Stock Crime Detection using Graph Mining

Jigyasha Arora
CSE Dept,
UTU, Dehradun.

Pawan Kumar Mishra
CSE Dept,
UTU, Dehradun.

Prakash Joshi
CSE Dept,
UTU, Dehradun.

ABSTRACT

Previously existing graph mining algorithm typically assumes that database is relatively static. To overcome that we proposed a new algorithm which deals with large database including the features which captures the properties of graph in few parameters and check the relationship among them in both left as well as right direction, thus adopting DFS as well as BFS approach. It further finds the sub graph by traversing the graph and extracting the desired pattern. The proposed algorithm is used for detection of crime in stock market by capturing the properties and identifying the relationship & associations that may exist between the person involved in that crime which prevent several crimes that might occur in future. We have used the ECLIPSE for the implementation of proposed algorithm and Neo4j is the graph database used for analysis.

Keywords

Graph database, Graph mining, DFSS, Sub graph.

1. INTRODUCTION

Nowadays the amount of data is increasing day by day, so accordingly the desire for data mining is also growing. Large database have to be searched to find the interesting properties of the graph and to establish a relationship among them. It is beneficial to model the complex data with the help of graph in which information is stored in nodes and edges represent the relationship among the nodes. Therefore having a graph database overcomes the constraint of relational database and helps in finding the super graph, sub graph, common graph and relationship between different graphs. This graph based data mining has become more and more popular in the last few years. Graph mining is the use of most important structure of graph to obtain frequent patterns of information. It has board range of applications.

This technique can be used to find the probability of persons doing crime in the stock market .Some case studies of people involved in stock crime were studied to obtain the attributes such as persons involved in crime, whether they are educated or not, style of crime, earning from the particular threat. These attribute lead to the construction of graph database and an algorithm has been proposed for traversing the graph in both directions left as well as right and establish relationship among different nodes which further generates a sub graph according to the query.Neo4j is the graph database used for analysis as the retrieval times of graph database are less than relational database as it looks only at records, it does not scan the entire group to find the nodes that met the search criteria. Analysis report from this implementation will also be helpful in planning the prevention of several crimes. The remainder of this paper is organized as follows. Section 2 introduces the problem statement of graph based data mining and existing algorithms; Section 3 describes our proposed algorithm used for traversing the graph database; Section 4 describes comparative study of our proposed technique with other existing technique; Conclusion & future scope in Section 5 and all the used references are given in Section 6.

2. OVERVIEW OF EXISTING ALGORITHM

2.1 Part Miner Algorithm

Each graph in the database is partitioned into smaller sub graphs. Part Miner can effectively reduce the number of candidate graphs by exploring the cumulative information of the units. This has led to a lot of cost savings. PartMiner is effective and scalable in finding sub graphs.

```
AlgorithmGraph Part
Input: G, the graph
Output: G1, G2, the two subgraphs of G
1: V = {vertices sorted according to their update frequency};
2: V*= Φ;
3: w (V*) = −∞
4: for (i = 0; i < |V |/2; i++) {
5: Vi = Φ;
6: call DFSScan(V, i, Vi);
7: Compute w(Vi);
8: if (w (Vi) > w(V∗ )) {
9: w (V*) = w (Vi);
10: V*= Vi;
11: }
12: }
13: G1 = {eij = (vi, vj )|vi ∈ V*, vj ∈ V*}∪{eij = (vi, vj )|vi ∈ V*, vj /∈V*}
14: G2 = {eij = (vi, vj )|vi /∈V*, vj /∈V*}
          ∪ {eij = (vi, vj )|vi ∈ V*, vj /∈V*}

Procedure DFSScan(V, i, Vi)
15: stack = Φ,m= 0;
16: stack.push(vi);
17: while(stack ≠ Φ ∧ m ≤ |V |/2){
18: v = stack.pop();
19: Vi = Vi ∪ {v};
20: m++;
21: choose the neighbor vertex vh, s.t. vh.visited = 0, and ∀vs,
    Vs.visited = 0 ∧ (v, vs) ∈ E, vs.ufreq < vh.ufreq;
22: stack.push(vh);
```

```
23 :}
Dividing graph database into units
Procedure DBPartition(D, k)
D, graph database;
K: number of units
1: D0,0 = D;
2: i = 1;
3: l = log2k;
4: while (i ≤ l) {
5: for (j = 0; j < 2i−1; j++)
6: DivideDBPart(Di-1,j,Di-2,j,Di,2j+1);
7: i++;
8 :}
9: for (j = 0; j < k – 2l; j ++)
10: DivideDBPart(Di-1,j , U2j , U2j+1);
Function DivideDBPart(Ds, D1,0, D1,1)
1: D1, 1 =Φ;
2: D1, 1 = Φ;
3: for each graph G ∈ Ds {
4: G1, G2 = calling GraphPart(G);
5: D1, 0= D1, 0 ∪ {G1};
6: D1, 1 = D1, 1 ∪ {G2}
```

2.2 gSpan Algorithm

Graph-Based Substructure Pattern Mining, which presented gSpan algorithm which discovers frequent substructures without candidate generation. gSpan builds a new lexicographic order among graphs ,and maps each graph to a unique minimum DFS code as its canonical label. Based on this lexicographic order, gSpan adopts the depth-first search strategy to mine frequent connected subgraphs efficiently. So, gSpan outperforms FSG by an order of magnitude and is capable to mine large frequent subgraphs in a bigger graph set with lower minimum supports.

GraphSet Projection (D,S).

```
1: sort the labels in by their frequency;
2: remove infrequent vertices and edges;
3: relabel the remaining vertices and edges;
4: $S^1$= all frequent 1-edge graphs in D ;
5: sort $S^1$ in DFS lexicographic order;
6: S →$S^1$
7: for each edge e € S1 do
8: initialize s with e, set S. D by graph which contains e
9: Subgraph Mining (D, S, s);
10: .D←D-e
11: if │D│< min Sup
12: break;
```

```
Subprocedure 1 Subgraph Mining(D,S,s)
1: if s ≠ min(S)
3: S←S U {s}
4: enumerate s in each graph in D and count its children;
5: for each c, c is s' child do
6: if support (C) > min Sup
7: s ←c
8: Subgraph Mining (D, S, s_);
```

2.3 RMAT Algorithm

In this recursive model for graph mining finding the properties of real graphs that seem to persist over multiple disciplines. We list such "laws" and, more importantly, we propose a simple, parsimonious model, the recursive matrix (R-MAT) model, which can quickly generate realistic graphs, capturing the essence of each graph in only a few parameters. R-MAT automatically generates graphs with the communities within communities' property. R-MAT can easily generate realistic weighted graphs directed graphs and bipartite graphs.

RMAT Algorithm

The adjacency matrix A of a graph of N nodes is an N _ N matrix, with entry a (i; j) = 1 if the edge(i; j) exists, and 0 otherwise. The basic idea behind R-MAT is to recursively subdivide the adjacency matrix into four equal-sized partitions, and distribute edges within these partitions with a unequal probabilities: starting o_ with an empty adjacency matrix, we \drop" edges into the matrix one at a time. Each edge chooses one of the four partitions with probabilities a; b; c; d respectively (see Figure 1). Of course, a + b + c + d = 1. The chosen partition is again subdivided into four smaller partitions, and the procedure is repeated until we reach a simple cell (=1_1 partition).This is the cell of the adjacency matrix occupied by the edge. The number of nodes in the R-MAT graph is set to 2n; typically n = dlog2 Ne. There is a subtle point here: we may have duplicate edges (i.e., edges which fall into the same cell in the adjacency matrix), but we only keep one of them. To smooth out actuations in the degree distributions, we add some noise to the (a; b; c; d) values at each stage of the recursion and then renormalize (so that a+b+c+d = 1).

2.4 gIndex Algorithm

Different from the existing path-based methods, our approach, called gIndex, makes use of frequent substructure as the basic indexing feature. Frequent substructures are ideal candidates since they explore the intrinsic characteristics of the data and are relatively stable to database updates.

Algorithm 1 Feature Selection

```
Input: Graph database D, Discriminative ratio,
Size-increasing support function,
Maximum fragment size maxL.
Output: Feature set F.
1: let F = { fΦ }, DfΦ = D, and l = 0;
2: while l <= maxL do
3: for each fragment x, whose size is l do
4: if x is frequent and discriminative then
5: F = F u {x}
```

```
6: l = l + 1;
7: return F;
```

Algorithm 2 Search

```
Input: Graph database D, Feature set F, Query q,
 Maximum fragment size maxL.
Output: Candidate answer set Cq.
1: let Cq = D;
2: for each fragment x is subset of  q and len(x) <=  maxL do
3: if x € F then
4: Cq = Cq Π Dx and return Cq.
```

Algorithm 3 Insert/Delete

```
Input: Graph database D, Feature set F,
Inserted (Deleted) graph g and its id gid,
Maximum  fragment  size maxL.
1: for each fragment x is subset of  g and len(x) <=  maxL do
2: if x € F then
3: Insert:
   insert gid into the id list of x;
4: Delete:
   delete gid from the id list of x;
5: return;
```

3. PROPOSED ALGORITHM

3.1 The proposed algorithm is better in performance than previous algorithm such as Part Miner, gSpan, RMAT and gIndex in terms of sorting and searching involving DFSS having both left and right relationship, graph property having user based query and relationship property. It includes the following steps.

1. Creation of nodes, property of nodes, and relationship among those nodes

2. Selection of property to be searched and sorting with the help of relationship.

3. Traversing to a particular node which needs to be searched in both left as well as right direction and store the relationship when the match occurred.

ALGORITHM FOR TRAVERSING

```
Setup
Step 1 Create Graph Database
Step 2 Create Node
Step 3 Set Property of nodes
Step 4 Create Relationship
Step 5 Select p              /* Property to be searched */
Step 6 Sort the graph by their relationship
Step 7 for Node position traverse <- depth
Step 8 if p == node.property // if required property        match
Step 9 S <- node-relationship // store relationship of    first
match
Step 10   if node.left.relationship ==S
Step 11   display properties
Step 12   continue traverse down
Step 13   else
Step 14   if node.right.relationship ==S
Step 15   Display properties
Step 16   continue traverse down
Step 17   else
Step 18     traverse <- down next node
Step 19    end
Step 20   end
Step 21   if p==node.property
Step 22    repeat Step 10 through18
Step 23   end
Step 24   end
Step 25   end
```

4. COMPARITIVE STUDY AND DISCUSSION

The proposed algorithm when compared with above existing algorithms works remarkably well in terms of parameters such as it stores the data in sorted way, searching takes place in both directions left as well as right, graph property based on user based query and also checks the relation whether it is one to one, one to many or many to many relationship. The comparison of existing algorithm with proposed algorithm is shown in Table 1.

Table1. Comparison of existing algorithm with proposed algorithm

Features	Part Miner	gSpan	RMAT	gIndex	Proposed Algo
Sorting	Yes	Yes	No	No	Yes
Approach	Top down	Top down	Top down	Top down	Top down
Search	DFSS	DFSS	DFSS	DFSS	DFSS, Left and Right Relationship
Partitioning	Yes	No	Yes (recursively)	Yes (recursive)	No
Large Database	Average	Good	Average	Average	Good
Graph Property	No	No	No	Feature based	Yes(user Query based)
Check Relationship	No	No (DFS code)	No	No	Yes
Iteration	Multiple	One	Multiple	Multiple	One

Some snapshots have been taken from the graph database to show the properties and relationship that exit among the different nodes such as BSE (Bombay Stock Exchange), NSE (National Stock Exchange) and NASDAQ.A Overview of Stock Crime Exchange is shown in figure 1.Stock Crime that has occurred inside NASDAQ is shown in figure2. Stock Crime that has occurred inside NSE is shown in figure3. Stock Crime that has occurred inside BSE is shown in figure4.Stock crime in NASDAQ having both has and knows relationship is shown in figure 5. Stock crime in NASDAQ having both has and knows relationship is shown in figure6. Stock crime in NASDAQ having both has and knows relationship is shown in figure7. Common node having relationship both in NSE & NASDAQ is shown in figure8 & figure9 which can be shown with the help of id which is common to both NSE & NASDAQ.

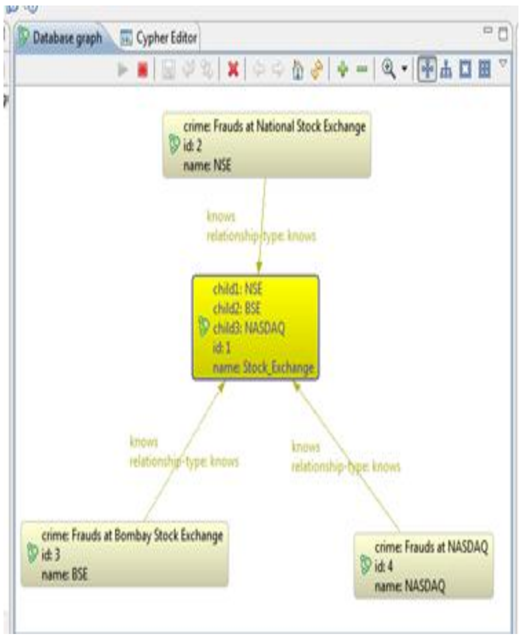

Figure1.Overview of Stock Crime Exchange

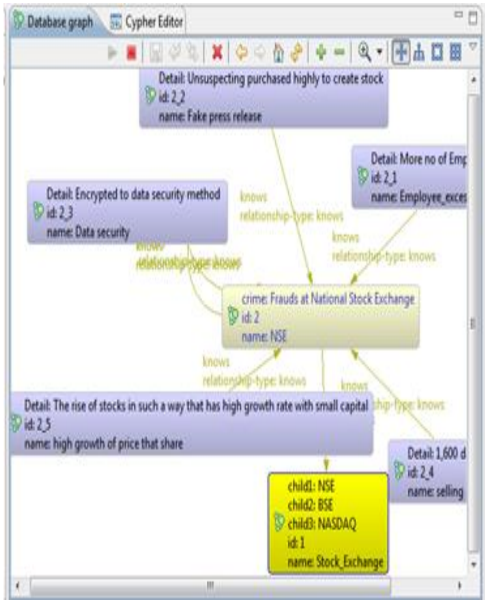

Figure2.Stock Crime that has occurred in NSE

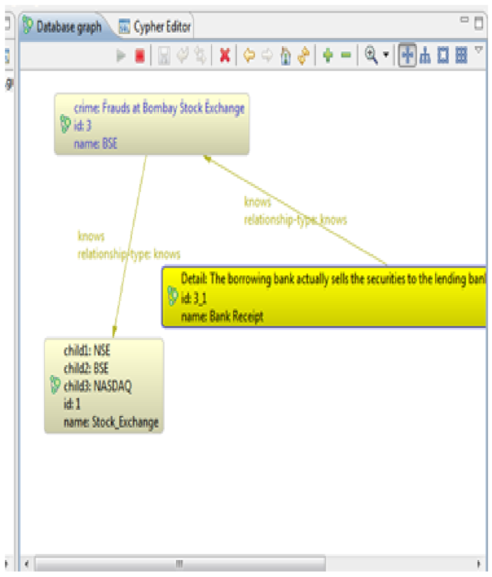

Figure3.Stock Crime that has occurred in BSE

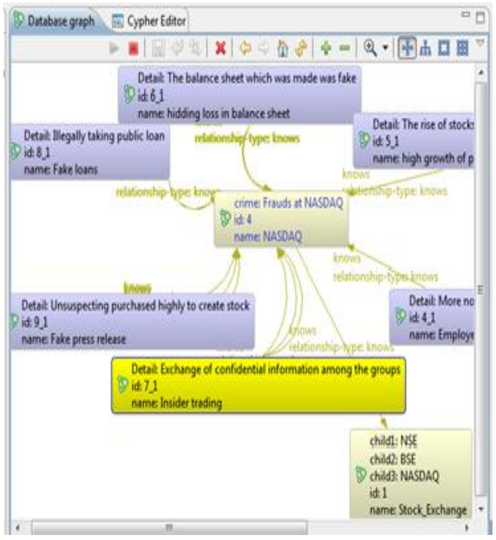

Figure4.Stock Crime that has occurred in NASDAQ

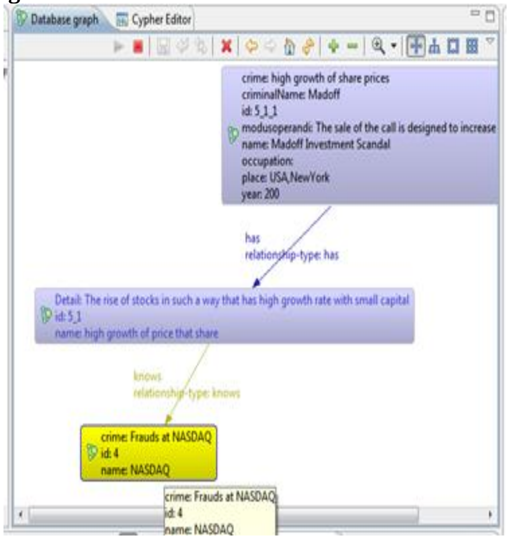
Database graph
Cypher Editor
crime: high growth of share prices
criminalName: Madoff
id: 5_1_1
modusoperandi: The sale of the call is designed to increase
name: Madoff Investment Scandal
occupation:
place: USA,NewYork
year: 200
has
relationship-type: has
Detail: The rise of stocks in such a way that has high growth rate with small capital
id: 5_1
name: high growth of price that share
knows
relationship-type: knows
crime: Frauds at NASDAQ
id: 4
name: NASDAQ
crime: Frauds at NASDAQ
id: 4
name: NASDAQ

Figure5.Stock Crime in NASDAQ having both knows & has relationship

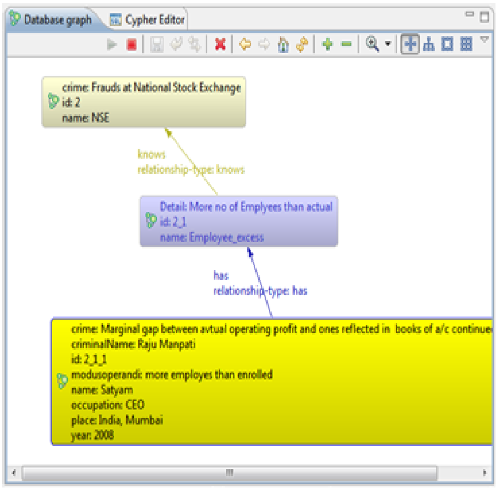
Database graph
Cypher Editor
crime: Frauds at National Stock Exchange
id: 2
name: NSE
knows
relationship-type: knows
Detail: More no of Emplyees than actual
id: 2_1
name: Employee_excess
has
relationship-type: has
crime: Marginal gap between avtual operating profit and ones reflected in books of a/c continue
criminalName: Raju Manpati
id: 2_1_1
modusoperandi: more employes than enrolled
name: Satyam
occupation: CEO
place: India, Mumbai
year: 2008

Figure6.Stock Crime in NSE having both knows & has relationship

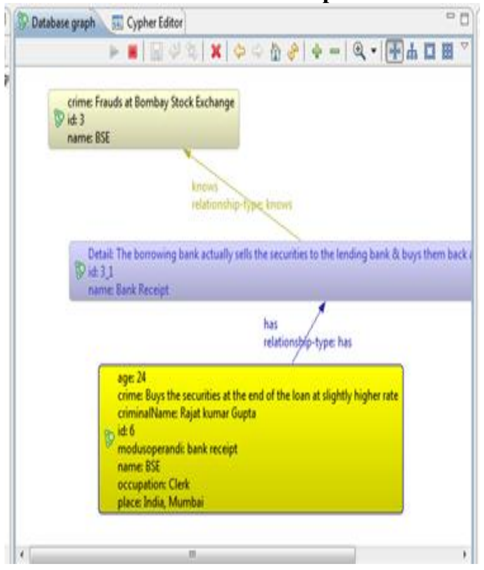
Database graph
Cypher Editor
crime: Frauds at Bombay Stock Exchange
id: 3
name: BSE
knows
relationship-type: knows
Detail: The borrowing bank actually sells the securities to the lending bank & buys them back
id: 3_1
name: Bank Receipt
has
relationship-type: has
age: 24
crime: Buys the securities at the end of the loan at slightly higher rate
criminalName: Rajat kumar Gupta
id: 6
modusoperandi: bank receipt
name: BSE
occupation: Clerk
place: India, Mumbai

Figure7.Stock Crime in BSE having both has & knows relationship

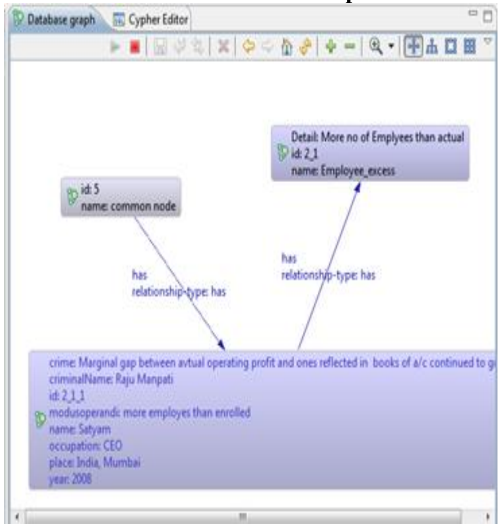
Database graph
Cypher Editor
Detail: More no of Emplyees than actual
id: 2_1
name: Employee_excess
id: 5
name: common node
has
relationship-type: has
has
relationship-type: has
crime: Marginal gap between avtual operating profit and ones reflected in books of a/c continued to g
criminalName: Raju Manpati
id: 2_1_1
modusoperandi: more employes than enrolled
name: Satyam
occupation: CEO
place: India, Mumbai
year: 2008

Figure8.Stock Crime in BSE having common id 5 & related to same crime that has occurred in NASDAQ

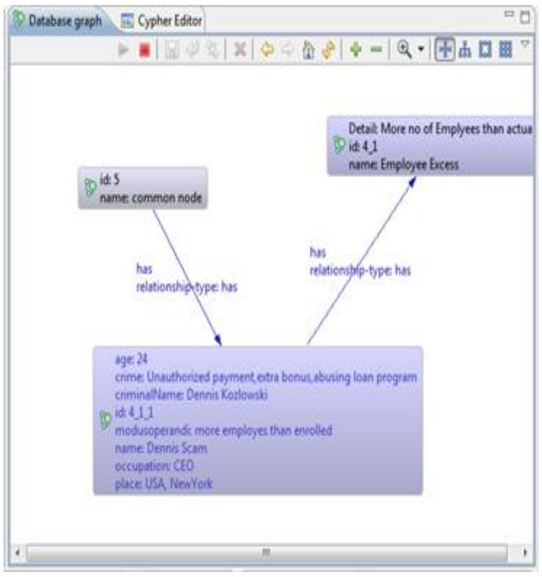

Figure9.Stock Crime that in NASDAQ having common id 5 & related to same crime that has occurred in BSE

5. CONCLUSION & FUTURE SCOPE

Although the current algorithm already performs quite well, it can be implemented in real time systems to trace the pattern of stock rise and fall in the share market and we can compare the current pattern of stock fluctuations with the pattern present in graph database, so that if it finds any resemblance in the pattern it can impose a security check over that particular stock and predict the future steps .This can be helpful in planning the prevention of several crimes which can contribute to the people who gets affected due to this share price manipulation. Graph mining is a currently very active research field. The application areas of graph mining are widespread ranging from biology & chemistry to internet applications.

6. REFERENCES

[1] Kamrul Abedin Tarafder, Shah Mostafa Khaled, moham Ashraful Islam," Reverse Apriori algorithm for frequent pattern mining , Medwell journals, 2008.

[2] Usama Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth," From Data Mining to Knowledge Discover Databases", AI MagazineVolume 17 Number 3 (1996) (© AAAI)

[3] Justin J. Miller,"Graph Database Applications and Concept with Neo4j",Proceedings of the Southern Association for Information System Conference, Atlanta, GA, USA March 23rd- 24th, 2013

[4] Ingrid Fischer and Thorsten Meinl,"Graph Based Molecular Data Mining - An Overview",∗ 0-7803-8566-7/04/$20.00 c.2004 IEEE

[5] Xifeng Yan and Jiawei Han,"gSpan: Graph-BasedSubstructure"http://oldwww.comlab.ox.ac.uk/oucl/groups/machlearn/PTE

[6] Deepayan Chakrabarti, Yiping Zhan and Christos Faloutsos,"R- MAT: A Recursive Model for Graph Mining", ‡ School of Computer Science, CMU.

[7] Frank Eichinger,Klemens B¨ohm and Matthias Huber, Improved Software Fault Detection with Graph Mining", Appearing in the 6th International Workshop on Mining and Learning with Graphs, Helsinki, Finland, 2008.

[8] JunmeiWang, WynneHsu Mong and Li Lee Chang Sheng,” “A Partition-Based Approach to Graph Mining”, Proceedings of the 22nd International Conference on Data Engineering (ICDE’06)8-7695-2570-9/06 $20.00 © 2006 IEEE

[9] Garima Jaiswal and Arun Prakash Agrawal”Comparativeanalysis of Relational and Graph databases”, IOSR Journal of Engineering (IOSRJEN).

[10] Ciro Cattuto, André Panisson, Marco Quaggiotto and Alex Averbuch,”Time-varying Social Networks in a GraphDatabase”http://www.sociopatterns.org

[11] Quist-Aphetsi Kester,”Criminal Geographical Profiling: Using FCA for Visualization and Analysis of Crime Data”,Email: kquist-aphetsi@gtuc.edu.gh / kquist@ieee.org

[12] G. Kishore Kumar, Dr. V. K. Jayaraman ” Clustering of Complex Networks and Community Detection Using Group Search Optimization”.

[13] Hsinchun Chen, WingyanChung, Jennifer Jie Xu, Gang Wang Yi Qin and Michael Chau,” Crime Data Mining: AGeneral Framework and Some Examples”, 0018-9162/04/$20.00 © 2004 IEEE

[14] Tibor Bosse, Charlotte Gerritsen, and Jan Treur,” Analysis of Criminal Behaviour ”.

[15] Sytske Besemer,” The impact of timing and frequency of parental criminal behaviour and risk factors on offspring offending”, an Institute of Criminology University of Cambridge, Cambridge, UK Version of record first published: 05 Nov 2012.