Communities and Balance in Signed Networks: A Spectral Approach

Pranay Anchuri, Malik Magdon-Ismail
{anchupa, magdon}@cs.rpi.edu
Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY 12180, USA

***Abstract*—Discussion based websites like Epinions.com and Slashdot.com allow users to identify both friends and foes. Such networks are called Signed Social Networks and mining communities of like-minded users from these networks has potential value. We extend existing community detection algorithms that work only on unsigned networks to be applicable to signed networks. In particular, we develop a spectral approach augmented with iterative optimization. We use our algorithms to study both communities and structural balance. Our results indicate that modularity based communities are distinct from structurally balanced communities.**

***Keywords*-signed networks; community detection; structural balance**

I. Introduction

Graphs have invariably been used to model relationships between objects in a wide range of domains ranging from metabolic signaling pathways [1] to the World Wide Web, and citation networks [2]. Detecting communities in social networks has gained a lot of prominence in recent times. In social networks, user communities provide better recommendations and clustering of web pages [3] can be used to provide more relevant search results.

It is almost never possible for people to have a single opinion on any topic. There are always different opinions, and social media like blogs and review websites have provided users a platform to express disagreement publicly or anonymously. Users in such a setting are connected either positively or negatively depending on whether they agree or disagree with another user's opinion. We can still use graphs to model these relationships by adding a sign on the edges. These social networks are called signed social networks. Most community detection algorithms are based on the assumption that the relationship between any pair of objects has the same meaning throughout the network. These algorithms cannot be applied directly to signed networks where the relationships between objects have multiple interpretations.

Modularity, proposed by Newmann [4], is a widely used technique to detect communities in an unsigned network. The idea behind this approach is to divide the network into non overlapping communities in such a way that the number of edges within a community is higher compared to the expected number of edges, had the edges between the vertices been randomly shuffled. The goal of community detection in signed networks is to find sets of nodes in the network that are densely connected with positive edges within the same set and negative edges between the sets.

Our main contributions in this paper are

- We propose an efficient two step spectral approach to optimize modularity and other objective functions in signed networks.
- We demonstrate that the two step approach is effective in detecting communities from real world datasets.
- We use our spectral approach to study communities obtained from modularity maximization with communities obtained from structural balance theory. Our results indicate that these communities are distinct.
- We give evidence for weak balance in Epinions.com and strong balance in Slashdot.com

A. Related Work

Several approaches have been proposed to detect communities in unsigned networks. Most of these methods work well even for weighted edges but not for signed networks. These algorithms fall under the category of graph partitioning and graph clustering. [5] proposed a heuristic to divide nodes of a graph into clusters (communities) such that the number of edges between the clusters is minimized. This method starts with an arbitrary clustering and tries to reduce the objective function by swapping the nodes between the clusters. There have been several random walk based approaches [6], to cluster nodes in a graph. The main idea behind these approaches is that a random walk starting from a node is likely to end up in the same cluster. There are also divisive algorithms like [7] and Girvan-Newman algorithm [8] where edges that are most "between" communities are deleted to find disjoint communities. [9] proposed a method to extract hierarchical community structure from very large networks. It is based on the modularity [4] and is efficient in extracting communities at different scales. A related problem [10], [11] is that of finding overlapping communities in social networks.

A few algorithms have been proposed to detect communities in signed networks. [12] defined a criterion function (frustration) for partitioning signed networks. Given the number of communities required, a locally optimized value of the criterion function is obtained by starting with a random partition of the network into k sets and moving to states which are neighbors of the current state until a local optima is reached. [13] analyzed the properties of signed networks at a global level (signed clustering coefficient), node level (popularity) and edge level (link prediction). [14] proposed a method to detect communities in signed networks by formulating the as a signed version of ratio cut. The objective function is solved by computing the eigen values of the Laplacian matrices. [15] presented a definition of modularity using the Potts model representation. [16] proposed an agent based approach to detect communities in signed networks. In this method, a random walk starting from an arbitrary node and is used to detect the community it belongs to. The number of steps in a given walk is a user defined parameter. Recently, [17] analyzed the structural balance in large signed networks. They concluded that most on-line networks available today exhibit structural balance.

The rest of the paper is organized as follows: Section II introduces the terminology used throughout the paper. In section III we briefly mention structural balance in social networks. In section IV we propose a simple strategy to extend existing graph partitioning techniques to signed networks. In section V we propose two objective functions to obtain two communities and show that they reduce to the same mathematical form. In section VI we generalize the method to detect multiple communities. In section VII we present the experimental results and conclude in section VIII.

II. Notation

Given a signed social network $G = (V, E, W)$, where V is the set of vertices or nodes, $E \subseteq V \times V$ denotes the set of edges, and $W : (V \times V) \longrightarrow \{-1, 0, 1\}$ is a function that assigns a value to the relationship between every pair of nodes. W assigns $+1$ to pairs of nodes that are connected positively, -1 to pairs connected by a negative relationship and 0 to pairs that are not connected. Let the number of vertices in V be n, and assume that the vertices are labeled 0 to $n-1$. A denotes the weighted adjacency matrix corresponding to G i.e $A_{ij} = W(i,j)$. A' denotes the unsigned version of the adjacency matrix i.e $A'_{ij} = |W(i,j)|$. Adjacency matrices corresponding to positive and negative relationships are defined respectively as

$$P_{ij} = \frac{A_{ij} + A'_{ij}}{2}, N_{ij} = \frac{A'_{ij} - A_{ij}}{2}$$

Number of non-zero entries in matrices A, P, N are respectively denoted by $2 \times m, 2 \times m_p, 2 \times m_n$. The positive degree of vertex i, p_i, is the number vertices to which i is connected to positively; similarly n_i is the number of vertices to which i is connected to negatively. The degree of the vertex d_i is $p_i + n_i$.

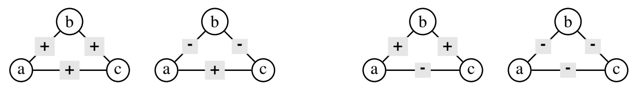
A cluster or community $\mathcal{C}$ is a non empty set of vertices. Our goal is to divide the network into communities $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$ maximizing some objective function. We only deal with non overlapping communities in this paper. $\therefore \mathcal{C}_i \cap \mathcal{C}_j = \emptyset$ for all $i \neq j$.

III. Structural Balance

In a signed network there are four possible configurations in which three nodes are interconnected (triad) as shown in figure 1. We can represent each configuration by the number of positive edges out of the three edges. A 3-configuration represents a scenario where every node connects positively to the other two nodes and 0-configuration is where each node is connected negatively to other two nodes. According to the theory of strong social balance which has its foundations in social psychology, the 1-configuration (a pair of friends who share an enemy) and the 3-configuration (mutual friends) are structurally stable and the other two configurations are unstable.

A general network with more than 3 nodes is structurally balanced if all the 3 CLIQUES are structurally stable. The basic structural theorem [18] proves that nodes in a structurally balanced network can be divided into two disjoint sets such that the positive edges lie within the sets and the negative edges between them. [19] proposed a generalized balance theorem where 0-configurations are also considered to be stable. Under this definition, a network is k-balanced iff the nodes can be partitioned in to k-subsets such that positive edges lie within the sets and the negative edges between them.

Real world networks are rarely structurally balanced. In most cases there are edges that participate in unstable configurations which disturb the stability of the entire network. The number of such edges is defined as the frustration of the network. Given a division of the network into communities, the frustration is the sum of the number of positive edges between nodes in different communities and the number of negative edges between nodes in the same community.



(a) Stable configurations (b) Unstable configurations

Figure 1: Showing structurally stable and unstable configurations

$$F(\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k) = \sum_{ij} N_{ij}\delta(c_i, c_j) + P_{ij}(1 - \delta(c_i, c_j)) \quad (1)$$

where c_v denotes the community to which the node v belongs and $\delta(x, y)$ is a function whose value is 1 if $x = y$ and 0 otherwise. Less frustration implies mutually antagonistic groups with few imbalances. Therefore, minimizing frustration can be used as an objective function to detect communities.

IV. A Clustering Heuristic for Unsigned Networks

A simple heuristic to minimize the frustration when the number of positive edges is significantly higher than the number of negative edges is to remove all the negative edges and cluster the network using any graph clustering algorithm. Isolated nodes are then added back such that the frustration is minimized.

Graph partitioning [20]–[23] is a well studied research problem and people have proposed various objective functions to capture the intuitive notion of clusters in the network. Most of these techniques find clusters that are densely connected internally as compared to external connections. One such objective function is the *Ratio Association*,R, which captures within cluster association relative to the size of the cluster.

$$R(\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k) = \sum_{i=1}^{i=k} \frac{links(\mathcal{C}_i, \mathcal{C}_i)}{|\mathcal{C}_i|} \tag{2}$$

where $links(A, B)$ is the number of edges that have one end in A and the other in B.

[24] proved that maximizing the *Ratio Association* is equivalent to k-means in a higher dimensional space for kernel matrix K equal to the adjacency matrix. As in any k-means based algorithm it is not easy to estimate the value of k beforehand. Such an algorithm only applies to unsigned networks.

V. Dividing into two communities: The Spectral Approach

We propose two new approaches to divide a signed network in to sub communities based on minimizing frustration and maximizing signed modularity. Though the objective function that each approach tries to optimize is different, we show that both the problems can be reduced to a similar mathematical form which can be solved by an efficient iterative algorithm.

Minimizing Frustration

The frustration of the entire network is equal to the number of negative edges because all the positive edges lie in the same community which is the network itself. A network can therefore be divided into two communities only if there is a division that leads to lower overall frustration. To check for the existence of such a division, we first derive an expression for the change in the frustration value when nodes are partitioned into two communities and then try to find a partition for which the change is negative.

The edges that affect the frustration value when a network $\mathcal{C}$ is divided into $\mathcal{C}_1$ and $\mathcal{C}_2$ are

- Positive edges between $\mathcal{C}_1$ and $\mathcal{C}_2$ which increase the frustration.
- Negative edges between $\mathcal{C}_1$ and $\mathcal{C}_2$ which decrease the frustration.

So the overall change in frustration can be written as

$$\begin{aligned} \delta F(\mathbf{s}) &= \sum_{i,j \in \mathcal{C}} \left(P_{ij} \times \frac{1 - s_i s_j}{2} - N_{ij} \times \frac{1 - s_i s_j}{2} \right) \\ &= \frac{1}{2} \sum_{i,j \in \mathcal{C}} \underbrace{A_{ij}}_{\text{constant}} - \mathbf{s}^{\mathsf{T}} \mathbf{A}\, \mathbf{s}, \end{aligned} \tag{3}$$

where $\mathbf{s}$ is called the assignment vector, s_i is $+1$ if the i^{th} node belongs to $\mathcal{C}_1$ and -1 otherwise. The overall frustration decreases if $\delta F(\mathbf{s})$ is negative, and our goal is to find an assignment vector such that the reduction is maximum. The problem is then equivalent to maximizing the term $\mathbf{s}^{\mathsf{T}} \mathbf{A}\, \mathbf{s}$ over all possible boolean vectors $\mathbf{s}$. We postpone the solution to this problem until we derive a very similar expression for optimizing a different objective function in the next section.

Maximizing Modularity

Modularity [25] for unsigned networks is the difference of number of edges within the communities and the expected number if the edges were randomly permuted. Modularity measures how "surprised" you are by the number of edges within the community. As modularity increases there are more edges within communities compared to random chance. A high surprise factor is what one expects from a good community detection algorithm. We now extend the concept of modularity to signed networks.

Though there are various methods for maximizing modularity, our algorithm is similar to the spectral approach proposed by [4]. A detailed explanation of the algorithm for unsigned networks can be found in [4]. Consider a division of an unsigned network $\mathcal{C}$ into two communities $\mathcal{C}_1$, $\mathcal{C}_2$. The modularity, up to a constant factor, can be written as

$$Q_U = \mathbf{s}^{\mathsf{T}} \sum_{\mathbf{i},\mathbf{j}} \overbrace{\left(A_{ij} - \frac{d_i d_j}{2m} \right)}^{B_{ij}} \mathbf{s} \tag{4}$$

where the symbols A, d_i, d_j, m have their usual meanings and $\mathbf{s}$ is a boolean assignment vector whose i^{th} term is 1 if vertex $i \in \mathcal{C}_1$ and -1 otherwise. The term $\frac{d_i \times d_j}{2 \times m}$ is the expected number of edges between nodes i and j. The matrix B = $[B_{ij}]$ is called the modularity matrix.

In the signed case, we need to consider the surprise factors contributed by both the positive edges within communities and the negative edges between communities to compute the modularity. Similar to the unsigned case, consider a division of a signed network $\mathcal{C}$ into two communities $\mathcal{C}_1$, $\mathcal{C}_2$. The signed modularity, up to a constant factor, can be written as

$$\begin{aligned} Q_S = &\sum_{i,j \in C_1} \left(P_{ij} - \frac{d_{p_i} d_{p_j}}{2m_p} \right) + \sum_{i,j \in C_2} \left(P_{ij} - \frac{d_{p_i} d_{p_j}}{2m_p} \right) \\ &+ \sum_{\substack{i \in C_1 \\ j \in C_2}} \left(N_{ij} - \frac{d_{n_i} d_{n_j}}{2m_n} \right) + \sum_{\substack{i \in C_2 \\ j \in C_1}} \left(N_{ij} - \frac{d_{n_i} d_{n_j}}{2m_n} \right) \end{aligned} \tag{5}$$

where the first two terms correspond to positive edges and the other two terms corresponds to negative edges. By using the fact that $A = P - N$, Q_S can be compactly written as

$$Q_S = \mathbf{s}^{\mathsf{T}} \sum_{\mathbf{i},\mathbf{j} \in \mathbf{C}} \overbrace{\left(A_{ij} + \frac{d_{n_i} d_{n_j}}{2m} - \frac{d_{p_i} d_{p_j}}{2m} \right)}^{B'_{ij}} \mathbf{s}. \tag{6}$$

The matrix $B' = [B'_{ij}]$ is the signed modularity matrix. If we define a function $f(M, \mathbf{s}) = \mathbf{s}^{\mathsf{T}} \mathbf{M}\, \mathbf{s}$ then from Equations (3) and (6) it can be seen that maximizing modularity is equivalent to maximizing $f(B', \mathbf{s})$, and minimizing frustration is equivalent to maximizing $f(A, S)$. There is an exact solution to this problem for $\mathbf{s} \in \mathbb{R}^n$ but unfortunately the solution we want is a boolean assignment vector

$\mathrm{s} \in \{-1,+1\}^n$. The s that maximizes $f(M, \mathrm{s})$ is equal to the eigen vector corresponding to the maximum eigen value of M . The top eigen vector can be computed using the *Power Iteration* method. The method starts with an arbitrary vector and iteratively updates it by premultiplying with the modularity matrix. As the modularity matrix can be represented as a sum of adjacency matrix (sparse) and product of degree vectors, the matrix multiplication can be performed very efficiently. One way to obtain $\mathrm{s}' \in \{-1,+1\}^n$ is to round off the values in s based on the sign i.e positive values are rounded to $+1$ and negative values to -1 [4].

A. Iterative Improvement

A better assignment vector s'' may be obtained starting from s and checking each node for the effect it has on the objective function if it jumps from the assigned community to the other community. This is very similar to the Kerninghan Lin update algorithm [5]. We use a variant of this algorithm to further minimize the frustration or maximize the modularity. Before we present the details of the algorithm, we will first deduce an expression for the change in the objective function when a single node moves.

B. Change in Frustration

Without loss of generality assume that node $i \in \mathcal{C}_1$ and let $p_{in}, n_{in}, p_{out}, n_{out}$ be the number of neighbors positively and negatively connected in $\mathcal{C}_1$ and $\mathcal{C}_2$ respectively. The contribution of i towards the overall frustration before and after the movement is $p_{out}+n_{in}$ and $p_{in}+n_{out}$ respectively. Therefore the change in frustration is

$$\delta F(i) = p_{in} + n_{out} - p_{out} - n_{in}. \tag{7}$$

C. Change in Modularity

The change in modularity can be computed very efficiently by considering only the node k and its neighbors. Let Q_1, s and Q_2, s' denote the modularity and assignment vector, before and after the vertex k moved. It is easy to see that $s'_k = -1 \times s_k$ and $s'_i = s_i$ for $i \neq k$. We obtain the following expressions for Q_1 and Q_2,

$$\begin{aligned} Q_1 = &\sum_{i,j}\left(P_{ij} - \frac{d_{p_i}d_{p_j}}{2m_p}\right)\left(\frac{1+s_is_j}{2}\right) \\ &+\sum_{i,j}\left(N_{ij} - \frac{d_{n_i}d_{n_j}}{2m_n}\right)\left(\frac{1-s_is_j}{2}\right) \end{aligned} \tag{8}$$

$$\begin{aligned} Q_2 = &\sum_{i,j}\left(P_{ij} - \frac{d_{p_i}d_{p_j}}{2m_p}\right)\left(\frac{1+s'_is'_j}{2}\right) \\ &+\sum_{i,j}\left(N_{ij} - \frac{d_{n_i}d_{n_j}}{2m_n}\right)\left(\frac{1-s'_is'_j}{2}\right). \end{aligned} \tag{9}$$

Define δq_{ij} as the change in modularity for the pair of nodes i, j. We find that

$$\delta q_{ij} = \begin{cases} 0 & \text{if } i,j \neq k \\ -\sum\limits_{i=k,j}(b_{ij}+c_{ij})\,s_is_j & \text{if } i=k, j\neq k \\ -\sum\limits_{i,j=k}(b_{ij}+c_{ij})\,s_is_j & \text{if } i\neq k, j=k \\ 0 & \text{if } i,j=k \end{cases}$$

$$\begin{aligned} \delta q_{ij} &= -2\sum_{i=k,j}(b_{ij}+c_{ij})s_is_j \\ &= -2\sum_{i=k,j}\left(p_{ij} - c_{ij} + \frac{d_{n_i}d_{n_j}}{2m_n} - \frac{d_{p_i}d_{p_j}}{2m_p}\right)s_is_j \\ &= -2 \times \left\{ s_k\sum_{i=k,j}a_{kj}s_j + s_kd_{n_k}\sum_j\left(\frac{d_{n_j}}{2m_n}\right)s_j \right. \\ &\qquad \left. s_kd_{p_k}\sum_j\left(\frac{d_{p_j}}{m_p}\right)s_j\right\} \end{aligned} \tag{10}$$

These expressions can be used to quickly compute the improvement when vertex k is moved.

The improvement algorithm is summarized in Procedure 3. Initially all the nodes are unmarked. In each iteration, we compute the change in objective by moving each unmarked node independently from its original community to the other community. Any node which leads to a maximum improvement in the objective is marked and is not considered in further iterations. After each iteration, the total change in the objective till that iteration is maintained. At the end, if there is a sequence of updates that result in an assignment closer to the optimal solution then, the optimal such sequence of updates are made permanent.

Though the improvement algorithm generally finds a better assignment of nodes, it is computationally expensive. It is quadratic in the number of nodes present in the community. We can obtain a significant improvement in efficiency by considering only a subset of nodes. This is due to the dual role played by the leading eigen vector. We used signs of the values to assign the nodes to communities, but the magnitude also plays an important role. It measures the extent to which a node belongs to its community. The subset on which we will perform the update is the set of nodes whose magnitudes are close to zero; these marginal nodes could be in either community. We generally don't find an improvement by moving nodes that strongly belong to a community. For a given ϵ, we define ambiguous nodes(U) of a community C as the nodes for which the values in the leading eigen vector corresponding to objective function matrix $\mathbf{M}$ lie in an ϵ interval around 0.

$$U(\epsilon) = \{v : |s_v| \leq \epsilon\}$$

where s_v is the real valued, component of the eigenvalue corresponding to node v.

For example consider the network shown in figure 2. Thick edges are positive and dashed edges are negative.

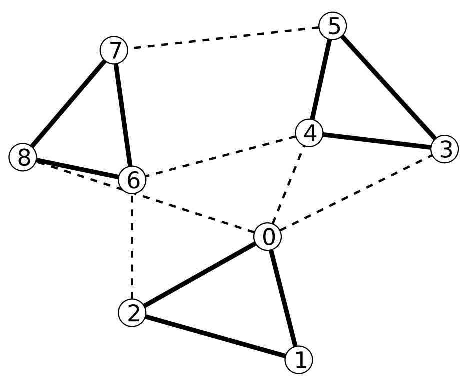


Figure 2: Sample network with CLIQUES of positive edges connected by negative edges.

Node	Leading Eigen Vector
0	0.55364898
1	0.27318962
2	0.25582182
3	-0.4405566
4	-0.45792439
5	-0.33426882
6	0.10928538
7	0.13242688
8	**-0.09162287**

Table I: Leading eigen vector of modularity matrix for network in figure 2

Table I shows the leading eigen vector corresponding to the generalized modularity of this network. Assigning nodes based on the sign separates node 8 (an ambiguous node if $\epsilon \geq 0.1$) from nodes $6, 7$ to which it is positively connected. As the further steps (see section VI) only subdivide the communities, node 8 is never placed in the same community as $6, 7$. The same effect is observed in case of frustration minimization. If $\epsilon = 0.1$ then the only node that is ambiguous is 8 and the change in the modularity when it moves from the "-" community to "+" community is 1.88. Hence, the improvement algorithm would move node 8 to the "+" community.

VI. More than two communities

There is no reason to restrict the community detection algorithm to find only two communities. In fact a community detection algorithm should automatically find the optimal number of communities without expecting any input parameter. An easy way to find multiple communities in the network is to repeatedly divide the communities until the objective cannot be improved any further. For minimizing frustration it means that there is no assignment that gives a lower frustration and for maximizing modularity it means that there is no assignment that increases the modularity. This is easy to check for frustration because the overall change in the frustration of the network by dividing a community is same as change in frustration considering only the sub-community. However this is not true when the objective function is maximizing modularity.

We now derive a general expression for the change in the modularity when a community is split into sub-communities. Consider the case where an existing community $\mathcal{C}_1$ is divided into two sub-communities $\mathcal{C}_{1'}$ and $\mathcal{C}_{1''}$. Equations (5) and (11) denote the modularity before (Q_S) and after (Q_{S^*}) the split. Canceling common terms and using $\mathcal{C}_{1'} \cup \mathcal{C}_{1''} = \mathcal{C}_1$, the change in modularity is as shown in Equation (12). Community $\mathcal{C}_1$ can be subdivided only if there is an assignment of nodes such the expression in (12) is positive. As before this can be checked by computing the leading eigen pair of the generalized signed modularity matrix.

$$\begin{aligned} Q_{S^*} = &\sum_{i,j \in C_{1'} \cup C_{1''}} \left(P_{ij} - \frac{d_{p_i} d_{p_j}}{2m_p}\right) + \sum_{i,j \in C_2} \overbrace{\left(P_{ij} - \frac{d_{p_i} d_{p_j}}{2m_p}\right)}^{b_{ij}} \\ &+ \sum_{\substack{i \in C_1 \\ j \in C_2}} \left(N_{ij} - \frac{d_{n_i} d_{n_j}}{2m_n}\right) + \sum_{\substack{i \in C_{1'} \\ j \in C_{1''}}} \left(N_{ij} - \frac{d_{n_i} d_{n_j}}{2m_n}\right) \\ &+ \sum_{\substack{i \in C_{1'} \\ j \in C_{1''}}} \left(N_{ij} - \frac{d_{n_i} d_{n_j}}{2m_n}\right) + \sum_{\substack{i \in C_2 \\ j \in C_1}} \underbrace{\left(N_{ij} - \frac{d_{n_i} d_{n_j}}{2m_n}\right)}_{c_{ij}} \end{aligned} \tag{11}$$

$$\begin{aligned} \delta Q \quad = \quad & \sum_{i,j \in C_1} \left(b_{ij} - \delta_{ij} \sum_{k \in C_1} b_{ik}\right) s_i s_j \\ & + \sum_{i,j \in C_1} \left(c_{ij} - \delta_{ij} \sum_{k \in C_1} c_{ik}\right) s_i s_j \\ = \quad & \mathbf{s^T} \sum_{\mathbf{i,j \in C}} \underbrace{\left(b_{ij} + c_{ij} - \delta_{ij} \sum_{k \in C_1} b_{ik} + c_{ik}\right)}_{G_{ij}} \mathbf{s} \end{aligned} \tag{12}$$

The matrix $G = [G_{ij}]$ is denoted the generalized signed modularity matrix. To summarize, the overall structure of the algorithm is presented in Procedure 1. Initially all the nodes belong to the same community and in each step we use Procedure 2 to divide an existing community into two nontrivial sub communities that improves the value of the objective function. The division algorithm computes the leading eigen vector and divides the community based on the assignment vector returned by the improvement algorithm.

VII. Experimental Evaluation

In this section we evaluate the effectiveness of the proposed algorithms in detecting communities from signed social networks. The datasets that we consider here are those of two popular websites Epinions.com and Slashdot.com.

Procedure 1 Clustering Signed Network

```
Input: Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$
Output: Communities $\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_k$ such that $\bigcup \mathcal{C}_i = V$
  $Communities \leftarrow Queue(V)$
  $Indivisible \leftarrow []$
  while Communities is not empty do
    $C \leftarrow Communities.dequeue()$
    $C_1, C_2 \leftarrow DivideCommunity(C)$
    if $C_1 = C$ or $C_2 = C$ then
      Indivisible.insert(C)
    else
      Communities.enqueue($C_1$)
      Communities.enqueue($C_2$)
    end if
  end while
  return Indivisible
```

Procedure 2 DivideCommunity

```
Input: A community $C$
Output: Optimal division $C_1$,$C_2$
  Compute the top eigen value $\lambda$ and vector $\mathbf{V}$ of $C$'s objective function
  if $\lambda \geq 0$ then
    Compute the vector $s' = (s'_1, s'_2, \ldots, s'_n)$ , $s'_i = 1$ if $V_i \geq 0$ and $-1$ otherwise
    $av \leftarrow U(\epsilon)$
    $s'' = Improve(\mathbf{s}, av)$
    $C_1 \leftarrow \{i | s''_i = 1\}$,$C_2 \leftarrow \{i | s''_i = -1\}$
  else
    $C_1 \leftarrow C$,$C_2 \leftarrow \emptyset$
  end if
  return $C_1, C_2$
```

Datasets

Epinions.com (E) is a consumer review website where users can post reviews about various items, which can help other users in their decision about buying that item. Users can rate reviews by other users on a scale ranging from "Very Helpful" to "Off Topic". Users are connected positively or negatively depending on whether they trust or distrust another user's review.

Slashdot.com (S) is a website where the administrators share technology related stories and it becomes a topic of discussion among the site's users. Users can tag other users as friends (positive edge) or foes (negative edge). The datasets were obtained from Stanford Large Network Dataset (http://snap.stanford.edu/data/index.html).

Table II shows the number of vertices and edges present in Epinions.com and Slashdot.com.

A. Structural Balance

Table III shows the number of times three nodes are inter connected (triad) in a given type of configuration in Epinions.com and Slashdot.com.

Procedure 3 Improve

```
Input: Assignment vector $s'$ and $av$
Output: Assignment vector $s''$ that improves the objective function.
  $unmoved \leftarrow av$
  $totalImprovement \leftarrow 0$
  $s'' \leftarrow s'$
  for $i = 1 \rightarrow |av|$ do
    for $j \in unmoved$ do
      $score[j] \leftarrow ComputeChange(j, s')$
    end for
    $b \leftarrow BestNode\{score\}$
    $s'_b \leftarrow -1 * s'_b$
    $totalImprovement += score[b]$
    $improve[i] = totalImprovement$
    $unmoved \leftarrow unmoved \setminus \{j'\}$
    $BestIndices[i] \leftarrow b$
  end for
  for $ch \in BestSequence$ do
    $s''[ch] \leftarrow -1 \times s''[ch]$
  end for
  return $s''$
```

Table II: Dataset Properties

Dataset	# Vertices	# Edges
Epinions.com	131828	379603
Slashdot.com	81871	214996

Table III: Number of Triads

Configuration Type	# Epinions.com	# Slashdot.com
+ + + (3)	4003085	414956
+ + - (2)	451711	66679
+ - - (1)	396548	76859
- - - (0)	58732	12075

B. Community Detection

Tables IV and V summarize the communities detected by various algorithms including the algorithms proposed in the paper. Column 2 in the table is the number of communities the algorithm detects. Column 3 is the size of largest community found and column 4 is the overall frustration value (reported as % of the number of negative edges.) of the communities.

We have used the kernel k-means algorithm as a baseline approach to evaluate the effectiveness of algorithms based on minimizing frustration and maximizing modularity. Though the value of k can be as low as 1, we choose a value which is comparable to the number of communities returned by other algorithms. It is important to note that the frustration value for $k = 1$ is equal to the number of negative edges which could be lower than the frustration obtained using detected communities, but minimizing frustration alone may not give insight into community structure. In these experiments we arbitrarily chose a value of 0.005 for ϵ to define ambiguous nodes of a community after the spectral step. A higher value of ϵ leads to larger set of nodes on which improvement step has to be performed.

Frustration as a measure : There is no universal definition of a community which makes it difficult to define a measure to compare algorithms that optimize different objective functions without any bias. We use frustration to compare the baseline and modularity based approaches as it is not biased towards either algorithm. It is also used later for a different purpose.

Modularity Maximization : Table IV shows the properties of the communities discovered by the graph clustering heuristics. We can see that maximizing modularity performs better compared to the baseline approach and with additional improvement it outperforms other algorithms in terms of the frustration. The improvement algorithm surprisingly found only 4 communities in Epinions.com, with largest community covering nearly 88% of the entire network. Without the improvement, this community is divided into smaller communities and resulted in a much higher frustration. A similar effect is observed in Slashdot.com, the largest community covers nearly 73% of the entire network. This shows the effectiveness of the improvement step.

Frustration Minimization : Table V shows the results for frustration minimization using different variations of the spectral algorithm. We can see that frustration is lower if we allow the algorithm to find more than two communities. This effect can be explained as follows : Consider a 0 configuration triad. The minimum frustration that can be obtained by restricting two communities is 1 whereas by allowing 3 communities we can obtain a frustration of zero. Existence of more than two communities with lower frustration is an evidence for the presence of weak balance in the network.

Strong Balance vs Weak Balance : From table V it is evident that in Slashdot.com, the frustration obtained by detecting 2 communities is nearly the same as the frustration obtained by detecting multiple communities (10). This suggests the existence of strong balance. On the other hand, the frustration in Epinions.com by dividing into two communities is significantly higher compared to the frustration obtained by detecting multiple communities (23). This suggests the existence of weak balance as opposed to strong balance in Epinions.com.

Discussion : The preference for balanced configurations in both the networks shows the existence of structural balance. However, these networks exhibit different degrees of balance (strong balance in Slashdot.com and weak balance in Epinions.com) which can be explained by the distribution of negative edges in the network. The negative incident ratio of a node k, $nir(k)$, can be defined as the ratio of the number of negative edges to the number of positive edges incident on k.

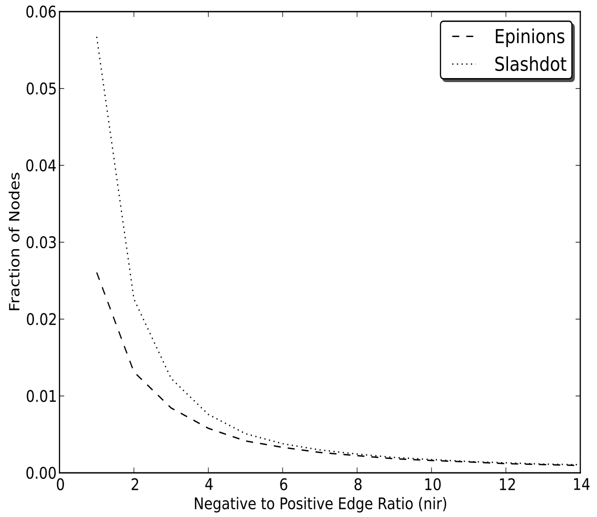


Figure 3: nir vs fraction of nodes in Epinions.com and Slashdot.com

It was shown that nodes with higher nir values contribute less to the overall frustration of the network [17]. In the extreme case where all the incident edges are negative, the node's contribution to the frustration can be made zero by placing it in a community of its own. Figure 3 shows, for a given r, the fraction of nodes with $nir(k) \geq r$. It can be seen that the fraction of nodes with a given nir is higher in $Slashdot.com$ when compared to the corresponding fraction in $Epinions.com$. This can be attributed to the trolling phenomena observed in $Slashdot.com$. It is one of the websites that is seriously affected by trollers. A troll user often posts offensive, inflammatory or off topic comments to provoke serious users of the website. These users are usually tagged as foes by serious users who form the majority of $Slashdot.com$ user community and as friends by other trollers. Hence, the troll users have a higher nir value and contribute less to the overall frustration of the network.

VIII. Conclusion

We have proposed an efficient two step algorithm to detect communities in signed social networks. It is an iterative approach that tries to maximize objective function in each step. The objective functions that we explored are minimizing frustration and maximizing modularity. Further, we used the detected communities to explain the structural balance in two real world social networks. In terms of future work, we would like to explore other objective functions and also extend our algorithm to detect communities in labeled graphs where the nodes and edges can have multiple attributes.

References

[1] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabasi, "The large-scale organization of metabolic networks," 2000.

Algorithm	# Communities	Largest	Frustration
	Epinions.com		
15-means	15	69802	175.07
40-means	40	59022	195.06
Modularity	15	56754	211.62
2 Step Modularity	**4**	**117072**	**100**
	Slashdot.com		
15-means	15	24460	259.93
40-means	40	21666	288.69
Modularity	14	40378	176.85
2 Step Modularity	**3**	**60031**	**141.72**

Table IV: Frustration for communities discovered by various community detection algorithms. k-means is the kernel k-means heuristic with k centers, modularity is the spectral algorithm and 2 step Modularity is the spectral algorithm with Kerninghan-Lin improvement. Observe that spectral with Kerninghan-Lin improvement produces the best communities as measured by Frustration. Frustration is reported as % of negative edges.

Algorithm	# Communities	Largest	Frustration
	Epinions.com		
No Improvement	20	69004	47.65
Improvement	**23**	**68990**	**43.92**
2 Division	2	74100	58.97
2 Div Improvement	2	73861	54.99
	Slashdot.com		
No Improvement	8	55479	62.52
Improvement	**10**	**57853**	**60.67**
2 Division	2	57824	66.34
2 Div Improvement	2	57853	64.71

Table V: (No) Improvement is spectral frustration minimization (without) improvement. 2 Division and 2 Div Improvement is the spectral frustration detecting 2 communities with and without improvement step. Observe that 2-communities are enough for Slashdot.com to get low frustration suggesting strong balance but more communities gives significant improvement in Epinions.com suggesting weak balance. Frustration is reported as % of the number of negative edges.

[2] J. Leskovec, J. Kleinberg, and C. Faloutsos, "Graphs over time: Densification laws, shrinking diameters and possible explanations," in In KDD. ACM Press, 2005, pp. 177–187.

[3] P. Ferragina and A. Gulli, "A personalized search engine based on web-snippet hierarchical clustering," in Special interest tracks and posters of the 14th international conference on World Wide Web, ser. WWW '05, 2005, pp. 801–810.

[4] M. E. J. Newman, "Modularity and community structure in networks," Proceedings of the National Academy of Sciences of the United States of America, vol. 103, no. 23, pp. 8577–8582, 2006.

[5] B. W. Kernighan and S. Lin, "An efficient heuristic procedure for partitioning graphs," Bell System Technical Journal, vol. 49, no. 2, pp. 291–307, 1970.

[6] S. Dongen, "A cluster algorithm for graphs," Amsterdam, The Netherlands, The Netherlands, Tech. Rep., 2000.

[7] J. R. Tyler, D. M. Wilkinson, and B. A. Huberman, Email as spectroscopy: automated discovery of community structure within organizations. Kluwer, B.V., 2003, pp. 81–96.

[8] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," Proceedings of the National Academy of Sciences, vol. 99, no. 12, pp. 7821–7826, 2002.

[9] V. D. Blondel, J. loup Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," 2008.

[10] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society." Nature, vol. 435, no. 7043, pp. 814–818, 2005.

[11] J. Baumes, M. Goldberg, and M. Magdon-ismail, "Efficient identification of overlapping communities," in In IEEE International Conference on Intelligence and Security Informatics (ISI, 2005, pp. 27–36.

[12] P. Doreian and A. Mrvar, "A partitioning approach to structural balance," Social Networks, vol. 18, no. 2, pp. 149–168, 1996.

[13] J. Kunegis, A. Lommatzsch, and C. Bauckhage, "The slashdot zoo: mining a social network with negative edges," in Proceedings of the 18th international conference on World wide web, ser. WWW '09, New York, NY, USA, 2009, pp. 741–750.

[14] J. Kunegis, S. Schmidt, A. Lommatzsch, J. Lerner, E. W. De, and L. S. Albayrak, "Spectral analysis of signed graphs for clustering, prediction and visualization."

[15] V. A. Traag and J. Bruggeman, "Community detection in networks with positive and negative links," Physical Review E (Statistical, Nonlinear, and Soft Matter Physics), vol. 80, no. 3, pp. 036 115+, 2009.

[16] B. Yang, W. K. Cheung, and J. Liu, "Community mining from signed social networks." IEEE Trans. Knowl. Data Eng., vol. 19, no. 10, pp. 1333–1348, 2007.

[17] G. Facchetti, G. Iacono, and C. Altafini, "Computing global structural balance in large-scale signed social networks," PNAS, vol. 108, no. 52 20953-20958, 2011.

[18] F. Harary, "On the notion of balance of a signed graph," The Michigan Mathematical Journal, vol. 2, no. 4, pp. 143–146, 1953.

[19] J. A. Davis, "Clustering and Structural Balance in Graphs," Human Relations, vol. 20, pp. 181–187, 1967.

[20] J. Shi and J. Malik, "Normalized cuts and image segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, pp. 888–905, 1997.

[21] I. S. Dhillon, Y. Guan, and B. Kulis, "Weighted graph cuts without eigenvectors: A multilevel approach," IEEE Trans. Pattern Anal. Mach. Intell, vol. 29, p. 2007, 2007.

[22] I. Dhillon, "A fast kernel-based multilevel algorithm for graph clustering," in In Proceedings of the 11th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2005, pp. 629–634.

[23] P. K. Chan, M. D. F. Schlag, and J. Y. Zien, "Spectral k-way ratio-cut partitioning and clustering," in Proceedings of the 30th international Design Automation Conference, ser. DAC '93, 1993, pp. 749–754.

[24] I. Dhillon, Y. Guan, and B. Kulis, "A unified view of kernel k-means, spectral clustering and graph cuts," Tech. Rep., 2004.

[25] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," Phys. Rev. E, vol. 69, p. 026113, 2004.