Concise integer linear programming formulation for clique partitioning problems

Miyuki Koshimura¹ · Emi Watanabe¹ · Yuko Sakurai¹,² · Makoto Yokoo¹

Abstract
A Clique Partitioning Problem (CPP) finds an optimal partition of a given edge-weighted undirected graph, such that the sum of the weights is maximized. This general graph problem has a wide range of real-world applications, including correlation clustering, group technology, community detection, and coalition structure generation. Although a CPP is NP-hard, due to the recent advance of Integer Linear Programming (ILP) solvers, we can solve reasonably large problem instances by formulating a CPP as an ILP instance. The first ILP formulation was introduced by Grötschel and Wakabayashi (Mathematical Programming, 45(1-3), 59–96, 1989). Recently, Miyauchi et al. (2018) proposed a more concise ILP formulation that can significantly reduce transitivity constraints as compared to previously introduced models. In this paper, we introduce a series of concise ILP formulations that can reduce even more transitivity constraints. We theoretically evaluate the amount of reduction based on a simple model in which edge signs (positive/negative) are chosen independently. We show that the reduction can be up to 50% (dependent of the ratio of negative edges) and experimentally evaluate the amount of reduction and the performance of our proposed formulation using a variety of graph data sets. Experimental evaluations show that the reduction can exceed 50% (where edge signs can be correlated), and our formulation outperforms the existing state-of-the-art formulations both in terms of memory usage and computational time for most problem instances.

Keywords Clique partitioning problem · Integer linear programming · Transitivity constraints

This work was supported by JSPS KAKENHI Grant Numbers JP19H04175, JP20H00609, JP18H03299.

✉ Miyuki Koshimura
koshi@inf.kyushu-u.ac.jp

1 Kyushu University, 744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan
2 National Institute of Advanced Industrial Science and Technology (AIST) 2-4-7 Aomi, Koto-ku, Tokyo 135-0064, Japan
1 Introduction

A Clique Partitioning Problem (CPP) finds an optimal partition of a given edge-weighted undirected graph that maximizes the sum of the weights within clusters [9]. An edge weight can be positive or negative. As the popularity of analyzing complex networks has increased (e.g., the Internet and social networks), CPPs can be found in a variety of important AI applications:

**Correlation clustering** [1]: For given objects (e.g., documents) with similarity scores, we are interested in finding clusters of similar objects (e.g., clusters of documents with the same topic). This problem can be formalized as a CPP where an object/document is a vertex and similarity scores among objects are edge weights.

**Group technology** [16, 19]: Assume we are developing a manufacturing system that requires several parts and machines for processing. The goal is to find a suitable partition of the parts and machines for efficient production. This problem can be formalized as a CPP where each part/machine is a vertex. If part $i$ is processed by machine $j$, edge $(i, j)$ has weight 1 (otherwise, its weight is $-1$). An edge between parts (or machines) has weight 0.

**Community detection** [2, 4, 8]: For a given network, we are interested in partitioning the vertices into communities. A community is a subset of vertices densely/positively connected within the same community and sparsely/negatively connected with other communities. Community detection in a network has a long research history (e.g., see the survey by Fortunato [8]). Community detection in bipartite networks, which can be formalized as a special case of a CPP called a modularity maximization problem, is proposed by Brandes et al. [4] and has gathered a significant amount of interest [2, 8].

**Coalition structure generation** [17]: A pair of agents can have positive/negative synergy, e.g., friends who collaborate well or rivals who do not. The goal is to partition agents into groups (coalitions) to maximize the sum of their synergies. This problem can be formalized as a CPP where an agent is a vertex and the synergies among the agents are edge weights.

Although a CPP is NP-hard, due to the recent advance of Integer Linear Programming (ILP) solvers (e.g., CPLEX and Gurobi), we can solve reasonably large problem instances by formulating a CPP as an ILP instance. Grötschel and Wakabayashi [9] proposed the first IPL formulation of a CPP, where the goal was selecting a subset of edges such that the sum of their weights is maximized and satisfies transitivity constraints. The number of transitivity constraints is $\Theta(n^3)$, where $n$ is the number of vertices. Thus, the growth of the number of transitivity constraints creates a bottleneck for solving larger CPP instances.

To address this scalability issue, several researchers have proposed more concise ILP formulations [3, 5, 6, 11, 14, 15]. Dinh and Thai [7] proposed a concise ILP formulation for the modularity maximization problem. Inspired by this work, Miyauchi and Sukegawa [12] proposed an ILP formulation for a general CPP by identifying redundant transitivity constraints by focusing on the signs of edge weights.

In this paper, we propose further reductions of the transitivity constraints that utilize the characterization of transitivity based on reachability among vertices.\(^1\) Our method can be used in conjunction with the state-of-the-art method proposed by Miyauchi et al. [13], whose formulation removed significantly more constraints, including non-redundant ones. As a result, the obtained ILP solution can be suboptimal. Thus, they proposed a method for obtaining an optimal solution by post-processing.

\(^1\) Liao et al. [10] also proposed a method to reduce the transitivity constraints for coalition structure generation problems based on reachability among vertices. Unlike our method, their work assumes that a problem instance is represented as a set of rules rather than a graph and deals with MaxSAT encoding.
We theoretically evaluate the amount of reduction based on a model in which edge signs are chosen independently and show that the ratio of the number of constraints between our proposed and the existing formulations is 50% when the ratio of the non-negative edges approaches 0. Furthermore, we experimentally evaluate the amount of reduction and the performance of our proposed formulation using well-known, real-world graph datasets related to CPP applications. We show that the reduction can exceed 50% when the edge signs can be correlated, and our formulations outperform the existing state-of-the-art formulations in terms of both memory usage and computational time for most problem instances.

2 Model

A Clique Partitioning Problem (CPP) instance is a complete undirected edge-weighted graph \( G = (V, E, w) \) consisting of vertices \{1, 2, ..., n\}, \( n(n-1)/2 = |E| \) edges, and weight function \( w : E \rightarrow \mathbb{R} \). For simplicity, we denote \( w_{i,j} = w(\{i, j\}) \) for each \( \{i, j\} \in E \). Clearly, \( w_{i,j} = w_{j,i} \). Set \( A \), which is a subset of \( E \), is called clique partitioning if partition \( \{V_1, V_2, ..., V_p\} \) of \( V \) exists such that \( A = \bigcup_{\ell=1}^{p} \{\{i, j\} \in E | i, j \in V_\ell\} \). We call each \( V_\ell \) a cluster (\( \ell = 1, ..., p \)). CPP’s goal is to find clique partitioning \( A \) to maximize the sum of weights \( \sum_{\{i,j\} \in A} w_{i,j} \).

3 Existing concise ILP formulations

Let us introduce the ILP formulation based on Grötschel and Wakabayashi [9]. Decision variable \( x_{i,j} \) is introduced for each edge \( \{i, j\} \in E \) where \( i < j \). \( x_{i,j} \) equals 1 if \( i \) and \( j \) are in the same cluster, and 0 otherwise.

Transitivity constraints are introduced for any \( i, j, k \in V \). If \( i \) and \( j \) are in the same cluster and \( j \) and \( k \) are in the same cluster, then \( i \) and \( k \) must also be in the same cluster. This ILP formulation, called \( P(G) \), is defined:

\[
\max \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} w_{i,j}x_{i,j}
\]

s.t.

\[
x_{i,j} + x_{j,k} - x_{i,k} \leq 1 \quad \forall (i, j, k) \in T,
\]

\[
x_{i,j} - x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in T,
\]

\[
-x_{i,j} + x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in T,
\]

\[
x_{i,j} \in \{0, 1\} \quad 1 \leq i < j \leq n,
\]

where

\[
T = \{(i, j, k) | 1 \leq i < j < k \leq n\}.
\]

This formulation has \( \left( \begin{array}{c} n \\ 2 \end{array} \right) = \Theta(n^2) \) variables and \( 3 \left( \begin{array}{c} n \\ 3 \end{array} \right) = \Theta(n^3) \) transitivity constraints.

Miyauchi and Sukegawa [12] proposed a more concise ILP formulation called \( RP(G) \), which reduces the number of transitivity constraints. More specifically, we replace (1) in the above formulation with:

\[
x_{i,j} + x_{j,k} - x_{i,k} \leq 1 \quad \forall (i, j, k) \in T_{1}^{1},
\]

\[
x_{i,j} - x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in T_{2}^{1},
\]

\[
-x_{i,j} + x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in T_{3}^{1},
\]

\[
x_{i,j} \in \{0, 1\} \quad 1 \leq i < j \leq n,
\]
where

\[ T^1_{\geq 0} = \{(i, j, k) \in T : w_{i,j} \geq 0 \lor w_{j,k} \geq 0\}, \]

\[ T^2_{\geq 0} = \{(i, j, k) \in T : w_{i,j} \geq 0 \lor w_{i,k} \geq 0\}, \]

\[ T^3_{\geq 0} = \{(i, j, k) \in T : w_{j,k} \geq 0 \lor w_{i,k} \geq 0\}. \]

A transitivity constraint, such that \( w_{i,j} < 0 \land w_{j,k} < 0 \) holds, is redundant and safely removed. \( \text{RP}(G) \) has \( O(nm_{\geq 0}) \) constraints, where \( m_{\geq 0} \) indicates the number of non-negative weighted edges in \( G \).

### 4 New concise ILP formulations

#### 4.1 Overview of our main idea and contributions

We introduce several terms to explain our main idea. Let \( x^* = (x^*_i,j)_{1 \leq i < j \leq n} \) denote an arbitrary optimal solution of an ILP formulation, \( E_{\geq 0} \) denote \( \{(i, j) \in E : w_{i,j} \geq 0\} \), and \( E^* \) denote \( \{(i, j) \in E : x^*_{i,j} = 1\} \). Furthermore, let \( \{(V_1, E^*_1), (V_2, E^*_2), \ldots, (V_p, E^*_p)\} \) denote a set of connected components of \( (V, E^*) \). For each \( \ell \in \{1, \ldots, p\} \), for a pair of vertices \( i, j \in V_\ell \) (where \( i < j \)), they are directly reachable if \( \{i, j\} \in E^*_\ell \), and non-negatively, directly reachable if \( \{i, j\} \in E^*_\ell \cap E_{\geq 0} \). They are also reachable if a path exists from \( i \) to \( j \) that consists of edges in \( E^*_\ell \), and non-negatively reachable if a path exists from \( i \) to \( j \) that consists of edges in \( E^*_\ell \cap E_{\geq 0} \).

**Example 1** Consider one connected component \((V_\ell, E^*_\ell)\) where \( V_\ell = \{1, 2, 3, 4, 5\} \) and \( E^*_\ell = \{\{1, 2\}, \{2, 3\}, \{2, 4\}, \{3, 4\}, \{4, 5\}\} \). Weight function \( w \) on \( E^*_\ell \) is as follows: \( w_{1,2} = -1, w_{2,3} = 0, w_{2,4} = 1, w_{3,4} = 0 \) and \( w_{4,5} = 2 \). Figure 1 illustrates the component. Here 1 and 2 are directly reachable but not non-negatively, directly reachable, and 3 and 4 are non-negatively, directly reachable. 3 and 5 are non-negatively reachable, and 1 and 5 are not non-negatively reachable.

The goal of transitivity constraints is to ensure that the following property holds.

If \( i \) and \( j \) are reachable, they are directly reachable, i.e., \( x_{i,j} = 1 \) must hold.

As shown by Lemma 1, if \( i \) and \( j \) are reachable, then they are non-negatively reachable. The intuition of this lemma is simple. If \( i, j \in V_\ell \) are not non-negatively connected, by removing only negative weight edges, we can divide \( V_\ell \) into two parts, contradicting the fact that \( x^* \) is an optimal solution. For example, in Fig. 1, since vertices 1 and 5 are not non-negatively reachable, the component is not optimal; we can divide \( V_\ell \) into two parts, i.e., \( \{1\} \) and \( \{2, 3, 4, 5\} \), by removing only negative weight edge \( \{1, 2\} \).

Thus, the above goal can be modified to guarantee the following property, which corresponds to Lemma 2:

**Fig. 1** Example of connected component

\[
\begin{array}{cccccc}
1 & 2 & 3 & 4 & 5 \\
-1 & 1 & 0 & 0 & 2
\end{array}
\]
If $i$ and $j$ are non-negatively reachable, they are directly reachable.

Transitivity constraints must be defined to ensure this property. With these terms, the transitivity constraints in [12] can be translated:

For any $i < k$ and for any $j$ s.t. $j \neq i, j \neq k$, if at least one of the following conditions holds, then $i$ and $k$ must be directly reachable (i.e., $x_{i,k} = 1$ must hold):

(i) $i$ and $j$ are directly reachable, and $j$ and $k$ are non-negatively, directly reachable, or

(ii) $i$ and $j$ are non-negatively, directly reachable, and $j$ and $k$ are directly reachable.

We argue that having only (i) instead of (i) and (ii) together is sufficient, i.e., (ii) is redundant if we have (i) to derive Lemma 2. Similarly, if we have (ii), (i) is redundant. This fact is verified easily by mathematical induction on the length of the non-negative path from $i$ to $k$.

In the rest of this section, we show a series of ILP formulations based on this idea, called CP($G$), CP*($G$), ACP($G$), and ACP*($G$). Intuitively, we can remove about half of the transitivity constraints. However, since (i) and (ii) can overlap, this intuition is not exact. In the rest of this section, we scrutinize the amount of reduction.

4.2 CP($G$)

We first present our new concise ILP formulation called CP($G$), which is based on RP($G$). It is identical to RP($G$) (or P($G$)) except that we replace (1) with:

$$
\begin{align*}
ix_{i,j} + x_{j,k} - x_{i,k} &\leq 1 \quad \forall (i, j, k) \in C^1_{\geq 0}, \\
ix_{i,j} - x_{j,k} + x_{i,k} &\leq 1 \quad \forall (i, j, k) \in C^2_{\geq 0}, \\
-x_{i,j} + x_{j,k} + x_{i,k} &\leq 1 \quad \forall (i, j, k) \in C^3_{\geq 0}, \\
x_{i,j} &\in \{0, 1\} \quad 1 \leq i < j \leq n,
\end{align*}
$$

where

$$
\begin{align*}
C^1_{\geq 0} &= \{(i, j, k) \in T : w_{j,k} \geq 0\}, \\
C^2_{\geq 0} &= \{(i, j, k) \in T : w_{i,k} \geq 0\}, \\
C^3_{\geq 0} &= \{(i, j, k) \in T : w_{j,k} \geq 0\}.
\end{align*}
$$

Here the requirement of (i) is given as $C^1_{\geq 0} \cup C^2_{\geq 0} \cup C^3_{\geq 0}$.

Let $x^* = \{(x^*_{i,j})_{1 \leq i < j \leq n}\}$ be an arbitrary optimal solution of CP($G$). We first present the following lemma where $\{(V_1, E^*_1), (V_2, E^*_2), \ldots, (V_p, E^*_p)\}$ denotes the connected components of $(V, E^*)$ and $E^* = \{(i, j) \in E | x^*_{i,j} = 1\}$.

Lemma 1 $(V_\ell, E^*_\ell \cap E_{\geq 0})$ is connected for each $\ell \in \{1, \ldots, p\}$.

Proof We prove this lemma in the same manner as the proof of Lemma 1 [12]. It suffices to show that for any partition $\{S_i, S_j\}$ of $V_\ell$, where $i \in S_i$ and $j \in S_j$, there exists a non-negative edge in $E^*_\ell \cap E_{\geq 0}$ whose one endpoint is in $S_i$ and its other is in $S_j$. 

 Springer
From the definition of \( V_\ell \), at least one edge exists in \( E_\ell^* \) between \( S_i \) and \( S_j \). Suppose that none of these connecting edges belong to \( E_{\geq 0} \), i.e., their weights are strictly negative. We focus on a CP solution obtained by changing the values of the variables that correspond to these edges from 1 to 0 on \( x^* \). This operation resembles partitioning \( V_\ell \) into \( S_i \) and \( S_j \).

This solution is feasible for CP because each transitivity constraint that includes a removed edge is satisfied since at least two terms equal 0. Thus, the objective value of this solution is strictly greater than that of \( x^* \). This contradicts the optimality of \( x^* \).

Lemma 1 argues that if \( i \) and \( j \) in \( V_\ell \) are reachable, then they are non-negatively reachable. Next we prove that if \( i \) and \( j \) are non-negatively reachable, then they are directly reachable. In the following, for notation simplicity, let \( x_{\{i,j\}} \) denote \( x_{i,j} \) when \( i < j \), and otherwise \( x_{j,i} \). With this notation, the three inequalities in (1) are merged into one inequality

\[
x_{\{i,j\}} + x_{\{j,k\}} - x_{i,k} \leq 1,
\]

where \( i < k \).

Lemma 2 \((V_\ell, E_\ell^*) \) is complete for each \( \ell \in \{1, \ldots, p\}\).

**Proof** Let \( i \) and \( j \) be two vertices in \( V_\ell \) \((i < j)\). According to Lemma 1, there exists path \( i = u_0, u_1, \ldots, u_q = j \) on \( E_\ell^* \cap E_{\geq 0} \). We prove \( x_{i,j}^* \) = 1 by induction on the path’s length \( q \).

(Base case): \( x_{i,j}^* = x_{u_0,u_1}^* = 1 \) is clearly true.

(Inductive step): Assume the induction hypothesis where the lemma is true for an arbitrary value of \( q - 1 \). Next we must show that \( x_{i,j}^* = 1 \), that is, \( x_{u_0,u_q}^* = 1 \). According to the induction hypothesis, \( x_{u_0,u_{q-1}}^* = 1 \). Since \( u_0 = i < j = u_q \) and the edge between \( u_{q-1} \) and \( u_q \) is in \( E_{\geq 0} \), constraint

\[
x_{\{u_0,u_{q-1}\}} + x_{\{u_{q-1},u_q\}} - x_{u_0,u_q} \leq 1,
\]

is contained in CP\((G)\). Thus, using \( x_{\{u_0,u_{q-1}\}} = 1 \) and \( x_{\{u_{q-1},u_q\}} = 1 \), we have \( x_{u_0,u_q} = 1 \).

Now we are ready to present our main theorem.

**Theorem 1** CP\((G)\) and P\((G)\) have the same set of optimal solutions.

**Proof** It is sufficient to show that any optimal solution \( x^* \) of CP\((G)\) is feasible for P\((G)\). This is equivalent to the statement that \((V_\ell, E_\ell^*) \) is complete for each \( \ell \in \{1, 2, \ldots, p\}\). This is what Lemma 2 argues.

It is straightforward to show that Theorem 1 holds if we choose (ii) instead of (i) for transitivity constraints.

According to Theorem 1 [12], RP\((G)\) and P\((G)\) have the same set of optimal solutions. Thus, we obtain the following corollary.

**Corollary 1** CP\((G)\) and RP\((G)\) have the same set of optimal solutions.

We examine the numbers of constraints in RP\((G)\) and CP\((G)\). Let \( a \) denote the probability that an edge has a non-negative weight and formally define it as follows:

\[
a = \frac{|\{(i, j) \in E \mid i < j, w_{i,j} \geq 0\}|}{\binom{n}{2}}.
\]
Let \( |RP(G)| \) and \( |CP(G)| \) denote \( |T^1_{\geq 0} \cup T^2_{\geq 0} \cup T^3_{\geq 0}| \) and \( |C^1_{\geq 0} \cup C^2_{\geq 0} \cup C^3_{\geq 0}| \), i.e., the number of constraints in \( RP(G) \) and \( CP(G) \).

**Theorem 2** When the probability in which the weight of each edge is non-negative is independent for each edge, the ratio of the number of constraints in \( CP(G) \) against that in \( RP(G) \) can be estimated:

\[
\frac{|CP(G)|}{|RP(G)|} = \frac{1}{2 - a}.
\]

**Proof** We first estimate the number of transitivity constraints in \( RP(G) \).

\[ |T| = |\{(i, j, k) | 1 \leq i < j < k \leq n\}| = \binom{n}{3}. \]

Tuple \((i, j, k)\) is included in \( T^1_{\geq 0} \) except when both \( w_{i,j} \) and \( w_{j,k} \) are negative. Thus, this probability is given as \( 1 - (1 - a)^2 = 2a - a^2 \). The same argument is applicable to \( T^2_{\geq 0} \) and \( T^3_{\geq 0} \). Then \( |T^1_{\geq 0} \cup T^2_{\geq 0} \cup T^3_{\geq 0}| \) can be estimated as \( 3(2a - a^2) \binom{n}{3} \).

Then we estimate the number of transitivity constraints in \( CP(G) \). Tuple \((i, j, k)\) is included in \( C^1_{\geq 0} \) only when \( w_{j,k} \geq 0 \), which occurs with probability \( a \). Thus, \( |C^1_{\geq 0} \cup C^2_{\geq 0} \cup C^3_{\geq 0}| \) is estimated: \( 3a \binom{n}{3} \).

Therefore, we obtain

\[
\frac{|CP(G)|}{|RP(G)|} = \frac{a}{2a - a^2} = \frac{1}{2 - a}.
\]

Since \( 0 \leq a \leq 1 \),

\[
\frac{1}{2} \leq \frac{|CP(G)|}{|RP(G)|} \leq 1.
\]

Furthermore, this ratio approaches 1/2 when \( a \) approaches 0, and it approaches 1 when \( a \) approaches 1. \( \square \)

**4.3 CP*(G)**

We present another concise ILP formulation called \( CP^*(G) \), based on \( RP^*(G) \) [13]. In a real-life network, most edges are commonly zero-weighted. By taking this fact into account, Miyauchi et al. [13] introduced \( RP^*(G) \), which is identical to \( RP(G) \) (or \( P(G) \)) except that we replace (1) with:

\[
\begin{align*}
&x_{i,j} + x_{j,k} - x_{i,k} \leq 1 \quad \forall (i, j, k) \in T^1_{>0}, \\
&x_{i,j} - x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in T^2_{>0}, \\
&-x_{i,j} + x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in T^3_{>0}, \\
&x_{i,j} \in \{0, 1\} \quad 1 \leq i < j \leq n,
\end{align*}
\]

where

\[
\begin{align*}
&T^1_{>0} = \{(i, j, k) \in T : w_{i,j} > 0 \lor w_{j,k} > 0\}, \\
&T^2_{>0} = \{(i, j, k) \in T : w_{i,j} > 0 \lor w_{i,k} > 0\}, \\
&T^3_{>0} = \{(i, j, k) \in T : w_{j,k} > 0 \lor w_{j,k} > 0\}.
\end{align*}
\]

Since \( RP^*(G) \) reduces non-redundant transitivity constraints, a connected component can be incomplete and the obtained partition can be infeasible for \( P(G) \). The following example presented by [13] shows the infeasibility.
Example 2 Let $G = (V, E, w)$ with $V = \{1, 2, 3, 4\}$, $w_{1,2} = 1$, $w_{1,3} = w_{2,3} = -1$, and $w_{1,4} = w_{2,4} = w_{3,4} = 0$. A 0-1 vector $\tilde{x}^* = (\tilde{x}_{i,j})^*$ such that $\tilde{x}_{1,2}^* = \tilde{x}_{1,4}^* = \tilde{x}_{2,4}^* = \tilde{x}_{3,4}^* = 1$ and $\tilde{x}_{1,3}^* = \tilde{x}_{2,3}^* = 0$ is one of the optimal solutions to $\text{RP}^*(G)$; however, transitivity constraint $-x_{1,3} + x_{3,4} + x_{1,4} \leq 1$ in $P(G)$ is violated.

Thus, Miyachi et al. [13] proposed $\text{RP}^*(G) + \text{pp}$, which first runs $\text{RP}^*(G)$ and then performs post-processing (pp) for the solution obtained by $\text{RP}^*(G)$ to recover the feasibility. We can modify $\text{RP}^*(G)$ using the same idea as $\text{CP}(G)$ and obtain $\text{CP}^*(G)$. Furthermore, by applying the same post-processing procedure pp, we obtain $\text{CP}^*(G) + \text{pp}$.

In $\text{CP}^*(G)$, we replace (1) with:

\[
\begin{align*}
&x_{i,j} + x_{j,k} - x_{i,k} \leq 1 \quad \forall (i, j, k) \in C^1_{>0}, \\
&x_{i,j} - x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in C^2_{>0}, \\
&-x_{i,j} + x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in C^3_{>0}, \\
&x_{i,j} \in \{0, 1\} \quad 1 \leq i < j \leq n,
\end{align*}
\]

where

\[
\begin{align*}
C^1_{>0} &= \{(i, j, k) \in T : w_{j,k} > 0\}, \\
C^2_{>0} &= \{(i, j, k) \in T : w_{i,k} > 0\}, \\
C^3_{>0} &= \{(i, j, k) \in T : w_{j,k} > 0\}.
\end{align*}
\]

The post-processing procedure pp is described as follows. Let $\check{x}^* = (\check{x}_{i,j})_{1 \leq i < j \leq n}$ denote an optimal solution of $\text{CP}^*(G)$. Let $\check{E}^*$ denote $\{(i, j) \in E \mid \check{x}_{i,j}^* = 1\}$, $E_{>0}$ denote $\{(i, j) \in E \mid w_{i,j} > 0\}$, and $\check{E}^*_E = \check{E}^* \cap E_{>0}$. Now let us consider a new graph $G_{\text{new}} = (V, \check{E}^*_E)$. We partition $V$ into a set of weakly connected components $\{(V_1, \check{E}^*_{1}), (V_2, \check{E}^*_{2}), \ldots, (V_p, \check{E}^*_{p})\}$ of $G_{\text{new}}$ by the depth-first search and outputs 0-1 vector $x^*$ that corresponds to partition $\{V_1, V_2, \ldots, V_p\}$, i.e., $x^*$, such that $x_{i,j}^* = 1$ if and only if $i, j \in V_\ell$ for some $\ell \in \{1, 2, \ldots, p\}$. Note that this procedure runs in time linear with the size of $G$.

The following lemma, which corresponds to Lemma 2 for $\text{CP}(G)$, holds for $\text{CP}^*(G)$. Here vertices $i$ and $j$ are positively reachable if a path exists from $i$ to $j$ that consists of edges in $E_{>0}$.

**Lemma 3** $(V_\ell, \check{E}^*_\ell)$ is complete for each $\ell \in \{1, \ldots, p\}$.

The proof is obtained from that of Lemma 2 by replacing ‘non-negatively’ with ‘positively’, $E_{>0}$ with $E_{>0}$, and $\text{CP}(G)$ with $\text{CP}^*(G)$.

Here we show that $\text{CP}^*(G) + \text{pp}$ returns an optimal solution to $P(G)$. This follows the proof in Subsection 4.1 in Miyachi et al. [13], which shows that $\text{RP}^*(G) + \text{pp}$ returns an optimal solution to $P(G)$. For the proof, it suffices to show that the objective value of $x^*$ remains identical as that of $\check{x}^*$ in $P(G)$ and $\text{CP}^*(G)$ because $x^*$ is feasible for $P(G)$ and $\check{x}^*$ is optimal to $\text{CP}^*(G)$, which is a relaxation of $P(G)$ (i.e., having fewer constraints). For convenience, we define $E^*_{in} = \{(i, j) \in E \mid x^*_{i,j} = 1\}$ and $E^*_{out} = E \setminus E^*_{in}$. Note that graph $(V, E^*_{in})$ consists of $p$ cliques $V_1, V_2, \ldots, V_p$.

We have the following lemmas.

**Lemma 4** It holds that

\[
\sum_{(i,j) \in E^*_{in}} w_{i,j}x^*_{i,j} = \sum_{(i,j) \in E^*_{in}} w_{i,j}\check{x}^*_{i,j}.
\]
Proof It suffices to show that for any $\ell \in \{1, 2, \ldots, p\}$, it holds that $\tilde{x}_{i,j}^* = 1$ for each $i, j \in V_\ell$ where $i < j$. Since $V_\ell$ is weakly connected by $\tilde{E}_{>0}$, a path exists on $\tilde{E}_{>0}$ that connects $i$ and $j$. This implies that every edge on it has a positive weight; $i$ and $j$ are positively reachable. According to Lemma 3, we conclude that $i$ and $j$ are directly reachable; $\tilde{x}_{i,j}^* = 1$.

Lemma 5 It holds that $\sum_{\{i, j\} \in E_{\text{out}}} w_{i,j} \tilde{x}_{i,j}^* \leq 0$.

Proof For each $\{i, j\} \in E_{\text{out}}$, we have $\{i, j\} \notin E_{>0}$. If otherwise, then $x_{i,j}^* = 1$ and thus $\{i, j\} \in E_{\text{in}}$. Therefore, for each $\{i, j\} \in E_{\text{out}}$, we have $\tilde{x}_{i,j}^* = 0$ or $w_{i,j} \leq 0$, which proves the lemma.

By Lemmas 4 and 5, we have $\sum_{\{i, j\} \in E} w_{i,j} \tilde{x}_{i,j}^* = \sum_{\{i, j\} \in E_{\text{in}}} w_{i,j} \tilde{x}_{i,j}^* \geq \sum_{\{i, j\} \in E_{\text{in}}} w_{i,j} \tilde{x}_{i,j}^* + \sum_{\{i, j\} \in E_{\text{out}}} w_{i,j} \tilde{x}_{i,j}^* = \sum_{\{i, j\} \in E} w_{i,j} \tilde{x}_{i,j}^*$. Thus, the following theorem holds.

Theorem 3 Any 0-1 vector returned by $CP^*(G)+pp$ is optimal to $P(G)$.

To estimate the ratio of $|CP^*(G)|$ to $|RP^*(G)|$, we define $b$ as the probability that an edge has a positive weight:

$b = \frac{|\{\{i, j\} \in E \mid i < j, w_{i,j} > 0\}|}{\binom{n}{2}}$.

Theorem 4 When the probability in which the weight of each edge is positive is independent for each edge, the ratio of the number of constraints in $CP^*(G)$ compared with that in $RP^*(G)$ can be estimated:

$\frac{|CP^*(G)|}{|RP^*(G)|} = \frac{1}{2 - b}$.

The proof is almost identical to Theorem 2. The ratio approaches $1/2$ when $b$ approaches 0, and it approaches 1 when $b$ approaches 1.

4.4 ACP(G)/ACP*(G)

As discussed in Section 4.1, either (i) or (ii) is required for transitivity constraints. We can adaptively switch between (i) or (ii) to minimize the number of transitivity constraints. Such a formulation, which introduces this idea to $CP(G)$, is called Adaptive $CP(G)$ (ACP(G)) and that for $CP^*(G)$ is called Adaptive $CP^*(G)$ (ACP*(G)).

Let $nn(i)$ denote the number of vertices $j \neq i$ that satisfy $w_{i,j} \geq 0$. In ACP(G), we replace (1) with:

$x_{i,j} + x_{j,k} - x_{i,k} \leq 1 \quad \forall (i, j, k) \in F^1_{\geq 0} \cup F^2_{\leq 0},$

$x_{i,j} - x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in F^3_{\geq 0} \cup F^4_{\leq 0},$

$-x_{i,j} + x_{j,k} + x_{i,k} \leq 1 \quad \forall (i, j, k) \in F^5_{\geq 0} \cup F^6_{\leq 0},$

$x_{i,j} \in \{0, 1\} \quad 1 \leq i < j \leq n,$
where
\[ F_{\geq 0}^1 = \{(i, j, k) \in T : w_{j,k} \geq 0 \land nn(i) > nn(k)\}, \]
\[ F_{\geq 0}^2 = \{(i, j, k) \in T : w_{i,j} \geq 0 \land nn(i) \leq nn(k)\}, \]
\[ F_{\geq 0}^3 = \{(i, j, k) \in T : w_{i,k} \geq 0 \land nn(j) > nn(k)\}, \]
\[ F_{\geq 0}^4 = \{(i, j, k) \in T : w_{i,j} \geq 0 \land nn(j) \leq nn(k)\}, \]
\[ F_{\geq 0}^5 = \{(i, j, k) \in T : w_{j,k} \geq 0 \land nn(i) > nn(j)\}, \]
\[ F_{\geq 0}^6 = \{(i, j, k) \in T : w_{i,k} \geq 0 \land nn(i) \leq nn(j)\}. \]

ACP*(G) is defined in the same manner based on CP*(G).

**Theorem 5** ACP(G) and P(G) have the same set of optimal solutions.

The proof resembles Theorem 1. By Theorem 1 [12] and this theorem, we obtain the following corollary.

**Corollary 2** ACP(G) and RP(G) have the same set of optimal solutions.

Compare the number of transitivity constraints in ACP(G) and CP(G). For given \( i, k \), s.t. \( i < k \), the number of transitivity constraints related to \( i, k \), i.e., the number of edges with non-negative weight between \( k \) and \( j \), s.t. \( j \neq i, j \neq k \), is given by binomial distribution \( B(n-2, a) \). The probability that the number of such edges equals \( y \) is given:

\[ p(y) = \binom{n-2}{y} a^y (1-a)^{n-2-y}. \]

The expected number of transitivity constraints related to \( i, k \) for CP(G) is given as \( \sum_{y=0}^{n-2} y \cdot p(y) = a(n-2) \). For ACP(G), we chose the smaller one from two values generated by binomial distribution \( B(n-2, a) \). The probability that the smaller one equals \( y \) is given:

\[ q(y) = (p(y))^2 + 2p(y) \sum_{z=y+1}^{n-2} p(z). \]

Then the expected number of transitivity constraints related to \( i, k \) for ACP(G) is given as \( \sum_{y=0}^{n-2} y \cdot q(y) \).

When \( n \) becomes large and the mean (which equals \( (n-2)a \)) and the variance (which equals \( (n-2)a(1-a) \)) are sufficiently large, we can approximate binomial distribution \( B(n-2, a) \) by normal distribution \( N((n-2)a, (n-2)a(1-a)) \). Then for each \( i, k \), the difference between the expected number of transitivity constraints for CP(G) and ACP(G) is given as \( \frac{1}{\sqrt{\pi}} \) of standard deviation \( \sqrt{(n-2)a(1-a)} \).

Thus, \( |CP(G)| - |ACP(G)| \) is approximately:

\[ \frac{(n-1)n \sqrt{(n-2)a(1-a)}}{2\sqrt{\pi}}. \]  \hspace{1cm} (2)

For ACP*(G), the following theorem holds.

**Theorem 6** Any 0-1 vector returned by ACP*(G) + pp is optimal to P(G).
We can prove this theorem based on the same reason as for Theorem 3. We can also estimate the reduction of the transitivity constraints in $\text{ACP}^*(G)$ against $\text{CP}^*(G)$ by replacing $a$ with $b$ in (2) for $\text{ACP}(G)$.

5 Experiments

We experimentally evaluated the performance of our new concise ILP formulations with Gurobi package version 9.0.3 with default parameters to solve all the ILP formulations. All problem instances were solved by a workstation equipped with an Intel Xeon Gold 6130 CPU @ 2.10GHz processor with 192GB RAM and Ubuntu 18.04.2 LTS. For each instance and formulation, we set the time limit to four hours.

5.1 Problem settings

We used for the experiments four types of problem settings, based on popular CPP application domains. The first three settings are based on real-world datasets from Miyauuchi et al. [13]. The last is based on synthetic (randomly generated) graphs, which were used as a benchmark for coalition structure generation algorithms [20].

(P) Correlation clustering: For instance $G = (V, E, w)$, $E = E_\rangle \cup E_\lessdot \cup E_0$ where $E_\rangle$, $E_\lessdot$ and $E_0$ are mutually disjointed. For each $\{i, j\} \in E$, we set $w_{i, j} = 1$ if $\{i, j\} \in E_\rangle$, $w_{i, j} = -1$ if $\{i, j\} \in E_\lessdot$, and $w_{i, j} = 0$ if $\{i, j\} \in E_0$. These instances were generated from a protein sequence dataset, http://www.paccanarolab.org/scps. The number of variables always equals $\left(\frac{n}{2}\right) = n(n - 1)/2$.

(G) Group technology: Instance $G = (V, E, w)$ is an undirected bipartite graph. $V$ is divided into two sets: $p$ parts and $q$ machines. For edge $\{i, j\} \in E$ between part $i$ and machine $j$, $w_{i, j} = 1$ if $i$ is proceeded by machine $j$ and $w_{i, j} = 0$ otherwise. These instances are generated based on a manufacturing cell formation dataset, http://mauricio.resende.info/data.

(C) Community detection: Instance $G = (V, E, w)$ is based on a bipartite modularity network. $V$ is divided into two sets, $V_R$ and $V_L$, so that each edge has one end point in $V_L$ and another in $V_R$. Edge $\{i, j\} \in E$ between $i \in V_L$ and $j \in V_R$ has weight $A_{ij} \frac{d_i d_j}{|E|^2}$, where $A_{ij}$ is the $(i, j)$ component of the adjacency matrix of $G$ and $d_i$ is the degree of $i \in N$.

(S) Coalition structure generation: We adopted the setting of the benchmark problem used in an existing work on coalition structure generation algorithms [20]. For each pair of nodes, an edge exists with probability 95%. With probability 5%, the edge weight is positive. Otherwise (i.e., with probability 95%), it is negative. A positive weight is chosen randomly from a range $[1, 100]$; a negative weight is chosen randomly from a range $[-100, -1]$.

Table 1 shows the ratio of the non-negative or positive edges for each instance.

5.2 Experimental results

We compared our four new concise ILP formulations against two existing ILP formulations: $\text{RP}(G)$ and $\text{RP}^*(G)$. Table 2 shows the number of transitivity constraints contained in the
Table 1  Details of instances used in experiments

| ID | Name            | n   | a         | b         |
|----|-----------------|-----|-----------|-----------|
| P1 | Protein 1       | 669 | 98.49%    | 4.22%     |
| P2 | Protein 2       | 587 | 98.21%    | 6.36%     |
| P3 | Protein 3       | 567 | 97.95%    | 6.39%     |
| P4 | Protein 4       | 654 | 98.87%    | 2.64%     |
| G9 | CH-8x20b        | 28  | 81.75%    | 24.07%    |
| G14| Mc-16x24        | 39  | 61.81%    | 11.47%    |
| G16| KI-16x43        | 59  | 67.15%    | 7.36%     |
| G17| Ca-18x24        | 42  | 60.05%    | 10.22%    |
| G18| MT-20x20        | 40  | 62.95%    | 14.23%    |
| G19| Ku-20x23        | 43  | 61.57%    | 12.51%    |
| G21| Bo-20x35        | 55  | 62.90%    | 10.03%    |
| G25| CH5-24x40       | 64  | 58.88%    | 6.50%     |
| G26| CH6-24x40       | 64  | 58.88%    | 6.50%     |
| G27| CH7-24x40       | 64  | 58.88%    | 6.50%     |
| G28| Mc-27x27        | 54  | 64.36%    | 15.30%    |
| G29| Ca-28x46        | 74  | 60.13%    | 7.81%     |
| G30| Ku-30x41        | 71  | 55.65%    | 5.15%     |
| G31| St-30x50-1      | 80  | 57.41%    | 4.87%     |
| G32| St-30x50-2      | 80  | 57.82%    | 5.28%     |
| G33| Ki-36x90        | 126 | 66.41%    | 4.23%     |
| G34| MC-37x53        | 90  | 75.43%    | 24.39%    |
| G35| Ch-40x100       | 140 | 63.21%    | 4.32%     |
| C1 | Wafa-CEO        | 41  | 63.54%    | 11.10%    |
| C2 | Divorces        | 59  | 85.33%    | 11.63%    |
| C3 | Hollywood movies| 102 | 56.79%    | 3.73%     |
| C4 | Scotland interlocks| 244 | 59.00%    | 1.21%     |
| C5 | Graph product   | 674 | 50.43%    | 0.27%     |
| C6 | Network science | 2,549 | 53.11% | 0.08% |
| S1 |                | 30  | 9.14%     | 5.03%     |
| S2 |                | 60  | 8.89%     | 4.87%     |
| S3 |                | 90  | 8.86%     | 4.85%     |
| S4 |                | 120 | 8.87%     | 4.86%     |

\[
a = \frac{|\{(i, j) \in E \mid i < j, w_{i,j} \geq 0\}|}{\binom{n}{2}}
\]

\[
b = \frac{|\{(i, j) \in E \mid i < j, w_{i,j} > 0\}|}{\binom{n}{2}}
\]

ILP formulations for P(G), RP(G), CP(G), and ACP(G), and Table 3 shows the number for RP*(G), CP*(G), and ACP*(G).

In the CP(G), ACP(G), CP*(G), and ACP*(G) columns, the pair of numbers in parentheses shows the ratio of the number of transitivity constraints. The first shows the actual
| ID  | P(G)       | RP(G)       | CP(G)       | ACP(G)       |
|-----|------------|-------------|-------------|--------------|
| P1  | 149,038,482| 148,995,305 | 146,928,630 | 146,043,257  |
| P2  | 100,614,735| 100,571,045 | 98,943,248  | 98,150,436   |
| P3  | 90,660,465 | 90,611,160  | 88,909,270  | 88,172,708   |
| P4  | 139,222,212| 139,199,019 | 137,814,213 | 137,092,183  |
| G14 | 27,417     | 23,366      | 17,919      | 15,502       |
| G16 | 97,527     | 84,297      | 73,040      | 65,929       |
| G17 | 34,440     | 28,920      | 21,585      | 19,090       |
| G18 | 29,640     | 25,661      | 18,640      | 17,713       |
| G19 | 37,023     | 31,606      | 23,228      | 21,465       |
| G21 | 78,705     | 67,196      | 53,392      | 44,554       |
| G25 | 124,992    | 102,860     | 80,159      | 66,221       |
| G26 | 124,992    | 102,875     | 80,185      | 66,375       |
| G27 | 124,992    | 102,887     | 80,147      | 66,463       |
| G28 | 74,412     | 65,114      | 47,504      | 45,826       |
| G29 | 194,472    | 161,752     | 125,506     | 105,382      |
| G30 | 171,465    | 137,420     | 101,470     | 88,240       |
| G31 | 246,480    | 199,476     | 154,625     | 127,212      |
| G32 | 246,480    | 200,359     | 155,769     | 128,038      |
| G33 | 842,520    | 716,019     | 634,888     | 481,179      |
| G34 | 352,440    | 326,247     | 272,067     | 241,138      |
| G35 | 1,342,740  | 1,121,684   | 959,479     | 736,143      |
| C1  | 31,980     | 27,466      | 18,651      | 18,281       |
| C2  | 97,527     | 93,475      | 78,292      | 76,540       |
| C3  | 515,162    | 409,432     | 261,430     | 257,942      |
| C4  | 7,176,508  | 5,484,694   | 3,907,530   | 3,494,964    |
| C5  | 152,410,272| 114,839,168 | 79,437,649  | 74,207,860   |
| C6  | 8,271,194,022| 6,335,107,844| 4,871,035,616| 3,912,482,999|
|     |            |             | (77%, 68%)  | (62%, 67%)   |
| S1  | 12,180     | 2,125       | 1,109       | 752 (35%, 35%)|
| S2  | 102,660    | 17,449      | 9,163       | 6,999 (40%, 40%)|
| S3  | 352,440    | 59,672      | 31,151      | 25,306 (42%, 42%)|
| S4  | 842,520    | 142,901     | 75,046      | 62,349 (44%, 44%)|

The second shows the theoretical ratio estimated by Theorems 2 and 4 or the approximation method presented in Section 4.4. These estimations are based on the ratios in Table 1. When the difference between the actual ratio and the theoretical one exceeds 10%, the larger one is shown in bold.

CP(G) contains slightly more transitivity constraints than the theoretical estimation for (G) as well as C4, C5 and C6. The ratio of CP(G) almost meets the theoretical estimation for...
| ID  | RP*(G) | CP*(G) | ACP*(G) |
|-----|--------|--------|---------|
| P1  | 12,106,852 | 4,753,162 (39%, 51%) | 3,230,495 (27%, 46%) |
| P2  | 12,191,742 | 5,227,235 (43%, 52%) | 3,777,089 (31%, 47%) |
| P3  | 10,995,760 | 4,726,615 (43%, 52%) | 3,235,986 (29%, 47%) |
| P4  | 7,161,805  | 2,720,318 (38%, 51%) | 1,790,304 (25%, 44%) |
| G14 | 5,902    | 2,830 (48%, 53%) | 2,240 (38%, 39%) |
| G16 | 13,560   | 5,441 (40%, 52%) | 4,385 (32%, 38%) |
| G17 | 6,640    | 3,129 (47%, 53%) | 2,438 (37%, 39%) |
| G18 | 7,839    | 4,200 (54%, 54%) | 3,273 (42%, 42%) |
| G19 | 8,646    | 4,375 (51%, 53%) | 3,341 (39%, 41%) |
| G21 | 14,938   | 6,537 (44%, 53%) | 5,906 (40%, 40%) |
| G25 | 15,750   | 7,007 (44%, 52%) | 6,237 (40%, 38%) |
| G26 | 15,765   | 7,033 (45%, 52%) | 6,447 (41%, 38%) |
| G27 | 15,777   | 6,995 (44%, 52%) | 6,556 (42%, 38%) |
| G28 | 21,044   | 11,000 (52%, 54%) | 9,322 (44%, 44%) |
| G29 | 28,840   | 12,178 (42%, 52%) | 10,508 (36%, 40%) |
| G30 | 17,222   | 8,110 (47%, 51%) | 6,403 (37%, 36%) |
| G31 | 23,508   | 10,145 (43%, 51%) | 9,564 (41%, 37%) |
| G32 | 25,405   | 11,289 (44%, 51%) | 10,011 (39%, 37%) |
| G33 | 68,435   | 29,968 (44%, 51%) | 18,650 (27%, 37%) |
| G34 | 146,067  | 76,507 (52%, 57%) | 63,198 (43%, 51%) |
| G35 | 112,904  | 48,739 (43%, 51%) | 40,149 (36%, 39%) |
| C1  | 6,640    | 4,026 (61%, 53%) | 2,590 (39%, 39%) |
| C2  | 20,116   | 15,640 (78%, 53%) | 5,729 (28%, 41%) |
| C3  | 37,532   | 21,548 (57%, 51%) | 12,322 (33%, 36%) |
| C4  | 174,246  | 82,062 (47%, 50%) | 58,906 (34%, 34%) |
| C5  | 822,272  | 390,337 (47%, 50%) | 274,947 (33%, 29%) |
| C6  | 13,130,379 | 5,386,394 (41%, 50%) | 4,160,816 (32%, 30%) |
| S1  | 1,196    | 610 (51%, 51%) | 348 (29%, 28%) |
| S2  | 9,752    | 5,010 (51%, 51%) | 3,409 (35%, 34%) |
| S3  | 33,330   | 17,011 (51%, 51%) | 12,640 (38%, 38%) |
| S4  | 79,111   | 41,014 (51%, 51%) | 31,600 (40%, 39%) |

The ratio of ACP*(G) almost meets the theoretical estimation for all the instances. The maximum difference between the actual and theoretical ratios is five percent. Note that ACP*(G) eliminates more transitivity constraints than CP*(G) for all the instances.

CP*(G) has almost the same or fewer constraints than the theoretical estimation for all the instances except for C2 and C3. ACP*(G) significantly reduces the redundant transitivity constraints compared to RP*(G) (by more than 60% in most instances). Similar to CP(G), the ratio of CP*(G) is close to the theoretical estimation for synthetic graphs (S). Note that ACP*(G) eliminates more transitivity constraints than CP*(G) for all the instances.
We obtained our theoretical estimation under the assumption that the sign of each edge (positive or negative) is determined independently. For real-world datasets, some correlations among edge signs likely exist. Thus, the actual reduction rate and our estimation do not exactly coincide except for synthetic graphs.

Table 4 shows the computation times in seconds. For the RP\(^*(G)\), CP\(^*(G)\), and ACP\(^*(G)\) results, we included the computational times for running the post-processing pp. The pp time was negligible in our experiments. In most cases it was less than 0.05 seconds. The

| ID | P(G) | RP(G) | CP(G) | ACP(G) | RP\(^*(G)\) +pp | CP\(^*(G)\) +pp | ACP\(^*(G)\) +pp |
|----|------|------|------|-------|-----------------|----------------|-----------------|
| P1 | OM   | OM   | OM   | OM    | 5,901.68       | 2,593.14       | 782.37          |
| P2 | OM   | OM   | OM   | OM    | (17.5%)        | (32.0%)        | 3,883.25        |
| P3 | OM   | OM   | OM   | OM    | (39.9%)        | (0.3%)         | (0.2%)          |
| P4 | OM   | OM   | OM   | OM    | (0.2%)         | (0.04%)        | (0.1%)          |
| G14 | 7.83 | 6.17 | 4.57 | 4.66  | 2.03            | 1.10           | 1.39            |
| G16 | 46.47| 41.58| 41.69| 23.89 | 5.59           | 1.15           | 1.29            |
| G17 | 2.74 | 2.28 | 2.01 | 1.3   | 0.43           | 0.77           | 0.87            |
| G18 | 501.40| 263.29| 110.17| 560.29| 98.71          | 286.02         | 598.89          |
| G19 | 19.44| 16.59| 12.93| 11.23 | 5.54           | 15.75           | 31.57           |
| G21 | 36.19| 35.33| 20.22| 19.09 | 5.78           | 9.47            | 4.85            |
| G25 | 39.01| 18.83| 13.07| 16.01 | 4.76           | 9.08            | 11.24           |
| G26 | 67.81| 123.53| 71.08| 46.07 | 9.11           | 68.45           | 70.91           |
| G27 | 260.04| 443.71| 308.67| 124.69| 27.04          | 212.21         | 232.55          |
| G28 | 483.03| 448.65| 364.75| 307.44| 138.01         | 1,998.26       | 1,289.23        |
| G29 | 5,458.76| 5,083.06| 4,296.99| 3,540.61| 192.73         | 444.61         | 430.4           |
| G30 | 5.94 | 4.95 | 6.67 | 5.31  | 1.38           | 0.92            | 0.81            |
| G31 | 36.34| 29.47| 12.7 | 18.95 | 1.90           | 4.59            | 4.33            |
| G32 | 153.42| 123.19| 100.27| 75.17 | 15.33          | 102.86         | 165.95          |
| G33 | (43.0%)| (29.7%)| (80.9%)| (31.0%)| (3.1%)         | 2,724.45       | 9,200.60        |
| G34 | 5,195.02| 4,361.58| 3,195.23| 2,910.43| 676.13         | 1,456.07       | 4,397.69        |
| G35 | 10.20| 8.94 | 7.61 | 5.49  | 0.66           | 0.22            | 0.20            |
| C1  | 9.34 | 8.64 | 5.97 | 5.43  | 1.88           | 4.67            | 1.42            |
| C2  | 378.34| 496.61| 440.59| 313.54| 21.32          | 234.93         | 1.62            |
| C3  | (1.9%)| (2.0%)| (1.6%)| (1.4%)| 13,042.87      | (2.6%)         | 6,112.13        |
| C4  | (8.4%)| (8.2%)| (8.4%)| (8.4%)| (0.4%)         | (0.8%)         | (0.5%)          |
| C5  | OM   | OM   | OM   | OM    | 12.95          | 5.21            | 3.02            |
| C6  | OM   | OM   | OM   | OM    | 298.58         | 175.85         | 72.02           |
| S1  | 0.11 | 0.03 | 0.02 | 0.02  | 0.02           | 0.02            | 0.01            |
| S2  | 1.68 | 0.44 | 0.27 | 0.22  | 0.30           | 0.19            | 0.15            |
| S3  | 18.55| 7.22 | 3.72 | 2.99  | 4.82           | 2.51            | 1.98            |
| S4  | 1,778.85| 520.60| 287.61| 239.49| 360.92         | 205.76         | 134.51          |
longest pp time was 0.15 seconds for C6, which has the most nodes (i.e., 2,549 nodes) in our experiments.

Bold numbers indicate the ILP formulation that obtained the best result. OM indicates that we failed to solve the instance due to insufficient memory, which occurs when the problem instance contains more than tens of millions of constraints. We show the relative gaps for problem instances that could be stored on the machine but failed to obtain an optimal solution within the time limit. More specifically, Gurobi returns an upper bound \((UB)\) and a lower bound \((LB)\) of the optimal value when terminated before reaching an optimal solution. We show \((UB - LB)/LB\). For each instance, the best computation time (or the relative gap) among the formulations is highlighted in bold.

ACP\(^\ast\)(G) successfully solved the most instances, i.e., 24 out of 27 in the (P), (G), and (C) categories, among the six formulations; RP(G), CP(G) and ACP(G) solved 18 instances, and RP\(^\ast\)(G) and CP\(^\ast\)(G) solved 22. There were no instances that ACP\(^\ast\)(G) could not solve that the other formulations could. Thus, ACP\(^\ast\)(G) is the best formulation with respect to the number of solved instances.

For the computation time, RP\(^\ast\)(G) works well on the (G) instances, except G33, even though CP\(^\ast\)(G) and ACP\(^\ast\)(G) have fewer transitivity constraints. On the other instance categories of (P), (C), and (S), ACP\(^\ast\)(G) shows the best performance.

As shown in our example, although reducing redundant constraints effectively shortened for the computational times in general, in some cases (mostly in (G) instances), our new formulations required more computational time than the existing methods, even though our formulations have fewer constraints. Our current conjecture is that even though some constraints are redundant, they can still be useful for constraint propagation and reducing the computational times.

6 Conclusions

We investigated the ILP formulations for CPP. Based on the existing ILP formulations, we developed a series of new ILP formulations that can significantly reduce transitivity constraints. We characterized the transitivity constraints based on their reachability among vertices and identified the redundant parts in the existing ILP formulations. All our ILP formulations are guaranteed to obtain optimal clique partitions. Experimental results showed that our new ILP formulations outperformed the existing ILP formulations for most instances. Transitivity constraints are ubiquitous in a variety of problems and their possible formulations. For CPP, we can use MaxSAT encoding instead of an ILP formulation. We would like to apply our idea to reduce transitivity constraints for MaxSAT encoding of CPP, as well as other discrete optimization problems, e.g., [10, 18, 21].

Declarations

Conflict of Interests The authors declare that they have no conflict of interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit http://creativecommons.org/licenses/by/4.0/.
References

1. Bansal, N., Blum, A., & Chawla, S. (2004). Correlation clustering. *Machine Learning*, 56(1-3), 89–113.
2. Barber, M. J. (2007). Modularity and community detection in bipartite networks. *Physical Review E*, 76, 066102.
3. Benati, S., Puerto, J., & Rodríguez-chía, A. M. (2017). Clustering data that are graph connected. *European Journal of Operational Research*, 261(1), 43–53.
4. Brandes, U., Delling, D., Gaertler, M., Görke, R., Hoefer, M., Nikoloski, Z., & Wagner, D. (2008). On modularity clustering. *IEEE Transactions on Knowledge and Data Engineering*, 20(2), 172–188.
5. Bruckner, S., Höffner, F., Komusiewicz, C., & Niedermeier, R (2013). Evaluation of ILP-based approaches for partitioning into colorful components. In *Proceedings of 12th International Symposium on Experimental Algorithms (SEA 2013)* (pp. 176–187).
6. Costa, A. (2015). MILP Formulations for the modularity density maximization problem. *European Journal of Operational Research*, 245(1), 14–21.
7. Dinh, T. N., & Thai, M. T. (2015). Toward optimal community detection: From trees to general weighted networks. *Internet Mathematics*, 11(3), 181–200.
8. Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486(3), 75–174.
9. Grötschel, M., & Wakabayashi, Y. (1989). A cutting plane algorithm for a clustering problem. *Mathematical Programming*, 45(1-3), 59–96.
10. Liao, X., Koshimura, M., Nomoto, K., Ueda, S., Sakurai, Y., & Yokoo, M. (2019). Improved WPM encoding for coalition structure generation under mc-nets. *Constraints*, 24(1), 25–55.
11. Miyauchi, A., & Miyamoto, Y. (2013). Computing an upper bound of modularity. *European Physical Journal B*, 86, 302.
12. Miyauchi, A., & Suzuki N. (2015). Redundant constraints in the standard formulation for the clique partitioning problem. *Optimization Letters*, 9(1), 199–207.
13. Miyauchi, A., Sonobe, T., & Suzuki N. (2018). Exact clustering via integer programming and maximum satisfiability. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence (AAAI-18)* (pp. 1387–1394).
14. Nogueira, L. L. H., Quiles, M. G., & Lorena L. A. N. (2019a). Improving the performance of an integer linear programming community detection algorithm through clique filtering. In *Proceedings of 19th International Conference on Computational Science and Its Applications (ICCSA 2019)* (pp. 757–769).
15. Nogueira, L. L. H., Quiles, M. G., Lorena, L. A. N., de Carvalho, A., & Cespedes J. G. (2019b). Qualitative data clustering: a new integer linear programming model. In *Proceedings of 2019 International Joint Conference on Neural Networks (IJCNN 2019)* (pp. 1–8).
16. Oosten, M., Rutten, J. H. G. C., & Spieksma F. C. R. (2001). The clique partitioning problem. *Networks*, 38(4), 209–226.
17. Rahwan, T., michalak, T. P., Wooldridge, M., & Jennings, N. R. (2015). Coalition structure generation: a survey. *Artificial Intelligence*, 229, 139–174.
18. Soh, T., Berre, D. L., Roussel, S., Banbara, M., & Tamura, N. (2014). Incremental SAT-based method with native boolean cardinality handling for the hamiltonian cycle problem. In *Proceedings of the 14th European conference on logics in artificial intelligence (JELIA 2014)* (pp. 684–693).
19. Wang, H., Alidacae, B., Glover, F., & Kochenberger, G. (2006). Solving group technology problems via clique partitioning. *International Journal of Flexible Manufacturing Systems*, 18, 77–97.
20. Watanabe, E., Koshimura, M., Sakurai, Y., & Yokoo, M. (2019). Solving coalition structure generation problems over weighted graph. In *Proceedings of the 22nd International Conference on Principles and Practice of Multi-Agent Systems (PRIMA-19)* (pp. 338–353).
21. Zha, A., Nomoto, K., Ueda, S., Koshimura, M., Sakurai, Y., & Yokoo, M. (2017). Coalition structure generation for partition function games utilizing a concise graphical representation. In *Proceedings of 20th International Conference on Principles and Practice of Multi-Agent Systems (PRIMA 2017)* (pp. 143–159).

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.