## A Comparative Study on

# Coterie Identification in Facebook Network

Tarun Kumar Ravipati
travipati1@student.gsu.edu
Phone: +1 970-691-7474

Bhavya Induri
binduri1@student.gsu.edu
Phone: +1 470-430-1589

## ABSTRACT

The identification of peers in a social network has been analyzed over giving importance to recognition in the way of connection and interaction between people. The coterie identification can be done by the structure of Facebook public pages. There are a lot of activities people perform which are diverse from each other. Since the different activities can't be identified, we choose newsgroup pages and focus on extraction of one set of users. For this, we use Label Propagation Algorithm, Fast Unfolding and Spectral Clustering methods for identification and compare the results of each method to get a fair understanding of which method calculates the no.of user nodes effectively. These algorithms base the results upon the posts, likes and comments for each user activity. In order to get the effective values as outputs we eliminate the isolated and enhance the quality of the peer network across multiple pages with the use of subgraphs containing nodes that are more densely linked to each other than to the remaining nodes of the graph. The peer identification problem is tackled by the above mentioned algorithms.

## Keywords

"Modularity", "LPA", " Fast Unfolding", "Spectral Clustering", "Synchronous and Asynchronous updation", "Oscillation", "Community Detection".

## 1. INTRODUCTION

The no.of users that are using Facebook have grown drastically during the last decade. Monitoring the total count of the users has never been difficult. But, as the count increased, the relationships between various users on the network is being obscured. This is making the analyzers face the problem of wrong identification, incorrect mapping and mistaken inferences. Beyond the scale of the graphs, the construction of subgraphs from the subsequent nodes is becoming more difficult than others. A graph has a community structure if the number of links into any subgraph is higher than the number of links between those subgraphs i.e. it is the sub-network in a network which are highly interconnected nodes.

**Motivation:**

The no.of users that are using Facebook have grown drastically during the last decade. Monitoring the total count of the users has never been difficult because of various data analysis techniques. But, as the count increased, the relationships/ interests of various users on the network is being lost. The relationship type is absolutely necessary for the data visualization in the review procedure of the network. Due to large number of data nodes and multiple interests for each user, the analyzers face the problem of wrong identification, incorrect mapping and mistaken inferences. Beyond the scale of the graphs, the construction of subgraphs from the subsequent nodes is becoming more difficult than others. To overcome this few algorithm techniques have been implemented.

This project is a comparative study of three algorithms that can effectively be used for community detection in the large networks such as Facebook for instance. Hence, we moved on to implement the occurrence of the algorithms of LPA, Fast Unfolding and Spectral Clustering. These algorithms make sure that the network administrator can successfully identify all the coterie structures from the Facebook community pages.

## 2. METRICS

The measurement metrics for the identification of Coterie/Peers depends on four major parameters and each parameter has a certain metric associated to it.

1. Internal Connectivity
2. External Connectivity
3. Combined Internal and External Connectivity
4. Network model

**Internal Connectivity Metrics:**

1. Internal Density
2. FOMD
3. TPR

**External Connectivity Metrics:**

1. Expansion
2. Cut Ratio

**Combined Internal and External Connectivity:**

1. Conductance
2. Normalized Cut
3. Out Degree Fraction

**Network model:**

1. Separability
2. Modularity
3. Density
4. Cohesiveness

## 3. RELATED WORK

Previously some of the methods like spatial interface clustering have been employed to obtain graphs that are able to generate the relationships among the peers that are on the same network. But this lead to major disappointments in the object oriented list of results obtained. The distance between the graphs are bound to vary among a sequence of nodes but this doesn't mean that the calculated distances are miscalculated. Shockingly, the results obtained are illogical and didn't provide any proof that the values are correct. On top of this, the nodes in the network started to intensify with increase of user count. This lead to the compromise of conventional methods of the calculations. In order to design a working principle that needed the correction of the node detection many methods have been introduced but none stood up to the expectations of the network analysis.

**Limitations:**

1. Complexities differ by the size of the Dataset.
2. Inconsistencies in the sub-graph can't be identified because of the complex structure of the network.
3. Look-up models constructed to obtain clarity on the modularity values of various sub-graphs don't give consistent values and often deviate from expected results.

4. While indicating the spots in the network, the network drawn by the radius of the ego node is also considered again.

The sample network of Facebook connected via edges is represented as follows –

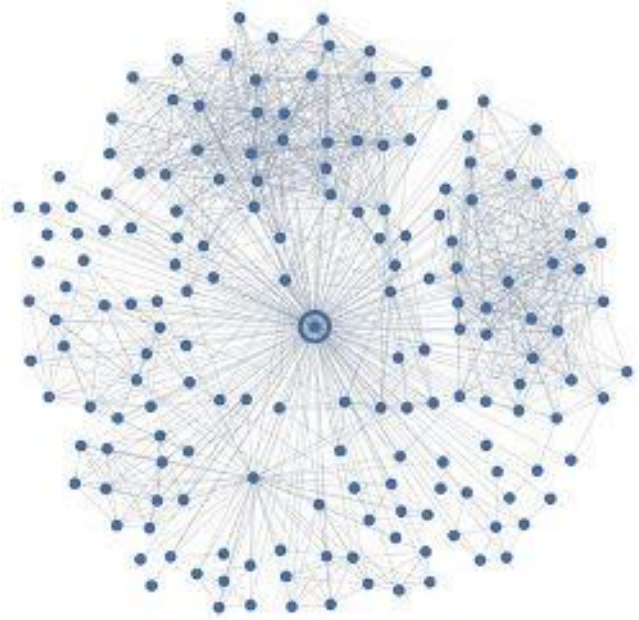


**Fig.1 Facebook data edge representation(of nodes)**

This network has been consistently undergoing changes that are inevitable with the count of all the breaking point values. The logical inferences of the Feistel Facebook structure might be compromised by giving up on the node connections, edge detection and cultural identification. The parameters that are to be recorded for this include edge count, node count, frequency analysis, operating node values, graph structure, opening clauses, updated structure,..etc. Based upon all of the above factors a node network will be identified which in-turn returns the values of the graphs that have accomplishment of director nodes. Of all the possible modularities the connectivity in the Facebook network is obtained as a whole unit rather than any optimized valued graphs that hide the true relationships among the original set of values. This optimized link-up possibility is used for optimization.

# 4. PROBLEM FORMULATION

**Input:** Facebook Dataset
**Output:** Communities detection in the ego network
**Equations:** Three type of equations are implemented resulting in the application of three algorithms that are to be implemented.
The major problems in the identification of the peer nodes are –

1. Huge content size
2. Lack of user scalability
3. Non-linear optimization
4. Low User-Aspire ratio
5. Larger implementation space

These five problems constitute for the major rupture of graph network thus taking a series of node structures away from each other. Each specific problem is identified as a single thread that doesn't combine to form a series of threads that destroys a graph structure representation. Each problem can be described as below–
**Huge Content size:** If the content size exceeds beyond a certain limit that is set during the construction of network, the peer nodes alter themselves leaving a proximity difference in the values of two or more sub-graphs belonging to the same network.
**Lack of User Scalability:** The combination of similar users over a period of time might result in a cluster with loops, devoiced edges and also many nodes that don't have a similarity with the remaining parts of the network.
**Non-Linear optimization:** In order to reduce the complexities on the graph we adjusted the node lengths and obtain linear set of optimizations thus resulting in the reduction of many loops in the graphs and dividing them to get a series of nodes.
**Low User-Aspire ratio:** The no.of users that are in the network can have the count less than the aspire value which is a part of the node influence factors.
If, U/A < 1, Nodes are distinct
If, U/A = 0, Nodes don't exist
If, U/A > 1, Nodes are cluttered, edges are looped
**Larger Implementation Space:** If the node percentage is beyond the capacity of the network, then there is a need for a larger implementation space. This space is required when handling a larger network such as the Facebook user community.

# 5. DATA SOURCE

For solving the above problem, we consider the Facebook data consisting of node, edges and no.of parameters that can be used for the identification of the adjoining peer nodes in the network.
**Dataset Name:** Facebook Data
This dataset consists of –
1. NodeID Edges
2. NodeID Circles
3. NodeID Features
4. Ego Network Features
5. NodeID Feature Names

**Description:**

1. **nodeId.edges:** The edges in the ego network for the node 'nodeId'. Edges are undirected for the facebook network. The 'ego' node does not appear, but it is assumed that they follow every node id that appears in this file.
2. **nodeId.circles:** The set of circles for the ego node. Each line contains one circle, consisting of a series of node ids. The first entry in each line is the name of the circle.
3. **nodeId.feat:** The features for each of the nodes that appears in the edge file.
4. **nodeId.egofeat:** The features for the ego user.
5. **nodeId.featnames:** The names of each of the feature dimensions. Features are '1' if the user has this property in their profile, and '0' otherwise. This file has been anonymized for Facebook users, since the names of the features would reveal private data.

**Statistics of the Dataset:**

1. Nodes – 4039
2. Edges – 88234
3. Nodes in largest WCC – 4039 (1.000)
4. Edges in largest WCC – 88234 (1.000)
5. Nodes in largest SCC – 4039 (1.000)
6. Edges in largest SCC – 88234 (1.000)
7. Average Clustering coefficient – 0.6055
8. Number of Triangles – 1612010
9. Fraction of closed triangles – 0.2647
10. Diameter(longest shortest path) – 8
11. 90-percentile effective diameter – 4.7

All the above parameters in the dataset define a particular set of nodes and edges that are to be related and recognized as a single group that has conventional properties of the actual graphs. The

content of this dataset can be missing, raw and contains some noise. So, before we proceed further the dataset is preprocessed, cleaned and processed properly.

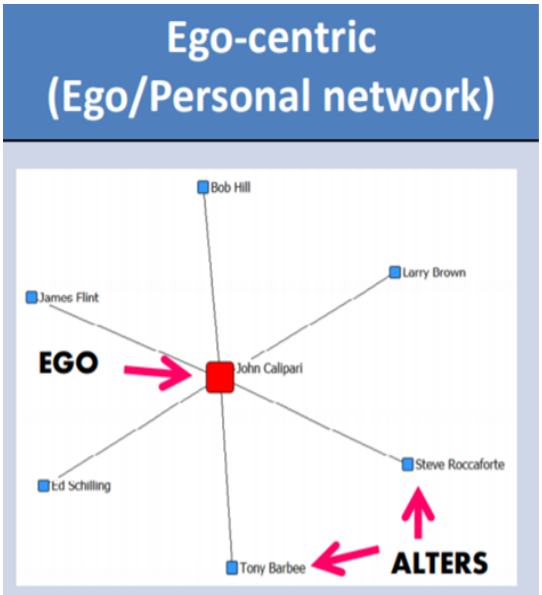



**Fig 2 Ego Centric Network Representation**

**Data Pre-processing:**

1. Inconsistencies are identified
2. Obtain the missing edges between nodes
3. Standard Graph Generation
4. Principal Component Analysis
5. Correlation of Nodes

The data consisted of a modest amount of inconsistencies in the form of unavailable and extreme/unrealistic node values. The unrealistic data is cleaned and merged by using the nearest neighbor method i.e, the endpoints of the gaps were used as estimates for all the missing values between any two nodes. Next, the standardization is done where features were shifted by the mean and scaled by their variance to obtain a standard normal distribution. This method of standardization is called Normalization. Finally PCA is used to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly correlated variables.

## 6. ALGORITHM DESIGN

In order to identify the coterie pairs we use three different algorithms and choose one best method that gives the actual relationship among the nodes in the network of Facebook community pages. The algorithms we are trying to implement –

1. LPA(Label Propagation Algorithm)
2. Fast Unfolding
3. Spectral Clustering

### 6.1 LPA(Label Propagation Algorithm)

The label propagation algorithm is semi-supervised machine learning algorithm which can be used to put the number sequences to the points that are not previously labelled. At the start of the algorithm, a (generally small) subset of the data points have labels (or classifications). LPA is to transfer label information constantly between nodes. The initial conditions are parameterized and the following procedure is implemented as a Feistel structure. The proper implementation of this algorithm is possible only when the network consists of nodes that are sequentially located and are in open configuration for each other thus ensuring the construction of a community. This is only possible for the dense networks containing more nodes.

The cypher projection for the label propagation algorithm is as shown the following code –

```
CALL algo.labelPropagation(
  'MATCH (p:User) RETURN id(p) as id, p.weight as weight, id(p
) as value',
  'MATCH (p1:User)-[f:FRIEND]->(p2:User)
   RETURN id(p1) as source, id(p2) as target, f.weight as weight',
  {graph:'cypher',write:true});
```

Keywords - Node Importance
Label Influence

LPA is to transfer label information constantly between nodes.

**Node Importance:**

Bayesian network property algorithm is used - Probability of importance

**Steps:**

1. Obtain user's basic interests(based in likes, comments, shares,..etc)
2. Calculate Cumulative no.of interests relating to other users
3. The user with most interests is labelled as Important User

Concurrent steps for **Label Influence** –

Calculate the total user influence through multiplying the influence of each attribute factor. The formula is as follows –

Where P(Inf) = Influence of the user

P(Inf | Attr) = Influence of each attribute of the user

Normalize the influence of all the users in the dataset, and the importance of each node is obtained.

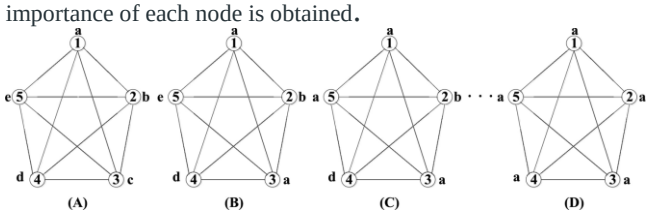



**Fig.3 Identifying Important Nodes and Ordering them**

At initial condition, the nodes carry a label that denotes the community they belong to. Membership in a community changes based on the labels that the neighboring nodes possess. This change is subject to the maximum number of labels within one degree of the nodes. Every node is initialized with a unique label, then the labels diffuse through the network. Consequently, densely connected groups reach a common label quickly.

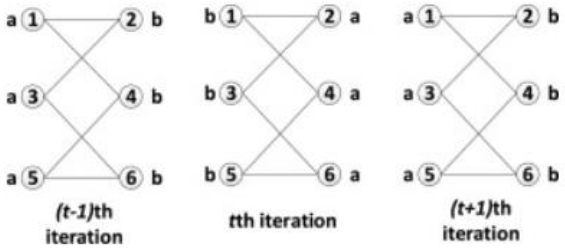



**Fig.4 Synchronous, Asynchronous Updation**

The log-normal distribution is required for this type of updation. Instead of spreading the log-in values, the complex information is taken back from the current algorithm and run on different platforms in order to successfully execute them to a scale. This is the major disadvantage as the system has to run loops continuously in order to sustain the overhead created by constant updation. More than

multiple communities in the network follow this updation pattern. To run the label propagation and call the write back functions in order to generate the LPA, we use the following syntax -

```
CALL algo.labelPropagation(label:String, relationship:String, {ite
rations:1,
   weightProperty:'weight', writeProperty:'partition', write:true, co
ncurrency:4})
YIELD nodes, iterations, didConverge, loadMillis, computeMillis,
writeMillis, write, weightProperty, writePropert
```

**Oscillation:**

The concept of oscillation is derived from the existence of labelling highly connected nodes and replacing the labels with the highest frequent node. In case of synchronous updation, the adjacent t+1 subgraphs alter between two different graphs over the scope of the entire algorithm. Whereas, in case of Asynchronous updation, after the t=0 i.e., first iteration, all the labels are updated to the final output and it iterates over for a period of time to result in the same output as the previous iteration. When many such dense (consensus) groups are created throughout the network, they expand outwards until it's impossible to do so.

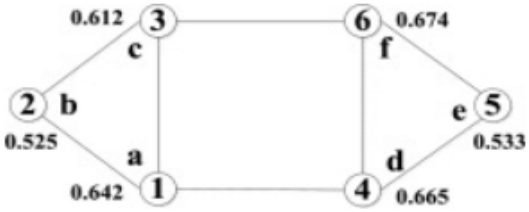



**Fig.5 Oscillation and Uncertainty**

**Uncertainty:**

In the above Figure 4, the possibility of 1(or)2 communities is decided by the adaption of node 1 and 4, i.e., if the node 4 gets labelled by node 1, then single community with all nodes labelled 'a' is obtained. On the flip side, if the above doesn't happen two communities are formed such that {1,2,3} fall into a community with all the labelled nodes named as 'a' and {4,5,6} fall into the community with all the labelled nodes named as 'f' as per our assumptions.

The set nodes are shared with the neighbors and are represented as shown in the above figure. We can make this model more intuitive and responsive by going beyond many other proximity caller tools such as SLPA(Speaker-listener Algorithm) and BMLPA(Balanced Multi-Label Algorithm). The extension factors have been very intuitive so as to give the desired relations.

1. Every node is initialized with a unique label, then the labels diffuse through the network.
2. Consequently, densely connected groups reach a common label quickly.
3. When many such dense (consensus) groups are created throughout the network, they continue to expand outwards until it is impossible to do so.

This algorithm is comprised of five steps –

1. Initialize the labels at all nodes in the network.

   For a given node x, $C_x(0) = x$.

2. Set t = 1.

3. Arrange the nodes in the network in a random order and set it to X.

4. For each $x \in X$ chosen in that specific order, let $C_x(t) = f(C_{xi1}(t), ...,C_{xim}(t),C_{xi(m+1)}(t-1), ...,C_{xik}(t-1))$. f here returns the label occurring with the highest frequency among neighbors. Select a label at random if there are multiple highest frequency labels.

5. If every node has a label that the maximum number of their neighbours have, then stop the algorithm. Else, set t = t + 1 and go to.

The next iteration t+1 is found out by using the following schematic representation-

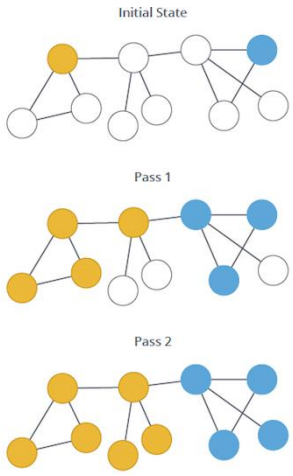



**Fig.6 Step-5 of the algorithm t = t + 1**

The community structure is same as that of the initial condition but the range of the solutions is bound to a sequence of aggregate multiplication structures with all main data stored in the network. In the context of upbringing the advantages of label propagation, the implemented structure of LPA is as follows –

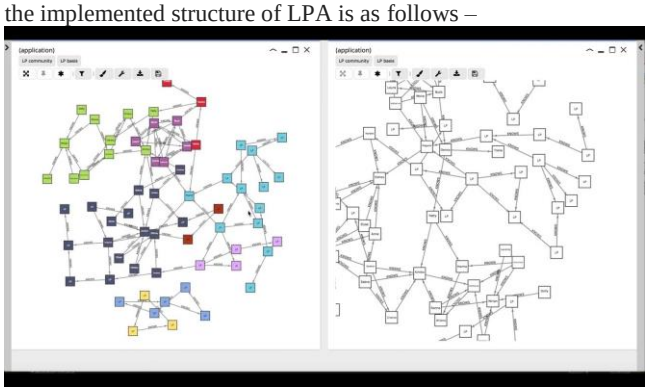


**Fig 7 A render of the Label Propagation algorithm after implementation**

**Time Complexity:**

Suppose n=|V| and m=|E|, and we firstly analyze the time complexity of the algorithm.

(1) The time complexity of initialization for all nodes in step 1: O(n).

(2) The time complexity of calculating the node importance of all nodes in step 2: O(n), and the time complexity of ranking the nodes in descending order of *NI* based on the fast sorting algorithm in step 2: O(nlog2(n)), so the whole time complexity of step 2 is O(nlog2(n)+n).

(3) The time complexity of normal label updating is O(m).

(4) The time complexity of assigning the nodes with the same labels to a community in step 5: O(n).

Step 4 is iterative, and maximum number of iterations is maxIter. The time complexity of the whole algorithm is

$$3 \times O(n) + 2 \times \max Iter \times O(m) + O(n \log_2(n))$$

**Space Complexity:**
(1) The space complexity of using adjacency list to store the node and edge information in step 1: O(m).
(2) The space complexity of ranking the nodes based on the fast sorting algorithm in step 2: O(nlog2(n)).
(3) The space complexity of assigning the nodes with the same labels to a community in step 5: O(n).
So the space complexity of the whole algorithm is 2×O(n)+O(m)+O(nlog2(n)).

## 6.2 Fast Unfolding

The fast unfolding algorithm is based on modularity optimization. The overall performance of the peer detection and relation identification hugely dependent on the computational time. The term of modularity is defined for the distance vectors than can be used for obtaining the prime location identity of ad-hoc modular networks.
Keywords – Modularity
As part of modularity optimization, we use the Louvain algorithm as a part of the context of multiple unfolding schemes.

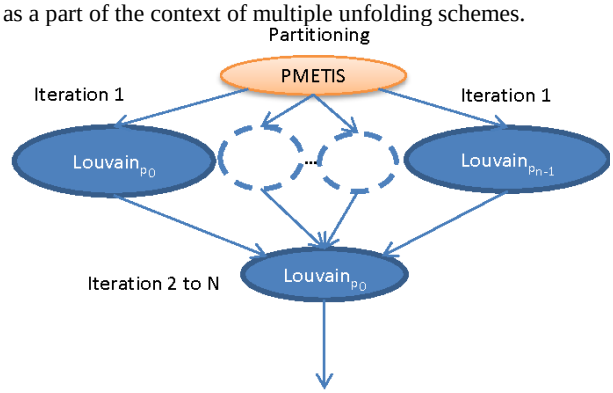



**Fig 8 Louvain Algorithm for Modularity Iteration**

**Modularity:**
It is defined as a scale value between -1 and +1 which can be defined and used to find the density of the link inside the communities when compared to the density between the communities. The formula for the modularity is as follows -

$$Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\delta(c_i, c_j)$$

where $A_{ij}$ = edge weight between the nodes i and j
$K_i$ = sum of weights of edges associated to node i
$K_j$ = sum of weights of edges associated to node j
$C_i$ , $C_j$ = communities of the nodes i and j
δ = Simple Delta Function.
This is highly used for random selection of the nodes rather than forming a community with the same neighboring nodes. Unlike label propagation, there is only one community structure that is existed in the graph.
This algorithm is done in two phases -
a. Identifying Communities
b. Merging these communities into a single community

**Phase - 1:**
1. Assign a different community to each node in a network.
2. Then for each node i consider node j and evaluate gain in modularity by removing node i from its community and placing it in community of j.
3. The node i is placed in the community for which it gains max modularity, but gain should be +ve. If -ve we do nothing.
4. The process is applied continuously until no further improvement can be achieved.
5. This completed the phase 1
A term called ΔQ is called the change in modularity. If the change in modularity is highest in a specific community, then that subgraph is called a community. The change in modularity is calculated by –

$$\Delta Q = \left[\frac{\Sigma_{in} + 2k_{i,in}}{2m} - \left(\frac{\Sigma_{tot} + k_i}{2m}\right)^2\right] - \left[\frac{\Sigma_{in}}{2m} - \left(\frac{\Sigma_{tot}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2\right]$$

where $\Sigma_{in}$ = sum of weights of links inside the community
$\Sigma_{tot}$ = sum of weights of links to nodes in the community i is moving into
$k_i$ = weighted degree of i
$k_{i,in}$ = sum of the weights of the links between i and the other nodes in the community that i is moving into
m = sum of the weights of all links in the network

**Phase - 2:**
1. Here we build nodes by merging all the nodes in community as a single node.
2. Weights of the link between the nodes is given by, sum of the weights of the links between nodes in corresponding two communities.
3. Links network nodes, of same community lead to self-loops for this community in the new network.
4. Repeat Phase 2 until no improvement can be made.

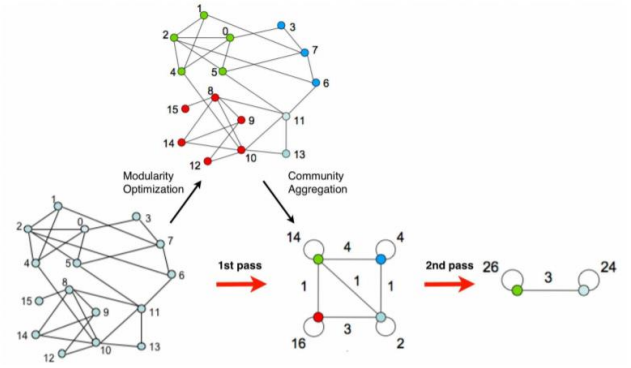



**Fig 9 Modularity Optimization and Community Aggregation**

Let's consider the orange nodes to be part of the community 1 and it's brief classification and counting the nodes is as follows –

For community-1:
0 → 2, 4, 5
1 → 2, 4
2 → 1, 0, 5, 4
4 → 2, 1, 0
5 → 0, 2
Total links within the community = 14

Hence the self-loop on community-1 has a total value of 14. Likewise all the communities can be grouped and aggregated into a single structure which in-turn is called the community of the given graph structure.
This is a heuristic method which is based on the modularity optimization technique and unfolding the logic behind the node representations of neighbors. We should set each node's label values in such a way as to obtain the highest modularity increase.
In replacement to the random forest algorithm, this fast unfolding is used to mark up and represent a modular set of open valued pairs collectively occupying the whole system of values in the network. The latest structural anomaly in the node system network has prompted us to go beyond the actual system nodes and look into more sub-sequential parts of the system. This network structure of unfolding leads back to the sign of leaving the references to the

nodes in the system behind. We evaluate the performance of this method using the time and space complexity.

**Time Complexity:**

(1) For each updation, O(m). Since there is only one iteration, this is the only possible outcome

(2) Modularity gain plays a major role in updating the time complexity from O(m) to O(nlog2(n))

(3) It speeds up the algorithm roughly 2-3 times. This is due to two factors: -

   a. A random neighbor is likely to be in a "good" community; and

   b. Random neighbors are likely to be hubs, helping the convergence.

**Space Complexity:**

Only assigning the nodes to the neighbors takes up the space and hence the space complexity is also O(nlog2(n))

## 6.3 Spectral Clustering

The similarity matrix in the special area of concern for the network will look beyond the exploited efficiency of the clustering and can be further enhanced to provide the solution that will actually improve the eigen count as a matrix tree structure and on the whole leaves the scars of manipulation and tell the user that the nodes are not very compatible to each other. This mathematical dedication is defined by the Lanczos algorithm.

Spectral clustering has become increasingly popular due to its simple implementation and promising performance in many graph-based clustering. It can be solved efficiently by standard linear algebra software, and very often outperforms traditional algorithms such as the k-means algorithm.

To perform a spectral clustering we need 3 main steps:

1. Create a similarity graph between our N objects to cluster.
2. Compute the first k eigenvectors of its Laplacian matrix to define a feature vector for each object.
3. Run k-means on these features to separate objects into k classes.

We consider spectral clustering algorithms for community detection under a general bipartite stochastic block model (SBM). Regularization of an appropriate adjacency or Laplacian matrix. A form of spectral truncation and a k-means type algorithm in the reduced spectral domain. We focus on the adjacency-based spectral clustering and for the first step, propose a new data-driven regularization that can restore the concentration of the adjacency matrix even for the sparse networks.

This result is based on recent work on regularization of random binary matrices, but avoids using unknown population level parameters, and instead estimates the necessary quantities from the data. A novel variation of the spectral truncation step is defined and we show how this variation changes the nature of the misclassification rate in a general SBM.

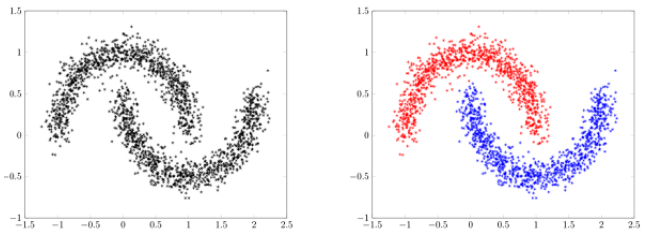


**Fig 10 Spectral Clustering Scenario**

* The **stochastic block model** is a generative model for random graphs. This model tends to produce graphs containing *communities*, subsets characterized by being connected with one another with particular edge densities.

We then show how the consistency results can be extended to models beyond SBMs, such as inhomogeneous random graph models with approximate clusters, including a graphon clustering problem, as well as general sub-Gaussian biclustering.

The Facebook graph structure being among the biggest references and the Laplacian matrix is not well conditioned and slow convergent to iterate a sequence in the clustering methodologies.

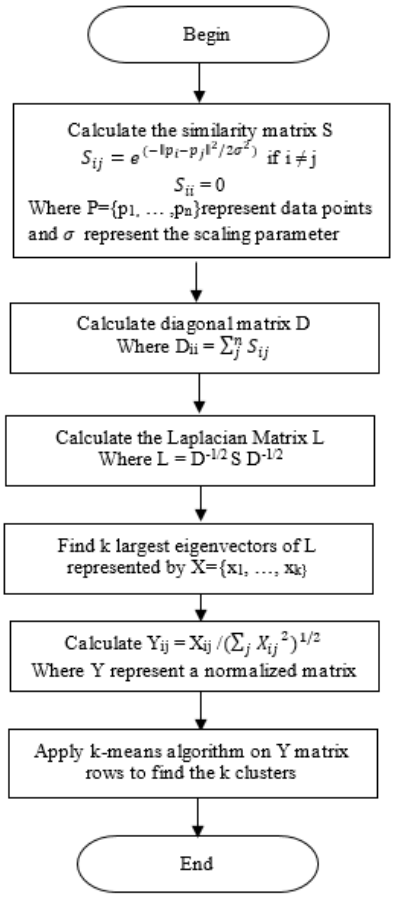



**Fig 11 Algorithm for Spectral Clustering**

Spectral clustering which is used for preconditioning to reduce errors and give up on the outliers. This has a relationship with k-means and it's occurrences are way beyond the exception limit. This equation has got beyond the expectations and got the maximum trace and rank them in the order. The constraints are taken to a graph in the maximum reference vector. The maximum trace is obtained by the following algorithm as a snapshot below.

$$\max_{\{C_s\}} \sum_{r=1}^{\kappa} w_r \sum_{x_i, x_j \in C_r} k(x_i, x_j).$$

Suppose $F$ is a matrix of the normalizing coefficients for each point for each cluster problem with n points and k clusters is given as,

$$\max_F \operatorname{trace}(KF)$$

such that

$$F = G_{n\times k} G^T_{k\times n}$$
$$G^T G = I$$

such that $\operatorname{rank}(G) = k$. In addition, there are identity constrains on $F$ given by,

$$F \cdot \mathbb{I} = \mathbb{I}$$

where $\mathbb{I}$ represents a vector of ones.

$$F^T \mathbb{I} = \mathbb{I}$$

This problem can be recast as,

$$\max_G \operatorname{trace}(G^T G).$$

**General Spectral Clustering Scenario –**

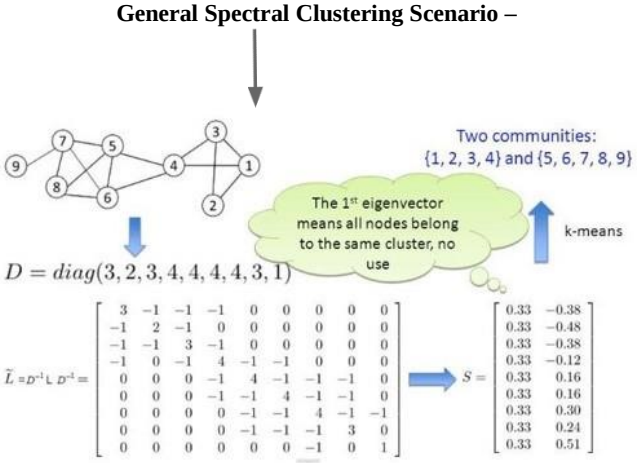



**Fig 12 The entire procedure for Spectral Clustering taking a relatively smaller sub-graph from the community structure of the Facebook Network**

The partition is set into a group of clusters with the help of the eigen vector values of the matrix notation. This algorithm can successfully retrieve the node count which in-turn is responsible for identifying the relationship among the user nodes through network connections in Facebook community pages.

**Time and Space Complexity:**

The time and space complexities are the same in case of spectral clustering. The break-down of this is as follows –

Since, the spectral clustering consists of three phases. Let's compute the time and space complexity for each phase and sum up the values

Step - 1: Adjacency Matrix - O(n)

Step - 2: Along with Truncation - O(n)

Step - 3: WIth K-Means spectral Domain - O(n)

While calculating the total complexities we go with algorithms that are superficially calculated by the sum of the above steps which is O(n) + O(n) + O(n) = 3 * O(n) i.e., it has a linear complexity.

The time and space complexities are the same which is O(n)

## 7. EXPERIMENTAL RESULTS

For initial scraping of the data from the data-site, the median of time for the data over the years is calculated using the method of initiation. The listing of the optimality/median over a scale is as follows –

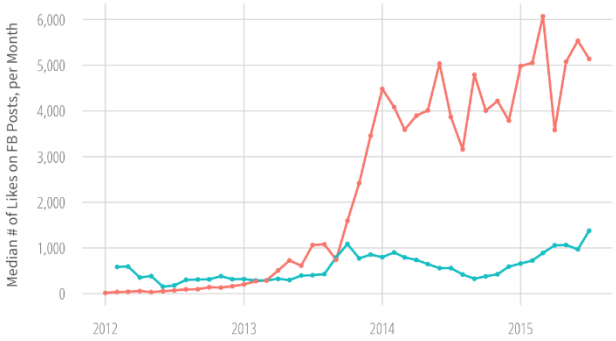



**Fig 13 Listing Medians of activities over the years**

After applying the Principal Component Analysis(PCA), the module graphs are optimized which in turn means the data set has been normalized such that all the values of the nodes and the edges between the nodes are present in a symmetric or asymmetric fashion. All the profile features are obscured and the remaining parts are linked and listed purposefully in the graph constructed.

All the values are optimized and drawn to a scale into the box-plots and the representation obtained is as follows –

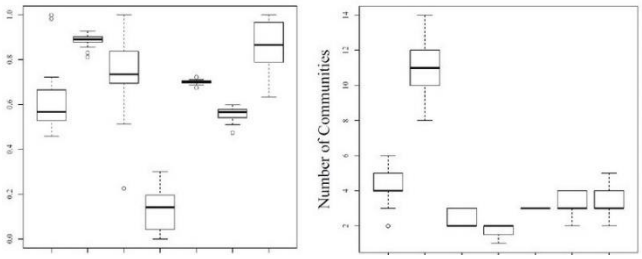



**Fig 14 Box Plots constructed after PCA**

This comparative study helps us to find out which algorithm provides us with the best result in forming the communities and thus we understand the instance in which the relevant algorithm has to be implemented. For each graph construction different parameters are considered for the graphs which result in proper understanding of the efficiency in each case.

The results of each method we used is as mentioned as below –

**a. Label Propagation:**

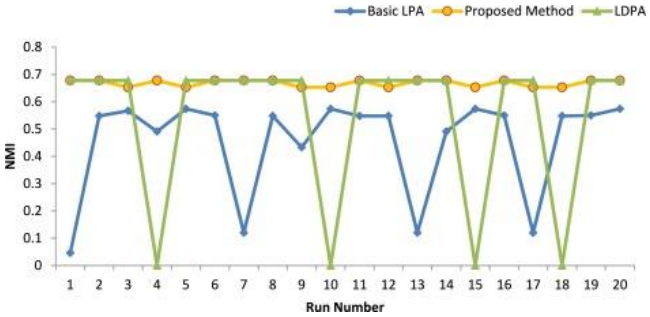



**Fig 15 Run the algorithms of LPA and our proposed LPA multiple times and compare the Non Maskable Interrupt at each stage of the process**

**b. Fast Unfolding:**

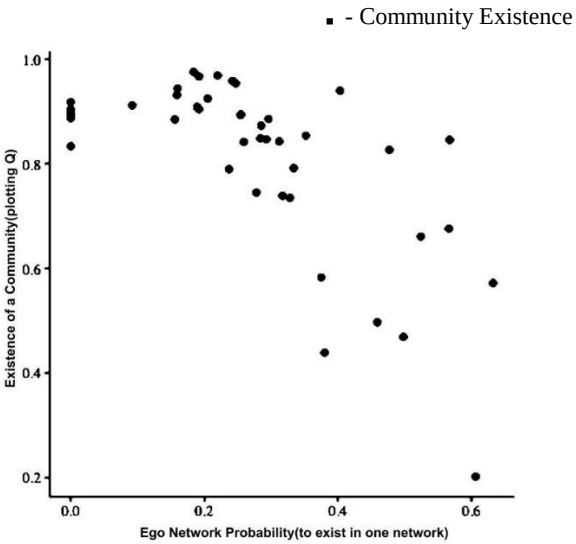



**Fig 16 By fast unfolding we implement the modularity ΔQ and draw it to a scale of Ego Network Probability checking for the existence of communities inside the Facebook network**

**c. Spectral Clustering**

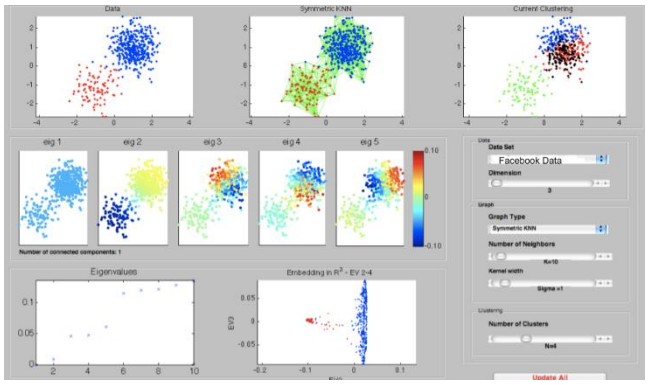



**Fig 17 Running the executable file of the spectral clustering in order to cluster the Facebook Data into communities**

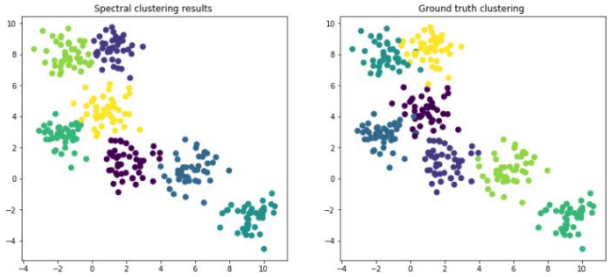
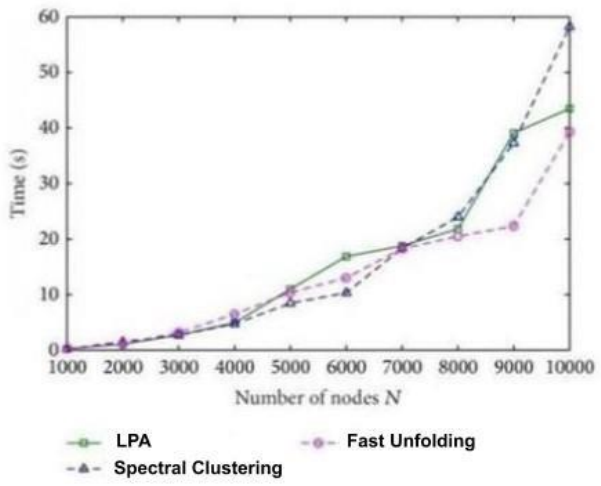



**Fig 18 Ground Truth Clustering is used to verify the Spectral Clustering Results. (In our case it is a valid match) making our procedure for implementation true**

**Overall Comparison Results:**

Since we have used different deployment procedures for all the algorithms, we need a common parameter to compare the performance of these algorithms. We choose that parameter to be the Time complexity of the algorithm.

In terms of Time Complexity –




**X - axis:** No.of Nodes in the Dataset
**Y - axis:** Time taken to successfully identify the groups or peers in the data nodes

**Fig 19 Comparison of the three algorithms implemented**

Since, we have used an incremental procedure(i.e., increased the number of users progressively, taking values for every 1000 nodes). The time linear equations are obtained. Critical points are marked subsequently. We compare the complexities owing to the three algorithms and plot them to a scale by varying the no.of elements in the dataset and hence we can successfully compare which algorithm has the highest efficiency.

**Advantages and Disadvantages:**

| | Label Propagation Algorithm | Fast Unfolding | Spectral Clustering |
|---|---|---|---|
| **Advantages** | 1. No amount of a priori information needed about the network structure (no parameter is required to be known beforehand).<br>2. No parameter is required to be known beforehand.<br>3. The computational costs are low. | 1. It is intuitive and easy to implement, and the outcome is unsupervised.<br>2. Faster compared to LPA. | 1. Does not make strong assumptions on the statistics of the clusters.<br>2. Easy to implement and gives good clustering results.<br>3. Reasonably fast for sparse data sets of several thousand elements. |
| **Disadvantages** | 1. The disadvantage is that it produces no unique solution, but an aggregate of many solutions. | 1. Modularity gain updation is computationally high. | 1. Use of K-Means clustering in the final step implies that the clusters are not always the same. They may vary depending on the choice of initial centroids.<br>2. Computationally expensive for large datasets. |

**Applications of Coterie Identification:**

1. To identify that are with an alike interest
2. The graph immersion and compression
3. The vertices classification vectors
4. Cohesiveness in transformations
5. Objective Analysis of Node structure in Facebook data organization as a graph.

## 8. CONCLUSION

From the graph, we can infer that -

For larger Datasets,

1. The Fast Unfolding algorithm is computationally more reliable because the computational costs are very less.
2. Following this, Label Propagation is not far behind but has more computational time compared to Fast Unfolding.
3. The method with highest complexity is Spectral Clustering. Though it is linear and in the order of O(n), for dense graphs the inverse calculation results in cubical powers sometimes.

Hence on Comparison, we declare that **with respect to the Computational Efficiency, Fast Unfolding** is the most efficient algorithm to detect the peers/communities/coterie in the Facebook Dataset that we have considered.

## 9. FUTURE WORK

Since, we obtained the LPA to be the least efficient method, we try to decrease the time complexity of this algorithm by introducing novel methods like Node Influence Based Label Propagation Algorithm (NIBLPA) and Evidence Label Propagation Algorithm (ELPA).

In order to decrease the complexities we introduce loop-less or single loop algorithm techniques where iterations are updated at the end of the entire algorithm rather than at the end of each step.

We also try to process huge amount of datasets where the processing takes a lot of computation time and we try to overcome this by introducing some of the breakthrough techniques and novel functions in various programming languages. Implementation of Standard Norms for Clustering and Optimization should be done because over a period of time, large sets result in generation of hashes. We should also find a way to tackle this problem.

## 10. References and Citations


[1] Leskovec, Jure, and Julian J. Mcauley. "Learning to discover social circles in ego networks." Advances in neural information processing systems. 2012.

[2] Lancichinetti, Andrea, and Santo Fortunato. "Community detection algorithms: a comparative analysis." Physical review E 80.5 (2009): 056117.

[3] Zhu, Xiaojin, and Zoubin Ghahramani. "Learning from labeled and unlabeled data with label propagation." (2002): 1.

[4] Fortunato, Santo. "Community detection in graphs." Physics reports 486.3 (2010): 75-174.

[5] https://en.wikipedia.org/wiki/Label_Propagation_Algorithm

[6] Blondel, Vincent D., et al. "Fast unfolding of communities in large networks." Journal of statistical mechanics: theory and experiment 2008.10 (2008): P10008.

[7]https://www.slideshare.net/NicolaBarbieri/community-detection-in-networks

[8] https://en.wikipedia.org/wiki/Spectral_clustering

[9] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte and Etienne Lefebvre, “Fast Unfolding of communities in large networks”.

[10] V.A. Traag, “Fast Unfolding of Communities: Speeding up the Louvain algorithm”.

[11] Julian McAuley, Jure Leskovee, “Learning to Discover Social Circles in Ego Networks”.

[12] P. Lazarsfeld and R. Merton. Friendship as a social process: A substantive and methodological analysis. In Freedom and Control in Modern Society. 1954.

[13] Y. Liu, A. Niculescu-Mizil, and W. Gryc. Topic-link LDA: joint models of topic and author community. In ICML, 2009.

[14] D. MacKay. Information Theory, Inference and Learning Algorithms. Cambrdige University Press, 2003.

[15] J. McAuley and J. Leskovec. Discovering social circles in ego networks. arXiv:1210.8182, 2012.

[16] M. McPherson. An ecology of affiliation. American Sociological Review, 1983.

[17] A. Menon and C. Elkan. Link prediction via matrix factorization. In ECML/PKDD, 2011.

[18] A. Mislove, B. Viswanath, K. Gummadi, and P. Druschel. You are who you know: Inferring user profiles in online social networks. In WSDM, 2010.

[19] P. Nasirifard and C. Hayes. Tadvise: A twitter assistant based on twitter lists. In SocInfo, 2011.

[20] M. Newman. Modularity and community structure in networks. PNAS, 2006.

[21] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. Nature, 2005.

[22] C. Rother, V. Kolmogorov, V. Lempitsky, and M. Szummer. Optimizing binary MRFs via extended roof.

[23] A. Streich, M. Frank, D. Basin, and J. Buhmann. Multi-assignment clustering for boolean data. JMLR, 2012.

[24] J. Ugander, B. Karrer, L. Backstrom, and C. Marlow. The anatomy of the Facebook social graph. preprint, 2011.

[25] C. Volinsky and A. Raftery. Bayesian information criterion for censored survival models. Biometrics, 2000.