Balanced Clique Computation in Signed Networks: Concepts and Algorithms

Zi Chen, Long Yuan, Xuemin Lin, Fellow, IEEE, Lu Qin, and Wenjie Zhang

Abstract—Clique is one of the most fundamental models for cohesive subgraph mining in network analysis. Existing clique model mainly focuses on unsigned networks. However, in real world, 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. Following the balanced clique model, we study the maximal balanced clique enumeration problem (MBCE) which computes all the maximal balanced cliques in a given signed network. Moreover, in some applications, users prefer a unique and representative balanced clique with maximum size rather than all balanced cliques. Thus, we also study the maximum balanced clique search problem (MBCS) which computes the balanced clique with maximum size. We show that MBCE problem and MBCS problem are both NP-Hard. For the MBCE problem, a straightforward solution 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. For the MBCS problem, we first propose a baseline solution. To overcome the huge search space problem of the baseline solution, we propose a new search framework based on search space partition. To further improve the efficiency of the new framework, we propose multiple optimization strategies regarding to redundant search branches and invalid candidates. We conduct extensive experiments on large real datasets. The experimental results demonstrate the efficiency, effectiveness and scalability of our proposed algorithms for MBCE problem and MBCS problem.

Index Terms—Balanced clique, structural balance theory, signed network, graph algorithm

1 INTRODUCTION

With the proliferation of graph applications, research efforts have been devoted to many fundamental problems in analyzing graph data [1], [2], [3], [4], [5], [6], [7]. 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 dominating set of the clique and the diameter of a clique is one [1]. 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 [3], [4], [5], [6]. Unfortunately, relationships between two entities in many real-world applications have completely opposite properties, such as friend-foe relationships between users in social networks [8], [9], support-dissent opinions in opinion networks [10], trust-distrust relationships in trust networks [11] and partnership-antagonism in protein-protein interaction networks [12]. Modelling these applications as signed networks with positive and negative edges allows them to capture more sophisticated semantics than unsigned networks [11], [13], [14], [15], [16], [17]. 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 [8], [11], [13], [14], [15], [16], [17], [18], [19]. 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 [19]. In a signed network, an imbalanced sub-structure is unstable and tends to evolve into a balanced state. Consider the graph $G$ shown...
in Fig. 1a. The negative edge between \(v_1\) and \(v_2\) makes \(G\) imbalanced. \(v_1\) 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 Fig. 1b 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, 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.

Moreover, in real signed networks, the number of maximal balanced cliques could be extremely large. For instance, in “Douban” network which is a Chinese score service website, there are more than a million balanced cliques in it. However, in some applications, users prefer a unique and representative balanced clique with maximum size rather than all balanced cliques. Maximum clique search problem is a fundamental and hot research topic in graph analysis. In the literature, numerous studies have been conducted, such as maximum clique search [20], [21], maximum quasi-clique search [22], maximum bi-clique search [23], \(k^*\)-partite clique with maximum edges [24], clique with maximum edge/vertex weight on weighted graph [25], [26]. Motivated by this, we aim to devise a maximum balanced clique search algorithm to find out the balanced clique with maximum vertex size, which can scale to large-scale real signed networks (with more than 100 million edges).

Applications for Maximal Balanced Clique Enumeration (MBCE). Maximal balanced clique enumeration can be used in many applications. For example,

(1) Opinion Leaders Detection In Opinion Networks. In an opinion network, each vertex represents a user and there is a positive/negative edge between two vertices if one user supports/dissents another user. Opinion leaders are people who are active involved in a community capturing the most representative standpoints in the social networks [27], like the rival groups of voters that support different political leaders [28]. The users in the rival groups actively involve in the opinion networks (every pair of them have a voting opinion with each other) and have their clear standpoints (support everyone in the same group and dissent everyone in the opposite group who votes for different political leaders). Our balanced clique model can naturally find a group of users in an opinion network such that any two of them have an opinion with each other, and the group can be further divided into two subgroups such that the intra-group users support each other and the inter-group users dissent each other. Therefore, the users in the balanced cliques are good candidates for opinion leaders in the opinion networks.

(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 balanced cliques in such networks reveals hostile groups of allied forces [8], [29]. We can extend it to find the alliances-rivalries commercial groups among business organizations similarly, such as (Pepsi, KFC) versus (Coke, McDonald) [30].

(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 [31]. 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 [32] and semantic expansion [33].

Applications for Maximum Balanced Clique Search (MBCS). In some applications, instead of all the maximal balanced cliques, the balanced clique with the largest number of vertices is preferred. For example,

(1) Portfolio Risk Management. In an investment network, a vertex represents a security, and there is a positive/negative edge between two vertices if the prices or values of these two corresponding securities tend to move in the same/opposite direction. A portfolio is a collection of financial securities or assets held by an investor. A group of securities only connected by positive edges represents a speculative portfolio, and the risk of such portfolio is high since all its securities tend to move in the same direction (profit or loss). On the other hand, a portfolio containing two groups of securities such that securities in the same group tend to move in the same direction while securities in different groups tend to move in opposite direction provides the investors with a hedging guarantee to reduce the investment risk. Clearly, balanced cliques model can capture this unique requirement, and the maximum balanced clique is a good candidate for such portfolio as the most number of securities are covered [34], [35].

(2) Conflict Detection. In the opinion network, our maximum balanced clique model can capture the biggest conflicting groups that hold clear but opposite standpoints, which may represent the core members of two polarized structures. Thus, computing the maximum balanced clique could help discover and prevent potential conflicts such as “flame war” on the social network [36].

In addition, as the number of maximal balanced cliques in some real signed networks could be extremely large, it is hard
to process such large number of balanced cliques. A general compromise solution for this problem is to consider the top $k$ diversified / representative maximal balanced cliques. Inspired by the well-studied top $k$ diversified clique problem [37], we can follow the same procedure by repeatedly removing the current maximum balanced clique from the signed network $k$ times to obtain the results. Clearly, the efficient computation of maximum balanced clique is the key of this problem.

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. An Efficient Maximum Balanced Clique Search Algorithm. To address the maximum balanced clique search problem, we first propose a baseline algorithm. In order to reduce the search space during the search process of baseline, we propose a search space partition-based algorithm MBCE-SSP by partitioning the whole search space into multiple search regions. In each search region, two size thresholds $\bar{k}$ and $k$ are used to search the result matching the size requirement specific to this search region, such that the search space is limited into a small area. To further improve the efficiency of MBCE-SSP algorithm, we also explore three optimization strategies to prune invalid search branches and candidates during the search process.

5. Extensive Performance Studies on Real Datasets. We first evaluate the performance of MBCE algorithms by conducting extensive experimental studies on real datasets. 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. Then, we evaluate the performance of our proposed MBCE algorithm. The baseline algorithm can not get the result within a reasonable time on large datasets, while our optimized algorithm shows high efficiency, effectiveness and scalability.

Due to limited space, more details of our MBCE approach can be found in our full conference version [38].

2 RELATED WORK

Signed Network Analysis. Structural balance theory is originally introduced in [18] and generalized in the graph formation in [13], [19]. After that, structural balance theory is developed extensively [11], [14], [15], [16], [17]. In these works, it is interesting to mention that the authors in [16] 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 $\mu \leq 0$. [39] provides a comprehensive survey on structural balanced theory.

Besides, a large body of literature on mining signed networks has been emerged. Among them, the most closely related work to ours is [40] in which an $(\alpha, k)$-clique model is proposed. Compared with our model, $(\alpha, k)$-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 $(\alpha, k)$-clique model essentially different from our model. In [41], a $k$-balanced trusted clique model is proposed. 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.

Clique on Unsigned Networks. Graph analysis is a significant research field [42], [43], [44], [45], [46]. Clique model is one of the most fundamental cohesive subgraph models [47], [48]. [3] proposes an efficient algorithm for maximal clique enumeration based on backtracking search. [43] first considers the memory consumption during the maximal clique enumeration. Based on [3], more efficient algorithms are investigated [5], [6], [47]. [5] proposes a novel branch pruning strategy, which can efficiently reduce the search space by ignoring the search process from the neighbors of the pivot [49] reviews recently advances in maximal clique enumeration. Based on clique, other cohesive subgraph models are also studied, such as $k$-core [50], [51], $k$-truss [52], [53], $k$-edge connected component [54], and $(r, s)$-nuclei [55], [56].

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. For each vertex $v \in G$, let $N^+_G(v)$ represents the positive neighbors of $v$, and let $N^-_G(v)$ represents the negative neighbors of $v$. We use $d^+_G(v)$ and $d^-_G(v)$ to denote the positive and negative degree of $v$, respectively. We also use $N^+_G(v)$ and $d^+_G(v)$ to denote the neighbors and degree of $v$, i.e., $N^+_G(v) = N^+_G(u) \cup N^+_G(v)$ and $d^+_G(v) = d^+_G(u) + d^+_G(v)$.

For simplicity, we omit $G$ in the above notations if the context is self-evident.

Definition 3.1. (Balanced Network [19]) Given a signed network $G = (V, E^+, E^-)$, it’s balanced if 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 $u \in C_R, v \in C_L \rightarrow (u, v) \in E^-$. 

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
Fig. 2. Maximal balanced clique in G (k = 2).

Definition 3.3. (Maximum Balanced Clique) Given a signed network \( G = (V, E^+, E^-) \), a maximum balanced clique \( C^* \) in \( G \) is a balanced clique with the maximum vertex size.

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 problems in the paper as follows:

Problem Statement. Given a signed network \( G \) and an integer \( k \),

- the maximal balanced clique enumeration (MBCE) problem aims to compute all the maximal balanced cliques \( C \) in \( G \) s.t. \( |C_L| \geq k \) and \( |C_R| \geq k \) for \( C \).
- the maximum balanced clique search (MBCS) problem aims to compute the balanced clique \( C^* \) in \( G \) s.t. \( |C_L^*| \geq k \), \( |C_R^*| \geq k \) and \( |C_L^*| + |C_R^*| \) is maximum.

Example 3.1. Consider the signed network \( G \) in Fig. 2 in which positive/negative edges are denoted by solid/dashed lines. Assume \( k = 2 \), there are 4 maximal balanced cliques in \( G \), namely, \( C_1 = \{v_1, v_2, v_3, v_5, v_7\} \), \( C_2 = \{v_4, v_6, v_8\} \), \( C_3 = \{v_9, v_1, v_5, v_6, v_8\} \), \( C_4 = \{v_9, v_14, v_13, v_15\} \), where vertices in \( C_L \) and \( C_R \) are marked with different colors. Among them, \( C_2 \) is the maximum balanced clique.

Problem Hardness. The MBCE problem is NP-Hard, which can be proved following the NP-Hardness of maximal clique enumeration problem [57], [58]. Given an unsigned network \( G = (V, E) \), we can transfer \( G \) to a signed network \( G \) in which \( \forall u \in C_L \rightarrow (v, u) \in E^+ \) and \( \forall w \in C_R \rightarrow (w, v) \in E^- \), then \( C' = \{C_L \cup \{v\}, C_R\} \) is also a balanced clique in \( G \).

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 \( \forall w \in C_R \rightarrow (w, v) \in E^- \), then \( C' = \{C_L \cup \{v\}, C_R\} \) is also a balanced clique in \( G \).

In this section, we present a new enumeration framework by considering the uniqueness of signed networks to overcome the memory consumption problem of baseline.

Algorithm of MBCEEnum. Following the above idea, our algorithm for MBCE is shown in Algorithm 1. For each vertex \( v_L \) in \( C_L \) (line 2), we enumerate all the maximal balanced cliques containing \( v_L \) (line 3–8). Note that \( v_0, v_1, \ldots, v_n \) are in the degeneracy order [60] of \( G \). We use \( C_L \) and \( C_R \) to maintain the balanced clique, which are initialized with \( v_L \) and \( v_R \), respectively (line 3). 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 MBCEEnumUtil to enumerate all the maximal balanced cliques containing \( v_L \).

Procedure MBCEEnumUtil 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, MBCEEnumUtil 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, MBCEEnumUtil 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.
Algorithm 1. MBCEnum \((G = (V, \mathcal{E}^{+}, \mathcal{E}^{-}), k)\)

1: Flag ← true;
2: for each \(v \in \{v_1, v_2, \ldots, v_{n-1}\} \in V\) do
3: \(C_L \leftarrow \{v\}, C_R \leftarrow \emptyset\);
4: \(P_L \leftarrow N_G^+(v) \cap \{v_{n+1}, \ldots, v_{n-k}\}\);
5: \(P_R \leftarrow N_G^-(v) \cap \{v_{n+1}, \ldots, v_{n-k}\}\);
6: \(Q_L \leftarrow N_G^+(v) \cap \{v_{n+1}, \ldots, v_{n-k}\}\);
7: \(Q_R \leftarrow N_G^-(v) \cap \{v_{n+1}, \ldots, v_{n-k}\}\);
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: \(\text{if } |C_L| \geq k \text{ and } |C_R| \geq k \text{ then return}\);
12: \(\text{output } C = \{C_L, C_R\};\)
13: return
14: Flag ← !Flag;
15: if Flag then
16: for each \(v \in P_L\) do
17: MBCEnumUtil \((C_L \cup \{v\}, C_R, N_G^+(v) \cap P_L, N_G^-(v) \cap P_L, N_G^+(v) \cap Q_L, N_G^-(v) \cap Q_L)\);
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 \((C_L, C_R \cup \{v\}, N_G^+(v) \cap P_L, N_G^-(v) \cap P_L, N_G^+(v) \cap Q_L, N_G^-(v) \cap Q_L)\);
21: \(P_R \leftarrow P_R \setminus \{v\}; Q_R \leftarrow Q_R \cup \{v\};\)
22: else
23: line 19–21; line 16–18;

6 ENUMERATION OPTIMIZATION STRATEGIES

6.1 In-Enumeration Optimization

Branch Pruning. Consider the maximal balanced clique search problem of Algorithm 1, 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 1.

In this paper, to maximize the benefits of pivot technology, we define the local degree for a vertex \(v \in P_L \cup Q_L\) as \(d_G(v) = |N_G^{+\text{v}}(v) \cap P_L| + |N_G^{-\text{v}}(v) \cap P_R|\). We choose the vertex \(v\) that satisfies \(\max_{v \in V} d_G(v)\) as the pivot, where \(V = P_L \cup P_R \cup Q_L \cup Q_R\).

Candidate Selection. In the search procedure of Algorithm 1, 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 1, we add vertices in the increasing order of their local degrees.

Early Termination. We consider different conditions that we can terminate the search early in Algorithm 1. 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 \(k\), we have the following rules:

- ET Rule 1: If \(|C_L| + |P_L| < k\) or \(|C_R| + |P_R| < k\), we can terminate current search directly.
- ET Rule 2: If \(\exists v \in Q_L\) s.t. \(P_L \subseteq N_G^{+\text{v}}(v)\) and \(P_R \subseteq N_G^{-\text{v}}(v)\) or \(\exists v \in Q_R\) s.t. \(P_L \subseteq N_G^{+\text{v}}(v)\) and \(P_R \subseteq N_G^{-\text{v}}(v)\), then we can terminate current search directly.
- ET Rule 3: If \(\forall v \in P_L\) s.t. \(P_L \subseteq \{p_L \cup N_G^{+\text{v}}(p_L)\} \) and \(P_R \subseteq N_G^{-\text{v}}(p_R)\) and \(P_L \subseteq N_G^{+\text{v}}(p_L)\) and \(P_R \subseteq N_G^{-\text{v}}(p_R)\), we can output \(C = (C_L \cup P_L, C_R \cup P_R)\) and terminate current search directly.

6.2 Pre-Enumeration Optimization

Vertex Reduction. To reduce the size of a signed network, we first consider the neighbors of each vertex \(v\), i.e., \(N_G^{+\text{v}}(v)\) and \(N_G^{-\text{v}}(v)\) to remove the unpromising vertices.

Definition 6.1. ((l, r)-Signed Core) Given a signed network \(G = (V, \mathcal{E}^{+}, \mathcal{E}^{-})\), two integers \(l\) and \(r\), a \((l, r)\)-signed core is a maximal subgraph \(G'\) of \(G\), s.t., \(\min_{v \in V} d_G^+(v) = l, \min_{v \in V} d_G^-(v) = r\).

Lemma 6.1. 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.

Therefore, in order to address MBC with respect to \(k\), we only need to compute the maximal balanced cliques in the corresponding \((k-1, k)\)-signed core of \(G\).

Algorithm of VertexReduction. Based on Definition 6.1, to compute the \((k-1, k)\)-signed core, we only need to identify the vertices 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 violate the degree constraints, we can further remove these vertices until no such vertices exist in \(G\).

Edge Reduction. In this part, we explore the opportunities to remove unpromising edges with respect to MBCEC 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 6.2. (Edge Common Neighbor Number) Given a signed network \(G = (V, \mathcal{E}^{+}, \mathcal{E}^{-})\), for a positive edge \((u, v)\),

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

for a negative edge \((u, v)\),

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

Lemma 6.2. Given a signed network \(G\) and an integer \(k\), let \(G'\) be the maximal sub-network of \(G\) s.t.,

1) \(\forall (u, v) \in E_G^+ \to \delta_G^+(u, v) \geq k-2 \land \delta_G^-(u, v) \geq k;\)
2) \(\forall (u, v) \in E_G^- \to \delta_G^-(u, v) \geq k-1 \land \delta_G^+(u, v) \geq k-1;\)
then, every maximal balanced clique \(C = (C_L, C_R)\) in \(G\) satisfying the size constraint with \(k\) is contained in \(G'\).
With Lemma 6.2, in order to address MBCE with respect $k$, we only need to keep edges in $G'$ shown in Lemma 6.2 and remove other unnecessary edges.

**Algorithm of EdgeReduction.** We first compute $\delta_{G}^{+}(u, v)$ and $\delta_{G}^{-}(u, v)$ for each positive edge of $G$ and $\delta_{G}^{+}(u, v)$ and $\delta_{G}^{-}(u, v)$ for each negative edge of $G$. Following Lemma 6.2, for each positive edge $(u, v)$ such that $\delta_{G}^{+}(u, v) < k - 2$ or $\delta_{G}^{-}(u, v) < k$, we remove $(u, v)$. 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 $(u, v)$ based on Definition 6.2. It's similar to negative edges. The algorithm terminates when all the edges satisfy conditions in Lemma 6.2.

### 7 Maximum Balanced Clique Search

Maximum clique search problem is a fundamental and hot research topic in graph analysis. In this section, we study the maximum balanced clique search problem.

#### 7.1 A Baseline Approach

As we aim to search the balanced clique with maximum size in the given graph, a straightforward idea is to enumerate all maximal balanced cliques and maintain the maximum balanced clique $C^*$ found so far. When the enumeration finishes, it is easy to verify that $C^*$ is the maximum balanced clique in the graph. During the enumeration process, if the candidates cannot form a larger balanced clique, i.e., $|C_L| + |C_R| + |P_L| + |P_R| \leq \epsilon$ where $\epsilon = |C^*|$, we can terminate such search branch directly. Following the above idea, we propose a baseline approach, namely MBCSear.

**Drawbacks of MBCSear.** Although the straightforward approach can find the maximum balanced clique, the complexity of MBCSear is the same as that of MBCEnum in the worst case. The search space of MBCSear is huge. In details, the drawbacks of MBCSear are twofold.

- **Lack of rigorous size constraints for $C_L$ and $C_R$.** Given a signed graph $G$, during the search process, MBCSear only holds the size constraint $|C_L| + |C_R| + |P_L| + |P_R| > \epsilon$ for each search branch. However, when $\epsilon$ is small, most of search branches have $|C_L| + |C_R| + |P_L| + |P_R|$ larger than $\epsilon$ which causes the fail of size constraint for most search branches. Unfortunately, as our algorithm constantly searches larger result than at present, the value of $\epsilon$ is gradually increasing from a small value, which makes MBCSear has to search the result with large search space.

- **Massive invalid search branches.** Although the search branches meet the size constraint with $\epsilon$, the structure of subgraph produced by $P_L$ and $P_R$ may be sparse which will generate invalid search branches. Hence, during the search process, more pruning techniques is needed urgently. Moreover, the optimization strategies based on $k$ in MBCEnum, like vertex reduction and edge reduction, are limited here, as $C^*$ usually has size much larger than $k$. Therefore, the remaining graph after reduction is still huge on large-scale signed network.

#### 7.2 Search Space Partition-Based Framework

Regarding to the first drawback of the baseline, the size constraint for $C_L$ and $C_R$ is too loose which leads to large invalid search space. To address this problem, more rigorous size constraints are urgently needed. Hence, in this subsection, we propose new rigorous size constraints under two size bounds $\kappa$ and $\pi$ for $C_L$ and $C_R$, respectively (assume $|C_L| \leq |C_R|$). We can search the maximum balanced clique under new size constraints, i.e., $|C_L| \geq \kappa$ and $|C_R| \geq \pi$ within narrow search space. Under different value of $\kappa$ and $\pi$, the search space is split into multiple partitions. As the balanced clique with large size can be found early with rigorous size constraints, the total search space can be significantly reduced in practice. Inspired by this, we propose a search space partition-based MBCS framework MBCSear-SSP. Given two certain values $\kappa_i$ and $\pi_i$, a search region (partition) is denoted as $(\kappa_i, \pi_i)$. In each search region, we keep searching larger result than at present. When all search regions are explored, $C^*$ can be found.

Moreover, in order to search balanced cliques with large size $\epsilon$ as priority, for the first search region $(\kappa_0, \pi_0)$, $\kappa_0$ is initialized as a large integer value. Obviously, as $\kappa_0$ value is large, benefited from the strict size constraint, most of search branches of MBCSear are ineligible now. Hence the result can be found quickly in this search region. Besides, to obey the size threshold $k$, we make $\kappa_0 = k$. Then, to cover the whole search space, for the later search regions, we keep increasing $\kappa$ and decreasing $\pi$ until $\kappa = \pi$. In another word, for $i < j$, $\kappa_i \leq \kappa_j$ and $\pi_i \geq \pi_j$.

Here, we first assign the possible maximum value to $\pi_0$, we have the following lemma:

**Lemma 7.1.** Given a signed network $G = (V, E^+, E^-)$, for every balanced clique, we have $\max\{|C_L|, |C_R|\} \leq \sigma + 1$, where $\sigma$ is the degeneracy number of $G^*$.

**Proof.** In unsigned networks, the degeneracy number plus 1 is an upper bound for the maximum size of cliques [21]. Based on Definition 3.2, in a signed network $G = (V, E^+, E^-)$, for every balanced clique $C = \{C_L, C_R\}$, $C_L$ and $C_R$ are traditional cliques in $G^*$. Therefore, $|C_L|$ and $|C_R|$ are must not greater than $\sigma + 1$, respectively.

Based on Lemma 7.1, we assign $(k, \sigma + 1)$ to the first search region $(\kappa_0, \pi_0)$. Then, in the later search region, we continue to seek larger balanced clique than the current one. However, not every search region can find a valid result. To skip invalid search regions, we have the following lemma:

**Lemma 7.2.** Given a signed network $G$, the maximum balanced clique found in the $i$-th search region $(\kappa_i, \pi_i)$ is denoted by $C_i^* = \{C_L^*_i, C_R^*_i\}$. Then, for the next search region $(\kappa_{i+1}, \pi_{i+1})$, we have $C_{i+1}^* = |C_i^*| - \pi_i$.

**Proof.** We prove it by contradiction. Following the $i$-th search region, in the next search region $(\kappa_{i+1}, \pi_{i+1})$, if we get a larger balanced clique $C_{i+1}^* = \{C_{L}^{i+1}, C_{R}^{i+1}\}$ than $C_i^*$. Based on our search framework, we have $\max\{|C_L^{i+1}|, |C_R^{i+1}|\} < \pi_{i+1}$. Otherwise, $C_{i+1}^*$ will be found in the $i$-th search region rather than the $(i+1)$-th search region. Now, we assume $\min\{|C_L^{i+1}|, |C_R^{i+1}|\} < |C_i^*| - \pi_i$. Combining with $\max\{|C_L^{i+1}|, |C_R^{i+1}|\} < \pi_{i+1}$, we have $|C_{i+1}^*| = |C_{L}^{i+1}| + |C_{R}^{i+1}| < |C_i^*|$. Obviously, it is against with our premise that $C_{i+1}^*$ is larger than $C_i^*$. Therefore, the assumption for $\min\{|C_L^{i+1}|, |C_R^{i+1}|\} < |C_i^*| - \pi_i$ does not hold. We get $\min\{|C_L^{i+1}|, |C_R^{i+1}|\} \geq |C_i^*| - \pi_i$, i.e., $\kappa_{i+1} = |C_i^*| - \pi_i$. \hfill $\Box$
Based on Lemma 7.2, after the \( i \)-th search region, the search regions with \( k < |C^*_i| - r \), can be skipped directly.

**Algorithm 2.** MBCSear-SSP \((G = (V, E^+, E^-), k)\)

1: compute degeneracy \( \sigma \) of \( G^+ = (V, E^+) \);
2: \( i \leftarrow 2k; k \leftarrow k; r \leftarrow \sigma + 1; \overline{r} \leftarrow -1; \)
3: while \( r \geq k \) and \( r < \overline{r} \) do
   4: \( \text{MBCSear}(G = (V, E^+, E^-), \epsilon) \) with adding size constraints:
      \( \min\{L, R\} < k \) or \( \max\{L, R\} < r \);
   5: \( r \leftarrow r; k \leftarrow \max\{\epsilon, r, k\}; \overline{r} \leftarrow \max\{\text{Dec}(r), \overline{r}\}; \)

Algorithm of MBCSear-SSP. Following the above idea, the new maximum balanced clique search algorithm MBCSear-SSP is shown at Algorithm 2. Given a signed network \( G = (V, E^+, E^-) \) and size threshold \( k \), we first compute the degeneracy number \( \sigma \) of \( G^+ = (V, E^+) \) (line 1). We initialize \( i = 2k, k = k; r = \sigma + 1 \) (line 2). Then, in each search region, Algorithm 2 invokes MBCSear to find the maximum balanced clique in current search region with adding size constraints: \( \min\{L, R\} < k \) or \( \max\{L, R\} < r \) on each search branch. For the first search region, we let \( k = \max\{\epsilon - \overline{r}, k\} \) (line 5), since the size threshold \( k \) is held for \( C_L \) and \( C_R \) as well. For the next value of \( k \), \( r = \max\{\text{Dec}(r), \overline{r}\} \), where \( \text{Dec}(r) \) is used to decrease \( r \). When \( r < k \), set \( r = k \). The search process finishes when \( r = k \) and \( r \) can not be reduced anymore, since if \( r \) is unchanged, this search region is covered by the previous region already (line 3). When the search process of the final search region is finished, Algorithm 2 terminates and returns the maximum balanced clique \( C^* \) in \( G \).

Then, we discuss how \( \text{Dec}(r) \) decreases the value of \( r \). The total running time of MBCSear-SSP can be formulated as \( T = \sum_{i=0}^{\omega} t_i \), where \( \omega \) is the number of search regions and \( t_i \) is the partial running time of the \( i \)-th search region. In this paper, to keep the efficiency of MBCSear-SSP, we give a heuristic way to decrease \( r \). In detail, if the last search region is time-consuming (set time threshold like 20 seconds), to make the total running time \( T \) as small as possible, we reduce the amount of search regions \( \omega \) by decreasing \( r \) by a large value, i.e., \( \text{Dec}(r) = \frac{r}{2} \), otherwise, \( \text{Dec}(r) = r - 2 \).

Now, we compare the search space between MBCSear and MBCSear-SSP. As shown at Fig. 3, since \( C^* \) is the maximum balanced clique, MBCSear searches all balanced cliques with size less than \( C^* \), until \( C^* \) is found. The search space of MBCSear is shown at Fig. 3. For the search space of MBCSear-SSP, benefited from the tight bounds for \( |C_L| \) and \( |C_R| \) at each search region, MBCSear-SSP searches local maximum balanced clique within small search space. For instance, as shown at Fig. 3, it finds the current maximum balanced clique \( C^*_0 \) in the first search region. Then, in the second search region, it searches balanced cliques with \( k_1 \leq |C_R| < k_0 \) and \( |C_L| \geq k_1 \) (assume \( |C_L| \leq |C_R| \)) until \( C^*_1 \) is found. As shown at Fig. 3, the search space of MBCSear-SSP is much smaller than MBCSear.

**Example 7.1.** Reconsidering the signed network \( G \) in Fig. 2, \( k = 2 \), Algorithm 2 first computes degeneracy number \( \sigma \) of \( G^+ = (V, E^+) \), so \( \sigma = 2 \). So, the first search region is \( (2, 3) \). Algorithm 2 find result \( C^*_0 = \{v_0, v_1, v_2\}, \{v_5, v_6, v_7\} \) at first, \( \epsilon = 6 \). Then, based on Lemma 7.2, \( k_1 \) is 3. As we should keep \( r \geq k \), \( k_1 \) is 3 as well. However, as the value of \( r \) is unchanged, this search region is covered by the first search region. Hence, Algorithm 2 is terminated and returns \( C^*_0 \) as \( C^* \). Comparing with MBCSear, the search space of MBCSear-SSP is reduced from (2.2) to (2.3).

### 7.3 Optimization Strategies

Regarding to the second drawback of MBCSear on the massive invalid search branches during search process, the existing constraints only focus on the size of subgraphs produced by candidates for balanced cliques but neglect the structure of the subgraphs. Obviously, when the structure is sparse, even their size is large, the balanced cliques in it will be still small which will cause massive invalid search branches. To address this drawback, we explore the chance to further improve the efficiency of our approach by considering the structure of the subgraphs produced by candidates. We first propose two optimization strategies, they are coloring-based branch pruning and vertex dominance-based candidate pruning, to prune invalid search branches and remove meaningless vertices from candidates. Then, we extend the vertex & edge reduction techniques of MBCE to prune more unnecessary vertices and edges in advance.

#### 7.3.1 Coloring-Based Branch Pruning

Given a search region \( (k, \overline{r}) \) and a search branch, if the upper bound of the balanced clique size in current search branch is less than the lower bounds \( k, \overline{r} \) and \( \epsilon \), current search branch can be pruned directly. Looking back to MBCSear, it uses the candidates size to form the upper bound. However, this upper bound is too loose, because although the number of candidates is large, the connectivity between candidates maybe sparse, which will lead to many invalid search branches. Hence, now, we aim to propose a tighter upper bound based on vertex coloring.

**Definition 7.1.** (Vertex Coloring [61]) Given a graph \( G \), vertex coloring in \( G \) aims to assign colors to each vertex such that vertices are different in color from their neighbors. The amount of colors needed in \( G \) is named chromatic number, denoted by \( \chi(G) \).

**Lemma 7.3.** Given a search branch \((C_L, C_R, P_L, P_R)\), the maximum balanced clique from this branch is denoted as \( C^* = \ldots \)
Given a graph $G$, the chromatic number $\gamma(G)$ is an upper bound of the maximum size of cliques in $G$ [21]. Based on it, the lemma can be proved.

Algorithm of ColoringPrune. We propose ColoringPrune algorithm to prune search branches. The pseudocode is shown at Algorithm 3. It first computes $\gamma(G_{L(R)})$ and $\gamma(G_{L(R)})$ (line 1–8). Then it returns true if the upper bound does not meet the size requirements, which means this search branch can be pruned directly, otherwise, returns false (line 9–11).

**Algorithm 3. ColoringPrune($C_L, C_R, P_L, P_R, \epsilon, k, \kappa$)**

1: $G_L \leftarrow G^+(P_L); G_R \leftarrow G^+(P_R);
2: \gamma(G_{L(R)}) \leftarrow 0; \text{col}(v) = 0$ for each $v \in P_{L(R)}$;
3: for each $v \in P_{L(R)}$ do
4: \hspace{0.5cm} \text{col}(v) \leftarrow 1;
5: while $\exists u \in N_{G_{L(R)}}(v)$, s.t., $\text{col}(u) = \text{col}(v)$ do
6: \hspace{1cm} $\text{col}(v) \leftarrow \text{col}(v) + 1;
7: \hspace{1cm} \text{if \ \gamma(G(v)) > \gamma(G_{L(R)})}$ then
8: \hspace{1.5cm} $\gamma(G_{L(R)}) \leftarrow G_{L(R)} + 1$;
9: \hspace{1.5cm} \text{if min}$\{\gamma(G(v)) + |C_L|, \gamma(G_R) + |C_R|\} < k$ or \text{max}$\{\gamma(G(v)) + |C_L|, \gamma(G_R) + |C_R|\} < \kappa$ or $\gamma(G(v)) + |C_L| + \gamma(G_R) + |C_R| < \epsilon$ then
10: \hspace{2cm} return true;
11: return false;

**Theorem 7.1.** The space complexity of Algorithm 3 is $O(|P_L| + |P_R| + |E_{G_L}| + |E_{G_R}|)$, the time complexity is $O(|P_L| + |P_R| + |E_{G_L}| + |E_{G_R}|)$.

Note that Algorithm 3 can be directly applied to MBCE problem with $k = k, \kappa = \kappa, \epsilon = 2k$. However, since $k$ is usually small, the effectiveness of Algorithm 3 is limited in MBCE problem.

**7.3.2 Vertex Dominance-Based Candidate Pruning**

To further improve the efficiency, we aim to reduce the number of candidates in $P_L$ and $P_R$ at each search branch by pruning invalid vertices from candidates. Our key idea is that if the neighbors of $v$ cover the neighbors of $u$ ($v$ connects all neighbors of $u$), $u$ cannot form a larger balanced clique than that of $v$, the search relevant to $u$ can be skipped safely as we only focus on the maximum balanced clique in the graph. In this paper, we describe this situation as the vertex dominance relationship, namely, $u$ is dominated by $v$. Combining the search process of our MBCE approach, to simplify the computation, we only consider the local neighborhood within candidates. We propose a formal definition of dominate vertex set below.

**Definition 7.2.** (Dominate Vertex Set) Given a signed network $G$ and a search branch with candidate sets $P_L$ and $P_R$, for a vertex $u$, if $N_I(u) \subseteq N_I(v)$ where $N_I(u/v)$ is the local neighbors of $u/v$ in $P_L/R$, then we say $v$ dominates $u$ and $u$ is dominated by $v$. A dominate vertex set of $v$ is the set of vertices dominated by $v$, denoted as $\Phi_v$.

Based on Definition 7.2, we have the following lemma:

**Lemma 7.4.** Given a signed network $G$ and a search branch with candidate sets $P_L$ and $P_R$, for each vertex $v \in P_L \cup P_R$, if $\exists w$, s.t., $u \in \Phi_w$, the further search for $u$ can be skipped.

The dominate vertex set of a candidate can be computed through the adjacency list join operation with time complexity $O(n^2)$, where $n = |\text{new}P_L| + |\text{new}P_R|$, which is time-consuming. Moreover, the amount of candidates to be computed could be very large. Therefore, there is a trade-off between the effectiveness and efficiency of the vertex dominance-based candidate pruning. To balance the effectiveness and efficiency, we only focus on the dominate vertex set of a single candidate who has strong pruning power and remove its dominate vertex set from candidates. Intuitively, pivot is the optimum choice as it has the most local neighbors. To obtain the dominate vertex set of pivot effectively, in this paper, we only consider four special cases with $n \leq 3$. The four special cases are introduced as follows, where $p$ is the pivot.

- **Case 1:** If $p \in Q_L \cup Q_R$ and $\text{new}P_L \cup \text{new}P_R = \{w\}$, then $\Phi_p = \{w\}$.
- **Case 2:** If $p \in Q_L \cup Q_R$, $\text{new}P_L \cup \text{new}P_R = \{w, q\}$ and $(w, q) \notin E$, then $\Phi_p = \{w, q\}$.
- **Case 3:** If $p \in P_L \cup P_R$, $\text{new}P_L \cup \text{new}P_R = \{p, w\}$, then $\Phi_p = \{w\}$.
- **Case 4:** If $p \in P_L \cup P_R$, $\text{new}P_L \cup \text{new}P_R = \{p, w, q\}$ and $(w, q) \notin E$, then $\Phi_p = \{w, q\}$.

Fig. 4 shows the four special cases of vertex dominance respectively. For case 1&3 (Fig. 4a), $p$ is selected as pivot, then, the surviving candidate is $w$ as other vertices are $p$’s neighbors. Since $N_I(w) \subseteq N_I(p)$, $w$ is dominated by $p$. For case 2&4 (Fig. 4b), $w$ and $q$ are two surviving candidates with pivot $p$, as $(w, q) \notin E$, they will not appear at a common balanced clique, hence, $N_I(w) \subseteq N_I(p)$, $N_I(q) \subseteq N_I(p)$, $\Phi_p = \{w, q\}$.

Based on Lemma 7.4, when we meet the above four special cases of vertex dominance, we skip the searches from vertices in $\Phi_p$ by deleting them from candidate sets. In this way, the invalid candidates can be further pruned effectively.

Note that, the reason that only four cases are considered is that the computation overhead for these four cases is little while lots of unnecessary computation can be pruned as verified in the experiments, and thus the whole
The performance is improved. Theoretically, more cases with candidates size greater than three can also be used. However, computing the dominate vertex set for these cases is costly and the extra overhead introduced may outweigh the benefits of pruning unnecessary computation accordingly, which leads to the whole performance decline. Therefore, in this paper, we only consider four special cases with few candidates.

### 7.3.3 Vertex & Edge Reduction Variants

Although the optimization strategies proposed in MBCE algorithm, like vertex reduction and edge reduction, are still applicable for MBCSearch, they only ensure that $C_L$ and $C_R$ are not less than $k$ but lack of binding force of our new size bounds $\kappa$, $\lambda$ and $\epsilon$. Moreover, the value of $\kappa$, $\lambda$ and $\epsilon$ are much larger than $k$, especially for $\kappa$ and $\lambda$, which makes the effectiveness of the reduction optimizations is limited in MBCS problem. Hence, we extend the vertex reduction and edge reduction such that they can support tighter bounds to prune more vertices and edges.

**Vertex reduction variant.** We first propose the vertex reduction variant considering the degree of vertices. We have the following lemma:

**Lemma 7.5.** Given a signed network $G = (V, E^+, E^-)$, a search region $(\kappa, \lambda, \epsilon)$ and $G = (V, E^+, E^-)$ is a subgraph of $G$, s.t., (a) $v \in V \min |d(v)| + 1, d^-(v)| \geq \kappa, \max |d(v)| + 1, d^+(v)| \geq \lambda, \kappa \geq \lambda > \epsilon$.

**Proof.** Given a search region $(\kappa, \lambda, \epsilon)$, let $C^*_i = \{C_L, C_R\}$ is the maximum balanced clique in current search region. Based on Algorithm 2, $C^*_i = \{C_L, C_R\}$ should satisfy $\min \{C_L, |C_R|\} \geq \kappa$ and $\max \{C_L, |C_R|\} \geq \lambda$. Meanwhile, based on Definition 3.2, $\forall v \in C_L$, $d^+(v) = |C_L| - 1$, $d^-(v) = |C_R|$. $\forall v \in C_R$, $d^+(v) = |C_L| - 1$, $d^-(v) = |C_R|$. Combining them, we get $\min \{d^+(v) + 1, d^+(v)\} \geq \kappa, \max \{d^+(v) + 1, d^+(v)\} \geq \lambda$. Moreover, as $C^*_i$ is the current maximum balanced clique, the degree of vertices in $C^*_i$ should not less than $\epsilon$.

**Algorithm of VertexReduction**. Based on Lemma 7.5, we can reduce the size of the candidate sets by continuously deleting the vertices that do not meet the degree constraints in Lemma 7.5. We propose VertexReduction algorithm, the pseudocode is shown at Algorithm 4. It continuously removes vertices until all vertices left in candidate sets meet the degree constraints.

### Algorithm 4. VertexReduction $(C_L, C_R, P_L, P_R, \epsilon, \kappa, \lambda, \rho)$

1. $V' = P_L \cup P_R \cap G = G(V')$
2. While $\exists v \in P_L(\cup) \text{s.t.} \min |d(v)| + 1, d^-(v)| \geq \kappa \lambda \max |d(v)| + 1, d^+(v)| \geq \lambda |C_L||C_R| \kappa \lambda \max |d(v)| + 1, d^+(v)| \geq \lambda |C_L||C_R| \kappa \lambda |C_L||C_R| \kappa \lambda |C_L||C_R| \kappa \lambda |C_L||C_R| \kappa \lambda |C_L||C_R| \kappa \lambda $ do
3. For each $u \in N^+_L(v)$ do
4. $d^+_L(u) \leftarrow d^+_L(u) - 1$
5. For each $u \in N^+_R(v)$ do
6. $d^+_R(v) \leftarrow d^+_R(v) - 1$
7. $G \leftarrow G \setminus v$
8. $P_L' \leftarrow P_L' \setminus v$
9. $P_R' \leftarrow P_R' \setminus v$

### Theorem 7.2. The space complexity of Algorithm 4 is $O(|P_L| + |P_R| + |E^+_L| + |E^-_L|)$.

**Edge reduction variant.** After the vertex reduction variant, we extend the edge reduction now. Inspired by the edge reduction technology utilized in MBCE, we continue to explore the edge reduction under a certain search region. Considering the edge common neighbor number introduced at Definition 6.2, we have the following lemma:

**Lemma 7.6.** Given a signed network $G$, a search region $(\kappa, \lambda, \epsilon)$ and $G = (V, E^+, E^-)$ is a subgraph of $G$, s.t.,

- $\forall (u, v) \in E^+ \leq \lambda \max \{d^+_L(u) + 2, \delta^-_L(u, v)\} \geq \kappa$ \lambda \max \{d^+_L(u) + 2, \delta^-_L(u, v)\} \geq \kappa$
- $\forall (u, v) \in E^- \leq \lambda \max \{d^-_L(u) + 1, \delta^-_L(u, v) + 1\} \geq \kappa$ \lambda \max \{d^-_L(u) + 1, \delta^-_L(u, v) + 1\} \geq \kappa$

Then, we can $C^*_i \subseteq \text{Algorithm of EdgeReduction}$.

**Based on Lemma 7.6, we propose EdgeReduction algorithm.** Given a signed network $G = (V, E^+, E^-)$, a search region $(\kappa, \lambda, \epsilon)$ and $G$, before the search starts, it removes the invalid edges that do not meet the requirements for the edge common neighbor number in Lemma 7.6 until no more edges can be removed.

---

The above text is a continuation of the discussion on algorithms and concepts for balanced clique computation in signed networks. The focus is on vertex and edge reduction variants, followed by the presentation of a theorem and a lemma related to space and edge complexity, respectively.
pruned. The pseudocode is omitted here. The time complexity of EdgeReduction^+ is $O(m^{1.5})$.

### 7.3.4 The Optimized Algorithm

Utilizing the above optimization strategies, i.e., coloring-based branch pruning, vertex dominance-based candidate pruning and vertex-edge reduction variants, we propose our optimized algorithm MBCSear^+ to search maximum balanced clique in a given search region. The pseudocode is shown at Algorithm 5. Given a search region $(k, \vec{x})$, for each search branch in this region, it first prunes candidates in $P_L$ and $P_R$ by invoking VertexReduction^+ algorithm (line 4). Then, the surviving candidate sets are judged to see whether it meets the size requirements (line 5–7). If the candidate sets and explored sets are both empty, we get a larger result and return it (line 8–11). Otherwise, Algorithm 5 tries to prune invalid branch by invoking ColoringPrune (line 12–13). After that, it chooses the pivot and distinguishes the four special cases for vertex dominance to further prune candidates (line 18–27). Then, it continuously search larger balanced clique in the remaining candidates by recursively calling itself. When all search branches are finished, Algorithm 5 can get the maximum balanced clique in the given search region.

**Algorithm 6. MBCSear-SSP^+** $(G = (V, E^+, E^-), k)$

1: compute degeneracy $\sigma$ of $G^+ = (V, E^+)$;
2: $e = 2k; \vec{k} = \vec{x} - \sigma + 1; \vec{r} = -1$;
3: while $\vec{r} \geq \vec{x}$ and $\vec{x} < \vec{r}$ do
4: $G' \leftarrow$ EdgeReduction^+ $(G, \epsilon, \vec{x}, \vec{r})$;
5: MBCSear$^+$ $(G', \epsilon)$;
6: $\vec{r} \leftarrow \vec{x}; \vec{x} \leftarrow \max \{\vec{x} - \vec{k}, \vec{k}\}; \vec{x} \leftarrow \max \{\frac{1}{2}, \vec{k}\};$

Based on MBCSear^+ algorithm, we are ready to propose the formal optimized algorithm MBCSear-SSP^+^, the pseudocode is shown at Algorithm 6. Given a signed network $G = (V, E^+, E^-)$ and size threshold $k$, for each search region $(\vec{x}, \vec{g})$, the algorithm first invokes EdgeReduction^+ to reduce the graph size by removing invalid edges before search start (line 4). Then, it invokes Algorithm 5 to search the maximum balanced clique in current search region (line 5). In the end, when finish the search in all search regions, it gets the maximum balanced clique $C^*$ in $G$ and terminates.

### 8 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.** For MBCE algorithms, they are 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.

For MBCS algorithms, they are MBCSear, MBCSear-SSP, MBCSear-01, MBCSear-02 and MBCSear-SSP^+. MBCSear is the baseline approach introduced at Section 7.1. MBCSear-SSP is proposed at Section 7.2. MBCSear-01 is the algorithm that utilizes the vertex dominance-based candidate pruning technique based on MBCSear-SSP. MBCSear-02 is the algorithm that utilizes the coloring-based branch pruning technique based on MBCSear-01. MBCSear-SSP^+ is the improved algorithm with three optimizations shown in Section 7.3. Note that, for fairness, we apply the EdgeReduction^+ proposed at Section 7.3.3 to all MBCS algorithms except MBCSear.

All algorithms are implemented in C++, using g++ compiler with -O3. If an algorithm cannot finish in 12 hours, we denote the processing time as INF.

**Real Datasets.** We evaluate our algorithms on nine real datasets. Slashdot and Epinions are signed networks in real world. AdjWordNet, DBLP and Douban are signed networks used in [28], [40], [62], we just follow the settings of these existing works. Specifically, AdjWordNet is a word network with synonyms and antonyms which is used in [28] and has been introduced in Section 1. DBLP is a co-authorship networks, where each vertex represents an author and each edge means that the two authors have co-authored at least one paper. An edge $(u, v)$ is a positive edge if the number of papers co-authored by $u$ and $v$ is greater than the threshold $t$ ($t = 2$ in our paper), otherwise, it’s a negative edge [40], [62]. Douban is a Chinese social network which allows users to contribute comments on movies. In the network, users have an average movie rating they give. The friendships between users are treated as positive edges. Then, if the difference of average movie rating between any two users is greater than 1, a negative edge is created between them [28]. For other datasets, we transfer them from unsigned network to signed network. In order to simulate the balanced clique as much as possible, the vertices in the graph are divided into two groups with a ratio of 4:1, the edges connecting vertices from the same group are positive edges, otherwise, they are negative edges. In this way, all the cliques in the original graph correspond to balanced cliques in the signed graph. For data source, AdjWordNet is downloaded from WordNet (https://wordnet.princeton.edu/). DBLP, Stack and Dpedia are downloaded from KONECT (http://konect.cc/). Douban is from authors in [63]. Other datasets are downloaded from SNAP (http://snap.stanford.edu). The details of each dataset are shown in Table 1.

**Table 1**

| 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 | 712,667 | 123,705 |
| DBLP        | 1,314,050 | 5,179,945 | 1,471,903 | 3,708,042 |
| Douban      | 1,588,565 | 13,918,375 | 9,034,357 | 4,883,838 |
| Pokec       | 1,632,803 | 30,622,564 | 15,179,203 | 7,022,761 |
| Stack       | 2,601,977 | 28,183,518 | 19,163,657 | 9,019,861 |
| Orkut       | 3,072,441 | 117,184,899 | 79,664,169 | 37,520,730 |
| Dpedia      | 18,268,992 | 126,890,209 | 86,002,736 | 40,887,473 |

8.1 The Performance of MBCE Algorithms

**Exp1: Efficiency of MBCE algorithms when varying k.** In this experiment, we evaluate the efficiency of three algorithms on all datasets with increasing $k$ from 4. On four large datasets Pokec, Stack, Orkut and Dpedia, we extend the maximum value of $k$ from 10 to 18 as maximal balanced cliques should be big on large datasets.

As shown in Fig. 5, baseline consumes the most time among three algorithms on all datasets when we vary $k$ and it can only handle the small datasets. 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 $k$. For instance, on Pokec, when $k = 6$, the running time of three algorithms are INF, 27451.7s and 89.9s, respectively. It’s because MBCEnum utilizes the in-enumeration optimization strategies, which demonstrates the effectiveness of optimization strategies.

Another phenomena shown in Fig. 5 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.

Exp-2: Memory consumption of MBCEnum algorithms when varying $k$. In this experiment, we compare the memory consumption between baseline, MBCEnum and MBCEnum' with different $k$ values.

As shown in Fig. 6, on both datasets, MBCEnum can sharply reduced the memory consumption compared with baseline on all $k$ values. For instance, on Epinions, when $k = 4$, the memory cost of three algorithms are 319MB, 79MB and 75MB, respectively. It’s because our MBCEnum and MBCEnum’ algorithms consider the uniqueness of signed graphs and enumerate the results on the graphs directly to avoid spending large memory to store massive mid-results (like maximal cliques in $G^+$).

Exp-3: Scalability of MBCEnum and MBCEnum’, $k=4$. 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 dataset 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.

8.2 The Performance of MBCE Algorithms

Exp-5: Efficiency of MBCE algorithms when varying $k$. To evaluate the efficiency of five MBCE algorithms, we record the running time of them on eight datasets, $k = [2 \to 10]$.

| TABLE 2                          |
|----------------------------------|
| Case study on AdjWordNet         |

| $C_L$               | $C_R$                  |
|---------------------|------------------------|
| raw, rough, rude    | refined, smooth, suave |
| relaxing, reposeful | relaxed, uneasy, ungratified |
| restful             | unsatisfied            |
| interior, internal, | away, foreign, outer, |
| intimate            | outside, remote        |
| assumed, false, fictitious | factual, genuine, |
| fictive             | literal, real         |
| mistaken, off-key   | tangible, touchable,   |
| pretended, put-on,  | true, truthful,        |
| sham, sour          | unfeigned, veridical   |
| untrue              | adynamic, asthenic,    |
| active              | debilitated, enervated |
| animated, combat-ready | undynamic, stagnant   |
| fighting            | light                  |
| participating       | ahead, in-the-lead,    |
| alive               | leading, preeminent,   |
| live                | prima, star,           |
| following, undermentioned | starring, stellar   |
| next                | cherished, treasured,  |
| undesirable, unsuitable | wanted, precious     |
As shown in Fig. 8, with the increasing of $k$, the running time of five algorithms decreases on most datasets. It’s because with large $k$, more balanced cliques can be neglected during the MBCS process. Moreover, MBCSear-SSP is faster than MBCSear, MBCSear-SSP* is the fastest algorithm among them. MBCSear-01 and MBCSear-02 perform as well as MBCSear-SSP* on small datasets such as Slashdot, Epinions and DBLP. On large datasets, MBCSear-01 can handle Pokec and Stack for all $k$ values, and can only handle Orkut for large $k$ ($k \geq 8$). On the other hand, MBCSear-02 can handle all datasets and is more efficient than MBCSear-01. MBCSear-SSP* is still the most efficient algorithm among them. For instance, on Stack when $k = 10$, the running time of five algorithms are INF, 29079.2s, 11394.8s, 815.7s and 655.2s, respectively. It’s because that MBCSear has to search the maximum balanced clique in the whole graph, while MBCSear-SSP utilizes the search space partition strategy to search the result within narrow search space. Based on MBCSear-SSP, MBCSear-01 can prune invalid candidates during the search process by the vertex dominance-based optimization. MBCSear-02 can further terminate the unnecessary search branches early due to the vertex color-based optimization. By applying all three optimization optimizations, MBCSear-SSP* can further reduce the search space, and thus it is the most efficient algorithm among them.

Exp-6: Effectiveness of MBCS Algorithms When Varying $k$. To intuitively compare the effectiveness of three MBCS algorithms, in this experiment, we record the amount of calculation of three algorithms on eight datasets, $k = [2 - 5]$. The calculation quantity is the time of invoking MBCSearUtil and MBCSearUtil*, which can intuitively represent the search space of different algorithms.

As shown in Fig. 9a, when $k = 2$, on all datasets, the calculation quantity of MBCSear-SSP* is much less than that of other algorithms. Meanwhile, MBCSear-SSP’s calculation quantity is less than MBCSear’s. For instance, on DBLP (DB), the calculation quantity of MBCSear, MBCSear-SSP and MBCSear-SSP* are 155,621, 328 and 183, respectively. When $k > 2$, the trend is similar. It’s because MBCSear-SSP* and MBCSear-SSP are based on search space partitions, which can reduce the total search space effectively. The experimental results also confirm the reason for the efficiency of MBCSear-SSP* at Exp-5.

Exp-7: Search Process of MBCSear-SSP* on Real Datasets. In this experiment, we show the search process of MBCSear-SSP* algorithm on Douban and Pokec datasets, $k=2$. Table 3 shows every search region $(E, k)$, the size of its input graph $G'$ including positive edges number $|E^+_G|$ and negative edges number $|E^-_G|$, the ratio of $G'$ in the original graph $G$, and the maximum balanced clique size $\epsilon$ found so far. Since Douban and Pokec are big datasets, the search process is time consuming, hence, MBCSear-SSP* adopts $Dec(E) = \frac{E}{|E|}$.

As Table 3 shows, on Douban, the first search region $(E_0, k_0)$ is $(2,78)$, and $\epsilon$ is initialized as 4. MBCSear-SSP* first invokes EdgeReduction* to pre-reduce useless edges in $G$. Due to the large value of $\epsilon_0$, all edges are pruned from $G$. Hence, $G'$ is empty here. For the second search region $(2,39)$, $G'$ is still empty. For the third search region $(2,20)$, $G'$ has 21,695 positive edges and 12,190 negative edges, it only holds 0.24% edges of the original graph which is much less than $G$. Then, MBCSear-SSP* finds the maximum balanced

| TABLE 3 |

The Search Process of MBCSear-SSP* on Douban, $k=2$

| Index | Search Region | $|E^+_G|$ | $|E^-_G|$ | $G'/G(\%)$ | $\epsilon$ |
|-------|---------------|-----------|-----------|-------------|-----------|
| 0     | (2,78)        | 0         | 0         | 0           | 4         |
| 1     | (2,39)        | 0         | 0         | 0           | 4         |
| 2     | (2,20)        | 21,695    | 12,190    | 0.24        | 33        |
| 3     | (13,13)       | 0         | 0         | 0           | 33        |

| TABLE 3 |

The Search Process of MBCSear-SSP* on Pokec, $k=2$

| Index | Search Region | $|E^+_G|$ | $|E^-_G|$ | $G'/G(\%)$ | $\epsilon$ |
|-------|---------------|-----------|-----------|-------------|-----------|
| 0     | (2,57)        | 0         | 0         | 0           | 4         |
| 1     | (2,19)        | 6,605     | 2,521     | 0.30        | 29        |
| 2     | (10,10)       | 0         | 0         | 0           | 29        |
clique $C^*$ on $G'$. The size of $C^*$ is 33. For the last search region (13,13), $G'$ is empty, the search process is finished. The search process on Pokec is similar.

By observing the search process of MBCSear-SSP on the two datasets, We find two significant phenomena. First, benefited from the search space partition paradigm, the number of search regions is limited. Second, the input graph $G'$ for each search region is much smaller than the original graph, because the edge reduction strategy can remove most of the invalid edges before the search starting.

Exp-8: Scalability of MBCS algorithms. In this experiment, we evaluate the scalability of MBCS algorithms on two biggest dataset as Exp-3. The results are shown in Fig. 10.

With the number of vertices increases, the running time of three algorithms increases as well. Among them, the growth rate of MBCSear-SSP is the most stable. For instance, on Dbpedia with 40% vertices, MBCSear cannot get result within a reasonable time, the running time of MBCSear-SSP and MBCSear-SSP are 11953.0s and 319.9s, respectively. On Dbpedia with more than 40% vertices, only MBCSear-SSP can get result within a reasonable time. The trend of running time on Orkut is similar. Therefore, MBCSear-SSP can scale to large-scale graphs.

9 CONCLUSION

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. Besides, we study the maximum balanced clique search problem, and propose a novel search space partition-based search framework. Moreover, we explore multiple optimization strategies to further reduce the search space during search process. The experimental results on real datasets demonstrate the efficiency, effectiveness and scalability of our solutions.

REFERENCES

[1] J. Pattillo, N. Youssef, and S. Butenko, “On clique relaxation models in network analysis,” Eur. J. Oper. Res., vol. 226, no. 1, pp. 9–18, 2013.
[2] L. Meng, L. Yuan, Z. Chen, X. Lin, and S. Yang, “Index-based structural clustering on directed graphs,” in Proc. IEEE Int. Conf. Data Eng., pp. 2831–2841.
[3] C. Bron and J. Kerbosch, “Finding all cliques of an undirected graph (algorithm 457),” Commun. ACM, vol. 16, no. 9, pp. 575–576, 1973.
[4] L. Yuan, L. Qin, X. Lin, L. Chang, and W. Zhang, “Diversified top-k clique search,” Int. J. Very Large Data Bases, vol. 25, no. 2, pp. 171–196, 2016.
[5] D. Eppstein, M. Löffler, and D. Strash, “Listing all maximal cliques in sparse graphs in near-optimal time,” in Proc. Int. Symp. Algorithms Comput., 2010, pp. 403–414.
[6] D. Eppstein and D. Strash, “Listing all maximal cliques in large sparse real-world graphs,” in Proc. Int. Symp. Exp. Algorithms, 2011, pp. 364–375.
[7] J. Zhang, L. Yuan, W. Li, L. Qin, and Y. Zhang, “Efficient label-constrained shortest path queries on road networks: A tree decomposition approach,” Proc. VLDB Endowment, vol. 15, no. 3, pp. 686–698, 2021.
[8] D. Easley and J. Kleinberg, Networks, Crowds, and Markets: Reasoning About a Highly Connected World, Cambridge, U.K.: Cambridge Univ. Press, 2010.
[9] S. Kumar, F. Spezzano, V. S. Subrahmanian, and C. Faloutsos, “Edge weight prediction in weighted signed networks,” in Proc. IEEE Int. Conf. Data Mining, 2016, pp. 221–230.
[10] J. Kunegis, A. Lommatzsch, and C. Bauckhage, “The slashdot zoo: Mining a social network with negative edges,” in Proc. World Wide Web, 2009, pp. 741–750.
[11] J. Leskovec, D. Huttenlocher, and J. Kleinberg, “Signed networks in social media,” in Proc. SIGCHI Conf. Hum. Factors Comput. Syst., 2010, pp. 1361–1370.
[12] L. Ou-Yang, D.-Q. Dai, and X.-F. Zhang, “Detecting protein complexes from signals by protein interaction networks,” IEEE/ ACM Trans. Comput. Biol. Bioinf., vol. 12, no. 6, pp. 1333–1344, Nov./Dec. 2015.
[13] D. Cartwright and F. Harary, “Structural balance: A generalization of heider’s theory,” Psychol. Rev., vol. 63, no. 5, 1956, Art. no. 277.
[14] S. A. Marvel, S. H. Strogatz, and J. M. Kleinberg, “Energy landscape of social balance,” Phys. Rev. Lett., vol. 103, no. 19, 2009, Art. no. 198701.
[15] P. Abell and M. Ludwig, “Structural balance: A dynamic perspective,” J. Math. Socid., vol. 33, no. 2, pp. 129–155, 2009.
[16] Ş. A. Marvel, J. Kleinberg, R. D. Kleinberg, and S. H. Strogatz, “Continuous-time model of structural balance,” Proc. Nat. Acad. Sci., vol. 108, no. 5, pp. 1771–1776, 2011.
[17] T. Derr, C. Aggarwal, and J. Tang, “Signed network modeling based on structural balance theory,” in Proc. ACM Int. Conf. Inf. Knowl. Manage., 2018, pp. 557–566.
[18] F. Heider, “Attitudes and cognitive organization,” J. Psychol., vol. 21, no. 1, pp. 107–112, 1946.
[19] F. Harary et al., “On the notion of balance of a signed graph,” Michigan Math. J., vol. 2, no. 2, pp. 143–146, 1953.
[20] L. Chang, “Efficient maximum clique computation and enumeration over large sparse graphs,” VLDB J., vol. 29, no. 5, pp. 999–1022, 2020.
[21] C. L. J. X. Yu, H. Wei, and Y. Zhang, “Finding the maximum clique in massive graphs,” Proc. VLDB Endowment, vol. 10, no. 11, pp. 1538–1549, 2017.
[22] J. Chen et al., “NuQClq: An effective local search algorithm for maximum quasi-clique problem,” in Proc. AAAI Conf. Artif. Intell., 2021, pp. 12299–12306.
[23] B. Lyu, L. Qin, X. Lin, Y. Zhang, Z. Qian, and J. Zhou, “Maximum biclique search at billion scale,” Proc. VLDB Endowment, vol. 13, no. 13, pp. 1359–1372, 2020.
[24] A. Zhou, Y. Wang, and L. Chen, “Finding large diverse communities on networks: The edge maximum k-partite clique,” Proc. VLDB Endowment, vol. 13, no. 11, pp. 2576–2589, 2020.
[25] E. Sevinc, and T. Dökeroglu, “A novel parallel local search algorithm for the maximum vertex weight clique problem in large graphs,” Soft Comput., vol. 4, no. 5, pp. 3551–3567, 2020.
[26] R. Li, X. Wu, H. Liu, J. Wu, and M. Yin, “An efficient local search for the maximum edge weighted clique problem,” IEEE Access, vol. 6, pp. 10743–10753, 2018.
[27] X. Song, Y. Chi, K. Hino, and B. Tseng, “Identifying opinion leaders in the blogosphere,” in Proc. ACM Int. Conf. Inf. Knowl. Manage., 2007, pp. 971–974.
[28] L. Chu, Z. Wang, J. Pei, J. Wang, Z. Zhao, and E. Chen, “Finding gangs in war from signed networks,” in Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Discov. Data Mining, 2016, pp. 1505–1514.
[29] R. Axelrod and D. S. Bennett, “A landscape theory of aggregation,” Brit. J. Political Sci., vol. 23, no. 02, pp. 211–233, 1993.
[30] M. Isoraiti, “Importance of strategic alliances in company’s activity,” Brit. J. Political Sci., vol. 1, no. 5, pp. 39–46, 2009.
[31] R. Müller, “Wordnet: A lexical database for english,” Commun. ACM, vol. 38, no. 11, pp. 39–46, 1995.
[32] V. Kumar, N. Joshi, A. Mukherjee, G. Ramakrishnan, and P. Jothi, “Cross-lingual training for automatic question generation,” in Proc. Conf. Assoc. Comput. Linguistics, 2019, pp. 4863–4872.
[33] A. Krishnan, D. P. S. Ranu, and S. Mehta, “Leveraging semantic resources in diversified query expansion,” World Wide Web, vol. 21, no. 4, pp. 1041–1067, 2018.

[34] F. Harary, M.-H. Lim, and D. C. Wunsch, “Signed graphs for portfolio analysis in risk management,” IMA J. Manage. Math., vol. 13, no. 2, pp. 201–210, 2002.

[35] R. M. V. de Figueiredo and Y. Frota, “The maximum balanced subgraph of a signed graph: Applications and solution approaches,” Eur. J. Oper. Res., vol. 236, no. 2, pp. 473–487, 2014.

[36] S. Kumar, W. L. Hamilton, J. Leskovec, and D. Jurafsky, “Community interaction and conflict on the web,” in Proc. World Wide Web Conf., 2018, pp. 933–943.

[37] L. Yuan, L. Qin, X. Lin, L. Chang, and W. Zhang, “Diversified top-k query,” in Proc. VLDB Conf., vol. 25, no. 2, pp. 171–196, 2016.

[38] Z. Chen, L. Yuan, X. Lin, L. Qin, and J. Yang, “Efficient maximal balanced clique enumeration in signed networks,” in Proc. World Wide Web Conf., 2020, pp. 339–349.

[39] X. Zheng and D. Zeng, “Social balance in signed networks,” Inf. Syst. Front., vol. 17, no. 5, pp. 1077–1095, 2015.

[40] R.-H. Li et al., “Efficient signed clique search in signed networks,” IEEE Trans. Knowl. Data Eng., 2018, pp. 245–256.

[41] F. Hao, S. S. Yau, G. Min, and L. T. Yang, “Detecting k-balanced trusted cliques in signed social networks,” IEEE Internet Comput., vol. 18, no. 2, pp. 24–31, 2014.

[42] L. Yuan, L. Qin, X. Lin, L. Chang, and W. Zhang, “Effective and efficient dynamic graph coloring,” Proc. VLDB Endowment, vol. 11, no. 3, pp. 338–351, 2017.

[43] A. Ouyang and E. A., “The enumeration of maximal cliques of large graphs,” SIAM J. Comput., vol. 2, no. 1, pp. 1–6, 1973.

[44] D. Ouyang, L. Yuan, L. Qin, L. Chang, and Y. Zhang, “Efficient shortest path index maintenance on dynamic road networks with theoretical guarantees,” Proc. VLDB Endowment, vol. 13, no. 5, pp. 602–615, 2020.

[45] C. Ma, L. Yuan, L. Han, M. Ding, R. Bhaskar, and J. Li, “Data level privacy preserving: A stochastic perturbation approach based on differential privacy,” IEEE Trans. Knowl. Data Eng., early access, 2021, doi: 10.1109/TKDE.2021.3137047.

[46] E. Tomita, A. Tanaka, and H. Takahashi, “The worst-case time complexity for generating all maximal cliques and computations: Experiments,” Theor. Comput. Sci., vol. 363, no. 1, pp. 28–42, 2006.

[47] L. Yuan, L. Qin, W. Zhang, L. Chang, and J. Yang, “Index-based densest clique percolation community search in networks,” IEEE Trans. Knowl. Data Eng., vol. 30, no. 5, pp. 922–935, May 2018.

[48] F. Fakhfakh, M. Tounsi, M. Mosbah, and A. H. Kacem, “Algorithms for finding maximal and maximum cliques: A survey,” in Proc. Int. Conf. Intel. Syst. Des. Appl., 2017, pp. 745–754.

[49] S. B. Seidman, “Network structure and minimum degree,” Soc. Netw., vol. 5, no. 3, pp. 269–287, 1983.

[50] B. Liu, L. Yuan, X. Lin, L. Qin, W. Zhang, and J. Zhou, “Efficient (α, β)-core computation: An index-based approach,” in Proc. World Wide Web, 2019, pp. 1130–1141.

[51] J. Cohen, “Trusses: Cohesive subgraphs for social network analysis,” Nat. Secur. Agency, Maryland, USA, Tech. Rep., 2008.

[52] Z. Chen, L. Yuan, L. Han, and Z. Qian, “Higher-order truss decomposition in graphs,” IEEE Trans. Knowl. Data Eng., Dec. 23, 2021, doi: 10.1109/TKDE.2021.3137955.

[53] R. Zhou, C. Liu, J. X. Yu, W. Liang, B. Chen, and J. Li, “Finding maximal k-edge-connected subgraphs from a large graph,” in Proc. 15th Int. Conf. Extending Database Technol., 2012, pp. 480–491.

[54] A. E. Sarıyüce, C. Seshadhri, A. Pinar, and U. V. Catalyurek, “Finding the hierarchy of dense subgraphs using nucleus decompositions,” in Proc. World Wide Web, 2015, pp. 927–937.

[55] A. E. Sarıyüce, C. Seshadhri, A. Pinar, and U. V. Catalyurek, “Nucleus decompositions for identifying hierarchy of dense subgraphs,” ACM Trans. Web, vol. 11, no. 3, pp. 16:1–16:27, 2017.

[56] J. Cheng, L. Zhu, Y. Ke, and S. Chu, “Fast algorithms for maximal clique enumeration with limited memory,” in ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2012, pp. 1240–1248.

[57] M. C. Schmidt, N. F. Samatova, K. Thomas, and B.-H. Park, “A scalable, parallel algorithm for maximal clique enumeration,” J. Parallel Distrib. Comput., vol. 69, no. 4, pp. 417–428, 2009.