GraphC: Parameter-free Hierarchical Clustering of Signed Graph Networks

When applied to signed graphs, spectral clustering methods often struggle to capture natural groupings accurately. Recent research shows that these methods become less effective as the size of the signed network increases. To address this problem, we present a new, scalable, hierarchical graph-clustering algorithm called *graphC* for identifying the best clusters in signed networks of various sizes. The *graphC* clustering approach aims to keep the proportion of positive edges high within each cluster while maximizing the negative edges between clusters. One key feature of *graphC* is that it does not require the number of clusters (k) to be predefined. We validate the performance of *graphC* by comparing it to ten other signed-graph clustering algorithms on fourteen datasets. We show *graphC* to be scalable to large signed graphs from Amazon, which include tens of millions of vertices and edges. On average, *graphC* outperforms the second-best algorithm by 18.6% in terms of maximizing the positive edge density within clusters and the negative edge density between clusters. This comparison excludes cases where the third-party algorithms fail to complete their tasks.

CCS Concepts: • **Theory of computation** → **Design and analysis of algorithms**; • **Information systems** → **Clustering**; • **Networks** → Network algorithms.

Additional Key Words and Phrases: Signed Network, Community Detection, Balance Theory, Balanced States

ACM Reference Format:

. 2024. GraphC: Parameter-free Hierarchical Clustering of Signed Graph Networks. 1, 1 (July 2024), 19 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Communities within a network are sets of vertices characterized by denser interconnections than with the broader network. These communities may exhibit distinct vertex sets or overlaps, where vertices participate in multiple communities. Detecting community structure is very important, as it gives us insights into relationships and dynamic mechanisms within complex networks spanning various domains, from biological to social networks. The inherent complexity of these networks frequently causes the resulting datasets to surpass the computational capacities of individual computers, making data partitioning for distributed processing a necessity.

The efficiency of such partitioning schemes in community-detection algorithms depends on various factors. The discovery of community structure is a challenging research topic, and numerous approaches have been proposed in the past that use local optimization [41], statistical inference [33], and machine learning [2]. While state-of-the-art community-discovery algorithms in unsigned graphs adeptly handle vast networks comprising millions of vertices and edges [39], the modeling of unsigned graphs falls short in capturing the complex relationships within the networks. Signed graphs capture positive and negative relationships well and offer more meaningful data representation [40]. However, community discovery methods for signed network graphs struggle to recover communities in graphs featuring mere thousands of vertices and hundreds of thousands of edges [18, 40]. The latest research underscores the challenges spectral methods pose in recovering the community structure in sparse networks, even with the incorporation of

Author's Contact Information:

Manuscript submitted to ACM

normalization techniques [12]. This finding aligns with our conclusions: there is a significant degradation in the performance of spectral methods for clustering large, sparse, signed networks derived from real-world data sources [17]. Balance theory is a pivotal concept in signed networks, explaining the evolution of attitudes within networks. Established by Heider [1] and subsequently formalized mathematically by Harary, who introduced k-way balancing [6, 22], balance theory has found applications in predicting edge sentiment, content and product recommendations, and anomaly detection in various domains [3, 14, 20, 25].

1.1 Spectral Clustering of Real Signed Graphs

Donath and Hoffman [15] introduced spectral clustering in 1973 to optimally partition the graph using the eigenvectors of the adjacency matrix. We outline the generic spectral clustering process in four steps: (1) computation of the Laplacian variant and identification of clusters (k); (2) derivation of k eigenvectors corresponding to the k smallest eigenvalues; (3) formation of the eigenvector matrix U to reduce dimensionality; and (4) application of k-means++ to cluster features. The SigNeT package, employed for spectral methods in signed networks, encompasses multiple Laplacian variants, all presumed to be positive semi-definite [19]. The scalability of spectral clustering algorithms is limited in the context of real signed networks: (i) the substantial time required for the solver to formulate eigenvectors with their associated eigenvalues, and (ii) when dealing with large matrices, the approximations can become unstable and lead to errors due to something called spectral pollution or eigenvalue pollution [5]. A recent survey has provided a proof-of-concept and practical evaluation of these limitations in actual signed graphs [17]. The survey falls short of delineating the breaking points of signed Laplacians concerning algorithmic assumptions (e.g., small world and diameter) and characteristics specific to signed networks (e.g., density or sparsity) [17].

A key limitation of state-of-the-art signed graph clustering algorithms is determining the optimal number of communities k before running the algorithm. The elbow method is widely adopted, albeit subjective and not reliable [31]. The eigenvector pollution, network sparsity, average degree, and overall structure can influence the choice of k in real large graphs [17]. If the ground truth is not available, the metrics adopted might not be able to measure the quality of communities discovered by the methods [17]. Thus, parameter-free techniques are preferable.

1.2 Research Contributions

This paper introduces a novel *graph clustering* algorithm called *graphC*. The main contributions are:

- We prove that Harary cuts from a *balanced* signed network are equivalent to an eigenvector with a zero eigenvalue of that balanced network.
- The *graphC* algorithm effectively implements this new duality finding by directly performing Harary cuts on the balanced signed graph.
- The *graphC* algorithm is k-independent: it does not require a predefined number of clusters to produce a high-quality clustering. The parameters used in the algorithm are dependent on computing resources.
- The *graphC* algorithm identifies optimal clusters without requiring a spectral decomposition solver. The algorithm is eigenvalue pollution-free for large sparse matrices.
- The *graphC* addresses the positive/negative edge imbalance that naturally occurs in most signed networks by equally considering the contribution of positive and negative edges to optimize the clustering.

2 Related Work

Existing approaches for community detection in signed networks form a diverse landscape. Xhiang et al. propose an algorithm for clustering signed graphs utilizing the balanced normalized cut, demonstrating commendable clustering results yielding an efficient approach [9]. The *Signed Positive over Negative Generalized Eigenproblem* spectral-based approach necessitates the specification of k since it formulates clustering as a generalized eigenvalue problem, enhancing a well-defined objective function [11]. Relying on the signed Hessian for clustering also requires a predefined k. It emphasizes the advantages of non-backtracking operators with the computational and memory efficiencies inherent in real symmetric matrices [36].

He et al. [24] use modified social network analysis and triangle balancing heuristics ("the friend of my friend is my friend") to address the issue of cluster discovery based on a modified version of Heider's balance theory. First, the authors employ a signed mixed-path aggregation (SIMPA) method to construct the vertex embedding. Consequently, this vertex embedding produces probabilities for cluster assignments. The clustering process involves training with a weighted combination of supervised and unsupervised loss functions, with the unsupervised component being a probabilistic balanced normalized cut. The algorithm uses SPONGE (SPO) labels for some signed graphs to execute the clustering, building a dependency on other signed graph clustering algorithms. Mercado et al. [32] propose to combine the positive Laplacians (part of the Laplacian matrix with positive edge weights) with negative Laplacian (part of the Laplacian matrix with negative edge weights) using matrix power means. The result is a modified version of the Laplacian matrix that considers both positive and negative relationships in the graph. Next, homophily is the tendency of a vertex to connect to similar ones, and negative connections introduce noise as those connections are to dissimilar vertices. The solutions to design the graph neural networks (GNN) for signed network clustering utilizing the balance theory showed initial promise [8, 29]. However, the issue of misrepresenting the negative edges in the resulting embedding proved to be a roadblock to further development of community discovery and recommender systems in this direction [13, 38]. The *neutrosophic* theory, introduced in [21], proposes a novel signed graph convolutional network (SGCN) to encapsulate better the architecture of the signed network in the compact space and overcome the embedding issues. The clustering in the neutrosophic feature space identifies overlapping communities in signed networks [21].

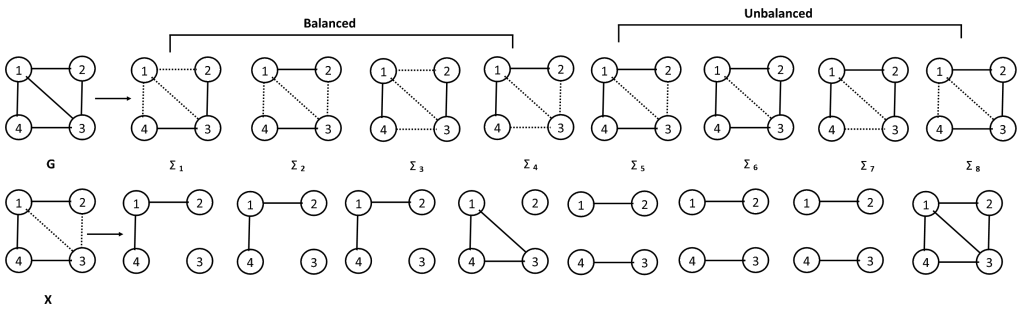

Fig. 1. **Dotted** lines are negative edges whereas **sold** lines are positive edges. Top: A graph G, a balanced graph Σ_1 obtained by switching (changing the sign of the edges originating in) v_1, a balanced graph Σ_2 obtained by switching v_1 and v_2, a balanced graph Σ_3 by switching v_2 and v_3, and a balanced graph Σ_4 by switching v_4. Σ_5, Σ_6, Σ_7 and Σ_8 are examples of unbalanced states. Bottom: Harary Cuts of nearest stable states [35] of Σ_5.

3 Balanced-Spectral Duality in Signed Networks

In this section, we present the terminology and properties of signed graphs and introduce spectral analysis and the duality theory of near-balanced states. Let Σ_i be a signed graph with the unsigned topology G, as illustrated in Figure 1. Let Σ_{ij} be the collection of vertex sets discovered by the clustering algorithm j. The *graphC* algorithm minimizes the fraction of positive edges *between* communities and the fraction of negative edges *within* communities for algorithm j applied to the signed graph Σ_i. We quantify the positive edges outside communities as $pos_{out}(\Sigma_{ij})$ and the negative edges within communities as $neg_{in}(\Sigma_{ij})$ for algorithm j operating on the signed graph Σ_i.

3.1 Fundamental Cycle Basis

Definition 3.1. *Graph Σ_i is a **subgraph** of a graph G if **all** edges and vertices of Σ_i are contained in G.*

Definition 3.2. *A **path** is a sequence of m edges that connect a sequence of n vertices in a graph. A **connected graph** has a path between every pair of vertices. A **cycle** is a path that begins and ends at the same vertex. A **cycle basis** is a set of simple cycles that forms a basis of the cycle space (see below).*

Definition 3.3. *Let T be a spanning tree of the underlying graph G, and let e be an edge in G between vertices x and y that is* NOT *in the spanning tree T. Since the spanning tree spans all vertices, the unique path in T between vertices x and y does not include m. A **fundamental cycle** in G is the cycle that is created when adding edge e to this path in T.*

Corollary 3.1. *For a given connected graph G, we create the fundamental cycle basis when we select all cycles formed by combining a path in the tree and a single edge outside the tree. For a graph G with n vertices and m edges, there are exactly $m - n + 1$ fundamental cycles because the spanning tree includes $n - 1$ edges.*

3.2 Balanced Signed Graphs

Definition 3.4. *The **signed graph** $\Sigma = (G, m)$ consists of the underlying unsigned graph G and an edge signing function $m \longrightarrow \{+1, -1\}$. The edge m can be positive m^+ or negative m^-. The **sign** of a sub-graph is the* product *of the signs of its edges. A **balanced signed graph** is a signed graph where every cycle is positive. The **frustration** of a signed graph is defined as the minimum number of candidate edges whose signs need to be switched for the graph to reach a balanced state. Figure 1 (top, balanced states) provides an example.*

Theorem 3.1 ([6]). *If a signed subgraph Σ is balanced, the following are equivalent:*

(1) *Σ is balanced (all circles are positive).*
(2) *For every vertex pair (v_i, v_j) in Σ, all (v_i, v_j)-paths have the same sign.*
(3) *$Fr(\Sigma) = 0$ (the frustration of Σ is zero).*
(4) *The vertices in the network can be grouped into two sets U and W where the edges between the vertices within the sets U and W are positive and the vertices between the sets are negative. (U, W) bi-set is called Harary-bipartition.*

The formation of the Harary cut starts with deleting the negative edges in each balanced state in the frustration cloud of the signed graph, as illustrated in Figure 1 (bottom). This figure's trivial unbalanced signed graph consisting of 4 vertices and five edges yields eight nearest balanced states. Note that a Harary cut can yield multiple connected components, and the input graph might contain many initially connected components where Harary cuts and balancing take place in every element. In prior work, we have characterized signed graphs through the frustration cloud [35], a

collection of nearest balanced states, and then introduced an improved and parallelizable way to efficiently discover the nearest balanced states in large signed graphs [16].

3.3 Spectral Clustering and Balanced States

A signed graph is *balanced* if the graph has no cycles with an odd number of negative edges. The *switch* operation flips the sign of all edges connected to a specific vertex as shown in Figure 1 when switching v_1 in G to obtain Σ_1. Switching equivalence in this context means that two balanced signed graphs can be transformed into each other through a series of switch operations, as shown in the following example. The balanced signed graphs of the same underlying graph G in Figure 1 (Top) are switching equivalent [30]. For example, Σ_2 and Σ_3 in Figure 1 (Top) are switching equivalent because we can obtain Σ_3 by switching v_1 and v_3 in Σ_2. The key point is that switching operations preserve the balanced nature of the graph as they do not alter the overall balance or imbalance of the graph [35]. We get the same eigenvalues, but the eigenvectors only differ by a multiple of -1 for each vertex switched.

Table 1. Balanced signed graphs Σ_0, Σ_1, and Σ_2 are isospectral: they have identical eigenvalues, but their eigenvectors differ. Σ_5 is an unbalanced graph, so its eigenvalues and eigenvectors differ.

	Balanced Graphs												Unbalanced Graph			
Graph	Σ_0				Σ_1				Σ_2				Σ_5			
Eigen values	0	2	4	4	0	2	4	4	0	2	4	4	$2-\sqrt{2}$	$3-\sqrt{3}$	$2+\sqrt{2}$	$3+\sqrt{3}$
	1	0	1	0	1	0	1	0	1	1	1	1	1	1	-1	1
Eigen	1	1	0	1	-1	1	0	1	1	1	0	1	$\sqrt{2}$	0	$\sqrt{2}$	0
vectors	1	0	-1	-2	-1	0	1	-2	-1	0	1	2	-1	1	1	1
	1	-1	0	1	-1	-1	0	1	-1	1	0	-1	0	$\frac{1+\sqrt{3}}{2}$	0	$\frac{1-\sqrt{3}}{2}$

Note that the spectral clustering yields as many zero eigenvalues as there are clusters in the data [30]. The zero eigenvalues have an eigenspace with a basis of vectors that only have zeros and ones as elements, and these vectors indicate which vertices belong to each cluster [30]. In Table 1, for the connected graph G of 4 vertices with all positive edges, the eigenvector with the zero eigenvalue consists entirely of ones. All balanced signed graphs are equivalent to all positive signed graphs with the same topology. Since the Laplacian matrices of balanced signed graphs of the same underlying graph are isospectral. Table 1 exemplifies this using the graph G from Figure 1 (Top, balanced states) whereas the Σ_5 column in Table 1 shows the spectral clustering of an unbalanced state.

Thus, when performing spectral analysis on such graphs, the eigenvalues remain the same, as illustrated in Table 1. The corresponding eigenvectors only differ by a multiple of -1 for each switched vertex, as illustrated in Table 1 for the eigenvalues and eigenvectors of G, Σ_1, and Σ_2 (from Figure 1). This property makes the study of spectral characteristics more manageable, allowing for a direct comparison between balanced signed graphs while preserving their essential structural features. The multiplicity of the 0 eigenvalue counts the number of connected components of the underlying graph when balanced. To see this fact, let $W \subseteq V$ be the set of vertices switched from the original, all-positive graph G, that produces another balanced graph Σ. Let $\mathbf{I}_W$ be the $V \times V$ diagonal matrix whose (v, v)-entry is -1 if $v \in W$ and $+1$ otherwise. Thus, the Laplacian for the balanced signed graph Σ is $\mathbf{L}_\Sigma = \mathbf{I}_W \mathbf{L}_G \mathbf{I}_W$.

We also present a simple new proof of the isospectrality of balanced states, highlighting the signed cycle structure and further indicating the need for non-spectral methods for sentiment analysis.

Theorem 3.2. *The Laplacian matrices of balanced signed graphs of the same underlying graph are isospectral.*

Proof. First, observe that switching does not change the cycle signs. This is because every vertex in a given circle has a degree equal to 2 in that circle subgraph. Switching reverses exactly two edge signs, which produces no change in the cycle sign. Next, we use the characteristic polynomial formulation for the Laplacian from [7]. In there, it was shown that the signs of the cycle subgraphs entirely determine the characteristic polynomial. Since each balanced signed graph of an underlying graph has all cycles positive, they must necessarily have the same characteristic polynomial and, hence, the same eigenvalues. □

Theorem 3.3. *For a connected balanced signed graph Σ, the entries of the 0-eigenvector are $+1$ or -1. Partitioning the vertices along the entry values of the 0-eigenvector corresponds to the Harary cut of the balanced signed graph.*

Proof. Since Σ is balanced, it is isospectral to the underlying graph G. Since G is connected, the dimension of the 0-eigenspace is 1, and the vector $\mathbf{1}$ consisting of all 1's is a basis for that eigenspace of G. Switching a vertex v negates both row and column v in $\mathbf{L}_G$ to produce $\mathbf{L}_\Sigma$, thus negating the v-entry of the basis vector $\mathbf{1}$. □

Note that since an unbalanced signed graph contains negative cycles, the proof of the previous theorem ensures that these cycles must have different eigenvalues. Therefore, if a signed graph is unbalanced, 0 is not an eigenvalue, and the structure and sentiment of the graph effectively merge into its geometry, as shown in Table 1 for Σ_5. In contrast, balanced graphs share the same eigenvalues and allow multiple optimal ways to divide the graph into two components. Spectral methods rely on the geometry of the eigenvectors rather than the sentiment since the sentiment is embedded in the signs of the vectors and not in the proximity of the data (such as in a k-means approach). Thus, we conclude that both the Harary cut and spectral methods involve bisecting the graph. We can create a balanced Laplacian matrix once a signed network is balanced. The spectral decomposition of this matrix reveals the eigenvectors and their eigenvalues. If the graph is balanced and connected, there will be a 1-dimensional kernel, and the corresponding 0 eigenvalue will provide the Harary cut.

4 Measuring Signed Graph Clustering Efficacy

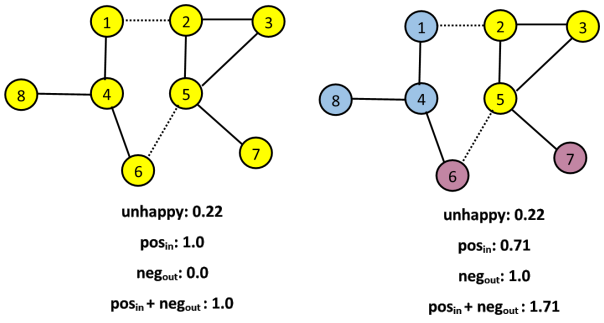

Fig. 2. An illustration emphasizing the advantage of measuring the quality of a clustering assignment in a signed graph using the summation of the fraction of positive edges within communities and the fraction of negative edges between communities. **Dotted** lines are negative edges whereas the **sold** lines are positive edges.

Consider two clustering assignments for the network illustrated in Figure 2. Without ground truth and training data for the community assignment in signed networks, the state-of-the-art uses the *unhappy ratio* to determine the quality

of the clustering [24]. The *unhappy ratio* U_i is defined in Eq. 1 for a set of communities produced by method j in the signed graph Σ_i:

$$U_{ij} = \frac{pos_{between} + neg_{within}}{pos_{between} + pos_{within} + neg_{between} + neg_{within}} \tag{1}$$

The $pos_{between}$, pos_{within}, neg_{within}, and $neg_{between}$ are the number of positive edges between communities, the number of positive edges within communities, the number of negative edges within communities, and negative edges between communities, respectively.

The *unhappy* ratio for both clustering approaches in Figure 2 is 0.22. The clustering assignment on the right seems more effective as it avoids disregarding *all* negative edges, successfully segregates two communities (1,4,8,6 and 2,3,5,7), and accurately classifies *most* positive edges. The *unhappy* ratio is unable to make these distinctions. It fails to capture the quality of the clustering of real signed networks because the *unhappy* ratio favors positive edges and trivial clustering cases in networks where the typical number of negative edges is 5-10 times smaller than the number of positive edges [17].

Consider another signed graph with 1000 edges, where 900 are positive, and 100 are negative, and two cluster assignments. In the first clustering, there are 400 positive edges between clusters ($pos_{between}$), 500 positive edges within clusters (pos_{within}), 50 negative edges within clusters (neg_{within}), and 50 negative edges between clusters ($neg_{between}$), yielding an unhappy ratio (U_1) of 0.45. The second clustering has zero positive edges between clusters, 900 positive edges within clusters, 100 negative edges within clusters, and zero negative edges between clusters, resulting in an unhappy ratio (U_2) 0.1. In this example, the distribution of positive and negative edges is imbalanced. The first clustering can separate some negative edges into different clusters while preserving positive edges within the same clusters. The second clustering is trivial because it places all vertices in a single cluster. Thus, the *unhappy ratio* as defined favors a trivial class when the number of positive edges is much higher than the number of negative edges and vice versa. In the edge sign imbalance case, the *unhappy ratio* is a low value for the clustering approach that puts all vertices into one community. As a remedy, we change the *unhappy ratio* to the *unhappy score*, which accounts for both positive and negative edges as follows:

$$US_{ij} = \frac{pos_{between}}{pos_{between} + pos_{within}} + \frac{neg_{within}}{neg_{within} + neg_{between}} \tag{2}$$

Intuitively, measuring the sum of the fraction of positive edges within communities (pos_{in}) and the fraction of negative edges between communities (neg_{out}) provides a better assessment of the clustering quality as shown in Figure 2 [17]. The unhappy ratio for both clustering assignments is the same (0.22), even though the left assignment ignores all negative edges, while the right assignment considers both positive and negative edges. In Figure 2, the left assignment has an updated $pos_{within} + neg_{between}$ of 1.0. The right assignment has an updated $pos_{within} + neg_{between}$ of 1.71, and the scores reflect better the true clustering quality.

The updated *unhappy score* is the new *graphC* loss measure for a signed graph Σ_i with an unbalanced edge ratio and clustering algorithm j. We rewrite it as a loss function since our objective in clustering is to minimize $\mathcal{L}(\Sigma_{ij})$ by splitting vertices into communities. The pos_{out} and neg_{in} are defined as

$$pos_{out} = \frac{pos_{between}}{pos_{between} + pos_{within}} \quad neg_{in} = \frac{neg_{within}}{neg_{within} + neg_{between}} \tag{3}$$

Thus, we propose to minimize the fraction of violating negative edges $neg_{in}(\Sigma_{ij})$ as well as the fraction of violating positive edges $pos_{out}(\Sigma_{ij})$ simultaneously. In this context, we define the violating edges ($\mathcal{V}$) as the total number of

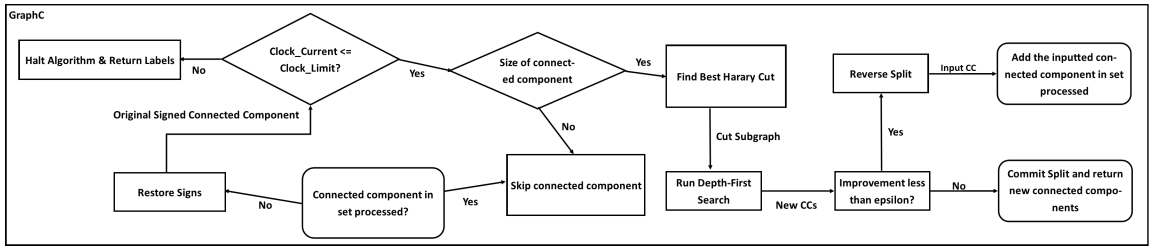

Fig. 3. The *graphC* pipeline: the starting state is "Connected component in set processed?" block.

positive edges between clusters and negative edges inside them.

$$\mathcal{V}_{ij} = \mathcal{L}_{\Sigma_{ij}} = pos_{out} + neg_{in} \tag{4}$$

Symmetrically, we redefine pos_{in} (Eq. 5) and neg_{out} (Eq. 6) measures in terms of fractions to account for the positive and negative edge imbalance.

$$pos_{in} = \frac{pos_{within}}{pos_{between} + pos_{within}} \tag{5}$$

$$neg_{out} = \frac{neg_{between}}{neg_{within} + neg_{between}} \tag{6}$$

Next, we generalize the loss function to accommodate for the graph characteristics:

$$\mathcal{L}_{\Sigma_{ij}}(\alpha, \beta) = \beta(\alpha pos_{out} + (1-\alpha) neg_{in}) + (1-\beta) * \frac{|V_{iso}|}{|V|} \tag{7}$$

The α controls the importance between pos_{out} and neg_{in}. Setting $\alpha = 0.5$ gives equal importance to both. The β parameter controls the fraction of isolated vertices $|V_{iso}|$ produced in the process; $|V|$ is the total number of vertices in the graph. If $\beta = 0$, then the algorithm will completely ignore minimizing the pos_{out} and neg_{in} and will find the cuts that yield the least amount of isolated vertices possible. Note that $\mathcal{L}_{\Sigma_{ij}}$ in Eq. 4 is equal to $\mathcal{L}_{\Sigma_{ij}}(0.5, 1)$ in Eq. 7.

5 The *graphC* Methodology

Section 3 showed that the Laplacian matrices of balanced signed graphs with the same underlying graph are isospectral and that spectral clustering reconstructs a Harary cut. The proposed *graphC* algorithm searches for the optimal Harary cut using the quality measure defined in Equation 7, along with a set of stopping criteria to automatically halt the algorithm and return the final clustering labels for each vertex. Figure 3 outlines the flow of the *graphC* algorithm. It starts by labeling all vertices within the same connected component with the same label, creating an initial clustering assignment set C, as described in Algorithm 1. Isolated vertices receive unique labels, and each isolated vertex is assigned a different label. Specifically, the algorithm first checks whether the connected component is in a set called *processed*. If not, it restores the original signs using the signed network before continuing. Next, the algorithm checks a time limit and ensures that the size of the connected component is greater than the threshold Γ. Once the algorithm identifies the best Harary cut based on the loss function in Equation 7, it evaluates whether the Harary split improves the overall clustering quality, as measured by Equation 8. Finally, if the improvement exceeds a threshold ϵ, the algorithm commits the split; otherwise, it terminates.

Algorithm 1: Split and Label Components

Data: Σ_{ij}, label_counter
Result: Clustering label assignment set $C = \{C_v\}, v \in V$

```
label_counter=0;
for CC, CC ∈ Σ_ij do
    for v, v ∈ CC do
        C_v=label_counter;
    end
    label_counter++;
end
```

Algorithm 2: Best Harary Cut

Data: Σ_{ij}, Set R, label_selected, C, α, β, I
Result: G^{FC}

```
for every vertex v in Σ_ij do
    if C_v! = label_selected then
        R = R + v ;
    end
end
G^F = G \ R;
Σ = GraphBplus(G^F, I) ;
G^FC = Σ|{minL_{α,β,G^F}};
```

Selecting the Best Harary Cut: We use the graph plus algorithm to create stable states and perform Harary cuts. To find the best Harary cut, we pick the one that results in the slightest loss, as defined in Eq. 7. After selecting the Harary cut, we employ Depth-First Search (DFS) to create the connected components. DFS starts at the root vertex and explores each branch as far as possible before backtracking. The process continues until *graphC* has visited all vertices connected to the root vertex. The time it takes for DFS to run is proportional to the number of vertices ($|V|$) plus the number of edges ($|E|$). Algorithm 2 demonstrates the process of selecting the best Harary cut.

Stopping Criteria: After finding the best Harary cut of a particular connected component, the algorithm computes the overall quality of the new clusters of the entire signed network, which is similar to Equation 7 and works as follows:

$$\mathcal{L}^{t}_{\Sigma_{ij}} = (pos_{out} + neg_{in}) \tag{8}$$

The *epsilon* parameter is a defined stopping condition for the graph clustering. If the new clusters result in an improvement of less than *epsilon*, the algorithm undoes the last split of the connected component and adds it to a set called *processed*. The *graphC* algorithm ignores any connected components in the *processed* set going forward. Next, *graphC* restores the original signs of the target connected component before invoking the graphBplus algorithm again. If the improvement of the split is greater than *epsilon*, the split is adopted, and *graphC* proceeds to the next connected component. Algorithm 3 outlines the steps of the proposed community discovery algorithm. The t_l is an optional time limit parameter for the algorithm. If $t_l = -1$, the algorithm will run until it meets the ϵ stopping criteria.

Algorithm 3: Choose component to split

Data: Signed graph Σ_{ij}, spanning tree sampling method M, I, α, β, ϵ, Γ, Time limit t_l

Result: Clustering label assignment set $C = \{C_v\}, v \in V$

label_counter=0;
Call Alg. 1 with inputs $G, label\_counter, C$;
set R = $\emptyset$, processed = $\emptyset$;
while *true* **do**
  if $t \geq t_l$ **then**
    break;
  end
  label_selected=-1 ;
  terminate=true ;
  for *label* $l, l \in C$ **do**
    if $|CC_l| \leq Gamma$ **then**
      continue;
    end
    if l *is not in set processed* **then**
      label_selected=l terminate = false;
    end
  end
  if *terminate == true* **then**
    break;
  end
  $C_t = C$ label_counter_temp = label_counter ;
  G^{FC} = Call Alg. 2 (Σ_{ij}, Set R,label_selected, C,α, β, I);
  for $CC \in G^{FC}$ **do**
    for *vertex* $i \in CC$ **do**
      C_i=label_counter;
    end
    label_counter++;
  end
  Compute $\mathcal{U}_{t+1,G}$;
  if $\mathcal{U}_{t,G} - \mathcal{U}_{t+1,G} \leq \epsilon$ **then**
    $C = C_t$ label_counter = label_counter_temp;
    Append label_selected to processed and exit loop;
  end
  Set $\mathcal{U}_{t,G} = \mathcal{U}_{t+1,G}$ $R = \emptyset$;
end

SOTA	Description
Laplacian_none [26]	spectral clustering using the signed graph Laplacian
Laplacian_sym [26]	spectral clustering using the symmetric Laplacian.
BNC_none [9]	balanced normalized cuts.
BNC_sym [9]	symmetric balanced normalized cuts.
SPONGE_none [11]	SPONGE implementation.
SPONGE_sym [11]	symmetric SPONGE implementation.
Hessian [36]	clustering based on signed Bethe Hessian.
A_sym [10]	spectral clustering using symmetric adjacency matrix.
dns [42]	clusters the graph using eigenvectors of the bns Laplacian matrix.
sns [42]	clusters the graph using eigenvectors of the sns Laplacian matrix.

Table 2. Ten state-of-the-art leading methods from the literature.

Diminishing Returns (Fostering Scalability): An intrinsic phenomenon in hierarchical clustering algorithms is that the deeper we go, the fewer potential improvements we get. Finding the best Harary cut of every connected component is computationally expensive, especially on massive signed graphs, as they might have millions of relatively small connected components, which could yield marginal improvements if any at all. This observation justifies using the optional Γ parameter to save computation time. It represents a trade-off between efficiency and performance. If the connected component's size is less than Γ, the algorithm ignores it and will not find the best Harary cut even if it is not in the *processed* set. Figure 5 illustrates a typical execution of the algorithm (on the signed PPI graph) and its gradually decreasing improvement over each Harary split. The overall improvement dropped from around 0.7 to a meager 0.0007 in the 5^{th} Harary cut.

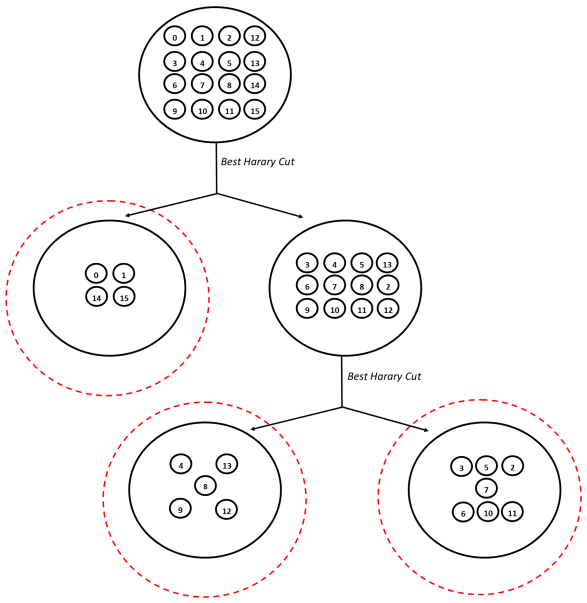

Fig. 4. Execution of the *graphC* algorithm on the Highland signed graph comprising 16 vertices and 58 edges.

Illustrative Execution on Highland Tribes Graph: Figure 4 shows the execution of our proposed algorithm. Assume that $I = 1000$, $\alpha = 0.5$ and $\beta = 1$, $Gamma = 2$, $\epsilon = 0.00000001$, and t_l = -1. First, the algorithm checks if it exceeded the time limit t_l; it proceeds since there is no time limit. Next, the *graphC* algorithm checks whether the initial component at level 0 is in *processed*. The first pass directly proceeds to the next step because the *processed* set is empty. Then, *graphC* restores the original signs and finds the best Harary cut for the processed component. In the first iteration, that is the largest connected component of the graph, and the result is two connected components. This step is iteratively repeated for both components. After performing the Harary split, the algorithm stops splitting the current component when the improvement is less than *epsilon*. Recall that it undoes the last split and commits the connected components found so far, as illustrated in Figure 4 with red dashed circles. The vertices within that component are given a unique final label. Eventually, the two components at level 1 also trigger the same stopping criteria, resulting in 3 clusters.

6 Proof of Concept Implementation

6.1 Implementation and Setup

The implementations of the state-of-the-art methods listed in Table 2 are in Python from the SIGNET repository [10]. The parameters we used for all Konect signed graphs are $I = 1000$, $\alpha = 0.5$, $\beta = 1$, $\epsilon = 0.00000001$, $Gamma = 2$, and $t_l = 100000$ except for WikiConflict and WikiPolitics, where we chose $Gamma = 10$ to speed up the execution. The parameters used for all Amazon signed graphs are $I = 50$, $\alpha = 0.5$, $\beta = 1$, $\epsilon = 0.00000001$, $Gamma = 40$, and t_l = -1. Next, we run the k-means++ clustering ten times with different centroid seeds for all spectral methods to cluster the eigenvectors. For all graphs, the number of communities k was chosen based on two recent papers [17, 24], as outlined in

Table 3. If the signed graph does not have a known k in the literature, we assign a k value equal to the k value of the most similar graph in size. We implemented the *graphC* algorithm in C++ to automatically discover optimal communities without a predefined number of clusters k. The *graphC* finds and commits the optimal Harary cut for each connected component if it satisfies the following conditions: it is not in in the *processed* set, does not exceed the time limit t_l, its size is more significant than Γ, and the Harary cut leads to an improvement greater than ϵ. The code for running the leading ten methods in Table 2 and for *graphC* implementation is available at https://anonymous.4open.science/r/graphC-2046/.

The operating system used for the experiments is Linux Ubuntu 20.04.3. The CPU is an 11th-generation Intel(R) Core(TM) i9-11900K @ 3.50GHz with 16 physical cores. It has one socket, two threads per core, and eight cores per socket. The architecture is x86_x64. The GPU is an NVIDIA GeForce RTX 3070 with 8GB of memory. The driver version is 495.29.05, and the CUDA version is 11.5.

6.2 Signed Graph Datasets

All signed graphs are pre-processed to purge self-edges, inconsistent edges, and duplicate edges (keeping only the first) and treat neutral edges as positive. Table 3 describes the graphs derived as the Konect [27] benchmark as follows: *Highland* is the Konect signed social network of tribes of the GahukuGama alliance structure of the Eastern Central Highlands of New Guinea, from Kenneth Read [34]. *Congress* is a signed network where vertices are politicians speaking in the United States Congress, and a directed edge denotes that a speaker mentions another speaker. In the *Chess* network, each vertex is a chess player, and a directed edge represents a game with the white player having an outgoing edge and the black player having an incoming edge, where the weight of the edge represents the outcome. *BitcoinAlpha* is a user-user trust/distrust network from the Bitcoin Alpha platform for trading bitcoins, and *BitcoinOTC* is a user-user trust/distrust network from the Bitcoin OTC platform for trading Bitcoins. *WikiElec* is the network of users from the English Wikipedia that voted for and against each other in admin elections. The *SlashdotZoo* graph is the reply network of the technology website Slashdot, where the vertices are users, and the edges are replies. The edges of *WikiConflict* represent positive and negative conflicts between users of the English Wikipedia. emphWikiPolitics is an undirected signed network that contains interactions between the users of the English Wikipedia that have edited pages about politics. Each interaction, such as text editing and votes, is valued positively or negatively. *Epinions* is the trust and distrust network of Epinions, an online product rating site. It incorporates individual users connected by directed trust and distrust links. Table 3 lists the characteristics of the Konect signed graphs, as well as the following signed graph networks: (1) the *TwitterRef* captures data from Twitter concerning the 2016 Italian Referendum. Different stances between users signify a negative tie, while the same stances indicate a positive link [28]; (2) the *Sampson25* models the sentiment over time between novice monks in a New England

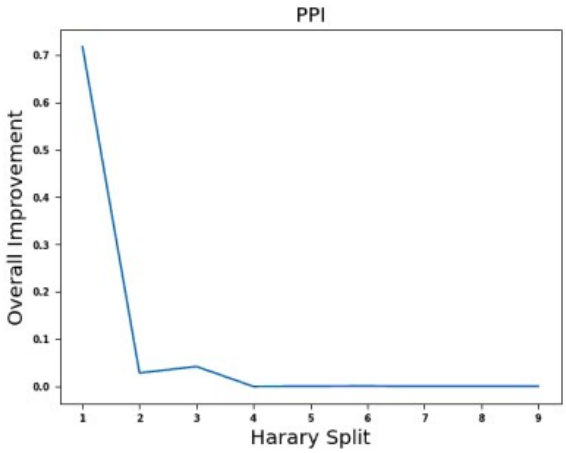

Fig. 5. Example output of the overall improvements with each **committed** Harary split of every connected component of our proposed algorithm on the signed PPI graph.

monastery captured by Sampson [37]; and (3) the *PPI* models the protein-protein interaction network [24]. *WikiRfa* describes voting information for electing Wikipedia managers [24].

Table 5 outlines the characteristics of the most significant connected components for the signed graphs derived from the Amazon ratings and reviews benchmark dataset. The Amazon dataset consists of *seventeen* signed graphs derived from the Amazon rating and review files [23]. The dataset contains Amazon's product reviews and metadata from May 1996 and July 2014. The rating score of 1 to 5 is mapped to an edge between the user and the product as follows $(5, 4) \rightarrow m^+$, $3 \rightarrow m$ (no sign), and $(2, 1) \rightarrow m^-$ [23]. Table 5 outlines the characteristics of the most significant connected component. For example, clustering a bipartite graph of users and items in a recommendation system context can reveal groups of users with similar preferences and groups of items that are popular among certain users. Another example is clustering a bipartite graph of authors and papers, which can identify research communities and topics.

7 Konect Signed Graphs Experiment

Table 3. Konect + benchmark: Konect [27] as well as TwitterReferendum [28], PPI [24], and WikiRfa [24] signed graphs. LCC stands for the largest connected component, k is the predefined number of clusters for the spectral clustering methods, and the *graphC* results list the number of clusters with at least five elements and the number of clusters with fewer than five elements. The last column is the number of splits produced after the algorithm's execution.

	LCC		k	GraphC		
Dataset	**# vertices**	**# edges**	**# clusters**	**# clusters≥5**	**# clusters<5**	**# splits**
Highland	16	58	3	1	2	2
Sampson25	25	165	4	2	2	3
Congress	219	521	11	2	9	3
PPI	3058	11l860	10	29	887	16
BitcoinAlpha	3l775	14,120	10	14	201	17
BitcoinOTC	5,875	21,489	20	22	518	10
Chess	7,115	55779	30	46	1,014	41
TwitterRef	10l864	251,396	50	4	234	8
SlashdotZoo	79l116	467l731	100	245	14,603	53
Epinions	119l130	704,267	100	584	27,498	237
WikiRfa	7,634	175787	30	25	1,419	24
WikiElec	7,066	100l667	30	38	1,528	25
WikiConflict	113,123	2,025,910	100	208	66,083	32
WikiPolitics	137,740	715,334	100	875	18,964	89

The *graphC* algorithm also counts the number of clusters (including isolated vertices) and the number of Harary split operations executed for various signed networks. We analyze and visualize the impact of increasing the number of subsequent Harary splits on the overall improvement in Equation 8 using the two evaluation metrics pos_{in} (Eq. 5) and neg_{out} (Eq. 6). Figure 6 shows the results of running our algorithm on the WikiElec signed graph. Initially, $pos_{in} = 1.0$ and $neg_{out} = 0.0$ as all edges belong to a single cluster. Our algorithm divides the connected component for each split to minimize the loss of pos_{within} and maximize the gain of $neg_{between}$. Figure 6 also shows that reducing pos_{within} with each split is inevitable, but the rate at which $neg_{between}$ increases is much higher than the rate at which pos_{within} decreases during the first few splits, and the overall measure of the quality of the clusters improves. Afterward, the performance starts to plateau, with diminishing gains as the algorithm makes deeper cuts into the connected components. The parameter Γ allows *graphC* to stop before reaching the plateau and saves computation time.

Table 4. Evaluation results of our proposed algorithm against ten different leading methods on 14 datasets in terms of $pos_{in}(G)$ and $neg_{out}(G)$, and time in seconds. Empty cells indicate the method failed to run on the dataset.

Method → Dataset ↓	*GraphC*			$\mathbf{A_{sym}}$			$\mathbf{L_{none}}$			$\mathbf{L_{sym}}$			$\mathbf{BNC_{none}}$			$\mathbf{BNC_{sym}}$		
	pos	**neg**	**t**	**pos**	**neg**	**t**	**pos**	**neg**	**t**	**pos**	**neg**	**t**	**pos**	**neg**	**t**	**pos**	**neg**	**t**
Highland	**93**	**100**	0.1	93	100	0.1	93	100	0.1	93	100	0.1	93	100	0.1	93	100	0.1
Sampson25	**70**	**90**	0.2	60	91	0.04	59	79	0.1	63	92	0.03	78	65	0.03	62	91	0.04
Congress	**93**	**100**	0.77	70	100	0.11	86	97	0.12	24	100	0.11	39	77	0.13	93	97	0.14
PPI	**81**	**98**	9	100	2	1	100	0.98	1	93	10	1	100	0.98	1			
BitcoinAlpha	**81**	**86**	14	100	0.5	2	30	68	3	72	24	3	55	66	2	100	3	2
BitcoinOTC	**82**	**90**	22	71	42	4	21	80	7	61	43	4	86	61	4	100	35	4
Chess	**39**	**84**	45	100	0.17	7							65	34	7	100	0.21	7
TwitterRef	**89**	**100**	446	65	90	31				67	94	31	23	93	30	100	0.66	31
WikiRfa	**61**	**81**	83	79	30	18	100	0.003	19	74	35	18	100	2	18	100	0.01	18
WikiElec	**70**	**79**	60	19	65	11	100	0.05	21	100	0.05	11	51	58	11	1	0.11	11

Method → Dataset ↓	*GraphC*			**Hessian**			$\mathbf{SPO_{none}}$			$\mathbf{SPO_{sym}}$			**dns**			**sns**		
	pos	**neg**	**t**	**pos**	**neg**	**t**	**pos**	**neg**	**t**	**pos**	**neg**	**t**	**pos**	**neg**	**t**	**pos**	**neg**	**t**
Highland	**93**	**100**	0.1	93	100	0.2	93	100	0.19	93	100	0.17	93	100	0.2	93	100	0.1
Sampson25	**70**	**90**	0.2	63	93	0.04	63	92	0.04	64	92	0.04	70	85	0.03	70	82	0.03
Congress	**93**	**100**	0.77	62	100	0.11	70	84	0.14	67	100	0.12	94	95	0.12	98	33	0.12
PPI	**81**	**98**	9	79	35	1	100	0.1	1	100	1	1	100	0.72	1	100	0.77	1
BitcoinAlpha	**81**	**86**	14	37	89	2	91	3	3	100	5	3	100	4	2	100	7	2
BitcoinOTC	**82**	**90**	22	30	94	4	62	30	8	100	28	4	100	8	4	100	8	4
Chess	**39**	**84**	45	40	70	7	100	0	12	100	0.51	8	100	0.2	7	100	0.2	7
TwitterRef	**89**	**100**	446	11	98	31	97	4	60	65	94	31	100	0.6	30	100	0.55	31
WikiRfa	**61**	**81**	83	54	59	18	9	11	2	91	6	19	100	0.02	18	100	0.01	18
WikiElec	**70**	**79**	60	19	92	11	92	12	20	1	0.74	11	1	0.17	11	1	0.16	11

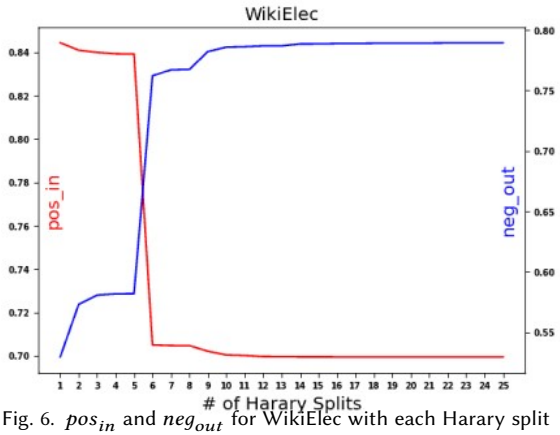

Fig. 6. pos_{in} and neg_{out} for WikiElec with each Harary split during algorithm execution

The methods with predefined k parameters force isolated vertices into arbitrary clusters to satisfy the k-value constraint [18]. In contrast, *graphC* allows for the isolated vertices to be automatically detected as long as the improvement in Equation 8 exceeds the threshold ϵ that specified that a vertex may split from its connected component if the overall improvement increases by ϵ value. The *graphC* algorithm finds a clustering that consistently achieves a good compromise between $pos_{in}(G)$ and $neg_{out}(G)$ (pos and neg in this table are percentages) on the Konect dataset with **both** having high values unlike other methods where they fail or solely focus on either evaluation metric. Table 3 summarizes the number of clusters and the number of Harary split operations performed on the Konect signed graphs, where the best results are underlined.

Next, we evaluate the performance and execution time of our algorithm against ten leading methods from the literature, which is also shown in Table 4. Empty cells in the table indicate two possible issues: either the state-of-the-art

algorithm took more than two days to run or failed due to non-convergence of the eigenvectors during the spectral decomposition, particularly for large signed graphs. In all tested signed graphs, regardless of size or density, our algorithm outperforms the state-of-the-art in terms of pos_{in} and neg_{out}. The execution time of our algorithm is longer than some other methods on certain graphs (e.g., WikiElec) because the k value in these methods is relatively small, leading to faster execution times. However, the state-of-the-art methods suffer significant performance degradation as the graph size and k-value increase while our algorithm maintains a more consistent execution time.

8 Amazon Ratings and Reviews Modeling

We map the Amazon ratings and reviews [23] to signed graphs with negative edges for ratings of 0, 1, and 2, no sign if the rating is 3, and a positive sign for ratings of 4 and 5. We demonstrate that our algorithm can scale to a signed network with millions of vertices and edges by running it on the Amazon signed graphs listed in Table 5. For instance, *graphC* successfully runs the largest Amazon graph (Books) in 31 hours while yielding high pos_{in} and neg_{out} percentages. As Table 5 shows, the bipartite graph of Amazon reviews/ratings has a skewed distribution of vertices and edges, where most of the vertices are sparsely connected, and only a few vertices dominate the market. For instance, in the Books signed graph, about 1 million clusters contain less than five users/books, and only 32 thousand clusters have more than five users/books. Although these clusters contain a mix of users and items, this may correlate with the "Long Tail Phenomenon" [4], where a large number of products or services that are not very popular can jointly have a significant share of the market that may overtake or be comparable to the current bestsellers.

Table 5. Amazon ratings and reviews [23] mapped to signed graphs. The *graphC* results include $pos_{in}(G)$ and $neg_{out}(G)$ (as percentages) and the resulting number of clusters with at least five elements as well as the number of clusters with less than five elements.

Amazon	**LCC**		**GraphC**					
Ratings	**# vertices**	**# edges**	**pos**	**neg**	**# splits**	**time (s)**	**# clusters≥5**	**# clusters<5**
Books	9,973,735	22,268,630	73	86	22	115,561	31,988	1,003,734
Electronics	4,523,296	7,734,582	88	80	7	23,516	9,208	862,970
Jewelry	3,796,967	5,484,633	81	90	11	25,650	15,802	648,695
TV	2,236,744	4,573,784	74	87	17	19,019	4,456	281,653
Vinyl	1,959,693	3,684,143	74	87	16	16,420	5,718	169,493
Outdoors	2,147,848	3,075,419	92	80	15	13,613	12,294	393,579
AndrApp	1,373,018	2,631,009	77	89	24	1,642	1,462	205,336
Games	1,489,764	2,142,593	82	22	1,662	280,356	8,111	272,245
Automoto	950,831	1,239,450	94	79	13	783	8,515	202,749
Garden	735,815	939,679	93	85	13	436	4,636	153,452
Baby	559,040	892,231	79	91	14	489	2,367	94,474
Music	525,522	702,584	87	83	23	571	6,124	109,287
Video	433,702	572,834	85	93	14	364	1,401	46,481
Instruments	355,507	457,140	90	85	15	281	3,536	61,964
Reviews	**# vertices**	**# edges**	**pos**	**neg**	**# splits**	**time (s)**	**clusters≥5**	**# clusters<5**
Core Music	9,109	64,706	52	84	14	9.57	38	746
Core Video	6,815	37,126	57	85	13	5.54	26	737
Core Instrum	2,329	10,261	51	92	8	1.90	20	159

8.1 Parameters Study & Analysis

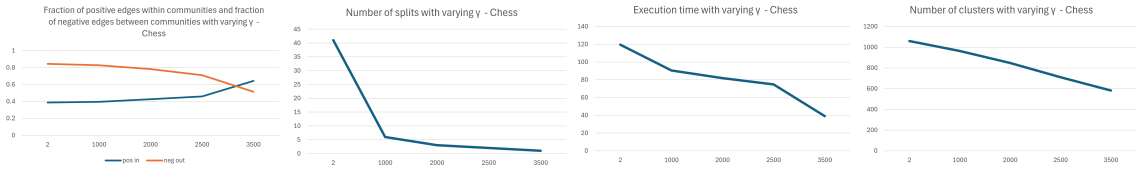

Fig. 7. The effect of varying Γ on the number of clusters, clustering quality, execution time, and Harary splits on the Chess signed graph. The y-axis is pos_{in} and neg_{out} as in equations 5 and 6, respectively, the number of splits, the execution time, and the number of communities for the four panels (from left to right). The x-axis is the Γ value. The *graphC* algorithm ignores connected components of size less than Γ.

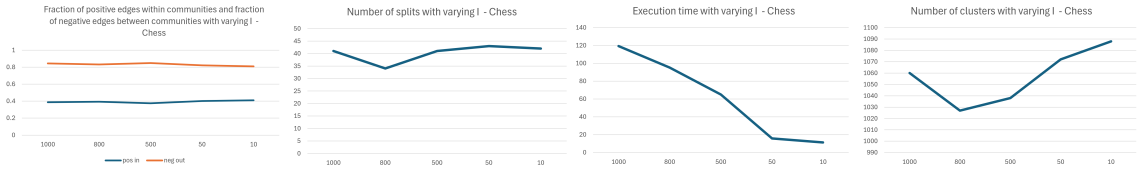

Fig. 8. The effect of varying I (number of iterations per connected component) on the number of clusters, clustering quality, execution time, and Harary splits on the Chess signed graph. The y-axis is pos_{in} and neg_{out} as in Equations 5 and 6, respectively, and the number of splits, the execution time, and the number of communities for the four panels (from left to right). The x-axis is the I value.

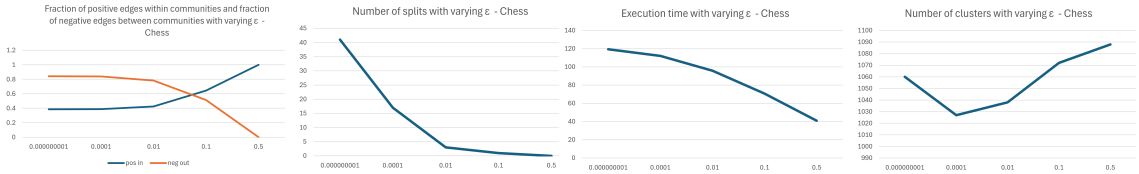

Fig. 9. The threshold ϵ specifies the minimum improvement value for the set split. This image summarizes the effect of varying ϵ on the number of clusters, clustering quality, execution time, and Harary splits on the Chess signed graph. From left to right, the y-axis is pos_{in} and neg_{out} as in Equations 5 and 6, respectively.

To better understand the parameters I, Γ, and ϵ and their effect on the clustering process, we run *graphC* on the Chess signed graph. First, we start by running the algorithm with $Gamma \in \{2, 1000, 2000, 2500, 3500\}$ while fixing the other parameters at $I = 50$, $\alpha = 0.5$, $\beta = 1$, $\epsilon = 0.00000001$, and t_l = -1. As the four panels of Figure 7 show, increasing Γ deteriorates the clustering quality while the number of splits and the execution time decrease, resulting in a lower number of communities. A higher value of Γ means that the algorithm will disregard even larger connected components that have the potential to improve the overall quality of the clustering if split. In return, we save computational time because the algorithm does not search for the best split for a connected component of size smaller than Γ.

Second, we run the algorithm with $I \in \{1000, 800, 500, 50, 10\}$ with the other parameters fixed at $Gamma = 2$, $\alpha = 0.5$, $\beta = 1$, $\epsilon = 0.00000001$, and t_l = -1. According to the four panels of Figure 8, decreasing the number of iterations when searching for the best Harary split for a connected component gradually degrades the overall quality of the clustering,

reducing the execution time tremendously. In contrast, the number of splits and clusters seems to stay the same or fluctuate around 41 splits and 1060 clusters. Finally, we run our algorithm with $\epsilon \in \{0.000000001, 0.0001, 0.01, 0.1, 0.5\}$ with the other parameters fixed at $Gamma = 2$, $\alpha = 0.5$, $\beta = 1$, $I = 1000$, and t_l = -1. The threshold ϵ specifies minimal overall improvement for the set split in the *graphC* algorithm. Thus, increasing ϵ reduces the probability of the split, as there is less chance improvement will be higher than ϵ, as illustrated in the four panels of Figure 9. Thus, higher values of ϵ degrade the quality of the clustering process and further reduce the execution time of the algorithm. The number of clusters in this ablations study stays about the same (1060) for different numbers of iterations I. In summary, high I and low Γ and ϵ values produce the most optimal clustering for the given resources regardless of the signed graph's type, density, or size.

9 Conclusion

The structure of communities within a network can reveal hidden patterns and insights about the relationships and dynamics in complex systems. Community detection is widely used in various fields, from biology to social networks, to extract such information. Recent research has pointed out challenges in recovering community structures in sparse networks, especially when using spectral methods. Current issues with signed-graph clustering algorithms include selecting the right number of clusters k, eigenvalue contamination, lack of scalability to large graphs, and dependence on ground truth for clustering validation. In response to these issues, we propose *graphC*, a new, scalable, hierarchical clustering algorithm for signed networks that automatically detects the optimal number of clusters without needing a predefined k. Our approach is flexible and supports optional parameters to control outcomes such as the number of isolated vertices detected.

Additionally, we introduce the parameter Γ to balance efficiency and scalability. We compare our algorithm to the leading methods from the literature using the pos_{in} and neg_{out} metrics. The results show that our algorithm consistently performs well on all tested datasets, achieving higher values for both metrics than the state-of-the-art methods, which struggle with scaling and producing high-quality cluster assignments. Future work is to parallelize our algorithm to speed up its execution and to integrate consensus features proposed in previous work [35] to enhance clustering quality.

References

[1] Robert P. Abelson and Milton J. Rosenberg. 1958. Symbolic psycho-logic: A model of attitudinal cognition. *Behavioral Science* 3, 1 (1958), 1–13.

[2] Alauddin Yousif Al-Omary and Mohammad Shahid Jamil. 2006. A new approach of clustering based machine-learning algorithm. *Knowledge-Based Systems* 19, 4 (2006), 248–258. https://doi.org/10.1016/j.knosys.2005.10.011

[3] Victor Amelkin and Ambuj K. Singh. 2019. Fighting Opinion Control in Social Networks via Link Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (Anchorage, AK, USA) *(KDD '19)*. Association for Computing Machinery, New York, NY, USA, 677–685. https://doi.org/10.1145/3292500.3330960

[4] Chris Anderson. 2006. *The Long Tail: Why the Future of Business Is Selling Less of More.* Hyperion.

[5] Lyonell Boulton. 2016. Spectral pollution and eigenvalue bounds. *Applied Numerical Mathematics* 99 (2016), 1–23. https://doi.org/10.1016/j.apnum.2015.08.007

[6] Dorwin Cartwright and Frank Harary. 1956. Structural balance: a generalization of Heider's theory. *Psychological Rev.* 63 (1956), 277–293.

[7] G. Chen, V. Liu, E. Robinson, L. J. Rusnak, and K. Wang. 2018. A Characterization of Oriented Hypergraphic Laplacian and Adjacency Matrix Coefficients. *Linear Algebra Appl.* 556 (2018), 323 – 341. https://doi.org/10.1016/j.laa.2018.07.012 arXiv:1704.03599 [math.CO]

[8] Yiqi Chen, Tieyun Qian, Huan Liu, and Ke Sun. 2018. "Bridge": Enhanced Signed Directed Network Embedding. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (Torino, Italy) *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 773–782. https://doi.org/10.1145/3269206.3271738

[9] Kai-Yang Chiang, Joyce Jiyoung Whang, and Inderjit S. Dhillon. 2012. Scalable clustering of signed networks using balance normalized cut. *Proceedings of the 21st ACM international conference on Information and Knowledge Management* (2012), 615 – 624.

[10] M. Cucuringu, P. Davies, A. Glielmo, and H. Tyagi. 2019. *SigNet.*

[11] Mihai Cucuringu, Peter Davies, Aldo Glielmo, and Hemant Tyagi. 2019. SPONGE: A generalized eigenproblem for clustering signed networks. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 89)*, Kamalika Chaudhuri and Masashi Sugiyama (Eds.). PMLR, Naha, Okinawa, Japan, 1088–1098. https://proceedings.mlr.press/v89/cucuringu19a.html

[12] Mihai Cucuringu, Apoorv Vikram Singh, DÃ©borah Sulem, and Hemant Tyagi. 2021. Regularized spectral methods for clustering signed networks. *Journal of Machine Learning Research* 22, 264 (2021), 1–79. http://jmlr.org/papers/v22/20-1289.html

[13] Tyler Derr, Yao Ma, and Jiliang Tang. 2018. Signed Graph Convolutional Network. *CoRR* abs/1808.06354 (2018). arXiv:1808.06354 http://arxiv.org/abs/1808.06354

[14] Tyler Derr, Zhiwei Wang, Jamell Dacon, and Jiliang Tang. 2020. Link and interaction polarity predictions in signed networks. *Social Network Analysis and Mining* 10, 1 (2020), 1–14.

[15] W. E. Donath and A. J. Hoffman. 1973. Lower Bounds for the Partitioning of Graphs. *IBM Journal of Research and Development* 17, 5 (1973), 420–425. https://doi.org/10.1147/rd.175.0420

[16] *Ghadeer Alabandi*, **Jelena Tešić**, Lucas Rusnak, and Martin Burtscher. 2021. Discovering and Balancing Fundamental Cycles in Large Signed Graphs. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* (St. Louis, Missouri) *(SC '21)*. Association for Computing Machinery, New York, NY, USA, Article 68, 17 pages. https://doi.org/10.1145/3458817.3476153

[17] *Maria Tomasso*, Lucas Rusnak, and **Jelena Tešić**. 2022. Advances in Scaling Community Discovery Methods for Signed Graph Networks. url = https://doi.org/10.1093/comnet/cnac013. *Oxford Journal of Complex Networks* 10, 3 (06 2022). https://doi.org/10.1093/comnet/cnac013

[18] *Maria Tomasso*, Lucas Rusnak, and **Jelena Tešić**. 2022. Cluster Boosting and Data Discovery in Social Networks. In *Proceedings of the 37th ACM/SIGAPP Symposium On Applied Computing (SAC)*. Association for Computing Machinery.

[19] Martin O'Reilly et al. 2018. Initial release of the SigNet package. https://doi.org/10.5281/zenodo.1435036

[20] Kiran Garimella, Tim Smith, Rebecca Weiss, and Robert West. 2021. Political polarization in online news consumption. In *Proceedings of the International AAAI Conference on Web and Social Media*, Vol. 15. AAAI Press, 152–162.

[21] Maryam Gholami, Amir Sheikhahmadi, Keyhan Khamforoosh, Mahdi Jalili, and Farshid Veisi. 2024. A new approach to finding overlapping community structure in signed networks based on Neutrosophic theory. *Journal of Complex Networks* 12, 1 (01 2024), cnad051. https://doi.org/10.1093/comnet/cnad051 arXiv:https://academic.oup.com/comnet/article-pdf/12/1/cnad051/55452104/cnad051.pdf

[22] Frank Harary and Dorwin Cartwright. 1968. On the Coloring of Signed Graphs. *Elemente der Mathematik* 23 (1968), 85–89. http://eudml.org/doc/140892

[23] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on WWW*. 507–517.

[24] Yixuan He, Gesine Reinert, Songchao Wang, and Mihai Cucuringu. 2022. SSSNET: Semi-Supervised Signed Network Clustering. arXiv:2110.06623 [cs.SI] https://arxiv.org/abs/2110.06623

[25] Ruben Interian, Ruslan G Marzo, Isela Mendoza, and Celso C Ribeiro. 2022. Network polarization, filter bubbles, and echo chambers: An annotated review of measures, models, and case studies. *arXiv preprint arXiv:2207.13799* (2022).

[26] Andrew V Knyazev. 2017. Signed Laplacian for spectral clustering revisited. *arXiv preprint arXiv:1701.01394* 1 (2017).

[27] Jérôme Kunegis. 2013. Konect: the koblenz network collection. In *Proceedings of the 22nd international conference on world wide web*. 1343–1350.

[28] Mirko Lai, Viviana Patti, Giancarlo Ruffo, and Paolo Rosso. 2018. Stance Evolution and Twitter Interactions in an Italian Political Debate. In *Natural Language Processing and Information Systems*, Max Silberztein, Faten Atigui, Elena Kornyshova, Elisabeth Métais, and Farid Meziane (Eds.). Springer International Publishing, Cham, 15–27.

[29] Yeon-Chang Lee, Nayoun Seo, Kyungsik Han, and Sang-Wook Kim. 2020. ASiNE: Adversarial Signed Network Embedding. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval* (Virtual Event, China) *(SIGIR '20)*. Association for Computing Machinery, New York, NY, USA, 609–618. https://doi.org/10.1145/3397271.3401079

[30] Ann Marsden. 2013. EIGENVALUES OF THE LAPLACIAN AND THEIR RELATIONSHIP TO THE CONNECTEDNESS. https://api.semanticscholar.org/CorpusID:17239810

[31] Luca Martino, Roberto San Millán-Castillo, and Eduardo Morgado. 2023. Spectral information criterion for automatic elbow detection. *Expert Systems with Applications* 231 (2023), 120705. https://doi.org/10.1016/j.eswa.2023.120705

[32] Pedro Mercado, Francesco Tudisco, and Matthias Hein. 2019. Spectral Clustering of Signed Graphs via Matrix Power Means. *CoRR* abs/1905.06230 (2019). arXiv:1905.06230 http://arxiv.org/abs/1905.06230

[33] Katherine S. Pollard and Mark J. van der Laan. 2002. Statistical inference for simultaneous clustering of gene expression data. *Mathematical Biosciences* 176, 1 (2002), 99–121. https://doi.org/10.1016/S0025-5564(01)00116-X

[34] Kenneth Read. 1954. Cultures of the Central Highlands, New Guinea. *Southwestern Journal of Anthropology* 10, 1 (1954), 1–43.

[35] Lucas Rusnak and **Jelena Tešić**. 2021. Characterizing Attitudinal Network Graphs through Frustration Cloud. *Data Mining and Knowledge Discovery* 6 (November 2021). https://doi.org/10.1007/s10618-021-00795-z

[36] Alaa Saade, Florent Krzakala, and Lenka Zdeborová. 2014. Spectral Clustering of graphs with the Bethe Hessian. *ArXiv* abs/1406.1880 (2014). https://api.semanticscholar.org/CorpusID:919316

[37] S. Sampson. 1968. A novitiate in a period of change: An experimental and case study of relationships. Ph.D. thesis, Cornell University.

[38] Changwon Seo, Kyeong-Joong Jeong, Sungsu Lim, and Won-Yong Shin. 2021. SiReN: Sign-Aware Recommendation Using Graph Neural Networks. *CoRR* abs/2108.08735 (2021). arXiv:2108.08735 https://arxiv.org/abs/2108.08735

[39] Vincent A. Traag and Lovro Šubelj. 2023. Large network community detection by fast label propagation. *Scientific Reports* 13, 1 (2023).

[40] Xiaolu Wang, Peng Wang, and Anthony Man-Cho So. 2022. Exact Community Recovery over Signed Graphs. In *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research, Vol. 151)*, Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera (Eds.). PMLR, Virtual, 9686–9710. https://proceedings.mlr.press/v151/wang22i.html

[41] Guangxia Xu, Xinkai Wu, Jun Liu, and Yanbing Liu. 2020. A Community Detection Method Based on Local Optimization in Social Networks. *IEEE Network* 34, 4 (2020), 42–48. https://doi.org/10.1109/MNET.011.1900472

[42] Q. Zheng and D.B. Skillicorn. 2015. Spectral Embedding of Signed Networks. In *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM)*. SIAM, Vancouver, British Columbia, Canada, 55–63. https://doi.org/10.1137/1.9781611974010.7 arXiv:https://epubs.siam.org/doi/pdf/10.1137/1.9781611974010.7

Received 21 November 2024