

# Clique Relaxations in Social Network Analysis: The Maximum $k$ -plex Problem

B. Balasundaram, S. Butenko, I. V. Hicks and S. Sachdeva

Department of Industrial and Systems Engineering,  
Texas A&M University,  
College Station, Texas 77843, USA.  
{baski,butenko,ivhicks,sandeeps}@tamu.edu

December 13, 2006

## Abstract

This paper introduces and studies the *maximum  $k$ -plex problem*, which arises in social network analysis, but can also be used in several other important application areas, including wireless networks, telecommunications, and graph-based data mining. We establish NP-completeness of the decision version of the problem on arbitrary graphs. An integer programming formulation is presented and basic polyhedral study of the problem is carried out. A branch-and-cut implementation is discussed and computational test results on the proposed benchmark instances and real-life *scale-free graphs* are also provided.

*keywords:* maximum  $k$ -plex problem; maximum clique problem; social network analysis; clique relaxations

## 1 Introduction

In the wake of information revolution, interest in studying the network structure of organizations, in particular criminal in nature, has increased manifold. Social network concepts, despite their versatility, have come to the forefront especially for these applications. A social network is usually represented by a graph, in which the set of vertices corresponds to the “actors” in a social network and the edges correspond to the “ties” between them [Scott, 2000]. Actors can be people, and examples of a tie between two actors include the acquaintance, friendship, or other type of association between them, such as visiting the same social event or place at the same time. Alternately, actors can be companies or other organizations, with ties representing various transactions between them. Thus, graphs can be used to conveniently model any such information and to make important deductions.

This paper introduces and studies the *maximum  $k$ -plex problem*, which arises in analysis of *cohesive subgroups* in social networks. *Social cohesion* is often used to explain and develop sociological theories. Members of a cohesive subgroup tend to share information, have homogeneity of thought, identity, beliefs, behavior, even food habits and illnesses [Wasserman and Faust, 1994]. Social cohesion is also believed to influence emergence of consensus among group members. Examples of cohesive subgroups include religious cults, terrorist cells, criminal gangs, military platoons, sports teams and conferences, work groups etc. Modeling a cohesive subgroup mathematically has

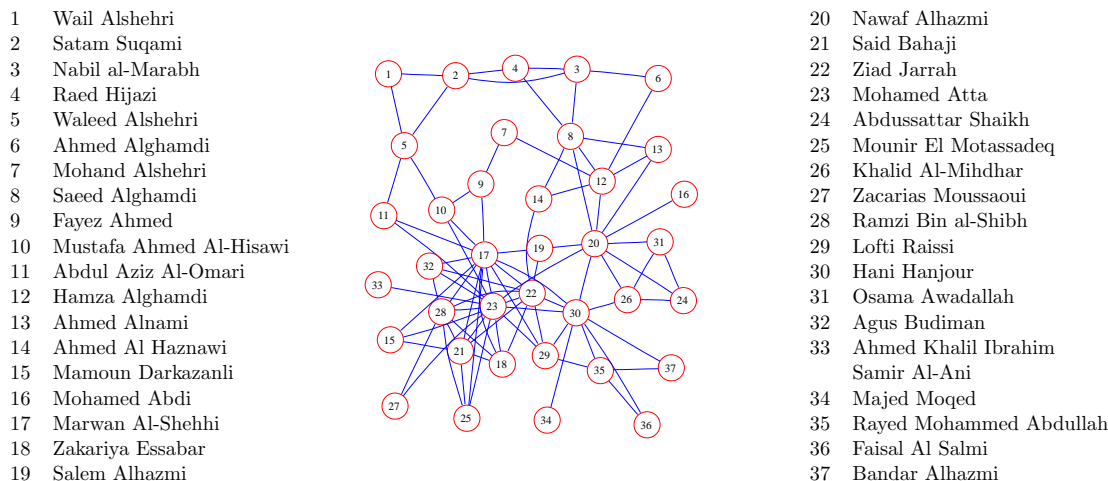
long been a subject of interest in social network analysis (SNA). One of the earliest graph models used for studying cohesive subgroups was the *clique* model [Luce and Perry, 1949]. A clique is a subgraph in which there is an edge between any two vertices. However, the clique approach has been criticized for its overly restrictive nature [Alba, 1973, Wasserman and Faust, 1994] and modeling disadvantages [Seidman and Foster, 1978, Freeman, 1992].

Alternative approaches were suggested that essentially relaxed the definition of cliques. Clique models idealize three important structural properties that are expected of a cohesive subgroup, namely, *familiarity* (each vertex has many neighbors and only a few strangers in the group), *reachability* (a low diameter, facilitating fast communication between the group members) and *robustness* (high connectivity, making it difficult to destroy the group by removing members). Different models relax different aspects of a cohesive subgroup. Luce [1950] introduced a distance based model called *k-clique*, and Alba [1973] introduced a diameter based model called *k-club*. These models were also studied along with a variant called *k-clan* by Mokken [1979]. All these models emphasize the need for high reachability inside a cohesive subgroup and have their own merits and demerits as models of cohesiveness. The focus of this paper is on a degree based model introduced by Seidman and Foster [1978] called *k-plex*. This model relaxes familiarity within a cohesive subgroup and implicitly provides reachability and robustness.

Some direct application areas of social networks include studying terrorist networks [Sageman, 2004, Berry et al., 2004], which is essentially a special application of criminal network analysis intended to study organized crimes such as terrorism, drug trafficking and money laundering [McAndrew, 1999, Davis, 1981, Sparrow, 1991]. Concepts of SNA provide suitable data mining tools for this purpose [Chen et al., 2004a]. Figure 1 shows an example of a terrorist network, which maps the links between terrorists involved in the tragic events of September 11, 2001. This graph was constructed in [Krebs, 2002] using the public data that were available before, but collected after the event. Even though the information mapped in this network is by no means complete, its analysis may still provide valuable insights into the structure of a terrorist organization. Traditionally, tools of SNA have been used with reasonable success in prosecuting crimes rather than in prevention [Krebs, 2002, Davis, 1981]. The concept of social cohesion, although quite successful in conventional social networks, is difficult to “implement” or model in a *covert* social network for various reasons. A major hurdle to overcome is the erroneous and incomplete nature of intelligence data [Sparrow, 1991, Davis, 1981] in addition to deciding who (or what) constitutes an “actor” and what constitutes a “tie” (weak or strong) in a covert social network. Clique relaxations alleviate the problem posed by erroneous and incomplete data to some extent by not being overly sensitive to missing edges, which is the most significant drawback of the clique model. Furthermore, the relaxations proposed in SNA are parameterized, in that they specialize to the classical clique model and allow one to define cohesion by the “clique-standard” and then systematically relax this notion.

Apart from the classical applications of SNA, the notion of cohesiveness and the use of clique relaxations are suitable in numerous other applications. These include clustering and mining biological networks, internet graphs, stock market graphs and wireless networks among others. For example, in internet web graphs, where cohesive subgroups correspond to collections of densely connected web sites [Terveen et al., 1999], topically related web sites are thus identified and organized to facilitate faster search and retrieval of information from the web. Graph-based data mining [Cook and Holder, 2000] has been used in studying structural properties of social networks [Mukherjee and Holder, 2004] and stock markets [Boginski et al., 2006], unraveling molecular structures to facilitate drug discovery and compound synthesis [Fischer and Meinl, 2004, Butenko and Wilhelm, 2006], and for identifying frequently occurring patterns in data sets (modeled as graphs) [Washio and Motoda, 2003, Boginski et al., 2004]. Clique and other low-diameter models have been popular in the area of wireless communication [Chen et al., 2004b, Krishna et al., 1997]. Clustering the

Figure 1: The network surrounding the tragic events of September 11, 2001.



*connectivity graph* of a wireless network introduces a hierarchy otherwise absent in these dynamic networks. Existence of a hierarchy facilitates routing of information through the network. Efficient resource management, routing and better throughput performance can be achieved through adaptive clustering of these mobile nodes [Stojmenovic, 2002]. A similar principle is also used in *organizational management*, where SNA is also used to study organizational structure to suggest better work practices and improve communication and work flow [Dekker, 2000].

Other interesting applications arise in systems biology, which studies organisms as biological systems. Technological advances such as microarrays, facilitate high-throughput experiments resulting in enormous amounts of experimental data, such as genomic and proteomic data. Thus, it is possible to capture a snapshot of the elements and their interactions at various levels – genes, transcripts, proteins, metabolites – that can be conveniently modeled as networks. Cliques have been used to cluster *gene co-expression networks* [Peng et al., 2004, Jiang et al., 2004], while cliques and high density subgraphs have been used to cluster *protein interaction networks* [Spirin and Mirny, 2003, Gagneur et al., 2004]. Graph-theoretic clique relaxations can provide interesting insights into these networks and allow to extract more information than what is revealed by cliques. The voluminous data generated by microarray experiments and other procedures are bound to contain significant percentage of errors. They could also be missing edges if interactions between elements are not yet known. Relaxing the restrictions imposed by clique models could reveal new protein interactions. In particular, structures where interactions of proteins occur through a central protein, which are likely to be found in similar biological processes, can be identified [Bader et al., 2004].

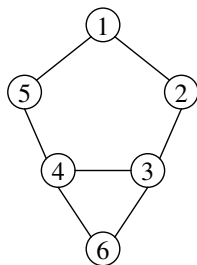
In spite of its potential applicability to a number of important practical situations, the optimization problems concerned with finding large  $k$ -plexes in a graph have not been studied from the mathematical programming perspective. Clique is undoubtedly one of the most important combinatorial objects, which plays a critical role in graph theory and mathematical programming in general, as well as in defining perfection where both areas elegantly come together. However, from the practical standpoint there is certainly a need to study clique relaxations in addition to investigating cliques. This paper introduces the maximum  $k$ -plex problem to the operations research community and analyzes its basic properties, including the computational complexity, mathematical programming formulations, and polyhedral study. Results of computational experience with a branch-and-cut algorithm for finding a maximum  $k$ -plex are also provided.

**Organization.** The remainder of this paper is organized as follows. Basic definitions, notations and background information of interest are presented in Section 2. Computational complexity analysis of the problem is carried out in Section 3. Section 4 introduces a binary integer programming formulation for the problem, presents some basic polyhedral properties of the problem, and develops valid inequalities. A branch-and-cut algorithm is presented in Section 5, including implementation details and computational test results. Furthermore, the proposed formulation is studied in the context of the maximum clique problem in Section 6. Finally, the paper is concluded with a summary and directions for future work in Section 7.

## 2 Definitions, Notations, and Background

Let  $G = (V, E)$  be a simple undirected graph representing a social network,  $d_G(u, v)$  denote the length of a shortest path between vertices  $u$  and  $v$  in  $G$ ,  $deg_G(v) = |\{u : (u, v) \in E\}|$  denote the degree of  $v$  in  $G$ , and  $diam(G) = \max_{u, v \in V} d_G(u, v)$  be the diameter of  $G$ . Denote by  $G[S] = (S, E \cap (S \times S))$ , the subgraph induced by  $S \subseteq V$ . A subset of vertices  $S \subseteq V$  is a  $k$ -clique if  $d_G(u, v) \leq k$  for all  $u, v \in S$ , and it is a  $k$ -club if  $diam(G[S]) \leq k$ . Note that a shortest path between two vertices in  $S$  may include vertices outside  $S$ . Hence, for a  $k$ -clique  $S$  there can exist two vertices  $u, v \in S$  such that  $d_G(u, v) \leq k$ , but  $d_{G[S]}(u, v) > k$  and thus  $u$  and  $v$  cannot exist in a  $k$ -club together. This is illustrated in Figure 2: The set  $\{2, 3, 4, 5, 6\}$  is a 2-clique but not a 2-club. In a social network, it may be unreasonable to expect a cohesive subgroup to require outside members, and the concept of  $k$ -club overcomes this weakness common to  $k$ -cliques by bounding the diameter of the induced subgraph. From the definitions and above example it follows that any  $k$ -club in  $G$  is also a  $k$ -clique, but the converse is not true. However,  $k$ -clubs have certain drawbacks that we illustrate using a simple example involving 2-clubs. It is possible that there exists one

Figure 2: 2-clique vs. 2-club



vertex in a 2-club that is adjacent to all other vertices, making it a 2-club, but these neighbors are poorly connected among themselves. This is demonstrated by *star graphs* which have diameter two as the central vertex is adjacent to all other vertices, but the neighbors of the central vertex have no edges between them. Although these models ensure reachability, they may lack cohesiveness in terms of degree and connectivity. In particular, removal of just one central vertex in a star graph completely disconnects the graph.

Degree based models of cohesion, which overcome the drawbacks inherent in the definitions of  $k$ -clique and  $k$ -club, were first introduced in [Seidman and Foster, 1978] and [Seidman, 1983]. Seidman [1983] introduced the concept of a  $k$ -core, which is a subgraph with minimum degree at least  $k$ . In other words,  $S \subseteq V$  is a  $k$ -core if  $|N(v) \cap S| \geq k \quad \forall v \in S$ , where  $N(v)$  denotes the set of neighbors of a vertex  $v \in V$  in  $G$ . However,  $k$ -cores were noted to only indicate dense regions of the graph and not necessarily identify a cohesive subgroup [Seidman, 1983, Wasserman and Faust,

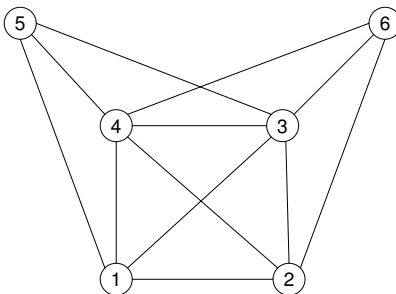
1994]. As suggested by Seidman (1983), this approach was only to produce global measures that captured the cohesive subgroups as well as regions surrounding them. We will now describe a simple greedy algorithm that finds the largest  $k$ -core in a graph in polynomial time. Pick a vertex  $v$  of minimum degree  $\delta(G)$ , if  $\delta(G) \geq k$  then we have a  $k$ -core. If  $\delta(G) < k$ , then that vertex cannot be in a  $k$ -core. Hence, delete the corresponding vertex,  $G \leftarrow G - v$  and continue recursively until the vertex set of  $G$  is a maximum  $k$ -core or the set is empty. Note that even though these structures are easy to find, they only point out dense regions of the graph where interesting subgroups may be found.

**Definition 1** A subset of vertices  $S$  is said to be a  $k$ -plex if the following condition holds:

$$\deg_{G[S]}(v) = |N(v) \cap S| \geq |S| - k \quad \forall v \in S.$$

That is, a subset of vertices  $S$  is said to be a  $k$ -plex if the degree of every vertex in the induced subgraph  $G[S]$  is at least  $|S| - k$ . A  $k$ -plex is said to be *maximal* if it is not strictly contained in any other  $k$ -plex. We propose calling the cardinality of the largest  $k$ -plex in the graph as *the  $k$ -plex number* and denote it by  $\omega_k(G)$ . The *maximum  $k$ -plex problem* is to find the largest  $k$ -plex of the given graph. Note that, as with the maximum  $k$ -clique and maximum  $k$ -club problems, this reduces to the *maximum clique problem* [Bomze et al., 1999] when  $k = 1$  and is a relaxation of the clique requirement for all other  $k > 1$ , allowing for at most  $k - 1$  non-neighbors inside the set. Figure 3 illustrates this concept: The set  $\{1, 2, 3, 4\}$  is a 1-plex (clique), sets  $\{1, 2, 3, 4, 5\}$  and  $\{1, 2, 3, 4, 6\}$  are 2-plexes (maximal and maximum) and the entire graph is a 3-plex.

Figure 3: Illustration of  $k$ -plexes for  $k = 1, 2, 3$



Apart from the basic definition that a graph  $G = (V, E)$  is a  $k$ -plex if  $\delta(G) \geq |V| - k$ , an alternate characterization of  $k$ -plexes has been established in the following theorem by Seidman and Foster [1978]. Let  $N[v]$  denote the closed neighborhood of a vertex  $v$ , that is  $N[v] = \{v\} \cup N(v)$ .

**Theorem 1 (Seidman and Foster, 1978)**  $G$  is a  $k$ -plex if and only if for any  $k$ -element subset of vertices  $\{v_1, \dots, v_k\} \subseteq V$ ,  $V = \bigcup_{i=1}^k N[v_i]$ .

In other words, the above theorem states that if  $G$  is a  $k$ -plex then any  $k$  vertices form a dominating set in the graph (a subset  $S$  of vertices of the graph  $G = (V, E)$  is called a *dominating set* if  $\cup_{i \in S} N[i] = V$ ). For instance, in the graph shown in Figure 3,  $\{1, 5\}$  is not a dominating set, but with any other vertex, say 6, the resulting set  $\{1, 5, 6\}$  is a dominating set, and so is any other triplet of vertices. Seidman and Foster [1978] also establish some of the basic graph theoretic properties of a  $k$ -plex that are stated here. Note that the vertex connectivity  $\kappa(G)$  is defined as the minimum number of vertices whose removal results in a disconnected or trivial graph [Harary, 1988].

**Theorem 2 (Seidman and Foster, 1978)** *Let graph  $G$  be a  $k$ -plex on  $n$  vertices. Then,*

1. *Any vertex-induced subgraph of  $G$  is a  $k$ -plex.*
2. *If  $k < \frac{(n+2)}{2}$ , then  $\text{diam}(G) \leq 2$ .*
3.  *$\kappa(G) \geq n - 2k + 2$ .*

By definition, members of a  $k$ -plex  $S$  can have at most  $k - 1$  non-neighbors inside  $S$ . Thus,  $k$ -plexes with low  $k$  values ( $k = 2, 3$ ) provide good relaxations of a clique that closely resemble the cohesive subgroups that can be found in real-life social networks. In addition, the above results indicate that a  $k$ -plex, besides being a natural generalization of a clique, also retains the properties of a clique such as low diameter and high connectivity for low values of  $k$ . The  $k$ -plex model overcomes the disadvantages of  $k$ -cliques and  $k$ -clubs by directly limiting the number of *non-neighbors* inside the cohesive subgroup. This structure imposes a degree bound that varies with the size of the group and hence ensures a cohesive subgroup even as the size of the group varies. Implicitly, it also achieves reachability and robustness. By allowing some “strangers” in a social group,  $k$ -plex provides a more realistic alternative to model cohesive subgroups in a social network.

The maximum clique problem is closely related to another well known graph problem, which is the *maximum independent set problem*. An independent set (or stable set) is a subset of pairwise nonadjacent vertices. In other words, the subgraph induced by an independent set is edgeless. A subset of vertices forms a clique in  $G = (V, E)$  if and only if it forms an independent set in the complement graph  $\bar{G} = (V, \bar{E})$ . Naturally, the  $k$ -plex can also be closely related to a similar problem on the complement graphs which we formalize as follows.

**Definition 2** *We call a subset of vertices  $S$  of a graph  $G = (V, E)$  a co- $k$ -plex if,*

$$\text{deg}_{G[S]}(v) = |N(v) \cap S| \leq k - 1 \quad \forall v \in S.$$

In other words, the induced subgraph  $G[S]$  has a maximum degree of  $k - 1$  or less. It should be noted that  $S$  is a co- $k$ -plex in  $G$  if and only if  $S$  is a  $k$ -plex in the complement graph  $\bar{G}$ . In particular, 1-plex is a clique and a co-1-plex is an independent set. Thus,  $k$ -plexes and co- $k$ -plexes provide a systematic way to generalize two important graph models.

The need for clique relaxations has been primarily dealt with in literature using various edge density based models such as *densest (heaviest)  $k$ -subgraph problem*, *maximum edge subgraph problem* [Corneil and Perl, 1984, S.S. Ravi, 1994] and the *maximum quasi-clique problem* [Abello et al., 1999, 2002] to cite a few examples. The motivation is clearly to relax the property that a clique has all possible edges. However, the density-based relaxations can result in subgraphs with high diameter, low connectivity and could contain vertices of low degree and even independent vertices. The structural guarantees are limited unless the required density is very high. The  $k$ -plex model is advantageous primarily because the relaxation is “controlled” and systematic resulting in good structural guarantees. It should be noted that our criticism of some of the existing models is not meant to undermine their usefulness. In fact, all these approaches have been applied successfully on real-life data. For instance, density based models are especially useful while relaxing independent sets by allowing more than zero edge density. More recently, Sherali and Smith [2006] proposed a related model for relaxing independent sets, in which an upper bound is placed on the number of edges in the induced subgraph. A generalized vertex packing GVP- $k$  is a subset of vertices  $I$  such that there are at most  $k$  edges in the induced subgraph  $G[I]$ . When  $k = 0$ , it represents an independent set and is a relaxation for  $k \geq 1$ . This approach, used to model problems in air-traffic control and national airspace planning, is studied using polyhedral methods in [Sherali and Smith,

2006]. The model is close to the co- $k$ -plex model for relaxing independent sets. Note that every GVP- $k$  is a co- $(k+1)$ -plex, but not vice versa.

Similarly, in applications where reachability is the only consideration  $k$ -cliques and  $k$ -clubs models are more appropriate. Furthermore, whenever relatively larger cohesive subgroups are required, with only reachability as the basic requirement, the following relationship should be taken into account. Let  $\omega(G)$ ,  $\omega_k(G)$ ,  $\bar{\omega}_k(G)$  and  $\tilde{\omega}_k(G)$  denote the size of a largest clique,  $k$ -plex,  $k$ -club and  $k$ -clique in the graph  $G$ , respectively. Then we have,  $\omega(G) \leq \omega_k(G) \leq \bar{\omega}_k(G) \leq \tilde{\omega}_k(G)$  whenever  $\omega_k(G) > 2k - 2$ . This emphasizes the fact that whenever sound structural guarantees, as provided by the clique definition are necessary,  $k$ -plex model should be considered as a meaningful alternative. Recently, Chesler and Langston (2006), while studying the clustering problem on gene co-expression networks, point out some of the issues that were raised before regarding the sensitivity of cliques to missing edges. A model robust enough to deal with this situation is necessary and  $k$ -plex is ideal for such purposes. In particular, the authors propose the use of *paracliques* that are close to the 2-plex model. The basic idea is to start with some maximum clique  $C$  and add new vertices to the set if they have at least  $g$  neighbors in the original  $C$  [Chesler and Langston, 2006]. They report successful results with  $g = |C| - 1$  where  $g$  is called the *glom factor* [Chesler and Langston, 2006]. Although the final result need not be a 2-plex, the notion of allowing one non-neighbor is clearly in the same spirit as a 2-plex approach. In the remainder of this paper, we will present our contributions to understanding the maximum  $k$ -plex problem.

### 3 Computational Complexity

This section presents computational complexity results for the problem of interest. The maximum  $k$ -plex problem is trivially NP-hard for arbitrary  $k$  as it includes the maximum clique problem as a special case. However, maximum  $k$ -plex problem for fixed  $k$  is of more interest as we need to solve the problem for  $k = 1, 2, 3$  and so on. The decision version of the maximum  $k$ -plex problem can be stated as follows:

**Definition 3** *k*-PLEX: *Given a simple undirected graph  $G = (V, E)$  and fixed positive integers  $c$ ,  $k$ , does there exist a  $k$ -plex of size  $c$  in  $G$ ?*

**Theorem 3** *k*-PLEX is NP-complete for any fixed positive integer  $k$ .

PROOF. We prove this by reducing CLIQUE [Garey and Johnson, 1979], a well-known NP-complete problem, to  $k$ -PLEX. Given an instance  $\langle G = (V, E), c \rangle$  of CLIQUE, we construct an instance  $\langle G' = (V', E'), c' \rangle$  in polynomial time such that  $G$  has a clique of size  $c$  if and only if  $G'$  has a  $k$ -plex of size  $c'$ . To construct  $G'$ , we expand  $G$  by adding  $k-1$  copies of the complete graph of order  $n = |V|$ . Denote the vertex set of the  $r^{\text{th}}$  such copy by  $V_r$ ,  $r = 1, \dots, k-1$ , where  $V_r = \{1_r, \dots, n_r\}$ , and let  $R = \bigcup_{r=1}^{k-1} V_r$ . Put  $V' = V \cup R$  and  $E' = E \cup \hat{E} \cup \tilde{E}$ , where

$$\hat{E} = \{(i, j_r) : i \in V, j_r \in V_r, i \neq j, r = 1, \dots, k-1\}$$

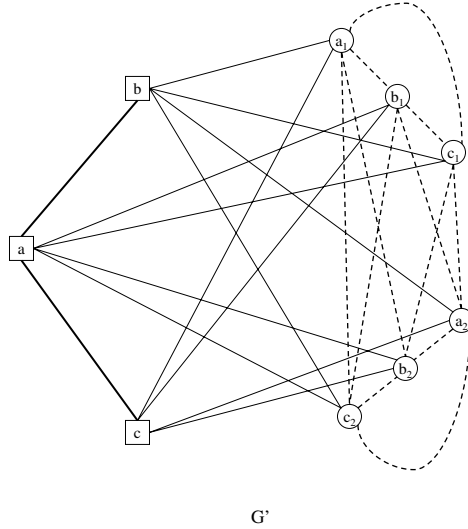
and

$$\tilde{E} = \{(i_p, j_r) : i_p \in V_p, j_r \in V_r, i \neq j, p, r = 1, \dots, k-1\}.$$

The set  $\hat{E}$  represents the edges between  $V$  and  $R$ , where every vertex  $u \in V$  is connected to every vertex in every complete graph except its copies, *i.e.*,  $u$  is adjacent to every vertex in  $R \setminus \{u_1, \dots, u_{k-1}\}$ . The set  $\tilde{E}$  includes the cross edges between distinct  $V_p$  and  $V_r$ , as well as all possible

edges between vertices in  $V_p$ ,  $p = 1, \dots, k-1$ . In other words, every vertex  $u_p \in V_p$ ,  $p = 1, \dots, k-1$  is adjacent to all the vertices in  $V_r \setminus \{u_r\}$ ,  $r = 1, \dots, k-1$ . Putting  $c' = c + (k-1)n$  completes the reduction. Note that the instance  $\langle G' = (V', E'), c' \rangle$  can be constructed in polynomial time. Figure 4 illustrates this transformation when  $G$  is a path on three vertices  $a, b$  and  $c$ .

Figure 4: Illustration of the 3-PLEX instance  $G'$ . Original graph  $G$  in box-vertices and bold solid edges.  $\hat{E}$  is denoted by regular solid edges and  $\tilde{E}$  is denoted by dashed edges.



We now show that if there exists a clique of size  $c$  in  $G$  then  $G'$  has a  $k$ -plex of size  $c'$ . Let  $C \subseteq V$  induce a clique of size  $c = |C|$  in  $G$ . We claim that the set  $S = C \cup R$ , where  $|S| = c + n(k-1) = c'$ , is a  $k$ -plex. For any  $u \in C$ , there exist  $c-1$  neighbors inside  $C$ , and  $(n-1)(k-1)$  neighbors in  $R$ . Thus, for  $u \in C$ ,  $\deg_{G[S]}(u) = c-1 + (n-1)(k-1) = c' - k$ . For any  $v_r \in R$ , there exist  $(n-1)(k-1)$  neighbors in  $R$  and  $c$  neighbors in  $C$  if  $v \notin C$ , and  $c-1$  neighbors in  $C$  if  $v \in C$ . Again, for  $v_r \in R$ ,  $\deg_{G[S]}(v_r) \geq c-1 + (n-1)(k-1) = c' - k$ . Hence,  $S$  induces a  $k$ -plex of size  $c'$ .

We now establish the other direction stating that if there exists a  $k$ -plex of size  $c'$  in  $G'$  then  $G$  has a clique of size  $c$ . Let  $S$  be a  $k$ -plex of size  $c' = c + n(k-1)$ . Let  $P = R \setminus S$  denote the set of vertices from  $R$  not included in the  $k$ -plex and let  $|P| = p$ . Then, the  $c'$  vertices in  $S$  consist of  $n(k-1) - p$  vertices in  $S \cap R$  and  $c + p$  vertices in  $S \cap V$ . Without loss of generality, suppose that  $S \cap V = \{1, \dots, c+p\}$  and further assume that for each  $i \in S \cap V$  there exist  $q_i$  copies of  $i$  in  $P$  that are left out of the  $k$ -plex. Since every  $i \in S \cap V$  has  $p - q_i$  neighbors in  $P$ , we know that

$$|N(i) \cap (S \cap R)| = (n-1)(k-1) - (p - q_i).$$

Since  $S$  is a  $k$ -plex,  $\forall i \in S \cap V$  :

$$\begin{aligned} \deg_{G[S]}(i) &= |N(i) \cap (S \cap R)| + |N(i) \cap (S \cap V)| \geq c + n(k-1) - k, \\ \Rightarrow |N(i) \cap (S \cap V)| &\geq c + p - 1 - q_i. \end{aligned} \quad (1)$$

Recall that each  $q_i$  is a non-negative integer counting copies of vertex  $i \in S \cap V$  in  $P$  and note that  $P$  can contain vertices that are not copies of any vertex in  $S \cap V$ . Thus, we have  $\sum_{i=1}^{c+p} q_i \leq p$ . Hence, there can exist at most  $p$  terms,  $q_i$ , in that sum that are strictly greater than 0, meaning that there exist at least  $c$  terms in that equation, that are equal to 0. Without loss of generality,

suppose that  $q_i = 0$ ,  $i \in \{1, \dots, c\}$ . Now, let  $C = \{1, \dots, c\}$ . We already know from (1) that for all  $i \in C \subseteq S \cap V = \{1, \dots, c+p\}$ :

$$|N(i) \cap (S \cap V)| \geq c + p - 1 - q_i = c + p - 1.$$

But  $|S \cap V| = c + p$ , so for all  $i \in C$ ,

$$|N(i) \cap (S \cap V)| = c + p - 1.$$

Thus, every vertex in  $C \subseteq S \cap V$  is adjacent to every vertex in  $S \cap V$ . Hence, every vertex in  $C$  is adjacent to every other vertex in  $C$ . Therefore  $C$  induces a clique of size  $c$  in  $G$ .  $\square$

This complexity result demonstrates that the maximum  $k$ -plex problem is hard not only because it is a generalization of the maximum clique problem, but it is a hard problem in its own respect, as Theorem 3 states that the decision version of the problem is NP-complete for every fixed  $k$ .

## 4 Mathematical Programming Approaches

This section presents an integer programming formulation of the maximum  $k$ -plex problem followed by a polyhedral study of the problem. Valid inequalities for the polytope under study are also presented.

### 4.1 Integer Programming Formulation

Given a graph  $G = (V, E)$  with  $|V| = n$ , recall that  $N[i]$  is the closed neighborhood of a vertex  $i$  and  $\omega_k(G)$  is the  $k$ -plex number of  $G$ . Let  $\bar{d}_i = |V \setminus N[i]|$  denote the degree of vertex  $i$  in the complement graph  $\bar{G} = (V, \bar{E})$ . Further assume that  $k > 1$  since  $k = 1$  yields the well known maximum clique problem. The following 0-1 program finds the largest  $k$ -plex in  $G$ .

$$\omega_k(G) = \max \sum_{i \in V} x_i \tag{2}$$

subject to:

$$\sum_{j \in V \setminus N[i]} x_j \leq (k-1)x_i + \bar{d}_i(1-x_i) \quad \forall i \in V, \tag{3}$$

$$x_i \in \{0, 1\} \quad \forall i \in V. \tag{4}$$

In this formulation,  $x_i = 1$  if and only if  $i \in V$  is in the  $k$ -plex and  $x_i = 0$  otherwise. Constraint (3) ensures that if a vertex  $i$  is in the  $k$ -plex then it has at most  $k-1$  non-neighbors inside the  $k$ -plex. The constraint is made redundant for vertices not in the  $k$ -plex.

### 4.2 Polyhedral Study

The  $k$ -plex polytope  $P_k(G)$  is the convex hull of feasible binary vectors of the aforementioned formulation (2) - (4). The following theorem establishes the trivial facets of the  $k$ -plex polytope.

**Theorem 4** Let  $P_k(G)$  denote the  $k$ -plex polytope of a given graph  $G = (V, E)$ , where  $k > 1$ . Then,

1.  $\dim(P_k(G)) = n$ .
2.  $x_i \geq 0$  induces a facet of  $P_k(G)$  for every  $i \in V$ .
3.  $x_i \leq 1$  induces a facet of  $P_k(G)$  for every  $i \in V$ .

PROOF. We will use the following notations in the proof. Let  $e_i$  be the unit vector with  $i^{\text{th}}$  component 1 and the rest 0;  $e_{ij} = e_i + e_j$

1. This is shown by demonstrating  $n + 1$  affinely independent points in  $P_k(G)$ . The points  $\mathbf{0}, e_1, e_2, \dots, e_n$  are clearly  $n+1$  affinely independent points in  $P_k(G) \subset \mathbb{R}^n$ . Hence,  $\dim(P_k(G)) = n$ .
2. Let  $F = \{x \in P_k(G) : x_i = 0\}$ . Since an empty set or any vertex by itself is a  $k$ -plex, we have  $\mathbf{0}, e_j$  for all  $j \in V \setminus \{i\}$  forming  $n$  affinely independent points in  $F$ . This shows that  $\dim(F) = n - 1$  and it is a facet.
3. Let  $F' = \{x \in P_k(G) : x_i = 1\}$ . We first observe that every vertex and any pair of vertices form a  $k$ -plex for any  $k$  such that  $1 < k < n$ . Then  $e_i$  and  $e_{ij}$  for all  $j \in V \setminus \{i\}$  form  $n$  affinely independent points in  $F'$ , indicating that  $\dim(F') = n - 1$  and it is a facet.  $\square$

### 4.3 Valid inequalities

The following valid inequalities are derived by identifying induced subgraphs that are not  $k$ -plexes and hence cannot be present in any  $k$ -plex. Although the result that every vertex-induced subgraph of a  $k$ -plex is a  $k$ -plex is presented in [Seidman and Foster, 1978], the following explanation is given here for the sake of clarity. Note that if  $G$  is a  $k$ -plex, the minimum degree  $\delta(G)$  is at least  $n - k$ . In the graph  $G'$  obtained by deleting any vertex from  $G$ , the degree of all the vertices and hence the minimum degree, can drop by at most 1. Hence, the new graph continues to be a  $k$ -plex as  $\delta(G') \geq (n - 1) - k$ . Also note that any  $k$ -element subset of vertices is a  $k$ -plex and any  $k$ -plex is also a  $k + r$ -plex ( $1 \leq r \leq n - k$ ). Before we introduce the first family of inequalities, we present the following two lemmas.

**Lemma 1** *Let  $k$  be even. Then, there does not exist a co- $k$ -plex that contains a  $k$ -plex of size  $2k - 1$ .*

PROOF. Let  $G$  be a co- $k$ -plex on  $n$  vertices. Assume that  $n \geq 2k - 1$  as the result is trivial otherwise. Now suppose that  $S$  is a  $k$ -plex of size  $2k - 1$  in  $G$ . Then we have

$$|N(i) \cap S| \geq 2k - 1 - k = k - 1 \quad \forall i \in S.$$

Since  $G$  is co- $k$ -plex we have,

$$|N(i) \cap S| \leq |N(i)| \leq k - 1 \quad \forall i \in S.$$

The two conditions then imply that the induced graph  $G[S]$  is regular with all degrees equal to  $k - 1$  and is of order  $2k - 1$ . But  $k - 1$  is odd and we cannot have an odd number of vertices of odd degree. This contradiction establishes that  $S$  does not exist.  $\square$

Note that this bound is sharp since the graph family  $G_k = K_k \cup K_{k-1}$ , the union of complete graphs, for each  $k$  forms a co- $k$ -plex of size  $2k - 1$ , which contains  $K_{k-1} \cup K_{k-1}$ , a  $k$ -plex of size  $2k - 2$ . Figure 5 illustrates this when  $k = 4$ .

**Lemma 2** *Let  $k$  be odd. Then, there does not exist a co- $k$ -plex that contains a  $k$ -plex of size  $2k$ .*

PROOF. As before, let  $G$  be a co- $k$ -plex on  $n$  vertices ( $n \geq 2k$ ). Suppose that  $S$  is a  $k$ -plex of size  $2k$  in  $G$ . Then we have

$$|N(i) \cap S| \geq 2k - k = k \quad \forall i \in S.$$

Since  $G$  is co- $k$ -plex we have,

$$|N(i) \cap S| \leq |N(i)| \leq k - 1 \quad \forall i \in S.$$

This contradiction establishes that  $S$  does not exist. □

This bound is also sharp since the following family of graphs have  $2k$  vertices forming a co- $k$ -plex containing a  $k$ -plex of size  $2k - 1$ . Construct the graph  $G_k = (V, E)$ , where

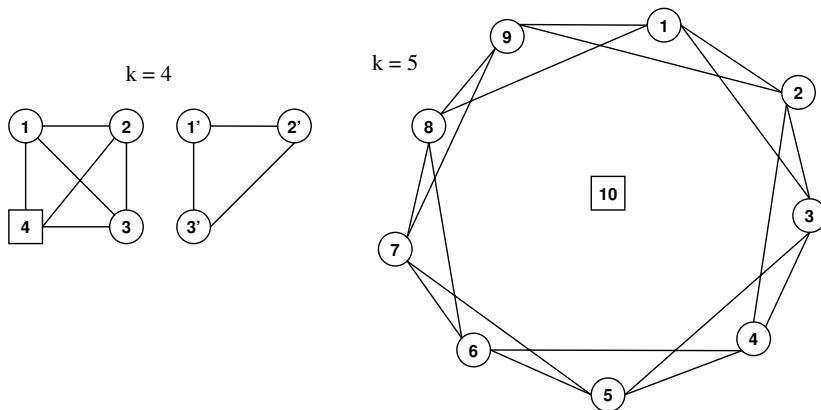
$$V = V' \cup \{2k\}, \quad V' = \{1, \dots, 2k - 1\},$$

and

$$E = \{(i, j) : i \in V' \text{ and } j = i + 1, \dots, \left(i + \frac{k - 1}{2}\right) \bmod (2k - 1)\}.$$

Maximum degree in  $G_k$  is  $k - 1$  and hence it is a co- $k$ -plex of order  $2k$ . The induced subgraph  $G_k[V']$  is a  $(k - 1)$ -regular  $k$ -plex of order  $2k - 1$  in which every vertex has exactly  $k - 1$  neighbors and non-neighbors each. It is also known as an *antiweb* and its complement is known as a *web*. Webs were introduced by Trotter [1975] to generalize odd hole and antihole inequalities developed by Padberg [1973] for the independent set polytope. Figure 5 illustrates this when  $k = 5$ .

Figure 5: Graphs demonstrating the sharpness of the bounds in Lemmas 1 and 2. The circled vertices form the said  $k$ -plexes



We next present three types of valid inequalities for the  $k$ -plex polytope: Independent set inequalities, hole inequalities, and co- $k$ -plex inequalities.

**Independent Set Inequalities.** Let  $I \subseteq V$  be an independent set. Note that no  $k$ -plex can contain an independent set of more than  $k$  vertices as  $k + 1$  or more independent vertices do not form a  $k$ -plex. Let  $\mathcal{I}_{k+1}$  represent the collection of all maximal independent sets of size  $k + 1$  or more in  $G$ . Then we have the following family of valid inequalities:

$$\sum_{i \in I} x_i \leq k \quad \forall I \in \mathcal{I}_{k+1}. \tag{5}$$

**Hole Inequalities.** Let  $H \subseteq V$  be a hole (induced chordless cycle). If  $|H| \leq k + 2$ , then  $H$  is a  $k$ -plex. Now suppose  $|H| > k + 2$ , then  $H$  is not a  $k$ -plex and for every proper subset  $S \subset H$ , we have  $\delta(G[S]) \leq 1$ . Hence, if  $|S| - k \geq 2$ ,  $S$  is not a  $k$ -plex. Thus, any  $k$ -plex can contain at most

$k + 1$  vertices from the hole and this bound is sharp. Let  $\mathcal{H}_{k+3}$  represent the collection of all holes of size  $k + 3$  or more in  $G$ . Then we have the following family of valid inequalities:

$$\sum_{i \in H} x_i \leq k + 1 \quad \forall H \in \mathcal{H}_{k+3}. \quad (6)$$

**Co- $k$ -plex Inequalities.** Lemmas 1 and 2, when combined, imply that the size of a maximum  $k$ -plex in any co- $k$ -plex is less than or equal to  $r_k = 2k - 1 - \frac{1+(-1)^k}{2}$ . Let  $\mathcal{J}_{r_k+1}$  represent the collection of all maximal co- $k$ -plexes of size more than  $r_k$  in  $G$ . We have the following family of valid inequalities:

$$\sum_{i \in J} x_i \leq r_k \quad \forall J \in \mathcal{J}_{r_k+1}. \quad (7)$$

Observe that, if  $J$  is a *maximal* co- $k$ -plex, there does not exist another co- $k$ -plex of which  $J$  is a proper subset. Then, for every  $v \in V \setminus J$ , at least one of the following conditions must hold.

1.  $\exists j \in J \cap N(v)$  such that  $|N(j) \cap J| = k - 1$  and including  $v$  would cause degree of  $j$  in the induced subgraph to be  $k$ .
2.  $|N(v) \cap J| \geq k$  and hence upon inclusion  $v$  would have degree  $k$  or more in the induced subgraph.

The next theorem uses this observation to show that for  $k = 2$  the co-2-plex inequalities actually form facets for the 2-plex polytope,  $P_2(G)$ .

**Theorem 5** *Let  $P_2(G)$  denote the 2-plex polytope described in Section 4.2. Then, the co-2-plex inequality given by*

$$\sum_{i \in J} x_i \leq 2, \quad (8)$$

where  $J$  is a maximal co-2-plex with  $|J| \geq 3$ , induces a facet of  $P_2(G)$ .

PROOF. First, recall that any 2 vertices from  $J$  form a 2-plex. Second, for every  $v \in V \setminus J$ , the above two conditions for a maximal co-2-plex imply the existence of two vertices  $u, w \in J$  such that  $\{v, u, w\}$  is a 2-plex. Indeed, if the first case holds, let  $u \in J \cap N(v)$ , then  $N(u) \cap J = \{w\}$  and  $\{v, u, w\}$  is a 2-plex. If the second case holds,  $\{u, w\} \subseteq J \cap N(v)$  and again  $\{v, u, w\}$  is a 2-plex. We use these observations to construct  $n$  affinely independent vectors that lie on the face defined by  $F = \{x \in P_2(G) : \sum_{i \in J} x_i = 2\}$ , so  $F$  is  $(n - 1)$ -dimensional and hence it is a facet. Without

loss of generality, assume that  $J = \{1, \dots, r\}$  and  $V \setminus J = \{r + 1, \dots, n\}$ , where  $r \geq 3$ . Let, as before,  $e_i \in \mathbb{R}^n$  denote the unit vector with  $i^{\text{th}}$  component one and all others zero. The said affinely independent vectors  $x^1, \dots, x^n$  are constructed as follows.

$$x^v = e_v + e_r, \quad \forall v = 1, \dots, r - 1;$$

$$x^r = e_1 + e_2 \text{ (note that } x^r \text{ is distinct from } x^1, \dots, x^{r-1} \text{ as } r \geq 3\text{);}$$

$$x^v = e_v + e_u + e_w, \quad \forall v = r + 1, \dots, n, \text{ where for each } v \in V \setminus J, u, w \in J \text{ are the particular vertices described before. Clearly, } x^v \in F \text{ and it can be easily verified that these vectors are affinely independent. Thus, the co-2-plex inequalities produce facets for the 2-plex polytope. } \square$$

Although co- $k$ -plex inequalities form facets of  $P_k(G)$  for  $k = 1, 2$ , they do not for  $k \geq 3$ . Consider a graph  $G = (V, \emptyset)$  with at least  $k$  vertices. Note that  $G$  is a co- $k$ -plex and the corresponding inequality  $\sum_{i \in V} x_i \leq r_k$  is not supporting since  $\omega_k(G) = k$  and there is no  $x \in P_k(G)$  that satisfies it at equality. Hence, these inequalities do not form facets of  $P_k(G)$  for all  $G$ . This is in contrast

to the results known for  $k = 1, 2$ . The reason is  $r_k = k$  for  $k = 1, 2$  and every graph  $G$  with at least  $k$  vertices has a  $k$ -plex of size  $r_k = k$ . The next natural question, if they form facets when  $G$  is a co- $k$ -plex with  $\omega_k(G) = r_k$ ,  $k \geq 3$  is also settled in the negative by the following counterexamples. Assume that  $k$  is even. Construct graphs  $G$  of arbitrary order  $n \geq r_k$  as the union of  $n - r_k$  clique components of size one and two clique components of size  $k - 1 = r_k/2$ . Then  $G$  is a co- $k$ -plex with the two “large” clique components forming a  $k$ -plex of size  $r_k$ . Suppose  $F = \{x \in P_k(G) : \sum_{i \in V} x_i = r_k\}$  is a facet of  $P_k(G)$ . Since  $P_k(G)$  is an integral polytope, the extreme points of  $F$  are also integral and  $F$  is a convex hull of those integral vectors. Consider one such binary vector  $x^o \in F$ . If  $x_i^o = 1$  for some  $i$  that is a one-vertex clique component of  $G$ , for  $x^o$  to be feasible we have  $\sum_{j \in V \setminus N[i]} x_j^o \leq k - 1$ . But since  $V \setminus N[i] = V \setminus \{i\}$ , we have  $\sum_{i \in V} x_i^o \leq k$ , which contradicts the fact that  $x^o \in F$  as  $r_k > k$ . Hence, the components of extreme points of  $F$  corresponding to one-vertex components of  $G$  are all zeros. Hence, there exists *exactly one extreme point* in  $P_k(G)$  that satisfies  $\sum_{i \in V} x_i \leq r_k$  at equality, which is the characteristic vector of  $K_{k-1} \cup K_{k-1}$ . Thus,  $F$  is 0-dimensional and not a facet.

For odd  $k$ , we can have arbitrarily large graphs by adding single vertex components to the antiweb  $G_k[V']$  constructed before. By using similar arguments, we can again show that there exists only one point in the  $k$ -plex polytope that satisfies the co- $k$ -plex inequality at equality. From these observations we can conclude that  $\omega_k(G) = r_k$  is only a necessary condition for the co- $k$ -plex inequality to induce a facet of  $P_k(G)$ . Identifying co- $k$ -plex graphs for which the co- $k$ -plex inequalities form facets of the  $k$ -plex polytope is important as these inequalities could be lifted to yield facets for a graph containing such structures. It is also a well-known fact that a graph is perfect if and only if its clique polytope is completely characterized by all the maximal independent set inequalities and non-negativity constraints [Cornuéjols, 2001]. Similarly, we could explore  *$k$ -plex perfectness* of graphs whose  $k$ -plex polytope can be completely described by the co- $k$ -plex inequalities described here and the trivial facets. This is an interesting topic for future research.

## 5 Branch & Cut Framework

Branch-and-cut (BC) methods are popular and effective in optimally solving a wide variety of combinatorial optimization and general integer programming problems [Padberg and Rinaldi, 1991, Applegate et al., 1998, Balas et al., 1996c,b,a]. These methods incorporate cutting planes in solving the linear programming (LP) relaxation at the nodes of a branch-and-bound tree to get tighter bounds. Although BC methods have been successfully applied to solve several hard combinatorial optimization problems, tailoring a BC algorithm to effectively solve a specific problem is a delicate task that requires attention in itself in terms of extensive experimentation and tuning. For more information on BC methods and for other useful references, see [Mitchell, 2002].

This section describes a branch-and-cut (BC) implementation incorporating the maximal independent set (MIS) cuts for the maximum  $k$ -plex problem. The aim of this part of the paper is to judge the effectiveness of MIS cuts which can be easily and quickly generated, in solving the problem of interest, the order and size of instances that can be solved under this framework, and to provide some benchmark instances for this problem. The experiments were conducted for  $k = 1$  and 2. Even though the case of  $k = 1$  corresponds to the well-researched maximum clique problem, these results illustrate the difference in the performance of our approach for two consecutive values of  $k$ . In addition, we describe our computational experience in solving the maximum  $k$ -plex problem on real-life *scale-free graphs* and the preprocessing techniques developed for the same.

## 5.1 Implementation Details

Details that are common to all our experiments are the following. All numerical experiments were conducted on *Dell Optiplex GX240*<sup>®</sup> computers with 2.20GHz PENTIUM IV<sup>®</sup> processor, 512MB RAM and 40GB HDD. The core of all our algorithms is a BC approach, implemented using ILOG CPLEX 9.0<sup>®</sup> [ILOG]. The biggest advantage of using the framework provided by CPLEX is the effective default settings that take care of the branching process, node selection, variable selection, primal heuristics, pre-solving among others, while the bounding is done by solving the LP relaxation with the user specified cuts.

MIS *local cuts*, which are valid at the node in which they are generated and for the sub-tree rooted at that node, are also generated in a greedy fashion and implemented using the CPLEX *goals* feature. The greedy algorithm starts by adding an arbitrary vertex and then removes its neighbors from the graph. Then it proceeds in a similar fashion by adding a vertex of minimum degree in the residual graph to the MIS and removing its neighbors until there are no more vertices to add. Local cuts were generated every SKIPFACTOR (constant set at 32) number of nodes in the BC tree. While generating local cuts, the vertices corresponding to variables fixed at zero in that node are deleted from the graph before MIS are found, and only variables with high fractional value ( $\geq$  HIFRACVAL, constant set at 0.75) are used as starting vertices for finding MIS. We ensure that the *round of cuts* generated are distinct, violated by the LP optimum and always fewer than MAXLOCALCUTSPERNODE (constant set at 3). All such violated MIS inequalities are added to the system and CPLEX re-solves the problem at that node and handles the cut management from that point onwards.

It should be noted that CPLEX generates its own classes of cuts to solve any given MIP. Based on preliminary experiments, we observed that adding a small number of Gomory fractional cuts slightly improved the runtimes, but the other CPLEX cuts were ineffective. However, all CPLEX cuts were turned off for our experiments and only MIS local cuts were used in the BC algorithm. In addition, the number of rows in the problem with cuts added is limited to 3 times the original number of rows by setting the CPLEX parameter *CutsFactor* to 3.

Apart from regular termination of an MIP (optimal or infeasible), CPLEX can be forced to terminate gracefully returning the best integer feasible solution and objective (if found) as well as a bound on the optimum, when an upper time limit is reached by setting the CPLEX parameter *TiLim* to the desired value (set at 3 hours). By setting CPLEX parameter *NodeFileInd* to 2, CPLEX can be forced to write the BC tree to hard disk without any compression. This enabled CPLEX to proceed without any memory shortage as the BC tree grows exponentially in size, without significant increase in runtime [ILOG].

The implementation described above helps us judge the effectiveness of MIS cuts without intensifying the BC algorithm. Note however that, several critical issues in designing a more sophisticated BC algorithm, such as using a dynamic global cut pool instead of the fixed size option offered by CPLEX and lifting local cuts so that they are globally valid can be considered. Even some of the basic settings can be improved such as using methods other than greedy algorithm to generate MIS cuts, distinguishing and selecting cuts from the violated cuts based on quality measures (such as maximum violation or maximum depth which is the Euclidean distance of the cut from the the point being separated), varying the frequency at which cuts are added dynamically (so that more cuts are generated and applied where the violation is “high”) and the size of the cut pool that is applied and its management. These are directions that need to be carefully considered in the future in order to develop a more powerful BC algorithm for this problem.

## 5.2 Pre-processing and Variable Fixing

**Peeling.** The peeling procedure (similar to the one used in Abello et al. [1999] for the maximum clique problem) removes vertices of low degree based on the size of a known  $k$ -plex,  $S$ . Vertices that cannot belong to an optimal solution are identified and removed recursively based on the necessary condition that every vertex in an optimal  $k$ -plex has degree at least  $|S| - k$ . For our implementation, the initial  $k$ -plex  $S$  is a greedily found maximal clique, and the following peeling procedure is called before the BC algorithm.

---

### Algorithm 1 Peeling Procedure

---

```

1: procedure PEEL( $G, S$ )
2:    $X \leftarrow \{v \in V(G) : deg_G(v) \leq |S| - k\}$ 
3:   if  $X \neq \emptyset$  then
4:      $G \leftarrow G[V \setminus X]$ ; go to step 2
5:   end if
6:   return  $G$ 
7: end procedure

```

---

**Non-dominated Variable Fixing.** Recall from Theorem 2 that any  $k$  vertices in a  $k$ -plex form a dominating set. Thus, in the BC tree, when  $k$  variables are fixed to one for the first time in node  $N$ , every vertex that is not adjacent to any of the  $k$  vertices can be fixed to zero for the subtree rooted at node  $N$ . This is valid since such vertices cannot belong to a  $k$ -plex containing the fixed  $k$  vertices as they are not dominated by them.

## 5.3 Description of the Test-bed

The test-bed of instances used in our experiments consists of two broad groups. The first group consists of graphs of various order and size generated using *Sanchis generators* [Sanchis and Jagota, 1996] and benchmark clique instances from the Second DIMACS challenge [DIMACS, 1995, Johnson and Trick, 1996]. The second group of instances are a set of Erdős collaboration networks [Grossman et al., 1995]; protein interaction networks of the yeast *Saccharomyces cerevisiae* [Jeong et al., 2001] and the gastric pathogen *Helicobacter Pylori* [Rain et al., 2004, BRITE]; a collaboration network of authors in computational geometry available from [Batagelj and Mrvar, 2006]; and a text-mining network based on *Reuters* terror news reporting following the tragic events of September 11 available from [Batagelj and Mrvar, 2006].

**Group I.** The Sanchis generator available at [DIMACS, 1995] produces graphs with known maximum clique size with a specified number of vertices, edges and a *construction parameter*,  $r$ . In our experiments, the maximum clique size was fixed at  $\lceil \frac{n}{3} \rceil$  (where  $\lceil a \rceil$  is the smallest integer greater than or equal to  $a$ ) and the construction parameter  $r$ , which has to be an integer from interval  $[0, \frac{n}{\omega(G)})$ , was set at 1. The number of vertices in the generated Sanchis graphs was varied from 60 to 1000 (not uniformly) and the edge density ( $d$ ) was varied from 0.4 to 0.9 resulting in a total of 108 instances. The number of edges was calculated as  $\lfloor \frac{dn(n-1)}{2} \rfloor$ , where  $\lfloor a \rfloor$  is the largest integer less than or equal to  $a$ . Table 1 presents information regarding the 20 DIMACS benchmark instances used in our experiments. Note that some of the DIMACS instances are also graphs arising in various applications. *Johnson* and *Hamming* graph families arise in coding theory and *c-fat* graphs arise

Table 1: DIMACS benchmarks.

Graphs	$ V $	$ E $	$d$
c-fat200-1.clq	200	1534	0.077
c-fat200-2.clq	200	3235	0.163
c-fat200-5.clq	200	8473	0.426
c-fat500-1.clq	500	4459	0.036
c-fat500-2.clq	500	9139	0.073
c-fat500-5.clq	500	23191	0.186
c-fat500-10.clq	500	46627	0.374
hamming6-2.clq	64	1824	0.905
hamming6-4.clq	64	704	0.349
hamming8-2.clq	256	31616	0.969
hamming8-4.clq	256	20864	0.639
hamming10-2.clq	1024	518656	0.990
hamming10-4.clq	1024	434176	0.829
johnson8-2-4.clq	28	210	0.556
johnson8-4-4.clq	70	1855	0.768
MANN_a9.clq	45	918	0.927
MANN_a27.clq	378	70551	0.990
MANN_a45.clq	1035	533115	0.996
san200_0.7_2.clq	200	13930	0.700
keller4.clq	171	9435	0.649

Table 2: Group II Instances.

Graph	$ V $	$ E $	$d$
COMP-GEOM-0.PAJ	7343	11898	0.000441383
COMP-GEOM-1.PAJ	7343	3939	0.000146126
COMP-GEOM-2.PAJ	7343	1976	7.33E-05
ERDOS-97-1.NET	472	1314	0.0118212
ERDOS-98-1.NET	485	1381	0.0117662
ERDOS-99-1.NET	492	1417	0.0117315
ERDOS-97-2.NET	5488	8972	0.0005959
ERDOS-98-2.NET	5822	9505	0.0005609
ERDOS-99-2.NET	6100	9939	0.0005343
H. Pylori	1570	1399	0.00113586
S. cerevisiae	2112	2203	0.00098824
DAYS-3.PAJ	13332	9000	0.000101278
DAYS-4.PAJ	13332	5159	5.81E-05
DAYS-5.PAJ	13332	3404	3.83E-05

in fault diagnosis. Description of these and other DIMACS instances can be found in [Hasselberg et al., 1993, Bomze et al., 1999, DIMACS, 1995].

**Group II.** The collaboration network of authors in computational geometry has vertices representing authors from this area, and for every two authors the number of joint works is available. This permits us to use a threshold for the edge to be included in the graph. Two authors are connected by an edge if they have (strictly) more than *threshold* joint works. We consider these graphs for *threshold*  $t = 0, 1, 2$  resulting in three instances named COMP-GEOM- $t$ .PAJ. Erdős collaboration networks are similar; here vertices represent mathematicians and an edge indicates that the mathematicians represented by the endpoints have published together. But these collaboration networks are centered around Paul Erdős and *Erdős number* of a mathematician is his or her shortest distance to Erdős in this network. We used the following Erdős collaboration networks available from [Batagelj and Mrvar, 2006, Grossman et al., 1995] in our experiments: ERDOS- $x - y$ .NET, where  $x$  represents the last two digits of the year for which the network was constructed, and  $y$  represents the largest *Erdős number* of a mathematician in that graph. We considered six such networks for years 1997-1999 and  $y = 1$  and 2. We will refer to ERDOS- $x - y$ .NET graph as the  $y$ -neighborhood Erdős network for year  $x$ . Note that in the instances we used the vertex corresponding to Erdős himself is excluded. Apart from the aforementioned social networks the following two biological networks, protein interaction networks of *H. Pylori* and *S. cerevisiae* were also used in testing. In these graphs, the vertices represent proteins and edges indicate that the pair of proteins forming the end points are known to interact. The text-mining network (DAYS.NET) from [Batagelj and Mrvar, 2006] is based on all stories released during 66 consecutive days beginning

at 9:00 AM EST 9/11/01 by the news agency *Reuters* concerning the September 11 attack. The network is based on information compiled by Steve Corman, Kevin Dooley and Robert McPhee at the LOCKS labs in Arizona State University [Corman et al., 2006, 2002]. The vertices of the network are selected words that appeared in the news. There is an edge between two words if they appear in the same text unit (sentence), and the edges are weighted with the number of co-appearances of its end-points. We use a threshold model as before and DAYS- $t$ .PAJ refers to this network with only edges of edge weight at least  $t + 1$  included. Graphs were constructed for  $t = 3, 4, 5$ .

Apart from the fact that Group II graphs are constructed from real-life data, another commonality among all these graphs is that they are extremely large and extremely sparse graphs that obey a power-law degree distribution. Such graphs are called scale-free graphs studied extensively recently [Barabási and Albert, 1999, Barabási et al., 2000b, Almaas and Barabási, 2005, Albert et al., 2000, Barabási et al., 2000a]. Table 2 presents information regarding these instances. It should be noted that in experiments with scale-free graphs our goal was to solve the maximum  $k$ -plex problem to optimality on large-scale, real-life instances. On the other hand, in the set of experiments with DIMACS and Sanchis graphs our intention was to get a sense of the influence of order and density of graphs on the BC algorithm and the effectiveness of MIS cuts.

#### 5.4 Computational Experience: Group I

We solve the maximum  $k$ -plex problem with  $k = 1, 2$  on Group I instances with the BC settings described in Section 5.1. First we apply the peeling procedure described in Section 5.2 before the BC algorithm, and the BC implementation includes variable fixing described therein. Table 3 and Table 4 report the runtime of Peel and BC (PBC) algorithm on Sanchis graphs to solve the  $k = 1$  and  $k = 2$  cases respectively. We indicate non-optimal terminations by † and unattempted instances by “-”. Whenever we observed CPLEX reach the upper time limit for a particular density for two consecutive orders, the runs for that density in higher order graphs were not conducted. Note that in some cases there are more than one non-optimal instances of consecutive orders for a particular density as the runs were conducted in batches. Decision to not attempt higher orders of that density was made once that batch run was completed.

It was observed that peeling was fast, but only effective on sparser Sanchis graphs with density 0.4 and 0.5. On graphs of higher density, the number of vertices was not reduced. But this observation was also encouraging since it can be expected to perform well on Group II instances. We do not report peeling times as they were under 20 seconds in all cases.

While solving the maximum 1-plex problem on Sanchis graphs, we observed 16 non-optimal terminations (including unattempted). We had 12 non-optimal instances with  $d = 0.9$ , 2 with  $d = 0.8$  and 1 each with  $d = 0.5$  and  $d = 0.6$ . Recall that  $\omega_1(G) = \omega(G) = \lceil \frac{n}{3} \rceil$  for these instances. It was observed that a maximum clique was indeed found in all cases we failed to solve to optimality. The respective optimality gaps are also included in the Table 3. In some cases, peeling procedure removed all the vertices in the original instance. These have zero running time and these are in fact the only cases where peeling contributed significantly. For  $k = 2$ , 25 instances were not optimally solved, primarily in the density 0.8 and 0.9 category and runtimes were also higher compared to the case of  $k = 1$  for all instances. As before, peeling was effective only on instances with density 0.4. The instances that were reduced to zero vertices when  $k = 1$  were again reduced to zero except for the 900-vertex instance, which was reduced to 600 vertices. Note that peeling procedure also depends on  $k$  value even though the same initial clique size is provided in both  $k = 1, 2$  cases. In Table 5, we present the 2-plex number (or the best 2-plex size) found by PBC algorithm for each Sanchis instance. For non-optimally solved cases, we report  $[l, u]$  where  $l$  is the size of best 2-plex

Table 3: Runtime in seconds for solving maximum 1-plex problem using PBC algorithm on Sanchis graphs.

$n$	$d = 0.4$	$d = 0.5$	$d = 0.6$	$d = 0.7$	$d = 0.8$	$d = 0.9$
60	0	0.109	0.266	0.266	0.296	1.25
80	0.359	0.328	0.328	0.329	0.5	6.344
100	0.579	0.625	0.328	0.344	0.969	37.001
120	0.75	0.812	0.844	0.766	1.609	242.737
140	1.547	1.625	1.531	1.36	2.157	1373.1
160	2.406	1.813	1.844	1.969	3.281	1763.77
180	4.422	3.375	3.328	2.906	6.047	10800.1 <sup>†</sup> (3.3%)
200	0	0.046	4.78	4.296	9.594	10800.4 <sup>†</sup> (6%)
250	16.94	18.545	11.218	9.391	25.594	-
300	0	23.938	20.328	28.047	66.345	-
350	0	37.891	39.375	52.673	140.486	-
400	74.22	111.798	121.97	65.422	208.127	-
500	0	120.299	152.986	231.644	795.964	-
600	231.535	459.743	434.535	646.347	2432.02	-
700	424.099	865.512	727.057	1344.6	5096.52	-
800	693.399	10801.5 <sup>†</sup> (9.7%)	10800.3 <sup>†</sup> (6%)	5736.85	10529.2	-
900	0	1814.69	1850.12	4364.82	10801.1 <sup>†</sup> (8.3%)	-
1000	1199.2	3036.01	3423.54	9055.25	10800.3 <sup>†</sup> (15.3%)	-

found and  $u$  is an upper-bound on  $\omega_2(G)$ .

The PBC implementation was also tested on established DIMACS benchmarks listed in Table 1. These results are presented in Table 6 where <sup>†</sup> indicates non-optimal terminations and the interval containing the optimum is reported for such cases.

## 5.5 Computational Experience: Group II

The results presented for the Group I instances indicate that MIS cuts and the BC framework are quite effective, especially on sparse graphs. We retained the same settings in our early attempts to solve the problem on Group II instances. Although they were successful on Erdős and protein networks, we faced memory issues with the others. Peeling by itself was not sufficient to reduce the size of computational geometries and Reuters news networks, resulting in a very large, very sparse graph given to CPLEX. The corresponding integer program built by CPLEX was dense and huge leading to a memory crash even before the root node solution was attempted. Hence we developed an iterative procedure which took advantage of another property of  $k$ -plexes in addition to peeling to solve maximum  $k$ -plex problem on many small graphs instead of one large graph. The settings for the BC procedure invoked in each iteration is the same as described in Section 5.1.

**IPBC Algorithm.** The iterative version of PBC algorithm is the IPBC Algorithm in which the basic idea is to fix a vertex  $v \in V$  in each iteration to find a maximum  $k$ -plex containing  $v$ . Here we assume that the largest  $k$ -plex in the graph  $S^*$  is large enough, *i.e.*,  $|S^*| > 2k - 2$ . If the assumption holds, and the fixed vertex  $v$  is in  $S^*$ , we are guaranteed by Theorem 2 that  $S^* \subseteq N_2[v]$  as  $\text{diam}(G[S^*]) \leq 2$ , where the 2-neighborhood of a vertex  $v$  in  $G$  is given by  $N_2[v] = \{i \in V(G) :$

Table 4: Runtime in seconds for solving maximum 2-plex problem using PBC algorithm on Sanchis graphs.

$n$	$d = 0.4$	$d = 0.5$	$d = 0.6$	$d = 0.7$	$d = 0.8$	$d = 0.9$
60	0	0.156	0.485	1.766	39.735	39.75
80	0.562	0.469	0.531	2.562	139.267	10800.3 <sup>†</sup>
100	2.109	1.094	1.047	4.782	64.829	10800.6 <sup>†</sup>
120	2.673	2.032	2.266	8.314	71.314	10800.4 <sup>†</sup>
140	3.781	2.939	3.404	11.013	71.936	10800.4 <sup>†</sup>
160	4.969	4.877	5.015	16.751	67.093	10800.5 <sup>†</sup>
180	7.327	8.72	10.673	36.187	132.457	10800.4 <sup>†</sup>
200	0	0.83	15.012	51.83	190.019	10800.4 <sup>†</sup>
250	19.297	21.516	26.266	110.141	348.486	10800.6 <sup>†</sup>
300	0	35.705	60.893	266.16	1025.52	10800.7 <sup>†</sup>
350	0	60.843	91.394	443.88	1759.3	10800.4 <sup>†</sup>
400	70.579	83.718	136.256	767.243	2819.15	10800.5 <sup>†</sup>
500	0	209.234	428.661	1862.08	8225.52	10800.6 <sup>†</sup>
600	250.566	428.328	802.203	3848.16	10800.1 <sup>†</sup>	-
700	407.268	724.614	1235.29	7643.69	10800.6 <sup>†</sup>	-
800	629.708	1390.81	2312.12	10800.4 <sup>†</sup>	10800.2 <sup>†</sup>	-
900	49.705	2320.16	4756.11	10800.5 <sup>†</sup>	-	-
1000	1323.75	3056.74	6822.67	-	-	-

Table 5: 2-Plex numbers of Sanchis graphs.

$n$	$d = 0.4$	$d = 0.5$	$d = 0.6$	$d = 0.7$	$d = 0.8$	$d = 0.9$
60	20	20	20	20	21	28
80	27	27	27	27	27	[33,36]
100	34	34	34	34	34	[38,45]
120	40	40	40	40	40	[43,54]
140	47	47	47	47	47	[48,64]
160	54	54	54	54	54	[54,70]
180	60	60	60	60	60	[60,78]
200	67	67	67	67	67	[67,85]
250	84	84	84	84	84	[84,105]
300	84	84	84	84	84	[100,127]
350	117	117	117	117	117	[117,149]
400	134	134	134	134	134	[134,172]
500	167	167	167	167	167	[167,219]
600	200	200	200	200	[200,213]	-
700	234	234	234	234	[234,266]	-
800	267	267	267	[267,277]	[267,316]	-
900	300	300	300	[300,321]	-	-
1000	334	334	334	-	-	-

Table 6: Results for PBC algorithm on DIMACS graphs.

Graph	$k = 1$		$k = 2$	
	BC Time (sec)	$\omega_1(G)$	BC Time (sec)	$\omega_2(G)$
c-fat200-1.clq	16.844	12	30.922	12
c-fat200-2.clq	19.688	24	31.672	24
c-fat200-5.clq	11.156	58	33.626	58
c-fat500-1.clq	221.095	14	557.1	14
c-fat500-2.clq	328.706	26	885.215	26
c-fat500-5.clq	555.269	64	1087.75	64
c-fat500-10.clq	279.705	126	1014.87	126
hamming6-2.clq	0.016	32	0.234	32
hamming6-4.clq	0.297	4	6.579	6
hamming8-2.clq	0.016	128	10800.3. <sup>†</sup>	[128,130]
hamming8-4.clq	10788.1	16	10800.4. <sup>†</sup>	[16,80]
hamming10-2.clq	0.187	512	10800.2. <sup>†</sup>	[512,534]
hamming10-4.clq	10800.2. <sup>†</sup>	[38,379]	10802.2. <sup>†</sup>	[45,458]
johnson8-2-4.clq	0.172	4	1.639	5
johnson8-4-4.clq	2.999	14	10800.3. <sup>†</sup>	[14,16]
MANN_a9.clq	0.094	16	0.144	26
MANN_a27.clq	10800.3. <sup>†</sup>	[125,148]	10800.5. <sup>†</sup>	[236,260]
MANN_a45.clq	10802. <sup>†</sup>	[342,422]	10800.8. <sup>†</sup>	[662,739]
san200_0.7_2.clq	10800.9. <sup>†</sup>	[17,27]	10800.4. <sup>†</sup>	[62,86]
keller4.clq	7510.53	11	10800.5. <sup>†</sup>	[40,45]

$d_G(v, i) \leq 2$ }. As we iterate over  $v \in V$  we only need to consider vertices in  $N_2[v]$ , assuming that there is a large  $k$ -plex. Under this assumption, for each  $v \in V$ , peeling procedure is called on the graph induced by  $N_2[v]$  along with the lower bound initialized as before with a maximal clique and then updated during the iterations. As before, BC is used after peeling to find the maximum  $k$ -plex containing  $v$  by adding the additional constraint  $x_v = 1$  to the system. The resulting solution is used to update the current best  $k$ -plex  $S$  if necessary. At the end of every iteration, vertex  $v$  can be removed from the graph, since from that point we are not interested in  $k$ -plexes containing  $v$ . Once the iterations are complete, if the best known  $k$ -plex was larger than  $2k - 2$ , our assumption was right and it can be returned as the optimal solution. If our assumption was incorrect, the solutions from the iterative procedure are not applicable and we re-solve maximum  $k$ -plex problem on the original graph with the additional constraint that  $\sum_{v \in V} x_v \leq 2k - 2$ . This is not especially disadvantageous since even a complete enumeration of all  $k$ -plexes of size  $k$  to  $2k - 2$  to find the optimum is still polynomial for fixed  $k$ . In this situation, the BC algorithm can easily find the optimum. Moreover,  $k$ -plex of size smaller than  $2k - 2$  might not be of much practical use since for most meaningful applications  $k$  is very low compared to  $n$ . Finally, the vertices are fixed in non-increasing order of vertex degrees, which ensured that a large  $k$ -plex was found early and subsequent preprocessing was effective. Algorithm 2 is the pseudo-code for this procedure.

---

**Algorithm 2** Iterative Peel-and-Branch-and-Cut (IPBC) Algorithm
 

---

```

1: procedure IPBC( $G$ )
2:    $\{v_1, \dots, v_n\}$  be vertices in non-increasing order of degrees
3:    $S \leftarrow S_o, G_o \leftarrow G$  ▷  $S_o$  is a greedily found maximal clique
4:   for  $i = 1$  to  $n$  do
5:     if  $|N_2[v_i]| > |S|$  then
6:        $\tilde{G} \leftarrow \text{PEEL}(G[N_2[v_i]], S)$ 
7:       if  $|V(\tilde{G})| > |S|$  then
8:          $\tilde{S} \leftarrow \text{BRANCH-AND-CUT}(\tilde{G}, x_{v_i} = 1)$  ▷ maximum  $k$ -plex containing  $v_i$ 
9:         if  $|\tilde{S}| > |S|$  then
10:            $S \leftarrow \tilde{S}$ 
11:         end if
12:       end if
13:     end if
14:      $G \leftarrow G - v_i$ 
15:   end for
16:   if  $|S| > 2k - 2$  then
17:     return  $S$  ▷ a maximum  $k$ -plex
18:   else
19:      $S \leftarrow \text{BRANCH-AND-CUT}(G_o, \sum_{v \in V} x_v \leq 2k - 2)$  ▷ assumption failed, re-solve
20:     return  $S$ 
21:   end if
22: end procedure

```

---

The approach taken in IPBC algorithm, such as simple checks on sizes in combination with peeling and the assumption of a large  $k$ -plex, is designed to enable us handle large sparse instances by decomposing the graph. The iterative scheme designed to look only at the 2-neighborhood of the fixed vertex in addition to peeling is necessary to build smaller IPs, avoid memory shortage and optimally resolve such instances by permitting longer preprocessing times instead. In our

Table 7: Group II: Number of vertices, edges, edge density, and  $k$ -plex numbers for  $k = 1, 2, 3$ .

Graph	$ V $	$ E $	$d$	$\omega_1(G)$	$\omega_2(G)$	$\omega_3(G)$
COMP-GEOM-0.PAJ	7343	11898	0.000441383	22	22	22
COMP-GEOM-1.PAJ	7343	3939	0.000146126	10	10	11
COMP-GEOM-2.PAJ	7343	1976	7.33E-05	8	8	10
ERDOS-97-1.NET	472	1314	0.0118212	7	8	9
ERDOS-98-1.NET	485	1381	0.0117662	7	8	9
ERDOS-99-1.NET	492	1417	0.0117315	7	8	9
ERDOS-97-2.NET	5488	8972	0.0005959	7	8	9
ERDOS-98-2.NET	5822	9505	0.0005609	7	8	9
ERDOS-99-2.NET	6100	9939	0.0005343	8	8	9
H. Pylori	1570	1399	0.00113586	3	5	6
S. cerevisiae	2112	2203	0.00098824	6	6	7
DAYS-3.PAJ	13332	9000	0.000101278	8	10	11
DAYS-4.PAJ	13332	5159	5.81E-05	7	8	9
DAYS-5.PAJ	13332	3404	3.83E-05	6	7	8

implementation, the only time limit imposed was via CPLEX parameter  $TiLim$  which was set to 3 hours as in PBC settings, but for each BC call. Hence the implementation in the worst-case could take  $3n$  hours to run. However, that was never observed in practice.

Table 7 presents the  $k$ -plex numbers for the Group II instances. Tables 8, 9 and 10 presents the runtime information for Group II instances. The column titled “IPBC Time” presents the total running time of the IPBC algorithm, “BC Time” presents the cumulative time spent on all BC calls to CPLEX and “#BC Calls” is the number of BC calls made to CPLEX for each instance. The runtime information is not reported for both H. Pylori and S. cerevisiae since these instances were resolved in under a total of 30 and 70 seconds respectively for all  $k$  values. It is clear from the results that very few BC calls are made compared to the number of vertices in each instance and the BC runtimes are relatively low. The burden of runtime has shifted from the BC procedure to the preprocessing as we expected, which appears to be effective in reducing the size of the instance solved in each BC call. The IPBC algorithm was quite successful in optimally resolving Group II instances.

## 6 On Maximum Clique Problem

Consider the following 0-1 formulation of the maximum clique problem on  $G = (V, E)$ . Let  $\bar{d}_i = |V \setminus N[i]|$  as before.

$$\omega(G) = \max \sum_{i \in V} x_i \quad (9)$$

subject to:

$$\sum_{j \in V \setminus N[i]} x_j \leq \bar{d}_i(1 - x_i) \quad \forall i \in V, \quad (10)$$

$$x_i \in \{0, 1\} \quad \forall i \in V. \quad (11)$$

Table 8: Runtime in seconds for Erdős networks using IPBC algorithm.

$k$	Graph	IPBC Time	BC Time	#BC Calls
1	ERDOS-97-1.NET	3.438	0.547	10
	ERDOS-98-1.NET	3.281	0.5	9
	ERDOS-99-1.NET	3.25	0.392	9
	ERDOS-97-2.NET	675.828	2.201	16
	ERDOS-98-2.NET	908.969	1.047	16
	ERDOS-99-2.NET	1058.89	1.123	14
2	ERDOS-97-1.NET	4.031	1.859	6
	ERDOS-98-1.NET	7.734	5.344	6
	ERDOS-99-1.NET	8.547	6.032	6
	ERDOS-97-2.NET	754.266	88.217	7
	ERDOS-98-2.NET	981.704	102.141	7
	ERDOS-99-2.NET	1151.31	101.702	9
3	ERDOS-97-1.NET	6.391	4.141	6
	ERDOS-98-1.NET	7.547	5.124	6
	ERDOS-99-1.NET	8.203	5.639	6
	ERDOS-97-2.NET	1007.7	342.936	7
	ERDOS-98-2.NET	1543.52	653.875	7
	ERDOS-99-2.NET	1891.3	837.015	9

Table 9: Runtime in seconds for computational geometers collaboration networks using IPBC algorithm.

$k$	Graph	IPBC Time	BC Time	#BC Calls
1	COMP-GEOM-0.PAJ	7381.83	1.235	1
	COMP-GEOM-1.PAJ	1360.36	0.375	3
	COMP-GEOM-2.PAJ	927.078	0.203	2
2	COMP-GEOM-0.PAJ	7888.33	498.656	1
	COMP-GEOM-1.PAJ	1409.63	58.5	2
	COMP-GEOM-2.PAJ	930.172	6.765	3
3	COMP-GEOM-0.PAJ	7829.72	360.312	1
	COMP-GEOM-1.PAJ	1407.47	54.297	2
	COMP-GEOM-2.PAJ	927.875	8.453	1

Table 10: Runtime in seconds for the Reuters terror news networks using IPBC algorithm.

$k$	Graph	IPBC Time	BC Time	#BC Calls
1	DAYS-3.PAJ	7600.34	66.516	33
	DAYS-4.PAJ	5634.09	13.609	23
	DAYS-5.PAJ	5068.78	0.921	17
2	DAYS-3.PAJ	8421.91	1002.91	28
	DAYS-4.PAJ	5842.64	210.423	21
	DAYS-5.PAJ	5046.03	60.375	17
3	DAYS-3.PAJ	9628.45	2605.95	25
	DAYS-4.PAJ	6036.13	505.735	20
	DAYS-5.PAJ	5069.31	68.375	17

This formulation is a special case of the maximum- $k$ -plex formulation (2) - (4) presented before with  $k = 1$ , as  $\omega_1(G) = \omega(G)$ . Constraint (10) ensures that no non-neighbor of a vertex is included in the clique for every vertex in the clique and becomes redundant for others. We will refer to the formulation (9) - (11) of the maximum clique problem as the *1-plex formulation* (OPF).

This case was excluded in the previous polyhedral study for the following reason. Unlike the maximum  $k$ -plex problem for  $1 < k < n$ ,  $x_i \leq 1$  is not always a facet for the maximum clique problem. It induces a facet *if and only if*  $V = N[i]$ . Since when the condition is satisfied, there exist  $n$  affinely independent vectors  $e_i$  and  $e_i + e_j$  for all  $j \in N(i) = V \setminus \{i\}$ , that lie on the face defined by  $x_i \leq 1$ , and when the condition is not satisfied, there exists  $j \in V \setminus N[i]$  such that  $x_i + x_j \leq 1$  is valid for the clique polytope. In other words,  $x_i \leq 1$  is a facet if and only if  $\{i\}$  is a maximal independent set. This is just a special case of a classical result presented in [Padberg, 1973].

Another interesting observation is that, to the best of our knowledge, this is the most compact *integer programming formulation* of the maximum clique problem with exactly  $n$  variables and  $n$  constraints. The classical (complement) *edge formulation* (EF) (12) - (14), has  $n$  variables and  $|\bar{E}|$  constraints which could be  $O(n^2)$  in the worst case:

$$\omega(G) = \max \sum_{i \in V} x_i \quad (12)$$

subject to:

$$x_i + x_j \leq 1 \quad \forall (i, j) \in \bar{E}, \quad (13)$$

$$x_i \in \{0, 1\} \quad \forall i \in V. \quad (14)$$

Another well known *MIS formulation* using maximal independent sets, where constraint (13) is replaced by (15), has  $n$  variables and  $O(3^n)$  constraints in the worst case [Moon and Moser, 1965].

$$\sum_{i \in I} x_i \leq 1 \quad \text{for each MIS } I \text{ in } G. \quad (15)$$

However, the compactness of the OPF comes at a price which we will make clear now. The EF is closely related to the OPF in the following sense. Note that constraint (13) can be rewritten as

$$x_i + x_j \leq 1 \quad \forall i \in V, j \in V \setminus N[i], \quad (16)$$

which amounts to repeating constraints in the EF. Now it is easy to see that constraint (10) in the OPF can be obtained by summing constraint (16) over all  $j \in V \setminus N[i]$ , for each  $i \in V$ , which results in  $n$  constraints. Let  $C(G)$  denote the feasible binary vectors of either formulation (characteristic vectors of cliques in  $G$ ),  $P(G)$  denote the LP relaxation polytope of the EF, and let  $R(G)$  denote the LP relaxation polytope of the OPF. Then we have,  $\text{conv}(C(G)) \subseteq P(G) \subseteq R(G)$  since  $R(G)$  is defined by a surrogate system [Glover, 2003] of  $P(G)$ .

The OPF has the advantage that it is compact, but it is a theoretically weaker relaxation and hence is expected to produce loose bounds in the BC tree. The EF can have a large number of constraints and hence a larger system being solved at the nodes of BC tree, but possibly producing tighter bounds. In order to compare them experimentally, we utilize BC implementations that are identical in every respect except for the formulation used. The implementation has non-neighbor fixing (the special case of non-dominated variable fixing introduced in Section 5.2 when  $k = 1$ ), MIS cutting planes (only), and parameter settings described in Section 5.1. No peeling or iterative schemes are employed. This was used to solve the maximum clique problem on DIMACS benchmarks and the results are presented in Table 11. For non-optimally solved cases indicated by †, we report  $[l, u]$  where  $l$  is the size of best clique found and  $u$  is an upper-bound on  $\omega(G)$ .

In terms of solution quality, OPF could not solve three instances to optimality that were optimally resolved using EF. In one instance that terminated non-optimally (with a huge termination gap) in both cases, OPF identified a larger clique than EF although its gap was 5% larger than that for EF. In terms of runtime for optimally solved instances, OPF took significantly smaller time on 7 instances while taking significantly larger time on 2 compared to EF. Apparently, even on this small set of 20 instances, both formulations are competitive and neither one dominates the other. One strong correlation that can be observed is that when the density is 50% or lower as in *c-fat* instances, OPF seems to allow significantly faster resolution compared to EF. It is hence worthwhile to study OPF in detail especially for sparse graphs, and also the possibility of “mixing” the two formulations to obtain one that combines the advantages of both formulations.

## 7 Conclusion and Future work

This paper studies the maximum  $k$ -plex problem, which is a graph-theoretic relaxation of the maximum clique problem introduced by Seidman and Foster (1978) in the context of social network analysis. Some important properties of a  $k$ -plex are reviewed and a detailed introduction to clique relaxation models in SNA is presented. We prove complexity results establishing the intractability of this problem for every fixed  $k$ . The problem is formulated as a binary integer program and basic polyhedral results are presented. In addition, classes of valid inequalities are developed for the problem, one of which is implemented in a branch-and-cut framework. The presented results of computational experiments indicate the effectiveness of the cuts and the framework used. The basic implementation is then modified to include preprocessing techniques that are found to be effective on very large and very sparse graphs such as scale-free graphs. This iterative approach is used to optimally resolve the problem on real-life scale-free graphs in reasonable running times. Further, the formulation developed is studied in the context of the maximum clique problem, for which a new compact formulation is obtained.

Several research problems and directions have been identified through the course of the paper that need attention. In particular, new facets of the  $k$ -plex polytope need to be discovered, the branch-and-cut algorithm may be modified and tuned to be able to solve larger instances to optimality, and the new formulation for the maximum clique problem needs to be explored further. Development of meta-heuristics capable of solving massive instances in a reasonable amount of

Table 11: Edge formulation Vs. 1-plex formulation on DIMACS graphs.

Graph	EF		OPF	
	BC Time (sec)	$\omega(G)$	BC Time (sec)	$\omega(G)$
c-fat200-1.clq	53.891	12	16.375	12
c-fat200-2.clq	50.625	24	19.922	24
c-fat200-5.clq	29.203	58	10.813	58
c-fat500-1.clq	5330.1	14	223.47	14
c-fat500-2.clq	3036.02	26	327.517	26
c-fat500-5.clq	4848.86	64	553.363	64
c-fat500-10.clq	3306.74	126	278.08	126
hamming6-2.clq	0.016	32	0.016	32
hamming6-4.clq	0.579	4	0.266	4
hamming8-2.clq	0.032	128	0.016	128
hamming8-4.clq	486.972	16	10788.1	16
hamming10-2.clq	0.187	512	0.141	512
hamming10-4.clq	10801.3 <sup>†</sup>	[32,205]	10800.4 <sup>†</sup>	[38,379]
johnson8-2-4.clq	0.015	4	0.11	4
johnson8-4-4.clq	0.033	14	2.719	14
MANN_a9.clq	0.047	16	0.125	16
MANN_a27.clq	9.157	126	10800.3 <sup>†</sup>	[125,148]
MANN_a45.clq	4579.08	345	10802 <sup>†</sup>	[338,422]
san200_0.7_2.clq	18.812	18	10800.9 <sup>†</sup>	[17,27]
keller4.clq	83.327	11	7510.53	11

time would be of a great practical value. While a maximum  $k$ -plex size can be viewed as a global measure characterizing the cohesiveness of a network, in practice one may be interested in finding all maximal (by inclusion)  $k$ -plexes in a graph. Designing algorithms for detecting all maximal  $k$ -plexes is another issue to address in the future. In addition, related  $k$ -plex and co- $k$ -plex partitioning problems seeking to partition the vertices of a graph onto a minimum number of  $k$ -plexes and co- $k$ -plexes, respectively, could be of interest. These problems provide natural generalizations of the well-known minimum clique partitioning and graph coloring problems.

## References

- J. Abello, P.M. Pardalos, and M.G.C. Resende. On maximum clique problems in very large graphs. In J. Abello and J. Vitter, editors, *External memory algorithms and visualization*, volume 50 of *DIMACS Series on Discrete Mathematics and Theoretical Computer Science*, pages 119–130. American Mathematical Society, 1999.
- J. Abello, M.G.C. Resende, and S. Sudarsky. Massive quasi-clique detection. *Lecture Notes in Computer Science*, 2286:598–612, 2002.
- R.D. Alba. A graph-theoretic definition of a sociometric clique. *Journal of Mathematical Sociology*, 3:113–126, 1973.
- R. Albert, H. Jeong, and A.-L. Barabási. Error and attack tolerance of complex networks. *Nature*, 406:378–382, 2000.
- E. Almaas and A.-L. Barabási. Power laws in biological networks. In E. Koonin, editor, *Power laws, scalefree networks and genome biology*. Landes Bioscience, 2005. To appear.
- D. Applegate, R. Bixby, V. Chvátal, and W. Cook. On the solution of traveling salesman problems. *Documenta Mathematica Journal der Deutschen Mathematiker-Vereinigung, International Congress of Mathematicians*, pages 645–656, 1998.
- J. S. Bader, A. Chaudhuri, J. M. Rothberg, and J. Chant. Gaining confidence in high-throughput protein interaction networks. *Nature Biotechnology*, 22(1):78–85, 2004.
- E. Balas, S. Ceria, and G. Cornuéjols. Mixed 0-1 programming by lift-and-project in a branch-and-cut framework. *Management Science*, 42:1229–1246, 1996a.
- E. Balas, S. Ceria, G. Cornuéjols, and N. Natraj. Gomory cuts revisited. *Operations Research Letters*, 19:1–9, 1996b.
- E. Balas, S. Ceria, G. Cornuejols, and G. Pataki. Polyhedral methods for the maximum clique problem. In D. S. Johnson and M. A. Trick, editors, *Cliques, Coloring, and Satisfiability: Second DIMACS Challenge, DIMACS Series in Discrete Mathematics and Theoretical Computer Science 26*, pages 11–28, Providence, RI, 1996c. American Mathematical Society.
- A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.
- A.-L. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: The topology of the World Wide Web. *Physica A*, 281:69–77, 2000a.

- A.-L. Barabási, R. Albert, H. Jeong, and G. Bianconi. Power-law distribution of the World Wide Web. *Science*, 287:2115a, 2000b.
- V. Batagelj and A. Mrvar. Pajek datasets. <http://vlado.fmf.uni-lj.si/pub/networks/data/>, 2006. Accessed June 2006.
- N. Berry, T. Ko, T. Moy, J. Smrcka, J. Turnley, and B. Wu. Emergent clique formation in terrorist recruitment. *The AAAI-04 Workshop on Agent Organizations: Theory and Practice, July 25, 2004, San Jose, California*, 2004. <http://www.cs.uu.nl/virginia/aotp/papers.htm>.
- V. Boginski, S. Butenko, and P. Pardalos. Network models of massive datasets. *Computer Science and Information Systems*, 1:79–93, 2004.
- V. Boginski, S. Butenko, and P. Pardalos. Mining market data: a network approach. *Computers & Operations Research*, 33:3171–3184, 2006.
- I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo. The maximum clique problem. In D.-Z. Du and P. M. Pardalos, editors, *Handbook of Combinatorial Optimization*, pages 1–74. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1999.
- BRITE. Biomolecular relations in information transmission and expression. Generalized protein interactions. [http://www.genome.jp/brite/generalized\\_interactions.html](http://www.genome.jp/brite/generalized_interactions.html), 2005. Accessed March 2005.
- S. Butenko and W. Wilhelm. Clique-detection models in computational biochemistry and genomics. *European Journal of Operational Research*, 173:1–17, 2006.
- H. Chen, W. Chung, J. J. Xu, G. Wang, Y. Qin, and M. Chau. Crime data mining: A general framework and some examples. *Computer*, 37(4):50–56, 2004a.
- Y. P. Chen, A. L. Liestman, and J. Liu. Clustering algorithms for ad hoc wireless networks. In Y. Pan and Y. Xiao, editors, *Ad Hoc and Sensor Networks*. Nova Science Publishers, 2004b. To be published.
- E. J. Chesler and M. A. Langston. Combinatorial genetic regulatory network analysis tools for high throughput transcriptomic data. Technical Report ut-cs-06-575, CS Technical Reports, University of Tennessee, 2006.
- D. J. Cook and L. B. Holder. Graph-based data mining. *IEEE Intelligent Systems*, 15(2):32–41, 2000.
- S. Corman, T. Kuhn, R. McPhee, and K. Dooley. Studying complex discursive systems: Centering resonance analysis of organizational communication. *Human Communication Research*, 28(2): 157–206, 2002.
- S. Corman, K. Dooley, and R. McPhee. LOCKS: Analysis of media coverage of the terrorist attacks. <http://locks.asu.edu/terror/>, 2006. Accessed June 2006.
- D. Corneil and Y. Perl. Clustering and domination in perfect graphs. *Discrete Applied Mathematics*, 9:27–39, 1984.
- G. Cornuéjols. *Combinatorial Optimization: Packing and Covering*. CBMS-NSF Regional Conference Series in Applied Mathematics. SIAM, Philadelphia, 2001.

- R. H. Davis. Social network analysis: An aid in conspiracy investigations. *FBI Law Enforcement Bulletin*, pages 11–19, 1981.
- A. H. Dekker. Social network analysis in military headquarters using CAVALIER. *Proceedings of Fifth International Command and Control Research and Technology Symposium*, pages 24–26, 2000.
- DIMACS. Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge. <http://dimacs.rutgers.edu/Challenges/>, 1995. Accessed September 2005.
- I. Fischer and T. Meinl. Graph based molecular data mining - an overview. In Wil Thissen, Peter Wieringa, Maja Pantic, and Marcel Ludema, editors, *IEEE SMC 2004 Conference Proceedings*, pages 4578–4582, 2004.
- L. C. Freeman. The sociological concept of “group”: An empirical test of two models. *American Journal of Sociology*, 98:152–166, 1992.
- J. Gagneur, R. Krause, T. Bouwmeester, and G. Casari. Modular decomposition of protein-protein interaction networks. *Genome Biology*, 5(8):R57.1–R57.12, 2004.
- M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W.H. Freeman and Company, New York, 1979.
- F. Glover. Tutorial on surrogate constraint approaches for optimization in graphs. *Journal of Heuristics*, 9:175–227, 2003.
- J. Grossman, P. Ion, and R. De Castro. The Erdős Number Project. <http://www.oakland.edu/enp/>, 1995. Accessed January 2005.
- F. Harary. *Graph Theory*. Narosa Publishing House, New Delhi, 1988.
- J. Hasselberg, P. M. Pardalos, and G. Vairaktarakis. Test case generators and computational results for the maximum clique problem. *Journal of Global Optimization*, 3:463–482, 1993.
- ILOG. ILOG CPLEX. <http://www.ilog.com/products/cplex/>, 2003. Accessed September 2005.
- H. Jeong, S. P. Mason, A.-L. Barabási, and Z. N. Oltvai. Centrality and lethality of protein networks. *Nature*, 411:41–42, 2001.
- D. Jiang, C. Tang, and A. Zhang. Cluster analysis for gene expression data: A survey. 16(11): 1370–1386, 2004.
- D.S. Johnson and M.A. Trick, editors. *Cliques, Coloring, and Satisfiability: Second DIMACS Implementation Challenge, October 11-13, 1993*, volume 26 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. AMS, Providence, RI, 1996.
- V. Krebs. Mapping networks of terrorist cells. *Connections*, 24:45–52, 2002.
- P. Krishna, N. Vaidya, M. Chatterjee, and D. Pradhan. A cluster-based approach for routing in dynamic networks. In *ACM SIGCOMM Computer Communication Review*, pages 49–65, 1997.
- R.D. Luce. Connectivity and generalized cliques in sociometric group structure. *Psychometrika*, 15:169–190, 1950.

- R.D. Luce and A.D. Perry. A method of matrix analysis of group structure. *Psychometrika*, 14: 95–116, 1949.
- D. McAndrew. The structural analysis of criminal networks. In D. Canter and L. Alison, editors, *The Social Psychology of Crime: Groups, Teams, and Networks, Offender Profiling Series, III*. Aldershot, Dartmouth, 1999.
- J. Mitchell. Branch-and-cut algorithms for combinatorial optimization problems. In P. M. Pardalos and M. G. C. Resende, editors, *Handbook of Applied Optimization*, pages 65–77. Oxford University Press, New York, 2002.
- R.J. Mokken. Cliques, clubs and clans. *Quality and Quantity*, 13:161–173, 1979.
- J. W. Moon and L. Moser. On cliques in graphs. *Israel Journal of Mathematics*, 3:23–28, 1965.
- M. Mukherjee and L. B. Holder. Graph-based data mining on social networks. In *Workshop on Link Analysis and Group Detection (LinkKDD2004)*, 2004.
- M. W. Padberg. On the facial structure of set packing polyhedra. *Mathematical Programming*, 5: 199–215, 1973.
- M. W. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric travelling salesman problems. *SIAM Review*, 33:60–100, 1991.
- X. Peng, M. A. Langston, A. M. Saxton, N. E. Baldwin, and J. R. Snoddy. Detecting network motifs in gene co-expression networks. In *Proc. International Conference for the Critical Assessment of Microarray Data Analysis (CAMDA 2004)*, Lecture Notes in Computer Science, 2004.
- J. C. Rain, L. Selig, H. De Reuse, V. Battaglia, C. Reverdy, S. Simon, G. Lenzen, F. Petel, J. Wojcik, V. Schachter, Y. Chemama, A. Labigne, and P. Legrain. The protein-protein interaction map of helicobacter pylori. *Nature*, 409(6817):211–215, 2004. Erratum in: *Nature* 409(6820):553 and 409(6821):743, 2001.
- M. Sageman. *Understanding Terrorist Networks*. University of Pennsylvania Press, 2004.
- L. A. Sanchis and A. Jagota. Some experimental and theoretical results on test case generators for the maximum clique problem. *INFORMS Journal on Computing*, 8(2):103–117, 1996.
- J. Scott. *Social Network Analysis: A Handbook*. Sage Publications, London, 2 edition, 2000.
- S. B. Seidman. Network structure and minimum degree. *Social Networks*, 5:269–287, 1983.
- S. B. Seidman and B. L. Foster. A graph theoretic generalization of the clique concept. *Journal of Mathematical Sociology*, 6:139–154, 1978.
- H. D. Sherali and C. J. Smith. A polyhedral study of the generalized vertex packing problem. *Mathematical Programming*, 107:367–390, 2006.
- M. K. Sparrow. The application of network analysis to criminal intelligence: An assessment of the prospects. *Social Networks*, 13:251–274, 1991.
- V. Spirin and L. A. Mirny. Protein complexes and functional modules in molecular networks. *Proceedings of the National Academy of Sciences*, 100(21):12123–12128, 2003.

- G.K. Tayi S.S. Ravi, D.J. Rosenkrantz. Heuristics and special case algorithms for dispersion problems. *Operations Research*, 42:299–310, 1994.
- I. Stojmenovic, editor. *Handbook of Wireless Networks and Mobile Computing*. Wiley InterScience, 2002.
- L. Terveen, W. Hill, and B. Amento. Constructing, organizing, and visualizing collections of topically related web resources. *ACM Transactions on Computer-Human Interaction*, 6:67–94, 1999.
- L. E. Trotter. A class of facet producing graphs for vertex packing polyhedra. *Discrete Mathematics*, 12:373–388, 1975.
- T. Washio and H. Motoda. State of the art of graph-based data mining. *SIGKDD Explor. Newsl.*, 5(1):59–68, 2003.
- S. Wasserman and K. Faust. *Social Network Analysis*. Cambridge University Press, 1994.