Physica A 482 (2017) 147–157

Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Eb&D: A new clustering approach for signed social networks based on both edge-betweenness centrality and density of subgraphs

Xingqin Qi [a,*], Huimin Song [a], Jianliang Wu [b], Edgar Fuller [c], Rong Luo [c], Cun-Quan Zhang [c]

[a] School of Mathematics and Statistics, Shandong University (Weihai), Weihai, 264209, China
[b] School of Mathematics, Shandong University, Jinan, 250100, China
[c] Department of Mathematics, West Virginia University, Morgantown, WV, 26506, USA

HIGHLIGHTS

- A clustering method is presented for signed networks.
- This method is a graph theory based approach.
- It does not need the number of clusters in advance.
- It performs better than other existing method with applying on several data sets.

ARTICLE INFO

Article history:
Received 5 July 2016
Received in revised form 28 February 2017
Available online 24 April 2017

Keywords:
Clustering
Signed network
Edge-betweenness
Density

ABSTRACT

Clustering algorithms for unsigned social networks which have only positive edges have been studied intensively. However, when a network has like/dislike, love/hate, respect/disrespect, or trust/distrust relationships, unsigned social networks with only positive edges are inadequate. Thus we model such kind of networks as signed networks which can have both negative and positive edges. Detecting the cluster structures of signed networks is much harder than for unsigned networks, because it not only requires that positive edges within clusters are as many as possible, but also requires that negative edges between clusters are as many as possible. Currently, we have few clustering algorithms for signed networks, and most of them requires the number of final clusters as an input while it is actually hard to predict beforehand. In this paper, we will propose a novel clustering algorithm called Eb&D for signed networks, where both the betweenness of edges and the density of subgraphs are used to detect cluster structures. A hierarchically nested system will be constructed to illustrate the inclusion relationships of clusters. To show the validity and efficiency of Eb&D, we test it on several classical social networks and also hundreds of synthetic data sets, and all obtain better results compared with other methods. The biggest advantage of Eb&D compared with other methods is that the number of clusters do not need to be known prior.

* Corresponding author.
E-mail address: qixingqin@163.com (X. Qi).

http://dx.doi.org/10.1016/j.physa.2017.04.037
0378-4371/

1. Introduction

Social network analysis has gained great attention in recent years. Commonly, vertices in a network represent people or individuals while the edges show relationships between them. In the usual setting, only positive relationships which denote proximity or similarity are considered. However, when a network has like/dislike, love/hate, respect/disrespect, or trust/distrust relationships, such a representation with only positive edges is inadequate as it fails to encode the sign of a relationship. This kind of networks can be modeled as signed networks, where edge weights can be either positive or negative, representing positive or negative relationships, respectively. Early uses of signed networks can be found in anthropology, where negative edges have been used to denote antagonistic relationships between tribes [1].

The most dominant theory related to signed networks is "social balance", whose intuition can be interpreted as "a friend of my friend is my friend", "an enemy of my friend is my enemy", and "an enemy of my enemy is my friend". A graph is *strongly balanced* if no part of the graph violates this intuition.

Cartwright and Harary [2,3] showed that a network which follows the strong balance notion can be clustered into **two** perfect antagonistic groups such that the edges within groups are positive and the edges between groups are negative. However, in real world not every network has the structure of *strongly balanced*, they may have more than two antagonistic groups. Davis [4] proposed the notion of "weak balance" as a generalization of social balance. They relax the balanced relationships by allowing "an enemy of one's enemy can still be an enemy".

Theorem 1 ([4]). *A network is "weakly balanced" if and only if either* (i) *all of its edges are positive, or* (ii) *the nodes can be clustered into k groups such that the edges within groups are all positive and the edges between groups are all negative. This kind of network is also called* ***perfect*** *k-weakly balanced (where* $k > 2$*).*

But real signed networks are seldom perfect k-weakly balanced, thus for a given signed network, we want to partition it into small groups such that (1) positive edges within clusters are as many as possible and (2) negative edges between clusters are as many as possible. This problem is called *clustering for signed networks*.

Clustering algorithms for unsigned networks with only positive edges have been studied intensively. The most widely used algorithms are spectral clustering method [5], the KL (Kernighan–Lin) algorithm [6], GN (Girvan–Newman) algorithm [7], Newman algorithm [8], and HITS (hyperlink induced topic search) algorithm [9] etc. Clustering for unsigned networks also can be seen as a process to detect dense subgraphs in G. In [10], an approach named Quasi-clique Merger Algorithm was proposed to detect all dense subgraphs in G with various levels and construct a hierarchically nested system to illustrate their inclusion relationships. Yang et al. [11] gave a comprehensive survey of the clustering algorithm for unsigned networks, where clustering algorithms are partitioned into two classes: optimization based methods and heuristic methods. Particularly, Malliaros and Vazirgiannis [12] gave a survey for clustering directed networks.

But clustering algorithms for signed networks cannot be successfully carried out by merely extending the theory and algorithms for unsigned networks in a straightforward manner. Many notions and algorithms for unsigned networks break down when edge weights are allowed to be negative. There are fewer algorithms for signed networks than for unsigned networks. Doreian et al. [13] proposed a local search strategy which is similar to the Kernighan–Lin algorithm. Nikihil et al. [14] also considered the clustering problem for signed graphs, though their formulation is motivated by correlation clustering. Yang et al. [15] proposed an agent-based method which essentially conducts a random walk on the graph. Anchuri et al. [16] proposed hierarchical iterative methods that solve 2-way frustration and signed modularity objectives using spectral relaxation at each hierarchy. Another framework for clustering signed graphs was proposed by Chiang et al. [17] who introduced a new criterion called balance normalized cut, and they developed a multilevel clustering framework "S-Graclus" for signed networks, which "*can be viewed as a generalization of Graclus* [18] *for the signed network setting*"(see Ref. [17]).

However, most of these existing algorithms (especially spectral clustering methods) required the number of final clusters k as an input, that is, k should be known in advance before clustering process. But in practice we cannot initially predict the number of communities beforehand. As said in SAS/STAT 9.2 User's Guide "*There are no completely satisfactory algorithms that can be used for determining the number of population clusters for many type of cluster analysis*". Thus, clustering methods for signed network which do not need the number of clusters in advance are in urgent need.

In this paper, we will propose a novel clustering algorithm for signed networks which avoids using k as an input. Motivated by the GN (Girvan–Newman) algorithm [7] which constructs clusters by progressively removing edges which are least central (i.e., the edges which are most "between" clusters), we will look for the "least central" edges as well in a signed network, but we will also take the density of subgraphs into consideration. A hierarchically nested system are constructed to illustrate the inclusion relationships of clusters. This paper is organized as follows. Some terminologies and notations are introduced in Section 2; In Section 3, a new clustering algorithm for signed networks named Eb&D is presented; Its validity is shown by applying it on some classical real data sets and hundreds of synthetic data sets in Section 4; Conclusions are made in Section 5.

2. Terminologies and notations

A signed network $G = (V, E)$ can be represented as an adjacency matrix that describes relationships between entities. Formally, let $G = (V, E)$ be a graph with weight $w(e) \in \{-1, 0, 1\}$ on each edge, the adjacency matrix A is defined as

follows:

$$A_{i,j} = \begin{cases} 1, & \text{if the relationship of } (i,j) \text{ is positive;} \\ -1, & \text{if the relationship of } (i,j) \text{ is negative;} \\ 0, & \text{if the relationship of } (i,j) \text{ is unknown.} \end{cases}$$

For traditional unsigned (or positive) networks, the *edge-betweenness* of an edge e is defined as the number of shortest paths between pairs of vertices that run along it. That is, let σ_{v_i,v_j} denote the number of shortest paths between nodes v_i and v_j, and let $\sigma_{v_i,v_j}(e)$ denote the number of shortest paths between v_i and v_j which go through e, *the edge betweenness* of an edge e, denoted by $EB(e)$, is defined as follows:

$$EB(e) = \sum_{v_i,v_j \in V} \frac{\sigma_{v_i,v_j}(e)}{\sigma_{v_i,v_j}}. \tag{1}$$

The*density* of a subgraph C is defined as [10],

$$d(C) = \frac{2 \sum_{e \in E(C)} w(e)}{|V(C)|(|V(C)| - 1)}. \tag{2}$$

From Eq. (2), if every pair of vertices in C has an edge e with $w(e) = 1$, then $d(C) = 1$, which means the subgraph C induces a special structure which is called a *clique* in graph theory. Eq. (2) can be generalized to a signed subgraph C where $w(e)$ can be negative. When a negative edge adds to a subgraph C, the density of the new subgraph will drop. If $d(C) \geq \Delta$ for some positive real number Δ, the subgraph C is called a *Δ-quasi-clique*.

3. Eb & D clustering algorithm for general signed networks

Let G be a general signed graph. In this section, we will design an algorithm to cluster G, such that (1) positive edges within clusters are as many as possible and (2) negative edges between clusters are as many as possible. To quantify these two conditions, we define two pre-defined parameters α, β, such that

(i) The density of each cluster C is not smaller than α, i.e. $d(C) \geq \alpha$
(ii) A cluster cannot contain too much negative edges, that is, the drop ratio of a cluster's density when considering only positive edges and when adding the negative edges should not be greater than β, i.e., $\frac{d^+(C)-d(C)}{d^+(C)} \leq \beta$,[1] where $d^+(C)$ is the density of cluster C when only considering positive edges.

3.1. The motivation of Eb&D clustering algorithm

Denote the subgraph consisting of all positive edges of G by G^+. Clearly, the connected components of G^+ is an initial clustering result for G. Similar to the GN (Girvan–Newman) algorithm [7], instead of trying to construct a measurement to tell which edges are most central to clusters, we focus on those edges which are least central and those edges which are most "between" clusters. But GN algorithm cannot be extended to signed networks directly because the criteria of clustering for signed networks should take the sign of each edge into consideration.

Our algorithm is motivated by the following observation:

Observation. *Positive edges between clusters are more likely to have higher betweenness centrality.*

Here, we use a real data from Ref. [1] to illustrate this observation. This data set describes the relations between sixteen tribal groups of the Eastern Central Highlands of New Guinea [19]. Relations between tribal groups in the Gahuku-Gama alliance structure can be friendly ("rova") or antagonistic ("hina"). We model the data set as a graph with edge weights $+1$ for friendship and -1 for enmity (see Fig. 1), whose matrix is also shown in the following:

$$\begin{matrix}
0 & 1 & -1 & -1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 1 \\
1 & 0 & -1 & 0 & -1 & -1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 1 & 1 \\
-1 & -1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & -1 & -1 \\
-1 & -1 & 1 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & 1 & 1 & -1 & 0 & 0 & -1 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & -1 & 0 & 0 \\
0 & -1 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 1 & -1 & 0 & 1 & 0 & -1 & 0 \\
0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & -1 & -1 & 0 & 1 & -1 & 0 & -1 & -1 \\
-1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & -1 & -1 & -1 \\
0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 1 & 1 & -1 & 0 & 0 & 1 & -1 & -1 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & 1 & 0 & 0 & -1 \\
1 & 1 & 0 & 0 & -1 & 0 & 0 & 0 & -1 & -1 & -1 & -1 & -1 & 0 & 0 & 1 \\
1 & 1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1 & 1 & 0
\end{matrix}$$

[1] Note that the density drop ratio is actually the ratio of negative edges to positive edges.

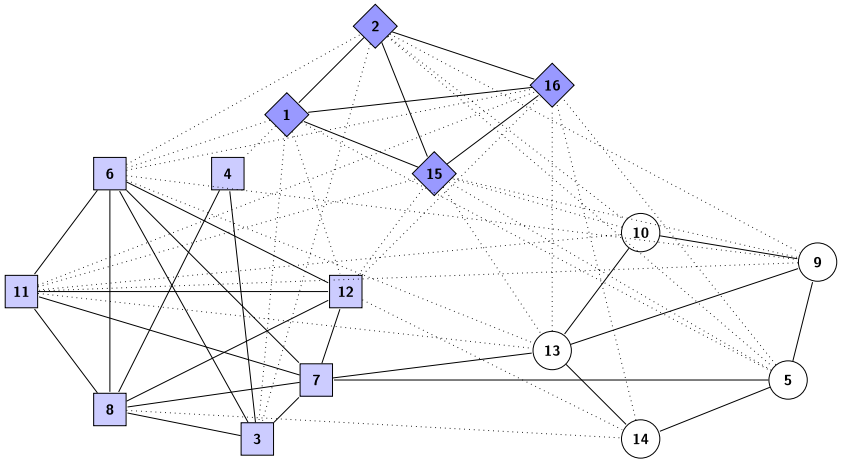


Fig. 1. The solid lines denote positive links, and the dotted lines denote negative links. The real communities are marked by different shapes (i.e., squares, diamonds and circles).

Table 1
Edge betweenness score of $G^+[V']$.

	v_3	v_4	v_6	v_7	v_8	v_{11}	v_{12}	v_5	v_9	v_{10}	v_{13}	v_{14}
v_3	0	4.5	2.17	9.67	1.67	0	0	0	0	0	0	0
v_4	4.5	0	0	0	6.5	0	0	0	0	0	0	0
v_6	2.17	0	0	6	1.5	1.33	1.33	0	0	0	0	0
v_7	9.67	0	6	0	9	6.33	6.33	**14.33**	0	0	**21.33**	0
v_8	1.67	6.5	1.5	9	0	2.33	2.33	0	0	0	0	0
v_{11}	0	0	1.33	6.33	2.33	0	1	0	0	0	0	0
v_{12}	0	0	1.33	6.33	2.33	1	0	0	0	0	0	0
v_5	0	0	0	**14.33**	0	0	0	0	6.33	0	0	5.33
v_9	0	0	0	0	0	0	0	6.33	0	2	5.33	0
v_{10}	0	0	0	0	0	0	0	0	2	0	9	
v_{13}	0	0	0	**21.33**	0	0	0	0	5.33	9	0	6.33
v_{14}	0	0	0	0	0	0	0	5.33	0	0	6.33	0

The true clusters of this data set are (1, 2, 15, 16), (3, 4, 6, 7, 8, 11, 12) and (5, 9, 10, 13, 14). At first, when we consider only the positive edges, we get two connected components, one is {1, 2, 15, 16} and the other is $V' = \{3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14\}$. Table 1 lists the edge betweenness scores of $G^+[V']$, where $G^+[V']$ is the subgraph of G^+ induced by V'. We can see that two positive edges (7, 13) and (5, 7) have the top two highest scores 21.33 and 14.33 respectively (marked in bold in Table 1), and they also are the only two positive edges between clusters (3, 4, 6, 7, 8, 11, 12) and (5, 9, 10, 13, 14).

3.2. The Eb& D clustering algorithm

The new Edge-betweenness & Density based Clustering Algorithm for signed networks (Eb&D Clustering Algorithm for short) is presented at Table 2. The connected components of G^+ is clearly an original cluster candidates. The output of Eb&D Clustering Algorithm is a hierarchical tree, whose size can be controlled by user predefined parameters α, β. This process will clearly highlight meaningful clusters, and it does not need the number of clusters k as an input in advance, which is its greatest advantage. Thus, Eb&D Clustering Algorithm should be more applicable for networks in which we cannot initially predict the number of clusters.

3.3. How to choose parameters α and β?

Parameters α and β are user-specified, which are used to control the density and the ratio of negative edges of a final cluster respectively. But how to choose their values? Here we give a guide for the default value of α and β.

The density of G is denoted by $d(G)$, and the density of G^+ is denoted by $d(G^+)$. Denote the graph from G by taking all edges as positive by $|G|$, and its density by $d(|G|)$. Obviously, $d(|G|) \geq d(G^+)$.

Define

$$\gamma = \frac{d(|G|) - d(G^+)}{d(|G|)},$$

which is actually the ratio of negative edges in the input graph G.

Table 2
Edge-betweenness & density based clustering algorithm for signed networks.

Input	A signed (unweighted) network G with both positive and negative edges
Output	A hierarchical system which illustrates the inclusion relationships of clusters
Step 0	G^+: the subgraph consist of all positive edges; *Cluster*[k]: the set of clusters detected in the kth level; C: a cluster (i.e., a set of vertices); $G^+[C]$: the subgraph G^+ induced by C; *Cluster_cont*[k]: the set of clusters in the kth level which need further partition; α: a parameter given by user to control the density of clusters; (default value is set as Eq. (4)) β: a parameter given by user to control the drop of the densities of C in G^+ and G; (default value is set as Eq. (3)) $k := 1$.
Step 1	Compute the connected components of G^+; Add them to *Cluster*[k].

Step 2	**For** every cluster C in *Cluster*[k]
	Calculate the density of C by Eq. (2)
	in both G (denoted by $d(C)$)and G^+ (by $d^+(C)$) respectively;
	If $d(C) \geq \alpha$ and $\frac{d^+(C)-d(C)}{d^+(C)} < \beta$
	C does not need further partition; Color it with red.
	Else;
	add C to the set *Cluster_cont*[k].

Step 3	**While***Cluster_cont*[k] is not empty Choose one cluster C from *Cluster_cont*[k] and remove it; Calculate the edge-betweenness for each edge of the subgraph $G^+[C]$; Keeping removing edges with maximum edge-betweenness until $G^+[C]$ becomes disconnected; Add all these connected components to the set *Cluster*[$k+1$]; **end** $k = k + 1$; go to Step 2.

As negative edges are likely to exist between clusters, γ can be considered as an upper bound of the ratio of negative edges that a true cluster C can contain, thus,

$$0 \leq \frac{\#negative\ edges\ in\ C}{\#positive\ edges\ in\ C} \leq \frac{\gamma}{1-\gamma},$$

which gives us a range that β can be chosen from. Here, we suggest to select

$$\beta = \frac{1}{2} \cdot \frac{\gamma}{1-\gamma} \tag{3}$$

as the default value.

The default value of α will be set as follows, which is depending on the ratio of negative edges in the input graph G:

$$\alpha = \begin{cases} d(G^+), & \text{if } \gamma > 0.5; \\ d(|G|), & \text{otherwise.} \end{cases} \tag{4}$$

Eq. (4) means that if there are many negative edges in a signed network then we should do not expect too much on the final clusters' density, while there are not too much negative edges in a signed network, then the density of the final clusters should be larger. The parameters α and β are user-specified, one can give their values as they desire. But as we will show in Section 4.3, it is more reasonable that choosing their values according to the input graph as we suggest in Eqs. (3) and (4).

3.4. The time complex of the algorithm

The time complexity of Eb&D Clustering Algorithm is polynomial. We mainly spend time on calculating the betweenness scores of all the edges in an unweighted (sub)graph C which involves calculating the shortest paths between all pairs of vertices of C, and this will take $O(|V(C)| \cdot |E(C)|)$ time with the Brandes's algorithm [20]. At each iteration, it will cost $O(|V(G)| \cdot |E(G)|)$ time with an input G. Let h be the number of iterations, and the time complexity of Eb&D Clustering Algorithm will be $O(h|V(G)| \cdot |E(G)|)$. The time complexity of S-Graclus algorithm proposed by Chiang et al. in [17] is $O(t \cdot [|E(G)| + k \cdot |V(G)|])$, where t is the number of iterations, and k is the number of clusters that we need to cluster into. For big data sets, S-Graclus algorithm will work faster than ours, but in S-Graclus algorithm, the number of clusters k should be known in advance, which is hard to predict in practice.

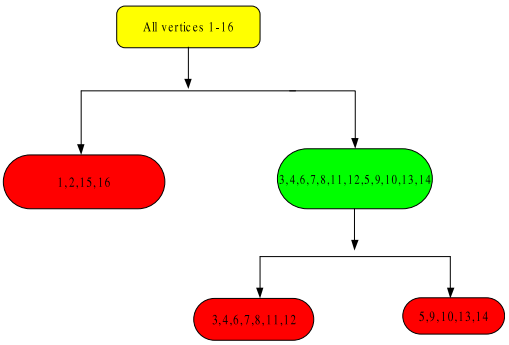


Fig. 2. The hierarchical tree after applying Eb&D clustering algorithm on the 16 tribal groups in the Gahuku-Gama alliance structure. Red rectangle means it do not need partitions any more, while green ones means need further partition. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4. Applications and experiments

To test the validity of Eb&D Clustering Algorithm, we will test it on several classical real-world data sets and also hundreds of synthetic data sets in the following subsections.

4.1. Benchmark data sets

As an application of signed network clustering to real world data, we first use the real data set from Ref. [1], see Fig. 1. The true clusters are (3, 4, 6, 7, 8, 11, 12), (1, 2, 15, 16) and (5, 9, 10, 13, 14). When we apply the new Eb&D Clustering Algorithm on this data set, we obtain exactly the same true correct clustering result (see the red rectangles of the hierarchical tree in Fig. 2).

Actually, since this data set is small, we can illustrate Eb&D's working process in detail. For this data set, the default α is set to be 0.4833, and the default β is set to be 0.5. Step 1, compute the connected components of G^+ which are $V_1 = \{1, 2, 15, 16\}$ and $V_2 = \{3, 4, 6, 7, 8, 11, 12, 5, 9, 10, 13, 14\}$ respectively; Step 2, the subgraph induced by V_1 has high density value $d(V_1) = 1 > \alpha$ and $\frac{d^+(V_1)-d(V_1)}{d^+(V_1)} = \frac{1-1}{1} < \beta$, thus V_1 does not need partition any more (color V_1 with red); while the subgraph induced by V_2 has lower density value $d(V_2) = 0.2424 < \alpha$, V_2 need further partition (color V_2 with green); Step 3, firstly we calculate the edge-betweenness for each (positive) edge of the induced subgraph $G^+[V_2]$, and find that edge (7, 13) has the highest edge-betweenness value 21.333, but $G^+[V_2] - (7, 13)$ is still connected, then keep removing the edge (7, 5) with the second highest edge-betweenness value 14.333, and we find that $G^+[V_2] - (7, 13) - (7, 5)$ becomes disconnected. We get two connected components which are $V_3 = \{3, 4, 6, 7, 8, 11, 12\}$ and $V_4 = \{5, 9, 10, 13, 14\}$ with density $d(V_3) = 0.7143$ and $d(V_4) = 0.6$ respectively, and there is no negative edges in either $G[V_3]$ or $G[V_4]$, which mean both V_3 and V_4 do not need further partitions (color both V_3 and V_4 red). There is no green subgraph any more, the algorithm stops, and the subgraphs with red color are recognized as the final clusters.

The second real-world signed network is a relation network of ten parties of the Slovene Parliament in 1994, see Fig. 3. The adjacency matrix is also shown below.

$$\begin{bmatrix}
0 & -215 & 114 & -89 & -77 & 94 & -170 & 176 & 117 & -210 \\
-215 & 0 & -217 & 134 & 77 & -150 & 57 & -253 & -230 & 49 \\
114 & -217 & 0 & -203 & -80 & 138 & -109 & 177 & 180 & -174 \\
-89 & 134 & -203 & 0 & 157 & -142 & 173 & -241 & -254 & 23 \\
-77 & 77 & -80 & 157 & 0 & -188 & 170 & -120 & -160 & -9 \\
94 & -150 & 138 & -142 & -188 & 0 & -97 & 140 & 116 & -106 \\
-170 & 57 & -109 & 173 & 170 & -97 & 0 & -184 & -191 & -6 \\
176 & -253 & 177 & -241 & -120 & 140 & -184 & 0 & 235 & -132 \\
117 & -230 & 180 & -254 & -160 & 116 & -191 & 235 & 0 & -164 \\
-210 & 49 & -174 & 23 & -9 & -106 & -6 & -132 & -164 & 0
\end{bmatrix}$$

The entries in the above matrix are the weights of links in the network estimated by 72 questionnaires among 90 members of the Slovene National Parliament. The questionnaires were designed to estimate the distance of the ten parties on a scale from -3 to 3, and the final weights were the averaged values multiplied by 100. The ten parties fall into two communities: (1, 3, 6, 8, 9) and (2, 4, 5, 7, 10), a hard-partition of the network.

When we apply Eb&D Clustering Algorithm on this data set, we regard it as an unweighted network. The resulting hierarchical tree is shown in Fig. 4, which is the same as the true clusters. Actually, after the Eb&D Clustering Algorithm finishes Step 1 (compute the connected components of G^+), we get the true clustering result $V_1 = \{1, 3, 6, 8, 9\}$ and $V_2 = \{2, 4, 5, 7, 10\}$ with densities satisfying the termination conditions.

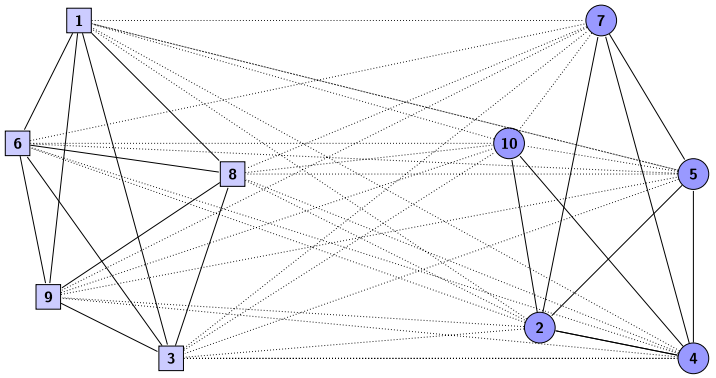


Fig. 3. The solid lines denote positive links, and the dotted lines denote negative links. The real communities are marked by different shapes (i.e., squares and circles).

4.2. Synthetic data sets

Section 4.1 presented two benchmark data sets whose clustering results are already known. Such data sets are desirable for testing purposes but are difficult to obtain. Thus to further test the accuracy of our algorithm, we may need to generate networks.

We generate a general signed network with k clusters by inputting a vector of cluster sizes $(s_1, s_2, \ldots, s_k)$ with two parameters: sparsity $0 < \rho < 1$ and noise value $0 < \epsilon < 1$. The sparsity ρ represents the percentage of edges sampled from the *complete* perfect k-weakly balanced signed networks, and the noise value ϵ specifies the percentage of edges flipping the sign. Initially, let $G = (V, E)$ be a complete graph on $\sum_1^k s_k$ vertices with a perfect k-clustering (edges within clusters are all positive and edges between clusters are all negative) with clusters of sizes $s_1, s_2, \ldots, s_k$ respectively. Next, we uniformly sample totally $\rho|E(G)|$ entries from G to form a *sparse* perfect k-weakly balanced network $G^{(1)}$, and then generate a new graph $G^{(2)}$ from $G^{(1)}$ by randomly choosing number of $\epsilon|E(G^{(1)})|$ edges in $G^{(1)}$ to flip the signs. That is, if $e = xy$ is one of such edges (and $x \neq y$) then we let $w'(e) = -w(e)$, where $w'(e)$ is the weight of e in $G^{(1)}$. Note that by randomness these changed edges can locate either between clusters or in a cluster.

4.2.1. Error functions

If an acceptable clustering for a given network is already established, as is the case in the two benchmark data sets in Section 4.1, we may use it as the reference or the ground truth, and then define an error function based on the number of vertices in incorrect clusters with respect to the known ground truth clustering.

We denote a clustering result $\mathbf{c}$ as a $n \times 1$ vector where $\mathbf{c}(i) \in \{1, 2, \ldots, n\}$ is the clustering result for vertex i and $n = |V(G)|$. The following is a pseudocode which was presented in [21] for such an error function. Note that this error function algorithm takes in account relabelings of clusters, because as we all know, the order of the clusters does not matter. As such, the lack of sensitivity is desirable.

Input : **c_test** : $n \times 1$ vector, clustering result obtained from algorithm; **c_true** : $n \times 1$ vector, established reference clustering result
Output : error E where $0 \leq E \leq 1$

```
A_test ← n × n zero matrix
A_true ← n × n zero matrix
k_test ← number of clusters in c_test
k_true ← number of clusters in c_true
for all k from 1 to k_test do
  current_cluster ← list of vertices i such that c_test(i) = k
  for all i, j in current_cluster do
    A_test(i, j) ← 1
  end for
end for
for all k from 1 to k_true do
  current_cluster ← list of vertices i such that c_true(i) = k
  for all i, j in current_cluster do
    A_true(i, j) ← 1
  end for
end for
E ← (sum of elements of |A_test − A_true|) / n^2
```

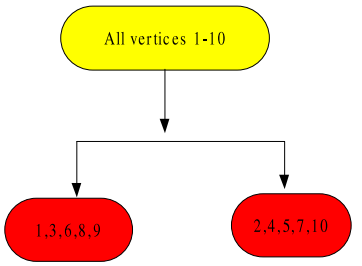


Fig. 4. The hierarchical tree after applying Eb&D on the relation network of ten parties of the Slovene Parliament in 1994.

4.2.2. Evaluation results

It is worth noting that we use the original noiseless graph G as the reference. However, as sparsity ρ decreases and noise ϵ increases, it may be the case that too much relationships between entities have been missed or changed that the reference clustering is no longer the most desired clustering. Thus, in the following experiments, the value of ϵ is chosen up to 0.5, and when ϵ's value gets bigger, the value of ρ should get big simultaneously such that enough "true" information are reserved and the reference are still reliable.

Firstly, we generate smaller signed networks with 10 clusters of size (20, 20, 20, 20, 20, 10, 10, 10, 10, 10) respectively. Considering there are two parameters ρ and ϵ when generating the networks, for each fixed noise value $\epsilon \in [0.1, 0.5]$, we generate 100 networks each with a random ρ value, and calculate the *average error* as the "error rate" corresponding to the fixed ϵ. The results are plotted in Fig. 5(a). We can see that Eb&D performs better than S-Graclus clustering algorithm.

Next we generate bigger signed networks with 5 cluster of size {200, 200, 200, 200, 200} respectively. We generate 100 signed networks for each noise value $\epsilon \in [0.1, 0.3]$ with fixed sparsity $\rho = 0.3$, and also generate 100 signed networks for each noise value $\epsilon \in [0.35, 0.5]$ with fixed sparsity $\rho = 0.5$, and calculate the *average error* as the "error rate" corresponding to the fixed ϵ. The results are plotted in Fig. 5(b) and (c). We can see that for these median size data sets, Eb&D also performs better than S-Graclus clustering algorithm.

4.3. The influences of α and β on the performance of Eb&D

Parameters α and β are user-specified to control the density and the ratio of negative edges of final clusters respectively. In Eqs. (3) and (4), we give a guide for their default values. But of course, one can define α and β as they desire. In the following we will show that it is more reasonable to use the default values than random values.

We generate 100 signed networks for each noise value $\epsilon \in [0.1, 0.5]$, and each of the 100 signed networks is with a random sparsity ρ value. For each signed network, we compute the error rate of Eb&D algorithm with the default values of α and β, and also compute the error rate of Eb&D algorithm with two randomly chosen values of α and β from [0, 1]. Results are shown in Fig. 6. When $\epsilon \in \{0.1, 0.3\}$, Eb&D algorithm with default values of α and β always performs better than that with random values; when $\epsilon \in \{0.4, 0.5\}$, Eb&D algorithm with default values of α and β still performs better most of the time.[2] These experiments prove the validity of our strategy of choosing values of α and β according to Eqs. (3) and (4) based on the input graph.

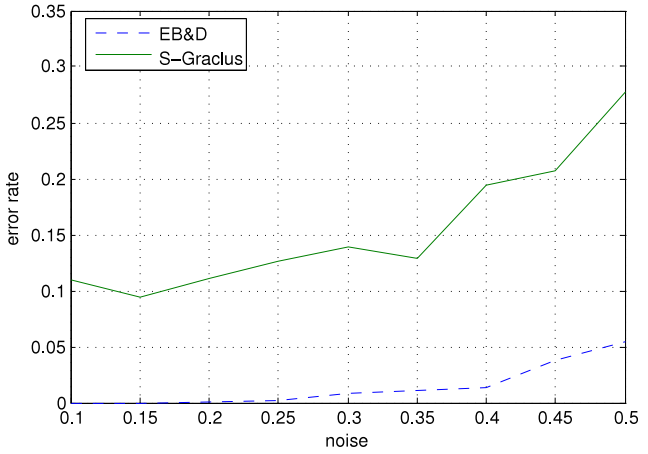
4.4. Advantage and disadvantage of Eb&d algorithm

Experiments have showed that Eb&d provides accurate results for both benchmark data sets and also generated data sets. For example, for the sixteen tribal groups of the Eastern Central Highlands of New Guinea, the S-Graclus algorithm returned the clustering (1, 2, 15, 16), (3, 4, 6, 8, 11, 12), and (5, 7, 9, 10, 13, 14), where 7 was misplaced, while Eb&D returned an accurate result.

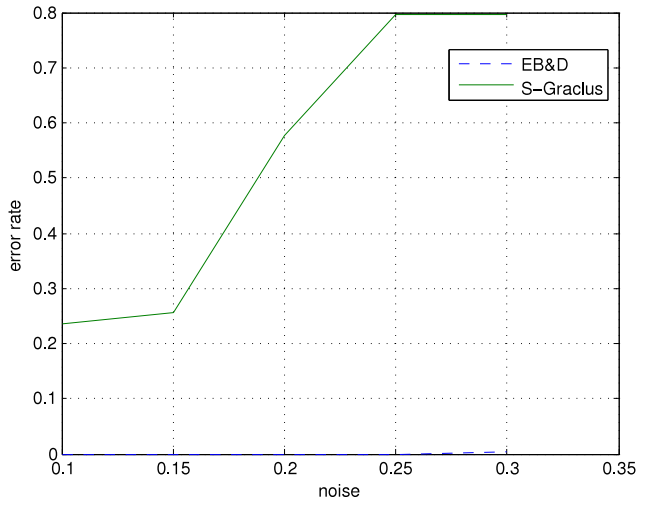
Beyond Eb&d's accuracy, the biggest advantage of Eb&D clustering algorithm is that it does not need the number of final clusters k in advance. For other most existing algorithms (such as S-Graclus), k should be known prior, but in practice, it is hard to predict k beforehand.

The biggest disadvantage of Eb&d Algorithm is time-consuming for very large networks. During its process, it requires to calculate the edge-betweenness of each subgraph (needed to partition) at each iteration of Step 3, which costs a lots of time for very large network. But for small or median size networks, Eb&D Clustering Algorithm can return more accurate results.

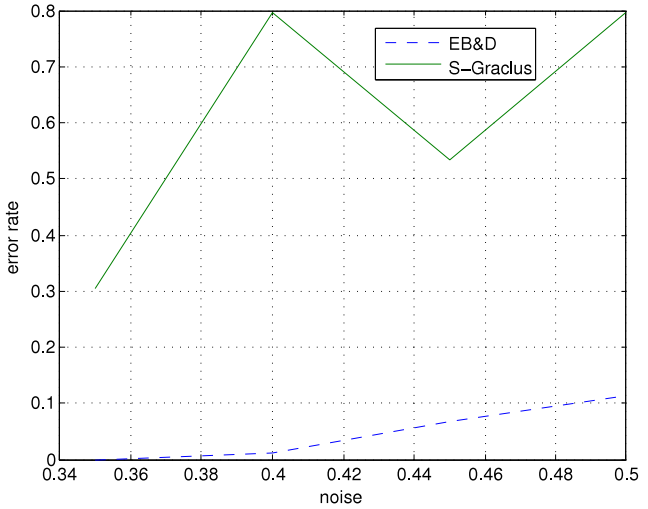
[2] Note that when $\epsilon \in \{0.4, 0.5\}$, the reference results are not reliable any more.



(a) For smaller data sets.

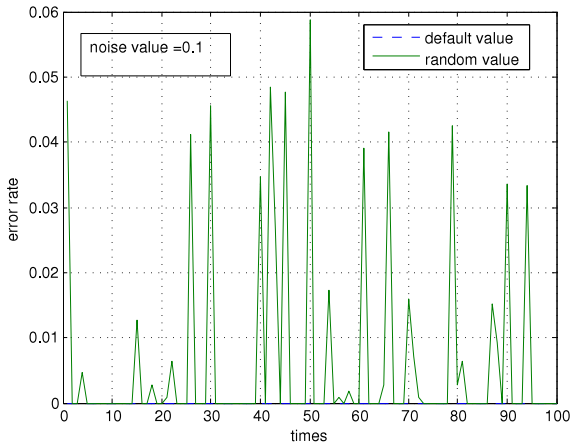


(b) For bigger data sets with fixed $\rho = 0.3$.

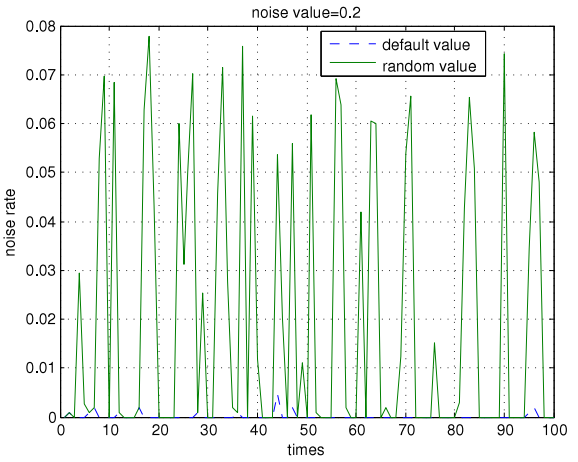


(c) For bigger data sets with fixed $\rho = 0.5$.

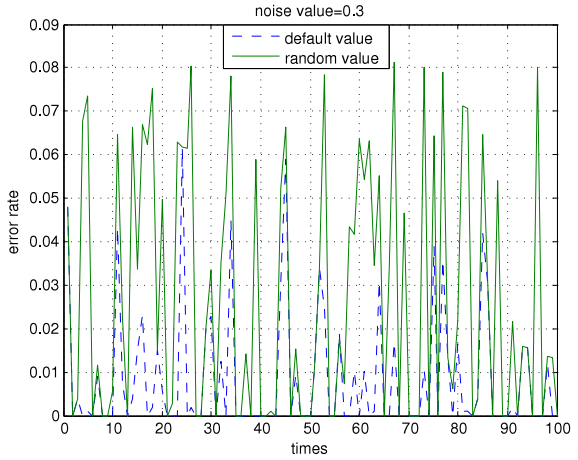
Fig. 5. Error rate tested on synthetic data sets (with respect to number of misplaced vertices) of Eb&D and S-Graclus algorithms.



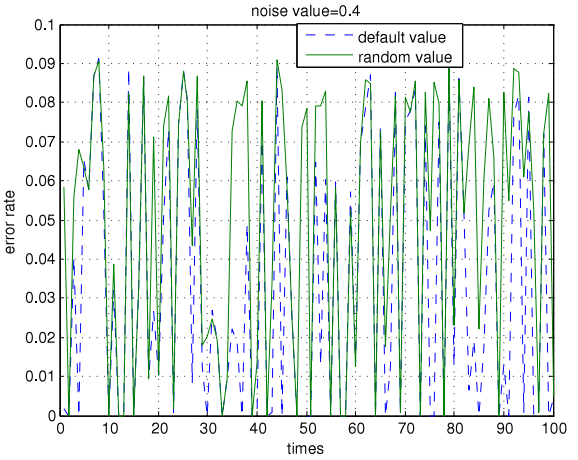
(a) $\epsilon = 0.1$.



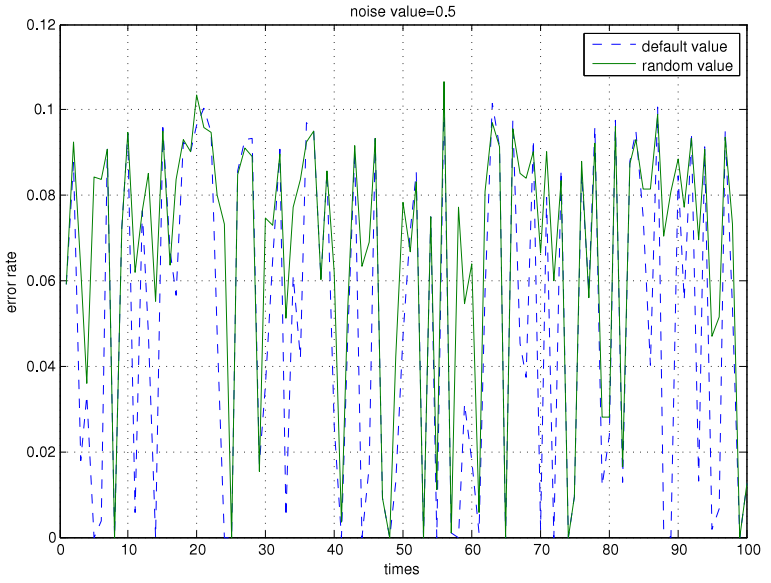
(b) $\epsilon = 0.2$.



(c) $\epsilon = 0.3$.



(d) $\epsilon = 0.4$.



(e) $\epsilon = 0.5$.

Fig. 6. Error rates tested on 100 synthetic data sets of Eb&D with default values and randomly choosing values of α and β respectively.

5. Conclusions

In this paper, we propose a new clustering method for signed networks, called Eb&D, which takes both the density and the edge-betweenness of edges of a subgraph into consideration. Eb&D algorithm is polynomial, and can provide accurate clustering results for small or median size networks, but we admit it is time-consuming for very large networks since it need to calculate the edge-betweenness. The biggest advantage of Eb&D is that it does not need the number of clusters in prior, while most current algorithms need the number of clusters k in advance as an input. Thus, Eb&D Clustering Algorithm should be more applicable for networks in which it is hard to predict the number of clusters.

Acknowledgments

We very appreciate Dr. Chiang et al.'s help to provide their S-Graclus code. We thank Editor and anonymous referees for their valuable comments, which help improve this work a lot. This work is supported by Natural Science Foundation of China with No. 11401346; Shandong Province Natural Science Foundation of China with No. ZR2014FM034; and also by China Postdoctoral Science Foundation with No. 2014M561910.

References

[1] K. Read, Cultures of the central highlands, new Guinea, Southwest. J. Anthropol. 1 (1954) 1–43.
[2] D. Cartwright, F. Harary, Structural balance: a generalization of heider's theory, Psychol. Rev. 63 (5) (1956) 277–292.
[3] F. Harary, On the notion of balance of a signed graph, Mich. Math. J. 2 (2) (1953) 143–146.
[4] J.A. Davis, Clustering and structural balance in graphs, Hum. Relat. 20 (2) (1967) 181–187.
[5] Jianbo Shi, Jitendra Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. Mach. Intell. 22 (8) (2000) 888–905.
[6] Brian W. Kernighan, Shen Lin, An efficient heuristic procedure for partitioning graphs, Bell Syst. Tech. J. 49 (2) (1970) 291–307.
[7] Michelle Girvan, Mark E.J. Newman, Community structure in social and biological networks, Proc. Natl. Acad. Sci. 99 (12) (2002) 7821–7826.
[8] Mark E.J. Newman, Fast algorithm for detecting community structure in networks, Phys. Rev. E 69 (6) (2004) 066133.
[9] Jon M. Kleinberg, Authoritative sources in a hyperlinked environment, J. ACM 46 (5) (1999) 604–632.
[10] Y. Ou, C.Q. Zhang, A new multimembership clustering method, J. Ind. Manag. Optim. 3 (4) (2007) 619–624.
[11] B. Yang, D.Y. Liu, D. Jin, H.B. Ma, Complex network clustering algorithms, J. Softw. 20 (1) (2009) 54–66.
[12] F.D. Malliaros, M. Vazirgiannis, Clustering and community detection in directed networks: A survey, Phys. Rep. 533 (4) (2013) 95–142.
[13] P. Doreian, A. Mrvar, A partitioning approach to structural balance, Soc. Networks 18 (2) (1996) 149–168.
[14] N. Bansal, A. Blum, S. Chawla, Correlation clustering, Mach. Learn. 56 (1–3) (2004) 89–113.
[15] B. Yang, W.K. Cheung, J. Liu, Community mining from signed social networks, IEEE Trans. Knowl. Data Eng. 19 (10) (2007) 1333–1348.
[16] P. Anchuri, M. Magdon-Ismail, Communities and balance in signed networks: A spectral approach, in: Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining, (ASONAM 2012), IEEE Computer Society, 2012, pp. 235–242.
[17] K.Y. Chiang, J. Whang, I.S. Dhillon, Scalable clustering of signed networks using balance normalized cut, in: Proceedings of the 21st ACM International Conference on Information and Knowledge Management, ACM, 2012, pp. 615–624.
[18] I.S. Dhillon, Y. Guan, B. Kulis, Weighted graph cuts without eigenvectors a multilevel approach, IEEE Trans. Pattern Anal. Mach. Intell. 29 (11) (2007) 1944–1957.
[19] P. Hage, F. Harary, Structural Models in Anthropology, first ed., Cambridge University Press, Cambridge, UK, 1983.
[20] Ulrik Brandes, A faster algorithm for betweenness centrality*, J. Math. Sociol. 25 (2) (2001) 163–177.
[21] Xingqin Qi, Ruth Luo, Eddie Fuller, Rong Luo, C.Q. Zhang, Signed quasi-clique merger: A new clustering method for signed networks with positive and negative edges, Int. J. Pattern Recognit. Artif. Intell. 30 (03) (2016) 1650006.