國立臺灣大學電機資訊學院資訊工程學系

碩士論文

Graduate Institute of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

基於同儕式網路之異質社群網路整合研究

P2P-iSN: A Peer-to-Peer Network for Integration of Heterogeneous Social Networks

鍾百俊

Pai-Chun Chung

指導教授：林風 博士

Advisor: Phone Lin, Ph.D.

中華民國一百零一年七月

July, 2012

Acknowledgement

兩年的碩士生涯轉眼間就過去了，又到了畢業的時候。首先我要感謝我的指導老師，林風教授。在兩年的碩士生涯中，林風老師扮演著亦師亦友的角色，讓我學到做研究的熱情以及態度，並且順利完成了這篇論文。接著要特別感謝顧磊博士，在數學分析上提供我許多寶貴的意見以及相關的參考資料，讓我能夠順利完成。同時，我 也非常感謝林一平教授、方玉光教授、周勝鄰博士以及黃天立博士擔任學生的口試委員，並且對本篇論文提供許多良好的建議，使得這篇論文更加完整。

接著我要感謝與我在這兩年一起奮鬥的實驗室同仁們。懷磊學長、啓維學長、家朋學長、有倫學長、亭佑學姊、厚鈞學長、思適學長、家綸學長、 宗哲學長、坤豐、冠銘、振翔、明峰、彥婷、恩豪，與你們相處的點滴永難忘懷，在你們的陪伴下，讓我十分快樂的度過這段研究的時光。

最後我要感謝我的家人，我的父母與姊姊，在這兩年中給我全力的支持與鼓勵，讓我可以全心全意的認眞讀書，順利地拿到碩士學位。

Chinese Abstract

社群網路快速的發展吸引了學者研究和分析大量的社群網路資料，而異質社群網路也啓發學者發展出新的應用程式來整合不同社群網路資源，以提供更多社群網路服務。 本篇研究著重於社群關係整合上，我們使用同儕式網路架構來整合不同的社群網路關係，稱作(P2P-iSN)。在P2P-iSN上透過Global Relationship Model計算出使用者彼此在社群網路上的關係強度，並且根據關係強度我們提出(i-Search)機制尋找出一條有意義的社群關係路徑來連結任兩個使用者。 這些特點可用來開發出許多應用程式如信任模組以及社群資源分享，使我們更加了解使用者彼此在異質網路上的社群網路關係，並且能夠設計出更多以人爲本的應用程式。

關鍵字(Keywords): 全域社群關係(Global social relationship)、異質社群網路(Heterogeneous social network)、同儕式網路(Peer-to-peer network)、關係強度(Relationship strength)

English Abstract

The unprecedented growth and influence of Social Network Sites (SNSs) have opened the possibility for researchers to explore an abundance of social and behavioral data. A landscape of heterogeneous SNSs further sparks research innovations to develop methods and applications that integrate resources and offer more seamless services across SNSs. Specifically aiming at the integration of social relations data, a much less studied subject, we propose a set of tools to aggregate social relations across multiple SNSs (P2P-iSN), calculate relationship strength between users (Global Relationship Model), and offer a social path that indicates how any two users are meaningfully connected in heterogeneous SNSs (i-Search mechanism). These key features allow for many future application developments, such as improved trust/prestige metrics and integrated content-sharing. With these tools that enhance our understanding of social relations in this heterogeneous SNS landscape, SNS developers can also design more user-centric applications and future SNS components.

Keywords: Global social relationship; Heterogeneous social network; Peer-to-peer network; Relationship strength;

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Hundreds of Social Network Sites (SNSs) have gathered users together and changed how users interact with one another. Although SNSs offer different services, one key feature shared among SNSs is how they are built around users and users pre-existing social networks [3, 11]. Furthermore, this is a landscape of heterogeneous SNSs, where a user carries multiple SNS accounts, interacts with contacts from different social networks, publishes and accesses different web content, and shares the content within each SNS community. With growing influence and complexity of SNSs, researchers are proposing methods to connect users and aggregate data across SNSs so that each SNS no longer stands alone. For example, the study [10] summarized how social network connect services allow users to leverage their information in multiple SNSs, from using single id to access multiple SNS accounts to publishing content simultaneously on multiple SNSs. However, the aggregation of social relations data has been much less studied, and this is the main purpose of our paper. We propose a model to define a large-scale and aggregated

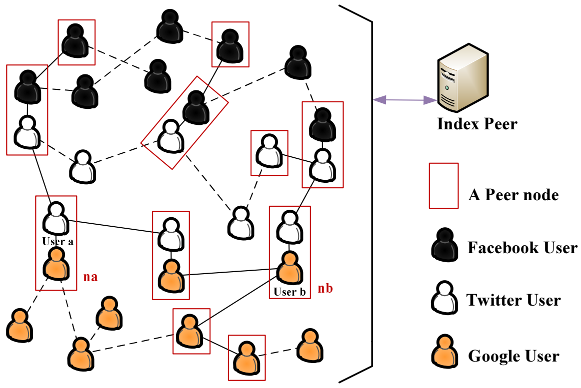


Figure 1.1: System Architecture of P2P-iSN

set of users social relations across heterogeneous SNSs.

First of all, within an SNS, if user b is in user a's friend list, we define that there is a directional social link denoted by "$a \rightarrow b$" between user a and user b. Building along these directional links, users in SNS form a social graph [1]. When there exists a social path between two users in an SNS, we define that there is a "relationship" between the two users. Secondly, we define "global relationship" as the social path between two users across different SNSs. These are basic notations that are applied and elaborated in our model. By identifying "global relationship" among users over heterogeneous SNSs, this paper aims to open the possibility for users from different SNSs to interlink their various networks and communicate with a larger audience more openly. A better evaluation of this heterogeneous SNS landscape can also help SNS developers design user-centric applications and design future SNS components [11].

In this paper, we first propose a peer-to-peer (P2P) network, namely P2P-iSN, to integrate heterogeneous SNSs as shown in Figure 1.1. P2P-iSN consists of two kinds of nodes: Peer node and Index Peer node. A Peer node is installed on an end-device (e.g., PDA or desktop) for the user to access SNSs, and its main functionality is to integrate heterogeneous SNSs. The user of a Peer node may register to one or more SNSs on his end-device, and login to one or more SNSs at the same time. To associate these different accounts of the same user from heterogeneous SNSs, a unique user ID is required. The concept is known as the OpenID[1] concept in [6]. A unique user ID can be some kind of ID information like user's cell phone number or email address. When the Peer node is turned on, it reports the online status, which includes the ID and IP address of the Peer node, to the Index Peer node. Upon receiving the online status, the Index Peer node updates the online status for the Peer node. If a user a of the Peer node **na** and a user b of the Peer node **nb** are on each other's friend list in a SNS, and **na** and **nb** are turned on, these two online Peer nodes can communicate with each other by using the corresponding IP addresses queried from the Index Peer node. The Peer nodes establish social paths among users from different SNSs and build our so-defined "global relationship".

With the peer-to-peer network architecture, P2P-iSN allows users from heterogeneous SNSs to communicate without involving the SNS, and the integration is independent from a specific SNS. In other words, the integration does not incur overhead to the SNSs. Then applying P2P-iSN, we propose a Global Relationship Model to identify global relationship strength between two users from heterogeneous SNSs. Based on the Global Relationship Model, we propose a searching mechanism, namely i-Search, to find the social

[1]OpenID is a protocol that authenticates a user's digital identity. A user could register on any one of the Identity Providers, which are websites that handle user authentication, including Facebook^{TM}, Google^{TM} and MySpace^{TM}. Once the register describes the identity of the user, the user, carrying the same ID, can browse all websites that support OpenID.

path between two users from heterogeneous SNSs. An analytical model is proposed to approximate the performance of the i-Search mechanism in terms of the "path found" probability with details to be elaborated later. We also conduct simulation experiments to validate the analysis results.

The rest of the paper is organized as follows: Chapter 2 describes the implementation of P2P-iSN. In Chapter 3, we detail the three functions in the PeerAgent class. In Chapter 4, we describe the analytical model. Chapter 5 is the performance evaluation. We conclude our work in Chapter 6. Some snapshots of P2P-iSN can be found in Appendix A.

Chapter 2

Implementation of P2P-iSN

P2P-iSN consists of two kinds of nodes: the Peer node and the Index Peer node. The main functionality of the Peer node is to integrate the heterogeneous SNSs through the Friend List maintenance (to be elaborated later). The Peer nodes communicate with each other directly and form a peer-to-peer network. The Index Peer node maintains the status and the IP address of the Peer node. The design and implementation for the two kinds of nodes are elaborated in the following subsections.

2.1 The Peer Node

The Peer node is installed on an end device (e.g., PDA or desktop) used by the user to access the SNS. A user may register to one or more SNSs on his end device, and login to one or more SNSs at the same time. Because a user may use different IDs in different SNSs, to associate these different accounts of a user, a unique user ID is required. The

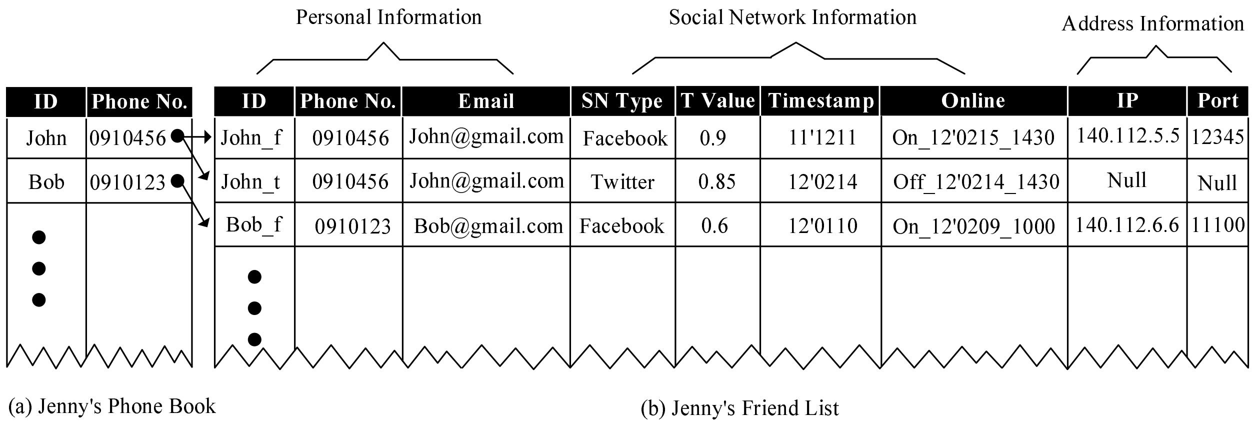


Figure 2.1: An Example of Friend List

concept is known as the OpenID concept in [6]. The unique user ID can be a user's cell phone number or email address. In this paper, we use the cell phone number as an example for the unique ID.

The phone book in a Peer Node (e.g., Jenny's end device) are used as the base to integrate the heterogeneous SNSs. Take (1) in Figure 2.1 (a) for example. Jenny has a friend John with phone number "0910456".

We maintain a database, Friend List, to store the information about the user's friends. Figure 2.1 (b) shows the format of a Friend List. The Friend List consists of three kinds of information: Personal Information, Social Network Information, and Address Information.

Personal Information stores the IDs of the user's friends, including the ID in SNS, phone number, and email address. In different SNSs, users may use different IDs. For example, Jenny's friend, John, uses the ID, "John_f", on Facebook^{TM} (see (1) in Figure 2.1 (b)) and use the ID, "John_t", on Twitter^{TM} (see (2) in Figure 2.1

(b)). The phone number associates the entry in the phone book with the entry in the Friend List. An entry in the phone book may be mapped to multiple entries in the Friend List.

The Social Network Information consists of four fields, including SN Type, T Value, Timestamp, and Online. The SN Type indicates which SNS the friend has registered. For example, in (1) in Figure 2.1 (b), Jenny's friend, John, registered to Facebook^{TM} using ID "John_f". The T Value stores the "trust value" which represents how much Jenny "trusts" John. It can be manually set by Jenny or calculated based on the interaction between Jenny and John on the SNS. We detail it an the next chapter. For example, in (1) in Figure 2.1 (b), the T Value for Jenny $\leftarrow$ John on Facebook^{TM} is 0.9. The Timestamp field stores the time when the T Value was calculated. The Online indicates that whether the friend is on the SNS now or not and when John logins to the Facebook^{TM} last time. If the value of Online is "On" ("Off"), the time is when John logins (logouts) Facebook^{TM}. For example, in (1) in Figure 2.1 (b), "On_12'0215_1430" implies that John_f logins Facebook^{TM} at 14:00 on Feb. 15, 2012, and is now on Facebook^{TM}.

The Address Information stores the IP address and the port number of the friend's end-device. This information is valid when the Peer node of the friend is turned on.

Figure 2.2 (1) shows the software architecture of the Peer node. The Peer node consists of five classes and a function, **PeerAgent** (see Figure 2.2 (1.5)), **FeedRequestListener** (see Figure 2.2 (1.1)), **SampleAuthListener** (see Figure 2.2 (1.2)), **CreateFriendListListener** (see Figure 2.2 (1.3)), **BackgroundService** (see Figure 2.2 (1.10)), and a Phone

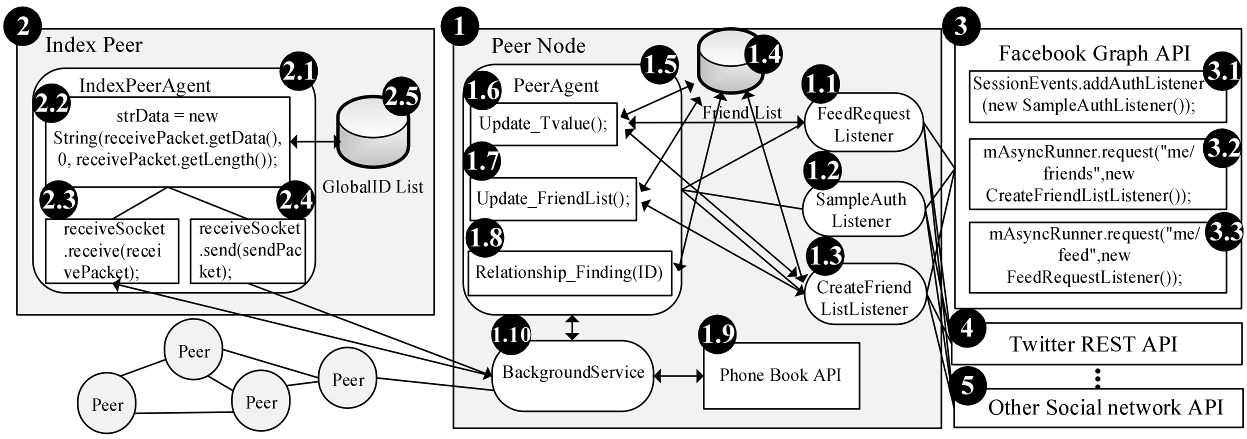


Figure 2.2: The Software Architecture of P2P-iSN

Book API (see Figure 2.2 (1.9)). The details of the implementation for the five classes are given below:

- The **FeedRequestListener** class (see Figure 2.2 (1.1)) is responsible to get the status of the user's social activities on SNS by invoking the API, mAsyncRunner.request("me/feed", new FeedRequestListener()) (see Figure 2.2 (3.3)), provided by the SNS (e.g., Facebook Graph API [7]).

- The **SampleAuthListener** class (see Figure 2.2 (1.2)) is responsible to authenticate a user when he turns on the Peer node and login an SNS. The **SampleAuthListener** class is implemented by using the API SessionEvents.addAuthListener(new SampleAuthListener()) (see Figure 2.2 (3.1)) provided by the SNS.

- The **CreateFriendListener** class (see Figure 2.2 (1.3)) is responsible to get the IDs of the user's friends in an SNS by invoking the API, mAsyncRunner.request("me/friends", new CreateFriendListListener()) (see Figure 2.2 (3.2)), and maintain the user's Friend List.

- The **BackgroundService** class (see Figure 2.2 (1.10)) is responsible for the message exchange between two Peer nodes and between the Peer node and the Index Peer node. The class provides the communication channel among Peer nodes for the i-Search mechanism. To be more specific, a Peer node uses this class to request another Peer node to execute the iSearch algorithm. The Peer node uses this class to inform his online status to the Index peer.

- The **PeerAgent** is the main class (see Figure 2.2 (1.5)). There are three functions defined in **PeerAgent** including the Update_Tvalue() function (see Figure 2.2 (1.6)), the Update_FriendList() function (see Figure 2.2 (1.7)), and the Relationship_Finding() function (see Figure 2.2 (1.8)). The Update_Tvalue() function and the Update_FriendList() function are used to respectively update the T Value and Online field in the Friend List. The Relationship_Finding() function implements the iSearch algorithm. We detail the three function in the next chapter.

- The Phone Book API is used to fetch the user's phone book friends and is provided by Android API [5]. It is executed in the Login procedure and will be elaborated later. By using the phone number, we can identify two or more accounts of the same user to integrate the different SNSs.

2.2 The Index Peer Node

The Index Peer node is a database that maintains the GlobalID List with the format as shown in Figure 2.1 (b). For each online Peer node, an entry is created in the GlobalID

ID	Phone No.	Email	IP	Port	SN Type
John_f	0910123	John@gmail.com	140.112.5.5	12345	Facebook
Bob_f	0910456	Bob@gmail.com	140.112.6.6	11100	Facebook
Bob_t	0910456	Bob@gmail.com	140.112.6.6	11100	Twitter
Jenny_f	0910789	Jenny@gmail.com	140.112.7.7	16161	Facebook
Jenny_t	0910789	Jenny@gmail.com	140.112.7.7	16161	Twitter
⋮					

Global ID List

Figure 2.3: An Example of GlobalID List

List for the Peer node. Similar to the Friend List, the GlobalID List consists of three kinds of information: Personal Information, Social Network Information, and Address Information for an online user.

The Personal Information stores the IDs of a user, including the ID in SNS that used by the user to login an SNS, phone number, and email address. Note that a user may turn on a Peer node by logining into one or more SNSs at the same time, there may be one or more SNS IDs for the same user (i.e., multiple entries for the same user exist in the GlobalID List). These multiple entries are linked used the phone number (or email address) of the user.

The Social Network Information stores the SN Type indicating which SNS the user logins currently (i.e., online).

The Address Information stores the IP address and the port number of the Peer node turned on by the user. This information is valid when the Peer node is turned on.

Figure 2.2 (2) shows the software architecture of the Index Peer node. There are one main class **IndexPeerAgent**(see Figure 2.2 (2.1)) and a database GlobalID List (see

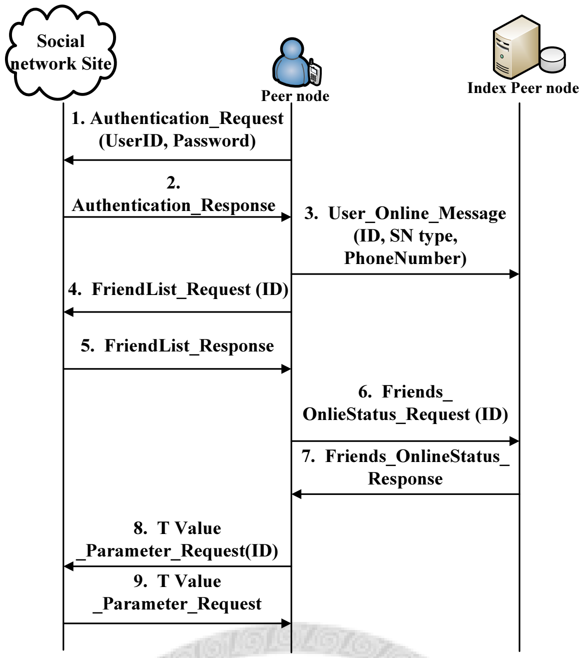


Figure 2.4: The Message Flow for The Login Procedure

Figure 2.2 (2.5)). In the the main class **IndexPeerAgent** (see Figure 2.2 (2.1)), the receiveSocket.receive() function (see Figure 2.2 (2.3)) is executed to receive the message sent from a Peer node. Upon receiving a message, the receivePacket.getData() function (see Figure 2.2 (2.2)) is invoked to get the information carried in this message. The receiveSocket.send() function (see Figure 2.2 (2.4)) is responsible to send the response message to a Peer node.

2.3 Turning on A Peer Node

This section describes the execution of a Peer node. When a user turns on the Peer node on his end device, the Login procedure. Figure 6.1 illustrates the message flow for the Login procedure with the following steps:

Step 1. When a user turns on the Peer node, a **SampleAuthListener** class is created, and the SessionEvents.addAuthListener(new SampleAuthListener()) function is exercised to authenticate the user in an SNS.

Step 2. If the authentication is successful, the SNS responses the user SNS ID in the return of the SessionEvents.addAuthListener() function.

Step 3. The Peer node creates a **BackgroundService** class to send a message (i.e., the User_Online_Message message) carrying the user's ID, Phone No., Email, IP address, port number, and SN Type, to the Index Peer. The Index Peer creates an entry for the user in the global ID list.

Steps 4 and 5. The Peer node creates a **CreateFriendListener** class (i.e., the FriendList_Request and FriendList_Response message pair) to get the IDs of the user's friends from the SNSs, and creates an entry for each friend in the Friend List.

Steps 6 and 7. The Peer node uses the **BackgroundService** class to send a message (i.e., the Friends_OnlineStatus_Request and Friends_OnlineStatus_Response message pair) to the Index Peer node to query the online friends of the user.

Steps 8 and 9. The Peer node creates a **FeedRequestListener** class to collect the social activity information to calculate the T value from the SNS by exchanging the T Value_Parameter_Request the T Value_Parameter_Response message pair.

Chapter 3

The Three Functions in the PeerAgent Class

Here we detail the three functions implemented in the **PeerAgent** class and explain the i-Search Mechanism used in the Relationship_Finding() function.

3.1 The Update_Tvalue() function and Update_FriendList() function

The execution of the Update_Tvalue() function is initiated by a user i and first it calls the mAsyncRunner.request in the Facebook Graph API to retrieve the JSON object [8], which contains the user i's Facebook information. We define three parameters $N_{m,u}$, $N_{r,u}$, and $N_{l,u}$ where $N_{m,u}$ denotes "the total number of message that a friend u posts on i's wall", $N_{r,u}$ denotes "the total number of message that a friend u replies to i", and $N_{l,u}$ denotes "the total number of "Likes" a friend u gives to i". The **FeedRequestListener** class parses the JSON object to obtain the three parameters $N_{m,u}$, $N_{r,u}$, and $N_{l,u}$.

Let $T_{i,u}$ denote the T value that a user i gives to his friend u. We define the T value calculation function as

$$T_{i,u} = \frac{W_m * \min\{N_{m,u}, \theta_m\}}{\theta_m} + \frac{W_r * \min\{N_{r,u}, \theta_r\}}{\theta_r} + \frac{W_l * \min\{N_{l,u}, \theta_l\}}{\theta_l} \quad (3.1)$$

where W_m, W_r, and W_l are the weight of $N_{m,u}$, $N_{r,u}$, and $N_{l,u}$, respectively. θ_m, θ_r, and θ_l are the threshold for $N_{m,u}$, $N_{r,u}$, and $N_{l,u}$, respectively. If $N_{m,u} > \theta_m$, the user is considered as a closed friend. The same applies to θ_r and θ_l. In our study, we set W_m, W_r to 0.4 and W_l to 0.2. We set the threshold θ_m, θ_r to 25 and θ_l to 50.

The concept of the T value calculation function is from the study [15]. The T value calculation function in the study is used in e-commerce communities and they computes the trust value of a peer u (i.e., a user) by a weighted average of the degree of satisfaction u receives for each transaction. We apply this concept to our P2P-iSN work and design a similar equation (3.1). We take the parameters (i.e., N_m, N_e, and N_l) of a user's SNS information which contains the interaction information with friends, and weight these parameters to derive an average interaction score that represent the trust value of the friend.

The Update_FriendList() function is invoked by a user and it sends a list of friend's ID to the Index Peer through the **BackgroundService** class. The Index Peer returns the IDs which are found in the GlobalID_List. These IDs indicate which friends are online. After receiving these IDs, the "Online" field of the corresponding IDs in the Friend List are set to "On", while the others are set to "Off". We use the "Polling" concept to update the Friend List, which is mean that, we do not update the Friend List when user is online or offline because this will enhance Index Peer node overhead.

3.2 i-Search Mechanism

In this chapter, we propose an i-Search mechanism to find a social path between two Peer nodes in P2P-iSN. Though searching in a social graph has been studied in the previous works [16], most of these studies considered a centralized searching, that is, a social graph is well maintained in a central node. Fewer studies have addressed searching in a P2P social network, which is the main focus of this paper.

The concept of the i-Search mechanism is similar to the flooding search that has been widely adopted in the large network studies (e.g., [2]). To convey this social path, we define that a *global relationship* exists between user 1 and user $L+1$. We propose a function $Z(\mathbf{P})$ to measure the strength of the global relationship between user 1 and user $L+1$, which is defined by

$$Z(\mathbf{P}) = \begin{cases} 1, & \text{if } L = 0; \\ \prod_{i=1}^{L} T_{i,i+1}, & \text{otherwise (i.e., } L \geq 1). \end{cases} \tag{3.2}$$

The i-Search mechanism establishes the social path link by link. When a link is added into a path, global relationship strength is calculated for the new path using the $Z(\cdot)$ function in (3.2). If the global relationship strength for the new path is below a threshold

Δ, the establishment of the social path stops.

Note that Δ is used to guarantee that the global relationship strength for the constructed path is strong enough so that users are motivated to use the global social relationship for further SNS applications. We set up Δ based on the research results in the sociology study [13]. As mentioned in [13], on average, the $T_{i,j} is 0.5$. If we consider a path **P** with length $|\mathbf{P}| = 4$, then using the $Z(\cdot)$ function in (3.2), the global relationship strength for the path is $Z(\mathbf{P}) = 0.5^4 = 0.0625$, which is considered a very weak relationship. Therefore, in the performance study later, we set $\Delta = 0.5^3 = 0.125$. In other words, it is likely that the social path (searched by the i-Search mechanism) has path length no larger than 3. As mentioned in [9], with path length no larger than 3, the flooding search is considered with low complexity. This is the main reason why we use the flooding search.

Details of the i-Search mechanism are given below: The Index Peer node maintains the online status (including the ID and IP address of the Peer node) for the online Peer nodes. A friend list is maintained in the Peer node, which stores the online information for all friends of the Peer node. To simplify our description, we use "the friend b of a Peer node a" to imply that the social link $a \rightarrow b$ exits.

When a Peer node is turned on, it reports its online status to the Index Peer node, and receives the latest online status for his friends from the Index Peer node. With the latest online information, the Peer node can determine whether his friend is online (i.e., a Peer node is turned on). A online Peer node can communicate with his online friends directly. We run a recursive algorithm, the iSearch algorithm, in the Peer node as shown in Algorithm 1. In this algorithm, the set **G** is the friend list of a Peer node. The input parameter s stores the ID of the Peer node who calls the iSearch algorithm, and r is the ID of the Peer node to be searched. Initially, we set $\mathbf{P} \leftarrow \emptyset$.

Algorithm 1: iSearch

Input: s, r, $\mathbf{P}$, $Z(\mathbf{P})$

Output: $\mathbf{P}_{new}$, $Z(\mathbf{P}_{new})$

foreach $v : v \in \mathbf{G}$- $\{\mathrm{P}\}$ **do**
    if $v = r$ **then**
        $\mathbf{P}_{new} \leftarrow \mathbf{P} \cup \{s \rightarrow v\}$;
        $Z(\mathbf{P}_{new}) \leftarrow Z(\mathbf{P})F(s, v)$;
        return;
    else if v *is online, and* $Z(\mathbf{P})F(s, v) > \Delta$ **then**
        v.iSearch(v, r, $\mathbf{P} \cup \{s \rightarrow v\}$, $Z(\mathbf{P})F(s, v)$);
    else if v *is off-line, or* $Z(\mathbf{P})F(s, v) \leq \Delta$ **then**
        quit;
    end
end

Consider the scenario where the Peer node a searches the Peer node d. A user a can "request" his friend b to execute the iSearch algorithm (i.e., b.iSearch() in Algorithm 1) through the direct communication if b is online. That is, the directional social path $\mathbf{P}$ is established along the online Peer nodes.

Note that the i-Search mechanism may find multiple global social relationships between two Peer nodes. For the Peer node who triggers the i-Search mechanism, he can use the one with the largest global social relationship strength. Furthermore, we can speed up the execution of the i-Search mechanism by caching the searching results on the Peer nodes. To simply our discussion, we do not include the study for the effects of the cache in this paper.

Chapter 4

Analytical Model

All Peer nodes and the corresponding social links in P2P-iSN form a social graph. A Peer node may be turned on or turned off during the execution of i-Search, and the iSearch request can reach the friends only when the friends are online. In other words, a social link $a \to b$ does not exist if Peer node a or b is turned off (i.e., off-line). Therefore, the physical network topology of P2P-iSN changes dynamically when the i-Search mechanism is being executed.

Let P_f be the "path found" probability that a directional social path exists when a Peer node a executes the i-Search mechanism to find a Peer node d. The online status of a Peer node affects the P_f probability significantly. In this chapter, we propose an analytical model to obtain an approximation value for P_f.

To simplify our discussion, we assume that the behaviors of the Peer nodes in P2P-iSN are i.i.d. As discussed in Chapter 3, in this paper, we set $\Delta = 0.5^3 = 0.125$ in the i-Search mechanism. In this analytical model, we use the constraint $|\mathbf{P}| \leq 3$ instead of

$\Delta \leq 0.125$, i.e., the i-Search mechanism quits when the path length reaches 3, and no global social path is found.

Assume that a Peer node is turned on (i.e., online) for a time period x (with the density function $f_x(\cdot)$ and mean $1/\mu_x$), and then it is turned off (i.e., off-line) for a time period y (with the density function $f_y(\cdot)$ and mean $1/\mu_y$). The Peer node alters between x and y. Suppose that iSearch request arrivals to a Peer node form a Poisson process. Then according to the alternating renewal process [12], the probability p_{on} that an iSearch request arrives when a Peer node is online can be obtained by

$$p_{on} = \frac{E[x]}{E[x] + E[y]} = \frac{\mu_y}{\mu_x + \mu_y} \tag{4.1}$$

Before the derivation, we generate the social graph for P2P-iSN using the W.S. model [14] with the three parameters α (i.e., the rewire probability), n (i.e., the total number of Peer nodes in P2P-iSN), and m (i.e., the average number of friends of a Peer node). With the setup:

$$0 < \alpha < 1 \text{ and } n \gg m \gg \ln n \gg 1 \tag{4.2}$$

the W.S. model has the small-world property, including short average length and high clustering. The small-world property can also apply to SNS [11]. The details of the W.S. model can be found at [14].

Let N_t denote the expected number of the Peer nodes that receive the iSearch request message during the execution of the i-Search mechanism. Consider the scenario that the Peer node a executes the i-Search mechanism to search a directional social path to d. If d belongs to one of the N_t peer nodes, then the directional social path from a to d is found.

Therefore, we have

$$P_f = \frac{N_t}{n}. \tag{4.3}$$

We derive N_t as follows. There are two types of nodes including "far-nodes" and "near-nodes" defined in the W.S. model. The far-nodes represents the Peer nodes that have social links after rewiring with probability α. The near-nodes represents the Peer nodes that have social links initially.

In the social graph of the P2P-iSN, let N_f and N_n respectively be the expected numbers of far-nodes and near-nodes that receive an iSearch request when the i-Search mechanism is executed. Then we have

$$N_t = N_f + N_n.$$

The N_f and N_n are obtained as follows. One round means that the iSearch request is delivered using a directional social link $a \to b$ when both Peer nodes a and b are online. In the i-Search mechanism, there are at most three rounds to construct a directional social path. In each round, a Peer node that triggers the round can be either a far-node or near-node:

Case 1: The Peer node that triggers the round is a far-node. In this case, there are on average $m\alpha p_{on}$ far-nodes and $m(1-\alpha)p_{on}$ near-nodes that can receive the iSearch request.

Case 2: The Peer node that triggers the round is a near-node. Because there is high probability that the near-node sends the iSearch request to another near-node that has received this iSearch request previously, we consider that only far-nodes can receive the iSearch request for the approximation. In this case, there are on average $m\alpha p_{on}$ far-nodes that can receive the iSearch request.

We use the following interative procedure to calculate the N_f and N_n.

Procedure 1.

Input parameters: α, m, μ_x, μ_y.

Output measures: N_f, N_n, N_t.

Step 1. Select initial values, $N_f \leftarrow 1$, $N_n \leftarrow 0$, and $round \leftarrow 0$;

Step 2. $N_f \leftarrow m\alpha \left(\dfrac{\mu_y}{\mu_x + \mu_y} \right) (N_f{+}N_n)$; $N_n \leftarrow m(1{-}\alpha) \left(\dfrac{\mu_y}{\mu_x + \mu_y} \right) N_f$; $round{+}{+}$.

Step 3. If ($round \leq 3$) then go to Step 2. Otherwise, go to the next step.

Step 4. $N_t \leftarrow N_f + N_n$; return.

The analytical model is validated by simulation experiments of a discrete event-driven simulation model, which has been widely adopted to simulate the mobile communications network in several studies (e.g., [4]). The simulation model simulates the online/off-line behavior of a Peer node and the behavior of the i-Search mechanism.

In the simulation model, we adopt the discrete event-driven approach in our simulation model, which has been widely applied in many networking studies (e.g., [4]). In our simulation model, we define five types of events listed below:

- The **QUERY_ARRIVAL** event represents that an online Peer node starts the i-Search mechanism to find another Peer node.
- The **QUERY_FORWARD** event represents that an online Peer node sends a iSearch request to his online friend.
- The **QUERY_RESPONSE** event represents that an online Peer node returns the results (i.e., a path is found) for the execution of the iSearch algorithm to the Peer node who sends the iSearch request.
- The **ONLINE** event represents that a Peer node is turned on.
- The **OFFLINE** event represents that a Peer node is turned off.

We maintain a timestamp t_s to indicate the time when an event occurs. The events are inserted into an event list and deleted/processed from the list in a non-decreasing timestamp order. During execution of the simulation, a simulation clock t_c is maintained, which indicates the progress of simulation. The following variables are used in the simulation model:

- N_r indicates the number of rounds that have been executed for an iSearch request.
- a is the ID of the Peer node who triggers the iSearch mechanism.
- d is the ID of the Peer node to be found.
- l indicates whither a social link exists between two Peer nodes.

We use the following counters in our simulation model to calculate the output measure:

- The C_f counter counts the total number of finding a path successfully.
- The C_q counter counts the total number of the **QUERY_ARRIVAL** events that have been processed.

We repeat the simulation runs until C_q exceeds $100,000$ to ensure the stability of the simulation results. Then we obtain the output measure:

$$P_f = \frac{C_f}{C_q}$$

Figures 5.1 and 5.2 show the comparison between the analytical and simulation results, whose details of the parameter setups are described in chapter 5. The figures indicates that the analysis results approximate the simulation results well.

Chapter 5

Performance Evaluation

In this chapter, we study the effects of the input parameters on the P_f performance for the i-Search mechanism. In our study, we set the input parameters following the constraints in (4.2), and we set the total number of Peer nodes $n = 1000$. The effects of the input parameters are described as follows. In Figures 5.1 and 5.2, we change μ_y/μ_x from 0.5 to 8. A larger μ_y/μ_x implies that the Peer node spends more time online. For example, when $\mu_y/\mu_x = 0.5$ and $\mu_y/\mu_x = 8$, from (4.1), we have $p_{on} = 1/3$ and $p_{on} = 8/9$, respectively. Both figures show that the path found probability P_f increases as μ_y/μ_x increases. It is worth noticing that we have P_f larger than 15% when $\mu_y/\mu_x = 8$ and $\alpha = 0.8$ as shown in Figure 5.1 (with $m = 6$), and P_f is around 40% when $\mu_y/\mu_x = 8$ and $m = 10$ as shown in Figure 5.2.

Observing Figure 5.1 where we set $m = 6$, we study the effects of α. A larger α implies that the social graph of P2P-iSN is sparser (i.e., more far-nodes). Figure 5.1 indicates that P_f increases as α increases, which means that in a sparser social graph, the

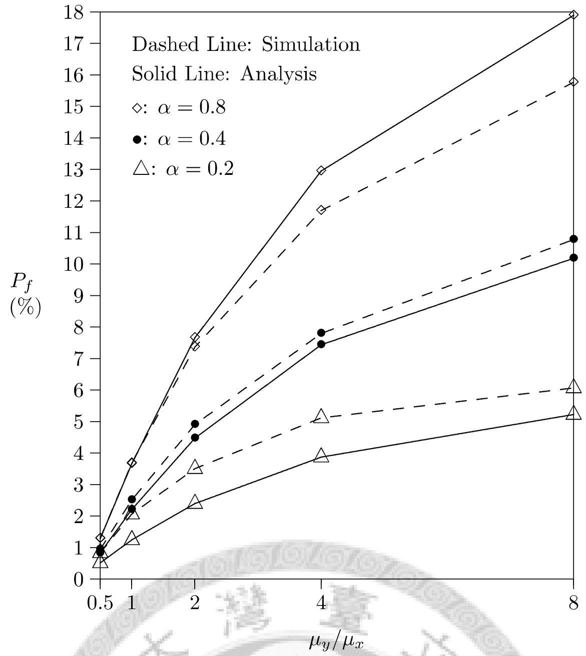


Figure 5.1: Effects of α ($n = 1000$; $m = 6$)

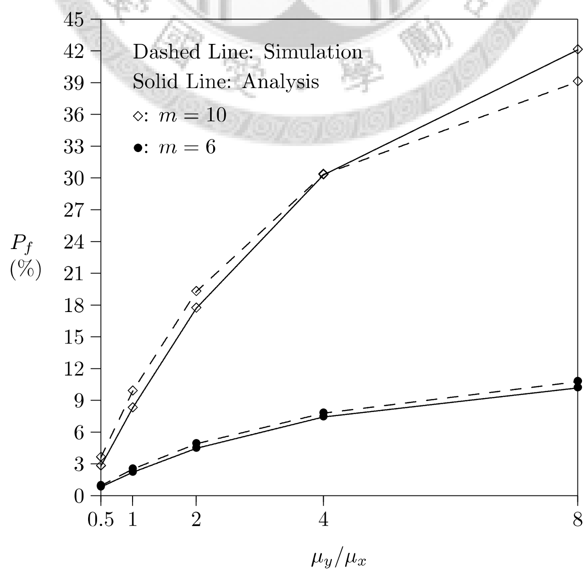


Figure 5.2: Effects of m ($n = 1000$; $\alpha = 0.4$)

i-Search mechanism attains better found probability. In Figure 5.2, we study the effects of m where we set $\alpha = 0.4$. A larger m implies more friends of a Peer node. Figure 5.2 shows that with more friends, the i-Search mechanism achieves better P_f performance.

To conclude, when in a sparser social graph and a peer node has more friends, there is 40% probability that the i-Search mechanism could find a global social relationship for the user, i.e. a social path with strong relationship strength.

Chapter 6

Conclusions

This paper studies the aggregation of social relations across heterogeneous SNSs with an end-goal to find a social path with strong strength between any two SNS users. P2P-iSN, a peer-to-peer network architecture, is proposed to integrate multiple SNSs without incurring overhead to the SNSs. We then develop an effective Global Relationship Model to calculate the global relationship strength between two users from heterogeneous SNSs with more precision. With P2P-iSN and the Global Relationship Model as foundation, we propose the i-Search mechanism that realizes social path finding in a P2P social network. Analytical and simulation models are conducted to investigate the performance of the i-Search in terms of the path found probability P_f.

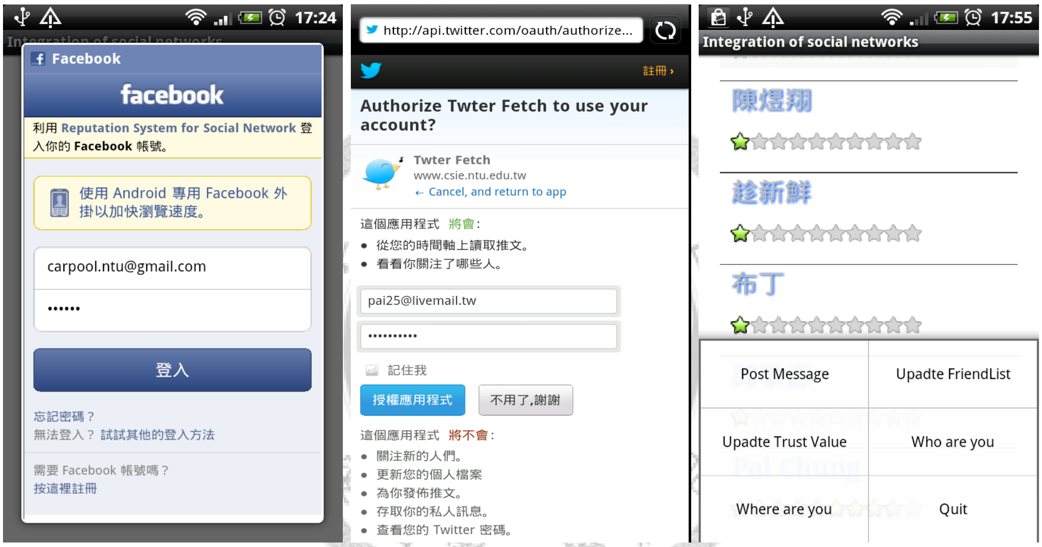
Our study indicates that when the social graph is sparser (e.g., $\alpha = 0.8$) and a Peer node has more friends (e.g., $m = 10$), the path found probability P_f is around 40%. The proposed i-Search mechanism can effectively find global social relationship for users from heterogeneous SNSs. The research results encourage us to implement the i-Search mechanism in real SNSs. Compared to identity data and content data aggregation across different SNSs, social path data aggregation has been much less studied and constitutes a major research challenge moving forward. This paper thus also offers important insights for further studies that aim to utilize the abundant social network and user behavior data.

Appendix A

Snapshots of P2P-iSN

This section we show some snapshots of P2P-iSN.

Figure 6.1 shows the snapshots that user can login to P2P-iSN by using different type social networks accounts. The login interface which is implemented by Facebook Graph API (see figure 6.1(a)) and Twitter REST API (see figure 6.1(b)).Figure 6.1(c) shows the functionality of P2P-iSN which include direct message communication, message post on the SNS, find the relationship between two users, query from Index Peer node that friend is on what type of SNS. Figure 6.2(a) shows the direct message communication between two users, the message does not pass through the SNS, it enhances the user privacy. Searching for the relationship between two users is showed in figure 6.2(b). User can use this functionality to find other user and know their relationship. Figure 6.2(c) shows the searching result for the request from 6.2(b), “Min Cai $\rightarrow$ Pai Chung $\rightarrow$ Alicia Lin” means that Alicia Lin is the friend of Pai Chung and Pai Chung is the friend of Min Cai. Figure 6.2(d) shows the request that passed to “Pai Chung”.



(a) (b) (c)

Figure 6.1: (a) Facebook Graph API (b) Twitter REST API (c) Functionality of P2P-iSN

Figure 6.2: (a) Direct message communication with friends (b) Search for the relationship between two users (c) The search result for the request from a friend (d) The request passed to user's friend

Bibliograghy

[1] Bae, J. A Global Social Graph as a Hybrid Hypergraph. *Proceedings of Fifth International Joint Conference on INC, IMS and IDC*, pages 1025–1031, August 2009.

[2] Chang, N.B. and Liu, M. Controlled Flooding Search in a Large Network. *IEEE/ACM Transactions on Networking*, 15(2):436–449, April 2007.

[3] Ellison, N. and Boyd, D. Social Network Sites: definition, history, and scholarship. *Journal of Computer-Mediated Communication*, 13(1):210–230, October 2007.

[4] Fu, H.-L., Chen, H.-C., Lin, P., and Fang, Y. Energy-Efficient Reporting Mechanisms for Multi-Type Real-time Monitoring in Machine-to-Machine Communications Networks. *Proceedings of IEEE INFOCOM 2012 Conference*, pages 136–144, March 2012.

[5] http://developer.android.com/reference/packages.html.

[6] http://openid.net.

[7] https://developers.facebook.com/docs/reference/api/.

[8] http://www.json.org.

[9] Jiang, S., Guo, L., Zhang, X., and Wang, H. LightFlood: Minimizing Redundant Messages and Maximizing Scope of Peer-to-Peer Search. *IEEE Transactions on Parallel and Distributed Systems*, 19:601–614, May 2008.

[10] Ko, M. N., Cheek, G. P., Shehab, M., and Sandu, R. Social-Networks Connect Services. *Computer*, 43(8):37–43, August 2010.

[11] Mislove, A., Marcon, M., Gummadi, K. P., Druschel, P., and Bhattacharjee, B. Measurement and Analysis of Online Social Networks. *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, pages 29–42, 2007.

[12] Nelson, R. *Probability, Stochastic Processes, and Queueung Theory*. Springer Verlag, 1995.

[13] Ronald, S. B. *STRUCTURE: A General Purpose Network Analysis Program Providing Sociometric Indices, Cliques, Structural and Role Equivalence, Density Tables, Contagion, Autonomy, Power and Qquilibria in Multiple Network Systems (Version 4.2).* New York: Columbia University Press, 1991.

[14] Watts, D. J. and Strogatz, S. H. Collective Dynamics of "Small-World" Networks. *Nature*, 393(6684):440–442, 1998.

[15] Xiong, L. and Liu, L. PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities. *IEEE Transactions on Knowledge and Data Engineering*, 16(7):843 – 857, 2004.

[16] Yu, B. and Singh, M. P. Searching Social Networks. *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS '03)*, pages 65–72, 2003.