



ISLAMIC UNIVERSITY OF TECHNOLOGY

---

**Community Recommendation in Social Network Using  
Strong Friends Based on Quasi-Clique Approach.**

---

*Authors:*

**Anjum Ibna Matin (104402)  
Sawgath Jahan (104444)**

*Supervisor:*

**DR. Mohammad Rezwanul Huq  
Assistant Professor  
Department of Computer Science and Engineering**

***A thesis submitted to the Department of CSE  
In partial fulfilment of the requirements for the degree***

***B. Sc. Engineering in CSE***

***Academic Year: 2013-2014***

A Subsidiary Organ of the Organization of Islamic Corporation  
Dhaka, Bangladesh

## ***Declaration of Authorship***

*This is to certify that the work presented in this thesis is the outcome of the analysis and investigation carried out by Anjum Ibna Matin and Md. Sawgath Jahan under the supervision of Dr. Mohammad Rezwanaul Huq in the Department of Computer Science and Engineering (CSE), IUT, Dhaka, Bangladesh. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given. Our work has been accepted in 8<sup>th</sup> International Conference on Electrical and Computer Engineering (ICECE 2014) and it's in the top 42% of total 530 submission.*

*Authors:*

---

Anjum ibna Matin  
Student ID – 104402

---

Sawgath Jahan  
Student ID – 104444

Supervisor:  
DR. Mohammad Rezwanaul huq  
Assistant Professor  
Department of Computer Science and Engineering

# Abstract

A social networking service is a platform to build social networks or social relations among people who, share interests, activities, backgrounds or real-life connections. Social network analysis is needed because the number of users is increasing rapidly day by day. Now days, users are involved themselves in to communities. They share post, their views, what they like in communities. So it is important for them to find suitable communities where they have common factors like friends, followers and their activities etc. Here we are working with a technique for recommending a community in social network like Facebook, twitter etc. We use some graph terminologies and graph mining techniques. Finding strong friends, we recommend communities for a user in a social network. We apply data mining techniques to help social users to pick out suitable community of a social network like Facebook, twitter etc. Big social network sites use their own algorithm. Here we are not improving an existing algorithm but giving a new method for community recommendation in social network. That is workable for both real and synthetic data. As real data are not given by any big social network so we use sample data to prove our algorithm. And we make a survey and thus we improve our **Community Recommendation Algorithm (CRA)**.

# Contents

Declaration of Authorship	<i>i</i>
Abstract	ii
List of Figures	iv
List of Tables	v
<i>Chapter 1:: Introduction</i> .....	<b>6</b>
<i>Chapter 2:: Problem Definition</i> .....	<b>8</b>
<i>Chapter 3:: Related Works</i> .....	<b>9</b>
<i>Chapter 4:: Ours Approach</i> .....	<b>10</b>
<i>Chapter 5:: Data Set</i> .....	<b>11</b>
<i>Chapter 6:: Definitions</i> .....	<b>13</b>
<i>Chapter 7:: How Algorithms Works</i> .....	<b>16</b>
7.1:: Step-1.....	<b>16</b>
7.2:: Step-2.....	<b>17</b>
7.3:: Step-3.....	<b>17</b>
7.4:: Step-4.....	<b>18</b>
7.5:: Algorithm (CRA) .....	<b>19</b>
<i>Chapter 8:: Experimental Result</i> .....	<b>23</b>
<i>Chapter 9:: Evaluation</i> .....	<b>26</b>
<i>Chapter 10:: Conclusion</i> .....	<b>28</b>
Coding Part	29
Bibliography	33

## List of Figures:

<a href="#"><u>Figure 6.1: Clique</u></a> .....	14
<a href="#"><u>Figure 6.2: Quasi-Clique</u></a> .....	15
<a href="#"><u>Figure 7.1: Dense Subgraph</u></a> .....	17
<a href="#"><u>Figure 7.2: Community C1</u></a> .....	18
<a href="#"><u>Figure 7.3: Community C2</u></a> .....	19
<a href="#"><u>Figure 8.1: Experimental Case-1</u></a> .....	23
<a href="#"><u>Figure 8.2: Experimental Case-2</u></a> .....	23
<a href="#"><u>Figure 8.3: Experimental Case-3</u></a> .....	24
<a href="#"><u>Figure 8.4: Experimental Case-4</u></a> .....	24
<a href="#"><u>Figure 8.5: Experimental Case-5</u></a> .....	24
<a href="#"><u>Figure 9.1: Equation of Algorithm[1]</u></a> .....	26
<a href="#"><u>Figure 9.2: Rule Based Mining</u></a> .....	27

## ***List of Tables:***

<a href="#"><u>Table 5.1 : Users and friends of each user in Social network</u></a> .....	11
<a href="#"><u>Table 5.2 : Communities of each User</u></a> .....	12
<a href="#"><u>Table 7.1 : Normalization</u></a> .....	16
<a href="#"><u>Table 9.1 : Specific Point of table</u></a> .....	26
<a href="#"><u>Table 5.1 : Strength of Relationship</u></a> .....	27

# Chapter 1

---

## Introduction

Now a days, the rapid growth and exponential use of social digital media has led to an increase in popularity of social networks and the emergence of social computing [1]. In general social networks are structures made of social entities that are linked by some specific types of interdependency (e.g., kinship, friendship, common interest, beliefs, or financial exchange). The main types of social networking services are those that contain category places (such as former school year or classmates), means to connect with friends, community and a recommendation system for selecting the appropriate community.

Recommendation system like social graph generation & forecasting using social network mining [1], Finding strong groups of friends [3], finding popular friends [4] in social network are well known. In this paper we develop a recommendation system based communities. Day by day community is increasing with rapid growth of social user. Nowadays we see the vast activity of social life. All those community can't be suitable for social users. So our recommendation system will find suitable community for a user. For example, a social user has created a new profile in social network service. He or she has some common interests with some stranger user or his/her friend. So they can create a community where being friend is not important but sharing the common interest is important. Suppose a cricketer always like to join in such groups which are involved with cricket. He must have some friends in his friend list who are interested in playing cricket and based on those friends who are connected with some groups associated with cricket we will recommend him a community which will be suitable for him.

We propose a mining algorithm to help users to find suitable community. Mining algorithms like F-Tree mining, Association rule mining seems difficult and time consuming to work with this large dataset. So we design a new algorithm for this recommendation system.



## Chapter 2

---

# Problem Definition

In this section, we present the problem formulation and basic definitions for mining set of friends having higher strength and set of communities which is associated with them.

Let us consider a sample database in Table-I that captures some users in a social network. The table consists of the friend lists of five users. For example, Jack records the list of jack's friends namely- Harry, Pet, James, Oliver and Kan. The list also shows the interaction strength (e.g., connectivity of each friend of Jack). For example, "James (15)" in the list may record that the interaction between them is 15, these interactions may be of various types like posting message on wall , chatting , giving likes to photos or status etc. in any social network like Facebook .

Now in table II, we have listed those set of communities that is associated with each user in a social Community. For example, Jack is a member of C3, C4 and C5 communities. Based on this connectivity information, we have find out which community is recommended for the particular user. For example, if we see Table-2, jack is connected with community C3, C4 and C5 but he is not connected with C1, C2 and C6 whereas his friend Harry, Pet, Kan, James, Oliver and Tom are connected with C1 and C2. Based on his friends we first differentiate his strong and weak friends and thus applying our algorithm we recommend a community between C1 and C2 for Jack.

## Chapter 3

---

### Related Works

As we work on recommending a community for a user in a social network, there are some existing works on it [1], [2], [9]. One of those is Social Graph Generation & Forecasting using Social Network Mining [1]. Here they used one way communication between the users and his corresponding friends and assigned a strength value for each of the friends and based on those values they came to an decision according their rule to recommend a community as well as the second one Graph Based Forecasting For Social Networking Site, "Communication [2], where the difference is the second one used two way directed graph to assign the strength values for defining the interactions between the users and his corresponding friends. But the problem is, they used some rules those are not well defined or proved and for a large dataset it will take more time and the accuracy was not so good. As we have used some basic graph techniques like clique and quasi-clique [7] and based on those techniques we divide all the friends of a user into two groups, strong and weak, and for these we can access a portion of his friends at a time. For finding strong friends [3] or popular friends [4] the existing researches worked on finding a group of friends not on finding a single friend who is strong or weak for a particular user. Here we have used normalization and a minimum strength value to detect a particular friend who is weak or strong for a particular user. Then we used our algorithm for comparing those communities and thus we recommend one for the particular user.

## Chapter 4

---

### Ours Approach

In section-3 (Related work), we have discuss the main problem of [1] (Social Graph Generation & Forecasting using Social Network Mining). Here they used one way communication between the users and his corresponding friends and assigned a strength value for each of the friends. Based on those values they came to a decision according their rule to recommend a community. In [2] (Graph Based Forecasting for Social Networking Site), two way directed graph is used to assign the strength values for defining the interactions between the users and his corresponding friends.

It is obvious that this algorithm can't work with big data set. A simple well defined reason is that they assign a range and point to identify the rank of users/ friends based on their communication [1].So our approach is solve the strength values for defining the interactions between the users and his corresponding friends in a well-defined method. Then we have to sort it so that we get our useful data. We have used some basic graph techniques like clique and quasi-clique [7] and based on those techniques we divide all the friends of a user into two groups, strong and weak, and for these we can access a portion of his friends at a time. Here we have used normalization and a minimum strength value to detect a particular friend who is weak or strong for a particular user. Then we used our algorithm for comparing those communities and thus we recommend one for the particular user.

# Chapter 5

---

## Data Set

In this section, we present our sample data set. Let us consider a sample database in Table-I that captures some users in a social network. The table consists of the friend lists of five users. In table II we can see that, Jack records the list of jack's friends namely- Harry, Pet, James, Oliver and Kan. The list also shows the interaction strength (e.g., connectivity of each friend of Jack). For example, "James (15)" in the list may record that the interaction between them is 15, these interactions may be of various types like posting message on wall, chatting, giving likes to photos or status etc. in any social network like Facebook.

Users ( $u_i$ ) and friends of each user in a social network

User( $u_i$ )	List of friends( $f_{ui}$ ) with interaction strength
Jack	Harry(35), Pet(25), James(15), Oliver(10), Kan(10), Tom(5)
Harry	Jack(15), Pet(5), Oliver(25)
Pet	Jack(15), Harry(10), James(5)
James	Olive(20), Pet(10), Jack(15)
Oliver	Harry(5), Jack(15), Kan(20)
Kan	Harry(5), Jack(15), Tom(20), Oliver(20)
Tom	Kan(15), Jack(20), Pet(20)

**Table-5.1**

Now in table III, we have listed those set of communities that is associated with each user in a social Community. For example, Jack is a member of C3, C4 and C5 communities.

Communities ( $c_i$ ) of each User( $u_i$ )

User( $u_i$ )	Communities( $c_i$ )
Jack	C3, C4,C5
Harray	C1, C2, C4
Pet	C1, C3
James	C1, C2, C6
Kan	C1, C2, C3, C5
Oliver	C1,C2,C3
Tom	C1,C2

**Table-5.2**

# Chapter 6

---

## Definitions

In the remainder of this section, we provide formal definitions and notations required to mine friend groups and communities from a social media database.

### **Definition 1:**

User ( $u_i$ ), we define a set  $U = \{u_1, u_2, \dots, u_n\}$  who are the users of a social network site. Example 1: Jack = {Harry (35), Pet (25), James (15), Oliver (10), Kan (10), Tom(5)}.

### **Definition 2:**

Community ( $c_{ui}$ ), A community is a group of people with a common interest among them. We define a set  $C = \{C_1, C_2, \dots, C_n\}$  that are the communities in a social network. Example 2: Pet is associated with  $C_1, C_3$  communities.

### **Definition 3:**

Normalized Interaction Strength ( $nis_{ui, u_j}$ ), Interaction strength is the value or weight that defines the interaction of a user ( $u_i$ ) with one of his friends ( $u_j$ ) and we normalize these values based on the total number of interactions of the user ( $u_i$ ). Example 3: Jack = {Harry (35), Pet (25), James (15), Oliver (10), Kan (10), Tom

(5)}. Here Harry and Jack are friends and the interaction value for them is 35 and the normalized interaction strength ( $nis_{ui,uj}$ ) is 0.35.

**Definition 4:**

Strong Friends ( $s_{ui}$ ), We define  $S_{ui} = \{s_1, s_2, \dots, s_n\}$  as a set of ordered based on  $nis_{ui,uj}$  strong friends of user ( $u_i$ ) whose normalized cumulative interaction strength (ascending order) is less than  $min_{str}$  value. The set indicates that given a  $min_{str}$  value, the user ( $u_i$ ) is most likely going to interact with a member of set ( $s_{ui}$ ).

**Definition 5:**

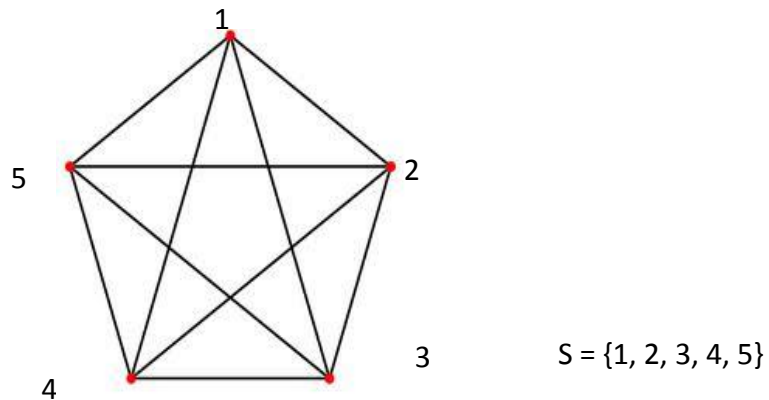
Weak Friends ( $w_{ui}$ ), we define  $W_i = \{w_1, w_2 \dots w_n\}$  as a set of weak friends whose normalized interaction strength is below a  $min_{str}$  value.

**Definition 6:**

Minimum strength ( $min_{str}$ ), Minimum strength value ( $min_{str}$ ) is the threshold or base value for recognizing strong and weak friends.

**Definition7:**

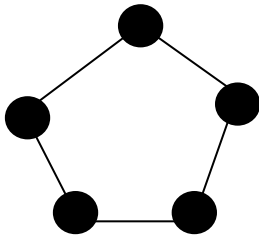
Clique, A set of vertices  $S$  is called a clique if the sub graph  $G(S)$  induced by  $S$  is complete and there is an edge between any two vertices in  $G(S)$ . [7]



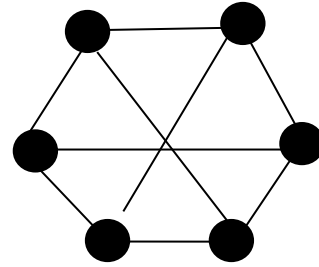
**Figure 6.1:** Clique

**Definition 8:**

$\gamma$ -quasiclique, A graph  $G = (V, E)$  is a  $\gamma$ -quasi-clique ( $0 \leq \gamma \leq 1$ ) if  $G$  is connected, and for every vertex  $v \in V$ ,  $\deg G(v) \geq \lceil \gamma \cdot (|V| - 1) \rceil$ . [7]



0.5-quasi-clique



0.6-quasi-clique

**Figure 6.2:**  $\gamma$ -quasi-clique:



# Chapter 7

---

## How Algorithms Works


Our proposed approach for recommending a community is based on several steps. We use some graph mining method to get result. Our algorithm first scans the friend database. After scanning, they follow some steps which are described below.

### 7.1 STEP-1(Normalization):

First we normalize the value of interaction strength. We calculate the normalization value of interactive strength for every friend by dividing the interaction strength with the total number of interaction. The total value of normalized interaction strength should be equal to 1. Example: For Jack in table 1 the normalized values for his friend set  $(f_i) = \{Harry (0.35), Pet (0.25), James (0.15), Oliver (0.10), Kan (0.10) \text{ and } Tom (0.05)\}$ . Then we arrange these normalized values in a descending order. The order of normalized interaction strength  $(nis_{ui, uj})$  for our input dataset is given below.

Normalization and  $min_{str} = 0.7$

Friends Of	Normalized Interaction Strength	Cumulative Normalized Interaction Strength
Harry	0.35	0.35
Pet	0.25	0.60
James	0.15	0.75
Oliver	0.10	0.85
Kan	0.10	0.95
Tom	0.05	1.00



Cumulative  $nis_{ui, uj} \leq min_{str}$

**Table-7.1:Normalization**

## 7.2 STEP-2 (Recognizing $s_{u_i}$ and $w_{u_i}$ ):

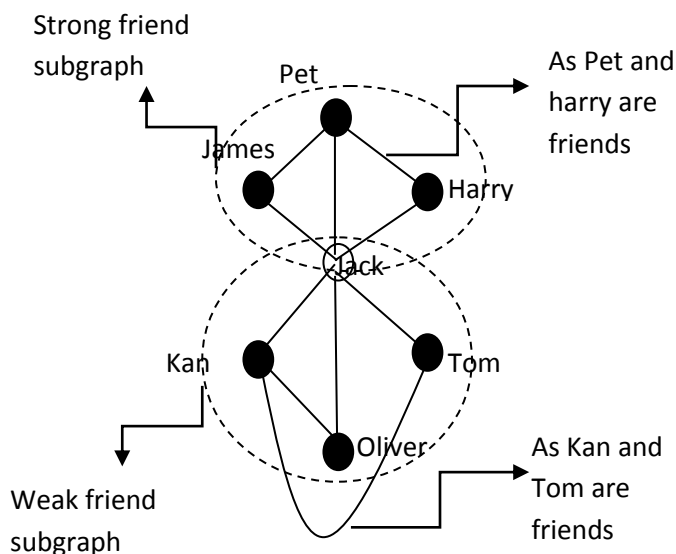
Here we divide the friend dataset for a particular user ( $u_i$ ) into two sets. One is strong friend set  $s_{u_i}$  and other one is weak friend set  $w_{u_i}$ . According to Table-3 the result for strong friend set ( $s_{u_i}$ ) and weak friend set ( $w_{u_i}$ ) is given below.

Set of strong friends  $S_{u_i} = \{\text{Harry (0.35), Pet (0.25), James (0.15)}\}$ .

Set of weak friends  $W_{u_i} = \{\text{Kan (0.10), Oliver (0.10), Tom (0.05)}\}$ .

## 7.3 STEP-3 (Creating Dense Sub graph):

Here we create the dense sub graph for differentiating strong and weak group of friends. The strong friends of a user ( $u_i$ ) creates a subgraph of strong friend group  $s_{u_i}$  and weak friends group  $w_{u_i}$ . Then we calculate the  $\gamma$ -quasi-clique value for each sub graph. The friends of a user who are also friend with one another from the friend database should be connected with one another. Suppose for our input data set Pet and Harry are also friends. So they should be connected.

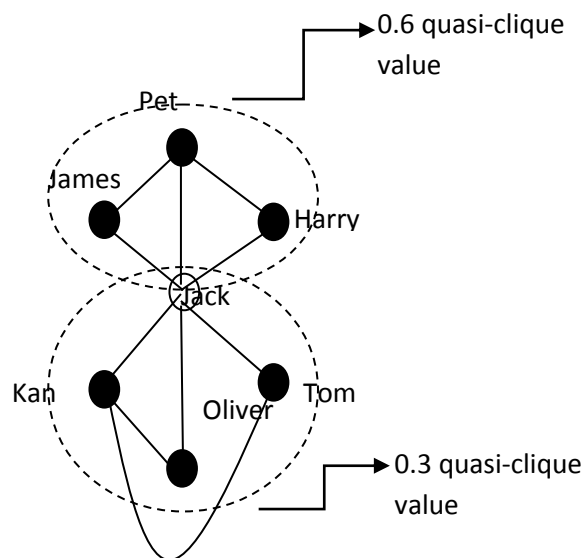


**Figure 7.1:** Dense Subgraph

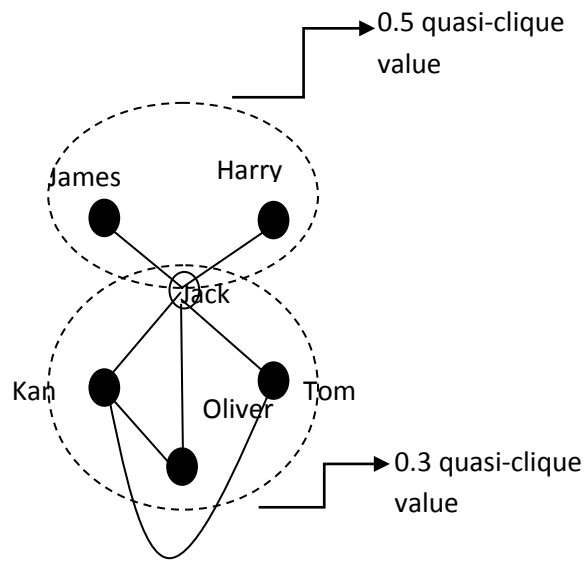
## 7.4 STEP -4 (Comparing Communities):

1. The user should not be connected with the communities which will be compared for recommendation.
2. Take two communities from the list of communities of the user's friends.
3. Only consider those friends with that particular community.

Find the  $\gamma$  -quasi-clique value for each sub graph for each community and then apply our comparison algorithm for recommending a community.



**Figure 7.2:** Community C1



**Figure 7.3:** Community C2

The steps are defining that first we consider only those communities with which the user is not connected. Then take two communities for comparing which are in the list of communities for the friends of ( $u_i$ ). Then calculate the  $\gamma$  -quasi-clique value for each subgraph and for each community.

### 7.5 Community Recommendation Algorithm (CRA):

Here  $q_i [s_{u_i}]$  denotes the  $\gamma$ -quasi-clique value for strong friends sub graph for  $i^{\text{th}}$  community and  $q_i [w_{u_i}]$  denotes the  $\gamma$  -quasi-clique value for weak friends sub graph for  $i^{\text{th}}$  community.  $n_i$  denotes the number of nodes for each sub graph and  $g_i$  denotes the corresponding subgraph.

#### Community recommendation algorithm (CRA):

**Input:** (i) Friend database (FDB), (ii) user specified minimum strength value ( $\text{min}_{\text{str}}$ ), (iii) Community database (CDB)

**Output:** A recommended community ( $c_i$ )

1. If no friends are connected with the community then  $q_i = 0$
2. Else if  $q_i [s_{ui}] > q_{i+1} [s_{ui}]$  and  $n_i [s_{ui}] > n_{i+1} [s_{ui}]$
3.           Then  $c_i = \text{Recommended}$
4. Else if  $n_i [s_{ui}] < n_{i+1} [s_{ui}]$  and  $g_i [s_{ui}] \subset g_{i+1} [s_{ui}]$
5.           Then  $c_{i+1} = \text{Recommended}$
6. Else if  $q_{i+1} [s_{ui}] > q_i [s_{ui}]$  and  $n_{i+1} [s_{ui}] > n_i [s_{ui}]$
7.           Then  $c_{i+1} = \text{Recommended}$
8. Else if  $n_{i+1} [s_{ui}] < n_i [s_{ui}]$  and  $g_{i+1} [s_{ui}] \subset g_i [s_{ui}]$
9.           Then  $c_i = \text{Recommended}$
10. Else if  $q_i [s_{ui}] = q_{i+1} [s_{ui}]$  then
11.           If  $q_i [w_{ui}] > q_{i+1} [w_{ui}]$  and  $n_i [w_{ui}] > n_{i+1} [w_{ui}]$
12.           Then  $c_i = \text{Recommended}$
13.           Else if  $n_i [w_{ui}] < n_{i+1} [w_{ui}]$  and  $g_i [w_{ui}] \subset g_{i+1} [w_{ui}]$
14.           Then  $c_{i+1} = \text{Recommended}$
15.           Else if  $q_{i+1} [w_{ui}] > q_i [w_{ui}]$  and  $n_{i+1} [w_{ui}] > n_i [w_{ui}]$

16. Then  $c_{i+1} = \text{Recommended}$
17. Else if  $n_{i+1}[w_{ui}] < n_i [w_{ui}]$  and  $g_{i+1}[w_{ui}] \subset g_i [w_{ui}]$
18. Then  $c_i = \text{Recommended}$
19. Else if  $q_i [s_{ui}] = q_{i+1} [s_{ui}]$  and  $q_i [w_{ui}] = q_{i+1} [w_{ui}]$  and  $n_i = n_{i+1}$
20. Then calculate the  $\sum n_{i s_{ui}, f_j}$  for each  $c_i$
21. IF for strong friend sub graph,
22.  $c_i [\sum n_{i s_{ui}, f_j}] > c_{i+1} [\sum n_{i s_{ui}, f_j}]$
23. Then  $c_i = \text{Recommended}$
24. Else If  $c_{i+1} [\sum n_{i s_{ui}, f_j}] > c_i [\sum n_{i s_{ui}, f_j}]$
25. Then  $c_{i+1} = \text{Recommended}$
26. Else If  $c_i [\sum n_{i s_{ui}, f_j}] = c_{i+1} [\sum n_{i s_{ui}, f_j}]$
27. Then for weak friend sub graph
28. If  $c_i [\sum n_{i s_{ui}, f_j}] > c_{i+1} [\sum n_{i s_{ui}, f_j}]$
29. Then  $c_i = \text{Recommended}$
30. Else if  $c_{i+1} [\sum n_{i s_{ui}, f_j}] > c_i [\sum n_{i s_{ui}, f_j}]$

31. Then  $c_{i+1}$  =Recommended

32. Else both  $c_i$  and  $c_{i+1}$  recommended.

According to our CRA algorithm for community C1 in Figure-IV the  $\gamma$  -quasi-clique value for strong friend sub graph is 0.6 and the number of nodes is 4 and in Figure-V for community C2 the  $\gamma$  -quasi-clique value is 0.5 and the number of nodes is 3. And it full fills CRA's first condition. So C1 is the selected community for the user.

# Chapter 8

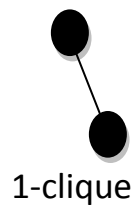
---

## Experimental Results

We made a survey on amount of hundred friends in Facebook. And there we found some interesting problem. Thus we bring out the concept of subgraph in our algorithm. Such cases are given below:

### Community-C11

#### CASE-1



### Community-C12

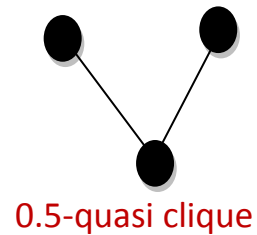


Figure 8.1: ExperimentalCase-1

#### CASE-2

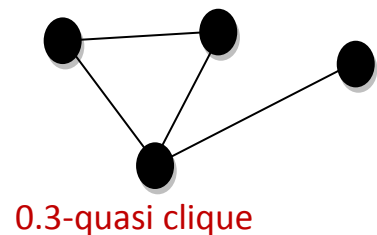
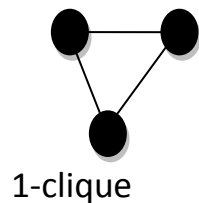
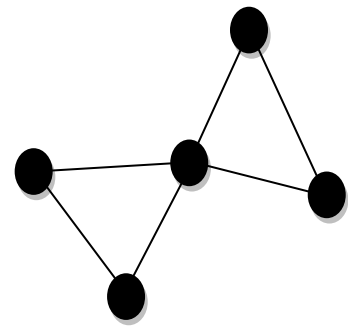
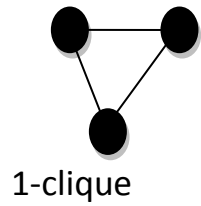


Figure 8.2: ExperimentalCase-2



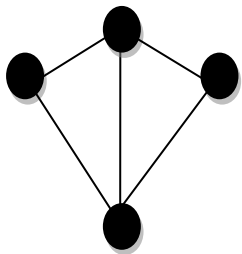
### CASE-3



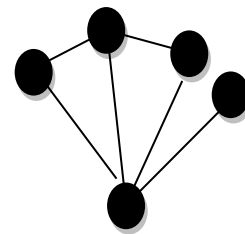
0.5-quasi clique

Figure 8.3: ExperimentalCase-3

### CASE-4



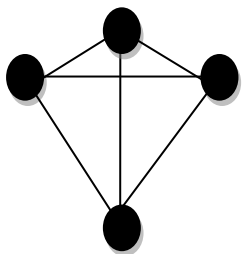
0.6-quasi clique



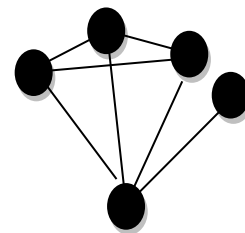
0.25-quasi clique

Figure 8.4: ExperimentalCase-4

### CASE-5



1-clique



0.25-quasi clique

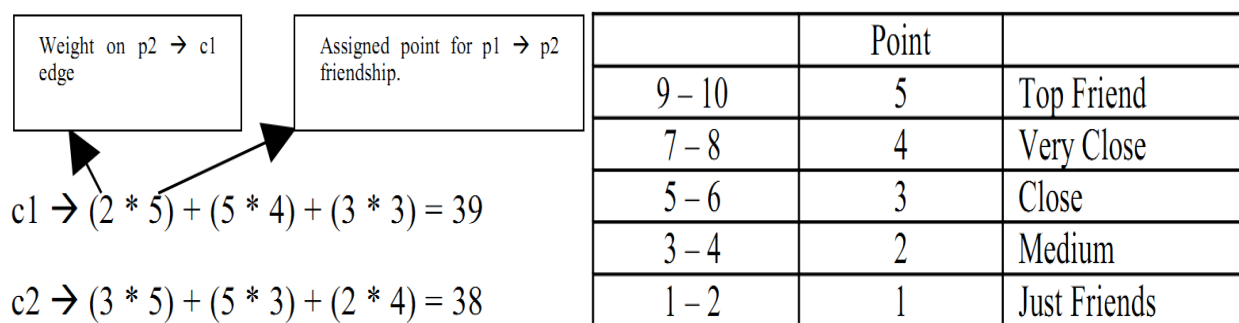
Figure 8.5: ExperimentalCase-5

For **case-4**, community C11 the value of  $\gamma$  -quasi-clique is 0.6 but for community C12 the value of  $\gamma$  -quasi-clique is 0.25. If we only consider the  $\gamma$  -quasi-clique value then the resultant community is C11. But it seems that C12 is more connected. To solve this problem in our algorithm, we introduce comparing sub graph. When the  $\gamma$  -quasi-clique value of one sub graph is greater than the sub graph of other community, we count the number of nodes of that sub graph and compare with the other ones nodes. If the sub graph which has higher  $\gamma$  -quasi-clique value and has more nodes than the other one, it is recommended. But if the number of nodes is less than the other one, we check whether it's a sub graph of other one or not. If it is, then we recommend the community which has lower quasi-clique value. Here C12 is a super graph of C11 though having lower  $\gamma$  -quasi-clique value. So we will recommend C12 for the user. As we work on a survey of the users of Facebook, we have found our algorithm more accurate. For the fresh start problem the solution will be the user opinion. Whenever a user connects with one or more friends than using our algorithm we can recommend him some communities according to his activities or interactions with his friends.

# Chapter 9

## Evaluation

In this paper, we proposed an algorithm to recommending a community to a particular social user. For that, we take many step like normalization, recognizing  $s_{ui}$  and  $w_{ui}$ , create dense sub graph etc. then we apply our algorithm. In [1] (Social Graph Generation & Forecasting using Social Network Mining), they build their algorithm based on specific points table (Table 9.1) and they propose an equation (Figure 9.1).which is used in selecting a community. Their approach is simple but this is not helpful for big social network like Facebook or Twitter or LinkedIn etc. In their paper, they introduce activeness which is very good approach, we didn't see any active use of it. In their equation, they only use weigh and Assigned point of table (Table 9.1).



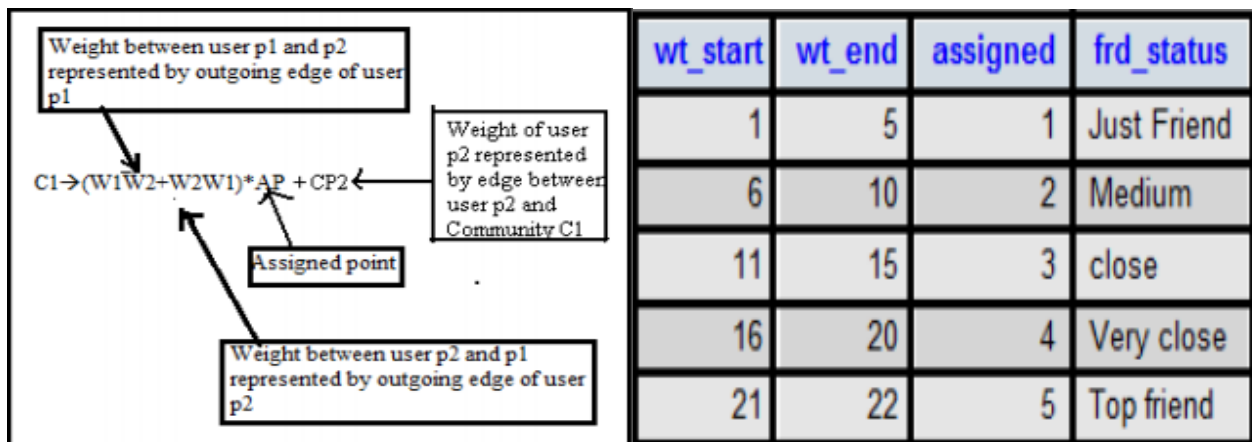
**Figure 9.1-** Equation of Algorithm [1]

**Table 9.1-** Specific point of table

At the early stage, their algorithm will work as they proposed but when the interaction increases and weight increases then many of the friends will be in

same category. At that stage their algorithm is insufficient. This algorithm can't handle large data set.

In Algorithm of "Graph Based Forecasting for Social Networking Site" [2], same thing reflect here again. They also have specific points table. They named it as Strength of Relationship (Table 9.2).They introduce weight in this algorithm. For specific weight limits, they assigned a value. Based on this value, they presents an equation(figure 9.2). Still they will face same problem, when the interaction increases and weight increases then many of the friends will be in same category.This algorithm also can't handle large data set.



**Table 9.2**-Rule based mining

**Figure 9.2:** Strength relationship

But according our approach we deal with the friendship strength by a different technique. If we see **Table-7.1** (Normalization) we see that if the interaction increases there is no problem with dividing them in strong and weak subgroups. And our mining process which is fully graph based and we used established graph mining techniques which are capable of dealing with large data set which we will explore later. We also consider the relation among friends of a user.

## Chapter 10

---

### Conclusion

Data processing takes time. A huge dataset represented in a database may not be as understandable as a visualized dataset like a graph. When the dataset is converted to a graph, processing of data becomes easier and more feasible due to a higher level of understandability. By only creating some mathematical rules applied on the graph, we could easily project the future activities of users in terms of community memberships. This is only a small subset of all the processing which can be performed on the generated graph. Marketing agents can take advantage of such predictions. Today lots of research is being carried out on machine learning and understandability. The integration of data mining and machine learning is still an open area to work on. The rapid growth and exponential use of social digital media has led to an increase in popularity of social networking sites and services such as Facebook, Google+, LinkedIn and Twitter. Groups of individual users of these sites form social networks. Taking

Idea from graph mining and machine learning, we build our algorithm to recommend a community to a social user. As ongoing work, we plan to simplify. Our algorithm and add experimental results on real social network data.

# Coding part

A few portions of implementation is given below:

## Normalization

```
253
254     //normalization
255     double finalScore = 0.0;
256     for(int i = 0; i < user1.friendList.size(); i++){
257         finalScore = finalScore + user1.friendList.get(i).getScore();
258
259     }
260
261
262     for(int i = 0; i < user1.friendList.size(); i++){
263         tempScore = user1.friendList.get(i).getScore();
264         user1.friendList.get(i).setScore(tempScore / finalScore);
265         // System.out.println(user1.friendList.get(i).getScore());
266     }
267
```

## Cumulative Score

```
270
271     //calculating cumulative score
272
273     for(int j = 0; j < user1.friendList.size(); j++){
274         cumulativeScore = cumulativeScore + user1.friendList.get(j).getScore();
275         if (cumulativeScore < minStrength) {
276             user1.strong.add(user1.friendList.get(j));
277             user1.friendList.get(j).setLinkStrStrong(1);
278             //System.out.println(user1.strong.get(j).getName());
279
280         }
281
282         else {
283
284             user1.weak.add(user1.friendList.get(j));
285             user1.friendList.get(j).setLinkStrWeak(1);
286
287         }
288     }

```

# Strong Friends

```
469 for(int i=0;i<user1Strong.length;i++){
470
471     if(user1Strong[i] != null){
472
473         for(int j=0;j<mainUser.user.size();j++){
474
475             if(user1Strong[i]==mainUser.user.get(j).getUserName()) {
476
477                 e=i;
478
479                 for(int k=0; k< mainUser.user.get(j).friendList.size(); k++){
480
481                     for(int l=0; l<user1Strong.length;l++){
482
483                         if(l !=e){
484
485                             if(mainUser.user.get(j).friendList.get(k).getName()==user1Strong[l]
486                                 && mainUser.user.get(j).friendList.get(k).getName() != user1.getUserUsername()){
487
488                                 for(int m=0; m<user1.strong.size(); m++) {
489
490                                     if(mainUser.user.get(j).getUserName().equals(user1.strong.get(m).getName())){
491
492                                         user1.strong.get(m).setLinkStrStrong(user1.strong.get(m).getLinkStrStrong() + 1);
493                                         // System.out.println(user1.strong.get(m).getLinkStrStrong()+ " "+ user1.strong.get(m).getName());
494                                         tempStrong=user1.strong.get(m).getLinkStrStrong();
495
496                                     }
497
498                                 }
499
500                             }
501                         }
502                     }
503                 }
504             }
505         }
506     }
507 }
```

# Weak Friends

```
563
564 for(int i=0;i<user1Weak.length;i++){
565
566     if(user1Weak[i] != null){
567
568         for(int j=0;j<mainUser.user.size();j++){
569
570             if(user1Weak[i]==mainUser.user.get(j).getUserName()) {
571
572                 g=i;
573
574                 for(int k=0; k< mainUser.user.get(j).friendList.size(); k++){
575
576                     for(int l=0; l<user1Weak.length;l++){
577
578                         if(l !=g){
579
580                             if(mainUser.user.get(j).friendList.get(k).getName()==user1Weak[l]
581                                 && mainUser.user.get(j).friendList.get(k).getName() != user1.getUserUsername()){
582
583                                 for(int m=0; m<user1.weak.size(); m++) {
584
585                                     if(mainUser.user.get(j).getUserName().equals(user1.weak.get(m).getName())){
586
587                                         user1.weak.get(m).setLinkStrWeak(user1.weak.get(m).getLinkStrWeak()+1);
588                                         // System.out.println(user1.weak.get(m).getLinkStrWeak()+ " "+ user1.weak.get(m).getName());
589                                         tempWeak=user1.weak.get(m).getLinkStrWeak();
590
591                                     }
592
593                                 }
594
595                             }
596                         }
597                     }
598                 }
599             }
600         }
601     }
602 }
```

## Quasi-clique Value for C1 (Strong)

```
539
540 for(int i=0; i<user1Strong.length;i++){
541
542     if(user1Strong[i] != null){
543
544         validUser1Strong++;
545         c1_strong_no++;
546     }
547
548 }
549
550
551
552 qclique_c1_strong= lowestLinkStrong / Math.ceil(validUser1Strong);
553 System.out.println("Strong friend quasi clique value for community c1"+ " "+ qclique_c1_strong);
554
555 //strong friend for community c1 done here
556
```

## Quasi-clique Value for C1 (Weak)

```
640 for(int i=0; i<user1Weak.length;i++){
641
642     if(user1Weak[i] != null){
643
644         validUser1Weak++;
645         c1_weak_no++;
646     }
647
648 }
649
650 qclique_c1_weak= (lowestLinkWeak / validUser1Weak);
651 System.out.println("Weak Friend quasi clique value for community c1"+ " "+ qclique_c1_weak);
652
653 //weak friend for community c1 done here
```



# Comparison

```
1136
1137 //comparison
1138
1139 if(qclique_c1_strong> qclique_c2_strong && c1_strong_no> c2_strong_no) System.out.println("C1 is recommended");
1140 else if(c2_strong_no > c1_strong_no && deg_frn_str_c2_1>deg_frn_str_c1_1 &&deg_frn_str_c2_2>deg_frn_str_c1_2 && deg_frn_str_c2_3>= deg_frn_str_c1_3
1141 else if(qclique_c2_strong> qclique_c1_strong && c2_strong_no> c1_strong_no) System.out.println("C2 is recommended");
1142 else if(c1_strong_no > c2_strong_no && deg_frn_str_c1_1 >= deg_frn_str_c2_1 && deg_frn_str_c1_2 >= deg_frn_str_c2_2 && deg_frn_str_c1_3>deg_frn_str_c
1143 else if(qclique_c1_strong == qclique_c2_strong){
1144     if(qclique_c1_weak> qclique_c2_weak && c1_weak_no > c2_weak_no) System.out.println("C1 is recommended");
1145     else if(c2_weak_no > c1_weak_no && deg_frn_weak_c2_1>= deg_frn_weak_c1_1 && deg_frn_weak_c2_2 >= deg_frn_weak_c1_2 && deg_frn_weak_c2_3 >= deg_frn
1146     else if(qclique_c2_weak > qclique_c1_weak && c2_weak_no > c1_weak_no) System.out.println("C2 is recommended");
1147     else if(c1_weak_no > c2_weak_no && deg_frn_weak_c1_1 >= deg_frn_weak_c2_1 && deg_frn_weak_c1_2 >= deg_frn_weak_c2_2 && deg_frn_weak_c1_3 >= deg_fr
1148
1149
1150 }
1151 else if(qclique_c1_strong == qclique_c2_strong && qclique_c1_weak == qclique_c2_weak){
1152 }
1153 }
```

# Result

```
"C:\Program Files (x86)\Java\jdk1.6.0_12\bin\java" -Didea.launcher.port=7533 "-Didea.launcher.bin.path=C:\Program Files (x86)\JetBrains\IntelliJ IDEA Community Edition 12.1.6\b
Strong friend quasi clique value for community c1 0.6666666666666666
Weak Friend quasi clique value for community c1 0.6666666666666666
Strong friend quasi clique value for community c2 0.5
Weak Friend quasi clique valu for community c2 0.6666666666666666
C1 is recommended
Process finished with exit code 0
```

# Bibliography

[1] Marjaneh Safaei, Merve Sahan and Mustafa Ilkan, "Social Graph Generation & Forecasting using Social Network Mining," 33rd Annual IEEE International Computer Software and Applications Conference (COMPSAC'09), 2009, pp. 31-35.

[2] Kadge, Sanam and Bhatia, Gresha, "Graph Based Forecasting For Social Networking Site", Communication, Information & Computing Technology (ICCICT) International Conference, 2012, , pp. 1-6.

[3] Cameron, Juan J and Leung, CK-S and Tanbeer, Syed Khairuzzaman, "Finding Strong Groups of Friends among Friends in Social Networks", Ninth IEEE International Conference, 2011, pp. 824-831.

[4] Jiang, Fan and Leung, CK-S and Tanbeer, Syed Khairuzzaman, "Finding Popular Friends in Social Networks", Second International Conference on Cloud and Green Computing, 2012, pp. 501-508.

[5] P.M. Zadeh, M.S. Moshkenani, "Mining Social Network for Semantic Advertisement", Third 2008 International Conference on Convergence and Hybrid Information Technology (ICCIT'08), 2008, Volume 1, pp. 11-13.

[6] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in Proc. ACM SIGMOD 1993, Volume 22, pp. 207–216.

**[7]** Silva, Arlei and Meira Jr, Wagner and Zaki, Mohammed J, “Structural correlation pattern mining for large graphs”, MLG '10 Proceedings of the Eighth Workshop on Mining And .Learning with Graphs, 2010.

**[8]**S. Mitra, A. Bagchi, A.K.Bandyopadhyay,“Complex Queries on Web Graph representing a Social Network”, 1st International Conference on Digital Information Management, IEEE Press, Dec. 2006, pp. 430-435.

**[9]** M.E.J. Newman, “Detecting community structure in networks”. The European Physical Journal B - Condensed Matter and Complex Systems (2004), Volume 38, Issue 2, pp. 321-330.

**[10]** Himel Dev, Mohammed Eunus Ali, Tanzima Hashem, “User Interaction Based Community Detection in Online Social Networks”, 19th International Conference, DASFAA 2014, pp. 296-310.

**[11]**Aaron Clauset, M. E. J. Newman, and Cristopher Moore, “Finding community structure in very large networks”, The American Physical Society, 2004,Phys. Rev. E 70, 066111.

**[12]**Qi, G.J., Aggarwal, C.C., Huang, T.S.: Community detection with edge content in social media networks. In: ICDE, pp. 534–545 (2012)