Comput Manag Sci (2016) 13:571–596
DOI 10.1007/s10287-016-0260-7

ORIGINAL PAPER

On the minimum-cost λ-edge-connected k-subgraph problem

Elham Sadeghi[1] · **Neng Fan**[1]

Received: 23 September 2015 / Accepted: 23 May 2016 / Published online: 21 June 2016

Abstract In this paper, we propose several integer programming (IP) formulations to exactly solve the minimum-cost λ-edge-connected k-subgraph problem, or the (k, λ)-subgraph problem, based on its graph properties. Special cases of this problem include the well-known k-minimum spanning tree problem (if $\lambda = 1$), λ-edge-connected spanning subgraph problem (if $k = |V|$) and k-clique problem (if $\lambda = k - 1$ and there are exact k vertices in the subgraph). As a generalization of k-minimum spanning tree and a case of the (k, λ)-subgraph problem, the $(k, 2)$-subgraph problem is studied, and some special graph properties are proved to find stronger and more compact IP formulations. Additionally, we study the valid inequalities for these IP formulations. Numerical experiments are performed to compare proposed IP formulations and inequalities.

Keywords Edge-connected subgraph · Minimum spanning tree · Integer programming · Graph connectivity · Valid inequalities

1 Introduction

Given an undirected graph $G = (V, E)$ consisting of vertex set V and edge set E, with nonnegative edge costs $c : E \to \mathbb{R}^+$ and an integer k, the k-minimum spanning tree, or k-MST, of G is a tree with minimum cost that spans at least k vertices of G. A k-MST does not have to be a subgraph of the minimum spanning tree (MST) of G. A similar problem to k-MST is the edge-weighted k-cardinality tree (KCT) problem,

✉ Neng Fan
nfan@email.arizona.edu

[1] Department of Systems and Industrial Engineering, University of Arizona, Tucson, AZ 85721, USA

which finds the minimum-cost tree of G with exactly k edges. On the other hand, the λ-edge-connected spanning subgraph problem is to find a subgraph $H = (V, E')$ ($E' \subseteq E$) of G with minimum cost such that H is λ-edge-connected, or removal of any $\lambda - 1$ edges from H leaves a connected graph.

As a generalization of the k-MST and λ-edge-connected spanning subgraph problems, the *minimum-cost λ-edge-connected k-subgraph problem*, or the (k, λ)-*subgraph problem*, is to find a minimum-cost λ-edge-connected subgraph of G with at least k vertices. Obviously, k-MST problem is equivalent to $(k, 1)$-subgraph problem; and λ-edge-connected spanning subgraph problem is a $(|V|, \lambda)$-subgraph problem. Additionally, the k-clique problem is to find a subgraph of k vertices in G with minimum cost such that there is an edge between any two vertices in the subgraph, and thus this problem is the $(k, k-1)$-subgraph problem with additional restriction to require *exact* k vertices.

The k-MST problem has been widely studied in history, and is shown to be NP-hard by reducing from the Steiner tree problem (see Karp 1972). There are many constant factor approximations for this problem, and the current best approximation is a 2-approximation by Garg (2005). For the similar KCT problem, some exact algorithms based on integer programming (IP) approaches were introduced in Chimani and Kandyba (2009), Quintpo et al. (2010), Simonetti et al. (2011) and Simonetti et al. (2013). For the k-clique problem, it is NP-hard (see Garey and Johnson 1979).

The λ-edge-connected spanning subgraph problem is also NP-hard (see Garey and Johnson 1979). There are numerous approximation algorithms in the literature for finding λ-edge-connected subgraphs, such as 2-approximation algorithm for 2-edge-connected subgraphs Frederickson and Ja'Ja' (1981), matroid-based 2-approximating Khuller and Vishkin (1994), 2-approximation Jain (2001), $\frac{3}{2}$-appro-ximation when multiple edges can be allowed and λ is even, and $(\frac{3}{2} + \frac{1}{2\lambda})$-approximation when λ is odd Goemans and Bertsimas (1993), $(1 + \frac{2}{\lambda})$-approximation algorithm when all edges have unit costs Gabow et al. (2009). Additionally, Magnanti and Raghavan (2005) considered two models for exactly solving λ-edge-connected subgraph problem, one of which is a cutset model and the other one based on a multicommodity flow. Cai and Sun (1989) obtained characterizations and algorithms for augmentation of any graph to be a λ-edge-connected graph. Bienstock et al. (1990) derived conditions on the class of minimum-weight λ-edge connected networks, where the distances between points satisfy the triangle inequality. In Bendali et al. (2007), several valid inequalities for λ-edge-connected subgraph were proposed by using branch-and-cut algorithm. For a similar problem to find the maximal λ-edge-connected subgraphs with most number of vertices, three approaches by vertex reduction, edge reduction and cut pruning were proposed in Zhou et al. (2012). As a special case, the 2-edge-connected subgraph problem has been studied in many references, for example, Sun (2013) presented an extensive study, while Fortz et al. (2006) studied the polyhedral results and proposed a branch-and-cut algorithm for 2-edge-connected subgraphs with bounded rings.

The (k, λ)-subgraph problem was recently proposed by Lau et al. (2009). They proposed an $O(log^2 n)$-approximation algorithm, where n is the number of vertices in G, for solving the $(k, 2)$-subgraph problem, and showed that the (k, λ)-subgraph problem in general is NP-hard. In Safari and Salavatipour (2008), Safari and Salavatipour

obtained an $O(1)$-approximation algorithm for (k, λ)-subgraph in a graph with metric costs. For the special $(k, 2)$-subgraph problem, Chekuri and Korula (2008) obtained an $O(\log n \cdot \log k)$-approximation. However, none of these approaches proposed for solving k-MST, λ-edge-connected subgraph or k-clique problems can be generalized directly for exactly solving the (k, λ)-subgraph problem.

In this paper, we propose several IP formulations for exactly solving the (k, λ)-subgraph problem, based on some graph properties, for example, requirements of cutsets for a division of the graph and paths between any two vertices. After that, we study four types of valid inequalities for this problem. Addtionally, we study the properties of $(k, 2)$-subgraphs, such as connectivity, bridgeless, and strong orientation properties. Based on these properties, we propose several stronger and more compact IP formulations for solving the $(k, 2)$-subgraph problem, which is a direct generalization of the k-MST problem.

The remainder of this paper is organized as follows. Section 2 introduces several IP formulations and algorithms for solving the (k, λ)-subgraph problem and introduces strong valid inequalities. Section 3 considers a special case of this problem, the $(k, 2)$-subgraph problem, and some stronger and more compact IP formulations are proposed. In Sect. 4, we perform some numerical experiments based on these IP formulations and inequalities, and compare their computational complexities on some randomly generated graphs. Finally, Sect. 5 concludes this paper with some further research directions.

2 Integer programming formulations for the (k, λ)-subgraph problem

In an undirected graph $G = (V, E)$, with nonnegative edge costs $c : E \to \mathbb{R}^+$ and two given integers k, λ, the (k, λ)-subgraph problem is to find a minimum-cost λ-edge-connected subgraph of G with at least k vertices. Assume that graph G has n vertices (i.e., $|V| = n$), and the nonnegative cost of edge $(i, j) \in E$ is c_{ij}.

Let $x_i \in \{0, 1\}$ be the decision variable associated with each vertex $i \in V$ such that $x_i = 1$ if i is selected into the (k, λ)-subgraph and $x_i = 0$ otherwise. Similarly, let $y_{ij} \in \{0, 1\}$ be the decision variable associated with each edge $(i, j) \in E$ such that $y_{ij} = 1$ if edge (i, j) is selected into the (k, λ)-subgraph and $y_{ij} = 0$ otherwise. In the following, x and y denote the vectors form by x_i for all $i \in V$ and y_{ij} for all $(i, j) \in E$, respectively.

Therefore, the (k, λ)-subgraph can be denoted by $G' = (V', E')$ with $V' = \{i \in V : x_i = 1\}$ and $E' = \{(i, j) \in E : y_{ij} = 1\}$. To ensure this subgraph has at least k vertices, the constraint $\sum_{i \in V} x_i \geq k$ should be added. However, to ensure the G' is λ-edge connected, some other constraints will be needed. In the following, we present different IP formulations to ensure the edge-connectivity of G'. First, we state the theorem by Menger (1927):

Lemma 1 *For any pair with distinct vertices s, t $(s \neq t)$ in a graph G, the cardinality of the minimum cutset for s and t is equal to the maximum number of edge-disjoint paths between s and t, where the minimum cutset is a minimum set of edges whose removal disconnects s and t.*

For a nonempty subset $S \subset V$ and a corresponding division $(S, V\backslash S)$ of V, any edge in the cutset by this devision has one edge in S and the other one in $V\backslash S$.

2.1 Cutset formulation and cutting plane algorithm

Based on Lemma 1, to ensure that the subgraph G' is λ-edge-connected, there should be at least λ edges in any cutset of V'. Therefore, the following IP formulation can be used to find the minimum (k, λ)-subgraph of G:

$$\min \sum_{(i,j)\in E} c_{ij} y_{ij} \tag{1a}$$

$$s.t. \sum_{i\in V} x_i \geq k \tag{1b}$$

$$\sum_{i\in S, j\in V\backslash S} y_{ij} \geq \lambda(x_i + x_j - 1), \forall S \subset V, S \neq \emptyset, i \in S, j \in V\backslash S \tag{1c}$$

$$y_{ij} \leq x_i, y_{ij} \leq x_j, \ \forall (i, j) \in E \tag{1d}$$

$$y_{ij} \in \{0, 1\}, \ \forall (i, j) \in E; \ x_i \in \{0, 1\}, \forall i \in V \tag{1e}$$

The objective (1a) is to minimize the cost associated with edges chosen into the (k, λ)-subgraph. The constraint (1b) is to ensure that there are at least k vertices. To ensure that the subgraph $G' = (V', E')$ chosen by x and y is λ-edge connected, by Lemma 1, the constraints

$$\sum_{i\in S, j\in V\backslash S} y_{ij} \geq \lambda, \forall S \subset V', S \neq \emptyset \tag{2}$$

should be satisfied for the cutset formed by the division $(S, V'\backslash S)$. Since the set V' is decided by x, we use constraints (1c) to ensure that (2) are satisfied. In fact, this can be proved through five cases of (1c) depending relations between S and V' and cases for $i \in S$, $j \in V\backslash S$:

(i) $S \subset V', i \in S$. If $j \in V'\backslash S$, then $x_i = x_j = 1$, and (1c) becomes $\sum_{i\in S, j\in V\backslash S} y_{ij} \geq \lambda$; otherwise, if $j \in V\backslash V'$, then $x_i = 1, x_j = 0$ and $\sum_{i\in S, j\in V\backslash S} y_{ij} \geq 0$, which is redundant.

(ii) $S \cap V' \neq \emptyset$, $S \cap (V\backslash V') \neq \emptyset$. For $i \in S$, $j \in V\backslash S$, there are four sub-cases:
- $i \notin V', j \in V'$. Then $x_i = 0, x_j = 1, \sum_{i\in S, j\in V\backslash S} y_{ij} \geq 0$;
- $i \notin V', j \in V\backslash V'$. Then $x_i = x_j = 0, \sum_{i\in S, j\in V\backslash S} y_{ij} \geq -\lambda$;
- $i \in V', j \in V'$. Then $x_i = x_j = 1, \sum_{i\in S, j\in V\backslash S} y_{ij} \geq \lambda$;
- $i \in V', j \in V\backslash V'$. Then $x_i = 1, x_j = 0, \sum_{i\in S, j\in V\backslash S} y_{ij} \geq 0$.

Except the 3rd sub-case, all of them are redundant.

(iii) $S = V'$. Then $x_i = 1$ for $i \in S$ and $x_j = 0$ for $j \in V\backslash S$. Thus, $\sum_{i\in S, j\in V\backslash S} y_{ij} \geq 0$, which is redundant.

(iv) $S \subset (V \backslash V')$. Then if $j \in V'$, $x_i = 0, x_j = 1$ and $\sum_{i \in S, j \in V \backslash S} y_{ij} \geq 0$; otherwise, if $j \in V \backslash (V' \bigcup S)$, $x_i = 0, x_j = 0$ and $\sum_{i \in S, j \in V \backslash S} y_{ij} \geq -\lambda$. Both are redundant.

(v) $S \supset V'$. If $i \in V'$, then $x_i = 1, x_j = 0$ and $\sum_{i \in S, j \in V \backslash S} y_{ij} \geq 0$; otherwise, if $i \notin V'$, then $x_i = x_j = 0$ and $\sum_{i \in S, j \in V \backslash S} y_{ij} \geq -\lambda$. Both are redundant.

Therefore, the constraints (2) are enforced in two sub-cases within case (i) and case (ii) under the inequality (1c). The set of constraints (1d) ensures that the chosen edges must connect two chosen vertices in the subgraph, while the last set of constraints (1e) is the binary requirements.

This cutset formulation is a direct formulation to find a minimum-cost (k, λ)-subgraph in G. However, as discussed above, there are too many redundant cases in (1c). In the worse case, the number of constraints for all possible subsets S of V in (1c) is exponential, and therefore, it cannot handle a problem in a large graph. Next, we discuss a cutting plane algorithm (**Cutset**) based on this formulation to solve the (k, λ)-subgraph problem.

Algorithm *Cutting Plane Algorithm for* (k, λ)*-subgraph problem* (**Cutset**)

Input: A graph $G = (V, E)$ with cost matrix $(c_{ij})_{n \times n}$, and two integers k, λ
Output: minimum-cost (k, λ)-subgraph of G
1: $t \leftarrow 0$
2: solve the problem consisting of (1a),(1b),(1d),(1e)
3: obtain optimal solutions x^t, y^t and define $V' := \{i \in V : x_i^t = 1\}$
4: **for** a division $(S, V' \backslash S)$ of V' $(S \subset V', S \neq \emptyset)$
5: **if** $\sum_{i \in S, j \in V' \backslash S} y_{ij}^t \geq \lambda$, **goto** Step 4 and find a distinct division
6: **else if** $\sum_{i \in S, j \in V' \backslash S} y_{ij}^t < \lambda$
7: add cut $\sum_{i \in S, j \in V' \backslash S} y_{ij} \geq \lambda$ to the problem in Step 2
8: **goto** Step 2, $t \leftarrow t + 1$
9: **end if**
10: **end for**
11: **if** no cut added in Steps 4-10, **exit**; x^t, y^t is optimal.

2.2 Path formulation

For any edge $(i, j) \in E$, assume it has a direction from i to j. Let E' denote set of edges with direction from j to i for any $(i, j) \in E$. Let $z_{ij}^{(st)l}$ denote the flow on edge $(i, j) \in E \cup E'$ of the lth type from s to t $(l = 1, \ldots, \lambda)$. Therefore, each type of flow from s to t ensures that there is a path from s to t.

From Lemma 1, a graph is λ-edge-connected if and only if for any pair s, t $(s \neq t)$ of vertices, there exist at least λ edge-disjoint paths from s to t. Now, we present a formulation to ensure that there are λ edge-disjoint paths for any pair of vertices in V', which instead has polynomial number of constraints.

$$\min \sum_{(i,j) \in E} c_{ij} y_{ij} \tag{3a}$$

$$s.t. \sum_{i \in V} x_i \geq k \tag{3b}$$

$$\sum_{j:(i,j)\in E\cup E'} z_{ij}^{(st)l} - \sum_{j:(j,i)\in E\cup E'} z_{ji}^{(st)l} = \begin{cases} x_s x_t, & i = s \\ -x_s x_t, & i = t \\ 0, & i \in V\backslash\{s,t\} \end{cases} \quad \forall s,t \in V, l = 1,\ldots,\lambda \tag{3c}$$

$$\sum_{l=1}^{\lambda} (z_{ij}^{(st)l} + z_{ji}^{(st)l}) \le y_{ij}, \ \forall s,t \in V(s \ne t), \forall (i,j) \in E \tag{3d}$$

$$0 \le z_{ij}^{(st)l} \le x_s, z_{ij}^{(st)l} \le x_t, \ \forall (i,j) \in E \cup E', \ \forall s,t \in V(s \ne t), l = 1,\ldots,\lambda \tag{3e}$$

$$y_{ij} \le x_i, y_{ij} \le x_j, \ \forall (i,j) \in E \tag{3f}$$

$$y_{ij} \in \{0,1\}, \ \forall (i,j) \in E; \ x_i \in \{0,1\}, \forall i \in V \tag{3g}$$

The objective (3a) and constraints (3b), (3g) are the same as those in (1). Constraints (3c) are included to ensure that there are λ paths from s to t if both vertices s, t are chosen into the subgraph, by sending λ types of flows from s to t (one unit for each type), and otherwise all $z_{ij}^{(st)l} = 0$. In fact, they are guaranteed through the following cases for pairs s, t:

(i) $s, t \in V'$, i.e., $x_t = x_s = 1$. By (3c)–(3e), we guarantee the existence of λ edge-disjoint paths between s, t;
(ii) $s \in V', t \notin V'$, i.e., $x_s = 1, x_t = 0$. $z_{ij}^{(st)l} = 0, \quad \forall l = 1,\ldots,\lambda$;
(iii) $s \notin V', t \in V'$, i.e., $x_s = 0, x_t = 1$. $z_{ij}^{(st)l} = 0, \quad \forall l = 1,\ldots,\lambda$; and
(iv) $s \notin V', t \notin V'$, i.e., $x_s = x_t = 0$. $z_{ij}^{(st)l} = 0, \quad \forall l = 1,\ldots,\lambda$,

where the case (i) ensures that for any chosen pair s, t within V', there are at least λ edge-disjoint paths, and cases (ii–iv) are redundant.

By constraints (3d), (3f), we guarantee that no edge outside of the V' is used to find edge-disjoint paths. An edge, chosen into G', can be used in most one path for sending flows, and this is ensured by constraints (3d)–(3e). Additionally, the constraints in (3d) ensures that all l paths from s to t are edge-disjoint.

The most complex constraints (3c) are required for all pairs of vertices in V, and there are $n(n-1)/2$ constraints. Comparing with the cutset formulations (1), this formulation, based on edge-disjoint paths, is compact. However, as shown in Lau et al. (2009) and Safari and Salavatipour (2008), the (k,λ)-subgraph is still very complex for large graphs, even by approximate methods.

The formulation (3), if $k = |V|$, reduces to the formulation introduced in Magnanti and Raghavan (2005) for the λ-edge-connected spanning subgraph problem. If $k = |V|, \lambda = 1$, this reduces to a formulation for the minimum spanning tree problem; and if $\lambda = 1$, this presents a formulation for the k-MST problem. If $\lambda = 2$, we consider a special case of this problem, $(k,2)$-subgraph problem.

The formulation (3) has nonlinear terms in the constraint (3c). We can linearize them by introducing d_{st} [for all $s, t \in V(s \ne t)$] and adding following four constraints:

$$d_{st} \leq x_s,\ d_{st} \leq x_t,\ d_{st} \geq x_s + x_t - 1,\ d_{st} \geq 0. \tag{4}$$

2.3 Stronger valid inequalities for IP formulations of (k, λ)-subgraph

An immediate result of Lemma 1 can be presented in the following theorem.

Theorem 1 *For any (k, λ)-subgraph G' of G, the degree of each vertex in the subgraph is at least λ, i.e., the choice of G' by x, y should satisfy*

$$\sum_{j:j\in V} y_{ij} \geq \lambda x_i \tag{5}$$

Theorem 2 *The number of edges in a (k, λ)-subgraph is at least $\frac{1}{2}k\lambda$, i.e., the inequality is a valid inequality for IP formulations (1), (3) to find the minimum-cost (k, λ)-subgraph,*

$$\sum_{i:i\in V} \sum_{j:j\in V} y_{ij} \geq \frac{1}{2}k\lambda. \tag{6}$$

More specially, for $\lambda = 1$, $\sum_{i:i\in V} \sum_{j:j\in V} y_{ij} \geq k - 1$.

Proof From Theorem 1, the degree for each vertex in the (k, λ)-subgraph is at least λ. The total number of edges in the subgraph is at least $\frac{1}{2}k\lambda$, as each edge contributes degree 2 in its two ends. For $\lambda = 1$, the problem has reduced to k-MST problem, which at least $k - 1$ edges in the tree. □

For corresponding values of k, λ, the valid inequalities (5)–(6) from Theorems 1 and 2 can be added directly to IP formulations for solving the (k, λ)-subgraph problem.

Theorem 3 *For any subset S of V, if its cardinality $|S|$ satisfies $n-k+1 \leq |S| \leq k-1$, the inequality*

$$\sum_{i\in S} \sum_{j\in V\setminus S} y_{ij} \geq \lambda \tag{7}$$

is valid for any IP formulations to solve the (k, λ)-subgraph problem.

Proof Since $n - k + 1 \leq |S| \leq k - 1$, the cardinality of $V\setminus S$ should also satisfy $n - k + 1 \leq |V\setminus S| \leq k - 1$. To find a subgraph with at least k vertices for the (k, λ)-subgraph problem, there is at least one vertex in S and at least one vertex in $V\setminus S$. Because of λ-edge-connectivity, the inequality is valid. □

In Theorem 3, the choices of S implies a relationship between k and n, i.e, $n-k+1 \leq k - 1$ or $k \geq \frac{n}{2} + 1$. That is, the inequality is valid only for finding subgraphs with more than half of total vertices. For example, when $n = 10$, Theorem 3 implies that there are inequalities (7) when $k = 6, 7, 8, 9, 10$. When $k = 6$, only subsets with cardinality 5 can be used to generate inequalities (7); when $k = 8$, any subsets with

cardinality 3, 4, 5, 6 or 7 can be used to generate inequalities (7); When $k = 10$, all nonempty proper subsets (i.e., with cardinality 1, 2, 3, 4, 5, 6, 7, 8 or 9) can be used to generate inequalities (7).

When $k = n$, the possible subsets satisfying the inequality of Theorem 3 have cardinality between 1 and $n-1$, and these are subsets S such that $S \subset V$, $S \neq \emptyset$. We have the following corollary:

Corollary 1 *If $k = |V|$, in the cutset formulation (1), the constraints in (1c) can be reduced to $\sum_{i\in S}\sum_{j\in V\setminus S} y_{ij} \geq \lambda$ for any $S \subset V$, $S \neq \emptyset$.*

When $k = |V|$, this problem reduces to the λ-edge-connected subgraph problem, as discussed in Sect. 1. The new formulation from above corollary is widely used for finding this subgraph (see Magnanti and Raghavan 2005).

Theorem 4 *The inequality*

$$\sum_{i\in S_1}\sum_{j\in S_2} y_{ij} + \sum_{i\in S_1}\sum_{j\in S_3} y_{ij} + \sum_{i\in S_2}\sum_{j\in S_3} y_{ij} \geq \frac{3}{2}\lambda \tag{8}$$

is valid for any IP formulations to solve the (k, λ)-subgraph problem, if the partition S_1, S_2, S_3 of the vertex set V has the cardinality requirements as $n-k+i_1 \leq |S_1| \leq k-i_2$, and $|V\setminus S_1|-i_2+1 \leq |S_2| \leq i_2-1$ and $|S_3| \geq 1$, for some $i_1 \in \{1, 2, \ldots, k-1\}$ and $i_2 \in \{2, 3, \ldots, k-1\}$.

Proof The condition $|S_1| \geq n-k+i_1$ implies $|S_2 \cup S_3| = |V\setminus S_1| \leq k-i_1$. That is, the (k, λ)-subgraph has at least i_1 vertices in S_1.

On the other hand, the condition $|S_1| \leq k-i_2$ implies that there are at most $k-i_2$ vertices in S_1, or equivalently, there are at least i_2 vertices in $S_2 \cup S_3$. From $|V\setminus S_1|-i_2+1 \leq |S_2|$, we have $|S_3| = |V\setminus S_1\setminus S_2| \leq i_2-1$. As given in the conditions $|S_2| \leq i_2-1$, there is at least one vertex to be chosen into the (k, λ)-subgraph from each of S_2 and S_3.

Therefore, the (k, λ)-subgraph has at least one vertex from each of the subsets S_1, S_2, S_3. Let d_{S_i} be the number of chosen edges into the subgraph with exactly one end in S_i $(i = 1, 2, 3)$. Therefore, the total number of chosen connecting distinct subsets is $\frac{1}{2}(d_{S_1} + d_{S_2} + d_{S_3})$.

Consider S_1, $S_2 \cup S_3$ as a bipartition of vertex set V, by Theorem 3, $d_{S_1} \geq \lambda$ should hold. Similarly, we have $d_{S_2} \geq \lambda$ and $d_{S_3} \geq \lambda$. Thus, $\frac{1}{2}(d_{S_1} + d_{S_2} + d_{S_3}) \geq \frac{3}{2}\lambda$. Therefore, the inequality (8) is valid for finding the (k, λ)-subgraph. □

In Theorem 4, the choices of i_1, i_2 in corresponding sets $i_1 \in \{1, 2, \ldots, k-1\}$ and $i_2 \in \{2, 3, \ldots, k-1\}$, should also satisfy the implicit conditions from $n-k+i_1 \leq |S_1| \leq k-i_2$, and $|V\setminus S_1| - i_2 + 1 \leq |S_2| \leq i_2 - 1$, or $k \geq n+2-i_2, k \geq \frac{n+i_1+i_2}{2}$, $|S_1| \geq n+2-2i_2$.

For example, when $n = 10$, Theorem 4 can generate inequalities when $k = 9, 10$. When $k = 9$, there are 4 partitions, with cardinalities 3, 3, 4, 2, 4, 4, 2, 3, 5, or 2, 2, 6. For $k = 10$, there are 8 partitions, with cardinalities 1, 1, 8 , 1, 2, 7, 1, 3, 6, 1, 4, 5, 2, 2, 6, 2, 3, 5, 2, 4, 4, or 3, 3, 4.

When $n = 15$ in a network, in case of k is 12, 13, 14 or 15, a inequality of type (8) can be generated. For example, when $k = 12$ there are 3 possible partitions to generate cuts, with cardinalities 4, 4, 7, 4, 5, 6, or 5, 5, 5. When $k = 13$ there are 7 partitions with cardinalities 3, 3, 9, 3, 4, 8, 3, 5, 7, 3, 6, 6, 4, 4, 7, 4, 5, 6, or 5, 5, 5. When $k = 14$, there are 12 partitions with cardinalities 3, 3, 9, 3, 4, 8, 3, 5, 7, 3, 6, 6, 4, 7, 4, 4, 5, 6, 4, 8, 3, 4, 9, 2, 5, 5, 5, 5, 8, 2, 6, 7, 2, or 2, 6, 7.

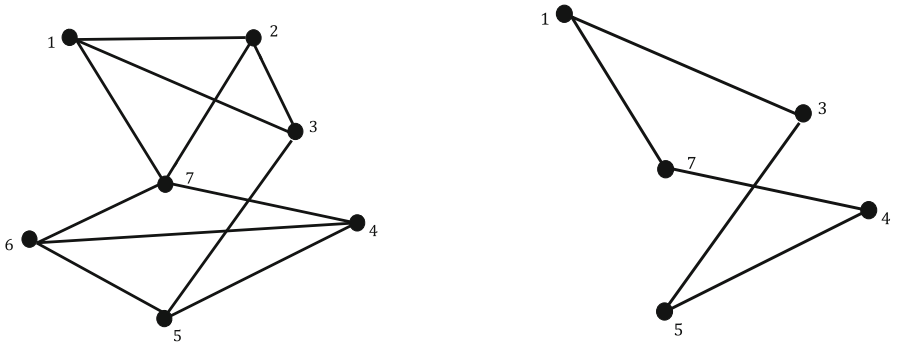
In general when $k = n$, each possible partition of V into three nonempty subsets gives a valid inequality (8). The total number of possible partitions for valid inequalities of type (8) is equal to the number $\llcorner \frac{n^2+6}{12} \lrcorner$ of partitions of an integer n into three positive integers.

In Theorem 3, we generate inequalities based on a bipartition of the vertex set V, while in Theorem 4, we generate inequalities based on a partition of V into three subsets. These theorems can be generalized into any possible partitions of the vertex set. For example, if there is at least one vertex in each subset of an m-partition (dividing V into m nonempty disjoint subsets) that the (k, λ)-subgraph should choose, the total number of edges, connecting distinct subsets, chosen into the subgraph should be larger than or equal to $\frac{m}{2}\lambda$.

3 Integer programming formulations for the $(k, 2)$-subgraph problem

Next, we study graph properties and stronger and more compact IP formulations for solving the $(k, 2)$-subgraph problem. This problem is a direct generalization of k-MST problem, but with higher connectivity requirement. Also, any optimal solution to the traveling salesman problem in a bridgeless graph leads to a feasible solution to the $(k, 2)$-subgraph problem (see Sebő and Vygen 2012). In Fig. 1, we give an example of $(k, 2)$-subgraph.

Lemma 2 *A graph G is 2-edge-connected if and only if one of the following properties holds:*



(a) Graph G with 7 vertices and 11 edges **(b)** A $(5, 2)$-subgraph of G

Fig. 1 An example for $(k, 2)$-subgraph

(i) *It is connected and bridgeless;*
(ii) *There exist two edge-disjoint paths between every pair of its vertices;*
(iii) (Robbins 1939) *it admits a strong orientation.*

A *bridge* of a graph is an edge whose deletion increases its number of connected components, and a graph is said to be *bridgeless* if it contains no bridges. From Lemma 2(ii) (a special case from Lemma 1), for a $(k, 2)$-subgraph, any two vertices must be connected by at least two edge-disjoint paths. In the following, we first prove a property of two edge-disjoint paths as a lemma for studying further properties of $(k, 2)$-subgraphs.

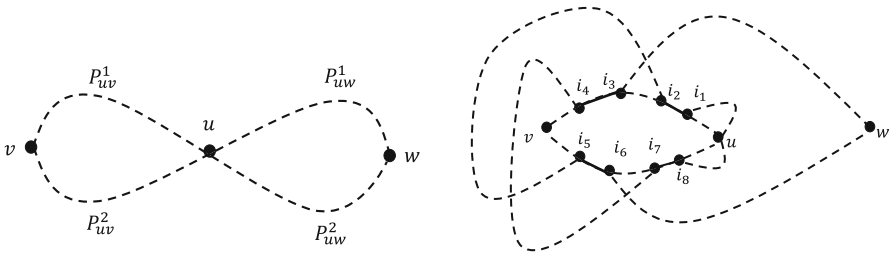
Lemma 3 *For any three distinct vertices $u, v, w \in V$ of graph G, if exist two edge-disjoint paths between vertices u, v and also two edge-disjoint paths between vertices u, w, there are two edge-disjoint paths between vertices v, w.*

Proof Let P^1_{uv}, P^2_{uv} be two edge-disjoint paths between vertices u, v and P^1_{uw}, P^2_{uw} edge-disjoint paths between vertices u, w. Thus, P^1_{uv} and P^2_{uv} do not share any edge, and also P^1_{uw} and P^2_{uw} do not.

If there is no edge shared by both paths P^1_{uv}, P^2_{uv} and P^1_{uw}, P^2_{uw}, two edge-disjoint paths can be found between v and w. The first path is formed by the reverse of P^1_{uv} from v to u and P^1_{uw} from u to w, while the second one is formed by the reverse of P^2_{uv} from v to u and P^2_{uw} from u to w (see Fig. 2a).

If there are edges shared by paths P^1_{uv}, P^2_{uv} and P^1_{uw}, P^2_{uw}. Without loss of generality, we consider the following four cases to find two edge-disjoint paths between v and w (see Fig. 2b, where P^1_{uv} goes through vertices i_1, i_2, i_3, i_4 and path P^2_{uv} goes through vertices i_8, i_7, i_6, i_5; and P^1_{uw} goes through i_1, i_2, i_5, i_6 and P^2_{uw} goes through vertices i_8, i_7, i_4, i_3):

(b.1) Only paths P^1_{uv} and P^1_{uw} have shared edges. Assume that shared edges are between segment i_1 to i_2 in both paths. The two disjoint paths between v and w can be found as follows: the first path consists of segment of v to i_2 in path P^1_{uv}, and segment i_2 to w in path P^1_{uw}, and the second one is formed by the reverse of P^2_{uv} from v to u and P^2_{uw} from u to w.
(b.2) Both paths P^1_{uv}, P^2_{uv} have shared edges with only path P^1_{uw}. Assume the shared edges for P^1_{uv} and P^1_{uw} are between segment i_1 to i_2, and the shared edges for



(a) Case of no shared edges **(b)** Case with shared edges

Fig. 2 Explanations of proof for Lemma 3

P^2_{uv} and P^1_{uw} are between i_5 and i_6. The two disjoint paths between v and w can be found as follows: the first path consists of segment of v to i_5 in path P^2_{uv}, and segment i_5 to w in path P^1_{uw}, and the second one is formed by the reverse of P^1_{uv} from v to u and P^2_{uw} from u to w.

(b.3) Paths P^1_{uv} and P^1_{uw} have shared edges, and paths P^2_{uv} and P^2_{uw} also have shared edges. Assume the shared edges for P^1_{uv} and P^1_{uw} are between segment i_1 to i_2, and the shared edges for P^2_{uv} and P^2_{uw} are between i_8 and i_7. The two disjoint paths between v and w can be found as follows: the first path consists of segment of v to i_2 in path P^1_{uv}, and segment i_2 to w in path P^1_{uw}, and the second one is formed by segment of v to i_7 in path P^2_{uv}, and segment i_7 to w in path P^2_{uw}.

(b.4) Both paths P^1_{uv}, P^2_{uv} have shared edges with path P^1_{uw}, and also they have shared edges with path P^2_{uw}. Assume the shared edges for P^1_{uv} and P^1_{uw} are between segment i_1 to i_2, and the shared edges for P^2_{uv} and P^1_{uw} are between i_5 and i_6. Assume the shared edges for P^1_{uv} and P^2_{uw} are between segment i_4 to i_3, and the shared edges for P^2_{uv} and P^2_{uw} are between i_8 and i_7. The two disjoint paths between v and w can be found as follows: the first path consists of segment of v to i_4 and then to i_3 in path P^1_{uv}, and segment i_3 to w in path P^2_{uw}, and the second one is formed by segment of v to i_5 and then to i_6 in path P^2_{uv}, and segment i_6 to w in path P^1_{uw}.

Therefore, under any case, we have found two edge-disjoint paths between v and w. □

In addition to formulations and methods discussed in Sect. 2, by assigning $\lambda = 2$ to solve the $(k, 2)$-subgraph problem, we will use Lemmas 2 and 3 to present more properties and IP formulations for this special case.

3.1 Connectivity and bridgeless for $(k, 2)$-subgraph

From Lemma 2(ii), we can have following theorem for finding a $(k, 2)$-subgraph.

Theorem 5 *A graph $G = (V, E)$ is 2-edge-connected if and only if for any edge $(i, j) \in E$, there exists at least one path between i and j such that the path does not contain (i, j), and additionally, the graph is connected.*

Proof ($\Rightarrow$) From Lemma 1, there exist at least two edge-disjoint paths between every pair of vertices. Thus, when $(i, j) \in E$ there exists a path between i and j such that does not contain (i, j).

Additionally, since there exists at least one path between any two vertices in G, this graph G is connected.

($\Leftarrow$) We claim that for every pair of vertices $s, t \in V (s \neq t)$, there exist two edge-disjoint paths between s and t. In fact, there are two cases:

(i) $(s, t) \in E$. By the assumption, there exists one path between s and t that does not contain (s, t). Thus, there exit two edge-disjoint paths between s and t.

(ii) $(s, t) \notin E$. As the graph G is connected, it has only one component. Also, according to the assumptions, each edge must be on at least one cycle. We claim

the graph is bridgeless. By contradiction, if there exists one bridge with edge (i_1, i_2) on it, this will contradict to the assumption of the existence of the path from i_1 to i_2 which does not contain edge (i_1, i_2). By Lemma 2(i) and 2(ii), there exist two edge-disjoint paths between s, t. This completes the proof. □

Let the decision variable $z_{ij}^{(st)}$ denote the flow on edge $(i, j) \in E$ from i to j to ensure the existence of a path from s to t, while $z_{ji}^{(st)}$ denotes the flow from j to i on edge $(i, j) \in E$. The decision variable $r_i \in \{0, 1\}$ for all $i \in V$ is an indicator for fixing a vertex i when $r_i = 1$. The decision variable u_{ij} indicates the amount of flow on edge $(i, j) \in E \cup E'$ to ensure the connectivity. Based on Theorem 5, we have the following IP formulation for solving the $(k, 2)$-subgraph problem:

$$\min \sum_{(i,j)\in E} c_{ij} y_{ij} \tag{9a}$$

$$s.t. \sum_{i\in V} x_i \geq k \tag{9b}$$

$$y_{ij} \leq x_i, y_{ij} \leq x_j, \ \forall (i, j) \in E \tag{9c}$$

$$z_{ij}^{(st)} \leq y_{ij}, z_{ji}^{(st)} \leq y_{ij}, \forall (s, t) \in E, (i, j) \in E \tag{9d}$$

$$\sum_{\substack{j:(i,j)\neq(s,t),(t,s)\\(i,j)\in E\cup E'}} z_{ij}^{(st)} - \sum_{\substack{j:(j,i)\neq(s,t),(t,s)\\(i,j)\in E\cup E'}} z_{ji}^{(st)} = \begin{cases} y_{st}, & i = s \\ -y_{st}, & i = t \\ 0, & i \in V\backslash\{s, t\} \end{cases} \quad \forall (s, t) \in E, \tag{9e}$$

$$\sum_{i\in V} r_i = 1 \tag{9f}$$

$$r_i \leq x_i, \forall i \in V \tag{9g}$$

$$\sum_{j:(j,i)\in E\cup E'} u_{ji} - \sum_{j:(i,j)\in E\cup E'} u_{ij} = x_i - r_i \sum_{j\in V} x_j, \ \forall i \in V \tag{9h}$$

$$\sum_{j:(j,i)\in E\cup E'} u_{ji} \leq (1 - r_i)n, \forall i \in V \tag{9i}$$

$$u_{ij} \leq y_{ij} \sum_{k\in V} x_k, u_{ji} \leq y_{ij} \sum_{k\in V} x_k, \forall (i, j) \in (E \cup E') \tag{9j}$$

$$y_{ij} \in \{0, 1\}, \ \forall (i, j) \in E; \ x_i, r_i \in \{0, 1\}, \forall i \in V \tag{9k}$$

$$u_{ij}, z_{ij}^{(st)} \geq 0, \ \forall (i, j) \in E \cup E', \ \forall (s, t) \in E \tag{9l}$$

The objective (9a) and constraints (9b) and (9c) are the same as those corresponding ones for the (k, λ)-subgraph problem. Constraints (9d)–(9e) guarantee one path from s to t for each $(s, t) \in E$ such that it does not contain (s, t) if $(s, t) \in E$. For $(s, t) \in E$, (i) if it is not chosen into the subgraph, $y_{st} = 0$, outflow of s and inflow of t are 0, and also the flow for other transshipment vertices is 0. (ii) if it is chosen into the subgraph,

$y_{st} = 1$ and by constraints (9e), there exists one unit of flow from s to t such that the path does not use edge (s, t).

Constraints (9f)–(9j) ensure connectivity of the subgraph, induced by $V' = \{i \in V : x_i = 1\}$, through a set of flow constraints from a root i with $r_i = 1$. Constraints (9f)–(9g) select such a root as the staring point for the flow. The constraint (9i) ensures the inflow of the selected root is 0 (if $r_i = 1$) or less than $|V|$, as an upper bound for all other vertices such that $r_i = 0$. Constraints (9h) ensure the balance of flow on each vertex. (i) if vertex i is chosen as the fixed vertex, i.e., $r_i = x_i = 1$, the outflow of is $\sum_j x_j - 1$. (ii) if the vertex i is chosen in the subgraph but not as the fixed vertex, i.e., $x_i = 1, r_i = 0$, the difference between the inflow and outflow will be 1. (iii) otherwise $r_i = x_i = 0$, inflow and outflow are both 0. Constraints (9j) indicate that the flow is 0 if an edge is not selected into the subgraph induced by V'. The idea by root and flows to ensure the connectivity was introduced in Fan and Watson (2012).

The last two sets of constraints in (9) are for binary and nonnegativity requirements for decision variables. Together, these constraints in (9) imply a $(k, 2)$-subgraph $G' = (V', E')$. This formulation (9) is an integer program, with some nonlinear terms, which can be linearized by approaches introduced in (4).

From Theorem 5, we know that any edge in a 2-edge-connected graph is with a circle. Therefore, for a $(k, 2)$-subgraph of G, each chosen edge into this subgraph must have at least one chosen edge connected to each end, and we have the following theorem:

Theorem 6 *The following inequalities,*

$$y_{ij} \leq \sum_{k:k\in V, k\neq i} y_{jk}, \text{ and } y_{ij} \leq \sum_{k:k\in V, k\neq i} y_{ki}, \quad \forall (i, j) \in E \tag{10}$$

are valid for any IP formulations to solve the $(k, 2)$*-subgraph problem.*

3.2 Strong orientation for (k, 2)-subgraph

For an undirected graph, a *strong orientation* is an assignment of a direction (an orientation) to each edge that makes the graph into a strongly connected graph. A directed graph is called *strongly connected* if there is a path in each direction between each pair of its vertices.

Lemma 4 *A directed graph is strongly connected if and only if for a fixed vertex, there exists a directed path from it to every other vertex and also there exists a directed path from every other vertex to this fixed one.*

Proof ($\Rightarrow$) Without loss of generality, assume that the fixed vertex is v_1, there exists a directed path from v_1 to every other vertex in this graph, and also a directed path from every other vertex to v_1, as the graph is strongly connected.

($\Leftarrow$) Assume the fixed vertex is v_1, and there exists a directed path from v_1 to every other vertex v_i $(i = 2, \ldots, N)$, and also a directed path from v_i to v_1. Now, we want to show that for each pair of vertices v_i, v_j $(i \neq j, i, j = 1, \ldots, N)$, there exists a

directed path from v_i to v_j and also there exits a directed path from v_j to v_i. In fact, we can show this by following two cases:

(i) One of v_i, v_j is the fixed vertex v_1. Without loss of generality, we assume $v_i = v_1$ in the pair. Therefore, for any v_j ($j \neq i$), by assumption, there is one path from v_1 to this v_j and another one from v_j to v_1.
(ii) None of v_i, v_j is the fixed vertex v_1. Now there is a directed path from v_1 to v_i and also a directed path from v_i to v_1. Similarly, two such paths exist between v_1 and v_j. Similar to the proof of Lemma 3, there exists a directed path from v_i to v_j and also a directed path from v_j to v_i.

Therefore, by summarizing these two cases, there exists a path between any two vertices in each direction, and the graph is strongly connected. □

By Lemma 2(iii), and Lemma 4, we have the following theorem for 2-edge-connected subgraphs.

Theorem 7 *A graph is 2-edge-connected, if and only if there exists an assignment of a direction to each edge such that for a fixed vertex, there exists a directed path from it to every other vertex and also there exists a directed path from every other vertex to this fixed one.*

Let the decision variable $r_i \in \{0, 1\}$ for all $i \in V$ be the indicator for fixing a vertex i when $r_i = 1$ and 0 otherwise. The decision variable y'_{ij} is used to indicate the direction of each edge (i, j) such that $y'_{ij} = 1$ means edge (i, j) having direction from i to j and $y'_{ji} = 1$ means edge (i, j) having direction from j to i. The decision variable z^k_{ij} denotes the flow from the fixed vertex (decided by r_i) to vertex k on the arc from i to j (direction decided by y'_{ij}) to ensure the existence of the path from the fixed vertex to k. The decision variable w^k_{ij} denotes the flow from vertex k to the fixed vertex on the arc from i to j to ensure the existence of the path from k to the fixed vertex. By Theorem 7, we have the following formulation for solving the $(k, 2)$-subgraph problem:

$$\min \sum_{(i,j)\in E} c_{ij} y_{ij} \tag{11a}$$

$$s.t. \sum_{i\in V} x_i \geq k \tag{11b}$$

$$y_{ij} \leq x_i, y_{ij} \leq x_j, \ \forall (i, j) \in E \tag{11c}$$

$$\sum_{i\in V} r_i = 1 \tag{11d}$$

$$r_i \leq x_i, \ \forall i \in V \tag{11e}$$

$$y'_{ij} + y'_{ji} = y_{ij}, \ \forall (i, j) \in E \tag{11f}$$

$$\sum_{j:(i,j)\in E\cup E'} z^k_{ij} - \sum_{j:(j,i)\in E\cup E'} z^k_{ji} = \begin{cases} r_i x_k, & i \in V\backslash\{k\}, \\ -x_k(1 - r_k), & i = k, \end{cases} \quad \forall k \in V \tag{11g}$$

$$z^k_{ij} \leq y'_{ij}, z^k_{ji} \leq y'_{ji}, \ \forall (i, j) \in E, k \in V \tag{11h}$$

$$\sum_{j:(i,j)\in E\cup E'} w_{ij}^k - \sum_{j:(j,i)\in E\cup E'} w_{ji}^k = \begin{cases} x_k(1-r_k), & i=k, \\ -r_i x_k, & i \in V\backslash\{k\}, \end{cases} \quad \forall k \in V \tag{11i}$$

$$w_{ij}^k \le y'_{ij},\ w_{ji}^k \le y'_{ji},\ \forall (i,j) \in E, k \in V \tag{11j}$$

$$x_i, u_i \in \{0,1\},\ \forall i \in V;\ y_{ij}, y'_{ij}, y'_{ji} \in \{0,1\},\ \forall (i,j) \in E \tag{11k}$$

$$z_{ij}^k, z_{ji}^k, w_{ij}^k, w_{ji}^k \ge 0,\ \forall (i,j) \in E, k \in V \tag{11l}$$

The objective (11a), and constraints (11b)–(11c) are the same as those corresponding ones for (k,λ)-subgraph problem. Constraints (11d) and (11e) are used to find a fixed vertex in the $(k,2)$-subgraph decided by x, y. Constraints (11f) are used to assign a direction for each chosen edge with $y_{ij} = 1$.

Constraints (11g)–(11h) guarantee one directed path from the fixed vertex (decided by r_i) to every other vertex k by using the arc from i to j (direction decided by y'_{ij}). For $k \in V$, (i) if it is not chosen into the subgraph, $r_k = x_k = 0$. Constraints (11g) indicate that $z_{ij}^k = 0$ for all $(i,j) \in E \cup E'$. (ii) if it is chosen into the subgraph and it is the fixed vertex, $r_k = x_k = 1$. Constraints (11g) become

$$\sum_j z_{ij}^k - \sum_j z_{ji}^k = 0,\ \forall i \in V$$

and thus $z_{ij}^k = 0$ for all $(i,j) \in E \cup E'$. (iii) if it is chosen into the subgraph but not the fixed vertex, $x_k = 1, r_k = 0$. Constraints (11g) become

$$\sum_j z_{ij}^k - \sum_j z_{ji}^k = \begin{cases} r_i, & i \in V\backslash\{k\}, \\ -1, & i = k, \end{cases}$$

where first part includes cases: if $i \in V\backslash\{k\}$ is the fixed vertex ($r_i = 1$), it is the start point to find a directed path to vertex k; if $i \in V\backslash\{k\}$ is not ($r_i = 0$), it may be a transshipment vertex in the directed path to k. From cases (i) to (iii), the set of constraints find a path from the fixed vertex to every other vertex.

Similarly, constraints (11i)–(11j) find one directed path from each vertex to the fixed one. The last two sets of constraints in (11) are the binary and nonnegativity requirements for decision variables. This formulation (11) is an integer program, with some bilinear terms, which can be linearized by approaches introduced in (4).

4 Numerical experiments

In this section, all IP formulations and inequalities discussed above are implemented in C++ using CPLEX 12.3 via IBM's Concert Technology library, version 2.9. All experiments were performed on a Linux workstation with 4 Intel(R) Core(TM)2 CPU 2.40GHz processors and 8GB RAM. In the following, all graphs and edge costs we test for $(k,2)$-subgraph problem and (k,λ)-subgraph problem are randomly generated with the weight $i + j$ for each edge $(i,j) \in E$, and 3000 s are the time limit for tests. Results of Tables 1, 2 and 3 are tested on the same group of randomly generated

Table 1 Computational results for the $(k, 2)$ subgraph problem with valid inequalities

$\|V\|$	$\|E\|$	k	Path formulation	Formulation	Strong orientation formulation (11)+						
			(3)+Thm. 1	(9)+Thm. 1	Thm. 1	Thm. 2	Thm. 6	Thm. 1, 2	Thm. 1, 6	Thm. 1, 2, 6	No other inequalities
5	10	3	0.05	0.04	0.01	0.04	0.03	0.01	0.01	0.01	0.04
		5	0.01	0.03	0.03	0.11	0.11	0.02	0.02	0.03	0.12
10	45	3	0.62	0.21	0.07	0.38	0.37	0.04	0.36	0.04	0.57
		7	1.15	0.31	0.12	1.88	1.89	1.01	1	1.01	1.9
		10	1.29	0.75	0.51	2.22	2.22	0.35	0.35	0.34	2.22
15	105	3	5.18	1.31	0.37	1.63	1.63	0.38	0.38	0.37	1.62
		7	23.47	1.77	0.82	6.9	6.97	0.81	0.8	0.81	6.94
		11	14.87	1.73	0.58	16.6	16.64	0.78	0.75	0.57	16.9
		15	7.55	2.07	1.97	107.89	107.96	1.97	1.97	2.07	108
20	114	3	14.12	3.94	0.84	5.17	5.18	0.84	0.84	0.84	5.16
		7	32.64	2.65	1.01	31.98	31.94	1.02	1.04	1.06	32.14
		11	98.68	15.54	3.62	587.77	590	3.62	3.57	3.6	599.86
		15	>3000	23.42	5.35	1359	1371	5.47	5.45	5.33	1374
		19	>3000	>3000	24.35	2246	2173	24.89	24.45	24.69	2179
		20	>3000	134.96	15.12	>3000	>3000	15.29	15.3	15.04	>3000
30	131	3	50.85	1.87	1.25	3.29	3.26	1.25	1.2	1.2	3.27
		7	211	6.4	3.44	61.04	61.38	3.31	3.32	3.3	61.18
		11	566.24	8.35	5.8	403.65	403.87	5.6	5.68	11.39	403.53
		15	>3000	526.28	11.7	>3000	>3000	11.68	11.65	11.62	>3000
		19	>3000	3515	23.03	>3000	>3000	23.49	19.17	23.11	>3000
		23	>3000	>3000	44.37	>3000	>3000	44.45	44.58	44.52	>3000
		27	>3000	59.92	44.82	>3000	>3000	45.24	45.01	44.54	>3000
		30	>3000	14.11	22.64	>3000	>3000	22.46	22.33	22.56	>3000

Table 1 continued

$\|V\|$	$\|E\|$	k	Path formulation	Formulation	Strong orientation formulation (11)+						
			(3)+Thm. 1	(9)+Thm. 1	Thm. 1	Thm. 2	Thm. 6	Thm. 1, 2	Thm. 1, 6	Thm. 1, 2, 6	No other inequalities
40	150	3	158.73	2.71	0.79	3.86	3.85	0.82	0.8	0.79	3.84
		11	2351	12.18	6.08	1051	1051.88	6.1	6.11	6.2	1050.87
		15	>3000	>3000	13.48	>3000	>3000	13.55	13.57	13.4	>3000
		19	>3000	>3000	34.43	>3000	>3000	33.91	34.37	33.73	>3000
		23	>3000	>3000	62.07	>3000	>3000	62.27	62.54	61.85	>3000
		27	>3000	>3000	150.76	>3000	>3000	150	149.78	150.25	>3000
		31	>3000	>3000	282.59	>3000	>3000	284.57	283.42	284.37	>3000
		35	>3000	>3000	215.54	>3000	>3000	213.9	217.08	214.24	>3000
50	230	3	679.61	6.86	1.44	9.16	9.21	1.44	1.41	1.42	9.08
		7	662	18.3	6.77	101.15	106.35	6.73	6.68	6.66	101.55
		11	>3000	236	15.39	2100	2150.58	15.01	15.15	15.15	2100.05
		15	>3000	119	23.63	>3000	>3000	23.67	23.66	23.58	>3000
		19	>3000	>3000	102.12	>3000	>3000	103	101	100.48	>3000
		23	>3000	>3000	224	>3000	>3000	229	233	218.36	>3000
		27	>3000	>3000	331	>3000	>3000	335	343	331.01	>3000
		31	>3000	>3000	619.38	>3000	>3000	643.34	628.53	649.63	>3000
		35	>3000	>3000	773.36	>3000	>3000	757.81	762.6	771.03	>3000

na the corresponding inequalities do not apply in this case

Table 2 Computational results of $(k, 2)$-subgraph problem with inequalities from Theorems 3 and 4

$\|V\|$	$\|E\|$	k	Strong orientation formulation (11)+						Best of results in Table 1
			Thm. 3	Thm. 4	Thm. 1, 3	Thm. 1, 4	Thm. 1, 3, 6	Thm. 1, 2, 3	
5	10	5	0.04	0.06	0.02	0.01	0.01	0.01	0.01
10	45	7	1.80	na	1.09	na	1.08	1.08	0.31
		9	2.08	3.76	0.40	0.70	0.40	0.40	0.42
		10	1.67	2.63	0.29	0.30	0.30	0.29	0.34
15	105	9	4.50	na	1.60	na	1.61	1.62	1.13
		11	22.72	na	0.59	na	0.60	0.59	0.58
		13	118.00	114.00	1.64	1.14	1.63	1.64	0.61
		15	137.00	164.00	1.87	2.32	1.74	2.00	1.97
20	114	11	1346.00	na	3.84	na	3.80	3.75	3.57
		13	1719.00	na	4.71	na	4.69	4.66	3.94
		15	2893.00	>3000	8.07	7.00	8.06	8.05	5.35
		17	>3000	>3000	10.67	21.98	14.51	14.50	12.46
		19	>3000	>3000	22.31	25.37	22.31	22.16	24.35
		20	>3000	>3000	10.32	10.01	10.33	10.31	15.12
30	131	23	>3000	>3000	44.19	44.22	44.52	34.50	44.37
		25	>3000	>3000	46.97	39.79	47.80	51.91	46.97
		27	>3000	>3000	44.50	67.86	44.54	39.92	44.82
		29	>3000	>3000	51.26	47.98	38.37	51.36	37.81
		30	>3000	>3000	29.98	21.99	22.56	29.66	22.33

Table 2 continued

$\|V\|$	$\|E\|$	k	Strong orientation formulation (11)+						Best of results in Table 1
			Thm. 3	Thm. 4	Thm. 1, 3	Thm. 1, 4	Thm. 1, 3, 6	Thm. 1, 2, 3	
40	150	23	>3000	na	70.17	na	61.85	70.05	61.85
		25	>3000	na	146.86	na	94.51	145.78	94.39
		27	>3000	na	127.78	na	150.25	126.78	149.78
		29	>3000	>3000	154.96	181.47	198.39	153.62	198.80
		31	>3000	>3000	268.12	293.11	284.37	265.35	282.59
		33	>3000	>3000	343.51	298.91	288.8	343.94	286.51
		35	>3000	>3000	301.10	324.25	214.24	300.04	213.9
50	230	27	>3000	na	336.00	na	331.01	264.11	331.00
		29	>3000	na	372.00	na	365.93	408.71	360.00
		31	>3000	na	596.30	na	649.63	595.09	619.38
		33	>3000	na	589.03	na	717.26	575.32	717.26
		35	>3000	>3000	822.27	985.09	771.03	809.11	757.81

"na" means that the corresponding inequalities do not apply in this case

Table 3 Computational results for the (k, λ)-subgraph problem with inequalities

$\|V\|$	$\|E\|$	k	λ	Cutset+	Path formulation (3)+				
				Thm. 1	Thm. 1	Thm. 2	Thm. 3	Thm. 4	No other inequalities
5	10	4	1	0	0.02	0.04	0.44	na	0.04
		4	3	0	0.07	0.09	0.13	na	0.08
		5	1	0	0.01	0.01	0	0.01	0.01
		5	3	0	0.02	0.01	0.02	0.03	0.02
		5	4	0	0.01	0.01	0.02	0.01	0.01
10	45	4	1	OM	0.53	6.93	na	na	7.05
		4	3	0.64	0.08	58.91	na	na	58.66
		5	1	OM	0.50	18.91	na	na	18.73
		5	3	0	1.27	288.04	na	na	285.24
		5	4	0.01	1.62	329.29	na	na	326.70
		6	1	OM	0.54	28.84	na	na	29.07
		6	3	0.71	1.97	311.60	na	na	308.00
		6	4	0.01	2.32	885.85	na	na	891.87
		7	1	OM	0.49	16.67	24.96	na	16.74
		7	3	0.16	2.08	378.28	41.69	na	378.80
		7	4	0.01	3.83	859.67	398.94	na	857.19
		8	1	OM	0.47	14.11	39.85	69.54	14.22
		8	3	0.01	2.45	275.64	83.19	49.91	275.22
		8	4	0.01	5.67	792.95	913.91	925.97	791.38
		9	1	OM	0.59	60.69	103.26	2.02	60.75
		9	3	0.01	5.51	24.48	59.00	49.07	24.67
		9	4	0	13.43	119.80	138.67	119.97	120.17
		10	1	OM	0.33	0.22	0.43	2.24	0.22
		10	3	0.04	2.82	3.38	2.32	1.86	3.37
		10	4	0	3.60	3.68	3.98	3.91	3.76
15	105	4	1	OM	6.65	5.31	na	na	5.44
		4	3	0.02	27.79	124.41	na	na	120.35
		5	1	OM	6.35	8.29	na	na	7.99
		5	3	0.01	13.83	313.12	na	na	313.76
		6	1	OM	6.69	16.51	na	na	16.42
		6	3	0.01	34.05	>3000	na	na	>3000
		7	1	OM	13.51	35.23	na	na	35.24
		7	3	0.01	15.44	>3000	na	na	>3000
		8	1	OM	19.70	86.39	na	na	87.31
		8	3	0.02	5.22	>3000	na	na	>3000
		9	1	OM	34.24	283.33	112.60	na	283.49
		9	3	0.01	20.28	>3000	>3000	na	>3000
		10	1	OM	12.01	68.39	81.02	na	69.48
		10	3	0.01	30.12	>3000	>3000	na	>3000

Table 3 continued

$\|V\|$	$\|E\|$	k	λ	Cutset+	Path formulation (3)+				
				Thm. 1	Thm. 1	Thm. 2	Thm. 3	Thm. 4	No other inequalities
		11	1	OM	12.35	88.17	77.71	115.21	89.01
		11	3	OM	44.68	1995.00	658.53	697.43	201.50
		12	1	OM	37.06	267.74	279.42	110.25	269.31
		12	3	OM	73.27	>3000	>3000	>3000	>3000
		13	1	OM	205.82	131.40	334.32	347.90	133.85
		13	3	OM	934.38	>3000	>3000	>3000	>3000
		14	1	OM	211.37	559.59	887.11	510.34	557.96
		14	3	OM	1757.40	>3000	>3000	>3000	>3000
		15	1	OM	113.71	867.01	136.94	65.32	868.28
		15	3	OM	24.30	>3000	>3000	>3000	>3000
20	114	4	1	OM	19.88	40.13	na	na	40.64
		4	3	0.04	59.13	1654.00	na	na	1687.00
		5	1	OM	18.36	63.92	na	na	64.37
		5	3	0.05	104.00	2997.00	na	na	>3000
		6	1	OM	23.80	225.88	na	na	225.41
		6	3	0.46	112.14	>3000	na	na	>3000
		7	1	OM	40.48	1524.24	na	na	1494.65
		7	3	0.03	107.07	>3000	na	na	>3000
		8	1	OM	940.00	>3000	na	na	>3000
		8	3	0.01	61.51	>3000	na	na	>3000
		9	1	OM	500.00	>3000	na	na	>3000
		9	3	0.03	211.94	>3000	na	na	>3000
		10	1	OM	>3000	>3000	na	na	>3000
		10	3	0.01	227.35	>3000	na	na	>3000

"na" means that the corresponding inequalities do not apply in this case; "OM" means "out of memory" on test server

graphs. For testing the k-clique problem (*the corresponding constraint* $\sum_{i\in V} x_i \geq k$ *is changed to* $\sum_{i\in V} x_i = k$ *for clique problems*), we generate some complete graphs for cases in Table 4. In Table 5, we test the $(k, 2)$-subgraph problem on graphs with 20 vertices but different densities, where the density of a graph is defined as ratio of the number of edges over $|V|(|V|-1)/2$ (in this case, 190 for graphs with 20 vertices).

First, we perform the numerical experiments for solving the $(k, 2)$-subgraph problem by different formulations. Initially, without adding any valid inequalities, we find that most cases in the randomly generated graphs will take more than 3000 s to obtain the optimal solutions [for example, by the formulation (11) from strong origination idea in Theorem 7; see the last column of Table 1]. Therefore, we add valid inequalities to test other formulations. Comparing the formulation by paths in (3) ($\lambda = 2$), formulations (9) and (11), with inequalities from Theorem 1 added, the computational

Table 4 Computational results for the k-clique problem

$\|V\|$	$\|E\|$	k	k-Clique	
			Path (3)+Thm. 1	Cutset+Thm. 1
5	10	3	0.05	0
		4	0.06	0
		5	0.01	0
10	45	3	0.62	0
		4	1.87	0.64
		5	4.79	0.01
		6	31.13	0.09
		7	90.20	0.02
		8	35.27	0.02
		9	10.20	0.01
		10	0.31	0
15	105	3	5.08	0.01
		5	17.48	0.01
		7	53.42	0.02
		9	90	0.02
		11	153.64	OM
		13	253.82	0M
		15	3.91	0M
20	190	3	33.89	0.02
		5	119.56	0.02
		7	284.01	0.02
		9	598.99	0.02
		11	1075.14	OM
		13	2009.92	OM
		15	>3000	OM
		17	>3000	OM
		20	23.71	OM

"OM" means "out of memory" on test server

seconds by (11) are better than (9), which is better than (3) (see columns 4-6 in Table 1).

Therefore, based on formulation (11) for solving the $(k, 2)$-subgraph problem, we test the efficiency of other inequalities (see columns 7–11 in Table 1, and also Table 2). As Theorems 3–4 can only be applied on certain cases, we put them in a separate Table 2. In this table and following tables with inequalities generated from Theorems 3–4, we only consider around $2k - n - 2$ inequalities generated by Theorem 3 and at least $\llcorner \frac{n^2+6}{12} \lrcorner$ inequalities by Theorem 4, because the number of possible inequalities generated from these two theorems is too large (see discussions in Section 2.3) and will cause the memory issues. Based on formulation (11), by adding a single type of inequalities from Theorems 1–4 and 7 (see columns 6–8 in Table 1 and columns 4–9 in Table 2), the best inequality is generated from Theorem 1. Therefore, we now

Table 5 Computational results of $(k, 2)$-subgraph problem with different graph densities

$\|V\|$	Density	$\|E\|$	k	Strong orientation formulation (11)+					
				Thm. 1	Thm. 2	Thm. 3	Thm. 4	Thm. 6	No other inequalities
20	1	190	3	0.79	2.71	na	na	2.76	2.72
			5	0.91	16.02	na	na	15.89	18.80
			7	1.05	131.60	na	na	131.09	131.24
			9	3.42	287.59	na	na	288.23	285.81
			11	4.49	1317.87	>3000	na	1319.00	1314.98
			13	7.86	2883.99	>3000	na	2904.30	2891.41
			15	9.50	>3000	>3000	>3000	>3000	3031.00
			17	7.54	>3000	>3000	>3000	>3000	>3000
			19	13.04	>3000	>3000	>3000	>3000	>3000
			20	30.74	>3000	>3000	>3000	>3000	>3000
20	0.8	150	3	0.19	4.20	na	na	3.97	4.26
			5	0.70	4.28	na	na	4.29	4.62
			7	1.99	35.68	na	na	35.87	35.63
			9	3.46	132.48	na	na	132.39	132.03
			11	6.20	288.69	1701.00	na	288.06	287.52
			13	6.01	758.50	3094.00	na	757.73	755.55
			15	9.17	1477.06	>3000	>3000	1470.00	1465.00
			17	15.39	1211.63	>3000	>3000	1204.63	1210.74
			19	17.81	882.8	>3000	>3000	881.72	880.97
			20	17.16	>3000	>3000	>3000	>3000	>3000
20	0.6	114	3	0.84	5.17	na	na	5.18	5.16
			5	0.29	3.65	na	na	3.63	3.63
			7	1.01	31.98	na	na	31.94	32.14
			9	2.45	129.00	na	na	128.61	128.59
			11	3.62	587.77	1346.00	na	590.00	599.86
			13	3.98	1305.00	1719.00	na	1296.00	1300.00
			15	5.35	1359.00	2893.00	>3000	1371.00	1374.00
			17	12.46	1512.00	>3000	>3000	1508.00	1515.00
			19	24.35	2246.00	>3000	>3000	2173.00	2179.00
			20	15.12	>3000	>3000	>3000	>3000	>3000
20	0.4	76	3	0.33	1.45	na	na	1.43	1.43
			5	0.64	3.29	na	na	3.31	3.30
			7	1.46	17.38	na	na	17.49	17.54
			9	0.75	12.84	na	na	12.83	12.86
			11	5.56	85.58	172.78	na	85.72	85.45
			13	2.63	346.54	647.04	na	347.64	347.61

Table 5 continued

$\|V\|$	Density	$\|E\|$	k	Strong orientation formulation (11)+					
				Thm. 1	Thm. 2	Thm. 3	Thm. 4	Thm. 6	No other inequalities
			15	3.86	681.65	766.93	742.57	681.74	681.11
			17	9.09	368.71	1129.05	985.88	272.00	364.51
			19	9.44	859.04	985.96	1020.47	869.41	858.33
			20	16.25	1074.82	925.02	997.33	1075.21	1077.32

Note: "na" means that the corresponding inequalities do not apply in this case

combine this type of inequality with others to see the effects. We observe that valid inequalities generated from Theorems 3 and 4 have improvements when k gets close to the value $|V|$, for example, cases of $k = 19, 20$ when $|V| = 20$, $k = 27, 31$ when $|V| = 40$ in Table 2.

Next, we perform numerical experiments for solving the (k, λ)-subgraph problem by formulation (3) and **Cutset** algorithm. Initially, without adding any valid inequalities, we find that most cases in the randomly generated graphs will take more than 3000 s to obtain the optimal solutions, for example, cases in last column of Table 3 by path formulation (3) when $n \geq 15, k \geq 5$. Therefore, we added inequalities generated from Theorems 1 to 4, to formulation (3). The **Cutset** algorithm is the most efficient one if it does not face the problem "out of memory" (see 5th column of Table 3). Based on formulation (3), we test the efficiency of valid inequalities from Theorems 1–4, and observe that the efficiency of valid inequalities generated from Theorem 1 is better than any others. For many cases, to which Theorems 3–4 can be applied, the efficiency by adding inequalities they generated are better than those ones generated from Theorem 2.

Additionally, in Table 3, when $\lambda = 1$, the corresponding results present the comparison for solving the k-MST problem. We can observe that proposed inequalities added on formulation (3) are quite fast to solve this type of problems, but the **Cutset** algorithm fails for memory issues.

We also test the **Cutset** algorithm and formulation (3) for solving the k-clique problem, i.e., $\lambda = k - 1$ (additionally requiring there are exact k vertices). For cases that **cutset** algorithm without memory issues, this algorithm for solving the k-clique problem is efficient. We can even solve a case with $|V| = 20$ and $k = 9$ less than one second (see the 5th column in Table 4).

In Table 1, for a graph with 20 vertices, we have tested different inequalities for solving the $(k, 2)$-subgraph problem on the graph with 114 edges. Now, in Table 5, we check the computational complexity by formulation (11) with different types of inequalities from Theorems 1–4, 6, for the graphs with different densities. Again, we conclude that the inequalities from Theorem 1 are the most efficient ones. From Table 5, under the same density, the computational seconds increase as k increases; under the same value of k, for the computational seconds, there are no big differences for graphs with different densities.

5 Conclusions

In this paper, we presented several IP formulations to solve the (k, λ)-subgraph problem, which is a generalization of the k-MST and λ-edge-connected spanning subgraph problems. Also, valid inequalities from several graph properties are studied to improve the efficiency of these IP formulations. As a special case, we considered the $(k, 2)$-subgraph problem, which can be considered as a generalization of the k-MST with higher connectivity requirement. Because of special properties for 2-edge-connected graphs, we proposed theorems based on strong orientation, and bridgeless and connectivity, to give stronger and more compact formulations. Numerical experiments performed on randomly generated graphs show that the formulation (3) with inequalities from Theorem 1 is the most efficient one to solve the (k, λ)-subgraph problem, and the formulation (11) with inequalities from Theorem 1 is the most efficient one to solve the $(k, 2)$-subgraph problem. Additionally, inequalities generated from Theorems 3–4 could also have improvements for solving the $(k, 2)$-subgraph problem when k gets close to the number of vertices. We only consider using valid inequalities on proposed IP formulations directly to solve these problems, and thus future directions include proposing decomposition algorithms, and studying facet properties of these inequalities.

References

Bendali F, Diarrassouba I, Didi Biha M, Mahjoub AR, Mailfert J (2007) The k-edge connected subgraph problem: valid inequalities and branch-and-cut. Design Reliab Commun Networks 1–8

Bienstock D, Brickell EF, Monma CL (1990) On the structure of minimum-weight k-connected spanning networks. SIAM J Discret Math 3(3):320–329

Cai GR, Sun YG (1989) The minimum augmentation of any graph to a k-edge-connected graph. Networks 19(1):151–172

Chekuri C, Korula N (2008) Min-cost 2-connected subgraphs with k terminals, manuscript 2008. Available at arXiv:0802.2528

Chimani M, Kandyba M, Ljubi I, Mutzel P (2009) Obtaining optimal k-cardinality trees fast. J Exp Algorithm 14(9):1–23

Fan N, Watson JP (2012) Solving the connected dominating set problem and power dominating set problem by integer programming. Lecture Notes Comp Sci 7402:371–383

Fortz B, Mahjoub AR, McCormick ST, Pesneau P (2006) Two-edge-connected subgraphs with bounded rings: Polyhedral results and branch-and-cut. Math Program 105:85–111

Frederickson GN, Ja'Ja' J (1981) Approximation algorithms for several graph augmentation problems. SIAM J Comp 10(2):270–283

Gabow HN, Goemans MX, Tardos E, Williamson DP (2009) Approximating the smallest k-edge connected spanning subgraph by LP-rounding. Networks 53(4):345–357

Garey MR, Johnson DS (1979) Computers and intractability: a guide to the theory of NP-Completeness. Freeman, San Francisco

Garg N (2005) Saving an epsilon: a 2-approximation for the k-MST problem in graphs. Theory Comp 37:396–402

Goemans MX, Bertsimas D (1993) Survivable networks, linear programming relaxations and the parsimonious property. Mathl Program 60:145–166

Jain K (2001) A factor 2 approximation algorithm for the generalized Steiner network problem. Combinatorica 21(1):39–60

Karp RM (1972) Reducibility among combinatorial problems. Compl Comp Comput 85–103

Khuller S, Vishkin U (1994) Biconnectivity approximations and graph carvings. SIAM J Comp 41(2):214–235

Lau LC, Naor J, Salavatipour MR, Singh M (2009) Surivable network design with degree or order constraints. SIAM J Comp 39(3):1062–1087

Magnanti TM, Raghavan S (2005) Strong formulations for network design problems with connectivity requirements. Networks 45(2):61–79

Menger K (1927) Zur allgemeinen Kurventheorie. Fund Mathem 10:96–115

Quintpo FP, da Cunha AS, Mateus GR, Lucena A (2010) The k-cardinality tree problem: reformulations and lagrangian relaxation. Discret Appl Math 158(12):1305–1314

Robbins HE (1939) A theorem on graphs, with an application to a problem of traffic control. Am Math Monthly 46(5):281–283

Safari MA, Salavatipour MR (2008) A constant factor approximation for minimum λ-edge-connected k-subgraph with metric costs. Lecture Notes Comp Sci 5171:233–246

Sebő A, Vygen J (2014) Shorter tours by nicer ears: $7/5$-approximation for graphic tsp, $3/2$ for the path version, and $4/3$ for two edge-connected subgraphs. Combinatorica. doi:10.1007/s00493-011-2960-3

Simonetti L, Protti F, Frota Y (2011) New branch-and-bound algorithms for k-cardinality tree problems. Electron Notes Discret Math 37:27–32

Simonetti L, da Cunha AS, Lucena A (2013) Polyhedral results and a branch-and-cut algorithm for the k-cardinality tree problem. Math Program 142:511–538

Sun Y (2013) Theoretical and experimental studies on the minimum size 2-edge-connected spanning subgrpah problem, Master thesis, University of Ottawa, Ottawa, Canada

Zhou R, Liu C, Xu Yu J, Liang W, Chen B, Li J (2012) Finding maximal k-edge-connected subgraphs from a large graph. Proceedings of the 15th International Conference on Extending Database Technology 480–491