Maximal Structural Balanced k-plex Enumeration in Signed Graphs

Kai Yao
The University of Sydney
kai.yao@sydney.edu.au

Xinchen Zhang
The University of Sydney
xzha9245@uni.sydney.edu.au

Lijun Chang
The University of Sydney
lijun.chang@sydney.edu.au

Abstract*—Signed graphs, which represent interactions as friendly (positive edges) or antagonistic (negative edges), offer a powerful model to capture novel and interesting structural properties of real-world phenomena. Recently, the structural balanced clique model has been proposed to identify polarized structures in signed graphs. A subgraph is a structural balanced clique if every pair of its vertices is connected by an edge and its vertices can be partitioned into two sets such that all edges inside each partition are positive and all cross-partition edges are negative. However, this model's rigidity restricts its applicability in some real signed graphs, causing the omission of many interesting patterns. In this paper, we formulate the structural balanced k-plex model, by relaxing the structural balanced clique model to allow each vertex missing a few edges. We prove that the problem of enumerating all maximal balanced k-plexes is #P-hard. To efficiently identify all maximal structural balanced k-plexes, we introduce the backtracking algorithm MBPE-BK and propose several optimization techniques to enhance its practical performance. However, MBPE-BK's efficiency is hindered by overlapping candidate sets. To overcome this challenge, we propose a novel algorithm MBPE with better time complexity than MBPE-BK (i.e., $\mathcal{O}^*(2^\delta)$ v.s. $\mathcal{O}^*(3^{\delta D})$). We also propose subgraph reduction and partition-based vertex reduction to improve its efficiency. Finally, we adopt the minimum-degree branching strategy to improve the worst-case time complexity of MBPE to $\mathcal{O}^*(\alpha_k^\delta)$, where $1 < \alpha_k < 2$ is a constant that depends only on k. Extensive experiments on both real-world and synthetic signed graphs are conducted to demonstrate the effectiveness and efficiency of our model and algorithms.

I. INTRODUCTION

Signed graphs provide an enriched representation of conventional graphs by incorporating the concept of *polarity* to signify relationships between entities/vertices using *positive* and *negative* edge signs [1]. This enhancement enables the capture of various types of relationships in different domains. For instance, in social networks, signed graphs can effectively model friend-foe relationships [2], while in opinion networks, they represent support-dissent opinions [3]. Trust networks can be analyzed to understand trust-distrust relationships [4], and protein-protein interaction networks can be studied in terms of activation-inhibition interactions [5].

In the literature, many traditional graph analysis tasks have been extended to signed graphs by treating positive and negative edges differently, such as community detection [6], [7], link prediction [8], [9] and recommendation systems [10], [11]. Among them, the problem of detecting structural balanced cliques receives an increasing attention recently [12]–[16]. A graph is a (structural) balanced clique if (1) it is a clique (i.e., every pair of its vertices is connected by an edge), and (2) it is structural balanced. The *structural balance theory* [17] states that a signed (sub)graph is structural balanced if its vertices can be partitioned into two sets such that all edges inside each partition have positive signs and all cross-partition edges have negative signs. That is, "the friend of my friend is my friend", and "the friend of my enemy is my enemy". Identifying balanced cliques holds significant relevance in various practical domains, including polarized community discovery [7], [18], conflict discovery in social networks [19], identification of synonym-antonym groups in word networks [6], and detecting protein complexes in protein-protein interaction networks [20].

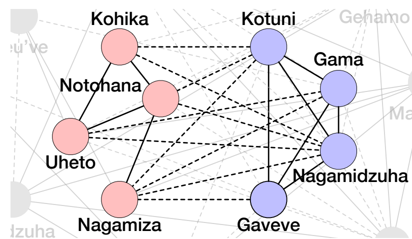


Fig. 1. Polarized tribes in Tribes [21]

However, the balanced clique concept is often too restrictive for applications, in view of the existence of data noise in real-world signed graphs. As a result, significant polarized structures in many signed graphs hardly appear as balanced cliques. As an illustration, Figure 1 presents a snippet of the real signed network Tribes [21], which depicts the friendship (solid lines) and enmity (dashed lines) among 16 tribes situated in the Eastern Central Highlands of New Guinea. In Figure 1, the red vertices and blue vertices exhibit a polarized structure, wherein each tribe holds a clear standpoint, fostering friendships with their allies while maintaining enmity towards hostile tribes. This results in a well-defined and polarized structure within the network. However, this polarized structure cannot be captured by the balanced clique model due to its stringent requirements. In fact, the largest balanced clique identified in Tribes consists of only 4 tribes, which is comparatively small and may not offer substantial insights for analysis.

In this paper, we formulate the structural balanced k-plex model, by relaxing the balanced clique model to allow each vertex missing a few edges. Specifically, given a signed graph G, two disjoint vertex subsets P_L and P_R induce a structural balanced k-plex if each vertex $u \in P_L \cup P_R$ has up to $k-1$ missing or wrong-signed adjacent edges. An edge is said to be *wrong-signed* w.r.t the partitioning (P_L, P_R) if (1) it has a negative sign and its two end-points are from the same partition, or (2) it has a positive sign and its two end-points are from different partitions. For simplicity, we refer to structural balanced k-plexes as *balanced k-plexes*, and denote it by (P_L, P_R). The polarized structure in Figure 1 (i.e., the red and blue vertices and their connections) is a balanced 4-plex, since each member has up to 3 missing or wrong-signed adjacent edges. Consider *Kohika*, it has missing edges to *Nagamiza*, *Gama*, and *Gaveve*. This example shows that the balanced k-plex model can unveil interesting patterns that are not apparent when using the balanced clique model.

We prove that the problem of enumerating all maximal balanced k-plexes is #P-hard. Nevertheless, we aim to solve this problem efficiently in practice in view of its intriguing applications. We first propose a baseline algorithm MBPE-BK, by drawing inspiration from the well-known backtracking algorithm, Bron-Kerbosch, that is initially proposed in [22] for enumerating all maximal cliques in unsigned graphs. The main idea of MBPE-BK is that, for each vertex v_i in the graph, we iteratively build up a partial balanced k-plex $P = (P_L, P_R)$, where $P_L = \{v_i\}$ and $P_R = \emptyset$ initially. In addition, we maintain two candidate sets C_L and C_R of vertices that are used for growing P_L and P_R, respectively. Then, we iteratively try each vertex of C_L to be added to P_L and each vertex of C_R to be added to P_R, to grow the solution and conduct the recursion. To improve the efficiency, we propose several optimization techniques including smaller candidate sets, vertex/edge reduction and pivoting technique. However, despite incorporating all these optimizations, the performance of MBPE-BK remains unsatisfactory. This is primarily due to the heavy overlap between the candidate sets C_L and C_R, i.e., $C_L \cap C_R \neq \emptyset$. To overcome this issue, we then propose the algorithm MBPE that adopts a different strategy at the root level of the backtracking search tree. Specifically, for each vertex v_i in the graph, we exhaustively enumerate all possible configurations of v_i's missing and wrong-signed adjacent edges. For each such configuration, the partial balanced k-plex $P = (P_L, P_R)$ is initialized by v_i and the other end-points of these edges; the candidate sets C_L and C_R are initialized as v_i's positive neighbors and negative neighbors, respectively. Then, we conduct a similar backtracking search as MBPE-BK to identify the maximal balanced k-plexes containing v_i. Through pre-determining the missing and wrong-signed adjacent edges of v_i, the issue of overlapping candidate sets is resolved; that is, it is guaranteed that $C_L \cap C_R = \emptyset$. To further improve the practical performance of MBPE, we propose a subgraph reduction technique by exploiting the fact that $C_L \cap C_R = \emptyset$, and propose the partition-based vertex reduction technique to reduce the number of exhaustively enumerated configurations. We prove that MBPE achieves a better theoretical time complexity than MBPE-BK (i.e., $\mathcal{O}^*(2^\delta)$ v.s. $\mathcal{O}^*(3^{\delta D})$). [1] Finally, we adopt the *minimum-degree pivoting* strategy from the k-plex enumeration literature [23]; that is, the minimum-degree vertex of $C_L \cup C_R$ is chosen as the branching vertex to generate new branches in the backtracking. We prove that this further improves the worst-case time complexity of MBPE to $\mathcal{O}^*(\alpha_k^\delta)$, where $1 < \alpha_k < 2$ is a constant depending only on k, e.g., $\alpha_2 = 1.839$, $\alpha_3 = 1.928$, and $\alpha_4 = 1.966$.

[1] For presentation simplicity, we also use the $\mathcal{O}^*$ notation, in addition to the usual $\mathcal{O}$ notation, for time complexity analysis. $\mathcal{O}^*$ notation focuses on the exponential part of the time complexity by hiding polynomial factors.

Our main contributions are summarized as follows.

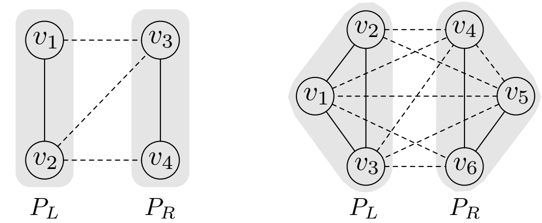
- We formalize the balanced k-plex model in signed graphs based on the structural balance theory, and prove that enumerating all maximal balanced k-plexes is #P-hard.
- We propose a basic backtracking algorithm MBPE-BK as well as several optimization techniques to improve its practical performance.
- We propose a novel algorithm MBPE with better time complexity than MBPE-BK (i.e., $\mathcal{O}^*(2^\delta)$ v.s. $\mathcal{O}^*(3^{\delta D})$). We also propose subgraph reduction and partition-based vertex reduction to improve its practical performance.
- We adopt the minimum-degree branching strategy to improve the worst-case time complexity of MBPE to $\mathcal{O}^*(\alpha_k^\delta)$, where $1 < \alpha_k < 2$ is a constant that depends only on k, e.g., $\alpha_2 = 1.839$, $\alpha_3 = 1.928$, and $\alpha_4 = 1.966$.
- Extensive experiments on real-world signed graphs demonstrate the efficiency of our algorithms and effectiveness of our balanced k-plex model.

Organization. The rest of the paper is organized as follows. Section II provides preliminaries and defines the problems studied in this paper. Section III presents our baseline algorithm MBPE-BK and its optimizations. Section IV proposes the advanced algorithm MBPE and a novel branching strategy. Experimental results are reported in Section V. Related works are discussed in Section VI. Finally, Section VII concludes the paper. All proofs are omitted due to space limitation.

II. Preliminaries

In the paper, we focus on undirected signed graphs $G = (V, E^+, E^-)$, where V is the set of vertices, and E^+ and E^- are the sets of positive and negative edges, respectively. The numbers of vertices and edges in G are denoted by $n = |V|$ and $m = |E^+| + |E^-|$. For each vertex $v \in V$, $N_G^+(v)$ denotes the positive neighbors of v, i.e., $N_G^+(v) = \{u \in V \mid (v, u) \in E^+\}$, and $N_G^-(v)$ denotes the negative neighbors of v, i.e, $N_G^-(v) = \{u \in V \mid (v, u) \in E^-\}$, $N_G(v) = N_G^-(v) \cup N_G^+(v)$ denotes neighbors of v. Additionally, we use $d_G^+(v) = |N_G^+(v)|$, $d_G^-(v) = |N_G^-(v)|$, and $d_G(v) = |N_G(v)|$ to respectively denote the positive degree, negative degree, and degree of vertex v. Given a vertex subset $S \subseteq V$, we use $G[S]$ to denote the induced subgraph of G that consists of all edges between vertices of S, i.e., $G[S] = (S, \{(u, v) \in E^+ \cup E^- \mid u, v \in S\})$. We use $N_G^2(v)$ to denote 2-hop neighbors of v, i.e., the set of vertices with distance exactly 2 to v in G. When the context is clear, we omit the subscript G.

Definition 1 (Structural Balanced Graph [17])**.** A signed graph $G = (V, E^+, E^-)$ is *structural balanced* if its vertices can be partitioned into two sets V_L and V_R such that all edges



(a) Strict structural balanced 2-plex (b) Structural balanced 2-plex

Fig. 2. Illustration of (strict) structural balanced k-plex

between vertices in the same set are positive and all edges between vertices from different sets are negative.

Definition 2 (k-Plex [24])**.** A signed graph $G = (V, E^+, E^-)$ is a *k-plex* if for each $u \in V$, its degree satisfies $d_G(u) \geq |V| - k$, i.e., u misses edges to at most k vertices (including u itself).

Definition 3 (Strict Structural Balanced k-Plex)**.** Given a signed graph $G = (V, E^+, E^-)$, two disjoint vertex subsets P_L and P_R induce a *strict structural balanced k-plex* if the subgraph of G induced by $P_L \cup P_R$ is structural balanced and is a k-plex.

For the signed graph in Figure 2(a), $P_L = \{v_1, v_2\}$ and $P_R = \{v_3, v_4\}$ form a strict structural balanced 2-plex since the subgraph induced by $P_L \cup P_R$, i.e., $G[P_L \cup P_R]$, is structural balanced and is a 2-plex. However, this model adheres strictly to the structural balance theory. Next, we propose a new model that further relaxes this constraint, making it more suitable for practical applications.

Given two disjoint vertex subsets P_L and P_R in a signed graph G, we define the *maximal structural balanced subgraph of G induced by* (P_L, P_R) as the subgraph of G whose vertices are $P_L \cup P_R$ and whose edges consist of (1) all positive edges of G between vertices of P_L, (2) all positive edges between vertices of P_R, and (3) all negative edges between vertices of P_L and vertices of P_R (i.e., removing edges that violate structural balance). For convenience we denote the maximal structural balanced subgraph induced by (P_L, P_R) as $g(P_L, P_R)$. It is easy to see that any $g(P_L, P_R)$ is structural balanced.

Definition 4 (Structural Balanced k-Plex)**.** Given a signed graph $G = (V, E^+, E^-)$, two disjoint vertex subsets P_L and P_R induce a *structural balanced k-plex* if the maximal structural balanced subgraph of G induced by P_L and P_R, i.e., $g(P_L, P_R)$, is a k-plex.

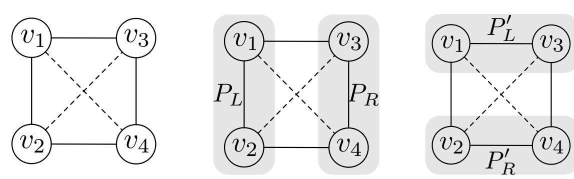
For example, consider the signed graph G in Figure 2(b). The maximal structural balanced subgraph of G induced by $P_L = \{v_1, v_2, v_3\}$ and $P_R = \{v_4, v_5, v_6\}$ contains all edges except (v_4, v_5); it is easy to see that this subgraph is a 2-plex. Therefore, ($P_L = \{v_1, v_2, v_3\}$,$P_R = \{v_4, v_5, v_6\}$) can be identified as a structural balanced 2-plex. The concept of a structural balanced k-plex (Definition 4) serves as a generalization of the strict structural balanced k-plex (Definition 3). By permitting a vertex to have a limited number of negative links to vertices on the same side and positive links to vertices on the opposite side, this model is more suited to practical applications. We will also evaluate the performance of these two models in the experimental part.

For simplicity, we refer to (P_L, P_R) as a *balanced k-plex* if P_L and P_R induce a structural balanced k-plex. We call P_L and P_R the two sides (e.g., left and right) of the balanced k-plex. For a balanced k-plex, the role of P_L and P_R can be swapped, i.e., (P_L, P_R) is same as (P_R, P_L). We measure the size of balanced k-plex by its number of vertices, i.e., $|P_L \cup P_R|$. The property of balanced k-plex is *hereditary*; that is, given a balanced k-plex (P_L, P_R), (P'_L, P'_R) is also a balanced k-plex for any $P'_L \subseteq P_L$ and $P'_R \subseteq P_R$. A balanced k-plex (P_L, P_R) is *maximal* if adding any vertex to either P_L or P_R would violate the balanced k-plex property.

In this paper, we aim to enumerate all maximal balanced k-plexes (P_L, P_R) in a given signed graph. Moreover, to avoid generating trivial solutions, we require the numbers of vertices in P_L and in P_R to be no less than a user-given threshold τ, i.e., $|P_L| \geq \tau$ and $|P_R| \geq \tau$. We also require $\tau \geq k$ such that each vertex v in the balanced k-plex possesses a sufficient number of neighbors (i.e., positive neighbors in the same side with v and negative neighbors in the opposite side with v). By satisfying this condition, we ensure that every member of the balanced k-plex holds a clear standpoint, resulting in a well-defined and balanced structure for the entire solution. The studied problem is formulated as follows.

Problem 1 (Maximal Balanced k-Plex Enumeration Problem)**.** Given a signed graph G and two positive integers k and τ such that $\tau \geq k$, the maximal balanced k-plex enumeration problem aims to report all maximal balanced k-plexes (P_L, P_R) in G s.t. $|P_L| \geq \tau$ and $|P_R| \geq \tau$.

Theorem 1. *The maximal balanced k-plex enumeration problem is #P-hard.*



(a) A signed graph (b) Case I (c) Case II

Fig. 3. Uniqueness of the balanced k-plex

Uniqueness of the balanced k-plex. It is noteworthy that two distinct k-plexes, denoted as (P_L, P_R) and (P'_L, P'_R), can share the same set of vertices, meaning $P_L \cup P_R = P'_L \cup P'_R$. For instance, in the signed graph illustrated in Figure 3(a), both ($P_L = \{v_1, v_2\}, P_R = \{v_3, v_4\}$) and ($P'_L = \{v_1, v_3\}, P'_R = \{v_2, v_4\}$) represent balanced 2-plexes, yet they consist of the same vertices. In this study, we regard these as distinct balanced k-plexes since different partitions imply different perspectives of the vertices. To gain a deeper understanding of balanced k-plexes, the following lemma demonstrates that when two distinct balanced k-plexes have the same set of vertices, certain constraints are imposed.

Lemma 1. *Given two different balanced k-plexes (P_L, P_R) and (P'_L, P'_R), if $P_L \cup P_R = P'_L \cup P'_R$, we must have $|P_L \cup P_R| \leq 4k - 4$.*

III. A Bron-Kerbosch-based Algorithm

The classic Bron-Kerbosch algorithm is proposed in [22] for enumerating all maximal cliques in unsigned graphs. In this section, we adopt the Bron-Kerbosch framework to solve the maximal balanced k-plex enumeration problem. The main idea is to iteratively build up two sets P_L and P_R such that (P_L, P_R) is a balanced k-plex. In addition, we maintain two candidate sets C_L and C_R of vertices that are used for growing P_L and P_R, respectively. Then, we iteratively try each vertex of C_L to be added to P_L and each vertex of C_R to be added to P_R, to grow the partial balanced k-plex and conduct the recursion. We also maintain exclusive sets X_L and X_R to record the vertices that have been processed to avoid generating duplicate solutions.

Before introducing the details of our algorithm, we analyze two properties of balanced k-plex that are important for our algorithm design.

Lemma 2. *Given a signed graph G, if the size of a balanced k-plex (P_L, P_R) is at least $2k-1$, the diameter of the subgraph of G induced by $P_L \cup P_R$ is bounded by 2.*

We categorize the 2-hop neighbors of v into four distinct types: positive-positive, positive-negative, negative-positive, and negative-negative neighbors. These are denoted as $N_G^{++}(v)$, $N_G^{+-}(v)$, $N_G^{-+}(v)$ and $N_G^{--}(v)$, respectively. It is important to note that each type of 2-hop neighbor set may overlap with the others.

Lemma 3. *Suppose v is an arbitrary vertex in a signed graph G. For any balanced k-plex (P_L, P_R) with size at least $2k-1$ and $v \in P_L$, we have that $P_L \subseteq \{v\} \cup N_G(v) \cup N_G^{++}(v) \cup N_G^{--}(v)$ and $P_R \subseteq N_G(v) \cup N_G^{+-}(v) \cup N_G^{-+}(v)$.*

The pseudo-code of Bron-Kerbosch-based algorithm is presented in Algorithm 1, denoted MBPE-BK. The major framework is firstly sort vertices in degeneracy order $\eta = \{v_1, \cdots, v_n\}$ (Line 1), and then iteratively search for all the maximal v_i-lead balanced k-plex, i.e., those balanced k-plexes in which v_i precedes all other vertices according to η (Lines 2-8). The degeneracy order $\eta = \{v_1, \cdots, v_n\}$ is defined such that for each $1 \leq i \leq n$, v_i is the minimum-degree vertex in $G[\{v_i, \ldots, v_n\}]$ [26]. This ordering minimizes the maximum number of neighbors that precede v_i for each $v_i \in V$ in $G[\{v_i, \ldots, v_n\}]$, potentially reducing the size of initial candidate sets. It uses P_L and P_R to maintain the balanced k-plex, which are initialized with v_i and $\emptyset$. It also uses C_L and C_R to maintain the set of candidates that are used to grow the partial balanced k-plex. Since we require $\tau \geq k$, the target balanced k-plex must have size of at least $2k - 1$. Thus Lemma 3 can be employed to optimize the initialization of C_L and C_R (Lines 4-5). Note that here we only need to consider the vertices that succeed v_i in the ordering η. To avoid outputting duplicate results, X_L and X_R are used to record the vertices that have been processed (Lines 6-7). After initializing these six sets, it invokes the enumeration procedure Enum to find all maximal v_i-lead balanced k-plexes (Line 8).

Algorithm 1: MBPE-BK

Input: A signed graph $G = (V, E^+, E^-)$ and integers k, τ
Output: All maximal balanced k-plexes
1 Sort V by degeneracy order as $\{v_1, \cdots, v_n\}$;
2 **for each** v_i *in degeneracy order* **do**
3 $\quad P_L \leftarrow \{v_i\}$, $P_R \leftarrow \emptyset$;
4 $\quad C_L \leftarrow \{v_{i+1}, \cdots, v_n\} \cap (N_G(v_i) \cup N_G^{++}(v_i) \cup N_G^{--}(v_i))$;
5 $\quad C_R \leftarrow \{v_{i+1}, \cdots, v_n\} \cap (N_G(v_i) \cup N_G^{+-}(v_i) \cup N_G^{-+}(v_i))$;
6 $\quad X_L \leftarrow \{v_1, \cdots, v_{i-1}\} \cap (N_G(v_i) \cup N_G^{++}(v_i) \cup N_G^{--}(v_i))$;
7 $\quad X_R \leftarrow \{v_1, \cdots, v_{i-1}\} \cap (N_G(v_i) \cup N_G^{+-}(v_i) \cup N_G^{-+}(v_i))$;
8 $\quad$ Enum$(P_L, P_R, C_L, C_R, X_L, X_R)$;

Procedure Enum$(P_L, P_R, C_L, C_R, X_L, X_R)$
9 $C_L \leftarrow \{v \in C_L : (P_L \cup \{v\}, P_R)$ is a balanced k-plex$\}$;
10 $C_R \leftarrow \{v \in C_R : (P_L, P_R \cup \{v\})$ is a balanced k-plex$\}$;
11 $X_L \leftarrow \{v \in X_L : (P_L \cup \{v\}, P_R)$ is a balanced k-plex$\}$;
12 $X_R \leftarrow \{v \in X_R : (P_L, P_R \cup \{v\})$ is a balanced k-plex$\}$;
13 **if** $C_L \cup C_R = \emptyset$ **and** $X_L \cup X_R = \emptyset$ **then**
14 $\quad$ **if** $|P_L| \geq \tau$ **and** $|P_R| \geq \tau$ **then** emit (P_L, P_R);
15 **for each** $v \in C_L$ **do**
16 $\quad C_L \leftarrow C_L \setminus \{v\}$;
17 $\quad$ Enum$(P_R, P_L \cup \{v\}, C_R \setminus \{v\}, C_L, X_R \setminus \{v\}, X_L)$;
18 $\quad X_L \leftarrow X_L \cup \{v\}$;
19 **for each** $v \in C_R$ **do**
20 $\quad C_R \leftarrow C_R \setminus \{v\}$;
21 $\quad$ Enum$(P_R \cup \{v\}, P_L, C_R, C_L \setminus \{v\}, X_R, X_L \setminus \{v\})$;
22 $\quad X_R \leftarrow X_R \cup \{v\}$;

Procedure Enum performs the maximal balanced k-plex enumeration based on the given six sets. Firstly, C_L, C_R, X_L and X_R are refined by removing vertices that cannot form a larger balanced k-plex with the partial solution (P_L, P_R) (Lines 9-12). If these four sets are empty and (P_L, P_R) satisfies the size constraint, (P_L, P_R) is selected as the result (Line 14). Otherwise, it expands (P_L, P_R) by adding each vertex $v \in C_L$ to P_L and recursively invoke itself to further enlarge the balanced k-plex (Line 17). Note that in the new recursion, v's copies in C_R and X_R (if there are) should be removed. After v is processed, it is added to X_L (Line 18). Then, the algorithm continues to expand (P_L, P_R) by adding each vertex $v \in C_R$ to P_R in a similar way (Lines 19-22). Note that at Lines 17 and 21, we swap the roles of P_L and P_R, C_L and C_R, X_L and X_R, such that we are adding vertices to the two sides of the growing balanced k-plex in alternating order; this is to avoid generating too skewed intermediate results.

Theorem 2. *The time complexity of* MBPE-BK *(Algorithm 1) is $\mathcal{O}(n(\delta D)^2 3^{\delta D})$, where δ and D are the degeneracy and maximum degree of G, respectively.*

A. *Optimizations on* MBPE-BK

In this subsection, we present optimizations to improve the efficiency of MBPE-BK.

Vertex reduction. By utilizing the size constraint τ, we can remove unpromising vertices that are not contained in any maximal balanced k-plex. We have the following lemma.

Lemma 4. *Given a signed graph G, and integers k and τ, the vertex v in a maximal balanced k-plex must satisfy:*

- $d_G^+(v) \geq \tau - k$;
- $d_G^-(v) \geq \tau - k + 1$;
- $d_G(v) \geq 2\tau - k$.

The pseudo-code of vertex reduction is shown in Algorithm 2. Firstly, it identifies the unqualified vertex v (Line 1). Since v will be removed from the graph, it decreases the positive (resp. negative) degree and degree of its positive (resp. negative) neighbors by 1 (Lines 2-5). Lastly, the algorithm removes v and its associated edges from the graph (Line 6). The algorithm repeats the above steps until all the remaining vertices satisfy the degree constraints. The time complexity of VertexReduction is $\mathcal{O}(n + m)$.

Algorithm 2: VertexReduction

Input: A signed graph $G = (V, E^+, E^-)$ and integers k, τ
Output: A reduced signed graph
1 **while** $\exists v \in V$ *s.t.* $d_G^+(v) < \tau - k$ **or** $d_G^-(v) < \tau - k + 1$ **or** $d_G(v) < 2\tau - k$ **do**
2   **for each** $u \in N^+(v)$ **do**
3     $d_G^+(u) \leftarrow d_G^+(u) - 1$; $d_G(u) \leftarrow d_G(u) - 1$;
4   **for each** $u \in N^-(v)$ **do**
5     $d_G^-(u) \leftarrow d_G^-(u) - 1$; $d_G(u) \leftarrow d_G(u) - 1$;
6   Remove v from G;

Edge reduction. We can also prune unpromising edges based on the common neighbors between two vertices. For a positive edge (u, v) in G, we define its positive-positive support as the number of vertices that positively link to both u and v (i.e., $\Delta_G^{++}(u, v) = |\{w \mid (u, w) \in E^+ \wedge (v, w) \in E^+\}|$), and define its negative-negative support as the number of vertices that negatively link to both u and v (i.e., $\Delta_G^{--}(u, v) = |\{w \mid (u, w) \in E^- \wedge (v, w) \in E^-\}|$). For a negative edge (u, v), we define its positive-negative support as the number of vertices that positively link to u and negatively link to v (i.e., $\Delta_G^{+-}(u, v) = |\{w \mid (u, w) \in E^+ \wedge (v, w) \in E^-\}|$), and define its negative-positive support as the number of vertices that negatively link to u and positively link to v (i.e., $\Delta_G^{-+}(u, v) = |\{w \mid (u, w) \in E^- \wedge (v, w) \in E^+\}|$). We have the following lemma.

Lemma 5. *Given a signed graph G, and integers k and τ. Suppose (P_L, P_R) is a valid balanced k-plex, any positive edge (u, v) in $g(P_L, P_R)$ must satisfy:*

- $\Delta_G^{++}(u, v) \geq \tau - 2k$;
- $\Delta_G^{--}(u, v) \geq \tau - 2k + 2$;
- $\Delta_G^{++}(u, v) + \Delta_G^{--}(u, v) \geq 2\tau - 2k$.

Similarly, any negative edge (u, v) in $g(P_L, P_R)$ must satisfy:

- $\Delta_G^{+-}(u, v) \geq \tau - 2k + 1$;
- $\Delta_G^{-+}(u, v) \geq \tau - 2k + 1$;
- $\Delta_G^{+-}(u, v) + \Delta_G^{-+}(u, v) \geq 2\tau - 2k$.

The pseudo-code of edge reduction is shown in Algorithm 3. Firstly, it computes $\Delta_G^{++}(u, v)$ and $\Delta_G^{--}(u, v)$ for each positive edge, and $\Delta_G^{+-}(u, v)$ and $\Delta_G^{-+}(u, v)$ for each negative edge (Lines 1-2). Then the positive edge (u, v) will be removed if it violates Lemma 5 (Line 5). At the same time, the algorithm updates the common neighbor numbers for those affected edges (Lines 6-11). The negative edge (u, v) will also be removed by Lemma 5 and the common neighbor numbers for those affected edges are updated similarly (Lines 13-19). The algorithm terminates if no edge violates Lemma 5 (Line 3). The time complexity of EdgeReduction is $\mathcal{O}(m^{1.5})$.

Algorithm 3: EdgeReduction

Input: A signed graph $G = (V, E^+, E^-)$ and integers k, τ
Output: A reduced signed graph
1 **for each** $(u, v) \in E^+$ **do** Obtain $\Delta_G^{++}(u, v)$ and $\Delta_G^{--}(u, v)$;
2 **for each** $(u, v) \in E^-$ **do** Obtain $\Delta_G^{+-}(u, v)$ and $\Delta_G^{-+}(u, v)$;
3 **while** $\exists (u, v) \in E^+ \cup E^-$ *violates Lemma 5* **do**
4   **if** $(u, v) \in E^+$ **then**
5   Remove (u, v) from G;
6   **for each** w *s.t.* $(u, w) \in E^+$ **and** $(v, w) \in E^+$ **do**
7       $\Delta_G^{++}(u, w) \leftarrow \Delta_G^{++}(u, w) - 1$;
8       $\Delta_G^{++}(v, w) \leftarrow \Delta_G^{++}(v, w) - 1$;
9   **for each** w *s.t.* $(u, w) \in E^-$ **and** $(v, w) \in E^-$ **do**
10       $\Delta_G^{+-}(u, w) \leftarrow \Delta_G^{+-}(u, w) - 1$;
11       $\Delta_G^{+-}(v, w) \leftarrow \Delta_G^{+-}(v, w) - 1$;
12   **else**
13   Remove (u, v) from G;
14   **for each** w *s.t.* $(u, w) \in E^+$ **and** $(v, w) \in E^-$ **do**
15       $\Delta_G^{--}(u, w) \leftarrow \Delta_G^{--}(u, w) - 1$;
16       $\Delta_G^{-+}(v, w) \leftarrow \Delta_G^{-+}(v, w) - 1$;
17   **for each** w *s.t.* $(u, w) \in E^-$ **and** $(v, w) \in E^-$ **do**
18       $\Delta_G^{-+}(u, w) \leftarrow \Delta_G^{-+}(u, w) - 1$;
19       $\Delta_G^{--}(v, w) \leftarrow \Delta_G^{--}(v, w) - 1$;

In the implementation, we apply vertex and edge reductions on the original graph G at the beginning of MBPE-BK. Note that the vertex pruning and edge pruning may trigger each other. That is, the removal of unqualified vertices may trigger some edges to become invalid, and the removal of unqualified edges may trigger some vertices to become invalid. Thus, we repeatedly apply vertex reduction and edge reduction until no vertex and edge can be pruned. Additionally, during the enumeration process, we perform reductions on the subgraph that the current search instance is focusing on. Specifically, at the beginning of Enum, we apply vertex reduction on the subgraph $G[P_L \cup P_R \cup C_L \cup C_R]$. Note that we do not apply edge reduction in Enum due to its high computational cost.

Pivoting technique. The Bron-Kerbosch algorithm designed for maximal clique enumeration uses an effective technique called pivoting to reduce the number of branches in the search tree. The main idea of pivoting is that each maximal clique must include either the vertex u (i.e., the pivot) or a non-neighbor of u. Otherwise, a clique only contains neighbors of u and thus we can enlarge it by adding u to it, which makes it non-maximal. Motivated by this, we present a generalized version of pivoting that is applicable to the problem of maximal balanced k-plex enumeration. For clarity, we use P to denote (P_L, P_R) and C to denote (C_L, C_R). We define $u \in C$

if $u \in C_L \cup C_R$. For a vertex $u \in C$, $P \cup \{u\}$ is defined as $(P_L \cup \{u\}, P_R)$ if $u \in C_L$ and as $(P_L, P_R \cup \{u\})$ if $u \in C_R$.

Lemma 6. *Let $P = (P_L, P_R)$ be a partial balanced k-plex, and $C = (C_L, C_R)$ be the corresponding candidate set. Given a vertex $u \in C$, any expanded balanced k-plex must contain either u or the vertex v that $v \neq u$ and satisfy one of the following conditions:*

(1) *$v \in C$ and u, v are non-neighbors in $g(P \cup \{u, v\})$;*
(2) *$v \in C$ and u, v are neighbors in $g(P \cup \{u, v\})$ and u, v have common non-neighbors in $g(P \cup \{u, v\})$.*

Lemma 6 suggests that if a candidate vertex v is a neighbor of candidate vertex u and does not share any common non-neighbor with u in $g(P \cup \{u, v\})$, then we do not need to branch on v without affecting the correctness of the algorithm. Suppose u is the pivot from C_L, we have the following branching rule. At Line 15, Algorithm 1 only generates branches for $v \in C_L$ such that $v \notin N^+(u)$ or $P_L \setminus (N^+(u) \cap N^+(v)) \neq \emptyset$ or $P_R \setminus (N^-(u) \cap N^-(v)) \neq \emptyset$. At Line 19, Algorithm 1 only generates branches for $v \in C_R$ such that $v \notin N^-(u)$ or $P_L \setminus (N^+(u) \cap N^-(v)) \neq \emptyset$ or $P_R \setminus (N^-(u) \cap N^+(v)) \neq \emptyset$. The branching rule can be derived similarly when the pivot u is from C_R. To reduce the number of recursive calls as many as possible, we pick the vertex u which cuts the most branches from the search tree as the pivot.

B. The Drawback of MBPE-BK

Despite incorporating all the above optimizations, the performance of MBPE-BK remains unsatisfactory. The primary reason is that the candidate sets C_L and C_R often overlap significantly, which increases the computational burden. This overlap can further weaken the effectiveness of vertex reduction during the enumeration process. To address this issue, we propose a new enumeration framework that ensures C_L and C_R are disjoint and contain only the neighbors of the lead vertex v_i. As a result, the candidate set will be of size δ, where δ represents the graph's degeneracy, typically a small value in real-world graphs.

IV. THE ADVANCED ALGORITHM

In this section, we first propose a new search framework in Section IV-A. Then, we optimize the enumeration procedure in Section IV-B by proposing a minimum degree pivoting technique. We discuss the extension of our algorithms to the strict balanced k-plex enumeration problem in Section IV-C.

A. A New Enumeration Framework

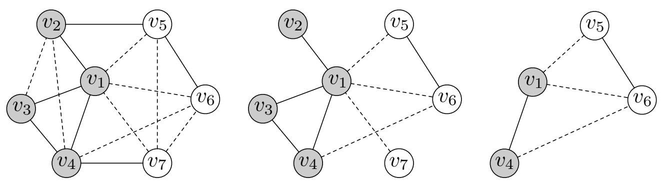
In MBPE-BK, the overlap between C_L and C_R arises from the fact that each vertex v_i in the balanced k-plex (P_L, P_R) is allowed to have at most k non-neighbors (including itself) in $g(P_L, P_R)$. In an ideal scenario, if we can determine in advance the set of v_i's non-neighbors in $g(P_L, P_R)$, then we only need to focus on its neighbors, i.e., C_L only contains v_i's positive neighbors, and C_R only contains v_i's negative neighbors. Thus C_L and C_R automatically become disjoint. To achieve this, we propose to exhaustively explore all possible initial balanced k-plex (P_L, P_R) such that v_i already has at most k non-neighbors in $g(P_L, P_R)$. For each such instance, we conduct a search to identify the maximal v_i-lead balanced k-plexes independently.

Algorithm 4: MBPE

Input: A signed graph $G = (V, E^+, E^-)$ and integers k, τ
Output: All maximal balanced k-plexes
1 Sort V by degeneracy order: $\eta = \{v_1, \cdots, v_n\}$;
2 **for each** v_i *in degeneracy order* **do**
3   **for each** $S \subseteq N_{\succ\eta}(v_i) \cup N^2_{\succ\eta}(v_i)$ *s.t.* $|S| \leq k-1$ **do**
4     $S_L \leftarrow \{u \in S \colon u \in N^-_{\succ_\eta}(v_i)\}$;
5     $S_R \leftarrow \{v \in S \colon v \in N^+_{\succ_\eta}(v_i)\}$;
6     $S_C \leftarrow \{v \in S \colon v \in N^2_{\succ_\eta}(v_i)\}$;
7   **for each** *partition* (T_L, T_R) *of* S_C **do**
8       $P_L \leftarrow \{v_i\} \cup S_L \cup T_L$;
9       $P_R \leftarrow S_R \cup T_R$;
10       $C_L \leftarrow N^+_{\succ_\eta}(v_i) \setminus S_R$;
11       $C_R \leftarrow N^-_{\succ_\eta}(v_i) \setminus S_L$;
12       $X_L \leftarrow N_{\prec_\eta}(v_i) \cup N^2_{\prec_\eta}(v_i) \cup (N^-_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)) \setminus (P_L \cup P_R)$;
13       $X_R \leftarrow N_{\prec_\eta}(v_i) \cup N^2_{\prec_\eta}(v_i) \cup (N^+_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)) \setminus (P_L \cup P_R)$;
14   Enum$(P_L, P_R, C_L, C_R, X_L, X_R)$;

The pseudo-code of our algorithm MBPE is shown in Algorithm 4. It first sorts the vertices in the degeneracy order $\eta = \{v_1, \cdots, v_n\}$ (Line 1). Then, from v_1 to v_n, the algorithm iteratively finds maximal v_i-lead balanced k-plexes (Lines 2-14). Let $N_{\prec_\eta}(v_i) = N(v_i) \cap \{v_1, \ldots, v_{i-1}\}$ and $N_{\succ_\eta}(v_i) = N(v_i) \cap \{v_{i+1}, \ldots, v_n\}$ be backward and forward neighbors of v_i, respectively; $N^2_{\prec_\eta}(v_i) = N^2(v_i) \cap \{v_1, \ldots, v_{i-1}\}$, $N^2_{\succ_\eta}(v_i) = N^2(v_i) \cap \{v_{i+1}, \ldots, v_n\}$ be the backward and forward 2-hop neighbors of v_i, respectively. For each vertex $v_i \in V$, it obtains the candidates that are at most 2-hop neighbors succeeding v_i. From $N_{\succ\eta}(v_i) \cup N^2_{\succ\eta}(v_i)$, it then picks a vertex subset S made up of v_i's potentially non-neighbors in $g(P_L, P_R)$ (Line 3). Next, it allocates these vertices to S_L, S_R, and S_C, respectively. S_L (resp. S_R) is the set of negative (resp. positive) neighbors of v_i, which will be put in the same (resp. opposite) side with v_i to become v_i's non-neighbors in $g(P_L, P_R)$, where (P_L, P_R) is the finally computed balanced k-plex (Lines 4-6). A further partition of S_C is done to ensure we exhaust all the possibilities (Line 7), since any vertex $u \in S_C$ will be v_i's non-neighbor in $g(P_L, P_R)$ no matter u is put in the same or opposite side with v_i. Then for each instance, we set up the initial balanced k-plex (P_L, P_R) (Lines 8-9). We also set up C_L and C_R to contain only the positive and negative neighbors of v_i by removing from C_L and C_R those vertices which are already in the initial balanced k-plex (Lines 10-11). The exclusive sets X_L, X_R include any other vertex that is possible to be added into the initial balanced k-plex (Lines 12-13). After initializing these six sets, it invokes procedure Enum to search for maximal v_i-lead balanced k-plexes under this instance (Line 14).

In MBPE, it is guaranteed that C_L and C_R are disjoint (i.e., C_L contains only v_i's positive neighbors and C_R contains only



(a) A signed graph G (b) Conflict edges removal (c) Vertex reduction

Fig. 4. Conflict edges removal and vertex reduction

v_i's negative neighbors). Thus, during the Enum procedure, we can reduce the subgraph $G[P_L \cup P_R \cup C_L \cup C_R]$ by removing *conflicting* edges that violate structural balance theory:

- negative edges between vertices of $P_L \cup C_L$,
- negative edges between vertices of $P_R \cup C_R$, and
- positive edges between a vertex of $P_L \cup C_L$ and a vertex of $P_R \cup C_R$.

This approach is justified as vertices from $P_L \cup C_L$ will belong to the same side of any derived balanced k-plex, and only positive edges among them will contribute to forming a balanced k-plex. Thus, removing negative edges between vertices of $P_L \cup C_L$ does not compromise the algorithm's correctness. Similarly, negative edges between vertices of $P_R \cup C_R$ can also be removed. Additionally, since any vertex $v \in P_L \cup C_L$ and any vertex $v' \in P_R \cup C_R$ will be on opposite sides of any derived balanced k-plex, only the negative edges between them will be counted for the form of balanced k-plex. Therefore, we can safely remove positive edges between vertices of $P_L \cup C_L$ and $P_R \cup C_R$.

Removing conflicting edges offers two significant advantages. Firstly, it leads to the sparsification of the subgraph $G[P_L \cup P_R \cup C_L \cup C_R]$. Secondly, it enhances the efficiency of the vertex reduction methods proposed in Section III-A, resulting in further reductions in the search space and computational costs. The following example illustrates this benefit.

Example 1. *Consider the signed graph G in Figure 4(a) with $k = 2$ and $\tau = 2$. Suppose v_1 precedes all other vertices in the ordering η. When processing v_1 with $S = \emptyset$, we have $P_L = \{v_1\}$, $P_R = \emptyset$, $C_L = \{v_2, v_3, v_4\}$, and $C_R = \{v_5, v_6, v_7\}$. Conflicting edges to be removed include negative edges (v_2, v_3), (v_2, v_4), (v_5, v_7), and (v_6, v_7), as well as positive edges (v_2, v_5) and (v_4, v_7). The subgraph after removing these conflicting edges is shown in Figure 4(b). We can then apply vertex reduction to further prune unpromising vertices, resulting in the final reduced subgraph shown in Figure 4(c). Without removing conflicting edges, the vertex reduction would not be able to identify unpromising vertices.*

Reduce the number of generated subset S. Our new algorithm, MBPE, ensures that candidate sets are disjoint by considering all possible subsets S of size at most $k-1$ from $N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$. In the worst case, the size of $N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$ can be as large as δD, leading to a substantial number of generated instances. To address this, we propose a partition-based vertex reduction technique aimed at decreasing the cardinality of $N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$. The main idea is to identify the minimum criteria that each vertex $u \in N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$ must meet to be part of a valid balanced k-plex with v_i. If a vertex u does not satisfy this criterion, it can be safely removed from $N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$, significantly reducing the number of subsets S.

Specifically, we partition $N_{\succ_\eta}(v_i)$ into $N^+_{\succ_\eta}(v_i) \cup N^-_{\succ_\eta}(v_i)$ and denote $L = \{v_i\} \cup N^+_{\succ_\eta}(v_i)$, $R = N^-_{\succ_\eta}(v_i)$, and $T = N^2_{\succ_\eta}(v_i)$. For each vertex $u \in L \cup R \cup T$, we define its L-part positive (or negative) neighbors as $N^+_L(u) = \{v \in L \mid (u,v) \in E^+\}$ (or $N^-_L(u) = \{v \in L \mid (u,v) \in E^-\}$). We also define its L-part positive (or negative) degree as $d^+_L(u) = |N^+_L(u)|$ (or $d^-_L(u) = |N^-_L(u)|$). Similarly, R-part positive and negative neighbors/degrees are defined by substituting L with R. In MBPE, for each vertex v_i, before generating all possible subsets S from $N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$, we can reduce the size of $N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$ using the following lemma.

Lemma 7. *Consider a signed graph $G = (V, E^+, E^-)$ and an arbitrary order of vertices $\eta = \{v_1, \cdots, v_n\}$. For a vertex $u \in N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$ to be part of a v_i-lead balanced k-plex, it must meet the following conditions:*

- *If $u \in N^+_{\succ_\eta}(v_i)$, it satisfies at least one of the following:*
  - $d^+_L(u) \geq \tau - k - |S| \wedge d^-_R(u) \geq \tau - k - |S| + 1$
  - $d^+_R(u) \geq \tau - k - |S| + 2 \wedge d^-_L(u) \geq \tau - k - |S| + 2$
- *If $u \in N^-_{\succ_\eta}(v_i)$, it satisfies at least one of the following:*
  - $d^+_L(u) \geq \tau - k - |S| + 1 \wedge d^-_R(u) \geq \tau - k - |S| + 3$
  - $d^+_R(u) \geq \tau - k - |S| \wedge d^-_L(u) \geq \tau - k - |S| + 1$
- *If $u \in N^2_{\succ_\eta}(v_i)$, it satisfies at least one of the following:*
  - $d^+_L(u) \geq \tau - k - |S| + 1 \wedge d^-_R(u) \geq \tau - k - |S| + 3$
  - $d^+_R(u) \geq \tau - k - |S| + 2 \wedge d^-_L(u) \geq \tau - k - |S| + 2$

The pseudo-code of PartitionVertexReduction is shown in Algorithm 5. Firstly, it computes the L-part and R-part positive/negative degrees for each vertex $u \in L \cup R \cup T$ (Lines 1-2). Then it identifies the vertex u that violates the degree constraints in Lemma 7. If u is from L, it decreases the L-part positive (resp. negative) degree by one for each of u's positive (resp. negative) neighbors in $L \cup R \cup T$ (Lines 4-6). If u is from R, it decreases the R-part positive (resp. negative) degree by one for each of u's positive (resp. negative) neighbors in $L \cup R \cup T$ (Lines 7-9). Then, it removes u from $L \cup R \cup T$ (Line 10). Note that the removal of the vertex from T will not cause the update of L-part and R-part degrees of other vertices. The algorithm terminates when no vertex violates Lemma 7.

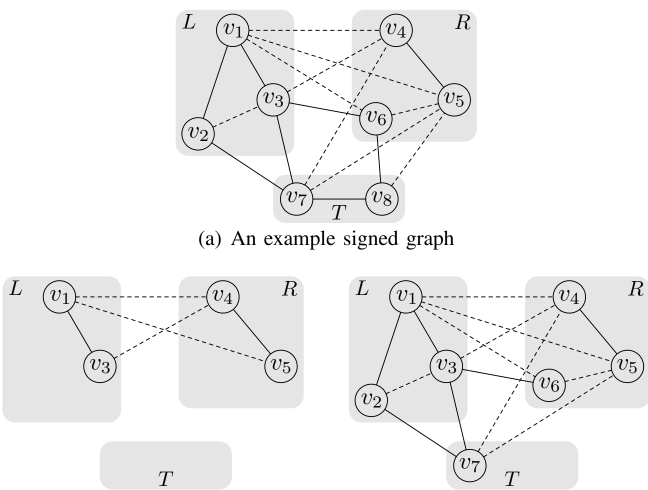
Example 2. *Consider the signed graph G in Figure 5(a) and let $k = 2, \tau = 2$. Suppose v_1 precedes all other vertices in the ordering η. We have $L = \{v_1, v_2, v_3\}$, $R = \{v_4, v_5, v_6\}$, and $T = \{v_7, v_8\}$. Firstly let us see how to prune unpromising vertices when $|S| = 0$. In this case, v_2 is unqualified since $d^-_R(v_2) = 0 < \tau - k - |S| + 1 = 1$ and $d^+_R(v_2) = 0 < \tau - k - |S| + 2 = 2$. Besides, v_6 is unqualified since $d^-_R(v_6) = 1 < \tau - k - |S| + 3 = 3$ and $d^+_R(v_6) + d^-_L(v_6) = 1 < 2\tau - k - |S| = 2$ and v_7 is unqualified since $d^-_R(v_7) = 2 < \tau - k - |S| + 3 = 3$ and $d^-_L(v_7) = 0 < \tau - k - |S| + 2 = 2$. v_8 is also unqualified due to the same reason as v_7. After removing*

Algorithm 5: PartitionVertexReduction

Input: A signed graph G, $L = \{v_i\} \cup N^+_{\succ_\eta}(v_i)$, $R = N^-_{\succ_\eta}(v_i)$, $T = N^2_{\succ_\eta}(v_i)$
Output: Reduced L, R, T

for each $u \in L \cup R \cup T$ **do**
  Compute $d^+_L(u)$, $d^-_L(u)$, $d^+_R(u)$, $d^-_R(u)$;
while $\exists u \in L \cup R \cup T$ *s.t.* $u \neq v_i$ *violates Lemma 7* **do**
  if $u \in L$ **then**
    for each $v \in N^+_{L\cup R\cup P}(u)$ **do** $d^+_L(v) \leftarrow d^+_L(v) - 1$;
    for each $v \in N^-_{L\cup R\cup P}(u)$ **do** $d^-_L(v) \leftarrow d^-_L(v) - 1$;
  else if $u \in R$ **then**
    for each $v \in N^+_{L\cup R\cup P}(u)$ **do** $d^+_R(v) \leftarrow d^+_R(v) - 1$;
    for each $v \in N^-_{L\cup R\cup P}(u)$ **do** $d^-_R(v) \leftarrow d^-_R(v) - 1$;
  Remove u from $L \cup R \cup T$;
return L, R, T;

all unpromising vertices, $N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$ is reduced as $\{v_3, v_4, v_5\}$, as shown in Figure 5(b). For the case of $|S| = 1$, only v_8 is unqualified since $d^-_R(v_8) = 1 < \tau - k - |S| + 3 = 2$ and $d^-_L(v_8) = 0 < \tau - k - |S| + 2 = 1$. In this case, $N_{\succ_\eta}(v_i) \cup N^2_{\succ_\eta}(v_i)$ is reduced as $\{v_2, v_3, v_4, v_5, v_6, v_7\}$, as shown in Figure 5(c).



(a) An example signed graph

(b) The case of $|S| = 0$

(c) The case of $|S| = 1$

Fig. 5. Partition vertex reduction

Theorem 3. *The time complexity of* MBPE *(Algorithm 4) is $\mathcal{O}(n2^{k-1}(\delta D)^{k+1}2^{\delta})$, where δ and D are the degeneracy and maximum degree of G, respectively.*

B. Minimum Degree Pivot

In our algorithm MBPE, Enum is the most critical component and has exponential time complexity. To improve performance, we propose a new enumeration procedure, Enum-MDP, equipped with a novel branching strategy. This approach greatly reduces the number of generated branches during the enumeration process, resulting in better worst-case time complexity compared to Enum. The main idea of this branching strategy is to select the vertex $u \in P_L \cup P_R$ with the minimum degree in $g(P_L \cup C_L, P_R \cup C_R)$. Then, based on u, we expand the partial balanced k-plex (P_L, P_R) by prioritizing the candidate vertices in $C_L \cup C_R$ that are u's non-neighbors in $g(P_L \cup C_L, P_R \cup C_R)$. When the number of u's non-neighbors in (P_L, P_R) reaches k, we no longer generate branches for u's remaining non-neighbors in $C_L \cup C_R$. In this way, the number of generated branches in the enumeration process can be reduced significantly. For simplicity, we will refer to $g(P_L \cup C_L, P_R \cup C_R)$ as g in the following sections.

Algorithm 6: Enum-MDP

Input: P_L, P_R, C_L, C_R, X_L, X_R
Output: Maximal balanced k-plexes

1 $C_L \leftarrow \{v \in C_L : (P_L \cup \{v\}, P_R)$ is a balanced k-plex$\}$;
2 $C_R \leftarrow \{v \in C_R : (P_L, P_R \cup \{v\})$ is a balanced k-plex$\}$;
3 $X_L \leftarrow \{v \in X_L : (P_L \cup \{v\}, P_R)$ is a balanced k-plex$\}$;
4 $X_R \leftarrow \{v \in X_R : (P_L, P_R \cup \{v\})$ is a balanced k-plex$\}$;
5 **if** $C_L \cup C_R = \emptyset$ **and** $X_L \cup X_R = \emptyset$ **then**
6   **if** $|P_L| \geq \tau$ **and** $|P_R| \geq \tau$ **then** emit (P_L, P_R);
7   **return**
  /* denote $g = g(P_L \cup C_L, P_R \cup C_R)$ */
8 $u \leftarrow$ the vertex of minimum degree in g;
9 **if** $d_g(u) \geq V(g) - k$ **then**
10   **if** *g is maximal and satisfies size constraint* **then**
11   emit $(P_L \cup C_L, P_R \cup C_R)$;
12 **else if** $u \in P_L \cup P_R$ **then**
13   Let $S = \{v_1, \cdots, v_{q_2}\}$ be the set of vertices in $(C_L \cup C_R) \setminus N_g(u)$;
14   $k' \leftarrow k - |(P_L \cup P_R) \setminus N_g(u)|$;
15   **for each** $i \in \{1, \cdots, k'\}$ **do**
16   **if** $v_i \in C_L$ **then**
17       $C_L \leftarrow C_L \setminus \{v_i\}$;
18   Enum-MDP$(P_L, P_R, C_L, C_R, X_L \cup \{v_i\}, X_R)$;
19       $P_L \leftarrow P_L \cup \{v_i\}$; $X_R \leftarrow X_R \setminus \{v_i\}$;
20   **else**
21       $C_R \leftarrow C_R \setminus \{v_i\}$;
22   Enum-MDP$(P_L, P_R, C_L, C_R, X_L, X_R \cup \{v_i\})$;
23       $P_R \leftarrow P_R \cup \{v_i\}$; $X_L \leftarrow X_L \setminus \{v_i\}$;
24   Enum-MDP$(P_L, P_R, C_L \setminus S, C_R \setminus S, X_L, X_R)$;
25 **else if** $u \in C_L$ **then**
26   Enum-MDP$(P_L, P_R, C_L \setminus \{u\}, C_R, X_L \cup \{u\}, X_R)$;
27   Enum-MDP$(P_L \cup \{u\}, P_R, C_L \setminus \{u\}, C_R, X_L, X_R \setminus \{u\})$;
28 **else**
29   Enum-MDP$(P_L, P_R, C_L, C_R \setminus \{u\}, X_L, X_R \cup \{u\})$;
30   Enum-MDP$(P_L, P_R \cup \{u\}, C_L, C_R \setminus \{u\}, X_L \setminus \{u\}, X_R)$;

The pseudo-code of Enum-MDP is shown in Algorithm 6. First, C_L, C_R, X_L and X_R are refined by removing vertices that cannot form a larger balanced k-plex with the current balanced k-plex (P_L, P_R) (Lines 1-4). If these four sets are empty and (P_L, P_R) satisfies the size constraint, it emits this maximal balanced k-plex (Lines 5-7). Next, it selects a vertex u with the minimum degree in g (Line 8) and check if g is maximal and satisfies the size constraint. If so, it emits $(P_L \cup C_L, P_R \cup C_R)$ and terminates the current branch (Lines 9-11). Otherwise, we know u has more than k non-neighbors in g and branches are generated as follows:

- If $u \in P_L \cup P_R$, let $q_1 = |(P_L \cup P_R) \setminus N_g(u)|$, representing the number of u's non-neighbors in $P_L \cup P_R$, and $q_2 = |(C_L \cup C_R) \setminus N_g(u)|$, representing the number of u's non-neighbors in $C_L \cup C_R$. Since u has more than k non-neighbors in g, we have $q_1 < k$ (otherwise, u cannot have more than k non-neighbors in g due to the refinement of

C_L and C_R at Lines 1-2). Let $k' = k - q_1$. At most k' vertices in $(C_L \cup C_R) \setminus N_g(u)$ can be included in the current balanced k-plex. If we use $\{v_1, \ldots, v_{q_2}\}$ to denote u's non-neighbors in the candidate sets $C_L \cup C_R$, $k' + 1$ branches will be generated (Lines 12-24):

1) The first branch moves v_1 from C_L (resp. C_R) to X_L (resp. X_R);
2) The second branch moves v_1 from C_L (resp. C_R) to P_L (resp. P_R) and v_2 from C_L (resp. C_R) to X_L (resp. X_R);
3) The i-th ($3 \leq i \leq k'$) branch moves $\{v_1, \cdots, v_{i-1}\}$ from C_L (resp. C_R) to P_L (resp. P_R) and v_i from C_L (resp. C_R) to X_L (resp. X_R);
4) The last branch moves $\{v_1, \cdots, v_{k'}\}$ from C_L (resp. C_R) to P_L (resp. P_R) and the rest of u's non-neighbors $\{v_{k'+1}, \cdots, v_{q_2}\}$ are removed from C_L (resp. C_R) to X_L (resp. X_R).

- If $u \notin P_L \cup P_R$, two branches are generated by moving u from C_L (resp. C_R) to X_L (resp. X_R) and by moving u from C_L (resp. C_R) to P_L (resp. P_R) for the case of $u \in C_L$ (resp. $u \in C_R$) (Lines 25-30). The second branch will fall into the first case.

The maximality check at Line 10 can be efficiently performed by verifying if there is any vertex in $X_L \cup X_R$ that can be added to g to form a larger balanced k-plex. This process can be efficient since if there is a vertex v in g that has already missed k vertices, we only need to consider v's neighbors in $X_L \cup X_R$. By substituting the Enum procedure with Enum-MDP in MBPE (Algorithm 4), we obtain a new algorithm, MBPE*. The time complexity of MBPE* is analyzed in the theorem below.

Theorem 4. *The time complexity of* MBPE* *is* $\mathcal{O}(n2^{k-1}(\delta D)^{k+1}\alpha_k^{\delta})$*, where δ and D are the degeneracy order the maximum degree of the graph respectively, α_k is strictly smaller than 2 and depends only on k, e.g., $\alpha_2 = 1.839$, $\alpha_3 = 1.928$, and $\alpha_4 = 1.966$.*

C. Extension to Strict Balanced k-plex Computation

Our proposed algorithm can also be easily adapted to enumerating the maximal strict balanced k-plex (Definition 3). Specifically, at Line 3 of Algorithm 4, we replace $S \subseteq N_{\succ\eta}(v_i) \cup N^2_{\succ\eta}(v_i)$ by $S \subseteq N^2_{\succ\eta}(v_i)$. This is because strict balanced k-plex requires each vertex is positively connected to other same-side vertices and negatively connected to other opposite-side vertices. Thus, S only contains v_i's 2-hop neighbors. We refer to the adapted algorithm for maximal strict balanced k-plex enumeration as MBPE*-Strict.

V. Experiments

In this section, we empirically evaluate the efficiency and effectiveness of our algorithms. We evaluate the following algorithms:

- MBPE-BK: The baseline algorithm proposed in Algorithm 1, equipped with all the optimizations proposed in Section III-A except the pivot technique.

TABLE I
Statistics of datasets

Dataset	$\lvert V\rvert$	$\lvert E\rvert$	$\frac{\lvert E^-\rvert}{\lvert E\rvert}$	D	δ	Category
Bitcoin	5,881	21,492	15%	795	21	Trade
AdjWordNet	16,259	76,845	32%	477	63	Language
WikiElection	7,118	100,693	22%	1065	53	Editing
Reddit	54,075	220,151	8%	4,164	76	Social
Referendum	10,884	251,406	5%	2,784	104	Political
Slashdot	82,140	500,481	24%	2,548	54	Social
Epinions	131,828	841,372	17%	3,558	121	Social
WikiConflict	116,836	2,027,871	62%	20,153	145	Editing
DBLP	2,387,365	11,915,023	72%	3,282	448	Cooperation
SN1	2,000,000	83,326,093	29%	1,301	489	Synthetic
SN2	3,000,000	124,990,528	31%	1,448	538	Synthetic

- MBPE-BK*: MBPE-BK algorithm equipped with all the optimizations proposed in Section III-A.
- MBPE: The algorithm proposed in Algorithm 4.
- MBPE*: Algorithm 4 equipped with the minimum degree pivot technique, i.e., by replacing the Enum procedure of MBPE with Enum-MDP.

Note that both MBPE and MBPE* apply the optimization techniques proposed in Section IV-A by default, i.e., subgraph reduction and partition based vertex reduction.

All the algorithms are implemented in C++, compiled with g++ 7.5.0 with the -O3 flag.[2] All the experiments are conducted on a machine with an Intel Core-i7 3.20GHz CPU and 64GB RAM running Ubuntu 18.04. The time cost is measured as the amount of wall-clock time elapsed during the program's execution. The time limit is set as 20 hours.

Datasets. We evaluate our algorithms on 9 publicly-available real-world signed graphs, whose main characteristics are summarized in Table I. Bitcoin[3] and Epinions[3] are who-trusts-whom graphs of users on Bitcoin OTC and Epinions, respectively. AdjWordNet[4] captures the synonym-and-antonym relation among adjectives in the English language. WikiElection[??] is the graph of users from the English Wikipedia that voted for and against each other during administration elections. Reddit[3] captures the sentiment among different subreddits on Reddit. Referendum [27] records Twitter data on the 2016 Italian Referendum, where interactions are negative for users with conflicting stances and positive otherwise. Slashdot[3] contains friend/foe links among Slashdot users. WikiConflict[??] represents positive and negative edit conflicts among the users of the English Wikipedia. DBLP is the signed network used in [28], which is downloaded from the *dblp database* [5] and processed in the same way as [28]. In addition, we also evaluate the algorithms on two synthetic datasets, SN1 and SN2, which are generated by the synthetic signed network generator SRN with default settings [29].

Exp-1: Efficiency when varying τ. In this experiment, we evaluate the efficiency of the algorithm by varying τ from 3 to 5 (with k fixed at 2). The running time for each algorithm is presented in Table II. The minimum running time for each

[2]Source code is available at https://github.com/kyaocs/MBPE
[3]https://snap.stanford.edu/
[4]https://wordnet.princeton.edu/
[5]https://dblp.uni-trier.de/

TABLE II
THE RUNNING TIME OF ENUMERATING MAXIMAL BALANCED k-PLEXES BY VARYING τ ($k = 2$)

Graphs	τ	Running time (sec)			
		MBPE-BK	MBPE-BK*	MBPE	MBPE*
Bitcoin	3	0.19	0.077	0.028	**0.021**
	4	0.06	0.015	0.013	**0.008**
	5	0.05	0.006	0.004	**0.002**
AdjWordNet	3	8930.63	15.15	8.39	**7.01**
	4	7923.66	9.57	4.26	**3.26**
	5	1929.86	4.69	2.30	**1.60**
WikiElection	3	15.38	14.31	1.21	**1.17**
	4	0.92	0.92	0.23	**0.23**
	5	0.87	0.87	0.16	**0.16**
Reddit	3	64.94	58.85	4.57	**4.45**
	4	7.77	7.67	0.69	**0.66**
	5	5.90	5.90	0.29	**0.29**
Referendum	3	1341.34	34.75	12.21	**10.36**
	4	114.98	6.91	2.74	**1.85**
	5	10.94	1.26	0.63	**0.39**
Slashdot	3	9.94	8.86	2.41	**2.33**
	4	0.68	0.67	0.64	**0.62**
	5	0.52	0.56	0.52	**0.50**
Epinions	3	594.79	483.13	42.82	**40.69**
	4	92.53	73.86	8.53	**7.32**
	5	27.83	19.60	2.69	**2.31**
WikiConflict	3	1143.19	1055.63	**141.43**	142.22
	4	4.42	4.43	3.10	**3.10**
	5	2.56	2.44	2.39	**2.26**
DBLP	3	17394.57	4620.07	381.52	**320.31**
	4	7305.79	2746.66	53.63	**46.87**
	5	46.21	46.50	32.01	**28.50**
SN1	3	-	19486.49	2249.52	**2136.59**
	4	2139.66	2136.74	428.77	**406.65**
	5	502.34	501.92	392.01	**388.82**
SN2	3	-	33747.44	3495.38	**3371.46**
	4	3130.09	3128.66	643.63	**635.24**
	5	735.60	734.53	585.92	**583.66**

instance is highlighted in bold. As τ increases, the running time decrease. This is because a larger size constraint τ enables the algorithm to prune more unpromising vertices early in the process. Consequently, the computation cost decreases. Notably, the pivot technique employed by MBPE-BK* significantly accelerates its performance, making it approximately two orders of magnitude faster than MBPE-BK. For instance, on AdjWordNet with $\tau = 4$, MBPE-BK* completes in 9.57 seconds, while MBPE-BK takes 7923.66 seconds. This highlights the effectiveness of the pivot technique. However, despite the gains made with the pivot technique, MBPE-BK* remains slower than MBPE in all cases, with a gap of approximately one order of magnitude. For instance, on Reddit with $\tau = 3$, MBPE takes 4.57 seconds, whereas MBPE-BK* takes 58.85 seconds. This discrepancy does demonstrate the effectiveness of the new search framework, thereby benefiting practical performance. Interestingly, the running time of MBPE and MBPE* are comparable, with MBPE* being slightly faster in most cases.

Exp-2: Efficiency when varying k. In this experiment, we evaluate the efficiency of the algorithm by varying k from 2 to 4 (setting $\tau = k + 2$). The results, shown in Table III, reveal that the running time of all algorithms increases with larger values of k. This is attributed to the fact that a larger k allows each member in a balanced k-plex to have more non-neighbors, significantly expanding the search space. In this experiment, the superiority of our new search framework becomes more evident, with MBPE outperforming MBPE-BK* significantly, sometimes achieving a speedup of up to three orders of magnitude. For example, on WikiElection, with $k = 3$, MBPE runs in 1.36 seconds, while MBPE-BK* takes 499.22 seconds. For $k = 4$, MBPE runs in 97.51 seconds, whereas MBPE-BK* fails to finish within 20 hours. Moreover, MBPE* consistently outperforms MBPE, especially when k becomes large. For instance, on Referendum with $k = 4$, MBPE* is around 2 times faster than MBPE, illustrating the improved efficiency from the minimum degree pivot techinque.

TABLE III
THE RUNNING TIME OF ENUMERATING MAXIMAL BALANCED k-PLEXES BY VARYING k ($\tau = k + 2$)

Graphs	k	Running time (sec)			
		MBPE-BK	MBPE-BK*	MBPE	MBPE*
Bitcoin	2	0.06	0.015	0.013	**0.008**
	3	0.35	0.079	0.097	**0.024**
	4	1.03	0.29	0.34	**0.081**
AdjWordNet	2	7923.66	9.57	4.26	**3.26**
	3	13492.20	564.26	50.34	**21.22**
	4	-	14356.45	920.71	**792.94**
WikiElection	2	0.92	0.92	0.23	**0.23**
	3	438.69	499.22	1.36	**1.31**
	4	-	-	97.51	**94.23**
Reddit	2	7.77	7.67	0.69	**0.66**
	3	5827.53	6667.67	84.66	**70.82**
	4	-	-	10403.30	**8759.93**
Referendum	2	114.98	6.91	2.74	**1.85**
	3	3818.18	670.82	122.70	**62.51**
	4	-	-	3213.36	**1676.66**
Slashdot	2	0.68	0.67	0.64	**0.62**
	3	1.75	1.45	0.68	**0.67**
	4	85.84	89.61	1.39	**1.34**
Epinions	2	92.53	73.86	8.53	**7.32**
	3	-	-	568.62	**465.23**
	4	-	-	53502.45	**46098.02**
WikiConflict	2	4.42	4.43	3.10	**3.10**
	3	2569.45	2613.50	6.50	**6.40**
	4	-	-	1413.70	**1389.39**
DBLP	2	7305.79	2746.66	53.63	**46.87**
	3	-	-	1614.75	**1487.92**
	4	-	-	37495.30	**29516.18**
SN1	2	2139.66	2136.74	428.77	**406.65**
	3	4921.64	4926.66	988.77	**936.65**
	4	-	-	18793.56	**18108.82**
SN2	2	3130.09	3128.66	643.63	**635.24**
	3	7470.37	7469.89	1560.08	**1548.87**
	4	-	-	27765.03	**25829.51**

Exp-3: Evaluation of each technique. In this experiment, we evaluate the effectiveness of each optimization technique by varying k from 2 to 4 (setting $\tau = k + 2$). We employ the algorithm MBPE* as our baseline and investigate three different variants of it. MBPE*-A does not apply vertex reduction and edge reduction on the original graph (proposed in Section III). MBPE*-B does not apply the subgraph reduction technique (proposed in Section IV). MBPE*-C does not apply the partition-based vertex reduction technique (proposed in Section IV). As shown in Table IV, MBPE* consistently outperforms all other algorithms. For example, on WikiConflict, when $k = 3$, MBPE* is three orders of magnitude faster than MBPE*-A (i.e., 6.40 seconds v.s. 6129.94 seconds). This demonstrates the effectiveness of our vertex and edge reduction techniques. On Reddit, when $k = 3$, MBPE* is around 6 times faster than MBPE*-B (i.e., 70.82 seconds v.s. 422.72 seconds), showcasing the effectiveness of our subgraph reduction technique. On WikiElection, when $k = 4$, MBPE* is around one order of magnitude faster than MBPE*-C (i.e., 94.23 seconds v.s. 839.88 seconds), validating the effectiveness of our partition based vertex reduction. This experiment demonstrates that each of our optimization techniques can contribute significantly to the algorithm efficiency.

TABLE IV
EVALUATION OF EACH TECHNIQUE BY VARYING k ($\tau = k + 2$)

Graphs	k	Running time (sec)			
		MBPE*-A	MBPE*-B	MBPE*-C	MBPE*
Bitcoin	2	0.27	0.009	0.009	**0.008**
	3	0.29	0.025	0.026	**0.024**
	4	0.36	0.090	0.081	**0.081**
AdjWordNet	2	62.82	3.38	3.80	**3.26**
	3	79.60	31.11	24.46	**21.22**
	4	865.18	1821.60	802.84	**792.94**
WikiElection	2	3.18	0.26	0.30	**0.23**
	3	8.89	4.97	3.92	**1.31**
	4	182.77	299.75	839.88	**94.23**
Reddit	2	199.03	1.26	0.83	**0.66**
	3	467.32	422.72	79.52	**70.82**
	4	-	-	36455.61	**8759.93**
Referendum	2	12.32	2.06	2.10	**1.85**
	3	94.94	77.46	66.61	**62.51**
	4	2477.84	2768.67	1825.48	**1676.66**
Slashdot	2	124.24	0.65	0.65	**0.62**
	3	235.22	0.71	0.70	**0.67**
	4	373.80	1.85	2.60	**1.34**
Epinions	2	2384.19	10.94	9.50	**7.32**
	3	4540.34	1775.48	531.59	**465.23**
	4	-	-	-	**46098.02**
WikiConflict	2	1498.75	3.14	3.24	**3.10**
	3	6129.94	12.72	19.10	**6.40**
	4	-	2383.00	11029.96	**1389.39**
DBLP	2	3394.64	53.14	159.25	**46.87**
	3	-	1957.65	9566.00	**1487.92**
	4	-	-	-	**29516.18**
SN1	2	36546.30	460.06	1513.33	**406.65**
	3	-	1112.40	8639.49	**936.65**
	4	-	-	-	**18108.82**
SN2	2	57143.52	696.56	2194.52	**635.24**
	3	-	1676.66	13239.49	**1548.87**
	4	-	-	-	**25829.51**

Exp-4: Evaluation of different models by varying k. In this experiment, we evaluate two proposed models: strict balanced k-plex (Definition 3) and balanced k-plex (Definition 4), by varying k from 2 to 4. The algorithm MBPE* aims to detect maximal balanced k-plexes, while MBPE*-Strict focuses on detecting maximal strict balanced k-plexes. As shown in Table V, MBPE* can detect more results than MBPE*-Strict in most cases, albeit with a slightly higher computational time. For instance, on Reddit, when $k = 3$ and $\tau = 5$, there are 3646 maximal balanced 3-plexes compared to 1361 maximal strict balanced 3-plexes, and MBPE* and MBPE*-Strict use 70.82 seconds and 43.66 seconds, respectively. This outcome aligns with theoretical expectations, as the balanced k-plex model generalizes the strict balanced k-plex model, allowing each vertex to hold a limited number of opposing opinions among its allies. Consequently, every strict balanced k-plex is also a balanced k-plex, but not vice versa.

TABLE V
EVALUATION OF DIFFERENT MODELS BY VARYING k ($\tau = k + 2$)

Graphs	k	MBPE*		MBPE*-Strict	
		# results	Time (sec)	# results	Time (sec)
Bitcoin	2	1026	0.008	1026	0.007
	3	7583	0.024	7583	0.023
	4	23739	0.081	23739	0.072
AdjWordNet	2	52374	3.26	52209	3.26
	3	3213645	21.22	3201555	20.98
	4	145899163	792.94	145455233	800.39
WikiElection	2	13	0.23	12	0.22
	3	7	1.31	5	1.00
	4	1	94.23	1	77.51
Reddit	2	752	0.66	520	0.64
	3	3646	70.82	1361	43.66
	4	11873	8759.93	2136	4897.61
Referendum	2	138924	1.85	138924	1.81
	3	3378920	62.51	3378920	62.11
	4	47040124	1676.66	47040124	1659.07
Slashdot	2	100	0.62	100	0.64
	3	40	0.67	40	0.60
	4	14	1.34	14	1.11
Epinions	2	319386	7.32	294265	7.26
	3	7961979	465.23	6702015	371.08
	4	146189731	46098.02	116817331	33502.45
WikiConflict	2	623	3.10	506	3.03
	3	762	6.40	528	5.93
	4	464	1389.39	291	1250.03
DBLP	2	612062	46.87	523948	43.54
	3	75018028	1487.92	63294958	1406.33
	4	820939485	29516.18	793049854	29176.66
SN1	2	102	406.65	97	423.59
	3	13	936.65	11	976.65
	4	2	18108.82	2	18108.82
SN2	2	120	635.24	114	650.01
	3	19	1548.87	16	1565.24
	4	1	25829.51	1	27462.79

Exp-5: Synonym and antonym group detection on AdjWordNet. In this experiment, we evaluate the effectiveness of the balanced k-plex by detecting synonym and antonym groups on AdjWordNet. This dataset is same as the one used for efficiency evaluation (see Table I), in which each vertex represents an adjective and the edge between a pair of synonyms (resp. antonyms) is positive (resp. negaitve). Using our algorithm MBPE*, we identify maximal balanced k-plexes with a size constraint $\tau = 6$ and $k = 3$. For comparison, we use MBCEnum [12] to detect maximal balanced cliques with the same constraint. Our results show $8,857$ maximal balanced k-plexes (sizes 12-64) and 183 maximal balanced cliques (sizes 12-60). Some synonym and antonym groups are uniquely detected by MBPE*, as k-plexes allow for some missing connections. Representative results are shown in Table VI. For instance, P_1 (shown in the second row) is a balanced 3-plex with $|P_L| = 7$ and $|P_R| = 7$. Compared with the maximal balanced clique, "up" and "pocket-size" (highlighted in gray) are newly appear since they do not connect to all other words. Similarly, P_2 (shown in the third row) is a balanced 3-plex with $|P_L| = 14$ and $|P_R| = 16$. Compared with the maximal balanced clique, "heavy" and "light" are

TABLE VI
CASE STUDY ON ADJWORDNET

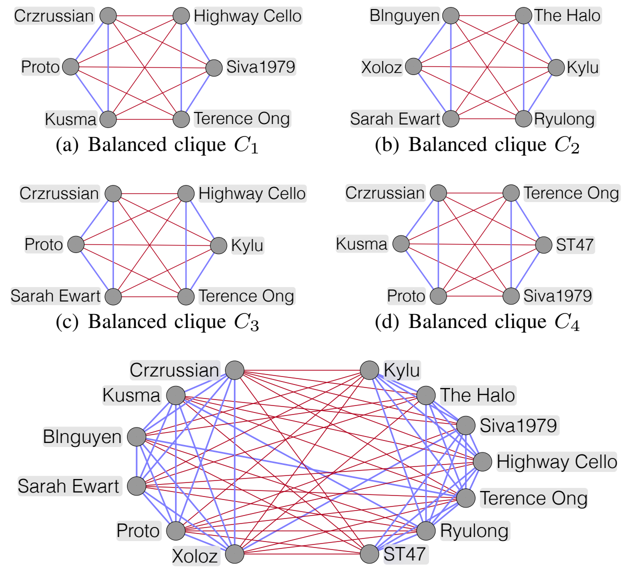
P_L	P_R
big, high, immodest, **up**, large, proud, tall	humble, modest, **pocket-size**, lowly, miserable, low, small
dirty, black, cloudy, ill-defined, dark, **heavy**, opaque, preserved, stale, unclean, unclear, unfair....	clean, blank, clear, fair, fresh, **light**, neat, unclouded, uncontaminated, uninfected, white....
...	...

newly appeared adjectives. From this experiment, we conclude that balanced k-plexes can identify more interesting synonym and antonym groups in AdjWordNet than balanced cliques.

Exp-6: Conflicts discovery on wiki-RfA. In this experiment, we evaluate the effectiveness of the balanced k-plex by detecting conflicts on wiki-RfA[3]. Wiki-RfA records the mutual voting results of Wikipedia editors running for administrator, which contains $11,381$ Wikipedia members (voters and votees) forming $189,004$ distinct voter/votee pairs. We build an undirected signed by regarding each user as a node and create a positive (or negative) edge between two nodes if there is an supporting (or opposing) vote between them. Our goal is to find polarized structures in wiki-RfA. First, we apply the MBCEnum algorithm [12], which identifies only maximal balanced cliques of size 6. Figures 6(a) to 6(d) display some of the detected maximal balanced cliques, with blue edges indicating positive connections and red edges indicating negative connections. Next, using our MBPE* algorithm with $k = 4$, we discover a balanced 4-plex of size 13 with $P_L = \{$"Crzrussian", "Kusma", "Blnguyen", $\cdots\}$ and $P_R = \{$"Kylu", "The Halo", "Siva1979", $\cdots\}$ (Figure 6(e)). This balanced 4-plex is a more significant polarized structure and includes all maximal balanced cliques detected by MBCEnum. Interestingly, we observe three positive edges connecting vertices from opposite sides, such as the positive edge between "Kusma" and "Terence Ong". This is allowed by our balanced k-plex model, and a limited number of such "conflicting edges" does not compromise the overall polarized structure. Although "Kusma" and "Terence Ong" support each other personally, they still align heavily with the polarized structure by supporting most of their allies and opposing most of their enemies). This highlights the advantages of our balanced k-plex model; by allowing slight deviations from structural balance theory, our model can uncover more intriguing patterns.

VI. RELATED WORKS

Signed graph was firstly studied by Harary et al. [17], where the notion of structural balanced is introduced. The problem of finding the largest (in terms of vertex number) vertex-induced subgraph that is structural balanced, known as the *maximum balanced subgraph problem*, is studied by Figueiredo et al. [30] and Ordozgoiti et al. [7]. The problem is NP-hard. A branch-and-cut exact algorithm, which only works for graphs with up-to a few thousand vertices, is proposed by Figueiredo et al. [30]. Heuristic algorithms without any guarantee on the solution optimality are investigated in [7],



(a) Balanced clique C_1 (b) Balanced clique C_2
(c) Balanced clique C_3 (d) Balanced clique C_4
(e) Balanced 4-plex

Fig. 6. Polarized structures detected in wiki-RfA

[30]. As the maximum balanced subgraph problem does not limit the non-neighbors and wrong-signed neighbors of each vertex in the identified subgraph, these techniques cannot be applied to our problem.

The notion of (structural) balanced clique is formulated by Chen et al. [12]. Recently, the detection of balanced cliques receives an increasing attention [12], [13], [15], [16]. However, balanced clique requires each pair of vertices are linked by an edge. Thus, their algorithms cannot be applied to our problem. Hao et al. [31] introduced the notion of *k-balanced trusted clique* for signed graphs, which is essentially the same problem as the traditional k-clique problem over unsigned graphs. Thus, the techniques proposed by Hao et al. [31] cannot be applied to solve our problem. The notion of (α, k)-clique is defined by Li et al. [28] for signed graphs, which is a clique such that each vertex has at most k negative neighbors and at least αk positive neighbors in the clique. As the structural balanced constraint is ignored, the techniques proposed by Li et al. [28] cannot be applied to our problem.

The problem of enumerating all maximal k-plexes in unsigned graph has received extensive attention [23], [25], [32]–[37]. Most of the maximal k-plex enumeration algorithms follow the framework of the Bron-Kerbosch algorithm, such as [25], [32], [35]. Aside from the Bron-Kerbosch variants, a polynomial delay algorithm is proposed by Berlowitz et al. [38]. Zhout et al. [36] proposed a novel branch scheme with a worst-case time complexity analysis. With their branch scheme, the running time of the algorithm is improved from $\mathcal{O}^*(2^n)$ to $\mathcal{O}^*(\gamma_k^n)$ in a graph with n vertices, where γ_k is related to k but strictly smaller than 2. Recently, Dai et al. [37] introduced a different branch strategy, which shares the same time complexity as the algorithm presented in [36]. Wand et al. [23] further improve the worst-case time complexity to $\mathcal{O}^*(\gamma_k^\delta)$, where δ is the degeneracy of the graph. However, these techniques cannot be directly applied to our problem due to the existence of edge signs and the enforcement of structural balanced constraint.

VII. Conclusion

In this paper, we formulated the balanced k-plex model in signed graphs and proved that the problem of enumerating all maximal balanced k-plexes is #P-hard. To solve this problem efficiently, we introduced the basic backtracking algorithm MBPE-BK and proposed several optimization techniques to enhance its practical performance. Then, we proposed a novel algorithm MBPE with better time complexity than MBPE-BK. We also proposed subgraph reduction and partition-based vertex reduction to improve its efficiency. Finally, we adopted the minimum-degree branching strategy to further improve the worst-case time complexity of MBPE. Experimental results on real-world signed graphs demonstrated the efficiency of our algorithms and effectiveness of our model.

References

[1] Jiliang Tang, Yi Chang, Charu Aggarwal, and Huan Liu. A survey of signed network mining in social media. *ACM Computing Surveys (CSUR)*, 49(3):1–37, 2016.

[2] David A. Easley and Jon M. Kleinberg. *Networks, Crowds, and Markets - Reasoning About a Highly Connected World*. Cambridge University Press, 2010.

[3] Jérôme Kunegis, Andreas Lommatzsch, and Christian Bauckhage. The slashdot zoo: mining a social network with negative edges. In *Proc. of WWW'09*, pages 741–750, 2009.

[4] Christos Giatsidis, Bogdan Cautis, Silviu Maniu, Dimitrios M Thilikos, and Michalis Vazirgiannis. Quantifying trust dynamics in signed graphs, the s-cores approach. In *Proc. of ICDM'14*, pages 668–676. SIAM, 2014.

[5] Le Ou-Yang, Dao-Qing Dai, and Xiao-Fei Zhang. Detecting protein complexes from signed protein-protein interaction networks. *IEEE/ACM transactions on computational biology and bioinformatics*, 12(6):1333–1344, 2015.

[6] Lingyang Chu, Zhefeng Wang, Jian Pei, Jiannan Wang, Zijin Zhao, and Enhong Chen. Finding gangs in war from signed networks. In *Proc. of SIGKDD'16*, pages 1505–1514, 2016.

[7] Bruno Ordozgoiti, Antonis Matakos, and Aristides Gionis. Finding large balanced subgraphs in signed networks. In *Proceedings of The Web Conference 2020*, pages 1378–1388, 2020.

[8] Jure Leskovec, Daniel Huttenlocher, and Jon Kleinberg. Predicting positive and negative links in online social networks. In *Proc. of WWW'10*, pages 641–650, 2010.

[9] Jihang Ye, Hong Cheng, Zhe Zhu, and Minghua Chen. Predicting positive and negative links in signed social networks by transfer learning. In *Proc. of WWW'13*, pages 1477–1488, 2013.

[10] Chien Chin Chen, Yu-Hao Wan, Meng-Chieh Chung, and Yu-Chun Sun. An effective recommendation method for cold start new users using trust and distrust networks. *Information Sciences*, 224:19–36, 2013.

[11] Jiliang Tang, Charu Aggarwal, and Huan Liu. Recommendations in signed social networks. In *Proceedings of the 25th International Conference on World Wide Web*, pages 31–40, 2016.

[12] Zi Chen, Long Yuan, Xuemin Lin, Lu Qin, and Jianye Yang. Efficient maximal balanced clique enumeration in signed networks. In *Proceedings of The Web Conference 2020*, pages 339–349, 2020.

[13] Zi Chen, Long Yuan, Xuemin Lin, Lu Qin, and Wenjie Zhang. Balanced clique computation in signed networks: Concepts and algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[14] Kai Yao, Lijun Chang, and Lu Qin. Computing maximum structural balanced cliques in signed graphs. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 1004–1016. IEEE, 2022.

[15] Renjie Sun, Yanping Wu, Xiaoyang Wang, Chen Chen, Wenjie Zhang, and Xuemin Lin. Clique identification in signed graphs: A balance theory based model. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[16] Kai Yao, Lijun Chang, and Lu Qin. Identifying large structural balanced cliques in signed graphs. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

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

[18] Francesco Bonchi, Edoardo Galimberti, Aristides Gionis, Bruno Ordozgoiti, and Giancarlo Ruffo. Discovering polarized communities in signed networks. In *Proc. of CIKM'19*, pages 961–970, 2019.

[19] Han Xiao, Bruno Ordozgoiti, and Aristides Gionis. Searching for polarization in signed graphs: a local spectral approach. In *Proceedings of The Web Conference 2020*, pages 362–372, 2020.

[20] Apichat Suratanee, Martin H Schaefer, Matthew J Betts, Zita Soons, Heiko Mannsperger, Nathalie Harder, Marcus Oswald, Markus Gipp, Ellen Ramminger, Guillermo Marcus, et al. Characterizing protein interactions employing a genome-wide sirna cellular phenotyping screen. *PLoS Comput Biol*, 10(9):e1003814, 2014.

[21] Kenneth E Read. Cultures of the central highlands, new guinea. *Southwestern Journal of Anthropology*, 10(1):1–43, 1954.

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

[23] Zhengren Wang, Yi Zhou, Mingyu Xiao, and Bakhadyr Khoussainov. Listing maximal k-plexes in large real-world graphs. In *Proceedings of the ACM Web Conference 2022*, pages 1517–1527, 2022.

[24] Stephen B Seidman and Brian L Foster. A graph-theoretic generalization of the clique concept. *Journal of Mathematical sociology*, 6(1):139–154, 1978.

[25] Alessio Conte, Tiziano De Matteis, Daniele De Sensi, Roberto Grossi, Andrea Marino, and Luca Versari. D2k: scalable community detection in massive networks via small-diameter k-plexes. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1272–1281, 2018.

[26] David W. Matula and Leland L. Beck. Smallest-last ordering and clustering and graph coloring algorithms. *J. ACM*, 30(3):417–427, 1983.

[27] Mirko Lai, Viviana Patti, Giancarlo Ruffo, and Paolo Rosso. Stance evolution and twitter interactions in an italian political debate. In *Natural Language Processing and Information Systems: 23rd International Conference on Applications of Natural Language to Information Systems, NLDB 2018, Paris, France, June 13-15, 2018, Proceedings 23*, pages 15–27. Springer, 2018.

[28] Rong-Hua Li, Qiangqiang Dai, Lu Qin, Guoren Wang, Xiaokui Xiao, Jeffrey Xu Yu, and Shaojie Qiao. Efficient signed clique search in signed networks. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pages 245–256. IEEE, 2018.

[29] Yansen Su, Bangju Wang, Fan Cheng, Lei Zhang, Xingyi Zhang, and Linqiang Pan. An algorithm based on positive and negative links for community detection in signed networks. *Scientific reports*, 7(1):10874, 2017.

[30] Rosa Maria Videira de Figueiredo and Yuri Frota. The maximum balanced subgraph of a signed graph: Applications and solution approaches. *Eur. J. Oper. Res.*, 236(2):473–487, 2014.

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

[32] Bin Wu and Xin Pei. A parallel algorithm for enumerating all the maximal k-plexes. In *Emerging Technologies in Knowledge Discovery and Data Mining: PAKDD 2007 International Workshops Nanjing, China, May 22-25, 2007 Revised Selected Papers 11*, pages 476–483. Springer, 2007.

[33] Sara Cohen, Benny Kimelfeld, and Yehoshua Sagiv. Generating all maximal induced subgraphs for hereditary and connected-hereditary graph properties. *Journal of Computer and System Sciences*, 74(7):1147–1159, 2008.

[34] Alessio Conte, Donatella Firmani, Caterina Mordente, Maurizio Patrignani, and Riccardo Torlone. Fast enumeration of large k-plexes. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 115–124, 2017.

[35] Zhuo Wang, Qun Chen, Boyi Hou, Bo Suo, Zhanhuai Li, Wei Pan, and Zachary G Ives. Parallelizing maximal clique and k-plex enumeration over graph data. *Journal of Parallel and Distributed Computing*, 106:79–91, 2017.

[36] Yi Zhou, Jingwei Xu, Zhenyu Guo, Mingyu Xiao, and Yan Jin. Enumerating maximal k-plexes with worst-case time guarantee. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2442–2449, 2020.

[37] Qiangqiang Dai, Rong-Hua Li, Hongchao Qin, Meihao Liao, and Guoren Wang. Scaling up maximal k-plex enumeration. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 345–354, 2022.

[38] Devora Berlowitz, Sara Cohen, and Benny Kimelfeld. Efficient enumeration of maximal k-plexes. In *Proceedings of the 2015 ACM SIGMOD International conference on management of data*, pages 431–444, 2015.

[39] Leslie G Valiant. The complexity of enumeration and reliability problems. *siam Journal on Computing*, 8(3):410–421, 1979.

[40] Vladimir Batagelj and Matjaz Zaversnik. An o (m) algorithm for cores decomposition of networks. *arXiv preprint cs/0310049*, 2003.

[41] Fedor V Fomin and Petteri Kaski. Exact exponential algorithms. *Communications of the ACM*, 56(3):80–88, 2013.