and high availability of computing and storage resources have fostered the recent explosive growth of P2P applications. Along with the development of Internet of Things, mobile computing and ubiquitous computing, the P2P networks are expected to become even more popular. P2P networks have been addressed as the future of computing paradigm [38, 39].

Table 2.1 The architectures of selected popular P2P applications

Name	**Type**	**Protocol**	**Architecture**
Napster	Music sharing	Napster	Semi-Centralised
Gnutella	File sharing	Gnutella	Unstructured
BitComet	File sharing	BitTorrent	Structured
eMule	File sharing	eDonkey2000	Unstructured
KaZaA	Music sharing	FastTrack	Super-peer
Skype	VoIP	KaZaA-alike	Super-peer
Tribler	File sharing	BitTorrernt	Structured

P2P networks can be broadly classified into structured P2P networks, unstructured P2P networks and super-peer based P2P networks in terms of the control over data location and network topology management. The architectures of the previously introduced popular P2P systems are summarised in Table 2.1. Existing search methods for service discovery using these three architectures are reviewed as follows.

2.2.1 Structured P2P Networks

The structured P2P networks impose *Distributed Hash Tables* (DHT) to achieve a consistent mapping between a data objects key and a peer node in the network. By using

DHT, each peer maintains a partial information about other peers and is responsible for a particular range of data objects. Based on the partial information, the routing procedure typically includes visiting several nodes, in order to get closer to the target node with each hop. The network is organised into a specific topology, and the DHT ensures that any node can efficiently search the network for any data object. Here are some representative models that utilise DHT for service discovery in structured P2P networks.

Chord [40] proposed in 2001, is one of the scalable protocols for data storage and lookups in dynamic P2P systems. Chord is implemented based on the consistent hashing techniques [41] using the SHA-1 algorithm. Each data and nodes are assigned with a unique identifier ranging from *0* to $2^m - 1$, which is ordered in a Chord ring structure and *m* is the length of the identifier. Each node stores a finger table that points other nodes on the circle. In the big-*O* notation, if the size of the network is *N*, then each node stores $O(logN)$ hashing information about other nodes. The big-O notation is a commonly used method of describing how the number of operations required for an algorithm scales as the input grows in size. The worst case for data lookup requires $O(logN)$ messages. Similar to Chord, Pastry [42] is among one of the earliest implementations of DHT. Instead of organising the identifier in a ring, Pastry utilises the numeric closeness of identifier. Node and data object uniquely associate *l*-bits (typically 128) identifiers, i.e. ranging from *0* to $2^l - 1$. Pastry views identifiers as strings of digits to the base 2^b. Routing table is made up of *l/b* rows with $2^b - 1$ entries per row. The complexity of data lookup is at the order of $O(logN)$. CAN [43] designs the DHT in a virtual multi-dimensional Cartesian coordinate space where identifier space can be viewed as a *D*-dimensional identifier space of Chord or Pastry. The coordinate space is independent of the physical location of the nodes and each node possesses a zone (i.e. a section of the DHT) that holds the IP address and zone of its neighbours in the coordinate space. Symphony [44] arranges all the nodes along a ring structure and connect them with long distance contacts via harmonic distributions. Symphony is a variation of DHT that exploits the small-world phenomenon to create a constant degree topology. It establishes directional links between nodes and uses bi-directional routing to search. Symphony achieves low cost to handle the network churn i.e. nodes arrivals and departures. Viceroy [45] presents a constant degree routing networks with a logarithmic diameter which approximates a butterfly topology. It improves the Chord protocol through a hierarchical structure of the identifier space. The maintenance of node's arrivals and departures requires only a constant number linkage changes in expectation. Kademlia [46] improves the Pastry design using a XOR routing metric to compute the distance of node's identifiers. The XOR metric forms an abelian group that is simple to calculate. Each routing iteration comes one bit closer to the target. In the worst case, a basic Kademlia network with N nodes only takes $O(logN)$ steps find a targeting node.

The DHT-based P2P networks can self-organise the distributed hash structure to conduct data lookup. The routing hops determine the speed of lookup algorithm, and the arrival and departure of nodes incur maintenance overhead of DHT management. In Table 2.2, the columns depict the average for the number of routing hops during a key lookup, the number of messages for node's arrivals and departures, where N is the number of participating nodes in the system, b is the identifier length, D is the dimension of CAN, c is a constant used in Symphony and Viceroy. It is worthy of note that the big-O notations in Table 2.2 are only valid for the original design of each system.

Table 2.2 Performance comparison of the discussed DHT systems.

System	**Routing**	**Arrival**	**Departure**
Chord	$O(\frac{1}{2}\log_2(N))$	$O(\log_2^2(N))$	$O(\log_2^2(N))$
Pastry	$O(\frac{1}{b}\log_2(N))$	$O(\log_{2^b}(N))$	$O(\log_b(N))$
CAN	$O(\frac{D}{2}N^{\frac{1}{D}})$	$O(\frac{D}{2}N^{\frac{1}{D}})$	$O(2D)$
Symphony	$O(\frac{c}{k}\log_2^2(N))$	$O(\log_2^2(N))$	$O(\log_2^2(N))$
Viceroy	$O(\frac{c}{k}\log_2^2(N))$	$O(\log_2(N))$	$O(\log_2(N))$
Kademlia	$O(\log_b(N))$	$O(\log_b(N))$	$O(\log_b(N))$

The structured P2P networks rely on DHTs to organise the network topology and data objects of the entire system. The efficiency and self-organising topology for data lookup offer great advantages such as fault tolerance, low latency and guaranteed search. However, the DHT-based systems also suffer some well-known drawbacks.

Firstly, DHT approaches are based on the underlying assumption that the data distribution is approximately even among the participating nodes, this is due to the fact that DHT approaches use hash functions to map data onto node key space and hash functions provide an even distribution of node keys and the corresponding data. If the data are concentrated onto a few nodes, the search efficiency becomes low and the system becomes less robust to handle query routing. Another drawback of DHT-based approaches is that they do not support well for the semantic match, range queries, complex queries or other kinds of data lookups. Since the terms and data are mapped onto hash space to generate a unique hash code, even semantically closed terms are represented in a completely different hash code. Although some approaches [47, 48] present multiple keywords match or fuzzy match, the overheads of storage cost and network traffic is high and the scalability of these approaches is still poor. Therefore, most structured P2P networks only support keyword-based exact match. Last but not least, DHT-based systems are very sensitive to the network churn which is the phenomenon of frequent node's arrivals and departures in P2P networks. Network churn can penalise the performance of DHT or even break the DHT when a large number of nodes leaves the network. Recent studies [49-52] have proposed active recovery mechanisms, reasonable timeout settings and neighbour selection strategies to control the impact of churn in DHT-based systems. However, these proposals do not scale well in large-scaled systems, and the cost of maintaining of the DHTs in such a dynamic environment is very expensive.

2.2.2 Unstructured P2P Networks

Unlike DHT-based structured P2P networks, unstructured P2P networks do not impose such strong restrictions on the network topology. Instead, nodes are connected with others in a flat symmetrical topology based on latency, geography or any other distance metrics. This loose organisation typically has lower overlay maintenance costs than structured P2P networks. Because all nodes in the network play an equivalent role and no structures have been globally imposed upon them, unstructured P2P networks are highly robust and resistant to high rates of churn and can easily build large-scale networks that allow for localised optimisations.

The first unstructured P2P network of its kind is Gnutella [53] that was one of the most popular file sharing applications with a market share of 40.5% as of 2007 [54]. The Gnutella network is a fully decentralised alternative to semi-centralised systems such as

Napster [30] and KaZaA [34]. Nodes in Gnutella network collaborate with each other during file discovery and query messages routing. The query routing is carried out in a blind fashion that does not utilise any knowledge such as the node's neighbourhood information whilst forwarding query messages. Moreover, Gnutella utilises a flooding method to route query messages, that is, forwarding messages blindly to all neighbours in each hop. However, the flooding-based routing incurs a significant overhead of network traffic and requires a large number of nodes to process any queries. The complexity of data looking up is $O\left(N^2\right)$ or even higher, where N is the total number of nodes in the network. The data looking up is not guaranteed in Gnutella, since the lifetime of query messages is restricted to a limited number of hops. Many alternative schemes have been proposed to address the problems of Gnutella. Instead of flooding a message to all neighbours, the k-walker random walk [55] approach forwards the message to k randomly selected neighbours, where k is a system-defined parameter. The query message terminates when the required object is found or Time-to-Live (TTL) expires. The advantage of the k-walker random walk is the reduction in message cost compared to the flooding approach of Gnutella. However, this performance may vary depending on the network topology due to the randomness in neighbour selection. Modified BFS (Breadth-First Searches) [56] is a variation imposed in the flooding approach, in which a ratio of its neighbour nodes is randomly chosen to forward a query. Although this approach reduces the duplicated messages compared to Gnutella, the network traffic overhead is still high and it suffers notable performance variations due to the randomness. Iterative depending [57] uses various depth of breadth-first searches with successively larger TTLs until the query is resolved. This approach has lower search cost compared to Gnutella, but may result in larger amounts of query messages for some queries especially those requesting rare resources.

The Gnutella-like unstructured P2P systems do not maintain explicit knowledge about the location of data objects in the distributed system. Therefore, to search for a data object, these systems require as many participating nodes as necessary. If a node needs to forward a query message, it utilises flooding or controlled flooding to forward this message to other nodes until a predefined TTL is exceeded. In addition, the messages grow exponentially in relation to the number of connections, which saturates the nodes having limited bandwidth resources. As a result, many query messages may only reach a very small part of the network and are often dropped before successfully locating the target information. Therefore, the blind search methods are unreliable, and the success of queries is not always guaranteed in a predefined TTL range.

2.2.3 Super-peer based P2P Networks

Pure unstructured and structured P2P systems tend to be inefficient [58, 59] since the data objects are distributed across the network, and the data lookup often results in large latency. In addition, the performance and capability of nodes are limited, which can create bottlenecks and limits the scale of the network. The super-peer based P2P networks address these issues by creating a hierarchical architecture that embeds centralised servers in the pure P2P networks. These centralised servers are often equipped with more computational resources such as CPU, memory and bandwidth, and the act as super-peers to manage the leaf peers in a centralised manner, therefore, the workload of the whole system is split across multiple super-peers. The routing communication is enabled for super-peers only. The data lookup is conducted using an index structure in a super-peer to find the location of the data among the leaf peers. Because each super-peer manages a group of leaf peers and handles the distributed workload of queries, the super-peer-based architecture has the advantages of load balancing and robustness [60]. This architecture has been adopted for both the unstructured P2P networks [61-63] and the structured P2P networks [64-67]. Additionally, the widely used instant messaging application Skype [35] also uses the super-peer based P2P network to provide efficient online text message and video chat services. Other P2P applications adopting such a similar super-peer architecture for file sharing include Emule [33], KaZaA [68] and Spotify [69].

There are a variety of super-peer based architectures that are designed to achieve a balanced trade-off between the centralised architecture and the pure P2P architecture. This super-peer based architecture has better search performance than both the pure unstructured and structured P2P networks, but still have a few drawbacks. Firstly, the dependence on centralised functionality may cause single points of failure due to malicious attacks [70]. If a super-peer fails to function normally, then the resources of leaf peers will become unavailable. In addition, the system scalability and search performance largely rely on the capability of the centralised super-peers. Therefore, the extra maintenance of the dedicated servers can be expensive to cope with query intensive workloads.

2.3 Knowledge-based Decentralised Service Discovery

2.3.1 Knowledge Representation

The knowledge in a decentralised system refers to a partial information of the network including connections, neighbour information and historical queries etc., which is stored in a node's local space to support the local query matching and query routing. The knowledge representation is usually the structure of knowledge and the efficiency of this structure is crucial for content indexing, query routing and topology adaptation for decentralised service discovery. Researchers have proposed numerous designs [57, 71-74] of this knowledge structure inside a node to represent and summarise local service entities and routing information.

2.3.1.1 Knowledge Index

Centralised indexing technology has been used by several semi-centralised systems such as Napster [30], PlanetP [75], Minerva [76]. Napster relies on a dedicated knowledge entity in the network to store the global indices which are the metadata of contents or locations of the participating nodes. Nodes must send query requests to the knowledge entity in order to obtain the routing information for the purpose of identifying the location of the desired services in the network. PlanetP presents a semantically indexed storage layer that combines the global indices and local indices. PlanetP gathers shared content information from others via a global diffusion approach in an unstructured P2P network. Every node maintains a local index of the global directory. PlanetP is designed for small-sized P2P systems since the collection of the global knowledge in large-scaled P2P systems is infeasible and the overhead of the global diffusion is significantly high especially when nodes frequently update their contents. Minerva uses a chord-like DHT to partition the semantical information over the network and maintains a global knowledge index with other node's content statistics. Every node keeps track of a list of nodes that are responsible for a particular term across the entire network using the DHT. However, Minerva suffers the usual limitations of the most DHT-based systems. The maintenance of the knowledge is very expensive when nodes frequently leave and join the network. Besides, the update of global information is also a costly operation in a large system.

Gnutella [77] utilises a local indexing structure that only stores own content of a node. Upon receiving a query, a node only evaluates the query against its local index. If there is no requested content in the local index, the node then utilises the flooding method to forward the query to its neighbours. In the super-peers architecture [78, 79], a super peer maintains a local index of aggregations of leaf peers in a hierarchical fashion, with other super peer's contact information being stored in its local index for query routing. Each leaf peer maintains its local content and handles the query requests assigned by the super peer. Then the match results will be sent back to the corresponding super peer. Other models such as Routing Indices [80], APS [81], IAPS [82] maintains a distributed indexing structure to store the information of neighbours or remote nodes. Routing Indices keep track of a goodness score of neighbours under a number of content topics. The routing process is carried out based on the goodness score against query terms. APS utilises the feedbacks from historical searches to maintain a relative probability of successfully matched neighbours. IAPS maintains a distributed index structure that records the success score for different file types based on previous searches in the network.

2.3.1.2 Vector Space Model

The Vector Space Model (VSM) [83] has been widely used in information retrieval fields to represent terms and document in the format of vectors. VSM is based on linear algebra that achieves a continuous degree of similarity. In the research of P2P networks, VSM is used to extract knowledge from a given node and represents a node's content as a vector. The node vector summarises the content and is used to compute the relevance between nodes. VSM has been used in many research works [61, 84-87]. In VSM, the classical TF-IDF (term frequency-inverted document frequency) weighting scheme is used to extract the keywords with high importance in a document repository. The number of dimensions of a node vector is determined by the size of keywords, which easily reach the range of millions when considering phases in a large repository. Due to the lack of semantic expressiveness and high dimensionality of the vectors, the similarity computation is expensive and causes unnecessary delay in a decentralised environment. Though there are models such as Latent Semantic Indexing [51][88] using Singular Value Decomposition (SVD) techniques to reduce the number of dimensions of node vectors, they are very difficult to construct and maintain in dynamic decentralised networks due to the lack of global document information. Moreover, there is no proof or guideline to determine the proper number of dimensions for node vectors.

The cosine similarity has been widely used to calculate the distance between two document-vectors in the information retrieval fields [89, 90]. However, the cosine similarity has some limitations. First of all, it neglects the term's correlations. In the VSM, terms in a document vector are considered as independent features or completely different factors. As a result, this term-independent assumption neglects important correlations between terms. Secondly, the similarity score can be overly biased by the terms with higher weight whereby not giving enough consideration to the common terms.

2.3.2 Informed Search

Due to the drawbacks of blind search methods, informed search methods have been proposed to reduce the unnecessary duplicated messages during query routing. In these methods, nodes store some knowledge to establish shortcuts among nodes, which facilitate the query routing by forwarding intelligently according to the knowledge. Compared to the blind search methods, the informed search can reduce overhead and search latency. However, the improvement in reducing messages is achieved at the cost of the maintenance of knowledge.

Routing Indices [80] allow nodes to forward queries to the neighbour that is most likely to have the required resources. Each node maintains a routing index that contains the aggregate count of documents for a predefined set of topics of its neighbours. If a node cannot answer the query, it forwards the query to a subset of its neighbours based on a goodness score rather than randomly selecting or flooding the network.

Because of this informed neighbour selection mechanism, the traffic overheads incurred in this method is much lower than the flooding based methods. The disadvantage of this approach is that it is necessary to maintain large amounts of information updates. The number of messages required to propagate the changes in the system could overload the network. If the information update process is delayed, a node can give invalid information for query routing. Moreover, the precision of this method depends on the number of categories considered in the search process.

Adaptive probabilistic search (APS) [81] is an algorithm proposed based on a combination of k-random walk algorithm and probabilistic forwarding. Each node has a local index that maintains one entry for each neighbour. The value of each entry is a tuple that contains the identifier of a neighbour and the probability that the neighbour will be selected next time based on previous searches. A query is forwarded to k neighbours having highest probabilities to resolve the query. The probabilities along a search path will be increased for a successful hit, otherwise, the probabilities will be decreased. APS achieves higher success rates compared to k-random walk search. However, one of the drawbacks of APS is imbalanced workloads for query messages. Nodes with popular data objects are overloaded with query requests, while nodes with rare data objects are overlooked by APS.

Modified BFS [56] proposes an intelligent search mechanism that allows peers to identify links that are likely to have the relevant information. A drawback of this algorithm is that a definite period of time is required to collect the information to improve the search performance. 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 used to guide the search process could overload some peers and leave other potential peers without traffic.

Freenet [91] employs a heuristic key-based routing method where each file is associated with a key, and files with similar keys tend to form a cluster on a similar set of nodes. Freenet uses a steepest ascent depth first strategy and query messages are likely to be forwarded to the node that is associated with the most similar key. However, Freenet searches in a greedy fashion, which does not guarantee 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 [92] is a decentralised adaptive search approach that adopts the historical information of previous searches to guide the peer nodes to make routing decisions. NeuroGrid relies on the previous query keywords to forward a query to the keyword-associated nodes. The search performance will be improved only for the cached query keywords. Another drawback of this method is that it does not work well for networks where nodes join and leave rapidly. The historical information does not reflect the node's availability and it may become invalid.

Similar to NeuroGrid, in ESLP [93], nodes caches knowledge about historical searches and a number of associated connections are stored in each node. The main improvement to NeuroGrid is that ESLP includes a number of correlated keywords that are semantically close to the query in the cached knowledge. The query routing utilises these correlated keywords to gain more knowledge from the associated connections. Furthermore, it utilises an adaptive forwarding degree to reduce the unnecessary traffic overhead. However, with a fixed knowledge size, ESLP suffers some limitations: the expansion of query keywords can saturate the knowledge with information only from a few associated connections, which leads to unbalanced workloads in the network; furthermore, ESLP does not update the knowledge accordingly when the associated nodes are not available in the network due to high rate of churn.

IAPS [60] is a fully distributed and bandwidth-efficient algorithm, which uses ant-colony optimisation and takes file types into consideration and a score based on previous searches in order to reduce the search overhead. The search performance is affected by the k-walker deployed in the system. The results showed that IAPS reduces duplicated messages and increases the success rate when compared to k-walker random walk and APS. However, the routing knowledge of IAPS only focuses on the score of file types and does not consider the semantic features of the query and the content. Therefore, when the knowledge only contains single file type or no file type information, the search performance will be the same as the random walk or even worse in some cases. Additionally, the frequent arrivals and departures of nodes also create undesirable impacts on the search performance.

2.3.3 Semantic Search

The informed search methods mainly utilise historical queries to guide future query routing. The search performance largely depends on how to manage the fix-sized knowledge cache to store various query topics that can cover many semantic aspects. Many of above-mentioned search methods suffer from several limitations, such as only supporting keyword-based searches and equipping large or static knowledge cache. To address the issue of semantic search and to achieve accurate knowledge-based query routing, many research works have proposed semantic search methods for efficient service discovery in decentralised systems.

SON [94] proposes a node classification approach based on semantic content that can efficiently process query routing and preserve node autonomy. Nodes with similar semantic content are clustered together using virtual links. Due to semantic clustering of nodes, network traffic is greatly reduced against the random overlay networks. In this design, there are multiple layered SONs covering different semantic aspects (i.e. classification hierarchies) in the network. During query routing, the messages are forwarded to the most relevant SON and then flooded within this SON to find the requested services. The semantic overlay networks are completely self-organised and maintained locally. However, SON does not guarantee that relevant nodes can be found in fixed TTL, especially in large and dynamic networks. Additionally, SON uses simple predefined classification hierarchies for the generation of SONs. The search performance may be varied when the hierarchies change or the distribution of hierarchy's changes in the network. Besides, the maintenance cost of SONs also grows with increasing number of nodes in the network.

Interest-based shortcut [95] presents an interest-based locality to improve the scalability of Gnutella. Nodes with similar interests are loosely organised into an interest's structure. This is based on the assumption that if a node has a particular service that a user is interested in, it is likely that it will have other services of that particular user's interest. The interest-based shortcuts are created between users after each successful query hit. The shortcuts are used to guide future queries to locate the requested information efficiently. However, this structure of interest-based shortcuts is built on top of Gnutella overlay, which increases the maintenance overhead for virtual links of shortcuts. Another limitation is that when there are no interest shortcuts or shortcuts fail, the query routing utilises flooding approach that incurs significant traffic overhead.

SETS [84] proposes a topic-segmentation based overlay and a topic-driven query routing for efficient search in P2P networks. This method organises nodes into an overlay network and query messages can quickly reach small regions of the overlay where the relevant services reside. However, the limitation of SETS is that it requires a distinguished node to cluster peers according to topic segmentation. Even through distinguished node only provide passive support and is not involved in the run-time, all participants are required to communicate to this node for topic information, which poses the potential single point of failure issue.

In order to cope with this single point of failure problem in SETS, GES [85] presents a fully distributed topology adaptation algorithm to organise node into semantic groups. In GES, a node content is represented as a notion of a vector that offers a good trade-off between search performance and bandwidth cost. The semantic distance between two nodes, or between a node and a query are calculated through the VSM [83]. The query routing is achieved as a biased walk through random links to locate relevant semantic group and then flooded within a semantic group. GES can achieve high recall by visiting a smaller fraction of nodes. However, the first drawback of GES is the node vector may not capture the document's content accurately since documents can belong to multiple semantic groups. The second drawback is that the random links are not attached to any semantic information and are not well-maintained, which may lead to a less efficient query routing.

pSearch [88] presents two algorithms such as pVSM and pLSI to support content-based full-text search in a structured P2P network. pSearch constructs a semantic overlay on top of an m-dimensional CAN [43]. The documents and queries are represented as semantic vectors using VSM [83] and LSI [96] and are mapped to semantic space as keys in DHT. However, pSearch relies on the global information statistics such as inverted document frequency and the dimension of LSI. This global information is inferred and the semantic vector calculation is very expensive. Moreover, the inferred semantic information may be inaccurate under dynamic environments, which leads to poor performance of semantic routing.

Other approaches use biologically inspired techniques to locate and organise resources. For instance, ant algorithms are suitable for unstructured P2P networks because they do not rely on global knowledge of the network. The algorithm proposed by Michlmyr et al. [59] uses ant algorithms to guide the search process in P2P networks. Every peer maintains a repository of documents, each of which has a keyword, the neighbour provides the document and the quality of pheromone. There are two types of ants in the system such as forward ants and backward ants. The forward ants navigate the network until the document is found or until the TTL finishes. The main drawback of this method is that the pheromone information is generated 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.

2.4 Decentralised Social Systems

P2P networks are spontaneous social systems and widely being used since the very first large-scale music sharing application Napster [30]. The P2P networks encourage people to share and collaborate with equality and openness on the Internet. Recent researches are largely investigating the way of exploiting knowledge about the social ties that connect the humans behind the nodes in a P2P system. The social ties can be established by social relationships such as friendships in Facebook, follower-followed relationships in Twitter or can be inferred from overlapping interests such as sharing same files and declared common hobbies. The social networks are proving to be an effective way to provide sustainable search and collaborations in decentralised systems. Social studies [97, 98] also confirmed that people are more generous toward their friends.

This section introduces the basic social theories found in real life social networks. Furthermore, recently proposed decentralised social models and systems are compared and discussed.

2.4.1 Social Network Theories

Social networks being a part of human life are generally a complex self-organised network. Sociological studies have discovered lots of valuable theories of social networks, such as the small-world theory [99], the scale free theory [100], the homophily theory [101] and the weak tie theory [102] etc. These classical sociological theories of complex networks play an important role in guiding the design of the topological structure of decentralised social networks.

The small world theory presented by Watts and Strogatz [103] defines a family of complex networks that exhibit two properties: small average path length and high average clustering coefficient. The first property means that the average distance between nodes is small and the second property shows that nodes tend to cluster together by forming complete triads in the network. Kleinberg [104] proposes a navigable small-world model that provides the research foundation for many small-world P2P topologies [105-107]. Recent studies [108-111] have confirmed that the presence of small-world phenomenon in OSNs, and the average clustering coefficients of Orkut, Flicker, LiveJournal, and YouTube generally stay between three and five orders of magnitude larger than their corresponding random graphs. Additionally, the average path length ranges between four and six, which is significantly shorter than the average path length of the web graph. Another study shows the degrees of separation of Facebook is three and a half, which is remarkably smaller than six degrees of separation from the real life social network study [99].

Homophily [101] is another important feature in social networks, which is the tendency of individuals to associate and form social ties with similar others. In the social network context, this feature can be described as that users tend to interact with people who share similar characteristics other than dissimilar ones [101]. Studies have also validated that homophily helps people to diffuse innovations and behaviours[112] and form opinions and social norms [113]. Moreover, the homophily influences information diffusion patterns over the social networks [114]. The works of [115] proposed a decentralised service discovery methodology in multi-agent systems, in which homophily has been utilised to create efficient decentralised and self-organised structures where agents have a greater probability of establishing links with similar agents than with dissimilar ones. Their experiment results showed that this homophily method achieves high recall and small average path in small-world networks.

The weak tie theory [102] characterises two types of ties such as strong ties and weak ties. The strong ties represent close friends or family members and the weak ties refer to acquaintances or distant friends. Strong ties have more influence on the ego circle of local community. Recent studies have shown that social circles overlapping between users are likely connected by strong ties [116]. On the other hand, the weak ties are favourable links that can spread information between different social communities. If the weak ties are removed, the system would consist of disjointed sub groups. The weak ties play important role as bridges between different communities during information diffusion [117-119]. Nodes are tightly connected via strong ties inside the same community, while loosely connected via weak ties between communities.

2.4.2 Community Structure

The social theories are interested in how and why people connect to each other. Homophily theory points out people are likely to meet others with similar social features such as behaviours and activities. From a global overview in the social networks, people are gradually grouped into social communities when the social network evolves. From the network topology perspective, the network community refers to the dense groups where the nodes connections within the community are relatively close, meanwhile, the connections of nodes among the different communities are relatively sparse. By identifying the relevant community, highly relevant resources can be recommended to the target users. According to the structural characteristics of network graph, community-based discovery methods can be divided into non-overlapping community discovery and overlapping community discovery [120, 121].

2.4.2.1 Non-overlapping Community Structure

Currently research on community detection algorithms for non-overlapping communities can be broadly categorised into hierarchical clustering based on sociology, and graphic segmentation based on graph theory. Research works [122, 123] have proposed hierarchical clustering algorithm is to recursively split or merge the network nodes according to the measure of cohesion for a given node. The clustering algorithms effectively divide the network into several community hierarchies. On the other hand, graphic segmentation is also an effective way to discover community. Typical representatives of graphic segmentation algorithms are the spectrum segmentation methods [124, 125], local search optimisation based algorithms [122, 126, 127] etc. However, these algorithms suffer some shortcomings. The spectrum segmentation algorithm treats the corresponding component of the matrix feature for a node as spatial coordinates, for the purpose of mapping the coordinates of network nodes to a multi-dimensional vector space. Then clustering algorithms are adopted to gather the vectors into different communities. These methods are inevitable to calculate the eigenvalues of the matrix, so the cost of calculation is considerably huge and unfeasible for large-scale networks. Moreover, the local search optimisation based algorithm is very sensitive to the input parameters such as the initial solution and the search strategies of solution space. As a result, different parameter settings often lead to a different clustering result.

2.4.2.2 Overlapping Community Structure

Most of the commonly used community discovery algorithms divide the entire network into multiple independent community structures. But in real world social network, the absolutely independent of community structures hardly exist. On the contrary, they are composed of many overlapping interrelated communities. For instance, a given user may have multiple social interests and belongs to more than one different groups such as family group, friends group, colleagues group etc. Such users are considered as overlapping part of communities, and this information is vital in social network analysis. Therefore, many algorithms designed to detect overlapping communities have been constantly raised, such as clique percolation method (CPM) algorithm [128], Local optimisation [129], hybrid probabilistic model algorithm [37] and edge clustering algorithms [38, 39]. However, overlapping community discovery methods also suffer some limitations, such as the measurement for segmentation of graph i.e. community is unclear depending on different methods and sensitive to noisy data which make the network graph inseparable in the worst case.

2.4.3 Topology Adaptation

Topology adaptation is an approach used to organise the overlay network in order to enhance the search performance of P2P networks. Most of the existing research works utilise the heterogeneity of nodes in the system to conduct topology adaptation. Literature in [77, 130-133] have proposed various topology adaptation algorithms based on the physical heterogeneous features such as bandwidth, node's capacity and underlying network distance between nodes to build the overlay network topology. The main idea behind these algorithms is to establish connections by selecting the nodes whose physical features are similar. In this way, it can effectively alleviate the topology mismatch between overlay network topology and physical network topology in the network, thus improving the efficiency of the searching and query routing process.

Research works [134-141] have proposed topology adaptation methods based on the node content similarity. The basic principle is to organise nodes with similar content in a group or a community in order. Semantic overlay networks are used to build connections among similar nodes and query routing is achieved based on a semantic-oriented neighbour selection method that utilities semantic distance between queries and node's content index. The semantic communities are able to effectively determine the likelihood of resolving a query using the knowledge of the neighbourhood and forward queries to the most suitable nodes in each routing hop.

Other research works have tried to optimise the P2P topology based on social network measurement studies. Literature [142] introduced a square-root topology where the connection degree of a node is proportional to the square root of the popularity of its content. The results showed that this topology is optimal for random searches and achieved significant performance improvement compared to the Scale-Free topology [100]. However, this square-root topology has made some strong assumptions that each type of resources has a replica in the network and the messages can travels across the network without the TTL restrictions. Some works [143-146] have tried to optimise the network topology into the small-world network. The basic idea is that when establishing connections between nodes, the small-world model [104] is used to generate the probability of connections. The small-world topology has high average clustering coefficient, which means in each cluster, nodes are grouped together with high degrees of connection among the nodes in the cluster. The maintenance and update of content changing are efficient and the query routing path tends to be short in the small-world topology. This small-world topology provides a favourable way to organise nodes with similar interests into clusters and offers short diameters for query routing. Furthermore, Scale-Free topologies have been widely studied in the research works [147-153]. The most prominent advantage of the Scale-Free topology is the fault tolerance [154], whereby it can stabilise the network under highly dynamic environments. According to recent measurement studies, the Gnutella leads to the formation of Scale-Free topologies [53, 155].

2.4.4 P2P Social Systems

This section reviews the related works in the area of decentralised social systems using P2P techniques. The social systems employ social theories to establish links among the nodes and carry out query routing based on the social knowledge stored in their local space. The discussions in this sub-section focus on the utilisation of social knowledge and routing mechanisms in the social systems.

2.4.4.1 Socially Aware P2P Systems

The term *"socially aware"* refers to exploiting social features in the P2P systems, in which nodes are considered as participants in the social networks. The connections among the nodes are treated as social ties, which can be established by declaring social relationships, or can be inferred from overlapping interests. The query routing in such systems is generally social network oriented. A number related works [36, 156-158] have utilised the social features to enable efficient query routing and service discovery.

Sprout [156] is a DHT-based routing algorithm that uses social links to improve the number of search results and to reduce the query delays. Similar to other DHT algorithm, upon forwarding a query to a node, this approach fetches the friend's information to obtain the closest identifier. If a friend identifier is found by the algorithm, then the query will be forwarded to the friend. Otherwise, it adopts the traditional strategy of Chord [40] for query routing. Their experiment results showed that the query routing efficiency is improved by more than 50% to that of Chord by using social links information to augment Chord routing.

Orkut [159] leverages the social network to improve the performance in P2P networks. This approach utilises user's interest information to guide the query routing process. Upon receiving a query, the BFS approach is used to search the friend's interests during routing. If there is no match in all of the friend's interest, the query will be flooded to the friends of friends. Based on the performance comparison with other two pure multicast routing protocols ESM [160] and NICE [161], this work claimed that their interest-based social routing approach achieves higher search performance and lower delay.

Tribler [36] proposed a social-based P2P file sharing system that utilises social incentives for content discovery and recommendation. This system is an extension of the implementation in BitTorrent [162] by using the buddycast algorithm to find resources with user preferences. The buddycast algorithm employs homophily feature to encourage user to connect people with similar tastes in content. In this way, users are grouped into communities based on their taste and these social communities give incentives for users to share more resources. As a result, more files will become available and the overall performance is improved significantly. The proposed collaborative downloading method achieves a significant speedup up to six fold than that of BitTorrent [162].

A super-peer based social approach [163] leverages virtual social groups on top of a P2P network to improve the search performance. The social relationships are incrementally built through user interactions and social groups are formed according to user preferences of file sharing. Each social group is handled by a super peer and the leaf peers maintain friendships with the super peer. During the query routing process, a query is firstly evaluated within a group. If the requested file cannot be found, the query will be propagated to other super peers to carry out the content match in that group. Their experiments evaluated the performance of the proposed super-peer approach against three methods including flooding method, random super-peer method and interest cluster method. The obtained results showed that the proposed approach yields the shortest average hit path length at 1.79 and highest search performance in terms of precision and recall.

Social-P2P [164] is another super-peer network that utilises social closeness and common interests for simultaneous and trustworthy file sharing. The common multi-interest nodes are grouped into clusters and the trust between nodes is measured by the frequency of interactions. In each cluster, comparably stable nodes are select as super peers to form a DHT for inter-cluster query routing. During the query routing process, a query will be forwarded to the neighbours with higher trust inside each cluster using the random walk strategy. When the requested file cannot be found, the query will be routed based on DHT to other super peers.

Other similar social systems [165-171] have been proposed to leverage social network features to improve the file sharing and resource discovering in P2P networks. In these systems, network connections rely on homophily among users such as common interests and friendships. This homophily knowledge is gradually gained during social interactions, and nodes can be clustered into groups based on their social interests or other social features. The query routing is achieved based on the network knowledge by determining the best neighbour to be selected in the next hop. A query is evaluated within a cluster and then forwarded to other clusters until the query has been resolved or when the TTL ends.

2.4.4.2 Decentralised Design of OSNs

P2P networks are inherently decentralised and resistant to censorship that makes them highly desirable for the design of DOSNs. In addition to the research on service discovery in socially aware P2P networks, research works [25, 27, 172-178] have proposed some rudimentary designs of OSNs based on P2P technologies. The P2P-based architecture for OSNs provides a decentralised solution, which allows users to enjoy the convenience of social networks, and at the same time, can give users full control of their privacy. Some representative designs for DOSNs using P2P architecture are reviewed as follows.

PeerSoN [25] presents a two-tier system architecture, where the first tier is used for node communications via a DHT structure and the second tier is used for data lookup. This approach provides an opportunistic and delay-tolerant mechanism to exchange data between users. The user's privacy is protected through symmetric encryption of content.

Vis-`a-Vis [172] proposed a decentralised design to achieve the privacy preserving OSN services. In this design, user's data are stored on a personal virtual machine called Virtual Individual Server (VIS) and users can be self-organised into social groups. Vis-`a-Vis utilises a distributed tree hierarchy to locate contents in the social groups.

Safebook [27] have focused on data availability and the privacy issues in centralised OSN. This approach proposes several privacy-preserving mechanisms for data storage and management. The overlays in Safebook consist of 3 main components including the matryoshkas, the P2P system and the trusted identification service. The matryoshkas provides trusted data storage, profile data retrieval and obscure communication for each node. The P2P system provides location services for data lookup, and the trusted identification service assigns users with a unique identifier based on hash functions.

LotusNet [173] is another attempt to mitigate the privacy issue of current OSNs. This approach adopts a DHT method that combines user's identifiers and overlay index nodes to provide confidentiality, access control and a set of core OSNs services, such as notifications, folksonomic content search, reputation management and storage service.

LifeSocial [174] stores user contents in a DHT structure and adopts symmetric PKI [179] to achieve privacy control. This approach distributes the load of maintaining OSN infrastructure to all the participating users in a P2P fashion and offers extendible social functionality using a plugin-based design pattern.

Apart from the above works, there are also efforts from the open source communities and commercial products in this context. Diaspora [175] is an implementation of privacy-aware, decentralised and user-owned OSN, which grant users with the power of controlling the location of their data storage. Diaspora operates the social network in an unstructured P2P network where all the participants are equal and no privileged entity controls the network. As of 2017, there are more than half million user accounts in Diaspora [180]. There are other commercial efforts such as Enthinnai [181] and FreedomBox [182], implementing the OSN services for users to build their social networks in a decentralised manner.

These systems build social networks infrastructure through the P2P technologies which allow users to establish a transparent connection with other users through personal networking equipment. Then using distributed environment encryption, key management and access control schemes, user's data and message are protected. Therefore, in a decentralised social networks environment, users have controls over their data and privacy and can use social networks services on the Internet under intermittently available circumstances without the need for a central server, and centralised storage, scheduling and management. There are many similar proposals such as SCOPE [183], MyNet [184], Magna [185] and Persona [186], all designed to handle the privacy issues in P2P OSNs. However, these P2P systems for OSNs are only at the preliminary experimental stage mainly focused on the privacy-preserving strategies. Moreover, these systems also lack a comprehensive architectural design and efficient data lookup mechanisms in highly dynamic network environments.

2.5 Summary

In this chapter, three types of P2P architectures such as unstructured, structured and super-peer based P2P networks have been introduced and critically reviewed from the perspectives of data management and query routing. Then, the knowledge based service discovery techniques have been discussed, along with a critical comparison of the state-of-the-art techniques. The main goal of the service discovery mechanisms is to route the query to obtain high-quality services while minimising the communication cost in the network. Further, decentralised social systems have been comprehensively reviewed and some prototypes based DOSNs have been introduced. Decentralised social systems leverage social network in P2P networks to improve the user engagement and search performance. Meanwhile, the DOSNs adopt P2P technologies to achieve decentralisation and privacy control.

With the OSN paradigm shifting from a centralised architecture to decentralised alternatives, P2P networks have attracted extensive attention from researchers. Notable properties of decentralised architectures such as decentralisation, self-organisation and fault tolerance, make P2P networks naturally scalable and attractive for designing large-scale DOSNs applications. The P2P OSNs need little or no administrative intervention to maintain the system and can avoid the single point of failure problems. Additionally, the nature of decentralisation in the P2P OSNs scales well for balancing the heavy workloads and user intensive computation.

The existing literature and research works have laid a platform for the design of DOSNs. However, existing proposals mainly focused on the privacy strategies in the decentralised social systems. The core functionalities of DOSNs such as the information exchange and service discovery lack in-depth research and face many challenges. Firstly, the main challenge is the decentralised architecture adaptation for node's self-organisation. Self-organising node's connections and contents under higher churn rate have been neglected by most of the current prototypes of DOSNs. Another challenge is how to effectively handle the lack of search guarantees and delays caused by inherent problems of P2P networks. In P2P networks, nodes are prone to become unreliable which can lead to the service unavailability. However, the impact of adequate service provision in decentralised social systems is not well studied in the existing research works.

To conclude, research in DOSNs is still at its very early stage and there are many problems yet to be resolved. This thesis proposes a novel self-organised architecture for OSNs focused on service discovery mechanisms to overcome the limitations of existing works and ultimately to achieve better scalability, availability and search performance.

3 A Self-organised Architecture

3.1 Overview

Previous chapter has reviewed the state-of-the-art research works and further validate the effectiveness of building decentralised architecture for OSNs to alleviate the key problems of centralised architecture such as the lack of user privacy and single point of failure. The promising features including decentralisation, self-organisation and fault tolerance make the P2P networks naturally attractive for designing DOSNs.

This chapter presents the systematic design of self-organised decentralised OSN (SDOSN) with a special focus on the network structure formation and maintenance. The aim of the architecture design is to provide cost-effective, highly scalable infrastructure of OSNs applications that self-organise social networks and support efficient social service discovery in a fully decentralised environment. The characteristics of the proposed architecture for OSNs can be summarised into three points 1) decentralisation: information exchanges and social communications are directly between users in the social networks; 2) self-organisation: participants exhibit a self-organising and self-sustaining nature. The formation of social relationships and interactions of participants are completed spontaneously without any support from external dedicated servers; 3) autonomy: it gives users full control over their data and avoids censorship and centralised control. In the meanwhile, users can access their personal data both online and offline.

A social overlay network is designed to form a bottom-to-top social network that resembles the real-life social graph. The social overlay network is based on the unstructured P2P network where connections of users and distribution of service contents are in an arbitrary fashion. Therefore, there is no strict topology control over the social network structures and the service content. In order to enable self-organisation, user's local service content, social connections and communications are defined and modelled with structured semantic abstractions, which facilitates the problem definition and conducting a unified comparison on heterogeneous service content. Then a distributed topology adaptation model is designed to provide bootstrapping and handle network dynamisms. Finally, a simulation platform is introduced and the correctness and efficiency of the proposed architecture is evaluated through simulations.

3.2 System Model

The SDOSN system consists of a set of autonomous users which provide their social functionality through a set of services. Users are scattered nodes in a network and have no global view of the network. Moreover, users can only communicate with a limited number of neighbours who are directly connected in the network. To avoid confusion, within this thesis a user and a node are considered to have one-to-one mapping; therefore, the terms *"user"* and *"node"* are totally interchangeable.

DEFINITION 3.1: *The social network is modelled as an undirected graph $G = (V, E)$, where $V=\{v_1,v_2,v_3\ldots,v_n\}$ is a finite set of users and $E \subseteq V \times V$ is the friendship relations among them.*

The social network, in this research, is a strongly connected graph. In other words, all nodes have at least one neighbour and there is a route between every two nodes.

DEFINITION 3.2: $\forall u \in V$ *, the function $N(u): V \to 2^V$ maps users to their set of friends. The set of first-degree neighbours (i.e. immediate friends) for u is defined as $N(u) = \{v \mid (u,v) \in E\}$. Further the second-degree neighbours (i.e. friends of u's friends) is defined as $N^2(u) = \{z \mid z \in N(v) \wedge v \in N(u) \wedge (z,v) \in E \wedge (u,v) \in E\}$.*

In the above definition, $N(u)$ indicates the direct relationship between u and its neighbouring nodes. Since the network is undirected, the relationship among the nodes is symmetric. Further, let $\tilde{N}(u) = N(u) \cup \{u\}$ be the social circle of u. In the graph, $\tilde{N}(u)$ is a sub-graph that consists of interconnections between u and its immediate friends.

A node is considered as a concrete social user, so it controls the local information about services it shares and the knowledge about its neighbours. The node has no information about the remaining nodes in the social network. This is one of the main challenges of decentralised social networks, i.e. effectively discovering required services without global knowledge. The services in the network are defined as:

DEFINITION 3.3: *The services possessed by u is defined as* $S(u)=\{s_1, s_2, s_3 \ldots, s_n\}$ *, and* $\Omega(S(V))_G$ *is all services in the network G. Each service is defined as the set of terms* $s_i=\{w_1, w_2, w_3 \ldots, w_n\}$ *.*

In this research, the services are defined as a set of terms to conduct effective information retrieval. It should be noted that the *"service"* is a general concept that is not restricted to well-defined web services, but many types of data including text, hashtags, friends and description of pictures or video clips. The service discovery should be considered a capability to locate varied, multi-source text-based social resources.

3.2.1 Social Overlay Network

SDOSN uses the unstructured P2P network to operate social networks. Social interactions are carried out collectively by users in the social network. The unstructured P2P network is self-organised and there is no global topology management. Due to the nature of fully decentralised P2P network, peer nodes resemble social users and every node is considered equal in the network.

This chapter adopts the *friend-to-friend* (F2F) approach [187] to build a social overlay network, where communication among nodes is enabled only if their owners know each other. In this design, users are arranged in a social overlay where one-to-one mapping mirrors the underlying network as shown in Figure 3.1. Users can use various devices to connect each other in the underlying networks, such as laptops, desktops, PDAs, mobiles etc. If a user uses multiple devices on the social network, it is assumed that this user does not log on from more than one device at the same time. A social overlay with 1000 participants is shown in Figure 3.2, where participants have an average of 50 social connections in the network and the degree of connection is depicted with different colour.

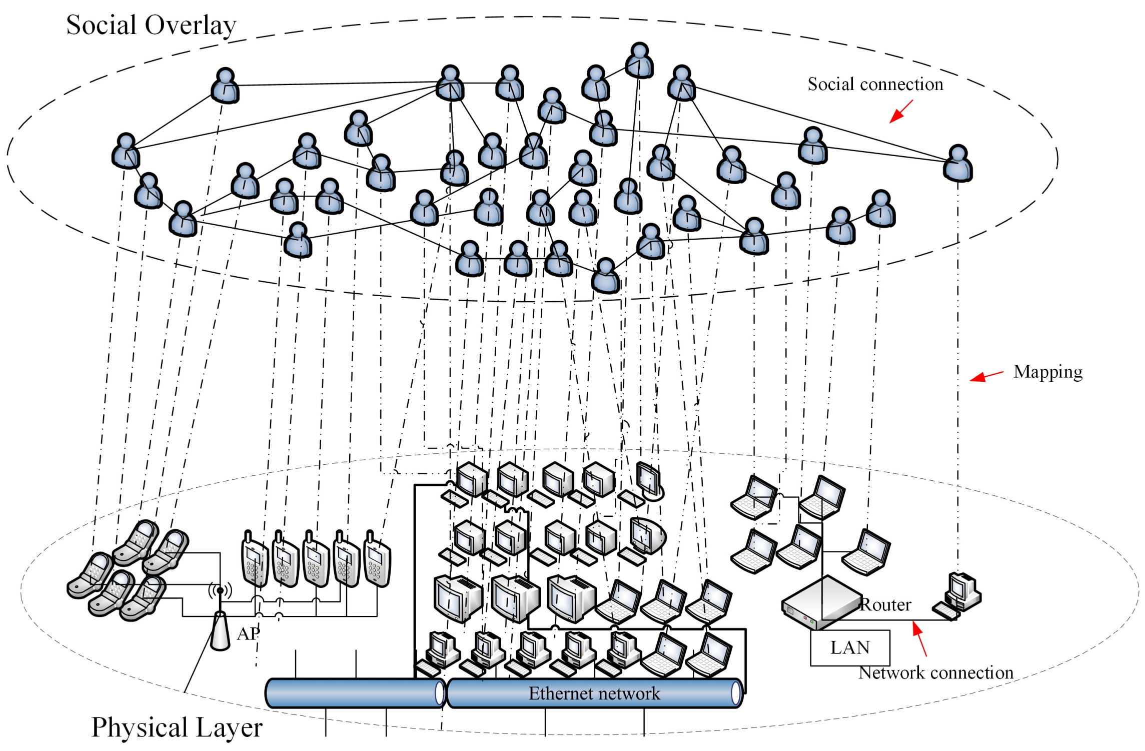


Figure 3.1 Social overlay network

The overlay network is built through logical connections i.e. a direct link on the overlay network does not imply that there is a direct link on the underlying physical network. For instance, a user located in the U.K. connects a user in the U.S. on the overlay network. The distance on the overlay network is one hop when they exchange messages, however, there is possibly a great number of hops between their physical devices via national and international routers.

The maintenance of the overlay network is achieved in a self-organised manner, which means node connections are established or detached in an autonomous fashion based on the social knowledge during user interactions without reporting to a central server. The social knowledge is historical information stored locally to maintain social relationships between nodes. Some of the main benefits of choosing the unstructured F2F social overlay are summarised as follows: Firstly, unlike links in a DHT, links in the social overlay represent friendships. The hypothesis here is that friends are more likely to interact and cooperate with other friends than with random strangers. A recent study [188] showed that approximately 78% of user interactions take part within one-hop neighbours, i.e. friends. Secondly, because people tend to talk to people they know, the social overlay enables shortcuts to friends' shared content and constrains communication traffic to small network sections at a time when handling frequent interactions between friends. Thirdly this overlay improves the locality of P2P social networks as people tend to connect to people that are alike, with geographical co-location playing a key role. For example, people tend to have many friends who are in the same city. Therefore, paths on the physical layer are likely to be short and localised. Moreover, privacy issues are mitigated since communication is restricted to friends; only the identity of the original user needs to be checked [189].

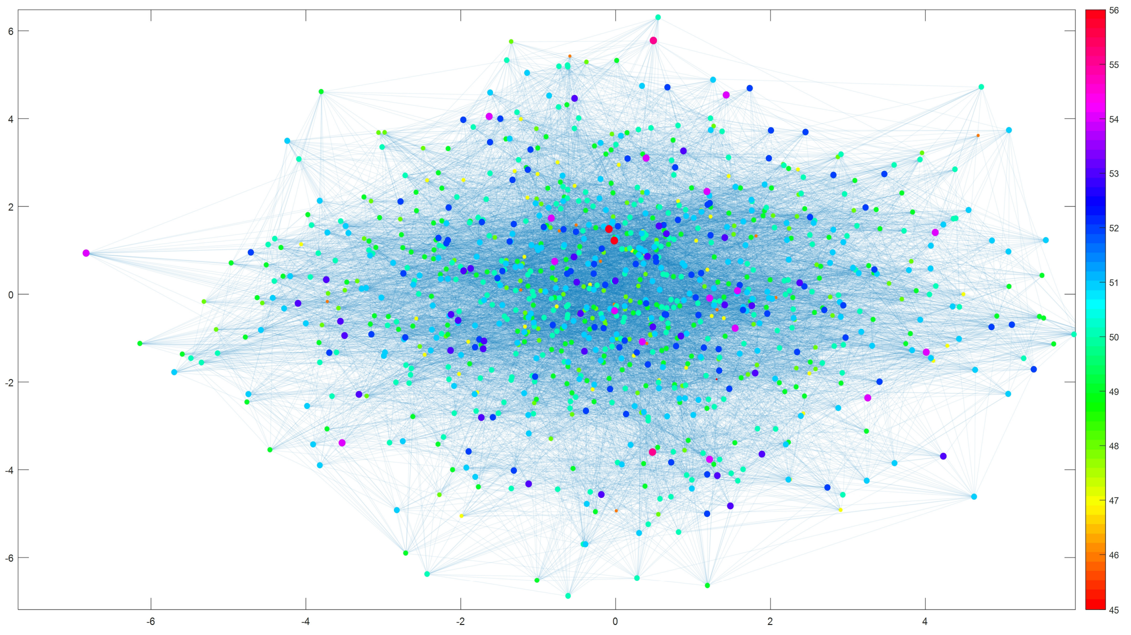


Figure 3.2 A social network graph with 1000 participants

The social overlay lays the foundation to support user's connection and communication in the social network. During social interactions, for instance, a user requests a service in the social network, SDOSN relies on hop-by-hop routing (friends to friends) to discover the identity of users who likely to hold the desired service. Because the location of social service provision (i.e. the location of user shared contents) has a significant impact on the search performance in a decentralised system, it is not surprising that content management and topology adaptation are two key research challenges in this context. In order to achieve high scalability and favourable search efficiency, each node (i.e. an individual user) maintains three kinds of distinct index: a local service index, a network shortcut index and a knowledge index in its local storage. The local service index stores location references of various types of local services, such as text documents, music and videos, etc. The network shortcut index keeps track of the historical successful discovery information and the knowledge index stores the information of immediate neighbours. The following section will discuss the detailed design of the three indexes.

3.2.2 Content Management

The content sharing in an unstructured P2P network is fully decentralised with the collaboration of independent participants. The participants store their social services locally and retain control over the local service, whereby controlling the access to the information. The term *"services"* designates an atomic piece of social information which all the users in the social network can contribute and discover. The services are created by the users and there is no definite standard for their formation.

The content management utilises abstractions of the data structure to model user's local contents. Three types of index abstractions are introduced in the following sections.

3.2.2.1 Local Service Index (LSI)

SDOSN does not impose a structured hash table on the overlay network. Participants in the social network have full control over their own shared services and maintain them through an index structure in an arbitrary fashion. The LSI utilises an inverted index that organises various types of local services with a user defined privacy level.

Basically, LSI represents the semantic knowledge of a node, which shows the semantic concepts contained in its local storage.

DEFINITION 3.4**:** *LSI for a node u is defined a* $LSI(u) = \{(c_i, k_i) \mid c_i \in \mathbb{N}, k_i \in KS(u)\}$*, where* c_i *is the number of relevant services reachable through node u for keyword* k_i*;* $KS(u)$ *is the keyword-service index and* $\mathbb{N}$ *is the natural number set.*

In user's local service repository, each service consists of a set of terms. In order to obtain the semantics in the service, these terms are pre-processed to filter out stop words and stem the root. Stop words refer to the most common words that are considered non-informative such as *"the"*, *"a"*, *"of"*, etc. Stemming words refers to the words that provide the root form for other related words. For example, *"search"* is the root for *"searching"*, *"searched"* and *"searches"*. The pre-processed terms are named keywords in this research. In this sense, each service is represented by a set of representative terms i.e. keywords which capture the main semantics for the service. The frequency of a keyword is also recorded as a weighted information for the keyword in that service. Shown in Figure 3.3, each node has an inverted index of keywords-services structure in the form of link list $\{keyword,(s,p,tf)\}$, where s represents the service, p is the privacy setting, and *tf* refers to the frequency of the *keyword* that appears in s. The inverted index $KS(u)$ is created from the local service content of node *u*. For each keyword in the index, a list of pointers is stored for all the occurrences and frequencies of that keyword in the service content. The tf_{ij} is an integer number which means the frequency of the *i-th* keyword in the *j-th* service on the inverted index structure.

Apart from the term frequency information in the inverted index structure, the LSI also organises the shared service objects with user-controlled privacy strategies $\{private, protected, public\}$. The privacy of local services is fully controlled by the users in SDOSN. The functionality of the three types of privacy strategies is to give users a relatively flexible control on whom the services are shared with.

Private services: private owned services that are only accessible to the user himself/herself. When a user changes the privacy attribute of a service to the state: *"private"*, the service will not be accessible to the user's friends and other users in the network.

Protected services: the shared services that can be accessible to direct neighbours or partial neighbours depending on the user's preferences. The *"protected"* attribute is the default privacy setting when users share their social services.

Public services: the services shared by users are accessible to all users in the network. For example, in the proposed design, the user declared social interests is a kind of the public services that can be seen by all users.

LSI can be viewed, browsed and managed by the owner, which provides a well-defined structure for social service sharing and discovery and cost-effective privacy control in the decentralised environment. The management for LSI includes setting access permissions, removing expired services, and automate updating the index upon file changes. Additionally, the inverted index is able to support efficient full-text search inside a node. With the inverted index created, the complexity of query-service match $O(MN)$ is reduced to $O(N)$ with sequential search or even $O(\log N)$ with binary search, where the M is the size of services and N is the size of keywords.

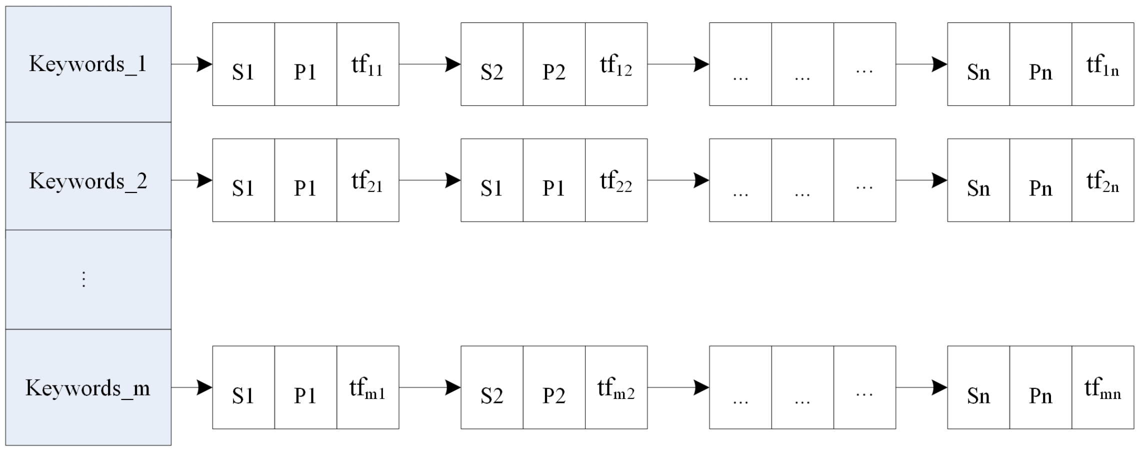


Figure 3.3 Inverted index of local services

This chapter emphasises on how to manage the social overlay to provide maximum performance of service discovery without losing user required privacy. The privacy issue is mitigated through the F2F social overlay since the personal social information is only shared with friends in the default setting. Furthermore, the main focus is utilising semantic text analysis on services, and keywords to improve the accuracy of service discovery. However, SDOSN does not attempt to resolve inconsistency among the copies of a service object in different user's cache. For instance, if a user retweets or forwards a service object from others, the service object will be downloaded to the local cache and maintained by the user using LSI regardless of changes of in the original sources.

3.2.2.2 Network Shortcut Index (NSI)

The F2F social overlay defines the communication mechanisms of participants, in which nodes only can communicate with their direct neighbours. When the number of direct neighbours is small, discovery performance of available services will be penalised. Moreover, the network traffic is imbalanced as nodes with the higher degree are more often visited because they have more social connections. In order to improve the performance of service discovery and balance the workloads, the network shortcuts are proposed to create additional links on top of the F2F social overlay. The network shortcuts can reduce to hop-by-hop delays in service discovery. Figure 3.4 depicts the F2F social overlay with two network shortcuts, represented by the red dotted lines. The grey circles represent service providers and they are discovered by hop-by-hop routing using directly connected neighbours. After successful discovery, the locations of service providers are stored in the network shortcuts cache of the query initiator. In the future, to avoid flooding, the service is located through the network shortcuts first. If no shortcuts have the requested service, then the query will be routed through neighbours. The definition of the shortcut is given as:

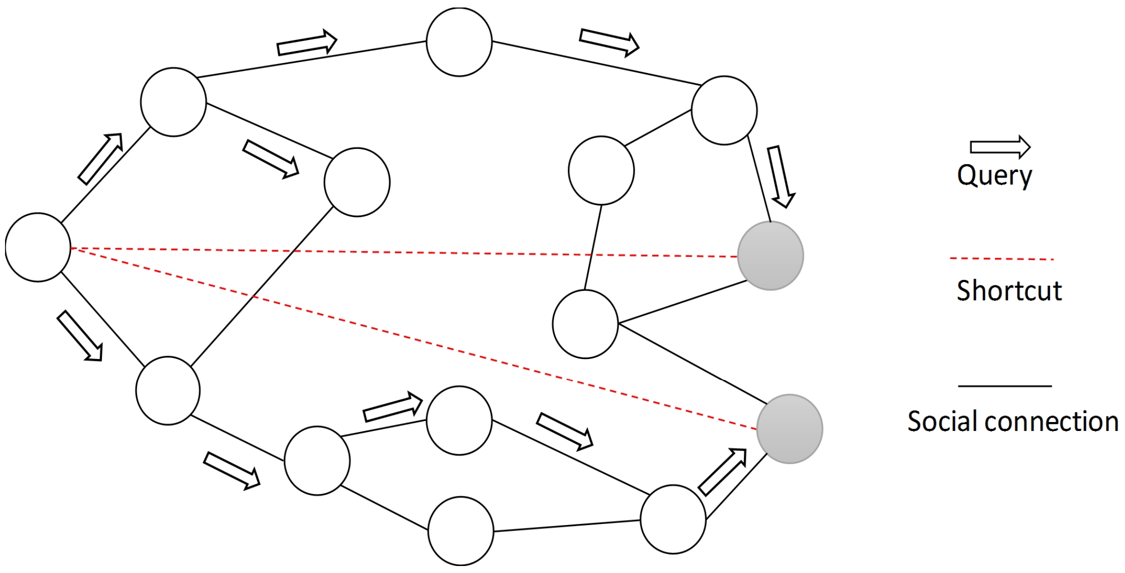


Figure 3.4 Network shortcuts shown in red dotted links

DEFINITION 3.5: $NSI(u) = \{(t_i, URI(v_j)) \mid t_i \in \Omega(V)_T, v_j \in V\}$ *is defined as the network shortcut index for node u, where* t_i *is a query topic,* $\Omega(V)_T$ *is the set of possible query topics.* $URI(v_j)$ *is the Uniform Resource Identifier that is used to identify a service in a remote node* v_j .

The network shortcut index is the successful query history for a period in the past with a fixed-size. The shortcuts are topic-URI pairs that contain the direct path to visit a service in a remote node using URI without flooding the network. The historical NSI information is stored in the local space and are attached with a time stamp of when the data being obtained and updated with the *Least Recently Used* (LRU) scheme.

Each node continuously maintains the NSI and updates new information when the size meets the upper bound. With the LRU scheme, it allows nodes to adapt to the dynamic environment and incrementally refine the shortcuts during interactions. The advantages of the NSI are twofold: 1) it is a performance-enhanced module that can be implemented onto the F2F social overlay. 2) it reduces the network traffic and latency using a one-hop service discovery for recent successful queries. The NSI acts as a separate module to enhance the F2F service discovery mechanisms with more efficiency and less overhead of network traffic.

3.2.2.3 Local Knowledge Index (LKI)

The LKI stores first-degree neighbours' information in the social overlay network based on historical interactions. The structure of LKI is a relational table: each row represents a first-degree neighbour, and the columns include the second-degree neighbour list, the neighbour's interests list and a pheromone table. The pheromone table can be viewed as the query hit during past searches and it is used to model the swarm intelligence to conduct adaptive service discovery, which will be introduced in Chapter 5.

This chapter formally represents the local knowledge index for a node *u* as follows:

DEFINITION 3.6: *The LKI is characterised as a 5-tuple data abstraction* $LKI(u)=\{(NFH(N(u),N(u),N^2(u),UII(N(u)),PHT(c_i,t_i,N(u)))|c_i\in\mathbb{N},t_i\in LSI(N(u))\}$ *where* $NFH(N(u))$ *is neighbour's node fingerprint hash,* $N(u)$ *is the first-degree neighbours of u,* $N^2(u)$ *is the second-degree neighbours of u,* $UII(N(u))$ *is the interest list of the first-degree neighbours,* $PH(c_i,t_i,N(u))$ *is the pheromone table of the first-degree neighbours and* c_i *is the number of hits through neighbours for topic* t_i.

In the LKI, a user knows not only the set of first-degree neighbours (i.e. friends), but also the neighbour list of the first-degree neighbours (friend's list of friends). This chapter assumes the identifiers and public services (such as declared interests) of second-degree neighbours are accessible. This assumption is reasonable because in modern OSNs, e.g. Facebook and LinkedIn, the set of friends is already a part of the profile and incremental updates (in the form of new friendship events involving your friends) are showed in the newsfeed [190]. For instance, if user *A* and user *B* are friends in a social network, *A* can see *B*'s friends list as this is a part of *B*'s profile and vice versa. If user *B* and user *C* have formed a new friendship, then user *A* will receive the update that *B* has a new friend *C* in the newsfeed.

Besides, LKI also integrates user's social interests to deal with personalised social information. Social interests are the terms that describe the personal concerns or involvement of a user in social networks. Interest terms are system-provided terms (e.g.

social tags) that do not necessarily appear in user's local repository. Interests are more semantically recapitulative concepts than keywords extracted from local services and are strong indicators of a user's preferences or expertise. The interests are explicitly declared by users and they are considered as public services that can be seen by all participants in the social network. Here the definition of user's interest is given as:

DEFINITION 3.7: $UII(u) = \{e_1, e_2, e_3 \ldots, e_n\}$ *is defined as the user interest for node u, where* e_i *is a user declared term of interest.*

Finally, LKI includes a routing table based on swarm intelligence. The pheromone table $PHT(c_i, t_i, N(u))$ is the query hit table through first-degree neighbours that aids a narrow and precise pathway route for other participants to follow during query routing. The pheromone table serves as service discovery *"hints"* and it does not provide direct service location. The term *"pheromone"* is a biological term that describes a special chemical secretion of species commonly known to lead other members of its own species towards the point of interest. Pheromone table is inspired by the food seeking strategies of biological swarms and details will be discussed in chapter 5. During the service discovery process, nodes visit the shorter paths more frequently, whereby leaves more trails of pheromones in the respective paths. The main purpose of the pheromone table is to provide performance-enhanced hints for identifying promising neighbour in the next hop. During the query routing process, a query will follow the hints in the pheromone table via hop-by-hop to the service provider which is the node that possesses the requested services. After successful discovery, the hints will be reinforced along the path from service requester and provider.

An example of LKI snippet for Node0 is shown in Figure 3.5. In this example, Node0 has three neighbours Node1, Node2 and Node3. Node0 maintains the knowledge of the neighbour list, interest list and the pheromone table of its three neighbours.

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

Figure 3.5 Example of the local knowledge index structure

3.2.3 Node Content Abstraction

In a decentralised network, nodes are often heterogeneous in terms of their operating systems, types of services and size of files. In addition, the semantic content of nodes is various and often has a very high dimensionality of concepts. The node similarity is a significant metric in the topology maintenance and service discovery. Utilising the heterogeneous nodes into a harmonious and accordant representation to calculate their similarity is a very challenging task.

This chapter proposes a node content abstraction model that summarises a user's content to generate an equal length hash value called *Node Fingerprint Hash* (NFH) for each user. NFH is derived from the service's textual information and is used to determine the similarity between users.

3.2.3.1 Fingerprint Generation Algorithm

In order to reduce the heterogeneity and dimensionality of semantic content and to conduct fast similarity match in the service discovery process, this research introduces a hash method to capture the contents with locality sensitive features. The idea is inspired from the simhash algorithm [191], which is one of the locality sensitive hash algorithms for quickly estimating the similarity of two sets. The goal of NFH is to generate a small hash digest for each node considering the local services to resolve the node similarity calculation. If two hash digests have a low distance between them, then it is likely that the corresponding nodes are also similar to each other. The nature of NFH differs from conventional hash functions such as MD5, Hash map etc., because it is aiming at maximising the probability of a collision of similar objects in a predefined dimension.

NFH utilises service documents as input objects and similar objects can map to the same bit in the fingerprint hash with high probability. The hash size is usually predefined by the algorithm as *32bits/64bits*, so that the number of bits is much smaller than the universe of possible input objects, in our case, the total number of possible social services.

In specific, the input objects of the algorithm are first pre-processed by natural language processing tools by removing stop words, stemming and extracting keyword and phrases. Every processed service document is represented as a set of keywords: $S_i = \{w_1, w_2, w_3, ... w_n\}$. Then a general hash function is used to calculate the hash code for each service document, which generates a fix-length *0-1* hash value: $Hash(S_i) = \{1,0\}_k$, where *k* is an integer, representing the number of bits for hash values. Ideally, the hash function will generate a different *0-1* hash value for each document.

Algorithm 3.1 Node Fingerprint Hash

```
Input: S = hashsize, D = service documents, N = NodeID

Output: Node Fingerprint Hash: V

1.  // initiate the hash function
2.  InitSimHash(S)
3.      // set the user defined hashsize
4.      let V ← [0] * hashsize
5.      ExtractFeatures(N,D,S)
6.      // break service document into phases
7.      DocumentShingling(D)
8.  HashProcess()
9.      for each d in D do
10.         d.hashcode ←Hash(S)
11.         //generate hash code for each document
12.         D.hashcodeList.add(d.hashcode)
13.     for bit←1 to S do
14.         // summarise document hashcode bit by bit
15.         for each d.hashcode in D.hashcode do
16.             if d.hashcode(bit).isSet() then
17.                 V[i] ← V[i] +1
18.             else V[i] ← V[i] – 1
19. HashMapping(V[i])
20.     for bit←1  to S do
21.         if V[i] > 0 then
22.             V[i].set(1)
23.         else  V[i].set(0)
24. // assign node fingerprint
25. AssignNFH(V,N)
```

Next, the weighting scheme utilises the frequency of service document (denoted as *DF)* to transform the *0-1* values into a sequence of frequency numbers. Each bit in the $Hash(S_i)\ 1<i<n$, where *n* is the number of the service documents in the node, multiplies the DF of S$_i$. The weight of a service document *S$_i$* is calculated as in Equation (3.1).

$$weight(S_i)_p = \begin{cases} (Hash(S_i)_p)\times DF \quad if\ \arg_{(bit_p=1)}(Hash(S_i)=1 \\ (-1)\times DF \quad otherwise \end{cases} \tag{3.1}$$

where *DF* is the frequency of services; *p* is the *p-th* bit in the hash value and the *arg* function is used to fetch the bit that equals *1* in the hash value.

The next step is to summarise the weighting values into an aggregate code. The *p-th* bit $1 < p < k$ in *k-bits* aggregated code is calculated as in Equation (3.2).

$$N_p = \sum_{p=1}^{k} \sum_{i}^{n} weight(S_i)_p \tag{3.2}$$

Finally, *k-bits* fingerprint is a piecewise function using one to one mapping from aggregated code N_p as in Equation (3.3).

$$NFH_p = \begin{cases} 1 \ \ if \ N_p > 0 \\ 0 \ \ otherwise \end{cases} \tag{3.3}$$

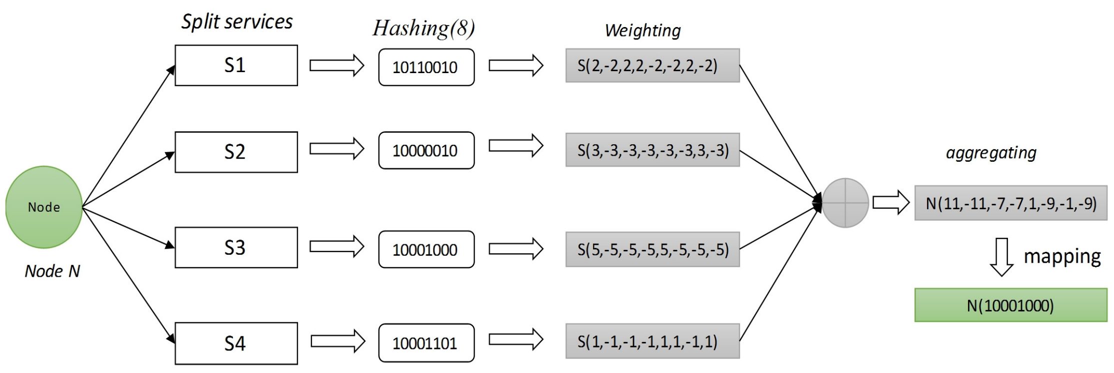


Figure 3.6 The calculation process for Node Fingerprint Hash with size=8

Figure 3.6 shows a simple example for NFH value calculation. In this example, the node has four distinct service documents. Using a predefined size, the hash value for each document is obtained with the same length. Four documents with 8 bits hash value are refined by the weighting technique. For instance, the S_1 has a frequency of 2 in the node, so that the hash value 10110010 is transformed to $(2,-2,2,2,-2,-2,2,-2)$ based on Equation (3.3). And the other three hash values adopt the same calculation to obtain the weighting hash values. Then all the weighting values are aggregated into one code by summarising four weighting values bit-by-bit. Ultimately, the mapping process uses a piecewise function to generate the fingerprint and assign to the node.

3.2.3.2 Similarity Comparison

NFH extracts the content knowledge automatically from a given node and generates an equal-length bit string of hash value for each node. NHF is used to determine the similarity between nodes and has attractive properties. The advantages of using NFH can be summarised in three points: 1) fast comparison of similarity. With the unified n-length hash value for all nodes, the Hamming Distance computation has much higher efficiency than the semantics-based methods; 2) low maintenance cost. When a certain number of services has been added or removed by the user, the NFH can be re-calculated incrementally. The hash value for each service only needs to be computed once and it will be stored in the cache. If a given service is added/removed from the user's repository, only the hash value of this service needs to generate a new NHF; 3) suitable for dynamic decentralised environments. NFH does not need global vocabularies across the network to model semantics inside a node. Other approaches such as VSM [83, 192, 193] requires global information of inverse document frequency (IDF) to build the node vector. The acquisition of global information is a very expensive operation in a decentralised environment.

The content of a node is represented as a binary sequence. The distance between two sequences is measured by the Hamming Distance which is the number of positions at which the corresponding symbols are different. The Hamming Distance is a number used to denote the difference between the two binary sequences.

The NHF is denoted by $\{0,1\}^n$ which is a collection of all *0-1* binary sequence with length *n*: $\mu = \mu_1\mu_2\mu_3\ldots\mu_n$ *where* $\mu_i \in \{0,1\}, i \in [1,n]$. For $\forall \mu,\gamma \in \{0,1\}^n$, the Hamming Distance is defined as:

$$HD(\mu,\gamma) = \sum_{i=1}^{n} |\mu_i - \gamma_i| \tag{3.4}$$

The normalised Hamming Distance is defined as:

$$NHD(\mu,\gamma) = \frac{1}{n}\sum_{i=1}^{n} |\mu_i - \gamma_i| \tag{3.5}$$

The metric space of NFH with hamming distance is equivalent to a metric space to the set of distances between vertices in a hypercube graph [194]. Since the Hamming

Distance is a metric that satisfies the separation, symmetry and triangle inequality properties, both the $(\{0,1\}^n, HD)$ and $(\{0,1\}^n, NHD)$ are both metric spaces. The metric space, in mathematics, is a set in which distances between all members are defined. Therefore, by using Hamming Distance, the similarity for all the possible values of NHF can be compared and quantified.

The higher distance between two NFH reflects the lower similarity between them. The similarity calculation for NFH is given as:

$$S_{NFH}(\mu,\gamma)=1-\frac{1}{n}\sum_{i=1}^{n}|\mu_i-\gamma_i| \tag{3.6}$$

The normalised Hamming Distance is a value between *0* and *1*, therefore the similarity range is also between *0* and *1*.

3.2.3.3 Examples of Calculation

Figure 3.7 shows an example of calculating the similarity between two NHFs. Algorithm 3.1 generates binary strings with a 32-bit length for node A and node B.

$NFH(A) = 00100011100000000011100001111010$

$NFH(B) = 00100011001000000011100001010010$

By applying Equation (3.4), the Hamming Distance between the two NFH is 4.

Then the normalised Hamming Distance transforms the value to the range between 0 and 1 by dividing the length of the NFH, which is calculated as $\frac{4}{n}=\frac{4}{32}=0.125$. In the practical computation process, the logical operation XOR has been used to compute the distance between two NFHs.

$NFH(A) \otimes NFH(B)$=00000000101000000000000000101000

Therefore, the Hamming Distance is equal to the number of '1' in the XOR operation. In this example, there are four '1' after XOR, so the Hamming Distance between A and B is 4. Finally, the similarity of NHF is derived from the normalised Hamming Distance, thus, the similarity between A and B is *0.875*.

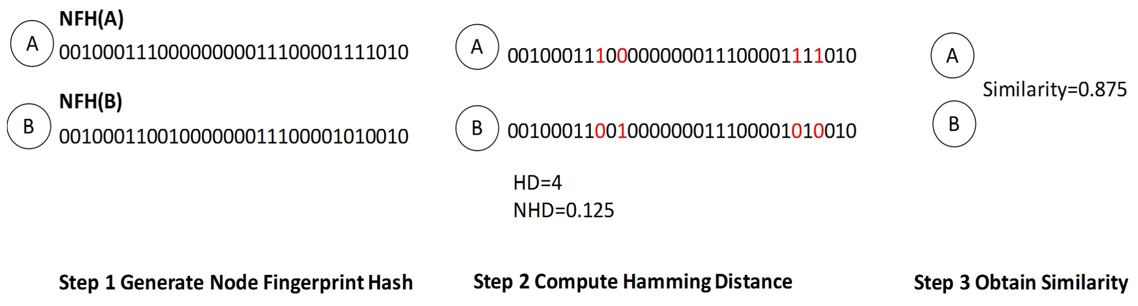


Figure 3.7 The calculation example for similarity between two NFHs

3.2.4 Communications

Previous section discussed the node content abstraction that characterises the service content within a given node. This section describes the query module to support the information exchange and communication between users in the social overlay network. In this research, the communications between users refers to the query and response process over the application layer, which lays the foundation for user's social profile updating and self-organised service discovery. The query module includes (1) query abstractions: which defines the structure of a query, and (2) query functionality: which specifies five basic functions of query routing in the decentralised environment.

3.2.4.1 Query Abstractions

In SDOSN, users utilise queries to self-organise the social network and communicate with each other. The communication is fundamental to support users to conduct service sharing, profile updating and service discovery. This research has designed five types of query messages to support application level communication in the social network.

DEFINITION 3.8: *The query message is defined as an 8-tuple set:*

$$Q = \{(n,u,v,s,p,t,f,c) \mid n \in \Im, u,v \in E, s \subset \Omega(V)_G, p \subset V, t \in Type, f \in \mathbb{R}^+, c \in \mathbb{N}\}$$

where n is the unique identifier for each query; u is the query initiator and v is the query range; s is the query content that is restricted to text vocabulary in the global service repository $\Omega(V)_G$; p is the path consisting of unique identifier of previously visited nodes; t defines the type of query messages; f is the frequency of the message; c is a constant that restricts the maximal hops of a query.

Identifier		Type	Frequency
Initiator	Range	TTL	Path
Content			Checksum

Figure 3.8 Structure of the query message

As shown in Figure 3.8, the query message includes head and body where head part consists of the globally unique identifier, type and frequency information; and the body part contains the query initiator, range, TTL, path and content information. The checksum is used to detect the errors that may be introduced during transmission or storage. The checksum techniques are commonly used in network protocols.

3.2.4.2 Query Functionality

There are five types of messages introduced in this chapter to carry out the communication, content updating and topology adaptation.

General Query (GQ) message: the function of this message is to search desired services in the social network. The content of GQ describes the desired services with user-defined attributes. GQ messages are issued by users to find specific information in the social network and they are automatically forwarded by the social network when the relevant information cannot be found locally.

Specifically, the default setting for the GQ message is given as:

$$Q_{(t=GQ)} = \{(n,u,v,s,p,t,f,c) \mid n \in \Im, u,v \in E, s \in \Omega(V)_G, p \subset V, t = GQ, f = \infty, c = \max TTL)\}$$

where the type is set to GQ, the frequency for this query is unlimited, which means users can issue this query at any time as they like, where the maximum hops are predefined as a constant *maxTTL* by the social network system. The *maxTTL* is the life cycle of a query message. When the hops meet the *maxTTL*, the messages will be dropped regardless of whether found the requested service.

Report Query (RQ) message: this message includes a URI in the query content *s* in order to respond to a GQ message. Upon receiving the RQ message, the GQ initiator uses the URI to retrieve desired services from remote nodes. Otherwise, failure status information, such as exceeding the predefined TTL and service not being available, will be also included in the content of RQ.

Specifically, the default setting for the RQ message is given as:

$$Q_{(t=RQ)} = \{(n,u,v,s,p,t,f,c) \mid n \in \Im, u,v \in E, s = URI, p \in V(u \leftarrow v), t = RQ, f = 1, c = \max TTL)\}$$

The RQ message is sent from the node where query terminates. Therefore, the query path for RQ is a reverse sequence of path response for GQ messages. The frequency of this message is set to 1, which means that there is one RQ message for one GQ message. The content of this message contains a URI or failure states to inform service requesters the results of their requests.

Virgin Query (VQ) message: this message is used to bootstrap newly joining peer nodes when the peer needs to connect other peers. The content of VQ message consists of the shared services and the declared social interests. The purpose of VQ is to obtain at least one neighbour who is already part of the social network, then the new node can join the social network and exchange information with others.

The default setting of VQ messages is given as:

$$Q_{(t=VQ)} = \{(n,u,v,s,p,t,f,c) \mid n \in \Im, u,v \in E, s \subset \{NFH(u), UII(u)\}, p \subset V, t = VQ, f = 1, c = \max TTL)\}$$

The virgin query is only applied when a newly joining user is establishing social connections for the very first time. The query content of this message contains the node fingerprint hash and its interests. The default frequency of VQ is defined as 1. Higher frequency will incur increased overheads since the VQ utilises the flooding approach to cover a wide range of existing network to find appropriate neighbours. The details of utilising VQ to bootstrap newly joining users in social network will be introduced in the next section.

Advertisement Query (AQ) message: this message is periodically multicasted among the first-degree neighbours to update the changes of shared services and interests. The shared service content is transformed into a digested information NFH to reduce the network overhead.

The default setting of AQ messages is given as:

$$Q_{(t=AQ)} = \{(n,u,v,s,p,t,f,c) \mid n \in \Im, u \in E, v \in N(u), s \subset \{NFH(u), UII(u), N(u)\}, p \in V(u \to v), t = AQ, f = \tau_0, c = 1)\}$$

where frequency f is a predefined timer τ_0 to periodically send AQ, and c i.e. the maximum number of hops is set to 1 as this message is only exchanged with u's first-degree neighbours.

Heartbeat Query (HQ) message: this message is used to detect the online presence of nodes, which periodically probe the presence of the first-degree neighbours. Because in an open decentralised social network, users can turn off their device any time, hence making their shared services inaccessible. The HQ is designed to detect nodes status in real-time communication. The default setting of HQ is given as:

$$Q_{(t=HQ)} = \{(n,u,v,s,p,t,f,c) \mid n \in \Im, u \in E, v \in N(u), s = PING, p \in V(u \to v), t = HQ, f = \tau_1, c = 1)\}$$

where the content includes a PING message to check the availability of nodes. The frequency f is a system predefined timer τ_1.

3.3 Distributed Topology Adaptation

3.3.1 The Requirements of Design

The topology adaptation is a fundamental part of decentralised systems. It aims not only to maintain a strong connected social graph, but also to achieve a self-adaptive topology that can provide better performance in the dynamic environment. To be more specific, in the F2F overlay, communications are only enabled when there is a direct social connection.

- The main requirement is to provide mechanisms to organise relevant nodes into semantic groups where nodes are considered relevant to the same query message. In the way, queries may be resolved with many related services in small ranges, hence achieving increased search performance.
- The second requirement is to design algorithms that can direct query messages to different semantic groups effectively via direct neighbours. Query messages can be intelligently routed to dissimilar neighbours to discover services that do not belong to the current semantic group within neighbourhood.

This research exploits an informed search approach by which each node maintains a small portion of the local knowledge of the services shared by other nodes. The index creates an unstructured overlay topology during social interactions and adapts the topology through a distributed information exchange among participants. In the simulation experiments, the index effectively improves the search performance and reduce network traffic in comparison with other search models.

3.3.2 Bootstrapping Stage

Bootstrapping is a vital core functionality required by all P2P networks. The bootstrapping task is a process that nodes should undergo when they are intending to participate in such an overlay network for the purpose of finding at least one neighbour that is already a part of the network. The goal of bootstrapping in a decentralised social network is to recommend some appropriate friend's connections for the newly joining users. Existing online social applications uses Facebook or Twitter accounts to locate the existing friends and to connect with them. However, in a self-organised decentralised system, such information is not available since no central server would host such gigantic information. Therefore, the first challenge here is to define similarity measures to establish some initial social connections for users without prior knowledge; the second challenge is that nodes after joining the social network should adapt social connections to dynamic environments for a guaranteed performance during service discovery.

3.3.2.1 Virgin Session for Friends Discovery

A node joining SDOSN for the first time has an empty friend's connections and empty local knowledge index. The virgin session is a special query process to 1) obtain social connections with those nodes that already present in the social network and 2) gather some useful topic information in the knowledge index to guide the future communication and service discovery. In the virgin session, a newly joining node utilises the VQ messages to obtain initial social connections and gather knowledge from their new neighbours. The goal of friend discovery is to enable each node to establish connections with some semantically similar nodes as well as some dissimilar nodes in the social network.

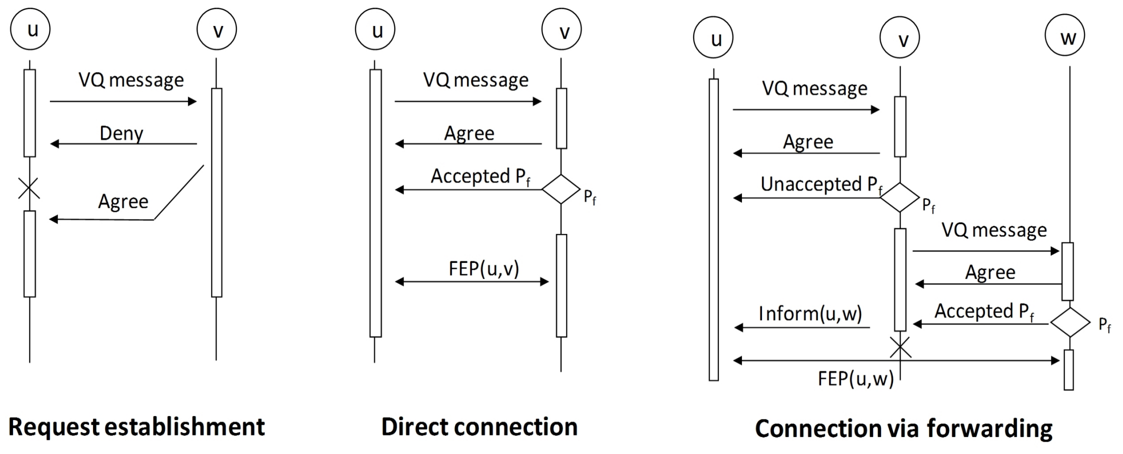


Figure 3.9 Bootstrapping protocol for newly joining nodes

Each node that is already part of the network is considered as an entry point with equal probability to bootstrap a new node. Specifically, if a new node *u* wants to join the social network, it follows the bootstrapping protocol shown in Figure 3.9.

1) Node *u* sends the VQ messages which contains fingerprint hash *NFH(u)* and interest *UII(u)* to a set of random selected bootstrapping nodes $BN = \{v_1, v_2, \ldots v_n\}$ in the system.
2) $\forall v_i \in BN$, if v_i responds to *u* with a refuse message, the interaction between *u* and v_i terminates.
3) Otherwise, v_i accepts the VQ message and starts the similarity calculation. The similarity is measured based on NFH similarity using Equation (3.6) $Sim(u,v_i) = S_{NFH}(u,v_i)$
4) The probability P_f for establishment of a friend connection between *u* and v_i is defined as:

$$P_f(u,v_i) = \left(\frac{1 - Sim(u,v_i)}{C}\right)^{-\eta} \tag{3.7}$$

where $C \in \mathbb{R}$ is a constant that indicates the precision degree of similarity measurement and $\eta \in \mathbb{R}$ is a similarity regulator and used as an adjustable parameter to determine user's preference of establishing friend connection. For $\eta = 0$, P_f will always be 1 regardless of the $Sim(u,v_i)$. In this sense, *u* connects to the entry nodes that have responded the VQ message without considering similarity. When η grows, the user tends to connect nodes with more similarity.

5) If u and v_i establish a friend connection, v_i stops forwarding VQ message from u. Otherwise, the AQ message will be forwarded to neighbours $N(v_i)$ until u receives friend connections via v_i or the TTL in VQ messages finishes.
6) The friend connections are established for both u and remote users through a friendship establishment protocol. Details of this protocol is presented in the next subsection.
7) During the connection between u and v, the *UII(u)* and *UII(v)* are exchanged and recorded in the knowledge index. After this step, the newly joining node u should have obtained the social interests of their neighbours and vice versa.

P_f considers the similarity of content between the two nodes using NFH. As a result, nodes have a greater probability to establish friend connection with other nodes who may have the similar services. The reason of using η to adjust similarity criterion is to give new nodes the chance of establishing social connections not only with similar nodes, but also with dissimilar ones. The dissimilar neighbours can effectively locate other nodes when the query message and query handling nodes (including the query initiator) falls into different semantic groups.

To facilitate the communication and service discovery, this chapter provides two concepts for the friend's connection:

DEFINITION 3.9:

Semantic social tie (Semantic tie for short): $\forall v_i \in N(u)$ *if* $Sim(u,v_i) \geq \varepsilon$, *the similarity of u and* v_i *is above the predefined threshold* ε. *A query route through a semantic tie is called an exploiting walk.*

Correlative social tie (Correlative tie for short): $\forall v_i \in N(u)$ *if* $Sim(u,v_i) < \varepsilon$, *the similarity of u and* v_i *is below the predefined threshold* ε. *A query route through a correlative tie is called an exploring walk.*

A user u's friend connections are classified into two categories based on the similarity threshold ε, such as semantic ties with high similarity and correlative ties with low similarity to u. The goal of defining these two types of social ties is to achieve high recall and low overhead for the query routing. The semantic ties are used to retrieve semantic relevant nodes to resolve a query. A query route via semantic ties indicates the query is semantically close to the current neighbours that potentially have the requested services. Therefore, the exploring walks are often achieved with a high forwarding degree within the neighbours. In this way, more relevant services would be retrieved from the semantic groups. In the meanwhile, correlative ties are used for locating services that differ from the current semantic group of the neighbourhood of u. In this case, the target nodes are not found in the neighbours. The query will be forwarded via correlative ties that have dissimilar semantic content with u. Exploring walks are often achieved with a small forwarding degree since the correlative ties only occupy a small portion of connections of u. In this fashion, the involved nodes to resolve the query is limited to a small fraction in the social network.

3.3.2.2 Friendship Establishment Protocol

The number of friend connections is a predefined number in the system. In some cases, multiple AQ messages will return more candidates than the predefined number of friend connections. When u reaches the maximum number of friend connections, u stops adding the responded remote nodes. However, some remote nodes may have already added u as a friend. In this case, the communication from the remote nodes in the future is not reachable for u, even if u is listed as their friends. This is considered as an incomplete friendship in SDOSN and will trigger a communication fault. Moreover, in the decentralised network, the message exchange is not always reliable as the nodes may join and leave at any time. Messages can be dropped or delayed during transmission, which may also cause the incomplete friendship problem.

In order to cope with this issue, the *Friendship Establishment Protocol* (FEP) is introduced to guarantee a mutual friend connection during the bootstrapping process. Before communication data can be transmitted, a reliable friend connection must be obtained. The FEP protocol is carried out based on a three-way handshake mechanism to enable complete friendships.

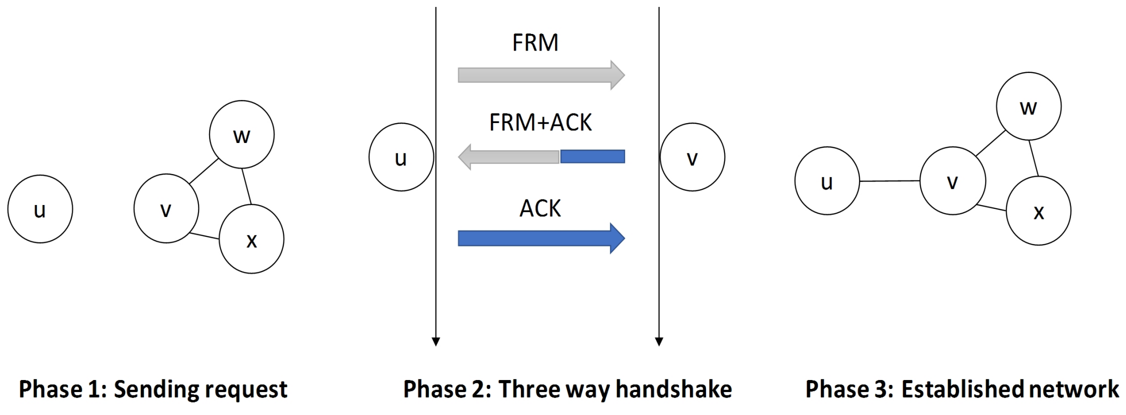


Figure 3.10 The three-way handshake FEP illustration

Friendship Establishment Protocol (FEP)

Input: Node u, v

Output: Friendship connection state

1. Node u sends a *FriendRequestMessage(FRM)* to Node v
2. Node v receives u's *FRM*
3. Node v sends a *FRM-ACKnowledgement*
4. Node u receives v's *FRM-ACK*
5. Node u sends *ACKnowledgement*
6. Node v receives *ACK*
7. Friend connection is *ESTABLISHED*

The three-way handshake is inspired by the TCP (Transmission Control Protocol) to build a reliable connection prior to any data can be transmitted. FEP is able to guarantee that the friendship is a mutual reciprocity between node A and B. The three-way handshake FEB is illustrated in Figure 3.10. In this illustration, node *u* sends a *FRM* to *v* in order to establish a friend connection with *v*. When *v* receives the *FRM* from *u*, *v* will determine whether to accept the request and send back a *FRM-ACK to u*. At this stage, *v* will not add *u* to its friend list until *v* receives *ACK* from *u*. The reason is that in the P2P network, the message exchange is not always reliable as the nodes may join and leave at any time. If *v* adds *u* to the friend list, but the *FRM-ACK* fails to reach back *u*, then *v* will not appear in *u*'s friend list. When *u* receives the *FRM-ACK* from *v* containing the information of acceptance of the friend request, *u* adds *v* to its friend list. Once *v* has been successfully added to *u*'s friend list, *u* will send an *ACK* to *v* in order to notify *v* that the acceptance has been received. Finally, *v* insert *u* to the friend list and update this status to its friends. The first and the second handshake are used to ensure *v* can receive messages and respond correctly to *u*; while the second and third handshake are used to ensure *u* can receive messages and respond correctly to *v*. After successfully applying the three-way handshake, *u* and *v* can establish a bi-directional link between each other.

3.3.2.3 Active Knowledge Accumulation

Initially, when a new node attempts to join a decentralised network for the first time, it has no connections to others and has no knowledge of the network. Therefore, a specific discovery called *Virgin Session* is required to connect the node to the social network. After joining the social network, the node sends a limited number of VQ messages to a set of selected neighbours. If the discovery is successful, the recently joined node updates its NSI and LKI to associate the nodes that have responded data successfully with the requested topic and establish connections to these nodes.

As defined in section 3.2.4, the content of the VQ message comprises the node fingerprint hash and terms of the publicly declared social interest. The node fingerprint hash is used to gain at least one neighbour that meets the similarity requirement. At the same time, the social interest terms are used to discover the contents from their latest neighbours and to build the topic-node shortcuts in the NSI and further to update the LKI about the new neighbours. The aim of the VQ message is to capture the neighbour's content at the beginning of the friendship, ultimately to increase the success rate of service discovery for the newly joined nodes.

Traditional methods utilise a query which consists of multiple terms to search for a service in the network. When a target node handles the query, it responds the query requester with information only corresponding to the query topics (the terms in the query). Often, a node needs to issue many query messages to gain the knowledge from the network. In this manner, message overheads are significantly high and it is inefficient to search user inexplicit query topics such as polysemanic terms and synonyms.

Therefore, this research proposes a fast knowledge acquisition approach to fetch a batch of information with VQ messages during the Virgin Session. The VQ 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 in the content of VQ, but also inform the originator of other associated topics shared within the same interest area. An interest area in SDOSN is a semantic group 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 [195], 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. For example, if the originator generates a query with the topic *"Paintball"* in the content, 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 VQ messages, the originator can gather more pieces of knowledge from each successful query; however, additional traffic will be generated when transferring this additional knowledge. This 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 the search is undertaken with only ordinary queries (without expansion), each new node accumulates the 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 utilise the VQ messages to gather a large amount of useful information quickly based on their interests. After the amount of cached knowledge reaching a certain threshold ratio with respect to the maximum size of the knowledge index, the nodes will utilise GQ messages to discover the required files with low traffic cost.

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 this process continues, each node will cache a great deal of useful knowledge that is effective in making future decisions to re-visit peer nodes having access to the required services.

3.3.3 Handling Dynamism

Since SDOSN operates the service discovery process on an unstructured P2P network, the network dynamism is one of the main challenges for maintaining the social overlay network and managing the services.

Dynamism features of the social overlay network are studied from two aspects such as friendship network and overlay network. In SDOSN, new friendship relationships are continually being formed or old ones are being severed. However, in real-world OSN systems, such changes are infrequent to the extent that they are insignificant within the time scale of the problems of information dissemination and service discovery. For this reason, friendship network is considered immutable.

The dynamism of the overlay network 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 may not be available for some periods. This phenomenon, in P2P terminology, is known as the churn. The overlay network undergoes frequent reconfiguration, which affects the performance of information propagation and service discovery.

When a node leaves the social network, it makes the services shared by the corresponding node unavailable. Therefore, if node u disconnects from the network, it notifies all online direct neighbours about its leaving. The offline neighbours of u, they detect this change when they come online for the next time by sending HQ messages to probe the presence of u. The neighbour nodes update their knowledge index by setting the neighbour's list of u as *-1*. When u re-joins the network, the neighbour's degree needs to be updated accordingly.

Each node also continuously maintains the content hash, social interest and friends list of neighbours by periodically sending advertisement query messages to its existing neighbours; neighbours responds to the request with its NHF, interests and friends list. Upon receiving this response, the up-to-date information of neighbours is obtained and stored in the knowledge index. The fresh information of NHF is used to renew the semantic group of node's neighbourhood. The node checks the similarity score of its semantic ties and correlative ties: if the similarity score of a semantic tie drops below the predefined threshold ε due to the content change of either nodes, SDOSN simply transforms the semantic tie to the correlative tie; similarly, when the similarity of a correlative tie increases above the threshold, the correlative tie becomes a semantic tie.

The tasks of neighbour maintenance are to obtain the availability of neighbours and to keep track of the up-to-date information. Because of dynamism handling, the social network can adapt to dynamic changes of the nodes and their shared services.

3.4 Simulation Methodology

Simulation is a cost-effective method than implementing physical systems to evaluate the performance of the methodology proposed in this research. A software simulation platform has been designed and implemented to simulate a real world unstructured P2P network with configurable routing protocols and with basic social network functions.

This section will discuss the design of experiments and evaluations of the proposed architecture of SDOSN. The performance of the proposed methodology has been evaluated by simulations under a dynamic environment. The simulator has been programmed in Java. Programming environment: Lenovo Thinkstation with specifications of Intel Core i5-6400 Processor, 16GB RAM, Windows 10 Pro X64. The implementation of the simulator is based on Java SDK 1.80 and Eclipse IDE. The main component of the SDOSN simulator is illustrated in Figure 3.11 .

3.4.1 Simulation Platform

The simulation platform is a P2P social network simulator which can simulate self-organised topological construction and service discovery under a dynamic network environment. Firstly, the components of the platform are explained and illustrated. Then the experimental dataset has been prepared and assigned to nodes. Then the performance metrics and parameter settings are introduced to compare the network topology structures and search performance.

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

At the network initialisation stage, the simulation creates a node instance by allocating a range of memory spaces and establishing the social attributes (such as service contents, friends list and a set of interest tags) in an object-oriented design. Then the attributes are assigned with values of real-world datasets. Furthermore, every node instance generates a local service index and a knowledge index to utilise 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 neighbours. The node instances, social connections and services are generated in a batch mode based on different network configurations.

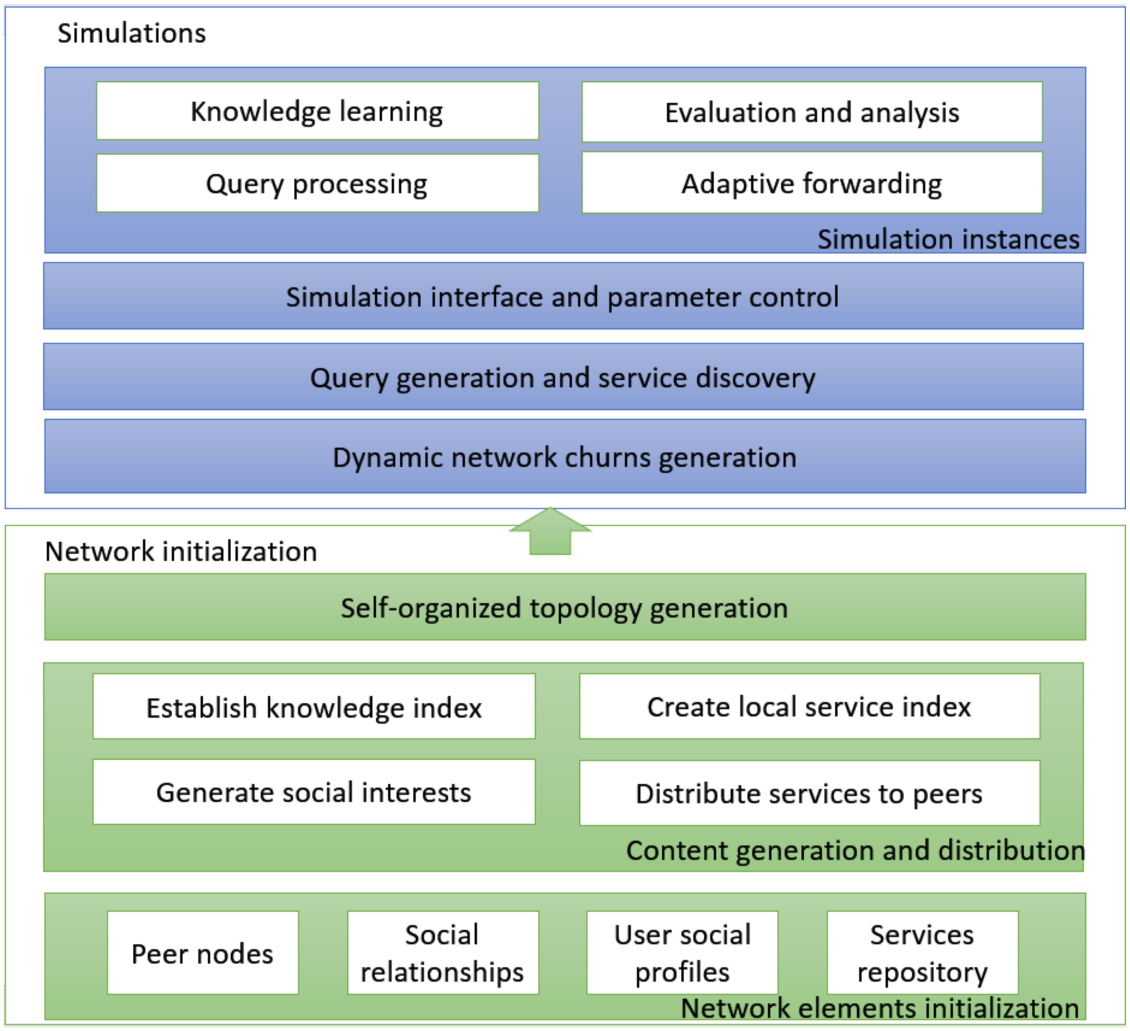

Simulations
Knowledge learning
Evaluation and analysis
Query processing
Adaptive forwarding
Simulation instances
Simulation interface and parameter control
Query generation and service discovery
Dynamic network churns generation
Network initialization
Self-organized topology generation
Establish knowledge index
Create local service index
Generate social interests
Distribute services to peers
Content generation and distribution
Peer nodes
Social relationships
User social profiles
Services repository
Network elements initialization

Figure 3.11 Simulation platform components

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 becoming unable to provide services in a simulation instance. This phenomenon is called as the churn in a decentralised social network, and it is very common in real P2P 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 parameter *maxTTL* is the upper bound hops limitations for a given query. A query is forwarded through neighbours to conduct service discovery by utilising the knowledge index, and the process continues until either the query has been resolved or until its TTL exceeds the *maxTTL*. 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.

3.4.2 Evaluation Setup

The simulation conducted in this research considers four main characteristics that are considered such as network topology, content distribution, query generation and search methods.

3.4.2.1 Topology Initialisation and Evolution

In order to simulate the evolution of the social network topology, at starting point, a random network structure comprising 100 nodes has been initially selected. Each node randomly connects to two nodes bi-directionally. Then a number of new nodes (approximately 30 nodes) join the network every minute until the size of network reaches 1000 nodes. Now the mature network comprises 1000 nodes and service discovery is conducted in this mature network.

3.4.2.2 Content Distribution

The service distribution among the social network can affect the search performances. In order to simulate a realistic environment, the service popularity has been considered in the simulation. A few popular services have many duplicates that appear in many nodes in the network, on the other hand, a large number of services are less popular and only appear in few nodes. In this way, popular services are easier to be discovered.

This study uses the DMOZ dataset [195] and the Social-ODP-2k9 dataset [196] for experiments. The DMOZ dataset is the most widely distributed database of Web content using a hierarchy topic structure and Social-ODP-2k9 is a dataset with data retrieved from the social bookmarking sites. This dataset comprises *12,616* unique URLs with their corresponding social annotations in the DMOZ. The annotated URLs are treated as social services in this experiment. The content of each service comprises 10 annotated topics.

These services are distributed among the nodes based on the following strategy: the Zipf law is in the form of $y \sim \frac{1}{x^{\gamma}}$, where y is the frequency, x is the rank and γ is a constant, and this law has been applied to generate service frequency in the entire network. The total number of services in the network is 173,846, with the max frequency 691 and min frequency 7. The frequency of services is shown in Figure 3.12 with $\gamma = 2$.Then a uniform distribution has been applied to distribute all service onto the nodes in the network. After this strategy, each node in the network has approximately 174 services.

Each node focuses on one or a few topics depending on its special interest. In this simulation, the interest topics are extracted from the DMOZ taxonomy. The interests are assigned to users based on the assumption that each node is an expert on some interest areas and possesses the services in these areas. Therefore, 80% interests are selected from the top frequent topics in user's content and the remaining 20% are randomly selected from the DMOZ. Note that topics of interests are all sub-topics of the DMOZ taxonomy. The hierarchical relationship between topics is only used to model the interests but not for query routing in this chapter.

3.4.2.3 Query Generation

The queries are launched uniformly within the network during the simulation. Query topics are randomly selected from the social annotations and the total number of query topics is 1000, which includes topics for both popular services and rare services. 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 queries. Each query message will be forwarded by the nodes to locate as many relevant services as possible until the query reaches the *maxTTL*. The *maxTTL* varied from 1 to 12 and the default value is set to 3.

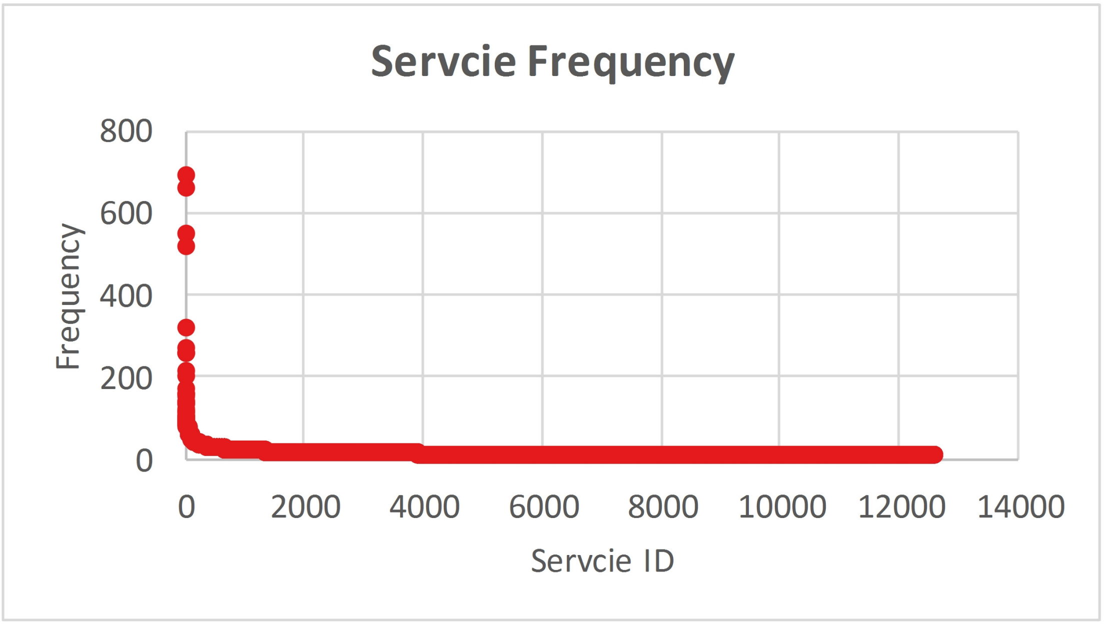


Figure 3.12 Service frequency based on the Zipf law

The network has been simulated for 10,000 GQ messages, each node issuing approximately 10 queries, to observe the average performance for all the queries issued by the same node. The query messages are first evaluated against the local knowledge index and then are propagated to neighbours according to search strategies.

3.4.2.4 Search Methods

The goal of search in this chapter is to identify the correctness of the proposed SDOSN. The underlying concept is to send query messages to find proper neighbours and desired services in the decentralised system. During simulation, online nodes are randomly chosen as service requesting nodes in order to start a search with a single topic. The query match is simply defined as the services inside a node containing the query topic in the annotation. The successfully matched node responds a report query message to the query initiator.

3.4.3 Performance Metrics

The network architecture provides the network topology for participants to carry out social interactions and service discovery. Ideally, an effective network architecture should have good search performance and low network maintenance overhead. The metrics used to validate the search performance and maintenance cost of SDOSN are summarised as follows:

- ***Success rate***: the ratio of the number of resolved queries to the number of all issued queries within a given time in the network. A query is deemed unsuccessful if the query initiator has not received responses at the end of the session.
- ***Resource cost***: the number of nodes involved to resolve the queries during the search process.
- ***Query efficiency***: it is the ratio of success rate to resource cost. This metric evaluates the overall effectiveness of a search strategy. Ideally, a service discovery approach with highest success rate and lowest resource cost is the most efficient strategy.

3.4.4 Performance Comparison

3.4.4.1 Evaluation of Bootstrapping

The connection probability P_f is the probability for a newly joining user *u* to connect *v*$_i$ that is already present in the social network, which is defined as $P_f(u,v_i) = (\frac{1-Sim(u,v_i)}{C})^{-\eta}\ i \in [1,n]$, where n is the total number of nodes in the network and the metric $Sim(u,v)$ measures the content similarity between two given node using their NFHs.

Based on the service distribution in the network, the similarity between all nodes are observed and the highest similarity value is 0.6. Therefore, the similarity for connection between nodes is evaluated in the range between 0.1 and 0.6, and the regulator η is evaluated in the range between 1 and 10. The constant *C* is set as 0.4 in the experiment. When a newly joining node initially establishes some social connections in the network, P_f is used to decide the connection probability based on the similarity between nodes. Figure 3.13 depicts the impacts of the regulator upon the connection probability.

With the regulator $\eta = 0$, the connection probability is always 1, which means that the node *u* connects *v*$_i$ directly without considering similarity. In this sense, all the new connections for *u* is randomly selected and there is no controlled similarity between *u* and its neighbours. With the increase in the regulator η, the connection probability rapidly decreases; when η approaches 10, the connection probability is nearly 0. It can be observed from Figure 3.13 that smaller η creates larger probability to connect those nodes characterising less similarity. Even with $Sim(u,v)=0.1$, the connection probability is approximately 0.44. When η approaches 0, the probability of connecting the most dissimilar node is nearly the same probability of connecting the most similar node.

In the parameter selection, in order to connect approximately 10% dissimilar nodes with $Sim(u,v) \le 0.1$, the regulator η is set as 2 in the bootstrapping process. In this way, newly joining nodes forms different semantic groups for query routing. Consequently, the average length of search paths in service discovery is shorter.

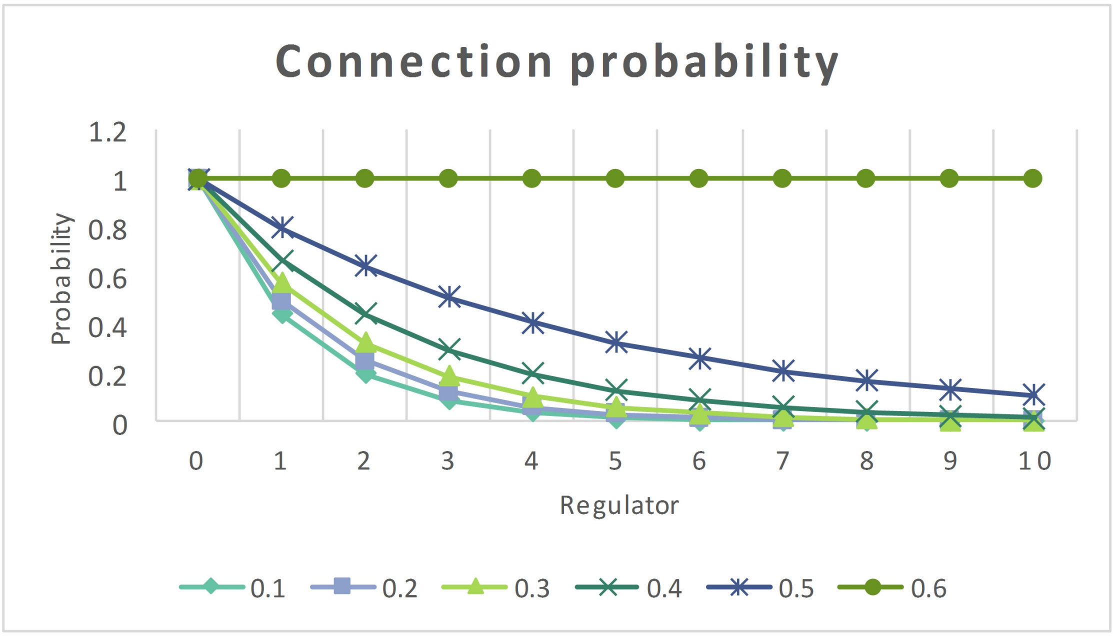


Figure 3.13 Effects of regulator upon connection probability

3.4.4.2 Topology Comparison

According to the small-world theory [99], human social networks exhibit clustering effects and has a short average path length as of 6. In other words, people are organised in tightly connected social communities, and the distance between two people in the social is six or fewer steps. Therefore, during the simulation, the topology evolution has been monitored to observe the effects of the below properties.

Average clustering coefficient (ACC): this metric quantifies the node cluster effect of the models. $ACC = \frac{1}{N}\sum_{i=1}^{N}\frac{2E}{k_i(k_i-1)}$ where k_i represents the number of neighbours of node *u*, and *E* is the number of possible connections between the k_i neighbours.

Average path length (APL): this metric calculates the average shortest distance between any two nodes in the network. It is a measure of the efficiency of information exchanged in a network. The shorter average path length is more desirable in social networks. $APL = \frac{2}{n \cdot (n-1)} \cdot \sum_{i \geq j} d(v_i, v_j)$, where $d(v_i, v_j)$ is the shortest graph distance between v_i and v_j.

Network topology comparison considers three types of topology construction in complex networks including random networks, scale-free networks and NFH-based networks. The topological establishment of these networks are achieved based on the following strategies:

Random Network: where connections between nodes are established randomly with probability $0 < p < 1$ according to Erdős–Rényi model [197].

Scale-Free Network: where connections between nodes are established based on the degree of connection and the degree distribution follows a power law $p(k) \sim k^{-\gamma}$. Nodes with a high degree of connectivity have a greater probability of receiving new links than agents with a low degree of connection [100]. The network is composed of a few number of nodes with a lot of social connections and several nodes with a very few connections accordingly.

NFH-based Network: where connections between nodes are established based on content similarity represented as NFH. Nodes tend to connect others who share greater similarity in content and small semantic communities are built in neighbourhood. The bridge between the communities is achieved with a small portion of dissimilar nodes which brings randomness and mutability in discovery among different semantic communities.

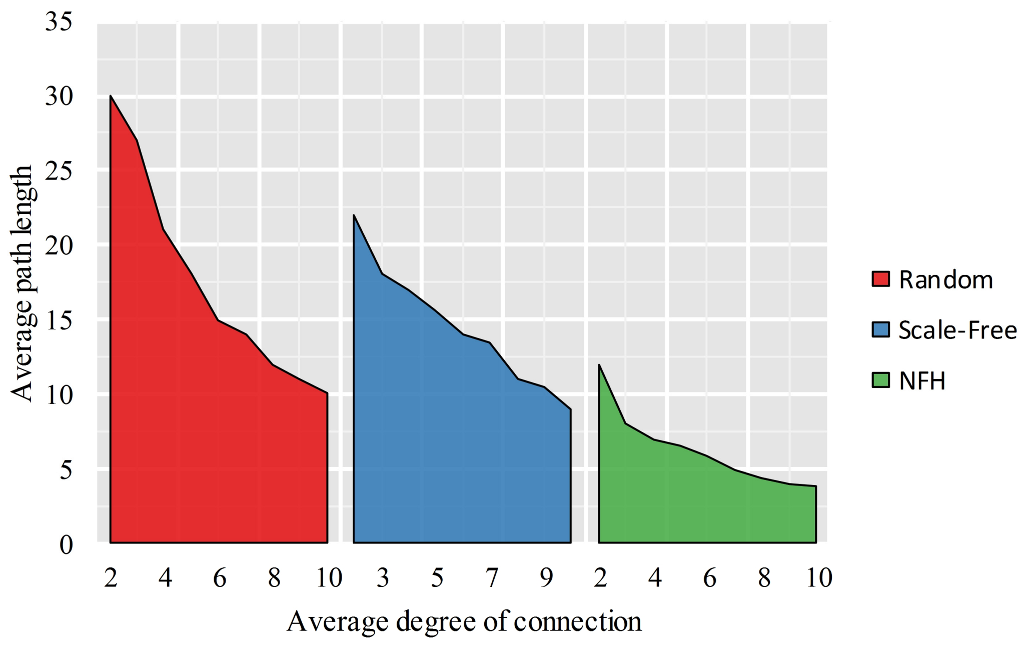


Figure 3.14 Influence of average degree of connection on average path length

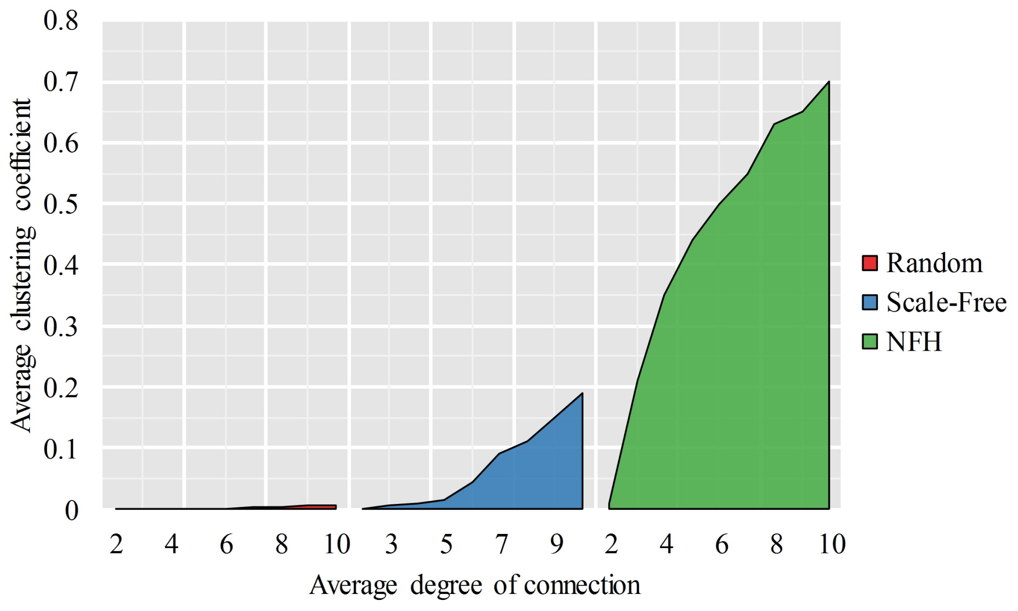


Figure 3.15 Influence of average degree of connection on average clustering coefficient

From the observation presented in Figure 3.14, the average number of connections increases when APL decreases. Three different networks including the random topology, scale-free topology and NFH-based topology perform similarly in this sense. The NFH topology characterises the shortest average path length with the same number of nodes and average degrees of connection. This is because the fact that the high degree of connection increases the possibility of finding available nodes and the smaller average path length facilitates the service discovery within a shorter range.

As shown in Figure 3.15, the ACC of NFH is much greater than the other two networks with the same number of connections. The ACC increases rapidly as the connections in NFH grows. A large network ACC value reflects better cluster capability of nodes having similar contents. From the results shown in Figure 3.14 and Figure 3.15, it can be observed that the small-world phenomenon in the NFH-based networks with a high clustering coefficient and a short average path length can effectively reduce the service discovery time and improve the efficiency by finding more relevant services within a short range of TTL.

3.4.4.3 Evaluation of NSI and LKI on Search Performance

The search strategies are generally classified as uninformed and informed approaches. The uniformed approach only relies on the local content and does not have any other information about other nodes in the network; the query routing is carried out in a blind way. But the informed approach utilises both local content and the historical search information to guide the query routing. In the experiments, the LSI-based search has been considered as the uninformed approach and it is used as the base line for evaluating the search performance. Compared routing strategies include the following:

LSI: a blind query routing strategy that utilises k-random walker algorithm to forward query messages to *k* direct neighbours.

NSI: a network shortcut based query routing strategy that utilises the historical network shortcuts to facilitate the query routing. If there is a match in NSI, a RQ message containing the URI information of the requested service is sent back to the query initiator. If there is no match found, the query is then forwarded to *k* direct neighbours.

LKI: a knowledge index based query routing strategy that utilises successful query topics via direct neighbours. This topic information provides only the hints instead of direct service access path i.e. URI. The query is forwarded to *k* neighbouring nodes exhibiting higher frequency for the interested topics.

Parameter setting: The query routing in LSI-based search has been achieved based on a k-random walker and with *k*=2. The forwarding degree in NSI and LKI is set as 2 in each hop. The *maxTTL* is evaluated from 1 to 12, which restricts the max travelling hops for a GQ query message. The topic size of NSI is set as 100, and a topic can have a list of URIs of service providers. The topic size of LKI is also set as 100, and a topic is only correlated with the neighbours that returned a success response. The simulation time is 30 minutes, each node is assigned approximately 10 queries uniformly during the entire simulation. The total query in the network during every minute is approximately 334. The query generation is shown in Figure 3.16.

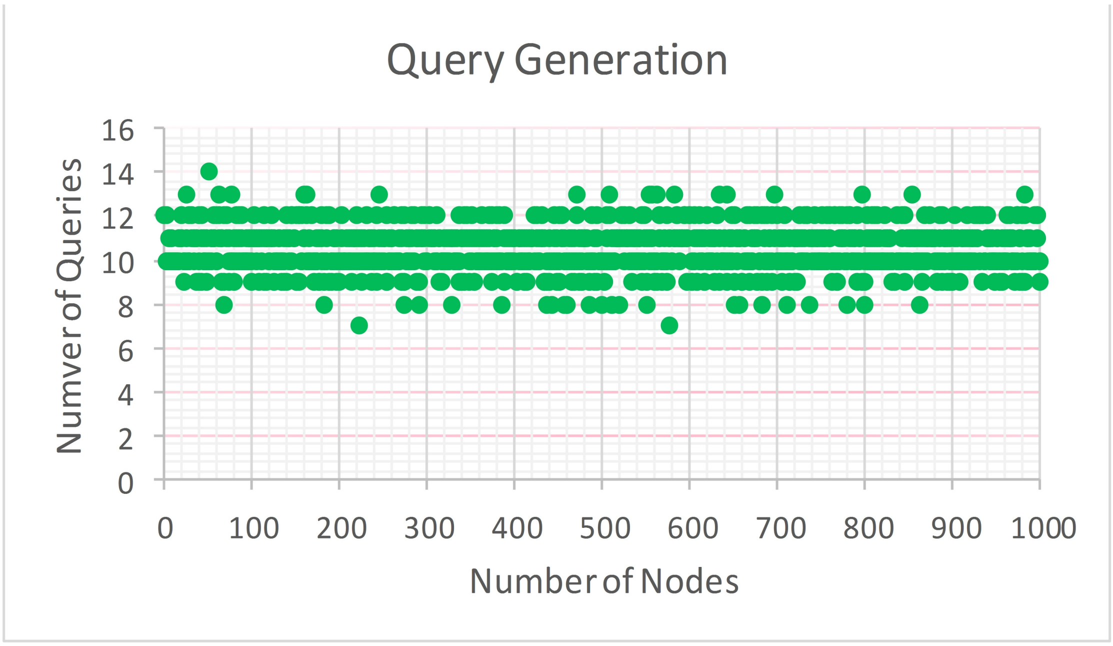


Figure 3.16 Uniformly query generation for each node

If the query has found a matching service in a node, the node sends a RQ message containing a URI to the query initiator. The service discovery is successful if the query initiator receives such a RQ message. After a query time out, the query messages are terminated and no message will be returned in order to reduce the network traffic. The success rate, resource cost and query efficiency are evaluated with 1000 nodes in the social network. The content distribution and friend connections are assumed static during the service discovery process.

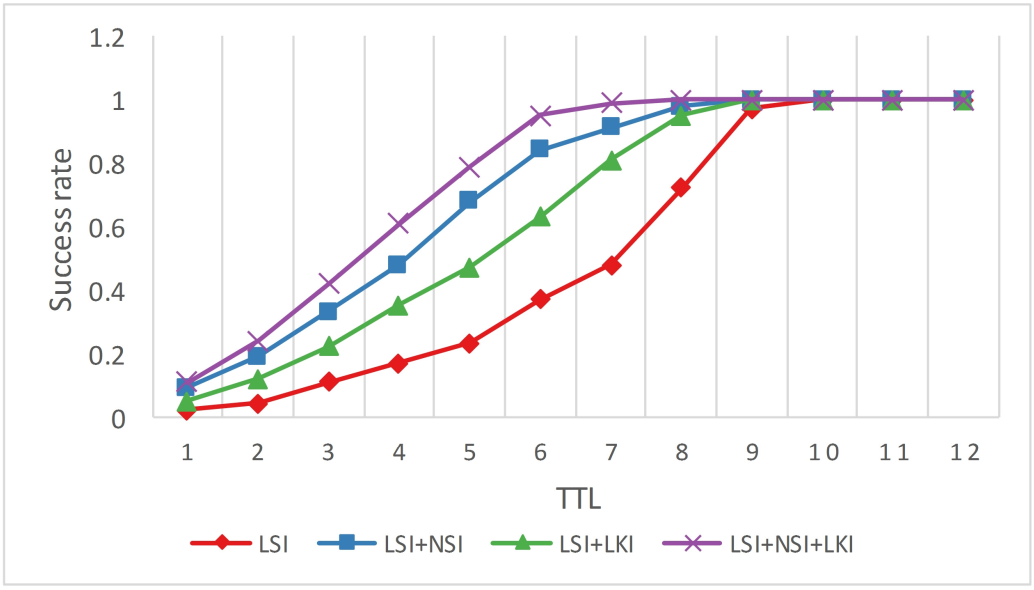


Figure 3.17 Evaluation of the success rate of LSI, NSI and LKI

The success rate of the compared approach is shown in Figure 3.17. Generally, the informed approaches LSI+NSI, LSI+LKI and LSI+NSI+LKI outperforms the uninformed approach LSI. As the TTL increases, the success rate of search in all approaches increases. This is due to the fact that more nodes will be involved in the search and more candidate services can be discovered. During the search process, for each fixed TTL value, the informed approaches achieve higher success rate than the uninformed approach, achieving nearly a 50% improvement on success rate. The informed approaches require less TTL for a successful search, reflecting that less number of nodes are visited in the network to locate the requested services. The success rate of NSI is higher than LKI, and the combination of NSI and LKI exhibits the highest success rate. The NSI uses the historical query hits, whereby facilitating a direct visit to the requested service for a matched topic in query messages. LKI only stores the successful topic hits via direct neighbours, which has a limited view of the network since the topics via neighbours only provide the hints for the query routing. Thus, NSI is exhibiting a better performance than LKI.

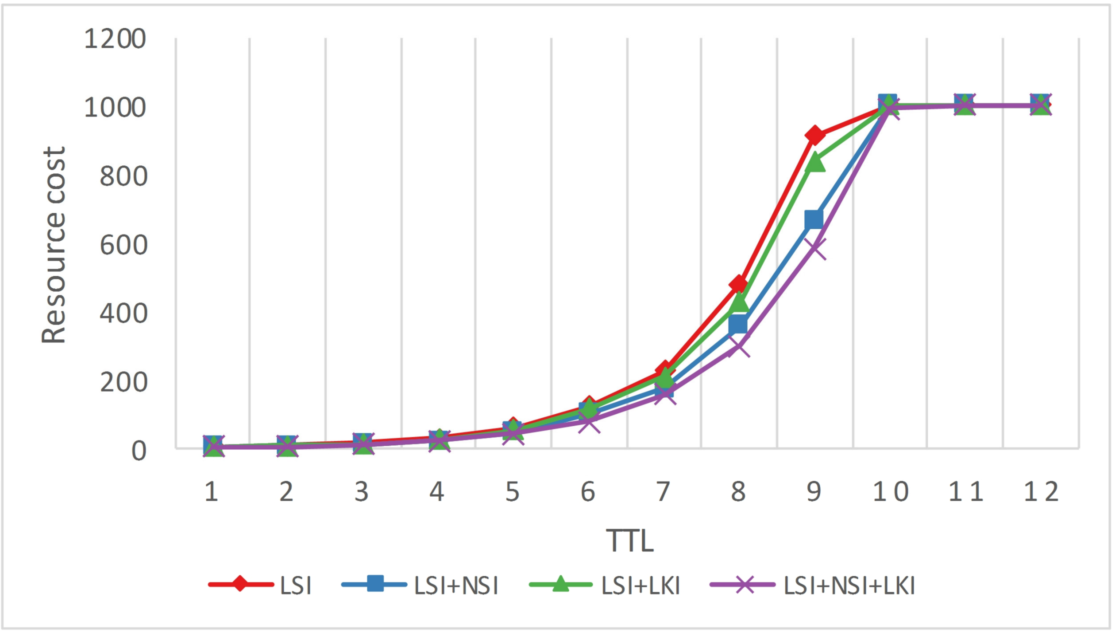


Figure 3.18 Evaluation of the resource cost of LSI, NSI and LKI

Figure 3.18 shows the average resource cost for resolving a query. The resource cost is usually the number of involved nodes to resolve the query. A lower resource cost reflects a better scalability for a given system. In order to resolve a query with less TTL, nodes should forward queries to as many nodes as possible in each hop. In our experiment, the forwarding degree is set to 2, which means that if there is no match in a certain hop, the total message will be doubled in the next step. If the requested service has been found, the service provider stops forwarding the query. As the TTL increases, all compared strategies visit more nodes in each hop until the query has travelled across all possible nodes in the network. At TTL=10, the query messages have saturated the network. As shown in Figure 3.18, both LSI+NSI and LSI+LKI outperforms the baseline method LSI, since the query routing is guided by historical knowledge as opposed to the blind strategy of LSI. In addition, LSI+NSI consumes fewer resources than LSI+LKI. This is because of the fact that the former strategy generates direct service shortcuts and stops forwarding when the shortcuts have been found. On the other hand, if a query matches the topics in LKI, the query still needs to be forwarded to the corresponding neighbours. When the TTL is less than 10, the resource cost for the combination of LSI+NSI+LKI is approximately 55% less than those of the LSI for answering a query. Since the resource cost is proportional to the duplicated query messages, hence, the network traffic is also reduced significantly.

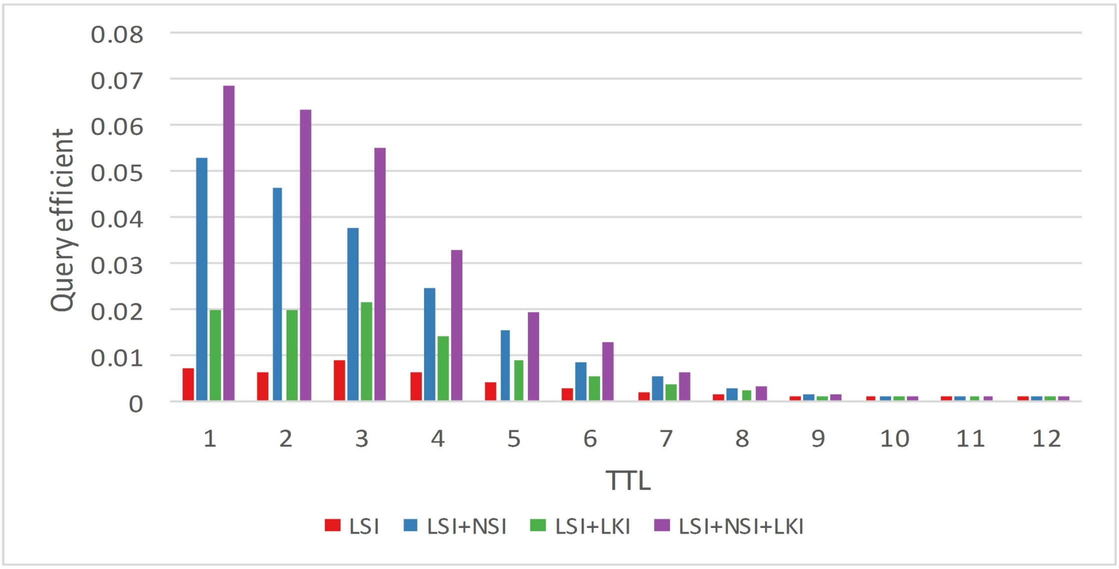


Figure 3.19 Evaluation of the query efficiency of LSI, NSI and LKI

The query efficiency is shown in Figure 3.19, demonstrating the query efficiency of different routing strategies under different TTL range. Since the informed approaches such as NSI and LKI have relatively higher success rate and less resource cost, its query efficiency outperforms the uninformed approach LSI. Within a small TTL, the search efficiency of LSI is very low since the query traverses only through to a limited number of nodes. However, informed approaches achieve a higher query efficiency especially when the TTL is less than 5, which shows the effectiveness of NSI and LKI for small TTL. This feature is favourable for service discovery in the decentralised environment, since it can discover more requested services without incurring a large number of duplicated messages in the network. As the TTL increases the resource costs increase exponentially, therefore the query efficiency considerably decreases for all the strategies. Hereby, it can be concluded that in order to improve the query efficiency, the search strategies should find the shortest path to resolve a query and to reduce the number of nodes involved in the query routing process.

3.4.4.4 Search Methods Comparison

In this section, three different search methods in decentralised service discovery has been evaluated through neighbour selection in each step. These strategies are described as follows:

Random: a search strategy that utilises random walks to forward queries [53, 77].

Degree: a search strategy that utilises degree information to forward queries to the neighbour that has the highest degrees of connection [57, 148, 198].

Knowledge: a search strategy that utilises LSI+NSI+LKI to forward queries to semantically relevant neighbours.

These three search methods have been evaluated in the NFH-based network with an average degree of 3.5. The results of experiments are shown in Figure 3.20, Figure 3.21 and Figure 3.22. In general, the influence of TTL parameter on the search performance is significant, since longer TTL can include more relevant nodes in the network, thereby achieving higher success rate as well as higher resource cost. This trend has been observed in all three search methods.

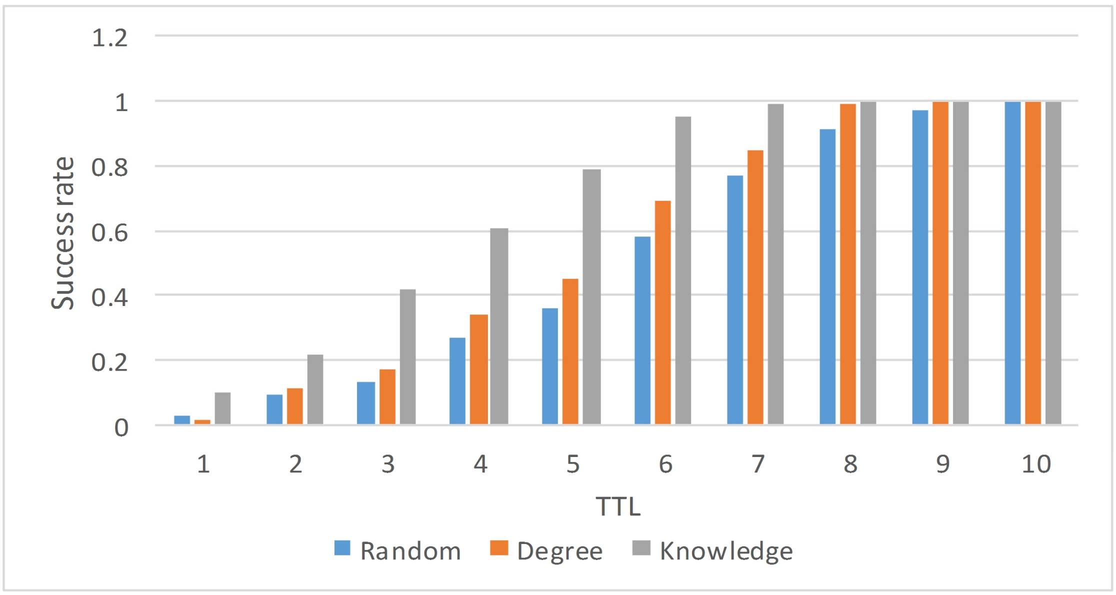


Figure 3.20 Success rate of different search methods

In Figure 3.20, the x-axis depicts the different search methods and the y-axis depicts the success rate, which is the percentage of queries resolved before the TTL expires. The knowledge-based search achieves the highest success rate for different TTL ranges. The success rate of the degree-based search is slightly better than that of the random search method. The average success rate of the knowledge-based search is approximately twice than that of the other two methods for the TTL ranging from 1 to 4, which means that the percentage of queries being resolved is doubled by utilising the knowledge in each query hop. For TTL=6, the success rate of the knowledge-based search achieves nearly reaches 100%, where the success rate of the degree-based search is 70% and the random search is less than 60%. When TTL approaches 10, the success rates of the latter two methods is observed to be reaching 100%. It can be concluded that the knowledge based search provides the highest success rate and can resolve most queries within short TTL range.

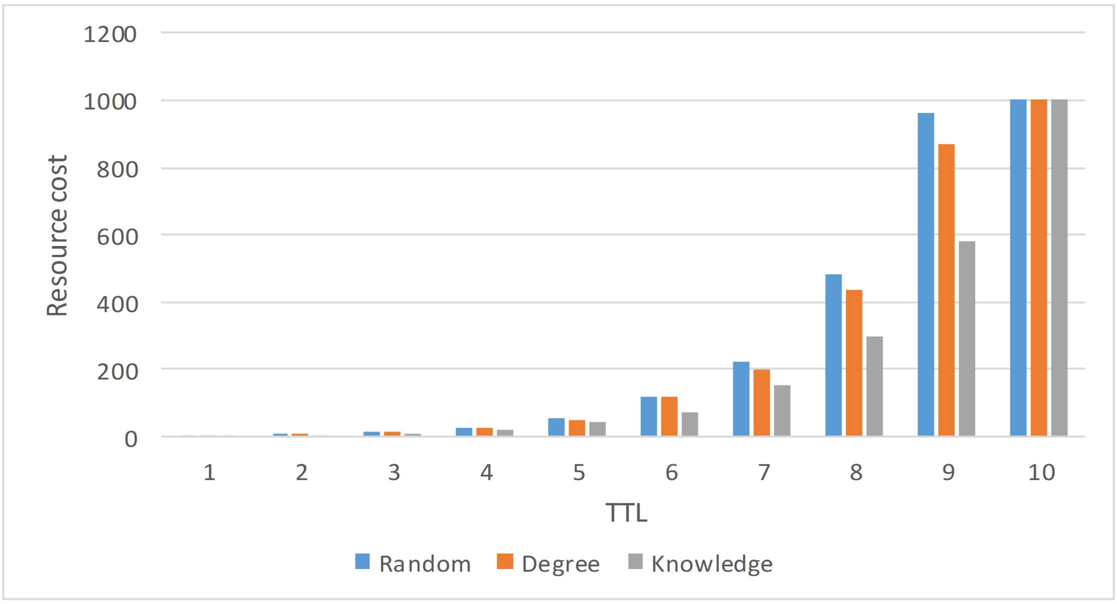


Figure 3.21 Resource cost of different search methods

TTL parameter is observed to exert a higher influence on the search performance. Longer TTL can achieve higher success rate. However, long TTL also can increase the resource cost and generate significant duplicated messages in the network. Figure 3.21 shows the resource cost for different search methods, where the x-axis represents the search methods under different TTL range, and the y-axis represents the average number of visited nodes required to resolve a query. The knowledge-based search method outperforms the other two methods. For TTL range 1 to 10, the knowledge based search resolves queries with least number of involved nodes. The degree-based search method has slightly less number of visited nodes than the random search method. It is worthy of note that the GUID of query messages has been applied to prevent queries with the same GUID from being forwarded to the same nodes twice. The involved nodes to resolve a query includes all the network nodes when the TTL hits 10.

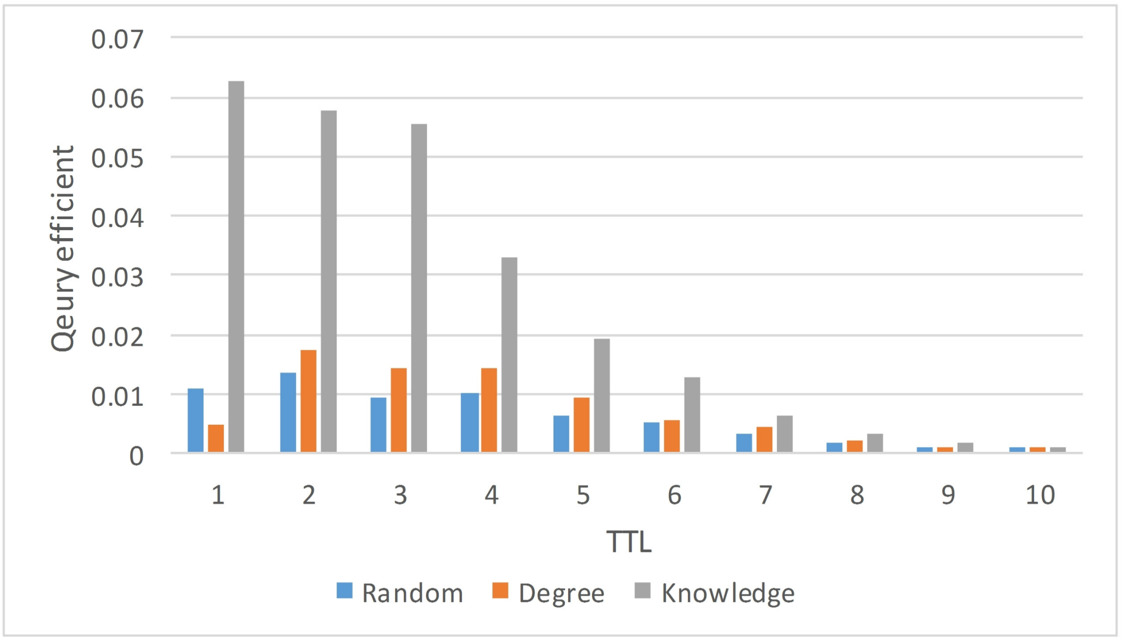


Figure 3.22 Query efficiency of different search methods

Figure 3.22 presents the query efficiency for the three search methods. There is no significant difference observed between the degree-based search method and random search method, due to the small-world nature of the NFH-based network. The NFH-based network comprises semantic communities those have connections between any two nodes within the community. As the result, variance among the degree of connection is small in such networks. Moreover, from the query efficiency observations, it can be concluded that the knowledge-based search is achieving the highest performance in search especially for small TTL values. This is because of the fact that it has the higher percentage of successful hits and the fewer number of visited nodes.

From these observations, it can be concluded that under different TTL ranges the knowledge-based search can locate the required services with a higher success rate and relatively lower resource cost. Thus, the query traffic is reduced and the efficiency is improved during the service discovery process.

3.4.4.5 Bootstrapping Overhead

Now, 30 new nodes are imposed to join the social network after a minute into the simulation. Nodes arrive randomly and send query requests to establish connections with the existing nodes in the social network. The initial social network is a randomly connected graph that has 100 nodes with an average degree of 4. The newly joining nodes send VQ messages to discover friendship candidates. The *maxTTL* for VQ messages is set to 3 as a default value. As shown in Figure 3.23, the size of the network grows steadily until a reaching predefined system size of 1000 nodes in our simulation. At 30 minutes of simulation, the network reaches the size of 1000 nodes and remains the same for the rest of the simulation. During every minute of simulation, each node in the network including the newly joined nodes issue one GQ messages with the *maxTTL* of 3.

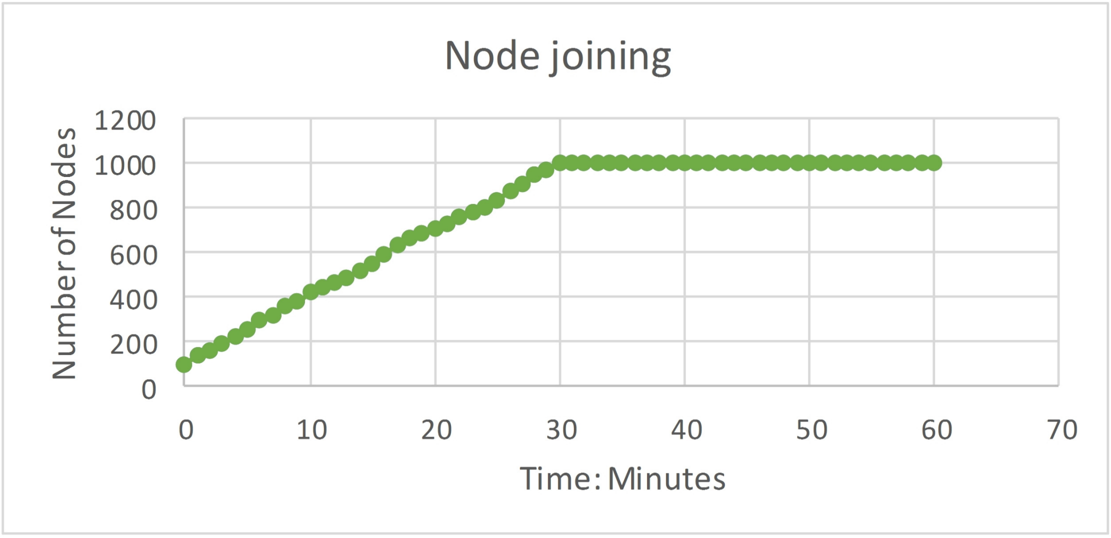


Figure 3.23 Nodes joining the social network

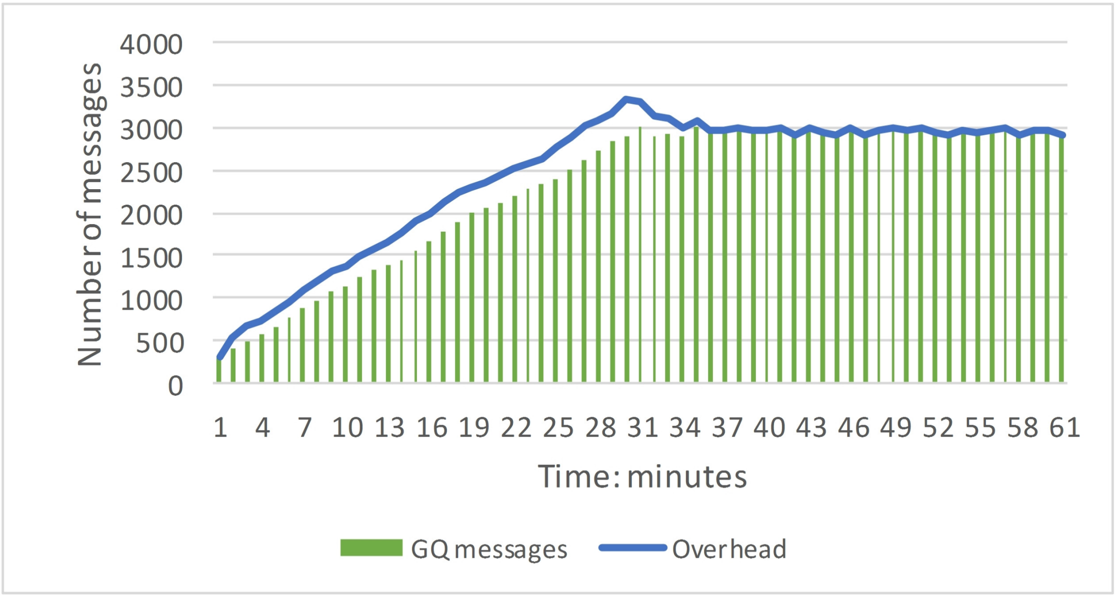


Figure 3.24 Message overhead for newly joining nodes

The network traffic consists of two parts such as the VQ messages and GQ messages as shown in Figure 3.24. GQ messages and VQ messages are both increasing when nodes join the network. The amount of overhead is marginal, represented in the area between the blue curve and green histogram. This overhead is incurred by the VQ messages flooding the network to establish new links. The total messages overhead reach the peak of 3337 messages in total at the 29th minute, witnessing a 14% increase incurred by the VQ messages. The highest increment during the simulation is observed to be approximately 33% at the 3rd minute of the simulation.

The network traffic depends on the entry points (bootstrapping nodes) of the social network, existing network connections and the *maxTTL* of the VQ messages. The entry points are selected randomly in this research since all participants are considered equally for the role of bootstrapping new nodes. In this sense, there is no extra maintenance incurred for bootstrapping the nodes, thus maintaining the workload balancing in the network. As more nodes joining the network, the connections of the nodes increase rapidly. Therefore, the messages are flooded among an increased number of nodes in each hop. However, the message overheads increase steadily during social connections whilst exerting a control over nodes joining the network. After 30 minutes of simulation, there are still some messages being transmitted in the network. This due to the fact that the TTL of VQ messages is not yet finished, thus imposing a delay in the response messages of friend’s connections.

3.5 Summary

This chapter proposed a decentralised alternative SDOSN featuring a self-organised architecture, in spite of overcoming the pressing issues of the centralised architecture, namely the lack of privacy control and single point of failure issues. SDOSN fills the research gap between the traditional P2P networks and the decentralised architecture of OSN, which can alleviate the privacy issue by giving users full control of their personal data, as well as providing adequate flexibility and capability using a self-organised architecture.

This self-organised architecture has been formally defined and characterised for decentralised OSNs with strong adaptability and high scalability without the need for costly central infrastructure. A social overlay network has been built on top of an instructed P2P network and users are allowed self-organise into semantic communities based on content similarity. The self-organised architecture exhibits the small-world characteristics with short average path length and large average clustering coefficient. Additionally, this chapter also defined and characterised the structure formation of the social overlay network and further proposed mechanisms for distributed topology adaptation to bootstrap newly joining nodes and to maintain the network structure. A node fingerprint hash technique has been presented to establish new connections considering the content similarity between users. Furthermore, the service content and user local knowledge including NSI, LKI have been defined with structured semantic abstractions. Five types of query messages have been formally defined to support communication and service discovery tasks. In the experiments, a configurable simulation platform has been presented and the main components include the adaptation of social overlay networks, decentralised service discovery and evaluation of search models. This platform can simulate a dynamic social network with knowledge-based routing protocols for service discovery. The correctness and efficiency of the proposed SDOSN architecture have been evaluated through extensive simulations. In the experiments, three different network structures including Random Network, Scale-Free Network and NFH-based Network have been compared and discussed. NFH-based network exhibited the small-world characteristics such as the large clustering coefficient and the short average path length. Moreover, the success rate, resource cost and query efficiency for informed approaches have been evaluated. The experiment results demonstrated that the combination of NSI and LKI converges fast in short TTL and delivers stable service discovery in the decentralised environment.

4 HOMOPHILY-BASED SOCIAL SERVICE DISCOVERY

4.1 Overview

Previous chapter introduced the semantical design of a self-organised architecture SDOSN. Service sharing and discovery are the core functionality of in SDOSN. Users in SDOSN exercise full autonomy over their shared services and social connections, and they interact with neighbours to locate the required services. However, with the scale of increasing number of social network users, the rate of possible interactions increases exponentially, hence large delay in service discovery. In addition, existing search methods in unstructured P2P often have low performance or generate high network traffic. Furthermore, the heterogeneous nature of social service data, dynamisms with nodes joining and leaving introduce more challenges for service discovery. Efficient service discovery without global knowledge of social network is a challenging task in large-scale decentralised environments.

This chapter presents an efficient service discovery approach based on homophily features that consider social relationships and user interests, and focuses on the neighbour selection for efficient query dissemination. To be more specific, firstly a semantic matchmaking method based soft-cosine similarity is proposed to capture the hidden semantic correlation between concepts and support the accurate match for complex queries. Then user's homophily features are characterised by the combination of status homophily and value homophily to capture the closeness between two users. The status homophily reveals the common social connections and the value homophily represents the common interests. Finally, during semantic query routing, a choice probability method is proposed to select the promising neighbours that have a high degree to disseminate the query and have a high probability to serve as friend candidates of the query initiator. As a result, the social network structure is self-adaptive according to the *choice probability* during service discovery process. Upon successful discovery, not only the requested services can be found, but also new connections between similar users may be established.

The proposed service discovery approach fits well with the real-world social networks, where the users expand their network during social interactions. With more social connections established during service discovery, users will have more social knowledge to resolve the query. The evaluation, performed in simulation using real-world datasets, shows that the homophily-based service discovery approach achieves better performance when compared with the state-of-the-art methods in a dynamic network environment.

In this thesis, service discovery specifically means finding a set of services published by their owners that satisfy a given set of requirements. It should be noted that the service is a general concept that is not restricted to well-defined web services, but many types of data including text, hashtags, friends, pictures or video clips in social networking systems. The service discovery should be considered a capability to locate varied, multi-source social resources.

4.2 Preliminary

Following assumptions are made in the proposed service discovery approach:

Network Topology: Participants $\{P_1, P_2, ..., P_m\}$ are self-organised in an unstructured P2P network with an average degree of connection γ. The social network is an undirected graph and only has one connected component, which means there is at least one path between any participants.

Social Communication: Each participant in the social network is able to communicate with its first-degree neighbours. Participants are considered as honest users who will respond the service request queries and help resolve the queries.

Social knowledge: Each participant maintains the social connections with their direct neighbours. The NFH, friends list and interests' information of their neighbours are obtained and stored in the local knowledge index. Note that NFH and friends list is considered as known information between direct neighbours by the default privacy setting. The common friends between any two users in the social network are assumed as known information. The interests are considered as user-declared public information that can be seen by all participants in the social network.

Service: this research assumes the user shared services are text documents. Each service document is represented as a weight vector $\vec{S} = (w_{s,1}, w_{s,2}, w_{s,3} \ldots w_{s,N})$ using the adapted term-frequency (TF) scheme.

Query: A query message contains a conjunction of *n* concepts. The query is represented as weighted vector $\vec{Q} = (w_{q,1}, w_{q,2}, w_{q,3} \ldots w_{q,N})$ using the TF scheme.

Similarity Calculation: the similarity between a query and a service is measured by the cosine of the angle between the vectors.

Semantic Group: The measurement of similarity between a query *Q* and node content $LSI(u)$ is using the soft-cosine similarity $soft\cos(Q, LSI(u))$ that returns a relevancy score between 0 and 1. The higher value of score, the higher relevancy it is. If the similarity $soft\cos(q, LSI(u)) > \theta$, the query is considered as the same semantic group with the content of node u. Otherwise, the query is considered beyond the semantic group.

Service Discovery: The service discovery task is to find as many relevant services that satisfy the query with a time-to-live (TTL) bound. A query can search for all available nodes and their shared services with a sufficiently large TTL through the collaboration of participants.

Service Provision: given a URI, a remote service can be requested directly from other participants to the service provider.

4.3 Problem Definition

The service discovery in centralised systems is the activity of obtaining information resources that are relevant to a user requirement from a resource repository. Full-text or other content-based indexing techniques are used to support efficiently searching in large datasets. A service discovery process begins when a user enters a query into the system. A query consists of formal statements of information needs, which is parsed into a conjunction of terms. The query identifies several objects that match terms in the query, and then objects is ranked by degrees of relevancy. The ranked list of objects is returned and shown to the user. Figure 4.1 shows the service discovery process in centralised systems.

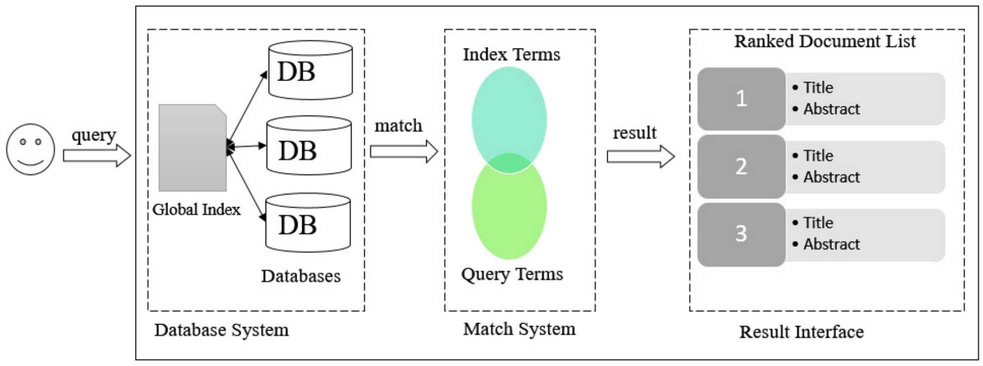


Figure 4.1 Centralised service discovery

The problem of service discovery in decentralised OSNs is known as *Semantic Query Routing Problem* (SQRP). The general goal of SQRP is to determine the shortest path from a user that issues a query to users that can appropriately resolve the query by providing the requested service. Furthermore, the number of services located is maximised and the number of steps to find the services is minimised. The query traverses the network moving from the initiating user to neighbouring users and then to neighbours of neighbours and so forth until it locates the requested service or exceeds a user defined limit.

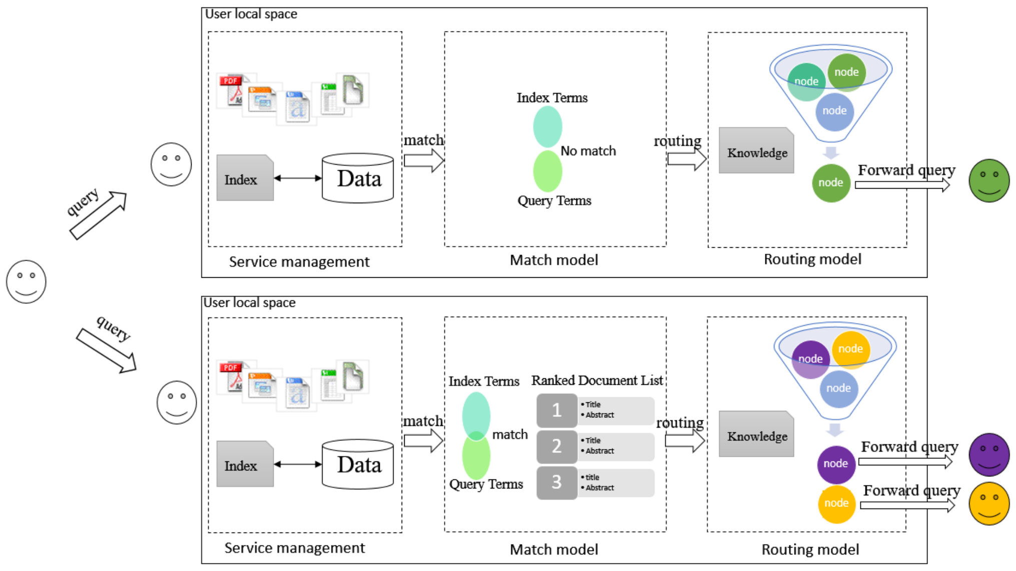


Figure 4.2 Decentralised service discovery

Similarly, a service discovery process begins when a user issues a query into the system. A query first searches a user's local space by matching the indexed data including text documents, description of images, audio and videos etc. If the query needs to be further forwarded, the user sends the query to a set of selected neighbours. The discovery process then is iterated in the neighbours' local space until sufficient relevant services have been found or TTL of the query meets the upper bound. Figure 4.2 depicts the service discovery process in the fully decentralised systems.

4.4 Discovery Framework

4.4.1 Knowledge-based Service Discovery

In social networks, people remember and update potentially useful knowledge from social interactions; as a consequence, random and diffuse behaviours gradually become highly organised [199].

In SDOSN, service discovery is also based on knowledge; a user only knows the information about their direct neighbours that contains the content hash NFH, friends list and declared interests. The service discovery is intended to locate the required service based on the knowledge and historical interactions among users. The historical information is the successful query history for a period in the past, which stores the direct shortcuts information URI to search a remote node without flooding the network. The historical information is attached with a time stamp of when the data being obtained and updated with the LRU scheme. With careful maintenance of the knowledge index, the service discovery can be guided with more accuracy and efficiency than blinded search, hence reducing the overhead of network traffic.

Figure 4.3 depicts the system structural frame of service discovery in the self-organised social network. The infrastructure is based on the unstructured P2P network. On the overlay network, users connect to others to build the F2F social network and the communication among users is enabled if users know each other, i.e. they are connected by the social network. Service discovery is a query-oriented information lookup in the social network. Users interact with each through query and response messages, and the historical information is cached in user's local space. Each user maintains two historical information about the social network including a network shortcut index (NSI) and a local knowledge index (LKI). The NSI contains a fix-sized inverted index of topic-URI pairs for successfully found services. The shortcuts can provide one-hop query routing for a recorded service location in remote nodes. The LKI stores information about immediate neighbours including their NFHs, friends list, declared interests and a pheromone table. When a new user joins the social network, the social connections and the knowledge index can be built rapidly through the virgin session. The social connections and the knowledge index are self-organised during social interactions in the network.

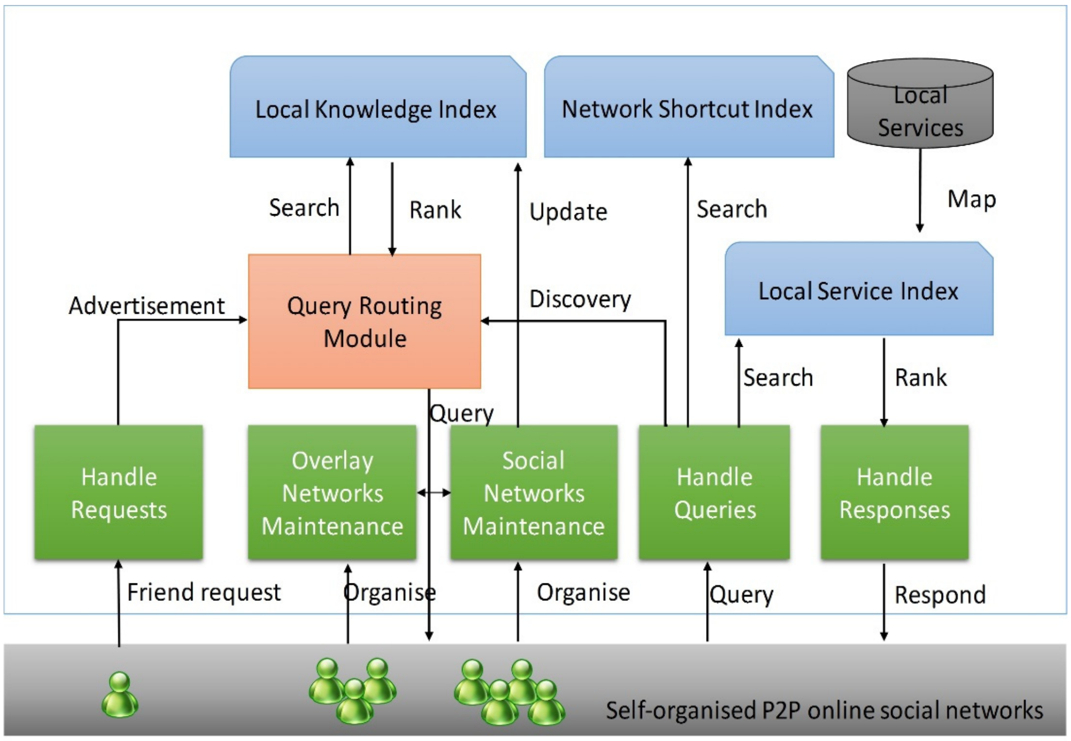
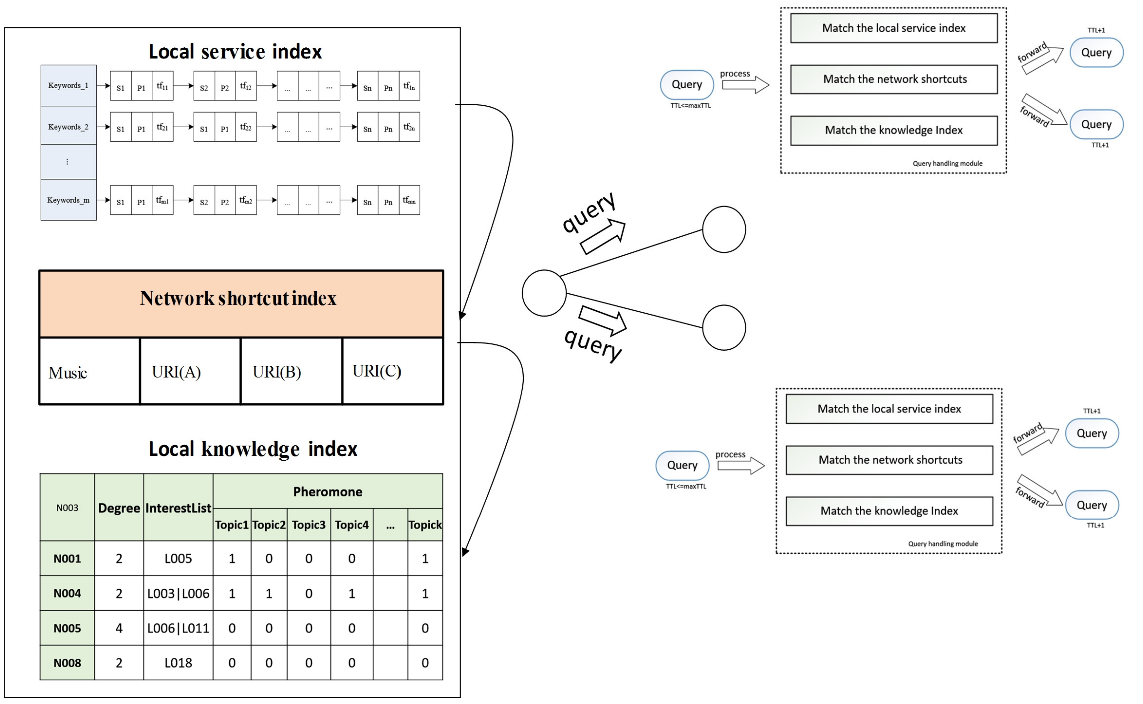


Figure 4.3 Frame diagram of service discovery in the self-organised social network

4.4.2 Query Processing Module

Within an established social network, users communicate with others through query messages that consist of service requirements. The query terminates when sufficient services have been found or the TTL exceeds the predefined upper bound. The query processing inside nodes consists of three modules such as local service match, network shortcuts match and local knowledge index match.

As shown in Figure 4.4, when a user receives a query message, the user first matches the query to the local service index to calculate the similarity between the query and local services. In order to fetch more services in the social network, the query then is processed by comparing the network shortcuts, which can provide direct access with the previous visited service in a remote node. If the query needs to be forwarded, the local knowledge index is utilised to select promising neighbours based on the homophily features and the pheromone information. When the query message is forwarded to a set of selective neighbours, the query will be iteratively processed through the query handling module.



N003	Degree	InterestList	Pheromone					
			Topic1	Topic2	Topic3	Topic4	...	Topick
N001	2	L005	1	0	0	0		1
N004	2	L003|L006	1	1	0	1		1
N005	4	L006|L011	0	0	0	0		0
N008	2	L018	0	0	0	0		0

Figure 4.4 Query handling module inside nodes

When a user receives a query, the query will be handled by the local query processing module which consists of local service index match, network shortcuts match and knowledge index match. The previous section introduced the service matchmaking method to effectively find relevant services that meet the system-defined similarity threshold. Then the query is processed through matching network shortcuts, the match process utilises the feature correlation matrix to calculate the similarity between query terms and the topics in topic-URI pairs. The matched shortcuts direct the query to the service providers, and the available service providers respond the URI service request by sending report query message to the current query handling node. If no service is found in shortcuts by URI, the query then will be forwarded to a selected set of suitable neighbours using LKI. This research utilises user's homophily features to discover the personalised information and potential social connections, which self-organises similar users into communities. Hence, with the evolution of social networks, the discovery path tends to be short and the unnecessary traffic is also reduced during service discovery.

During query routing, a book-keeping technique is adopted to avoid redundant query messages. Each query message is assigned with a globally unique identifier GUID by the query initiator. In the local cache, each node keeps track of received query GUIDs. If a node receives a query with the same GUID that has been recorded already, the node will discard the query. This technique reduces the probability that a query traverses to the same node to sidestep routing circles.

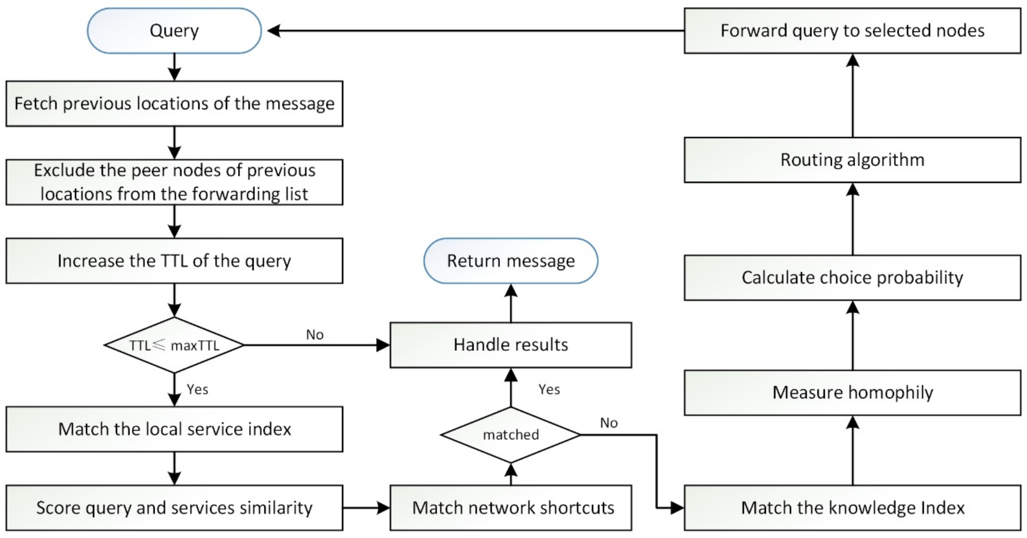

Query
Forward query to selected nodes
Fetch previous locations of the message
Routing algorithm
Exclude the peer nodes of previous locations from the forwarding list
Increase the TTL of the query
Return message
Calculate choice probability
No
TTL ≤ maxTTL
Handle results
Yes
Yes
Measure homophily
Match the local service index
matched
No
Score query and services similarity
Match network shortcuts
Match the knowledge Index

Figure 4.5 Flow chart of query routing process

It is necessary to define at which point a GQ message for service discovery should be terminated. In decentralised systems, the services are scattered across the networked nodes and the behaviour of search relies on query routing via neighbour. The search task is to find some unknown nodes that may provide the requested service and the service availability is not guaranteed. Therefore, a stopping point must be defined to prevent query messages from running infinitely in the network. This research uses TTL parameter to define the hops of query messages. One hop means the query travels from one node to another node. After each hop, the TTL in the query increases by one. The *maxTTL* parameter is predefined to confine the maximum hops of a query message. When TTL reaches the *maxTTL*, the query will be terminated. The query forwarding process flow chart is shown in Figure 4.5. After a query has reached a target node that can provide the service, the query messages can terminate the travel or continue to find more target nodes until it reaches the *maxTTL*. In this research, the query messages are kept alive after first successful discovery and continue the discovery process until they reach the maximum time constraint i.e. *maxTTL*. It aims to maximise the number of results by searching more appropriate nodes. This strategy is beneficial for individual users since it can find as many services as possible that meets the user requirement in the decentralised social network.

4.4.3 Service Discovery

The service discovery comprises two stages: the advertisement stage and the search stage. The advertisement stage is the knowledge updating phase when each participant periodically issues an AQ message containing digest information NFH about services to be shared along with their social connections and interests. The digest information is pre-processed based on the locality-sensitive hash [200] techniques that are able to compress similar user shared services into similar hash values. The NFH is a hash-based, space-efficient data structure that significantly reduces the size of messages transferring in the network layer. Due to the dynamic nature of the social network, contents, connections and interests of participants are changing constantly. Therefore, the advertisement stage is critical for effective service discovery in such a dynamic environment. After this stage, participants can obtain the up-to-date information and cache the NHFs, connections and interests of their neighbours.

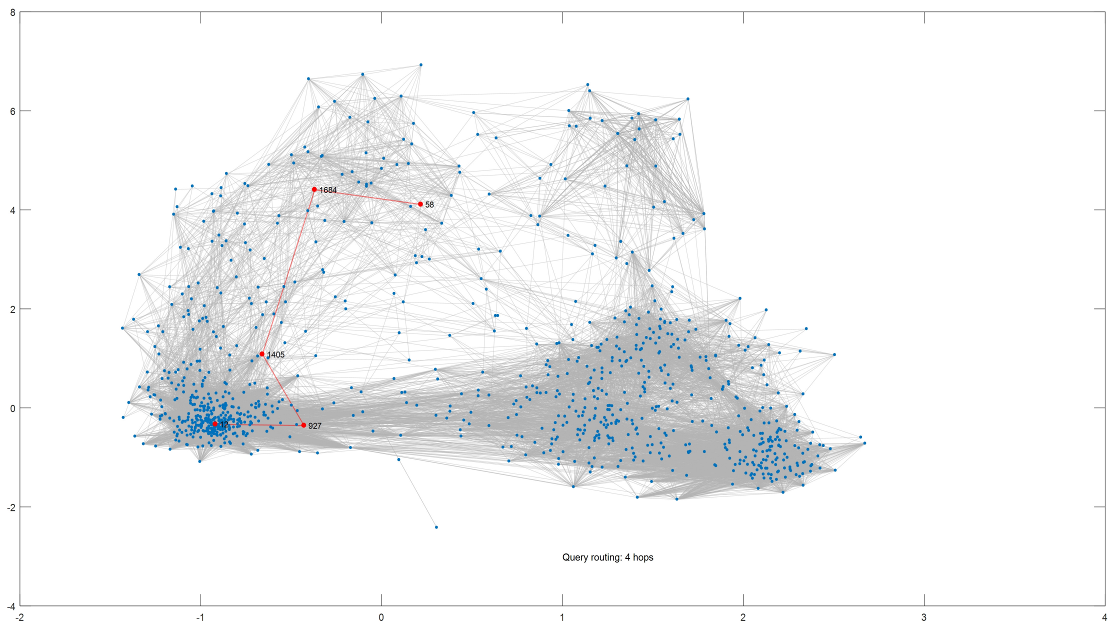


Figure 4.6 Service discovery: query path from node 12 to node 58 requires 4 hops

In the search stage, when a participant searches for a specific service, it deploys a GQ message to find participants holding the desired service. A service discovery example is shown in Figure 4.6. In this example, the identifier of query initiator is 12, and the requested service is possessed in node 58. The query travels four hops from node 12 to its neighbour 927, then to 927's neighbour 1405, next to 1405's neighbour 1684, and finally via 1684 to the service provider 58. The query routing is in a hop-by-hop fashion and query messages are forwarded to selective intermediate neighbours based on their semantic similarity to a query. To be more specific, if the semantic similarity is above a predefined threshold, the query is forwarded to the neighbours with semantic ties; otherwise, the query is forwarded via correlative ties. The semantic ties and correlative ties are pre-calculated during advertisement stage. If the target node is located, a RQ message will be returned to the requestor and pheromone information is laid along the routing path. In the future, if other query messages are seeking similar services, the query messages can follow previously query hit information and not traverse the network in a blind way.

4.5 Semantic Matchmaking

The service discovery is a query-oriented search process that defines the searching and matching logics in SDOSN. It allows users to issue query messages to obtain information of neighbours and locate services from remote peers in the social network. The effectiveness of service discovery significantly depends on the similarity measurements of relevance between queries and services in node content. The aim of semantic matchmaking is to rapidly determine whether a node can resolve the query or not.

The searching process inside a node is based on two assumptions: firstly, a query message with a conjunction of terms can search all shared services through LSI in a user's local space; secondly, a service is considered as a successful match to a query when the relevance score is above a relevance threshold θ. Otherwise, they are deemed to be irrelevant.

4.5.1 Motivation

The traditional VSM approach adopts cosine similarity [201] to capture the similarity between query messages and services inside nodes. The cosine similarity is widely used to calculate the distance between two document-vectors in the information retrieval fields [89, 90]. However, the cosine similarity adopted by VSM suffers some limitations: 1) it neglects the terms correlations. VSM considers terms in a document vector as independent features or completely different factors. As a result, this term-independent assumption neglects important correlations between terms and cannot accurately measure the similarity between two documents. For example, suppose there are two keywords *"play"* and *"game"*. In VSM, the two different words are mapped to different dimensions; the similarity between them is 0. However, it is obvious that *"play"* and *"game"* are related semantically. If the correlation between terms is taken into consideration, the cosine similarity can be measured more accurately. 2) it does not cope with scale problems. The similarity score is overly biased by the terms with higher weight and gives less consideration on the number of common terms. For instance, consider three vectors with example values $\vec{a} = (1,0,10,0,2)$, $\vec{b} = (0,0,10,0,0)$ and $\vec{c} = (1,0,1,0,1)$. The cosine similarities for $\vec{a}$ and $\vec{b}$, and $\vec{a}$ and $\vec{c}$ are calculated as $\cos(\vec{a},\vec{b}) = 0.975$ $\cos(\vec{a},\vec{c}) = 0.734$. From observation, $\vec{a}$ is more similar to $\vec{c}$ than it is to $\vec{b}$ since there are 3 positive elements matching in the same dimensions between $\vec{a}$ and $\vec{c}$ but only 1 matching for $\vec{a}$ and $\vec{b}$. However, the cosine score does not reflect the intuitive observation. Even through $\vec{a}$ and $\vec{c}$ share more common words, the higher weight in the third dimension of $\vec{b}$ has gained biased score and significantly reduces the impact of low weight values in other dimensions.

4.5.2 Semantic Matchmaking

In order to capture the accurate semantic match between a query and a LSI, the soft-cosine similarity is adopted to calculate the similarity considering the underlying correlation between terms in VSM. The soft-cosine similarity integrates the generalized concepts of cosine measure and semantic correlations of the terms [202]. Here the formal definition is given as:

DEFINITION 4.1: *the feature correlation matrix* $FC = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{pmatrix}$*, where* $c_{ij} = sim(t_i, t_j)$*,* t_i *and* t_j *are the term features in a finite document repository, and* $sim(t_i, t_j)$ *is a similarity measure to quantify the semantic similarity between terms* t_i *and* t_j.

There are numerous ways of computing the similarity between terms, such as widely used synonym wordnet [203], word2vec [204], WS4J [205]. With correlations of terms becoming available, our goal is to be able to calculate the similarity between two vectors considering the feature correlations.

DEFINITION 4.2: *the service and the query are defined as vector $\vec{S}$ and $\vec{Q}$, respectively.*

$\vec{S} = (w_{s,1}, w_{s,2}, w_{s,3} \dots w_{s,N})$ *is a N-dimensional vector of LSI, where* $w_{s,i} (1 \le i \le N)$ *is the weight the for i-th term feature.*

$\vec{Q} = (w_{q,1}, w_{q,2}, w_{q,3} \dots w_{q,N})$ *is a N-dimensional vector of query, where* $w_{s,i} (1 \le i \le N)$ *is the weight the for i-th term feature.*

The weight function is defined as:

$$w_{x,t} = 1 + \log f_{x,t} \tag{4.1}$$

where the $f_{x,t}$ is the term t's frequency in a document that can be a service or a query.

The weigh is assigned using an adapted *term-frequency* (TF) scheme. This weight function does not require global information and can be calculated locally inside nodes.

The soft-cosine similarity between $\vec{S}$ and $\vec{Q}$ is given as:

$$soft\cos(\vec{S},\vec{Q} \mid FC) = \vec{S} \times FC \times \vec{Q}^T = \frac{\sum\sum_{i,j}^{N} a_i c_{ij} b_j}{\sqrt{\sum\sum_{i,j}^{N} a_i c_{ij} a_j}\sqrt{\sum\sum_{i,j}^{N} b_i c_{ij} b_j}} \tag{4.2}$$

where FC is the feature correlation matrix in which $c_{ij} = sim(s_i, q_j)$ and $c_{ii} = 1,\ c_{ij} = c_{ji}$.

If there is no similarity between two features, then the feature elements in FC is zero. The formula computes the soft-cosine for the query and service in time complexity of $O(N^2)$.

The soft-cosine similarity normalises document length during the comparison. The similarity value between a query and document is in a closed range between 0 and 1, in which the resulting value with 1 indicating exactly match, with 0 indicating no correlation, and in-between values indicating intermediate similarity level.

In order to illustrate the soft-cosine similarity, an example are given as follows:

For instance, suppose there are total 4-dimension of term features in the vector model including the elements in $\left(download,\ itunes, jazz, music\right)$. A service vector $\vec{S}$ contains three term features $\left(download, itunes, jazz\right)$; by using weight function, the service vector is transformed to $\vec{S} = (1,1,1,0)$. A query message has one term feature $\left(music\right)$, and the query vector is represented as $\vec{Q} = (0,0,0,1)$.

$FC = \begin{pmatrix} 1 & 0.8 & 0.1 & 0.2 \\ 0.8 & 1 & 0.4 & 0.7 \\ 0.1 & 0.4 & 1 & 0.6 \\ 0.2 & 0.7 & 0.6 & 1 \end{pmatrix}$ is the feature correlation matrix that provides the terms similarity between each pair of in the set:

$\{\left(download, itunes\right), (download, jazz), (download, music), (itunes, jazz), (itunes, music), (jazz, music)\}$

Using the traditional cosine similarity:

$$\text{cosine}(\vec{\text{S}},\vec{\text{Q}}) = \frac{\vec{S}\cdot\vec{Q}}{\|\vec{S}\|_2\|\vec{Q}\|_2} = \frac{\sum_{i=1}^{n} S_i Q_i}{\sqrt{\sum_{i=1}^{n} S_i^2}\sqrt{\sum_{i=1}^{n} Q_i^2}}$$

$$= \frac{1\times 0 + 1\times 0 + 1\times 0 + 0\times 1}{\sqrt{1^2+1^2+1^2+0^2}\sqrt{0^2+0^2+0^2+1^2}} = \frac{0}{\sqrt{3}} = 0.0$$

Since the music is in the different dimension with other terms, the similarity calculated by cosine similarity is 0; yet the query: $\left(music\right)$ is very much similar to the service $\left(download, itunes, jazz\right)$ as they all related to a semantic group. Due to the lack of considering underlying semantic correlations between terms, the traditional cosine similarity only has a limited capability for query-service matchmaking. As a result, many services with high relevance to queries are not retrieved, thus low search performance during service discovery process.

The soft-cosine similarity that can capture the correlation between term features with additional relatedness knowledge is calculated as:

$$soft\cos(\vec{S},\vec{Q}\mid FC)=\vec{S}\times FC\times\vec{Q}^{T}=\frac{\sum\sum_{i,j}^{N}a_i c_{ij} b_j}{\sqrt{\sum\sum_{i,j}^{N}a_i c_{ij} a_j}\sqrt{\sum\sum_{i,j}^{N}b_i c_{ij} b_j}}$$

$$=\frac{1.9\times 0+2.2\times 0+1.5\times 0+1.5\times 1}{\sqrt{1.9\times 1+2.2\times 1+1.5\times 1+1.5\times 0}\sqrt{0.2\times 0+0.7\times 0+0.6\times 0+1\times 1}}=\frac{1.5}{\sqrt{5.6}}=0.633$$

From the above result, it can be seen that the correlation between term features can bring a difference in the similarity measurement. The soft-cosine similarity achieves more accurate semantic match and has advantages in dealing with the inexplicit query that often has low precisions and recalls in the service discovery.

4.6 Homophily-based Service Discovery

In the discovery stage, nodes utilise query messages to search the network in a decentralised manner. Since services are scattered in the social network and lack hierarchical organisation or centralised control, most queries travel a long time to retrieve relevant services. Moreover, the traditional service discovery methods in P2P networks are mainly supported by keyword-based solutions that cannot provide a context-aware and personalised search for end users. As a result, many services that are irrelevant to a specific user's request may be considered during the service discovery process. These unrelated services do not provide the request information and cost bandwidth.

In order to deal with these problems, a homophily-based approach is proposed to exploit service content and user's social interest. The main objective is to discover those nodes that have a high probability of resolving the query messages and have potential to serve as friendship candidates. The homophily-based approach can self-organise nodes in the social network into small semantic communities that can hand frequent queries. With the social network evolving, service discovery in the homophily-based social network can have high efficiency and the query path become short.

4.6.1 Homophily Formation

When a user is being queried, SDOSN not only considers the similarity based on queries and services, but also takes user's common features into account. Furthermore, according to recent studies [101, 117, 206], 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 [114].

Homophily is one of the most salient properties in complex networks [207-209]. The idea behind homophily is that individuals tend to interact and establish links with similar individuals. In their original formulation of homophily, Lazarsfeld and Merton [101] distinguished the concept between status homophily and value homophily. This subsection introduces the formal definition of status homophily and value homophily to approach the similarity of users.

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

$$H(u,v) = \delta SH(u,v) + (1-\delta)VH(u,v) \tag{4.3}$$

Where δ is a parameter that regulates the importance of the influence of status homophily and value homophily in the overall homophily of two users. $H(u,v)$ is a measurement of interpersonal similarity that influences the service matchmaking and forwarding degree in the service discovery process. Users are more likely to successfully provide the services or guide the discovery when they are similar to each other [201].

The status homophily defines the social familiarity, which calculates the common friends (neighbours) 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 $SH(u,v)$. The computation utilises the Jaccard index [210] to compare the similarity of two sets. $SH(u,v)$ is defined as the size of the neighbourhood intersection divided by the size of the neighbourhood union:

$$SH(u,v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|} \tag{4.4}$$

where $N(u)$, $N(v)$ are the neighbour set of user *u* and *v,* respectively. Given a set $N(u)$, the cardinality of $N(u)$ denoted $|N(u)|$ counts how many elements are in $N(u)$. The intersection between $N(u)$ and $N(v)$ is denoted $N(u) \cap N(v)$ and reveals the elements in both sets. The union between $N(u)$ and $N(v)$ is denoted $N(u) \cup N(v)$ and reveals the elements in either set. If the $N(u)$ and $N(v)$ are both empty, then define $SH(u,v) = 0$.

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. Therefore, 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:

$$VH(u,v) = \frac{|T(u) \cap T(v)|}{|T(u) \cup T(v)|} \tag{4.5}$$

where $T(u)$ and $T(v)$ are the interest set of user *u* and *v*, respectively. If $T(u)$ and $T(v)$ are both empty, define $VH(u,v) = 0$.

Equation (4.4) and (4.5) apply the Jaccard index metric in the neighbour set and the interest set to achieve fast calculation of homophily score. Consider set $N(u) = \{0,1,5\}$ and $N(v) = \{0,5,6\}$ where the entries in *N(u)* and *N(v)* are the identifiers of nodes, set $T(u) = \{sports, arts\}$ and $T(v) = \{photography, arts, religion\}$ where the entries in *T(u)* and *T(v)* are terms of social interest for *u* and *v*, respectively. For simplicity of this example, the identifier of nodes are represented as numbers, and the interest's terms are mapped into a flat way without the semantic hierarchy.

The status homophily is calculated: $SH(u,v) = \frac{|N(u) \cap N(v)|}{|N(u) \cup N(v)|} = \frac{|\{0,5\}|}{|\{0,1,5,6\}|} = 0.5$

The value homophily is calculated:

$$VH(u,v) = \frac{|T(u) \cap T(v)|}{|T(u) \cup T(v)|} = \frac{|\{arts\}|}{|\{\text{sports,arts,photograpy,religion}\})|} = 0.25$$

If $\delta = 0.5$, then $H(u,v) = \delta SH(u,v) + (1-\delta) VH(u,v) = 0.5 \times 0.5 + 0.5 \times 0.25 = 0.375$

Hence, the neighbours and interest can capture the homophily feature between two users. The users who have high overall homophily score are considered strongly share common social characteristics, which facilitate new relationship formations thereby making communication more efficient in the decentralised social network.

4.6.2 Homophily-based Query Processing

4.6.2.1 Query Modelling

This research proposes a query routing method by modelling a GQ message as a virtual node with potential services that contains the query requests and homophily features of the query initiator. This method is beneficial for the service discovery for two reasons. Firstly, the virtual node not only has the information of service requirement, but also the information about the query initiator, which enriches the expression ability of a query, make the matchmaking process more accurate and reduces mismatch of the vague terms that has multiple meanings. Secondly, the virtual node integrates homophily to discover personalised information that increases the frequency of interaction with socially similar nodes during neighbour selection.

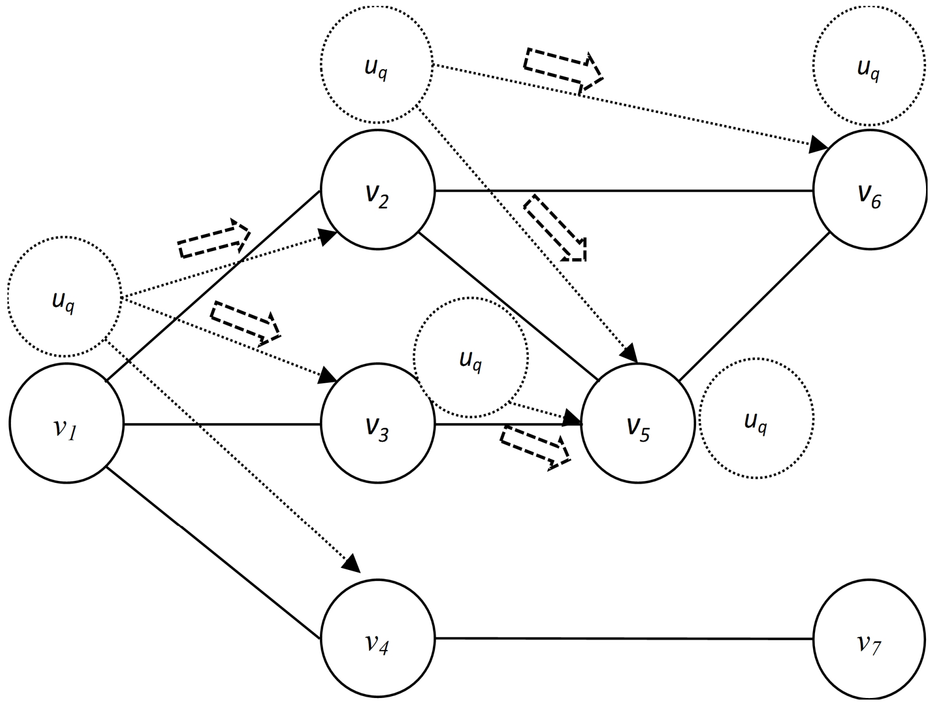


Figure 4.7 Homophily-based query processing

As shown in Figure 4.7, a GQ message q issued by node u is modelled as a dotted node of u_q. u_q, a temporary shadow of u, contains the information on the profile of node u including interests, neighbour list as well as the service requirements in the content. In the personalised service discovery, the virtual 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 [115]. The lifecycle of u_q starts from the query being issued and ends with the *maxTTL* expiry. The u_q is intended to discover those nodes that have potential to serve as friendship candidates and have a high probability of resolving the query messages.

4.6.2.2 Homophily-based Matchmaking

The matchmaking process is tailored specifically to an individual's social feature by incorporating information about the individual beyond specific query provided, which is shown in Figure 4.8. 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 neighbour 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 user's profile vectors. As a result, heterogeneous information, such as queries and service contents, as well as user interests and neighbours, 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 neighbours, 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 soft-cosine similarity to calculate the overall similarity between the query vector and user vector. A threshold of cosine similarity that determines a user can provide a relevant service to the query requester is set to 0.25 in the system. This threshold is evaluated in the simulation section.

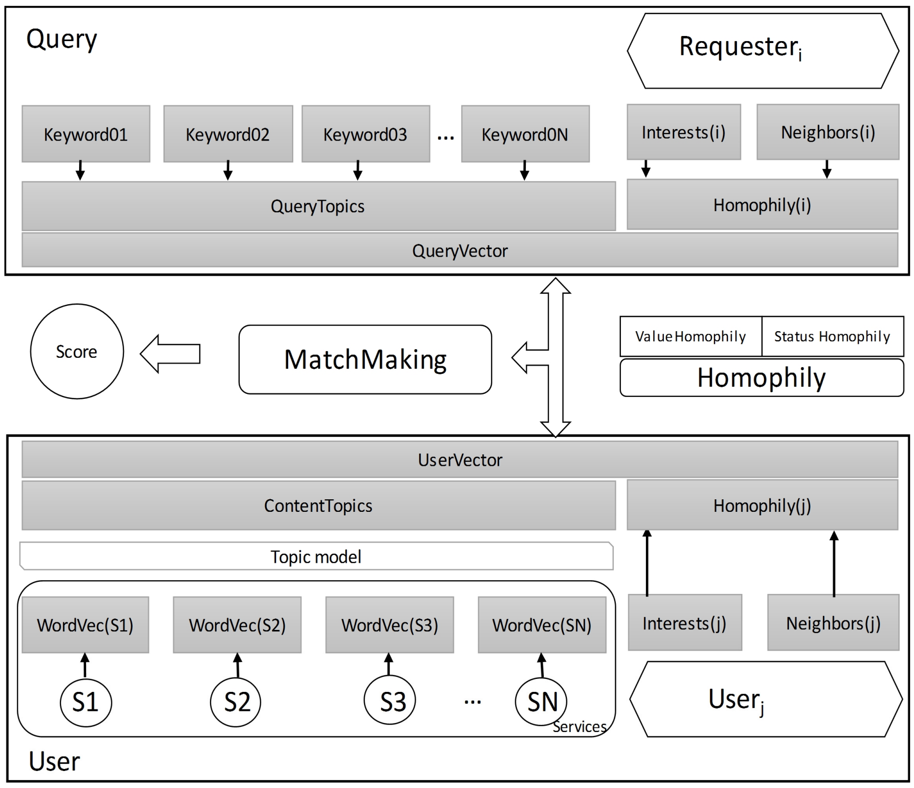


Figure 4.8 Homophily-based matchmaking module

4.6.2.3 Choice Probability

When node v_y receives a query u_q, v_y starts the local query processing module. It utilises the soft-cosine approach to calculate the semantic distance between the query topics and LSI. If the similarity score is below the predefined threshold θ, the query then will be forwarded to the neighbours. This research utilises homophily-based factor and degree-based factor (number of neighbours) in an exponential function to measure the probability of a neighbouring node to be capable of resolving the query. This probability will be used to determine whether to forward a query to a neighbour and so the probability is called choice probability. The most promising neighbour is the most similar neighbour to the virtual node and has the highest number of connections. Therefore, when node v_y forwards query u_q to its neighbour set N_{v_y} the following equation calculates choice probability of

$\forall v_x \in N_{v_y}$

$$CP(v_x,u_q)=1-\left(1-\left(\frac{H(v_x,u_q)}{\sum_{v\in N_{v_y}} H(v,u_q)}\right)\right)^{|N_{v_x}|} \tag{4.6}$$

The choice probability *CP* considers the homophily between the node v_x and the virtual node u_q. Moreover, it also considers the degree of the neighbours. 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 make more impact on the network traffic. The exponential function used in Equation (4.6) 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 neighbouring node and the virtual 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.

An example calculation of choice probability is presented in Figure 4.7. Before forwarding the query to the neighbours, 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, the choice probability is calculated by using homophily score and connection degree. 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, since the connection degree plays a more important role in Equation (4.6).

4.6.2.4 Query Routing Strategy

The query routing aims at finding the promising neighbours that a have high probability of resolving the query or know who can resolve the query. This research considers the semantic information of neighbourhood using node fingerprint NFH that captures the content similarity between nodes. Based on the similarity, the neighbours are logically classified as two kinds of social ties: the semantic social ties and correlative social ties. The semantic query routing consists of two strategies: the exploitation strategy and the exploration strategy. The functionality of the two strategies are described as following:

- The exploitation strategy utilises the current gained results to search more relevant result in the neighbourhood. It selects the neighbours with high *choice probability* (CP) in the semantic ties.
- The exploration strategy provides mutability and randomness to discover new semantic groups. It is achieved by routing queries to dissimilar neighbours via correlative ties.

The two strategies supplement each by utilising the two kinds of social ties in a self-adaptive manner. Given a query, SDOSN first utilises exploitation strategy to locate a node within the relevant semantic group through correlative ties, and then multicast the query to the semantic ties that are likely to be relevant to the query too. By iteratively applying the two strategies in the query routing, the query can be navigated to the service provider with less traffic using small forwarding degree on correlative ties, and it is able to discover more services in the neighbourhood of a service provider.

4.6.3 Social Connection Adaptation

In most online social networking systems, users can meet new users and establish a friend connection during social interactions. In order to facilitate the growth of social networks, most systems provide recommendations of new friends who have common friends or common interests. In this way, users can establish new friend connections and expand their social network. Hence, this research proposes mechanisms for users to establish new friendship during service discovery.

The task of neighbour adaptation in this research is to select new neighbours (i.e. friends) from candidates during service discovery and to keep track of the LKI of the existing neighbours. The selected new neighbours will be connected through friendship establishment protocol. Homophily is a measurement of social closeness and can enhance the discovery process to select the service providers that share common social features with the query initiator. And the choice probability enhances the discovery towards the nodes with a high degree of connection.

Algorithm 4.1 Social Connection Adaptation

Input: Node u, a finite query set: $Q=\{q_1,q_2,\ldots q_c\}$, Choice probability threshold φ

Output: Friend connection update

```
1.  //generate query messages for service discovery
2.  for $i \leftarrow 1$ to $c$ do
3.      while $GenerateQuery().Next() \neq \varnothing$ do
4.          $q_i \leftarrow Q.Next()$
5.          $q_i.ttl \leftarrow q_i.setTTL()$
6.          $q_i.\mathrm{homophily} \leftarrow u.getHomophily()$
7.          $QueryRouting(q_i)$
8.          $Hits\,QueryHits(q_i)$
9.      end while
10. end for
11. //generate candidates from success discovery
12. while $v \leftarrow Hits.next() \neq \varnothing$ do
13.     if $v.CP \geq \varphi$ do
14.         $candidateList.add(v)$
15.         //Friendship establishment protocol
16.         $FEP(u,v)$
17.         //Incremental update of local knowledge index
18.         $updateLKI(u)$
19.     end if
20. end while
21. //u sends AQ message to inform its neighbours
22. $sendNotification(N(u))$
23. //Neighbours update the changes
24. $updateLKI(N(u))$
25. return $u.connections()$
```

Following a successful discovery, these nodes are cached in the network shortcut index for future search. Moreover, the successful discovery returns a friend candidate list of service providers that have resolved the query and also have higher choice probability than a predefined threshold φ. Each node periodically updates its neighbours by adding new neighbours from the friend candidate list. New friendships are formed by using FEP protocol if the current number of social connection is less than the maximum connection. After successful connection, neighbours are categorised into semantic ties or correlative ties based on the NFH similarity score. The information of new connected neighbours then advertised to the existing neighbours through AQ messages. The existing neighbours will be notified with the messages and updates the neighbour information in LKI accordingly. This strategy gradually establishes a homophily-based social network. It allows users to expand their social network with nodes that have common social features and high degrees of connection. The algorithm for social connection adaptation is shown in Algorithm 4.1.

The degree of connection is an important factor in query routing. In a single hop of routing process, those nodes with a higher degree have more influence on the information dissemination, since they can forward queries to more nodes. In overall routing process, with more high-degree nodes being selected in the routing process, queries can reach a larger portion of the network within a fixed number of hops. Since the degree factor can significantly impact on the efficiency of query routing, Equation (4.6) lays more emphasis on degree factor to generate a marginal improvement of ranking score for high connection nodes in every hop. The reason of using the degree as an exponential factor is that the larger degree nodes are more sociable and have a greater number of candidate neighbours to be explored in the next hop. As a result, the query has a higher probability of being resolved within the next hop. Therefore, placing more importance on the degree will accelerate the service discovery process.

4.7 Simulation and Results

Previous experiments have evaluated the correctness of the architecture design of SDOSN. The focus of this section is on the evaluation of the search performance of service discovery, and network overhead of semantic query routing based on this architecture. The experiments compare performance by the commonly used metrics for information retrieval (IR) in P2P networks.

4.7.1 Performance Metrics

Performance is evaluated with the following measures.

- ***Recall***: It is a commonly used information retrieval metric to measure the quantity of the search results. The definition of recall is the number of retrieved services divided by the number of relevant services in the whole repository. The equation to calculate the recall is given as:

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

- ***Precision***: The definition of precision is the number of retrieved relevant services divided by the number of the retrieved services. This metric is generally adopted to measure the quality of search results. The equation to calculate the precision is given as:

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

- ***F-1 measure***: It evaluates overall effects of precision and recall. This metric is defined as the harmonic mean of precision and recall and is given as:

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

- ***Success rate***: the ratio of the number of resolved query to the number of all issued query within a given TTL in the network. A query is deemed unsuccessful if the query initiator has not received responses at the end of the simulation.
- ***Resource cost***: this metric is used to quantify the number of nodes visited to resolve the query during the service discovery process.
- ***Query efficiency***: it is the ratio of success rate to resource cost. This metric evaluates the overall effectiveness of a search strategy. Ideally, an approach with highest success rate and lowest resource cost is the most efficient strategy.

4.7.2 Simulation Setup

The simulation setup used for the subsequent experiments is the same as section 3.4. The content generation and query generation are the same in section 3.4.2 and the same metrics used are in line with section 3.4.3. Since the simulation covers the semantic matchmaking, additionally, the benchmarking dataset and the feature correlation matrix are created for the calculation of precision and recall.

Benchmarking dataset: the Social-ODP-2k9 [196] includes the annotation data and document data. This dataset comprises *12,616* unique URLs and corresponding HTML documents. This dataset is separated into two parts: 1) the annotations of a document are considered as the relevancy judgement dataset. 500 annotations are randomly selected to construct the candidate query set. 2) the content of each document including title, keywords, and descriptions is extracted to construct a structured document-topic vector where each document vector has average 10 topics achieved by applying the topic model [33]. The document-topic vectors are used as service content data, which is distributed among the nodes based on the power law.

Table 4.1 Word similarity example

computer		**book**		**music**	
Word	Similarity	Word	Similarity	Word	Similarity
software	0.703264	novel	0.721062	musical	0.707323
mainframe	0.686567	essay	0.707682	jazz	0.686187
computing	0.679052	memoir	0.693306	dance	0.657294
microcomputer	0.660982	autobiography	0.656579	musics	0.615135
realtime	0.646826	graphic_novel	0.636255	hip_hop	0.606102
hardware	0.640497	novella	0.629417	recording	0.599797
minicomputer	0.636557	pamphlet	0.620085	songs	0.598778
microprocessor	0.634268	poem	0.617383	tunes	0.594010
workstation	0.627294	foreword	0.612422	recordings	0.587729

Feature correlation matrix: the term correlation is used to capture the semantic similarity distance between terms. A NLP tool, Gensim [211] is adopted in the calculation of the correlation matrix, which serves to generate a similarity between two given terms. The matrix is constructed by the following steps: 1) English Wikipedia dump (download data is 20 March 2017) is processed to plain text by removing stop words and punctuation. 2)

The Word2vec model provided by Gensim is used to train wiki text to vectors that provide the word similarity distance. There are more than 3 billion words in the wiki text. Table 4.1 shows three words computer, book and music and their top-9 similarity words. 3) Construct the feature correlation matrix for this simulation. Apparently, this research cannot test all the words in wiki text. Unique annotations and topics (in total 35700 terms) have been selected into the feature vocabulary. Then the similarity distance between any two terms are calculated and stored in the feature correlation matrix.

4.7.3 Performance Evaluation

4.7.3.1 Evaluation of Query-service Similarity Threshold

In service discovery, a query message can return many relevant services to the query initiator. Most of the current P2P search models only focus on recall metric that validates the capability of retrieving relevant results and the precision of the retrieved results is often neglected. Precision validates the accuracy and correctness of search models. Therefore, both recall and precision metrics are important in the service discovery process.

To determine the performance of search model, the retrieved services from SDOSN are compared with relevancy judgement in benchmarking dataset. The recall and precision are evaluated in the experiment. Ideally, an optimal search model should have the highest recall and highest precision. However, in most information retrieval systems, recall and precision are inversely proportional, which means a search model either can have high recall and relatively low precision, or have high precision and relatively low recall. In the experiment, the similarity threshold θ is evaluated, which is the parameter determining the relevancy of services to a query. If the similarity between a service and a query is above the threshold during matchmaking, the service is considered as relevant to the query and is retrieved to the query initiator. Otherwise, the services are disregarded during matchmaking.

In the scenario shown in Figure 4.9, the threshold θ is evaluated through a service discovery task based on soft-cosine similarity, which is to return a set of relevant enough services for a given query. Precision is the probability that retrieved services are relevant. The recall is the probability of retrieving a relevant service in a search. The benchmarking dataset provides the ground truth of relevance between a query and a service. It can be seen from Figure 4.9, the recall and precision are inversely proportional; the precision usually increases at the cost of reducing the recall whenever the cosine similarity threshold increases. It can be observed that the two curves intersect with each other for the defined threshold value of 0.25.

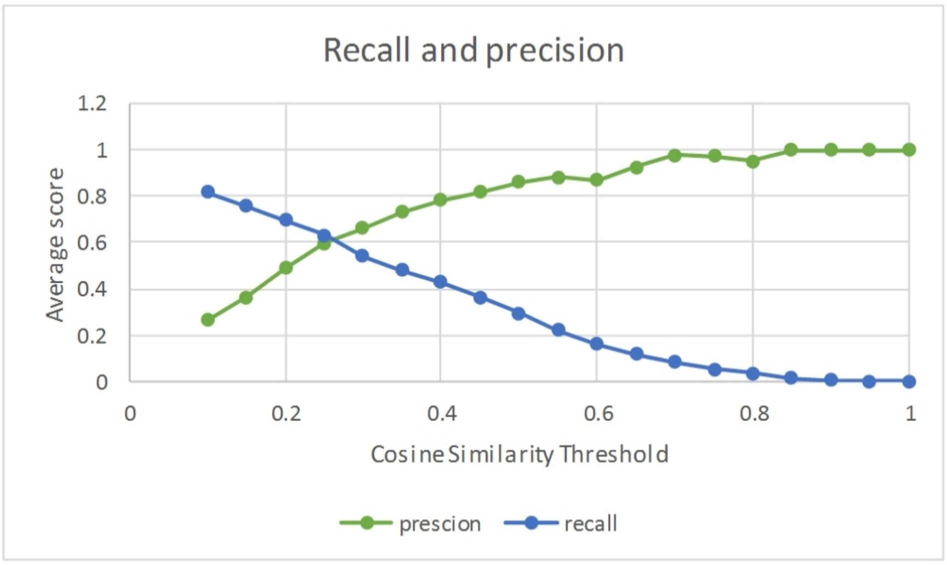


Figure 4.9 Evaluation of similarity threshold on recall and precision

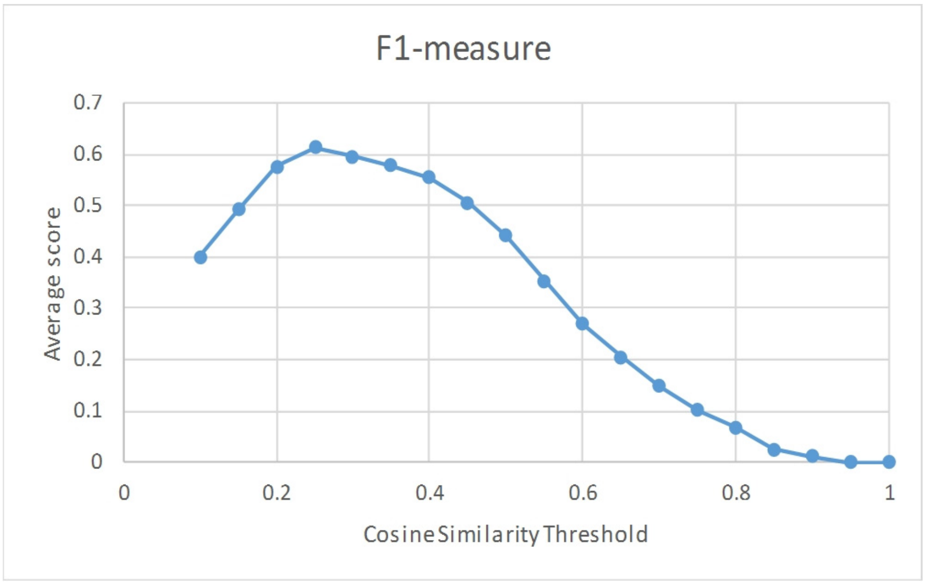


Figure 4.10 F1 measure on similarity threshold 0.25

Typically, the precision and the 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. Figure 4.10 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 can be observed that the F1-measure reaches its peak value when the cosine similarity threshold approaches 0.25. Based on the observation in Figure 4.9 and Figure 4.10, the parameter of similarity threshold is set as 0.25 in order to determine a relevant service.

4.7.3.2 Evaluation of Semantic Matchmaking on Complex Queries

The natural way of finding a specific service in the decentralised social network is to use keywords that specify various properties or attributes of the service. One of the favourable advantages of the proposed SDOSN is that it can support complex queries to deal with accurate semantic service matchmaking. This research refers the complex queries as multi-attribute queries in particular, which contains more than one semantic concepts in one query. The concepts can be combined in a disjunctive way (i.e. OR connections of attributes) or a conjunctive way (i.e. AND connections of attributes). The disjunctions may require multiple input nodes to resolve, while the conjunctions can be processed at a single node which meet all the attributes in the query.

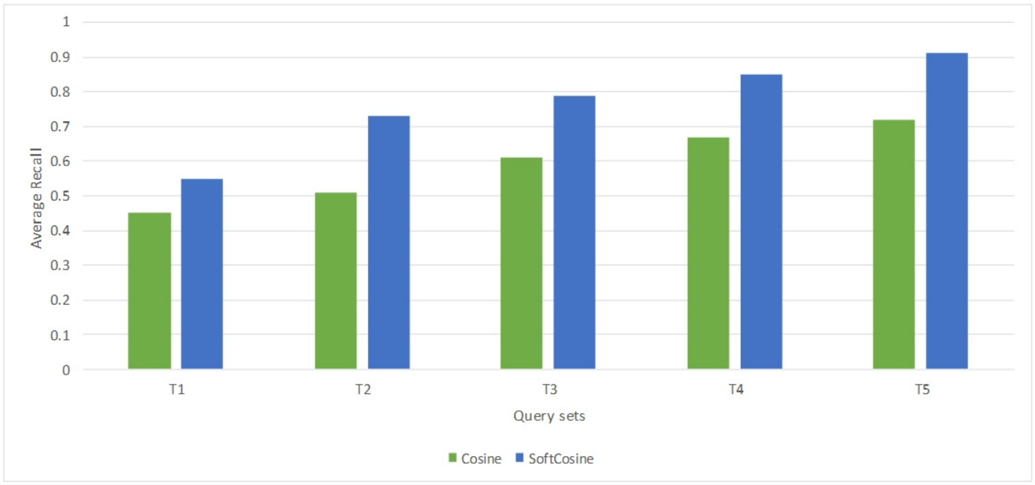


Figure 4.11 Average recall for cosine and soft-cosine match methods

In this section, the effectiveness and correctness of soft-cosine similarity are evaluated through simulations. Five query sets have been constructed including T1 (queries with one term), T2 (queries with two terms), T3 (queries with three terms), T4 (queries with four terms) and T5 (queries with five terms). Each query set has 20 queries randomly selected from the benchmarking datasets, in which disjunctive queries and conjunctive queries are distributed in the uniform distribution. The experiment results are shown in Figure 4.11, Figure 4.12, and Figure 4.13. The proposed soft-cosine method is compared with traditional cosine method through commonly used metrics in information retrieval including average recall, average precision and F1 measure.

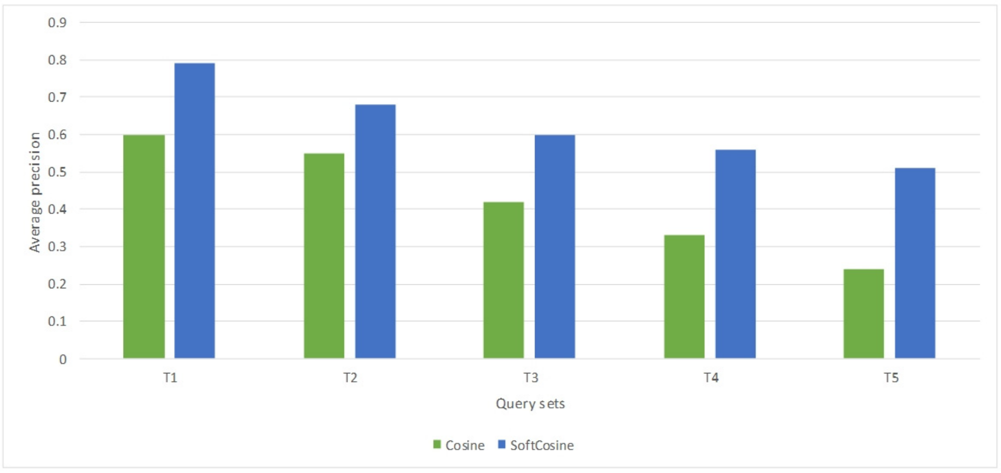


Figure 4.12 Average precision for cosine and soft-cosine match methods

In general, for the complex queries with more than two terms, the average recall is higher than the average precision for both methods in the experiment. As shown in Figure 4.11, the query set with more terms has a higher average recall. On the other hand, it can be observed that Figure 4.12 shows query set with more terms has lower average precision. This is due to the nature of similarity-based match methods that intend to retrieve more documents that could possibly satisfy the query.

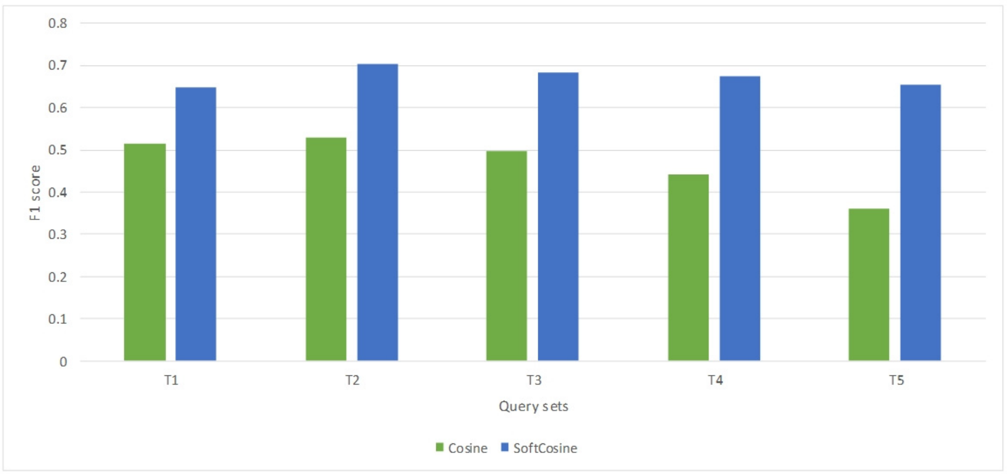


Figure 4.13 F1 measure for cosine and soft-cosine match methods

As shown in Figure 4.13, in all five query sets, the soft-cosine method has a better overall performance of F1 measure than the cosine method. The reason is that soft-cosine similarity can capture the comprehensive correlations between query terms and document terms using feature correlation matrix. While the cosine method treats multiple terms as loose *OR* relationship. For instance, *"play"* and *"game"* are treated as highly correlated terms in the soft-cosine method, but they are considered as independent terms in the cosine method.

4.7.3.3 Evaluation of Network Self-adaptation based on Homophily

Homophily-based networks are established based on the homophily value and the degree of connection controlled by the choice probability *CP*. During bootstrapping phase, the connections are established based on the content similarity using the NFH. After the bootstrapping, each newly joining node connects 2 to 4 nodes in the network. Then during service discovery, new connections are established based on homophily until the average degree of connection reaches a system defined value, in this experiment the value is 24. In other words, users in such social network have an average of 24 friends. The social graph is shown in Figure 4.14.

The δ parameter regulates the importance of the influence of status homophily and value homophily to determine the social similarity between two users in Equation (4.3). In this section, the influence of δ on the network structure and on the search performance is evaluated and analysed.

Table 4.2 Influence of value homophily and status homophily

Properties / δ	**Average degree**	**ACC**	**APL**	**Diameter**
0	24	0.023	2.527	4
0.25	24	0.032	2.543	4
0.5	24	0.104	2.589	4
0.75	24	0.313	3.724	6
1	24	0.621	4.397	8

As shown in Table 4.2, the homophily parameter δ affects the structural properties of SDOSN. These properties include the ACC, APL and network diameter. These changes in properties allow us to observe the influence of parameter δ. When $\delta = 0$, the network is constructed purely based on value homophily that is social interest among users. As δ

increases, the status homophily has more impact on the network adaptation. The ACC grows rapidly as the status homophily encourages users to connect more common neighbours. According to the calculation of ACC, the more connections within the neighbourhood, the greater ACC is. Furthermore, the APL increases gradually since in a highly connected social network, the APL tends to be short. From the range of 0 to 0.5, it can be seen that the APL remains nearly the same value approximate 2.5; from range 0.5 to 1, the APL increases and reaches the largest distance of 4.379 when $\delta = 1$. The network diameter measures the longest graph distance between any two nodes in the network. The network diameter shows the similar trend along with APL. This is because they are both the metrics to measure the node's graph distance in the network. The longer diameter means that it requires more query cost to find services between the two distant nodes.

The degree distribution of the social network is also influenced by the parameter δ. It can be observed from Figure 4.15, the degree distribution follows a normal distribution for evaluated value 0.25, 0.5, 0.75. When the δ increases, the bell-shape of the degree distribution becomes narrow and taller, which means the frequency of the average degree grows. In mathematical terms, the variance of degree become smaller when δ increases.

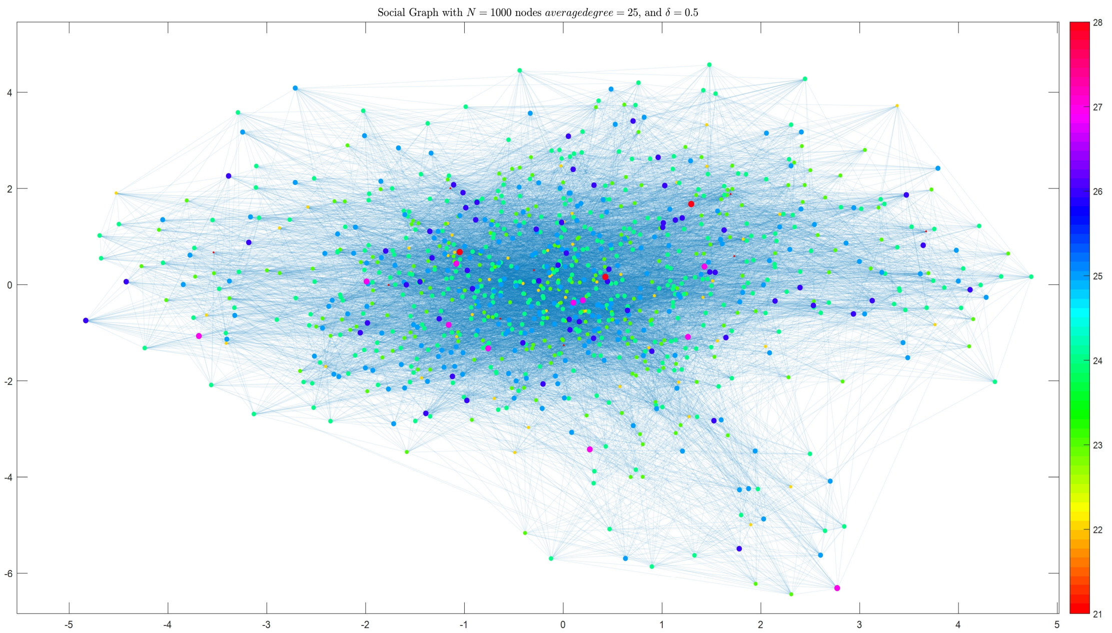


Figure 4.14 Social graph with 1000 nodes, average degree 24 and delta=0.5

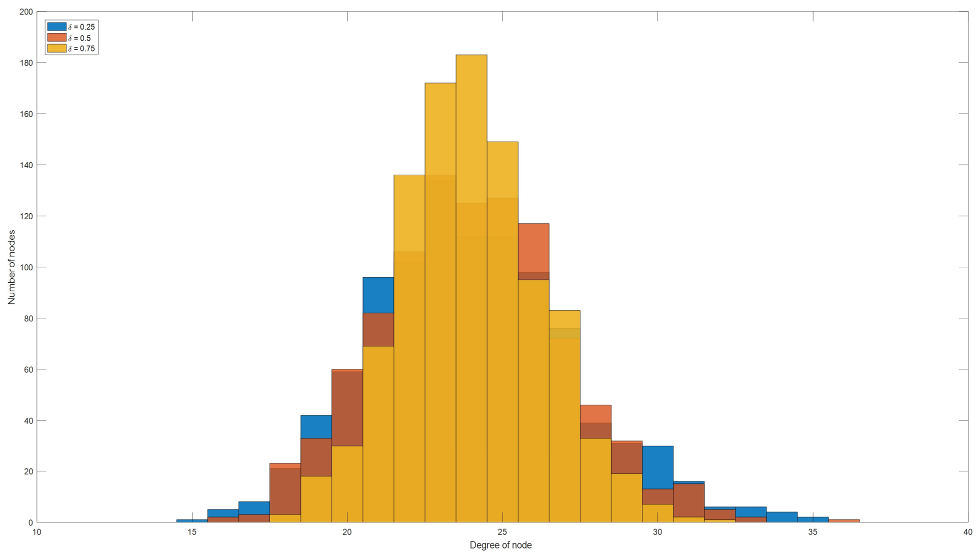


Figure 4.15 Degree distribution for delta 0.25, 0.5 and 0.75

These properties of network structure indicate the choice of the proper homophily parameter δ for neighbour self-adaptation. The preferable range of δ is between 0 and 0.5. In this range, the connection brings more importance on the value homophily, which is biased to the common interest. The interest can create semantically similar connections for users. The ACC indicates that small semantic communities are created and APL between communities is short. Such networks have great advantages in service discovery since queries can find large number of similar services within a semantic community and the query routing paths tend to be short. Specifically, the best parameter for δ is set as 0.5 in the following experiments.

4.7.3.4 Comparison of Search Methods

This section compares various search strategies proposed for service discovery in decentralised systems. The content of query message consists of two features: homophily information of the query initiator and the topics that describe the service requirement.

The network structure of real-world decentralised system can be highly heterogeneous. During simulation, based on the measurement studies of complex networks, three common network structures considered and constructed as follows:

Random Network structure (RN): where social connections between nodes are established randomly with probability $0 < p < 1$ according to Erdős–Rényi model [197].

Scale-Free Network structure (SFN): where social connections between nodes are established based on the degree of connection. Nodes with a high degree of connectivity have a greater probability of receiving a new link than agents with a low degree of connection [100].

Homophily-based Network structure (HN): where social connections are established based on the homophily value and the degree of connection controlled by the choice probability *CP*.

In order to conduct experiments with fairness, the network structures are constructed with the nearly same average degrees of connection. The detailed network properties are shown in

Table 4.3.

Table 4.3 Network properties

Topolog	**Average**	**Min degree**	**Max**	**ACC**	**APL**	**Diamet**
Random	4	2	11	0.002	6.14	13
Scale-	4.173	2	112	0.034	7.53	24
NFH	4	2	10	0.066	5.53	9

The proposed search model *Choice Probability* for decentralised service discovery has also been compared with various search models used in complex networks, differing by the way of neighbour selection in each step. These models are listed as follows:

Random walk (RW): a search strategy utilising k-walker random walks, which forward a query *k* random selected neighbours.

Routing Index (RI): a search model utilising neighbours' semantic information to forward a query to the neighbour that has the highest similarity [80].

Gnutella with Efficient Search (GES): a search model utilising semantic group and random walk to forward a query to the relevant semantic group [85].

Neurogrid: a search model utilising network shortcut to forward a query to the directly matched topic-nodes [92].

ESLP: a search model utilising network shortcut to forward a query to the direct matched and associated topic-nodes [93].

EDSD: a search model utilising node similarity and degrees of connection to forward a query to the selected neighbours [115].

Choice probability (CP): the proposed search model that utilises shortcuts, semantic group and choice probability during query routing process.

During service discovery process, the forwarding degree is set as 2, and *maxTTL* is set as 5 for all the search models. The summary of search models is shown in Table 4.4. The topic size in knowledge-based models such as RI, Neurogrid, ESLP and CP is set as 20. The similarity between a query and a service is calculated based on soft-cosine method. The similarity threshold θ to determine the relevancy is set as 0.25 for all the compared search strategies.

Table 4.4 Summary of search strategies

Model	**Description of strategy**
RW	Search based on random walk
RI	Search based on semantic group
GES	Search based on semantic group and random group
NeuroGrid	Search based on network shortcuts
ESLP	Search based on network shortcuts with associate topics
EDSD	Search based on node similarity and degree
CP	Search based on shortcuts, semantic group and choice probability

4.7.3.5 Evaluation of Search Efficiency

In this section, the performance of the decentralised search models is evaluated and compared in three network structures including Random Network, Scale-Free network and Homophily Network are shown from Figure 4.16 to Figure 4.24. General conclusions can be made from observations of the experimental results are: 1) the success rate, resource cost and query efficiency of the proposed CP model outperforms other models under three network structures. 2) Homophily Network exhibits higher overall performance for compared search models over Scale-Free Network and Random Network.

3) The network shortcuts based models such as NeuroGrid, ESLP and CP have relatively higher query efficiency than those methods without considering network shortcuts including RW, RI, GES and EDSD.

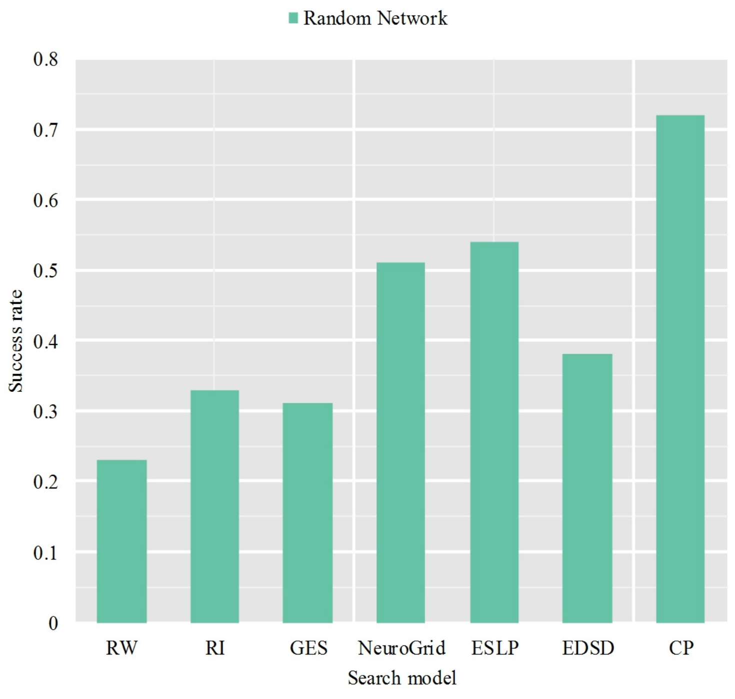


Figure 4.16 Success rate under a Random Network

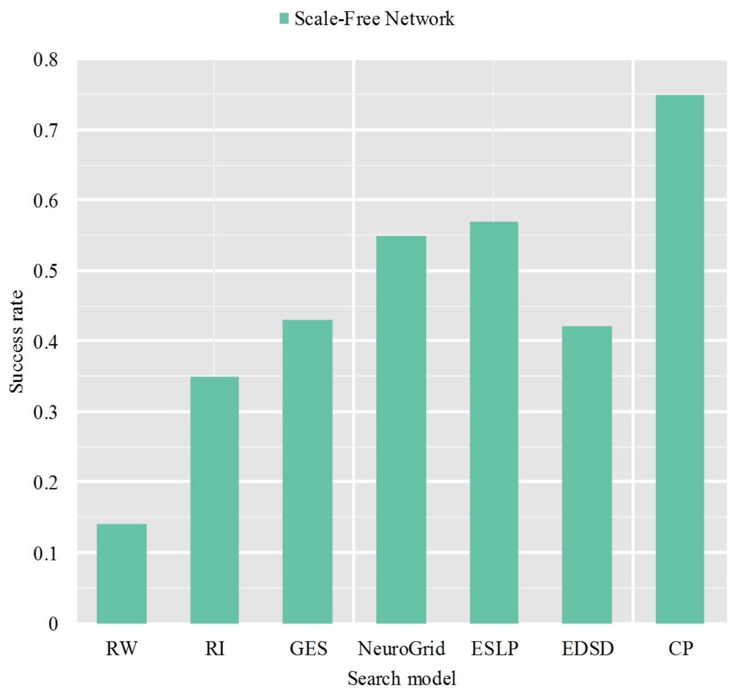


Figure 4.17 Success rate under a Scale-Free Network

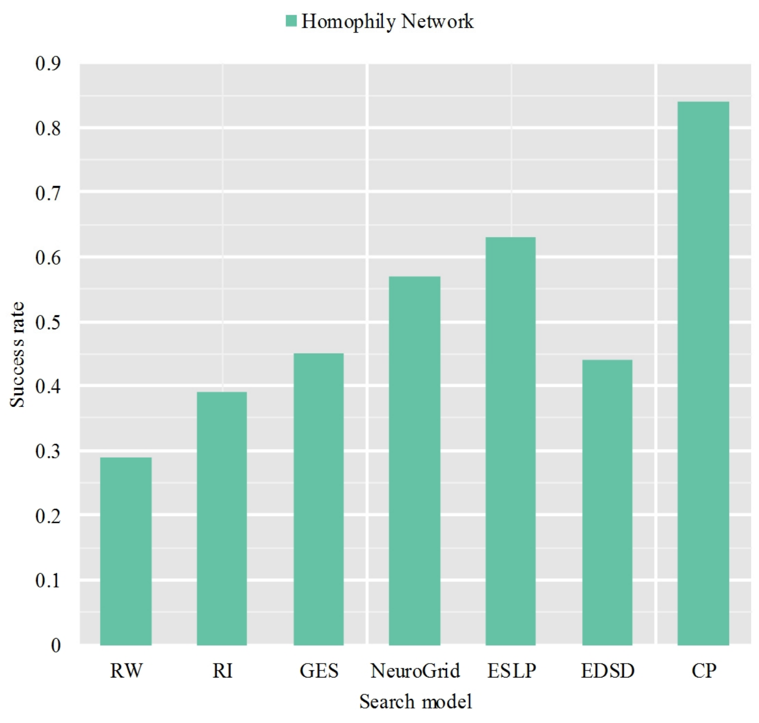


Figure 4.18 Success rate under a Homophily Network

Experiment results for the success rate of compared models under three network structures are shown in Figure 4.16, Figure 4.17 and Figure 4.18. The results reveal that the proposed model CP achieves the best performance of success rate over other search models regardless the variances of the network structures. The success rate of CP is over 70% in Random Network and Scale-Free Network and reaches 84% in Homophily Network, which means CP can resolve most queries within the TTL limit. The next best model is the ESLP model that achieves 55%, 57% and 63% under three network structures respectively. From observed results, CP achieves more than 30% increase in the success rates. In other words, CP has found 30% more requested services averagely under the same TTL and network structures. This is because of the fact that the proposed CP integrates the semantic group and choice probability to guide the query routing process. The similarity metrics of the semantic group can determine different query routing strategies such as exploitation strategy and exploration strategy in each hop, which can effectively locate the relevant nodes. Besides, during neighbour selection, CP has a higher probability of selecting the nodes with high degrees of connection, which expands the search range rapidly, hence, more candidates will be returned for similarity evaluation in each hop. As a whole, more relevant services can be found during query routing. On the other hand, other methods consider only partial information of semantics or degree of connections, which leads to relative low success rates in the search.

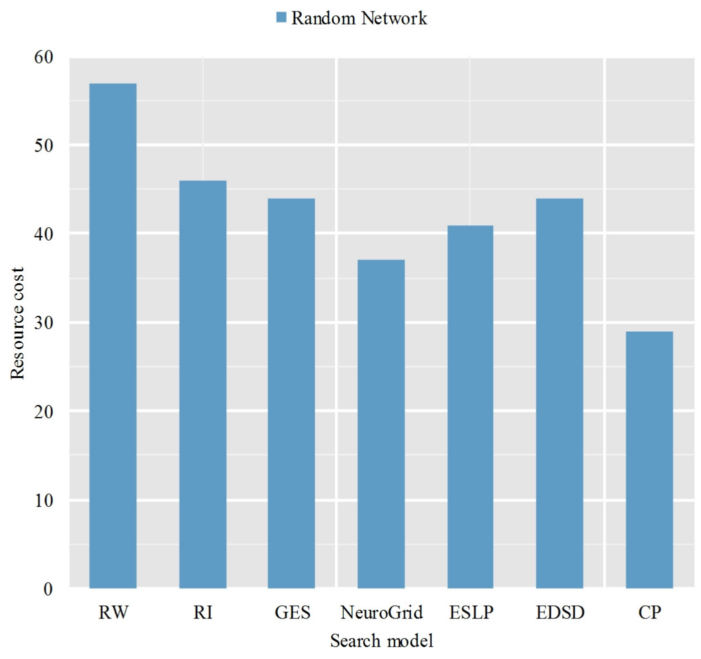


Figure 4.19 Resource cost under a Random Network

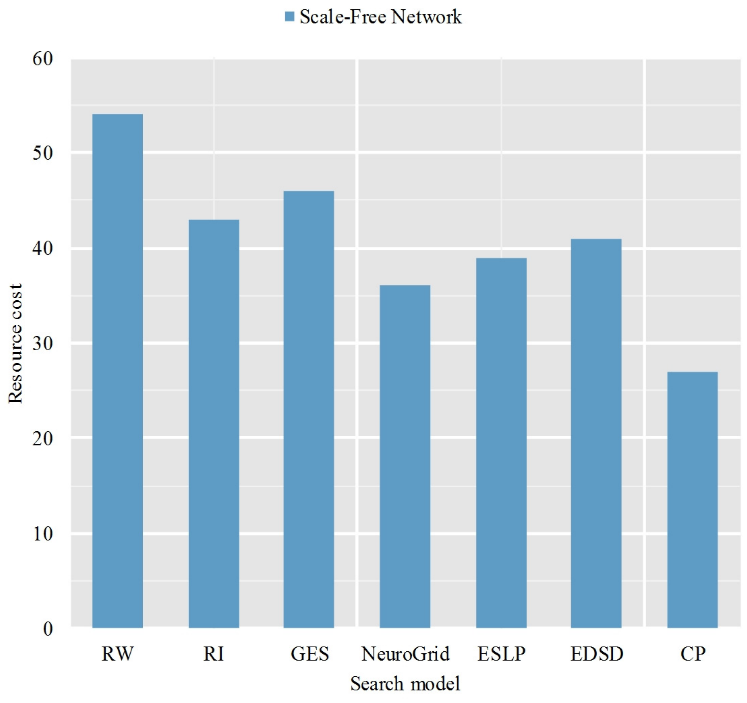


Figure 4.20 Resource cost under a Scale-Free Network

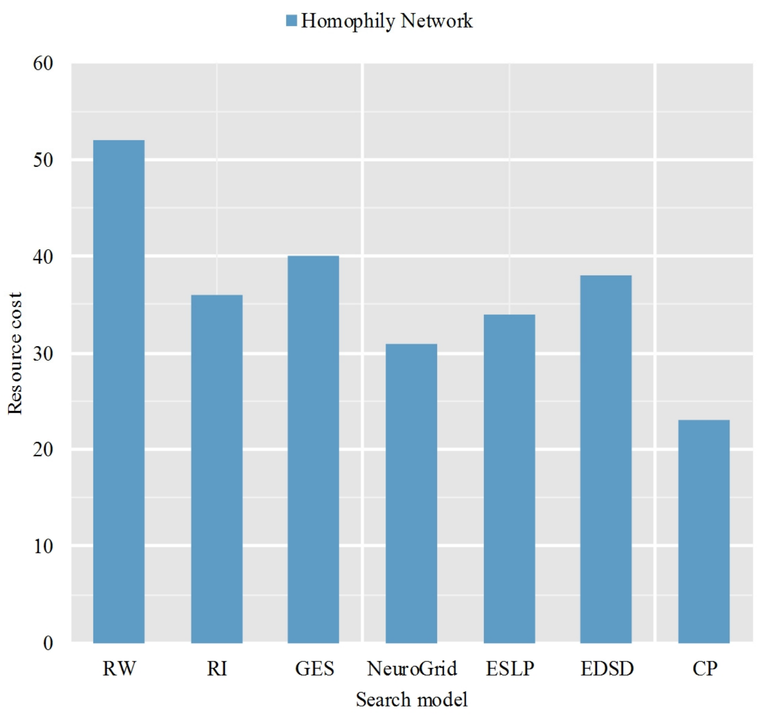


Figure 4.21 Resource cost under a Homophily Network

Experiment results for the resource cost are depicted with histograms in Figure 4.19, Figure 4.20 and Figure 4.21. The obtained results show the average number of visited nodes of compared models to resolve a given query. The fewer nodes involving the query routing process cost less bandwidth and incurs fewer duplicate messages. Across three figures, the proposed model CP achieves the least resource cost over other compared models in three network structures, whereby CP shortens routing path lengths using topology adaptation. This topology adaptation employs the homophily feature of CP that is able to form personalised semantic communities via previous successful search. Those nodes that have a higher homophily score with the service requester can serve as candidates to establish friend connections. In this way, similar nodes are grouped in the same semantic community in which relevant queries can be effectively resolved. Other search models rely on static topologies and the personalised information are neglected during query routing.

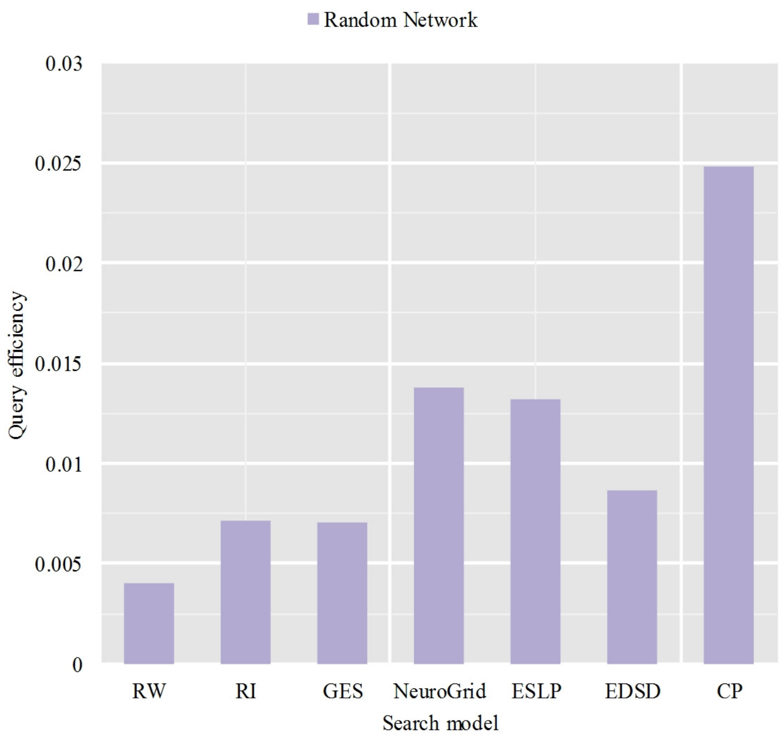


Figure 4.22 Query efficiency under a Random Network

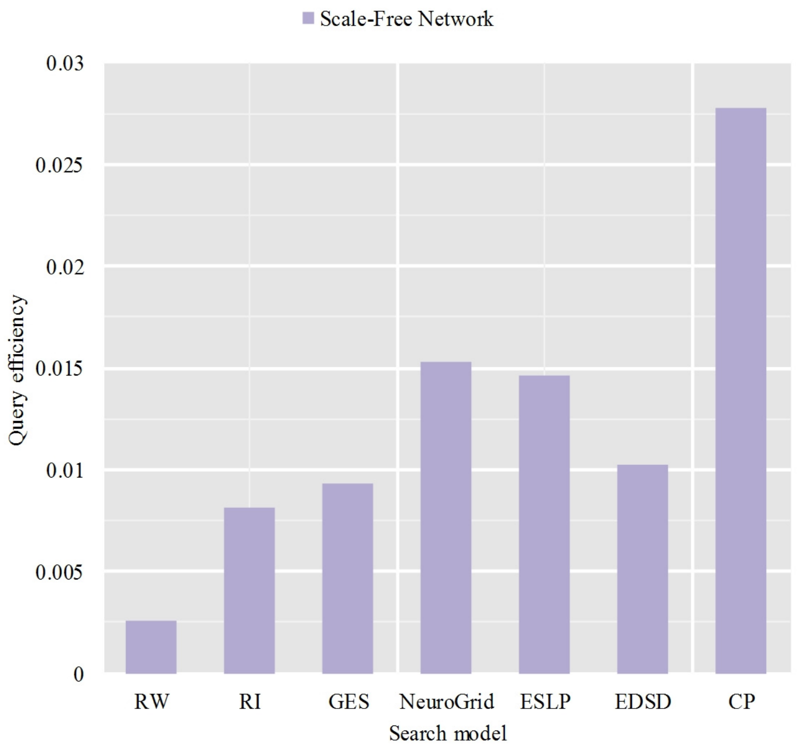


Figure 4.23 Query efficiency on a Scale-Free Network

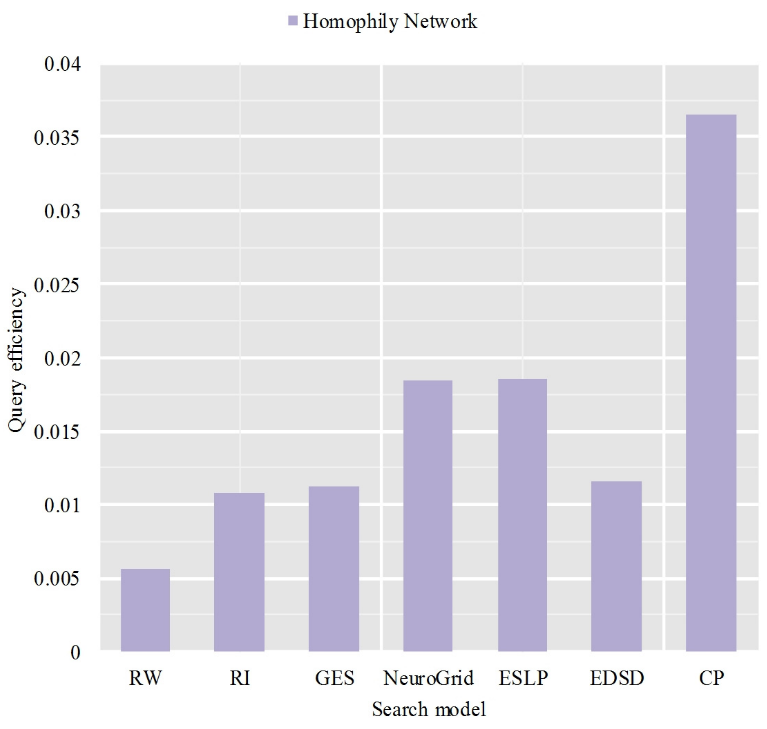


Figure 4.24 Query efficiency on a Homophily Network

The results of query efficiency are shown in Figure 4.22, Figure 4.23 and Figure 4.24 presenting an overall performance evaluation on the success rate against its resource cost. The shortcuts based models such as NeuroGrid, ESLP and CP achieved relatively higher query efficiency than RW, RI, GES and EDSD. This is because the fact that network shortcuts can effectively direct query to the service providers, which leads to matches without forwarding queries to neighbours. The success rate depends on the sizes of the network shortcuts and update strategy. With simple LRU scheme, the frequent queries can be handled effectively. The CP achieves the highest query efficiency over NeuroGrid and ESLP. The key difference that the proposed model CP makes is that CP utilises social similarity and degree of connection to enhance the performance of network shortcuts when a query is not recorded or has been replaced by other queries.

Moreover, from the above results, it can be seen that network structures also play an important role during search discovery process. In the Random Network, social links are connected randomly regardless the degree of connection and user similarity. The service discovery in such network often results in a long average routing path to resolve query messages due to the relevant services are scattered randomly in the network. With a fix

TTL range, the success rate of query messages is very low. For the Scale-Free Network, a certain number of nodes have very high degrees of connection and most nodes only have low degrees of connection. Scale-Free Network does not consider user's similarity; hence, the relevant services are also scattered. Those nodes with high degrees of connection can serve as social hubs that has more candidate neighbours for query routing. As a result, the success rate in Scale-Free Network is marginally higher than the Random Network. The Homophily Network connects nodes with social similarity i.e. homophily. The outcome of the constructed social network shows a small-world property that has large ACC and short APL. Users with similarity interests are self-organised in communities in which the semantic distance between nodes tends to be short. Many query messages that related to user's interests can be resolved within the community. Therefore, the success rate for all the compared search models achieves the best results in the Homophily Network.

4.8 Summary

This chapter presented an efficient service discovery approach to utilise homophily features in semantic query routing. Firstly, a semantic matchmaking method based on soft-cosine similarity has been proposed to capture the hidden semantic correlation between concepts and support the accurate semantic match for complex queries. Then, a homophily-based approach has been proposed to carry out the query routing for the decentralised service discovery using effective social network adaptation. This homophily approach integrates two perspectives of social features: the status homophily and the value homophily to define the closeness between users. Then the social network adaptation is spontaneously conducted through user interactions during service discovery, whereby nodes can accumulate social knowledge and find more suitable candidates to establish new connections using choice probability. More specifically, this choice probability determines the likelihood of resolving the query using user's closeness and degree of connection. Those neighbours that have the high score of choice probability will be selected to disseminate the query in the next hop. After successful service discovery, the service providers also serve as friend candidates to adapt user's social connections. In this sense, with the adaptation of personal social network, users are self-organised into communities then form a homophily network. In such network, the average discovery paths tend to be short and communities exhibit high clustering coefficients. As a result, the semantic query routing evaluates user’s closeness and degree of connection to select suitable neighbours to disseminate the query in the network. This approach achieves high success rate and low overhead in various network structures.

Extensive experiments have been conducted through simulations to verify the search performance of the proposed service discovery approach. In the evaluation on semantic matchmaking, the soft-cosine outperforms the traditional cosine methods for complex queries in terms of recall, precision and F1 measure. The proposed service discovery approach CP is compared with other search models under three network structures. The success rate, resource cost and query efficiency are discussed. The results show the proposed approach achieves the best performance in three networks. In addition, the Homophily Network constructed during service discovery shows the higher query efficiency than Random Network and Scale-Free Network.

To conclude, the homophily-based service discovery approach can cope with complex queries and exhibits good performance in different network structures. The query routing is based on a greedy strategy that selects neighbours with high choice probability score. However, queries are forwarded to a fixed number of neighbours and the routing efficiency is not optimised. Next chapter will introduce an advanced routing strategy that optimises the routing performance and network overheads.

5 Advanced Query Routing with Adaptive Forwarding Degree

5.1 Overview

Previous chapter presented a homophily-based service discovery approach to accelerate semantic query routing through effective social network adaptation. The efficiency of service discovery in SDOSN depends on the topology design and the decentralised search model. The topology determines whether the nodes know each other in the social overlay network. In order to achieve high recalls of services against network dynamic changes and data unavailability, the topology needs to connect users with a relatively large set of friends to facilitate service discovery. As a downside, given the lack of centralised knowledge, flooding the neighbours causes significant network traffic and the duplicate messages saturate the network.

To cope with the traffic overhead, many research works have proposed k-walker routing [164, 212, 213] that selects k neighbours randomly or deterministically in each routing hop and greedy routing [91, 115, 214] that only selects the most suitable neighbour to forward the query to. Both the k-walker routing and greedy routing can prevent the duplicated messages to a certain extent. However, they have some limitations in terms of query efficiency. The query efficiency is defined as the ratio of success rate and resource cost. The k-walker routing requires predefined k for each decentralised system and the query efficiency varies with different values of k and different sized systems. There are no theoretical guidelines for choosing the proper *k* in query routing. Meanwhile, the greedy routing indeed reduces the network traffic in each hop. However, the total number of found services is very limited within a fixed TTL, hence low query efficiency. This chapter proposes a swarm intelligence routing algorithm with adaptive forwarding degree to address the limitations of existing works. The proposed algorithm can effectively adapt to different network structures and content changes, and optimise the trade-off problems between the network traffic and search performance to achieve high query efficiency.

5.2 Problem Definition

The use of Equation (4.6) has generated a probability of resolving the query q for each of the neighbouring nodes of v_y. Node v_y can forward the query to its neighbourhood 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 [82, 92, 115, 215] 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 reduced to the lowest level. However, those less promising nodes also can possess the requested service. As a result, the recall of this greedy neighbour selection method is low since only partial relevant services are returned during a query routing session. On the contrary, if nodes utilise a large fixed forwarding degree by flooding the neighbours 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.

Figure 5.1 and Figure 5.2 shows a simple scenario of how the forwarding degree affects the performance of service discovery. There are two routing strategies with a static number of receivers per hop ($d = 3$) in Figure 5.1 and an adaptive number of receivers per hop ($d = 2 \sim 4$) in Figure 5.2. The black dots represent the nodes that share requested services, while the grey dots show the nodes that are highly correlated with the query and know who has the requested services. As shown in this scenario, the fixed forwarding degree method chooses a fixed number of neighbours to forward a query, while the adaptive forwarding degree method selects a dynamic number of neighbours in each hop.

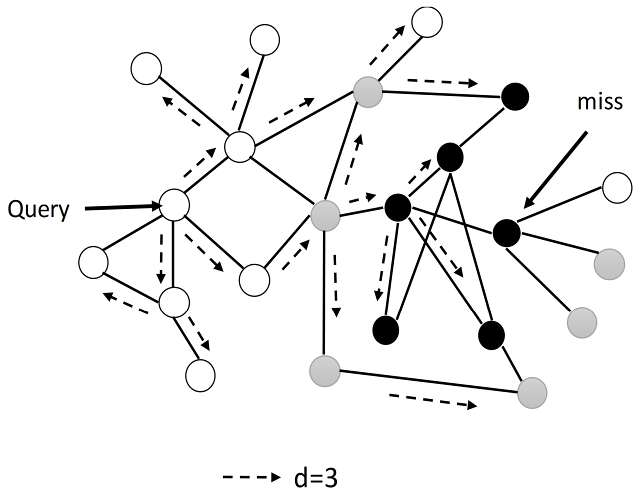


Figure 5.1 Query routing with a fixed forwarding degree

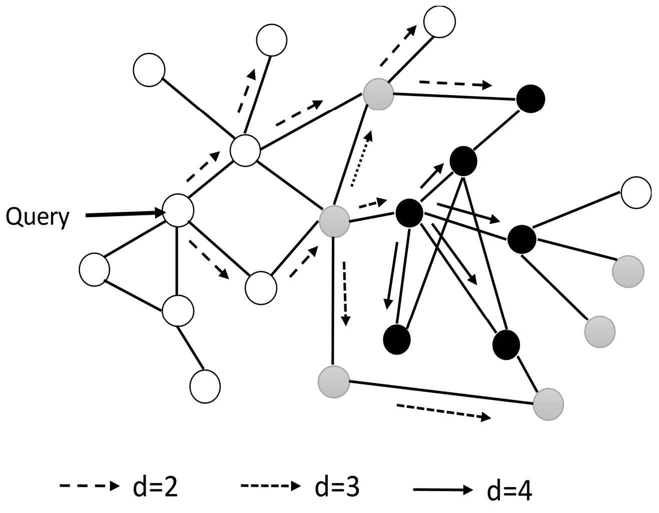


Figure 5.2 Query routing with an adaptive forwarding degree

When forwarding a query to a node within its semantic group, the query can be resolved by both methods. However, in the homophily-based network, many neighbours are connection based on social similarity, which means potentially a query can find more relevant service in the neighbourhood. Figure 5.1 shows the fixed forwarding degree method retrieves 3 out of 4 relevant nodes due to the predefined fix number of forwarding. On the other hand, Figure 5.2 shows the adaptive forwarding degree finds 4 out of 4 relevant nodes in the same hop. As a result, adaptive forwarding can discover more relevant services in the service discovery process. Moreover, the number of duplicate query messages is also controlled and can be reduced significantly during query navigation to the service providers. The navigation is the routing path that directs the query to the nodes within the relevant semantic group that can resolve the query. In the routing path, those nodes that are beyond the relevant semantic group should have a relatively low forwarding degree, as they have very little information to answer the query. Based on this observation, the adaptive routing method has the most favourable advantage over the fixed forwarding degree method.

In order to improve the trade-off situation between network traffic and recall and to 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 in the routing process to adapt volatile decentralised networks. The goal of query routing is to determine the shortest query paths while the number of found services is maximised and the steps to locate the services are minimised.

The formal definition of the problem is given as:

Participants $PN = \{P_1, P_2, ..., P_m\}$ are self-organised in an unstructured P2P network with the maximum degree of connection λ. Given a query Q issued by P_i, the relevant *service set* $\Omega(S)_Q = \{S_1, S_2, \ldots S_n\}$ located in *r* nodes. $PS = \{P_{s1}, P_{s2} \ldots, P_{sr}\}$ is denoted as the *service provider set*, where $|PS| = r$, $r \geq 1$ and $PS \subseteq PN$. During each query hop in the query routes, the forwarding degree is variable $d \in [1, \lambda]$. The number of visited nodes in one successful query route from P_i to $P_{s1} \in PS$ is denoted as $n_1(d)$. Therefore, the number of visited nodes in all successful query routes is summarised as $\sum_{j=1}^{r} n_j(d)$. The total number of the discovered service providers during the life cycle of Q is denoted as $\psi(d)$. The goal of adaptive forwarding is to achieve the following objective function: $\arg\min_d f(d) := \{d \mid (\sum_{j=1}^{r} n_j(d) \big/ \psi(d))\}$. In another words, the objective function aims at finding appropriate forwarding degree *d* to minimise the resource cost during query routing.

5.3 A Swarm Intelligence Inspired Approach

5.3.1 Swarm Intelligence

This section introduces a 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 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 [216]. During the forging process, ants visit the shorter paths more frequently, leaving 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.

Service discovery in decentralised social network 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 neighbours. The behaviour of users discovering target services resembles the phenomenon of biological swarms locating food resources. The likelihood of finding the target services relies on the user's knowledge about the search path, which is 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, which resembles the process of pheromone diffusion. Besides, the pheromone in the ecosystem evaporates over time, while the knowledge index has a fixed sizes and unused knowledge will vanish and be replaced with new information through time.

5.3.2 Adaptive Forwarding Framework

The proposed OSS algorithm is designed by adopting the service discovery behaviour of swarm intelligence. This algorithm maximises the phase of pheromone exploitation by utilising 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). Individuals with the lowest level of sensibility can select any locations marked with pheromones in each hop for maximising the forwarding degree. When the level of sensibility increases, individuals ignore the locations marked with the lower level of pheromones and select the 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 utilise 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 Figure 5.3.

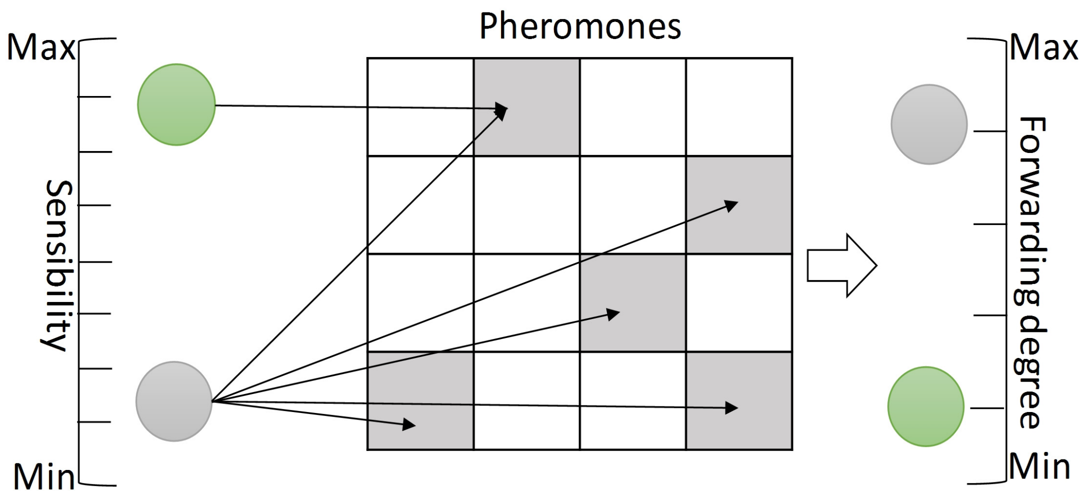
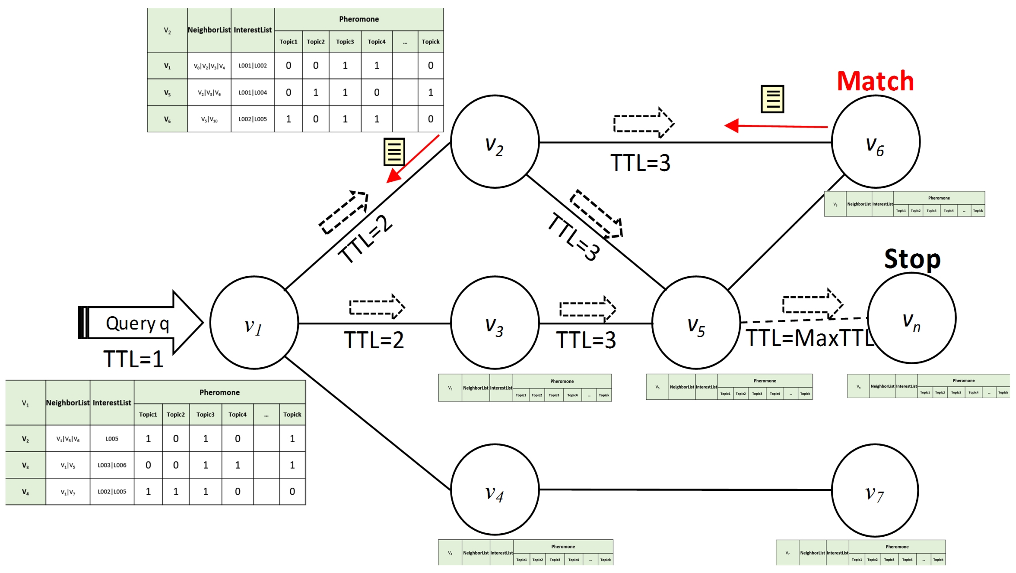


Figure 5.3 The impacts of the olfactory sensibility on forwarding degree

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 flavour using their olfactory sensibility. During the exploration, each one gets some flavours and distributes a pheromone in an amount proportional to the amount of the found flavour. The flavours can be seen as the historical query topics via director neighbours in the social network. The pheromone can be seen as the topic hits, so it will be enhanced after each successful service discovery. Note the topic hits do not necessarily indicates the neighbour is the service provider for the corresponding topic; they provide the hints information about the route to the topics.

5.3.3 Query Routing Scenario based on Pheromone

The pheromone information is stored in the knowledge index; a scenario of the discovery process is illustrated in Figure 5.4. 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 utilises its LKI to calculate the choice probability of its neighbours v_2, v_3 and v_4 based on the *"Neighbourlist"* and *"Interestlist"* columns. Having obtained the values of choice probability in the neighbourhood, v_1 learns the pheromone regarding the query q using *"Pheromone"* column. Using the OSS algorithm, v_1 determines the overall scores of its neighbours and forwards the query to the selected neighbours v_2 and v_3. Then the query is handled by v_2 and v_3, and remains unsolved. v_2 and v_3 utilise their knowledge index to forward the query to selected neighbours that have not received the query before. It can be seen 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 v6, 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 the parameter *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.



V_2	NeighborList	InterestList	Pheromone					
			Topic1	Topic2	Topic3	Topic4	...	Topick
V_1	$V_0\|V_2\|V_3\|V_4$	L001|L002	0	0	1	1		0
V_5	$V_2\|V_3\|V_6$	L001|L004	0	1	1	0		1
V_6	$V_5\|V_{10}$	L002|L005	1	0	1	1		0

V_1	NeighborList	InterestList	Pheromone					
			Topic1	Topic2	Topic3	Topic4	...	Topick
V_2	$V_1\|V_5\|V_6$	L005	1	0	1	0		1
V_3	$V_1\|V_5$	L003|L006	0	0	1	1		1
V_4	$V_1\|V_7$	L002|L005	1	1	1	0		0

Figure 5.4 Query routing based on pheromone table in the knowledge index

5.4 Olfactory Sensitive Search Algorithm

OSS algorithm is an adaptive forwarding algorithm, which selects a subset of promising neighbours 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 that 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.

5.4.1 Algorithm Notations

Now the notations used in the remainder of this chapter are introduced as follows.

Swarm member *j* is the current query handling node*; m* is the total number of nodes in the network; *n* is the number of neighbours of *j* and *k* is the number of flavours (i.e. topics) of pheromone in a neighbouring node of *j*.

Search Space: The search space of *j* $(1 \le j \le m)$ is defined as $X_{ij} = (x_{0ij}, x_{1ij}, \ldots, x_{kij})$ *where* *i* $(1 \le i \le n)$ is the *n*-dimensional neighbour space, and $t\,(1 \le t \le k)$ is the k-dimensional topic space in each dimension of the neighbouring space.

Location: $x_{tij}(1 \le t \le k, 1 \le i \le n)$ is the location information of a topic *t* marked by swarm member *i* for the receiving member *j*.

Query Queue: $Q_j = \{q_1, q_2, \ldots, q_c\}$ is a queue of queries to be resolved by *j*. Queries are sorted by the receiving time and is process by the first in and first out (FIFO) scheme.

5.4.2 Adaptive Search Algorithm

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

In the initialisation 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 initialises the pheromone P_{0j} and the olfactory sensibility S_{0j}.

Define $P_{0j} = S_{0j} = \max CP(X_{ij}, u_q)$ *where Xij is j's neighbour* $(1 \le i \le n)$; $CP(X_{ij}, u_q)$ is calculated by using Equation (4.6), and the max choice probability of *j*'s neighbours is defined as the initial sensibility of *j*.

The initial sensibility S_{0j} is calculated by the choice probability of utilising the homophily score and connection degree of neighbourhood to estimate the max likelihood of resolving a query before exploring its pheromone table.

During the discovery phase, the walk behaviour of swarm member *j* is described as $f_{ti} = f(x_{tij})$, where $t\ (1 \le t \le k)$ is the flavour of pheromone, i.e., the topic of a neighbour in the knowledge index of *j*. Each walk of *j* has *k* small steps to get the flavours of *i*. The *f* function is a match function between a query *q* and *t* based on the distance between the semantic concepts in the DMOZ. The function presented by [217] is used to calculate the distance between the semantic concepts. The *f* is defined as:

$$\underset{1 \le t \le k}{f_{ti}(q,t)} = e^{-\alpha l} \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} \tag{5.1}$$

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 organisational ontology, in this research, the DMOZ. *l* is the depth distance of the two concepts in the ontology. $\alpha \ge 0$ and $\beta \ge 0$ are parameters scaling the contribution of shortest path length and depth, respectively.

$$P_{i(i \in R_j)} = \max_{1 \le t \le k} f_{ti} \tag{5.2}$$

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

$$S_{max} = P_{max};\ S_{min} = P_{min} \tag{5.3}$$

S_{min} and S_{max} are the minimal and maximal possible values of the sensibility of the node *j*. 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 *Smin*, *j* selects all the locations marked with pheromone. As a result, the forwarding degree will be maximised.

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

$$S_{ij} = S_{\max} - (S_{\max} - S_{\min}) \times S_{0j} \qquad (5.4)$$

Sij 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 the choice probability. 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 is sensed and considered by *j* in the next forwarding process. By comparing the adaptive sensibility and marked pheromones in the neighbourhood, new locations for next hop are generated as:

$$X'_{0j} = \begin{cases} X_{ij} & if\,(P_i \geq S_{ij})\; 1 \leq i \leq n \\ X_{0j} & otherwise \end{cases} \qquad (5.5)$$

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 neighbour 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 only selects the neighbours whose pheromone is greater than or equal to *j*'s sensibility for forwarding the query. Neighbours 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 predefined parameter *maxTTL*. Then, response messages will be sent back to the query requester. Finally, the memory resource of real-time computation will be released. The OSS algorithm is presented in the Algorithm 5.1.

Algorithm 5.1 Olfactory Sensitive Search (OSS)

Input: Query processing node j, neighbour list of $j: R_j = \{r_1, r_2, \ldots, r_n\}$, a finite queue $Q = \{q_1, q_2, \ldots, q_c\}$ and maximum Time-to-Live value $MaxTTL$

Output: The forwarding list of node j

```
1.  $Initialization\left(SeachSpace\ X_{tij}, QueryQueue\ Q\right)$
2.        let $X_{tij} \leftarrow loadKnowledgeIndex(j, R_j)$
3.  // Q contains the queries that need to be forwarded by $j$
4.  $Q \leftarrow getForwardingQueryMessages()$
5.  for $l \leftarrow 1$ to $c$ do
6.        if $getQuery().Next() \neq \varnothing$ then
7.            $q_l \leftarrow Q.Next()$
8.            $q_l.ttl \leftarrow q_l.getTTL()$
9.        $U_{q_l} \leftarrow q_l.getQueryRequestNode()$
10.       $P_{0j} = S_{0j} \leftarrow HomophilyScore(j, U_{q_l})$
11.       if $ttl \leq MaxTTL$ then
12.           for $i \leftarrow 1$ to $n$ do
13.               // Iterate over all neighbours of $j$
14.               if $getNeighbors().Next() \neq \varnothing$ then
15.                   $r_i \leftarrow getNeighbors(j).Next()$
16.               for $t \leftarrow 1$ to $k$ do
17.                   //Iterate over all topics in the i-th neighbour
18.                   if $getTopics().Next() \neq \varnothing$ then
19.                       $x_{tij} \leftarrow getTopics(r_i).Next()$
20.                   $f_{ti} \leftarrow f(x_{tij})$
21.                   //Semantic distance between query $q_h$ and topic $t$
22.                   $f_{ti}(q_k, t) = e^{-\alpha l}\dfrac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}}$
23.                   //Learn Pheromone from $i$
24.               $P_i = maxf_{ti}$
25.               $PheromoneList.Add(P_i)$
26.           //Update sensibility of $j$
27.           $PheromoneList.Sort()$
28.           $P_{\max} = PheromoneList().Max\ P_{\min} = PheromoneList().Min$
29.           $S_j = S_{\max} - (S_{\max} - S_{\min}) \times S_{0j}$
30.       for $i \leftarrow 1$ to $n$ do
31.           if $P_i \geq S_j$ then
32.               $ForwardingList(j).Add(r_i)$
33.               return $X_{ij}$
```

```
34.         else
35.             return $X_{0j}$
36.     $forwardQueryto(q_h, ForwardingList(j))$
37.     $q_l.setTTL() = q_l.ttl + 1$
38. return $ForwardingList(j)$
```

5.4.3 Algorithm Example

The mechanism of sensibility and pheromones determining the forwarding nodes for a query *q* between *v1* and its neighbour *v2*, *v3* and *v4* is illustrated as follows.

Using the same example in showed in Figure 5.4. In the initialisation phase, by using Equation (4.6), the choice probabilities of *v2*, *v3* and *v4* 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 *v1* is assigned as $S_{0v1}=\max[0.61,0.55,0.64]=0.64$.

In the discovery phase, assuming the pheromone values from neighbour space are $Pv_2=0.7, Pv_3=0.5, Pv_4=0.2$ based on Equation (5.1) and (5.2) considering semantic similarity, the *v1*'s sensibility range is between 0.2 and 0.7 after learning pheromone information.

By adjusting the sensibility range, the adaptive sensibility of *v1* 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, it obtains the following results: $Pv_2 > S_{v1}, Pv_3 > S_{v1}, Pv_4 < S_{v1}$. Using the decision function Equation (5.5), eventually *v1* forwards the query to *v2* and *v3*. Note in this example, though *v4* has the highest choice probability based on the homophily feature and connection degree, OSS excludes *v4* from the forwarding list based on its adaptive sensibility, since the pheromone information Pv_4 for query *q* is below the level of the sensibility of *v1*.

To conclude, when a node receives a query message, it first checks whether the query has been already received during the past. Redundant queries will be discarded without further processing. Then the node utilises its local service index to score the similarity 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 initialises 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 neighbours. Intuitively, the pheromone of a neighbour is the topic of historical success in query routing via that neighbour. 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 neighbourhood. The olfactory sensibility is an adaptive value depending on the level of pheromone and the choice probability. With a higher level of pheromone information and large choice probability, nodes utilise a larger forwarding degree to exploit the current neighbourhood to locate more relevant services. On the contrary, lower level of pheromone information and small choice probability will lead to a smaller forwarding degree for bandwidth-efficient discovery.

5.5 Evaluation

The simulation platform generates various network structures to simulate complex connectivity of decentralised social overlays. In order to focus on the core value of service discovery in the decentralised environment, the platform can conduct simulations of service discovery under different network structures.

5.5.1 Simulation Methodology

The simulation setup used for the subsequent experiments is the same as section 3.4. The content generation and query generation are the same in line with the method in section 3.4.2.

5.5.2 Simulation Design

Since the simulation in this chapter covers the semantic routing based on pheromone information, the semantic ontology DMOZ is used for calculation of the semantic distance between two topics. Topic ontology construction: The DMOZ project is a taxonomy for classification of web pages. To prepare the topic ontology, the RDF dump of structure and the content was downloaded as of 21st March 2016. All topics were converted to lower-case and in singular format, e.g. "*Computers*" is converted to *“computer”*. The sub-category Top/World is excluded from the experiment, because this subcategory was designed for the multi-lingual purpose for the DMOZ project. In total, the ontology has 682 sub-topics and the average topic path length is 5.73.

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 *random networks* (RN), *Scale-Free networks* (SFN) and *homophily-based networks* (HN). The construction of this network structures has been presented in section 4.7. In each experiment has been repeated for 10 networks for each of the structures and 10000 query messages have been generated in each network. 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. The content of query message consists of two features: homophily information of the query initiator and the query topics that describe the service requirements.

5.5.3 Performance Metrics

- ***Recall***: the number of retrieved services divided by the number of relevant services in the whole repository.
- ***Recall on Time-to-Live (RTTL)***: it 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***: this metric counts the path length for a query from initiator to the first found service provider. Lesser the number of visited nodes, the shorter is the discovery path.

In this chapter, the precision metric is not considered, because all the compared search models use the same match criteria, all the retrieved services are considered as true positives.

5.5.4 Performance Evaluation

5.5.4.1 Evaluation of the Size of Pheromone Table

The size of pheromone table determines how many topics are stored in the LKI that can direct queries to suitable neighbours. Intuitively, if the number of stored topics are greater, the search efficiency will also increase. But the maintenance and computation also become more expensive when the size grows. Therefore, this experiment is designed to evaluate the size of pheromone table under a fixed *maxTTL* of 5 in the homophily-based structure with 1000 nodes in the social network. Six search simulations are conducted with different size of pheromone table. Each simulation assigns all nodes with the same size of pheromone table and each node randomly issues 1000 queries during the simulation. The LRU scheme is used to replace the records in pheromone table when the number of query topics exceeds the predefined size. The average results of each simulation are observed to guide the choice of a proper size to achieve good search quality and a relatively low maintenance cost.

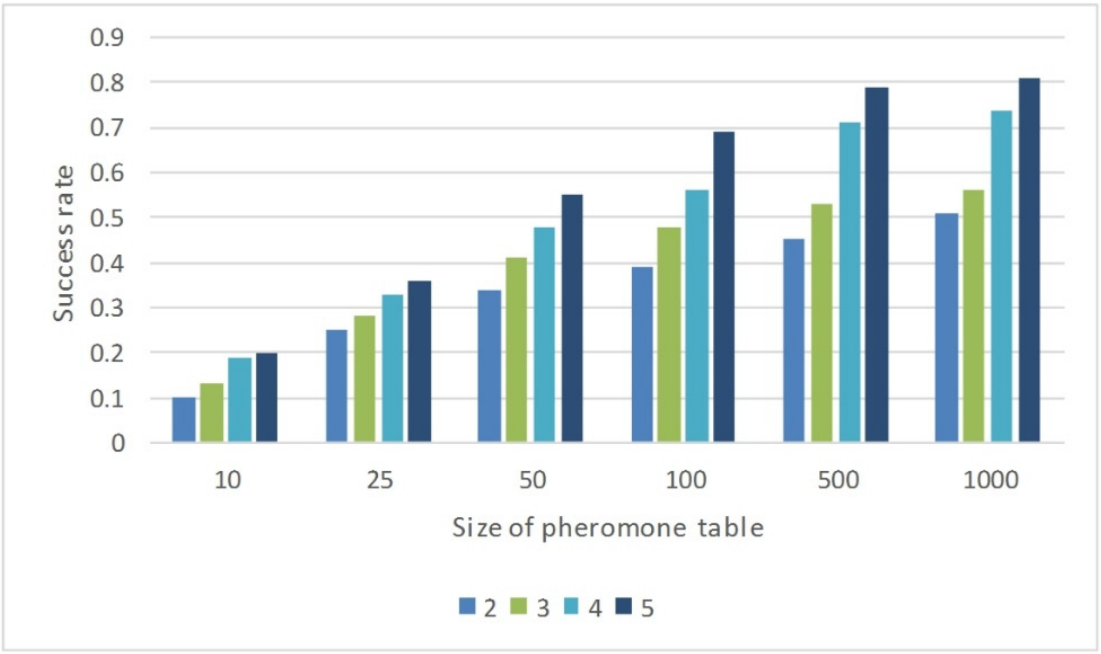


Figure 5.5 Evaluation of the size influence on the success rate

Figure 5.5 shows the influence on success rate of the size of pheromone table. The x-axis shows the 6 groups of histograms where the topic size of topics ranges from 10 to 1000 and in each group the degree of connection changes from 2 to 5. The y-axis shows the average success rate for different sizes. The overall results showed that the larger degrees of connection and the larger size of topics can achieve higher success rate. For the same degree of connection, as the size of topics increases, the success rate gains increment rapidly. In the first and second histogram, with the increase of topic size from 10 to 25, the success rate gains more than 150% for the average degree of connection of 2. When the size reaches 1000, the success rates are all above 50% for the compared degrees of connection, and the highest success rate hits 81% with the degree of connection of 5. On the other hand, when the degree of connection increases, the success rate also exhibits increment for a fixed size of topics in pheromone table. In each group of histograms, it can be seen that there is marginal increment for the first two groups of histograms (topic size 10 and 25) and a relatively large increment for the remaining groups of histograms (topic size 50,100,500 and 1000). The reason behind this phenomenon is because a node with higher degrees of connection is considered more sociable and more queries can be resolved via the node. As a result, more successfully resolved query topics are stored in the pheromone table, which leads to the higher increment in success rate.

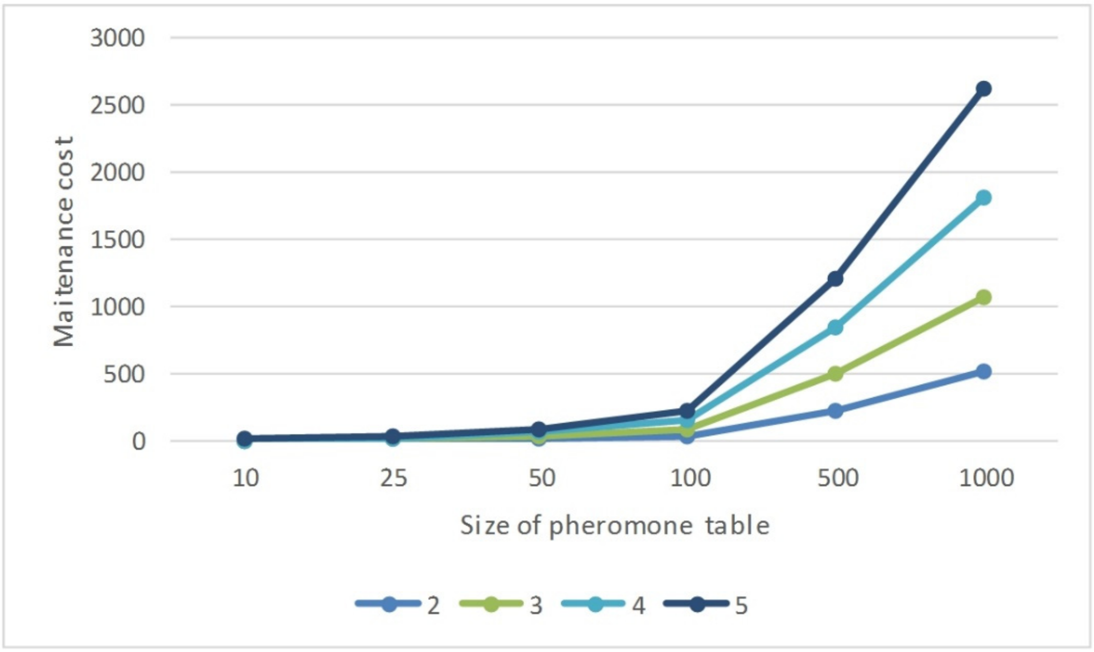


Figure 5.6 Evaluation of the pheromone maintenance cost

Figure 5.6 shows the maintenance cost for different size of pheromone table. The x-axis shows the topic size of pheromone table and y-axis represents the maintenance cost which is defined as the number of recorded topics in pheromone table during the simulation. It can be seen that the maintenance cost exhibits exponential growth when the topic size increases. However, from the results shown in Figure 5.5, it can be observed that when the size of topics increases to a certain level e.g. size 100, the increment of success rate becomes marginal. From topic size 500 to 1000 the success rate for the same degree of connection remains nearly stable. To balance the trade-off between success rate and maintenance cost, the size of pheromone table is set as 100 in the homophily-based network, which can achieve a relatively high success rate and a low maintenance cost.

5.5.4.2 Evaluation of Search Performance in Volatile Networks

Since SDOSN operates the service discovery on an unstructured P2P network, the network dynamism is one of the main challenges for maintaining the social overlay network and supporting effective service discovery. 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 overlay network undergoes frequent reconfiguration, which affects the performance of information propagation and service discovery. Therefore, it is necessary to test whether if the simulation environment can cope with the dynamic changes. In order to simulate a dynamic environment, during the simulation each node is assigned with an absent probability. The experiment uses the network setting in

Table 4.3 and TTL is set as 25. During simulation, some nodes are randomly removed and added during simulation according to their absent probability. Figure 5.7 depicts the influence of churn rate on the success rate in three network structures. The x-axis is the absent portion of nodes in the social network, and the y-axis is the success rate. The experimental result shows the Scale-Free network (SN) suffers the most impact when nodes leaving the network; the RN has the medium success rate and Homophily-based network (HN) has the best tolerance to network churn. In SN, a few nodes have very large degrees of connection, and these nodes are considered highly influential participants that affect the overall network connections significantly. When these nodes are absent from the network, the success rate declines sharply. In HN, the reason for the best tolerance is that nodes are connected by homophily which can handle a great number of queries in a short path. Moreover, the neighbours in HN who share common interests and may provide similar services. Therefore, even when nodes are absent from the network, the HN shows favourable resistance to the churn. This feature is very desirable in real-world dynamic systems since users can turn off their physical device anytime and the services possessed by these devices will no longer be available. The homophily-based social network provides higher success rate in such dynamic environment that can guarantee a better search performance for service discovery.

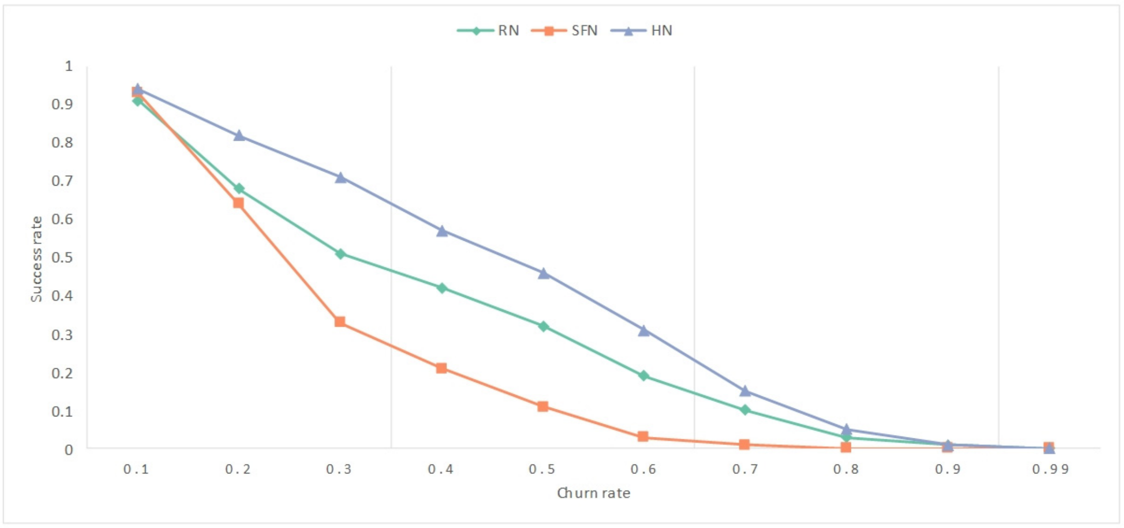


Figure 5.7 Churn rate influence on search performance

5.5.4.3 Comparison of Related Search Models

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

Random: a search strategy utilising random walks with a fixed forwarding degree [53, 77]

Degree: a search strategy utilising only the degree of connection information with a fixed forwarding degree [57, 148, 198].

FreeNet: a search model utilising similarity with deep first search with a fixed forwarding degree [91].

Neurogrid: a search model utilising similarity with broad first search with a fixed forwarding degree [92].

ESLP: a search model utilising similarity with broad first search with an adaptive forwarding degree [93].

EDSD: a search model utilising similarity integrated with degree search in greedy search mode [115].

IAPS: a search model utilising resource type score with a fixed forwarding degree [82].

5.5.4.4 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 [218]. Also in the experiments, with the TTL of 8, query messages can reach almost all the nodes in the 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 network shortcut index is 100. The soft-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 d in the experiment is evaluated from 1 to 5. The average forwarding degree of all the models in the simulations is approximated to 2 based on the experimental observations. It should be noted that the original model of FreeNet, EDSD forwards queries only to a single neighbour in each hop to reduce the query messages. In order to conduct a fair comparison of the forwarding degree, the forwarding degree of FreeNet and EDSD are altered to the same with other models. Furthermore, simulations include a churn rate to simulate dynamic decentralised systems, where approximately 20% of peer nodes are presented less than 30% of the time. Table 5.1 shows the default values of simulation parameters and the function description.

Table 5.1 Simulation parameters and their default values

Parameters	**Value**	**Function**
maxTTL	2~8	Maximum TTL of a query
m	1000	Number of nodes in the social network
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	Similarity threshold: FreeNet, Neurogrid, ESLP, EDSD, OSS
d	1~5	Adaptive forwarding degree range

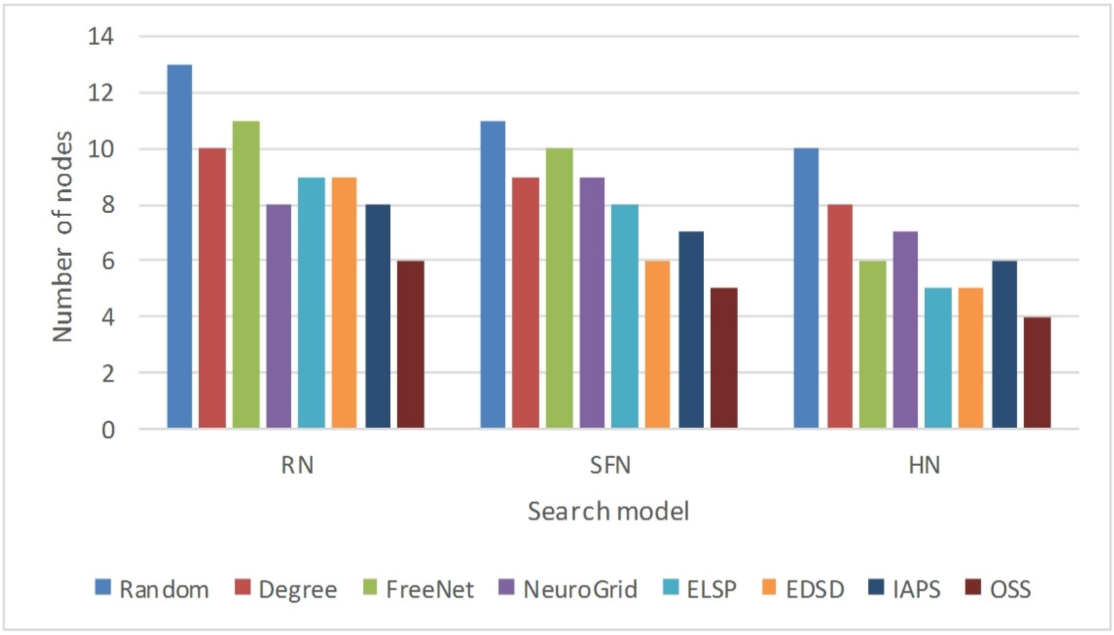


Figure 5.8 Number of visited nodes under different network structures

Figure 5.8 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. The Random Network structure (RN) and the Scale-Free Network structure (SFN) connects the nodes without considering the interests and semantic content of the nodes. Considering the differences between the content of nodes and their neighbours, the Homophily Network structure (HN) establishes user connections with similar interest and content. It can be observed from Figure 5.8 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 neighbours, thus the number of visited nodes to resolve queries are lower than RN and SFN. Furthermore, it is clearly evident that the 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 utilises 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.

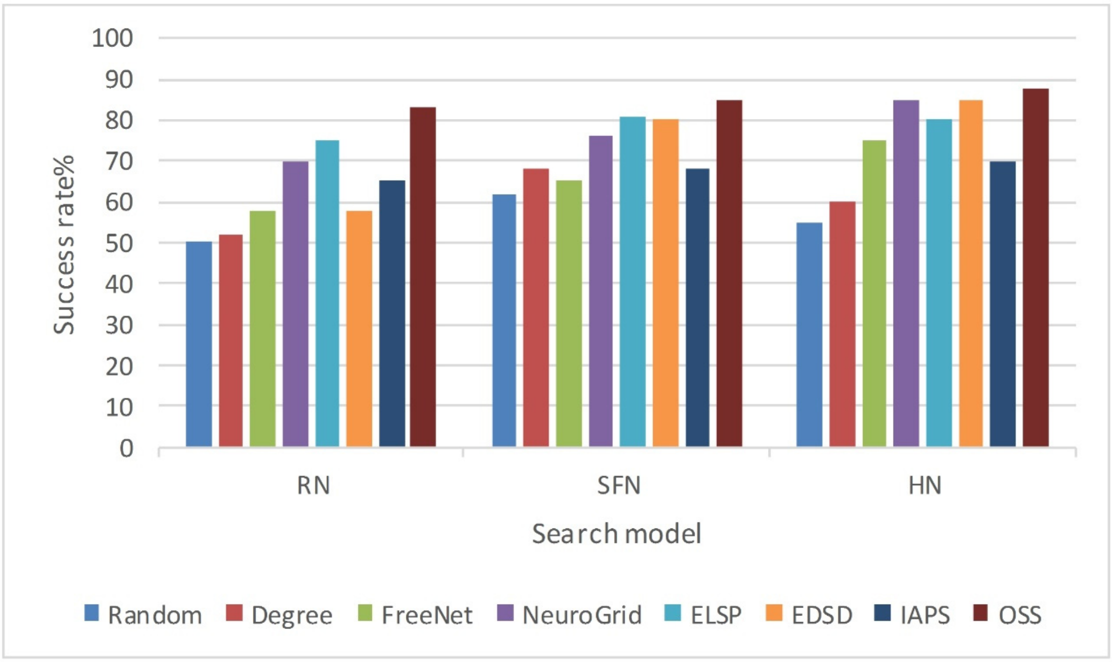


Figure 5.9 Success rates under different network structures

Figure 5.9 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, ESLP, 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, ESLP and OSS are exhibiting a higher average success rate than FreeNet which uses DFS based search. Comparing ESLP and OSS based on the adaptive forwarding degree, OSS has a more comprehensive knowledge index for gaining more social information about the neighbours including interests, social relationships. ESLP only has a topic information of immediate neighbours in its knowledge index, but the neighbour’s degree and the homophily are not considered in ESLP. During the service discovery process, OSS obtains homophily information in the social network and utilises a more accurate neighbour selection than ESLP, thus OSS exhibits a better success rate than ESLP. 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 neighbours 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 utilises semantic similarity but also considers neighbours' 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.

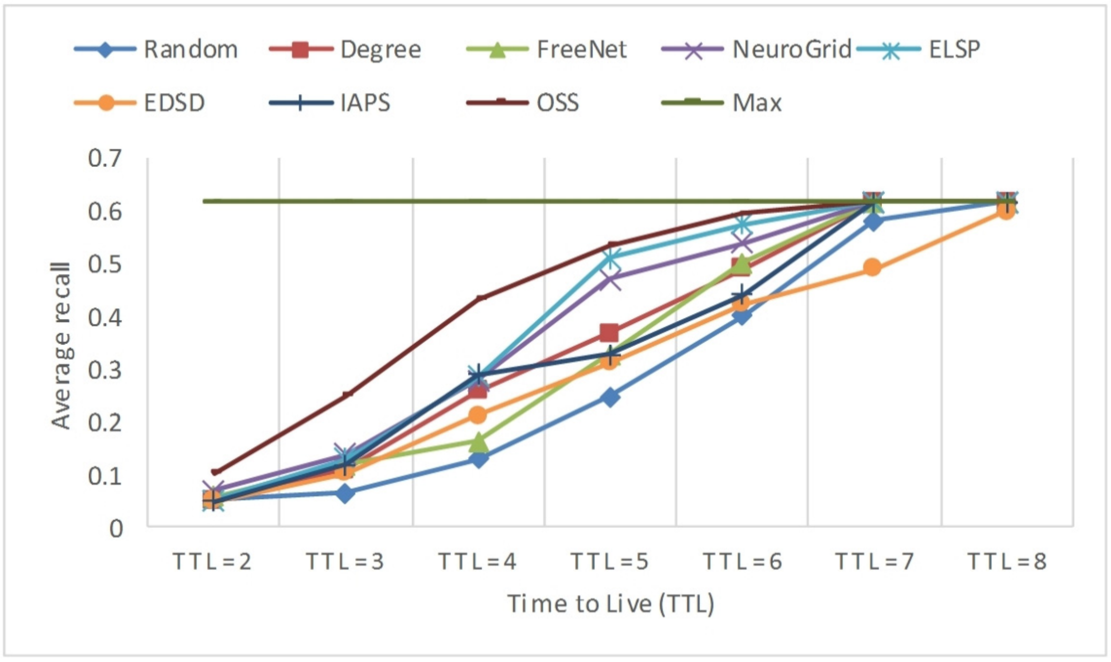


Figure 5.10 Average recalls on time-to-live (TTL)

Figure 5.10 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 decentralised environment. It can be observed that the TTL determines 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. The 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 can utilise 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.

5.6 Summary

This chapter introduced a novel swarm intelligence inspired search algorithm, which aims to optimise the service discovery process by finding more relevant services with less traffic overhead. This algorithm characterises the decentralised service discovery as the food foraging method of swarms in the wild. The biologic features pheromone and olfactory sensibility are studied and formally defined in this chapter. This algorithm performs a swarm-based query routing strategy that employs an adaptive forwarding degree in each hop. The forwarding degree of swarms is self-adaptive to the environmental changes and achieves favourable search performance in dynamic networks. The proposed OSS algorithm 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 this approach is the maintenance of a knowledge index in user's local storage. The knowledge index contains associations between a node and its immediate neighbours based on historical search results, which includes the neighbour's neighbour list, the neighbour's interests list and the pheromone table. On one hand, the neighbour list and interest list do not need frequent updates. In this research, when the social overlay network has been established, the neighbour 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 neighbour 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 the simulation, the size of pheromone table has been defined as 100 topic items. When the number of topic records reaches the maximum size, old records will be replaced with new records using the least recently used strategy. It can be observed from the experimental results that the proposed algorithm achieved more than 30% higher performance than uninformed search models by utilising a small-sized knowledge index. Besides, this algorithm has achieved 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, it can be concluded that with the significant improvement of service discovery performance, the maintenance overhead of the small-sized knowledge index in the proposed algorithm is reasonable and cost-effective.

6 CONCLUSION

6.1 Conclusion

This thesis presents a novel self-organised architecture and adaptive mechanisms to support efficient service discovery for decentralised OSNs. The research objectives have been achieved through five core components: literature review, architecture design, supporting algorithm development, model optimisation and simulation.

A comprehensive and systematic review of the state-of-the-art literature on P2P networks and decentralised OSNs has been presented in chapter 2. This includes deeply analysing the P2P network topology, knowledge-based service discovery, and the architecture design requirements of decentralised social systems. In addition, social network theories and user behaviours have been vastly studied including social knowledge maintenance, communication mechanisms and social relationship adaptation. This literature review discussed the key building blocks of decentralised OSNs and laid the foundation for this research towards two core components such as a self-organised architecture and an efficient service discovery model for decentralised OSNs.

Chapter 3 detailed the system model design of the proposed self-organised architecture. The content management structure, communication and distributed topology adaptation have been characterised and defined in order to develop self-sustaining mechanisms to maintain the network structure and to support social service discovery without the involvement of a dedicated central server. A social overlay network has been built on top of an unstructured P2P network which facilitates users to self-organise into semantic communities based on content similarity. This self-organised architecture exhibits the small-world characteristics with short average path length and large average clustering coefficient. Additionally, a distributed topology adaptation has been proposed to bootstrap newly joining nodes for the purpose of maintaining the network structure. A node fingerprint hash technique has been presented to achieve a fast bandwidth-efficient similarity calculation between users. The conducted experiments validate the effectiveness and correctness of the proposed architecture design under three different network structures including Random Network, Scale-Free Network and NFH-based Network respectively. The obtained results demonstrated that the proposed architecture can deliver correct and stable service discovery in decentralised environments.

Chapter 4 presented an efficient service discovery model that utilises homophily features in semantic query routing. The proposed model can efficiently handle complex queries and exhibits good performance under different network structures. In order to capture the hidden semantic correlation between concepts, a semantic matchmaking method that utilises the soft-cosine similarity has been employed to achieve an accurate semantic match for complex queries. The homophily feature of social networks has been studied and modelled, which integrates common social connections and common social interest to form communities in the social network. Moreover, the social network adaptation has been spontaneously conducted through user interactions. During this network adaptation, nodes can accumulate social connections and find more suitable candidates to connect as friends. In the homophily-based social network, the average discovery paths tend to be short and communities exhibit high clustering coefficients. Experimental results showed this proposed service discovery model can achieve high success rate and low overhead.

In order to optimise the query routing efficiency, chapter 5 introduced a novel swarm intelligence inspired search algorithm OSS, which is able to locate more relevant services with fewer messages. The decentralised service discovery has been achieved through a food foraging method of insect swarms in the wild. The biological features such as pheromone and olfactory sensibility of swarms have been adopted and characterised as indicators for the query routing process. The proposed algorithm employs an adaptive forwarding degree in each hop and it is self-adaptive to the environmental changes and achieves favourable search performance in dynamic networks. In the experiments, the OSS outperformed the compared search models in terms of the achieved search performance and reduced network overhead.

In order to evaluate the proposed model and algorithms, a configurable simulator has been developed using Java, as introduced in chapter 1. The main components of this simulator include adaptation to network structure, decentralised service discovery and evaluation metrics. This simulator can simulate a dynamic social network with knowledge-based routing protocols for service discovery. The efficiency and effectiveness of the proposed architecture and model have been evaluated against the state-of-the-art methods using real world datasets.

6.2 Reflections

OSNs are the most prevalent Internet applications that provide appealing online activities for people to share information and to communicate with the wider world. However, a series of urgent issues faced by existing centralised OSNs such as single-point of failure and privacy concerns, have motivated this research to investigate the possibility of developing new decentralised architectures with better scalability and user privacy control as an alternative to the traditional centralised architectures. The booming social network applications on mobile devices further urges the research on DOSNs. This thesis presented a novel service discovery model aimed at facilitating the development of next generation OSNs. The proposed model entails a self-organised architecture and effective decentralised service discovery algorithms to support the core functionality of DOSNs. The potential practical impacts of this research in the context of DOSNs can be concluded as follows.

First of all, like the Internet itself, the proposed self-organised architecture is not controlled by any sole entity. Users in this architecture have full ownership of their personal social data and can control their data storage and sharing policies. Additionally, promising properties such as decentralisation, self-organisation and fault tolerance provide the architecture with strong adaptability and high scalability. This allows the proposed architecture to serve as attractive infrastructure for large-scale social systems.

Next, the proposed service discovery approach accelerates query routing through effective social network adaptation. This approach achieves a semantic-aware and social-aware search, which encompasses a service matchmaking module to capture the hidden semantic information and a homophily-based query module to characterise user's common social status and interests. Important features such as supporting complex queries and fuzzy match, high query efficiency and dynamic social network adaptation, make the service discovery approach flexible and accurate in decentralised environments.

Finally, the swarm intelligence algorithm is self-adaptive to environmental changes and achieves favourable search performance in dynamic social networks. This algorithm optimises the query efficiency by finding more relevant services with less traffic overhead.

In decentralised social systems, this algorithm can provide users with more service candidates for each query and the request delays can be significantly reduced due to its less bandwidth cost requirement for query routing. Therefore, the user experience in such social system can be massively improved.

6.3 Future Directions and Insights

Although this research has addressed several issues of service discovery in DOSNs, there are still other open research problems to be considered into account in the future.

The first limitation of the proposed model is that the research only considers user interests and social relationships to characterise the homophily features. However, in real-life social networks, users are sophisticated and volatile. As a future work, other social features such as authority, centrality and time/location attributes will be considered for optimisation, which may further enrich the homophily definition. Additionally, user's interests and social relationships dynamically change over time. Investigating the impacts of these dynamic changes upon the overall performance will be another research direction. The volatility of homophily during social network evolution will be studied to capture more comprehensive and accurate social features. The second limitation is that the discovery may not be guaranteed in an open dynamic system. Unlike a closed simulation system, the topology structure of real-world DOSNs is ad-hoc in nature. With nodes joining and leaving the system frequently and contents being generated and updated quickly, service availability and resilience under high network churn are a few commonly prevailing issues in service discovery. In a high churn rate environment, the limitations of the proposed social overlay network may witness low service availability. In the current design, service availability depends on user's engagement of online presence in the social network. If a large number of users are absent, the success rate of discovery declines. In addition, when the number of direct neighbours is small, discovery performance will be further penalised as the network may be segmented. As part of future work, service replica schemes and incentives for user's online presence will be considered to improve the service availability. Additionally, nodes with a smaller number of connections require more dedicated service cache mechanisms and personalised service recommendation algorithms to achieve a reasonable search quality in dynamic environments. Furthermore, testing the performance of the proposed model in a real open system is also a future research direction.

REFERENCES

[1] Number of social media users worldwide from 2010 to 2020 [Online]. 2016. Available: https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/. [Accessed: 1 March 2017]

[2] P. Vagata and K. Wilfong. Scaling the Facebook data warehouse to 300 PB [Online]. 2014. Available: https://code.facebook.com/posts/229861827208629/scaling-the-facebook-data-warehouse-to-300-pb/. [Accessed: 14 Jan 2017]

[3] P. Mell and T. Grance, "The NIST definition of cloud computing," 2011.

[4] Google privacy policy [Online]. 2017. Available: https://www.google.com/policies/privacy/. [Accessed: 1 July 2017]

[5] J. Ratkiewicz, M. Conover, M. Meiss, B. Gonçalves, S. Patil, A. Flammini*, et al.*, "Truthy: mapping the spread of astroturf in microblog streams," in *Proceedings of the 20th international conference companion on World wide web*, 2011, pp. 249-252.

[6] F. Ciulla, D. Mocanu, A. Baronchelli, B. Gonçalves, N. Perra, and A. Vespignani, "Beating the news using social media: the case study of American Idol," *EPJ Data Science,* vol. 1, p. 8, 2012.

[7] D. Maynard, K. Bontcheva, and D. Rout, "Challenges in developing opinion mining tools for social media," *Proceedings of the@ NLP can u tag# usergeneratedcontent,* pp. 15-22, 2012.

[8] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *Security and Privacy, 2009 30th IEEE Symposium on*, 2009, pp. 173-187.

[9] G. Wondracek, T. Holz, E. Kirda, and C. Kruegel, "A practical attack to de-anonymize social network users," in *Security and Privacy (SP), 2010 IEEE Symposium on*, 2010, pp. 223-238.

[10] L. Backstrom, C. Dwork, and J. Kleinberg, "Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography," in *Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 181-190.

[11] Data Policy:How do we use this information [Online]. 2016. Available: https://www.facebook.com/about/privacy/your-info. [Accessed: 1 July 2017]

[12] K. HILL, "Facebook Privacy Policy Change Paves Way For Off-Facebook Advertising," *Forbers: Tech,* 2012.

[13] S. Jahid, S. Nilizadeh, P. Mittal, N. Borisov, and 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.

[14] D. Etherington. Large DDoS attacks cause outages at Twitter, Spotify, and other sites [Online]. 2016. Available: https://techcrunch.com/2016/10/21/many-sites-including-twitter-and-spotify-suffering-outage/. [Accessed: 1 July 2017]

[15] B. Krebs. DDoS on Dyn Impacts Twitter, Spotify, Reddit [Online]. 2016. Available: https://krebsonsecurity.com/2016/10/ddos-on-dyn-impacts-twitter-spotify-reddit. [Accessed: 1 July 2017]

[16] S. Kumar and K. M. Carley, "Understanding DDoS cyber-attacks using social media analytics," in *Intelligence and Security Informatics (ISI), 2016 IEEE Conference on*, 2016, pp. 231-236.

[17] E. V. BUSKIRK. Denial-of-Service Attack Knocks Twitter Offline [Online]. 2009. Available: https://www.wired.com/2009/08/twitter-apparently-down/. [Accessed: 2 July 2017]

[18] Z. Whittaker. Apple iCloud hack threat gets worse: Here's what we've learned [Online]. 2017. Available: http://www.zdnet.com/article/icloud-accounts-breach-gets-bigger-here-is-what-we-know/. [Accessed: 15 June 2017]

[19] S. Thielman. Yahoo hack: 1bn accounts compromised by biggest data breach in history [Online]. 2016. Available: https://www.theguardian.com/technology/2016/dec/14/yahoo-hack-security-of-one-billion-accounts-breached. [Accessed: 1 July 2017]

[20] S. Perez. 117 million LinkedIn emails and passwords from a 2012 hack just got posted online [Online]. 2016. Available: https://techcrunch.com/2016/05/18/117-million-linkedin-emails-and-passwords-from-a-2012-hack-just-got-posted-online/. [Accessed: 1 July 2017]

[21] S. Perez. Recently confirmed Myspace hack could be the largest yet [Online]. 2016. Available: https://techcrunch.com/2016/05/31/recently-confirmed-myspace-hack-could-be-the-largest-yet/. [Accessed: 1 June 2017]

[22] C. Shu. Passwords for 32M Twitter accounts may have been hacked and leaked [Online]. 2016. Available: https://techcrunch.com/2016/06/08/twitter-hack/. [Accessed: 1 July 2017]

[23] D. Guarini. Experts Say Facebook Leak Of 6 Million Users' Data Might Be Bigger Than We Thought [Online]. 2013. Available: http://www.huffingtonpost.com/2013/06/27/facebook-leak-data_n_3510100.html. [Accessed: 1 July 2017]

[24] W. Gordon. 5 Million Online Passwords Leaked, Check Yours Now [Online]. 2014. Available: http://lifehacker.com/5-million-gmail-passwords-leaked-check-yours-now-1632983265. [Accessed: 1 July 2017]

[25] S. Buchegger, Schi, D. Berg, L. H. Vu, and A. Datta, "PeerSoN: P2P social networking: early experiences and insights," in *In Proc Acm Workshop on Social Network Systems*, 2009, pp. 46-52.

[26] Z. Li and 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.

[27] L. A. Cutillo, R. Molva, and T. Strufe, "Safebook: A privacy-preserving online social network leveraging on real-life trust," *IEEE Communications Magazine,* vol. 47, 2009.

[28] S. Buchegger and 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.

[29] C.-m. A. Yeung, I. Liccardi, K. Lu, O. Seneviratne, and 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.

[30] S. Fanning and S. Parker. Napster [Online]. 1999. Available: https://us.napster.com/home. [Accessed: 4 June 2017]

[31] J. Frankel and T. Pepper. Gnutella Protocol Development [Online]. 2002. Available: http://rfc-gnutella.sourceforge.net/. [Accessed: 12 April 2017]

[32] B. DevGroup, "BitComet," ed.

[33] B. Hendrik. eMule [Online]. 2017. Available: http://www.emule.com/. [Accessed: 1 June 2017]

[34] J. Tallinn. Kazaa [Online]. 2006. Available: https://en.wikipedia.org/wiki/Kazaa. [Accessed: 1 June 2017]

[35] S. A. Baset and H. Schulzrinne, "An analysis of the skype peer-to-peer internet telephony protocol," *arXiv preprint cs/0412017,* 2004.

[36] J. A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, *et al.*, "TRIBLER: a social ‐ based peer ‐ to ‐ peer system," *Concurrency and computation: Practice and experience,* vol. 20, pp. 127-138, 2008.

[37] H. Schulze and K. Mochalski, "Internet Study 2008/2009," *Ipoque Report,* vol. 37, pp. 351-362, 2009.

[38] X. S. Shen, H. Yu, J. Buford, and M. Akon, *Handbook of peer-to-peer networking* vol. 34: Springer Science & Business Media, 2010.

[39] J. Buford, H. Yu, and E. K. Lua, *P2P networking and applications*: Morgan Kaufmann, 2009.

[40] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," *ACM SIGCOMM Computer Communication Review,* vol. 31, pp. 149-160, 2001.

[41] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, "Consistent hashing and random trees: Distributed caching protocols for relieving hot spots on the World Wide Web," in *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, 1997, pp. 654-663.

[42] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," in *IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing*, 2001, pp. 329-350.

[43] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, *A scalable content-addressable network* vol. 31: ACM, 2001.

[44] G. S. Manku, M. Bawa, and P. Raghavan, "Symphony: Distributed Hashing in a Small World," in *USENIX Symposium on Internet Technologies and Systems*, 2003, p. 10.

[45] D. Malkhi, M. Naor, and D. Ratajczak, "Viceroy: A scalable and dynamic emulation of the butterfly," in *Proceedings of the twenty-first annual symposium on Principles of distributed computing*, 2002, pp. 183-192.

[46] P. Maymounkov and D. Mazieres, "Kademlia: A peer-to-peer information system based on the xor metric," in *International Workshop on Peer-to-Peer Systems*, 2002, pp. 53-65.

[47] P. Reynolds and A. Vahdat, "Efficient peer-to-peer keyword searching," in *Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware*, 2003, pp. 21-40.

[48] M. Harren, J. Hellerstein, R. Huebsch, B. Loo, S. Shenker, and I. Stoica, "Complex queries in DHT-based peer-to-peer networks," *Peer-to-peer systems,* pp. 242-250, 2002.

[49] R. Mahajan, M. Castro, and A. Rowstron, "Controlling the cost of reliability in peer-to-peer overlays," *Peer-to-Peer Systems II,* pp. 21-32, 2003.

[50] J. Li, J. Stribling, T. M. Gil, R. Morris, and M. F. Kaashoek, "Comparing the Performance of Distributed Hash Tables Under Churn," in *Iptps*, 2004, pp. 87-99.

[51] Z. Xu, R. Min, and Y. Hu, "Reducing maintenance overhead in DHT based peer-to-peer algorithms," in *Peer-to-Peer Computing, 2003.(P2P 2003). Proceedings. Third International Conference on*, 2003, pp. 218-219.

[52] R. Rodrigues and C. Blake, "When multi-hop peer-to-peer lookup matters," in *International Workshop on Peer-to-Peer Systems*, 2004, pp. 112-122.

[53] 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.

[54] E. Bangeman, "Study: Bittorrent sees big growth, limewire still♯ 1 p2p app," *ars technica,* 2008.

[55] C. Gkantsidis, M. Mihail, and A. Saberi, "Random walks in peer-to-peer networks," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 2004.

[56] V. Kalogeraki, D. Gunopulos, and 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.

[57] B. Yang and H. Garcia-Molina, "Improving search in peer-to-peer networks," in *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, 2002, pp. 5-14.

[58] B. B. Yang and H. Garcia-Molina, "Designing a super-peer network," in *Data Engineering, 2003. Proceedings. 19th International Conference on*, 2003, pp. 49-60.

[59] C. Mastroianni, D. Talia, and O. Verta, "A super-peer model for resource discovery services in large-scale grids," *Future Generation Computer Systems,* vol. 21, pp. 1235-1248, 2005.

[60] A. Saini, "Super-peer architectures for distribtuted computing," ed: Tech. Rep, 2002.

[61] V. Cholvi, P. Felber, and E. Biersack, "Efficient search in unstructured peer‐to‐peer networks," *Transactions on Emerging Telecommunications Technologies,* vol. 15, pp. 535-548, 2004.

[62] Z. Zhuang, Y. Liu, L. Xiao, and L. M. Ni, "Hybrid periodical flooding in unstructured peer-to-peer networks," in *Parallel Processing, 2003. Proceedings. 2003 International Conference on*, 2003, pp. 171-178.

[63] C. Gkantsidis, M. Mihail, and A. Saberi, "Hybrid search schemes for unstructured peer-to-peer networks," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, 2005, pp. 1526-1537.

[64] Y. Li, X. Huang, F. Ma, and F. Zou, "Building efficient super-peer overlay network for DHT systems," in *GCC*, 2005, pp. 787-798.

[65] Y. Zhu, H. Wang, and Y. Hu, "A Super-Peer Based Lookup in Structured Peer-to-Peer Systems," in *ISCA PDCS*, 2003, pp. 465-470.

[66] L. Garces-Erice, E. W. Biersack, K. W. Ross, P. A. Felber, and G. Urvoy-Keller, "Hierarchical peer-to-peer systems," *Parallel Processing Letters*, vol. 13, pp. 643-657, 2003.

[67] S. Zoels, Z. Despotovic, and W. Kellerer, "Cost-based analysis of hierarchical DHT design," in *Peer-to-Peer Computing, 2006. P2P 2006. Sixth IEEE International Conference on*, 2006, pp. 233-239.

[68] N. Leibowitz, M. Ripeanu, and A. Wierzbicki, "Deconstructing the kazaa network," in *Internet Applications. WIAPP 2003. Proceedings. The Third IEEE Workshop on*, 2003, pp. 112-120.

[69] G. Kreitz and F. Niemela, "Spotify--large scale, low latency, P2P music-on-demand streaming," in *Peer-to-Peer Computing (P2P), 2010 IEEE Tenth International Conference on*, 2010, pp. 1-10.

[70] Q. H. Vu, M. Lupu, and B. C. Ooi, *Peer-to-peer computing: Principles and applications*: Springer Science & Business Media, 2009.

[71] D. Tsoumakos and N. Roussopoulos, "Analysis and comparison of P2P search methods," in *Proceedings of the 1st international conference on Scalable information systems*, 2006, p. 25.

[72] D. Tsoumakos and N. Roussopoulos, "A Comparison of Peer-to-Peer Search Methods," in *WebDB*, 2003, pp. 61-66.

[73] K. Aberer and M. Hauswirth, "An Overview of Peer-to-Peer Information Systems," in *WDAS*, 2002, pp. 171-188.

[74] H. T. Shen, Y. Shu, and B. Yu, "Efficient semantic-based content search in P2P network," *IEEE Transactions on Knowledge and Data Engineering,* vol. 16, pp. 813-826, 2004.

[75] F. M. Cuenca-Acuna, C. Peery, R. P. Martin, and T. D. Nguyen, "PlanetP: Infrastructure support for P2P information sharing," 2001.

[76] M. Bender, S. Michel, P. Triantafillou, G. Weikum, and C. Zimmer, "Minerva: Collaborative p2p search," in *Proceedings of the 31st international conference on Very large data bases*, 2005, pp. 1263-1266.

[77] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham, and 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.

[78] E. Ayorak and A. B. Bener, "Super peer web service discovery architecture," in *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, 2007, pp. 1360-1364.

[79] W. Nejdl, M. Wolpers, W. Siberski, C. Schmitz, M. Schlosser, I. Brunkhorst*, et al.*, "Super-peer-based routing and clustering strategies for RDF-based peer-to-peer networks," in *Proceedings of the 12th international conference on World Wide Web*, 2003, pp. 536-543.

[80] A. Crespo and H. Garcia-Molina, "Routing indices for peer-to-peer systems," in *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, 2002, pp. 23-32.

[81] D. Tsoumakos and 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.

[82] M. Shojafar, J. H. Abawajy, Z. Delkhah, A. Ahmadi, Z. Pooranian, and A. Abraham, "An efficient and distributed file search in unstructured peer-to-peer networks," *Peer-to-Peer Networking and Applications,* vol. 8, pp. 120-136, 2015.

[83] G. Salton, A. Wong, and C.-S. Yang, "A vector space model for automatic indexing," *Communications of the ACM,* vol. 18, pp. 613-620, 1975.

[84] M. Bawa, G. S. Manku, and P. Raghavan, "SETS: search enhanced by topic segmentation," in *Proceedings of the 26th annual international ACM SIGIR*

conference on Research and development in informaion retrieval, 2003, pp. 306-313.

[85] Y. Zhu and Y. Hu, "Enhancing search performance on Gnutella-like P2P systems," *IEEE Transactions on Parallel and Distributed Systems,* vol. 17, pp. 1482-1495, 2006.

[86] J. Huang, X. Li, and J. Wu, "A class-based search system in unstructured P2P networks," in *Advanced Information Networking and Applications, 2007. AINA'07. 21st International Conference on*, 2007, pp. 76-83.

[87] M. Yu, J. Wang, Q. Wang, X. Liu, Z. Zhao, and Y. Zhang, "Semantic-based query routing in P2P social networks," in *Wireless Communications, Networking and Information Security (WCNIS), 2010 IEEE International Conference on*, 2010, pp. 674-678.

[88] C. Tang, Z. Xu, and M. Mahalingam, "pSearch: Information retrieval in structured overlays," *ACM SIGCOMM Computer Communication Review,* vol. 33, pp. 89-94, 2003.

[89] G. Salton, E. A. Fox, and H. Wu, "Extended Boolean information retrieval," *Communications of the ACM,* vol. 26, pp. 1022-1036, 1983.

[90] R. Mihalcea, C. Corley, and C. Strapparava, "Corpus-based and knowledge-based measures of text semantic similarity," in *AAAI*, 2006, pp. 775-780.

[91] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A distributed anonymous information storage and retrieval system," in *Designing Privacy Enhancing Technologies*, 2001, pp. 46-66.

[92] S. Joseph, *NeuroGrid: Semantically Routing Queries in Peer-to-Peer Networks*: Springer Berlin Heidelberg, 2002.

[93] L. Liu, N. Antonopoulos, S. Mackin, J. Xu, and D. Russell, "Efficient resource discovery in self-organized unstructured peer-to-peer networks," *Concurrency & Computation Practice & Experience,* vol. 21, pp. 159–183, 2009.

[94] A. Crespo and H. Garcia-Molina, "Semantic Overlay Networks for P2P Systems," *Lecture Notes in Computer Science,* vol. 3601, pp. 1-13, 2002.

[95] K. Sripanidkulchai, B. Maggs, and H. Zhang, "Efficient content location using interest-based locality in peer-to-peer systems," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, 2003, pp. 2166-2176 vol.3.

[96] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American society for information science,* vol. 41, p. 391, 1990.

[97] J. L. Sachs, U. G. Mueller, T. P. Wilcox, and J. J. Bull, "The evolution of cooperation," *The Quarterly Review of Biology,* vol. 79, pp. 135-160, 2004.

[98] P. Brañas-Garza and M. Paz Espinosa, "Altruism with social roots: an emerging literature," *Desarrollo y Sociedad,* 2006.

[99] J. Travers and S. Milgram, "The small world problem," *Phychology Today,* vol. 1, pp. 61-67, 1967.

[100] R. Pastor-Satorras and A. Vespignani, "Epidemic spreading in scale-free networks," *Physical review letters,* vol. 86, p. 3200, 2001.

[101] M. McPherson, L. Smith-Lovin, and J. M. Cook, "Birds of a feather: Homophily in social networks," *Annual review of sociology,* pp. 415-444, 2001.

[102] M. S. Granovetter, "The strength of weak ties," *American journal of sociology,* vol. 78, pp. 1360-1380, 1973.

[103] D. J. Watts and S. H. Strogatz, "Collective dynamics of'small-world'networks," *nature,* vol. 393, p. 440, 1998.
[104] J. M. Kleinberg, "Navigation in a small world," *Nature,* vol. 406, p. 845, 2000.
[105] M. Li, W.-C. Lee, and A. Sivasubramaniam, "Semantic small world: An overlay network for peer-to-peer search," in *Network Protocols, 2004. ICNP 2004. Proceedings of the 12th IEEE International Conference on*, 2004, pp. 228-238.
[106] K. Y. Hui, J. C. Lui, and D. K. Yau, "Small world overlay P2P networks," in *Quality of Service, 2004. IWQOS 2004. Twelfth IEEE International Workshop on*, 2004, pp. 201-210.
[107] Y. Ren, C. Sha, W. Qian, A. Zhou, B. C. Ooi, and K.-L. Tan, "Explore the" small world phenomena" in pure P2P information sharing systems," in *Cluster Computing and the Grid, 2003. Proceedings. CCGrid 2003. 3rd IEEE/ACM International Symposium on*, 2003, pp. 232-239.
[108] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 2007, pp. 29-42.
[109] J. Scott, *Social network analysis: A Handbook*: Sage, 2017.
[110] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," *Knowledge and Information Systems,* vol. 42, pp. 181-213, 2015.
[111] S. A. Myers, A. Sharma, P. Gupta, and J. Lin, "Information network or social network?: the structure of the twitter follow graph," in *Proceedings of the 23rd International Conference on World Wide Web*, 2014, pp. 493-498.
[112] N. A. Christakis and J. H. Fowler, "The spread of obesity in a large social network over 32 years," *n engl j med,* vol. 2007, pp. 370-379, 2007.
[113] D. Centola, R. Willer, and M. Macy, "The emperor's dilemma: A computational model of self-enforcing norms," *American Journal of Sociology,* vol. 110, pp. 1009-1040, 2005.
[114] M. Yavaş and G. Yücel, "Impact of homophily on diffusion dynamics over social networks," *Social Science Computer Review,* vol. 32, pp. 354-372, 2014.
[115] E. D. Val, M. Rebollo, and V. Botti, "Enhancing decentralized service discovery in open service-oriented multi-agent systems," *Autonomous Agents and Multi-Agent Systems,* vol. 28, pp. 1-30, 2014.
[116] E. Gilbert and K. Karahalios, "Predicting tie strength with social media," in *Proceedings of the SIGCHI conference on human factors in computing systems*, 2009, pp. 211-220.
[117] E. Bakshy, I. Rosenn, C. Marlow, and 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.
[118] J.-P. Onnela, J. Saramäki, J. Hyvönen, G. Szabó, D. Lazer, K. Kaski, *et al.*, "Structure and tie strengths in mobile communication networks," *Proceedings of the national academy of sciences,* vol. 104, pp. 7332-7336, 2007.
[119] P. De Meo, E. Ferrara, G. Fiumara, and A. Provetti, "On Facebook, most ties are weak," *Communications of the ACM,* vol. 57, pp. 78-84, 2014.
[120] S. Fortunato, "Community detection in graphs," *Physics reports,* vol. 486, pp. 75-174, 2010.
[121] A. Lancichinetti and S. Fortunato, "Community detection algorithms: a comparative analysis," *Physical review E,* vol. 80, p. 056117, 2009.
[122] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *Journal of statistical mechanics: theory and experiment,* vol. 2008, p. P10008, 2008.

[123] J. Huang, H. Sun, J. Han, H. Deng, Y. Sun, and Y. Liu, "SHRINK: a structural clustering algorithm for detecting hierarchical communities in networks," in *Proceedings of the 19th ACM international conference on Information and knowledge management*, 2010, pp. 219-228.

[124] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on pattern analysis and machine intelligence,* vol. 22, pp. 888-905, 2000.

[125] X. Zhang, Y. Yin, M. Zhang, and B. Zhang, "Web service community discovery based on spectrum clustering," in *Computational Intelligence and Security, 2009. CIS'09. International Conference on*, 2009, pp. 187-191.

[126] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," *The Bell system technical journal,* vol. 49, pp. 291-307, 1970.

[127] R. Guimera and L. A. N. Amaral, "Functional cartography of complex metabolic networks," *nature,* vol. 433, p. 895, 2005.

[128] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *arXiv preprint physics/0506133,* 2005.

[129] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New Journal of Physics,* vol. 11, p. 033015, 2009.

[130] J. Hu, M. Li, W. Zheng, D. Wang, N. Ning, and H. Dong, "SmartBoa: Constructing p2p Overlay Network in the Heterogeneous Internet Using Irregular Routing Tables," in *IPTPS*, 2004, pp. 278-287.

[131] M. Waldvogel and R. Rinaldi, "Efficient topology-aware overlay network," *ACM SIGCOMM Computer Communication Review,* vol. 33, pp. 101-106, 2003.

[132] A. Malatras, "State-of-the-art survey on P2P overlay networks in pervasive computing environments," *Journal of Network and Computer Applications,* vol. 55, pp. 1-23, 2015.

[133] Z. Hu, "Research and implementation of peer-to-peer network topology based on balanced binary tree," *Wseas Transactions on Computers,* 2012.

[134] V. Carchiolo, M. Malgeri, G. Mangioni, and V. Nicosia, "Emerging structures of P2P networks induced by social relationships," *Computer Communications,* vol. 31, pp. 620-628, 2008.

[135] H. Rostami, J. Habibi, and E. Livani, "Semantic partitioning of peer-to-peer search space," *Computer Communications,* vol. 32, pp. 619-633, 2009.

[136] S. Lemmouchi, M. Haddad, and H. Kheddouci, "Robustness study of emerged communities from exchanges in peer-to-peer networks," *Computer Communications,* vol. 36, pp. 1145-1158, 2013.

[137] Y. Ma, Z. Tan, G. Chang, and X. Gao, "A P2P network topology optimized algorithm based on minimum maximum k-means principle," in *Hybrid Intelligent Systems, 2009. HIS'09. Ninth International Conference on*, 2009, pp. 396-399.

[138] S. Antaris, D. Stasi, M. Högqvist, G. Pallis, and M. Dikaiakos, "A Socio-Aware Decentralized Topology Construction Protocol," in *Hot Topics in Web Systems and Technologies (HotWeb), 2015 Third IEEE Workshop on*, 2015, pp. 91-96.

[139] S. Manfredi and E. Di Tucci, "A decentralised topology control to regulate global properties of complex networks," *The European Physical Journal B,* vol. 90, p. 62, 2017.

[140] L. Chang, C. Zhang, X. Zhang, and X. Chen, "Decentralised regulation of nonlinear multi-agent systems with directed network topologies," *International Journal of Control,* pp. 1-11, 2016.

[141] B. Yuan, L. Liu, and N. Antonopoulos, "Efficient service discovery in decentralized online social networks," *Future Generation Computer Systems,* 2017.

[142] B. F. Cooper, "An optimal overlay topology for routing peer-to-peer searches," in *Proceedings of the ACM/IFIP/USENIX 2005 International Conference on Middleware*, 2005, pp. 82-101.

[143] G. Chen, C. P. Low, and Z. Yang, "Enhancing search performance in unstructured P2P networks based on users' common interest," *IEEE Transactions on Parallel and Distributed Systems,* vol. 19, pp. 821-836, 2008.

[144] H. Jin, X. Ning, and H. Chen, "Efficient search for peer-to-peer information retrieval using semantic small world," in *Proceedings of the 15th international conference on World Wide Web*, 2006, pp. 1003-1004.

[145] B. Zeng and R. Wang, "An Efficient DHT Routing Protocol with Small-world Features for Structured P2P Network," *International Journal of Grid and Distributed Computing,* vol. 9, pp. 1-12, 2016.

[146] R. Akavipat, L.-S. Wu, and F. Menczer, "Small world peer networks in distributed Web search," in *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*, 2004, pp. 396-397.

[147] M. Ripeanu, I. Foster, and A. Iamnitchi, "Mapping the gnutella network: Properties of large-scale peer-to-peer systems and implications for system design," *arXiv preprint cs/0209028,* 2002.

[148] L. Liu, J. Xu, D. Russell, P. Townend, and D. Webster, "Efficient and scalable search on scale-free P2P networks," *Peer-to-Peer Networking and Applications,* vol. 2, pp. 98-108, 2009.

[149] P. Fraigniaud, P. Gauron, and M. Latapy, "Combining the use of clustering and scale-free nature of user exchanges into a simple and efficient p2p system," in *European Conference on Parallel Processing*, 2005, pp. 1163-1172.

[150] H. Guclu and M. Yuksel, "Limited scale-free overlay topologies for unstructured peer-to-peer networks," *IEEE Transactions on Parallel and Distributed Systems,* vol. 20, pp. 667-679, 2009.

[151] X. Lu, E. Bulut, and B. Szymanski, "Towards limited scale-free topology with dynamic peer participation," *Computer Networks,* vol. 106, pp. 109-121, 2016.

[152] C. Mitra, J. Kurths, and R. V. Donner, "Rewiring hierarchical scale-free networks: Influence on synchronizability and topology," *arXiv preprint arXiv:1707.04057,* 2017.

[153] E. Bulut and B. K. Szymanski, "Constructing limited scale-free topologies over peer-to-peer networks," *IEEE Transactions on Parallel and Distributed Systems,* vol. 25, pp. 919-928, 2014.

[154] R. Cohen, K. Erez, D. Ben-Avraham, and S. Havlin, "Resilience of the internet to random breakdowns," *Physical review letters,* vol. 85, p. 4626, 2000.

[155] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," in *proceedings of Multimedia Computing and Networking*, 2002, p. 152.

[156] S. Marti, P. Ganesan, and H. Garcia-Molina, "SPROUT: P2P Routing with Social Networks," in *EDBT Workshops*, 2004, pp. 425-435.

[157] N. Kourtellis, J. Finnis, P. Anderson, J. Blackburn, C. Borcea, and A. Iamnitchi, "Prometheus: User-controlled p2p social data management for socially-aware

applications," in *Proceedings of the ACM/IFIP/USENIX 11th International Conference on Middleware*, 2010, pp. 212-231.

[158] A. Iamnitchi, M. Ripeanu, E. Santos-Neto, and I. Foster, "The small world of file sharing," *IEEE Transactions on Parallel and Distributed systems,* vol. 22, pp. 1120-1134, 2011.

[159] Z. Anwar, W. Yurcik, V. Pandey, A. Shankar, I. Gupta, and R. H. Campbell, "Leveraging social-network infrastructure to improve peer-to-peer overlay performance: Results from orkut," *Arxiv preprint cs/0509095,* 2005.

[160] Y.-h. Chu, S. G. Rao, S. Seshan, and H. Zhang, "A case for end system multicast," *IEEE Journal on selected areas in communications,* vol. 20, pp. 1456-1471, 2002.

[161] S. Banerjee, B. Bhattacharjee, and C. Kommareddy, *Scalable application layer multicast* vol. 32: ACM, 2002.

[162] B. Cohen. BitTorrent [Online]. 2001. Available: http://www.bittorrent.com/. [Accessed: 17 June 2017]

[163] S. J. Yang, J. Zhang, L. Lin, and J. J. Tsai, "Improving peer-to-peer search performance through intelligent social search," *Expert Systems with Applications,* vol. 36, pp. 10312-10324, 2009.

[164] H. Shen, Z. Li, and K. Chen, "Social-P2P: an online social network based P2P file sharing system," *IEEE Transactions on Parallel and Distributed Systems,* vol. 26, pp. 2874-2889, 2015.

[165] R. Farahbakhsh, N. Crespi, Á. Cuevas, S. Adhikari, M. Mani, and T. Sanguankotchakorn, "socP2P: P2P content discovery enhancement by considering social networks characteristics," in *Computers and Communications (ISCC), 2012 IEEE Symposium on*, 2012, pp. 000530-000533.

[166] C. Huang, J. Li, and K. W. Ross, "Can internet video-on-demand be profitable?," *ACM SIGCOMM Computer Communication Review,* vol. 37, pp. 133-144, 2007.

[167] J. Blackburn, N. Kourtellis, and A. Iamnitchi, "Vulnerability in socially-informed peer-to-peer systems," in *Proceedings of the 4th Workshop on Social Network Systems*, 2011, p. 7.

[168] L. Liu, N. Antonopoulos, and S. Mackin, "Social peer-to-peer for resource discovery," in *Parallel, Distributed and Network-Based Processing, 2007. PDP'07. 15th EUROMICRO International Conference on*, 2007, pp. 459-466.

[169] R. Haw, C. S. Hong, and C.-H. Kang, "A social P2P networking based on interesting keywords," in *Information Networking (ICOIN), 2011 International Conference on*, 2011, pp. 509-512.

[170] P. H. Ha, P. Tsigas, and O. J. Anshus, "Socionet: A social-based multimedia access system for unstructured p2p networks," *IEEE Transactions on parallel and distributed systems,* vol. 21, pp. 1027-1041, 2010.

[171] K. Chen, H. Shen, K. Sapra, and G. Liu, "A social network based reputation system for cooperative P2P file sharing," *IEEE Transactions on Parallel and Distributed Systems,* vol. 26, pp. 2140-2153, 2015.

[172] A. Shakimov, H. Lim, R. Cáceres, L. P. Cox, K. Li, D. Liu, *et al.*, "Vis-a-vis: Privacy-preserving online social networking via virtual individual servers," in *Communication Systems and Networks (COMSNETS), 2011 Third International Conference on*, 2011, pp. 1-10.

[173] L. M. Aiello and G. Ruffo, "LotusNet: Tunable privacy for distributed online social network services," *Computer Communications,* vol. 35, pp. 75-88, 2012.

[174] K. Graffi, C. Gross, D. Stingl, D. Hartung, A. Kovacevic, and R. Steinmetz, "LifeSocial. KOM: A secure and P2P-based solution for online social networks,"

in *Consumer Communications and Networking Conference (CCNC), 2011 IEEE*, 2011, pp. 554-558.

[175] I. Zhitomirskiy, D. Grippi, M. Salzberg, and R. Sofaer. A privacy-aware, distributed, open source social network [Online]. 2010. Available: https://diasporafoundation.org/. [Accessed: May 23 2017]

[176] D. Koll, "Towards a Robust and Secure Decentralized Online Social Network," 2015.

[177] S. Fu, L. He, X. Liao, C. Huang, K. Li, and C. Chang, "Analyzing and Boosting the Data Availability in Decentralized Online Social Networks," *International Journal of Web Services Research (IJWSR),* vol. 12, pp. 47-72, 2015.

[178] N. Charjan and S. Bohra, "Review of Access Control in Decentralized Online Social Network," *International Journal,* vol. 4, 2016.

[179] C. Adams and S. Lloyd, *Understanding PKI: concepts, standards, and deployment considerations*: Addison-Wesley Professional, 2003.

[180] J. Robinson. Statistics of diaspora [Online]. 2016. Available: https://the-federation.info/. [Accessed: 20 June 2017]

[181] A. Rao. EnThinnai - A Novel Approach to Social Networking [Online]. 2010. Available: https://www.enthinnai.com/. [Accessed: 1 June 2017]

[182] E. Moglen, B. Garbee, and Y. Benkler. Freedombox Enabling private conversations online [Online]. Available: http://freedomboxfoundation.org/. [Accessed: 16 March 2017]

[183] M. Mani, A.-M. Nguyen, and N. Crespi, "SCOPE: A prototype for spontaneous P2P social networking," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2010 8th IEEE International Conference on*, 2010, pp. 220-225.

[184] D. N. Kalofonos, Z. Antoniou, F. D. Reynolds, M. Van-Kleek, J. Strauss, and P. Wisner, "Mynet: A platform for secure p2p personal and social networking services," in *Pervasive Computing and Communications, 2008. PerCom 2008. Sixth Annual IEEE International Conference on*, 2008, pp. 135-146.

[185] magna. (2 July 2011). *magna*. Available: https://gitorious.org/magna

[186] R. Baden, A. Bender, N. Spring, B. Bhattacharjee, and D. Starin, "Persona: an online social network with user-defined privacy," in *ACM SIGCOMM Computer Communication Review*, 2009, pp. 135-146.

[187] W. Galuba, "Friend-to-Friend computing: Building the social web at the internet edges," 2008.

[188] F. Benevenuto, T. Rodrigues, M. Cha, and V. Almeida, "Characterizing user navigation and interactions in online social networks," *Information Sciences,* vol. 195, pp. 1-24, 2012.

[189] K. Graffi, P. Mukherjee, B. Menges, D. Hartung, A. Kovacevic, and R. Steinmetz, "Practical security in p2p-based social networks," in *2009 IEEE 34th Conference on Local Computer Networks*, 2009, pp. 269-272.

[190] G. Mega, A. Montresor, and G. P. Picco, "Efficient dissemination in decentralized social networks," in *Peer-to-Peer Computing (P2P), 2011 IEEE International Conference on*, 2011, pp. 338-347.

[191] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, 2002, pp. 380-388.

[192] J.-C. Huang, X.-Q. Li, and J. Wu, "A semantic searching scheme in heterogeneous unstructured P2P networks," *Journal of Computer Science and Technology,* vol. 26, pp. 925-941, 2011.

[193] G. Ramachandran and K. Selvakumar, "Dynamic Caching for Semantic Oriented Adaptive Search in Unstructured P2P Networks," *Asian Journal of Information Technology,* vol. 13, pp. 138-149, 2014.

[194] Y. Saad and M. H. Schultz, "Topological properties of hypercubes," *IEEE Transactions on computers,* vol. 37, pp. 867-872, 1988.

[195] https://www.dmoz.org/, "DMOZ - The Directory of the Web," 2016.

[196] A. Zubiaga, R. Martínez, and 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.

[197] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *science,* vol. 286, pp. 509-512, 1999.

[198] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman, "Search in power-law networks," *Physical review E,* vol. 64, p. 046135, 2001.

[199] H. Tajfel, *Human groups and social categories: Studies in social psychology*: CUP Archive, 1981.

[200] A. Broder and M. Mitzenmacher, "Network applications of bloom filters: A survey," *Internet mathematics,* vol. 1, pp. 485-509, 2004.

[201] M. Steinbach, G. Karypis, and V. Kumar, "A comparison of document clustering techniques," in *KDD workshop on text mining*, 2000, pp. 525-526.

[202] G. Sidorov, A. Gelbukh, H. Gómez-Adorno, and D. Pinto, "Soft similarity and soft cosine measure: Similarity of features in vector space model," *Computación y Sistemas,* vol. 18, pp. 491-504, 2014.

[203] T. Pedersen, S. Patwardhan, and J. Michelizzi, "WordNet:: Similarity: measuring the relatedness of concepts," in *Demonstration papers at HLT-NAACL 2004*, 2004, pp. 38-41.

[204] Y. Goldberg and O. Levy, "word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method," *arXiv preprint arXiv:1402.3722,* 2014.

[205] H. Shima, "Wordnet similarity for java (ws4j)," *HYPERLINK" https: llcode. google. com/p/ws4jl" https: llcode. google. comlp/ws4j,* 2016.

[206] D. J. Watts, P. S. Dodds, and M. E. Newman, "Identity and search in social networks," *science,* vol. 296, pp. 1302-1305, 2002.

[207] D. B. Kandel, "Homophily, selection, and socialization in adolescent friendships," *American journal of Sociology,* pp. 427-436, 1978.

[208] M. Ruef, H. E. Aldrich, and N. M. Carter, "The structure of founding teams: Homophily, strong ties, and isolation among US entrepreneurs," *American sociological review,* pp. 195-222, 2003.

[209] S. Aral, L. Muchnik, and A. Sundararajan, "Distinguishing influence-based contagion from homophily-driven diffusion in dynamic networks," *Proceedings of the National Academy of Sciences,* vol. 106, pp. 21544-21549, 2009.

[210] A. Jain, A. Jain, N. Chauhan, V. Singh, and N. Thakur, "Information Retrieval using Cosine and Jaccard Similarity Measures in Vector Space Model," *International Journal of Computer Applications,* vol. 164, 2017.

[211] R. Řehůřek and P. Sojka, "Gensim—Statistical Semantics in Python," 2011.

[212] B. Shah and K.-I. Kim, "Towards enhanced searching architecture for unstructured peer-to-peer over mobile ad hoc networks," *Wireless personal communications,* vol. 77, pp. 1167-1189, 2014.

[213] I. Memon, K. Domenic, H. Memon, R. Akhtar, W. Yong, and F. Zhang, "Rumor riding: An anonymity approach for decentralized peer to peer systems," *Wireless personal communications,* vol. 79, pp. 647-660, 2014.

[214] D. Das, A. Chatterjee, N. Pal, A. Mukherjee, and M. K. Naskar, "A degree-first greedy search algorithm for the evaluation of structural controllability of real world directed complex networks," *Network Protocols and Algorithms,* vol. 6, pp. 1-18, 2014.

[215] S. Joseph, "P2P metadata search layers," in *International Workshop on Agents and P2P Computing*, 2003, pp. 101-112.

[216] E. O. Wilson and M. Pavan, "Glandular sources and specificity of some chemical releasers of social behavior in dolichoderine ants," *Psyche,* vol. 66, pp. 70-76, 1959.

[217] P. Fu, S. Liu, H. Yang, and L. Gu, "Matching algorithm of web services based on semantic distance," in *Proceedings of 2009 International Workshop on Information Security and Application*, 2009.

[218] S. Jiang, L. Guo, X. Zhang, and H. Wang, "Lightflood: Minimizing redundant messages and maximizing scope of peer-to-peer search," *IEEE Transactions on Parallel and Distributed Systems,* vol. 19, pp. 601-614, 2008.