Lower Bounds and Exact Algorithms for the Quadratic Minimum Spanning Tree Problem (Submitted to Computers & Operations Research on May 2013)

Dilson Lucas Pereira *
CAPES Foundation, Ministry of Education of Brazil,
Brasília - DF 70.040-020, Brazil
dilson@dcc.ufmg.br

Michel Gendreau
École Polytechnique de Montréal,
Montréal - QC, Canada
michel.gendreau@cirrelt.ca

Alexandre Salles da Cunha†
Departamento de Ciência da Computação, Universidade Federal de Minas Gerais,
Belo Horizonte - MG, Brazil
acunha@dcc.ufmg.br

December 2013

Abstract

Given a connected and undirected graph, the quadratic minimum spanning tree problem consists of finding one spanning tree that minimizes a quadratic cost function. We first propose an integer programming formulation based on the reformulation-linearization technique and show that such a formulation is stronger than previous ones in the literature. We then introduce a novel type of formulation, based on the idea of partitioning spanning trees into subgraphs of fixed size. This idea offers a hierarchy of formulations of increasing strength, such that the first hierarchy level is precisely the model obtained with the application of the reformulation-linearization technique. Many possible relaxations of the hierarchy are also studied. On the computational side, three Lagrangian relaxation procedures and two parallel branch-and-bound algorithms are developed. For the first time, several instances in the literature were solved to optimality, including some with 50 vertices.

Keywords: Quadratic 0-1 programming, Lagrangian relaxation, Spanning trees

1 Introduction

Given a connected and undirected graph $G = (V, E)$ with $n = |V|$ vertices and $m = |E|$ edges, and a matrix $Q = (q_{ij})_{i,j \in E}$ of *interaction costs* between the edges of G, the quadratic minimum spanning tree problem (QMSTP) is a quadratic 0-1 programming problem that consists of finding a spanning tree of G, whose incidence vector $\mathbf{x} \in \mathbb{B}^m$ minimizes the function

$$\sum_{i,j \in E} q_{ij} x_i x_j.$$

*Partially funded by CAPES, BEX 2418/11-8
†Partially funded by CNPq grants 305423/2012-6, 477863/2010-8 and FAPEMIG PRONEX APQ-01201-09.

QMSTP was proven to be NP-Hard by Assad and Xu [5], by means of a polynomial reduction from the quadratic assignment problem (QAP) [6]. If the objective function is linear (i.e., if Q is diagonal), QMSTP reduces to the minimum spanning tree problem (MSTP), for which several polynomial time algorithms are known [29, 19].

An application of the QMSTP can be found in the context of wireless sensor networks, where the communication between sensor nodes occurs by means of radio transmissions. Assuming that the radio frequency assigned for each possible communication link in the network has been defined beforehand, one might wish to find a communication spanning tree that minimizes the radio interference between pairs of links. Clearly, the interference between pairs of links can be modeled by the off-diagonal entries of Q. Other applications can be found in the context of telecommunication, transportation, and hydraulic networks [5].

1.1 Literature Review

A common procedure for computing lower bounds for constrained quadratic 0-1 problems is that of Gilmore [16] and Lawler [20] (the so called Gilmore-Lawler procedure/lower bound). The procedure was introduced for the QAP and later adapted for many other problems like, for example, the quadratic 0-1 knapsack problem [28]. The Gilmore-Lawler procedure has an important role in many exact solution algorithms for constrained quadratic 0-1 problems, where it can be used either as a lower bounding procedure on itself or as a procedure for the resolution of subproblems in Lagrangian relaxation schemes.

In [5], Assad and Xu proposed a lower bounding procedure and a branch-and-bound algorithm for QMSTP. The procedure can be seen as a dual ascent algorithm for obtaining near optimal multipliers in a Lagrangian relaxation scheme. Given a choice of Lagrangian multipliers, the Gilmore-Lawler procedure is used to solve the resulting Lagrangian subproblem. Such a lower bounding procedure was embedded into a branch-and-bound algorithm that managed to solve instances defined over complete graphs with up to 12 vertices. When interactions costs are given only for adjacent edges, the branch-and-bound algorithm in [5] solved instances with up to 15 vertices.

Another branch-and-bound algorithm that also relies on the Gilmore-Lawler lower bounding procedure was proposed by Cordone and Passeri [9]. At publication time, that algorithm solved instances with 10 and 15 vertices, depending on whether the input graph was respectively complete or sparse.

Öncan and Punnen [24] also introduced a procedure based on Lagrangian relaxation. That algorithm makes use of the QMSTP formulation in [5], strengthened with new valid inequalities, that were relaxed and dualized in a Lagrangian fashion. A Lagrangian heuristic that uses subgradient optimization was implemented and tested, but the procedure was not embedded into a branch-and-bound search tree.

Computational results for a QMSTP algorithm proposed by Cordone and Passeri [10] are also available on-line, at [8]. Details on the algorithm and on the computational experiments for the results reported in [8] are not available since, the paper [10] is yet unpublished and, hence, is not publicly available.

For the instances introduced in [9], computational results reported in [8] indicate that those defined over complete graphs (resp. defined over sparse graphs) with up to 15 (resp. 20) vertices are solved to proven optimality. For the instances of [24], they report solving problems defined over complete graphs with up to 20 vertices.

One common approach to solve quadratic 0-1 problems consists of linearizing the non-linear terms, in order to obtain a mixed integer linear program. In the particular case of the QMSTP, that is accomplished by introducing additional variables $y_{ij} := x_i x_j, \forall i, j \in E$, used to replace the quadratic terms $x_i x_j$ in the objective function. After linearization takes place, one is interested in describing the convex hull of points $(\mathbf{x}, \mathbf{y}) \in \{0,1\}^{m+\frac{m(m-1)}{2}}$, such that $\mathbf{x}$ is the incidence vector of a spanning tree of G and vector $\mathbf{y}$ satisfies $y_{ij} := x_i x_j, \forall i, j \in E$. Closely related polytopes are the boolean quadric polytope (BQP) and the boolean quadric forest polytope (BQFP). BQP is the convex-hull of the vectors that satisfy the integer programming (IP) formulation that results after applying the linearization procedure for the unconstrained quadratic 0-1 program. Because of that, valid inequalities for BQP [25] are valid for all constrained quadratic 0-1 problems, QMSTP included. Many facet defining inequalities for BQFP were studied by Lee and Leung [21], but no computational use of them was reported. Some of these inequalities derive from the reformulation-linearization technique (RLT) [3, 30]; others are special cases of valid inequalities for BQP introduced in [25].

Heuristic procedures for QMSTP span a wide range of solution approaches. Three methods are proposed in [5]; two of them are constructive algorithms, while the third is a Lagrangian heuristic. A genetic algorithm is proposed in [32]. Greedy heuristics and a tabu search are proposed in [9]. Sundar and Singh [31] propose an algorithm based on artificial bee colony. A local search based on tabu thresholding is suggested in [24]. Algorithms based on simulated annealing, genetic algorithms, and tabu search are also discussed in [27].

1.2 Outline of the Paper and Main Contributions

In Section, 2.1, we develop an integer programming formulation for QMSTP, based on a partial application of RLT [3, 30]. An effective Lagrangian relaxation scheme is developed to obtain its linear programming (LP) relaxation bounds. It is also shown that the LP lower bounds provided by the formulation dominate previous QMSTP lower bounds in literature.

In section 2.2, we introduce a novel formulation for the problem that is based on the idea of partitioning spanning trees into subgraphs of fixed size. We show that this formulation offers a hierarchy of LP lower bounds of increasing strength. In particular, at the first hierarchy level, it has the first formulation as a subcase.

The second formulation has a huge number of variables and constraints. With the purpose of developing less computationally intensive lower bounding procedures, we propose strategies to reduce the number of variables in the formulation and study the complexity of solving many of its relaxations. Two Lagrangian relaxation schemes are then presented.

In Section 3, we develop two branch-and-bound (BB) algorithms for the problem. One of them is based on the Lagrangian relaxation scheme for the first formulation, while the other is based on one of the Lagrangian relaxation schemes suggested for the second. Given the difficulty to solve QMSTP in practice, these BB algorithms are implemented with parallel programming. As a result of new features suggested here for load balancing, our parallel algorithms obtain very high rates of parallel efficiency (around 80%).

In Section 4, we present computational experiments conducted on two instance sets from the literature. For the first time, many of these instances are solved to proven optimality. In particular, we provide optimality certificates for instances with up to 50 vertices.

In order to shorten the main text body, separation algorithms and proofs for some theoretical results are provided in an attached supplementary document. Detailed computational results for the algorithms introduced here are also provided there.

2 Formulations, Linear Programming and Lagrangian Relaxation Bounds

Given a subset $V' \subseteq V$ denote by $E(V') = \{\{u,v\} \in E : u,v \in V'\}$ the set of edges with both endpoints in V' and by $\delta(V') = \{\{u,v\} \in E : u \in V', v \notin V'\}$ the set of edges with exactly one endpoint in V'. Given a vector $\mathbf{x} = (x_i)_{i \in M}$ and $M' \subseteq M$, define $\mathbf{x}(M') = \sum_{i \in M'} x_i$. Given any formulation P for QMSTP, let $Z(P)$ denote its LP lower bounds. Define $\mathbb{B} = \{0,1\}$.

2.1 Lagrangian Bounds from a Partial RLT Application

Consider a vector $\mathbf{x} = (x_i)_{i \in E}$ of binary variables such that $x_i = 1$ if and only if edge $i \in E$ is selected to be part of the tree we are looking for. A canonical quadratic 0-1 programming formulation for QMSTP is given by:

$$\min \left\{ \sum_{i,j \in E} q_{ij} x_i x_j : \mathbf{x} \in X \cap \mathbb{B}^m \right\},$$

where X denotes the convex hull of the incidence vectors of spanning trees of G [11], i.e., the set of vectors in $\mathbb{R}^m$ that satisfy:

$$\mathbf{x}(E) = n-1, \tag{1}$$

$$\mathbf{x}(E(S)) \leq |S| - 1, \quad S \subset V, |S| \geq 2, \tag{2}$$

$$x_i \geq 0, \quad i \in E. \tag{3}$$

To obtain a linear 0-1 programming formulation for the QMSTP, we apply a scheme that consists of two steps:
Reformulation step. Each constraint (1)-(3) is multiplied by each variable x_i, $i \in E$, resulting in new, non-linear, constraints.
Linearization step. Linearization variables $\mathbf{y} = (\mathbf{y}_i)_{i \in E}$, where $\mathbf{y}_i = (y_{ij})_{j \in E}$, are introduced to replace the products $x_i x_j$, $i, j \in E$, in the non-linear constraints and in the objective function. The constraints obtained after linearization takes place, together with $y_{ij} = y_{ji}$, $i < j \in E$, and $y_{ii} = x_i$, $i \in E$, are added to (1)-(3), to obtain a new IP formulation for QMSTP. Observe that, in the linearization step, explicit distinction is made between $y_{ij} = x_i x_j$ and $y_{ji} = x_j x_i$. This is done so that a special structure in the resulting formulation can be exploited. For convenience, we also replace the powers x_i^2 by y_{ii} instead of simply x_i (see [7] for an exposition of such an approach for constrained quadratic 0-1 programs in general).

Once the linearization scheme has been applied, the following linear 0-1 formulation is obtained

$$F_1: \qquad \min \left\{ \sum_{i,j \in E} q_{ij} y_{ij} : (\mathbf{x}, \mathbf{y}) \in P_1 \cap \mathbb{B}^{m+m^2} \right\},$$

where P_1 refers to the polyhedral region defined by:

$$\mathbf{x} \in X, \tag{4}$$
$$\mathbf{y}_i \in X(x_i), \qquad i \in E, \tag{5}$$
$$y_{ij} = y_{ji}, \qquad i < j \in E, \tag{6}$$

and the symbol $X(x_i)$, $i \in E$, denotes the set of vectors in $\mathbb{R}^m$ that satisfy:

$$\mathbf{y}_i(E) = (n-1)x_i, \tag{7}$$
$$\mathbf{y}_i(E(S)) \leq (|S|-1)x_i, \quad S \subset V, |S| \geq 2, \tag{8}$$
$$y_{ii} = x_i, \tag{9}$$
$$y_{ij} \geq 0, \qquad j \in E. \tag{10}$$

Notice that if $x_i = 0$, constraints (7)-(10) imply that $\mathbf{y}_i = 0$. On the other hand, if $x_i = 1$, (7)-(10) describe the convex hull of the incidence vectors of the spanning trees of G containing edge i. Whenever $\mathbf{y}_i$ is the incidence vector of a spanning tree of G, in an abuse of language, it will be referred to as an *interaction tree* for i.

The reformulation process just applied consists of a partial application of the first level of the RLT [2, 30]. The complete RLT scheme would also involve the multiplication of (1)-(3) by $(1 - x_i)$, $i \in E$, followed by the linearization step. As such, the full application of the first RLT level would provide the following additional valid inequalities for QMSTP:

$$(\mathbf{x} - \mathbf{y}_i)(E(S)) \leq (|S|-1)(1-x_i), \qquad i \in E, S \subset V, |S| \geq 2, \tag{11}$$

which are obtained from (2). The multiplication of constraints (1) and (3) by $(1 - x_i), i \in E$, leads to redundant inequalities.

It was shown by Lee and Leung [21] that (8), (10), (11), the clique, the cut inequalities for the BQP [25] and an extension of the latter (also presented in [21]) define facets for the BQFP. LP based polynomial time separation algorithms were given for (8) and (11). We remark that these two sets of valid inequalities can also be separated by modified versions of the algorithm of Padberg and Wolsey [26] for separating subtour breaking constraints. We present these modifications in Section A of the supplementary document to this paper. In respect to the remaining inequalities proposed by [21], we are not aware of any polynomial time separation algorithm.

Preliminary computational experiments conducted here with the instances in [24] with $n = 15$ indicated that constraints (11) do not significantly strengthen bounds $Z(F_1)$. According to our testings, such LP bounds increased by only 0.08%, while the average computational time needed to evaluate them (by explicitly solving LPs) increased by 69.8%. That explains why our formulation and solution techniques do not consider these inequalities.

Due to the large number of variables and constraints, computing the bound $Z(F_1)$ by means of a cutting plane algorithm where inequalities (9) are dynamically separated is too time demanding. Therefore, we adopt an alternative strategy to that aim. We relax and dualize constraints (6) by attaching to them unconstrained Lagrangian multipliers $\theta = (\theta_{ij})_{i<j\in E}$ (assume $\theta_{ij} = -\theta_{ji}$ in case $i > j \in E$), to obtain the problem:

$$F_1^{(6)}(\theta): \qquad L_1^{(6)}(\theta) = \min \left\{ \sum_{i,j\in E} q'_{ij} y_{ij} : (\mathbf{x}, \mathbf{y}) \in P_1^{(6)} \cap \mathbb{B}^{m+m^2} \right\},$$

where polytope $P_1^{(6)}$ is obtained by relaxing (6) in the definition of P_1, i.e., $P_1^{(6)}$ is given by (4) and (5). Lagrangian modified costs are defined as $q'_{ij} = q_{ij} + \theta_{ij}$ for $i \neq j \in E$ and $q'_{ii} = q_{ii}$ for $i \in E$. The corresponding Lagrangian dual is:

$$DF_1: \qquad L_1^{*(6)} = \max \left\{ L_1^{(6)}(\theta) : \theta \in \mathbb{R}^{\frac{m(m-1)}{2}} \right\}.$$

In order to develop a procedure to solve $F_1^{(6)}$, let us investigate its optimal solution $(\overline{\mathbf{x}}, \overline{\mathbf{y}}) \in \mathbb{B}^{m+m^2}$ for a given a set of multipliers $\overline{\theta} \in \mathbb{R}^{\frac{m(m-1)}{2}}$. It is not difficult to check that, if $\overline{x}_i = 1$ for some $i \in E$, then $\overline{\mathbf{y}}_i$ is the incidence vector of a spanning tree that minimizes:

$$\overline{q}_i = \min \left\{ \sum_{j\in E} q'_{ij} y_{ij} : \mathbf{y}_i \in X(x_i = 1) \cap \mathbb{B}^m \right\}. \tag{12}$$

That means that the selection of edge i implies in a interaction cost given by $\overline{q}_i$. Consequently, one can solve $F_1^{(6)}(\overline{\theta})$ by computing the spanning tree that minimizes:

$$\overline{q}_0 = \min \left\{ \sum_{i\in E} \overline{q}_i x_i : \mathbf{x} \in X \cap \mathbb{B}^m \right\}, \tag{13}$$

and setting $\mathbf{y}$ appropriately. Algorithm 1 summarizes the main steps.

Algorithm 1:

1. For each edge $i \in E$, solve (12) and denote the solution vector by $\widetilde{\mathbf{y}}_i \in \mathbb{B}^m$.
2. Solve (13), denote by $\widetilde{\mathbf{x}} \in \mathbb{B}^m$ the solution vector.
3. An optimal solution $(\overline{\mathbf{x}}, \overline{\mathbf{y}}) \in \mathbb{B}^{m+m^2}$ of cost $L_1(\overline{\theta}) = \overline{q}_0$ for $F_1^{(6)}(\overline{\theta})$ is given by $\overline{\mathbf{x}} = \widetilde{\mathbf{x}}$ and $\overline{\mathbf{y}}_i = \widetilde{x}_i \widetilde{\mathbf{y}}_i$, $i \in E$.

When $\overline{\theta} = \mathbf{0}$, Algorithm 1 provides the Gilmore-Lawler lower bound for QMSTP. In our implementation, each minimum spanning tree problem is solved in $O(m \log n)$ time complexity (with Prim's algorithm [29]). As a result, Algorithm 1 runs in $O(m^2 \log n)$ time complexity.

In order to discuss the strength of the Lagrangian dual DF_1, we present the next result that states that $P_1^{(6)}$ has integer extreme points.

Proposition 1. *$P_1^{(6)}$ is an integral polytope.*

Proof. To simplify the notation, assume that the elements of E are referred as integers from 1 to m, i.e., $E = \{1, \ldots, m\}$. Denote by $\mathcal{T}$ the set of all incidence vectors of spanning trees of G and define $\mathcal{T}_i = \{\mathbf{t} \in \mathcal{T} : t_i = 1\}$, the set of all incidence vectors of spanning trees containing edge $i \in E$. Observe that any integer vector $(\overline{\mathbf{x}}, \overline{\mathbf{y}}) \in \mathbb{B}^{m+m^2}$ in $P_1^{(6)}$ has $\overline{\mathbf{x}} \in \mathcal{T}$ and $\overline{\mathbf{y}}_i = \overline{x}_i \mathbf{t}$, $\mathbf{t} \in \mathcal{T}_i$.

We will show that any vector in $P_1^{(6)}$ can be written as a convex combination of integer vectors in $P_1^{(6)}$, which will prove the claim.

Note that, by (4), in any vector $(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}}_1, \ldots, \widetilde{\mathbf{y}}_m) \in \mathbb{R}^{m+m^2}$ in $P_1^{(6)}$, $\widetilde{\mathbf{x}}$ is a convex combination of elements of $\mathcal{T}$, i.e.,

$$\widetilde{\mathbf{x}} = \sum_{\mathbf{t}_0 \in \mathcal{T}} \alpha^{\mathbf{t}_0} \mathbf{t}_0, \qquad \sum_{\mathbf{t}_0 \in \mathcal{T}} \alpha^{\mathbf{t}_0} = 1, \qquad \alpha^{\mathbf{t}_0} \geq 0, \mathbf{t}_0 \in \mathcal{T}, \tag{14}$$

and by (7)-(10), $\widetilde{\mathbf{y}}_i$ is a convex combination of elements of $\mathcal{T}_i$ subsequently multiplied by $\widetilde{x}_i$, i.e.,

$$\widetilde{\mathbf{y}}_i = \widetilde{x}_i \sum_{\mathbf{t}_i \in \mathcal{T}_i} \alpha_i^{\mathbf{t}_i} \mathbf{t}_i, \qquad \sum_{\mathbf{t}_i \in \mathcal{T}_i} \alpha_i^{\mathbf{t}_i} = 1, \qquad \alpha_i^{\mathbf{t}_i} \geq 0, \mathbf{t}_i \in \mathcal{T}_i \tag{15}$$

Observe that

$$(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}}) = \sum_{\mathbf{t}_0 \in \mathcal{T}} \alpha_0^{\mathbf{t}_0} \cdots \sum_{\mathbf{t}_m \in \mathcal{T}_m} \alpha_m^{\mathbf{t}_m} (\mathbf{t}_0, t_{01}\mathbf{t}_1, \ldots, t_{0m}\mathbf{t}_m),$$

and that for any choice of indices in the sum, the combination between parenthesis is a integer vector in $P_1^{(6)}$. Since by (14) and (15) each individual sum has value 1, we have

$$\sum_{\substack{\mathbf{t}_0 \in \mathcal{T} \\ \vdots \\ \mathbf{t}_m \in \mathcal{T}_m}} \alpha_0^{\mathbf{t}_0} \ldots \alpha_m^{\mathbf{t}_m} = 1,$$

which shows that the linear combination is convex. □

From proposition 1 and from a well known result in Lagrangian duality theory [14], we have the following result.

Corollary 1. $L_1^{*(6)} = Z(F_1)$. □

One of the Lagrangian relaxation algorithms introduced here, denoted Lag_1, is based on such a Lagrangian relaxation scheme. In order to solve the Lagrangian dual DF_1, we used the subgradient method [18]. The simplicity of the latter combined with the efficiency of Algorithm 1 provided a lower bounding scheme that is much more tractable than directly solving the LP relaxation of F_1, by means of a cutting plane algorithm.

Next, we investigate the strength of formulation F_1 compared to other formulations in the QMSTP literature.

Proposition 2. *Denote by F_{AX92} the formulation introduced by Assad and Xu [5], given by F_1 with constraints* (6) *replaced by*

$$\sum_{i \in E} y_{ij} = (n-1)x_j, \qquad j \in E. \tag{16}$$

Denote by F_{OP10} the formulation proposed by Öncan and Punnen [24], given by F_{AX92} plus the valid inequalities

$$\sum_{i \in \delta(v)} y_{ij} \geq x_j, \qquad j \in E, v \in V. \tag{17}$$

We have the following:

$$Z(F_{\text{AX92}}) \leq Z(F_{\text{OP10}}) \leq Z(F_1).$$

Proof. Constraints (16) are clearly implied by (7) and (6). To check that constraints (17) are also implied by F_1, formulate (8) in terms of an edge j and set $S = V \setminus \{v\}$, for a given $v \in V$. Then subtract the resulting inequality from (7) (also formulated for $j \in E$), to obtain:

$$\sum_{i \in \delta(v)} y_{ji} \geq x_j,$$

which together with (6) implies (17). □

Although formulation F_1 is at least as strong as F_{AX92} and F_{OP10}, duality gaps implied by $Z(F_1)$ are sometimes quite large. This observation motivates the study of stronger lower bounding approaches for QMSTP.

2.2 Lagrangian Bounds from a Subgraph Enumeration Formulation

For any factor $K > 0$ of $n-1$, the set of edges of any spanning tree of G can be partitioned into $(n-1)/K$ subsets of K edges each. Thus, in order to construct a spanning tree of G, one can combine $(n-1)/K$ of its acyclic subgraphs with K edges. This is the core idea behind a novel formulation for QMSTP introduced next.

Let E^K be the collection of all sets $H \subseteq E$ such that: (i) $|H| = K$ and (ii) the edges in H induce an acyclic subgraph of G. Let $o = |E^K|$. Define $E_i^K = \{H \in E^K : i \in H\}$ as the set of the elements of E^K that contain $i \in E$.

The formulation uses a vector of binary variables $\mathbf{s} = (s_H)_{H \in E^K}$, such that $s_H = 1$ if and only if $H \in E^K$ is selected to be part of the spanning tree we are looking for. The formulation also uses a vector of binary variables $\mathbf{t} = (\mathbf{t}_H)_{H \in E^K}$, where $\mathbf{t}_H = (t_{Hi})_{i \in E}$, $H \in E^K$. For the new formulation, $\mathbf{t}_H$ has a role similar to that of $\mathbf{y}_i$ in F_1. For F_1, $\mathbf{y}_i$ defines an interaction tree for edge $i \in E$. Likewise, $\mathbf{t}_H$ defines an interaction tree for the set of edges H. That being stated, QMSTP can be formulated as:

$$F_2: \qquad \min \left\{ \sum_{i,j \in E} q_{ij} y_{ij} : (\mathbf{x}, \mathbf{y}, \mathbf{s}, \mathbf{t}) \in P_2 \cap \mathbb{B}^{m+m^2+o+om} \right\},$$

where polytope P_2 is given by:

$$\begin{aligned} &\mathbf{x} \in X, && && (18)\\ &x_i = \mathbf{s}(E_i^K), && i \in E, && (19)\\ &\mathbf{y}_i = \mathbf{t}(E_i^K), && i \in E, && (20)\\ &\mathbf{t}_H \in X(s_H), && H \in E^K, && (21)\\ &y_{ij} = y_{ji}, && i < j \in E, && (22) \end{aligned}$$

and the symbol $X(s_H)$ denotes the set of points in $\mathbb{R}^m$ that satisfy:

$$\begin{aligned} &\mathbf{t}_H(E) = (n-1)s_H, && && (23)\\ &\mathbf{t}_H(E(S)) \le (|S|-1)s_H, && S \subset V, |S| \ge 2, && (24)\\ &t_{Hi} = s_H, && i \in H, && (25)\\ &t_{Hi} \ge 0, && i \in E. && (26) \end{aligned}$$

Observe that the size of E^K is $O(m^K)$, which is polynomial in n if K is a constant. In the case $K = 1$, formulations F_1 and F_2 are equivalent. If $K = n-1$, E^K will be the set of all spanning trees of G and $Z(F_2)$ will be the optimal solution value of the QMSTP. For other values of K, we have the following result.

Proposition 3. *Given a factor $K > 0$ of $n-1$, denote by $P_2(K)$ the polytope defined by (18)-(22) for this particular value of K. Denote by $Proj_{\mathbf{xy}}(P_2(K))$ the projection of $P_2(K)$ onto the vector space of the variables $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{m+m^2}$. For two factors K and L of $n-1$, $L > K$, the following holds:*

$$Proj_{\mathbf{xy}}(P_2(L)) \subseteq Proj_{\mathbf{xy}}(P_2(K)).$$

Proof. Let $o(K) = |E^K|$ and $o(L) = |E^L|$. Consider a vector $(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{s}}, \overline{\mathbf{t}}) \in P_2(L) \subseteq \mathbb{R}^{m+m^2+o(L)+o(L)m}$. We are going to show that there is a vector $(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}}, \widetilde{\mathbf{s}}, \widetilde{\mathbf{t}}) \in P_2(K) \subseteq \mathbb{R}^{m+m^2+o(K)+o(K)m}$ such that $\widetilde{\mathbf{x}} = \overline{\mathbf{x}}$ and $\widetilde{\mathbf{y}} = \overline{\mathbf{y}}$.

For every $H \in E^L$ define the set $E^K(H) = \{I \in E^K : H \cap I = K\}$, i.e., $E^K(H)$ is the set of the subsets of H that contain K edges. Observe that each edge of H appears in exactly $c = (L-1)!/((L-K)!(K-1)!)$ elements of $E^K(H)$.

Now, consider

$$\widetilde{s}_I = \frac{1}{c} \sum_{H \in E^L : I \in E^K(H)} \overline{s}_H,$$

for all $I \in E^K$. Thus,

$$\overline{x}_i = \sum_{H \in E_i^L} \overline{s}_H = \sum_{H \in E_i^L} \frac{1}{c} \sum_{I \in E^K(H): i \in I} \overline{s}_H = \sum_{I \in E_i^K} \frac{1}{c} \sum_{H \in E^L : I \in E^K(H)} \overline{s}_H = \sum_{I \in E_i^K} \widetilde{s}_I = \widetilde{x}_i,$$

which shows that (19) is satisfied in $P_2(K)$.

For every $I \in E^K$, consider

$$\widetilde{\mathbf{t}}_I = \frac{1}{c} \sum_{H \in E^L : I \in E^K(H)} \overline{\mathbf{t}}_H.$$

From this,

$$\sum_{i \in E} \widetilde{t}_{Ii} = \sum_{i \in E} \frac{1}{c} \sum_{H \in E^L : I \in E^K(H)} \overline{t}_{Hi} = \frac{1}{c} \sum_{H \in E^L : I \in E^K(H)} (n-1)\overline{s}_H = (n-1)\widetilde{s}_I,$$

which proves the satisfaction of the cardinality constraint (23) in $X(\widetilde{s}_I)$.

The satisfaction of the remaining constraints in $X(\widetilde{s}_I)$ can be proved in a similar fashion. Finally, for (20) we have

$$\overline{\mathbf{y}}_i = \sum_{H \in E_i^L} \overline{\mathbf{t}}_H = \sum_{H \in E_i^L} \frac{1}{c} \sum_{I \in E^K(H): i \in I} \overline{\mathbf{t}}_H = \sum_{I \in E_i^K} \frac{1}{c} \sum_{H \in E^L : I \in E^K(H)} \overline{\mathbf{t}}_H = \sum_{I \in E_i^K} \widetilde{\mathbf{t}}_I = \widetilde{\mathbf{y}}_i,$$

which completes the proof. □

It might be the case that a given spanning tree of G can be partitioned into subgraphs of K edges in many different ways. However, we only need to grant that at least one such decomposition does exist. By eliminating redundant possibilities from E^K, we can reduce the number of variables of F_2 and improve its LP relaxation lower bound. The next result shows how that can be accomplished for $K \leq 4$.

Proposition 4. *Let T be a spanning tree of G and K be a factor of $n-1$. If $K = 2$, the set of edges of T can be partitioned into subsets of two adjacent edges each. If $K = 3$ or $K = 4$, then the set of edges of T can be partitioned into subsets of K edges, each subset inducing no more than two connected components of G.*

Proof. Note that, for $K = 2$, either T has edges $\{i,j\}$ and $\{j,k\}$ such that i and k are leaves, or i is a leaf and j is not connected to any vertex other than i or k. No matter the case, we remove these two edges to obtain a subgraph of T that is connected and has an even number of edges. The argument is then applied recursively.

For $K = 3$, remove $(1/3)(n-1)$ edges $\{i,j\}$ of T, one at a time, under the condition that i is a leaf. The remaining subgraph has $(2/3)(n-1)$ edges and is connected; apply the procedure for $K = 2$ to this subgraph. For each resulting set of two edges add one of the edges that were previously removed.

For $K = 4$, apply the procedure for $K = 2$, group the resulting pairs of adjacent edges into sets of four edges. □

Even in the light of Proposition 4, computing $Z(F_2)$ by explicitly solving LPs is impractical, due to the large number of variables and constraints. As an attempt to speed up the computation of lower bounds derived from F_2, we investigate the relaxation and the dualization of constraints (22), in a Lagrangian fashion. To that aim, consider again that unconstrained multipliers $\theta = (\theta_{ij})_{i<j \in E}$ are assigned to (22), $\theta_{ij} = -\theta_{ji}$ if $i > j \in E$. Such a relaxation strategy leads to the following Lagrangian subproblem:

$$F_2^{(22)}(\theta): \quad L_2^{(22)}(\theta) = \min \left\{ \sum_{i,j \in E} q'_{ij} y_{ij} : (\mathbf{x}, \mathbf{y}, \mathbf{s}, \mathbf{t}) \in P_2^{(22)} \cap \mathbb{B}^{m+m^2+o+om} \right\},$$

where $P_2^{(22)}$ is obtained by relaxing (22) in P_2, i.e., $P_2^{(22)}$ is represented by (18)-(21). Lagrangian modified costs are defined as $q'_{ij} = q_{ij} + \theta_{ij}$ for $i \neq j \in E$ and $q'_{ii} = q_{ii}$ for $i \in E$.

Observe that, by (20), the objective function of $F_2^{(22)}$ can be written as:

$$\sum_{i,j \in E} q'_{ij} y_{ij} = \sum_{i,j \in E} \sum_{H \in E_i^K} q'_{ij} t_{Hj} = \sum_{H \in E^K} \sum_{i \in H} \sum_{j \in E} q'_{ij} t_{Hj}.$$

Therefore, using the fact that in $F_2^{(22)}$ the choice of the vector $\mathbf{t}_H$ depends only on s_H, it can be concluded that in an optimal solution $(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{s}}, \overline{\mathbf{t}}) \in \mathbb{B}^{m+m^2+o+om}$ for $F_2^{(22)}(\overline{\theta})$, $\overline{\theta} \in \mathbb{R}^{\frac{m(m-1)}{2}}$, if $\overline{s}_H = 1$, $\overline{\mathbf{t}}_H$ will be the incidence vector of the spanning tree that minimizes:

$$\overline{q}_H = \min \left\{ \sum_{i \in H} \sum_{j \in E} q'_{ij} t_{Hj} : \mathbf{t}_H \in X(s_H = 1) \cap \mathbb{B}^m \right\}. \tag{27}$$

Thus, problem $F_2^{(22)}$ can be solved with the resolution of

$$\overline{q}_0 = \min \left\{ \sum_{H \in E^K} \overline{q}_H s_H : \mathbf{x} \in X, x_i = \mathbf{s}(E_i^K), \forall i \in E, (\mathbf{x}, \mathbf{s}) \in \mathbb{B}^{m+o} \right\}, \tag{28}$$

followed by the appropriate adjustment of $(\mathbf{y}, \mathbf{t}) \in \mathbb{B}^{m^2+om}$. This process is summarized in the following algorithm.

Algorithm 2:

1. For every $H \in E^K$, solve (27) and denote by $\widetilde{\mathbf{t}}_H \in \mathbb{B}^m$ its solution vector.
2. Solve problem (28) to obtain a solution $(\widetilde{\mathbf{x}}, \widetilde{\mathbf{s}}) \in \mathbb{B}^{m+o}$.
3. Obtain a solution $(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{s}}, \overline{\mathbf{t}}) \in \mathbb{B}^{m+m^2+o+om}$ of cost $L_2^{(22)}(\overline{\theta}) = \overline{q}_0$ for $F_2^{(22)}(\overline{\theta})$ by making $\overline{\mathbf{x}} = \widetilde{\mathbf{x}}$, $\overline{\mathbf{s}} = \widetilde{\mathbf{s}}$, $\overline{\mathbf{t}}_H = \overline{s}_H \widetilde{\mathbf{t}}_H, H \in E^K$, and $\overline{\mathbf{y}}_i = \overline{\mathbf{t}}(E_i^K)$, $i \in E$.

While Algorithm 2 actually solves $F_2^{(22)}$, Proposition 5 shows that the problem is in fact NP-Hard for $K \geq 3$. Consequently, it is unlikely that one can come up with an efficient algorithm to solve (28).

Proposition 5. *Problem $F_2^{(22)}$ is NP-Hard for $K \geq 3$*

Proof. The idea of the proof is to present a polynomial reduction from the problem of finding a minimum spanning tree of a k-uniform hypergraph [4] to $F_2^{(22)}$. The detailed proof is presented in Section B of the supplementary document. □

Given the complexity of solving $F_2^{(22)}$, we study two possible alternative approaches for deriving lower bounds from relaxations of formulation F_2.

2.2.1 First Approach - Selecting Edge-disjoint Subgraphs

Consider the subtour elimination constraints (2). Note that for integer solutions of F_2, these constraints are already implied by the remaining ones in the formulation. To check that, observe that

$$y_{ij} = y_{ji} \leq x_j, \qquad i, j \in E,$$

and as $\mathbf{y}_i$ defines a spanning tree, $\mathbf{x}$ also defines a spanning tree. Furthermore, if we also relax and dualize (2) in $F_2^{(22)}$, we obtain a Lagrangian subproblem that is easy to solve for a particular value of K. Such an approach is discussed next.

Consider unconstrained dual multipliers $\theta = (\theta_{ij})_{i<j \in E}$ and non-negative multipliers $\mu = (\mu_S)_{S \subseteq V, |S| \geq 2}$, respectively associated to (22) and (2). Assume $\theta_{ij} = -\theta_{ji}$ for $i > j \in E$. Once the constraints are dualized, the following Lagrangian subproblem results:

$F_2^{(22),(2)}(\theta, \mu)$:

$$L_2^{(22),(2)}(\theta, \mu) = C + \min \left\{ \sum_{i,j \in E} q'_{ij} y_{ij} : (\mathbf{x}, \mathbf{y}, \mathbf{s}, \mathbf{t}) \in P_2^{(22),(2)} \cap \mathbb{B}^{m+m^2+o+om} \right\},$$

where $P_2^{(22),(2)}$ is obtained by relaxing (22) and (2) in P_2, i.e., $P_2^{(22),(2)}$ is defined by (1), (3), (19)-(21). Lagrangian costs q'_{ij} are defined as $q'_{ij} = q_{ij} + \theta_{ij}$ if $i \neq j \in E$, $q'_{ii} = q_{ii} + \sum_{S \subseteq V : i \in E(S)} \mu_S$, $i \in E$, and $C = -\sum_{S \subseteq V, |S| \geq 2} \mu_S$.

The associated Lagrangian dual is:

$$DF_2 \qquad : \qquad L_2^{*(22),(2)} \qquad = \qquad \max\left\{L_2^{(22),(2)}(\theta,\mu) : (\theta,\mu) \in \mathbb{R}^{\frac{m(m-1)}{2}} \times \mathbb{R}_+^{|\{S\subseteq V,|S|\geq 2\}|}\right\}.$$

In order to develop a procedure for solving $F_2^{(22),(2)}$, we study an optimal solution $(\overline{\mathbf{x}},\overline{\mathbf{y}},\overline{\mathbf{s}},\overline{\mathbf{t}}) \in \mathbb{B}^{m+m^2+o+om}$ for that problem, given a choice $(\overline{\theta},\overline{\mu}) \in \mathbb{R}^{\frac{m(m-1)}{2}} \times \mathbb{R}_+^{|\{S\subseteq V,|S|\geq 2\}|}$ of multipliers. We note that if $\overline{s}_H = 1$, then $\overline{\mathbf{t}}_H$ will be the incidence vector of the spanning tree that minimizes (27). Also, after the relaxation of (2), variables $\mathbf{x}$ can be projected out of the formulation. This way, $F_2^{(22)}(\overline{\theta},\overline{\mu})$ can be solved with the resolution of

$$\overline{q}_0 = C + \min\left\{\sum_{H\in E^K} \overline{q}_H s_H : \mathbf{s}(E_i^K) \leq 1, i \in E, \mathbf{s}(E) = \frac{n-1}{K}, \mathbf{s} \in \mathbb{B}^o\right\}, \tag{29}$$

and the appropriate adjustment of $(\mathbf{x},\mathbf{y},\mathbf{t}) \in \mathbb{B}^{m+m^2+om}$. Problem (29) is a set packing problem with an additional cardinality constraint. This problem can be efficiently solved for $K=2$ as it will be shown later. However, for $K \geq 3$, (29) is NP-Hard, which implies that $F_2^{(22),(2)}$ is also NP-Hard.

Proposition 6. *Problem $F_2^{(22),(2)}$ is NP-Hard for $K \geq 3$.*

Proof. The idea of the proof is to present a polynomial time reduction from the maximum weight k-set packing problem [17] to $F_2^{(22),(2)}$. The detailed proof is given in Section B of the supplementary document. □

If $K = 2$, problem (29) states that $(n-1)/2$ disjoint elements of E^K must be selected. In other words, one has to find a minimum cost matching of cardinality $(n-1)/2$, in an auxiliary graph $\overline{G} = (\overline{V},\overline{E})$, defined with vertex set $\overline{V} = E$ and edge set $\overline{E} = E^K$.

In order to solve this problem, consider the auxiliary graph $\overline{G}$. Add to $\overline{V}$ a set U of $m-(n-1)$ auxiliary vertices and add to $\overline{E}$ edges of zero cost, connecting each vertex from U to all the vertices in $\overline{V}\setminus U$. Since G is connected, the subgraph of $\overline{G}$ induced by $\overline{V}\setminus U$ has a matching of $(n-1)/2$ edges ($n-1$ vertices), while the $m-(n-1)$ remanescent non-matched vertices can be matched without additional costs to the vertices of U. Conversely, for any perfect matching of $\overline{G}$, each one of the $m-(n-1)$ vertices in U needs to be matched to vertices in $\overline{V}\setminus U$, letting $(n-1)$ vertices of $\overline{V}\setminus U$ to be matched among themselves, what results in a matching of cardinality $(n-1)/2$ and the same cost for the subgraph of $\overline{G}$ induced by $\overline{V}\setminus U$. Thus, there is an equivalence between perfect matchings of $\overline{G}$ and matchings of cardinality $(n-1)/2$ of the subgraph of $\overline{G}$ induced by $\overline{V}\setminus U$.

Consequently, in order to solve $F_2^{(22),(2)}(\overline{\theta},\overline{\mu})$ when $K=2$, we can proceed by computing the costs (27), followed by the resolution of (29) as outlined above. Algorithm 3 summarizes the main steps.

Algorithm 3:

1. Solve problem (27) for each $H \in E^K$ and let $\widetilde{\mathbf{t}}_H \in \mathbb{B}^m$ be the minimizing vector.
2. Solve (29) as described above to obtain a solution $\widetilde{\mathbf{s}} \in \mathbb{B}^o$.
3. Obtain a solution $(\overline{\mathbf{x}},\overline{\mathbf{y}},\overline{\mathbf{s}},\overline{\mathbf{t}}) \in \mathbb{B}^{m+m^2+o+om}$ of cost $L_2^{(22),(2)}(\overline{\theta},\overline{\mu}) = \overline{q}_0$ for $F_2^{(22),(2)}(\overline{\theta},\overline{\mu})$ by letting $\overline{\mathbf{s}} = \widetilde{\mathbf{s}}$. If $\overline{s}_H = 1$ for some $H \in E_i^K$ and $i \in E$, let $\overline{x}_i = 1$, otherwise $\overline{x}_i = 0$. Let $\overline{\mathbf{t}}_H = \overline{s}_H\widetilde{\mathbf{t}}_H, H \in E^K$, and $\overline{\mathbf{y}}_i = \overline{\mathbf{t}}(E_i^K)$, $i \in E$.

The first step of Algorithm 3 runs in $O(om\log n) = O(m^3\log n)$ time complexity. We employ an algorithm of time complexity $O(|\overline{V}|^2|\overline{E}|)$ [15] for solving the minimum cost perfect matching problem. This way, step 2 is performed in $O(m^4)$, which determines the worst case time complexity of the algorithm.

As a result of the discussion above, the solutions to the Lagrangian subproblem $F_2^{(22),(2)}(\theta,\mu)$ implicitly satisfy all valid inequalities for the matching polytope. Note also that blossom inequalities [13] (facet defining inequalities for the matching polytope) are missing from $F_2^{(22),(2)}$. Therefore, the Lagrangian dual bound provided by DF_2 might well be stronger than $Z(F_2)$.

Corollary 2. *$L_2^{*(22),(2)} \geq Z(F_2)$.* □

The evaluation (approximation) of $L_2^{*(22),(2)}$ requires finding optimal (near optimal) multipliers for an exponential number of constraints (2). One of the known algorithmic alternatives to deal with exponentially many inequalities candidates to Lagrangian dualization is the relax-and-cut approach [22]. Due to the already excessive number (though polynomial in n, m) of other dualized constraints, the benefits of implementing a relax-and-cut algorithm for the evaluation of $L_2^{*(22),(2)}$ are quite small: in practice, small lower bound improvements are obtained at a substantial increase of CPU time. For these reasons, we decided to set $\mu = \mathbf{0}$ in our implementation, i.e., we do not update multipliers μ in the course of the subgradient method. We denote this Lagrangian relaxation scheme by Lag_2.

2.2.2 Second Approach - Variable Splitting

In this section, we reformulate F_2 by rewriting its objective function and augmenting its variable space, keeping intact, however, the set of integer feasible solutions in $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{m+m^2}$. The LP relaxation bounds are also unchanged and a Lagrangian relaxation scheme is developed for their evaluation.

Consider the replacement of each variable s_H, $H \in E^K$, by K new binary variables s_{iH}, one for each $i \in H$, so that now we have $\mathbf{s} = (\mathbf{s}_i)_{i \in E}$ and $\mathbf{s}_i = (s_{iH})_{H \in E_i^K}$, $i \in E$. Likewise, consider the replacement of each vector $\mathbf{t}_H$, $H \in E^K$, by K new binary vectors $\mathbf{t}_{iH}$, $i \in H$. Therefore, $\mathbf{t} = (\mathbf{t}_i)_{i \in E}$, $\mathbf{t}_i = (\mathbf{t}_{iH})_{H \in E_i^K}$, and $\mathbf{t}_{iH} = (t_{iHj})_{j \in E}$. QMSTP can be formulated as:

$$F_3 : \qquad \min \left\{ \sum_{i,j \in E} q_{ij} y_{ij} : (\mathbf{x}, \mathbf{y}, \mathbf{s}, \mathbf{t}) \in P_3 \cap \mathbb{B}^{m+m^2+Ko+Kom} \right\}, \tag{30}$$

where P_3 denotes the polytope given by:

$$\begin{aligned}
&\mathbf{x} \in X, && && (31)\\
&x_i = \mathbf{s}_i(E_i^K), && i \in E, && (32)\\
&\mathbf{y}_i = \mathbf{t}_i(E_i^K), && i \in E, && (33)\\
&\mathbf{t}_{iH} \in X(s_{iH}), && i \in E, H \in E_i^K, && (34)\\
&y_{ij} = y_{ji}, && i < j \in E, && (35)\\
&s_{iH} = s_{jH}, && i < j \in E, H \in E_i^K \cap E_j^K. && (36)
\end{aligned}$$

Later on, we show that the relaxation of constraints (35) and (36) results in a problem that is easy to solve for any factor K, what allows us to develop a tractable lower bounding procedure based on F_3. Before discussing that, observe that constraints of type (36) were not imposed for $\mathbf{t}$, which implies that $Z(F_3)$ may be weaker than $Z(F_2)$. However, by conveniently rewriting the objective function in F_3, we show that does not apply. To that aim, notice that if $t_{iHj} = 1$, $i, j \in E$, $H \in E_i^K$, then

$$\sum_{k \in H} t_{kHj} = K. \tag{37}$$

Using (33) and (37), the objective function in (30) can be rewritten as:

$$\sum_{i,j \in E} q_{ij} y_{ij} = \sum_{i \in E} \sum_{H \in E_i^K} \sum_{j \in E} q_{ij} t_{iHj} = \sum_{i \in E} \sum_{H \in E_i^K} \sum_{j \in E} \sum_{k \in H} \frac{1}{K} q_{ij} t_{kHj} t_{iHj}$$

Now, note that $t_{iHj} t_{kHj} = t_{iHj} = t_{kHj}$ for $i, j, k \in E$, $H \in E_i^K \cap E_k^K$. Therefore

$$\begin{aligned}
\sum_{i \in E} \sum_{H \in E_i^K} \sum_{j \in E} \sum_{k \in H} \frac{1}{K} q_{ij} t_{kHj} t_{iHj} &= \sum_{i \in E} \sum_{H \in E_i^K} \sum_{j \in E} \sum_{k \in H} \frac{1}{K} q_{ij} t_{kHj} \\
&= \sum_{i \in E} \sum_{H \in E_i^K} \sum_{j \in E} \sum_{k \in H} \frac{1}{K} q_{kj} t_{iHj}
\end{aligned} \tag{38}$$

In other words, (38) states that the cost of the tree defined by $\mathbf{t}_{iH}$, $i \in E$, $H \in E_i^K$, depends equally on all the edges in H and their interaction costs. In the remainder of this section, we assume that the objective in (30) is rewritten according to (38). Bearing that in mind, we have the following result.

Proposition 7. $Z(F_2) = Z(F_3)$.

Proof. We make use of an argument based on the application of Lagrangian relaxation to the LP relaxations of F_2 and F_3. We dualize constraints (35) with unconstrained Lagrangian multipliers $\theta = (\theta_{ij})_{i<j\in E}$, $\theta_{ij} = -\theta_{ji}$. This gives the the Lagrangian subproblem

$$F_3^{(35)}(\theta): \qquad L_3^{(35)}(\theta) = \min\left\{ \sum_{i\in E}\sum_{H\in E_i^K}\sum_{j\in E}\sum_{k\in H} \frac{1}{K} q'_{kj} t_{iHj} \right.$$
$$\left. : (\mathbf{x},\mathbf{y},\mathbf{s},\mathbf{t}) \in P_3^{(35)} \cap \mathbb{B}^{m+m^2+Ko+Kom} \right\},$$

where $P_3^{(35)}$ is obtained by relaxing (35) in P_3, i.e., $P_3^{(35)}$ is defined by (31)-(34) and (36). Lagrangian modified costs are defined as $q'_{ij} = q_{ij} + \theta_{ij}$ for $i \neq j \in E$ and $q'_{ii} = q_{ii}$ for $i \in E$.

We now show that for any $\overline{\theta} \in \mathbb{R}^{\frac{m(m-1)}{2}}$, $Z(F_3^{(35)}(\overline{\theta})) = Z(F_2^{(22)}(\overline{\theta}))$, which proves the claim.

Given a feasible solution $(\overline{\mathbf{x}},\overline{\mathbf{y}},\overline{\mathbf{s}},\overline{\mathbf{t}}) \in \mathbb{R}^{m+m^2+o+om}$ for the LP relaxation of $F_2^{(22)}(\overline{\theta})$, letting $\widetilde{\mathbf{x}} = \overline{\mathbf{x}}$, $\widetilde{\mathbf{y}} = \overline{\mathbf{y}}$, $\widetilde{s}_{iH} = \overline{s}_H$, and $\widetilde{\mathbf{t}}_{iH} = \overline{\mathbf{t}}_H$, $i \in H$, $H \in E_i^K$, we obtain a feasible solution $(\widetilde{\mathbf{x}},\widetilde{\mathbf{y}},\widetilde{\mathbf{s}},\widetilde{\mathbf{t}}) \in \mathbb{R}^{m+m^2+Ko+Kom}$, with the same objective value of the LP relaxation of $F_3^{(35)}(\overline{\theta})$.

Conversely, given a solution for the linear relaxation of $F_3^{(35)}(\overline{\theta})$, there is always a feasible solution $(\widetilde{\mathbf{x}},\widetilde{\mathbf{y}},\widetilde{\mathbf{s}},\widetilde{\mathbf{t}}) \in \mathbb{R}^{m+m^2+Ko+Kom}$ with the same objective value, such that $\widetilde{\mathbf{t}}_{iH} = \widetilde{\mathbf{t}}_{jH}$, $i, j \in E$, $H \in E_i^K \cap E_j^K$. Letting $\overline{\mathbf{x}} = \widetilde{\mathbf{x}}$, $\overline{\mathbf{y}} = \widetilde{\mathbf{y}}$, $\overline{s}_H = \widetilde{s}_{iH}$, and $\overline{\mathbf{t}}_H = \widetilde{\mathbf{t}}_{iH}$, for any $i \in H$, $H \in E^K$, we obtain a feasible solution $(\overline{\mathbf{x}},\overline{\mathbf{y}},\overline{\mathbf{s}},\overline{\mathbf{t}}) \in \mathbb{R}^{m+m^2+o+om}$ with the same objective value for the LP relaxation of $F_2^{(22)}(\overline{\theta})$. □

When constraints (35) and (36) are relaxed in F_3, we obtain a problem that is easy to solve for any value of K. Thus, we again apply Lagrangian relaxation once again. Assume that unconstrained dual multipliers $\theta = (\theta_{ij})_{i<j\in E}$ and $\pi = (\pi_{ijH})_{i<j\in E, H\in E_i^K\cap E_j^K}$ are respectively attached to (35) and (36). For $i > j \in E$, assume $\theta_{ij} = -\theta_{ji}$, $\pi_{iiH} = 0$, and $\pi_{ijH} = -\pi_{jiH}$, $H \in E_i^K \cap E_j^K$. We obtain the Lagrangian subproblem:

$$F_3^{(35)(36)}(\theta,\pi):$$
$$L_3^{(35)(36)}(\theta,\pi) = \min\left\{ \sum_{i\in E}\sum_{H\in E_i^K}\sum_{j\in H} (\pi_{ijH} s_{iH} + \sum_{k\in E} \frac{1}{K} q'_{jk} t_{iHk}) \right.$$
$$\left. : (\mathbf{x},\mathbf{y},\mathbf{s},\mathbf{t}) \in P_3^{(35)(36)} \cap \mathbb{B}^{m+m^2+Ko+Kom} \right\},$$

where $P_3^{(35)(36)}$ is obtained by relaxing (35) and (36) in P_3, i.e., $P_3^{(35)(36)}$ is defined by (31)-(34). The modified costs are defined as $q'_{ij} = q_{ij} + \theta_{ij}$ for $i \neq j \in E$ and $q'_{ii} = q_{ii}$ for $i \in E$. The associated Lagrangian dual is:

$$DF_3: \qquad L_3^{*(35)(36)} = \max\left\{ L_3^{(35)(36)}(\theta,\pi) : (\theta,\pi) \in \mathbb{R}^{\frac{m(m-1)}{2}+\frac{oK(K-1)}{2}} \right\}.$$

To see how $F_3^{(35)(36)}$ can be solved, we investigate an optimal solution $(\overline{\mathbf{x}},\overline{\mathbf{y}},\overline{\mathbf{s}},\overline{\mathbf{t}}) \in \mathbb{B}^{m+m^2+Ko+Kom}$ for that problem, given a choice $(\overline{\theta},\overline{\pi}) \in \mathbb{R}^{\frac{m(m-1)}{2}+\frac{oK(K-1)}{2}}$ of Lagrangian multipliers. We see that if $\overline{x}_i = 1$ for some $i \in E$, and $\overline{s}_{iH} = 1$ for some $H \in E_i^K$, then $\overline{\mathbf{t}}_{iH}$ is the incidence vector of the spanning tree that minimizes

$$\overline{q}_{iH} = \frac{1}{K} \min\left\{ \sum_{j\in H}\sum_{k\in E} q'_{jk} t_{iHk} : \mathbf{t}_{iH} \in X(s_{iH} = 1) \cap \mathbb{B}^m \right\}. \tag{39}$$

Therefore, if $\overline{x}_i = 1$, $i \in E$, we will have $\overline{s}_{iH} = 1$ for the element of H_i^K that minimizes

$$\overline{q}_i = \min\left\{ \overline{q}_{iH} + \sum_{j\in H} \pi_{ijH} : H \in E_i^K \right\}. \tag{40}$$

Thus, $F_3^{(35)(36)}$ can be solved by solving

$$\overline{q}_0 = \min\left\{\sum_{i\in E} \overline{q}_i x_i : x \in X \cap \mathbb{B}^m\right\}, \tag{41}$$

followed by the appropriate adjustment of $\mathbf{y} \in \mathbb{B}^{m^2}$, $\mathbf{s} \in \mathbb{B}^{Ko}$, and $\mathbf{t} \in \mathbb{B}^{Kom}$. The following algorithm summarizes how $F_3^{(35)(36)}$ is solved.

Algorithm 4:

1. Solve (39) and obtain $\overline{q}_{iH}$ for each $i \in E$ and $H \in E_i^K$. Denote the minimizing vector by $\widetilde{\mathbf{t}}_{iH}$. Observe that (39) needs to be solved only once for each $H \in E^K$, i.e., find $\overline{q}_{iH}$ for some $i \in H$ and let $\overline{q}_{jH} = \overline{q}_{iH}$ and $\widetilde{\mathbf{t}}_{jH} = \widetilde{\mathbf{t}}_{iH}$ for $j \neq i \in H$.
2. Solve (40) for each $i \in E$ to obtain $\overline{q}_i$. Let $\widetilde{s}_{iH} = 1$ for the minimizing element $H \in E_i^K$ and $\widetilde{s}_{iI} = 0$ for the remaining $I \neq H \in E_i^K$.
3. Solve the minimum spanning tree problem in (41) and denote the minimizing vector by $\widetilde{\mathbf{x}}$.
4. Obtain a solution vector $(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{s}}, \overline{\mathbf{t}}) \in \mathbb{B}^{m+m^2+Ko+Kom}$ of cost $\overline{q}_0 = L_3^{(35)(36)}(\overline{\theta}, \overline{\pi})$ for $F_3^{(35)(36)}(\overline{\theta}, \overline{\pi})$ by letting $\overline{\mathbf{x}} = \widetilde{\mathbf{x}}$, $\overline{s}_{iH} = \widetilde{x}_i \widetilde{s}_{iH}$ and $\overline{\mathbf{t}}_{iH} = \overline{s}_{iH} \widetilde{\mathbf{t}}_{iH}$, $i \in E$, $H \in E_i^K$, and $\overline{\mathbf{y}}_i = \overline{\mathbf{t}}_i(E_i^K)$, $i \in E$.

Steps 1 and 2 can be executed in $O(om \log n)$ and $O(Ko)$ time, respectively. Step 3 takes $O(m \log n)$ elementary operations. Thus, Algorithm 4 runs in $O(om \log n)$ time.

In order to evaluate the strength of DF_3, we first show the following.

Proposition 8. *$P_3^{(35)(36)}$ is an integral polytope.*

Proof. The proof is quite similar to the proof of Proposition 1 and is presented in Section B of the supplementary document. □

In the light of Propositions 7 and 8, we have the following result.

Corollary 3. $L_3^{*(35)(36)} = Z(F_2) = Z(F_3)$. □

To solve DF_3, we have to deal with the large number of dualized constraints, $O(Ko + m^2)$ to be more precise. As a consequence of convergence difficulties we were faced with when we first applied standard subgradient optimization to this relaxation, we employ the heuristic described next to adjust the multipliers assigned to (36). The Lagrangian multipliers for (35) are adjusted according to the subgradient method.

Firstly, let us we clarify the reasoning behind the heuristic. Given a certain $\overline{H}$ that minimizes (40), observe that for any $H \neq \overline{H} \in E_i^K$, $\overline{q}_{iH} + \sum_{j\in H} \pi_{ijH} \geq \overline{q}_{i\overline{H}} + \sum_{j\in \overline{H}} \pi_{ij\overline{H}}$. Notice that there is a margin for the decrease of π_{ijH}, $j \in H$, without any change in $\overline{q}_i$. This decrease, and consequently the increase of π_{jiH}, can cause the increase of $\overline{q}_j$, in case H is the minimizing element of (40) for j. Note that $\sum_{j\in H} \pi_{ijH}$ can be decreased by at most $\lambda = \overline{q}_{iH} + \sum_{j\in H} \pi_{ijH} - (\overline{q}_{i\overline{H}} + \sum_{j\in\overline{H}} \pi_{ij\overline{H}})$, without resulting in any alteration in $\overline{q}_i$. These ideas are employed in the algorithm below, during the resolution of $F_3^{(35)(36)}$.

Algorithm 5:

1. Let $\overline{q}_i = \infty$ for all $i \in E$.
2. For each $H \in E^K$:
   (a) Let $\widetilde{\mathbf{t}}_H$ be the minimizing vector of

   $$\overline{q}_H = \min\{\sum_{j\in H}\sum_{k\in E} q'_{jk} t_{Hk} : \mathbf{t}_H \in X \cap \mathbb{B}^m : t_{Hj} = 1 \forall j \in H\}$$

   (b) Consider $\lambda_i = \overline{q}_H / K$ for all $i \in H$.
   (c) Assume an ordering $(e_1, \ldots, e_K)$ for the elements of H and for i going from 1 to K do the following.
   If $\overline{q}_H < \overline{q}_{e_i}$, let $\overline{q}_{e_i} = \overline{q}_H$, $\widetilde{s}_{e_i H} = 1$ and $\widetilde{s}_{e_i I} = 0$ for $I \neq H \in E_i^K$.
   If $\overline{q}_H > \overline{q}_{e_i}$, let $\lambda_{e_j} = \lambda_{e_j} + (\overline{q}_H - q_{e_i})/(K - i)$ for $i < j \leq K$.
3. Solve the minimum spanning tree problem (41) and denote by $\widetilde{\mathbf{x}}$ the minimizing vector.

4. Obtain the solution vector $(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{s}}, \overline{\mathbf{t}}) \in \mathbb{B}^{m+m^2+Ko+Kom}$ by letting $\overline{\mathbf{x}} = \widetilde{\mathbf{x}}$, $\overline{s}_{iH} = \widetilde{x}_i \widetilde{s}_{iH}$ and $\overline{\mathbf{t}}_{iH} = \overline{s}_{iH} \widetilde{\mathbf{t}}_H$ for $i \in E$ and $H \in E_i^K$, and $\overline{\mathbf{y}}_i = \overline{\mathbf{t}}_i(E_i^K)$ for $i \in E$.

The second step of Algorithm 5 runs in $O(om \log n)$ time complexity and defines the overall complexity of the algorithm. The resulting Lagrangian relaxation scheme is denoted by Lag_3.

In Section 4, we conduct computational experiments to compare the bounds given by Lag_1, Lag_2, and Lag_3. Before that, in the next section, we describe how two BB algorithms based on Lag_1 and Lag_2 were implemented. As we will demonstrate in Section 4, Lag_3 is computationally expensive for $K \geq 3$ and the evaluation of the implied Lagrangian bounds suffers very badly from convergence problems. For $K = 2$, a preliminary BB algorithm that makes use of Lag_3 was largely outperformed by the one based on Lag_2. As such, we decided not to proceed with the implementation of a BB algorithm based on Lag_3.

3 Branch-and-bound Algorithms

In this section, we describe the main implementation details of two branch-and-bound algorithms, BB_1 and BB_2, respectively based on the Lagrangian relaxation lower bounding procedures Lag_1 and Lag_2. BB_1 and BB_2 are quite similar, differing only when explicitly mentioned in the exposition that follows.

3.1 Initial Upper Bounds

Initial valid QMSTP upper bounds are obtained by means of the following multi-start heuristic. We first randomly select an initial spanning tree T of G and then apply local search. The latter is implemented by evaluating the trees obtained by inserting into T an edge i not in T and removing the edges in the (unique) resulting cycle. If the best removal for edge i results in a tree with better cost than T, that tree immediately becomes the current solution. The process stops when the insertion of every edge not in T is evaluated and no cost improvement is detected. The procedure is repeated 100 times, each one starting with a different initial spanning tree.

3.2 Lower Bounds and Node Selection

Algorithm BB_1 makes use of the bounding procedure Lag_1, while algorithm BB_2 employs Lag_2. In an attempt to accelerate the resolution of the problem at each non-root BB node and preserve the quality of the lower bounds, Lagrangian multipliers at a given node are initialized with the best multipliers found for the parent node. This way, we expect to obtain near optimal multipliers with a smaller number of steps of the subgradient method. As a drawback, a table of size $O(m^2)$ needs to be stored at each node. For this reason, a best bound search strategy becomes prohibitive, since a huge amount of memory is needed to store the node list. Consequently, both algorithms implement depth-first searches.

3.3 Branching and Variable Selection

Assume that $\overline{\mathbf{y}} \in \mathbb{B}^{m^2}$ is part of the optimal solution vector to the Lagrangian subproblem that provided the best lower bound at a given BB node. If $\overline{y}_{ij} = \overline{y}_{ji}$ for all $i < j \in E$, no branching is needed since that node can be pruned by optimality. Otherwise, the edges i and j such that $\overline{y}_{ij} \neq \overline{y}_{ji}$ are candidates for branching. Once the branching edge is determined, two new nodes are created. For one of them, we force the edge to be selected. For the other node, the edge is forbidden.

The way branching constraints are enforced is one of the few differences between BB_1 and BB_2. For BB_1, it suffices to force (resp. to prevent) the appearance of the edge in the trees obtained in steps 1 and 2 of Algorithm 1. For BB_2, if an edge i is imposed (resp. forbidden) it is necessary to grant that in step 2 of Algorithm 3 exactly one (resp. none) of the subgraphs containing i is selected. To guarantee such a condition, we remove from the auxiliary graph $\overline{G}$ any edge that connects i to a vertex of U (resp. $\overline{V}$).

In order to decide the branching variable, we do the following. For each candidate branching variable, we compute the Gilmore-Lawler bounds implied when the variable is imposed and forbidden. If any of these two bounds is larger than the best incumbent value, the variable is then fixed accordingly. If any variable is fixed in this step, we apply another round of subgradient optimization, starting with the best multipliers obtained during the previous round. Otherwise, we select as the branching variable the one for which the minimum between these two bounds is the maximum. Notice that this a strong branching strategy [1] applied to Lagrangian relaxation.

3.4 Redistribution of the Costs of Fixed Variables

Assume that at a given branch-and-bound node, a nonempty set F of edges is forced to be included in the solution for that node. The objective function can be written as $\sum_{i \in E \setminus F} \sum_{j \in E} q_{ij} x_i x_j + \sum_{i \in F} \sum_{j \in E} q_{ij} x_i x_j$. Multiplying the second term by $\sum_{i \in E} x_i/(n-1) = 1$, using $x_i = 1$ for $i \in F$ and using the linearization variables, we obtain the following expression for the objective function:

$$\sum_{i \in E \setminus F} \sum_{j \in E} (q_{ij} + \frac{\sum_{k \in F} q_{kj}}{n-1}) y_{ij} + \sum_{i \in F} \sum_{k \in E} \frac{\sum_{j \in F} q_{jk}}{n-1} y_{ij}.$$

For the optimal Lagrangian multipliers, the bounds are unaffected by the change indicated above. However, since multipliers are not necessarily optimized exactly by the subgradient method, this reformulation might help to obtain better lower bounds in practice. In order to illustrate the reasoning behind such a claim, consider as an example the Lagrangian relaxation scheme Lag_1. During the resolution of the Lagrangian subproblems, we compute the best interaction tree for each edge $i \in E$. With the reformulation above, if $i \notin F$, the cost of selecting $j \in E$ in its interaction tree becomes $q_{ij} + \sum_{k \in F} q_{kj}/(n-1)$. That means that it is necessary to take into account a fraction of the interaction costs of the fixed edges and j, leading to a better estimate of the actual cost of using i in a solution. On the other hand, all $i \in F$ will have the same interaction tree.

3.5 Parallelization

Our computational experiments are conducted with a multi-processor shared memory system. In order to take advantage of this hardware, our BB algorithms are implemented in parallel, following the guidelines proposed in [23] for a parallel BB algorithm implemented for the QAP. We give a brief description of their strategy and show how we improved that implementation by introducing an effective load balancing mechanism.

As in [23], BB nodes are kept in disjoint lists: a global list and a local list for each processor. Each processor explores its local list independently. Whenever a processor finishes solving all nodes in its list, it requires access to the global list. After obtaining access to the global list, the processor explores that list until a node at level d or greater is found. This node is then added to the local list of the processor, which releases the access to the global list and goes back to exploring its own list.

In [23], a processor stops after it certifies that the global list is empty. This might lower parallel efficiency, since after that moment that processor no longer works. In order to overcome that, we proceed in a different way. After detecting that the global list is empty, the processor waits, but periodically checks the global list. On the other hand, a processor that is working also periodically checks if there are processors waiting for duty. In positive case, this processor removes some nodes from the head of its local list and add them to the global list. Those nodes will be available to the waiting processors and parallel efficiency should improve. Under this framework, a processor only stops when its local list is empty and all the other processors are waiting.

4 Computational Experiments

In this section, we present computational experiments that allowed us to address the quality of the lower bounds provided by Lag_1, Lag_2 and Lag_3. We also present computation results for BB_1 and BB_2 and compare them with existing solution approaches in the literature.

4.1 Test Instances

The algorithms were tested with two sets of instances from the literature:

- CP, introduced by Cordone and Passeri [9]: These instances comprise graphs with $n \in \{10, 15, \ldots, 50\}$ and densities $d \in \{33\%, 67\%, 100\%\}$. Depending on how the diagonal (q_{ii}) and the off-diagonal (q_{ij}) entries of Q relate, four types of instances were generated, for each tuple (n, d). These types are denoted CP1, CP2, CP3, and CP4. For CP1, values for q_{ii} and q_{ij} correspond to integers randomly chosen from $\{1, \ldots, 10\}$, with uniform probability. Similarly, for CP2, $q_{ii} \in \{1, \ldots, 10\}$ and $q_{ij} \in \{1, \ldots, 100\}$. For CP3, $q_{ii} \in \{1, \ldots, 100\}$ and $q_{ij} \in \{1, \ldots, 10\}$. Finally, for CP4, $q_{ii}, q_{ij} \in \{1, \ldots, 100\}$.
- OP, introduced by Öncan and Punnen [24]: These instances comprise complete graphs of different sizes $n \in \{6, 7, \ldots, 18\}$ and $n \in \{20, 30, 40, 50\}$. For each n, ten instances of three different types, OP1, OP2, and OP3 were generated. Instances of type OP1 have integer costs q_{ii} and q_{ij} randomly chosen from $\{1, \ldots, 100\}$ and $\{1, \ldots, 20\}$, respectively. For OP2, an integer weight w_v randomly chosen from $\{1, \ldots, 10\}$ is assigned to each vertex $v \in V$. Given two different edges $i = \{a, b\}$ and $j = \{c, d\}$, $q_{ij} = w_a w_b w_c w_d$. For an edge i, q_{ii} is an integer randomly chosen from $\{1, \ldots, 10000\}$. For OP3, each vertex represents a 2-dimensional point with coordinates randomly chosen in the interval $[0, 100]$. The value of q_{ii} is given by the euclidian distance between the extremities of i, while q_{ij}, $i \neq j$, is the distance between the midpoints of i and j.

4.2 Computational Results

Computational experiments were performed with a machine with two Intel Xeon processors, each one with six cores running at 2.4GHz and a total of 32GB of shared RAM memory, running under Linux Operating System. All algorithms were coded in `C++` and compiled with `G++ 4.6.3`, optimization flag `-O3` turned on. OpenMP was used to implement the parallel BB algorithms.

In our subgradient method implementation, 5000 iterations are performed, with an initial step size of 2 in the direction of the normalized subgradient. The step size is halved whenever 500 iterations have past with no improvement in the Lagrangian dual function. For non-root BB nodes, 100 iterations are performed with the same initial step size of 2. The step size is halved at every 10 iterations without improvements.

In Table 1, we report lower bounds for the formulations of Assad and Xu [5] (F_{AX92}), Öncan and Punnen [24] (F_{OP10}), F_1, F_2 for $K = 2$, and F_3 for $K \in \{2, 3, 4\}$. The bounds we indicate in columns under headings F_1 and F_2 are $L_1^{*(6)}$ and $L_2^{*(22),(2)}$, respectively. These bounds were approximated with the Lagrangian relaxation schemes Lag_1 and Lag_2, respectively. Likewise, columns under headings F_3 depict an approximation of the bound $L_3^{*(35)(36)}$ provided by Lag_3, for values of $K \in \{2, 3, 4\}$.

The lower bounds we report for F_{AX92} and F_{OP10} were evaluated by ourselves, by means of Lagrangian relaxation algorithms implemented as described in those references. We found differences between the bounds we evaluated and those reported by Öncan and Punnen [24]. In order to further validate the correctness of the Lagrangian bounds provided by Lag_1, we computed the LP relaxation bounds $Z(F_1)$, by means of a LP based cutting plane algorithm. The computational results indicate that the bounds provided by Lag_1 are close to $Z(F_1)$, but never exceed them. However, the bounds reported for F_{OP10} in [24] quite often exceed $Z(F_1)$. We provide an in depth discussion of this matter in section C of the supplementary document to this paper.

The first four columns of Table 1 give the number of vertices (n), the number of edges (m), the type (type), and the best known upper bound (ub), for each instance. Subsequent columns provide the lower bound (lb) and the computational time (t) (in seconds) taken by each formulation/lower bounding procedure. A time limit of 10 hours was specified. The best overall lower bounds are indicated in boldface.

In Table 2, we compare algorithms BB_1, BB_2, and BB_{CP}, the BB algorithm in [10]. Since no implementation details were available, we could not implement and test BB_{CP}. Therefore, computational results we report for BB_{CP} are precisely those provided in [8]. For BB_1 and BB_2, a time limit of 100 hours was imposed. The stopping criteria for BB_{CP}, however, was not the same for all instances. For some of them, the algorithm was stopped after a time limit of 3600 seconds was reached. For others, after 10^6 nodes were investigated. Differently from Table 1, where each row refers to a particular instance, Table 2 presents aggregated results for ranges of n and m. Detailed computational results for each instance in our test bed are provided in section D of the supplementary document.

The first four columns of Table 2 present the range of n and m, the type and total number of instances (total) in that range. Next, for each algorithm, we present the total number of instances solved to optimality (solv.), the maximum number of nodes investigated (max nodes) and the maximum time (max t) in seconds needed by the algorithm to solve a single instance in the range (considering only those instances solved to optimality). An entry "-" indicates that all instances in the range were left unsolved by the algorithm under consideration.

From both tables, it is clear that the bounds implied by F_1 are much stronger than the previous bounds in the literature (16,6% stronger than the bounds of [5] and 26,7% stronger than the bounds of [24], for OP1 instances). Compared to the other schemes, the Lagrangian relaxation algorithm Lag_1 seems to offer a good trade-off between lower bound quality and computational effort. That claim is validated by how BB_1 and BB_2 do compare to each other.

Formulation F_2 (Lag_2) provides lower bounds that are significantly stronger than those provided by F_1 (Lag_1), F_{AX92}, and F_{OP10} (65% stronger than F_1 (Lag_1), 90,6% stronger than F_{AX92}, and 81,2% stronger than F_{OP10}, for CP instances). Consequently, the number of nodes investigated by BB_2 is orders of magnitude smaller than BB_1 and BB_{CP} counterparts. However, BB_2 is dominated by BB_1 in terms of computational time due to the high costs demanded to run Lag_2.

Lower bounds offered by F_3 (Lag_3) with $K = 2$ are quite close to those offered by F_2 (Lag_2), but demand less computational effort. As expected, these bounds get stronger as K grows. Bounds provided by F_3 with $K = 4$ are the overall best but demand a high computational effort. However, since the computational effort involved for their evaluation is huge, Lag_3 actually provided a poor approximation for the true bound $L_3^{*(35)(36)}$. That behavior can be observed, for example, for OP2 and OP3 instances with $n = 13$ vertices.

Compared to BB_1, BB_{CP} explores many more nodes. A fair comparison between BB_1 and BB_{CP} is not trivial to state, since they were tested in different computational environments and make use of different stopping criteria. In addition, the enumeration tree of BB_1 (and BB_2) was explored in parallel (parallel efficiencies around 80% were achieved) whereas no indication on whether BB_{CP} was implemented in parallel or not is available.

With BB_1, for the first time in the QMSTP literature, the following sets of instances were solved to proven optimality: all instances of Cordone and Passeri with 20 vertices and 127 edges, OP2 and OP3 instances of Öncan e Punnen with $n \in \{30, 50\}$ and all OP1 instances with $n = \{16, 17, 18\}$. Instances of type CP2 and CP3 are not very difficult; most of them were solved at the root node by BB_1 and BB_2.

5 Conclusion

In this paper, we investigated formulations and exact solution approaches for the quadratic minimum spanning tree problem. Initially, we introduced a linear 0-1 programming formulation based on the reformulation-linearization technique and derived a Lagrangian relaxation algorithm based on it. We have shown that the formulation over which the Lagrangian subproblem is defined has the integrality property and we presented an efficient algorithm for solving it. This lower bounding scheme was embedded in a branch-and-bound algorithm.

We also introduced a novel linear 0-1 programming formulation, based on the idea of decomposing spanning trees into subgraphs with a fixed number of edges. That formulation was used to derive two Lagrangian relaxation bounding procedures. A second branch-and-bound algorithm based on one of them was implemented. Although the Lagrangian bounds behind the second method are stronger than those provided by the reformulation-linearization technique, the second algorithm was dominated by the first, in terms of overall running time. That happens because the evaluation of its lower bounds demand excessive CPU running time.

The first branch-and-bound algorithm benefits a lot from the good trade-off between lower bound quality and the computational effort involved in its evaluation. As a result, a parallel implementation of that branch-and-bound algorithm managed to solve several instances in the literature for the first time, including some with $n = 50$ vertices.

References

[1] T. Achterberg, T. Koch, and A. Martin. Branching rules revisited. *Operations Research Letters*, 33(1): 42–54, 2005.

Instance				F_{AX}		F_{OP}		F_1		F_2 $(K=2)$		F_3 $(K=2)$		F_3 $(K=3)$		F_3 $(K=4)$	
n	m	type	ub	lb	t	lb	t	lb	t	lb	t	lb	t	lb	t	lb	t
25	100	CP1	2185	1115.5	0	1193.2	3	1285.1	2	1718.7	40	1715.8	20	1764	1592	**1877.3**	18998
25	100	CP2	19976	8170.6	0	8988.8	3	10061.1	2	14860.8	52	14829.5	20	15371.2	1595	**16617.2**	19846
25	100	CP3	2976	2069.1	0	1961.3	3	2289.2	2	2652.5	41	2645.9	20	2660.9	1592	**2749.9**	19174
25	100	CP4	21176	9296.8	0	10089.2	3	11190	2	15977	55	15947.7	20	16470.8	1601	**17736.8**	20125
25	200	CP1	2023	755.1	0	801.3	12	828	3	1316.7	305	1315	135	1431.6	21610	**1522.7**	36000
25	200	CP2	18251	4154.4	0	4564.5	12	5028.6	3	10497.4	362	10480	136	11780.2	21618	**12795.7**	36000
25	200	CP3	2546	1468.1	0	1385.3	12	1626.8	3	2071	261	2068.2	136	2135.2	21472	**2185**	36000
25	200	CP4	19207	5183.6	0	5560.5	12	6065.4	3	11522	360	11504.6	134	12798.5	21666	**13812.8**	36000
25	300	CP1	1943	668.5	0	705.2	23	715.4	6	1143.2	1012	1141.3	392	1268.7	36000	**1348.7**	36000
25	300	CP2	17411	2879.4	0	3161.8	24	3443.2	6	8533.5	1118	8518.3	394	9920.1	36000	**10868.2**	36000
25	300	CP3	2471	1279.2	0	1213.9	23	1405.6	6	1875.3	918	1871.8	400	**1957.4**	36000	1874.4	36000
25	300	CP4	18370	3865.2	0	4086.6	24	4451.3	6	9563.4	1135	9542.9	397	10947.8	36000	**11854.5**	36000
13	78	OP1	1022	513.7	0	475.9	2	606.3	0	842.6	21	838.9	14	847.5	655	**901**	9947
13	78	OP1	1089	592.9	0	532.1	2	702	0	900.1	22	891.5	14	905.5	648	**941.9**	9892
13	78	OP1	1163	609.8	0	576.3	2	697	0	945.2	22	941.5	14	952.1	654	**1003.8**	9763
13	78	OP1	1129	703.5	0	659	1	803.1	0	1033.3	21	1028.9	14	1041	640	**1086**	9763
13	78	OP1	1023	663.2	0	588.4	2	748.8	0	1001.4	21	997.3	14	1005.8	664	**1054.7**	9838
13	78	OP1	982	586.9	0	546.2	1	715.4	0	933	21	928.6	14	931	652	**970.8**	9714
13	78	OP1	1048	520.8	0	466	2	613.3	0	838.3	22	833.7	14	832	656	**880.4**	9866
13	78	OP1	1045	611.8	0	571.1	2	712.8	0	929.4	22	926.6	14	935.8	645	**976.2**	9724
13	78	OP1	1065	637.6	0	594.3	2	741.2	0	980.3	22	974	14	978.1	653	**1017.2**	9755
13	78	OP1	1160	618.2	0	572.1	2	720.3	0	978.6	21	976.6	14	991.4	648	**1050.7**	9575
13	78	OP2	45586	44885	0	44693	1	**45586**	0	**45586**	12	42642.2	4	38844.2	195	37228.7	2940
13	78	OP2	49313	48747.1	0	45717	1	**49313**	0	**49313**	11	45373.6	4	37705.3	183	35443.9	2825
13	78	OP2	44513	44257.5	0	43676.5	1	**44513**	0	**44513**	11	36545.3	4	34665.2	181	34181.8	2866
13	78	OP2	37250	37250	0	37054	1	**37250**	0	**37250**	11	31793.1	4	30504.9	183	25435.2	2811
13	78	OP2	50990	49908	0	46969	1	**50990**	0	**50990**	11	48493.8	4	45486.7	198	43695.1	2846
13	78	OP2	43261	42380	0	41140	1	**43261**	0	**43261**	12	33263	4	24401.7	181	25020	2797
13	78	OP2	36085	35809.1	0	35135	1	**36085**	0	**36085**	11	34055.6	4	32091.6	190	30169.3	2804
13	78	OP2	34474	34442.6	0	33775	1	**34474**	0	**34474**	10	30467.7	4	26480.4	180	24826.8	2829
13	78	OP2	28566	28360.2	0	27653	1	**28566**	0	**28566**	10	24213.1	4	22879.2	178	21686	2842
13	78	OP2	34847	34493	0	33909	1	**34847**	0	**34847**	13	32670.2	4	26926.7	187	27950.4	2902
13	78	OP3	1731	1595.6	0	1648.3	1	**1731**	0	**1731**	9	1720.2	8	1730.4	601	1730.5	9058
13	78	OP3	2484	2341.4	0	2318.7	1	**2484**	0	**2484**	10	2332.5	4	2210.2	218	2270.2	3524
13	78	OP3	2440	2228.8	0	2297.2	1	2436.6	0	**2440**	12	2407.4	9	2430.1	595	2426.5	5881
13	78	OP3	2489	2307.5	0	2272.5	1	2453.2	0	**2483**	23	2420.1	6	2453.9	591	2187	3398
13	78	OP3	2044	1932.8	0	1915	1	**2044**	0	**2044**	11	1940.9	4	2029.6	609	1910.1	3421
13	78	OP3	1806	1655.7	0	1634	1	1805	0	**1806**	11	1754.8	5	1692.4	268	1796.6	8852
13	78	OP3	2185	2041.9	0	2035	1	**2185**	0	**2185**	10	2162.6	5	2184.1	615	2167	4859
13	78	OP3	2275	2081	0	2134.2	1	2272.8	0	**2275**	11	2269.9	11	2265.8	597	2270.2	8853
13	78	OP3	1968	1741.5	0	1857.6	1	1943.1	0	**1957.7**	21	1931.5	13	1942.4	616	1948	7916
13	78	OP3	2331	2241.4	0	2252	1	**2331**	0	**2331**	10	2283.1	5	2097.8	226	2330.4	7964

Table 1: Lower bound comparisons.

Instance				BB_{CP}			BB_1			BB_2		
n	m	type	total	solv.	max nodes	max t	solv.	max nodes	max t	solv.	max nodes	max t
10-20	15-105	CP	28	28	51880837	887	28	144309	946	28	24106	3170
20	127	CP	4	0	-	-	4	24431331	271761	0	-	-
10-15	45-105	OP1	60	60	7922195	184	60	19239	174	60	4057	775
16-17	120-136	OP1	20	0	-	-	20	449565	6386	20	48463	20080
18	153	OP1	10	0	-	-	10	5351735	93178	0	-	-
10-20	45-190	OP2	100	100	583379	29	100	7	15	100	3	133
30	435	OP2	10	0	-	-	10	1	151	10	1	1900
50	1225	OP2	10	0	-	-	10	1	1731	0	-	-
10-20	45-190	OP3	100	98	979125	36	100	21	12	100	13	218
30	435	OP3	10	0	-	-	10	129	491	10	81	17857
50	1225	OP3	10	0	-	-	10	735	19045	0	-	-

Table 2: Comparison of branch-and-bound algorithms.

[2] W. P. Adams and H. D. Sherali. A tight linearization and an algorithm for zero-one quadratic programming problems. *Management Science*, 32(10):1274 – 1290, 1986.

[3] W. P. Adams and H. D. Sherali. A tight linearization and an algorithm for zero-one quadratic programming problems. *Management Science*, 32(10):1274–1290, 1986.

[4] L. D. Andersen and H. Fleischner. The np-completeness of finding a-trails in eulerian graphs and of finding spanning trees in hypergraphs. *Discrete Applied Mathematics*, 59(3):203–214, 1995.

[5] A. Assad and W. Xu. The quadratic minimum spanning tree problem. *Naval Research Logistics (NRL)*, 39(3):399–417, 1992.

[6] R. E. Burkard, E. Çela, P. M. Pardalos, and L. S. Pitsoulis. The quadratic assignment problem. In D.-Z. Du and P. M. Pardalos, editors, *Handbook of Combinatorial Optimization*, volume 3, pages 241 – 337. Kluwer Academic Publishers, 1998.

[7] A. Caprara. Constrained 0–1 quadratic programming: Basic approaches and extensions. *European Journal of Operational Research*, 187(3):1494–1503, 2008.

[8] R. Cordone. The quadratic minimum spanning tree problem (qmstp), March 2013. URL `http://homes.di.unimi.it/~cordone/research/qmst.html`.

[9] R. Cordone and G. Passeri. Heuristic and exact approaches to the quadratic minimum spanning tree problem. In *Seventh Cologne-Twente Workshop on Graphs and Combinatorial Optimization*, 2008.

[10] R. Cordone and G. Passeri. Heuristics and exact algorithm for the quadratic minimum spanning tree problem. *Computers and Operations Research*, 2012. Submitted.

[11] J. Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1(1):127–136, 1971.

[12] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM*, 19(2):248–264, 1972.

[13] Jack Edmonds. Maximum matching and a polyhedron with $0, 1$ vertices. *J. of Res. the Nat. Bureau of Standards*, 69 B:125–130, 1965.

[14] A. M. Geoffrion. Lagrangian relaxation for integer programming. *Mathematical Programming Study*, 2: 82–114, 1974.

[15] A. M. H. Gerards. Matching. In M.O. Ball, T.L. Magnanti, C.L. Monma, and G.L. Nemhauser, editors, *Network Models*, volume 7 of *Handbooks in Operations Research and Management Science*, chapter 3, pages 135–224. Elsevier, 1995.

[16] P. C. Gilmore. Optimal and suboptimal algorithms for the quadratic assignment problem. *Journal of the Society for Industrial and Applied Mathematics*, 10(2):305–313, 1962.

[17] E. Hazan, S. Safra, and O. Schwartz. On the complexity of approximating k-set packing. *Computational Complexity*, 15(1):20–39, 2006.

[18] M.H. Held, P. Wolfe, and H.D Crowder. Validation of subgradient optimization. *Mathematical Programming*, 6:62–88, 1974.

[19] J.B. Kruskal. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proceedings of the American Mathematical Society*, 7:48–50, 1956.

[20] E. L. Lawler. The quadratic assignment problem. *Management Science*, 9(4):586–599, 1963.

[21] J. Lee and J. Leung. On the boolean quadric forest polytope. *INFOR*, 42(2):125–141, 2004.

[22] A. Lucena. Non delayed relax-and-cut algorithms. *Annals of Operations Research*, 140(1):375–410, 2005.

[23] B. Mans, T. Mautor, and C. Roucairol. A parallel depth first search branch and bound algorithm for the quadratic assignment problem. *European Journal of Operational Research*, 81(3):617 – 628, 1995.

[24] T. Öncan and A. P. Punnen. The quadratic minimum spanning tree problem: A lower bounding procedure and an efficient search algorithm. *Computers and Operations Research*, 37(10):1762–1773, 2010.

[25] M. Padberg. The boolean quadric polytope: some characteristics, facets and relatives. *Mathematical Programming*, 45(1–3):139–172, 1989.

[26] M. W. Padberg and L. A. Wolsey. Trees and cuts. In *Combinatorial Mathematics Proceedings of the International Colloquium on Graph Theory and Combinatorics*, volume 75, pages 511–517. 1983.

[27] G. Palubeckis, D. Rubliauskas, and A. Targamadzė. Metaheuristic approaches for the quadratic minimum spanning tree problem. *Information Technology and Control*, 39:257–268, 2010.

[28] David Pisinger. The quadratic knapsack problem—a survey. *Discrete Applied Mathematics*, 155(5):623 – 648, 2007.

[29] R.C. Prim. Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36: 1389–1401, 1957.

[30] H. D. Sherali and W. P. Adams. A hierarchy of relaxations and convex hull characterizations for mixed-integer zero—one programming problems. *Discrete Applied Mathematics*, 52(1):83 – 106, 1994.

[31] S. Sundar and A. Singh. A swarm intelligence approach to the quadratic minimum spanning tree problem. *Information Sciences*, 180(17):3182–3191, 2010.

[32] G. Zhout and M. Gen. An effective genetic algorithm approach to the quadratic minimum spanning tree problem. *Computers & Operations Research*, 25(3):229–237, 1998.

Supplementary Document for "Lower Bounds and Exact Algorithms for the Quadratic Minimum Spanning Tree Problem" (Submitted to Computers & Operations Research on May 2013)

Dilson Lucas Pereira Michel Gendreau Alexandre Salles da Cunha

December 16, 2013

A Separation

In this section we provide polynomial time algorithms for the separation of inequalities (8) and (11). First, we address the separation of (8). Without loss of generality, we assume that constraints (8) are formulated for all $S \subset V$, $|S| \geq 1$. As such, their separation problem can be stated as: *Given $i \in E$, $\overline{x}_i$, and $\overline{\mathbf{y}}_i \in \mathbb{R}^m$, find a set $\overline{S} \subset V$, $|\overline{S}| \geq 1$, for which (8) is violated or certify that no such set does exist.*

Padberg and Wolsey [26] give a polynomial time algorithm that solves the separation problem of (2), which we now adapt for the separation problem of (8). The algorithm finds $\overline{S}$ such that

$$\overline{S} \in \arg \min_{S \subset V, |S| \geq 1} \{|S|\overline{x}_i - \overline{\mathbf{y}}_i(E(S))\}. \tag{42}$$

Clearly, there is a violated inequality (8) if and only if the minimum is smaller than $\overline{x}_i$.

Algorithm 6

1. Create a directed graph $\widehat{G} = (\widehat{V}, \widehat{A})$ with $\widehat{V} = V \cup \{s, t\}$ and $\widehat{A} = \{(u, v), (v, u) : \{u, v\} \in E\} \cup \{(s, u), (u, t) : u \in V\}$.
2. For each $j = \{u, v\} \in E$ assign capacities $c_{uv} = c_{vu} = \frac{1}{2}\overline{y}_{ij}$ to the arcs (u, v) and (v, u) in $\widehat{A}$.
3. For each $u \in V$ assign capacities $c_{su} = \max\{\frac{1}{2}\overline{\mathbf{y}}_i(\delta(u)) - \overline{x}_i, 0\}$ and $c_{ut} = \max\{\overline{x}_i - \frac{1}{2}\overline{\mathbf{y}}_i(\delta(u)), 0\}$ to the arcs (s, u) and (u, t) in $\widehat{A}$.
4. Find the cut $(\overline{S} \cup \{s\}, V \cup \{t\} \setminus \{\overline{S}\})$ of minimum capacity of $\widehat{G}$. $\overline{S}$ minimizes (42).

Assuming that G is connected, Steps 1-3 are easily seen to be performed in $O(E)$. Step 4 takes $O(n)$ maximum flow computations, which can be performed in $O(nm^2)$ with the algorithm of Edmonds and Karp [12], for example. Thus, the complexity of Algorithm 6 is $O(n^2m^2)$. Observe, however, that when employed to solve the linear relaxation of F_1 with dynamic generation of cuts, Algorithm 6 has to be applied for each $i \in E$, which gives the total complexity of $O(n^2m^3)$.

To prove the validity of the procedure, observe that the capacity of any cut $(\widetilde{S} \cup \{s\}, V \cup \{t\} \setminus \{\widetilde{S}\})$ is

$$\sum_{u \in \widetilde{S}} \max\left\{\overline{x}_i - \frac{1}{2}\overline{\mathbf{y}}_i(\delta(u)), 0\right\} + \sum_{u \in V \setminus \widetilde{S}} \max\left\{\frac{1}{2}\overline{\mathbf{y}}_i(\delta(u)) - \overline{x}_i, 0\right\} + \frac{1}{2} \sum_{\substack{i = \{u,v\} \in E, \\ u \in \widetilde{S}, v \notin \widetilde{S}}} \overline{y}_{ij}$$

$$= \sum_{u \in \widetilde{S}} \left(\max\{\overline{x}_i - \frac{1}{2}\overline{\mathbf{y}}_i(\delta(u)), 0\} - \max\{\frac{1}{2}\overline{\mathbf{y}}_i(\delta(u)) - \overline{x}_i, 0\} \right)$$

$$+ \sum_{u \in V} \max\{\frac{1}{2}\overline{\mathbf{y}}_i(\delta(u)) - \overline{x}_i, 0\} + \frac{1}{2} \sum_{\substack{i=\{u,v\} \in E, \\ u \in \widetilde{S}, v \notin \widetilde{S}}} \overline{y}_{ij}$$

$$= |\widetilde{S}|\overline{x}_i - \overline{\mathbf{y}}(E(\widetilde{S})) + \sum_{u \in V} \max\{\frac{1}{2}\overline{\mathbf{y}}_i(\delta(u)) - \overline{x}_i, 0\}.$$

Since $\sum_{u \in V} \max\{\frac{1}{2}\overline{\mathbf{y}}_i(\delta(u)) - \overline{x}_i, 0\}$ is constant, the set that yields the cut of minimum capacity of $\widehat{G}$ is the set $\overline{S}$ that minimizes (42).

We now deal with the separation of constraints (11). As before, assume that these constraints are formulated for all $S \subset V$, $|S| \geq 1$. Their associated separation problem can be stated as: *Given* $(\overline{\mathbf{x}}, \overline{\mathbf{y}}) \in \mathbb{R}^{m+m^2}$*, find an edge* $i \in E$ *and a set* $\overline{S} \subset V$*,* $|\overline{S}| \geq 1$*, for which* (11) *is violated or certify that no such set does exist.*

Again, we adapt the algorithm of Padberg and Wolsey [26]. We consider one edge $i \in E$ at a time. It is assumed that $(\overline{\mathbf{x}}, \overline{\mathbf{y}})$ satisfy $y_{ij} \leq x_j$ for all $j \in E$, which is the case when using dynamic cut generation to strengthen the LP relaxation of formulation F_1, for example.

The algorithm below will find $\overline{S}$ such that

$$\overline{S} \in \arg \min_{S \subset V, |S| \geq 1} \{|S|(1 - \overline{x}_i) - (\overline{\mathbf{x}} - \overline{\mathbf{y}}_i)(E(S))\}. \tag{43}$$

It is clear that there is a violated inequality (11) if and only if the minimum is smaller than $1 - \overline{x}_i$.

Algorithm 7

1. Create a directed graph $\widehat{G} = (\widehat{V}, \widehat{A})$ with $\widehat{V} = V \cup \{s,t\}$ and $\widehat{A} = \{(u,v),(v,u) : \{u,v\} \in E\} \cup \{(s,u),(u,t) : u \in V\}$.

2. For each $j = \{u,v\} \in E$ assign capacities $c_{u,v} = c_{v,u} = \frac{1}{2}(\overline{x}_j - \overline{y}_{ij})$ to the arcs (u,v) and (v,u) in $\widehat{A}$.

3. For each $u \in V$ assign capacities $c_{s,u} = \max\{\frac{1}{2}(\overline{\mathbf{x}} - \overline{\mathbf{y}}_i)(\delta(u)) - \overline{x}_i, 0\}$ and $c_{u,t} = \max\{\overline{x}_i - \frac{1}{2}(\overline{\mathbf{x}} - \overline{\mathbf{y}}_i), 0\}$ to the arcs (s,u) and (u,t) in $\widehat{A}$.

4. Find the cut $(\overline{S} \cup \{s\}, V \cup \{t\} \setminus \{\overline{S}\})$ of minimum capacity of $\widehat{G}$. $\overline{S}$ minimizes (43).

Algorithm 7 has the same complexity of Algorithm 6. Considering again the application of the algorithm for each $i \in E$ we obtain a total complexity of $O(n^2m^3)$. Its validity proof is similar to that for Algorithm 6 and is omitted.

B Proofs

In order to prove Proposition 5 we need the following lemma.

Lemma 1. *Problems* $F_2^{(22)}$ *and* (28) *are equivalent.*

Proof. We assume $K \geq 2$. It is clear that the polynomial time resolution of (28) implies the polynomial time resolution of $F_2^{(22)}$, as discussed earlier.

To show the converse, given an instance for (28), defined in terms of a graph $\overline{G} = (\overline{V}, \overline{E})$, $\overline{F} \subseteq \overline{E}^K$ and costs $\overline{Q} = (\overline{q}_H)_{H \in F}$, we will construct a instance for $F_2^{(22)}$, defined in terms of a graph $G = (V, E)$, $F \subseteq E^K$ and matrix $Q' = (q'_{ij})_{i,j \in E}$. The construction will be such that each feasible solution for $F_2^{(22)}$ defined over G, F, and Q' will give a feasible solution with the same cost for (28) defined over $\overline{G}$, $\overline{F}$, and $\overline{Q}$.

Initially, consider $V = \overline{V}$, $E = \overline{E}$, and $F = \overline{F}$. Select a vertex $v \in \overline{V}$ and for all $H \in \overline{F}$, add K new vertices $u_H^1, \ldots, u_H^K$ to V and add to E $K-1$ new edges: $\{v, u_H^1\}$, and $\{u_H^i, u_H^{i+1}\}$, $1 \leq i < K$. Consider an ordering $(H_1, ..., H_{|\overline{F}|})$ for the elements of $\overline{F}$. For all H_k, $1 \leq k \leq |\overline{F}|$, add to E edges $\phi_{H_k} = \{u^{K-1}_{H_k}, u^K_{H_1}\}$, edges $\{u^{K-1}_{H_k}, u^K_{H_k}\}$, for $k > 1$, and $\{u^{K-1}_{H_k}, u^K_{H_{k+1}}\}$, for $k < |\overline{F}|$. Add to F the sets $\{\{v, u^1_{H_k}\}, \ldots, \{u^i_{H_k}, u^{i+1}_{H_k}\}, \ldots, \{u^{K-1}_{H_k}, u^K_{H_1}\}\}$, the sets $\{\{v, u^1_{H_k}\}, \ldots, \{u^i_{H_k}, u^{i+1}_{H_k}\}, \ldots, \{u^{K-1}_{H_k}, u^K_{H_k}\}\}$, for $k > 1$, and $\{\{v, u^1_{H_k}\}, \ldots, \{u^i_{H_k}, u^{i+1}_{H_k}\}, \ldots, \{u^{K-1}_{H_k}, u^K_{H_{k+1}}\}\}$, for $k < |\overline{F}|$. After that, $O(K|\overline{F}|)$ vertices and edges will have been added to V and E, respectively, and $O(|\overline{F}|)$ new subsets will have been added to F. Therefore, the transformation is polynomial in the size of the input data.

Now, define matrix $Q' = (q'_{ij})_{i,j \in E}$ as follows. For all $H \in \overline{F}$, fix an edge $i \in H$, define $q'_{i\phi_H} = \overline{q}_H + (K-1)M$, where M is a sufficiently large number, $q'_{j\phi_H} = -M$, $j \neq i \in H$, and $q'_{j\phi_I} = KM$ if $j \notin I$, $j \in H$, $I \in \overline{F}$. All other entries of Q' are set to zero. This step takes $O((|\overline{E}| + K|\overline{F}|)^2)$, which is polynomial in the size of the input problem.

Assuming that the problem defined by (28) over $\overline{G}$, $\overline{F}$, and $\overline{Q}$ admits at least one feasible solution, then, given the way F was constructed, $F_2^{(22)}$ defined over G, F, and Q' also admits a feasible solution. Notice that, for any feasible solution for $F_2^{(22)}$, if $S \subseteq F$ is the set of elements of F used in this solution, then a unique solution $S \cap \overline{F}$ is readily available for (28).

In case $H \in \overline{F}$ is used in an optimal solution for $F_2^{(22)}$, the cost implied by its selection will be as in (27). Let us now investigate the optimal spanning tree for that problem. All the vertices, with exception of $u^K_{H_1}$, can be reached without any cost. To reach $u^K_{H_1}$, if ϕ_I, $I \in \overline{F}$, is used and if $|H \cap I| = 0$, a cost K^2M will be implied. If $|H \cap I| = a$, $0 < a < K$, the smallest cost possible is $(K-a)KM - aM > M$. If $|H \cap I| = K$, a cost $\overline{q}_H$ will be implied, which is the minimum possible. Thus, all interaction trees of minimum cost for $H \in \overline{F}$ will have cost $\overline{q}_H$ (the minimum interaction trees for $H \in F \setminus \overline{F}$ do not imply any costs). Consequently, a solution with the same objective value is readily available for (28) and the proof is complete. □

Proposition 5. *Problem $F_2^{(22)}$ is NP-Hard for $K \geq 3$*

Proof. A k-uniform hypergraph is a hypergraph whose edges have the same cardinality k. Deciding whether a k-uniform hypergraph has a spanning tree is NP-Hard for $k \geq 4$, which implies that if weights are attached to the edges, finding the spanning tree of minimum cost is also NP-Hard [4]. Although it is possible to decide whether a 3-uniform hypergraph has spanning tree in polynomial time, we are not aware if it is possible to find the minimum spanning tree of a 3-hypergraph in polynomial time. The case $k = 2$ corresponds to a minimum spanning tree problem on a ordinary graph.

The idea of the proof is to present a polynomial reduction from the problem of finding a minimum spanning tree of a k-uniform hypergraph to problem (28) and apply Lemma 1.

Consider then the problem of finding a minimum spanning tree of a $(K+1)$-uniform hypergraph $\widetilde{G} = (\widetilde{V}, \widetilde{E})$ $(|\widetilde{E}| = \widetilde{m})$ with costs $\widetilde{Q} = (\widetilde{q}_i)_{i \in \widetilde{E}}$ assigned to its hyperedges.

Now, for all $i \in \widetilde{E}$, define $\phi(i)$ as the set of edges of any ordinary connected and acyclic graph over the elements of i. Consider the graph $G = (V, E)$, with $V = \widetilde{V}$ and $E = \{i : \exists j \in \widetilde{E} : i \in \phi(j)\}$. Define $F = \{\phi(i) : i \in \widetilde{E}\}$, note that $F \subseteq E^K$. Consider also costs $\overline{Q} = (\overline{q}_H)_{H \in F}$, where $\overline{q}_H = \widetilde{q}_j$ if $H = \phi(j)$, $j \in \widetilde{E}$. Observe that G, F, and $\overline{Q}$ can be constructed in time $O(K\widetilde{m})$.

Observe that the elements in $T \subseteq F$ induce a spanning tree of G if and only if $\widetilde{T} = \{i \in \widetilde{E} : \phi(i) \in T\}$ is a spanning tree of $\widetilde{G}$, both with the same cost. Therefore, to solve the input problem we can resort to (28) over G, F, and $\overline{Q}$. The result then follows from Lemma 1. □

The next Lemma is used to prove Proposition 6.

Lemma 2. *Problems $F_2^{(22),(2)}$ and* (29) *are equivalent.*

Proof. Apply the same construction used for the proof of Lemma 1. □

Proposition 6. *Problem $F_2^{(22),(2)}$ is NP-Hard for $K \geq 3$.*

Proof. Given a set $\widetilde{E}$ and a set $\widetilde{F} \subseteq \{H : H \subseteq \widetilde{E}, |H| = k\}$, the maximum k-set packing problem is the problem of finding a set $S \subseteq \widetilde{F}$ of pairwise disjoint elements that has maximum size [17]. For $k = 2$, the problem reduces to the polynomially solvable maximum weight matching problem [15]. However, this problem is NP-Hard for $k \geq 3$. Clearly, if non-negative weights are attached to the elements of $H \in \widetilde{F}$, finding the solution of maximum weight is still a NP-Hard problem.

The idea of the proof is to present a polynomial time reduction from the maximum weight k-set packing problem to problem (29) and apply Lemma 2.

Assume we are given an instance consisting of $\widetilde{E}$, $\widetilde{F} \subseteq \{H : H \subseteq \widetilde{E}, |H| = k\}$, and weights $\widetilde{Q} = (\widetilde{q}_H \geq 0)_{H \in \widetilde{F}}$ for the maximum weight k-set packing problem. Assume $k = K \geq 3$ since, otherwise, both problems are trivial. Define $\widetilde{m} = |\widetilde{E}|$ and $\widetilde{o} = |\widetilde{F}|$. Consider $G = (V, E)$, with $V = \{v_i : 1 \leq i \leq \widetilde{m} + 1\}$ and $E = \{e_i = \{v_i, v_{i+1}\} : 1 \leq i \leq \widetilde{m}\}$. Assume, without loss of generality, that $\widetilde{m}$ is a multiple of K. Assume also that there is a bijective mapping ϕ from $\widetilde{E}$ to E. The mapping ϕ implies the mapping $\phi(H) = \{\phi(i) : i \in \widetilde{E}\}$ for $H \in \widetilde{F}$. Define $F = \{\phi(H) : H \in \widetilde{F}\} \subseteq E^K$. Consider also costs $\overline{Q} = (\overline{q}_H)_{H \in F}$, where $\overline{q}_H = -\widetilde{q}_{\widetilde{H}}$ if $H = \phi(\widetilde{H})$, $\widetilde{H} \in \widetilde{F}$.

Now, add to E $\widetilde{m}$ new edges: $\{v_1, v_i\}$, $3 \leq i \leq \widetilde{m}$, and $\{v_2, v_{\widetilde{m}}\}$. Partition these new edges into $\widetilde{m}/K$ sets H, $q_{\overline{H}} = 0$, let J be the set of all such H and $F = F \cup J$.

Observe that any solution $\widetilde{T} \subseteq \widetilde{F}$ of cost $\widetilde{C}$ for the input problem has $|\widetilde{T}| \leq \widetilde{m}/K$. We can obtain a solution of cost $-\widetilde{C}$ for (29) by adding to $T = \{\phi(H) : H \in \widetilde{T}\}$ $\widetilde{m}/K - |T|$ elements of J to obtain exactly $\widetilde{m}/K$ disjoint sets of edges. Conversely, for any $T \subseteq F$ of cost C, consisting of $\widetilde{m}/K$ disjoint elements, $\widetilde{T} = \{H \in \widetilde{F} : \phi(H) \in T\}$ is a feasible solution of cost $-C$ for the input problem.

Observe that the construction of G, F, and $\overline{Q}$ is performed in time $O(K\widetilde{o})$. Thus, an algorithm with polynomial time complexity for the problem formulated by (29) implies an algorithm with polynomial time complexity for the input problem. Finally, to complete the proof, apply Lemma 2. □

Proposition 8. *$P_3^{(35)(36)}$ is an integral polytope.*

Proof. The proof is similar to the proof of Proposition 1. We show that any vector in $P_3^{(35)(36)}$ can be represented as a convex combination of the integer vectors in that polytope.

To simplify the notation, assume for a moment, that $E = \{1, \ldots, m\}$ and $E^K = \{1, \ldots, o\}$. Assume also that $\mathbf{s}_i = (s_{iH})_{H \in E^K}$, where, for $i \in E$ and $H \in E^K$, s_{iH} is defined as before if $i \in H$, and $s_{iH} = 0$ if $i \neq H$. In a similar way, $\mathbf{t}_i = (\mathbf{t}_{iH})_{H \in E^K}$, where, for $i \in E$ and $H \in E^K$, $\mathbf{t}_{iH}$ is defined as before if $i \in H$, and $\mathbf{t}_{iH} = \mathbf{0}$ if $i \neq H$.

Denote by $\mathcal{T} = \{\mathbf{u}^1, \ldots, \mathbf{u}^{|\mathcal{T}|}\}$ the set of all incidence vectors of spanning trees of G. Define $\mathcal{S} = \{\mathbf{v}^1, \ldots, \mathbf{v}^{|\mathcal{S}|}\}$ as the set of vectors in $\mathbb{B}^o$ that have exactly one entry with value one.

Observe that any integer vector $(\overline{\mathbf{x}}, \overline{\mathbf{y}}, \overline{\mathbf{s}}, \overline{\mathbf{t}}) \in \mathbb{B}^{m+m^2+mo+m^2o}$ in $P_3^{(35)(36)}$ has the following form: $\overline{\mathbf{x}} \in \mathcal{T}$, $\overline{\mathbf{s}}_i = \overline{x}_i \mathbf{v}$, $\mathbf{v} \in \mathcal{S}$, $i \in E$, $\overline{\mathbf{t}}_{iH} = \overline{s}_{iH}\mathbf{u}$, $\mathbf{u} \in \mathcal{T}$, $i \in E$, $H \in E^K$, and $\overline{\mathbf{y}}_i = \sum_{H \in E^K} \overline{\mathbf{t}}_{iH}$, $i \in E$.

Consider vector $(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}}, \widetilde{\mathbf{s}}, \widetilde{\mathbf{t}}) \in \mathbb{R}^{m+m^2+mo+m^2o}$ in $P_3^{(35)(36)}$. From (31) we have

$$\widetilde{\mathbf{x}} = \sum_{a=1}^{|\mathcal{T}|} \alpha^a \mathbf{u}^a, \qquad \sum_{a=1}^{|\mathcal{T}|} \alpha^a = 1, \qquad \alpha^a \geq 0, 1 \leq a \leq |\mathcal{T}|,$$

and, from (32),

$$\widetilde{\mathbf{s}}_i = \widetilde{x}_i \sum_{b_i=1}^{o} \beta_i^{b_i} \mathbf{v}^{b_i}, \qquad \sum_{b_i=1}^{o} \beta_i^{b_i} = 1, \qquad \beta_i^{b_i} \geq 0, 1 \leq b_i \leq o,$$

for all $i \in E$. From (34)

$$\widetilde{\mathbf{t}}_{iH} = \widetilde{s}_{iH} \sum_{c_{iH}=1}^{|\mathcal{T}|} \gamma_{iH}^{c_{iH}} \mathbf{u}^{c_{iH}}, \qquad \sum_{c_{iH}=1}^{|\mathcal{T}|} \gamma_{iH}^{c_{iH}} = 1, \qquad \gamma_{iH}^{c_{iH}} \geq 0, 1 \leq c_{iH} \leq |\mathcal{T}|,$$

for all $i \in E$ and $H \in E^K$. It is not difficult to see that this solution is obtained by the linear combination

$$(\widetilde{\mathbf{x}}, \widetilde{\mathbf{y}}_1, \ldots, \widetilde{\mathbf{y}}_m, \widetilde{\mathbf{s}}_1, \ldots, \widetilde{\mathbf{s}}_m, \widetilde{\mathbf{t}}_{11}, \ldots, \widetilde{\mathbf{t}}_{1o}, \ldots, \widetilde{\mathbf{t}}_{m1}, \ldots, \widetilde{\mathbf{t}}_{mo}) =$$

$$\sum_{a=1}^{|\mathcal{T}|} \alpha^a \sum_{b_1=1}^{o} \beta_1^{b_1} \cdots \sum_{b_m=1}^{o} \beta_m^{b_m} \cdots \sum_{c_{11}=1}^{|\mathcal{T}|} \gamma_{11}^{c_{11}} \cdots \sum_{c_{1o}=1}^{|\mathcal{T}|} \gamma_{1o}^{c_{1o}} \cdots \sum_{c_{m1}=1}^{|\mathcal{T}|} \gamma_{m1}^{c_{m1}} \cdots$$

$$\sum_{c_{mo}=1}^{|\mathcal{T}|} \gamma_{mo}^{c_{mo}}(\mathbf{u}^a, u_1^a \sum_{H\in E^K} v_H^{b_1}\mathbf{u}^{c_{1H}}, \ldots, u_m^a \sum_{H\in E^K} v_H^{b_m}\mathbf{u}^{c_{mH}}, u_1^a\mathbf{v}^{b_1}, \ldots,$$

$$u_m^a\mathbf{v}^{b_m}, u_1^a v_1^{b_1}\mathbf{u}^{c_{11}}, \ldots, u_1^a v_o^{b_1}\mathbf{u}^{c_{1o}}, \ldots, u_m^a v_1^{b_m}\mathbf{u}^{c_{m1}}, \ldots, u_m^a v_o^{b_m}\mathbf{u}^{c_{mo}}).$$

Observe that for each possible instance of indices, the vector obtained by the concatenation on the right-hand-side is an integer vector in $P_3^{(35)(36)}$. Since each individual sum adds up to one, we have

$$\sum_{a=1}^{|\mathcal{T}|}\sum_{b_1=1}^{o}\cdots\sum_{b_m=1}^{o}\cdots\sum_{c_{11}=1}^{|\mathcal{T}|}\cdots\sum_{c_{mo}=1}^{|\mathcal{T}|} \alpha^a\beta_1^{b_1}\ldots\beta_m^{b_m}\gamma_{11}^{c_{11}}\ldots\gamma_{mo}^{c_{mo}} = 1,$$

which shows that the combination is convex and completes the proof. □

C Considerations about the Bounds Reported by Öncan and Punnen [24]

In this section, we make two observations about the results in [24]. The Lagrangian relaxation in that reference is based on formulation F_{OP10}, given by F_1 with constraints (6) replaced by (16) and the addition of the valid inequalities (17). It is proposed a lower bounding scheme based on Lagrangian relaxation, where constraints (17) are relaxed and dualized in a Lagrangian fashion.

The authors claim, in Proposition 2 of that study, that the resulting Lagrangian subproblem can be solved by the Gilmore-Lawler algorithm. However, their Lagrangian subproblem is still the QMSTP, albeit with a modified objective function. Clearly, the QMSTP is not solved by the Gilmore-Lawler algorithm alone. Thus, that is claim is not true. Indeed, constraints (16) are not satisfied by the solution given by the Gilmore-Lawler algorithm. As a matter of fact, these constraints had been previously dualized in the work of Assad and Xu [5]. Nevertheless, the procedure proposed in [24] still provides a lower bound for QMSTP, in which constraints (16) are in relaxed and dualized with zero valued multipliers (with no multiplier adjustment).

As we pointed out in Section 4, computational results reported in that work are not in accordance with Proposition 2 of our study. The bounds reported by the Lagrangian relaxation scheme in [24] are stronger than $Z(F_1)$. To further validate the Lagrangian relaxation bounds we present here, we evaluated $Z(F_1)$ by LP means. That is accomplished by a LP cutting plane algorithm that separates (2) and (8), on the fly. To solve the separation algorithms, we used the algorithms described in Section A.

Table 1 presents average lower bounds, as evaluated by ourselves and as reported in [24]. The first three columns present n, m and the instance type. The next three columns present the lower bounds reported in [24] (Lag_{OP}), the lower bounds computed by our implementation of the procedure described in that work (Lag'_{OP}), the lower bounds computed by Lag_1, and $Z(F_1)$. In each line we report the average for 10 instances.

Instance						
n	m	type	Lag_{OP}	Lag'_{OP}	Lag_1	$Z(F_1)$
10	45	OP1	547,9	414,8	529,5	529,6
11	55	OP1	613,2	459	584,1	584,3
12	66	OP1	652,1	500,2	648,3	648,9
13	78	OP1	713	558,1	706	707,5

Table 1: Lower Bounds.

Note that bounds $Z(F_1)$ and those provided by Lag_1 are quite similar, what supports the validity of our Lagrangian lower bounds. The bounds reported by [24], however, are above the theoretical bound $Z(F_1)$. The bounds evaluated by our implementation of their strategy, however, are in accordance with the theoretical results.

D Detailed Branch-and-bound Results

In Tables 2-11, we provide detailed BB results for BB_1 and BB_2, and also results for BB_{OP} [10, 8]. The first three columns of each table present information concerning the instances: the number of nodes (n), the number of edges (m), the type of the instance ($type$), and the best known upper bound (ub). For BB_{CP}, the number of nodes (n_{nodes}) and the total computational time $t(s)$ are presented. For BB_1 and BB_2, we present: the initial upper bound (ub_{heu}) obtained by the heuristic described in Section 3 and the respective computational time ($t_{heu}(s)$), the lower bound at the root node of the BB tree (lb_{root}), the computational time for solving the root node ($t_{root}(s)$), the total number of BB nodes explored (n_{nodes}), and the total time to solve the problem ($t(s)$).

We only present results for instances that were solved by at least one of these three algorithms. Entries with the symbol "-" indicate that the corresponding algorithm was not able to solve the instance within the specified time limit.

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
10	33	CP1	350	19	0	350	0	350	0	1	1	350	0	350	0	1	0
10	33	CP2	3122	17	0	3232	0	3122	0	1	1	3232	0	3122	0	1	0
10	33	CP3	646	9	0	646	0	646	0	1	1	646	0	646	0	1	0
10	33	CP4	3486	19	0	3486	0	3486	0	1	1	3486	0	3486	0	1	0
10	67	CP1	255	967	0	255	0	202.1	0	21	1	255	0	242.7	2	13	3
10	67	CP2	2042	1351	0	2042	0	1398.6	0	31	1	2042	0	1884.9	2	15	3
10	67	CP3	488	183	0	488	0	482.8	0	1	1	488	0	487.9	1	1	1
10	67	CP4	2404	1101	0	2404	0	1793	0	39	1	2404	0	2255	2	11	3
10	100	CP1	239	8205	0	239	0	159.2	0	139	2	245	0	210.4	6	72	15
10	100	CP2	1815	8157	0	1842	0	933.6	0	193	2	1891	0	1519.3	6	101	14
10	100	CP3	426	1011	0	426	0	386	0	11	1	426	0	423.2	6	3	6
10	100	CP4	2197	7931	0	2197	0	1313.8	0	157	2	2197	0	1894.7	6	71	14
15	33	CP1	745	22743	0	750	0	578	0	275	2	750	0	679.4	2	106	6
15	33	CP2	6539	25289	0	6556	0	4684.2	0	315	2	6556	0	5789.8	3	111	6
15	33	CP3	1236	2871	0	1236	0	1180	0	17	1	1236	0	1232.7	4	7	4
15	33	CP4	7245	25913	0	7245	0	5355.2	0	305	2	7245	0	6483.7	3	143	6
15	67	CP1	659	4252005	63	659	0	384.3	0	16893	59	659	0	529.8	17	3377	215
15	67	CP2	5573	3538983	53	5573	0	2579.7	0	13785	49	5573	0	4211	20	2843	175
15	67	CP3	966	55881	1	966	0	846.2	0	59	5	966	0	941.4	16	13	23
15	67	CP4	6188	3706107	55	6188	0	3204.9	0	14865	52	6188	0	4804.5	18	3115	192
15	100	CP1	620	24242281	520	620	1	319.2	1	109729	718	620	1	475.8	44	12683	2479
15	100	CP2	5184	27631419	613	5184	1	1822.5	1	118607	773	5184	1	3580.7	55	15415	2412
15	100	CP3	975	802457	16	975	1	778.8	1	871	14	975	1	901.2	40	215	155
15	100	CP4	5879	33476125	736	5879	1	2479.1	1	144309	946	5879	1	4232.5	54	20207	3169
20	33	CP1	1379	51880837	886	1390	0	886.9	1	96307	305	1388	0	1137.4	12	24106	1103
20	33	CP2	12425	49240971	879	12463	0	7037.5	1	90135	290	12463	0	9773.4	14	22285	992
20	33	CP3	1972	2230621	38	1972	0	1672.4	1	897	8	1972	0	1868.8	15	157	38
20	33	CP4	13288	46873773	823	13357	0	7914	1	90819	289	13357	0	10662.5	15	19715	879
20	67	CP1	1252	-	-	1254	5	598.3	2	24431331	271760	-	-	-	-	-	-
20	67	CP2	10893	-	-	10893	5	3797.3	1	15202397	168843	-	-	-	-	-	-
20	67	CP3	1792	-	-	1794	5	1306.1	2	89595	1527	-	-	-	-	-	-
20	67	CP4	11893	-	-	11893	4	4671.6	1	21244515	238189	-	-	-	-	-	-

Table 2: Branch-and-bound results. Instances of Cordone and Passeri [9].

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
10	45	OP1	613	4497	0	613	0	444,2	0	65	2	613	0	546,2	5	55	10
10	45	OP1	596	1115	0	596	0	493,5	0	11	1	647	0	596	4	1	4
10	45	OP1	757	3807	0	757	0	601,2	0	55	2	757	0	711,4	5	23	10
10	45	OP1	588	863	0	588	0	514,1	0	5	1	588	0	588	4	1	4
10	45	OP1	710	2753	0	710	0	574,5	0	35	1	710	0	681	6	11	9
10	45	OP1	647	2141	0	647	0	524,7	0	21	1	647	0	621,4	6	13	8
10	45	OP1	599	1295	0	599	0	489,7	0	13	1	599	0	585,1	6	7	8
10	45	OP1	653	2113	0	653	0	518,7	0	35	1	653	0	630,4	6	11	8
10	45	OP1	753	2577	0	753	0	627,2	0	19	1	753	0	718,1	6	15	9
10	45	OP1	623	2187	0	623	0	506,8	0	25	1	623	0	606	6	9	9
11	55	OP1	755	9799	0	755	0	541	0	103	3	755	0	693,4	9	43	16
11	55	OP1	750	3203	0	750	0	627	0	13	2	750	0	736,8	9	11	11
11	55	OP1	807	10327	0	809	0	583,3	0	131	2	809	0	732,3	8	61	16
11	55	OP1	816	8833	0	816	0	598,9	0	87	2	860	0	747,1	9	42	18
11	55	OP1	759	12111	0	761	0	539,8	0	119	2	759	0	703,6	8	39	14
11	55	OP1	877	12933	0	877	0	649,2	0	159	2	877	0	799,8	8	87	20
11	55	OP1	734	8063	0	734	0	529,7	0	73	2	734	0	685,3	7	35	13
11	55	OP1	843	13063	0	843	0	631,3	0	117	2	843	0	773,7	9	39	19
11	55	OP1	797	14593	0	804	0	575,9	0	153	2	804	0	723,5	8	49	18
11	55	OP1	721	3849	0	721	0	564,9	0	37	2	721	0	689,8	9	11	12
12	66	OP1	975	90541	1	985	0	659,2	0	701	4	985	0	833,2	15	327	46
12	66	OP1	903	28745	0	953	0	661,7	0	243	4	953	0	824	16	115	40
12	66	OP1	977	78679	1	977	0	655,4	0	591	4	977	0	848,5	16	259	43
12	66	OP1	936	11781	0	936	0	702,6	0	99	3	936	0	879,6	16	21	26
12	66	OP1	863	25075	0	863	0	601,7	0	169	3	874	0	782,4	16	69	34
12	66	OP1	991	49673	0	991	0	674,5	0	425	4	995	0	862,8	16	191	42
12	66	OP1	848	13657	0	848	0	645,7	0	69	4	848	0	786,9	16	25	27
12	66	OP1	842	52331	0	845	0	558,1	0	331	3	845	0	734,1	16	137	38
12	66	OP1	965	31605	0	965	0	699,1	0	191	3	965	0	877,6	15	83	33
12	66	OP1	885	26523	0	885	0	625	0	153	4	885	0	803,2	16	45	36

Table 3: Branch-and-bound results. Instances of Öncan and Punnen [24], type 1.

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
13	78	OP1	990	174049	2	1009	0	606,3	0	1005	8	1009	0	842,6	21	223	54
13	78	OP1	1022	83093	1	1022	0	702	0	351	5	1026	0	900,1	22	115	52
13	78	OP1	1089	199691	3	1089	0	697	0	909	9	1089	0	945,2	22	213	63
13	78	OP1	1163	183977	2	1165	0	803,1	0	879	7	1165	0	1033,3	21	175	53
13	78	OP1	1129	193607	3	1133	0	748,8	0	869	7	1133	0	1001,4	21	190	55
13	78	OP1	1023	80917	1	1023	0	715,4	0	269	5	1023	0	933	21	47	42
13	78	OP1	982	174309	2	982	0	613,3	0	693	7	982	0	838,3	22	249	64
13	78	OP1	1048	100115	1	1048	0	712,8	0	429	5	1048	0	929,4	22	123	54
13	78	OP1	1065	56937	0	1065	0	741,2	0	291	6	1065	0	980,3	22	49	56
13	78	OP1	1160	480411	7	1189	0	720,3	0	4743	22	1189	0	978,6	21	1445	131
14	91	OP1	1246	982445	18	1246	0	762,3	0	2837	21	1246	0	1029,1	33	799	157
14	91	OP1	1197	338485	6	1197	1	783,9	0	933	12	1197	1	1045,2	32	181	96
14	91	OP1	1398	2025109	37	1430	0	868,4	0	12245	71	1430	0	1148,9	33	2821	396
14	91	OP1	1276	1021699	19	1276	0	791,2	0	3243	24	1276	0	1043,9	34	1239	216
14	91	OP1	1266	603819	11	1266	1	809,6	0	1879	16	1266	1	1074,1	35	469	120
14	91	OP1	1188	517429	10	1206	0	736,8	0	1823	18	1206	0	1012	36	435	129
14	91	OP1	1312	1127341	21	1312	0	806,3	0	4155	29	1312	0	1076,8	35	1535	263
14	91	OP1	1171	170639	3	1171	1	807,9	0	389	7	1171	0	1071,8	33	53	86
14	91	OP1	1301	816627	16	1301	1	830,8	0	2437	20	1301	1	1097,3	32	785	176
14	91	OP1	1143	223427	4	1143	1	766,5	0	679	9	1143	1	1014,7	34	125	90
15	105	OP1	1401	3735849	87	1401	1	804,2	1	9419	84	1401	1	1155,5	45	1499	343
15	105	OP1	1404	2141949	49	1436	1	841,3	1	7063	64	1436	1	1178,1	43	858	216
15	105	OP1	1384	3593455	84	1412	1	812,6	1	8835	79	1384	1	1135,8	46	1465	333
15	105	OP1	1376	7922195	184	1383	0	741,2	1	19239	158	1383	0	1083,6	44	4057	775
15	105	OP1	1295	1810267	44	1295	1	766,5	1	3143	33	1295	1	1090,1	47	523	181
15	105	OP1	1473	2807121	64	1473	1	899,2	1	6857	65	1474	1	1227,4	44	1441	347
15	105	OP1	1389	4091815	95	1389	1	782	1	10865	92	1389	1	1128,7	44	1645	368
15	105	OP1	1407	2879425	66	1407	0	847,4	1	6189	173	1407	0	1176,2	44	937	246
15	105	OP1	1333	2191531	51	1342	1	774,2	1	6041	55	1342	1	1112,6	43	741	201
15	105	OP1	1440	3039535	71	1440	1	871,7	1	7755	66	1440	1	1198,9	44	1137	278

Table 4: Branch-and-bound results. Instances of Öncan and Punnen [24], type 1.

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
16	120	OP1	1624	-	-	1632	1	863,2	1	66679	733	1624	2	1241,7	73	14389	4766
16	120	OP1	1667	-	-	1702	2	913	1	87299	923	1667	2	1297,6	74	18641	5928
16	120	OP1	1629	-	-	1629	1	900,2	1	44877	473	1629	2	1265,1	84	8993	2891
16	120	OP1	1659	-	-	1661	1	903,3	1	72119	783	1672	1	1271,4	78	20374	6708
16	120	OP1	1695	-	-	1695	1	930,7	1	67577	733	1718	1	1311,9	69	15880	5170
16	120	OP1	1518	-	-	1549	1	833,4	1	27547	289	1521	1	1225,2	72	2449	948
16	120	OP1	1652	-	-	1721	1	864,8	1	128423	1365	1721	1	1256,1	74	21287	7029
16	120	OP1	1687	-	-	1687	1	959,2	1	43201	457	1687	1	1347,2	73	7975	2626
16	120	OP1	1543	-	-	1543	2	785	1	72719	777	1557	2	1162,2	71	19273	6233
16	120	OP1	1619	-	-	1619	2	887,3	1	42193	454	1619	2	1266,4	76	7445	2530
17	136	OP1	1843	-	-	1852	4	973,2	1	191595	2712	1852	3	1415,6	104	24776	10771
17	136	OP1	1828	-	-	1837	2	984,8	1	101253	1546	1837	2	1441,4	95	11588	5171
17	136	OP1	1859	-	-	1859	4	1021,3	1	127597	1848	1859	3	1459,8	98	15239	6611
17	136	OP1	1839	-	-	1859	2	941,4	1	287547	4263	1859	2	1398,2	97	38661	16345
17	136	OP1	1795	-	-	1834	2	904,8	1	449565	6385	1798	2	1359,8	116	46463	18853
17	136	OP1	1817	-	-	1831	3	946,1	1	215273	3168	1825	3	1388	98	29827	13176
17	136	OP1	1893	-	-	1899	2	980,4	1	382019	5370	1899	2	1438,8	89	48463	20079
17	136	OP1	1818	-	-	1836	2	987,4	1	109887	1602	1836	2	1436,4	104	12866	5936
17	136	OP1	1734	-	-	1734	3	932,4	1	67545	1098	1734	3	1369,1	97	7507	3392
17	136	OP1	1812	-	-	1830	3	922,7	1	242045	3582	1836	3	1365,3	103	35631	15514
18	153	OP1	2153	-	-	2156	4	1075,6	2	1516447	26497	-	-	-	-	-	-
18	153	OP1	2125	-	-	2136	3	1037,5	2	1890921	32555	-	-	-	-	-	-
18	153	OP1	2108	-	-	2108	7	1094,6	2	568609	9659	-	-	-	-	-	-
18	153	OP1	2026	-	-	2033	5	1029,8	2	857377	14439	-	-	-	-	-	-
18	153	OP1	2028	-	-	2030	6	941,8	2	1238159	23234	-	-	-	-	-	-
18	153	OP1	2023	-	-	2023	6	976,4	2	1121977	20526	-	-	-	-	-	-
18	153	OP1	1951	-	-	1951	4	961,2	2	552413	9755	-	-	-	-	-	-
18	153	OP1	2089	-	-	2089	3	957,3	2	3191969	55061	-	-	-	-	-	-
18	153	OP1	2138	-	-	2158	3	995,5	2	5351735	93178	-	-	-	-	-	-
18	153	OP1	2169	-	-	2169	5	1141	2	1010763	18211	-	-	-	-	-	-

Table 5: Branch-and-bound results. Instances of Öncan and Punnen [24], type 1.

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
10	45	OP2	32105	59	0	32105	0	32105	0	1	1	32105	0	32105	3	1	3
10	45	OP2	23863	65	0	23863	0	23863	0	1	1	23863	0	23863	3	1	4
10	45	OP2	18338	55	0	18338	0	18338	0	1	1	18338	0	18338	3	1	3
10	45	OP2	24613	105	0	24613	0	24613	0	1	1	24613	0	24613	3	1	3
10	45	OP2	35699	173	0	35699	0	35699	0	1	1	35699	0	35699	3	1	3
10	45	OP2	21401	99	0	21401	0	21401	0	1	1	21401	0	21401	3	1	3
10	45	OP2	26997	95	0	26997	0	26997	0	1	1	26997	0	26997	3	1	3
10	45	OP2	22992	123	0	22992	0	22992	0	1	1	22992	0	22992	3	1	3
10	45	OP2	28833	109	0	28833	0	28833	0	1	1	28833	0	28833	3	1	3
10	45	OP2	22597	63	0	22597	0	22597	0	1	1	22597	0	22597	3	1	3
11	55	OP2	40359	893	0	40359	0	40359	0	1	1	40359	0	40359	5	1	5
11	55	OP2	22735	175	0	22735	0	22735	0	1	1	22735	0	22735	5	1	5
11	55	OP2	27723	333	0	27723	0	27723	0	1	1	27723	0	27723	6	1	6
11	55	OP2	35474	105	0	35474	0	35474	0	1	1	35474	0	35474	4	1	5
11	55	OP2	29778	281	0	29778	0	29778	0	1	1	29778	0	29778	5	1	5
11	55	OP2	23877	219	0	23877	0	23877	0	1	1	23877	0	23877	5	1	5
11	55	OP2	37495	339	0	37495	0	37495	0	1	1	37495	0	37495	4	1	4
11	55	OP2	22705	49	0	22705	0	22705	0	1	1	22705	0	22705	4	1	5
11	55	OP2	19020	111	0	19020	0	19020	0	1	1	19020	0	19020	4	1	5
11	55	OP2	33910	311	0	33910	0	33910	0	1	1	33910	0	33910	4	1	5
12	66	OP2	35542	89	0	35542	0	35542	0	1	1	35542	0	35542	8	1	9
12	66	OP2	20585	129	0	20585	0	20585	0	1	2	20585	0	20585	7	1	8
12	66	OP2	38153	1103	0	38153	0	38153	0	1	1	38153	0	38153	8	1	8
12	66	OP2	32015	361	0	32015	0	32015	0	1	1	32015	0	32015	7	1	8
12	66	OP2	34136	227	0	34136	0	34136	0	1	1	34136	0	34136	9	1	9
12	66	OP2	42814	83	0	42814	0	42814	0	1	1	42814	0	42814	9	1	10
12	66	OP2	30153	353	0	30153	0	30153	0	1	1	30153	0	30153	8	1	8
12	66	OP2	25646	227	0	25646	0	25646	0	1	1	25646	0	25646	9	1	9
12	66	OP2	34183	145	0	34183	0	34183	0	1	1	34183	0	34183	8	1	8
12	66	OP2	32551	155	0	32551	0	32551	0	1	1	32551	0	32551	8	1	9
13	78	OP2	45586	347	0	45586	0	45586	0	1	2	45586	0	45586	12	1	13
13	78	OP2	49313	2185	0	49313	0	49313	0	1	2	49313	0	49313	11	1	12
13	78	OP2	44513	509	0	44513	0	44513	0	1	2	44513	0	44513	11	1	12
13	78	OP2	37250	91	0	37250	0	37250	0	1	2	37250	0	37250	11	1	12
13	78	OP2	50990	601	0	50990	0	50990	0	1	2	50990	0	50990	11	1	11
13	78	OP2	43261	481	0	43261	0	43261	0	1	2	43261	0	43261	12	1	12
13	78	OP2	36085	281	0	36085	0	36085	0	1	2	36085	0	36085	11	1	12
13	78	OP2	34474	63	0	34474	0	34474	0	1	2	34474	0	34474	10	1	11
13	78	OP2	28566	235	0	28566	0	28566	0	1	2	28566	0	28566	10	1	11
13	78	OP2	34847	357	0	34847	0	34847	0	1	2	34847	0	34847	13	1	14

Table 6: Branch-and-bound results. Instances of Öncan and Punnen [24], type 2.

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
14	91	OP2	41065	179	0	41065	1	41065	0	1	3	41065	1	41065	16	1	18
14	91	OP2	41656	747	0	41656	0	41656	0	1	2	41656	0	41656	18	1	19
14	91	OP2	48541	273	0	48541	1	48541	0	1	3	48541	1	48541	16	1	17
14	91	OP2	40335	269	0	40335	1	40335	0	1	2	40335	1	40335	19	1	20
14	91	OP2	39599	297	0	39599	1	39599	0	1	3	39599	1	39599	17	1	18
14	91	OP2	45279	813	0	45279	1	45279	0	1	3	45279	1	45279	16	1	17
14	91	OP2	55026	613	0	55102	1	55026	0	1	2	55102	1	55026	18	1	19
14	91	OP2	47163	981	0	47163	1	47163	0	1	2	47163	1	47163	16	1	17
14	91	OP2	43475	183	0	43475	1	43475	0	1	2	43475	1	43475	15	1	16
14	91	OP2	40265	237	0	40265	1	40265	0	1	2	40265	1	40265	17	1	18
15	105	OP2	62365	1101	0	62365	2	62365	0	1	3	62365	2	62365	20	1	23
15	105	OP2	50019	1477	0	50019	1	50019	0	1	3	50019	1	50019	23	1	25
15	105	OP2	56720	1007	0	56720	1	56720	0	1	3	56720	1	56720	24	1	26
15	105	OP2	58992	625	0	58992	2	58992	0	1	3	58992	1	58992	22	1	24
15	105	OP2	31530	421	0	31530	1	31530	0	1	3	31530	1	31530	21	1	22
15	105	OP2	54726	1723	0	54726	1	54726	0	1	3	54726	1	54726	28	1	29
15	105	OP2	49804	1435	0	49804	1	49804	0	1	3	49804	1	49804	26	1	28
15	105	OP2	49286	1223	0	49286	1	49286	0	1	3	49286	1	49286	29	1	30
15	105	OP2	41852	829	0	41852	1	41852	0	1	3	41852	1	41852	29	1	30
15	105	OP2	52922	1857	0	52922	1	52922	0	1	3	52922	1	52922	25	1	27
16	120	OP2	42342	699	0	42342	2	42342	1	1	4	42342	2	42342	39	1	42
16	120	OP2	45092	1819	0	45092	2	45092	1	1	4	45092	2	45092	33	1	35
16	120	OP2	41701	2053	0	41701	3	41701	1	1	5	41701	3	41701	39	1	42
16	120	OP2	46319	4351	0	46319	3	46319	1	1	5	46319	3	46319	32	1	36
16	120	OP2	41371	645	0	41371	2	41371	1	1	4	41371	2	41371	32	1	35
16	120	OP2	45108	1707	0	45108	3	45108	1	1	5	45108	3	45108	34	1	37
16	120	OP2	42021	1471	0	42021	2	42021	1	1	4	42021	2	42021	37	1	40
16	120	OP2	30880	445	0	30880	2	30880	1	1	4	30880	2	30880	32	1	35
16	120	OP2	37409	459	0	37409	2	37409	1	1	4	37409	2	37409	35	1	37
16	120	OP2	47159	16111	0	47192	1	46906.4	1	7	4	47192	2	46946.2	71	3	80
17	136	OP2	36696	797	0	36696	4	36696	1	1	6	36696	3	36696	46	1	49
17	136	OP2	48774	7163	0	48774	6	48774	1	1	9	48774	6	48774	46	1	52
17	136	OP2	45025	22963	0	45025	3	45025	1	1	5	45025	3	45025	45	1	48
17	136	OP2	33751	6503	0	33751	2	33751	1	1	5	33751	2	33751	61	1	63
17	136	OP2	44692	3463	0	44692	3	44692	1	1	6	44692	3	44692	53	1	57
17	136	OP2	45465	11419	0	45465	4	45465	1	1	7	45465	4	45465	49	1	54
17	136	OP2	38428	4139	0	38428	4	38428	1	1	7	38428	4	38428	63	1	67
17	136	OP2	40172	4933	0	40172	4	40172	1	1	6	40172	4	40172	50	1	54
17	136	OP2	44717	10113	0	44717	2	44717	1	1	5	44717	2	44717	51	1	54
17	136	OP2	40470	1987	0	40470	4	40470	1	1	7	40470	4	40470	47	1	52

Table 7: Branch-and-bound results. Instances of Öncan and Punnen [24], type 2.

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
18	153	OP2	36845	6101	0	36845	5	36845	1	1	8	36845	5	36845	73	1	79
18	153	OP2	51694	49937	1	51694	6	51694	1	1	9	51694	6	51694	63	1	70
18	153	OP2	42143	14247	0	42143	6	42143	1	1	8	42143	6	42143	64	1	70
18	153	OP2	40601	5411	0	40601	4	40601	1	1	6	40601	4	40601	71	1	75
18	153	OP2	41237	1653	0	41237	5	41237	1	1	8	41237	5	41237	71	1	76
18	153	OP2	50000	21035	0	50000	6	50000	1	1	9	50000	7	50000	71	1	78
18	153	OP2	52766	29727	1	52766	6	52766	1	1	9	52766	6	52766	73	1	80
18	153	OP2	54200	29233	1	54200	3	54200	1	1	6	54200	3	54200	60	1	63
18	153	OP2	46867	3177	0	46867	4	46867	1	1	7	46867	4	46867	74	1	79
18	153	OP2	44949	10359	0	44949	5	44949	1	1	7	44949	5	44949	63	1	68
20	190	OP2	50610	29637	1	50610	11	50610	2	1	14	50610	11	50610	109	1	120
20	190	OP2	53427	72855	3	53427	9	53427	2	1	13	53427	10	53427	108	1	118
20	190	OP2	51497	87843	3	51497	8	51497	2	1	12	51497	8	51497	117	1	126
20	190	OP2	57638	27955	1	57638	7	57638	2	1	10	57638	7	57638	120	1	127
20	190	OP2	56344	330715	15	56344	10	56344	2	1	14	56344	10	56344	114	1	125
20	190	OP2	54615	93887	4	54615	8	54615	2	1	12	54615	9	54615	117	1	126
20	190	OP2	61214	85795	4	61214	11	61214	2	1	15	61214	11	61214	121	1	132
20	190	OP2	52650	63967	3	52650	10	52650	2	1	14	52650	11	52650	112	1	123
20	190	OP2	64980	583379	29	64980	7	64980	2	1	11	64980	8	64980	105	1	113
20	190	OP2	50287	461769	21	50287	9	50287	2	1	13	50287	9	50287	108	1	117
30	435	OP2	82953	-	-	82953	71	82953	12	1	85	82953	70	82953	1383	1	1454
30	435	OP2	76977	-	-	76977	102	76977	12	1	115	76977	101	76977	1186	1	1287
30	435	OP2	88098	-	-	88098	71	88096	13	1	86	88098	70	88096	1826	1	1899
30	435	OP2	90361	-	-	90361	86	90361	12	1	99	90361	90	90361	1319	1	1410
30	435	OP2	69976	-	-	69976	104	69976	12	1	117	69976	101	69976	1327	1	1429
30	435	OP2	78864	-	-	78864	137	78864	12	1	150	78864	138	78864	1245	1	1383
30	435	OP2	73015	-	-	73015	97	73015	12	1	111	73015	97	73015	1112	1	1209
30	435	OP2	73619	-	-	73619	120	73619	12	1	133	73619	116	73619	1170	1	1287
30	435	OP2	81534	-	-	81534	108	81534	12	1	121	81534	108	81534	1393	1	1501
30	435	OP2	74602	-	-	74602	73	74602	12	1	86	74602	75	74602	1362	1	1437
50	1225	OP2	172157	-	-	172157	1160	172157	177	1	1338	-	-	-	-	-	-
50	1225	OP2	170915	-	-	170915	1140	170915	175	1	1317	-	-	-	-	-	-
50	1225	OP2	160256	-	-	160256	1325	160256	176	1	1503	-	-	-	-	-	-
50	1225	OP2	152830	-	-	152830	1214	152830	174	1	1389	-	-	-	-	-	-
50	1225	OP2	174926	-	-	174926	956	174926	176	1	1133	-	-	-	-	-	-
50	1225	OP2	154341	-	-	154341	1557	154341	173	1	1731	-	-	-	-	-	-
50	1225	OP2	180023	-	-	180023	1036	180023	176	1	1214	-	-	-	-	-	-
50	1225	OP2	153578	-	-	153578	1118	153578	175	1	1295	-	-	-	-	-	-
50	1225	OP2	179932	-	-	179932	1224	179932	174	1	1399	-	-	-	-	-	-
50	1225	OP2	155241	-	-	155241	1279	155241	175	1	1456	-	-	-	-	-	-

Table 8: Branch-and-bound results. Instances of Öncan and Punnen [24], type 2.

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
10	45	OP3	1047	801	0	1047	0	1047	0	1	1	1047	0	1047	3	1	3
10	45	OP3	1454	3383	0	1454	0	1454	0	1	1	1454	0	1454	3	1	3
10	45	OP3	1508	269	0	1508	0	1508	0	1	1	1508	0	1508	3	1	3
10	45	OP3	1490	1351	0	1490	0	1470.5	0	5	1	1490	0	1478	4	3	5
10	45	OP3	1880	549	0	1880	0	1878.7	0	1	1	1880	0	1880	3	1	3
10	45	OP3	1430	1273	0	1466	0	1411.6	0	3	1	1430	0	1430	3	1	3
10	45	OP3	1275	1257	0	1275	0	1274	0	1	1	1275	0	1275	3	1	3
10	45	OP3	1412	239	0	1429	0	1393.1	0	3	1	1429	0	1412	3	1	3
10	45	OP3	1522	649	0	1522	0	1522	0	1	1	1522	0	1522	3	1	3
10	45	OP3	1253	219	0	1253	0	1248.3	0	1	1	1253	0	1253	3	1	3
11	55	OP3	1175	1037	0	1175	0	1170.9	0	1	1	1175	0	1175	5	1	5
11	55	OP3	1614	207	0	1614	0	1614	0	1	1	1614	0	1614	5	1	5
11	55	OP3	2065	519	0	2065	0	2045.3	0	1	1	2065	0	2065	5	1	5
11	55	OP3	1854	1841	0	1854	0	1830.7	0	3	1	1854	0	1854	4	1	5
11	55	OP3	1424	159	0	1424	0	1424	0	1	1	1424	0	1424	4	1	4
11	55	OP3	1442	1123	0	1442	0	1442	0	1	1	1442	0	1442	4	1	4
11	55	OP3	1208	17495	0	1208	0	1208	0	1	1	1208	0	1208	4	1	4
11	55	OP3	1444	18889	0	1444	0	1432.6	0	5	1	1444	0	1437.4	9	9	12
11	55	OP3	1716	841	0	1716	0	1716	0	1	1	1716	0	1716	4	1	4
11	55	OP3	1509	755	0	1509	0	1509	0	1	1	1509	0	1509	4	1	4
12	66	OP3	1586	5449	0	1586	0	1582.4	0	1	1	1594	0	1586	8	1	8
12	66	OP3	2132	5079	0	2140	0	2086.2	0	7	1	2140	0	2121.9	16	5	19
12	66	OP3	1798	997	0	1798	0	1798	0	1	1	1798	0	1798	7	1	8
12	66	OP3	2227	2039	0	2227	0	2226.8	0	1	1	2227	0	2227	7	1	8
12	66	OP3	1768	1795	0	1768	0	1768	0	1	1	1768	0	1768	7	1	7
12	66	OP3	1488	40719	0	1526	0	1485	0	3	1	1526	0	1488	8	1	8
12	66	OP3	1813	319	0	1813	0	1813	0	1	1	1813	0	1813	7	1	8
12	66	OP3	2057	13491	0	2128	0	2023.9	0	3	1	2128	0	2042.2	17	7	19
12	66	OP3	2071	787	0	2071	0	2071	0	1	1	2071	0	2071	8	1	8
12	66	OP3	2076	1035	0	2270	0	2076	0	1	1	2270	0	2076	8	1	8
13	78	OP3	1731	15789	0	1731	0	1731	0	1	2	1731	0	1731	9	1	10
13	78	OP3	2484	917	0	2484	0	2484	0	1	2	2484	0	2484	10	1	11
13	78	OP3	2440	5533	0	2440	0	2436.6	0	1	2	2440	0	2440	12	1	12
13	78	OP3	2489	1881	0	2493	0	2453.2	0	9	2	2489	0	2483	23	3	25
13	78	OP3	2044	2549	0	2044	1	2044	0	1	2	2044	1	2044	11	1	12
13	78	OP3	1806	6509	0	1806	0	1805	0	1	2	1806	0	1806	11	1	11
13	78	OP3	2185	2363	0	2185	0	2185	0	1	2	2185	0	2185	10	1	11
13	78	OP3	2275	21009	0	2275	0	2272.8	0	1	2	2275	0	2275	11	1	11
13	78	OP3	1968	125889	2	1968	0	1943.1	0	3	2	1968	0	1957.7	21	3	24
13	78	OP3	2331	937	0	2331	0	2331	0	1	2	2331	0	2331	10	1	11

Table 9: Branch-and-bound results. Instances of Öncan and Punnen [24], type 3.

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
14	91	OP3	1955	35983	0	1955	1	1955	0	1	2	1955	1	1955	16	1	17
14	91	OP3	2555	1297	0	2555	0	2555	0	1	2	2555	0	2555	15	1	16
14	91	OP3	3182	10799	0	3182	1	3169.8	0	3	2	3182	1	3182	16	1	17
14	91	OP3	2516	72339	1	2516	1	2516	0	1	2	2516	1	2516	17	1	18
14	91	OP3	2551	9005	0	2551	1	2551	0	1	2	2551	1	2551	16	1	17
14	91	OP3	2818	119579	2	2818	1	2797	0	5	4	2895	2	2816.2	38	7	46
14	91	OP3	2457	23607	0	2457	0	2457	0	1	2	2457	0	2457	16	1	17
14	91	OP3	2491	14763	0	2491	1	2482.9	0	1	3	2491	1	2491	17	1	19
14	91	OP3	2293	237	0	2293	1	2293	0	1	3	2293	1	2293	15	1	17
14	91	OP3	2461	5509	0	2461	1	2454.1	0	1	3	2461	1	2460.9	19	1	20
15	105	OP3	2181	97099	2	2181	1	2177.7	0	1	3	2181	1	2181	22	1	23
15	105	OP3	2840	3911	0	2840	1	2840	0	1	3	2840	1	2840	23	1	25
15	105	OP3	2921	4111	0	2921	2	2921	0	1	4	2921	2	2921	22	1	24
15	105	OP3	2780	3641	0	2780	1	2780	0	1	3	2780	1	2780	23	1	24
15	105	OP3	2340	84369	1	2369	2	2305.5	1	15	5	2369	2	2340	29	1	31
15	105	OP3	2917	11983	0	2917	1	2904.9	1	3	3	2917	1	2917	25	1	26
15	105	OP3	2291	27405	0	2291	2	2265	1	5	4	2291	1	2291	24	1	25
15	105	OP3	2537	781	0	2537	2	2537	0	1	4	2537	2	2537	23	1	25
15	105	OP3	2504	6853	0	2516	0	2495	1	3	2	2504	0	2504	24	1	24
15	105	OP3	2577	4879	0	2577	2	2575.9	0	1	4	2577	2	2577	21	1	24
16	120	OP3	2418	211997	5	2418	2	2408.5	1	1	5	2418	2	2418	36	1	39
16	120	OP3	2507	26651	0	2507	1	2507	1	1	4	2507	2	2507	36	1	38
16	120	OP3	3241	33041	0	3241	4	3214.6	1	1	7	3241	4	3241	33	1	37
16	120	OP3	3600	662209	16	3626	2	3585.5	1	3	5	3626	2	3593	76	12	106
16	120	OP3	3097	24013	0	3097	2	3022.5	1	5	6	3097	2	3096.8	73	1	76
16	120	OP3	2741	8043	0	2741	3	2741	1	1	5	2741	3	2741	32	1	35
16	120	OP3	3264	60473	1	3264	3	3216.6	1	15	7	3264	4	3248.7	70	13	94
16	120	OP3	2958	88063	2	2958	1	2943.7	1	1	4	2958	1	2958	41	1	42
16	120	OP3	3079	38665	0	3079	2	3079	1	1	4	3108	2	3079	36	1	38
16	120	OP3	2896	605	0	2896	3	2896	1	1	6	2896	3	2896	34	1	38
17	136	OP3	2842	987395	31	2842	3	2835.7	1	1	6	2842	3	2842	51	1	54
17	136	OP3	3734	1000000	36	3734	3	3724.9	1	3	5	3734	2	3734	64	1	67
17	136	OP3	3543	687011	21	3543	3	3455.7	2	9	7	3543	3	3513.9	85	9	115
17	136	OP3	3165	382535	12	3165	3	3150.1	1	5	7	3165	3	3161.2	92	5	104
17	136	OP3	2920	4437	0	2920	4	2920	1	1	7	2920	4	2920	42	1	46
17	136	OP3	3445	17131	0	3445	4	3442.4	1	1	7	3445	4	3445	42	1	47
17	136	OP3	3483	380767	12	3483	4	3419.1	2	7	9	3483	4	3482.9	104	1	108
17	136	OP3	3321	352711	10	3368	2	3252.6	1	21	7	3368	2	3316.9	86	11	117
17	136	OP3	3460	52593	1	3460	3	3460	1	1	6	3460	3	3460	45	1	48
17	136	OP3	3809	71073	2	3832	4	3757.4	1	3	8	3832	4	3805.1	115	11	150

Table 10: Branch-and-bound results. Instances of Öncan and Punnen [24], type 3.

Instance				BB_{CP}		BB_1						BB_2					
n	m	$type$	ub	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$	ub_{heu}	$t_{heu}(s)$	lb_{root}	$t_{root}(s)$	n_{nodes}	$t(s)$
18	153	OP3	2970	979125	35	2970	5	2968.1	2	1	8	2970	5	2970	64	1	69
18	153	OP3	3733	154609	5	3740	4	3706.6	2	5	8	3740	4	3730.2	156	11	217
18	153	OP3	3563	327861	10	3563	6	3543.6	2	1	9	3563	6	3563	68	1	75
18	153	OP3	4060	1000000	33	4060	7	4044.7	2	1	10	4060	7	4058.5	148	1	167
18	153	OP3	3300	127447	4	3333	5	3300	1	1	8	3333	5	3300	63	1	68
18	153	OP3	3191	61887	2	3191	4	3191	1	1	7	3191	4	3191	62	1	66
18	153	OP3	3700	376161	16	3700	4	3692	2	1	7	3700	4	3700	68	1	73
18	153	OP3	3560	4955	0	3560	7	3546.5	2	1	11	3560	7	3560	60	1	68
18	153	OP3	3990	73855	2	3990	7	3964.2	2	3	10	3990	7	3990	72	1	79
18	153	OP3	3623	778303	29	3623	7	3561.7	2	9	11	3623	7	3612.8	134	3	155
20	190	OP2	22428	17	0	22756	0	22428	0	1	1	22756	0	22428	0	1	0
20	190	OP2	15253	21	0	15253	0	15253	0	1	1	15253	0	15253	0	1	0
20	190	OP2	21570	23	0	21570	0	21570	0	1	1	21570	0	21570	0	1	0
20	190	OP2	19535	9	0	19535	0	19535	0	1	1	19535	0	19535	0	1	0
20	190	OP2	24964	15	0	24964	0	24964	0	1	1	24964	0	24964	0	1	0
20	190	OP2	9821	21	0	9821	0	9821	0	1	1	9821	0	9821	0	1	0
20	190	OP2	11810	11	0	11810	0	11810	0	1	1	11810	0	11810	0	1	0
20	190	OP2	13869	11	0	13869	0	13869	0	1	1	13869	0	13869	0	1	0
20	190	OP2	7742	13	0	7742	0	7742	0	1	1	7742	0	7742	0	1	0
20	190	OP2	15747	7	0	15747	0	15747	0	1	1	15747	0	15747	0	1	0
30	435	OP2	6992	-	-	6992	138	6865.3	15	35	214	6992	137	6964.2	2437	15	5157
30	435	OP2	9057	-	-	9066	183	8700.3	16	129	491	9066	184	8859.5	2069	81	17856
30	435	OP2	7823	-	-	7823	165	7823	13	1	179	7823	161	7823	1233	1	1395
30	435	OP2	7936	-	-	7936	179	7886.1	14	1	200	7936	177	7936	1214	1	1392
30	435	OP2	8092	-	-	8163	219	8042.1	18	9	272	8163	217	8092	1380	1	1597
30	435	OP2	8566	-	-	8568	159	8438.3	15	15	223	8568	159	8533.5	2928	9	5691
30	435	OP2	7525	-	-	7538	248	7364.4	14	41	342	7538	246	7472.3	2079	31	4958
30	435	OP2	8645	-	-	8645	163	8409.5	15	49	318	8645	161	8545.9	2211	29	8001
30	435	OP2	8692	-	-	8739	183	8526.1	15	31	296	8739	176	8653.4	2348	28	5015
30	435	OP2	7239	-	-	7239	179	7209.2	14	1	197	7239	179	7239	1422	1	1601
50	1225	OP2	17524	-	-	17650	4255	16933.3	195	735	19045	-	-	-	-	-	-
50	1225	OP2	16780	-	-	16780	4871	16558.9	192	13	6061	-	-	-	-	-	-
50	1225	OP2	13198	-	-	13231	5608	12940.4	189	71	8244	-	-	-	-	-	-
50	1225	OP2	15137	-	-	15218	4768	14708.5	193	123	11761	-	-	-	-	-	-
50	1225	OP2	16358	-	-	16358	6341	15677.9	195	405	14950	-	-	-	-	-	-
50	1225	OP2	14996	-	-	14996	7265	14781.4	190	21	8791	-	-	-	-	-	-
50	1225	OP2	17282	-	-	17282	4987	17222.5	188	1	5219	-	-	-	-	-	-
50	1225	OP2	14975	-	-	14975	4653	14959.9	185	1	4856	-	-	-	-	-	-
50	1225	OP2	13594	-	-	13594	3730	13591.8	185	1	3920	-	-	-	-	-	-
50	1225	OP2	18062	-	-	18163	3799	17736	193	111	8141	-	-	-	-	-	-

Table 11: Branch-and-bound results. Instances of Öncan and Punnen [24], type 3.