Integer Programming Formulations for Minimum Spanning Forests and Connected Components in Sparse Graphs

Conference Paper · December 2014

DOI: 10.1007/978-3-319-12691-3_46

CITATIONS
7

READS
3,076

2 authors, including:

Mehdi Golari
Amazon
6 PUBLICATIONS **132** CITATIONS

SEE PROFILE

Integer Programming Formulations for Minimum Spanning Forests and Connected Components in Sparse Graphs

Neng Fan(✉) and Mehdi Golari

Systems and Industrial Engineering Department, University of Arizona
Tucson, AZ 85721, USA
{nfan,golari}@email.arizona.edu

Abstract. In this paper, we first review several integer programming formulations for the minimum spanning tree problem, and then adapt these formulations for solving the minimum spanning forest problem in sparse graphs. Some properties for the spanning forest and connected components are studied, and we then present the integer programming formulation for finding the largest connected component, which has been widely used for network vulnerability analysis.

Keywords: Minimum spanning forest, Minimum spanning tree, Connected components, Network vulnerability analysis, Integer programming

1 Introduction

Given an undirected graph $G = (V, E)$, where V is the vertex set with cardinality $|V| = n$, and E is the edge set. A graph is *connected* if there exists a path between any pair of its vertices. A maximal connected subgraph of G is called its *connected component*.

A connected graph has exactly one connected component, consisting of the whole graph. Generally, the number of connected components in graph G can be computed several approaches:

(a) It can be interpreted as the zeroth Betti number of the graph G, in topological theory (see [1]).
(b) It equals to the multiplicity of 0 as an eigenvalue of the Laplacian matrix of the graph G, in algebraic graph theory.
(c) It is also the index of the first nonzero coefficient of the chromatic polynomial of a graph.

In this paper, we assume that the number m of connected components in graph G is known and given, and the second approach will be used to compute the exact value of m for a given graph for numerical experiments purpose. The Laplacian matrix for graph G is defined as $L := (l_{ij})_{n\times n}$ such that $L = D - A$, where D is the degree matrix and and A is the adjacency matrix of G, or directly expressed as

$$l_{ij} = \begin{cases} deg(v_i), & \text{if } i = j, \\ -1, & \text{if } i \neq j \text{ and } v_i \text{ is adjacent to } v_j, \\ 0, & \text{otherwise,} \end{cases}$$

where $deg(v_i)$ is degree of the vertex v_i.

A *tree* is a connected graph without cycles. A subgraph of a connected graph G is its *spanning tree* T if it is a tree connecting all vertices of G. A *spanning forest* F of a disconnected graph G is a collection of spanning trees, one for each of its connected components. Given a graph G with weight matrix $W = (w_{ij})_{n\times n}$, where $w_{ij} \geq 0$ is the weight of edge $(i, j) \in E$ such that $w_{ij} = w_{ji}$ and $w_{ii} = 0$ for all $i = 1, \cdots, n$, a *minimum spanning forest (MSF)* of a weighted graph is a spanning forest such that the sum of its edge-weight is minimum. When the number of connected components in G is 1, the MSF problem reduces to the well-known *minimum spanning tree (MST)* problem.

There are two algorithms commonly used to find the MST in a graph: Prim's algorithm and Kruskal's algorithm. If the graph is not connected, the Kruskal's algorithm finds the MSF directly. Running Prim's algorithm for each connected component, we can also find the MSF. The Kruskal's algorithm has computational complexity as $O(|E|\log|V|)$ while Prim's algorithm can have $O(|E|+|V|\log|V|)$. For MSF problem, in [2,3], more efficient algorithms to find MSF runs in $O(\log|V|)$ time were designed. These are all heuristic algorithms. Recently, because of advances in high-performance computing and efficient algorithms found in commercial and open-source software, integer programming (IP) methods are more broadly used for obtaining exact solutions to many combinatorial optimization problems. For example, Pop [4] presented a survey on different integer programming formulations of the generalized MST problem.

In this paper, we study the problem of finding the minimum spanning forest in a given graph (mainly in sparse graphs, i.e., graphs with only a few edges) by integer programming. Besides the MST problem, the MSF problem also has close relations with other classic combinatorial optimization problems, such as finding the largest connected component of a graph, interdependent power network disruptor problem for vulnerability analysis, etc. Therefore, we also discuss these problems and their relations to network analysis.

The remainder of this paper is organized as follows. In Section 2, several IP formulations for MSF problem will be presented and discussed with their corresponding algorithms. Section 3 will discuss problems related to largest connected component in a network. Finally, Section 4 concludes this paper.

2 IP Formulations for the MSF Problem

Given an undirected graph $G = (V,E)$ with a nonnegative weight matrix W, let $x_{ij} \in \{0,1\}$ denote whether edge $(i, j) \in E$ is selected into the forest if $x_{ij} = 1$ or not if $x_{ij} = 0$. Let x denote the vector formed by x_{ij}'s for all $(i, j) \in E$. The MSF found by optimal x^*, denoted F^*, will be a subgraph $F^* = (V,E^*)$, where $E^* = \{(i, j) \in E : x^*_{ij} = 1\}$ denotes the selected edge into the spanning forest.

2.1 IP formulations for MST problem

If this graph is connected, i.e., $m = 1$, the MSF problem becomes the MST problem. Recall that there are several basic properties related to tree:

Proposition 1. *A tree is an undirected simple graph T (with n vertices) that satisfies any one of the following equivalent conditions:*

(i) T has no simple cycles and has $n-1$ edges;
(ii) T is connected and has $n-1$ edges;

Based on each of these properties, the following IP formulations are presented correspondingly for the MST problem:

(1) Subtour elimination formulation is based on the fact that G has no simple cycles and has $n-1$ edges (Proposition 1(i)):

$$\textbf{[MST1]} \quad \min_{x} \sum_{(i,j)\in E} w_{ij}x_{ij}$$
$$s.t. \begin{cases} \sum_{(i,j)\in E} x_{ij} = n-1 \\ \sum_{(i,j)\in E(S)} x_{ij} \le |S|-1,\ \forall S \subset V, S \neq V, S \neq \emptyset \\ x_{ij} \in \{0,1\},\ \forall (i,j) \in E \end{cases}$$

where $E(S) \subset E$ is a subset of edges with both ends in subset $S \subset V$. Constraint $\sum_{(i,j)\in E(S)} x_{ij} \le |S|-1$ ensures that there is no cycles in subset S.

(2) Cutset formulation based on the fact that G is connected and has $n-1$ edges (Proposition 1(ii)):

$$\textbf{[MST2]} \quad \min_{x} \sum_{(i,j)\in E} w_{ij}x_{ij}$$
$$s.t. \begin{cases} \sum_{(i,j)\in E} x_{ij} = n-1 \\ \sum_{(i,j)\in \delta(S)} x_{ij} \ge 1,\ \forall S \subset V, S \neq V, S \neq \emptyset \\ x_{ij} \in \{0,1\},\ \forall (i,j) \in E \end{cases}$$

where the cutset $\delta(S) \subset E$ is a subset of edges with one end in S and the other end in $V \setminus S$. Constraints $\sum_{(i,j)\in \delta(S)} x_{ij} \ge 1$ ensures that subsets S and $V \setminus S$ are connected.

(3) Single-commodity flow formulation. Additionally, if considering the Proposition 1(ii), the single-commodity flow constraints can be used to ensure the connectivity of a tree found by x. Correspondingly, the following formulation can be presented:

$$\textbf{[MST3]} \quad \min_{x,f} \sum_{(i,j)\in E} w_{ij}x_{ij}$$
$$s.t. \begin{cases} \sum_{(i,j)\in E} x_{ij} = n-1 \\ \sum_j f_{j1} - \sum_j f_{1j} = n-1 \\ \sum_j f_{ji} - \sum_j f_{ij} = 1, \forall i \in V, i \neq 1 \\ f_{ij} \le (n-1)x_{ij},\ f_{ji} \le (n-1)x_{ij}, \forall (i,j) \in E \\ f_{ij} \ge 0,\ \forall (i,j) \in E \cup E' \\ x_{ij} \in \{0,1\},\ \forall (i,j) \in E \end{cases}$$

where the decision variable f_{ij} denotes the flow from vertex v_i to vertex v_j (f is vectors for flows on all possible pairs in two directions). The constraints in **[MST3]** to find the MST include (in order): the total number of selected edges into the tree is $n-1$; the total number of flows from vertex v_1 is $n-1$; each vertex consumes exactly one unit of flow; the flow from v_i to v_j or from v_j to v_i may have flow if edge (i,j) is chosen into the spanning tree, and have upper bounds $n-1$; the flow is nonnegative, where E' denotes set of edges with opposite direction for edges in E; and binary choice for edge selection into the spanning tree. Through this formulation, $n-1$ edges selected by x connects all n vertices, and by Proposition 1(ii), x finds the MST.

Let P_{sub} be polyhedron formed by constraints of **[MST1]**, i.e.,

$$P_{sub} = \{x \in \mathbb{R}^{|E|} : 0 \le x_{ij} \le 1, \forall (i,j) \in E, \sum_{(i,j)\in E} x_{ij} = n-1, \sum_{(i,j)\in E(S)} x_{ij} \le |S|-1, \ \forall S \subset V, S \neq V, S \neq \emptyset\}.$$

Similarly,

$$P_{cut} = \{x \in \mathbb{R}^{|E|} : 0 \le x_{ij} \le 1, \forall (i,j) \in E, \sum_{(i,j)\in E} x_{ij} = n-1, \sum_{(i,j)\in \delta(S)} x_{ij} \ge 1, \ \forall S \subset V, S \neq V, S \neq \emptyset\}$$

is the polyhedron formed by constraints of **[MST2]**, and

$$P_{flow} = \{x \in \mathbb{R}^{|E|} : 0 \le x_{ij} \le 1, \forall (i,j) \in E, \text{ all constraints in } \textbf{[MST3]} \text{ except the integraty conditions}\}$$

is the polyhedron formed by constraints of **[MST3]**. There are some properties regarding these two sets: (i) The polyhedron P_{sub} has integral extreme points; (ii) $P_{sub} \subseteq P_{cut} \subseteq P_{flow}$. These properties can be used in solving IP formulations, and we refer to [5,6] for the first property for more details, and [7,8,6] for the second property.

Both formulations **[MST1]** and **[MST2]** have exponential number of constraints in the worst-case, while formulation **[MST3]** has polynomial number of constraints. Additionally, in [9], Martin presented a reformulation with polynomial number of constraints for the MST problem. This method was also used in [10] by Yannakakis, who attributed to [9], and was recently mentioned in [6,11].

(4) Martin's formulation:

$$\textbf{[MST4]} \quad \min_{x,y} \sum_{(i,j)\in E} w_{ij}x_{ij}$$

$$s.t. \begin{cases} \sum_{(i,j)\in E} x_{ij} = n-1 \\ y^k_{ij} + y^k_{ji} = x_{ij}, \ \forall (i,j) \in E, k \in V \\ \sum_{k\in V\setminus\{i,j\}} y^j_{ik} + x_{ij} = 1, \ \forall (i,j) \in E \\ x_{ij}, y^k_{ij}, y^k_{ji} \in \{0,1\}, \ \forall (i,j) \in E, k \in V \end{cases}$$

where the decision variable $y_{ij}^k \in \{0,1\}$ for each $(i,j) \in E, k \in V$ denotes that edge (i,j) is in the spanning tree and vertex k is on the side of j (i.e., vertex k is within the resulted part containing j after removal of (i,j) from the spanning tree) if $y_{ij}^k = 1$, and edge (i,j) is on the tree and k is not on side of j or if edge (i,j) is not in the tree if $y_{ij}^k = 0$.
Here, in this formulation, the first constraint ensures that there are exactly $n-1$ edges selected into the spanning tree. The second constraint for $(i,j) \in E, k \in V$ grantees that if $(i,j) \in E$ is selected into the tree ($x_{ij} = 1$), any vertex $k \in V$ must be either on the side of j ($y_{ij}^k = 1$) or on the side of i ($y_{ji}^k = 1$); if $(i,j) \in E$ is not in the tree ($x_{ij} = 0$), any vertex k cannot be on the side of j nor i ($y_{ij}^k = y_{ji}^k = 0$). The third constraint for $(i,j) \in E$ ensures that if $(i,j) \in E$ is in the tree ($x_{ij} = 1$), edges (i,k) who connects i are on the side of i ($y_{ik}^j = y_{ij}^k = 0, y_{ij}^k = 1$); if $(i,j) \in E$ is not in the tree ($x_{ij} = 0$), there must be an edge (i,k) such that j is on the side of k ($y_{ik}^j = 1$ for some k). These constraints with the the binary requirements of all x_{ij}'s and y_{ij}^k's ensures that E^* presents a spanning tree for a connected graph, and they are proposed by Martin in [9] and discussed in [11].

2.2 IP formulations for MSF problem

In the following, we will first generalize these formulations for a general graph with m connected components, and therefore, a MSF can be found.

Without loss of generality, assume that the m connected components of G have vertex sets as $V_1, V_2 \cdots, V_m$, respectively. Definitely, these sets are disjoint and their union is V. Correspondingly, assume E_k ($k = 1,2,\cdots,m$) is the edge set induced by vertices in V_i from graph G. Thus, each connected component of G can be considered as a subgraph $G_k = (V_k, E_k)$ ($k = 1,2,\cdots,m$) of G. Therefore, a spanning forest of G consists of spanning trees of all G_k's, and one for each G_k. The following theorem is a direct result.

Theorem 1. *For the graph G with m connected components, denoted by $G_1, G_2, \cdots, G_m$, the forest F^*, consisting of spanning trees $T_1^*, T_2^*, \cdots, T_m^*$, is a minimum spanning forest of G if and only if each T_i^* is a minimum spanning tree for subgraph G_i ($i = 1,2,\cdots,m$). Furthermore, the number of edges in a spanning forest of G is $n-m$.*

From this theorem, if a simple graph has no simple cycles and exactly $n-m$ edges, where m is the number connected components in the graph, it is a forest. Now, we adapt subtour elimination constraints and cutset constraints to find the MSF in a graph with m connected components.

Considering $S \subset V, S \neq \emptyset, S \neq V$, there are three cases for the subtour elimination constraints and cutset constraints:

(i) if $S \subset V_i$, $\sum_{(i,j)\in E(S)} x_{ij} \leq |S| - 1$; $\sum_{i\in S, j\in V\setminus S} x_{ij} \geq 1$;
(ii) if $S \subset V_{i_1} \cup V_{i_2} \cup \cdots \cup V_{i_k}$ ($2 \leq k \leq m$) and $S \cap V_{i_1} \neq \emptyset, \cdots, S \cap V_{i_k} \neq \emptyset$, $\sum_{(i,j)\in E(S)} x_{ij} \leq |S| - k$; $\sum_{i\in S, j\in V\setminus S} x_{ij} \geq k$;
(iii) if $S = V_{i_1} \cup V_{i_2} \cup \cdots \cup V_{i_k}$ ($1 \leq k < m$), $\sum_{(i,j)\in E(S)} x_{ij} \leq |S| - k$; $\sum_{i\in S, j\in V\setminus S} x_{ij} \geq 0$.

The first set of constraints in case (i) for this type of subsets are enough to ensure that x_{ij}'s generate all spanning trees for all connected components, and thus, they also ensure a spanning forest for G. Therefore, for a general subset $S \subset V$, if we can make constraints in cases (ii) and (iii) be redundant, a slight modification of formulations **[MST1]** and **[MST2]** will find the MSF of G.

For subtour elimination constraints, the case (ii) can have the relations as $\sum_{(i,j)\in S} x_{ij} \leq |S| - k < |S| - 1$, and the case (iii) also has $\sum_{(i,j)\in S} x_{ij} \leq |S| - k \leq |S| - 1$. Therefore, changing them to $\sum_{(i,j)\in S} x_{ij} \leq |S| - 1$ will make them be redundant for these types of subsets S's. An IP formulation, similar to **[MST1]** for the MST problem, is presented for the MSF problem:

$$\begin{aligned}
\textbf{[MSF1]} \quad \min \quad & \sum_{(i,j)\in E} w_{ij}x_{ij} \\
s.t. \quad & \sum_{(i,j)\in E} x_{ij} = n - m \\
& \sum_{(i,j)\in E(S)} x_{ij} \leq |S| - 1, \ \forall S \subset V, S \neq V, S \neq \emptyset \\
& x_{ij} \in \{0,1\}, \ \forall (i,j) \in E
\end{aligned}$$

where the first constraint ensures that there are $n - m$ edges in the spanning forest.

For the cutset constraints, using the one $\sum_{i\in S, j\in V\setminus S} x_{ij} \geq 1$ in case (i) to replace the one $\sum_{i\in S, j\in V\setminus S} x_{ij} \geq k > 1$ will make the constraints be redundant for S's in case (ii). However, the one $\sum_{i\in S, j\in V\setminus S} x_{ij} \geq 0$ in case (iii) is stronger than the one in case (i). Let $a_{ij} = 1$ if the pair $(i,j) \in E$, and $a_{ij} = 0$ if $(i,j) \notin E$ for any $i, j \in V$ (a_{ij} is actually an element in adjacency matrix A). By introducing the variable, the case (iii) can be fixed as follows:

$$\begin{aligned}
\textbf{[MSF2]} \quad \min \quad & \sum_{(i,j)\in E} w_{ij}x_{ij} \\
s.t. \quad & \sum_{(i,j)\in E} x_{ij} = n - m \\
& \sum_{i\in S, j\in V\setminus S, (i,j)\in E} x_{ij} \geq \max_{i\in S, j\in V\setminus S} a_{ij}, \ \forall S \subset V, S \neq V, S \neq \emptyset \\
& x_{ij} \in \{0,1\}, \ \forall (i,j) \in E
\end{aligned}$$

where the right-hand-side $\max_{i\in S, j\in V\setminus S} a_{ij}$ of the second constraints will be 1 for S chosen as in cases (i) and (ii), and 0 for S in case (iii).

From arguments discussed above in cases (i)-(iii), both formulations **[MSF1]** and **[MSF2]** will find the minimum spanning forest directly without finding the connected components. In the following, we will propose several IP formulations to find the connected components in a graph first and then to find the minimum spanning tree of each component, which can form the minimum spanning forest as shown in Theorem 1.

Next, we present a formulation to find all components such that vertices in the same component are connected and vertices in different components are not connected.

Theorem 2. *Assume that graph G has m connected components $G_1, G_2, \cdots, G_m$. Let $z_{ik} \in \{0,1\}$ be a decision variable such that $z_{ik} = 1$ if vertex v_i is in set V_k of component G_i, for all $v_i \in V, k = 1,2,\cdots,m$. Additionally, let u_{ij} be a decision variable such that $u_{ij} = 1$ if vertices v_i, v_j are in the same component, and $u_{ij} = 0$ otherwise, for all $v_i, v_j \in V, i \neq j$. The formulation*

$$u_{ij} \geq a_{ij}, \ \forall i,j \in V, i \neq j \tag{1a}$$

$$\begin{cases} u_{i_1 i_2} + u_{i_2 i_3} - u_{i_1 i_3} \leq 1, \\ u_{i_1 i_2} - u_{i_2 i_3} + u_{i_1 i_3} \leq 1, \\ -u_{i_1 i_2} + u_{i_2 i_3} + u_{i_1 i_3} \leq 1, \end{cases} \quad \forall \text{ triplets } i_1, i_2, i_3 \in V \tag{1b}$$

$$u_{ij} = \sum_{k=1}^{m} z_{ik} z_{jk}, \ \forall i,j \in V, i \neq j \tag{1c}$$

$$\sum_{k=1}^{m} z_{ik} = 1, \ \forall i \in V \tag{1d}$$

$$z_{ik} \in \{0,1\}, u_{ij} \in \{0,1\}, \ \forall i,j \in V, k = 1,\cdots,m \tag{1e}$$

will find all m connected components in graph G, i.e., the vertex set V_k of the kth connected component can be expressed as $V_k = \{v_i \in V : z_{ik} = 1\}$, and vertices in V_k should be connected ($k = 1,2,\cdots,m$).

Proof. Inequalities in (1a) imply that v_i, v_j are in the same component (i.e., $u_{ij} = 1$) if there exists an edge $(i,j) \in E$ (i.e., $a_{ij} = 1$) and $u_{ij} = 0$ otherwise. Inequalities (1b) imply that for all triplets $i_1, i_2, i_3 \in V$, if i_1, i_2 are in the same component and i_2, i_3 are in the same component, then i_1, i_3 are in the same component. Equalities in (1c) ensure that if i, j are partitioned into the same component, $u_{ij} = 1$ and otherwise $u_{ij} = 0$. Equalities in (1d) ensure that each vertex is partitioned into one connected component, while the last ones ensure the binary choices of z_{ik}'s and u_{ij}'s.

□

In Theorem 2, we use the pairwise connectivity to find all connected components. In the formulation in Theorem 2, a feasible solution to constraints (1a)-(1b) with $u_{ij} \in \{0,1\}$ for all $i,j \in V, i \neq j$ can decide every pair of vertices in the same component or not. By all values of u_{ij}'s from this feasible solution, constraints (1c)-(1d) with $z_{ik} \in \{0,1\}$ for all $i \in V, k = 1,\cdots,m$ can then divide the graph into components directly.

Also, by similar method in **[MST3]**, we can propose another approach by the multi-commodity flow to find all m connected components by sending m types of flows. By Martin constraints, a MST will be found for each connected component and thus a MSF can be found directly. Therefore, different from **[MSF1]** and **[MSF2]**, both of

which have exponential number of constraints, the following formulation **[MSF3]** has polynomial number of constraints.

$$\textbf{[MSF3]} \quad \min_{x,y,z,u} \sum_{(i,j)\in E} w_{ij}x_{ij} \tag{2a}$$

$$s.t. \text{ Constraints in (1a)-(1e)} \tag{2b}$$

$$\sum_{(i,j)\in E} x_{ij} = n-m \tag{2c}$$

$$y_{ij}^{kl} + y_{ji}^{kl} = x_{ij}z_{lk},\ \forall (i,j)\in E, l\in V, k=1,\cdots,m \tag{2d}$$

$$\sum_{l\in V\setminus\{i,j\}} y_{il}^{kj} z_{lk} + x_{ij} = 1,\ \forall (i,j)\in E, k=1,\cdots,m \tag{2e}$$

$$y_{ij}^{kl}, y_{ji}^{kl} \le \min\{z_{ik}, z_{jk}, z_{lk}, x_{ij}\},\ \forall (i,j)\in E, l\in V, k=1,\cdots,m \tag{2f}$$

$$x_{ij}, y_{ij}^{kl}, y_{ji}^{kl} \in \{0,1\},\ \forall (i,j)\in E, l\in V, k=1,\cdots,m \tag{2g}$$

The objective (2a) is the same as those one in **[MSF1]** and **[MSF2]** to minimize the cost associated the spanning forest. Constraint (2c) ensures the the number of chosen edges into the spanning forest. Constraints in (2d)-(2f) ensure that the chosen edges by x_{ij} imply a spanning tree in each connected component by Martin constraints.

Proposition 2. *In graph G, let $N(i) = \{v_j \in V : (i,j)\in E\}$ be the set of neighbors of vertex v_i. If v_i is not an isolated vertex, i.e., $N(i)\neq\emptyset$, $\sum_{j\in N(i)} x_{ij} \ge 1$ is a valid inequality for above IP formulations for the MSF problem.*

Proposition 3. *The anticycle inequalities*

$$z_{11} = 1, z_{21} + z_{22} = 1, \cdots, z_{m1} + z_{m2} + \cdots + z_{mm} = 1, \tag{3}$$

can be added to **[MSF3]** *to reduce the computational complexity.*

The m equalities in (3) denote that: the first vertex is in the 1st component; the second vertex can be either in the 1st or 2nd component; similarly, the mth vertex can be in one of the first m components.

Remark. If there exist some isolated vertices, the formulations in **[MSF1], [MSF2], [MSF3]** can still find the minimum spanning forests in sparse graphs.

3 Largest connected component

Following the method for the formulation **[MSF3]**, the connected component G_k has number of vertices as $|V_k| = \sum_{i:v_i\in V} z_{ik}$. Now based on this method, we present an IP formulation for finding the largest connected component (LCC) in a graph G with m

components:

$$\textbf{[LCC]} \quad \max \sum_{k=1}^{m} \delta_k \sum_{i:v_i \in V} z_{ik} \tag{4a}$$

$$s.t. \sum_{k=1}^{m} \delta_k = 1 \tag{4b}$$

$$\text{Constraints in (1a)-(1e)} \tag{4c}$$

$$\delta_k \in \{0,1\},\ k = 1,2,\cdots,m \tag{4d}$$

where the objective with constraint (4b) and binary choices of δ_k's ensures that for some k, $\delta_k = 1$ and the corresponding $|V_k|$ is the largest one among all components. The one k with $\delta_k = 1$ indicates the largest connected component, and the objective $\sum_{k=1}^{m} \delta_k \sum_{i:v_i \in V} z_{ik}$ gives the size of the components as all other δ_k's are 0. Similarly, inequalities in (3) can be added to **[LCC]** to reduce the computational complexity.

The largest connected component has been used as a measurement for power network's vulnerability in case of cascading failures. In [12], the *interdependent power network disruptor* (IPND) problem was presented to find the minimum size of the largest connected component in case of node failures. Following the notations and definitions in [12], the interdependent network can be defined as a system, consisting of two graphs $G_s = (V_s, E_s)$ and $G_c = (V_c, E_c)$ and their interdependencies E_{sc}, where V_s, V_c are node sets and E_s, E_c, E_{sc} are edge set (called *intra-links*). The set E_{sc} includes *inter-links* coupling G_s and G_c, and is denoted by $E_{sc} = \{(u,v) : u \in V_s, v \in V_c\}$. The IPND problem is to find a subset $T \subseteq V_s$ of a fix size, such that the size of the largest connected component of G_s after the cascading failures caused by removal of T is minimized.

4 Conclusions

In this paper, we first reviewed several integer programming formulations for the minimum spanning tree problem, and then adapted these formulations for solving the minimum spanning forest problem in sparse graphs. Some properties for the spanning forest and connected components are studied, and we then present the integer programming formulation for finding the largest connected component, which has been widely used for network vulnerability analysis. Further research directions include studying the polyhedral properties of constraints in different integer programming formulations, and also integer programming formulations for network vulnerability analysis, especially by largest connected component, for social networks, and related combinatorial optimization problems, such as k-minimum spanning tree problem or K-cardinality tree problem, subgraph isomorphism problem, etc.

References

1. Carlsson, G.: Topology and data. Bulletin of the American Mathematical Society 46(2), 255–308 (2009)

2. Cole, R., Klein, P.N., Tarjan, R.E.: Finding minimum spanning forests in logarithmic time and linear work using random sampling. Proceeding SPAA '96 Proceedings of the eighth annual ACM symposium on Parallel algorithms and architectures, 243–250 (1996).
3. Pettie, S., Ramachandran, V.: A randomized time-work optimal parallel algorithm for finding a minimum spanning forest. SIAM Journal on Computing 31(6), 1879–1895 (2002)
4. Pop, P.C.: A survey of different integer programming formulations of the generalized minimum spanning tree problem. Carpathian J. Math. 25(1), 104–118 (2009)
5. Edmonds, J.: Matroids and the greedy algorithm. Mathematical Programming 1, 127–136 (1971)
6. Conforti, M., Cornuéjols, G., Zambelli, G.: Extended formulations in combinatorial optimization. 4OR(8), 1–48 (2010)
7. Myung, Y. S., Lee C. H., Tcha D. W.: On the generalized minimum spanning tree problem. Networks 26(4) 231–241 (1995)
8. Feremans, C., Labbe, M., Laporte, G.: A comparative analysis of several formulations for the generalized minimum spanning tree problem. Networks 39(1) 29–34 (2002)
9. Martin, R.K.: Using separation algorithms to generate mixed integer model reformulations. Operations Research Letters 10, 119–128 (1991)
10. Yannakakis, M.: Expressing combinatorial optimization problems by linear programs. Journal of Computer and System Sciences 43(3), 441–466 (1991)
11. Kaibel, V., Pashkovich, K., Theis, D.O.: Symmetry matters for the sizes of extended formulations. Integer Programming and Combinatorial Optimization (IPCO 2010), 135–148 (2010)
12. Nguyen, D.T., Shen, Y., Thai, M.T.: Detecting critical nodes in interdependent power networks for vulnerability assessment. IEEE Trans. Smart Grid 4(1), 151–159 (2013)