Software Engineering Department
Ort Braude College
Course 61401: Extended Project in Software Engineering

Detecting unexplained human behaviors in Social Networks

Final Project in Software Engineering (course 61401)

Ort Braude College

Karmiel – January 2016

Matatov Evgeni 323576009

Kayumov Victor 314357559

Supervisors:

Prof. Kirzner Valery

Dr. Avros Renata

Contents

1. INTRODUCTION

Abstract: More and more social network users rely on public information, such as shared posts, reviews, ads and more. This led to the growth of the techniques for spam and malicious software spreading. User's private information also becomes vulnerable to the Sybil attack: the situation where an attacker forges many identities, each called a Sybil, and joins a target system for various advertising objectives and for even more harmful activity.

In our project we survey the most popular Sybil attacks in social networks. Different techniques are being applied to distinguish potentially bad behavior from normal behavior. We test some defensive schemes against Sybil activity in order to find optimal solution for the problem.

Keywords: Sybil detector models, abnormal behavior in social networks, Fast unfolding algorithm, Sybil attacks

1.1. Anomalous user behavior in social networks

Today’s online social network systems are open; any user can join the system by providing an identity that is issued by either the system itself or by a trusted third-party.

In identity-based systems (Facebook, Twitter, LinkedIn) each user is intended to have a single profile and is expected to use it when interacting with other accounts in the system. Without strong identity binding social network systems are quite vulnerable.

Sybil attackers behave similarly to normal users. To find out whether an account is Sybil or not is not an easy task. Moreover, when the attackers change strategy, the detection of abnormal behavior becomes extremely difficult. Therefore, this makes the defense against Sybil of great importance in social and information systems.

The project's aim is to verify the Sybil activity detection schemes in social networks.

2. THEORY

Driven by similar interests, social users could form a virtual online community or society in order to exchange information and share multimedia resources. Here Sybil activity comes out.

In this chapter, several types of Sybil attacks are presented.

2.1. Graph representation

In order to understand Sybil means for attacks, a social graph model is presented. We build an indirect social graph denoted as G with n nodes and E edges. We denote by H the set of honest nodes and Sybil nodes by S. The object instances or users in social graph are represented by nodes. The edge between every pair of nodes is an integer value between 1 and N, which is characterized by social relationships between accounts.

For each identity we consider a vector V of properties with ratings. For each pair of different nodes i, j function f converts two vectors V1 and V2 into real number w that assumed to be an edge weight. Nodes that are tightly interconnected define community. Such groups or communities have several properties. The clustering coefficient (cc) – is a parameter that reflects the closeness of nodes within social network [1].

Two or more tightly connected communities define cluster structure. Each community has several properties such as density or popularity distribution coefficient D, conductance coefficient C [2].

To understand the behavior of the Sybil, we describe several types of attacks.

2.2. Types of Sybil attacks

Sybil attacks exist to maliciously manipulate the systems. In this section, two types of Sybil attacks: SA-1 and SA-2 are described.

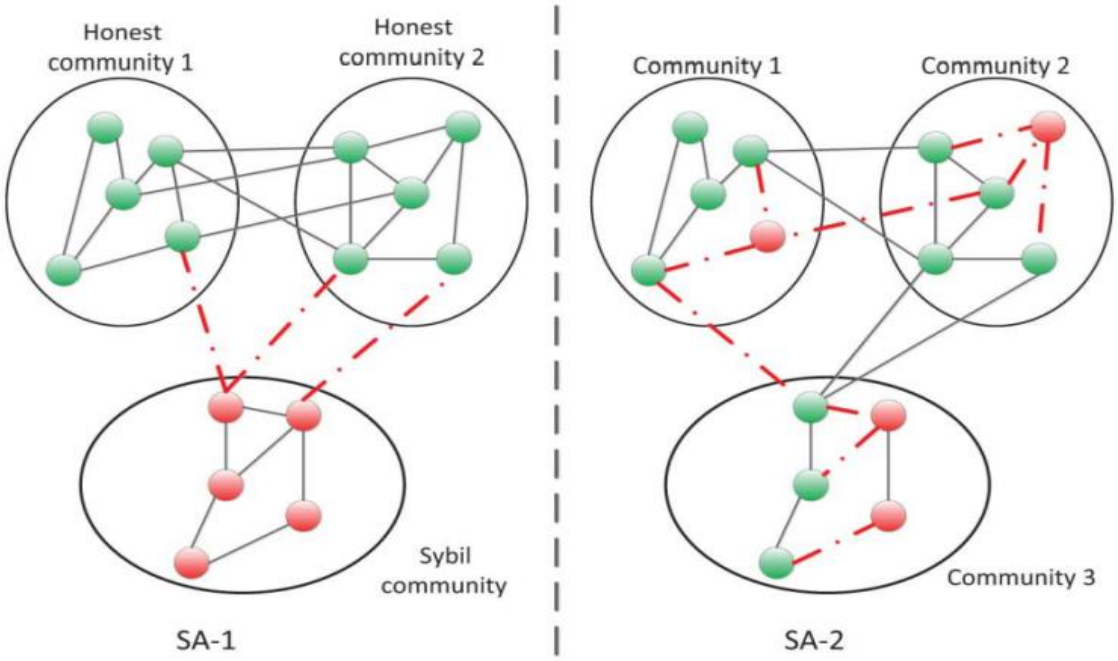


Fig. 1 Types of Sybil attacks

2.2.1. Sybil Attack SA-1

The SA-1 attackers usually build connections within the Sybil community as shown in Fig. 1, i.e., Sybil nodes tightly connect with other Sybil nodes. However, the SA-1's capability of building social connections with honest nodes is not strong. The number of social connections between Sybil nodes and honest ones (attack edges) is limited.

One of the main goals of SA-1 attacks can be intended for various network manipulations. For example, in an online voting system, SA-1 can illegally create a massive number of identities to act as normal users and submit the fake votes. The final voting result might be manipulated by the SA-1attackers, since a considerable portion of votes are from the SA-1 attackers.

Therefore, in some cases, the behaviors of Sybil attackers are indistinguishable from the normal users.

2.2.2. Sybil Attack SA-2

Unlike SA-1, SA-2 is able to create the social connections not only among Sybil identities but also with the normal users. In other words, the ability of SA-2 to simulate the normal user's social structures in the social graph is strong. Therefore, is very difficult to distinguish a SA-2 by using social graph partition.

The goal of SA-2 is to spread spam, advertisements, and malware; steal and violate user's privacy; and maliciously manipulate the reputation system.

In addition, SA-2 could generate a lot of positive or negative review comments in the service evaluation systems to overrate or underestimate them.

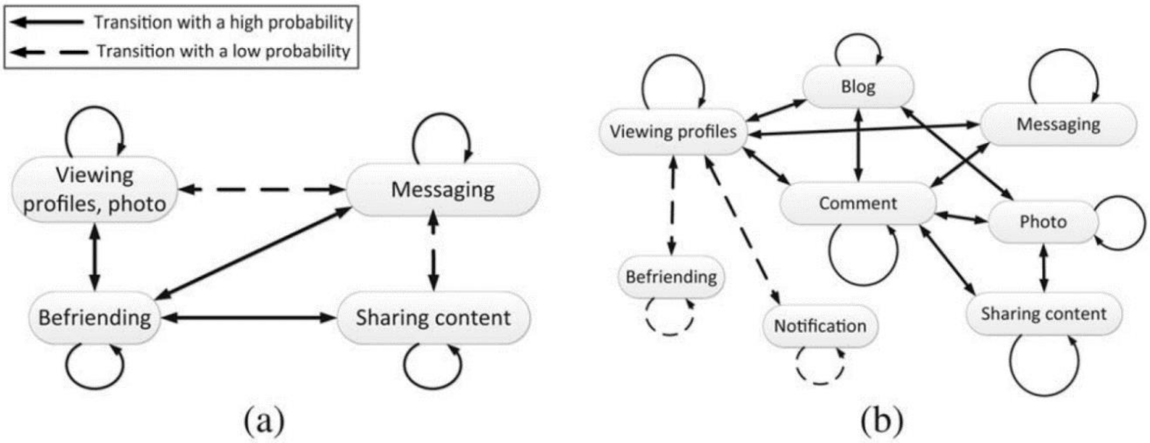


Fig.2 Online social networking's behaviors and transition probabilities of Sybil attackers and normal users.

(a) State transitions for a Sybil user, (b) State transitions for a normal user.

Categories of Sybil attacks	Social graph features	Attack goal	Behavior discrimination
SA-1	Sybil exist in the same region of community, and the number of attack edges is limited	Maliciously or purposely upload the biased reports or comments (positive or negative) to manipulate the overall option and dominate the whole system	Perform as the normal users, and repeat specific behaviors frequently
SA-2	Sybil may tightly connect with normal users, and generate more attack edges	Disseminate spam and malware to launch some other attacks, camouflage as normal users, or violate others users' privacy	Purposely repeat some specific behaviors in the high frequency

Table 1. Sybil attacks

3. SYBIL DETECTOR SCHEMES

The Sybil accounts can be identified by using of the following schemes:

1) Social graph-based Sybil detection scheme (SGSD),

2) Behavior classification-based Sybil detection scheme (BCSD).

3.1. Social Graph-based Sybil Detection model

In this chapter, the Social Graph-based Sybil Detection (SGSD) scheme is described.

We guess Sybil is much more aggressive in sending requests than normal users. For example, invitation frequency can be used to detect a Sybil behavior.

Another distinguishing feature is the fraction of outgoing friend requests confirmed by the recipient. Non-Sybil users generally have high accepted ratios, normal users typically send invites to people with whom they have prior relationships, whereas Sybil target strangers. The incoming requests accepted by non-Sybil users are spread across the board. In contrast, Sybil accounts are nearly uniform in that they accept all incoming friend requests.

Since normal users tend to have a small number of well-connected social cliques, we expect them to have much higher cc values than Sybil accounts.

3.1.1 Social Community-Based Sybil Detection

The possibility of using social community detection algorithms to detect SA-1 is validated in [3]. The main idea of this model is to analyze interaction among communities in order to distinguish between Sybil groups and honest ones.

For example, Cai and Jermaine [4] used the latent community model and machine learning to detect Sybil attacks. Even though some certain communities are compromised by Sybil attackers, the attack communities can also be detected. By using multi-community social network structure, Shi [5] proposed SybilShield, an agent-aided SCSD scheme. SybilShield use trust relationships among users to form the social graph. However, due to the fact that two honest nodes belonging to the two different social communities may not tightly connect with each other, SybilShield deploy the agents and ensures the honest nodes tightly connect with other honest ones. In [5], the first random walk algorithm [6] is adopted as SybilGuard. Then, some agents of a verifier are selected to run another round of random walk, called agent walk, where the agents pass over all of the verifier’s edges to confirm the suspect nodes. SybilShield relies on Assumption.

Assumption: Sybil nodes cannot tightly connect with honest nodes in the multiple honest communities since honest nodes would not trust Sybil ones. Honest nodes can tightly inter-connect with others in the honest community.

This protocol is based on the “social network” among user identities, where an edge between two identities indicates a human-established trust relationship. Malicious users can create many identities but few trust relationships.

3.1.2. Behavior Classification-Based Sybil Detection

Another detection scheme is the Behavior Classification-Based Sybil Detection (BCSD). It is known that Sybil users can create a lot of social connections with the honest users [7] and rarely set up social connections with other Sybil users. Therefore, only relying on the SGSD schemes can't effectively detect Sybil attacks since the Assumption may not always hold. Therefore, another approach for Sybil detection schemes is required. We observe users activity in social networks in order to distinguish the Sybil by comparing their anomalous behavior with the normal users. For example browsing and clicking habits described in [8]. According to the statistics, the primary activities of Sybil users are friending (especially, sending friend requests), viewing photos and profiles of others, and sharing contents with others.

The normal users spend a large part of online time to view photo, and perform other activities, such as viewing profiles, sending messages, sharing contents with a similar frequency. Both Sybil and normal users share content or send messages at similar frequencies. Sharing content or sending messages are the common approaches for Sybil to distribute spam in online social networks (OSNs). Normal users usually perform various OSN behaviors, and the transformations among states are really complicated (see diagram on fig 2). The Sybil users repeat some specific operation in a high frequency such as average clicks per session, average session length, and average inter-arrival time between two clicks. Empirical approach by analyzing network operations statistics could be examined and applied as an instrument for Sybil attack detection model. If Sybil attackers simulate almost the same click patterns or habits of normal user than the BCSD cannot effectively detect them as well.

However, such adaptation process will consume a lot of time to simulate the normal user's behavior.

In summary, these BCSD schemes can detect the Sybil attackers, according to the user's behavior learning and classification. So, this idea is used in our project implementation.

Sybil defense scheme	Type of Sybil attacks	Preliminary technique	Base\assumption
SCSD	SA-1	Community detection	Assumption
BCSD	SA-2	Behavior classification	Behavior difference

Table 2. Sybil detection: a comparison

3.2. Fast unfolding of communities in large networks

SDSG schemes could be applied using multi-community social network structure. Therefore we choose the Fast unfolding algorithm that can transform the graph into community-based structure. All nodes will be grouped into communities using common properties and relevant social connections. We apply algorithm that finds high modularity partitions of large networks in short time. It unfolds a complete hierarchical community structure for the network, thereby giving access to different resolutions of community detection. The modularity of a partition is a scalar value between 0 and 1. It measures the density of links inside communities as compared to links between communities. The algorithm is divided in two phases that are repeated iteratively. Assume that we start with a weighted network of N nodes [9, 10]. First, we assign a different community to each node of the network.

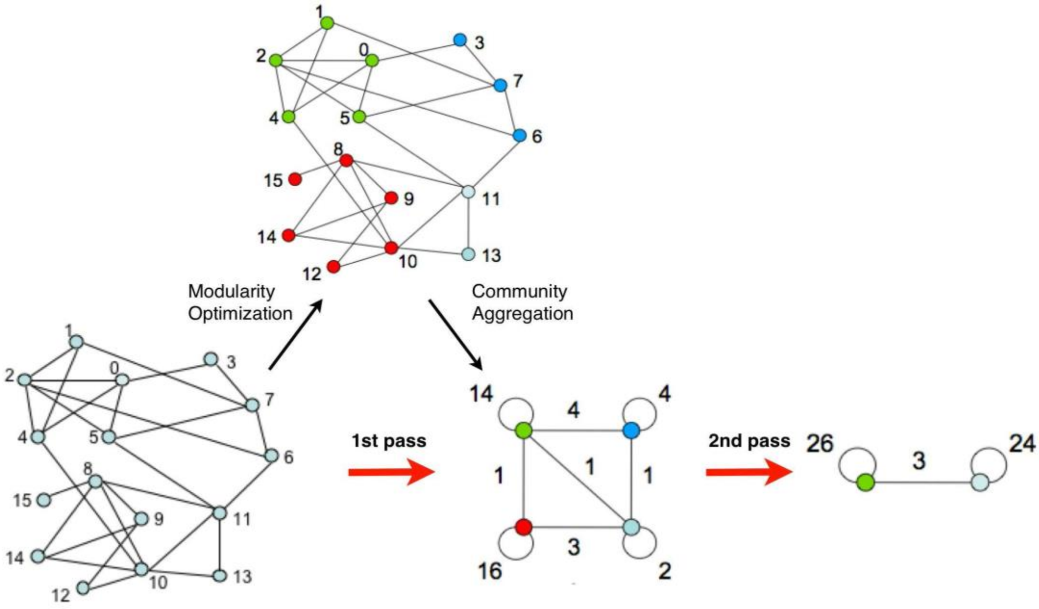


Fig.3 Fast unfolding algorithm of clustering

So, in this initial partition there are as many communities as there are nodes. Then, for each node i' we consider the neighbors j' of i' and we evaluate the improvement of modularity that would take place by removing i' from its community and by placing it in the community of j'. The node i' is then placed in the community for which this gain is maximum (in case of a tie we use a breaking rule), but only if this gain is positive. If no positive gain is possible, i' stays in its original community. This process is applied repeatedly and sequentially for all nodes until no further improvement can be achieved and the first phase is then complete [11]. This first phase stops when a local maximum of the modularity is attained, i.e. when no individual move can improve the modularity.

The second phase of the fast folding algorithm consists in the creation of a new network whose nodes (communities) found during the first phase [12]. To do so, the weights of the links between the new nodes are given by the sum of the weight of the links between nodes in the corresponding two communities.

Links between nodes of the same community lead to self-loops for this community in the new network.

After this second phase is completed, the first phase of the algorithm can start run again to the resulting weighted network and to iterate. The number of communities decreases at each pass.

The passes are iterated until there are no more changes and a maximum of modularity is attained.

3.3 Approach

We use database from network of Q&A site named academia.stackexchange.com. It is real data snapshot between dates 2012-02-01 and 2015-03-03. Archive includes users' information table, users' activities such as published posts, comments and votes archives in posts.

First we try to build suitable environment for applying Sybil detecting model. We build transformation function from database tables into network graph of nodes and edges. As was described before in 2.1 network users are represented by nodes and edges are integer numbers from 1 to N (edges weights). By choosing available parameters from database tables we build network graph.

Clustering process is the next step of transformation network graph into network of clusters or communities. Users in each community are grouped by connection strength between them.

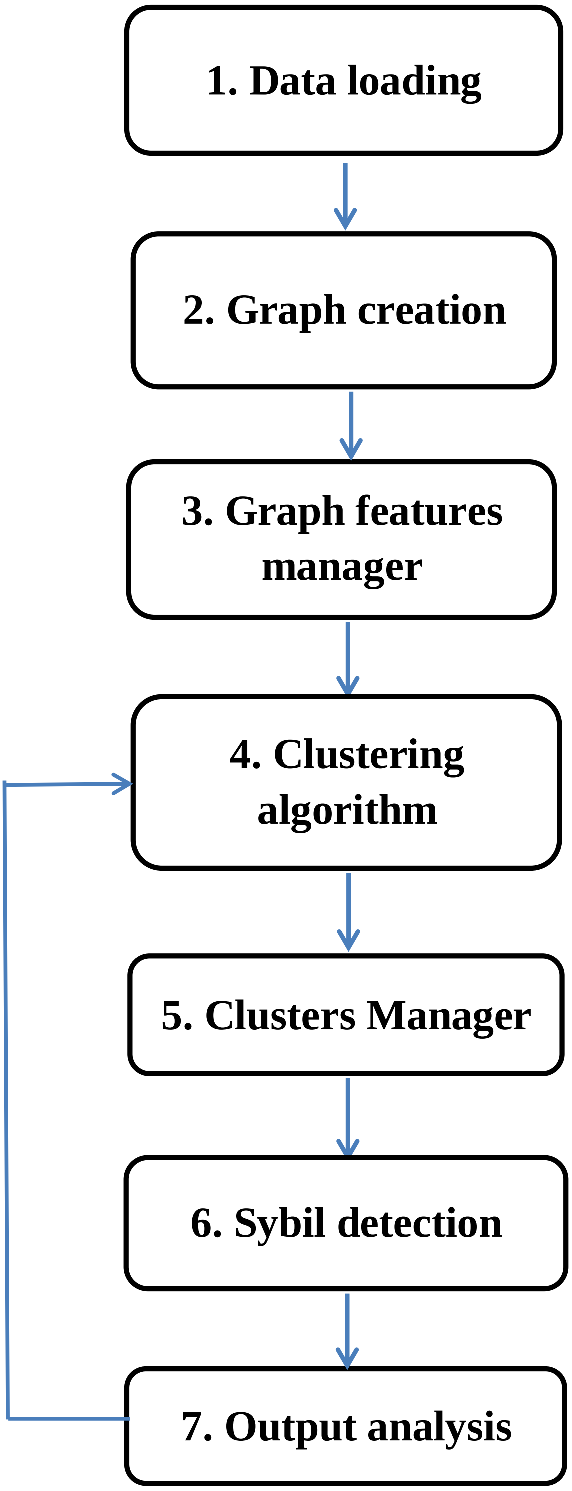
To characterize Sybil behavior, we identify several significant attributes that are unique to attacker, and use them to build a Sybil detector. In order to verify and test our detection model we added number of nonexistent users into database. The activity of all artificial users will simulate abnormal behavior of social network users.

We create different types of rules that can help to identify Sybil activity. One or more rules can be applied simultaneously.

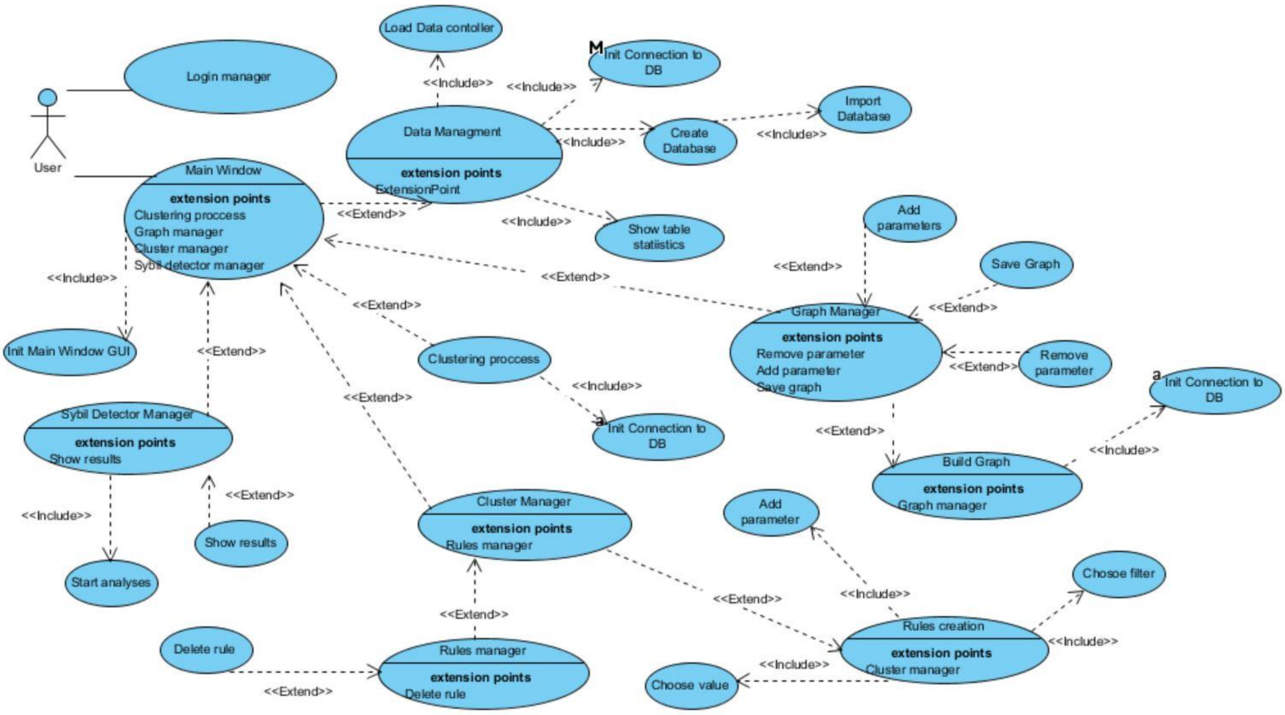
After individual analysis of rules, output file with results for each filter is created. We also present the visualization of several analysis results in our project.

4. PROJECT DEVELOPMENT

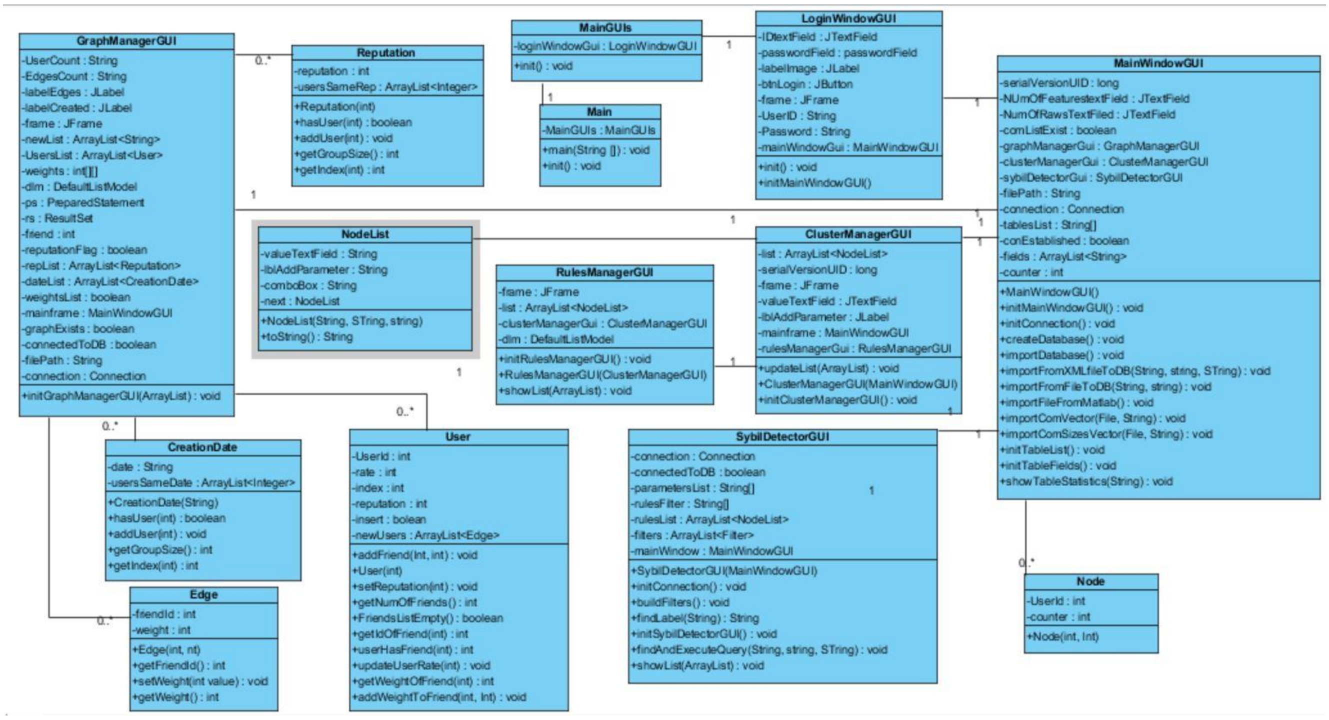
4.1. Flow chart



4.2. Use case diagram



4.3. Class diagram


GraphManagerGUI
-UserCount : String
-EdgesCount : String
-labelEdges : JLabel
-labelCreated : JLabel
-frame : JFrame
-newList : ArrayList<String>
-UsersList : ArrayList<User>
-weights : int[][]
-dlm : DefaultListModel
-ps : PreparedStatement
-rs : ResultSet
-friend : int
-reputationFlag : boolean
-repList : ArrayList< Reputation>
-dateList : ArrayList<CreationDate>
-weightsList : boolean
-mainframe : MainWindowGUI
-graphExists : boolean
-connectedToDB : boolean
-filePath : String
-connection : Connection
+initGraphManagerGUI(ArrayList) : void
Reputation
-reputation : int
-usersSameRep : ArrayList<Integer>
+Reputation(int)
+hasUser(int) : boolean
+addUser(int) : void
+getGroupSize() : int
+getIndex(int) : int
MainGUIs
-loginWindowGui : LoginWindowGUI
+init() : void
Main
-MainGUIs : MainGUIs
+main(String []) : void
+init() : void
LoginWindowGUI
-IDtextField : JTextField
-passwordField : passwordField
-labelImage : JLabel
-btnLogin : JButton
-frame : JFrame
-UserID : String
-Password : String
-mainWindowGui : MainWindowGUI
+init() : void
+initMainWindowGUI()
MainWindowGUI
-serialVersionUID : long
-NUmOfFeaturestextField : JTextField
-NumOfRawsTextFiled : JTextField
-comListExist : boolean
-graphManagerGui : GraphManagerGUI
-clusterManagerGui : ClusterManagerGUI
-sybilDetectorGui : SybilDetectorGUI
-filePath : String
-connection : Connection
-tablesList : String[]
-conEstablished : boolean
-fields : ArrayList<String>
-counter : int
+MainWindowGUI()
+initMainWindowGUI() : void
+initConnection() : void
+createDatabase() : void
+importDatabase() : void
+importFromXMLfileToDB(String, string, STring) : void
+importFromFileToDB(String, string) : void
+importFileFromMatlab() : void
+importComVector(File, String) : void
+importComSizesVector(File, String) : void
+initTableList() : void
+initTableFields() : void
+showTableStatistics(String) : void
NodeList
-valueTextField : String
-lblAddParameter : String
-comboBox : String
-next : NodeList
+NodeList(String, STring, string)
+toString() : String
RulesManagerGUI
-frame : JFrame
-list : ArrayList<NodeList>
-clusterManagerGui : ClusterManagerGUI
-dlm : DefaultListModel
+initRulesManagerGUI() : void
+RulesManagerGUI(ClusterManagerGUI)
+showList(ArrayList) : void
ClusterManagerGUI
-list : ArrayList<NodeList>
-serialVersionUID : long
-frame : JFrame
-valueTextField : JTextField
-lblAddParameter : JLabel
-mainframe : MainWindowGUI
-rulesManagerGui : RulesManagerGUI
+updateList(ArrayList) : void
+ClusterManagerGUI(MainWindowGUI)
+initClusterManagerGUI() : void
CreationDate
-date : String
-usersSameDate : ArrayList<Integer>
+CreationDate(String)
+hasUser(int) : boolean
+addUser(int) : void
+getGroupSize() : int
+getIndex(int) : int
Edge
-friendId : int
-weight : int
+Edge(int, nt)
+getFriendId() : int
+setWeight(int value) : void
+getWeight() : int
User
-UserId : int
-rate : int
-index : int
-reputation : int
-insert : bolean
-newUsers : ArrayList<Edge>
+addFriend(Int, int) : void
+User(int)
+setReputation(int) : void
+getNumOfFriends() : int
+FriendsListEmpty() : boolean
+getIdOfFriend(int) : int
+userHasFriend(int) : int
+updateUserRate(int) : void
+getWeightOfFriend(int) : int
+addWeightToFriend(int, Int) : void
SybilDetectorGUI
-connection : Connection
-connectedToDB : boolean
-parametersList : String[]
-rulesFilter : String[]
-rulesList : ArrayList<NodeList>
-filters : ArrayList<Filter>
-mainWindow : MainWindowGUI
+SybilDetectorGUI(MainWindowGUI)
+initConnection() : void
+buildFilters() : void
+findLabel(String) : String
+initSybilDetectorGUI() : void
+findAndExecuteQuery(String, string, STring) : void
+showList(ArrayList) : void
Node
-UserId : int
-counter : int
+Node(int, Int)
0..*
1

5. GUI APPLICATION DESIGN

5.1. System login

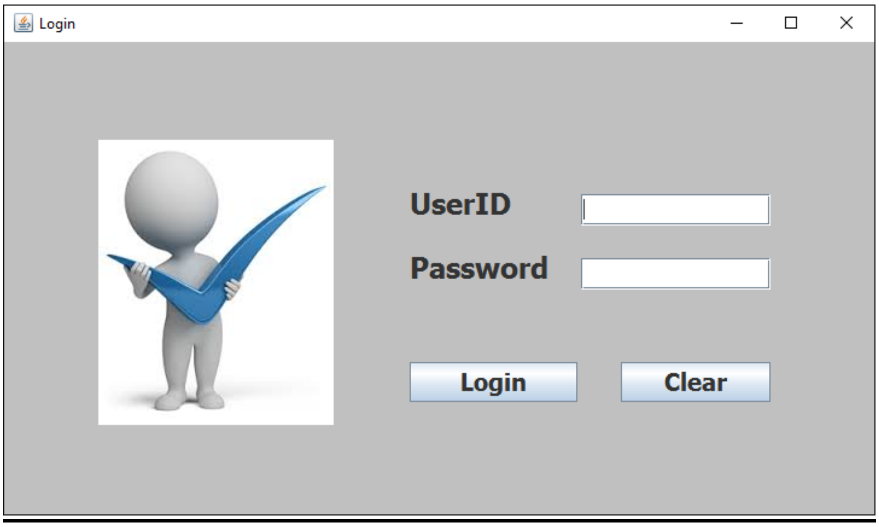


Fig. 4 Login window

First, in order to enter the system, the user presses the login button.

5.2. Main window

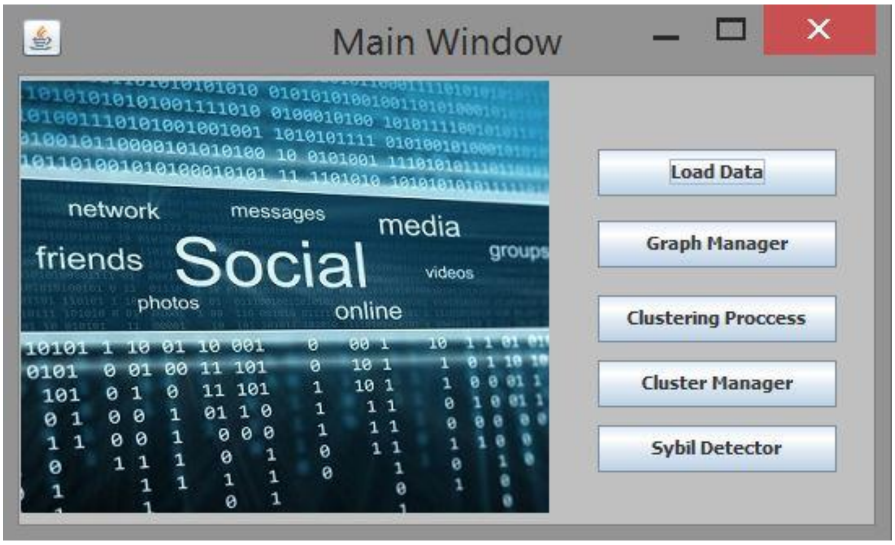
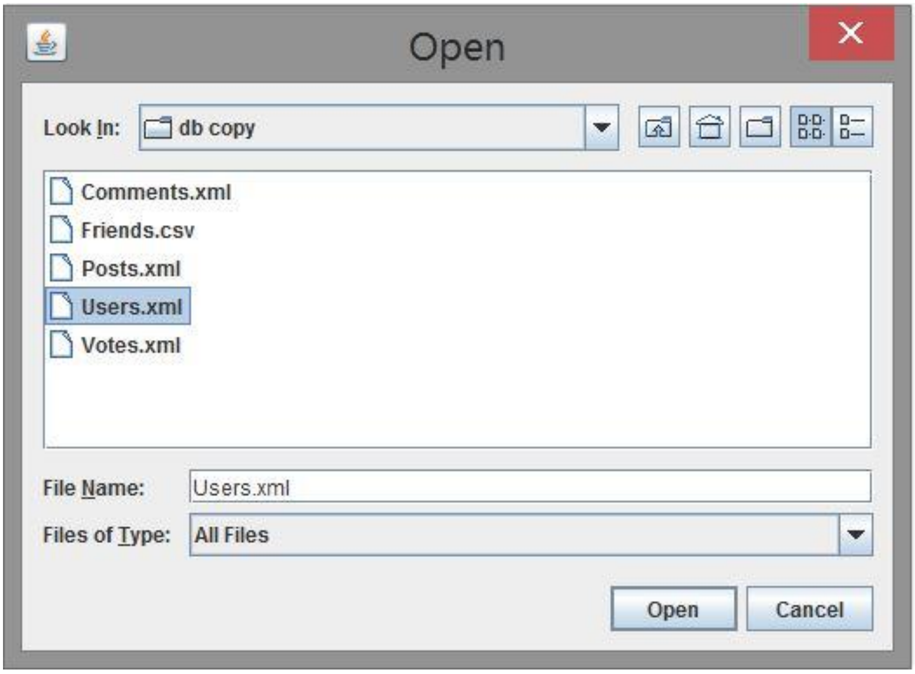


Fig. 5 Main window

- The user can import the data tables into MySQL server database.
- **Graph manager** allows building graph.
- **Clustering process** uses the fast unfolding algorithm to cluster network users by groups.
- **Cluster manager** allows creating rules/filters for further analysis of network users.
- **Sybil detector** performs analysis of network data by using the created rules.

5.3. Load data



Look In: db copy

Comments.xml
Friends.csv
Posts.xml
Users.xml
Votes.xml

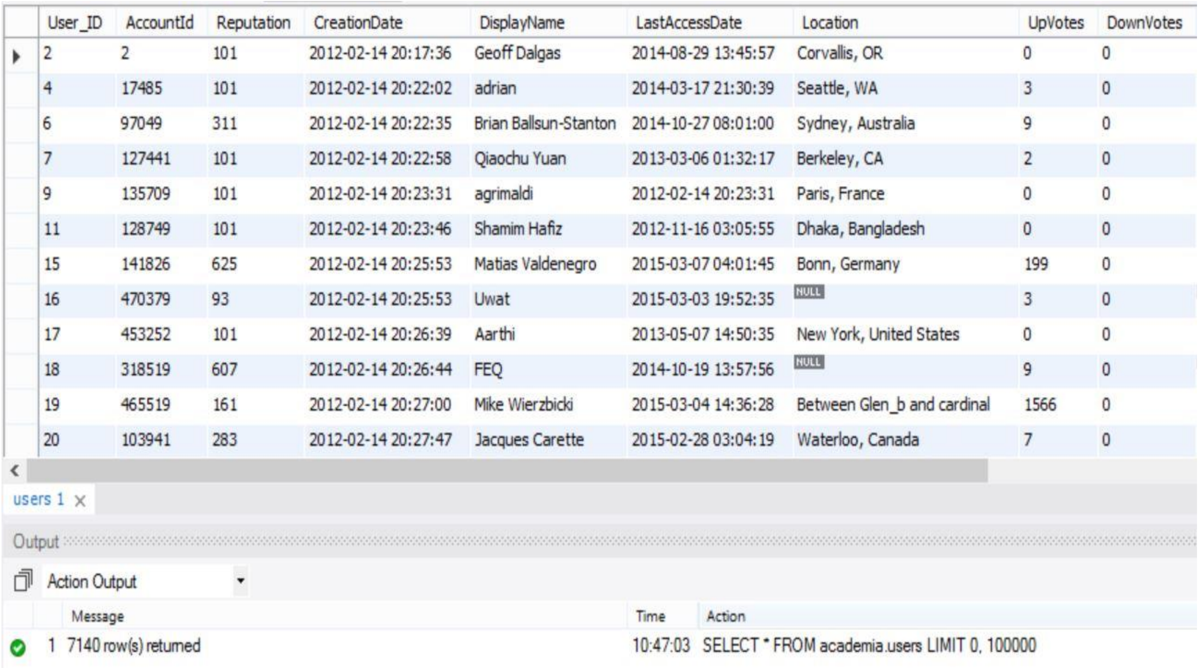
File Name: Users.xml

Files of Type: All Files

Fig. 6 Load data window

In order to import the data from external file, the user must press the "Load Data" button on the Main Window and choose the file. The details of all imported tables are described below.

Database table (Users)

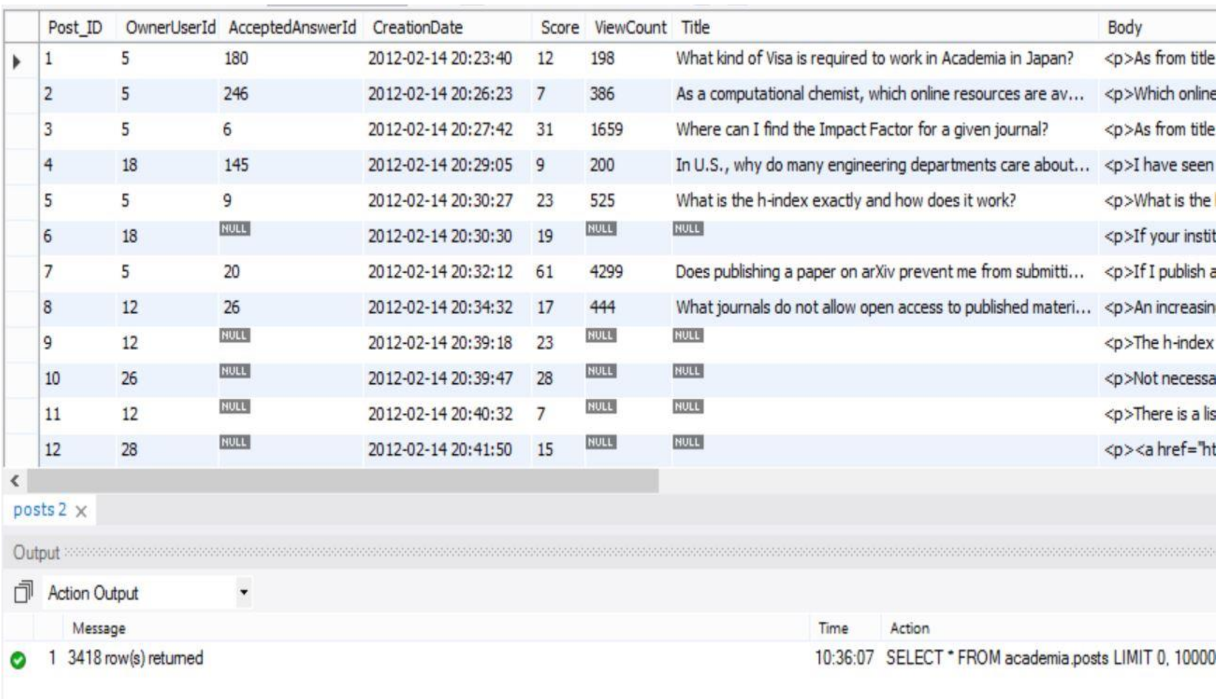


User_ID	AccountId	Reputation	CreationDate	DisplayName	LastAccessDate	Location	UpVotes	DownVotes
2	2	101	2012-02-14 20:17:36	Geoff Dalgas	2014-08-29 13:45:57	Corvallis, OR	0	0
4	17485	101	2012-02-14 20:22:02	adrian	2014-03-17 21:30:39	Seattle, WA	3	0
6	97049	311	2012-02-14 20:22:35	Brian Ballsun-Stanton	2014-10-27 08:01:00	Sydney, Australia	9	0
7	127441	101	2012-02-14 20:22:58	Qiaochu Yuan	2013-03-06 01:32:17	Berkeley, CA	2	0
9	135709	101	2012-02-14 20:23:31	agrimaldi	2012-02-14 20:23:31	Paris, France	0	0
11	128749	101	2012-02-14 20:23:46	Shamim Hafiz	2012-11-16 03:05:55	Dhaka, Bangladesh	0	0
15	141826	625	2012-02-14 20:25:53	Matias Valdenegro	2015-03-07 04:01:45	Bonn, Germany	199	0
16	470379	93	2012-02-14 20:25:53	Uwat	2015-03-03 19:52:35	NULL	3	0
17	453252	101	2012-02-14 20:26:39	Aarthi	2013-05-07 14:50:35	New York, United States	0	0
18	318519	607	2012-02-14 20:26:44	FEQ	2014-10-19 13:57:56	NULL	9	0
19	465519	161	2012-02-14 20:27:00	Mike Wierzbicki	2015-03-04 14:36:28	Between Glen_b and cardinal	1566	0
20	103941	283	2012-02-14 20:27:47	Jacques Carette	2015-02-28 03:04:19	Waterloo, Canada	7	0

	Message	Time	Action
1	7140 row(s) returned	10:47:03	SELECT * FROM academia.users LIMIT 0, 100000

Fig. 6.1 Database table Users

Database table (Posts)



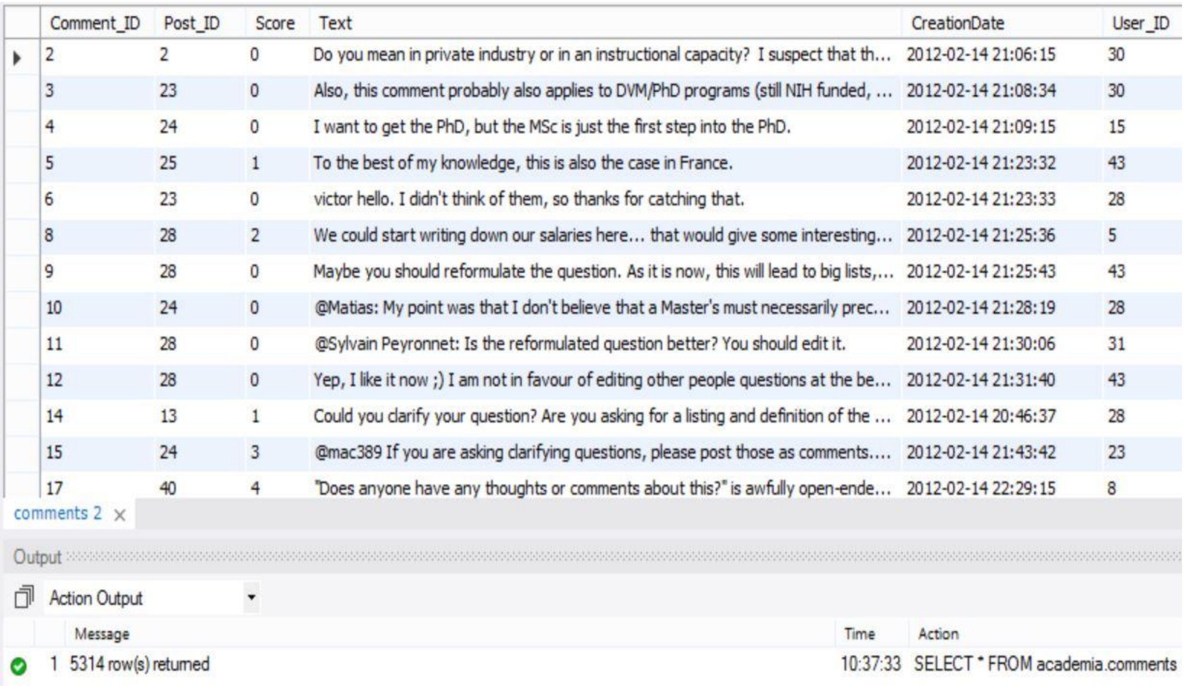
Post_ID	OwnerUserId	AcceptedAnswerId	CreationDate	Score	ViewCount	Title	Body
1	5	180	2012-02-14 20:23:40	12	198	What kind of Visa is required to work in Academia in Japan?	<p>As from title
2	5	246	2012-02-14 20:26:23	7	386	As a computational chemist, which online resources are av...	<p>Which online
3	5	6	2012-02-14 20:27:42	31	1659	Where can I find the Impact Factor for a given journal?	<p>As from title
4	18	145	2012-02-14 20:29:05	9	200	In U.S., why do many engineering departments care about...	<p>I have seen
5	5	9	2012-02-14 20:30:27	23	525	What is the h-index exactly and how does it work?	<p>What is the
6	18	NULL	2012-02-14 20:30:30	19	NULL	NULL	<p>If your instit
7	5	20	2012-02-14 20:32:12	61	4299	Does publishing a paper on arXiv prevent me from submitti...	<p>If I publish a
8	12	26	2012-02-14 20:34:32	17	444	What journals do not allow open access to published materi...	<p>An increasin
9	12	NULL	2012-02-14 20:39:18	23	NULL	NULL	<p>The h-index
10	26	NULL	2012-02-14 20:39:47	28	NULL	NULL	<p>Not necessa
11	12	NULL	2012-02-14 20:40:32	7	NULL	NULL	<p>There is a lis
12	28	NULL	2012-02-14 20:41:50	15	NULL	NULL	<p><a href="ht

	Message	Time	Action
1	3418 row(s) returned	10:36:07	SELECT * FROM academia.posts LIMIT 0, 10000

Fig. 6.2 Database table Posts

Owner of the post is the primary key in the Users database table.

Database table (Comments)



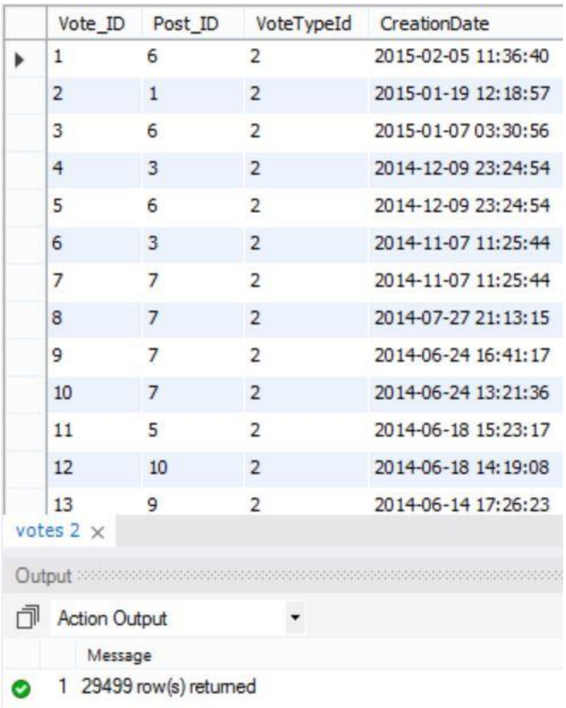
Comment_ID	Post_ID	Score	Text	CreationDate	User_ID
2	2	0	Do you mean in private industry or in an instructional capacity? I suspect that th...	2012-02-14 21:06:15	30
3	23	0	Also, this comment probably also applies to DVM/PhD programs (still NIH funded, ...	2012-02-14 21:08:34	30
4	24	0	I want to get the PhD, but the MSc is just the first step into the PhD.	2012-02-14 21:09:15	15
5	25	1	To the best of my knowledge, this is also the case in France.	2012-02-14 21:23:32	43
6	23	0	victor hello. I didn't think of them, so thanks for catching that.	2012-02-14 21:23:33	28
8	28	2	We could start writing down our salaries here... that would give some interesting...	2012-02-14 21:25:36	5
9	28	0	Maybe you should reformulate the question. As it is now, this will lead to big lists,...	2012-02-14 21:25:43	43
10	24	0	@Matias: My point was that I don't believe that a Master's must necessarily prec...	2012-02-14 21:28:19	28
11	28	0	@Sylvain Peyronnet: Is the reformulated question better? You should edit it.	2012-02-14 21:30:06	31
12	28	0	Yep, I like it now ;) I am not in favour of editing other people questions at the be...	2012-02-14 21:31:40	43
14	13	1	Could you clarify your question? Are you asking for a listing and definition of the ...	2012-02-14 20:46:37	28
15	24	3	@mac389 If you are asking clarifying questions, please post those as comments....	2012-02-14 21:43:42	23
17	40	4	"Does anyone have any thoughts or comments about this?" is awfully open-ende...	2012-02-14 22:29:15	8

	Message	Time	Action
1	5314 row(s) returned	10:37:33	SELECT * FROM academia.comments

Fig. 6.3 Database table Comments

The Comments table includes information about user’s comments in the published posts. Post ID is the primary key in the Posts database table.

Database table (Votes)



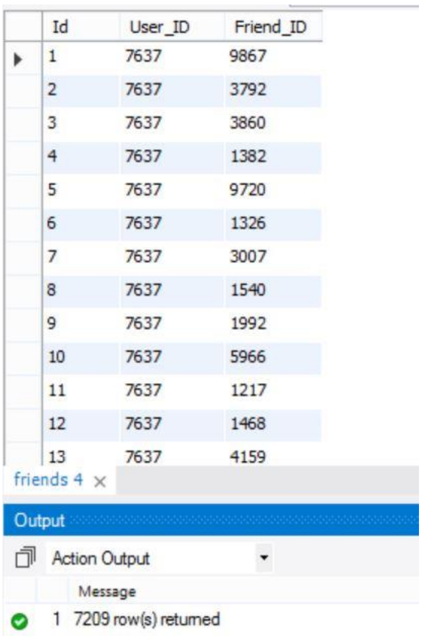
Vote_ID	Post_ID	VoteTypeId	CreationDate
1	6	2	2015-02-05 11:36:40
2	1	2	2015-01-19 12:18:57
3	6	2	2015-01-07 03:30:56
4	3	2	2014-12-09 23:24:54
5	6	2	2014-12-09 23:24:54
6	3	2	2014-11-07 11:25:44
7	7	2	2014-11-07 11:25:44
8	7	2	2014-07-27 21:13:15
9	7	2	2014-06-24 16:41:17
10	7	2	2014-06-24 13:21:36
11	5	2	2014-06-18 15:23:17
12	10	2	2014-06-18 14:19:08
13	9	2	2014-06-14 17:26:23

1 29499 row(s) returned

Fig. 6.4 Database table Votes

Votes table includes information about votes in different posts.

Database table (Friends)



Id	User_ID	Friend_ID
1	7637	9867
2	7637	3792
3	7637	3860
4	7637	1382
5	7637	9720
6	7637	1326
7	7637	3007
8	7637	1540
9	7637	1992
10	7637	5966
11	7637	1217
12	7637	1468
13	7637	4159

1 7209 row(s) returned

Fig. 6.5 Database table Friends

Friends table includes information about users’ friend connections in the social network.

Load data

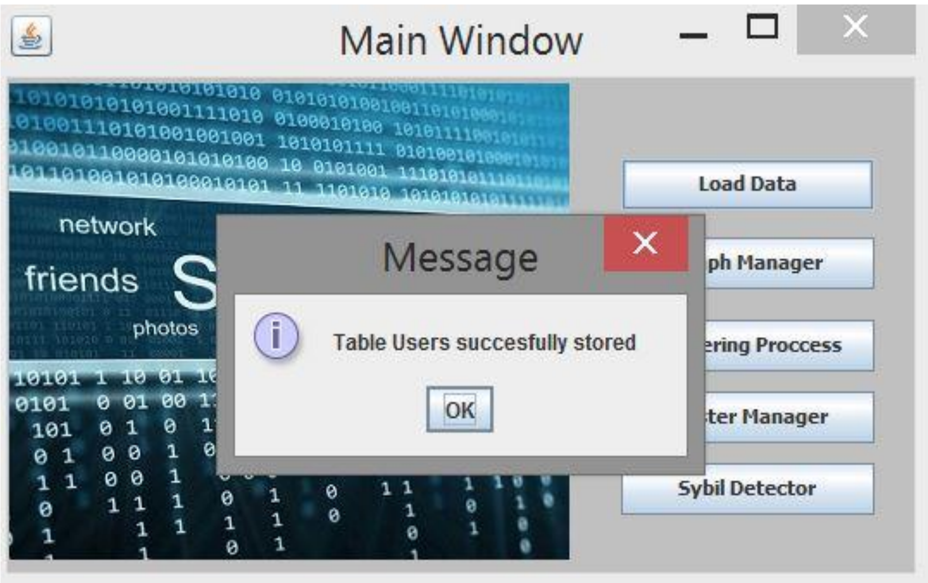
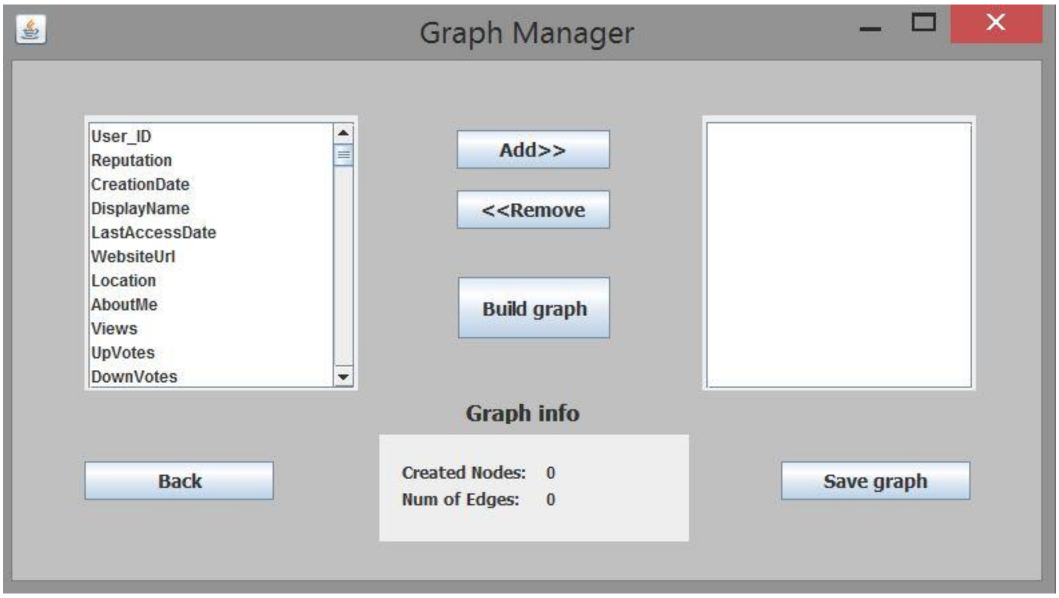


Table Users succesfully stored

Fig. 7 Load data window

First the user must load the existent tables into database. We work with MYSQL server in order to manage database tables. A message about successful storage will be showed.

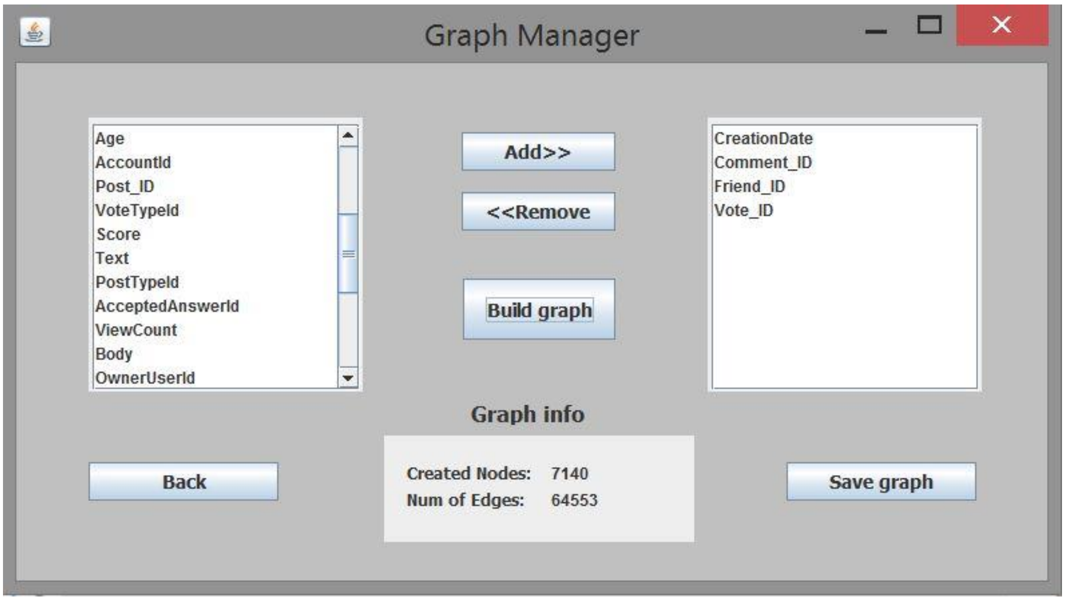
5.4. Graph manager



Created Nodes: 0
Num of Edges: 0

Fig. 8 Graph manager window

Graph manager window allows building network graph. The user chooses available parameters from the list. The list includes all available fields and features from the database tables. After "Build graph" button have been pressed, all selected features will be used in order to build network graph.

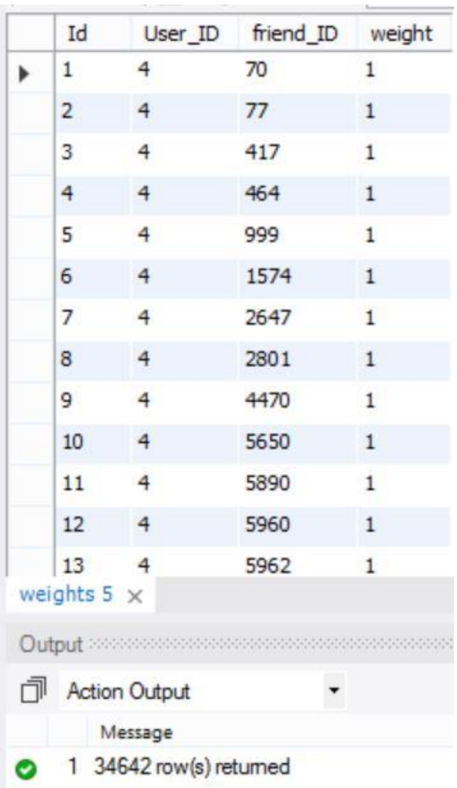


Created Nodes: 7140
Num of Edges: 64553

Fig. 9 Graph manager window

Build graph algorithm creates the weights table in the database. After the end of the algorithm, the user can see the information about created network graph. An additional option is to save network graph into external file by pressing “Save graph” button.

Database table (weights)



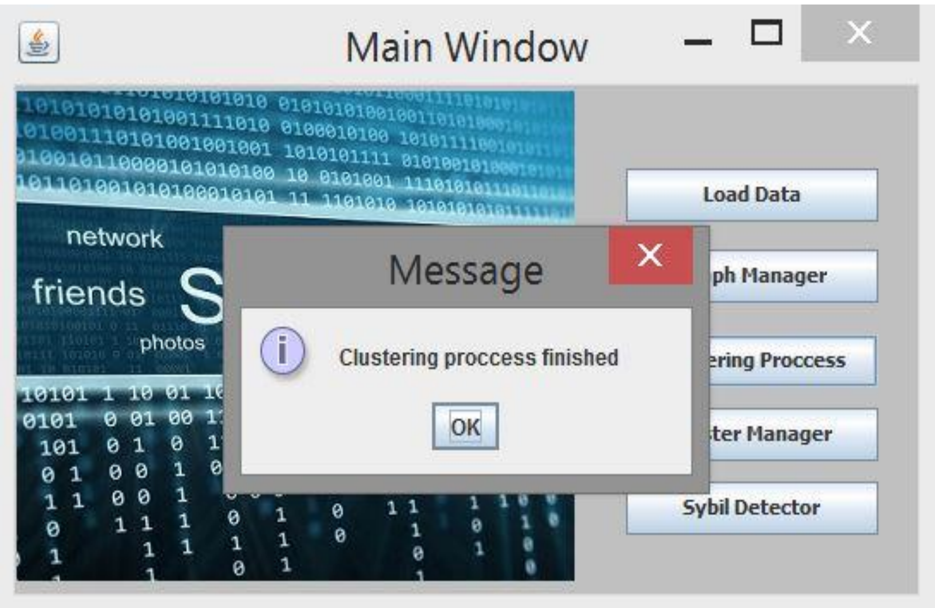
Id	User_ID	friend_ID	weight
1	4	70	1
2	4	77	1
3	4	417	1
4	4	464	1
5	4	999	1
6	4	1574	1
7	4	2647	1
8	4	2801	1
9	4	4470	1
10	4	5650	1
11	4	5890	1
12	4	5960	1
13	4	5962	1

1 34642 row(s) returned

Fig.9.1 Graph manager window (weights)

Weight parameter represents the values of the graph edges.

5.5. Clustering process



Clustering proccess finished

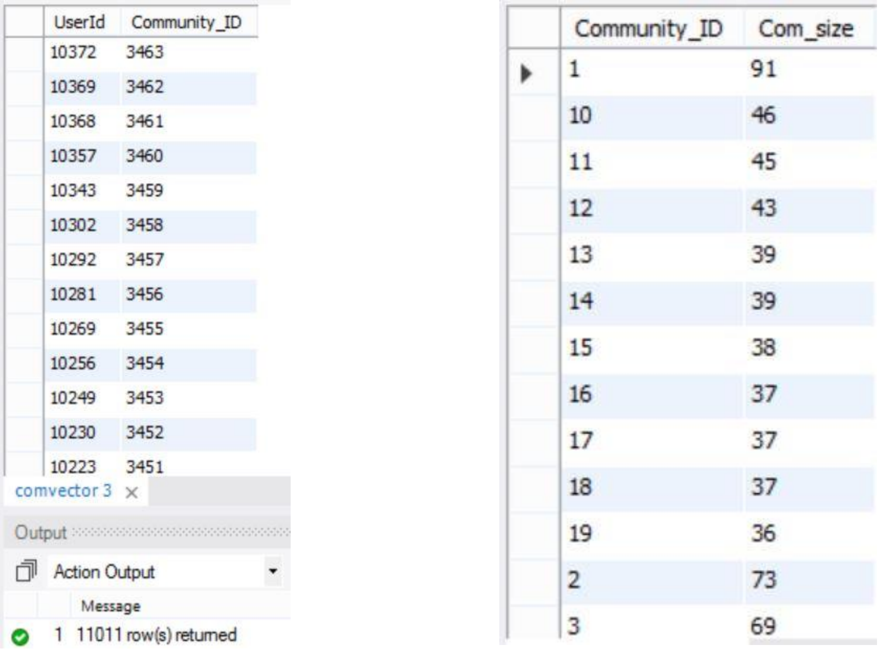
OK

Fig. 10 Clustering process algorithm

The Clustering Algorithm described in the 3.2, is applied by pressing “Clustering process” button.

After completing the process, additional information about user’s community index and community size is available in the database tables.

Database table (Community tables)



UserId	Community_ID
10372	3463
10369	3462
10368	3461
10357	3460
10343	3459
10302	3458
10292	3457
10281	3456
10269	3455
10256	3454
10249	3453
10230	3452
10223	3451

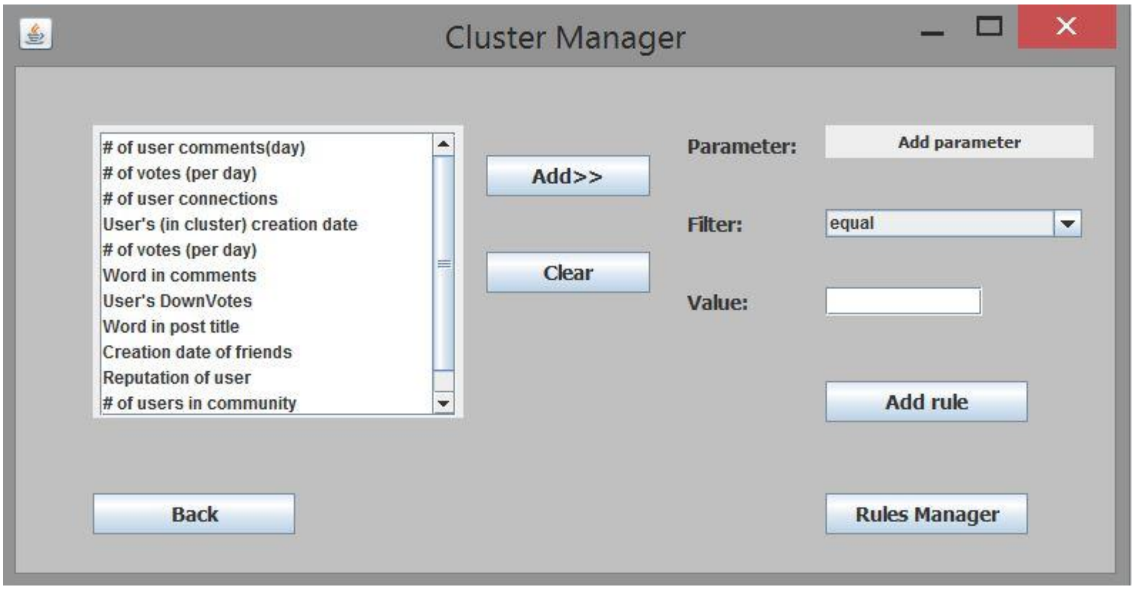
```
Message
1  11011 row(s) returned
```

Community_ID	Com_size
1	91
10	46
11	45
12	43
13	39
14	39
15	38
16	37
17	37
18	37
19	36
2	73
3	69

Fig. 10.1 Clustering process algorithm (Community tables)

Each user belongs to the specific community. Size of each community is also known.

5.6. Clusters manager



# of user comments(day)
# of votes (per day)
# of user connections
User's (in cluster) creation date
# of votes (per day)
Word in comments
User's DownVotes
Word in post title
Creation date of friends
Reputation of user
# of users in community

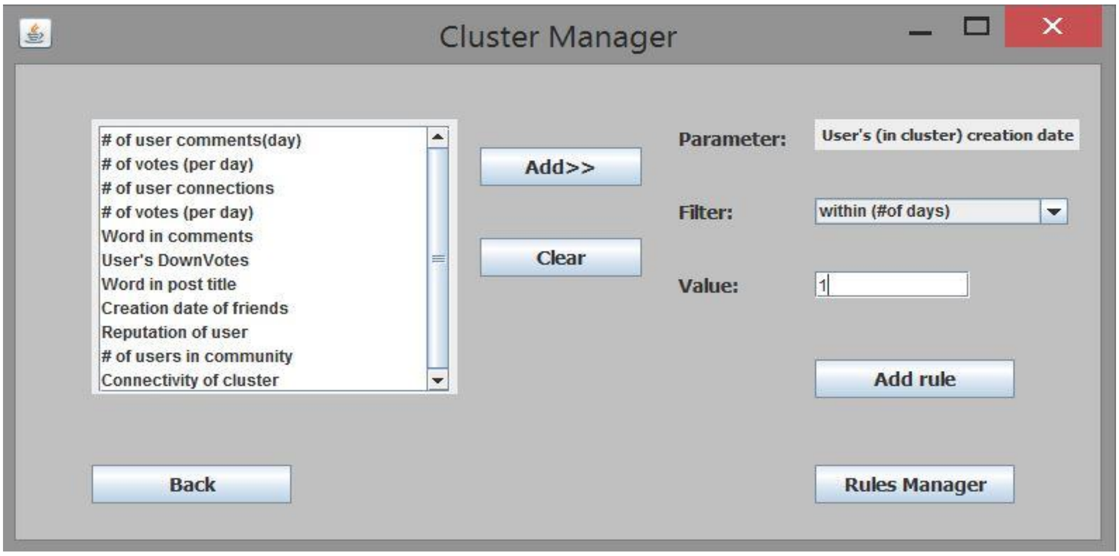
Parameter: Add parameter

Filter: equal

Value:

Fig. 11 Cluster manager window

Cluster Manager Window allows creating rules. The user can filter specific cluster or community data in order to detect Sybil behavior. (For example, to find all users that created more than 100 comments a day).



# of user comments(day)
# of votes (per day)
# of user connections
# of votes (per day)
Word in comments
User's DownVotes
Word in post title
Creation date of friends
Reputation of user
# of users in community
Connectivity of cluster

Parameter: User's (in cluster) creation date

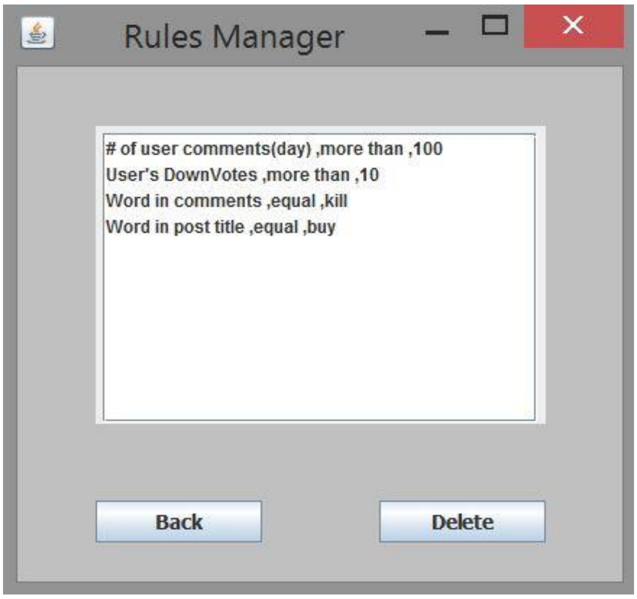
Filter: within (#of days)

Value: 1

Fig. 12 Cluster manager window

Choosing the appropriate filters from the list, the user creates rules. The number of rules may be one or more. The user can manage filters in the Rules Manager window.

5.7. Rules manager

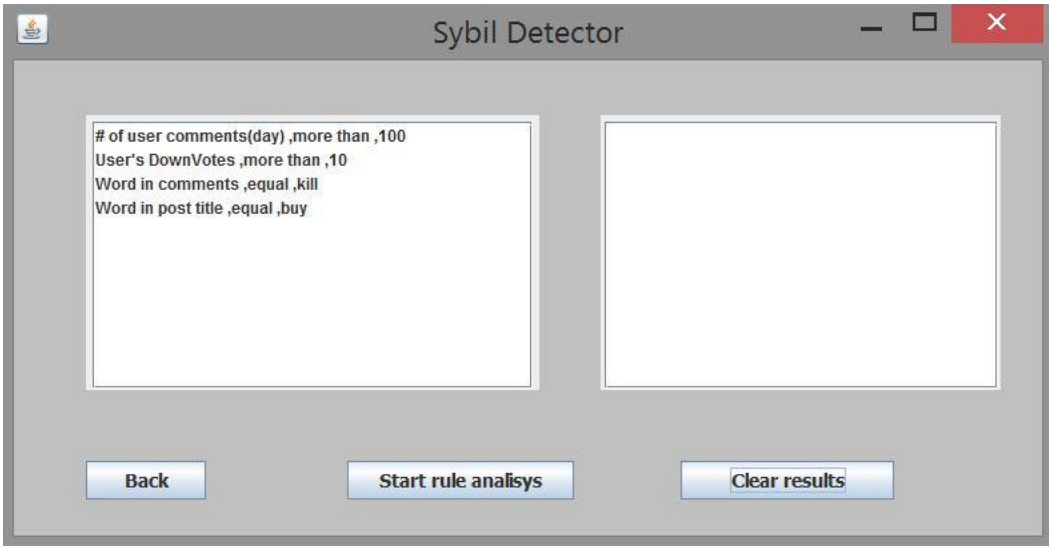


# of user comments(day) ,more than ,100
User's DownVotes ,more than ,10
Word in comments ,equal ,kill
Word in post title ,equal ,buy

Fig. 13 Rules manager window

The user can view or delete some rule from Rules Manager window. In order to delete rule the user must select this rule from the list and press “Delete” button.

5.8. Sybil Detector

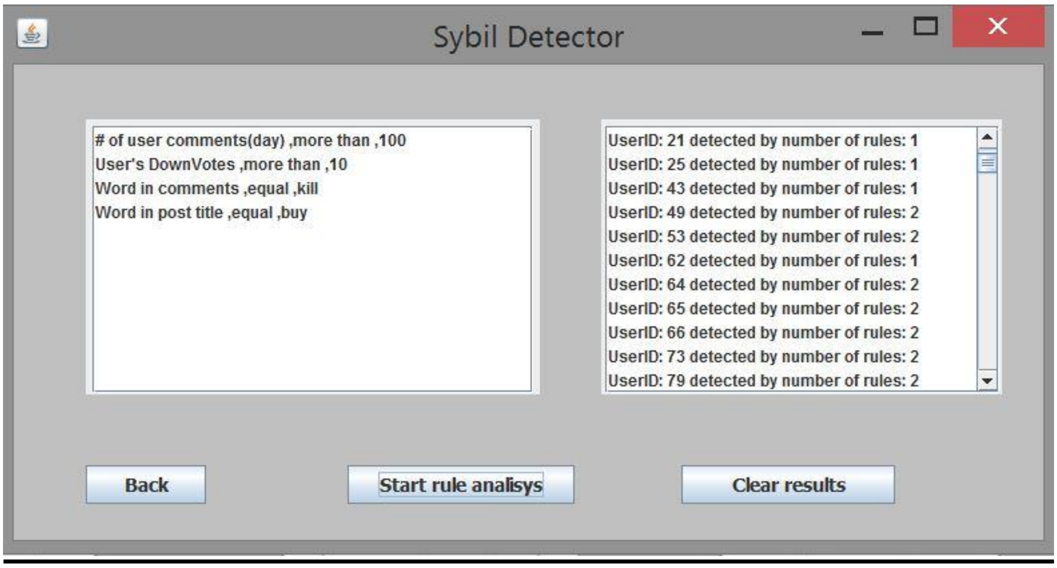


# of user comments(day) ,more than ,100
User's DownVotes ,more than ,10
Word in comments ,equal ,kill
Word in post title ,equal ,buy

Fig. 14 Sybil detector window

All created rules will be displayed here. In order to begin analysis "Start rule analysis" button should be pressed.

Sybil Detector



```
# of user comments(day) ,more than ,100
User's DownVotes ,more than ,10
Word in comments ,equal ,kill
Word in post title ,equal ,buy
```

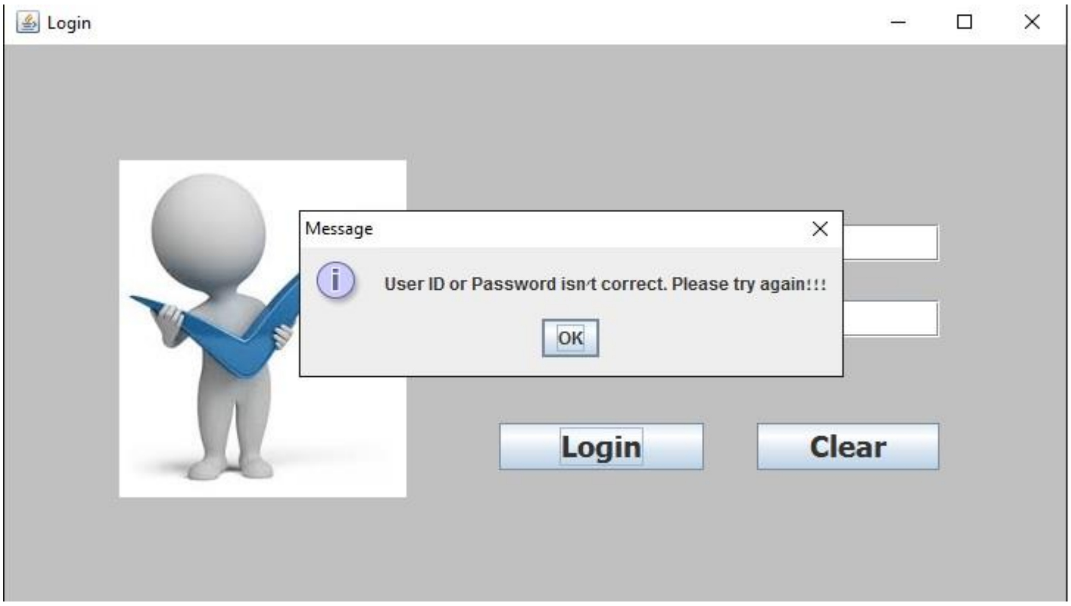
```
UserID: 21 detected by number of rules: 1
UserID: 25 detected by number of rules: 1
UserID: 43 detected by number of rules: 1
UserID: 49 detected by number of rules: 2
UserID: 53 detected by number of rules: 2
UserID: 62 detected by number of rules: 1
UserID: 64 detected by number of rules: 2
UserID: 65 detected by number of rules: 2
UserID: 66 detected by number of rules: 2
UserID: 73 detected by number of rules: 2
UserID: 79 detected by number of rules: 2
```

Fig. 15 Sybil detector window

Algorithm analyzes each rule and shows the results in the right window. Algorithm creates the results file for each rule. Output file is named by rule details. (For example output file of the rule “Word in post title, equal, bad word” will be “Rule Word in post title_equal_bad.txt”). Each output file will include the users’ details.

6. TESTING

6.1. Login test

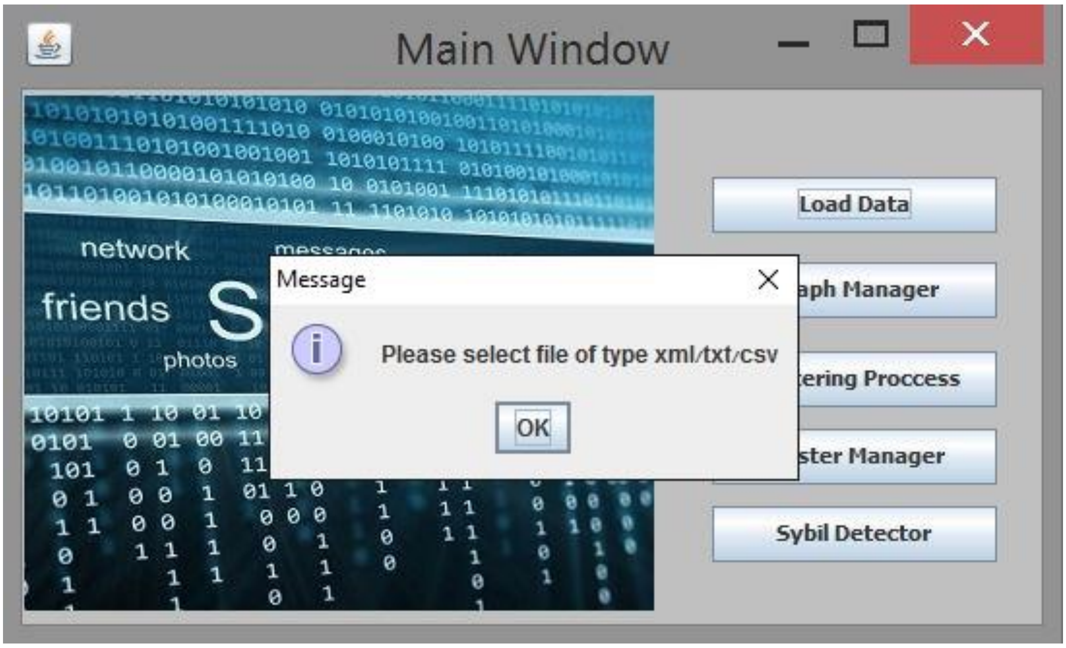


User ID or Password isn't correct. Please try again!!!

Fig. 16 Login test

In order to enter the program interface, the user must input correct an User ID and a Password

6.2. Load data test



Please select file of type xml/txt/csv

Fig. 17 Loading data test

Only **xml/txt/csv** types of file can be imported into database.

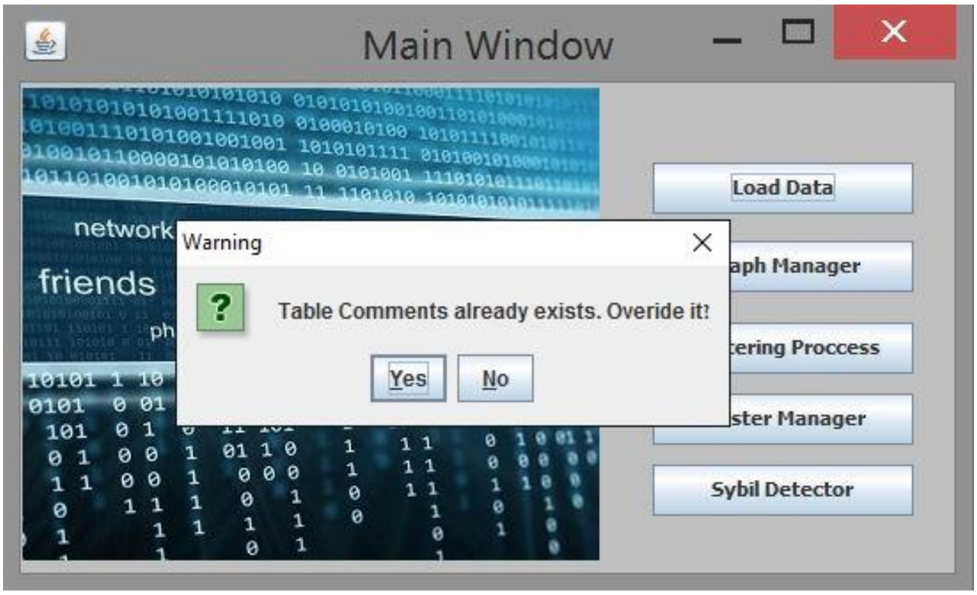


Table Comments already exists. Overide it!

Fig. 18 Load data test

When the user loads data file into database and table already exists, it can be overridden.

6.3. Graph manager test

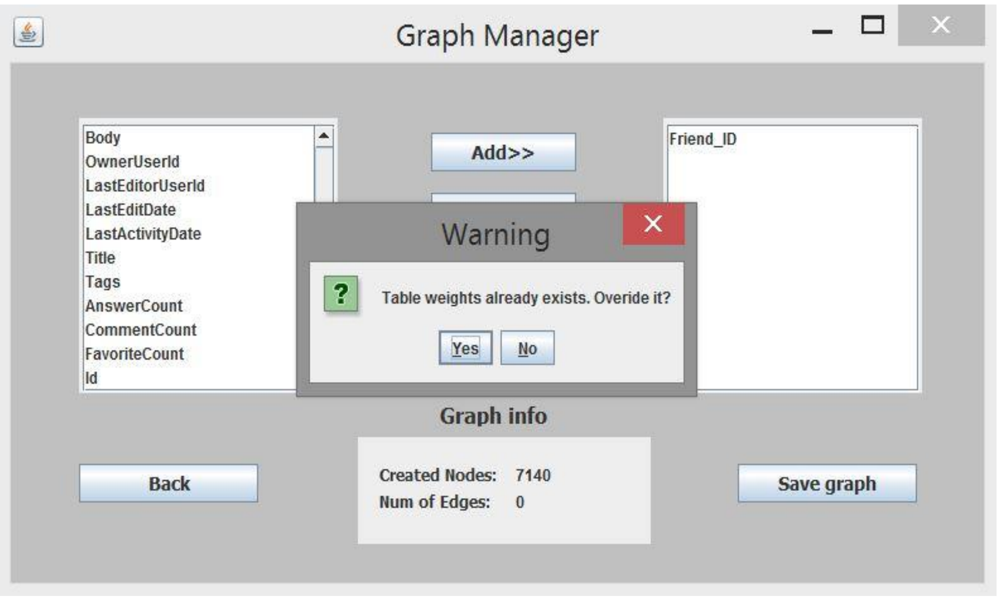


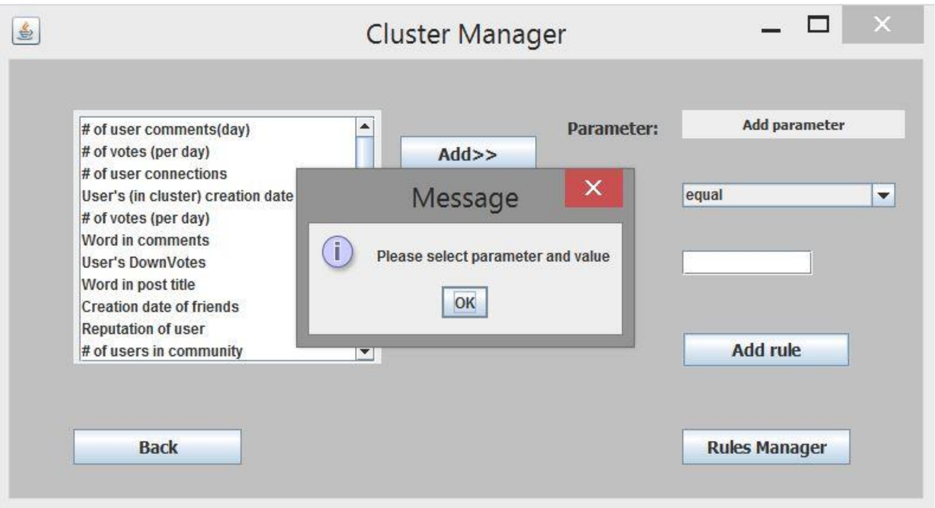
Table weights already exists. Overide it?

Created Nodes: 7140
Num of Edges: 0

Fig. 19 Graph manager test

Any existent table in the database can be overridden.

6.4. Cluster manager test



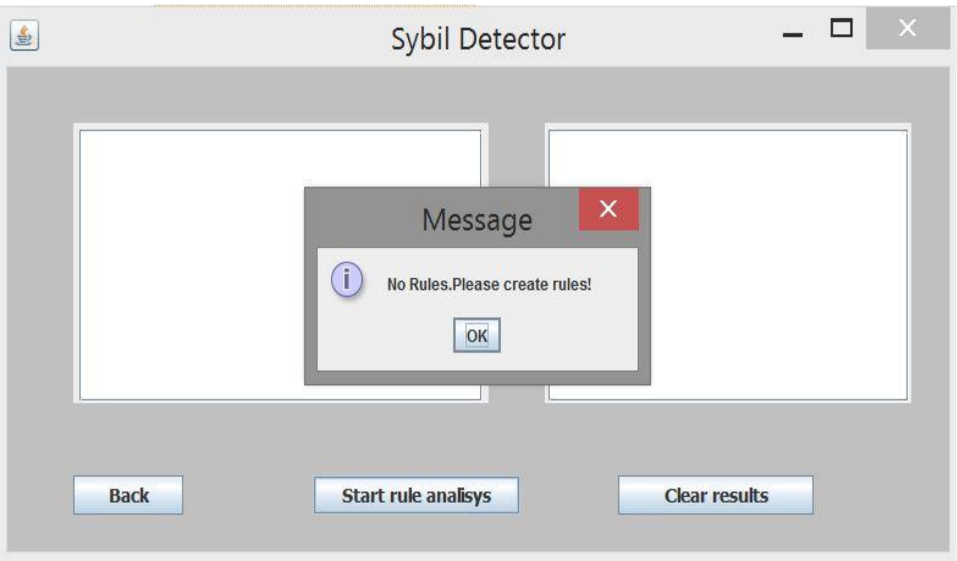
Please select parameter and value

equal

Fig. 20 Cluster manager test

In order to create a rule, the user must select parameter, filter and enter value. Otherwise, an error message will be displayed.

6.5. Sybil Detector test



No Rules.Please create rules!

Fig. 21 Sybil detector test

In order to start rule analysis, the user must create at least one rule.

6.6 Tests table

	Test subject	Expected result
1.	Login	Can login to the application only if user inserts correct User ID and Password, otherwise error message expected.
2.	Load data	User must select files with the extension xml/txt/csv, in order to import data to DB, otherwise error message expected.
3.	Overriding data	User can load data files to DB, but if they already exist user can override them and error message will be showed.
4.	Graph builder	Allows build graph only in case database exists, otherwise error message expected.
5.	Clustering process	Enable to start clustering process only after valid graph creation and features selection.
6.	Cluster manager	User must choose a parameter, filter and value, otherwise error message expected.
7.	Back button	Return back to previous window after press on the button.
8.	Apply Sybil detector without rule/filter selection	Not possible. Error message expected.
9.	Sybil detector termination	Show results on the screen.

7. RESULTS

We examined and tried to understand and evaluate the best features we need to detect the potentially Sybil users. We have to decide what definition of user's unexplained behavior in the social network is. Then we try to create rules that will help us to distinguish the normal users from the Sybil ones. We expect to verify our model by generating a lot of rules and analyze the obtained results.

8. CONCLUSIONS

We have experimented and created a sufficient number of filters and rules in order to detect abnormal behavior in the social network. We have analyzed a lot of results in order to verify our Sybil detecting model. Our algorithm detected all nonexistent users. All expected results were correct therefore our method is reliable.

We came across lots of algorithms that are used to clustering data .We chose one of them the "Fast unfolding of communities in large networks" to be useful and implemented in our project and some are not. We encountered some problems and even managed to fix some of them. For example one of the problems that we encountered with was the input matrix size for the fast unfolding clustering algorithm. Algorithm exceeds the RAM memory size with too large matrix of users' connections as an input. Therefore we reduced the number of rows in the database tables.

REFERENCES

[1] Leskovec, J. Kleinberg, and C. Faloutsos. *Graphs over time: densification laws, shrinking diameters and possible explanations*. In KDDWS, 2005.

[2] Subramani, K. *Density-based community detection in social networks*.

[3] B. Viswanath, A. Post, K. Gummadi, and A. Mislove, "*An analysis of social network-based Sybil defenses,*" in Proc. ACM SIGCOMM Conf., 2010, pp. 363–374.

[4] Z. Cai and C. Jermaine, "*The latent community model for detecting Sybils in social networks*," in Proc. 19th Annu. Netw. Distrib. Syst. Security Symp. (NDSS), 2012, pp. 1–13.

[5] L. Shi, S. Yu, W. Lou, and Y. T. Hou, "*SybilShield: An agentaided social network-based Sybil defense among multiple communities,*" in Proc. IEEE Conf. Comput. Commun. (INFOCOM), 2013, pp. 1034–1042.

[6] Reversible Markov *Chains and Random Walks on Graphs* - David Aldous and James Allen Fill.

[7] J Z. Yang et al., "*Uncovering social network Sybils in the wild," in Proc. Int. Micro-electron. Conf. (IMC),* 2011, pp. 259–268.

[8] G. Wang et al., "*You are how you click: Clickstream analysis for Sybil detection*," in Proc. 22nd USENIX Security Symp., 2013, pp. 241–255.

[9] Girvan M and Newman M E J, 2002 Proc. Natl. Acad. Sci. USA 99 7821.

[10] Newman M E J, 2006 Proc. Natl. Acad. Sci. USA 103 8577.

[11] Fortunato S and Barth´elemy M, 2007 Proc. Natl. Acad. Sci. USA 104 36.

[12] Arenas A, Duch J, Fern´andez A and G´omez S, 2007 N. J. of Phys. 9 176.