Mathematical Programming Computation (2020) 12:109–130
https://doi.org/10.1007/s12532-019-00165-3

FULL LENGTH PAPER

Exploiting sparsity for the min *k*-partition problem

Guanglei Wang[1] · **Hassan Hijazi**[1,2]

Received: 19 October 2017 / Accepted: 17 June 2019 / Published online: 2 July 2019

Abstract
The minimum k-partition problem is a challenging combinatorial problem with a diverse set of applications ranging from telecommunications to sports scheduling. It generalizes the max-cut problem and has been extensively studied since the late sixties. Strong integer formulations proposed in the literature suffer from a large number of constraints and variables. In this work, we introduce two more compact integer linear and semidefinite reformulations that exploit the sparsity of the underlying graph and develop theoretical results leveraging the power of chordal decomposition. Numerical experiments show that the new formulations improve upon state-of-the-art.

Keywords Integer programming · Graph partitioning · Semidefinite programming · Chordal graph

Mathematics Subject Classification 90C99

1 Introduction

Given an undirected, weighted graph $G = (V, E)$, the minimum k-partition problem (MkP) consists of partitioning V into at most k disjoint subsets, minimizing the total weight of the edges joining vertices in the same subsets. This problem has been shown to be strongly $\mathcal{NP}$-hard [33]. MkP was firstly defined in [10] in the context of scheduling where activities must be assigned to a limited number of facilities. The authors formulate the problem as a quadratic program and suggest a method returning local minima.

✉ Guanglei Wang
guanglei.wang13@gmail.com

Hassan Hijazi
hlh@lanl.gov

1 The Australian National University, Acton, Canberra 2601, Australia

2 Los Alamos National Laboratory, Los Alamos, NM, USA

The complement of MkP is a max-k-cut problem where one is seeking to maximize the sum of weights corresponding to intra-partition edges. Related investigations can be found in [14,15,19]. The well-known Integer Linear Programming (ILP) formulation of max-k-cut can be obtained by complementing the variables of the ILP formulation of MkP. Thus solving max-k-cut is equivalent to solving MkP. When $k = 2$, the minimum 2-partition is equivalent to the max-cut problem, which is $\mathcal{NP}$-hard and has been extensively studied in the literature (see e.g., [3,16,24]). Hence, in the general case, both MkP and max-k-cut are $\mathcal{NP}$-hard (see also [12,36]). However, they have different approximability results. It has been shown in [19] that max-k-cut can be solved $(1 - k^{-1})$-approximately in polynomial time. But MkP is not $\mathcal{O}(|E|)$-approximable in polynomial time unless $\mathcal{P} = \mathcal{NP}$ [17].

In [12], Chopra and Rao introduce two ILPs for MkP. The first one uses node and edge variables associated with the graph G. The second one uses edge variables but requires completing the original graph G with missing edges. To strengthen the formulations, several families of strong valid linear inequalities have been proposed (see e.g., [11,12,14,18] for references).

In this paper, we propose two new MkP formulations. One is an ILP formulation based on Chopra and Rao's [12] edge-based model. The other is an Integer Semidefinite Programming (ISDP) formulation based on the Eisenblätter's [17] model. Both models exploit the sparsity of the graph and thus have less integer variables than their counterparts. We also show that if G is chordal, our proposed ILP formulation has $|E|$ variables (according to [12], no existing formulations live in that space). To validate the proposed formulations, several theoretical results are established. The computational experiments show that the proposed reformulations are able to improve computational efficiency by several orders of magnitude.

Outline The rest of the paper is organized as follows. In Sect. 2, the literature on the existing MkP formulations is reviewed. Applications of chordal graphs to mathematical reformulations are also discussed. In Sect. 3, we provide our reformulations based on the maximal clique set. Some theoretical results are established. In Sect. 4, we describe some computational experiments and analyze the results. Finally, some concluding remarks are made in Sect. 5.

Notation For a finite set S, $|S|$ represents its cardinality. $\mathbb{S}^n$ represents the set of real symmetric matrices in $\mathbb{R}^{n\times n}$. $\mathbb{S}^n_+$ denotes the cone of real positive semidefinite matrices in $\mathbb{S}^n$, i.e., $\mathbb{S}^n_+ = \{S \in \mathbb{S}^n : S \succeq 0\}$. For a matrix M, M_C represents the principle sub-matrix composed by set C of columns and set C of rows of M. We also use (C, D) to denote the matrix where C and D are concatenated by rows assuming they have the same number of columns. The inner product between two matrices $A, B \in \mathbb{R}^{m\times n}$ is denoted by $\langle A, B\rangle = \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}B_{ij}$.

2 Literature review

In this section, we review some existing formulations of MkP and existing literature on chordal graph based techniques.

2.1 Formulations of M*k*P

Following [11,12], M*k*P can be expressed in Model 1. Binary variable x_{ij} is 1 if

Model 1 The edge formulation of M*k*P

$$\textbf{variables:}\quad x \in \{0,1\}^{\frac{n(n-1)}{2}} \tag{1a}$$

$$\textbf{minimize:}\quad \sum_{\{i,j\}\in E} w_{ij}x_{ij} \tag{1b}$$

$$\textbf{s.t.:}\quad x_{ij} + x_{ih} - x_{jh} \le 1, \tag{1c}$$

$$x_{ij} + x_{jh} - x_{ih} \le 1,$$

$$x_{ih} + x_{jh} - x_{ij} \le 1,\ \forall \{i,j,h\} \subseteq V$$

$$\sum_{i,j\in Q: i<j} x_{ij} \ge 1,\ \forall Q \subseteq V, \text{where } |Q| = k+1 \tag{1d}$$

and only if i and j are in the same partition. The *triangle* constraints (1c) enforce consistency with respect to partition membership. Namely, for a given subset $\{i, j, h\}$, if $x_{ij} = 1$ and $x_{jh} = 1$, indicating that i, h and j are in the same partition, this implies $x_{ih} = 1$. For every subset of $k+1$ vertices, the *clique* constraints (1d) force at least two vertices to be in the same partition. Together with constraints (1c), this implies that there are at most k subsets. In total, there are $3\binom{|V|}{3}$ triangle inequalities and $\binom{|V|}{k+1}$ clique inequalities. Following [18], we call this formulation the *edge* formulation. When $3 \le k \le |V|-1$, we denote by P^k the set of feasible solutions of Model 1 as:

$$P^k = \left\{ x \in \{0,1\}^{V^2} : x \text{ satisfies (1c) and (1d)} \right\}, \tag{2}$$

where $V^2 := \{\{i,j\} \in V \times V : i < j\}$ represents the edges of the complete graph induced by vertices V.

It is known that the convex hull of P^k is a full-dimensional polytope [11, 12,16]. Chopra and Rao give several valid and facet defining inequalities for P^k including *general clique, wheel* and *bicycle* constraints [12]. It has been noticed that the exact separation of the clique inequalities is $\mathcal{NP}$-hard in general, and the complete enumeration is intractable even for small values of k [23].

Chopra and Rao [12] also propose an alternative ILP, presented in Model 2. It has $kn + m$ variables, where n and m are respective numbers of nodes and edges. Binary variable x_{ic} is 1 if node i lies in the cth subset. Binary variable y_{ij} is 1 if i and j are in the same subset. Constraints (3c) express that each node must be assigned to exactly one subset. Constraints (3d) indicate that if $\{i, j\}$ is an edge and i and j are in the same subset then the edge $\{i, j\}$ is not cut by the partition (i.e., $y_{ij} = 1$). Conversely, if nodes i, j are in two disjoint subsets, (3e) and (3f) enforce that the edge $\{i, j\}$ is cut by the partition (i.e., $y_{ij} = 0$). We refer to Model 2 as the node-edge formulation. Some valid inequalities have been proposed to strengthen Model 2 (see, e.g., [12,18]).

Model 2 The node-edge formulation of MkP

$$\textbf{variables: } x \in \{0,1\}^{kn}, y \in \{0,1\}^m \tag{3a}$$

$$\textbf{minimize: } \sum_{\{i,j\}\in E} w_{ij} y_{ij} \tag{3b}$$

$$\textbf{s.t.: } \sum_{c=1}^{k} x_{ic} = 1 \; i \in V \tag{3c}$$

$$x_{ic} + x_{jc} - y_{ij} \leq 1 \;\; \{i,j\} \in E, c = 1,\ldots,k \tag{3d}$$

$$x_{ic} - x_{jc} + y_{ij} \leq 1 \;\; \{i,j\} \in E, c = 1,\ldots,k \tag{3e}$$

$$-x_{ic} + x_{jc} + y_{ij} \leq 1 \;\; \{i,j\} \in E, c = 1,\ldots,k. \tag{3f}$$

Eisenblätter [17] provides an exact ISDP reformulation of Model 1. We introduce variables $X_{ij} \in \{\frac{-1}{k-1}, 1\}$, where $X_{ij} = 1$ if and only if node i and j are in the same partition. This formulation is presented in Model 3.

Model 3 The integer SDP formulation of MkP

$$\textbf{variables: } X_{ij} \in \frac{-1}{k-1}, 1 \;\; (\forall i,j \in V : i < j). \tag{4a}$$

$$\min \sum_{\{i,j\}\in E} w_{ij} \frac{(k-1)X_{ij}+1}{k} \tag{4b}$$

$$\text{s.t. } X \in \mathbb{S}^n_+ \tag{4c}$$

$$X_{ii} = 1, \;\; \forall i \in V. \tag{4d}$$

Replacing the constraint $X_{ij} \in \{\frac{-1}{k-1}, 1\}$ with $\frac{-1}{k-1} \leq X_{ij} \leq 1$ yields a Semidefinite Programming (SDP) relaxation. Notice that $X_{ij} \leq 1$ can be dropped since it is implicitly enforced by the SDP constraint and $X_{ii} = 1$. This SDP relaxation was used in combination with randomized rounding to obtain a polynomial time approximation algorithm for max k-cut by Frieze and Jerrum [19]. Eisenblätter [17] shows that one can recast Model 3 using binary variables. Indeed, we introduce binary variables Y_{ij} such that

$$X_{ij} = \frac{-1}{k-1} + \frac{k}{k-1} Y_{ij}. \tag{5}$$

Thus, $X_{ij} = \frac{-1}{k-1}$ if $Y_{ij} = 0$ and $X_{ij} = 1$ if $Y_{ij} = 1$. This leads to

$$\min \sum_{\{i,j\}\in E} w_{ij} Y_{ij}$$

$$\text{s.t. } \frac{-1}{k-1} J + \frac{k}{k-1} Y \in \mathbb{S}^n_+, \tag{6a}$$

$$Y_{ii} = 1, \quad \forall i \in V, \tag{6b}$$
$$Y_{ij} \in \{0, 1\} \quad \forall i, j \in V : i < j. \tag{6c}$$

where J is the all-one matrix. Note that the mapping function (5) is bijective or one-to-one. This bijective relationship will be used in Sect. 3.2.

Several SDP based branch-and-bound frameworks for MkP (or max-k-cut) have appeared in the literature [23,38]. Typically, these approaches identify several families of valid inequalities to tighten the SDP relaxation during the solution procedure. However, SDP solvers generally tend to be significantly slower than LP solvers [4,41] for the respective SDPs and LPs with the same number of variables.

2.2 Graph terminology

We shall assume familiarity with basic definitions from graph theory. We will also use results from the algorithmic graph theory for chordal graphs and we refer readers to [8,25] for relevant references. Let us now introduce some of the graph notation and terminology that will be used in this article.

Let $G = (V, E)$ be an *undirected graph* formed by vertex set V and edge set V. The number of vertices is denoted by $n = |V|$ and the number of edges by $m = |E|$. A graph is called *complete* if every pair of vertices are adjacent. A *clique* of a graph is an induced subgraph which is complete, and a clique is *maximal* if its vertices do not constitute a proper subset of another clique. A sequence of nodes $\{v_0, v_1, v_2, \ldots, v_k, v_0\} \subseteq V$ is a *cycle* of length $k + 1$ if $\{v_{i-1}, v_i\} \in E$ for all $i \in \{1, \ldots, k\}$ and $\{v_k, v_0\} \in E$.

A graph is said to be *chordal* if every cycle of length greater than or equal to 4 has a *chord* (an edge joining two nonconsecutive vertices of the cycle). Given a graph $G = (V, E)$, we say that a graph $G_F = (V, F)$ is a *chordal extension* of G if G_F is chordal and $E \subseteq F$. Throughout this article, we also assume that the graph G is connected.

2.3 Chordal graphs and reformulations

Investigations on chordal graph were initiated by [5,21] for the characterisation of perfect graphs. Chordal graphs have a wide range of applications in combinatorial optimization [2,22], matrix completion [26,32] and more recently in solving sparse SDPs [20,29,35,40]. Comprehensive reviews on the theory and applications are given by Blair and Peyton [8] and Vandenberghe and Andersen [39].

To the best of our knowledge, research on mathematical reformulations using chordal graphs started with Fulkerson and Gross [21], where the authors establish connections between interval graphs (a special case of chordal graph) and matrix total unimodularity. Later, the authors in [20,35] introduce chordal graph techniques to accelerate the solution procedure of Interior Point Methods (IPMs) for SDPs by reformulating the underlying large matrix with an equivalent set of smaller matrices. The reformulation is based on a clique decomposition of a chordal extension of the *aggregated sparsity pattern graph*. This approach accelerates the computation of the solution (as a search direction in IPMs) of the Schur complement equation, thereby

speeding up the solution procedure of IPMs. The reformulation method is often called conversion method or clique decomposition method. Chordal graphs are also used to exploit the *correlative sparsity pattern* in the context of polynomial optimization for deriving hierarchies of SDP relaxations (see [31,40]).

More recently, Bienstock and Ozbay [7] establish a connection between the well-known Sherali-Adams reformulation operator for 0-1 integer programs and chordal extensions. Later, the authors in [6] show that polynomial-size LP reformulations can be constructed for certain classes of mixed-integer polynomial optimization problems by exploiting structured sparsity.

To distinguish and relate the results developed in this paper from the mentioned techniques, we remark the following: (1) our main results are ILP (Model 4) and ISDP (Model 5) reformulations, which is different from LP or SDP reformulations discussed above; (2) in addition to the known chordal graph properties in the literature, our reformulations rely on the proposed theoretical results (Theorems 1 and 2); (3) to the best of our knowledge, the theoretical results are novel in the sense that they are not mentioned by any results in the literature. Nevertheless it has been known for decades that many intractable combinatorial problems may become efficiently solvable when the treewidth of the input graph is bounded by a constant [2]. And our main results are consistent with this. It is also interesting that our reformulation is also related to the conversion method proposed in [20,35]. As will be shown later, both reformulations rely on a clique decomposition of the underlying graph, though with different purposes.

3 Main results

In this section, we propose two reformulations of MkP by exploiting the structured sparsity of the underlying graph. Let $\mathcal{K}$ be the set of all maximal cliques in G_F, representing the chordal extension of G, i.e., $\mathcal{K} = \{C_1, \ldots, C_l\}$, $C_i \subseteq V \ \forall i \in \{1, \ldots, l\}$, such that $F = \bigcup_{r=1}^{l} C_r \times C_r \supseteq E$. We will next show that the following property holds:

Property 1 (Completion Property) *For any vector $x_F \in \mathbb{R}_+^{|F|}$, if x_F satisfies the following triangle constraints for each $\{i, j, h\} \subseteq C_r, r = 1, \ldots, l$,*

$$\begin{aligned} x_{ij} + x_{ih} - x_{jh} &\leq 1, \\ x_{ij} + x_{jh} - x_{ij} &\leq 1, \\ x_{ih} + x_{jh} - x_{ij} &\leq 1, \end{aligned} \tag{7}$$

and the following clique constraints and binary constraints

$$\sum_{i,j \in Q} x_{ij} \geq 1, \ \forall Q \subseteq C_r, \text{where } |Q| = k+1, r = 1, \ldots, l, \tag{8a}$$

$$x_F \in \{0, 1\}^{|F|}, \tag{8b}$$

then there exists $x_T \in \{0,1\}^{|V^2 \setminus F|}$ such that $x = (x_F, x_T)$ satisfies (1c) and (1d).

Under the conditions above, we observe that the value of the objective (1b) can be determined by values of entries x_F and independently of values in x_T. In addition, the resulting formulation has $\sum_{r=1:|C_r|\geq 3}^{l} 3\binom{|C_r|}{3} + \sum_{r=1:|C_r|\geq k+1}^{l} \binom{|C_r|}{k+1}$ constraints and $|F|$ binary variables and it becomes attractive if this number is less than $3\binom{|V|}{3} + \binom{|V|}{k+1}$.

3.1 An ILP reformulation

Let P_F^k represent the set of feasible solutions satisfying (7), (8a), (8b), i.e.,

$$P_F^k = \left\{ x \in \mathbb{R}_+^F : (7), (8a), (8b) \right\}. \tag{9}$$

We show that for any $x \in P^k$, the components indexed by set F represent a member of P_F^k. We denote by $\text{Proj}_F\, S$ the projection of S into the space defined by the components in F. Specifically, for a given set $S \subset \mathbb{R}^I$, where $F \subset I \subseteq V^2$, we define

$$\text{Proj}_F\, S = \left\{ x_F \in \mathbb{R}^F : \exists x_T \in \mathbb{R}^{I \setminus F}, (x_F, x_T) \in S \right\}. \tag{10}$$

Lemma 1 $\forall k,\ 2 \leq k \leq n,\ \text{Proj}_F\, P^k \subseteq P_F^k$.

Proof Let $\bar{x}$ be an arbitrary point in P^k. It is easy to verify that entries of $\bar{x}_F$ satisfy constraints in P_F^k and therefore $\text{Proj}_F\, P^k \subseteq P_F^k$. □

Lemma 2 (Lemma 3, [26]) *Given that G_F is chordal, for any pair of vertices u and v with $u \neq v$, $\{u, v\} \notin E$, the graph $G_F + \{u, v\}$ has a unique maximal clique which contains both u and v.*

Lemma 3 (Lemma 4, [26]) *Given that G_F is chordal, there exists a sequence of chordal graphs*

$$G_i = (V, F_i), \quad i = 0, \ldots, s, \tag{11}$$

such that $G_0 = G_F$, G_s is the complete graph, and G_i and F_i are obtained by adding an edge to G_{i-1} for all $i = 1, \ldots, s$.

Theorem 1 $\forall k,\ 2 \leq k \leq n,\ P_F^k = \text{Proj}_F\, P^k$.

Proof By Lemma 1, $\text{Proj}_F\, P^k \subseteq P_F^k$. We now show that the reverse also holds. Given G_F, let $\{G_0, G_1, \ldots G_s\}$ be a sequence of chordal graphs satisfying Lemma 3. Denote by $\{i_1, j_1\}$ the edge of G_1 that is not a member of G_0 (w.l.o.g., we assume $i_1 < j_1$). Let $\bar{x}_F$ be a point in P_F^k. If we can show that there exists a vector $\bar{x}_1 = (\bar{x}_F, x_{i_1 j_1})$ satisfying the constraints in $P_{G_1}^k$, then by induction, we can show the existence of a point $x_s \in P^k$.

By Lemma 2, there is a unique maximal clique C of G_1 containing nodes i_1, j_1. Without loss of generality, we may reorder indices of the nodes in C and let $C = \{l_0, \ldots, l_p, i_1, j_1\}$ with $l_0 < \cdots < l_p < i_1 < j_1$. Let x_C be the vector corresponding to the maximal clique C. Since any clique containing $\{i_1, j_1\}$ is a subset of C, $x_1 \in P^k_{G_1}$ iff $x_C \in P^k_C$, where P^k_C is formed by replacing V with C in (2).

Hence we only need to show that $x_C \in P^k_C$ for some value of $x_{i_1 j_1}$. Given x_F, we construct $x_{i_1 j_1}$ as follows: if $C = \{i_1, j_1\}$, let $x_{i_1 j_1} = 1$ and the solution is feasible. Otherwise, three cases can occur:

1. $(x_{hi_1}, x_{hj_1}) = (0, 0), \forall h \in \{l_0, \ldots, l_p\}$.
2. $\exists h \in \{l_0, \ldots, l_p\}$ such that $x_{hi_1} + x_{hj_1} = 1$.
3. $\exists h \in \{l_0, \ldots, l_p\}$ such that $x_{hi_1} + x_{hj_1} = 2$.

We now construct the solution as follows:

1. If case 1 or case 3 occurs, let $x_{i_1 j_1} = 1$.
2. otherwise (i.e., case 2 occurs), let $x_{i_1 j_1} = 0$.

In order to show that the constructed solution is valid, we next show that case 1 and case 3 are exclusive with respect to case 2. It is obvious that case 1 is exclusive with case 2. We now show that case 2 and case 3 are exclusive:

- The result is straightforward when $p = 0$.
- When $p \geq 1$, we proceed by contradiction: suppose there exists $h, h' \in V : h \neq h'$ such that $x_{hi_1} + x_{hj_1} = 1$ and $x_{h'i_1} + x_{h'j_1} = 2$. Let us consider $(x_{hi_1}, x_{hj_1}) = (0, 1)$. The other case $(x_{hi_1}, x_{hj_1}) = (1, 0)$ will follow symmetrically. Note that $\{h, h', i_1\}$ and $\{h, h', j_1\}$ are cliques of length 3 in G_F. Thus by (7), we have, on one hand, $x_{hh'} + x_{h'i_1} - x_{hi_1} \leq 1$ leading to $x_{hh'} \leq 0$. On the other hand, we have $x_{h'j_1} + x_{hj_1} - x_{hh'} \leq 1$ leading to $x_{hh'} \geq 1$, contradiction.

We now verify that the extended solution $x_1 = (x_F, x_{i_1 j_1})$ satisfies constraints in P^k_C. Since $\{l_0, \ldots, l_p, i_1\}$ and $\{l_0, \ldots, l_p, j_1\}$ are cliques in G_F, the associated constraints have been imposed by P^k_F. Thus, we just need to verify that x_1 satisfies the following inequalities

$$x_{hi_1} + x_{hj_1} - x_{i_1 j_1} \leq 1, \quad \forall h \in \{l_0, \ldots, l_p\} \tag{12a}$$

$$x_{hi_1} + x_{i_1 j_1} - x_{hj_1} \leq 1, \quad \forall h \in \{l_0, \ldots, l_p\} \tag{12b}$$

$$x_{hj_1} + x_{i_1 j_1} - x_{hi_1} \leq 1, \quad \forall h \in \{l_0, \ldots, l_p\} \tag{12c}$$

$$\sum_{\{h,f\} \in C_q} x_{hf} + x_{i_1 j_1} \geq 1, \quad \forall Q \subseteq C, |Q| = k + 1, \; i_1, j_1 \in Q \tag{12d}$$

where $C_q = \{\{h, f\} \in Q^2 : \; h < f, \; \{h, f\} \neq \{i_1, j_1\}\}$.

- First, let us show that the constructed solution satisfies the above triangle inequalities (12a)–(12c). For case 1, for each $h, i_1, j_1 \in C$, the unique solution $(x_{hx_1}, x_{hj_1}, x_{i_1 j_1}) = (0, 0, 0)$ is feasible. For case 2, both solution $(x_{hi_1}, x_{hj_1}, x_{i_1 j_1}) = (1, 0, 0)$ and $(x_{hi_1}, x_{hj_1}, x_{i_1 j_1}) = (0, 1, 0)$ are feasible. For case 3, the unique solution $(x_{hi_1}, x_{hj_1}, x_{i_1 j_1}) = (1, 1, 1)$ is feasible.

- Second, we need to verify that the constructed solution is feasible for the clique inequalities (12d) when $|C| \geq k+1$. It is easy to see that this is true for case 1 and case 3. For case 2, we consider $|C| = k+1$ and $C \geq |k+2|$. Let us denote by h^* the index such that $x_{h^* i_1} + x_{h^* j_1} = 1$. Recall that $x_{i_1 j_1} = 0$.

  1. When $|C| = k+1$, it holds that $Q = C$, $\{h^*, i_1\} \in C_q$ and $\{h^*, j_1\} \in C_q$. Then we have

$$\sum_{\{h,f\} \in C_q} x_{hf} + x_{i_1 j_1} \geq 1.$$

  2. When $|C| \geq k+2$, there are multiple subsets Q. If Q contains h^*, the desired result follows. If $h^* \notin Q$, we have

$$\sum_{\{h,f\} \in C_q} x_{hf} + x_{i_1 j_1} = \sum_{\{h,f\} \in C_q : f < i_1} x_{hf} + \sum_{\{h,i_1\} \in C_q} x_{hi_1} + \sum_{\{h,j_1\} \in C_q} x_{hj_1}$$

Note that if the following disjunction is true:

$$\sum_{\{h,f\} \in C_q : f < i_1} x_{hf} \geq 1 \vee \sum_{\{h,i_1\} \in C_q} x_{hi_1} \geq 1 \vee \sum_{\{h,j_1\} \in C_q} x_{hj_1} \geq 1$$

then we have $\sum_{\{h,f\} \in C_q} x_{hf} + x_{i_1 j_1} \geq 1$, the result follows. We now show that it is not possible to have a solution $x_F \in P_F^k$ satisfying

$$\sum_{\{h,f\} \in C_q : f < i_1} x_{hf} = 0 \tag{13a}$$

$$\sum_{\{h,i_1\} \in C_q} x_{hi_1} = 0 \tag{13b}$$

$$\sum_{\{h,j_1\} \in C_q} x_{hj_1} = 0. \tag{13c}$$

We proceed by contradiction. Assume that (13a)–(13c) hold. Then we have:

$$\begin{aligned} 0 &= \sum_{\{h,f\} \in C_q : f < i_1} x_{hf} + \sum_{\{h,i_1\} \in C_q} x_{hi_1} + \sum_{\{h,j_1\} \in C_q} x_{hj_1} \\ &= \sum_{\{h,i_1\} \in C_q} x_{hi_1} + \sum_{\{h,j_1\} \in C_q} x_{hj_1} \end{aligned}$$

$$\begin{aligned}
&\geq 2 - 2 \sum_{\{h,f\} \in C_q : f < i_1} x_{hf} - 2 \sum_{\{h,h^*\} \in C_q} x_{hh^*} - x_{h^* i_1} - x_{h^* j_1} \\
&\geq 1 - 2 \sum_{\{h,h^*\} \in C_q} x_{hh^*} \\
&= 1,
\end{aligned}$$

which is a contradiction. The first"$\geq$" comes from the fact that $Q \setminus \{i_1\} \cup \{h^*\}$ and $Q \setminus \{j_1\} \cup \{h^*\}$ are cliques of size $k+1$. Thus by clique inequalities (8a) enforced in P_F^k, we have

$$\begin{aligned}
\sum_{\{h,f\} \in C_q : f < i_1} x_{hf} + \sum_{\{h,j_1\} \in C_q} x_{hj_1} + \sum_{\{h,h^*\} \in C_q} x_{hh^*} + x_{h^* j_1} \geq 1, \\
\sum_{\{h,f\} \in C_q : f < i_1} x_{hf} + \sum_{\{h,i_1\} \in C_q} x_{hi_1} + \sum_{\{h,h^*\} \in C_q} x_{hh^*} + x_{h^* i_1} \geq 1.
\end{aligned}$$

The second "$\geq$" comes from (13a) and $x_{h^* i_1} + x_{h^* j_1} = 1$. The last equality comes from the fact that for each $h \in Q : h < i_1, h \neq h^*$, $\{h, h^*, i_1\}$ and $\{h, h^*, j_1\}$ are cliques in G_F. Thus, by (13b)–(13c) and the triangle inequality (7) in P_F^k, we have

$$x_{hh^*} \leq 1 + x_{hi_1} - x_{h^* i_1} \quad \text{and} \quad x_{hh^*} \leq 1 + x_{hj_1} - x_{h^* j_1}, \quad \forall h \in Q : h < i_1.$$

which leads to $x_{hh^*} = 0, \forall h \in Q : h < i_1, h \neq h^*$.

□

This immediately yields the following desired result.

Corollary 1 *Model 1 is equivalent to Model 4 in the sense that for any feasible solution for Model 1, there exists a feasible solution for 4 such that their objective values are equal; conversely, for any feasible solution x_F for Model 4, there also exists a vector extending x_F such that it is feasible for Model 1.*

Model 4 The clique-based reformulation of MkP

variables: $x \in \{0, 1\}^{|F|}$
minimize: (1b)
s.t.: (7), (8a).

Proof Observe that the objective (1b) is determined by variables $x_{ij}, \forall \{i, j\} \in E$. Given that $E \subseteq F$ and based on Theorem 1, the result follows. □

Observe that Model 4 has less binary variables than Model 1 when G is not complete. In the most favorable case, where the given graph $G = (V, E)$ is chordal, Model 4 involves no additional binary variables and thus becomes more attractive than Model 2. In Section 2 of [12], the authors state that if G is not complete, they are not aware of any formulation that uses only edge variables. Here, we see from Corollary 1 that if G is chordal, Model 4 uses exactly $|E|$ variables.

3.2 An ISDP reformulation

We show in this section that there exists a clique-based reformulation of Model 3, which has less number of variables and constraints. It is presented in Model 5.

Model 5 The clique-based integer SDP formulation of MkP

$$\textbf{variables: } X_{ij} \in \left\{\frac{-1}{k-1}, 1\right\} \quad (\forall \{i,j\} \in F) \tag{14a}$$

$$\min \ (4b) \tag{14b}$$

$$\text{s.t. } X_{ii} = 1, \ \forall i \in V.$$

$$X_{C_r} \in \mathbb{S}_+^{|C_r|}, \ \forall C_r \in \mathcal{K}. \tag{14c}$$

We will next show that Model 5 is a valid formulation of MkP, which entails showing that

$$\begin{aligned}&\text{Proj}_F \left\{ (X_{ij})_{i<j} \in \left\{\frac{-1}{k-1}, 1\right\}^{\frac{n(n-1)}{2}} : (4c), (4d) \right\} \\ &= \left\{ (X_{ij})_{\{i,j\}\in F} \in \left\{\frac{-1}{k-1}, 1\right\}^{F} : (14b), (14c) \right\},\end{aligned}$$

where the projection operator $\text{Proj}_F\, S$ has been defined in (10). With the bijective mapping defined in (5), we see that the left-hand side and the right-hand side of the above equation correspond to the respective $\mathcal{F}$ and $\mathcal{F}'$ presented below,

$$\mathcal{F} = \left\{ (Y_{ij})_{i<j} \in \{0,1\}^{\frac{n(n-1)}{2}} : (6a), (6b) \right\},$$

$$\mathcal{F}' = \left\{ (Y_{ij})_{\{i,j\}\in F} \in \{0,1\}^{F} : \frac{-1}{k-1} J_{C_r} + \frac{k}{k-1} Y_{C_r} \in \mathbb{S}_+^{|C_r|}, \ \forall C_r \in \mathcal{K}, \ Y_{ii} = 1, \ \forall i \in V \right\}.$$

This implies that we simply need to prove that $\text{Proj}_F\, \mathcal{F} = \mathcal{F}'$. To this end, we exploit a technical lemma that was proposed in [19]. We remark that a similar lemma was used by Eisenblätter's [17] to prove the equivalence of Model 3 and 1.

Lemma 4 (Lemma 1, [19]) *For all integers n and k satisfying $2 \le k \le n+1$, there exist k unit vectors $\{u_1, \ldots, u_k\} \in \mathbb{R}^n$ such that $\langle u_l, u_h \rangle = \frac{-1}{k-1}$, for $l \ne h$.*

Theorem 2 *Given integer* $2 \leq k \leq n$, $\text{Proj}_F \mathcal{F} = \mathcal{F}'$.

Proof As remarked in Sect. 2.1, we have $\mathcal{F} = P^k$, where P^k is defined in (2). Thus it holds that $\text{Proj}_F \mathcal{F} = \text{Proj}_F P^k$. By Theorem 1, $\text{Proj}_F P^k = P_F^k$. Thus, it is sufficient to show that $P_F^k = \mathcal{F}'$. Although the proof is close to the reasoning of proving $P^k = \mathcal{F}$ given by Eisenblätter's [17], we state it below for the sake of completeness.

- $\mathcal{F}' \subseteq P_F^k$: Let $Y \in \mathbb{S}^n$ such that $(Y_{ij})_{\{i,j\}\in F} \in \mathcal{F}'$. We show that Y satisfies the triangle inequalities (7) and clique inequalities (8a).

  1. Triangle inequalities (7): Suppose there exists $i, j, h \in \mathcal{C}_r$ such that (Y_{ij}, Y_{ih}, Y_{jh}) violates (7). Then, (Y_{ij}, Y_{ih}, Y_{jh}) can only be assigned the values $(1, 1, 0)$, $(1, 0, 1)$ or $(0, 1, 1)$. Since $(Y_{ij})_{\{i,j\}\in F} \in \mathcal{F}'$, the principle sub-matrix of $X_{C_r} = \frac{-1}{k-1} J_{C_r} + \frac{k}{k-1} Y_{C_r}$ corresponding to indices (i, j, h) in the following form

$$\begin{pmatrix} 1 & X_{ij} & X_{ih} \\ X_{ij} & 1 & X_{jh} \\ X_{ih} & X_{jh} & 1 \end{pmatrix} \tag{15}$$

  is positive semidefinite. This implies its determinant $1 + 2X_{ij}X_{ih}X_{jh} - {X_{ij}}^2 - {X_{ih}}^2 - {X_{ij}}^2 \geq 0$. One can verify that if (Y_{ij}, Y_{ih}, Y_{jh}) is assigned value $(1, 1, 0)$, $(1, 0, 1)$ or $(0, 1, 1)$, the determinant becomes $-(\frac{k}{k-1})^2 < 0$, contradiction.
  2. Clique inequalities (8a): Suppose there exists $Q \subseteq \mathcal{C}_r : |Q| = k + 1$ such that Y_Q violates the clique constraint (8a). This implies that $X_{ij} = \frac{-1}{k-1}$, $\forall i, j \in Q : i < j$, leading to $\sum_{i,j\in Q: i<j} X_{ij} = \frac{-k(k+1)}{2(k-1)}$. Note that $X_Q = \frac{-1}{k-1} J_Q + \frac{k}{k-1} Y_Q \in \mathbb{S}_+^{|k+1|}$, suggesting that $\mathbf{1}^T X_Q \mathbf{1} = \sum_{i,j\in Q: i<j} 2X_{ij} + k + 1 \geq 0$, where $\mathbf{1}$ is the all ones vector. This leads to $\sum_{i,j\in Q: i<j} X_{ij} \geq \frac{-(k+1)}{2} > \frac{-(k+1)}{2} \times \frac{k}{k-1}$, contradiction.

- $P_F^k \subseteq \mathcal{F}'$: for any $(Y_{ij})_{\{i,j\}\in F} \in P_F^k$, we show that for each $C_r \in \mathcal{K}$ ($|C_r| \geq 3$), the matrix $X_{C_r} = \frac{-1}{k-1} J_{C_r} + \frac{k}{k-1} Y_{C_r}$ formed by Y_{C_r} is positive semi-definite. Given k, let $\{u_1, \ldots, u_k\}$ be a set of unit vectors satisfying Lemma 4. Now we construct a real matrix $B \in \mathbb{R}^{n\times|C_r|}$ in the following way. For each $i, j \in C_r$,

  1. if $y_{ij} = 0$, then assign column i, j of matrix B with different unit vectors from set $\{u_1, \ldots, u_k\}$;
  2. otherwise, assign column i, j with the same unit vector from set $\{u_1, \ldots, u_k\}$.

  One can now verify that matrix $X_{C_r} = B^T B$, showing that X_{C_r} is positive semidefinite.

□

In summary, the relationship between the aforementioned four constraint sets, i.e., P^k, $\mathcal{F}$, P_F^k and $\mathcal{F}'$, can be depicted in Fig. 1, where the dotted arrow represents the projection operator Proj_F.

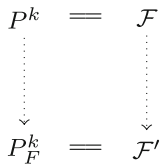

Fig. 1 The relationship between the four constraint sets

Note that the number of integer variables in Model 5 is generally smaller than that of Model 3, making it attractive. The number of SDP constraints in Model 5 is equal to the size of the clique set $\mathcal{K}$. X_{C_r} is a sub-matrix of X in Model 5. For its implementation with conic solvers (e.g., MOSEK [34]), one might follow ideas in [35] and introduce SDP cone variables that are linked by additional constraints. Specifically we do a clique-tree decomposition of an input graph and represent the clique tree as $T = (\mathcal{K}, \mathcal{E})$, where $\mathcal{K}$ is the set of maximal cliques and $\mathcal{E} \subset \mathcal{K} \times \mathcal{K}$. Then we replace constraints $X_{C_r} \in \mathbb{S}_+^{|C_r|}$ of Model 5 with the following constraints

$$\begin{aligned} &X^r \in \mathbb{S}_+^{|C_r|}, \quad r = 1, \ldots, |\mathcal{K}|, \\ &X_{ij}^r = X_{ij}^s \quad \forall (i,j) \in (C_r \cap C_s) \times (C_r \cap C_s), (C_r, C_s) \in \mathcal{E} \end{aligned}$$

and $X_{ii} = 1$ with

$$X_{ii}^{r_i} = 1$$

where C_{r_i} is first clique in $\mathcal{K}$ having index i, i.e., $r_i = \min\{r : i \in C_r, C_r \in \mathcal{K}\}$. Similarly the variables X_{ij} in the objective function of Model 5 are replaced with $X_{ij}^{r_{ij}}$ where $r_{ij} = \{r : i, j \in C^r, C_r \in \mathcal{K}\}$.

The benefit of this implementation is that one can reduce the number of the linking constraints. Moreover, when the number of introduced constraints is large we can merge two small maximal cliques to a bigger one. We direct interested readers to Algorithm proposed by Nakata et al. [35].

Analogously, the four continuous relaxations of the four constraint sets can be formed by relaxing the respective integrality constraints. We denote by $\overline{P^k}$, $\overline{\mathcal{F}}$, $\overline{P_F^k}$ and $\overline{\mathcal{F}'}$ the respective continuous relaxation sets of P^k, $\mathcal{F}$, P_F^k and $\mathcal{F}'$. It has been shown in [17] that $\overline{P^k}$ and $\overline{\mathcal{F}}$ are not contained in one another. We remark that similar results also hold for $\overline{P_F^k}$ and $\overline{\mathcal{F}'}$. This will be illustrated numerically in Sect. 4.

3.3 The construction of the clique set

Model 4 and Model 5 are attractive when they involve less constraints and integer variables than the respective Model 1 and Model 3. Hence one would like to find an "optimal" (maximal) clique set $\mathcal{K}$ that minimizes the number of constraints and variables in these models. Similar to [20,35], we remark that it is hard to determine such a chordal extension G_F and consequently the clique set $\mathcal{K}$. Alternatively, one may want to find a clique set such that the number of edges of G_F is minimized, which however is $\mathcal{NP}$-complete [44].

Nevertheless, for many practical purposes, there are good methods to find chordal extensions such that the size of $\mathcal{K}$ (measured by $\max_{C_r \in \mathcal{K}} |C_r|$) is small. We refer readers to [9] for an excellent survey. Here, we employ the `greedy fill-in heuristic` [30] to obtain G_F and $\mathcal{K}$. The greedy fill-in heuristic attempts to create few new edges, which leads to less integer variables in Model 4 and Model 5. To make the text self-contained, we present the algorithm below, where the term *fill-in* of a vertex refers to the number of pairs of its non-adjacent neighbors.

Algorithm 1: Heuristic to find a chordal extension [30]

input : Graph $G = (V, E)$
output: A chordal graph $G_F = (V, F)$ of G and cliques K_i
Initialisation: $H = G, i = 0, G_F = (V, F)$ with $F = E$
while # *Vertices of* $H \geq 2$ **do**
  if *all fill-in values of vertices in* H *is* 0 **then**
    terminates
  else
    Choose a vertex v with the smallest number of fill-in edge
    Label v with i (i.e., v_i)
    Make the neighbouring vertices $N_H(v_i)$ of v_i a clique
    $K_i = N_H(v_i) \cup \{v_i\}$
    $F = F \bigcup \{(u, v_i) : u \in N_H(v_i)\}$
    $i = i + 1$
    Remove v_i from H
  end
end

Note that we need to extract the maximal clique set $\mathcal{K}$ from $\{K_i : i = 1, \ldots, n-2\}$. Due to [21], it is known that $\mathcal{K}$ contains exactly sets K_i for which there exists no $K_i, K_j : i < j$, such that $K_j \subset K_i$. This shows that the cardinality of $\mathcal{K}$ is bounded by $(n-2)$. It should also be noted that for each two distinct (maximal) cliques $(C_r, C_s) \in \mathcal{K} \times \mathcal{K}$ such that $\left|C_r \bigcap C_s\right| \geq 3$, there are redundant triangle inequalities (7) in variables $x_{ij}, \{i, j\} \in C_r \bigcap C_s$. Similarly, for each two distinct (maximal) cliques $(C_r, C_s) \in \mathcal{K} \times \mathcal{K}$ such that $\left|C_r \bigcap C_s\right| \geq k+1$, there are redundant clique inequalities (8a). This redundancy can be avoided by checking the occurrence of each associated tuple.

3.4 The separation of valid inequalities

Model 4 can also suffer from a prohibitive number of clique inequalities when the size of some maximal clique set $C_r \in \mathcal{K}$ is large. This issue can be alleviated by a separation algorithm approach. For a maximal clique set $C_r \in \mathcal{K}$ with $|C_r| \geq k+1$, the number of clique constraints is $\binom{|C_r|}{k+1}$. This number grows roughly as fast as $|C_r|^k$ as long as $2k \leq |C_r|$. Some heuristics for the separation of cliques and other inequalities have been proposed in [18,28,38] and these algorithms might be adapted for both Model 4 and 5 in a branch and bound algorithm to solve MkP to global optimality. We leave the sufficiently thorough investigation for future research.

Table 1 Computational evaluation for Model 1 and Model 4 with $k = 3$

	Model 1				Model 4			
	Root node		Branch-and-bound		Root node		Branch-and-bound	
$(k, \|V\|)$	Time (s)	Gap (%)	Time (s)	Nodes	Time (s)	Gap (%)	Time (s)	Nodes
`spinglass2g`								
$(3, 3 \times 3)$	0.03	0.00	0.03	0	0.03	0.00	0.03	0
$(3, 4 \times 4)$	0.20	0.00	0.34	0	0.02	0.00	0.38	0
$(3, 5 \times 5)$	1.42	0.00	5.54	0	0.09	0.17	0.16	0
$(3, 6 \times 6)$	8.12	0.00	9.69	0	0.17	0.00	0.30	0
$(3, 7 \times 7)$	25.32	0.48	14,075.40	4	0.76	0.98	10.80	0
$(3, 8 \times 8)$	127.13	0.00	–	–	2.05	0.00	2.52	0
$(3, 9 \times 9)$	–	–	–	–	4.66	0.91	40.98	0
$(3, 10 \times 10)$	–	–	–	–	6.19	0.00	6.74	0
$(3, 11 \times 11)$	–	–	–	–	5.00	0.00	46.11	0
$(3, 12 \times 12)$	–	–	–	–	7.86	0.00	9.82	0
$(3, 13 \times 13)$	–	–	–	–	14.02	0.00	229.40	0
$(3, 14 \times 14)$	–	–	–	–	16.12	0.00	23.13	0
$(3, 15 \times 15)$	–	–	–	–	19.57	0.03	1420.80	0
$(3, 16 \times 16)$	–	–	–	–	16.01	0.18	58,900.00	28
`spinglass2pm`								
$(3, 3 \times 3)$	0.02	0.00	0.05	0	0.03	0.00	0.04	0
$(3, 4 \times 4)$	0.20	0.00	0.26	0	0.05	0.00	0.06	0
$(3, 5 \times 5)$	1.72	0.00	1.93	0	0.09	0.00	0.12	0
$(3, 6 \times 6)$	6.89	2.20	8.89	0	0.21	2.31	0.42	0
$(3, 7 \times 7)$	26.53	2.00	43.88	4	0.77	2.10	6.96	0
$(3, 8 \times 8)$	81.55	0.00	–	–	2.05	0.00	4.97	0
$(3, 9 \times 9)$		–	–	–	2.03	1.21	31.52	0
$(3, 10 \times 10)$		–	–	–	2.78	0.16	21.11	0
$(3, 11 \times 11)$		–	–	–	4.28	0.00	6152.60	28
$(3, 12 \times 12)$		–	–	–	6.73	0.00	18.27	0
$(3, 13 \times 13)$		–	–	–	9.61	1.08	859.37	0
$(3, 14 \times 14)$		–	–	–	12.63	1.18	–	–

4 Numerical experiments

Results in this section illustrate the following key points:

1. Model 4 is more scalable than Model 1 for general graphs.
2. Compared with Model 2, Model 4 is quite attractive when the underlying graph is chordal. For general random graphs, it is less competitive for the standard branch-and-bound algorithm although it provides stronger continuous relaxation bounds. This is mainly due to the exponential number of clique inequalities (8a).

Table 2 Continuous relaxations of Model 2 and Model 4

	Model 2				Model 4			
	Root node		Branch-and-bound		Root node		Branch-and-bound	
$(k, \|V\|)$	Time (s)	Gap (%)	Time (s)	Nodes	Time (s)	Gap (%)	Time (s)	Nodes
`band`								
(3, 50)	0.09	100.00	1588.65	345246	0.03	5.38	0.28	0
(3, 100)	0.25	100.00	–	–	0.04	5.47	0.76	0
(3, 150)	0.51	98.64	–	–	0.06	4.78	1.58	0
(3, 200)	0.35	100.00	–	–	0.07	5.51	5.14	0
(3, 250)	0.47	100.00	–	–	0.08	5.52	5.61	0
(4, 50)	0.09	107.14	–	–	0.04	12.21	4.86	0
(4, 100)	0.25	111.40	–	–	0.08	13.39	13.05	0
(4, 150)	0.38	112.79	–	–	0.09	13.78	36.36	0
(4, 200)	0.35	112.55	–	–	0.16	13.59	70.74	122
(4, 250)	0.44	113.14	–	–	0.21	13.78	112.85	254
`spinglass2g`								
(3, 10 × 10)	0.06	3.79	10.28	58	6.19	0.00	6.74	0
(3, 11 × 11)	0.07	8.50	17.63	145	5.00	0.00	46.11	0
(3, 12 × 12)	0.09	10.61	16.00	89	7.86	0.00	9.82	0
(3, 13 × 13)	0.11	9.41	34.01	584	14.02	0.00	229.40	0
(3, 14 × 14)	0.13	9.01	67.20	2126	16.12	0.00	23.13	0
(3, 15 × 15)	0.15	10.01	77.30	860	19.57	0.03	1420.80	0
(4, 10 × 10)	0.07	10.38	27.00	3075	6.99	0.00	22.70	0
(4, 11 × 11)	0.09	7.50	30.50	3654	15.212	0.00	50.47	0
(4, 12 × 12)	0.11	9.50	75.40	3068	24.696	0.00	144.87	0
(4, 13 × 13)	0.14	9.19	79.03	4211	76.670	0.00	–	–
(4, 14 × 14)	0.17	8.10	42.68	882	76.940	0.00	–	–
(4, 15 × 15)	0.34	7.20	430.39	11,314	279.98	0.00	–	–

3. The continuous relaxation of Model 5 is computationally more scalable than that of Model 3 when the underlying graph has structured sparsity.
4. The continuous relaxation of Model 4 and that of 5 do not dominate each other.

4.1 Test instances

To illustrate the above key points, we randomly generate four sets of sparse graphs. The first set includes `band` graph instances, which were used in [20,35]. The other sets are generated by the package `rudy` [37]. Similar instances have been used in [23,38] for numerical demonstration.

- `band`: we generate graphs with edges set $E = \{\{i, j\} \in V \times V : j - i \leq \alpha, i < j\}$, where α is 1 plus the partition parameter k, i.e., $\alpha = k + 1$. The 50% of edge weights are -1 and the others are 1.

Table 3 Continuous relaxations for Model 3 and Model 5

	Model 3		Model 5	
$(k, \|V\|)$	Time (s)	Gap (%)	Time (s)	Gap (%)
`spinglass2g`				
$(3, 11 \times 11)$	125.84	2.60	1.74	2.46
$(3, 12 \times 12)$	359.43	2.48	2.29	2.68
$(3, 13 \times 13)$	866.25	2.02	4.00	2.09
$(3, 14 \times 14)$	2215.69	2.09	5.64	2.23
$(3, 15 \times 15)$	5072.07	2.72	10.33	2.89
$(4, 11 \times 11)$	130.806	2.07	1.88	2.19
$(4, 12 \times 12)$	343.846	2.39	2.43	2.58
$(4, 13 \times 13)$	1164.69	2.95	4.34	3.04
$(4, 14 \times 14)$	2820.33	2.01	5.59	2.11
$(4, 15 \times 15)$	6951.17	2.25	10.88	2.30
`spinglass2pm`				
$(3, 11 \times 11)$	876.87	8.75	1.21	9.11
$(3, 12 \times 12)$	2227.27	7.83	1.66	8.19
$(3, 13 \times 13)$	5500.92	7.94	2.54	8.41
$(3, 14 \times 14)$	11349.41	9.14	3.37	9.51
$(3, 15 \times 15)$	33188.43	10.54	5.67	10.98
$(4, 11 \times 11)$	102.62	9.09	1.26	10.26
$(4, 12 \times 12)$	291.59	8.52	1.73	8.86
$(4, 13 \times 13)$	729.29	9.41	2.80	9.84
$(4, 14 \times 14)$	1905.11	9.52	3.56	9.94
$(4, 15 \times 15)$	4791.15	11.05	6.67	11.48

- `springlass2g`: These consist of instances of a toroidal two dimensional grid with gaussian interactions. The graph has $n =$(rows $\times$ columns) vertices.
- `spinglass2pm`: It generates a toroidal two-dimensional grid with ± 1 weights. The grid has size $n =$ (rows $\times$ columns). The percentage of negative weights is 50%.
- `rndgraph`: we generate a random graph of n nodes and density 10%. The edge weights are all 1.

4.2 Implementation and experiments setup

Models 1, 2, 3, 4 and 5 are implemented in GRAVITY [27]. Source code and test instances can be found online [43]. MIP problem instances are solved by CPLEX 12.7 [13] with default settings. The SDP relaxation instances are solved by the state-of-the-art solver MOSEK 8 [34] with default tolerance settings, which exploits the sparsity for performance and scalability gains. As remarked in Sect. 3.2, we adopt the conversion method in [35] for the implementation of the continuous relaxation of Model 5 with MOSEK 8.

Table 4 Continuous relaxations of Model 4 and Model 5 with $k = 3$ (`rndgraph`)

	Model 4		Model 5	
$(k, \|V\|)$	Time (s)	Value	Time (s)	Value
(3, 10)	0.01	0.00	0.06	0.00
(3, 20)	0.00	0.00	0.03	0.00
(3, 30)	0.01	0.00	0.03	0.00
(3, 40)	0.07	0.67	0.05	0.14
(3, 50)	0.61	0.00	0.12	0.00
(3, 60)	1.97	1.11	0.25	0.44
(3, 70)	3.75	4.01	0.49	2.53
(3, 80)	12.44	4.94	1.09	3.53
(3, 90)	24.46	10.96	1.69	9.86
(3, 100)	49.76	14.67	3.73	16.45
(3, 110)	85.13	23.53	5.62	26.11
(3, 120)	209.22	28.86	12.21	35.53
(3, 130)	278.67	37.00	41.07	48.58
(3, 140)	–	–	54.40	61.60
(3, 150)	–	–	98.33	78.46

The experiments are conducted on a Mac with Intel Core i5 clocked at 2.7 GHz and with 8 GB of RAM. For a fair computational comparison, CPU time is used for all computations. A wall-clock time limit of 10 h was used for all computations. If no solution is available at solver termination or the solution process is killed by the solver, (–) is reported.

4.3 Analysis of the computational results

Results on Model 1 and Model 4

As remarked in the previous sections, both Models 1 and 4 suffer from a prohibitive number of clique inequalities. Thus we fix $k = 3$. For each problem instance, we measure the computational performance of each formulation by its respective computational time and optimality gap (if it has). All computational time is measure by the CPU time in seconds. As all relaxations lead to a lower bound of the optimum, we quantify optimality gap as

$$\text{Gap} = \frac{\text{Optimum - Lower bound}}{|\text{Optimum}|} \times 100\%$$

The numerical results are presented in Table 1. For each problem instance, the statistics on root node relaxation and the full branch-and-bound procedure are reported. These tested cases illustrate the following key points.

1. Model 4 is remarkably more scalable for all tested instances than Model 1.

2. The continuous relaxations of both models are strong. It is worth mentioning that the optimality gaps at root nodes for Model 4 are nearly the same with Model 1.
3. For problem instances with over 100 vertices, the computational time for Model 4 grows exponentially as the problem size increases. This is probably due to the fact that the size of each clique in $\mathcal{K}$ becomes larger, leading to an exponential number of clique inequalities (8a).

Results on Model 4 and Model 2

Let us now compare the results of Models 4, 2 presented in Table 2. First, for `band` instances, the performance of Model 4 is significantly better than Model 2. This is largely because Model 4 has much less binary variables and constraints due the small sizes of its maximal clique sets. Second, for `spinglass2g` problem instances, the performance of Models 4 and 2 are similar. When k or the sizes of instances get larger, Model 4 is less attractive than Model 2. This is probably because the sizes of the maximal clique sets are large, causing a great number of clique inequalities. Third, the strength of Model 4 is generally much stronger than that of 2. This is illustrated by instances of `spinglass2g`, where the continuous relaxation of Model 4 leads to 0 optimality gap.

Results on Model 3 and Model 5

We now compare the performance of Model 3 and Model 5 with respect to the optimality gap and the solution time. The numerical results are summarized in Table 3.

Overall, we remark that that the continuous relaxation of Model 5 reduces significantly the computational time compared with that of Model 3, though a bit inferior in terms of the optimality gap. In addition, as expected, the computational time for Model 5 grows linearly with respect to the size of the graph while the time for Model 3 grows more significantly as the instance size increases.

Results on Model 4 and Model 5

We compare the strength of Model 4 and Model 5 with respect to their continuous relaxation values and solution time.

Our previous numerical results show that the computational time and bounds of both Model 4 and Model 5 are quite similar for test instances of `band, spinglass2g, spinglass2pm`. To contrast these two compact models, we present the numerical results for tests instances of `rndgraph`. Our experiments indicate that the standard branch-and-bound algorithm with CPLEX 12.7 is not able to return global optimal values for graphs with 100 vertices within reasonable time. Thus the optimality gap for both relaxations is not available. Instead, the their continuous relaxation values are reported. Since MkP is a minimization problem, the higher the value is, the stronger is the lower bound. The numerical results are summarized in Table 4. We outline the following key observations.

1. The strength of the continuous relaxation of Model 5 neither dominates nor is dominated by that of 4. See, for instance, problem instance (3, 40) and (3, 100).

2. For larger problem instances ($|V| \geq 100$), Model 5 appears much more attractive than Model 4 in terms of computational scalability.

5 Conclusion

This work introduces two more compact MkP reformulations exploiting the structured sparsity of the underlying graph. The first model is an ILP while the second is an ISDP. Both are based on the maximal clique set corresponding to the chordal extension of the original graph. Numerical results show that the proposed models numerically dominate state-of-the-art formulations.

Based on the results presented in this paper, several research directions can be considered. First, alternative algorithms may be implemented to obtain optimal clique sets minimizing the number of integer variables. Second, separation algorithms for valid inequalities can be investigated. Third, specialized branch-and-bound algorithms [42] for Model 5 can be considered in comparison with existing results in [1]. Lastly, combining Models 4 and 2 to obtain a novel ILP is also left for future research.

Acknowledgements The authors are grateful to the associate editor and the anonymous referees for their helpful suggestions that greatly improved the quality of the paper. This research was partly funded by the Australia Indonesia Centre.

References

1. Anjos, M.F., Ghaddar, B., Hupp, L., Liers, F., Wiegele, A.: Solving k-way graph partitioning problems to optimality: the impact of semidefinite relaxations and the bundle method. In: Jünger, M., Reinelt, G. (eds.) Facets of Combinatorial Optimization: Festschrift for Martin Grötschel, pp. 355–386. Springer, Heidelberg (2013)
2. Arnborg, S., Proskurowski, A.: Linear time algorithms for NP-hard problems restricted to partial k-trees. Discrete Appl. Math. **23**(1), 11–24 (1989)
3. Barahona, F., Mahjoub, A.R.: On the cut polytope. Math. Program. **36**(2), 157–173 (1986)
4. Ben-Ameur, W., Ouorou, A., Wang, G.: Convex and concave envelopes: revisited and new perspectives. Oper. Res. Lett. **45**(5), 421–426 (2017)
5. Berge, C.: Some classes of perfect graphs. In: Harary, F. (ed.) Graph Theory and Theoretical Physics, pp. 155–165. Academic Press, New York (1967)
6. Bienstock, D., Muñoz, G.: LP formulations for polynomial optimization problems. SIAM J. Optim. **28**(2), 1121–1150 (2018)
7. Bienstock, D., Ozbay, N.: Tree-width and the Sherali–Adams operator. Discrete Optim. **1**(1), 13–21 (2004)
8. Blair, J.R., Peyton, B.: An introduction to chordal graphs and clique trees. In: George, A., Gilbert, J.R., Liu, J.W.H. (eds.) Graph Theory and Sparse Matrix Computation, pp. 1–29. Springer, New York (1993)
9. Bodlaender, H.L., Koster, A.M.: Treewidth computations I. Upper bounds. Inf. Comput. **208**(3), 259–275 (2010)
10. Carlson, R., Nemhauser, G.L.: Scheduling to minimize interaction cost. Oper. Res. **14**(1), 52–58 (1966)
11. Chopra, S., Rao, M.: Facets of the k-partition polytope. Discrete Appl. Math. **61**(1), 27–48 (1995)
12. Chopra, S., Rao, M.R.: The partition problem. Math. Program. **59**(1–3), 87–115 (1993)
13. Cplex: Cplex 12.7 user's manual. ILOG (2010)
14. Deza, M., Grötschel, M., Laurent, M.: Complete descriptions of small multicut polytopes. In: Gritzmann, P., Sturmfels, B. (eds.) Applied Geometry and Discrete Mathematics, pp. 221–252. AMS, Providence (1991)

15. Deza, M., Grötschel, M., Laurent, M.: Clique-web facets for multicut polytopes. Math. Oper. Res. **17**(4), 981–1000 (1992)
16. Deza, M., Laurent, M.: Geometry of Cuts and Metric Embeddings. Springer, New York (1997)
17. Eisenblätter, A.: The semidefinite relaxation of the k-partition polytope is strong. In: Cook, W., Schulz, A. (eds.) Proceedings of the 9th International Conference on Integer Programming and Combinatorial Optimization, Lecture Notes in Computer Science, vol. 2337, pp. 273–290. Springer, Heidelberg (2002)
18. Fairbrother, J., Letchford, A.N.: Projection results for the k-partition problem. Discrete Optim. **26**, 97–111 (2017)
19. Frieze, A., Jerrum, M.: Improved approximation algorithms for max-k-cut and max bisection. Algorithmica **18**(1), 67–81 (1997)
20. Fukuda, M., Kojima, M., Murota, K., Nakata, K.: Exploiting sparsity in semidefinite programming via matrix completion I: general framework. SIAM J. Optim. **11**(3), 647–674 (2001)
21. Fulkerson, D., Gross, O.: Incidence matrices and interval graphs. Pac. J. Math. **15**(3), 835–855 (1965)
22. Gavril, F.: Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. SIAM J. Comput. **1**(2), 180–187 (1972)
23. Ghaddar, B., Anjos, M.F., Liers, F.: A branch-and-cut algorithm based on semidefinite programming for the minimum k-partition problem. Ann. Oper. Res. **188**(1), 155–174 (2011)
24. Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. J. ACM **42**(6), 1115–1145 (1995)
25. Golumbic, M.C.: Algorithmic Graph Theory and Perfect Graphs. Academic Press, New York (1980)
26. Grone, R., Johnson, C.R., Sá, E.M., Wolkowicz, H.: Positive definite completions of partial Hermitian matrices. Linear Algebra Appl. **58**, 109–124 (1984)
27. Hijazi, H., Wang, G., Coffrin, C.: Gravity: A mathematical modeling language for optimization and machine learning. Machine Learning Open Source Software Workshop at NIPS 2018 (2018)
28. Kaibel, V., Peinhardt, M., Pfetsch, M.E.: Orbitopal fixing. Discrete Optim. **8**(4), 595–610 (2011)
29. Kim, S., Kojima, M., Mevissen, M., Yamashita, M.: Exploiting sparsity in linear and nonlinear matrix inequalities via positive semidefinite matrix completion. Math. Program. **129**(1), 33–68 (2011)
30. Koster, A.M., Bodlaender, H.L., Van Hoesel, S.P.: Treewidth: computational experiments. Electron. Notes Discrete Math. **8**, 54–57 (2001)
31. Lasserre, J.B.: Convergent SDP-relaxations in polynomial optimization with sparsity. SIAM J. Optim. **17**(3), 822–843 (2006)
32. Laurent, M.: A connection between positive semidefinite and euclidean distance matrix completion problems. Linear Algebra Appl. **273**(1), 9–22 (1998)
33. Michael, R.G., David, S.J.: Computers and Intractability: A Guide to the Theory of NP-Completeness. Freeman, New York (1979)
34. MOSEK: MOSEK Fusion API for C++. http://docs.mosek.com (2017)
35. Nakata, K., Fujisawa, K., Fukuda, M., Kojima, M., Murota, K.: Exploiting sparsity in semidefinite programming via matrix completion II: implementation and numerical results. Math. Program. **95**(2), 303–327 (2003)
36. Papadimitriou, C., Yannakakis, M.: Optimization, approximation, and complexity classes. In: Proceedings of the Twentieth Annual ACM Symposium on Theory of computing, pp. 229–234. ACM, New York (1988)
37. Rinaldi, G.: Rudy, a graph generator. https://www-user.tu-chemnitz.de/helmberg/sdp_software.html (1998)
38. de Sousa, V.J.R., Anjos, M.F., Le Digabel, S.: Computational study of valid inequalities for the maximum k-cut problem. Ann. Oper. Res. **265**(1), 5–27 (2018)
39. Vandenberghe, L., Andersen, M.S., et al.: Chordal graphs and semidefinite optimization. Found. Trends Optim. **1**(4), 241–433 (2015)
40. Waki, H., Kim, S., Kojima, M., Muramatsu, M.: Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity. SIAM J. Optim. **17**(1), 218–242 (2006)
41. Wang, G.: Relaxations in mixed-integer quadratically constrained programming and robust programming. Ph.D. thesis, Evry, Institut national des télécommunications (2016)
42. Wang, G., Ben-Ameur, W., Ouorou, A.: A Lagrange decomposition based branch and bound algorithm for the optimal mapping of cloud virtual machines. Eur. J. Oper. Res. **276**(1), 28–39 (2019)
43. Wang, G., Hijazi, H.: An implementation for solving the min k-partition problem. https://zenodo.org/record/3252476. https://doi.org/10.5281/zenodo.3252476 (2019)

44. Yannakakis, M.: Computing the minimum fill-in is NP-complete. SIAM J. Algebr. Discrete Methods **2**(1), 77–79 (1981)