Efficient Maximal Balanced Clique Enumeration in Signed Networks

Zi Chen
East China Normal University
zchen@stu.ecnu.edu.cn

Long Yuan*
Nanjing University of Science and Technology
longyuan@njust.edu.cn

Xuemin Lin
University of New South Wales
lxue@cse.unsw.edu.au

Lu Qin
University of Technology, Sydney
lu.qin@uts.edu.au

Jianye Yang
Hunan University
jyyang@hnu.edu.cn

ABSTRACT

Clique is one of the most fundamental models for cohesive subgraph mining in network analysis. Existing clique model mainly focuses on unsigned networks. In real world, however, many applications are modeled as signed networks with positive and negative edges. As the signed networks hold their own properties different from the unsigned networks, the existing clique model is inapplicable for the signed networks. Motivated by this, we propose the balanced clique model that considers the most fundamental and dominant theory, structural balance theory, for signed networks, and study the maximal balanced clique enumeration problem which computes all the maximal balanced cliques in a given signed network. We show that the maximal balanced clique enumeration problem is NP-Hard. A straightforward solution for the maximal balanced clique enumeration problem is to treat the signed network as two unsigned networks and leverage the off-the-shelf techniques for unsigned networks. However, such a solution is inefficient for large signed networks. To address this problem, in this paper, we first propose a new maximal balanced clique enumeration algorithm by exploiting the unique properties of signed networks. Based on the new proposed algorithm, we devise two optimization strategies to further improve the efficiency of the enumeration. We conduct extensive experiments on large real and synthetic datasets. The experimental results demonstrate the efficiency, effectiveness and scalability of our proposed algorithms.

KEYWORDS

Maximal Balanced Clique, Signed Network, Graph Algorithm

ACM Reference Format:
Zi Chen, Long Yuan, Xuemin Lin, Lu Qin, and Jianye Yang. 2020. Efficient Maximal Balanced Clique Enumeration in Signed Networks. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3366423.3380119

*Zi Chen and Long Yuan contribute equally to this study. Long Yuan is the corresponding author.

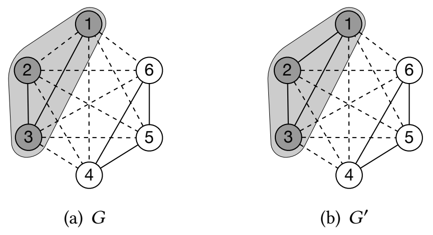


Figure 1: Imbalanced Graph and Balanced Graph

1 INTRODUCTION

With the proliferation of graph applications, research efforts have been devoted to many fundamental problems in analyzing graph data [15, 28, 36, 37, 39, 49, 53, 55]. Clique is one of the most fundamental cohesive subgraph models in graph analysis, which requires each pair of vertices has an edge. Due to the completeness requirement, clique model owns many interesting cohesiveness properties, such as the distance of any two vertices in a clique is one, every one vertex in a clique forms a dominate set of the clique and the diameter of a clique is one [38]. As a result, clique model has wide application scenarios in social network mining, financial analysis and computational biology and has been extensively investigated for decades. Existing studies on clique mainly focus on the unsigned networks, i.e., all the edges in the graph share the same property [4, 12, 13, 51]. Unfortunately, relationships between two entities in many real-world applications have completely opposite properties, such as friend-foe relationships between users in social networks [11, 23], support-dissent opinions in opinion networks [25], trust-distrust relationships in trust networks [26] and partnership-antagonism in protein-protein interaction networks [35]. Modelling these applications as signed networks with positive and negative edges allows them to capture more sophisticated semantics than unsigned networks [1, 5, 10, 26, 32, 33]. Consequently, existing studies on clique ignoring the sign associated with each edge may be inappropriate to characterize the cohesive subgraphs in a signed network and there is an urgent need to define an exclusive clique model tailored for the signed networks.

For the signed networks, the most fundamental and dominant theory revealing the dynamics and construction of the signed networks is the *structural balance theory* [1, 5, 10, 11, 18, 19, 26, 32, 33]. The intuition underlying the structural balance theory can be described as the aphorisms: "The friend (resp. enemy) of my friend (resp. enemy) is my friend, the friend (resp. enemy) of my enemy (resp. friend) is my enemy". Specifically, a signed network G is structural balanced if G can be split into two subgraphs such that the edges in the same subgraph are positive and the edges between subgraphs are negative [18]. In a signed network, an imbalanced sub-structure is unstable and tends to evolve into a balanced state. Consider the graph G shown in Figure 1 (a). The negative edge between v_1 and v_2 makes G imbalanced. Closely observing G, we can find that v_1 and v_2 have a mutual "friend" v_3 and mutual "enemies" v_4, v_5 and v_6. It means v_1 and v_2 share more common grounds than differences. According to *structural balance theory*, v_1 and v_2 tend to be allies as time goes by. G' shown in Figure 1 (b) is the evolved balanced counterpart of G. In G', the sign of the edge between v_1 and v_2 becomes positive. $\{v_1, v_2, v_3\}$ and $\{v_4, v_5, v_6\}$ form two alliances and the edges in the same alliance are positive and the edges connecting different alliances are negative. As illustrated in this example, structural balance reflects the key characteristics of the signed networks.

According to the above analysis, clique model is a fundamental cohesive subgraph model in graph analysis and can be used in many applications, but there is no appropriate counterpart in the signed networks. Meanwhile, the structure of the signed networks is expected to be balanced based on the structure balance theory. Motivated by this, we propose a maximal balanced clique model in this paper. Formally, given a signed network G, a maximal balanced clique C is a maximal subgraph of G such that (1) C is complete, i.e., every pair of vertices in C has an edge. (2) C is balanced, i.e., C can be divided into two parts such that the edges in the same part are positive and the edges connecting two parts are negative. This definition not only catches the essence of the clique model in the unsigned networks but also guarantees that a detected clique is stable in the signed networks. In this paper, we aim to devise efficient algorithms to enumerate all maximal balanced cliques in a given signed network.

Applications. Maximal balanced clique enumeration can be used in many applications, for example:

(1) Opinion leaders detection in opinion networks. Opinion leaders are people who are active in a community capturing the most representative opinions in the social networks [44]. Maximal balanced clique enumeration can be used to detect opinion leaders in the opinion networks. In an opinion network, each vertex represents a user and there is a positive/negative edge between two vertices if one user support/dissent another user. A maximal balanced clique in an opinion network represents a group of users that any two of them have an opinion with each other and can be further divided into two subgroups such that the intra-group users support each other and the inter-group users dissent each other. Since these users actively involve in the opinion networks (every two of them have an opinion with each other) and have their clear standpoints (support everyone in the same group and dissent everyone in the opposite group), the users in the maximal balanced cliques are good candidates of opinion leaders in the opinion network.

(2) Finding international alliances-rivalries groups. The international relationships between nations can be modeled as a signed network, where each vertex represents a nation, positive and negative edges indicate alliances and rivalries, respectively. Computing the maximal balanced cliques in such networks reveals hostile groups of allied forces, such as the Allied and Axis power during World War II or the North Atlantic Treaty Organization and the Warsaw Pact during the Cold War [3, 11]. We can extend it to find the alliances-rivalries commercial groups among business organizations similarly, such as {Pepsi, KFC} vs {Coke, McDonald}[21].

(3) Synonym and antonym groups discovery. In a word network, each vertex represents a word and there is a positive edge between two synonyms and a negative edge between two antonyms[34]. In such signed networks, our model can discover synonym groups that are antonymous with each other, such as, {interior, internal, intimate} and {away, foreign, outer, outside, remote}. These discovered groups may be further used in applications such as automatic question generation [24] and semantic expansion [22].

Contributions. In this paper, we make the following contributions:

(1) The first work to study the maximal balanced clique model. We formalize the balanced clique model in signed networks based on the structural balance theory. To the best of our knowledge, this is the first work considering the structural balance of the cliques in signed networks. We also prove the NP-Hardness of the problem.

(2) A new framework tailored for maximal balanced clique enumeration in signed networks. After investigating the drawbacks of the straightforward approach, we propose a new framework for the maximal balanced clique enumeration. Our new framework enumerates the maximal balanced cliques based on the signed network directly and its memory consumption is linear to the size of the input signed network.

(3) Two effective optimization strategies to further improve the enumeration performance. We explore two optimization strategies, in-enumeration optimization and pre-enumeration optimization, to further improve the enumeration performance. The in-enumeration optimization can avoid the exploration for unpromising vertices during the enumeration while the pre-enumeration techniques can prune unpromising vertices and edges before enumeration.

(4) Extensive performance studies on real and synthetic datasets. We conduct extensive experimental studies to evaluate the proposed algorithms on real and synthetic datasets, one of which contains 3 million vertices and 105 million edges. As shown in our experiments, the baseline approach only works on small datasets while our approach can complete the enumeration efficiently on both small and large datasets.

Outline. Section 2 reviews the related work. Section 3 provides preliminaries including the definition of balanced clique model and problem statement. Section 4 introduces the baseline algorithm. Section 5 presents our new enumeration framework. Section 6 shows several optimization techniques. Section 7 reports the results of experimental studies. Section 8 concludes our paper.

2 RELATED WORK

Signed network analysis. Signed network analysis has attracted much attention in the literature. In these works, the theories explaining the potential social dynamics process in signed networks have been extensively studied. Among these theories, *structural* *balance theory* is the most fundamental and dominant one [58]. Structural balance theory is originally introduced in [19] and generalized in the graph formation in [5, 18]. After that, structural balance theory is developed extensively [1, 10, 26, 32, 33]. In these works, it is interesting to mention that the authors in [32] model the evolving procedure of a signed network and theoretically prove that the network would evolve into a balanced clique when the mean value of the initial friendliness among the vertices $\mu \leq 0$. [58] provides a comprehensive survey on structural balanced theory.

Besides theories on signed networks, a large body of literature on mining signed networks has been emerged. Among them, the most closely related work to ours is [27] in which an (α, k)-clique model is proposed. Given a signed network G, an (α, k)-clique is defined as a maximal clique C such that the negative degree for each vertex in C is not greater than k and the positive degree for each vertex in C is not less than αk. Compared with our model, (α, k)-clique model only considers the amount of positive and negative edges in the clique and the structural balance of the clique is totally ignored, which makes (α, k)-clique model essentially different from our model. In [17], a k-balanced trusted clique model is proposed. A k-balanced trusted clique is defined as a clique with k vertices consisting with positive edges only. Although the k-balanced trusted clique model has a similar name with our model, it ignores the negative edges in the clique, which means the information of the negative edges are totally missed.

Community detection in signed networks is also related to our work. For example, [8, 16, 29–31, 45] aim to find the antagonistic communities in a signed network. These works mainly focus on exploring several groups of dense subgraphs and most of them don't have a clear structural definition of their community model, while our work aims to enumerate the clique structure in a signed network. Moreover, these solutions generally involve a complicated optimization procedure, thereby, they are hard to handle large signed networks, while our proposed algorithm is scalable to enumerate all the maximal balanced cliques in large signed networks with hundreds of millions of edges as verified in our experiments. A survey on signed network mining can be found in [46].

Clique on unsigned networks. Clique model is one of the most fundamental cohesive subgraph models. [4] proposes an efficient algorithm for maximal clique enumeration based on backtracking search.[2] first considers the memory consumption during the maximal clique enumeration. Based on [4], more efficient algorithms for maximal clique enumeration are investigated [12, 13, 47].[12] proposes a novel branch pruning strategy, pivot pruning, which can efficiently reduce the search space by ignoring the search process from the neighbors of the pivot. [57] studies the maximal biclique enumeration problem on bipartite graphs.[57] keeps growing the vertex set in one side and peeling the vertex set in another side to enumerate the maximal biciques. It also utilizes some techniques to further improve the enumeration performance, such as choosing vertex with small degree from candidate set to reduce the search tree depth and pruning vertices which may produce non-maximal bicliques. These techniques for biclique enumeration inspire our techniques presented in Section 6.1. [14] reviews recently advances in maximal clique enumeration. Based on clique, other cohesive subgraph models are also studied recently, such as k-core [43], k-truss[9, 20], k-edge connected component[52, 54, 59], (r, s)-nuclei [40, 41]. Note that our balanced clique model is different from the existing cohesive subgraph models on unsigned networks and it cannot be well solved by the existing works. If we just consider the positive edge in the signed network and use the traditional methods on unsigned networks for community detection, the found results would ignore the negative edges and half meaningful information in the signed network is lost.

3 PROBLEM STATEMENT

In this paper, we consider an undirected and unweighted signed network $G = (V, E^+, E^-)$, where V denotes the set of vertices, E^+ denotes the positive edges and E^- denotes the negative edges connecting the vertices in G. We denote the number of vertices and number of edges by n and m, respectively, i.e., $n = |V|$ and $m = |E^+| + |E^-|$. For each vertex $v \in G$, let $N_G^+(v)$ represents the positive neighbors of v, i.e, $N_G^+(v) = \{u|(v, u) \in E^+, u, v \in V\}$, and let $N_G^-(v)$ represents the negative neighbors of v, i.e, $N_G^-(v) = \{u|(v, u) \in E^-, u, v \in V\}$. We use $d_G^+(v)$ and $d_G^-(v)$ to denote the positive and negative degree of v, respectively, i.e., $d_G^+(v) = |N_G^+(v)|$ and $d_G^-(v) = |N_G^-(v)|$. We also use $N_G(v)$ and $d_G(v)$ to denote the neighbors and degree of v, i.e., $N_G(v) = N_G^-(v) \cup N_G^+(v)$ and $d_G(v) = d_G^+(v) + d_G^-(v)$. For simplicity, we omit G in the notations if the context is self-evident.

Definition 3.1. **(Balanced Network [18])** Given a signed network $G = (V, E^+, E^-)$, it's balanced iff it can be split into two subgraphs G_L and G_R, s.t. $\forall(u, v) \in E^+ \rightarrow u, v \in G_L$ or $u, v \in G_R$, and $\forall(u, v) \in E^- \rightarrow u \in G_L, v \in G_R$ or $u \in G_R, v \in G_L$.

Definition 3.2. **(Maximal Balanced Clique)** Given a signed network $G = (V, E^+, E^-)$, a maximal balanced clique C is a maximal subgraph of G that satisfies the following constraints:

- Complete: C is complete, i.e, $\forall u, v \in C \rightarrow (u, v) \in E^+ \cup E^-$.
- Balanced: C is balanced, i.e, it can be split into two sub-cliques C_L and C_R, s.t. $\forall u, v \in C_L$ or $u, v \in C_R \rightarrow (u, v) \in E^+$, and $\forall u \in C_L, v \in C_R$ or $u \in C_R, v \in C_L \rightarrow (u, v) \in E^-$.

In this paper, we aim to enumerate all maximal balanced cliques in a given signed network. Since many real applications require that the number of vertices in C_L and C_R is not less than a fixed threshold, we add a size constraint on $|C_L|$ and $|C_R|$ s.t. $|C_L| \geq k$ and $|C_R| \geq k$. With the size constraint, users can control the size of the returned maximal balanced cliques based on their specific requirements. We formalize the studied problem as follows:

Problem Statement. Given a signed network G and an integer k, maximal balanced clique enumeration (MBCE) computes all the maximal balanced cliques C in G s.t. $|C_L| \geq k$ and $|C_R| \geq k$ for C.

Example 3.3. Consider the signed network G in Figure 2 in which positive/negative edges are denoted by solid/dashed lines. Assume $k = 2$, there are 4 maximal balanced cliques in G, namely, $C_1 = \{\{v_0, v_1, v_3\}, \{v_5, v_6, v_7\}\}$, $C_2 = \{\{v_0, v_1, v_2, v_3\}, \{v_5, v_6\}\}$, $C_3 = \{\{v_0, v_1\}, \{v_5, v_6, v_8\}\}$, $C_4 = \{\{v_0, v_{14}\}, \{v_{13}, v_{15}\}\}$, where vertices in C_L and C_R are marked with different colors. Take C_4 as an example, it is complete as any two vertices in C_4 have an edge. It is balanced as it can be split into $\{v_0, v_{14}\}$ and $\{v_{13}, v_{15}\}$, and two positive edges (v_0, v_{14}) and (v_{13}, v_{15}) exist in E^+. v_0 has negative

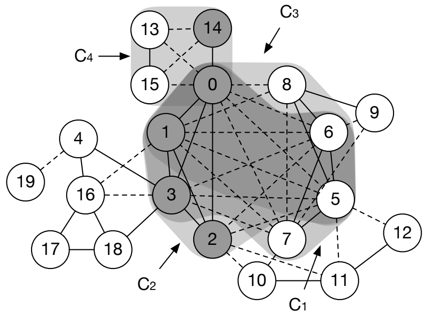


Figure 2: Maximal Balanced Clique in G ($k = 2$)

edges to v_{13}, v_{15} and similar negative edges exist for v_{13}, v_{14} and v_{15}. C_4 is maximal because no more vertices can be added into it to make it complete and balanced.

Problem Hardness. The MBCE problem is NP-Hard, which can be proved following the NP-Hardness of maximal clique enumeration problem [6, 42]. Given an unsigned network $G = (V, E)$, we can transfer G to a signed network G' as follows: we first keep all the vertices of G in G' and all the edges of G as positive edges in G'; then, we add a new vertex v to G' and connect v to all the remaining vertices in G' with negative edges. It's clear that each maximal clique C in G corresponds a maximal balanced clique $\{\{v\}, C\}$ in G' (assume $k = 1$), and vice versa, which means the maximal clique enumeration problem in G can be reduced to the MBCE problem in G'. As the maximal clique enumeration problem is NP-Hard [6, 42], our problem is also NP-Hard.

4 A BASELINE ALGORITHM

We first propose a baseline algorithm to address MBCE problem based on existing methods for maximal clique enumeration [13] and maximal biclique enumeration [57] in unsigned networks. For a signed network $G = (V, E^+, E^-)$, we can treat it as the combination of two unsigned networks $G^+ = (V, E^+)$ and $G^- = (V, E^-)$. For any maximal balanced clique $C = \{C_L, C_R\}$ in G, it is clear that C_L (resp. C_R) is a clique in G^+ and the subgraph induced by vertices in C_L and C_R in G^- is a biclique. Therefore, we can enumerate the maximal balanced cliques in G in two steps: 1) compute all the maximal cliques in G^+ with [13]; 2) for each pair of the computed maximal cliques C_i and C_j in G^+, compute the maximal bicliques in the bipartite subgraph induced by the vertices in C_i and C_j in G^- with [57]. The returned maximal bicliques in G^- are the maximal balanced cliques in G. The pseudocode of Baseline solution is shown in Algorithm 1. As the pseudocode is self-explained, we omit the description. Note that although all the maximal cliques in G^+ are enumerated in line 1 of Algorithm 1, Algorithm 1 does not require that the two component cliques of a maximal balanced clique are maximal in G^+. Algorithm 1 just considers all maximal cliques as candidate subgraphs for further processing in step 2.

Example 4.1. Consider G in Figure 2, assume $k = 2$, Baseline first enumerates all the maximal cliques in G^+ with size not less than 2, such as $\{v_0, v_1, v_2, v_3\}$, $\{v_5, v_6, v_7\}$, $\{v_5, v_6, v_8\}$, $\{v_0, v_{14}\}$, $\{v_{13}, v_{15}\}$. After that, for each pair of computed maximal cliques, computes the maximal bicliques in the induced bipartite subgraph in G^-.

Algorithm 1 Baseline($G = (V, E^+, E^-)$, k)

```
1: enumerate the maximal cliques in $G^+ = (V, E^+)$ with size not less than $k$ by [13];
2: for each pair of computed maximal cliques $C_i$ and $C_j$ do
3:     enumerate the maximal bicliques in the bipartite subgraph induced by $C_i$ and $C_j$ in $G^- = (V, E^-)$ with size not less than $k$ for both two parts by [57];
4: remove the duplicate bicliques computed in line 3;
```

Take $\{v_0, v_1, v_2, v_3\}$ and $\{v_5, v_6, v_7\}$ as an example, the maximal biclique in the induced bipartite subgraph in G^- are $\{\{v_0, v_1, v_3\}, \{v_5, v_6, v_7\}\}$ and $\{\{v_0, v_1, v_2, v_3\}, \{v_5, v_6\}\}$ which correspond C_1 and C_2 in G, respectively. The remaining maximal balanced cliques can be enumerated similarly.

Theorem 4.2. *Given a signed network G,* Baseline *enumerates all the maximal balanced cliques in G correctly.*

Proof. We first prove that all the maximal balanced cliques in G are found. Based on Definition 3.2, if there exists a maximal balanced clique $C = \{C_L, C_R\}$ in G, the vertices in C_L (resp. C_R) must be contained in a maximal cliques C'_L (resp. C'_R) in G^+. As Baseline considers all the maximal cliques in G^+, C_L and C_R are not missed in step 1. Following Definition 3.2, C_L and C_R form a maximal biclique in G^-. In step 2, Baseline enumerates all the maximal bicliques in the induced subgraph in G^- by every pair of enumerated maximal cliques in step 1. Thus, Baseline can find all the maximal balanced cliques in G. Moreover, as a maximal balanced clique maybe contained in multiple pairs of maximal cliques, Baseline removes all the duplicates in line 4. Therefore, Baseline outputs each maximal balanced clique once. The theorem is proved. □

Drawbacks of baseline. Since Baseline does not consider the uniqueness of the signed networks and processes MBCE with the techniques for the unsigned networks, it has two drawbacks:

- Memory consumption. Baseline has to store all the maximal cliques in G^+ in memory. The number of maximal cliques could be exponential to the number of vertices [12], which makes Baseline unable to handle large networks.
- Efficiency. In baseline, all the maximal cliques in G^+ are enumerated and every pair of maximal cliques are explored. The time complexity of Baseline is $O(T_{\mathrm{Cli}} + \eta^2 \cdot T_{\mathrm{BiCli}})$, where $T_{\mathrm{BiCli}}/T_{\mathrm{Cli}}$ represent the time complexity of maximal (bi)clique enumeration, and η is the number of enumerated maximal cliques in G^+. Considering the maximal (bi)clique enumeration is time-consuming and the number of maximal cliques could be very large, it is inefficient for MBCE problem.

5 A NEW ENUMERATION FRAMEWORK

Revisiting baseline, the root leading to its drawbacks discussed above is that it treats the signed network as a specific combination of two unsigned networks and utilizes the existing techniques designed for the unsigned networks. Therefore, we have to explore new techniques by considering the uniqueness of signed networks to overcome the drawbacks of Baseline and improve the efficiency

Algorithm 2 MBCEnum($G = (V, E^+, E^-), k$)

```
1:  Flag ← true;
2:  for each $v_i \in \{v_0, v_1, \cdots, v_{n-1}\} \in V$ do
3:      $C_L \leftarrow \{v_i\}, C_R \leftarrow \emptyset$
4:      $P_L \leftarrow N^+_G(v_i) \cap \{v_{i+1}, \cdots, v_{n-1}\}$;
5:      $P_R \leftarrow N^-_G(v_i) \cap \{v_{i+1}, \cdots, v_{n-1}\}$;
6:      $Q_L \leftarrow N^+_G(v_i) \cap \{v_0, \cdots, v_{i-1}\}$;
7:      $Q_R \leftarrow N^-_G(v_i) \cap \{v_0, \cdots, v_{i-1}\}$;
8:      MBCEnumUtil($C_L, C_R, P_L, P_R, Q_L, Q_R$);
9:  Procedure MBCEnumUtil($C_L, C_R, P_L, P_R, Q_L, Q_R$)
10: if $P_L = \emptyset$ and $P_R = \emptyset$ and $Q_L = \emptyset$ and $Q_R = \emptyset$ then
11:     if $|C_L| \geq k$ and $|C_R| \geq k$ then
12:         output $C = \{C_L, C_R\}$;
13:     return
14: Flag ←!Flag;
15: if Flag then
16:     for each $v \in P_L$ do
17:         MBCEnumUtil($C_L \cup \{v\}, C_R, N^+_G(v) \cap P_L, N^-_G(v) \cap P_R, N^+_G(v) \cap Q_L, N^-_G(v) \cap Q_R$);
18:         $P_L \leftarrow P_L \setminus \{v\}$; $Q_L \leftarrow Q_L \cup \{v\}$;
19:     for each $v \in P_R$ do
20:         MBCEnumUtil($C_L, C_R \cup \{v\}, N^-_G(v) \cap P_L, N^+_G(v) \cap P_R, N^-_G(v) \cap Q_L, N^+_G(v) \cap Q_R$);
21:         $P_R \leftarrow P_R \setminus \{v\}$; $Q_R \leftarrow Q_R \cup \{v\}$;
22: else
23:     line 19-21; line 16-18;
```

of the enumeration. In this section, we present a new enumeration framework which aims to address the memory consumption problem. In next section, we further optimize the enumeration framework to improve the efficiency.

Lemma 5.1. *Given a signed network G, for a balanced clique $C = \{C_L, C_R\}$ in G, if there is a vertex v in G such that $\forall u \in C_L \rightarrow (v, u) \in E^+$ and $\forall w \in C_R \rightarrow (v, w) \in E^-$, then $C' = \{C_L \cup \{v\}, C_R\}$ is also a balanced clique in G.*

Proof. It can be proved following Definition 3.2 directly. □

According to Lemma 5.1, if we maintain a balanced clique $C = \{C_L, C_R\}$, let P_L be the set of vertices that are positive neighbors of all the vertices in C_L and negative neighbors of all the vertices in C_R, let P_R be the set of vertices that are positive neighbors of all the vertices in C_R and negative neighbors of all the vertices in C_L, we can enlarge C by adding vertices from P_L and P_R into C_L and C_R, respectively. Furthermore, if we update the P_L and P_R based on the new C_L and C_R accordingly and repeat the above enlargement procedure, we can obtain a maximal balanced clique when no more vertices can be added into C_L or C_R.

Algorithm. Following the above idea, our algorithm for MBCE is shown in Algorithm 2. For each vertex v_i in G (line 2), we enumerate all the maximal balanced cliques containing v_i (line 3-8). Note that $v_0, v_1, \ldots, v_n$ are in the degeneracy order [48] of G. We use C_L and C_R to maintain the balanced clique, which are initialized with v_i and $\emptyset$, respectively (line 3). Similarly, we also initialize P_L and P_R as discussed above (line 4-5). Moreover, we use Q_L and Q_R to record the vertices that have been processed to avoid outputting duplicate maximal balanced cliques (line 6-7). After initializing these six sets, we invoke procedure MBCEnumUtil to enumerate all the maximal balanced cliques containing v_i (line 8).

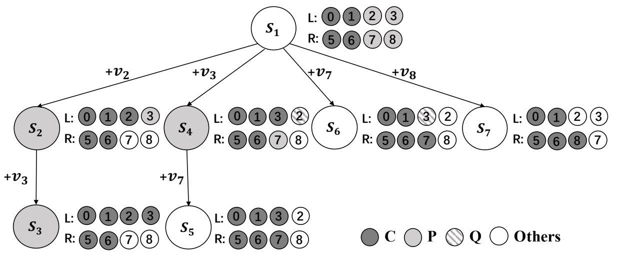


Figure 3: Search Tree for MBCEnum

Procedure MBCEnumUtil performs the maximal balanced clique enumeration based on the given six sets. If P_L, P_R, Q_L and Q_R are empty, which means current balanced clique $C = \{C_L, C_R\}$ cannot be enlarged and it is a maximal balanced clique, MBCEnumUtil checks whether C_L and C_R satisfy the size constraint. If the size constraint is satisfied, it outputs the maximal balanced clique C (line 11-12). Otherwise, MBCEnumUtil adds a vertex from P_L to C_L, updates the corresponding P_L, P_R, Q_L and Q_R, and recursively invokes itself to further enlarge the balanced clique (line 17). When $v \in P_L$ is processed, v is removed from P_L and added in Q_L (line 18). Similar processing steps are applied on vertices in P_R (line 19-21). Variable Flag (line 1) is used to control the order of adding new vertex into C_L or C_R. With the switch operation in line 14, we can guarantee that we add vertex into C_L, then into C_R, recursively.

Correctness of Algorithm 2. We show the correctness of Algorithm 2 from three aspects: (1) the balanced clique outputted in line 12 is maximal. Assume that a balanced clique C outputted in line 12 is not maximal, then based on the vertices maintained in P_L and P_R regarding C, at lease P_L or P_R is not empty, which contradicts with the outputting condition in line 10. Therefore, the balanced clique outputted in line 12 is maximal. A special case that needs to note is the balanced clique exploration caused by the initialization of P_L and P_R. For a vertex v_i, its positive (negative) neighbors in $v_0, \cdots, v_{i-1}$ are not added into P_L (P_R). As a result, for a maximal balanced clique C containing v_i and other vertices in $v_0, \cdots, v_{i-1}$, due to the initialization of P_L and P_R, the vertices in $v_0, \cdots, v_{i-1}$ are not contained in C in Algorithm 2, and P_L and P_R are empty regarding C in line 10. However, in this case, Q_L or Q_R is not empty and C still cannot be outputted based on the condition in line 10. (2) Algorithm 2 outputs all the maximal balanced cliques in G. In line 2, Algorithm 2 visits each vertex v_i. Based on the recursive structure of MBCEnumUtil, all the maximal balanced cliques containing v_i are explored. Therefore, it can be proved. (3) No duplicate maximal balanced cliques are outputted in Algorithm 2. During the recursive enumeration procedure, when we finish the maximal balanced clique enumeration containing a vertex v, we add the vertex into Q_L (line 6, line 18) or Q_R (line 7, line 21). Therefore, when we explore a maximal balanced clique C containing a vertex v_i and C has been outputted when processing v_j ($j < i$). Then, v_j will be in Q_L or Q_R in line 10 and C will not be outputted duplicately. Combining above three aspects together, the correctness of Algorithm 2 is proved.

Example 5.2. The enumeration procedure of MBCEnum can be illustrated as a search tree. Figure 3 shows part of the search tree when we conduct the MBCE on G in Figure 2 through MBCEnum.

$S_1, S_2, \ldots$ represent different search states during the enumeration. At S_1, we assume that we have a balanced clique $C = \{C_L = \{v_0, v_1\}, C_R = \{v_5, v_6\}\}$, P_L=$\{v_2, v_3\}$, P_R=$\{v_7, v_8\}$ at this state. We first grow search branch by adding v_2 from P_L into C_L. Since v_7 and v_8 are not v_2's negative neighbors, they are removed from P_R at S_2. Because P_R is empty at S_2, we keep expending C_L by adding v_3 from P_L. At S_3, P_L, R_L, Q_L and Q_R are empty, we obtain a maximal balanced clique $C_2 = \{\{v_0, v_1, v_2, v_3\}, \{v_5, v_6\}\}$ and this search branch starting from v_2 finishes. We return back to S_1 and v_2 is moved to Q_L. Then, we add v_3 into C_L at S_4 and add v_7 into C_R at S_5 and obtain $C_1 = \{\{0, 1, 3\}, \{5, 6, 7\}\}$. The search continues in a similar way until all the vertices in P_L and P_R at S_1 are explored.

Based on Algorithm 2, it is clear that the memory consumption of our enumeration framework is linear to the size of the input signed network. Therefore, the drawback of large memory consumption in Baseline is avoided.

6 OPTIMIZATION STRATEGIES

Although Algorithm 2 addresses the memory consumption problem in MBCE, the efficiency of Algorithm 2 is disappointing. In this section, we present two optimization strategies, namely in-enumeration optimization and pre-enumeration optimization, to further improve the efficiency of the enumeration.

6.1 In-Enumeration Optimization

Branch Pruning. Branch pruning aims to prune the unfruitful branches in the search tree of Algorithm 2 to improve the performance.

Pivot Choosing. Consider the maximal balanced clique search procedure of Algorithm 2, assume that we currently have C_L, C_R, P_L and P_R, and we add a vertex v from P_L to C_L in line 17. After finishing the search starting from v, we do not need to further explore the positive neighbors of v in the for loop of line 16 and the negative neighbors of v in the for loop of line 19. The reasons are as follows: w.o.l.g, let v' be a positive neighbor of v, although we skip the maximal balanced clique search starting from v', these maximal balanced cliques containing v' must be explored by the searching branches starting v or neighbors of v'. Therefore skipping the search starting from v's neighbors does not affect the correctness of Algorithm 2.

In this paper, to maximum the benefits of pivot technology, we define the local degree for a vertex $v \in P_L \cup Q_L (P_R \cup Q_R)$ as $d_l(v) = |N^{+(-)}(v) \cap P_L| + |N^{-(+)}(v) \cap P_R|$, and we choose the vertex v that satisfies $\max_{v \in V'}\{d_l(v)\}$ as the pivot, where $V' = P_L \cup P_R \cup Q_L \cup Q_R$.

Candidate Selection. In the search procedure of Algorithm 2, heuristically, search starting from a vertex with small local degree will have a short and narrow search branch, which means the search starting from the vertex will be finished very fast. Moreover, due to the search finish of the vertex, the vertex will be added into the excluded set and it can be used to further prune other search branches. Therefore, instead of adding vertices from P_L and P_R into C_L and C_R randomly in line 16 and 19 of Algorithm 2, we add vertices in the increasing order of their local degrees.

Algorithm 3 MBCEnum*($G = (V, E^+, E^-), k$)

```
1:  line 1-7 of Algorithm 2;
2:  MBCEnumUtil*($C_L$, $C_R$, $P_L$, $P_R$, $Q_L$, $Q_R$);
3:  **Procedure** MBCEnumUtil*($C_L$, $C_R$, $P_L$, $P_R$, $Q_L$, $Q_R$)
4:  line 10-13 of Algorithm 2;
5:  if $|C_L| + |P_L| < k$ or $|C_R| + |P_R| < k$ then
6:      return;                                          // ET Rule 1
7:  if $\exists v \in Q_L$, s.t., $P_L \subseteq N_G^+(v)$ and $P_R \subseteq N_G^-(v)$ or $\exists v \in Q_R$, s.t.,
    $P_R \subseteq N_G^+(v)$ and $P_L \subseteq N_G^-(v)$ then
8:      return ;                                         // ET Rule 2
9:  if $\forall p_l \in P_L$, s.t., $P_L \subseteq \{\{p_l\} \cup N_G^+(p_l)\}$ and $P_R \subseteq N_G^-(p_l)$ and
    $\forall p_r \in P_R$, s.t., $P_R \subseteq \{\{p_r\} \cup N_G^+(p_r)\}$ and $P_L \subseteq N_G^-(p_r)$ then
10:     output $C = \{C_L \cup P_L, C_R \cup P_R\}$;
11:     return ;                                         // ET Rule 3
12: Flag ←!Flag;
13: $p \leftarrow \mathrm{argmax}_{v \in P_L \cup P_R \cup Q_L \cup Q_R}\{d_l(v)\}$;   // Pivot Choosing
14: /* assume $p$ from $P_L \cup Q_L$ */
15: newP_L ← $P_L \setminus N_G^+(p)$;
16: newP_R← $P_R \setminus N_G^-(p)$;
17: sort(newP_L); sort(newP_R);                          // Candidate Selection
18: if Flag then
19:     for each $v \in$ newP_L
20:         line 17-18 replacing MBCEnumUtil with MBCEnumUtil*;
21:     for each $v \in$ newP_R
22:         line 20-23 replacing MBCEnumUtil with MBCEnumUtil*;
```

Early Termination. We consider different conditions that we can terminate the search early in Algorithm 2. For a balanced clique $C = \{C_L, C_R\}$, the maximal possible size of C_L (C_R) for the final maximal balanced clique is $|C_L| + |P_L|$ ($|C_R| + |P_R|$). Based on the size constraint of k, we have the following rule:

- **ET Rule 1:** If $|C_L| + |P_L| < k$ or $|C_R| + |P_R| < k$, we can terminate current search directly.

In Algorithm 2, we use Q_L and Q_R to store such vertices that the maximal balanced cliques containing them have been enumerated. Therefore, during the enumeration, if there exists a vertex $v \in Q_L(Q_R)$ such that $P_L(P_R) \subseteq N_G^+(v)$ and $P_R(P_L) \subseteq N_G^-(v)$, then we can conclude that the maximal balanced cliques have been enumerated. Following this, we have our second rule:

- **ET Rule 2:** If $\exists v \in Q_L$, s.t., $P_L \subseteq N_G^+(v)$ and $P_R \subseteq N_G^-(v)$ or $\exists v \in Q_R$, s.t., $P_R \subseteq N_G^+(v)$ and $P_L \subseteq N_G^-(v)$, then we can terminate current search directly.

In a certain search of Algorithm 2, if all the vertices in P_L (P_R) consist a clique formed by positive edges and every vertex in P_L (P_R) has negative edges to all the vertices in P_R (P_L), then P_L and P_R consist a balanced clique. Then, based on Definition 3.2, $C_L \cup P_L$ and $C_R \cup P_R$ consist a maximal balanced clique. Therefore, we have our third early termination rule:

- **ET Rule 3:** If $\forall p_l \in P_L$, s.t., $P_L \subseteq \{\{p_l\} \cup N_G^+(p_l)\}$ and $P_R \subseteq N_G^-(p_l)$ and $\forall p_r \in P_R$, s.t., $P_R \subseteq \{\{p_r\} \cup N_G^+(p_r)\}$ and $P_L \subseteq N_G^-(p_r)$, we can output $C = (C_L \cup P_L, C_R \cup P_R)$ and terminate current search directly.

Note that, in order to avoid outputting duplicate maximal balanced cliques, ET Rule 3 must be applied after ET Rule 2.

Algorithm. The maximal balanced clique enumeration algorithm with in-enumeration optimization strategies is shown in Algorithm 3. Since the pseudocode is self-explained, we omit the detailed description here.

Theorem 6.1. *Given a signed network G, the time complexity of Algorithm 3 to enumerate the maximal balanced cliques in G is $O(\sigma n \cdot 3^{\sigma/3})$, where σ is the degeneracy number of G.*

Proof. Given a graph G, the degeneracy number of G is σ[1]. Let $P = P_L \cup P_R$, $Q = Q_L \cup Q_R$, we first prove the size constraint for P. In line 2 of Algorithm 2, we iterates v_i in the degeneracy order of G and vertices with a lower order than v_i are not included in P. Therefore, for P regarding v_i, we have $|P| \leq \sigma$. Then, we analyse the time complexity of MBCEnumUtil*. In detail, ET Rule 1 can be done in $O(1)$ time. For ET Rule 2, we need to get local neighbors (within P) for each vertex in Q, it costs $O(|Q||P|)$ time. Similarly, for ET Rule 3, the time complexity for getting local neighbors for vertices in P is $O(|P|^2)$. Moreover, pivot selection and candidates sort consume $O(|P|+|Q|)$ time and $O(|P|\log|P|)$ time, respectively, based on above computation. So far, the time complexity is $O(|P|(|P|+|Q|))$. And because each recursion for MBCEnumUtil* can invoke at most $|P|$ further recursion, so the further time complexity is $O(|P|^2(|P| + |Q|))$. Now, we formulate the time complexity function for MBCEnumUtil* with parameters $|P|$ and $|Q|$, namely $T(|P|, |Q|) = max_s\{\sum_{i=1}^{s}[T(|P'_i|, |Q|)]\} + |P|^2(|P| + |Q|)$, note that P'_i is the new candidates reduced by neighbors of i-th vertex in the rest candidates. As we choose a vertex with maximum pruning size as pivot, we get $|P'_i| < |P| - s$ where s is the size of the rest candidates after pivot pruning. Hence we get $T(|P|, |Q|) \leq max_s\{sT(|P| - s, |Q|)\} + |P|^2(|P| + |Q|)$, it's proved that $T(|P|, |Q|)$ can be bounded by $O((\sigma + |Q|) \cdot 3^{|P|/3})$[12]. Sum the time of $T(|P|, |Q|)$ for n vertices, the finally time complexity is $\sum_{i=1}^{n}[(\sigma + |Q_i|) \cdot 3^{|P_i|/3}] = O((\sigma n + m) \cdot 3^{\sigma/3}) = O(\sigma n \cdot 3^{\sigma/3})$ due to the size constraint of P. □

6.2 Pre-Enumeration Optimization

In pre-enumeration optimization, we aim to remove the unpromising vertices and edges that not contained in any maximal balanced cliques based on their structural information. We explore two optimization strategies based on the neighbors of a vertex and the common neighbors of an edge.

Vertex Reduction. To reduce the size of a signed network, we first consider the neighbors of each vertex v, i.e., $N^+_G(v)$ and $N^-_G(v)$ to remove the unpromising vertices. We first define:

Definition 6.2. **((l, r)-signed core)** Given a signed network $G = (V, E^+, E^-)$, two integers l and r, a (l, r)-signed core is a maximal subgraph C of G, s.t., $\min_{v\in C}\{d^+_C(v)\} = l$, $\min_{v\in C}\{d^-_C(v)\} = r$.

Lemma 6.3. *Given a signed network G and threshold k, a maximal balanced clique satisfying the size constraint with k is contained in a $(k-1, k)$-signed core.*

Proof. We can prove it by contradiction. Assume there is a vertex v in a maximal balanced C satisfying the size constraint with k but not in a $(k-1, k)$-signed core. Based on Definition 3.2, the positive degree of v in C is not less than $k-1$ and the negative degree of v in C is not less than k. This contradicts with our assumption. Thus, the lemma holds. □

[1]Given a graph G, its degeneracy number, namely σ, is the least d such that the vertices of G can be arranged in a sequence so that each vertex is adjacent to at most d of the vertices that follow it in the sequence [48].

Algorithm 4 VertexReduction($G = (V, E^+, E^-)$, k)

```
1: while $\exists v \in V$, s.t. $d^+_G(v) < k-1$ or $d^-_G(v) < k$ do
2:     for each $u \in N^+_G(v)$ do
3:         $d^+_G(u) \leftarrow d^+_G(u) - 1$;
4:     for each $u \in N^-_G(v)$ do
5:         $d^-_G(u) \leftarrow d^-_G(u) - 1$;
6:     $G \leftarrow G \setminus v$;
```

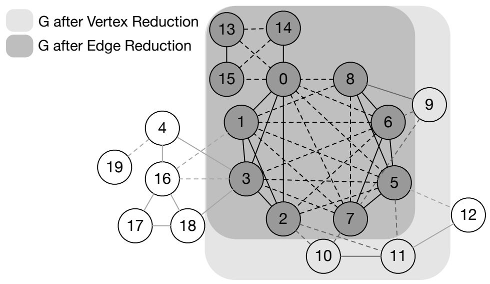


Figure 4: Vertex Reduction and Edge Reduction

Therefore, in order to compute the maximal balanced cliques in a given signed network G with integer k, we only need to compute the maximal balanced cliques in the corresponding $(k-1, k)$-signed core of G. The remaining problem is how to efficiently compute the $(k-1, k)$-signed core. We propose a linear algorithm to address this problem, which is shown in Algorithm 4.

Algorithm. Based on Definition 6.2, to compute the $(k-1, k)$-signed core in the signed network G, we only need to identify the vertices like v with $d^+_G(v) < k-1$ or $d^-_G(v) < k$ and remove them from G. Due to the removal of such vertices, more vertices will be with positive degree less than $k-1$ or negative degree less than k, we can further remove these vertices until no such kind of vertices exist in G. Following this idea, in Algorithm 4, we first identify a vertex v with $d^+_G(v) < k-1$ or $d^-_G(v) < k$ (line 1). Since v will be removed from G, we decrease the positive degree by 1 for each positive neighbor of v (line 2-3) and decrease the negative degree by 1 for each negative neighbor of v (line 4-5). Then, we remove v from G (line 6). The algorithm terminates when no vertex with $d^+_G(v) < k-1$ or $d^-_G(v) < k$ exists in G (line 1). It is clear that Algorithm 4 correctly computes the $(k-1, k)$-signed core of G. And we have the following theorem regarding its efficiency.

Theorem 6.4. *Given a signed network G and an integer k, the time complexity of Algorithm 4 is $O(n + m)$.*

Proof. In Algorithm 4, we use a queue to store vertices that should be removed in line 6. Since every vertex is pushed in and popped from the queue at most once, the total processing time for this part is $O(n)$. Moreover, when a vertex is removed, we have to update the degrees for their neighbors once, the total time cost is $O(m)$. Therefore, the time complexity of Algorithm 4 is $O(n+m)$. □

Example 6.5. Let $k = 2$, Figure 4 shows an example of vertex reduction by Algorithm 4 on the signed network G in Figure 2.

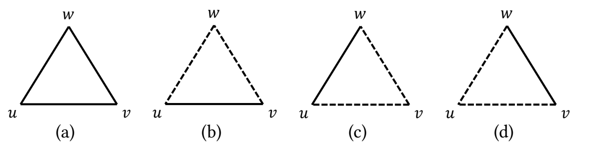


Figure 5: Different types of common neighbors for (u, v)

$v_4, v_{12}, v_{16}, v_{17}, v_{18}, v_{19}$ are pruned, because they are not contained in the $(1, 2)$-signed core.

Edge Reduction. In this part, we explore the opportunities to remove unpromising edges with respect to MBCE by considering the common neighbors of an edge formed by different types of edges. Specifically, for a positive/negative edge (u, v), we define the edge common neighbor number:

Definition 6.6. **(Edge Common Neighbor Number)** Given a signed network $G = (V, E^+, E^-)$, for a positive edge (u, v), we define:

- $\delta_G^{++}(u, v) = |\{w|(u, w) \in E^+ \wedge (v, w) \in E^+\}|$
- $\delta_G^{--}(u, v) = |\{w|(u, w) \in E^- \wedge (v, w) \in E^-\}|$

for a negative edge (u, v), we define:

- $\delta_G^{+-}(u, v) = |\{w|(u, w) \in E^+ \wedge (v, w) \in E^-\}|$
- $\delta_G^{-+}(u, v) = |\{w|(u, w) \in E^- \wedge (v, w) \in E^+\}|$

Figure 5 shows the different types of common neighbors used in Definition 6.6. For a positive edge (u, v), Figure 5 (a) and (b) show the common neighbor w used in $\delta_G^{++}(u, v)$ and $\delta_G^{--}(u, v)$, respectively. For a negative edge (u, v), Figure 5 (c) and (d) show the common neighbor w used in $\delta_G^{+-}(u, v)$ and $\delta_G^{-+}(u, v)$, respectively. Note that G is undirected and every edge is stored once in G. Based on Definition 6.6, we have the following lemma:

Lemma 6.7. *Given a signed network G and an integer k, let G' be the maximal sub-network of G s.t.,*

(1) $\forall (u, v) \in E_{G'}^+, \rightarrow \delta_{G'}^{++}(u, v) \geq k - 2 \wedge \delta_{G'}^{--}(u, v) \geq k;$
(2) $\forall (u, v) \in E_{G'}^-, \rightarrow \delta_{G'}^{+-}(u, v) \geq k - 1 \wedge \delta_{G'}^{-+}(u, v) \geq k - 1;$

then, every maximal balanced clique $C = \{C_L, C_R\}$ in G satisfying the size constraint with k is contained in G'.

Proof. This lemma can be proved similarly as Lemma 6.3. □

Algorithm. With Lemma 6.7, in order to enumerate the maximal balanced cliques in a given signed network G with respect k, we only need to keep the edges in G' shown in Lemma 6.7 and the positive/negative edges not in G' can be safely pruned. Now we focus on efficiently computing G' and our algorithm is shown in Algorithm 5. We first compute $\delta_G^{++}(u, v)$ and $\delta_G^{--}(u, v)$ for each positive edge of G (line 1-2) and $\delta_G^{+-}(u, v)$ and $\delta_G^{-+}(u, v)$ for each negative edge of G (line 3-4). Following Lemma 6.7, for each positive edge (u, v) such that $\delta_G^{++}(u, v) < k - 2$ or $\delta_G^{--}(u, v) < k$, we remove (u, v) (line 9). After that, we decrease the corresponding edge common neighbor numbers that have been changed due to the removal of (u, v) for the edge incident to (u, v) (line 10-15) based on Definition 6.6. Similarly, for each negative edge not satisfying the conditions in Lemma 6.7, we remove it and decrease the corresponding edge common neighbor numbers (line 17-24). The algorithm terminates when all the edges satisfy conditions in Lemma 6.7.

Algorithm 5 EdgeReduction($G = (V, E^+, E^-), k$)

```
1: for each $(u, v) \in E^+$ do
2:     compute $\delta_G^{++}(u, v)$ and $\delta_G^{--}(u, v)$;
3: for each $(u, v) \in E^-$ do
4:     compute $\delta_G^{+-}(u, v)$ and $\delta_G^{-+}(u, v)$;
5: Stop← false;
6: while Stop = false do
7:     Stop ← true;
8:     if $\exists (u, v) \in E^+$, s.t $\delta_G^{++}(u, v) < k - 2$ or $\delta_G^{--}(u, v) < k$ then
9:         $G \leftarrow G \setminus (u, v)$;
10:        for each $w$ s.t. $(u, w) \in E^+$ and $(v, w) \in E^+$ do
11:            $\delta_G^{++}(u, w) \leftarrow \delta_G^{++}(u, w) - 1$;
12:            $\delta_G^{++}(v, w) \leftarrow \delta_G^{++}(v, w) - 1$;
13:        for each $w$ s.t.$(u, w) \in E^-$ and $(v, w) \in E^-$ do
14:            $\delta_G^{+-}(u, w) \leftarrow \delta_G^{+-}(u, w) - 1$;
15:            $\delta_G^{+-}(v, w) \leftarrow \delta_G^{+-}(v, w) - 1$;
16:        Stop← false;
17:    if $\exists (u, v) \in E^-$ s.t. $\delta_G^{+-}(u, v) < k - 1$ or $\delta_G^{-+}(u, v) < k - 1$ then
18:        $G \leftarrow G \setminus (u, v)$;
19:        for each $w$ s.t. $(u, w) \in E^+$ and $(v, w) \in E^-$ do
20:            $\delta_G^{--}(u, w) \leftarrow \delta_G^{--}(u, w) - 1$;
21:            $\delta_G^{-+}(v, w) \leftarrow \delta_G^{-+}(v, w) - 1$;
22:        for each $w$ s.t. $(u, w) \in E^-$ and $(v, w) \in E^+$ do
23:            $\delta_G^{-+}(u, w) \leftarrow \delta_G^{-+}(u, w) - 1$;
24:            $\delta_G^{--}(v, w) \leftarrow \delta_G^{--}(v, w) - 1$;
25:        Stop← false;
```

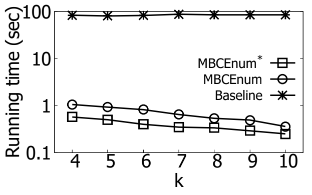
Theorem 6.8. *Given a signed network $G = (V, E^+, E^-)$, an integer k, the time complexity of Algorithm 5 is $O(m^{1.5})$.*

Proof. We first prove that the time complexity of line 1-2 of Algorithm 5 can be bounded by $O(m^{1.5})$, which follows the idea in [7]. Let $d_G(v) = d_G^+(v) + d_G^-(v)$ and assume that $d_G(u) < d_G(v)$. To compute the $\delta_G^{++}(u, v)$, we can do it as follows: for each $w \in N_G^+(u)$, we check whether $(v, w) \in E^+$. If $(v, w) \in E^+$ then $\delta_G^{++}(u, v)$ increases by 1. Otherwise, $\delta_G^{++}(u, v)$ keep the same. In this way, the time complexity of line 1-2 to compute $\delta_G^{++}(u, v)$ is $O(\sum_{u \in V}(d^+(u) \cdot |N_{\geq u}^+|))$, where $|N_{\geq u}^+|$ is the number of positive neighbors v of u s. t. $d_G(u) < d_G(v)$. Obviously, $O(\sum_{u \in V}(d^+(u) \cdot |N_{\geq u}^+|))$ can be bounded by $O(\sum_{u \in V}(d(u) \cdot |N_{\geq u}|))$, where $|N_{\geq u}|$ is the number of neighbors v of u s. t. $d_G(u) < d_G(v)$. $O(\sum_{u \in V}(d(u) \cdot |N_{\geq u}|))$ can be bounded by $O(m^{1.5})$. This is because if $d_G(u) \leq \sqrt{m}$, $|N_{\geq u}| \leq d_G(u) \leq \sqrt{m}$ and $\sum_{u \in V}(d_G(u) \cdot |N_{\geq u}|) \leq 2m^{1.5}$. If $d_G(u) > \sqrt{m}$, $|N_{\geq u}| \leq \sqrt{m}$ as well for $d_G(u) \cdot |N_{\geq u}| \leq \sum_{v \in |N_{\geq u}|} d_G(v) < 2m$, and $\sum_{u \in V}(d_G(u) \cdot |N_{\geq u}|) \leq 2m^{1.5}$. Similarly, we can prove that line 3-4 can be bounded by $O(m^{1.5})$. And similar to line 6-25. So, the time complexity of Algorithm 5 is $O(m^{1.5})$. □

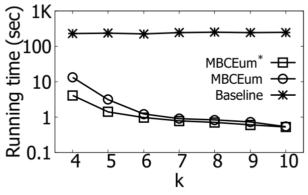
Example 6.9. Let $k = 2$, Figure 4 shows the result of edge reduction on G in Figure 2. As shown in Figure 4, $\{v_9, v_{10}, v_{11}\}$ cannot be pruned by vertex reduction directly, as they are contained in a $(1, 2)$-signed core. However, with edge reduction, as $\delta_G^{--}(v_8, v_9)$=1<2, (v_8, v_9) is removed. Then, (v_6, v_9) and (v_7, v_9) are removed as $\delta_G^{-+}(v_6, v_9)$=0<1 and $\delta_G^{-+}(v_7, v_9)$=0<1. As v_9 has no edges afterwards, it is pruned. v_{10} and v_{11} are pruned similarly.

Table 1: Statistic for the real datasets

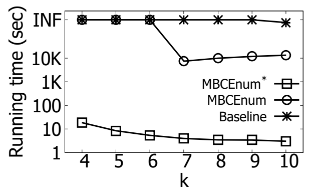
Dataset	n	m	$\|E^+\|$	$\|E^-\|$
AdjWordNet	21,247	426,896	378,993	47,903
Slashdot	77,357	516,575	396,378	120,197
Epinions	131,828	841,372	717,667	123,705
DBLP	1,314,050	5,179,945	1,471,903	3,708,042
Douban	1,588,565	13,918,375	9,034,537	4,883,838



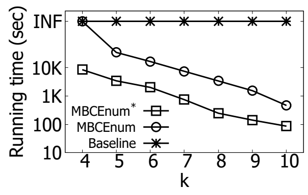
(a) Slashdot (vary k)



(b) Epinions (vary k)



(c) DBLP (Vary k)



(d) Douban (Vary k)

Figure 6: Running time of different algorithms varying k

7 PERFORMANCE STUDIES

In this section, we present our experimental results. All the experiments are performed on a machine with two Intel Xeon 2.2GHz CPUs and 64GB RAM running CentOS 7.

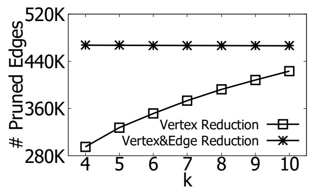
Algorithms. We compare three algorithms: Baseline , MBCEnum and MBCEnum*. Baseline is the baseline solution shown in Section 4. MBCEnum is our algorithm shown in Section 5. MBCEnum* is the algorithm with the in-enumeration optimization shown in Section 6.1. Note that the pre-enumeration optimization strategies can be also used in Baseline and MBCEnum, thus, we apply them for all three algorithms for fairness.

All algorithms are implemented in C++, using g++ complier with -O3. The time cost is measured as the amount of wall-clock time elapsed during the program's execution. If an algorithm cannot finish in 12 hours, we denote the processing time as INF. We evaluate our algorithms on real and synthetic signed networks.

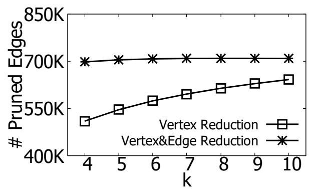
Real datasets. We first evaluate our algorithms on five real datasets. Slashdot and Epinions are signed networks in real world. AdjWordNet, DBLP and Douban are signed networks used in [8], [27] and [50], respectively. Slashdot and Epinioins are downloaded from SNAP (http://snap.stanford.edu). Douban is from authors in [50]. AdjWordNet is downloaded from WordNet (https://wordnet.princeton.edu/). DBLP is downloaded from KONECT (http://konect.uni-koblenz.de/) and processed the same as shown in [27]. The details of each dataset are shown in Table 1.

Exp-1: Efficiency when varying k. In this experiment, we evaluate the efficiency of three algorithms when varying k from 4 to 10 and the results are shown in Figure 6.

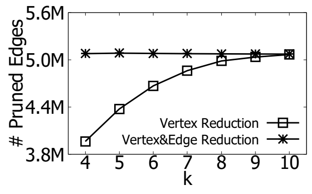
As shown in Figure 6, Baseline consumes the most time among three algorithms on all datasets when we vary k and it can only handle the small datasets. For example, on Slashdot (Figure 6 (a)), MBCEnum and MBCEnum* are at least two orders of magnitude faster than Baseline . On Douban (Figure 6 (d)), Baseline cannot finish the enumeration in 12 hours. This is because Baseline does not consider the uniqueness of the signed networks and lots of unnecessary computations are involved in the enumeration of Baseline . MBCEnum is faster than Baseline on most of the test cases as MBCEnum takes the uniqueness of the signed networks into consideration and enumerates the maximal balanced cliques based on the signed network directly. MBCEnum* is the most efficient algorithm on all datasets when varying k due to the utilization of in-enumeration optimization strategies, which reveals the effectiveness of in-enumeration optimization strategies. Another phenomena shown in Figure 6 is that the running time of all algorithms decreases as k increases. This is because as k increases, the pruning power of the optimization strategies proposed in Section 6 strengthens.



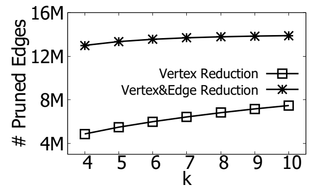
(a) Slashdot (Vary k)



(b) Epinions (Vary k)

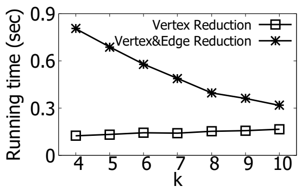


(c) DBLP (Vary k)

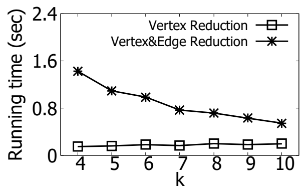


(d) Douban (Vary k)

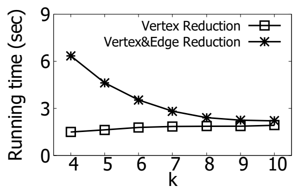
Figure 7: Pruned edges by pre-enumeration optimizations



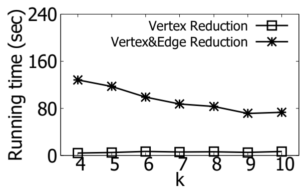
(a) Slashdot (Vary k)



(b) Epinions (Vary k)



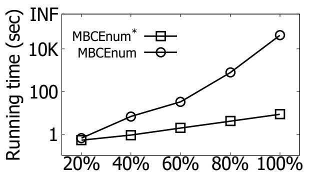
(c) DBLP (Vary k)



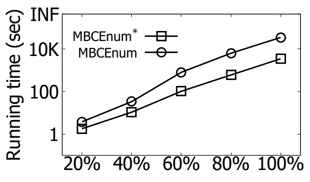
(d) Douban (Vary k)

Figure 8: Running time of pre-enumeration optimizations

Exp-2: Evaluation of the pre-enumeration optimization. In this experiment, we evaluate the effectiveness and efficiency of the pre-enumeration optimization strategies proposed in Section 6.2. We report the number of pruned edges for VertexReduction and the sum of pruned edges for VertexReduction and EdgeReduction when varying k in Figure 7. Figure 8 shows the running time.



(a) DBLP (Vary n)



(b) Douban (Vary n)

Figure 9: Scalability of MBCEnum and MBCEnum*

As shown in Figure 7, for VertexReduction, as k increases, the number of pruned edges increases as well. This is because as k increases, more vertices are not contained in the corresponding $(k-1, k)$-signed core. These vertices together with their incident edges are pruned. Figure 7 also reveals that EdgeReduction prunes much more edges than VertexReduction. This is because EdgeReduction adopts a more restrict pruning condition. Figure 8 shows that as k increases, the running time of VertexReduction increases. Since as k increases, more vertices are explored by VertexReduction. On the other hand, the running time of them together decreases. This is because as k increases, more vertices are pruned by VertexReduction. As a result, EdgeReduction takes less time to conduct the pruning. Meanwhile, EdgeReduction is more time-consuming compared with VertexReduction. Thus, the running time together decreases.

Exp-3: Scalability testing. In this experiment, we test the scalability of MBCEnum and MBCEnum* on two large datasets DBLP and Douban by varying their vertices from 20% to 100%. Figure 9 shows the results.

As shown in Figure 9, when n increases, the running time of both algorithms increases as well, but MBCEnum* outperforms MBCEnum for all cases on both datasets. For example, on DBLP, when we sample 20% vertices, the running time of MBCEnum and MBCEnum* is 0.6 seconds and 0.5 seconds, respectively, while when sampling 80% vertices, their running times are 770.6 seconds and 4.0 seconds, respectively. It shows that MBCEnum* has a good scalability in practice.

Exp-4: Case study on AdjWordNet. In this experiment, we perform a case study on the real dateset AdjWordNet. In this dataset, two synonyms have a positive edge and two antonyms have a negative edge, and Table 2 shows some results obtained by our algorithm. As shown in Table 2, words in C_L or C_R have similar meaning while each word from C_L is an antonym to all words in C_R. This case study verifies that maximal balanced clique enumeration can be applied in the applications to find synonym and antonym groups on dictionary data.

Exp-5: Efficiency on synthetic datasets. In this experiment, we evaluate our algorithms on synthetic datasets. We use the synthetic signed network generator, SRN, to generate the synthetic datasets with default settings [45, 56]. We generate four synthetic signed networks SN1-4 (details in Table 3) in different sizes and evaluate the efficiency of MBCEnum* and MBCEnum on SN1-4 similarly as Exp-1. The results are shown in Figure 10.

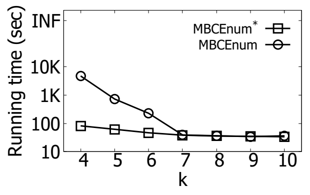
As shown in Figure 10, the trends on the synthetic datasets are similar to that on the real datasets. MBCEnum* outperforms MBCEnum when we vary k, especially when k is small.

Table 2: Case study on AdjWordNet

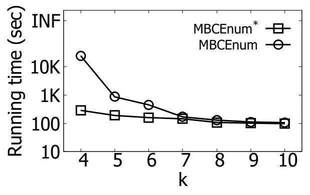
C_L	C_R
raw, rough, rude	refined, smooth, suave
relaxing, reposeful, restful	restless,uneasy, ungratified, unsatisfied
interior, internal, intimate	away, foreign, outer, outside, remote
assumed, false, fictitious, fictive, mistaken, off-key, pretended, put-on, sham, sour, untrue	actual, existent, existing, factual, genuine, literal, real, tangible, touchable, true, truthful, unfeigned, veridical
active, animated, combat-ready, dynamic, dynamical, fighting, participating, alive, live	adynamic, asthenic, debilitated, enervated, undynamic, stagnant, light
following, undermentioned, next	ahead, in-the-lead, leading, preeminent, prima, star, starring, stellar
undesirable, unsuitable, unwanted	cherished, treasured, wanted, precious

Table 3: Statistic for the synthetic datasets

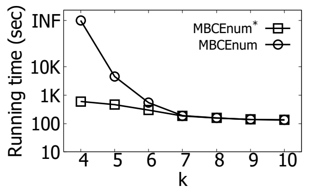
Dataset	n	m	$\|E^+\|$	$\|E^-\|$
SN1	500,000	17,535,536	10,192,418	7,343,118
SN2	1,000,000	35,054,718	20,373,721	14,680,997
SN3	2,000,000	70,156,704	40,776,222	29,380,482
SN4	3,000,000	105,218,600	61,159,457	44,059,143



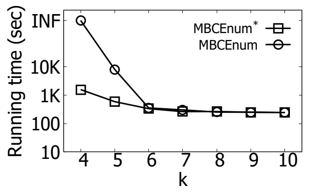
(a) SN1 (Vary k)



(b) SN2 (Vary k)



(c) SN3 (Vary k)



(d) SN4 (Vary k)

Figure 10: Running time of different algorithms varying k

8 CONCLUSIONS

In this paper, we study the maximal balanced clique enumeration problem in signed networks. We propose a new enumeration algorithm tailored for signed networks. Based on the new enumeration algorithm, we explore two optimization strategies to further improve the efficiency of the enumeration algorithm. The experimental results on real and synthetic datasets demonstrate the efficiency, effectiveness and scalability of our solution.

Acknowledge. Long Yuan is supported by NSFC61902184 and NSF of Jiangsu Province BK20190453. Xuemin Lin is supported by 2018YFB1003504, NSFC61232006, ARC DP180103096 and DP170101628. Lu Qin is supported by ARC DP160101513.

REFERENCES

[1] Peter Abell and Mark Ludwig. 2009. Structural balance: a dynamic perspective. *Journal of Mathematical Sociology* 33, 2 (2009), 129–155.

[2] Akkoyunlu and E. A. 1973. The Enumeration of Maximal Cliques of Large Graphs[J]. *SIAM J. Comput.* 2, 1 (1973), 1–6.

[3] R. Axelrod and D. S. Bennett. 1993. A landscape theory of aggregation. *BJPS* 23, 02 (1993), 211–233.

[4] Coenraad Bron and Joep Kerbosch. 1973. Finding all cliques of an undirected graph (algorithm 457). *Commun. ACM* 16, 9 (1973), 575–576.

[5] Dorwin Cartwright and Frank Harary. 1956. Structural balance: a generalization of Heider's theory. *Psychological review* 63, 5 (1956), 277.

[6] James Cheng, Linhong Zhu, Yiping Ke, and Shumo Chu. 2012. Fast algorithms for maximal clique enumeration with limited memory. In *Proceedings of SIGKDD.* 1240–1248.

[7] N. Chiba and T. Nishizeki. 1985. Arboricity and subgraph listing algorithms. *SIAM J. Comput* 14, 1 (1985), 210–223.

[8] Lingyang Chu, Zhefeng Wang, Jian Pei, Jiannan Wang, Zijin Zhao, and Enhong Chen. 2016. Finding gangs in war from signed networks. In *Proceedings of SIGKDD.* 1505–1514.

[9] J. Cohen. 2008. Trusses: Cohesive subgraphs for social network analysis. *IEEE Transactions on Knowledge and Data Engineering* (2008).

[10] Tyler Derr, Charu Aggarwal, and Jiliang Tang. 2018. Signed Network Modeling Based on Structural Balance Theory. In *Proceedings of CIKM.* 557–566.

[11] David Easley and Jon Kleinberg. 2010. *Networks, crowds, and markets: Reasoning about a highly connected world.* Cambridge University Press.

[12] David Eppstein, Maarten Löffler, and Darren Strash. 2010. Listing all maximal cliques in sparse graphs in near-optimal time. In *International Symposium on Algorithms and Computation.* 403–414.

[13] David Eppstein and Darren Strash. 2011. Listing all maximal cliques in large sparse real-world graphs. In *International Symposium on Experimental Algorithms.* 364–375.

[14] Faten Fakhfakh, Mohamed Tounsi, Mohamed Mosbah, and Ahmed Hadj Kacem. 2017. Algorithms for Finding Maximal and Maximum Cliques: A Survey. In *International Conference on Intelligent Systems Design and Applications.* 745–754.

[15] Xing Feng, Lijun Chang, Xuemin Lin, Lu Qin, Wenjie Zhang, and Long Yuan. 2018. Distributed computing connected components with linear communication cost. *Distributed and Parallel Databases* 36, 3 (2018), 555–592.

[16] Ming Gao, Ee-Peng Lim, David Lo, and Philips Kokoh Prasetyo. 2016. On detecting maximal quasi antagonistic communities in signed graphs. *Data mining and knowledge discovery* 30, 1 (2016), 99–146.

[17] Fei Hao, Stephen S Yau, Geyong Min, and Laurence T Yang. 2014. Detecting k-balanced trusted cliques in signed social networks. *IEEE Internet Computing* 18, 2 (2014), 24–31.

[18] Frank Harary et al. 1953. On the notion of balance of a signed graph. *The Michigan Mathematical Journal* 2, 2 (1953), 143–146.

[19] Fritz Heider. 1946. Attitudes and cognitive organization. *The Journal of psychology* 21, 1 (1946), 107–112.

[20] Xin Huang, Hong Cheng, Lu Qin, Wentao Tian, and Jeffrey Xu Yu. 2014. Querying k-truss community in large and dynamic graphs. In *SIGMOD.* 1311–1322.

[21] Margarita Išoraitė. 2009. Importance of strategic alliances in company's activity. *BJPS* 1, 5 (2009), 39–46.

[22] Adit Krishnan, Deepak P, Sayan Ranu, and Sameep Mehta. 2018. Leveraging semantic resources in diversified query expansion. *World Wide Web* 21, 4 (2018), 1041–1067.

[23] Srijan Kumar, Francesca Spezzano, V. S. Subrahmanian, and Christos Faloutsos. 2016. Edge Weight Prediction in Weighted Signed Networks. In *IEEE 16th International Conference on Data Mining.* 221–230.

[24] Vishwajeet Kumar, Nitish Joshi, Arijit Mukherjee, Ganesh Ramakrishnan, and Preethi Jyothi. 2019. Cross-Lingual Training for Automatic Question Generation. In *ACL.* 4863–4872.

[25] Jérôme Kunegis, Andreas Lommatzsch, and Christian Bauckhage. 2009. The slashdot zoo: mining a social network with negative edges. In *Proceedings of WWW.* 741–750.

[26] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. 2010. Signed networks in social media. In *Proceedings of SIGCHI.* 1361–1370.

[27] Rong-Hua Li, Qiangqiang Dai, Lu Qin, Guoren Wang, Xiaokui Xiao, Jeffrey Xu Yu, and Shaojie Qiao. 2018. Efficient Signed Clique Search in Signed Networks. In *Proceedings of ICDE.* 245–256.

[28] Boge Liu, Long Yuan, Xuemin Lin, Lu Qin, Wenjie Zhang, and Jingren Zhou. 2019. Efficient (α, β)-core Computation: an Index-based Approach. In *Proceedings of WWW.* 1130–1141.

[29] Weiwei Liu, Ivor W. Tsang, and Klaus-Robert Müller. 2017. An Easy-to-hard Learning Paradigm for Multiple Classes and Multiple Labels. *J. Mach. Learn. Res.* 18 (2017), 94:1–94:38.

[30] Weiwei Liu, Donna Xu, Ivor W. Tsang, and Wenjie Zhang. 2019. Metric Learning for Multi-Output Tasks. *IEEE Trans. Pattern Anal. Mach. Intell.* 41, 2 (2019), 408–422.

[31] David Lo, Didi Surian, and Kuan Zhang. 2011. Mining direct antagonistic communities in explicit trust networks. In *Proceedings of CIKM.* 1013–1018.

[32] Seth A Marvel, Jon Kleinberg, Robert D Kleinberg, and Steven H Strogatz. 2011. Continuous-time model of structural balance. *Proceedings of the National Academy of Sciences* 108, 5 (2011), 1771–1776.

[33] Seth A Marvel, Steven H Strogatz, and Jon M Kleinberg. 2009. Energy landscape of social balance. *Physical review letters* 103, 19 (2009), 198701.

[34] G. A. Miller. 1995. Wordnet: a lexical database for english. *Commun. ACM* 38, 11 (1995), 39–41.

[35] Le Ou-Yang, Dao-Qing Dai, and Xiao-Fei Zhang. 2015. Detecting protein complexes from signed protein-protein interaction networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 12, 6 (2015), 1333–1344.

[36] Dian Ouyang, Long Yuan, Lu Qin, Lijun Chang, and Ying Zhang. 2020. Efficient Shortest Path Index Maintenance on Dynamic Road Networks with Theoretical Guarantees. *PVLDB* 13, 5 (2020), 602–615.

[37] Dian Ouyang, Long Yuan, Fan Zhang, Lu Qin, and Xuemin Lin. 2018. Towards Efficient Path Skyline Computation in Bicriteria Networks. In *Proceedings of DASFAA.* 239–254.

[38] Jeffrey Pattillo, Nataly Youssef, and Sergiy Butenko. 2013. On clique relaxation models in network analysis. *European Journal of Operational Research* 226, 1 (2013), 9–18.

[39] Zhu Qing, Long Yuan, Fan Zhang, Lu Qin, Xuemin Lin, and Wenjie Zhang. 2018. External Topological Sorting in Large Graphs. In *Proceedings of DASFAA.* 203–220.

[40] Ahmet Erdem Sariyuce, C Seshadhri, Ali Pinar, and Umit V Catalyurek. 2015. Finding the hierarchy of dense subgraphs using nucleus decompositions. In *Proceedings of WWW.* 927–937.

[41] Ahmet Erdem Sariyüce, C. Seshadhri, Ali Pinar, and Ümit V. Çatalyürek. 2017. Nucleus Decompositions for Identifying Hierarchy of Dense Subgraphs. *TWEB* 11, 3 (2017), 16:1–16:27.

[42] Matthew C. Schmidt, Nagiza F. Samatova, Kevin Thomas, and Byung-Hoon Park. 2009. A Scalable, Parallel Algorithm for Maximal Clique Enumeration. *J. Parallel Distrib. Comput.* 69, 4 (2009), 417–428.

[43] Stephen B Seidman. 1983. Network structure and minimum degree. *Social networks* 5, 3 (1983), 269–287.

[44] Xiaodan Song, Yun Chi, Koji Hino, and Belle Tseng. 2007. Identifying opinion leaders in the blogosphere. In *Proceedings of CIKM.* 971–974.

[45] Yansen Su, Bangju Wang, Fan Cheng, Xingyi Zhang, and Linqiang Pan. 2017. An algorithm based on positive and negative links for community detection in signed networks. In *Nature Scientific Reports.*

[46] Jiliang Tang, Yi Chang, Charu Aggarwal, and Huan Liu. 2016. A survey of signed network mining in social media. *Comput. Surveys* 49, 3 (2016), 42.

[47] Etsuji Tomita, Akira Tanaka, and Haruhisa Takahashi. 2006. The worst-case time complexity for generating all maximal cliques and computational experiments. *Theoretical Computer Science* 363, 1 (2006), 28–42.

[48] D. Wen, L. Qin, Y. Zhang, X. Lin, and J. X. Yu. 2019. I/O Efficient Core Graph Decomposition: Application to Degeneracy Ordering. *IEEE TKDE* 31, 1 (2019), 75–90.

[49] Xudong Wu, Long Yuan, Xuemin Lin, Shiyu Yang, and Wenjie Zhang. 2019. Towards Efficient k-TriPeak Decomposition on Large Graphs. In *Proceedings of DASFAA.* 604–621.

[50] Tong Xu, Dong Liu, Enhong Chen, Huanhuan Cao, and Jilei Tian. 2012. Towards annotating media contents through social diffusion analysis. In *Proceedings of ICDM.* 1158–1163.

[51] Long Yuan, Lu Qin, Xuemin Lin, Lijun Chang, and Wenjie Zhang. 2016. Diversified top-k clique search. *VLDB J.* 25, 2 (2016), 171–196.

[52] Long Yuan, Lu Qin, Xuemin Lin, Lijun Chang, and Wenjie Zhang. 2016. I/O Efficient ECC Graph Decomposition via Graph Reduction. *PVLDB* 9, 7 (2016), 516–527.

[53] Long Yuan, Lu Qin, Xuemin Lin, Lijun Chang, and Wenjie Zhang. 2017. Effective and Efficient Dynamic Graph Coloring. *PVLDB* 11, 3 (2017), 338–351.

[54] Long Yuan, Lu Qin, Xuemin Lin, Lijun Chang, and Wenjie Zhang. 2017. I/O efficient ECC graph decomposition via graph reduction. *VLDB J.* 26, 2 (2017), 275–300.

[55] Long Yuan, Lu Qin, Wenjie Zhang, Lijun Chang, and Jianye Yang. 2018. Index-Based Densest Clique Percolation Community Search in Networks. *IEEE TKDE* 30, 5 (2018), 922–935.

[56] Yujie Zeng and Jing Liu. [n.d.]. Community detection from signed social networks using a multi-objective evolutionary algorithm. In *Proceedings of IES*, Vol. 1.

[57] Yun Zhang, Charles A Phillips, Gary L Rogers, Erich J Baker, Elissa J Chesler, and Michael A Langston. 2014. On finding bicliques in bipartite graphs: a novel algorithm and its application to the integration of diverse biological data types. *BMC bioinformatics* 15, 1 (2014), 110.

[58] Xiaolong Zheng and Daniel Zeng. 2015. Social balance in signed networks. *Information Systems Frontiers* 17, 5 (2015), 1077–1095.

[59] R. Zhou, C. Liu, J. X. Yu, W. Liang, B. Chen, and J. Li. 2012. Finding maximal k-edge-connected subgraphs from a large graph. *in EDBT* (2012).