LINKCUBE:
A TOOL FOR ANOMALY DETECTION IN SOCIAL NETWORK USING GBAD

by

RAMESH PAUDEL

A PROJECT REPORT

submitted in partial fulfillment of the requirements for the degree

MASTERS OF SCIENCE

Department of Computer Science
College of Engineering

TENNESSEE TECHNOLOGICAL UNIVERSITY
Cookeville, Tennessee

2014

Approved by:

Dr. William (Bill) Eberle
Dr. Doug Talbert
Dr. Ambareen Siraj

Table of Contents

List of Figures

Acknowledgements

I am pleased to acknowledge Dr. Bill Eberle for his invaluable guidance during the course of this project work. Without his continuous help and guidance, this project would have been an uphill task. Also I am grateful to Dr. Doug Talbert and Dr. Ambareen Siraj, members of graduate advisory committee for their continuous support and guidance.

I extend my sincere thanks to all the member of Knowledge Discovery Lab, TTU for their inspiring weekly meeting and idea they have shared which were valuable during the course of this project work.

I am also grateful to Department of Computer Science, Tennessee Technological University for providing all the resource required for the project.

I would like to thank, Mr. Luke Jackson who helped a lot during the initial stage of the project for setting up necessary hardware.

March 2014 Ramesh Paudel

Chapter 1 - Introduction

Anomaly detection is one of the major areas of computer security [6]. Research into anomaly detection has led to the successful implementation of approaches in a wide-range of areas: Network intrusion detection, credit card or cell phone fraud detection, healthcare diagnostics, industrial damage detection, image processing etc. Many algorithms are used for anomaly detection in these areas. In particular, graph based approaches are used in domains where the data can be converted into a graph. One particular approach to graph-based anomaly detection is the GBAD approach which scans graphs looking for the substructure which is less likely to occur. This less likely occurring substructure is designated as an anomaly [1].

This master's project presents the implementation of a tool called LinkCube constructed to facilitate anomaly detection in social networking data. The purpose of LinkCube is to provide the user with the ability to take Twitter, Facebook and LinkedIn data feeds, construct a graph, and send the resulting graph to a third-party graph-based anomaly detection tool called GBAD. GBAD allows the user to analyze a graph for potentially anomalous substructures [1].

1.1 Motivation

The widespread use and popularity of social network has tremendously increased the volume of data that can be found on the internet. It is believed that social networks are one of the reasons that 90% of all the data in the world has been generated in the last two years alone [3]. The data generated by social networks is particularly important because it is personal in nature and provides the insight of personal life. The knowledge that can be mined from social networks is of great importance [16]. There might be various anomalous activities going on in the social networks that may or may not be threatening. One potential mechanism for discovering unusual social network activity could be through the use of graph-based anomaly detection tools [1].

Graph Based Anomaly Detection (GBAD) is a tool for finding anomalies in graph based data [1]. One could look at a social network as a large social graph. In order to analyze a graph of social network data using a tool like GBAD we need a general approach for constructing a graph from various social network data. To our knowledge there are no tools that can extract data from a social network, preprocess the data and convert the data to the input format expected by GBAD. While there are various applications available to extract data from social networks, the data is not available in the format needed by GBAD, nor is generalizable to other graph-based knowledge discovery tools. The main motivation for this project is to assist researchers in applying graph-based anomaly detection to social networks by providing a simple means of data extraction and preprocessing.

1.2 Project Goal

The project presented in this report was coordinated at the Department of Computer Science, Tennessee Tech University. The goal of the project is defined as follows:

Project Goal*: *Design a tool for anomaly detection in a social network using GBAD that will provide the end-user with the ability to extract data from the social network and convert into the desired format for GBAD and other visualization tools (XDot, Gephi etc).

In order to achieve this goal, the tool we call LinkCube will be designed, implemented and tested with GBAD, allowing us to validate such an approach.

1.3 Structure of the Report

The remainder of the report is divided into five different chapters. Chapter 2 contains a background about the various topics that are related to the project. This section will give the basic idea of the scope of the project. Chapter 3 introduces LinkCube and its architecture. This chapter talks about LinkCube in detail. Chapter 4, 5 and 6 are about the project implementation, project management, and finally some analysis and conclusions.

Chapter 2 - Background

LinkCube provides a tool for anomaly detection on a social network using the GBAD graph-based approach. The following sections provide a brief overview of these topics.

2.1 Social Network

A social network is an interactive structure created among users when they share information, ideas, interests, activities and content amongst themselves. Social networks have existed almost as long as societies themselves have existed. Traditionally, a social network can be thought as workplace, schools, universities, social clubs or organization etc. Today, social networks use websites as an online community to share ideas and contents. These websites also allow a user to create their profile and share it with other people. Once you create a profile you can start networking with people who share common interests and make your own communities.

Traditionally, social networks were localized to a small area or confined to their individual community. But with the emergence of online social networks, people connected to or shared their ideas with another person who shared common interests across a large area irrespective of political, social, economic, or geographic borders. Eventually membership in online social networks increased tremendously from tens and hundreds (in traditional social networks) to thousands and millions even billions (in current online social networks) [22]. The most popular online social networks these days include Facebook [13], LinkedIn [15], Twitter [12], Flickr [17], Google+ [19], Orkut [20] and Cyworld [18].

Twitter, Facebook and LinkedIn data are used as the social network data for this project. Twitter is a micro blogging service that allows people to communicate with a short message (140 characters) [4]. Twitter has a feature to broadcast a short message to the world and conversely a user can discover people online and follow them or their message. In short, one can view these actions as a graph that models the connections between people and their interests (which is reflected by their message). This type of graph has tremendous possibilities in the area of data mining.

Facebook is another social network that provides a great way to meet friends and share thoughts, pictures, videos and articles by creating a profile. Users can post almost anything to their timeline (a snapshot of what is happening in their social circle at any given time) [13] or send an instant message to their friends. The Facebook platform is a social graph representing all online relationships between people, places, things etc. Facebook contains a graph API that provides an automated way for web pages to extract the social connections between Facebook users. A Facebook social graph can have a great importance in data mining.

LinkedIn is a social network for business professionals that provide a way to connect, network and relate to other professional people or groups. As Facebook is primarily for everyone, LinkedIn is for professionals. Like the Facebook and Twitter network, the LinkedIn network could also be useful in data mining.

2.2 GBAD

GBAD is a graph based anomaly detection technique. GBAD finds anomalies in a graph-based data where the anomalous substructure in a graph is part of (or attached to or a missing from) a normative substructure [1]. In other word this approach tries to find a substructure that is similar to the normal substructure but not exactly the same because if a person or entity is trying to commit a fraud, they will do so by imitating the legitimate transaction or at least close to the legitimate transaction. Hence the similar substructures are anomalous substructures.

GBAD assumes there is a graph G with a normative substructure S, where a substructure S' is considered anomalous if the difference d between S and S' satisfies $0<d<=X$, where X is a user defined threshold and d is a measure of the unexpected structural difference between two sub-graphs of a graph [1]. Basically GBAD deals with three different types of anomalies in graphs namely modifications, insertions, and deletions. A modification has unexpected label on a vertex or edge, an insertion has an unexpected vertex or edge, and a deletion is the absence of an edge or a vertex. For a graph based anomaly, the following situation can be considered anomalies in a graph [1]:

1. A vertex exists that is unexpected.
2. An edge exists that is unexpected.
3. The label on vertex is different than was expected.
4. The label on edge is different than was expected
5. An expected vertex is absent.
6. An expected edge between two vertices is absent.

GBAD has been successfully implemented in various domains like insider threat detection; anomaly detection in homeland security cargo screening, cybercrime detection etc. [8, 9, 10, 11]. Anomaly detection in social networks is a relatively new area of research in graph-based anomaly detection.

2.3 Anomaly Detection

Anomaly detection is the process of identifying cases that are unusual within data that is seemingly homogenous. The general steps for anomaly detection is to build a profile of the normal behavior and then use the normal profile to detect the anomalies. An anomaly is a pattern in the data that does not conform to the expected behavior or the observations whose characteristics differ significantly from the normal profile. Anomalies are also called outliers, exceptions, peculiarities, or surprises [7]. An anomaly could signify irregularities, like credit card fraud, calling card fraud, accounting fraud, suspicious cargo shipments, electronic auction fraud, and many others [6].

There are three categories of anomaly detection. Unsupervised anomaly detection helps to identify anomalies in unlabeled test data set [7]. Here the majority of the data instances are normal and anomalous instance are the ones that fit least to the reminder of the data sets. Supervised anomaly detection helps to identify anomalous instances in the data set where data instances are labeled as normal and anomalous and also have a training set [7]. Semi-supervised anomalous detection constructs a model representing normal behavior from the given normal training data set, and then tests the likelihood of a test instance to be generated by the learnt model [7].

Successful anomaly detection has been demonstrated in the following domains [7]:-

1. Network intrusion detection
2. Insurance/credit card/cell phone fraud detection
3. Healthcare informatics/medical diagnostic
4. Industrial damage detection
5. Image processing/video surveillance
6. Novel topic detection in text mining etc.

The Graph Based Anomaly Detection (GBAD) approach used in the project is an unsupervised anomaly detection technique.

2.4 Anomaly in a Social Network

An anomaly in a social network will vary based on the data set and the kind of problem we are trying to solve. Anomaly detection on graphs of social or communication networks has important security applications. There are various approaches already implemented for detecting anomalies in a social network. Some use graph based approaches, while others use statistical modelling for identifying anomalies. This project works with a graph based approach. The following are some examples of anomalies in a social network [5]:

- *Missing connected sub graph*

  If an edge is the only link to the sub graph, losing an edge due to some reason will result in missing the portion of a graph. This type of anomaly is called missing connected sub graph [5]. In the following example, shown in Figure 2.1, an edge 'retweet' that goes from a node 'text' to an another node 'text' is missing as a result the graph is partitioned into two sub graphs.

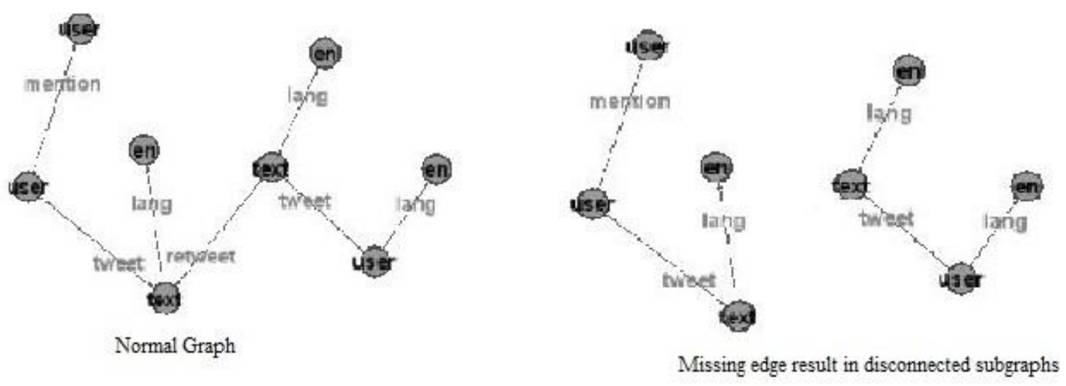


Fig 2.1: Disconnected sub graph anomaly

- *Missing vertex or edge*

  Sometime an expected vertex or edge will be missing from the graph [5]. Figure 2.2 shows an example of a missing expected edge from the Twitter graph.

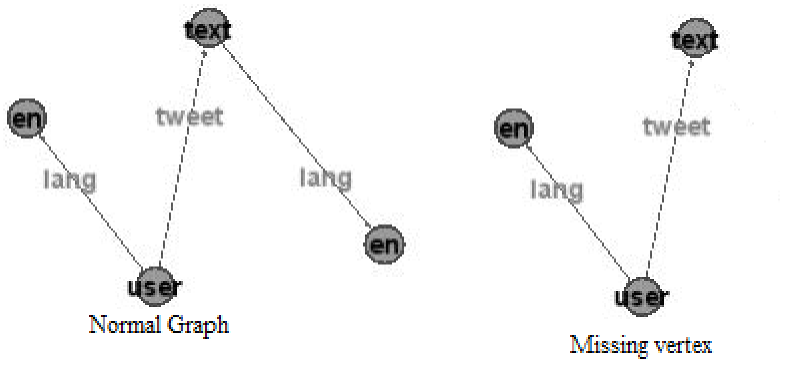


Fig 2.2: Missing edge anomaly

- *Connectivity change*

  Connectivity change refers to anomalies where scalar graph properties such as the total numbers of vertices and edges do not change but for some vertices, the edges these vertices are adjacent to change [5]. In the following example, shown in Figure 2.3, the label of the vertex change but the total number of vertices and edges remains the same.

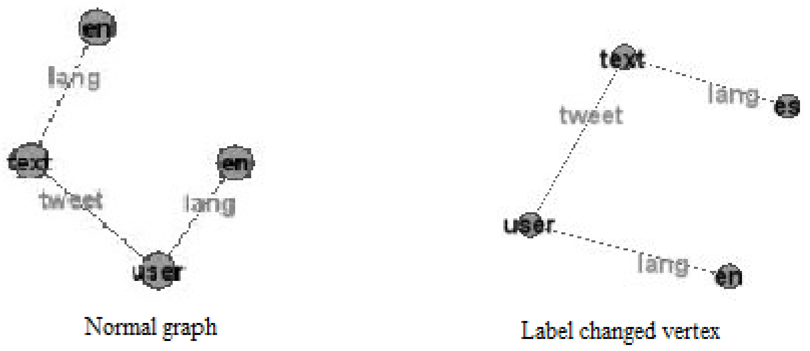


Fig 2.3: Connectivity change anomaly (Label of the vertex change)

Chapter 3 - Project Overview

LinkCube is command-line based tool written in Python that runs on a Linux operating system. LinkCube processes the data between a social network that resides on the internet and GBAD. LinkCube is composed of two layers: the first layer is the data extraction layer, and the second layer is the data processing layer. The data extraction layer has three modules that invoke data requests to each of the three social networks (Twitter, Facebook and LinkedIn). Each module receives data in the JSON format and forwards it to their respective parser in the data processing layer. The data processing layer gathers the data, processes the data, and converts the data into a graph. The data processing layer has an individual parser for each of the modules in the data extraction layer. The output from the data processing layer is in the form of a graph. This output graph can then be run in GBAD to discover an anomaly.

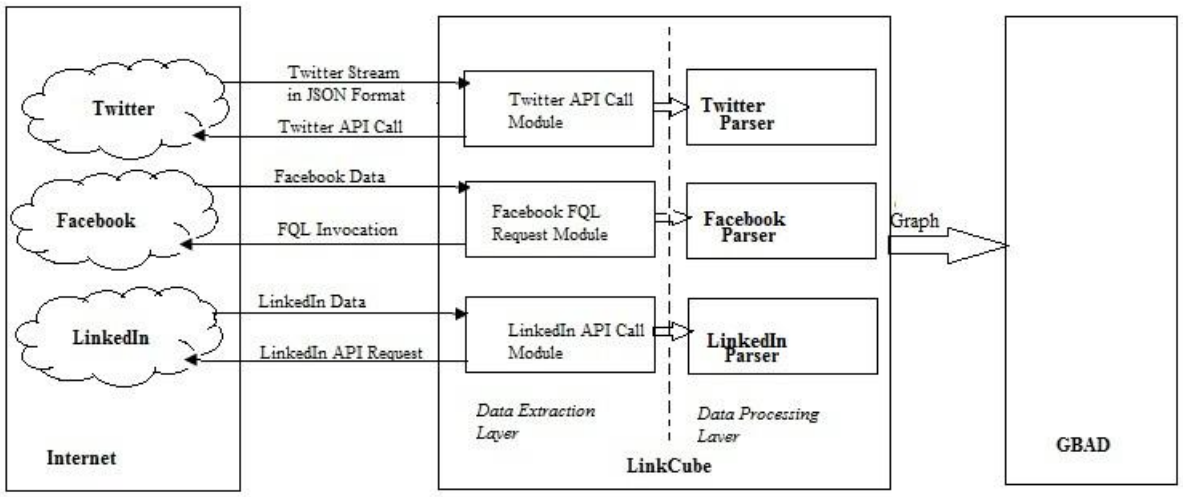


Fig 3.1 System Architecture of LinkCube

3.1 Data Extraction Layer

The data extraction layer connects to social networks through various modules. The main task of the data extraction layer is to send data requests to the social networks and save the response for further processing by the data processing layer. Since we are using Twitter, Facebook and LinkedIn for this project, there is an individual module for each of these social networks. A request to each of the social networks varies according to their API methods and the security they impose on the data.

3.1.1 Twitter API Call Module

Twitter is an information network and communication mechanism that produces millions of tweets every day. The Twitter platform offers access to its data via Twitter APIs [12]. Twitter has different types of APIs, such as Search, REST and Streaming [12]. The Twitter API Call Module in the data extraction layer of LinkCube is built upon the Twitter streaming API. Use of the streaming API allows us to find a set of tweets with specific keywords. The advantage of using the streaming API is it allows us to query the search at a high speed and get trending topic's tweet [12]. The Twitter API is HTTP based and we send a GET request to Twitter to retrieve the data for the feed.

The Twitter module uses a consumer key, consumer secrete key, access token and access token secret key to authenticate the user. In addition to the above authentication, we have to provide Twitter with a username and password. The username and password can be provided from the command line during runtime. The command to run this module is:

```
python twitter.py -u <username> -p <password> -t <tweet_keyword>
```

The third option is a keyword. The tweets are extracted based on the keyword used in the tweet message. The result returned by Twitter is a JSON data file. The pseudo code given below explains the general flow of the program:

```
callListener()
{
    if (dataFound)
    {
        loadStream;
        return stream;
    }
    else if(timeout)
    {
        return timeoutMessage;}
    else if (error)
    { return error; }
}
```

```
main()
{
    define  CONSUMER_KEY, CONSUMER_SECRET,
            ACCESS_TOKEN, ACCESS_TOKEN_SECRET;
    loginInfo = getUserLoginInformation();
    key = getKeyword();

    //login_info has twitter account username and password
    if (authenticateUser(loginInfo))        {
        api = createApi();
        if( authenticateApi(api))   //using consumer key and access token
        {
            createStreamListener;
            stream = callListener();
            if (stream)
            {
                filterStream(stream,key);
                writeStreamToFile;
            }
            else
            {
                print error;
            }
        }
    }
    else
        end;
}
```

3.1.2 Facebook FQL Request Module

The Facebook FQL Request Module uses a Facebook Graph API to get access to a Facebook social graph. Essentially there are three methods of using the Graph API: requesting data, posting data, and deleting data [13]. The Facebook platform offers access to its data via the requesting data API. The main objective of the Facebook FQL Request Module is to retrieve data from Facebook using their Graph API. Some data in Facebook is public so it can be requested without authentication while most data is closed and requires an authentication. This module can access authenticated data from the user network if they provide an access key of their Facebook profile. Beyond the personal network it only accesses the data that is publicly available and needs no authentication. The Facebook Graph API has a tool called FQL Query which provides a privilege to query data from Facebook tables. We send the FQL query as a GET request to extract the data from Facebook. The command to run this module is:

python facebook.py -a <ACCESS_TOKEN> -q <QUERY>

The first option is an access token for the Graph API. An access token is required to authenticate the user's Facebook profile. Each user must provide their Facebook access token to use this module. The second option is an FQL query where the user can enter a query based on the data they want back from the Facebook. All of the data received from Facebook is encoded as JSON. The pseudo code given below explains the general flow of the program:

```
main()
{
    accessToken = getAccessToken();
    query = getFql();
    define baseUrl;
    url = createUrl (baseUrl,query,accessToken);
    if(sendRequest(url))
    {
        createFile();
        writeDateToFile();
    }
    else
    {
        end;
    }
}
```

3.1.3 LinkedIn API Call Module

Like Facebook and Twitter, LinkedIn also provide an API to access its data. The LinkedIn API divides information into resources for people, companies, groups, and jobs [15]. The LinkedIn module uses a consumer key, consumer secrete key, user token and user secret key to authenticate the user. The command to run this module is:

python linkedin.py -f <fields like "a,b,c">

We can provide the fields we need from the profile of a user's connection through the command line options. All of the data received from LinkedIn is also encoded as JSON. The pseudo code given below explains the general flow of the program:

```
main()
{
        define  CONSUMER_KEY, CONSUMER_SECRET,
                USER_TOKEN, USER_SECRET
        define baseUrl;
        define oauth2;

        fields = getField();
        url = createUrl (baseUrl,fields);
        consumer = oauth(CONSUMER_KEY, CONSUMER_SECRET);
        token = oauth(USER_TOKEN, USER_SECRET);
        client = oauth(consumer, token);

        if(sendRequest(client,url))
        {
                createFile();
                writeDateToFile();
        }
        else
        {
                end;
        }
}
```

3.2 Data Processing Layer

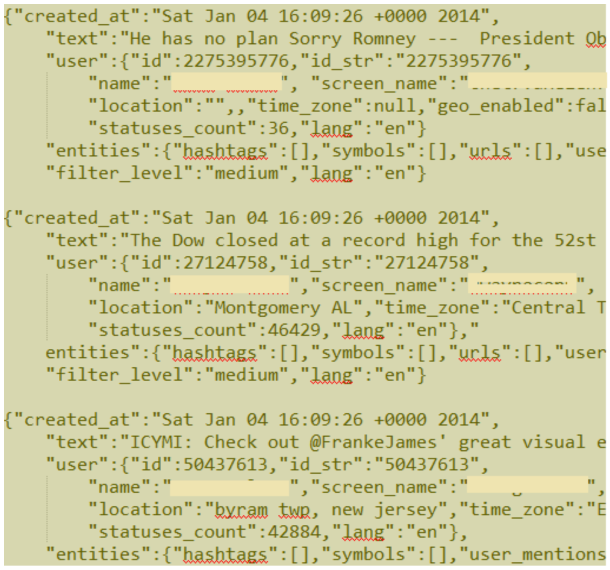
The data processing layer takes the data from data extraction layer and converts it into a graph. This layer has an individual parser for each of the three modules in the data extraction layer. The data generated by these modules are in a different structure. The structure of the graph that needs to be generated also may differ significantly from one social network to another so an individual parser is built for each module.

3.2.1 Twitter Parser

The Twitter Parser takes the JSON file generated by the Twitter API module and converts it into the graph. The command to run this module is:

python twparser.py -s <SOURCE FILE> -g <GRAPH FILE>

The first option is the name of the JSON source file and the second option is the name of graph file we are trying to generate. An example portion of a JSON file and a graph file are shown in the Fig 3.2.

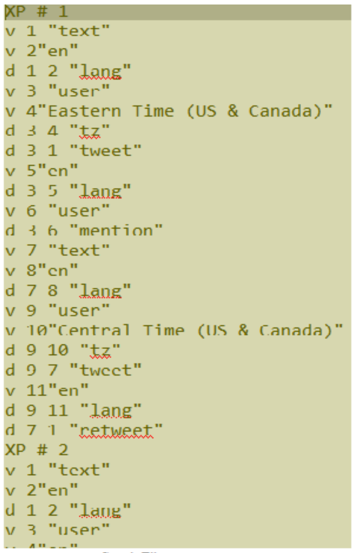


```
{"created_at":"Sat Jan 04 16:09:26 +0000 2014",
    "text":"He has no plan Sorry Romney ---  President Ob
    "user":{"id":2275395776,"id_str":"2275395776",
        "name":"[redacted]", "screen_name":"[redacted]
        "location":"",,"time_zone":null,"geo_enabled":fal
        "statuses_count":36,"lang":"en"}
    "entities":{"hashtags":[],"symbols":[],"urls":[],"use
    "filter_level":"medium","lang":"en"}

{"created_at":"Sat Jan 04 16:09:26 +0000 2014",
    "text":"The Dow closed at a record high for the 52st
    "user":{"id":27124758,"id_str":"27124758",
        "name":"[redacted]","screen_name":"[redacted]",
        "location":"Montgomery AL","time_zone":"Central T
        "statuses_count":46429,"lang":"en"},"
    entities":{"hashtags":[],"symbols":[],"urls":[],"user
    "filter_level":"medium","lang":"en"}

{"created_at":"Sat Jan 04 16:09:26 +0000 2014",
    "text":"ICYMI: Check out @FrankeJames' great visual e
    "user":{"id":50437613,"id_str":"50437613",
        "name":"[redacted]","screen_name":"[redacted]",
        "location":"byram twp, new jersey","time_zone":"E
        "statuses_count":42884,"lang":"en"},
    "entities":{"hashtags":[],"symbols":[],"user_mentions
```

JSON File



```
XP # 1
v 1 "text"
v 2"en"
d 1 2 "lang"
v 3 "user"
v 4"Eastern Time (US & Canada)"
d 3 4 "tz"
d 3 1 "tweet"
v 5"en"
d 3 5 "lang"
v 6 "user"
d 3 6 "mention"
v 7 "text"
v 8"en"
d 7 8 "lang"
v 9 "user"
v 10"Central Time (US & Canada)"
d 9 10 "tz"
d 9 7 "tweet"
v 11"en"
d 9 11 "lang"
d 7 1 "retweet"
XP # 2
v 1 "text"
v 2"en"
d 1 2 "lang"
v 3 "user"
```

Graph File

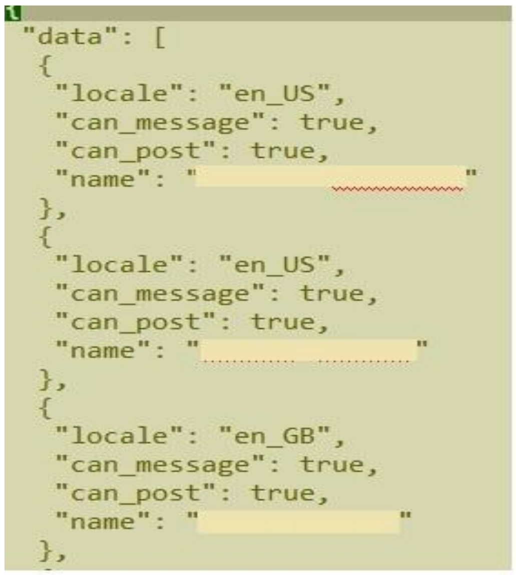
Fig 3.2 Output of Twitter API Module and Twitter Parser

3.2.2 Facebook Parser

The Facebook Parser takes the JSON file generated by the Facebook FQL Request module and converts it into a graph. The command to run this module is:

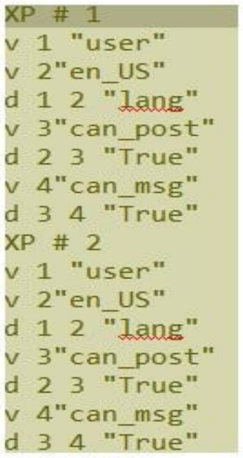
python fbparser.py -s <SOURCE FILE> -g <GRAPH FILE>

The first option is the name of the JSON source file and the second option is the name of graph file we are trying to generate. An example of portions of a JSON file and a graph file are shown in the Fig 3.3.



```
"data": [
  {
   "locale": "en_US",
   "can_message": true,
   "can_post": true,
   "name": "[redacted]"
  },
  {
   "locale": "en_US",
   "can_message": true,
   "can_post": true,
   "name": "[redacted]"
  },
  {
   "locale": "en_GB",
   "can_message": true,
   "can_post": true,
   "name": "[redacted]"
  },
```

JSON File



```
XP # 1
v 1 "user"
v 2"en_US"
d 1 2 "lang"
v 3"can_post"
d 2 3 "True"
v 4"can_msg"
d 3 4 "True"
XP # 2
v 1 "user"
v 2"en_US"
d 1 2 "lang"
v 3"can_post"
d 2 3 "True"
v 4"can_msg"
d 3 4 "True"
```

Graph File

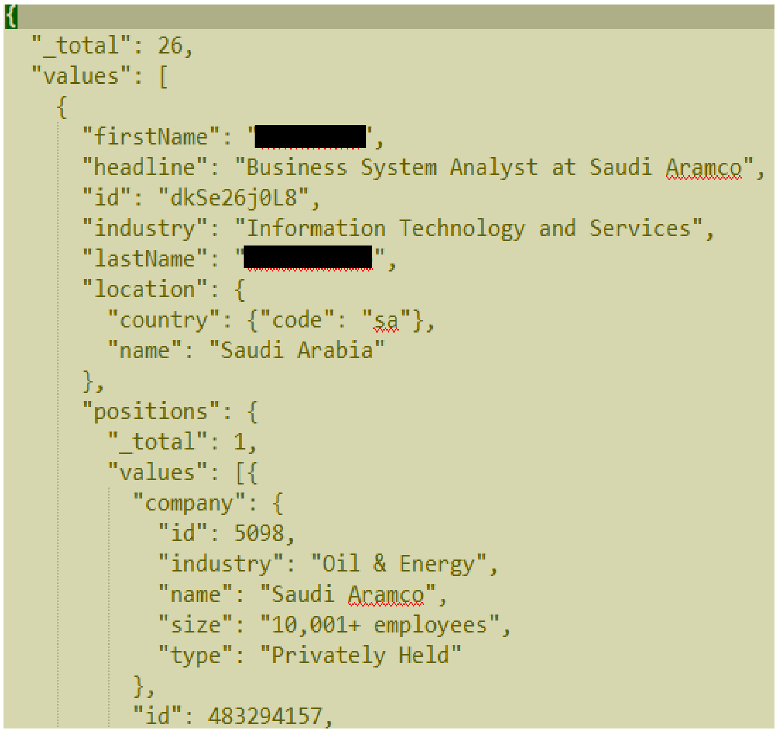
Fig 3.3 Output of Facebook FQL Request Module and Facebook Parser

3.2.3 LinkedIn Parser

The LinkedIn Parser takes the JSON file generated by the LinkedIn API module and converts into a graph. The command to run this module is:

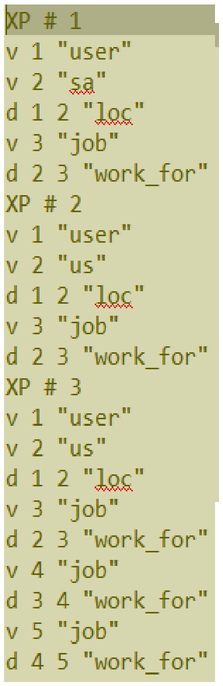
python lkparser.py -s <SOURCE FILE> -g <GRAPH FILE>

The first option is the name of the JSON source file and the second option is the name of graph file we are trying to generate. An example portion of a JSON file and a graph file are shown in the Fig 3.4.



```
{
  "_total": 26,
  "values": [
    {
      "firstName": "[redacted]",
      "headline": "Business System Analyst at Saudi Aramco",
      "id": "dkSe26j0L8",
      "industry": "Information Technology and Services",
      "lastName": "[redacted]",
      "location": {
        "country": {"code": "sa"},
        "name": "Saudi Arabia"
      },
      "positions": {
        "_total": 1,
        "values": [{
          "company": {
            "id": 5098,
            "industry": "Oil & Energy",
            "name": "Saudi Aramco",
            "size": "10,001+ employees",
            "type": "Privately Held"
          },
          "id": 483294157,
```

JSON File



```
XP # 1
v 1 "user"
v 2 "sa"
d 1 2 "loc"
v 3 "job"
d 2 3 "work_for"
XP # 2
v 1 "user"
v 2 "us"
d 1 2 "loc"
v 3 "job"
d 2 3 "work_for"
XP # 3
v 1 "user"
v 2 "us"
d 1 2 "loc"
v 3 "job"
d 2 3 "work_for"
v 4 "job"
d 3 4 "work_for"
v 5 "job"
d 4 5 "work_for"
```

Graph File

Fig 3.4 Output of LinkedIn API Module and LinkedIn Parser

The flowchart of the system which can be found in Appendix A explains the detail flow of the LinkCube.

Chapter 4 - Project Implementation

LinkCube is written using the Python programming language in a Linux operating system. It is command-line based tool. The system specification about the project implementation and its testing is discussed in the following sections.

4.1 System Specification

4.1.1 Software Requirement

The necessary software for the project development are as follows:

- Python 2.7.6 for system development.
- Python plug-ins like facebook-sdk, requests, tweepy and linkedin.
- Ubuntu 12.04 64-bit version as an operation system.
- GBAD for testing the results.
- XDot for visualizing graphs generated by LinkCube.
- EdrawMax for creating figures.

4.1.2 Hardware Requirement

This project requires hardware that supports the above software. However, for the purpose of system development and design the following specifications were used:

- Computer with Inter(R) Core 2 Duo CPU E8400 @ 3.00 GHz x 2.
- 2.0 GB Ram.
- 160 GB Hard Disk.
- XVGA graphics capable screen supporting resolution of 1280 × 800 pixel.
- Internet connectable system.

4.1.3 Experience Requirement

The following technical experience is required for the development and use of LinkCube:

- Extensive knowledge of Python programming language.
- Social network analysis.
- Web applications and HTTP.
- Graph Based Anomaly Detection (GBAD).
- Knowledge of Twitter, Facebook and LinkedIn API’s.

4.2 System Testing

The purpose of testing is to confirm that the system satisfies requirements and functions without error. Testing of LinkCube was performed by using various methods throughout the development cycle with a single goal: to uncover errors in content, function, and performance.

4.2.1 Unit Testing

The first form of testing is unit testing. Unit testing is the process of ensuring the individual components of an application work as intended. LinkCube is designed as a collection of subsystems that can function independently. All the modules in data extraction layer and data processing layer are developed independently and work as a single unit. Therefore the process of unit testing is very straightforward. This meant each module of the system was scrutinized and tested in isolation without considering the impact of other modules in the same layer or another layer.

4.2.2 Requirement Testing

Another testing process is requirement testing. It is a continuous process that was carried throughout the life of the software development process. The requirement testing involved testing whether or not the software was satisfying the goal stated at the beginning of the project (can be found in section 1.2 of this report). It not only tested whether the stated goal was met but also helped to keep track of the progress during the software development stage.

The main goal of the project LinkCube was to generate a GBAD compatible graph from social network data. The JSON file returned by the data extraction layer is used as input to the data processing layer. For each module of Twitter, LinkedIn and Facebook the JSON file from their respective module in the data extraction layer are provided to their respective parser. The graph file is then generated by the parser. The testing is done on these graphs. The GBAD algorithm is run on these graphs and checked if the output graphs are in desired format. If after running the graphs on GBAD, it doesn't gets an error then the graphs are compatible.

Furthermore the graph generated by the data processing layer of LinkCube was tested for anomaly detection using GBAD. Fig 4.1 shows an example of the structure of a graph generated by LinkCube from Twitter data. All three GBAD algorithms, probabilistic, maximum partial substructure and MDL (Anomalous Modification) were tested on the graph. For example, the MDL algorithm detected the anomalous instant consisting of a modified vertex name as shown in Fig 4.2.

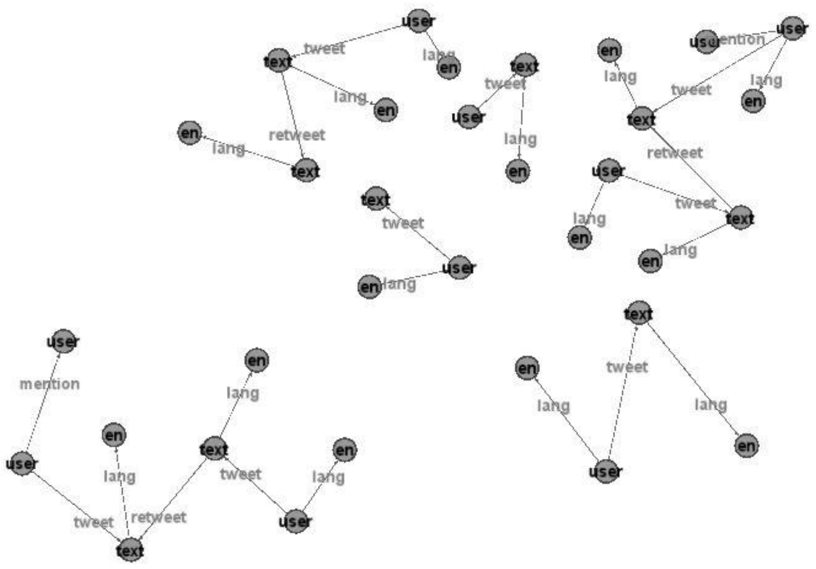


Fig 4.1 Part of Twitter Network Graph

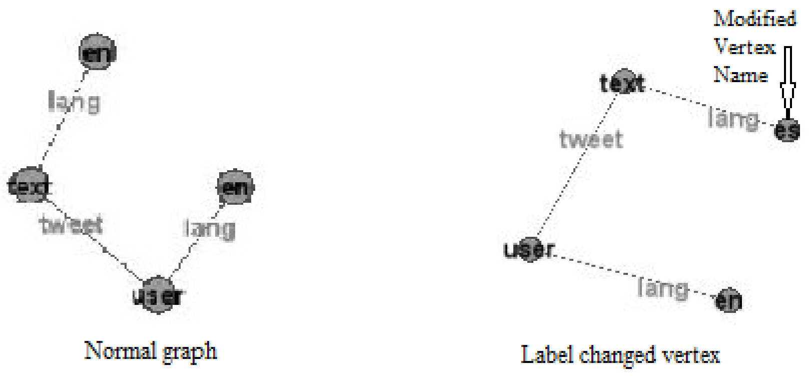


Fig 4.2 Anomalous Instance Detected by MDL Algorithm in Twitter Network

Similarly, the Maximum Partial Substructure (MPS) algorithm also found an anomalous deletion instance in the Twitter network. The anomalous instance was the occurrence of a substructure with a missing vertex. Fig 4.3 shows the normal substructure and anomalous substructure in a twitter graph after running the MPS algorithm.

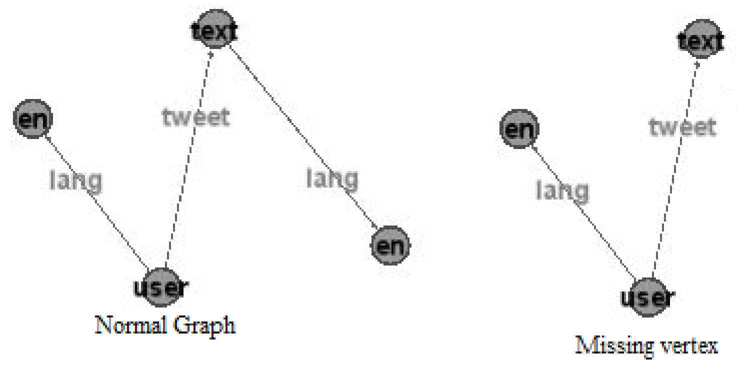


Fig 4.3 Normal and Anomalous Substructure Generated by MPS in Twitter graph

The Facebook and LinkedIn social network graph constructed by their respective parsers in the data processing layer have also resulted in anomalies being detected by GBAD. The results of running these two network graphs using the MDL (Anomalous Modification) algorithm in GBAD are shown in Figures 4.4 and 4.5.

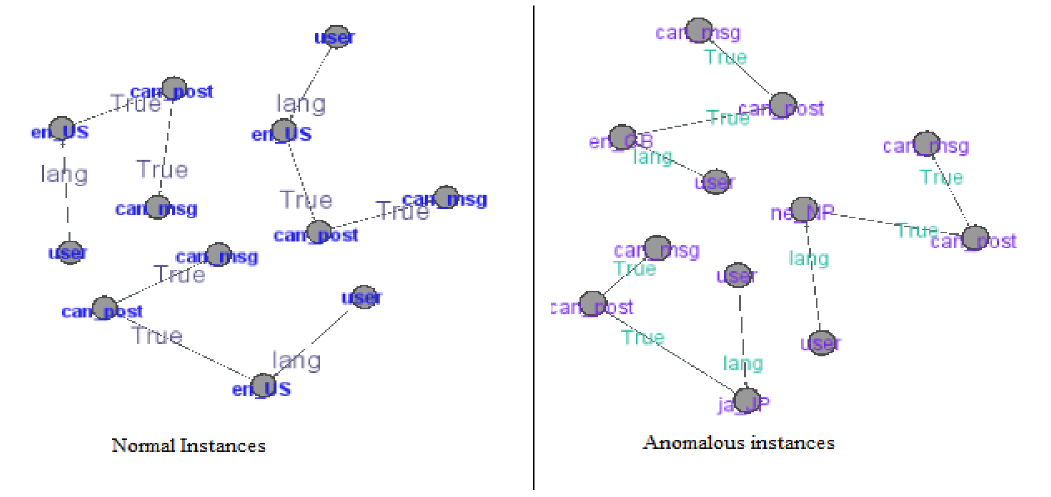


Fig 4.4 Normal and Anomalous Substructure Generated by MDL in Facebook graph

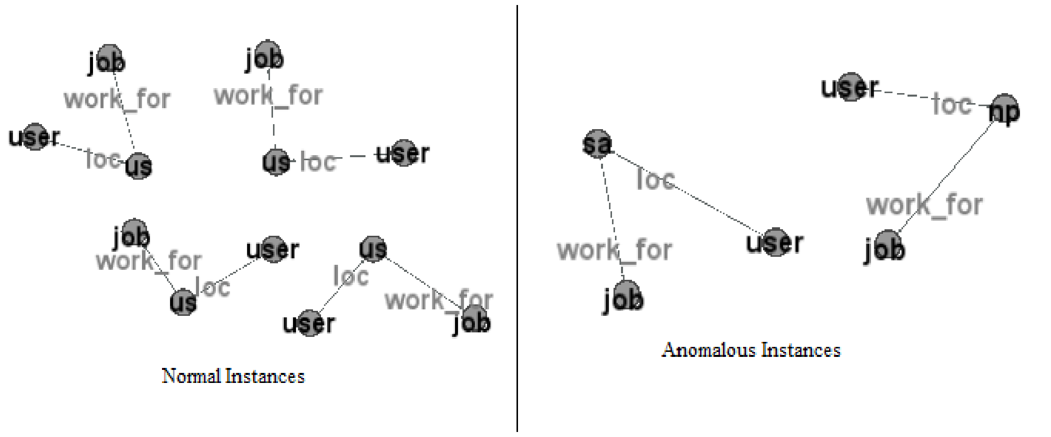


Fig 4.5 Normal and Anomalous Substructure Generated by MDL in LinkedIn graph

Chapter 5 - Project Management

5.1 Planning

The planning process gives an idea of how the system will be developed. It is characterized for defining the design and development model to be used, listing the various design guidelines of the system and assigning the time boundary of the whole project. For the development of the project entitled LinkCube, we used the Waterfall Model as shown in Fig 5.1. The first aspect of the Waterfall Model of design is to break the project goal down into a number of tasks and subtasks. Each of the tasks are assigned start and finish times by estimating their time of completion as well considering bottlenecks that may occur during the process of completing these subtasks [21].

During the planning stage any foreseeable problems that may arise at any point throughout the project are taken into consideration. To identify these problems, problem analysis is done which is very helpful to divide the project goal into sub tasks and plan their time frames. The sub tasks thus identified are used to produce the Gantt chart, which can be found in Appendix B. The Gantt chart gives a clear breakdown of the tasks and their detailed schedule. The Gantt chart is also crucial in keeping track of the project goal and the deliverables of the each subtask. To reflect the deviations from the original plan the Gantt chart is updated during the duration of the project.

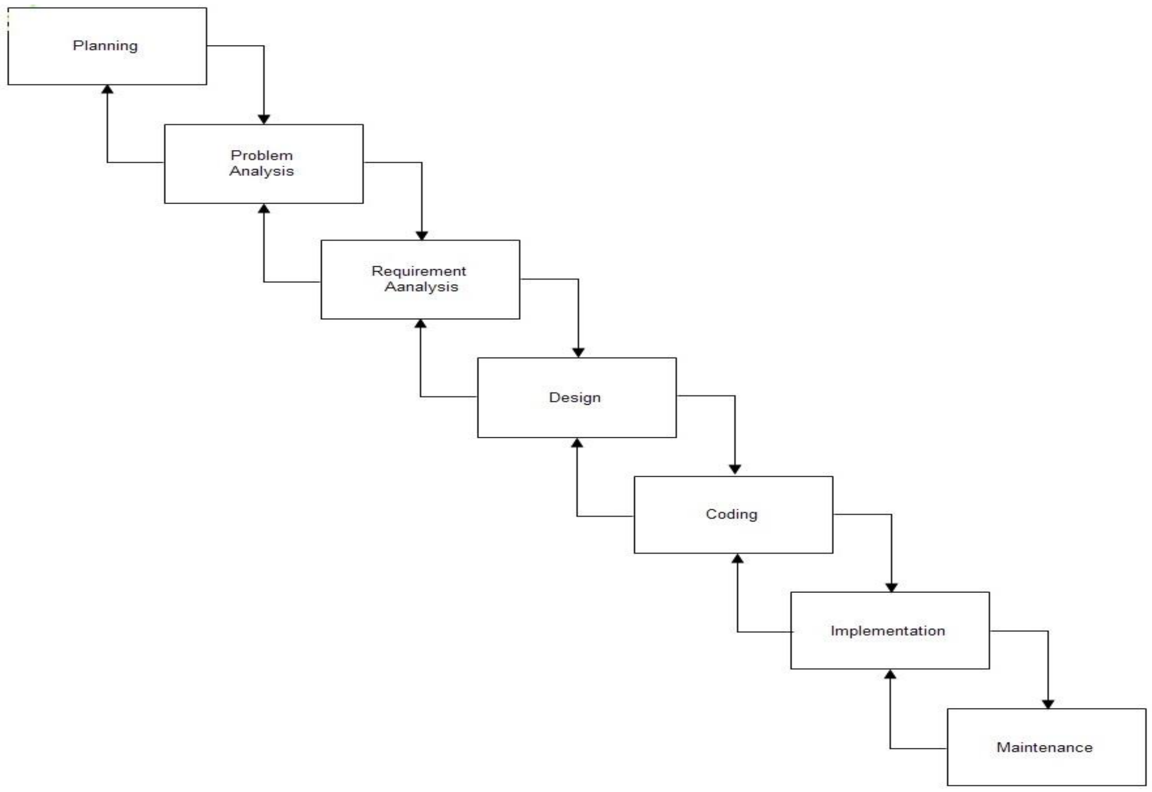


Fig 5.1 Waterflow Development Model

5.2 Management

Project management deals with managing changes during the course of project work. A folder structure was maintained to keep a backup of all reports, code, diagrams and notes in a dropbox. Also a backup of every major version were kept. This was done not only to minimize the impact of lost work (drive failure, theft, etc) but also to refer back to previous working versions if there is any faulty code needing to be reverted to a previous state.

Weekly advisory meetings with my project advisor were attended to develop ideas and ensure the project was keeping on track. Coding standards were adhered to during the development of the project, keeping uniform layouts and styles throughout all source files to increase readability including the use of informative comments. Since the code is written in python indentation plays a vital role for the compiler. The code is written with a proper indentation of 4 throughout the project. Proper naming convention is followed for naming variable and functions. The name of variables and functions are descriptive and is mixed case starting with lower case. Named constants are all uppercase using an underscore to separate the words, and abbreviations were avoided. Variables are initialized where they are declared and any potential dual meanings of variable were avoided. Use of global variables is minimized and is declared in the smallest scope possible. All comments are written in English and are included relative to their position in the code.

Chapter 6 - Conclusion

In the end, the success of a project comes down to one thing: did it work? With all requirements, aims and objectives satisfied I consider this project to be a success. All the modules were successfully tested and verified to be working. Working on LinkCube has proved to be a challenging yet extremely rewarding experience. Implementing a software development project on this scale I have gotten the chance to apply the knowledge gained over the course of the past few years and also learned new practices along the way.

6.1 Future Work

There were a few problem encountered during the implementation of this project. Firstly, Twitter allows access to their data but Facebook and LinkedIn has lots of privacy and security implemented to their data. Getting access to these data from Facebook and LinkedIn was challenging so we implemented LinkCube only on small datasets from these two social networks. One possibility for future work is to implement LinkCube on large datasets of Facebook and LinkedIn.

Secondly, LinkCube is command line tool so the user should know how to execute command-line tools (as opposed to GUI tools) beforehand. A possibility for future work is to design a GUI for LinkCube so that it will provide a simple user interface.

References:

[1]William Eberle, Lawrence Holder, "Anomaly Detection in Data Represented as Graphs." *Intelligent Data Analysis: An International Journal*. Volume 11, Number 6, pp. 663-689. 2007.

[2] Caleb C. Noble, Diane J. Cook, “Graph-Based Anomaly Detection”, in *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining,* New York, 2003

[3] http://www.sciencedaily.com/releases/2013/05/130522085217.htm

[4] https://Twitter.com/

[5] Papadimitriou, Panagiotis, Ali Dasdan, and Hector Garcia-Molina. "Web graph similarity for anomaly detection." *Journal of Internet Services and Applications*1.1 (2010): 19-30.

[6] http://docs.oracle.com/cd/B28359_01/datamine.111/b28129/anomalies.htm#CIHGAJAE

[7] http://en.wikipedia.org/wiki/Anomaly_detection

[8] William Eberle, Lawrence Holder, Jeffery Graves, “Insider Threat detection using graph based approach”, *Journal of Applied Security Research,* Vol. 6, Issue 1, January 2011.

[9] William Eberle, Lawrence Holder, “Applying Graph-based Anomaly Detection Approaches to the Discovery of Insider Threats”, *IEEE International Conference on Intelligence and Security Informatics (ISI)*, June 2009.

[10] William Eberle, Lawrence Holder, Jeffrey Graves, “Using a Graph-Based Approach for Discovering Cybercrime”, *International Conference of the Florida AI Research Society (FLAIRS)*, Florida, May 2010.

[11] William Eberle, Lawrence Holder, Beverly Massengill, "Graph-Based Anomaly Detection Applied to Homeland Security Cargo Screening", *International Conference of the Florida AI Research Society (FLAIRS)*, May 2012.

[12] https://dev.Twitter.com

[13] https://www.Facebook.com/about/pages

[14] http://fbdevwiki.com/wiki/FQL:stream

[15] https://developer.linkedin.com/documents

[16] Gundecha, Pritam, and Huan Liu. "Mining social media: A brief introduction."*Tutorials in Operations Research* 1.4 (2012).

[17] http://www.flickr.com/

[18] http://cyworld.co.kr/

[19] https://plus.google.com

[20] http://www.orkut.com/

[21] http://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm

[22] http://en.wikipedia.org/wiki/Social_networking_service

Appendix A - **Flow Chart**

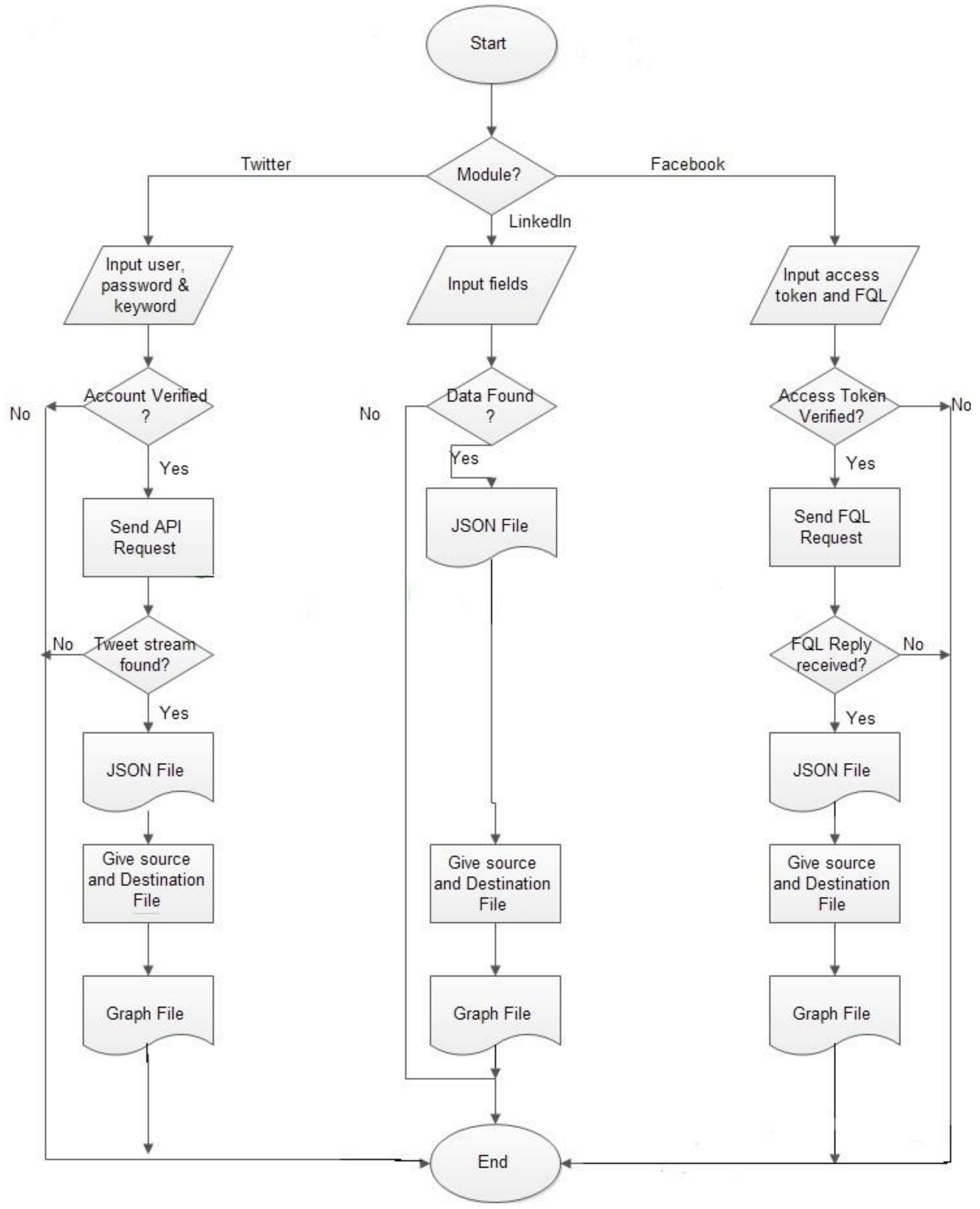


Fig A.1 Flowchart of the system

Appendix B - Gantt chart

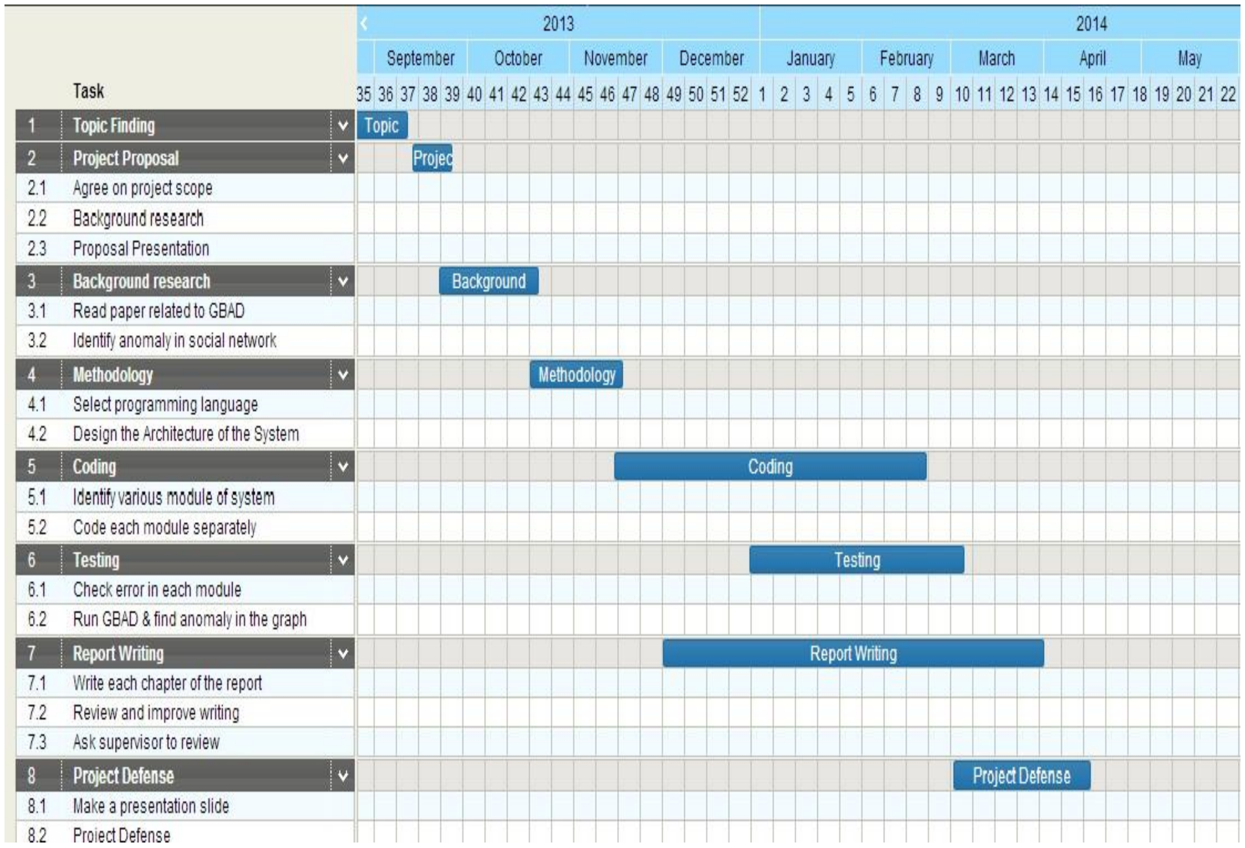


Fig A.1 Gantt Chart