Computers and Operations Research 81 (2017) 322–332

Contents lists available at ScienceDirect

Computers and Operations Research

journal homepage: www.elsevier.com/locate/cor

A branch-and-cut algorithm for the minimum branch vertices spanning tree problem

Selene Silvestri[a,*], Gilbert Laporte[b], Raffaele Cerulli[c]

[a] Department of Computer Science, University of Salerno, Via Giovanni Paolo II 132, 84084 Fisciano, Italy
[b] CIRRELT, HEC Montréal, 3000 chemin de la Côte-Sainte-Catherine, Montréal, Canada H3T 2A7
[c] Department of Mathematics, University of Salerno, Via Giovanni Paolo II 132, 84084 Fisciano, Italy

ARTICLE INFO

Article history:
Received 26 November 2015
Revised 23 September 2016
Accepted 9 November 2016
Available online 15 November 2016

Keywords:
Spanning tree
Branch vertices
Branch-and-cut

ABSTRACT

Given a connected undirected graph $G = (V, E)$, the Minimum Branch Vertices Problem (MBVP) asks for a spanning tree of G with the minimum number of vertices having degree greater than two in the tree. These are called *branch vertices*. This problem, with applications in the context of optical networks, is known to be NP-hard. We model the MBVP as an integer linear program, with undirected variables, we derive valid inequalities and we prove that some of these are facet defining. We then develop a hybrid formulation containing undirected and directed variables. Both models are solved with branch-and-cut. Comparative computational results show the superiority of the hybrid formulation.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

Given a connected undirected graph $G = (V, E)$, with $n = |V|$ vertices and $m = |E|$ edges, the Minimum Branch Vertices Problem (MBVP) aims to find a spanning tree T of G with the minimum number of branch vertices, i.e. vertices with a degree greater than two. For the input graph given on the left of Fig. 1, we depict two spanning trees with different numbers of branch vertices. The spanning tree in the middle has one branch vertex and the one on the right has two. The best known application of MBVP arises in the context of optical networks. In such networks, an optical signal has to be split whenever it enters a node having degree greater than two. The split has to be performed using an appropriate light switch. These switches must be located at all the branch vertices, which can significantly increase the cost of the network.

The MBVP was introduced by Gargano et al. [8], who proved it to be NP-hard. Since then, the problem has been extensively investigated by several authors [2–4,11,13,19–21]. Carrabs et al. [2] consider four IP formulations. The first formulation contains the well-known Dantzig et al. [6] subtour elimination constraints. Due to the exponential number of constraints, the authors consider than this formulation is not suitable for solving instances of significant size, but they solve it in a Lagrangian relaxation fashion. The second formulation is the most studied. It guarantees connectivity by sending one unit of flow from a source vertex to every other vertex of the graph. The third formulation is based on a multi-commodity flow. The fourth formulation makes use of the Miller–Tucker–Zemlin subtour elimination constraints [14]. Finally, Marín [11] presents a branch-and-cut algorithm based on a strengthened single commodity flow formulation. The author also provides a two-stage heuristic to reduce the computational time and to produce good feasible solutions when the optimum cannot be found within a reasonable time.

Our aim is to develop new formulations and a polyhedral-based exact branch-and-cut algorithm for the MBVP. The remainder of the paper is organized as follows. In Section 2, the problem is formulated as an integer linear program with variables associated with the edges of G. In this section, we also investigate some properties of the problem and we analyze its LP relaxation. In Section 3, we derive the dimension of the polyhedron as well as some facet related results, and we introduce some valid inequalities. In Section 4, we present a reformulation based on a directed graph and we adapt several properties of the problem and some valid inequalities to this formulation to obtain a hybrid formulation. The branch-and-cut algorithm is described in Section 5. Comparative computational results and conclusions are presented in Section 6 and 7, respectively.

2. Undirected formulation and properties

Given a connected undirected graph $G = (V, E)$, the MBVP can be formulated as an integer linear program (ILP) as follows. Let x_e

* Corresponding author at: CIRRELT, HEC Montréal, 3000 chemin de la Côte-Sainte-Catherine, Montréal, Canada H3T 2A7.
E-mail addresses: selene.silvestri@gmail.com (S. Silvestri), gilbert.laporte@cirrelt.ca (G. Laporte), raffaele@unisa.it (R. Cerulli).

http://dx.doi.org/10.1016/j.cor.2016.11.010
0305-0548/© 2016 Elsevier Ltd. All rights reserved.

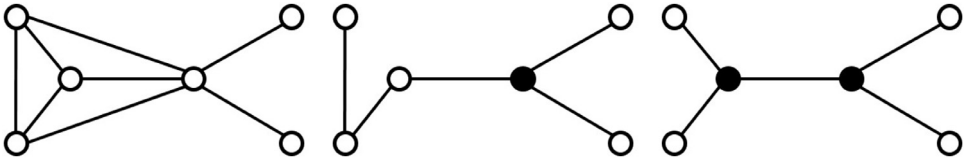

Fig. 1. For a given graph on the left, two spanning trees with one and two branch vertices.

be a binary variable equal to 1 if and only if edge $e \in E$ belongs to the spanning tree T. For the sake of simplicity, in the following we will refer to variables x_e, which are variables associated with the edges of the undirected graph G, as *undirected variables*. Moreover, let y_v be a binary variable equal to 1 if and only if vertex $v \in V$ has degree greater than or equal to 3 in T, i.e. v is a branch vertex. In addition, for $S \subset V$, define $E(S) = \{e = (v,u) \in E : v,u \in S\}$ and $\delta(S) = \{e = (v,u) \in E : v \in S,\ u \in V \setminus S\}$. If $S = \{v\}$, we simply write $\delta(v)$ instead of $\delta(\{v\})$. Moreover, we denote by $d(v)$ the degree of vertex v, that is $d(v) = |\delta(v)|$, $v \in V$. We write $\delta_{G'}(v)$ and $d_{G'}(v)$ to denote the set of incident edges and the degree of vertex v, respectively, when they refer to a subgraph G' of G. The ILP formulation is then

$$\text{minimize } z = \sum_{v \in V} y_v \tag{1}$$

subject to

$$\sum_{e \in E(S)} x_e \le |S| - 1 \quad S \subset V,\ |S| \ge 3 \tag{2}$$

$$\sum_{e \in E} x_e = n - 1 \tag{3}$$

$$\sum_{e \in \delta(v)} x_e - 2 \le (d(v) - 2) y_v \quad v \in V \tag{4}$$

$$x_e \in \{0,1\} \quad e \in E \tag{5}$$

$$y_v \in \{0,1\} \quad v \in V. \tag{6}$$

In this formulation, constraints (2) are the classical Dantzig et al. [6] subtour elimination constraints. They guarantee that the edges in the solution cannot form cycles. Constraint (3) forces the selection of exactly $n-1$ edges. Constraints (4) are logical constraints linking the binary variables x_e with the binary variables y_v. They guarantee that v is a branch vertex whenever at least three edges incident to it are selected. Even if they do not explicitly force $y_v = 0$ when $\sum_{e \in \delta(v)} x_e \le 2$ holds, due to the objective function used, this condition is satisfied for any optimal solution. The objective (1) requires the minimization of the number of branch vertices. This formulation is already known (see Melo et al. [12]) and represents a stronger version of one of the mathematical formulations proposed by Carrabs et al. [2]. Note that variables y_v will represent exactly a set of branch vertices if we add the following constraints

$$2y_v \le \sum_{e \in \delta(v)} x_e - 1 \quad v \in V. \tag{7}$$

Constraints (7) together with constraints (4) ensure that y_v, $v \in V$, is equal to 1 if and only if v is a branch vertex. Let P_{STP} denote the spanning tree polytope, that is the linear relaxation of (5) plus constraints (2) and (3). Moreover, let P_L denotes the intersection of the spanning tree polytope P_{STP} with the linear relaxation of (6), plus constraints (4) and (7).

2.1. Spanning tree properties

We now present some properties that a spanning tree must satisfy and we make some observations that will allow us to preprocess the instances. For a given vertex v, we can write the set of incident edges $\delta(v)$ as

$$\delta(v) = \delta^L(v) \cup \delta^I(v),$$

where $\delta^L(v) = \{(v,u) \in \delta(v) : d(u) = 1\}$ and $\delta^I(v) = \{(v,u) \in \delta(v) : d(u) > 1\}$. As Marín [11] observed, each edge belonging to the set $\delta^L(v)$, for a given vertex v, must belong to an optimal tree T:

$$x_e = 1 \quad v \in V, \quad e \in \delta^L(v). \tag{8}$$

Moreover,

$$y_v = 0 \quad v \in V :\ d(v) \le 2$$

$$y_v = 1 \quad v \in V :\ d(v) \ge 3,\ |\delta^L(v)| \ge 2.$$

Note that for each vertex v such that $|\delta^L(v)| = 1$, constraints (4) and (7) become respectively

$$\sum_{e \in \delta^I(v)} x_e - 1 \le (|\delta^I(v)| - 1) y_v \quad v \in V :\ |\delta^L(v)| = 1 \tag{9}$$

$$y_v \le \sum_{e \in \delta^I(v)} x_e - 1 \quad v \in V :\ |\delta^L(v)| = 1. \tag{10}$$

Constraints (9) come from (4) by simply replacing some constants. Constraints (10) derive from the observation that since $|\delta^L(v)| = 1$, v is not a branch vertex if at most one edge belonging to $\delta^I(v)$ is selected. To ensure the connectivity property, the inequalities

$$\sum_{e \in \delta^I(v)} x_e \ge 1 \quad v \in V : |\delta^I(v)| > 0 \tag{11}$$

must be satisfied.

Marín [11] defines a *bridge* as an edge $e \in E$ such that the graph $(V, E\backslash\{e\})$ becomes disconnected and defines a 2-*cocycle* as a set of two edges $\{e, f\} \subset E$ such that the graph $(V, E\backslash\{e, f\})$ becomes disconnected, but e and f are not bridges. It is easy to see that all bridges of a connected graph must belong to the edge set of any spanning tree:

$$x_e = 1 \quad e \in E :\ (V, E \setminus \{e\}) \text{ is disconnected.} \tag{12}$$

Moreover, at least one of the edges of a 2-cocycle set must belong to any feasible solution:

$$x_e + x_f \ge 1 \quad e, f \in E :\ \{e, f\} \text{ is a 2-cocycle.} \tag{13}$$

Note that all edges belonging to the set $\bigcup_{v \in V} \delta^L(v)$ are particular bridges. Removing any one of them isolates a vertex, therefore constraints (8) represent a subset of constraints (12). Identifying bridges and 2-cocycle sets can be achieved with the algorithm proposed by Schmidt [18] which is used by Marín [11].

In graph theory [1] a *cut vertex* is a vertex $v \in V$ such that the graph $G \setminus v = (V \setminus \{v\}, E \setminus \delta(v))$ is disconnected. Let $\bar{c}_v$ be the number of connected components of the graph $G \setminus v$ and let $C_i(v) = (V_{C_i}(v), E_{C_i}(v))$, $i = 1, \ldots, \bar{c}_v$, be the corresponding connected components, such that $\bigcup_{i=1}^{\bar{c}_v} V_{C_i}(v) = V \setminus \{v\}$ and $\bigcup_{i=1}^{\bar{c}_v} E_{C_i}(v) = E \setminus \delta(v)$.

For a given cut vertex v, we can write the set of incident edges $\delta(v)$ as

$$\delta(v) = \bigcup_{i=1}^{\bar{c}_v} \delta^i(v),$$

where $\delta^i(v) = \{(v,u) \in \delta(v) : u \in V_{C_i}(v)\}$. Gargano et al. [8] observed that any cut vertex v such that $G \setminus v$ becomes disconnected in at least three connected components is necessarily a branch vertex in any spanning tree of G. Therefore, if we denote V_B the set of cut vertices, it is easy to see that

$$y_v = 1 \quad v \in V_B : \ \bar{c}_v \geq 3.$$

Moreover, it is easy to see that the following inequalities hold true:

$$\sum_{e \in \delta^i(v)} x_e - 1 \leq (|\delta^i(v)| - 1) y_v \quad v \in V_B : \ \bar{c}_v = 2, \ i = 1, 2 \tag{14}$$

$$\sum_{e \in \delta^i(v)} x_e \geq 1 \quad v \in V_B, \ i = 1, \ldots, \bar{c}_v. \tag{15}$$

Note that inequalities (14) and (15) are a restricted version of (4) and (11) respectively, i.e. they are the same type of constraints but they involve only a subset of the set $\delta(v)$. A connected graph G is *2-connected* if G contains no *cut vertex*. In this paper we call *2-disconnected* a connected graph G such that G contains no cut vertex v such that $G \setminus v$ is *disconnected* into more than two connected components. Note that if all the vertices of a graph are cut vertices or have degree equal to one, the graph is a forest. If the graph is also connected, it is a tree. From now on, given a subgraph $\tilde{G} = (V, \tilde{E})$ of $G = (V, E)$ and given $\tilde{E} \subseteq E$, we will denote by $\tilde{G} \cup \tilde{E}$ the subgraph $(V, \tilde{E} \cup \tilde{E})$ unless it leads to confusion.

Lemma 1. *Let $G = (V, E)$ be a 2-disconnected graph. Then, for any $v \in V$, there exists a spanning tree T in G such that v is not a branch vertex in T.*

Proof. Since G is 2-disconnected, $G \setminus v$ can be connected or disconnected into two connected components. If it is connected, there exists a spanning tree T_v in $G \setminus v$, therefore $T = T_v \cup \{e\}$ is a spanning tree in G, for any $e \in \delta(v)$, and then $d_T(v) = 1$. If $G \setminus v$ is disconnected, there exist two spanning trees T_1 and T_2 in $C_1(v)$ and $C_2(v)$, respectively. Hence, for an arbitrary $e_1 \in \delta^1(v)$ and $e_2 \in \delta^2(v)$, $T = T_1 \cup T_2 \cup \{e_1, e_2\}$ is a spanning tree in G such that $d_T(v) = 2$. □

Assume G is not 2-disconnected and let $v \in V$ be a vertex such that $G \setminus v$ becomes disconnected into $s \geq 3$ connected components. Thus, v is going to be a branch vertex in any spanning tree of G. Melo et al. [12], propose an *obligatory branches based decomposition* and a *cut edges based decomposition* which, starting from $G = (V, E)$, build a new graph $G_0' = (V_0', E_0')$ that becomes disconnected if at least one compulsory branch or one cut edge (bridge) is encountered. Moreover, they provide the following proposition.

Proposition 1. *An optimal solution to the minimum branch vertices problem can be obtained from the solutions of the s connected components of G_0' and its optimal value is*

$$z = |L_0| + \sum_{i=1}^{s} z_i,$$

where L_0 represents the set of compulsory vertices, i.e. vertices that are branches in any spanning tree of G, and z_i, $i = 1, \ldots, s$, the optimal value obtained for the i^{th} connected component.

3. Polyhedral analysis of the undirected formulation

We now study some properties of the following polytope P:

$$P = \text{conv}\left\{(x, y) \in \mathbb{R}^{|E|+|V|} : (x, y) \text{ satisfies } (2) - (6)\right\},$$

and we provide some valid inequalities of the following polytope P^+:

$$P^+ = \text{conv}\left\{(x, y) \in \mathbb{R}^{|E|+|V|} : (x, y) \text{ satisfies } (2) - (7)\right\}.$$

Note that we define P as the convex hull of the solutions satisfying (2)–(6), namely solutions such that y does not necessarily represent exactly a set of branch vertices. Moreover, we define P^+ as the convex hull of the solutions satisfying (2)–(7), namely solutions such that y represent exactly a set of branch vertices. In order to provide our polyhedral results, we need some preliminary results.

Definition 1. A polyhedron $S = \{x \in \mathbb{R}^k : Ax \leq b\}$ is full-dimensional if $dim(S) = k$, where (A, b) is an $h \times (k+1)$ matrix.

Definition 2. Let $X \subseteq \mathbb{R}^k$ be a set of k-dimensional vectors. The *affine hull* of X, aff(X), is the set of all affine combinations of elements of X, namely

$$\text{aff}(X) = \left\{ \sum_{i=1}^{t} \lambda_i \xi_i \mid t \geq 0, \xi_1, \ldots \xi_t \in X, \sum_{i=1}^{t} \lambda_i = 1, \lambda_1, \ldots \lambda_t \in \mathbb{R} \right\} \tag{16}$$

Let $H = \{1, \ldots, h\}$, $H^= = \{i \in H : a^i x = b_i \text{ for all } x \in S\}$ and $H^< = \{i \in H : a^i x < b_i \text{ for some } x \in S\} = H \setminus H^=$, where a^i represents the i^{th} row of A. Let $(A^=, b^=)$ and $(A^<, b^<)$ the corresponding rows of (A, b). According to this notation, the following theorem holds true (see Chapter 3, Theorem 3.17 of Conforti et al. [5]):

Theorem 1. *Let $S = \{x \in \mathbb{R}^k : Ax \leq b\}$ be a nonempty polyhedron. Then*

$$\text{aff}(S) = \{x \in \mathbb{R}^k : A^= x = b^=\} = \{x \in \mathbb{R}^k : A^= x \leq b^=\}.$$

Furthermore, $\dim(S) = k - \text{rank}(A^=)$.

We represent subsets of vertices and edges by their characteristic vectors $y \in \mathbb{B}^n$ and $x \in \mathbb{B}^m$, respectively. Therefore, $V' \subseteq V$ is represented by the vector $y^{V'}$, where $y_v^{V'} = 1$ if $v \in V'$ and $y_v^{V'} = 0$ otherwise, and $E' \subseteq E$ is represented by the vector $x^{E'}$, where $x_e^{E'} = 1$ if $e \in E'$ and $x_e^{E'} = 0$ otherwise. Moreover, $\mathbb{0}$ and $\mathbb{1}$ are the vectors of all zeros and all ones, respectively.

Proposition 2. *Let $G = (V, E)$ be a 2-connected graph. The affine hull of P is given by*

$$\text{aff}(P) = \left\{ (x, y) \in \mathbb{R}^{|E|+|V|} : \sum_{e \in E} x_e = n - 1 \right\}$$

Proof. Let

$$p^t x + q^t y = r \tag{17}$$

be an equality that is satisfied by all points in P.

i. Note that since G is 2-connected, G is a 2-disconnected graph. Due to Lemma 1, for any $v \in V$ there is a spanning tree $T = (V, E_T)$ in G such that v is not branch, therefore $(x^{E_T}, \mathbb{1} \setminus y^{\{v\}})$ belongs to P. Moreover, it is easy to see that $(x^{E_T}, \mathbb{1})$ also belongs to P. This implies $q_v = 0$.
ii. Let $T_1 = (V, E_{T_1})$ and $T_2 = (V, E_{T_2})$ be two spanning trees in G such that $E_{T_2} = \{E_{T_1} \setminus \{e\}\} \cup \{f\}$. Such trees exist because G is 2-connected (see Diestel [7]). Hence $p_e = p_f$ holds, for any $e, f \in E$. Let ξ denote the value p_e, $e \in E$.

We obtain $r = \xi(n-1)$ and therefore the equality (17) must be a multiple of $\sum_{e\in E} x_e = n-1$. □

Corollary 1. *The dimension of P is* $dim(P) = |V| + |E| - 1$.

The following theorem is useful to establish whether a valid inequality is a facet (see Theorem 3.6 of Nemhauser and Wolsey [15]).

Theorem 2. *Let* $(A^=, b^=)$ *be the equality set of* $S \subseteq \mathbb{R}^k$ *and let* $F = \{x \in S : \pi x = \pi_0\}$ *be a proper face of S. The following two statements are equivalent:*

- *F is a facet of S.*
- *If* $\lambda x = \lambda_0$ *for all* $x \in F$ *then*

  $(\lambda, \lambda_0) = (\alpha\pi + uA^=, \alpha\pi_0 + ub^=)$ for some $\alpha \in \mathbb{R}$ and some $u \in \mathbb{R}^{|H^=|}$.

Proposition 3. *Let* $G = (V, E)$ *be a 2-connected graph. For* $v \in V$, $y_v \le 1$ *defines a facet of P.*

Proof. We prove the result by showing that the conditions of Theorem 2 hold. Consider a fixed vertex $v \in V$. Without loss of generality, we can assume that $v = v_1$, where $V = \{v_1, \dots, v_n\}$. We can therefore write the valid inequality $y_v \le 1$ as

$$(0, \dots, 0)x^T + (1, 0, \dots, 0)y^T \le 1,$$

and hence $F_0^y = \{(x, y) \in P : (0, \dots, 0)x^T + (1, 0, \dots, 0)y^T = 1\}$. It is easy to see that F_0^y is a proper face. Therefore in order to prove that F_0^y represents a facet of P, from Theorem 2, it is sufficient to show that if $\lambda(x, y)^T = \lambda_0$ for all $(x, y) \in F_0^y$, then (λ, λ_0) can be expressed as $(\alpha\pi + uA^=, \alpha\pi_0 + ub^=)$, for some $\alpha \in \mathbb{R}$, $u \in \mathbb{R}^{|H^=|}$. As shown above, in our case $(\pi, \pi_0) = (0, \dots, 0, 1, 0, \dots, 0, 1)$, $(A^=, b^=) = (1, \dots, 1, 0, \dots, 0, n-1)$ and $|H^=| = 1$. For convenience, we represent (λ, λ_0) as

$$(\lambda, \lambda_0) = (s_1, \dots, s_m, t_1, \dots, t_n, \lambda_0).$$

Let $T_w = (V, E_{T_w})$ be a spanning tree of G such that w is not a branch vertex in T_w, where $w \in V$, $w \ne v$. It is easy to see that $(x^{E_{T_w}}, \mathbb{1} \setminus y^{\{w\}})$ and $(x^{E_{T_w}}, \mathbb{1})$ belong to F_0^y, and therefore satisfy $\lambda(x, y)^T = \lambda_0$. Consequently, $\lambda(x^{E_{T_w}}, \mathbb{1} \setminus y^{\{w\}})^T - \lambda(x^{E_{T_w}}, \mathbb{1})^T = 0$ and hence $t_w = 0$. This implies that $t_w = 0$ for any $w \in V \setminus \{v\}$.

Let $T_1 = (V, E_{T_1})$ and $T_2 = (V, E_{T_2})$ be two spanning trees of G such that $E_{T_2} = \{E_{T_1} \setminus \{e\}\} \cup \{f\}$. Note that $(x^{E_{T_1}}, \mathbb{1})$ and $(x^{E_{T_2}}, \mathbb{1})$ belong to F_0^y, therefore $\lambda(x^{E_{T_1}}, \mathbb{1})^T - \lambda(x^{E_{T_2}}, \mathbb{1})^T = 0$ and hence, through simple algebraic manipulations, we obtain $s_e = s_f$. Since T_1 is a generic spanning tree, we can conclude that $s_1 = \dots = s_m$. From now on, we will denote this coefficient vector as s.

From these observations $\lambda(x, y)^T = \lambda_0$ becomes

$$s\sum_{e\in E} x_e + t_1 y_v = \lambda_0.$$

Moreover for any $(x, y) \in F_0^y$, $\sum_{e\in E} x_e = n-1$ and $y_v = 1$, and hence $\lambda_0 = s(n-1) + t_1$. Therefore we obtain

$$(\lambda, \lambda_0) = (s, \dots, s, t_1, 0, \dots, 0, s(n-1) + t_1).$$

Note that

$$(\alpha\pi + uA^=, \alpha\pi_0 + ub^=) = (u, \dots, u, \alpha, 0, \dots, 0, \alpha + u(n-1)).$$

Hence, setting $\alpha = t_1$ and $u = s$ completes the proof. □

Proposition 4. *Let* $G = (V, E)$ *be a 2-connected graph. For* $v \in V$, *if* $G \setminus v$ *is 2-disconnected,* $y_v \ge 0$ *defines a facet of P.*

Proof. We prove the result by showing that the conditions of Theorem 2 hold. Consider a fixed vertex $v \in V$. Without loss of generality, we can assume that $v = v_1$, where $V = \{v_1, \dots, v_n\}$. We can therefore write the valid inequality $y_v \ge 0$ as

$$(0, \dots, 0)x^T + (1, 0, \dots, 0)y^T \ge 0,$$

and hence $F_1^y = \{(x, y) \in P : (0, \dots, 0)x^T + (1, 0, \dots, 0)y^T = 0\}$. It is easy to see that F_1^y is a proper face. As shown above, in our case $(\pi, \pi_0) = (0, \dots, 0, 1, 0, \dots, 0, 0)$, $(A^=, b^=) = (1, \dots, 1, 0, \dots, 0, n-1)$ and $|H^=| = 1$. For convenience, we represent (λ, λ_0) as

$$(\lambda, \lambda_0) = (s_1, \dots, s_m, t_1, \dots, t_n, \lambda_0).$$

Before proceeding with the proof, we need the following remark. □

Remark 1. Due to Lemma 1, for any $w \in V \setminus \{v\}$ there is a spanning tree $T_w = (V \setminus \{v\}, E_{T_w})$ in $G \setminus v$ such that w is not a branch vertex. Since G is 2-connected $d(v) > 1$, therefore there exists a spanning tree $T_{v,w} = (V, E_{T_{v,w}})$ in G such that $E_{T_{v,w}} = E_{T_w} \cup \{e\}$, where $e \in \delta(v)$, $e \ne (v, w)$. $T_{v,w}$ is a spanning tree of G such that w and v are not branch vertices.

Due to Remark 1, for any $w \in V \setminus \{v\}$ there exists a spanning tree $T_{w,v} = (V, E_{T_{w,v}})$ in G such that v and w are not branch vertices, therefore $(x^{E_{T_{w,v}}}, \mathbb{1} \setminus y^{\{v,w\}})$ and $(x^{E_{T_v}}, \mathbb{1} \setminus y^{\{v\}})$ belong to F_1^y, and therefore they satisfy $\lambda(x, y)^T = \lambda_0$. Consequently, $\lambda(x^{E_{T_{w,v}}}, \mathbb{1} \setminus y^{\{v,w\}})^T - \lambda(x^{E_{T_{w,v}}}, \mathbb{1} \setminus y^{\{v\}})^T = 0$. This implies that $t_w = 0$ for any $w \in V \setminus \{v\}$.

Let $T_1 = (V, E_{T_1})$ and $T_2 = (V, E_{T_2})$ be two spanning tree of G such that $E_{T_2} = \{E_{T_1} \setminus \{e\}\} \cup \{f\}$ and v is not a branch vertex in the two trees. Note that $(x^{E_{T_1}}, \mathbb{1} \setminus y^{\{v\}})$ and $(x^{E_{T_2}}, \mathbb{1} \setminus y^{\{v\}})$ belong to F_1^y, therefore $\lambda(x^{E_{T_1}}, \mathbb{1} \setminus y^{\{v\}})^T - \lambda(x^{E_{T_2}}, \mathbb{1} \setminus y^{\{v\}})^T = 0$ and hence, through simple algebraic manipulations, we obtain $s_e = s_f$. Since T_1 is a generic spanning tree, we can conclude that $s_1 = \dots = s_m$. From now on, we will denote this coefficient vector as s.

Due to these observations $\lambda(x, y)^T = \lambda_0$ becomes

$$s\sum_{e\in E} x_e + t_1 y_v = \lambda_0.$$

Moreover for any $(x, y) \in F_1^y$, $\sum_{e\in E} x_e = n-1$ and $y_v = 0$, and hence $\lambda_0 = s(n-1)$. Therefore we obtain

$$(\lambda, \lambda_0) = (s, \dots, s, t_1, 0, \dots, 0, s(n-1)).$$

Note that

$$(\alpha\pi + uA^=, \alpha\pi_0 + ub^=) = (u, \dots, u, \alpha, 0, \dots, 0, \alpha + u(n-1)).$$

Hence, setting $\alpha = 0$ and $u = s$ completes the proof.

Proposition 5. *Let* $G = (V, E)$ *be a 2-connected graph. For* $e \in E$, $x_e \le 1$ *defines a facet of P.*

Proof. Consider a fixed edge $e \in E$. Without loss of generality, we can assume that $e = e_1$, where $E = \{e_1, \dots, e_m\}$. We can therefore write the valid inequality $x_e \le 1$ as

$$(1, \dots, 0)x^T + (0, 0, \dots, 0)y^T \le 1,$$

and hence $F_1^x = \{(x, y) \in P : (1, \dots, 0)x^T + (0, 0, \dots, 0)y^T = 1\}$. Since G is 2-connected, e is not a bridge edge in G and therefore F_1^x is a proper face. As shown above, in our case $(\pi, \pi_0) = (1, 0, \dots, 0, 0, \dots, 0, 1)$, $(A^=, b^=) = (1, \dots, 1, 0, \dots, 0, n-1)$ and $|H^=| = 1$. For convenience, we represent (λ, λ_0) as

$$(\lambda, \lambda_0) = (s_1, \dots, s_m, t_1, \dots, t_n, \lambda_0).$$

Let $T_w = (V, E_{T_w})$ be a spanning tree of G such that w is not a branch vertex in T_w, where $w \in V$, and e belongs to T_w. It is easy to see that $(x^{E_{T_w}}, \mathbb{1} \setminus y^{\{w\}})$ and $(x^{E_{T_w}}, \mathbb{1})$ belong to F_1^x, and therefore they satisfy $\lambda(x, y)^T = \lambda_0$. Consequently, $\lambda(x^{E_{T_w}}, \mathbb{1} \setminus y^{\{w\}})^T - \lambda(x^{E_{T_w}}, \mathbb{1})^T = 0$ and hence $t_w = 0$. This implies that $t_w = 0$ for any $w \in V$.

Let $T_f = (V, E_{T_f})$ and $T_g = (V, E_{T_g})$ be two spanning trees of G such that $e \in E_{T_f}$ and $E_{T_g} = \{E_{T_f} \setminus \{f\}\} \cup \{g\}$, with $f \ne e$. Note that $(x^{E_{T_f}}, \mathbb{1})$ and $(x^{E_{T_g}}, \mathbb{1})$ belong to F_1^x, therefore $\lambda(x^{E_{T_f}}, \mathbb{1})^T -$

$\lambda(x^{E_{T_g}}, \mathbb{1})^T = 0$ and hence, through simple algebraic manipulations, we obtain $s_f = s_g$, for all $f, g \neq e$. From now on, we will denote this coefficients as s.

Due to these observations $\lambda(x, y)^T = \lambda_0$ becomes

$$s_1 x_e + s \sum_{f \in E \setminus \{e\}} x_f = \lambda_0.$$

Moreover for any $(x, y) \in F_1^x$, $\sum_{f \in E} x_f = n - 1$. Since we are assuming $x_e = 1$, then $\sum_{f \in E \setminus \{e\}} x_f = n - 2$ and hence $\lambda_0 = s(n-2) + s_1$. Therefore we obtain

$$(\lambda, \lambda_0) = (s_1, s, \ldots, s, 0, \ldots, 0, s(n-2) + s_1).$$

Note that

$$(\alpha\pi + uA^{=}, \alpha\pi_0 + ub^{=}) = (\alpha + u, u, \ldots, u, 0, \ldots, 0, \alpha + u(n-1)).$$

Hence, setting $\alpha = s_1 - s$ and $u = s$ completes the proof. □

Proposition 6. *For $v \in V$ and $S \subseteq \delta(v)$ with $|S| \geq 3$,*

$$\sum_{e \in S} x_e - 2 \leq (|S| - 2) y_v \tag{18}$$

is valid for P and for P^+.

Proof. It is easy to see that for any subset S of $\delta(v)$, if more than two edges belong to the optimal solution, then vertex v has to be a branch vertex. Note that, for $S = \delta(v)$ we obtain constraints (4), therefore (18) represent a generalized version. □

Proposition 7. *For any cut vertex $v \in V_B$ with $\bar{c}_v = 2$, for $C_i(v)$, $i = 1, 2$ such that $|V_{C_i}(v)| \geq 2$, for $D \subseteq \delta^i(v)$ with $|D| = 2$,*

$$\sum_{e \in D} x_e \leq 1 + y_v \tag{19}$$

is valid for P and for P^+.

Proof. Note that, v being a cut vertex with $\bar{c}_v = 2$, as stated above, at least one edge connecting v with $C_i(v)$ for $i = 1, 2$, has to be selected. As soon as a second edge connecting v with one of the two connected components is selected, v becomes a branch vertex and y_v has to be activated. □

Proposition 8. *For $v \in V$ and $Q \subset \delta(v)$ such that $|Q| = d(v) - 2$*

$$y_v \leq \sum_{e \in Q} x_e \tag{20}$$

is valid for P^+.

Proof. This inequality means that if there exists at least one $Q \subset \delta(v)$ such that all the edges in Q do not belong to the spanning tree, then v cannot be a branch vertex. □

Proposition 9. *Let $R = (V_R, E_R)$ be a cycle of cardinality three, i.e. $V_R = \{a, b, c\}$ and $E_R = \{(a, b), (a, c), (b, c)\}$. For $v \in V_R$ such that $d(v) = 3$, without loss of generality assume that $v = a$,*

$$y_a + x_f \leq 1 \quad f = (b, c) \tag{21}$$

is valid for P^+. Moreover, if there exist at least two vertices a and b in the cycle having degree 3 in the graph, then

$$y_a + y_b \leq x_g \quad g = (a, b) \tag{22}$$

is valid for P^+. Finally, if the three vertices all have degree 3, then

$$y_a + y_b + y_c \leq 1. \tag{23}$$

Proof. Constraints (21) state that if a is a branch vertex, then the edge $f = (b, c)$ cannot be selected otherwise the solution would contain a cycle. Conversely, if edge $f = (b, c)$ belongs to the solution, then a will not be a branch vertex. Constraints (22) impose that only one vertex between a and b can be a branch vertex whenever edge $g = (a, b)$ is selected. If the three vertices have degree 3, then constraints (23) state that at most one of them can be a branch vertex. □

4. Directed and hybrid reformulations

Problems originally defined over undirected graphs can often be reformulated over corresponding directed graphs. In this section we consider a directed integer linear programming reformulation (DILP) of MBVP as a spanning arborescence problem. To develop a model for this directed version of the problem, we choose an arbitrary vertex $r \in V$ as the root vertex and we consider the directed graph $D = (V, A)$ obtained by replacing each edge $(v, u) \in E$ by arcs (v, u) and (u, v) in A. In addition to the previously defined variables y_v, $v \in V$, for each arc $a \in A$, we define z_a as a binary variable equal to 1 if and only if arc a belongs to the spanning arborescence A. In association with graph D, we define $\delta^+(w) = \{(v, u) \in A : v = w\}$, $\delta^-(w) = \{(v, u) \in A : u = w\}$ and $A(S) = \{a = (v, u) \in A : v, u \in S\}$. Note that, from the definition of A, it follows that $|\delta^+(v)| = |\delta^-(v)| = d(v)$, $v \in V$. The DILP formulation is then

$$\text{minimize } z = \sum_{v \in V} y_v \tag{24}$$

subject to

$$\sum_{a \in A(S)} z_a \leq |S| - 1 \quad S \subset V, \ |S| \geq 3 \tag{25}$$

$$\sum_{a \in A} z_a = n - 1 \tag{26}$$

$$\sum_{a \in \delta^-(v)} z_a = 1 \quad v \in V \setminus \{r\} \tag{27}$$

$$\sum_{a \in \delta^+(v)} z_a - 1 \leq (d(v) - 2) y_v \quad v \in V \setminus \{r\} \tag{28}$$

$$\sum_{a \in \delta^+(r)} z_a - 2 \leq (d(r) - 2) y_r \tag{29}$$

$$2y_v \leq \sum_{a \in \delta^+(v)} z_a \quad v \in V \setminus \{r\} \tag{30}$$

$$2y_r \leq \sum_{a \in \delta^+(r)} z_a - 1 \tag{31}$$

$$z_a \in \{0, 1\} \quad a \in A \tag{32}$$

$$y_v \in \{0, 1\} \quad v \in V. \tag{33}$$

Constraints (25)–(27) and the linear relaxation of (32) characterize the spanning arborescence polytope. Note that inequalities (4) and (7), for the undirected graph formulation, are split into inequalities (28), (29) and (30), (31), respectively, for the directed graph reformulation. Also observe that due to (27), one unit is subtracted in the left-hand side of (28) instead of two units in the corresponding inequalities (4).

It is easy to see that several of the properties described for the undirected formulation can be easily adapted to the directed case. Moreover, with the only exception of the root vertex r, no more than one outward pointing arc may be incident to a vertex which is not a branch vertex. Hence the inequalities

$$\sum_{a \in W} z_a - 1 \leq (|W| - 1) y_v \quad v \in V \setminus \{r\}, \ W \subset \delta^+(v) : |W| \geq 2 \tag{34}$$

are clearly valid for the directed formulation.

Let D_{STP} denote the polytope obtained by the linear relaxation of (32) plus constraints (25)–(27) and

$$x_e = z_{vu} + z_{uv} \quad e = (v, u) \in E, \tag{35}$$

in the x-space. Moreover, let D_L denote the polytope defined by the intersection of D_{STP} with the linear relaxation of (33) plus (28)–(31), in the x-space.

Proposition 10. *The undirected and directed formulations for the Minimum Branch Vertex Spanning Tree Problem are equivalent if constraints (35) are introduced in the DILP model.*

Proof. D_{STP} yields an alternative description of the spanning tree polytope P_{STP}, hence $P_{STP} = D_{STP}$ (see [10] for the details). Note that summing up (27) and (28) we obtain (4), and by summing up (27) and (30) we obtain (7). Therefore, $D_L \subseteq P_L$. Conversely, suppose that $(\bar{x}, \bar{y})$ is feasible in P_L, then $\bar{x}$ satisfies (2) and (3), therefore there exists $\bar{z}$ that satisfies (25)–(27), for an arbitrary $r \in V$. For any $v \neq r$ we can rewrite (4) and (7) as follows:

$$\sum_{a \in \delta^+(v)} z_a + \sum_{a \in \delta^-(v)} z_a - 2 \leq (d(v) - 2) y_v \quad v \in V \setminus \{r\} \tag{36}$$

$$2y_v \leq \sum_{a \in \delta^+(v)} z_a + \sum_{a \in \delta^-(v)} z_a - 1 \quad v \in V \setminus \{r\}. \tag{37}$$

Note that (36) and (27) imply (28), and that (37) and (27) imply (30). Then $(\bar{x}, \bar{y}, \bar{z})$ is feasible in D_L, and hence $P_L = D_L$. □

Constraints (2)–(7), (27)–(29), (32) and (35) define the set of feasible solutions for the MBVP. We call this formulation the *hybrid reformulation* (HILP). Note that we keep constraints (4) in this formulation even if they are redundant in the presence of constraints (27)–(29), (32) and (35). However the formulation is still correct and our test results show that this choice helps CPLEX to find an optimal solution faster. Let H_L denote the polytope obtained by the linear relaxation of (5), (6) and (32), plus (2)–(4),(7), (27)–(29) and (35).

Algorithm 1: Branch-and-cut algorithm.

Input: integer program P.
Output: an optimal solution of P.

```
1  Identify initial feasible solution T_0. Get number b_0 of branch
   vertices in T_0
2  ub ← b_0, L = ∅
3  Define a first subproblem and insert it in the list L
4  while L is not empty do
5      chose the subproblem and remove it from L
6      solve the subproblem to obtain the lower bound lb
7      if lb < ub then
8          if the solution is integer then
9              if the solution is feasible then
10                 ub ← lb
11                 update incumbent solution
12             else
13                 search and add SEC on integer solutions
14         else
15             search violated constraints
16             if root node then
17                 search SEC on non-integer solutions
18             if violated constraints are identified then
19                 add them to the model
20             else
21                 branch on a variable and add the corresponding
                   subproblems in L
```

5. Branch-and-cut algorithm

We solve the MBVP by means of a branch-and-cut algorithm which is summarized in Algorithm 1 . Before executing the algorithm we apply a preprocessing phase in which the graph is reduced by exploiting the properties introduced in Section 2.1. In line 1, an initial feasible solution is identified by searching a minimum spanning tree using Prim's algorithm [17]. With any edge $e = (v, u)$ we associate a weight $w_e = n$ if $\min\{d(v), d(u)\} \leq 2$, otherwise $w_e = n - \max\{d(v), d(u)\}$. In line 1, the first subproblem is obtained by relaxing the subtour elimination constraints (2), except for the case where $|S| = 3$, as well as the integrality constraints on the variables. We also identify all the bridges, the 2-cocycles and the cut vertices of the graph and we add the corresponding constraints (13), (14) and (15). In line (13), a search for violated constraints (2) is performed on the integer solutions by identifying the connected components and by adding the subtour elimination constraints induced by the subsets of vertices of all the connected components containing at least one cycle. In line 1, at a non-integer solution, constraints (2) are separated using the max-flow algorithm proposed by Padberg and Wolsey [16]. The max-flow obtained with this algorithm is $f = |\bar{S}| - \sum_{e \in E(\bar{S})} x_e + C$, where $\{\bar{S}, V \setminus \bar{S}\}$ represents the cut-set associated with the max-flow and C is a constant with a value that depends on the vertex set V. Therefore a constraint is violated if $f - C < 1$. To avoid adding constraints with a small violation, a constraint is generated whenever $f - C$ is less than $1 - \epsilon$, for a fixed ϵ depending on the instance. For non-integer solutions, we run the max-flow procedure only at the root node.

The branch-and-cut algorithm was applied to both undirected and hybrid formulations. In the first case, in line 1, a search for violated inequalities (18) and (19) is performed. Valid inequalities (20), (21), (22) and (23) turned out to be ineffective and were not considered. A subset of the most violated inequalities (19) is added to the cut-pool. The separation procedure used for inequalities (18) is the one of Lucena et al. [9]. Let $(\bar{x}, \bar{y})$ be a feasible solution for the linear programming relaxation, and for every $v \in V$ such that $d(v) \geq 3$, order the elements in $\{\bar{x}_e : e \in \delta(v)\}$ in decreasing value. Then, for $(\bar{x}, \bar{y})$, $v \in V$, and every $k \in \{3, \ldots, d(v) - 1\}$, compute $\sum_1^k \bar{x}_{e_k} - (k-2)\bar{y}_v$. This procedure identifies a set S of cardinality k where the largest value on the left-hand side of (18) corresponds to vertex v. If this value is greater than 2, we have the most violated inequality, otherwise, no violated inequality (18) exists for v. For any vertex $v \in V$ with $d(v) \geq 3$, we first consider all $S \subseteq \delta(v)$ such that $|S| = 3$ and we add a subset of the most violated inequalities (18) in the current relaxed solution. Moreover, we run the procedure previously described for $k \in \{4, \ldots, d(v) - 1\}$ and we add at most one violated constraint for each value of k. In line (19) all the violated constraints identified are added to the model. In the implementation for the hybrid formulation a search for violated inequalities (34) is also performed in line (15). The separation procedure is the same as the one described for inequalities (18). As for the previous case, we first look for all subsets $W \subset \delta^+(v)$ such that $|W| = 2$ and a subset of the most violated inequalities is added to the cut-pool. Then, the separation procedure is performed for $k \geq 3$. In line (21), branching takes place in priority on the y_v variables.

6. Computational results

The branch-and-cut algorithm was coded in C and solved using IBM ILOG CPLEX 12.5.1. The computational experiments were performed on a 64-bit GNU/Linux operating system with 96 GB of RAM and one processor Intel Xeon X5675 running at 3.07 GHz. In our tests the *MIPEmphasis* parameter is set to the best bound value, the *VarSel* parameter is set to the default value, i.e. CPLEX chooses

Table 1
Undirected formulation: computational results for small instances.

n	*m*	ub	lb	opt	bridge	2-cocycle	cut vertex	nodes	cuts	sec
20	27	4.6	2.3	2.4	6.2	5.6	4.2	0.0	3.2	0.0
20	34	5.4	0.6	1.2	2.4	5.0	2.2	0.0	5.2	0.0
20	42	3.6	0.0	0.2	0.8	2.2	0.8	0.0	0.0	0.0
20	49	3.6	0.0	0.0	0.0	1.2	0.0	0.0	0.4	0.0
20	57	3.0	0.0	0.0	0.2	0.0	0.2	0.0	1.8	0.0
40	50	12.2	6.5	7.4	16.2	13.4	9.2	0.0	15.8	0.0
40	60	9.2	2.8	3.4	7.4	13.6	5.6	0.4	22.2	0.0
40	71	10.8	1.1	1.6	5.2	7.4	4.6	0.0	18.2	0.0
40	81	8.4	0.3	0.8	2.2	6.6	2.2	0.0	19.4	0.0
40	92	8.2	0.1	0.6	2.2	4.4	2.2	0.0	13.8	0.0
60	71	19.6	11.9	13.0	28.4	27.6	15.6	0.0	16.0	0.0
60	83	18.0	6.9	8.2	17.8	21.0	11.6	1.8	56.0	0.1
60	95	15.4	3.9	5.4	12.0	18.8	9.8	25.0	189.8	0.4
60	107	15.6	2.2	3.4	7.2	13.2	6.4	1.0	126.4	0.2
60	119	12.8	0.9	1.6	4.8	11.6	4.8	7.6	151.6	0.3
80	93	24.0	15.3	16.4	40.8	35.0	21.2	1.6	27.0	0.1
80	106	23.6	10.4	12.0	27.4	30.0	17.4	7.0	77.8	0.1
80	120	22.4	6.9	8.8	19.4	23.4	13.8	36.4	195.4	0.5
80	133	21.0	4.0	5.6	12.4	25.0	10.6	12.8	186.6	0.8
80	147	18.6	2.3	3.4	9.8	19.2	8.4	17.2	199.6	0.4
100	114	31.6	22.8	23.8	56.8	38.6	27.4	4.2	25.4	0.1
100	129	32.0	14.6	16.4	38.6	35.6	22.4	8.0	109.2	0.5
100	144	29.8	10.2	11.8	26.2	32.2	18.0	18.8	189.2	0.6
100	159	27.4	6.4	8.4	18.6	32.4	14.8	47.4	334.2	1.1
100	174	24.4	4.7	6.2	15.4	25.2	11.8	4937.0	2220.6	126.9
120	136	39.6	28.2	29.6	69.8	45.6	33.4	10.4	36.4	0.1
120	152	38.8	20.1	21.8	48.4	48.4	27.8	19.2	124.4	0.4
120	169	34.6	13.8	16.0	36.4	38.4	23.2	28.6	214.4	0.8
120	185	33.2	9.2	11.6	25.4	41.4	18.2	162.0	455.4	1.8
120	202	31.8	6.7	8.6	20.4	34.6	15.0	93.4	442.8	2.3
140	157	45.4	32.5	34.2	79.8	71.0	38.6	14.0	64.0	0.3
140	175	43.6	23.5	25.8	59.0	57.6	33.4	15.4	141.6	0.7
140	193	40.6	16.0	18.8	41.6	52.8	28.4	124.8	329.8	1.8
140	211	39.2	13.0	15.2	35.6	41.2	24.0	128.2	466.4	1.9
140	229	36.0	8.2	10.6	23.8	43.0	19.2	253.0	750.4	4.8
160	179	52.6	38.5	39.8	94.0	64.6	44.8	0.0	28.8	0.2
160	198	49.4	28.8	31.2	69.2	68.2	37.8	45.6	179.8	1.1
160	218	47.2	20.7	23.4	50.2	62.8	31.2	112.6	359.2	1.9
160	237	44.6	15.0	17.4	39.4	50.8	27.4	198.0	542.6	2.9
160	257	44.0	10.8	13.4	32.2	43.0	24.8	248.4	799.0	6.6
180	200	58.6	44.8	46.4	111.6	76.4	51.4	12.0	52.4	0.4
180	221	55.6	32.4	35.0	79.8	67.0	44.2	99.6	215.0	1.5
180	242	54.2	22.7	25.4	58.8	69.6	37.0	204.4	491.2	3.7
180	263	53.2	18.0	21.0	46.6	61.0	32.4	805.0	905.8	14.5
180	284	47.6	14.2	17.6	39.4	56.0	29.6	528.2	1100.8	11.9
200	222	63.6	49.0	50.6	127.8	74.8	57.0	14.6	69.4	0.6
200	244	62.0	37.1	39.4	92.4	77.8	49.6	49.8	174.0	1.3
200	267	59.4	27.7	30.4	69.0	72.0	40.2	130.8	390.0	3.7
200	289	56.4	20.8	24.8	56.8	68.8	38.4	2166.4	1185.8	56.6
200	312	57.2	15.4	[(1)]25.8	42.2	57.6	30.2	5464.6	3942.2	732.5

variable to branch and the *NodeSel* parameter is set to the depth-first search value. Moreover, other parameters are set as follows: *Reduce* is set to no primal or dual reductions; *PreLinear* to only linear reductions; *PreInd* to not apply presolve; *PreslvNd* to no node presolve, *Probe* to no probing and *Threads* to 1. Cuts and heuristics parameters are the default ones. For all the instances the constant ϵ introduced to identify violated constrains (2) in the non-integer solutions is set equal to 0.7. Experiments for the MBVP were conducted on benchmark instances. Carrabs et al. [2] generated instances with n between 20 and 1000 and different densities. Note that dense graphs often contain a Hamiltonian path, therefore the authors generated sparse graphs. These instances were also used by Marín [11]. In his paper the author divides the instances into two groups: medium instances (with $n \leq 500$) and large instances (with $n \geq 600$). Here, the instances with $n \leq 200$ are considered *small*, those with $250 \leq n \leq 500$ are *medium* and those with $n \geq 600$ are *large*. Silva et al. [20] proposed six classes of dense instances. Melo et al. [12] noted that only three of those sets were available for experiments and solved all of them with their algorithm. These instances contain a Hamiltonian path, therefore the optimal solution value is zero for all of them. We call these *branch-free solution instances*.

Tables 1 and 2 report the results for the undirected formulation applied to the small and medium instances. In the tables each line represents an average over five instances with the same number of vertices and edges. In both tables the first two columns represent the instances, columns **ub**, **lb**, **opt** and **sec** report the average of the upper bounds found with Prim's algorithm [17], the average of the LP lower bounds obtained by relaxing the integrality constraints in the formulation, the average of the optimal solution values and the average of the computational time needed to compute the optimal solutions, re-

Table 2
Undirected formulation: computational results for medium instances.

n	m	ub	lb	opt	bridge	2-cocycle	cut vertex	nodes	cuts	sec
250	273	81.4	64.5	66.0	164.4	100.2	71.4	1.4	51.0	0.8
250	297	78.6	50.2	53.0	120.8	110.8	60.8	312.0	318.8	5.0
250	321	75.8	40.4	43.4	101.8	93.8	57.6	277.4	514.8	7.4
250	345	74.6	30.8	34.4	76.2	90.0	47.8	1616.2	930.2	47.9
250	369	70.8	22.8	26.2	60.0	85.2	40.2	732.4	1352.8	42.7
300	326	97.4	79.0	81.0	203.0	121.8	87.4	30.2	127.2	1.9
300	353	95.0	65.0	67.8	160.2	116.4	78.6	171.4	323.6	6.2
300	380	92.6	50.4	54.6	124.8	114.0	69.0	572.4	785.4	21.9
300	407	89.6	41.8	46.2	104.6	103.4	61.8	1808.0	1619.6	75.9
300	434	85.0	32.6	37.2	86.4	89.4	56.2	1657.2	1933.0	143.8
350	378	113.4	92.1	94.6	238.8	143.2	102.8	70.2	152.6	5.0
350	406	111.6	76.8	80.6	190.0	145.6	93.6	452.4	476.8	10.1
350	435	108.0	60.4	65.6	151.0	150.8	84.4	2016.2	1379.4	85.6
350	463	107.2	50.5	56.6	124.2	128.4	75.8	11731.0	1945.2	663.4
350	492	102.6	40.0	45.4	103.6	123.8	67.2	5569.6	2322.0	444.0
400	429	130.8	109.2	111.8	282.6	167.2	119.6	56.2	123.6	3.9
400	459	128.0	90.2	94.0	226.4	165.0	109.4	851.6	782.6	21.0
400	489	126.2	73.9	(1)88.4	184.8	152.4	99.0	2315.8	5068.8	742.9
400	519	122.2	62.2	68.4	154.2	154.4	88.4	9878.8	2517.8	979.4
400	549	118.4	51.0	56.0	131.2	141.2	80.2	3204.6	2962.6	350.2
450	482	148.6	123.1	125.8	318.6	177.8	135.4	33.6	116.0	4.8
450	515	146.0	103.1	107.4	250.6	202.8	121.6	1298.6	846.2	75.4
450	548	140.0	85.1	90.4	208.8	184.2	110.4	3686.0	4059.0	835.4
450	581	139.2	71.3	(1)77.6	176.6	167.8	100.4	12719.2	2901.4	1363.9
450	614	133.2	59.0	(3)66.4	151.8	153.8	93.8	17717.4	3686.4	2766.6
500	534	164.6	139.2	141.6	361.0	191.2	150.6	70.4	149.2	10.2
500	568	160.8	117.3	120.8	294.2	187.0	137.2	948.6	770.0	53.8
500	603	158.2	99.6	105.6	246.0	198.4	126.8	3089.4	1981.2	260.3
500	637	151.6	83.4	(2)117.2	210.6	181.2	116.8	4850.8	6615.4	1902.9
500	672	148.4	69.3	(5)122.8	170.0	194.6	104.4	13407.2	11163.8	3600.0

Table 3
Hybrid formulation: computational results for small and medium instances.

n	m	ub	lb	opt	bridge	2-cocycle	cut vertex	nodes	cuts	sec
20	41.8	4.0	0.6	0.8	1.9	2.8	1.5	0.0	1.8	0.0
40	70.8	9.8	2.2	2.8	6.6	9.1	4.8	0.2	33.6	0.1
60	95.0	16.3	5.2	6.3	14.0	18.4	9.6	0.0	67.4	0.5
80	119.8	21.9	7.8	9.2	22.0	26.5	14.3	1.0	83.9	0.7
100	144.0	29.0	11.7	13.3	31.1	32.8	18.9	1.7	108.7	1.0
120	168.8	35.6	15.6	17.5	40.1	41.7	23.5	2.6	135.0	1.1
140	193.0	41.0	18.6	20.9	48.0	53.1	28.7	6.2	178.8	2.0
160	217.8	47.6	22.7	25.0	57.0	57.9	33.2	2.8	165.6	1.9
180	242.0	53.8	26.4	29.1	67.2	66.0	38.9	9.0	212.3	2.5
200	266.8	59.7	30.0	32.6	77.6	70.2	43.1	6.8	213.4	3.1
250	321.0	76.2	41.7	44.6	104.6	96.0	55.6	5.8	209.8	3.1
300	380.0	91.9	53.7	57.4	135.8	109.0	70.6	6.0	230.2	4.2
350	434.8	108.6	64.0	68.6	161.5	138.4	84.8	7.9	298.8	6.9
400	489.0	125.1	77.3	81.8	195.8	156.0	99.3	21.0	355.2	9.1
450	548.0	141.4	88.3	93.4	221.3	177.3	112.3	17.5	333.7	9.5
500	602.8	156.7	101.7	106.7	256.4	190.5	127.2	10.3	332.0	9.8

spectively. Moreover, whenever α instances of a group are not solved to optimality within the time limit of 1 h, we write (α) close to the solution value. The numbers of bridges, 2-cocycles and cut vertices are also reported. Columns **nodes** and **cuts** represent the number of nodes in the search tree and the number of cuts added.

Results for small, medium and large instances for the hybrid formulation are reported in Table 3 and 4. In Table 3 each line represents an average over 25 instances with the same number of vertices. The table reports the results for both small and medium instances. In Table 4 each line represents an average over five instances with the same number of vertices and edges. These two tables have the same structure than Tables 1 and 2. Note that all the instances were solved optimally in this case. Our results show that the hybrid formulation is more efficient and faster. It allows us to solve all the small and medium instances in less than 10 s, while the undirected formulation could not find an optimal solution on 13 instances after 1 h. Moreover, we can solve all the large instances, up to $n = 1000$ in 90.5 s on average. Finally, note that the number of nodes in the search tree is relatively small and all families of cuts are useful.

Tables 5 and 6 report the results obtained with the MIP solver without heuristic and cuts, and with the same parameters used for solving the hybrid formulation. The first two columns represent the instances and the third one the optimal solution. Columns **nodes**, **cuts** and **sec** represent the number of nodes in the search tree, the number of cuts added, and the average of the computational time needed to compute the solutions using the branch-and-

Table 4
Hybrid formulation: computational results for large instances.

n	m	ub	lb	opt	bridge	2-cocycle	cut vertex	nodes	cuts	sec
600	637.0	197.6	180.4	183.8	493.6	68.8	188.0	0.0	74.0	3.2
600	674.0	192.6	163.8	167.2	437.4	71.6	176.4	0.0	148.8	8.7
600	712.0	188.0	147.2	150.6	394.4	68.6	168.6	1.6	229.0	10.3
600	749.0	182.2	136.1	138.8	363.4	55.6	161.0	21.2	335.6	17.6
600	787.0	173.8	123.9	125.8	333.6	49.4	153.2	18.2	333.8	16.2
700	740.0	232.0	211.0	214.4	576.8	91.4	218.6	0.0	100.4	8.7
700	780.0	224.8	193.7	198.0	518.4	89.2	206.8	2.6	176.6	11.0
700	821.0	218.0	175.7	180.0	470.2	79.4	198.2	0.6	257.2	12.5
700	861.0	212.4	160.8	164.0	436.6	62.8	191.4	3.2	291.0	17.4
700	902.0	205.0	151.2	154.2	403.2	63.6	183.2	1.0	293.6	14.7
800	843.0	265.4	242.0	245.6	666.8	90.6	252.2	0.0	102.0	10.3
800	886.0	256.8	223.4	227.6	599.4	98.8	237.4	1.8	169.0	11.2
800	930.0	253.6	204.3	208.4	546.6	89.2	228.8	10.2	321.6	22.7
800	973.0	245.2	189.9	194.2	505.8	82.0	221.4	72.4	658.4	48.8
800	1017.0	232.2	172.4	176.2	468.2	71.4	212.8	23.8	479.8	37.1
900	944.0	300.6	275.2	279.6	756.4	105.4	284.8	0.0	118.4	12.6
900	989.0	290.0	254.0	259.2	685.6	110.4	271.4	188.8	339.6	66.2
900	1034.0	286.6	235.7	240.6	633.0	105.0	262.2	28.2	405.4	30.2
900	1079.0	281.4	218.0	223.2	583.6	98.0	251.2	12.6	489.8	90.5
900	1124.0	269.0	202.2	206.0	547.6	83.2	242.4	2.0	372.0	30.7
1000	1047.0	332.6	307.5	312.0	849.6	110.2	317.0	8.4	148.8	26.2
1000	1095.0	323.2	283.0	290.0	767.0	121.0	303.2	0.0	209.2	17.0
1000	1143.0	318.6	265.4	271.2	705.0	121.2	290.2	74.2	613.4	57.1
1000	1191.0	310.4	245.0	251.0	657.6	109.8	279.8	53.6	621.2	75.4
1000	1239.0	303.8	230.3	235.2	609.8	105.6	268.4	45.8	735.4	62.6

Table 5
Hybrid formulation: MIP solver without heuristics and cuts for small and medium instances.

n	m	opt	nodes	cuts	sec	$\overline{nodes}$	$\overline{cuts}$	$\overline{sec}$
20	41.8	0.8	0.0	1.8	0.0	0.8	9.2	0.0
40	70.8	2.8	0.2	33.6	0.1	3.2	64.2	0.1
60	95.0	6.3	0.0	67.4	0.5	6.4	93.4	0.1
80	119.8	9.2	1.0	83.9	0.7	6.6	119.3	0.2
100	144.0	13.3	1.7	108.7	1.0	6.2	138.1	0.3
120	168.8	17.5	2.6	135.0	1.1	9.2	181.1	0.5
140	193.0	20.9	6.2	178.8	2.0	24.3	244.9	1.0
160	217.8	25.0	2.8	165.6	1.9	13.9	220.1	1.0
180	242.0	29.1	9.0	212.3	2.5	24.0	287.7	1.7
200	266.8	32.6	6.8	213.4	3.1	16.6	277.0	1.9
250	321.0	44.6	5.8	209.8	3.1	150.2	684.6	93.0
300	380.0	57.4	6.0	230.2	4.2	52.6	364.8	6.2
350	434.8	68.6	7.9	298.8	6.9	76.5	513.4	9.7
400	489.0	81.8	21.0	355.2	9.1	60.6	476.8	10.6
450	548.0	93.4	17.5	333.7	9.5	62.1	554.4	18.4
500	602.8	106.7	10.3	332.0	9.8	57.7	538.2	15.1

Table 6
Hybrid formulation: MIP solver without heuristics and cuts for large instances.

n	m	opt	nodes	cuts	sec	$\overline{nodes}$	$\overline{cuts}$	$\overline{sec}$
600	637.0	183.8	0.0	74.0	3.2	13.4	135.0	7.7
600	674.0	167.2	0.0	148.8	8.7	23.0	219.4	11.3
600	712.0	150.6	1.6	229.0	10.3	31.6	375.4	15.7
600	749.0	138.8	21.2	335.6	17.6	266.2	1343.0	102.4
600	787.0	125.8	18.2	333.8	16.2	56.0	529.2	28.1
700	740.0	214.4	0.0	100.4	8.7	21.2	145.6	11.2
700	780.0	198.0	2.6	176.6	11.0	32.2	274.0	15.1
700	821.0	180.0	0.6	257.2	12.5	143.6	576.2	39.0
700	861.0	164.0	3.2	291.0	17.4	867.8	2870.0	(1)744.0
700	902.0	154.2	1.0	293.6	14.7	48.6	667.6	74.7
800	843.0	245.6	0.0	102.0	10.3	20.8	164.2	14.1
800	886.0	227.6	1.8	169.0	11.2	28.4	296.0	19.9
800	930.0	208.4	10.2	321.6	22.7	118.8	735.0	49.5
800	973.0	194.2	72.4	658.4	48.8	145.8	731.8	91.9
800	1017.0	176.2	23.8	479.8	37.1	119.2	825.4	70.0
900	944.0	279.6	0.0	118.4	12.6	26.0	183.6	19.5
900	989.0	259.2	188.8	339.6	66.2	66.6	386.2	36.2
900	1034.0	240.6	28.2	405.4	30.2	113.2	566.2	53.0
900	1079.0	223.2	12.6	489.8	90.5	239.4	800.8	115.3
900	1124.0	206.0	2.0	372.0	30.7	39.0	625.8	49.8
1000	1047.0	312.0	8.4	148.8	26.2	38.0	234.6	42.2
1000	1095.0	290.0	0.0	209.2	17.0	48.4	408.4	32.9
1000	1143.0	271.2	74.2	613.4	57.1	215.0	1019.2	99.1
1000	1191.0	251.0	53.6	621.2	75.4	158.8	920.2	94.7
1000	1239.0	235.2	45.8	735.4	62.6	164.6	1051.6	109.6

cut algorithm described previously. Finally, columns $\overline{\textbf{nodes}}$, $\overline{\textbf{cuts}}$, $\overline{\textbf{sec}}$ provide the number of nodes in the search tree, the number of cuts added, and the average of the computational time needed to compute the solutions with the MIP solver without heuristic and cuts, and with the same parameters used for solving the hybrid formulation. The results show that our hybrid formulation remains efficient without using CPLEX heuristics and CPLEX cuts, and they also show that only one instance, with 700 nodes and 861 edges, cannot be solved to optimality within the time limit of 1 h. We write (1) close to the time value $\overline{\textbf{sec}}$.

6.1. LP lower bounds and duality gaps

We present the LP lower bounds obtained by adding one valid inequality at a time to the hybrid formulation. Note that all the lower bounds are calculated with the MIP solver without parallel, heuristic, preprocessing and cuts, and the parameters are the default ones. In Table 7 each line is an average over 25 instances, while in Table 8 the average is computed over five instances. In both tables the first two columns represent the instances and the third one the optimal solution. The next columns provide lower bounds $w(H_L)$, $w(H_L^1)$, $w(H_L^2)$, $w(H_L^3)$, $w(H_L^4)$ and $w(H_L^5)$, where H_L denotes the polytope obtained by relaxing the integrality constraints in the hybrid formulation, defined by constraints (2)–(7), (27)–(29), (32) and (35), while H_L^1 and H_L^2 denote the intersection of H_L with (34) for $W \subset \delta^+(v)$ with $|W| = 2$ and $|W| \geq 3$, respectively. Moreover, H_L^3 denotes the intersection of H_L with (19), while

Table 7
Hybrid formulation: lower bounds for MBVP on small and medium instances.

n	m	opt	$w(H_L)$	sec(H_L)	$w(H_L^1)$	sec(H_L^1)	$w(H_L^2)$	sec(H_L^2)	$w(H_L^3)$	sec(H_L^3)	$w(H_L^4)$	sec(H_L^4)	$w(H_L^5)$	sec(H_L^5)
20	41.8	0.8	0.59	0.00	0.65	0.00	0.64	0.00	0.60	0.00	0.61	0.00	0.62	0.00
40	70.8	2.8	2.16	0.01	2.31	0.04	2.29	0.07	2.21	0.02	2.23	0.04	2.25	0.05
60	95.0	6.3	5.18	0.03	5.64	0.08	5.56	0.09	5.33	0.05	5.42	0.09	5.44	0.08
80	119.8	9.2	7.79	0.09	8.44	0.13	8.25	0.13	8.05	0.11	8.19	0.14	8.10	0.13
100	144.0	13.3	11.75	0.10	12.47	0.19	12.28	0.25	12.06	0.21	12.23	0.23	12.16	0.18
120	168.8	17.5	15.60	0.23	16.46	0.37	16.24	0.38	16.02	0.33	16.20	0.39	16.14	0.33
140	193.0	20.9	18.64	0.40	19.64	0.53	19.36	0.58	19.18	0.79	19.47	0.69	19.17	0.61
160	217.8	25.0	22.74	0.46	23.75	0.79	23.55	0.93	23.25	0.78	23.55	0.77	23.31	1.08
180	242.0	29.1	26.39	1.65	27.62	1.23	27.30	1.47	27.08	0.88	27.41	1.24	27.06	2.70
200	266.8	32.6	30.00	4.63	31.22	1.45	30.91	3.69	30.55	3.28	30.94	2.18	30.68	3.18
250	321.0	44.6	41.72	1.58	43.06	1.81	42.68	1.95	42.43	1.72	42.96	2.37	42.42	1.34
300	380.0	57.4	53.74	4.48	55.55	5.02	54.93	6.34	54.73	5.52	55.23	5.50	54.51	7.37
350	434.8	68.6	63.96	8.39	65.93	4.86	65.36	8.24	65.40	6.68	66.16	10.34	65.00	9.03
400	489.0	81.8	77.31	8.71	79.33	6.55	78.68	25.40	78.96	9.33	79.62	6.40	78.21	31.43
450	548.0	93.4	88.32	81.74	90.66	14.86	89.75	34.63	90.04	14.68	90.79	20.25	89.29	108.96
500	602.8	106.7	101.75	15.68	104.26	15.94	103.43	47.07	103.48	40.31	104.16	23.18	102.86	23.71

Table 8
Hybrid formulation: lower bounds for MBVP on large instances.

n	m	opt	$w(H_L)$	sec(H_L)	$w(H_L^1)$	sec(H_L^1)	$w(H_L^2)$	sec(H_L^2)	$w(H_L^3)$	sec(H_L^3)	$w(H_L^4)$	sec(H_L^4)	$w(H_L^5)$	sec(H_L^5)
600	637	183.8	180.40	4.16	180.94	6.51	180.70	6.78	182.03	4.47	182.79	6.37	180.63	3.71
600	674	167.2	163.83	13.05	164.63	9.71	164.47	10.88	164.97	29.23	166.11	37.86	164.45	41.82
600	712	150.6	147.24	5.15	148.31	14.13	147.99	13.17	148.10	11.31	148.96	22.03	147.73	12.24
600	749	138.8	136.09	5.50	136.97	13.23	136.82	29.51	136.50	11.30	136.83	17.20	136.75	19.90
600	787	125.8	123.87	1436.26	124.66	98.91	124.66	120.82	124.17	713.52	124.46	379.27	124.85	425.97
700	740	214.4	211.03	6.02	211.56	10.98	211.30	12.36	212.89	6.24	213.68	14.96	211.08	6.11
700	780	198.0	193.67	12.24	194.79	23.16	194.56	583.03	195.11	64.68	196.52	60.49	194.30	726.18
700	821	180.0	175.72	20.35	177.14	134.69	176.91	729.02	177.11	21.07	178.28	214.58	176.56	71.40
700	861	164.0	160.81	34.02	161.82	732.00	161.73	244.80	161.29	784.56	161.87	168.69	161.77	70.80
700	902	154.2	151.16	758.59	152.43	115.37	152.42	420.73	151.71	228.43	152.17	95.60	152.50	760.61
800	843	245.6	242.02	8.50	242.55	40.72	242.25	15.81	244.04	13.46	245.08	17.13	242.17	8.24
800	886	227.6	223.44	6.56	224.24	16.22	224.15	21.32	224.82	10.27	226.47	33.38	223.67	6.72
800	930	208.4	204.26	34.75	205.36	42.62	205.26	875.43	205.55	42.09	206.92	35.75	204.82	28.72
800	973	194.2	189.87	660.59	191.48	107.55	191.15	350.10	190.73	1036.30	191.68	139.47	191.12	756.66
800	1017	176.2	172.37	767.53	173.72	38.56	173.63	473.45	172.99	71.40	173.49	85.26	173.79	43.47
900	944	279.6	275.15	29.70	275.76	12.19	275.33	13.81	277.72	10.44	278.88	21.45	275.17	28.26
900	989	259.2	253.97	62.55	255.29	110.73	254.81	64.46	256.15	18.22	257.71	35.00	254.26	18.85
900	1034	240.6	235.66	27.99	236.88	29.03	236.84	57.55	237.38	105.78	239.09	67.88	236.37	40.64
900	1079	223.2	218.02	13.65	219.98	71.61	219.59	110.59	219.52	40.19	220.61	62.95	219.09	19.95
900	1124	206.0	202.19	727.97	203.78	735.15	203.50	745.99	202.87	397.24	203.41	759.04	203.47	96.45
1000	1047	312.0	307.48	37.12	308.28	27.04	307.97	79.40	310.08	33.45	311.33	23.69	307.63	51.69
1000	1095	290.0	283.03	22.26	284.62	25.38	284.23	36.97	286.72	52.87	288.50	26.43	283.78	20.94
1000	1143	271.2	265.37	94.31	266.94	777.18	266.76	217.49	267.40	446.09	269.43	275.63	266.37	92.61
1000	1191	251.0	244.99	96.29	246.92	783.48	246.68	1448.73	246.82	758.26	248.22	123.72	246.29	30.05
1000	1239	235.2	230.27	30.94	232.16	58.52	231.90	91.77	231.10	743.89	231.92	592.85	231.74	95.08

H_L^4 and H_L^5 denote the intersection with (18) for $S \subseteq \delta(v)$ with $|S| = 3$ and $|S| \geq 4$, respectively. It is easy to see from the tables that all the cuts help to improve the lower bound. In particular, $w(H_L^1)$, $w(H_L^3)$ and $w(H_L^4)$ seem to yield the best lower bounds in most cases. Inequalities (34) for $W \subset \delta^+(v)$ with $|W| = 2$ and (18) for $S \subseteq \delta(v)$ with $|S| = 3$ are the most useful cuts.

6.2. Branch-free solution instances

We provide next our computational results using the branch-free solution instances. Due to the high density of these instances the time required for the preprocessing phase would be too high. Therefore we decided not to look for bridges, 2-cocycles and cut-vertices. Test results show (Table 9) that even if the initial feasible solutions are not good the computational time needed to find a Hamiltonian path is reasonable compared with the dimension of the instances.

7. Conclusions

We have modeled and solved the Minimum Branch Vertices Spanning Tree Problem. We have provided two mathematical formulations based on an undirected and a directed graph, as well as a hybrid formulation obtained by merging the first two. Moreover, we have derived some properties and some valid inequalities for the problem. A branch-and-cut approach was proposed for the undirected and the hybrid formulations. Results show that the hybrid formulation is superior to the undirected formulation and that our branch-and-cut algorithm applied to the hybrid solves all benchmark instances to optimality.

Acknowledgments

This research was partly supported by the Canadian Natural Sciences and Engineering Research Council under grant

Table 9
Hybrid formulation: computational results for branch-free solution instances.

id	*n*	*m*	ub	lb	opt	nodes	cuts	sec
alb1000	1000	1998	227	0	0	0	0	30.0
alb2000	2000	3996	476	0	0	0	0	786.8
alb3000a	3000	5999	711	0	0	0	0	704.3
alb4000	4000	7997	980	0	0	0	0	1805.7
steind11	1000	5000	142	0	0	0	0	42.6
steind12	1000	5000	160	0	0	0	0	29.7
steind13	1000	5000	156	0	0	0	0	35.1
steind14	1000	5000	154	0	0	0	0	49.1
steind15	1000	5000	151	0	0	0	0	61.2
le450_5a	450	5714	39	0	0	0	0	3.2
le450_5b	450	5734	38	0	0	0	0	9.3
le450_5c	450	9803	29	0	0	0	0	35.2
le450_5d	450	9757	29	0	0	0	0	29.2
le450_15a	450	8168	27	0	0	0	0	12.5
le450_15b	450	8169	29	0	0	0	0	104.3
le450_15c	450	16679	17	0	0	0	0	152.5
le450_15d	450	16750	18	0	0	0	0	73.6
le450_25a	450	8260	29	0	0	0	0	43.4
le450_25b	450	8263	27	0	0	0	0	27.3
le450_25c	450	17343	17	0	0	0	0	203.7
le450_25d	450	17425	20	0	0	0	0	151.2

2015-06189. This support is gratefully acknowledged. Thanks are also due to the two referees for their valuable comments.

References

[1] Bondy JA, Murty USR. Graph theory with applications, vol. 290. London: Macmillan; 1976.
[2] Carrabs F, Cerulli R, Gaudioso M, Gentili M. Lower and upper bounds for the spanning tree with minimum branch vertices. Comput Optim Appl 2013;56:405–38.
[3] Cerrone C, Cerulli R, Raiconi A. Relations, models and a memetic approach for three degree-dependent spanning tree problems. Eur J Oper Res 2014;232:442–53.
[4] Cerulli R, Gentili M, Iossa A. Bounded-degree spanning tree problems: models and new algorithms. Comput Optim Appl 2009;42:353–70.
[5] Conforti M, Cornuéjols G, Zambelli G. Integer programming. Berlin: Springer; 2014.
[6] Dantzig GB, Fulkerson DR, Johnson SM. Solution of a large-scale traveling-salesman problem. J. Oper. Res. Soc. Am. 1954;2:393–410.
[7] Diestel R. Graph theory. Graduate texts in mathematics, vol. 173. Berlin and Heidelberg: Springer-Verlag GmbH & amp; 2000.
[8] Gargano L, Hell P, Stacho L, Vaccaro U. Spanning trees with bounded number of branch vertices. In: Automata, languages and programming. Berlin, Heidelberg: Springer; 2002. p. 355–65.
[9] Lucena A, Maculan N, Simonetti L. Reformulations and solution algorithms for the maximum leaf spanning tree problem. Comput Manage Sci 2010;7:289–311.
[10] Magnanti TL, Wolsey LA. Optimal trees. In: Ball MO, Magnanti T, Monma C, Nemhauser G, editors. Network models. Handbooks in operations research and management science, vol. 6. Amsterdam: North-Holland; 1995. p. 503–615.
[11] Marín A. Exact and heuristic solutions for the minimum number of branch vertices spanning tree problem. Eur J Oper Res 2015;245:680–9.
[12] Melo RA, Samer P, Urrutia S. An effective decomposition approach and heuristics to generate spanning trees with a small number of branch vertices. Comput Optim Appl 2016;65:821–44.
[13] Merabet M, Durand S, Molnar M. Minimization of branching in the optical trees with constraints on the degree of nodes. In: ICN'12: the eleventh international conference on networks. Saint-Gilles, Réunion Island; 2012. p. 235–40.
[14] Miller CE, Tucker AW, Zemlin RA. Integer programming formulation of traveling salesman problems. J Assoc Comput Mach 1960;7:326–9.
[15] Nemhauser GL, Wolsey LA. Integer and combinatorial optimization. New York: Wiley; 2014.
[16] Padberg MW, Wolsey LA. Trees and cuts. In: Combinatorial mathematics. In: Annals of discrete mathematics, vol. 17. Amsterdam: North-Holland; 1983. p. 511–17.
[17] Prim RC. Shortest connection networks and some generalizations. Bell Syst Tech J 1957;36:1389–401.
[18] Schmidt JM. A simple test on 2-vertex- and 2-edge-connectivity. Inf Process Lett 2013;113:241–4.
[19] Silva DM, Silva RMA, Mateus GR, Gonçalves JF, Resende MGC, Festa P. An iterative refinement algorithm for the minimum branch vertices problem. In: Experimental algorithms. Berlin, Heidelberg: Springer; 2011. p. 421–33.
[20] Silva RMA, Silva DM, Resende MGC, Mateus GR, Gonçalves JF, Festa P. An edge-swap heuristic for generating spanning trees with minimum number of branch vertices. Optim Lett 2014;8:1225–43.
[21] Sundar S, Singh A, Rossi A. New heuristics for two bounded-degree spanning tree problems. Inf Sci 2012;195:226–40.