Submitted to *INFORMS Journal on Computing*
manuscript (Please, provide the manuscript number!)

A Dual Bounding Framework through Cost Splitting for Binary Quadratic Optimization

Mahdis Bayani
Polytechnique Montréal, CIRRELT, and GERAD, Canada
mahdis.bayani@polymtl.ca

Borzou Rostami
Alberta School of Business, University of Alberta, Edmonton, Canada
borzou@ualberta.ca

Yossiri Adulyasak
HEC Montréal and GERAD, Canada
yossiri.adulyasak@hec.ca

Louis-Martin Rousseau
Polytechnique Montréal and CIRRELT, Canada
louis-martin.rousseau@cirrelt.net

Binary quadratic programming (BQP) is a class of combinatorial optimization problems comprising binary variables, quadratic objective functions and linear/non-linear constraints. This paper examines a unified framework to reformulate a BQP problem with linear constraints to a new BQP with an exponential number of variables defined on a graph. This framework relies on the concept of stars in the graph to split the quadratic costs into adjacent and non-adjacent components indicating in-star and out-of-star interactions. We exploit the star-based structure of the new reformulation to develop a decomposition-based column generation algorithm. In our computational experiments, we evaluate the performance of our methodology on different applications with different quadratic structures. The quadratic component of the problem is dealt with in the column generation master problem and its subproblem. Results indicate the superiority of the framework over one of the state-of-the-art solvers, GUROBI, when applied to various benchmark reformulations with adjacent-only or sparse quadratic cost matrices. The framework outperforms GUROBI in terms of both dual bound and computational time in almost all instances.

Key words: binary quadratic programming, combinatorial optimization, column generation, semi-assignment problem, multiple object tracking problem

1. Introduction

Binary quadratic programming (BQP) is a large class of combinatorial optimization problems that arise from modeling real-life applications, for example, in management, engineering, logistics and

network design (Punnen et al., 2019). Given graph $G=(V,E)$ with node set $V=\{1,2,\ldots,|V|\}$ and edge set $E=\{1,2,\ldots,m\}$, a BQP problem with linear constraints on graph G can be specified using a quadratic cost matrix $\boldsymbol{q}\in\mathbb{R}^{m\times m}$ and a linear cost vector $\boldsymbol{c}\in\mathbb{R}^m$ and is formulated as follows:

$$\text{BQP:}\quad \min \quad \sum_{e\in E} c_e x_e + \sum_{(e,f)\in\mathcal{E}} q_{ef} x_e x_f \qquad \text{s.t.} \quad \boldsymbol{x}\in X, \tag{1}$$

where $X\subseteq\{0,1\}^m$ is the set of feasible binary vectors and $\mathcal{E}=\{(e,f)\in E\times E : e<f\}$. Without loss of generality, we only consider the symmetric quadratic costs in the objective function as one can replace q_{ef} and q_{fe} by their average $(q_{ef}+q_{fe})/2$ for the case of asymmetric quadratic costs. Note that the matrices and vectors are shown in bold text throughout the paper.

Many quadratic combinatorial optimization problems can be naturally formulated in this fashion. Some important examples include the quadratic assignment problem (Çela, 2013), the quadratic knapsack problem (Pisinger, 2007), the quadratic travelling salesman problem (Fischer, 2014; Rostami et al., 2016; Punnen et al., 2017), the quadratic shortest path problem (Hu and Sotirov, 2018; Rostami et al., 2018), the quadratic spanning tree problem (Assad and Xu, 1992; Rostami and Malucelli, 2015; Pereira and da Cunha, 2020) and the quadratic set covering problem (Escoffier and Hammer, 2007; Punnen et al., 2019).

The main difficulty of solving BQP stems from the quadratic structure of the objective function rather than the combinatorial nature of the problem. For example, when $X=\{0,1\}^m$, problem (1) is equivalent to unconstrained quadratic binary optimization and hence to the max-cut problem, which is NP-hard (Barahona, 1983). In general, BQP problems are NP-hard, even if the linear optimization problem over the same feasible set is tractable. This is the case for many BQP problems, including the quadratic spanning tree problem (Assad and Xu, 1992) and the quadratic assignment problem (Çela, 2013), while their linear counterparts are polynomially solvable.

One way to deal with the challenges of the BQP is to identify patterns and structures in the objective function, constraint matrices, or instances (Punnen et al., 2017; Bettiol et al., 2022). In this paper, we employ such a technique to exploit the structure of the quadratic cost matrix $\boldsymbol{q}$. Our methodology relies on splitting the quadratic cost $\boldsymbol{q}$ into (i) in-star interactions in which the quadratic costs between adjacent edges are investigated and (ii) out-of-star interactions corresponding to the quadratic costs between non-adjacent edges. This leads to a new reformulation of the BQP with exponentially many variables, each associated with a star in G. Inspired by the star-reformulation of Pereira et al. (2013) for the adjacent quadratic minimum spanning tree problem, we develop a column generation to derive bounds for some classes of BQPs with different levels of in-star and out-of-star quadratic costs.

1.1. Literature Review

One of the most natural ways to solve the BQP problems with an exact method is to linearize the quadratic terms of the model and solve the resulting mixed-integer linear programming (MILP) using state-of-the-art solvers. The standard linearization technique is one of the most well-known linearizations in the literature of BQPs (Glover and Woolsey, 1974). However, two main concerns appear when dealing with the MILP reformulation: the increased size of the problem (in terms of variables and constraints) and the quality of the obtained dual bounds. There have been many attempts to deal with these concerns in the literature. Adams and Sherali (1990), Adams and Forrester (2005) and Sherali and Smith (2007) provide different reduced-size MILP reformulations of the BQP, while Liberti (2007) introduces a compact linearization approach for a general class of binary quadratic problems subject to assignment constraints. This approach is then revised in Mallach (2018) by proposing two new necessary and sufficient conditions to achieve consistent linearization for this class of problems. In another study, Jünger and Mallach (2021) investigate the solution methods proposed in the literature for unconstrained BQPs based on linear programming. They provide some enhancements to the algorithm of the central separation problem arising in solving these problems. Following this study, Charfreitag et al. (2022) present a solver called McSparse, based on a branch-and-cut algorithm to obtain exact solutions for sparse unconstrained binary quadratic programming problems. Furthermore, Hahn et al. (2012), Sherali and Adams (2013) and Rostami and Malucelli (2015) develop the reformulation-linearization technique (RLT), which generally provides stronger MILP reformulations. Recently, Mallach (2023) investigates the effectiveness of the inductive linearization technique for some applications of BQPs. They represent that although this linearization technique is more compact in terms of constraints, the continuous relaxation is at least as tight as the standard linearization technique.

Semi-definite programming (SDP), quadratic reformulation, and cutting-plane methods are alternative approaches used to generate strong relaxations of BQP. In SDP, which is considered an extension of MILP reformulations, non-negativity constraints are replaced by positive semi-definiteness constraints (Helmberg et al., 2000; Oustry, 2001). In quadratic reformulations, one must alter the objective function of a BQP problem and transform it into an equivalent convex/non-convex BQP problem to generate tighter dual bounds (Billionnet et al., 2009; Rostami et al., 2022). The use of valid inequalities, which are generated and added in a cutting-plane fashion, is another approach commonly adopted in the literature (Fischer, 2014).

Another relevant approach to obtain a stronger reformulation for the BQP is to use decomposition techniques. Variants of decompositions such as Lagrangian decomposition, graph partitioning, and CG methods are employed to explore the bounds of unconstrained BQP problems (Mauri and Lorena, 2011, 2012). For constrained BQP problems, Chen et al. (2017) represent bounds for the

BQP using a Lagrangian-based heuristic method. Some examples of using decompositions to tackle the problem of investigating bounds for some specific BQP problems are observed for the quadratic knapsack problem and the minimum spanning tree problem (Billionnet and Soutif, 2004; Pereira and da Cunha, 2020).

There are a few papers in the literature reformulating a BQP model for a specific application into a MILP with an exponential number of variables, which is solved by CG. Aloise et al. (2010) reformulate the mixed 0-1 quadratic programming model of the modularity maximization problem and solve the reformulation using a stabilized CG. In a related study, a CG heuristic is used in a districting problem to produce the best territories for the purpose of financial product pricing (De Fréminville et al., 2015). Rostami et al. (2016) propose a lower bounding procedure for the asymmetric quadratic traveling salesman problem. They reformulate the problem as a MILP with an exponential number of cycles as variables and solve the relaxation using a CG. Recently, Yarkony et al. (2020) developed an extended MILP formulation for correlation clustering. They consider solving correlation clustering for several computer vision applications through CG, Benders decomposition, and dynamic programming (DP).

The most related works to our study are the papers devoted to the reformulation of the adjacent-only quadratic minimum spanning tree problem (AQMSTP) to a MILP and solution algorithms applied to solve the reformulation. More specifically, a star-reformulation is introduced in Pereira et al. (2013) based on the concept of stars in graph theory to reformulate the AQMSTP by a stronger linear programming (LP). Then, an algorithm based on dynamic column and row generation is proposed to determine the dual bounds of the problem. In another study, Pereira et al. (2015) present a branch-and-cut-and-price algorithm based on this reformulation of the AQMSTP and a branch-and-cut algorithm based on projecting out the decision variables of this model. There are also other reformulations and algorithms for the AQMSTP in the literature based on the proposed star-based model (e.g., Pereira and da Cunha (2018, 2020)).

In recent years, identifying patterns and structures inherent in large-scale optimization problems and using them to efficiently tackle these real-life problems has drawn researchers' attention. Exploring the constraint matrix structures (Punnen et al., 2019), finding important patterns in a specific problem class (Khaniyev, 2018), and revealing the structure of a data instance related to a large-scale problem (Khaniyev et al., 2020) are among the notable techniques in this area. One of the structures explored in the MILP literature is the singly bordered block diagonal (BBD) structure. Exploiting this structure in the constraint matrices of a MILP leads to Dantzig-Wolfe (DW) decomposition, Lagrangian relaxation, and branch-and-price (Bergner et al., 2015; Khaniyev et al., 2018). Bergner et al. (2015) provide a computational proof-of-concept to show that the DW reformulation can be automated and applied to all MILPs by exploiting and rearranging the

structure of the constraint's matrix in various ways. They suggest a score to measure the quality of each decomposition and identify the most useful one for the DW reformulation of a MIP. To our knowledge, there are few methodological studies concerning structures in BQP problems. Bettiol et al. (2022) tackle BQP from the CG perspective and construct an approach to study the structure of block-decomposable problems in BQP. They present two types of relaxations that acquire strong lower bounds for general BQP and block-decomposable BQP specifically. The relaxations are based on DW reformulation, while CG is used as their solution method. In addition, some studies explore the linearizability of the quadratic cost matrices. Punnen et al. (2017) investigate the structure of quadratic cost matrices to propose necessary and sufficient conditions for linearizability of the quadratic traveling salesman problem. In another related work, Hu and Sotirov (2021) present a linearization-based lower bounding scheme applicable to several BQP problems using a certificate for a quadratic function to be non-negative on the feasible set.

To the best of our knowledge, the idea of exploring the effectiveness of the star-reformulation on more general cases rather than the AQMSTP has not been investigated in the literature. Our major focus here is to explore the generalizability and usability of this reformulation by splitting the quadratic cost to model and solve more general problems in BQP.

1.2. Main Contributions

Our main contributions are summarized as follows:

- We investigate the idea of the star-reformulation proposed for the AQMSTP in the literature on the adjacent-only BQP problems. Moreover, we show how to exploit such a structure to split the general BQP problems' quadratic costs into adjacent and non-adjacent components.
- We develop a column generation (CG) for each specific reformulation to derive valid dual bounds.
- To demonstrate the potential of the star-reformulation, the cost-splitting framework and the solution methodology, we consider three BQP problems (quadratic semi-assignment, adjacent-only quadratic semi-assignment, and multiple object tracking) whose reformulations lead to different master problems and pricing subproblems, i.e., (i) quadratic master problem and unconstrained BQP pricing subproblem, (ii) linear programming master problem and unconstrained BQP pricing subproblem, and (iii) linear programming master problem and constrained BQP pricing subproblem.
- We perform extensive computational experiments to evaluate the proposed reformulations and CG algorithms on the adjacent-only and general BQP problems. The special quadratic matrix structure of adjacent-only BQPs and BQP problems with sparse quadratic costs derive the most significant benefit from the star reformulation and the proposed splitting framework.

The rest of the paper is organized as follows. In Section 2, we present the idea of star-reformulation and introduce the adjacent-only BQP class of problems. Section 3 is corresponding to the proposed cost-splitting framework and then we explain a column generation to solve the reformulation. Section 4 is dedicated to presenting and reformulating one BQP example to evaluate the cost-splitting reformulation and two adjacent-only BQP problems, the AQSAP and MOT, as our illustrative examples to generalize star-reformulation. Finally, in Section 5, we explain our test sets and display the computational results from the introduced problems and in Section 6, we wrap up the paper and provide some future possible directions for the research.

2. Star-Reformulation For Adjacent-only BQP Problems

In many real-life applications modeled as BQP on graphs, the quadratic costs appear only between adjacent edges. As such, interaction costs are zero for pairs of edges that do not share a common endpoint. Some notable examples include the adjacent-only quadratic minimum spanning tree problem (Pereira et al., 2013; Pereira and da Cunha, 2020), the quadratic traveling salesman problem (Fischer, 2014; Punnen et al., 2017), the adjacent quadratic assignment problem (Fischer et al., 2009), the adjacent quadratic shortest path problem (Rostami et al., 2015; Hu and Sotirov, 2018) and variants of the correlation clustering and the modularity maximization problems (Bonizzoni et al., 2008; Aloise et al., 2010; Yarkony et al., 2020).

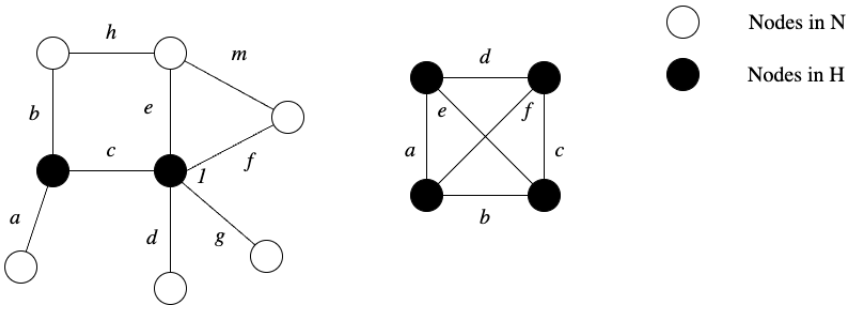
To exploit such a unique structure in developing a solution strategy, we follow Pereira et al. (2013) to reformulate the adjacent-only BQP to a new model with exponentially many variables based on the concept of stars in a graph. To this end, we first provide some notations to introduce the concept in Section 2.1 and then present the star-based reformulation in Section 2.2. For further information on the star-reformulation, we refer the readers to Pereira et al. (2013) where the idea is introduced for the adjacent-only QMSTP.

2.1. Definitions

Consider the BQP problem on graph $G=(V,E)$ in (1). Without loss of generality, we assume that $V=N\cup H$ where N and H are disjoint sets. We also take into account the possibility of N being an empty set. Thus, in this situation, the set V is equal to the set H. For each $v\in H$, we define $\delta(v)\subseteq E$ as the set of edges incident to node v and let $A=\cup_{v\in H}\delta(v)$ be the set of all edges with exactly one endpoint in H, considering exceptionally, when the set N is empty, both endpoints of all edges of the related graph are thereby included in H. So, if $N\neq\emptyset$, A is the set of edges with exactly one endpoint in H and if $N=\emptyset$, then $A=E$. Two distinct edges of A, say $e=\{i,j\}$ and $f=\{k,\ell\}$, are adjacent if they share a common endpoint v in H, i.e., if $\{i,j\}\cap\{k,\ell\}=v\in H$. We denote by $\mathcal{A}$ the set of distinct pairs of adjacent edges in A:

$$\mathcal{A}=\Big\{(e,f)\in A\times A\ : e=\{i,j\}, f=\{k,\ell\},\ \{i,j\}\cap\{k,\ell\}=v\in H\Big\} \tag{2}$$

Figure 1 **Two examples of graph to demonstrate the concepts of set A, adjacency set $\mathcal{A}$ and star s**



Note: Graph on the left: $A=\{a,b,d,e,f,g\}$, $\mathcal{A}=\{(a,b),(d,e),(d,f),(d,g),(e,f),(e,g),(f,g)\}$
Graph on the right: $A=\{a,b,c,d,e,f\}$,
$\mathcal{A}=\{(a,b),(a,d),(a,e),(a,f),(b,c),(b,e),(b,f),(c,d),(c,e),(c,f),(d,e),(d,f)\}$,
Node1 is specified to motivate a star center candidate

The graphs indicated in Figure 1 illustrate the concept of adjacent edges by defining sets A and $\mathcal{A}$.

These definitions will lead to the following definition for the star-shaped sub-graph s. That is, for each $v \in H$ we define a star s centered at node v as any subset of $\delta(v)$ whose elements are in the set A, including an empty set and let S^v be the set of all stars centered at node v. The mathematical representation of the aforementioned definitions can be expressed as:

$$S^v = \{s : s \subseteq (\delta(v) \cap A)\}.$$

Therefore, $S = \bigcup_{v \in H} S^v$ includes all the possible stars centered at nodes $v \in H$ in the graph. As an example, in the left graph of Figure 1, $s = \{e, f, g, d\}$ is the largest possible star centered at node 1. Note that as both endpoints of the edge c are in the set H, this edge is not included in any possible star of this graph.

2.2. Star-Reformulation

In the adjacent-only cases, interaction costs are zero for pairs of edges that do not share a common endpoint in the defined subset H. Due to equation (2), in this BQP class, the quadratic cost $\boldsymbol{q}$ for the pairs of edges (e, f) that are not covered by the set $\mathcal{A}$ are zero. Based on the represented concepts and the fact that each star $s \in S$ consists of a subset of pairs in $\mathcal{A}$, we can reformulate an adjacent-only BQP problem in terms of stars. For each $s \in S$ let $C_s = \sum_{e \in s} c_e + \sum_{e,f \in s} q_{ef}$ represent the total linear and quadratic cost of star s. We define a new binary decision variable λ_s for each star $s \in S$ to indicate if the corresponding star s is included in the solution of the BQP problem or not. We also denote $\mathcal{F}$ to depict the feasible space of the problem. By preserving the definition of variable $\boldsymbol{x} \in X$, we provide the following star-based reformulation:

$$\min \quad \sum_{s \in S} C_s \lambda_s + \sum_{e \in E \setminus A} c_e x_e \tag{3}$$

$$\text{s.t.} \quad (\boldsymbol{x}, \boldsymbol{\lambda}) \in \mathcal{F}(\boldsymbol{x}, \boldsymbol{\lambda}) \tag{4}$$

$$\boldsymbol{x} \in \{0,1\}^m \tag{5}$$

$$\boldsymbol{\lambda} \in \{0,1\}^{|S|} \tag{6}$$

The objective function of the reformulation minimizes the total cost of the problem and consists of two different parts. The first part corresponds to the cost of star s, including the linear costs of the edges inside the star and the interaction between adjacent edges of that star. The second term of the objective function reflects the linear cost of the edges which are not incorporated in any possible star. Constraint (4) links the feasible region of the problem to the stars by coupling the original variables $\boldsymbol{x}$ and new variables $\boldsymbol{\lambda}$. It can also include the constraints which are only related to the variables $\boldsymbol{\lambda}$ and the constraints which are only associated with variables $\boldsymbol{x}$. We assume, without loss of generality, that such linking constraints can always be found, because for each $v \in H$ and each $e \in \delta(v)$, there exist parameters $b_{es} \in [0,1]$ such that $x_e = \sum_{s \in S^v} b_{es}\lambda_s$ (e.g., see Section 4). Since some of the constraints in the original model can be included in the definition of the star in this reformulated counterpart, the $\boldsymbol{x}$-only constraints can be considered as a subset of X in (1).

To elaborate on the star-reformulation consider the right graph of Figure 1. In this graph, a given set of star centers (H) potentially comprises the whole node set of the graph (V) (i.e., $H = V$). Intuitively, for BQP problems with this property, set A contains all the graph's edges. To illustrate this, consider the AQMST problem (Assad and Xu, 1992), in which the set H is equal to all nodes of the graph ($N = \emptyset$). Thus, in the reformulated model of the AQMST of Pereira et al. (2013), the objective function consists of only a linear term to represent the cost of stars. Nonetheless, in the left graph of Figure 1, some of the nodes are not considered as possible star centers ($N \neq \emptyset$), so we keep the second term of the reformulation's objective function to demonstrate the linear cost of the edges which are not included in any possible star. In this study, we employ the star-reformulation to model two applications, more details on modeling and solution methodology are presented in Section 4.

3. Cost-Splitting and Star-Reformulation For General BQP Problems

In order to tackle the complexity of the general BQP problem of (1) using decomposition and inspired by the idea of the star-reformulation of Section 2, we propose a cost-splitting framework to reformulate BQP problems. Our methodology relies on the concept of star structures to partition the objective function of (1) at any feasible solution $\bar{\boldsymbol{x}}$ into four parts: in-star linear costs, out-of-star linear costs, in-star quadratic costs, and out-of-star quadratic costs. More precisely, given a feasible solution $\bar{\boldsymbol{x}} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_{m_1}, \ldots, \bar{x}_m) \in X \subseteq \{0,1\}^m$, we can rewrite it as $\bar{\boldsymbol{x}} = (\bar{\boldsymbol{x}}^1, \bar{\boldsymbol{x}}^2)$ with $\bar{\boldsymbol{x}}^1 \in \{0,1\}^{m_1}$ and $\bar{\boldsymbol{x}}^2 \in \{0,1\}^{m-m_1}$ and where $\bar{\boldsymbol{x}}^1$ is related to feasible solutions of edges in set A

and $\bar{\boldsymbol{x}}^2$ stands for feasible solutions of edges in $E \setminus A$. Therefore, the objective function of (1) at $\bar{\boldsymbol{x}}$ can be written as:

$$\sum_{e \in A} c_e \bar{x}_e + \sum_{e \in E \setminus A} c_e \bar{x}_e + \sum_{(e,f) \in \mathcal{A}} q_{ef} \bar{x}_e \bar{x}_f + \sum_{(e,f) \in \mathcal{E} \setminus \mathcal{A}} q_{ef} \bar{x}_e \bar{x}_f$$

As presented in Section 2, the first and third terms of this formula together depict the costs of inside stars, while the last term as a quadratic cost is associated with the interaction between pairs of non-adjacent edges. Therefore, the cost-splitting reformulation of the BQP problem of (1) is proposed where the constraints are maintained as in (4)-(6) and the objective function is as follows:

$$\min \quad \sum_{s \in S} C_s \lambda_s + \sum_{e \in E \setminus A} c_e x_e + \sum_{(e,f) \in \mathcal{E} \setminus \mathcal{A}} q_{ef} x_e x_f \tag{7}$$

$$(4) - (6). \tag{8}$$

Although modeling BQP problems based on separating adjacent edge interactions and non-adjacent edge interactions is the principal notion behind the proposed model, each term of the objective function can potentially be eliminated based on the specific problem structure in different applications. Nonetheless, we keep the first term of the objective function as the basis of our reformulation. As an example, in the quadratic minimum spanning tree (QMST) problem (Assad and Xu, 1992), since the quadratic cost comprises the interactions between all pairs of edges (adjacent and non-adjacent) and the set of possible star centers consists of the complete node set V, in the objective function of the cost-splitting reformulated model of the QMST problem, we incorporate a linear term to represent the cost of stars and a quadratic term for interactions between non-adjacent edges. Another notable example is the uncapacitated single allocation p-Hub median problem (USApHMP) (O'Kelly, 1987; Meier et al., 2016) which can be formulated as a BQP problem. This problem can be reformulated using our star-based model, consisting of linear costs inside the possible stars and quadratic interactions between stars with different centers.

3.1. Column Generation Approach

The use of the star representation in the proposed reformulation of the BQP in (7)-(8) results in an exponential number of variables $\lambda_s, s \in S$. Pereira et al. (2013) suggest a column generation for the star-reformulation of the AQMST problem. Inspired by this idea, CG is applied to deal with the proposed cost-splitting of (7)-(8).

CG is an efficient iterative algorithm for providing dual bounds for problems with an exponential number of variables (columns) (Dantzig and Wolfe, 1960). In each iteration of the algorithm, it solves one restricted master problem (RMP) which is the problem restricted to a small subset of the variables, and one or several pricing subproblems. Using the dual information of the RMP, the

pricing subproblem is solved to verify the optimality of the master problem and the CG algorithm stops if the optimality condition is satisfied. Otherwise, one or more new variables determined by the subproblem will be added to the RMP and the updated RMP will be solved in the new iteration.

Since the generic reformulation (7)-(8) is a non-convex quadratic problem, the standard CG procedure cannot be directly applied. If the model is convex, meaning that the matrix $\boldsymbol{q}$ is positive semidefinite, then either the underlying nature of that problem brings out a convex quadratic subproblem or a convex quadratic master problem. In the case of a convex RMP, the primal and dual solutions of the RMP can be obtained by state-of-the-art solvers. However, in our proposed framework, we do not restrict the definition of $\boldsymbol{q}$ to positive semidefinite matrices. Therefore, one has to deal with the quadratic term of the objective function. Different types of convexification (Billionnet et al., 2009), linearization, semidefinite programming relaxation and BQP relaxation for block-decomposable problems (Bettiol et al., 2022) are alternative options for handling the BQP master problem. Nevertheless, some of these methods require additional complex constraints and variables which may adversely affect their performance. Our approach is based on the idea of converting the quadratic cost to linear cost by separating the in-star and out-of-star costs and then replacing each of them by a linear term. Here, we follow linearization techniques to transform the quadratic objective function into an equivalent MILP. We define y_{ef} as the linearized variable to replace the quadratic terms $x_e x_f$, $(e,f) \in \mathcal{E} \setminus \mathcal{A}$, impose a set of linking constraints $\mathcal{P}(\boldsymbol{x}, \boldsymbol{y})$ to guarantee $y_{ef} = x_e x_f$, and consider $\bar{S} \subseteq S$ as a subset of possible stars, so we obtain the following RMP for the LP relaxation of the problem:

$$\min \quad \sum_{s \in \bar{S}} C_s \lambda_s + \sum_{e \in E \setminus A} c_e x_e + \sum_{(e,f) \in \mathcal{E} \setminus \mathcal{A}} q_{ef} y_{ef} \tag{9}$$

$$\text{s.t.} \quad (\boldsymbol{x}, \boldsymbol{\lambda}) \in \bar{\mathcal{F}}(\boldsymbol{x}, \boldsymbol{\lambda}) \tag{10}$$

$$(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{P}(\boldsymbol{x}, \boldsymbol{y}) \tag{11}$$

$$\boldsymbol{y} \in \mathbb{R}_+^{|(e,f) \in \mathcal{E} \setminus \mathcal{A}|} \tag{12}$$

$$\boldsymbol{\lambda} \in [0,1]^{|\bar{S}|} \tag{13}$$

where $\bar{\mathcal{F}}(\boldsymbol{x}, \boldsymbol{\lambda})$ is a subset of $\mathcal{F}(\boldsymbol{x}, \boldsymbol{\lambda})$ restricted to $\bar{S}$.

In each iteration, we add a subset of columns $s \in S \setminus \bar{S}$ which potentially improves the objective function of (9). To this end, we solve an auxiliary problem to find the most negative reduced cost column to add to the master problem. Thus, a column entering the basis can be found by computing the minimum reduced cost star with respect to the quadratic and linear costs of the edges inside the star.

According to the definition of the cost of stars C_s, the pricing subproblem can either be a linear problem or a BQP problem. In a simple setting, this subproblem can take the form of an unconstrained BQP (UBQP) problem. Nevertheless, it is possible that one must explicitly incorporate constraints in the binary quadratic subproblem in some specific applications. Given that the essence of solving a BQP problem exactly is NP-hard, intuitively adding columns with negative reduced cost without solving the subproblem to optimality can be a promising alternative when applying a CG algorithm. However, to provide a/some valid dual bound(s), we need to solve the subproblem to optimality (Aloise et al., 2010; De Fréminville et al., 2015). Specifically, in the case of a UBQP subproblem, several solution approaches based on greedy and heuristic methods are proposed to solve this problem (Kochenberger et al., 2014). Even in the case where the pricing is a constrained BQP problem and obtaining an exact solution is necessary, the size of the problem is much smaller than the compact formulation in Section 1, meaning the problems are relatively easy to solve. In addition, when the subproblem is a constrained BQP problem, we have better options in terms of linearization techniques such as RLT to solve it exactly. In the following section, we use examples to demonstrate how to deal with each of these cases.

Although the model presented in (7)-(8) is a generic reformulation of the BQP, special structure assumptions for the quadratic matrix $\boldsymbol{q}$ to have only adjacent quadratic costs offer promising properties for solving this type of problems. The third term of the objective function in (9) is therefore eliminated and the stars interact with each other only through linear costs. The new formulation for this particular class is an integer linear problem and consequently the constraints (11) and (12) are removed leading to a model which is more tractable to solve than the generic model. Noteworthy mentioning that the quadratic interaction between two adjacent edges may depend on their common endpoint, or may be independent of it, resulting in different sparsity levels of the matrix $\boldsymbol{q}$.

4. Illustrative Examples

The objective of this section is to demonstrate how the star-reformulation, the cost-splitting framework, and the solution strategy described in Sections 2 and 3 can be applied to different BQP problems. To this end, we consider three BQP problems whose formulations lead to different master problems and pricing subproblems, described as follows:

- Quadratic master problem and unconstrained BQP pricing subproblem
- Linear programming master problem and unconstrained BQP pricing subproblem
- Linear programming master problem and constrained BQP pricing subproblem

We consider the quadratic semi-assignment problem (QSAP) as an example to demonstrate the performance of the cost-splitting idea in addition to two problems from the adjacent-only class of

the BQP, the adjacent-only quadratic semi-assignment problem (AQSAP) and the multiple object tracking (MOT) problem, to outline the computational advantages of the star-reformulation in this class. In the following subsections, we provide a brief description and literature review for each problem, as well as the compact BQP models. Then, we describe how to reformulate them as (7) to (8) and obtain dual bounds using our CG solution method.

4.1. Quadratic Semi-Assignment Problem (QSAP)

In this problem, we are given a set of clients $N = \{1, ..., n\}$ and a set of servers $H = \{1, ..., h\}$. Suppose there is a linear cost $c_e, e = \{i, j\}$, associated with the assignment of client $i \in N$ to server $j \in H$ and there is a quadratic cost $q_{ef}, e = \{i, j\}, f = \{k, l\}$, associated with the assignment of client $i \in N$ to server $j \in H$ and client $k \in N$ to server $l \in H$ simultaneously. We transfer this problem to the previously-defined graph G, with the node set V consisting of n clients and h servers and the edge set $E = \{(i, j) | i \in N, j \in H\}$. By recalling the concepts of Section 3, we define the binary decision variable as x_e. Hence, denoting by variable x_e equals 1 if the edge e is chosen (the client i is assigned to server j), and 0 otherwise, the BQP problem of semi-assignment on graphs is formulated as follows:

$$\min \quad \sum_{e \in A} c_e x_e + \sum_{(e,f) \in \mathcal{E}} q_{ef} x_e x_f \tag{14}$$

$$\text{s.t.} \quad \sum_{e \in \delta(i)} x_e = 1 \qquad \forall i \in N \tag{15}$$

$$x_e \in \{0, 1\} \qquad \forall e \in E. \tag{16}$$

The QSAP has a variety of applications in the area of scheduling (Stone, 1977; Chrétienne, 1989) and partitioning (Hansen and Lih, 1992). The hub network design problem is also considered as a special case of the QSAP (Saito et al., 2009). The problem is known to be NP-hard, and solving it even for small size instances is very time-consuming (Sahni and Gonzalez, 1976; Magirou and Milis, 1989; Malucelli, 1996). Using RLT is a common approach in the literature to solve the QSAP and there are also some studies on polynomial algorithms, heuristics and lower bounding methods for special cases of the QSAP. We refer the reader to Saito et al. (2009) and the references therein for more details.

4.1.1. Reformulation. We reformulate the QSAP as the general model (7) to (8). To this end, without loss of generality, we assume that every server j is a center of a star-shaped sub-graph s. So, the binary variable λ_s corresponds to selecting this star. We define parameter $B_{js} \in \{0, 1\}$ to indicate if server j is the center of star s or not, the parameters $D_{is} \in \{0, 1\}$ to identify if client i is included in star s, and $D_{es} \in \{0, 1\}$ to denote whether edge e belongs to star s or not. The out-of-star interactions in the QSAP result in a quadratic reformulation, so based on the notation

given above, and according to the formulation (9)-(13), the linearized RMP can be expressed as follows:

$$\text{[RMP-QSAP]:} \quad \min \quad \sum_{s\in\bar{S}} C_s\lambda_s + \sum_{(e,f)\in\mathcal{E}\setminus\mathcal{A}} q_{ef}y_{ef} \tag{17}$$

$$\text{s.t.} \quad \sum_{s\in\bar{S}} B_{js}\lambda_s \leq 1 \qquad \forall j \in H \tag{18}$$

$$\sum_{s\in\bar{S}} D_{is}\lambda_s = 1 \qquad \forall i \in N \tag{19}$$

$$\sum_{s\in\bar{S}} D_{es}\lambda_s = x_e \qquad \forall e \in A \tag{20}$$

$$(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{P}(\boldsymbol{x},\boldsymbol{y}) \tag{21}$$

$$0 \leq x_e \leq 1 \qquad \forall e \in A \tag{22}$$

$$\boldsymbol{y} \in \mathbb{R}_+^{|(e,f)\in\mathcal{E}\setminus\mathcal{A}|} \tag{23}$$

$$\boldsymbol{\lambda} \in [0,1]^{|\bar{S}|} \tag{24}$$

where y_{ef} are the linearization variables. Constraints (18) impose that at most one star can be chosen among all the stars centered at j. Constraints (19) are the set partitioning linking constraints, which impose that each client must be included in exactly one star. Constraints (20) are the linking constraints, and enforce that if an edge is selected in an optimal solution, then it has to be included in only one selected star.

4.1.2. Column Generation. Starting from a subset of stars ($\{s|\lambda_s = 1\}$) which are feasible with respect to the constraints of the reformulation as initial columns, the algorithm solves the [RMP-QSAP] restricted to the current set of stars in each iteration. The corresponding dual solutions construct one pricing subproblem for each server $j \in H$, which aims to find a star related to j with the minimum reduced costs. In the next iteration of the algorithm, these stars are added to [RMP-QSAP] as the new columns to possibly improve the objective value of the master problem. The algorithm terminates when there are no more columns with a negative reduced cost to be added.

Let π_j, ρ_i and γ_{ij} be the dual solutions corresponding to constraints (18)-(20), respectively. For each server j we consider a set of incident edges to that server. Therefore, we can use them to rewrite the dual solutions $\gamma_{i,j}$ as γ_e, where $e = \{i,j\}$. In addition, ρ_i is not dependent on j, and is fixed for all star centers, so we use $\rho_e, e = \{i,j\}$ instead of ρ_i in our formulation to keep it consistent with the rest of the terms. The star with minimum reduced cost can be found by solving the following pricing subproblems on graphs, one for each server j:

$$\min \quad \sum_{e\in\delta(j)} (c_e - \rho_e - \gamma_e)z_e + \sum_{e,f\in\delta(j)} q_{ef}z_e z_f - \pi_j \tag{25}$$

$$\text{s.t.} \quad z_e \in \{0,1\} \qquad \forall e \in \delta(j) \tag{26}$$

where binary decision variable $z_e, e = \{i,j\}$, indicates if client i is part of the star centered at server j.

4.2. Adjacent-only Quadratic Semi-Assignment Problem (AQSAP)

In this section, we consider a special class of the QSAP in which the quadratic costs are restricted to the adjacent edges only. Consider the QSAP of assigning n clients to h servers, this time in a distributed processing system. If client $i \in N$ is assigned to server $j \in H$, the required processing time c_{ij}, is computed based on the processing speed of the server and the client's demand. In this type of problem, where multiple clients are assigned to the same server, there is no predefined priority and the order of processing the requirements is unknown. In this situation, every client $i \in N$ aims to minimize the worst-case completion time. Hence, the completion time of client i is set to $CT_i = \sum_{j \in H} x_{ij} \sum_{k \in N} c_{kj} x_{ij} x_{kj}$, where the binary variable $x_{ij} = 1$ indicates assigning client i to server j (Drwal, 2014). The goal in this problem is obtaining an assignment to minimize the total completion time for all clients, $\sum_{i \in N} CT_i$. According to the definitions presented in Section 4.1 and considering edge $e = \{i,j\}$ and $f = \{k,j\}$, the objective function of this problem on the graph G is:

$$\min \quad \sum_{e \in A} c_e x_e + \sum_{(e,f) \in \mathcal{A}} q_{ef} x_e x_f \tag{27}$$

where the constraints of the problem are the same as (15) and (16) and the equation below holds:

$$q_{ef} = c_e + c_f. \qquad \forall (e,f) \in \mathcal{A} \tag{28}$$

This formulation is valid for all assignment problems in which multiple clients compete for a single machine and each assigned client has to undergo the completion time of the machine. This problem includes a large class of the QSAP, while the structure of its quadratic matrix narrows it down to the class of problems described in Section 2. We denote this problem by adjacent-only QSAP (AQSAP) in the rest of the paper. Note that the explanations provided here are based on an application of AQSAP for clarity. Nevertheless, AQSAP is a general problem and can be applied in various applications such as specialized variants of scheduling and selfish resource allocation on the Internet. We observe two fundamental properties of the current problem: (i) similar to the QSAP, the linear costs for the edges which are not covered by stars are zero, and (ii) the non-adjacent edges do not interact with each other. Therefore, there are no out-of-star interactions between edges in the AQSAP, which in turn leads to the following reformulation:

$$\min \quad \sum_{s \in S} C_s \lambda_s \tag{29}$$

$$\text{s.t.} \quad \sum_{s\in S} B_{js}\lambda_s \leq 1 \qquad \forall j \in H \tag{30}$$

$$\sum_{s\in S} D_{is}\lambda_s = 1 \qquad \forall i \in N \tag{31}$$

$$\boldsymbol{\lambda} \in [0,1]^{|S|} \tag{32}$$

In this case, the reformulation is linear and CG is directly applicable. In order to solve the reformulation with CG, the pricing subproblem on graphs for every star center j can be written as:

$$\min \quad \sum_{e\in\delta(j)} (c_e - \rho_e)z_e + \sum_{e,f\in\delta(j)} q_{ef}z_e z_f - \pi_j \tag{33}$$

$$\text{s.t.} \quad z_e \in \{0,1\} \qquad \forall e \in \delta(j) \tag{34}$$

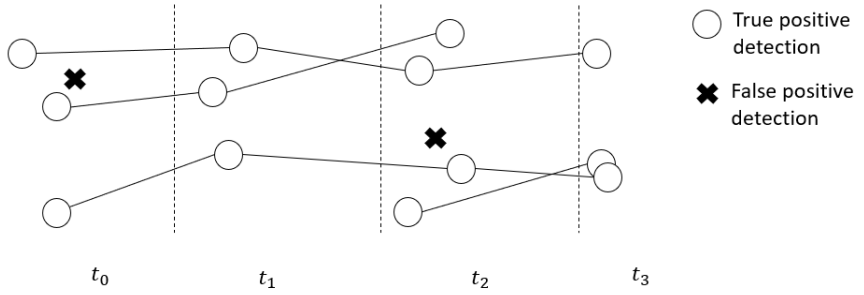
where $\pi_j, j \in H$, and $\rho_i, i \in N$ are the optimal duals associated with constraints (30), and (31), respectively, and we transfer them on the edges incident to j. Similar to the QSAP, here we end up with a UBQP pricing subproblem. Implementation details of the CG approach, and extensive computational experiments to find dual bounds, are provided in Section 5.

4.3. Multiple Object Tracking

Multiple object tracking (MOT) and, more specifically, multiple people tracking is a well-known application in computer vision that aims to track multiple objects (people) in a sequence of video frames. MOT is associated with a variety of applications like self-driving cars, human-computer interaction, security and video surveillance, sports analysis, some games like Microsoft Kinect, traffic analysis, etc. (Shen et al., 2018; Emami et al., 2018). Despite recent developments in this area, it is still a very challenging task due to occlusion and scene cluttering. As a result of the advancement of object detection technologies, detection-based methods are the most dominant techniques in MOT (Tang et al., 2017; Shen et al., 2018). MOT consists of three main components: (i) detecting the objects, in which a person detector is utilized for each individual frame to find the potential locations of all the people, (ii) affinity, or score estimation, which demonstrates how likely detections are related to a single identity, and (iii) data association, in which these hypotheses are linked across the frames based on the estimated scores to form tracks (Henschel et al., 2018).

In general, while object detection and score determination are deep learning tasks, data association is a combinatorial optimization problem. Once the detections and their unary and pair-wise scores are computed, they are given as inputs to the data association problem to generate the associated tracks. In the context of MOT, a sequence of frames (from t_0 to t_3) of a scene containing a few objects (such as people) is depicted in Figure 2. These potential objects are identified as a priori through the use of a machine-learning-based object detection method which provides a list

Figure 2 **Graphical representation of a solution of the data association in MOT applied to detections of four video frames**



of potential objects and the likelihood scores associated with them. This list and the detection scores are then used as the inputs in the optimization model. The detected objects are represented by circles and crossmarks in this figure where the detection scores produced by the detection algorithm are included as a part of the linear and quadratic costs in the MOT model. This figure represents the results of the data association step. In this step, an optimization model is solved to determine the set of detections that belong to the same object across multiple frames based on the cost parameters. The figure shows that, after solving the optimization model, the solution indicates four tracks (people) which are depicted by connecting lines in the figure and two false detections, represented by crossmarks. As shown in Figure 2, two of the objects are captured in all four frames while one of the objects moves through frames t_0 to t_2 and the other object only moves from frame t_2 to the last frame. Data association algorithms can be categorized as online or offline algorithms (Emami et al., 2018). Here we consider the offline data association case.

Essentially, a data association problem can be modeled with respect to a graph $G = (V, E)$. A detection $i \in N$ is represented by a node in this graph. We consider another set of nodes $H = \{1, 2, \ldots, h\}$ which are dummy nodes related to the tracks (target people). h is an upper bound on the number of target people in the video, which is predefined as an input. More precisely, we define the graph where the vertex set V consists of all detections, i.e., potential bounding boxes of potential candidates of people N in a video sequence and all possible tracks (target people) H. Similar to Section 2, consider A as a subset of edges E which are incident in a node in H. Therefore, edge $e = \{i, j\} \in A$ denotes a possible linking of a detection to a track (person).

Given $T = \{1, 2, \ldots, \mathcal{T}\}$ as the set of all frames, each detection i belongs to a frame $t \in T$. We introduce two other subsets of edges based on our definitions in Section 2.1: $\delta(i) \subseteq A$ is a subset of edges in A incident to node i and $\delta^t(i)$ is a subset of $\delta(i)$ when the edges stem from the frame t.

Each edge $e = \{i, j\} \in A$ has a cost $c_e \in \mathbb{R}$ defined via a logit function and reflects the likelihood of detection $i \in N$ being a correct detection. This cost is called unary cost in the computer vision literature (Henschel et al., 2018). Unary cost is fixed for each detection i and is not dependent on

tracks. For every pair of edges (e, f) which are incident in a node in H, a pair-wise cost $q_{ef} \in \mathbb{R}^{m \times m}$ is to be paid. Note that the interaction between edges e and f is non-zero if, and only if, the detections $i \in N$ and $k \in N$ are assigned to a distinct track. Pair-wise cost identifies how likely two detections belong to the same person. The probabilities which determine the costs are inferred based on detection scores. There are several ways to estimate these score terms from the geometry features, color histogram, appearance, and other features related to the image data. Considering p as this probability, the quadratic cost is computed as $log(1-p)/p$, so depending on the unary and pair-wise probabilities the costs can be negative, positive or zero, resulting in non-convexity of the problem (Dehghan and Shah, 2017; Henschel et al., 2018).

Similar to the model presented in Henschel et al. (2018), the MOT data association problem to minimize the total cost of labeling is expressed as:

$$\min \quad \sum_{e \in A} c_e x_e + \sum_{(e,f) \in \mathcal{A}} q_{ef} x_e x_f \tag{35}$$

$$\text{s.t.} \quad \sum_{e \in \delta(i)} x_e \leq 1 \qquad \forall i \in N \tag{36}$$

$$\sum_{e \in \delta^t(j)} x_e \leq 1 \qquad \forall j \in H, \quad \forall t \in T \tag{37}$$

$$x_e \in \{0, 1\} \qquad \forall e \in A \tag{38}$$

where constraints (36) are needed to mandate that more than one track assignment is not possible for every detection i. Constraints (37) restrict the model to select at most one detection associated with each track inside every frame.

Although the BQP model from Henschel et al. (2018) only contains constraints (36), different types of potential side constraints can be added to this primary model. These constraints can be formulated based on prior knowledge or hypotheses associated with the scenes that video frames come from, the types of cameras taping the frames, or the features of the object detector as well as other constraints based on the prior knowledge with respect to the objects in the scenes (Assari et al., 2016; Dehghan and Shah, 2017). The goal here is to input possible additional knowledge associated with the scenes, detections, and other available information as the constraints to the model. Thus, the obtained tracks will be more accurate with respect to the ground truth. In this study, based on the assumption of using only one detector (body detector) and having one detection per person as inputs to the problem, we append the frame constraints in (37) to the basis model from Henschel et al. (2018), guaranteeing that no two detections inside a frame are associated with the same person.

There are a few works in the literature on object tracking which are related to our modeling and solution method. Leal-Taixe et al. (2012) study the problem of tracking multiple objects across

multiple cameras. Their LP minimum cost flow formulation of the problem has block structural properties and they explore the results using a branch-and-price algorithm. Wang et al. (2017) model the MOT problem through an ILP and suggest using CG for MOT and solving the associated pricing subproblem with dynamic programming. They consider a pool of constructed tracklets (short tracks along with a few frames) as input for their model and choose an optimal among them based on DP. However, in this study, we deal with the quadratic interactions between many detections simultaneously.

4.3.1. Reformulation. To reformulate the problem as the general star-based model, (7) to (8), we identify each track j as the center of a possible star s. According to the definition of the problem, the pair-wise costs of MOT correspond only to the pairs of edges associated with the same person, meaning that if the edges are adjacent in H, their corresponding quadratic cost is non-zero and otherwise it is zero. Therefore, we can exploit the special structure of the adjacent-only class in this problem. Moreover, since the cost c_e is zero for the edges that are not incident to nodes in H, the objective function of our reformulation is reduced to the cost of stars. Thus, the star-based reformulation of MOT is given below:

$$\text{[RMP-MOT]:} \quad \min \quad \sum_{s\in\bar{S}} C_s\lambda_s \tag{39}$$

$$\text{s.t.} \quad \sum_{s\in\bar{S}} \lambda_s \le h \tag{40}$$

$$\sum_{s\in\bar{S}} D_{is}\lambda_s \le 1 \qquad \forall i \in N \tag{41}$$

$$\boldsymbol{\lambda} \in [0,1]^{|\bar{S}|} \tag{42}$$

The star-based model for LP relaxation of MOT consists of one star-only constraint (40) enforcing the maximum number of tracks, and one set of coupling constraints (41) to impose labeling every detection with at most one track.

4.3.2. Column Generation. The CG process starts from an empty set of feasible stars where we solve a pricing subproblem for each person $j \in H$ as a star center. Let π and $\rho_i, i \in N$, be the optimal solutions of dual variables associated with constraints (40) and (41), respectively and by transferring the definition of ρ_i to the edge $e = \{i, j\}$, we rewrite it as ρ_e. We further define binary variable $z_e = 1$ when edge e is selected in the star and $z_e = 0$ otherwise. Given this, the pricing subproblem corresponding to center $j \in H$ is as follows:

$$\min \quad \sum_{e\in\delta(j)} (c_e - \rho_e)z_e + \sum_{e,f\in\delta(j)} q_{ef} z_e z_f - \pi \tag{43}$$

$$\text{s.t.} \quad \sum_{e\in\delta^t(j)} z_e \le 1 \qquad \forall t \in T \tag{44}$$

$$z_e \in \{0,1\} \qquad \forall e \in \delta(j) \tag{45}$$

where constraint (44) restricts each star to select a maximum of one detection per frame.

Note that this CG process requires only one subproblem to be solved at each iteration. This is due to the fact that neither linear nor quadratic costs are dependent on the star centers; the centers can be realized identically. More specifically, suppose that $j, l \in H$ are possible star centers (tracks) and $i, k \in V \setminus H$ are the detections. The quadratic interaction between $e_1 = \{i, j\}$ and $f_1 = \{k, j\}$, $q_{e_1 f_1}$ is equal to the quadratic interaction between $e_2 = \{i, l\}$ and $f_2 = \{k, l\}$, $q_{e_2 f_2}$.

We remark that the pricing subproblem in this case is a constrained BQP problem. The implementation details are provided in Section 5.

5. Computational Experiments

In this section, we provide a rigorous experimental study to evaluate our suggested framework on test instances of the quadratic semi-assignment, the adjacent-only variant of the QSAP and MOT in terms of dual bound and computing time. We attempt to answer these fundamental questions through our experiments: How effective and applicable is the star-reformulation on other adjacent-only problems like the AQSAP and MOT? To what extent employing the cost-splitting framework is worthwhile for a BQP problem like the QSAP? And for each of the selected problems, which type of formulation performs better?

To answer these questions, we compare the star-reformulation, the cost-splitting and the CG framework with both standard linearization technique (SLT) and RLT as two most commonly used methods in the literature of BQP. For the QSAP and the AQSAP, we also compare our results with the most effective exact method proposed (Rostami et al., 2022). GUROBI version 9.0.1 is chosen as our benchmark mixed-integer programming (MIP) solver and we also solve the BQP models directly using GUROBI. Note that, through the CG procedure, we consider various possible reformulations which result in different pricing subproblems (i.e., BQP and different linearizations). This leads to different reformulations and solution strategies to solve the original BQP models which can be implemented and solved by a commercial solver like GUROBI. We consider a time limit of 3 hours to solve each instance.

For the sake of comparison, the root node dual bound of the RMP is selected as a reliable indicator to be compared with the solver dual bound. Yet, the dual bound is not the only measure of efficiency; we single out the computation time needed to complete the CG process in the root node and for GUROBI to solve the problem.

We compare the trade-off between computation time and the quality of the dual bound using the methodology of performance profiles (Dolan and Moré, 2002; Bergner et al., 2015) in the related

section of each problem. The profile plot shows the cumulative distribution function (CDF) of the ratio of the time taken by each method to solve each problem or the bound obtained by each method to the best acquired time or bound among all the approaches for that problem. These two criteria are described below in more detail:

- Dual bound performance profile: The first set of graphs is based on the dual bound quality regardless of the time required to compute that bound. For each instance and each method, we compute the ratio between the dual bound of each method and the best bound among them. The horizontal axis reports this ratio, thus the vertical axis corresponds to the fraction of instances with at least this ratio of bound performance displayed for each method. A large value is considered for the ratio where the method could not provide any dual bound for an instance within the time limit.

- Time performance profile: The second type of graphs is generated based on the time needed to obtain the best dual bound. In this analysis, for each instance, we first find the best dual bound among all the obtained dual bounds by different methods. Then, for every method which yielded the best dual bound, we consider the ratio between the time required by that method to attain the best dual bound and the shortest time among all of them. This ratio provides the performance index in the horizontal axis. We report the fraction of instances with a maximum of a specific time ratio in the vertical axis. A large value is assigned to the ratio where the method is not able to achieve the best dual bound for an instance. Before moving on, let us make the following remarks:

Remark 1. For each method, the probability that the method will win over the rest of the methods is defined by the fraction of instances with the best performance (Dolan and Moré, 2002). Hence, we refer to the ratio of instances with the performance equal to one, as the number of wins of the associated method.

Remark 2. When running the CG algorithm, we do not have to wait until the termination of the CG procedure to obtain a valid lower bound. If the CG procedure could not converge in our desired time, we have information about the intermediate quality of the dual bound in each iteration at the expense of slightly more computations (Lübbecke and Desrosiers, 2005). Nonetheless, to attain an exact dual bound, we must solve the pricing of that iteration to optimality.

Remark 3. Primal bounding methodologies are beyond the scope of the current study, thus we do not embed the CG procedure in a branch-and-bound tree. However, we compute primal bounds by applying a trivial heuristic to the solutions of the CG for each instance of the problem. In this heuristic, we solve the IP model for the master problem of the last iteration where all available columns are considered as binary variables. In Appendix C, we discuss the obtained upper bounds in more detail.

REMARK 4. The reported computation time for the experiments is comprised of the time to attain both LB and UB.

REMARK 5. It is well-accepted that difficulties in proving optimality may appear when column generation is solving a degenerate, large-scale problem. In addition, dual variables may oscillate from a good one to a much worse one, deriving the same value for many iterations of the CG (Amor et al., 2004; Desaulniers et al., 2006). Computational experiments show that it is possible to alleviate these effects using stabilization techniques such as dual-optimal inequalities and stabilized column generation algorithms. In this study, we implemented BoxPen, in-out separation, and interior-point stabilization techniques (Du Merle et al., 1999; Ben-Ameur and Neto, 2007; Rousseau et al., 2007). Since these methods need tuning of several parameters and our main focus is on the application of the standard CG algorithm, further specialized CG enhancements and tuning procedures for specific problem structures were not thoroughly investigated. Our computational results demonstrate improvements in some CG experiments, but not for all of them. Thus, we have decided to keep the unstabilized results in our reports. However, one can test different techniques to accelerate the CG algorithm to solve the proposed star-reformulation and cost-splitting in an arbitrary application.

All the algorithms and models were implemented in the Python programming language. They were performed on a shared cluster with a four-core, 3.05GHz processor and 128GB RAM running under Linux 7.8. In the following sections, we present the test instances, parameter settings and computational results for the data association in MOT and both versions of the QSAP. The instance-by-instance tables in Appendix B provide more details on the results of each problem.

5.1. QSAP and Adjacent-only QSAP Experiments

Instances in the literature that can be used for the QSAP experiments are very limited and many papers rely on data instances that were randomly generated (Silva et al., 2021; Rostami et al., 2022). More importantly, since our main objective was to investigate the instances with varying cost structures, we generate random instances for QSAP, introduced in Section 4.1, where the processing cost c_{ij} for assigning a client i to a machine j is computed as $dm_i \times pr_j$. The client unit of demand for processing i is identified by dm_i, while pr_j indicates the required time for processing a unit in machine j. They are randomly generated over $(0, 100)$ and $(0, 10)$, respectively, from uniform distribution. We carry out our experiments considering different combinations of parameters $n \leq 50$ and $h \leq 14$, since in practice h is smaller than n. To investigate the impacts of the sparsity and structure of the quadratic matrix on the performance of the cost-splitting framework, we run the experiments on 5 randomly-generated data sets with different sparsity of the quadratic matrices, each comprised of 41 instances. We embark on our experiments, using an adjacent-only QSAP data

set and adding out-of-star interactions incrementally to observe the effects. Indeed, the quadratic matrix of the first problem consists of in-star interactions only, while the next problems include 10%, 15%, 20%, and 25% out-of-star interactions, respectively, in addition to the in-star interactions. We should take into account that, in the real-life applications of BQP, the quadratic matrix is mostly very sparse (Furini et al., 2019). Therefore, adding just 10% out-of-star interactions on top of the in-star interactions results in a fairly dense quadratic matrix. Data generated by altering parameters n and h are defined in the QSAP-associated tables in Appendix B.

As mentioned in Section 4.1, the pricing subproblems (25) and (33) are unconstrained BQP problems. In order to heuristically solve these subproblems, we employ an open-source solver, qbsolv (Booth et al., 2017). Based on divide and conquer and dynamic programming, the solver partitions the problem into multiple subproblems and solves them using a tabu search algorithm. When the heuristic solver fails to find an improving column to add, we switch to an iteration of an exact method. In addition, as mentioned before, to retrieve information on the intermediate dual bounds, either the pricing has to be solved to optimality or a valid lower bound on the optimal reduced cost must be available. Hence, we apply a hybrid strategy to solve the pricing subproblem in which, after calling qbsolv for a fixed number of CG iterations, it switches to an exact method (branch-and-bound) for one iteration. In the exact iteration, either the BQP formulation of the subproblem or the linearized version is solved by GUROBI. We chose the standard linearization technique to construct the set $\mathcal{P}(x, y)$ in constraints (21). The concept of the standard linearization is presented in Appendix A.

As mentioned earlier, the RLT is, in general, the most effective linearization approach to provide tight bounds for the QSAP in the literature (Billionnet and Elloumi, 2001; Schüle et al., 2009). Therefore, we compare the results of our reformulation with the results of GUROBI applied to the RLT model of the QSAP as well as the SLT model of the QSAP. To investigate the effectiveness of the proposed reformulation, we also implement the reformulation and outer approximation suggested in Rostami et al. (2022) for a class of BQP problems including the quadratic semi-assignment and report the results on the same instances. A brief description of the different reformulations and methods used to solve instances of the QSAP and the AQSAP are provided here.

BQP: The BQP model (14)-(16) solved by GUROBI.

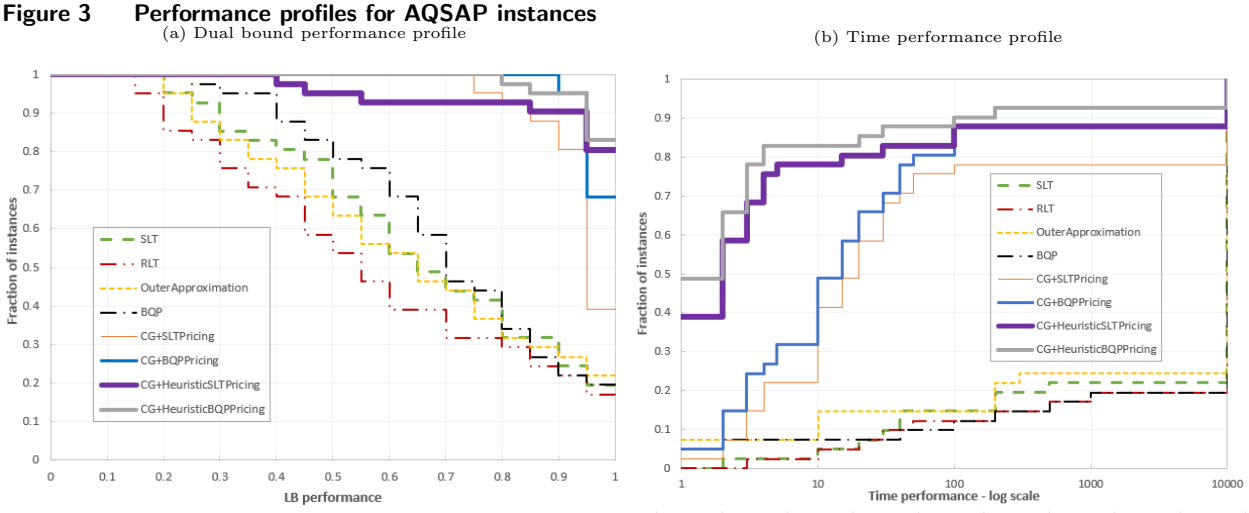
SLT: Linearized reformulation (using SLT) of the BQP model (14)-(16) solved by GUROBI.

RLT: Linearized reformulation (using RLT) of the BQP model (14)-(16) solved by GUROBI.

OuterApproximation: Convex reformulation and outer approximation (Rostami et al., 2022) applied to the BQP model (14)-(16) solved by GUROBI.

CG+BQPPricing: CG algorithm for the model (17)-(23) where the UBQP pricing is solved by GUROBI

Figure 3 **Performance profiles for AQSAP instances**



CG+SLTPricing: CG algorithm for model (17)-(23) where the standard linearization of the pricing subproblem is solved by GUROBI.

CG+HeuristicBQPPricing: CG algorithm for model (17)-(23) where the UBQP pricing is solved by using the hybrid heuristic method described above.

CG+HeuristicSLTPricing: CG algorithm for model (17)-(23) where the UBQP pricing is solved using the hybrid heuristic in which the exact iteration solves the standard linearized pricing subproblem by GUROBI.

Note that, although we use GUROBI to solve all the obtained models in different methods, in the rest of the work, we only refer to SLT, BQP and RLT as GUROBI methods for the sake of simplicity.

Figures 3 to 7 show the results of all methods in terms of performance profile for both the AQSAP and the QSAP. In each figure, the left diagram compares all GUROBI, CG and the OuterApproximation methods in terms of dual bound performance, while the right one gives the time performance comparison of the methods. According to the description provided for the dual bound performance profile, the dual bound ratio is between zero and one for this application. However, the diagrams on the right-hand side corresponding to time performance always consist of performance ratios that are greater than or equal to 1. Clearly, for both bound and time performance profiles, a method with a larger fraction of instances with a ratio closer to 1 is preferable.

AQSAP results. The results for the AQSAP instances are given in Figure 3. According to the LB performance analysis in Figure 3a, CG hybrid methods have the most wins in terms of providing the best LB (83% and 80% for the CG+HeuristicBQPPricing and the CG+HEURISTICBQPPricing, respectively) among all the methods, though the CG+BQPPricing outperforms the hybrid methods in a few quantiles. For instance, all the instances are solved by

the CG+BQPPricing to 90% of the best LB, while only 95% of them could reach this ratio when solved by the CG+HeuristicPricing. We also observe that in general, GUROBI solves the BQP model (BQP) slightly better than it solves the linearization models (SLT and RLT). If we seek a method that can achieve at least 20% of the best LB, then all of the tested methods achieve this. However, if we increase the requirement to 40%, we can observe that all the CG methods perform better than the non-CG GUROBI methods and the OuterApproximation method. Finally, looking for a method with 100% performance, the CG+HeuristicBQPPricing is the best choice.

The next analyses are related to computing time. The time performance profile in Figure 3b shows that the best time to find the best LB for almost 50% of instances belongs to the CG+HeuristicBQPPricing method. It also demonstrates the superiority of this method in all other quantiles. Comparing all the methods together, the GUROBI methods and the OuterApproximation method are inefficient at finding the best LBs, where the best of them in terms of time (the OuterApproximation method) provides the best time in only 7% of the cases. As another example to illustrate the superiority of CG methods, consider that using the SLT method, only 15% of the instances and using the BQP, only 12% of instances could converge to the best LB within 2 orders of magnitude of the best time.

Looking at both graphs simultaneously confirms the superiority of the CG-based methods over the GUROBI and the OuterApproximation methods, with a large gap for almost all intervals. Likewise, comparing the different methods of CG indicates that, as theory suggests, combining heuristics and exact methods to solve the pricing problems improves the results both in terms of computing time and LB for most of the instances.

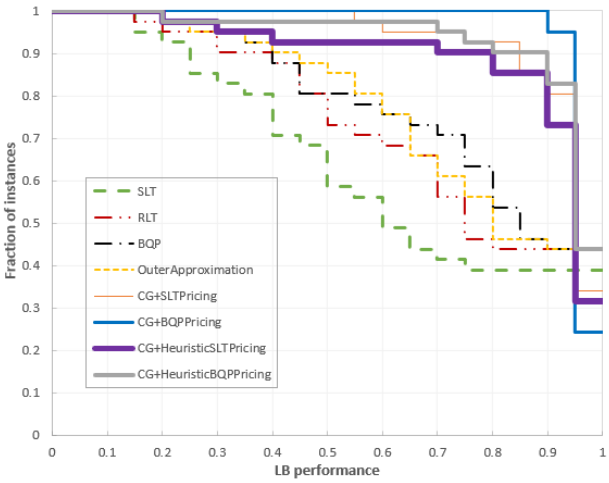
QSAP results. The results for the QSAP instances are shown in Figures 4 and 7 where to increase the density of the quadratic matrix, we add more out-of-star interactions to the AQSAP problem each time to generate our data sets.

Considering the graph 4a, where we have only 10% out-of-star interactions (quadratic costs), the overall interpretation of the results is very similar to adjacent-only QSAP. The CG+HeuristicBQPPricing has the greatest number of instances with the best LB among all the methods (44%); however, this time the CG+BQPPricing outperforms the hybrid counterpart in more intervals compared to the AQSAP. The performance of the GUROBI methods and the Outer-Approximation is almost the same as before. The BQP and the OuterApproximation methods are superior to the other GUROBI methods while they still have a huge LB performance gap compared to all the CG methods.

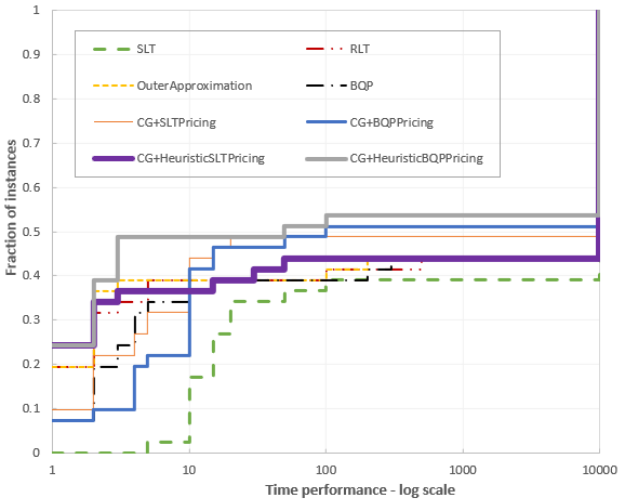
Nevertheless, in the next LB performance profile represented in Figure 5a, all GUROBI methods and the OuterApproximation outperform CGs in the interval $[0.95, 1]$. Comparing the CG methods

Figure 4 **QSAP**-10% **out-of-star density**

(a) Dual bound performance profile



(b) Time performance profile

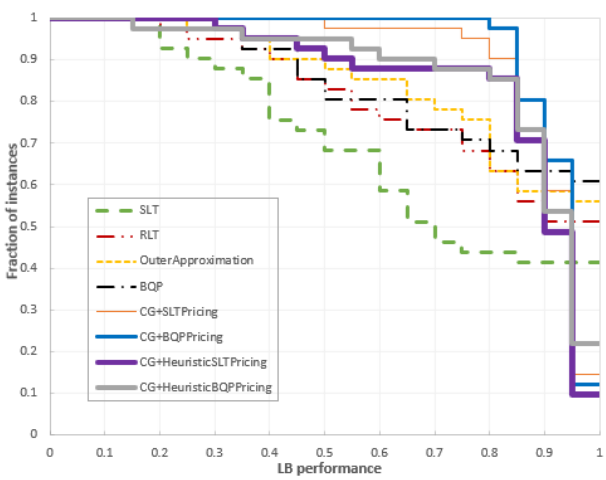


with one another in this figure demonstrates that the CG methods with exact pricing generally outperform heuristic pricing methods. Considering this figure and the next LB performance figures, 6a and 7a, the number of wins for the GUROBI and the OuterApproximation methods is more than for CG methods. Indeed, adding more out-of-star interactions incrementally to the data sets in Figures 6a and 7a results in enhancing the performance of the GUROBIs and the OuterApproximation method. However, even for the densest quadratic matrix in Figure 7a, the CG methods still outperform non-CG methods in some quantiles. For example, we can observe the superiority of both CG+BQPPricing and CG+SLTPricing to GUROBI methods when a ratio smaller than 70% is considered and to the OuterApproximation method when the ratio is smaller than 60%. An overview of LB performance profiles for all of the data sets in the QSAP demonstrates that CG methods are more robust because they can achieve satisfactory LB performance for most of the instances. While if we need higher LB ratios, we aim to use GUROBI and the OuterApproximation. For instance, in the case of 20% out-of-star density and for a minimum requirement of 60% LB performance, we would opt for the CG+BQPPricing as all the instances reach this ratio of the best bound when they are solved using this method. In contrast, the best non-CG method, the Outer-Approximation, satisfies this requirement in only 90% of the problem instances. Assessing non-CG methods, the plots suggest increasing the performance of both RLT and the OuterApproximation compared to BQP and SLT by adding more out-of-star costs, to an extent that they outperform BQP and SLT in most of the quantiles in Figure 7a.

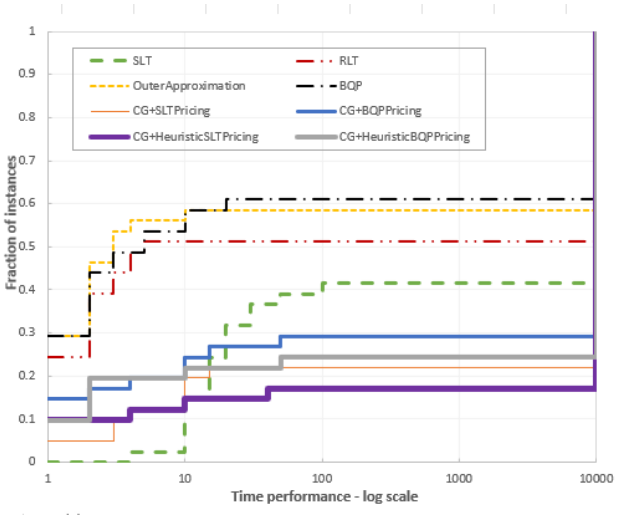
The second analysis is related to the time performance profiles. As shown in Figure 3b, all CG methods perform better than GUROBI methods in all quantiles when it comes to AQSAP. Starting from the plot associated with time performance in Figure 4, when we add out-of-star interactions,

Figure 5 **QSAP**-15% **out-of-star density**

(a) Dual bound performance profile



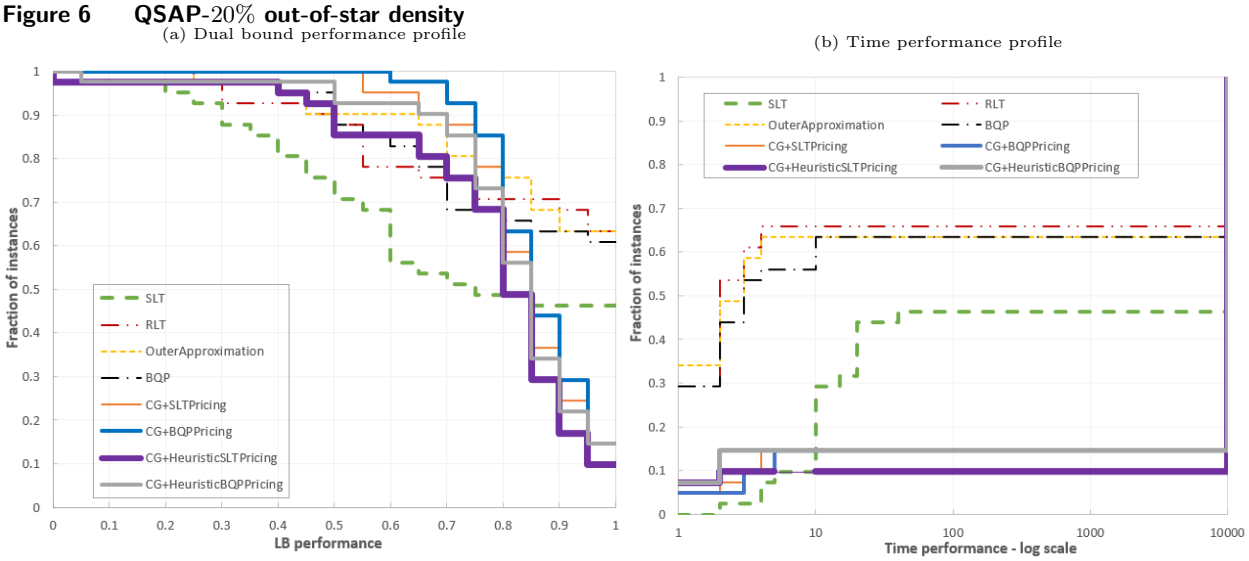
(b) Time performance profile



the GUROBI methods and the OuterApproximation move upside of the figure and get closer to the CG methods. This is due to the fact that the speed of the GUROBI and the OuterApproximation methods increases when the instances consist of some out-of-star interactions in addition to the in-star interactions. However, still, in Figure 4b, most wins belong to CG methods. Starting from Figure 5b, SLT, RLT, and BQP methods outperform all CG methods in all quantiles. Comparing non-CG methods together, similar to the LB analysis, the OuterApproximation outperforms the GUROBI methods in terms of time and it is superior even when we add more out-of-star costs.

It is interesting to note that, based on the last vertical lines of Figure 4b, around 47% of the instances could not obtain the best LB when they are solved by the best CG method (CG+HeuristicBQPPricing), while this number is 57% for the best GUROBI method and the OuterApproximation method. Although inefficiency at finding the best LB increases for CG methods by adding more out-of-star costs, GUROBI methods can reach the best lower bound for more instances, in denser matrices, due to their speed improvement. Regarding the vertical lines for the BQP method in Figures 5b and 6b, our interpretation is that this method could not obtain the best LB in the time limit for more than 35% of instances, whereas it obtained the best LB in the best time for the other 65% of instances. Moreover, it is observed that as often seen in the literature, CG methods with exact pricing are slower than their hybrid counterparts for all instances of adjacent-only problems. Nonetheless, the advantages of using the hybrid methods are weakened for more dense quadratic matrices. As an example, Figure 7b shows that the CG+HeuristicBQPPricing method outperforms the CG+BQPPricing in only 4% of the cases in terms of time performance. Looking at the trend in all of the time performance profiles in Figures 4b to 7b, we observe that the GUROBI methods and the OuterApproximation method generally outperform CG methods

Figure 6 **QSAP-20% out-of-star density**

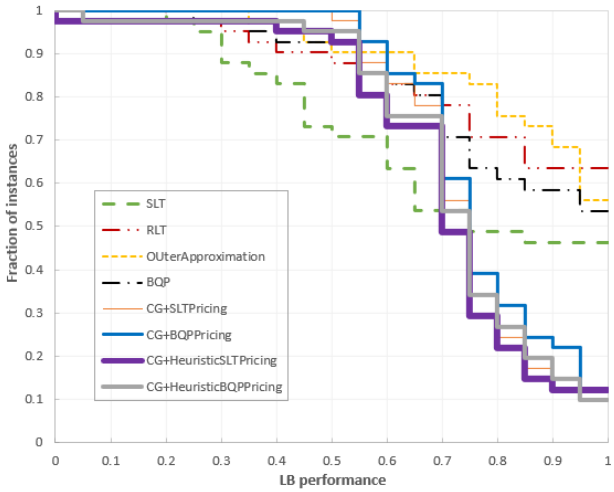


when the quadratic cost matrix of the QSAP is more dense. For instance, in Figure 7b where we have 25% out-of-star quadratic costs, in approximately 63% of the instances, the best GUROBI method (RLT) is within one order of magnitude with respect to the best time. Nevertheless, when we solve the instances using the best CG method (CG+BQPPricing), only around 18% of the cases are within this order. However, we should remark that, in our time performance analysis, we do not reflect the computing time of the instances which could not obtain the best LB. In an extreme example, suppose that CG stopped in a few seconds with a high ratio of the best LB while GUROBI found the best LB in 3 hours. In this situation, since CG did not yield the best LB, we assign a very large number (10000) to its time performance. If the GUROBI method is the only one that obtains the best LB, we consider its corresponding time performance equal to 1.

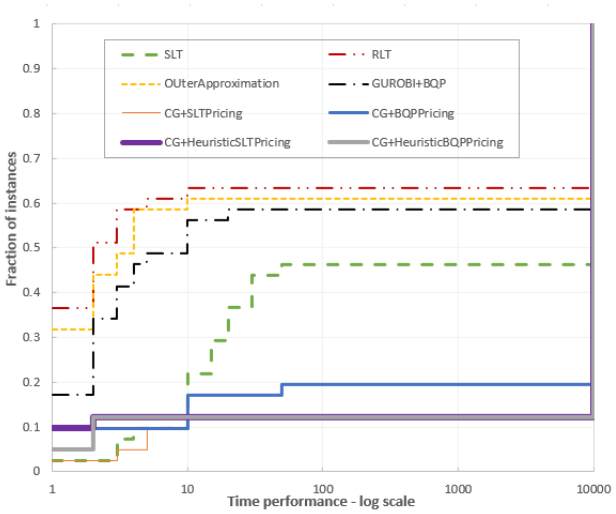
Considering both performance profile analyses and the instance-by-instance details of Appendix B, it can be deduced that the star-based reformulation and CG, even with a basic implementation, outperform GUROBI in terms of computing time and LB for all the instances of the AQSAP. Additionally, in a large number of the instances, given a heuristic solution for the optimal value, optimality can be proved in the root node for this problem. By evaluating the general QSAP, the performance of the cost-splitting framework is reduced by augmenting the out-of-star quadratic matrix and it might not be highly effective when the matrix is fairly dense. It should be noted that, since the majority of real applications of the BQP consist of sparse quadratic matrices (Furini et al., 2019), 10% to 25% out-of-star interactions in addition to in-star interactions provides the proper condition to investigate the effects of having dense matrices in our proposed reformulation. One notable remark is that, although the performance of the CG methods appears to deteriorate when solving the instances with more out-of-star costs, their performance to solve larger instances is still generally good. The results shown in instance-by-instance tables in Appendix B suggest that for the

Figure 7 **QSAP-25% out-of-star density**

(a) Dual bound performance profile



(b) Time performance profile



majority of large instances, the CG methods generally provide the best bounds when the time limit is reached. In this case, GUROBI and OuterApproximation method's dual bounds are relatively weak. In practice, we may choose CG methods over non-CG methods for even denser quadratic matrices. For example, when we have 15% out-of-star density, all the instances solved by the CG+BQPPricing provide an overall LB with the ratio of at least 80% of the best LB, while this ratio is 30% when using the best GUROBI method, BQP. Although GUROBI and OuterApproximation methods can solve small instances quite efficiently and obtain the best bounds, they still have a huge optimality gap when solving large instances after 3 hours. On the other hand, while CG methods are not able to converge to the best LB in many cases, they obtain a rather strong LB compared to the other methods in the majority of the instances which were not solved to optimality within the time limit.

5.2. MOT Experiments

For the MOT data association problem, we perform our tests based on a well-known benchmark, the MOT challenge datasets (Milan et al., 2016). Our tests are performed on different instances from the same video sequence (MOT16-09) of this benchmark and produce instances by altering the input parameters in the sequence. These parameters include the number of frames in the video ($\mathcal{T}$), the maximum number of tracks (h), and the maximum number of considered adjacent frames (d). Trivially, the estimated upper bound h has to be increased by enlarging the number of investigated data frames to avoid missing track of any person in a real case. However, one has to consider the effects of this parameter on the growth of the problem size in the pre-estimation. In addition, we set the quadratic cost of two nodes that are more than d frames apart to zero in our implementation. We alter this parameter to demonstrate its influence on enlarging the quadratic matrix and problem

size and generating various instances. The majority of the dataset configurations in this section are based on Henschel et al. (2018).

It is worth noting that to be able to compare two entire MOT algorithms, the detection of objects and the precision of estimating unary and pair-wise costs, which are normally obtained using deep learning techniques, are very crucial. However, since in the current paper , we aim to compare the proposed reformulation of data association and CG with the results of a MIP solver, computing real accurate costs is beyond the scope of our research. Therefore, although we explore the real detection of the MOT challenge dataset, we estimate naive unary and pair-wise costs for these detections based on some basic factors such as the distance between the detections. This method is exploited in several papers in data association. For example, Yarkony et al. (2020) assume that the costs are provided by learning methods and are given to their algorithms.

We should remark that each frame in the MOT16-09 sequence includes 15 to 25 detections; therefore, the number of decision variables in the represented formulation of (35) is between $15 \times h \times \mathcal{T}$ and $25 \times h \times \mathcal{T}$. To better realize the large scale of the problem, assume we investigate a data instance related to 10 frames of a video sequence consisting of an average of 20 detections in each frame and we aim to track a maximum number of 35 people in the sequence. This instance includes 7000 decision variables in the represented BQP formulation, which generates a graph of the problem with 235 nodes in total. We show 27 generated instances from the mentioned data set in Tables 11 and 12 of Appendix B.

Similar to the semi-assignment problem, here we evaluate the efficiency of the star-based reformulation of MOT compared with the solver. In Section 4.3, we discussed that the pricing subproblem of MOT is a constrained BQP problem. So, we can apply a naturally tighter linearization, RLT, to solve the subproblems in addition to previously-used standard linearization in the QSAP. In Appendix A, we clarify this linearization when it is applied to the MOT formulation. Here, the solution methods are briefly introduced:

BQP: The BQP model (35)-(38) solved by GUROBI.

SLT: Linearized reformulation (using SLT) of the BQP model (35)-(38) solved by GUROBI.

RLT: Linearized reformulation (using RLT) of the BQP model (35)-(38) solved by GUROBI.

CG+BQPPricing: CG algorithm for model (39)-(42) where the constrained BQP pricing is solved by GUROBI.

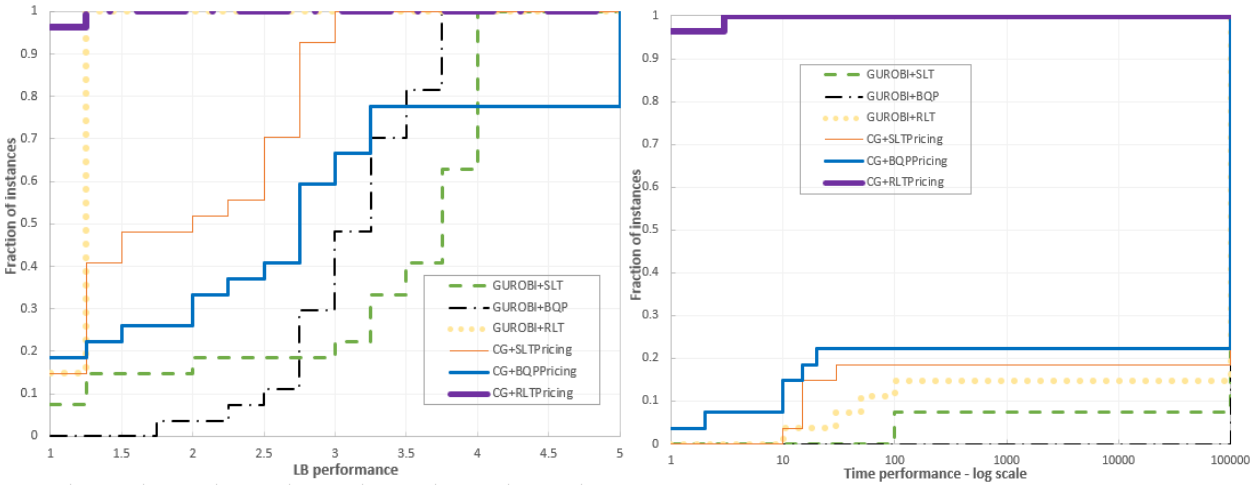
CG+SLTPricing: CG algorithm for the model (39)-(42) where the standard linearization of the pricing subproblem is solved by GUROBI.

CG+RLTPricing: CG algorithm for the model (39)-(42) where the linearized pricing subproblem using RLT is solved by GUROBI.

Figure 8 **Performance profiles for the MOT problem**

(a) Dual bound performance profile

(b) Time performance profile



As discussed in the previous section, we investigate the performance of the methods through two types of performance profiles. Moreover, we report the experimental details for each instance of the problem in instance-by-instance tables in Appendix B.

According to the LB performance plot in Figure 8a, all the CG-based methods outperform their GUROBI-based counterparts in almost all of the intervals. The only exception occurs when it comes to comparing the CG+BQPPricing method with the BQP method. The GUROBI method demonstrates superior performance in almost 20% of the instances. As we mentioned in the definition of the dual bound performance profile, we assign a large number for the performance ratio when the method fails to provide a valid dual bound for an instance within the time limit. Given that 5 is considered as a large ratio in our analyses, the figure indicates that in 22% of the instances, the CG+BQPPricing could not obtain an LB within the time limit. We identify the dual bound for these instances by NA in Table 12. Another remark related to the LB performance graph is that all three CG methods have a number of wins that is equal to, or greater than the best GUROBI method (RLT). Evidently, when we apply RLT to solve the pricing subproblem of the CG reformulation, the number of wins is the highest among all the methods with a large gap. Overall, the best method is the CG+RLTPricing, since it obtains the best LB for 96% of the instances and the worst LB outcome for this method for the rest of the instances is less than 1.25 times the best LB. It should be noted that the negative objective function of the data association problem in MOT is reflected in an LB performance ratio greater than 1.

Although according to Figure 8a the ratio of the obtained LB by the RLT to the best LB is at most 1.25, Figure 8b shows that its computational time is not competitive compared to CGs. More specifically, less than 20% of the test set is solved by the RLT within 2 orders of magnitude with respect to the fastest method. Hence, it still outperforms other GUROBI methods. The

performance profiles in Figure 8 delineate the superiority of the RLT method. Evidently, when the RLT is directly applied to the BQP formulation (the BQP+RLT method), it outperforms the other GUROBIs. Moreover, when it is used as the method for solving the pricing subproblem of CG (the CG+RLTPricing method), it has better performance than the other CG methods. Considering both LB and time performance, we observe that the CG+RLTPricing obtains not only the best LB in 96% of the cases but also in the shortest time for almost all of these cases. We detail the experiments on dual bound and time, as well as UB and the parameters of the instances, later in the appendices.

Given the instance-by-instance tables in Appendix B and the performance analyses, we can infer that the star-based reformulation and the CG methodology computationally outperform the GUROBI solver in obtaining LB for the data association problem in MOT. Moreover, similar to the CG+RLTPricing, which outperforms the RLT, the other CG methods outperform their GUROBI counterparts both in terms of LB and computational time. Evidently, the CG+RLTPricing achieves the best LB in nearly all the cases and, in the situation where it stops before the time limit, it converges to an optimal solution for the vast majority of instances.

6. Conclusion

In this study, we investigated the generalizability and performance of the star-reformulation on a large class of BQP problems, adjacent-only BQP problems. We employed the star-reformulation on two adjacent-only BQP problems with different characteristics: the quadratic semi-assignment problem and the multiple object tracking problem. Moreover, we developed a cost-splitting reformulation framework and solution methodology using column generation for general BQP problems. This framework, based on in-star and out-of-star interaction between pairs of edges in the BQP problem's graph, exploits the quadratic matrix structure. To evaluate the efficiency of our framework, we perform extensive experiments on the QSAP. We compare the lower bound and computing time of the reformulation and CG methods with a state-of-the-art MIP solver on instances of this problem as well as the AQSAP and MOT. According to the proposed framework, the adjacent-only class of problems inherits a special structure of the quadratic matrix which resulted in a huge improvement to solve these problems.

One notable outcome of this study is that, in the adjacent-only class of problems, a basic implementation of the presented framework can already compete with the solver, showing large improvements both in terms of dual bound and computation time. When out-of-stars quadratic costs are added to the problem incrementally, the potential of the framework to compete with the solver decreases. Nevertheless, it is interesting to note that even in the case of QSAP with a fairly dense out-of-star quadratic matrix, the cost-splitting and CG methods still obtain promising results for

some instances. Particularly in larger instances, where all the tested methods meet the time limit, CG methods outperform the MIP solver in many cases.

A possible future research direction is to explore the possibility of incorporating the proposed framework in a branch-and-bound tree to improve the primal bounds in addition to the dual bounds. Investigating this idea on other BQP problems with different constraint structures (such as the general case of quadratic minimum spanning tree) is another avenue for further research. Alternatively, exploring the star-reformulation idea on more problems in the adjacent-only BQP class such as the adjacent quadratic assignment problem is of interest for future studies. It would also be interesting to explore how effective the cost-splitting technique is on general BQP problems such as the quadratic assignment problem.

Acknowledgments

We wish to gratefully thank Mitacs Accelerate Program for providing funding for this project. In addition, the second author gratefully acknowledges the funding provided by the Canadian Natural Sciences and Engineering Research Council (NSERC) under the Discovery Grant RGPIN- 2020-05395. We also thank the associate editor and anonymous reviewers for their valuable comments.

References

Adams WP, Forrester RJ (2005) A simple recipe for concise mixed 0-1 linearizations. *Operations Research Letters* 33(1):55 – 61, ISSN 0167-6377, URL `http://www.sciencedirect.com/science/article/pii/S0167637704000562`.

Adams WP, Sherali HD (1990) Linearization strategies for a class of zero-one mixed integer programming problems. *Operations Research* 38(2):217–226.

Aloise D, Cafieri S, Caporossi G, Hansen P, Perron S, Liberti L (2010) Column generation algorithms for exact modularity maximization in networks. *Physical Review E* 82(4):046112.

Amor HB, Desrosiers J, Frangioni A (2004) *Stabilization in column generation* (Groupe d'études et de recherche en analyse des décisions).

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

Assari SM, Idrees H, Shah M (2016) Human re-identification in crowd videos using personal, social and environmental constraints. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 9906 LNCS:119–136, ISSN 16113349, URL `http://dx.doi.org/10.1007/978-3-319-46475-6{\_}8`.

Barahona F (1983) The max-cut problem on graphs not contractible to K5. *Operations Research Letters* 2(3):107–111.

Ben-Ameur W, Neto J (2007) Acceleration of cutting-plane and column generation algorithms: Applications to network design. *Networks: An International Journal* 49(1):3–17.

Bergner M, Caprara A, Ceselli A, Furini F, Lübbecke ME, Malaguti E, Traversi E (2015) Automatic Dantzig-Wolfe reformulation of mixed integer programs. *Mathematical Programming* 149(1-2):391–424.

Bettiol E, Bomze I, Létocart L, Rinaldi F, Traversi E (2022) Mining for diamonds—matrix generation algorithms for binary quadratically constrained quadratic problems. *Computers & Operations Research* 142:105735.

Billionnet A, Elloumi S (2001) Best reduction of the quadratic semi-assignment problem. *Discrete Applied Mathematics* 109(3):197–213.

Billionnet A, Elloumi S, Plateau MC (2009) Improving the performance of standard solvers for quadratic 0-1 programs by a tight convex reformulation: The qcr method. *Discrete Applied Mathematics* 157(6):1185–1197.

Billionnet A, Soutif É (2004) An exact method based on Lagrangian decomposition for the 0–1 quadratic knapsack problem. *European Journal of Operational Research* 157(3):565–575.

Bonizzoni P, Della Vedova G, Dondi R, Jiang T (2008) On the approximation of correlation clustering and consensus clustering. *Journal of Computer and System Sciences* 74(5):671–696.

Booth M, Reinhardt SP, Roy A (2017) Partitioning optimization problems for hybrid classical/quantum execution. *GitHub repository* URL `https://github.com/dwavesystems/qbsolv/blob/master/qbsolv_techReport.pdf`.

Çela E (2013) *The quadratic assignment problem: theory and algorithms*, volume 1 (Springer Science & Business Media).

Charfreitag J, Jünger M, Mallach S, Mutzel P (2022) Mcsparse: Exact solutions of sparse maximum cut and sparse unconstrained binary quadratic optimization problems. *2022 Proceedings of the Symposium on Algorithm Engineering and Experiments (ALENEX)*, 54–66 (SIAM).

Chen WA, Zhu Z, Kong N (2017) A Lagrangian decomposition approach to computing feasible solutions for quadratic binary programs. *Optimization Letters* ISSN 1862-4480, URL `http://dx.doi.org/10.1007/s11590-017-1125-x`.

Chrétienne P (1989) A polynomial algorithm to optimally schedule tasks on a virtual distributed system under tree-like precedence constraints. *European Journal of Operational Research* 43(2):225–230.

Dantzig GB, Wolfe P (1960) Decomposition principle for linear programs. *Operations research* 8(1):101–111.

De Fréminville PDLP, Desaulniers G, Rousseau LM, Perron S (2015) A column generation heuristic for districting the price of a financial product. *Journal of the Operational Research Society* 66(6):965–978.

Dehghan A, Shah M (2017) Binary quadratic programing for online tracking of hundreds of people in extremely crowded scenes. *IEEE transactions on pattern analysis and machine intelligence* 40(3):568–581.

Desaulniers G, Desrosiers J, Solomon MM (2006) *Column generation*, volume 5 (Springer Science & Business Media).

Dolan ED, Moré JJ (2002) Benchmarking optimization software with performance profiles. *Mathematical programming* 91(2):201–213.

Drwal M (2014) Algorithm for quadratic semi-assignment problem with partition size coefficients. *Optimization Letters* 8(3):1183–1190.

Du Merle O, Villeneuve D, Desrosiers J, Hansen P (1999) Stabilized column generation. *Discrete Mathematics* 194(1):229–237.

Emami P, Pardalos PM, Elefteriadou L, Ranka S (2018) Machine Learning Methods for Solving Assignment Problems in Multi-Target Tracking URL `https://arxiv.org/abs/1802.06897`.

Escoffier B, Hammer PL (2007) Approximation of the quadratic set covering problem. *Discrete Optimization* 4(3-4):378–386.

Fischer A (2014) An analysis of the asymmetric quadratic traveling salesman polytope. *SIAM Journal on Discrete Mathematics* 28(1):240–276.

Fischer F, Jaeger G, Lau A, Molitor P (2009) Complexity and algorithms for the traveling salesman problem and the assignment problem of second order. *Lecture Notes in Comput. Sci* 5165:211–224.

Furini F, Traversi E, Belotti P, Frangioni A, Gleixner A, Gould N, Liberti L, Lodi A, Misener R, Mittelmann H, et al. (2019) QPLIB: a library of quadratic programming instances. *Mathematical Programming Computation* 11(2):237–265.

Glover F, Woolsey E (1974) Converting the 0-1 polynomial programming problem to a 0-1 linear program. *Operations research* 22(1):180–182.

Hahn PM, Zhu YR, Guignard M, Hightower WL, Saltzman MJ (2012) A level-3 reformulation-linearization technique-based bound for the quadratic assignment problem. *INFORMS Journal on Computing* 24(2):202–209.

Hansen P, Lih KW (1992) Improved algorithms for partitioning problems in parallel, pipelined, and distributed computing. *IEEE Transactions on Computers* 41(6):769–771.

Helmberg C, Rendl F, Weismantel R (2000) A semidefinite programming approach to the Quadratic Knapsack Problem. *Journal of Combinatorial Optimization* 4(2):197–215, ISSN 1382-6905, URL `http://dx.doi.org/10.1023/A:1009898604624`.

Henschel R, Leal-Taixé L, Cremers D, Rosenhahn B (2018) Fusion of head and full-body detectors for multi-object tracking. *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 1428–1437.

Hu H, Sotirov R (2018) Special cases of the quadratic shortest path problem. *Journal of Combinatorial Optimization* 35(3):754–777.

Hu H, Sotirov R (2021) The linearization problem of a binary quadratic problem and its applications. *Annals of Operations Research* 307(1):229–249.

Jünger M, Mallach S (2021) Exact facetial odd-cycle separation for maximum cut and binary quadratic optimization. *INFORMS Journal on Computing* 33(4):1419–1430.

Khaniyev T (2018) Data-driven structure detection in optimization: Decomposition, hub location, and brain connectivity .

Khaniyev T, Elhedhli S, Erenay FS (2018) Structure detection in mixed-integer programs. *INFORMS Journal on Computing* 30(3):570–587.

Khaniyev T, Elhedhli S, Erenay FS (2020) Spatial separability in hub location problems with an application to brain connectivity networks. *INFORMS Journal on Optimization* 2(4):320–346.

Kochenberger G, Hao JK, Glover F, Lewis M, Lü Z, Wang H, Wang Y (2014) The unconstrained binary quadratic programming problem: a survey. *Journal of Combinatorial Optimization* 28(1):58–81.

Leal-Taixe L, Pons-Moll G, Rosenhahn B (2012) Branch-and-price global optimization for multi-view multi-target tracking. *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 1987–1994 (IEEE).

Liberti L (2007) Compact linearization for binary quadratic problems. *4OR* 5(3):231–245.

Lübbecke ME, Desrosiers J (2005) Selected topics in column generation. *Operations research* 53(6):1007–1023.

Magirou V, Milis J (1989) An algorithm for the multiprocessor assignment problem. *Operations research letters* 8(6):351–356.

Mallach S (2018) Compact linearization for binary quadratic problems subject to assignment constraints. *4OR* 16:295–309.

Mallach S (2023) Inductive linearization for binary quadratic programs with linear constraints: a computational study. *4OR* 1–41.

Malucelli F (1996) A polynomially solvable class of quadratic semi-assignment problems. *European Journal of Operational Research* 91(3):619–622.

Mauri GR, Lorena LAN (2011) Lagrangean decompositions for the unconstrained binary quadratic programming problem. *International Transactions in Operational Research* 18(2):257–270.

Mauri GR, Lorena LAN (2012) A column generation approach for the unconstrained binary quadratic programming problem. *European Journal of Operational Research* 217(1):69 – 74, ISSN 0377-2217, URL `http://dx.doi.org/http://dx.doi.org/10.1016/j.ejor.2011.09.016`.

Meier JF, Clausen U, Rostami B, Buchheim C (2016) A compact linearisation of euclidean single allocation hub location problems. *Electronic Notes in Discrete Mathematics* 52:37 – 44, ISSN 1571-0653, URL `http://dx.doi.org/http://dx.doi.org/10.1016/j.endm.2016.03.006`, INOC 2015 – 7th International Network Optimization Conference.

Milan A, Leal-Taixe L, Reid I, Roth S, Schindler K (2016) MOT16: A Benchmark for Multi-Object Tracking URL `https://arxiv.org/abs/1603.00831`.

O'Kelly ME (1987) A quadratic integer program for the location of interacting hub facilities. *European Journal of Operational Research* 32(3):393–404.

Oustry CLF (2001) SDP relaxations in combinatorial optimization from a Lagrangian viewpoint. *Advances in Convex Analysis and Global Optimization: Honoring the Memory of C. Caratheodory (1873-1950)* 54:119–134.

Pereira DL, da Cunha AS (2018) Polyhedral results, branch-and-cut and Lagrangian relaxation algorithms for the adjacent only quadratic minimum spanning tree problem. *Networks* 71(1):31–50.

Pereira DL, da Cunha AS (2020) Dynamic intersection of multiple implicit dantzig–wolfe decompositions applied to the adjacent only quadratic minimum spanning tree problem. *European Journal of Operational Research* 284(2):413–426.

Pereira DL, Gendreau M, Salles da Cunha A (2013) Stronger lower bounds for the quadratic minimum spanning tree problem with adjacency costs. *Electronic Notes in Discrete Mathematics* 41:229–236.

Pereira DL, Gendreau M, Salles da Cunha A (2015) Branch-and-cut and branch-and-cut-and-price algorithms for the adjacent only quadratic minimum spanning tree problem. *Networks* 65(4):367–379.

Pisinger D (2007) The quadratic knapsack problem—a survey. *Discrete applied mathematics* 155(5):623–648.

Punnen AP, Pandey P, Friesen M (2019) Representations of quadratic combinatorial optimization problems: A case study using quadratic set covering and quadratic knapsack problems. *Computers & Operations Research* 112:104769.

Punnen AP, Walter M, Woods BD (2017) A characterization of linearizable instances of the quadratic traveling salesman problem. *arXiv preprint arXiv:1708.07217* .

Rostami B, Chassein A, Hopf M, Frey D, Buchheim C, Malucelli F, Goerigk M (2018) The quadratic shortest path problem: complexity, approximability, and solution methods. *European Journal of Operational Research* 268(2):473–485.

Rostami B, Errico F, Lodi A (2022) A convex reformulation and an outer approximation for a large class of binary quadratic programs. *Operations Research* .

Rostami B, Malucelli F (2015) Lower bounds for the quadratic minimum spanning tree problem based on reduced cost computation. *Computers & Operations Research* 64:178 – 188, ISSN 0305-0548, URL `http://dx.doi.org/https://doi.org/10.1016/j.cor.2015.06.005`.

Rostami B, Malucelli F, Belotti P, Gualandi S (2016) Lower bounding procedure for the asymmetric quadratic traveling salesman problem. *European Journal of Operational Research* 253(3):584–592.

Rostami B, Malucelli F, Frey D, Buchheim C (2015) On the quadratic shortest path problem. Bampis E, ed., *Experimental Algorithms*, 379–390 (Cham: Springer International Publishing), ISBN 978-3-319-20086-6.

Rousseau LM, Gendreau M, Feillet D (2007) Interior point stabilization for column generation. *Operations Research Letters* 35(5):660–668.

Sahni S, Gonzalez T (1976) P-complete approximation problems. *Journal of the ACM (JACM)* 23(3):555–565.

Saito H, Fujie T, Matsui T, Matuura S (2009) A study of the quadratic semi-assignment polytope. *Discrete Optimization* 6(1):37 – 50, ISSN 1572-5286, URL `http://dx.doi.org/http://dx.doi.org/10.1016/j.disopt.2008.08.003`.

Schüle I, Ewe H, Küfer KH (2009) Finding tight RLT formulations for quadratic semi-assignment problems. *Proceedings of 8th Cologne-Twente Workshop on Graphs and Combinatorial Optimization, CTW*, 109–112.

Shen H, Huang L, Huang C, Xu W (2018) Tracklet Association Tracker: An End-to-End Learning-based Association Approach for Multi-Object Tracking URL `https://arxiv.org/abs/1808.01562`.

Sherali HD, Adams WP (2013) *A reformulation-linearization technique for solving discrete and continuous nonconvex problems*, volume 31 (Springer Science & Business Media).

Sherali HD, Smith JC (2007) An improved linearization strategy for zero-one quadratic programming problems. *Optimization Letters* 1(1):33–47.

Silva A, Coelho LC, Darvish M (2021) Quadratic assignment problem variants: A survey and an effective parallel memetic iterated tabu search. *European Journal of Operational Research* 292(3):1066–1084.

Stone HS (1977) Multiprocessor scheduling with the aid of network flow algorithms. *IEEE transactions on Software Engineering* (1):85–93.

Tang S, Andriluka M, Andres B, Schiele B (2017) Multiple people tracking by lifted multicut and person re-identification. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, ISBN 9781538604571, URL `http://dx.doi.org/10.1109/CVPR.2017.394`.

Wang S, Wolf S, Fowlkes CC, Yarkony J (2017) Tracking objects with higher order interactions via delayed column generation. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017*.

Yarkony J, Adulyasak Y, Singh M, Desaulniers G (2020) Data association via set packing for computer vision applications. *Informs Journal on Optimization* 2(3):167–191.

Online Supplement

Appendix A: Linearization

The standard linearization technique (SLT) is a known technique in the literature to transform the BQP model into its equivalent mixed integer program (MIP) model by substituting the quadratic terms with extra binary variables y_{ef}. For instance, linearized constraints for the MOT problem are as follows:

$$y_{ef} \geq x_e + x_f - 1 \qquad \forall (e,f) \in \mathcal{A} \tag{46}$$

$$y_{ef} \leq x_e \qquad \forall (e,f) \in \mathcal{A} \tag{47}$$

$$y_{ef} \leq x_f \qquad \forall (e,f) \in \mathcal{A} \tag{48}$$

$$y_{ef} \geq 0 \qquad \forall (e,f) \in \mathcal{A} \tag{49}$$

where it suggests $O(n^3)$ decision variables and constraints be added to the BQP model.

The reformulation linearization technique (RLT) is another mathematical programming technique used for linearization. In this study, we exploit a tighter linearization which is similar to the RLT to linearize the BQP formulation of MOT as well as to linearize the star-based reformulation of MOT. Considering (37) and multiplying it once by $x_f, \forall f \in \delta^{t'}(j)$, where t' represents any frame except frame t and then generating the same constraint for x_e, we obtain the following strong valid inequalities instead of the SLT constraints (47) and (48):

$$\sum_{e \in \delta^t(j)} y_{ef} \leq x_f \qquad \forall t \in T, \forall j \in H, \forall f \in \delta^{t'}(j) : t' \neq t \tag{50}$$

$$\sum_{f \in \delta^t(j)} y_{ef} \leq x_e \qquad \forall t \in T, \forall j \in H, \forall e \in \delta^{t'}(j) : t' \neq t \tag{51}$$

The RLT model applied to the quadratic semi-assignment problem of (14)-(16) is formulated as follows:

$$\min \quad \sum_{e \in A} c_e x_e + \sum_{\substack{i,k \in N \\ i \neq k}} \sum_{\substack{e \in \delta(i) \\ f \in \delta(k)}} q_{ef} y_{ef} \tag{52}$$

$$\text{s.t.} \quad \sum_{e \in \delta(i)} x_e = 1 \qquad \forall i \in N \tag{53}$$

$$\sum_{e \in \delta(i)} y_{ef} = x_f \qquad \forall i,k \in N, i \neq k, \forall f \in \delta(k) \tag{54}$$

$$\sum_{f \in \delta(k)} y_{ef} = x_e \qquad \forall i,k \in N, i \neq k, \forall e \in \delta(i) \tag{55}$$

$$y_{ef} \geq 0 \qquad \forall i,k \in N, i \neq k, \forall e \in \delta(i), \forall f \in \delta(k) \tag{56}$$

$$x_e \in \{0,1\} \qquad \forall e \in A. \tag{57}$$

Appendix B: Instance-by-instance tables

In the following, we detail the computational results for each problem in two tables consisting of dual bounds and computational time. For each problem, one table demonstrates the experiments for the instances in which at least one of the compared methods stops within the time limit. The other table contains instances

in which all the methods reach the time limit; thus, the three hours of fixed computation time is not reported in this table to avoid repetition. As an index of efficiency for each method, we report the ratio of the obtained lower bound to the best known feasible solution (BFS). The BFS is computed by comparing the obtained upper bounds of all methods in the three hour timeframe.

Table 1 AQSAP – Comparing different methods – At least one of the methods stops within the time limit

Instance	BFS	BQP		CG+BQPPricing		CG+HeuristicBQPPricing		CG+HeuristicSLTPricing		SLT		CG+SLTPricing		RLT		OuterApproximation	
(m-n)		LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time
10-3	3038.8	**1**	**0**	1	2	1	8	1	6	1	1	1	2	1	1	1	0.1
10-4	2829.3	**1**	**0**	1	4	1	11	1	8	1	2	1	1	1	2	1	0.1
15-3	32202.4	1	52	1	56	1	18	1	14	1	40	1	16	1	30	**1**	**6**
15-4	21336.7	1	524	1	57	1	28	1	28	1	314	**1**	**17**	1	593	1	120
15-6	4472.7	1	327	1	41	1	25	1	16	1	233	**1**	**10**	1	1395	1	95
18-3	8168.7	1	43	1	298	1	35	1	24	1	25	1	142	1	61	**1**	**13**
18-4	6747.3	1	1852	1	142	**1**	**5**	1	8	1	909	1	127	1	1910	1	535
18-6	5906.2	0.82	10800	1	96	1	53	**1**	**43**	0.81	10800	1	147	0.73	10800	0.82	10800
18-8	4916	0.85	10800	1	96	1	83	**1**	**82**	0.8	10800	1	160	0.74	10800	0.79	10800
20-3	15549.8	1	1099	1	430	**1**	**10**	1	25	1	345	1	531	1	422	1	68
20-4	13657.3	0.92	10800	1	333	1	58	**1**	**25**	0.94	10800	1	331	0.86	10800	1	7406
20-6	9781.5	0.72	10800	1	227	1	101	**1**	**52**	0.68	10800	1	293	0.55	10800	0.66	10800
20-8	8559.1	0.62	10800	1	105	**1**	**10**	1	31	0.62	10800	1	176	0.55	10800	0.62	10800
20-10	4810.8	0.83	10800	1	151	**1**	**18**	1	59	0.81	10800	1	195	0.61	10800	0.79	10800
22-3	42873.5	0.97	10800	1	1744	**1**	**51**	1	74	1	7360	1	1612	0.98	10800	1.00	4807
22-4	24085.5	0.87	10800	1	500	**1**	**27**	1	76	0.84	10800	1	590	0.75	10800	0.80	10800
22-6	18316.3	0.72	10800	1	237	**1**	**28**	1	66	0.63	10800	1	404	0.46	10800	0.52	10800
22-8	7394.2	0.74	10800	1	205	**1**	**13**	1	31	0.67	10800	1	293	0.61	10800	0.67	10800
22-10	6017.8	0.78	10800	1	188	1	95	**1**	**49**	0.73	10800	1	284	0.59	10800	0.73	10800
22-12	4519.2	0.83	10800	1	158	1	100	**1**	**73**	0.76	10800	1	197	0.56	10800	0.80	10800
25-3	42794.9	0.94	10800	1	3127	**1**	**51**	1	72	0.98	10800	1	2417	0.89	10800	0.92	10800
25-4	14012.7	0.9	10800	1	598	**1**	**16**	1	79	0.9	10800	1	656	0.82	10800	0.88	10800
25-6	9478.1	0.68	10800	1	1937	1	684	**1**	**174**	0.59	10800	1	1477	0.46	10800	0.54	10800
25-8	8807.1	0.68	10800	1	581	1	566	**1**	**252**	0.64	10800	1	614	0.47	10800	0.57	10800
25-10	6318.8	0.7	10800	1	975	**1**	**98**	1	99	0.61	10800	1	591	0.43	10800	0.56	10800
25-12	4744	0.61	10800	1	265	1	228	**1**	**179**	0.52	10800	1	348	0.38	10800	0.47	10800
30-3	48847.1	0.89	10800	1	1741	**1**	**70**	1	98	0.94	10800	1	1863	0.90	10800	1.00	8424
30-4	20369.8	0.74	10800	0.99	10800	1	3665	**1**	**1143**	0.73	10800	1	9636	0.62	10800	0.65	10800
30-6	14167	0.52	10800	1	10800	**1**	**2001**	1	2287	0.5	10800	1	10800	0.39	10800	0.45	10800
30-8	10322.5	0.58	10800	1	6378	1	2612	**1**	**964**	0.5	10800	1	2953	0.31	10800	0.45	10800
30-14	6680.8	0.47	10800	1	1875	**1**	**214**	1	831	0.41	10800	1	818	0.31	10800	0.38	10800
40-14	10365.5	0.4	10800	0.98	10800	1	10800	**1**	**4062**	0.34	10800	1	10800	0.21	10800	0.31	10800
50-4	49468.9	0.43	10800	1	10800	**1**	**6895**	1	10800	0.37	10800	0.97	10800	0.29	10800	0.35	10800
50-6	40831	0.67	10800	1	10800	**1**	**4156**	0.56	10800	0.25	10800	0.82	10800	0.17	10800	0.22	10800
50-8	34960.5	0.5	10800	1	10800	**1**	**7146**	0.46	10800	0.24	10800	0.78	10800	0.18	10800	0.22	10800

Table 2 AQSAP – Comparing different methods – None of the methods stop within the time limit

Instance (m-n)	BFS	BQP	CG+BQPPricing	CG+Heuristic BQPPricing	CG+Heuristic SLTPricing	SLT	CG+SLTPricing	RLT	Outer Approximation
		LB/BFS	LB/BFS	LB/BFS	LB/BFS	LB/BFS	LB/BFS	LB/BFS	LB/BFS
40-3	46075.8	0.66	0.94	**1**	**1**	0.58	0.93	0.51	0.58
40-4	37264	0.46	**0.89**	0.80	0.80	0.41	**0.89**	0.28	0.35
40-6	17322.9	0.36	0.86	0.86	0.86	0.28	**0.88**	0.20	0.26
40-8	16018.9	0.34	0.94	0.84	**1**	0.29	0.90	0.21	0.27
50-3	52448.4	0.61	0.99	**1**	**1**	0.50	0.94	0.47	0.50
50-14	11696.4	0.29	0.99	**1**	0.41	0.34	0.78	0.21	0.29

A closer look at Tables 3 to 10 proves that in some cases of the QSAP with out-of-star interactions, one has to choose the solution methodology by considering the trade-off between time and the quality of LB. The reason is that CG may stop in a shorter time with a slightly worse LB, which we do not consider in our performance analysis. In addition, for larger-size instances of different data sets in QSAP, a significant pattern is observed in the results. When the number of servers (h) is relatively large compared to the number of clients (n), one of the CG methods yields the best results, while instances with a smaller number of servers are solved by GUROBI and the OuterApproximation methods with better bounds. Interestingly, in certain cases where none of the techniques stops prior to the time limit, CG exhibits a superior LB. As various techniques (such as acceleration and stabilization techniques) can be applied to further improve the performance of CG, there is room for enhancements in our methodology to make it more effective in tackling large-scale problems.

According to the tables associated with the MOT problem, increasing the parameter d for a problem with a fixed number of frames ($\mathcal{T}$) results in more computational time, highlighting the effects of the density of the quadratic matrix in solving the problem. Table 12 demonstrates that in many cases CG algorithms converge to an optimal solution of the problem. Specifically, when we use CG+RLTPricing and the problem can be solved within the time limit, optimality is proved for almost all the instances.

Table 3 QSAP – 10% out-of-star quadratic matrix density – Comparing different methods – At least one of the methods stops within the time limit

Instance	BFS	BQP		CG+BQPPricing		CG+ HeuristicBQPPricing		CG+ HeuristicSLTPricing		SLT		CG+SLTPricing		RLT		OuterApproximation	
(m-n)		LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time
10-3	3061.4	**1**	**0.1**	1	6	1	6	1	3	1	2	1	2	**1**	**0.1**	**1**	**0.1**
10-4	2829.3	1	1	1	5	1	5	1	5	1	1	1	2	**1**	**0.1**	**1**	**0.1**
15-3	33075.4	1	4	0.98	77	0.98	10	0.98	6	1	11	0.98	22	1	2	**1**	**1**
15-4	21665.4	1	21	0.99	71	0.99	18	0.99	14	1	348	0.99	17	1	10	**1**	**5**
15-6	4497.2	1	7	0.99	40	0.99	15	0.99	15	1	79	0.99	64	**1**	**5**	1	6
18-3	8383	1	4	0.99	164	0.99	41	0.99	32	1	13	0.99	180	**1**	**1**	1	2
18-4	6884.3	1	23	0.98	144	0.98	29	0.98	24	1	119	0.98	169	**1**	**12**	1	25
18-6	5957.5	1	248	0.99	123	0.99	47	**0.99**	**39**	1	4021	0.99	135	**1**	**239**	1	395
18-8	4956.1	1	4762	0.99	125	0.99	43	**0.99**	**40**	0.76	10800	0.99	129	**1**	**3748**	1	6587
20-3	16245.3	1	20	0.97	533	0.97	103	0.97	55	1	75	0.97	561	1	25	**1**	**5**
20-4	13942.2	1	492	0.98	459	0.98	53	**0.98**	**25**	1	1701	0.98	502	1	170	**1**	**37**
20-6	9919.2	1	4867	0.99	224	0.99	42	**0.99**	**23**	0.51	10800	0.99	270	1	9593	**1**	**2166**
20-8	8697.8	0.75	10800	0.98	156	0.98	76	**0.98**	**31**	0.53	10800	0.98	250	0.71	10800	0.76	10800
20-10	4878.3	0.88	10800	0.99	194	0.99	87	**0.99**	**34**	0.64	10800	0.99	280	0.74	10800	0.81	10800
22-3	44517.4	1	444	0.96	1092	0.96	34	**0.96**	**21**	1	1632	0.96	1432	1	328	**1**	**183**
22-4	25071.6	1	4806	0.96	589	0.96	89	**0.96**	**36**	1	9336	0.96	768	1	3453	**1**	**2033**
22-6	18872.5	0.79	10800	0.97	312	0.97	89	**0.97**	**31**	0.55	10800	0.97	391	0.76	10800	0.70	10800
22-8	7587.9	0.82	10800	0.97	322	0.97	109	**0.97**	**59**	0.62	10800	0.97	511	0.75	10800	0.80	10800
22-10	6121.4	0.84	10800	0.99	336	**0.99**	**84**	0.99	107	0.65	10800	0.99	432	0.72	10800	0.78	10800
22-12	4566.7	0.88	10800	0.99	353	**0.99**	**111**	0.99	126	0.62	10800	0.99	478	0.71	10800	0.90	10800
25-3	44875.1	**1**	**206**	0.96	3207	0.96	516	0.96	176	1	918	0.96	4427	1	181	1	196
25-4	14811.7	**1**	**83**	0.95	1256	0.95	122	0.95	66	1	567	0.95	1298	1	184	1	153
25-6	9887.9	0.79	10800	0.96	2361	0.96	376	**0.96**	**195**	0.47	10800	0.96	2629	0.74	10800	0.79	10800
25-8	8999.1	0.73	10800	0.98	2051	0.98	673	**0.98**	**402**	0.53	10800	0.98	1492	0.64	10800	0.67	10800
25-10	6509.8	0.74	10800	0.97	2698	**0.97**	**367**	0.97	543	0.39	10800	0.97	1822	0.55	10800	0.63	10800
25-12	4921.6	0.47	10800	0.96	1130	0.96	1024	**0.96**	**916**	0.35	10800	0.96	1712	0.45	10800	0.57	10800
30-3	53502.1	**1**	**203**	0.93	8255	0.94	412	**0.94**	**79**	1	2902	0.94	1863	1	361	1	208
30-4	21896.3	**1**	**1359**	0.93	10800	0.93	4229	0.93	4350	1	8366	0.93	9636	1	1101	1	2049
30-6	14736.8	0.73	10800	0.95	10800	**0.96**	**2324**	0.96	3722	0.43	10800	0.96	10800	0.65	10800	0.64	10800
30-8	10807.5	0.40	10800	0.95	8560	0.95	4120	**0.95**	**2720**	0.24	10800	0.95	8120	0.44	10800	0.47	10800
30-14	6978.9	0.36	10800	0.96	7894	**0.96**	**1063**	0.96	10800	0.26	10800	0.96	10800	0.33	10800	0.39	10800

Table 4 QSAP – 10% out-of-star quadratic matrix density – Comparing different methods – None of the methods stops within the time limit

Instance (m-n)	BFS	BQP LB/BFS	CG+BQPPricing LB/BFS	CG+Heuristic BQPPricing LB/BFS	CG+Heuristic SLTPricing LB/BFS	SLT LB/BFS	CG+SLTPricing LB/BFS	RLT LB/BFS	Outer Approximation LB/BFS
40-3	51693.8	0.75	0.83	0.68	**0.87**	0.63	0.83	0.68	0.72
40-4	41684.2	0.52	**0.78**	0.71	0.71	0.22	**0.78**	0.40	0.48
40-6	19765.3	0.31	**0.76**	0.18	0.18	0.17	0.68	0.24	0.27
40-8	18865.9	0.21	**0.43**	0.35	0.35	0.17	**0.43**	0.23	0.26
40-14	11586.1	0.25	**0.51**	**0.51**	0.38	0.21	**0.51**	0.25	0.28
50-3	60996.3	0.83	0.87	**0.88**	0.82	0.58	0.79	0.73	0.65
50-4	62141.8	0.46	0.79	**0.81**	0.76	0.26	0.73	0.44	0.53
50-6	51281.2	0.27	0.78	**0.79**	0.27	0.16	0.50	0.18	0.23
50-8	44208.2	0.18	0.77	**0.79**	0.33	0.15	0.45	0.15	0.19
50-14	14426.3	0.22	**0.37**	0.25	0.30	0.20	0.29	0.17	0.23

Appendix C: Primal bounding

As mentioned before, primal bounding methodologies are beyond the scope of this project. However, to show the strength of the proposed reformulation framework and CG for the adjacent-only class of BQPs, we obtain the primal bounds for all of the methods. In the next sections we first discuss the obtained UB for the adjacent-only problems and then we point out some discussions on the QSAP as an example of general BQPs.

C.1. Upper bound for adjacent-only problems

Although we use a very trivial heuristic to find the feasible solutions after the CG methods terminate, in a relatively large fraction of instances in adjacent-only problems, the BFS is obtained through the CG, meaning that the reformulation and CG outperform other formulations and methods in terms of both primal bound and dual bound for these problems.

To find the BFS for each instance, we compare the primal bounds obtained by different methods. In Figure 9, we present the fraction of instances where the BFS is found through each method. This figure proves the capabilities of the CG methods to find the best upper bounds in both the AQSAP and the MOT problem. According to Figure 9a, the CG+HeuristicBQPPricing obtains the BFS for the largest fraction of instances (93%) among all of the methods. For MOT, Figure 9b indicates that the CG+RLTPricing is the best method, finding the BFS for 70% of instances.

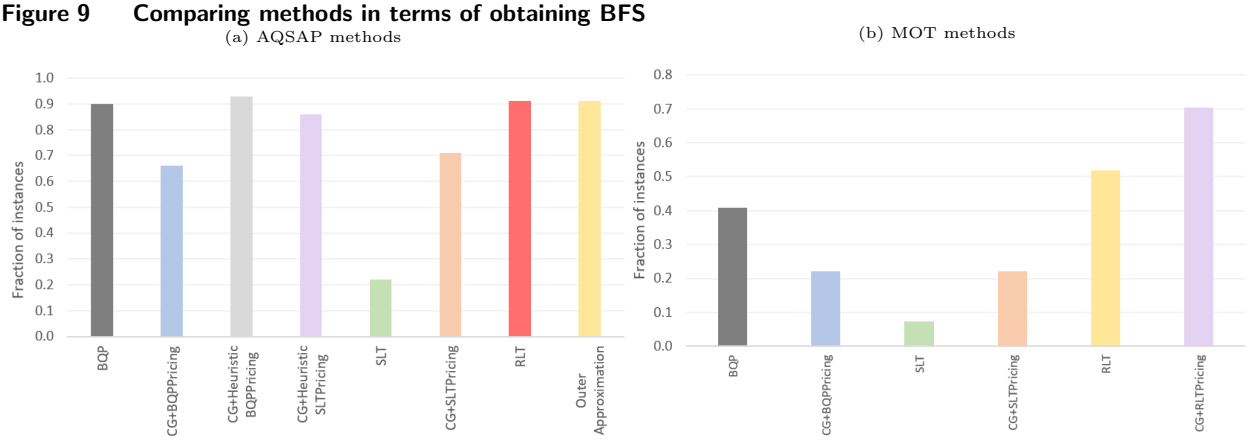
Table 5 QSAP – 15% out-of-star quadratic matrix density – Comparing different methods – At least one of the methods stops within the time limit

Instance	BFS	BQP		CG+BQPPricing		CG+HeuristicBQPPricing		CG+HeuristicSLTPricing		SLT		CG+SLTPricing		RLT		OuterApproximation	
(m-n)		LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time
10-3	3172.5	**1**	**0**	0.97	4	0.97	4	0.97	4	1	2	0.97	1	**1**	**0**	**1**	**0**
10-4	2881.7	1	1	0.99	5	0.99	5	0.99	4	1	1	0.99	1	**1**	**0**	**1**	**0**
15-3	33324.7	1	2	0.97	54	0.97	14	0.97	10800	1	8	0.97	18	**1**	**1**	**1**	**1**
15-4	22318.5	1	6	0.96	63	0.96	26	0.96	20	1	65	0.96	65	1	5	**1**	**4**
15-6	4568.5	1	4	0.98	41	0.98	16	0.98	21	1	38	0.98	56	**1**	**3**	1	4
18-3	8669.3	1	2	0.96	182	0.96	24	0.96	22	1	6	0.96	212	**1**	**0**	1	1
18-4	6978.7	1	10	0.97	194	0.97	29	0.97	26	1	38	0.97	193	**1**	**4**	1	13
18-6	6126.4	1	130	0.96	122	0.96	62	0.96	59	1	1544	0.96	137	**1**	**75**	1	136
18-8	5034.8	**1**	**419**	0.98	97	0.98	57	**0.98**	**52**	1	7584	0.98	122	1	500	1	840
20-3	16434.4	1	9	0.96	600	0.96	71	0.96	45	1	27	0.96	550	1	6	**1**	**2**
20-4	14529.7	1	97	0.94	349	0.94	32	0.94	19	1	788	0.94	444	1	76	**1**	**19**
20-6	10257.6	1	1707	0.95	206	0.95	57	**0.95**	**25**	0.85	10800	0.95	283	1	1724	**1**	**391**
20-8	9062.6	**1**	**9703**	0.95	227	0.95	84	**0.95**	**36**	0.6	10800	0.95	289	0.85	10800	0.85	10800
20-10	4965.2	**1**	**4169**	0.97	196	0.97	110	**0.97**	**50**	0.65	10800	0.97	315	1	10800	1	6248
22-3	45955.1	1	222	0.93	897	0.93	52	**0.93**	**37**	1	1037	0.93	1525	1	151	**1**	**88**
22-4	26366.2	**1**	**899**	0.91	464	0.91	121	**0.91**	**44**	1	4682	0.91	654	1	868	**1**	**899**
22-6	19512	1	8792	0.94	256	0.94	73	**0.94**	**29**	0.61	10800	0.94	433	0.89	10800	**1**	**7364**
22-8	8214.5	**1**	**4946**	0.9	525	0.90	106	**0.90**	**78**	0.64	10800	0.9	492	1	6424	1	8687
22-10	6349.8	0.85	10800	0.95	259	**0.95**	**168**	0.95	237	0.65	10800	0.95	368	0.85	10800	1	9966
22-12	4821.5	0.70	10800	0.93	433	0.93	160	**0.93**	**131**	0.65	10800	0.93	490	1	10800	**1**	**8706**
25-3	48713.8	**1**	**117**	0.89	4418	0.89	287	0.89	202	1	377	0.89	4186	**1**	**117**	1	129
25-4	16521.4	**1**	**46**	0.86	1685	0.86	121	0.86	75	1	241	0.86	1631	1	81	1	122
25-6	10827.9	1	7729	0.88	1932	0.88	384	**0.88**	**191**	0.51	10800	0.88	1895	**1**	**7087**	1	8188
25-8	9745.2	0.61	10800	0.9	4584	0.9	952	**0.9**	**511**	0.65	10800	0.9	2642	0.72	10800	0.76	10800
25-10	7063.3	0.59	10800	0.89	3336	0.89	942	**0.89**	**840**	0.39	10800	0.89	2130	0.61	10800	0.68	10800
25-12	4738.4	0.44	10800	**0.85**	**1328**	0.85	10156	0.85	10283	0.32	10800	0.85	10800	0.45	10800	0.58	10800
30-3	57082.6	**1**	**93**	0.88	10800	0.88	1751	0.88	264	1	1025	0.88	10800	1	103	1	118
30-4	24213	**1**	**109**	0.84	10800	0.85	2085	0.85	1132	1	2207	0.85	10800	1	398	1	322
30-6	18614.7	0.68	10800	0.77	10800	**0.77**	**809**	0.77	2739	0.41	10800	0.77	10800	0.69	10800	0.64	10800
30-8	14294	0.38	10800	0.78	5906	0.78	5366	**0.78**	**3479**	0.25	10800	0.78	8978	0.39	10800	0.44	10800
30-14	10284	0.31	10800	0.80	9593	0.80	1371	**0.80**	**1365**	0.23	10800	0.80	10800	0.29	10800	0.34	10800
40-14	17160.5	0.17	10800	0.36	10800	0.22	1540	0.19	10800	0.15	10800	**0.37**	**10800**	0.17	10800	0.2	10800

Table 6 QSAP – 15% out-of-star quadratic matrix density – Comparing different methods – None of the methods stops within the time limit

Instance (m-n)	BFS	BQP LB/BFS	CG+BQPPricing LB/BFS	CG+Heuristic BQPPricing LB/BFS	CG+Heuristic SLTPricing LB/BFS	SLT LB/BFS	CG+SLTPricing LB/BFS	RLT LB/BFS	Outer Approximation LB/BFS
40-3	55376	**0.85**	0.75	0.62	0.82	0.65	0.76	0.78	0.76
40-4	47234.8	0.68	**0.70**	0.64	0.64	0.32	0.69	0.41	0.50
40-6	23518.1	0.29	**0.62**	0.22	0.22	0.15	0.60	0.25	0.27
40-8	27886.8	0.16	**0.29**	0.17	0.17	0.13	**0.29**	0.17	0.20
50-3	65385.1	**0.91**	0.81	0.82	0.77	0.66	0.76	0.83	0.80
50-4	70572.9	0.57	0.69	**0.70**	0.66	0.29	0.63	0.54	0.58
50-6	62755.8	0.24	0.63	**0.65**	0.23	0.14	0.51	0.17	0.25
50-8	53024.2	0.16	0.63	**0.66**	0.32	0.13	0.35	0.14	0.18
50-14	26288.9	0.12	**0.17**	0.03	0.16	0.12	0.16	0.10	0.14

Figure 9 Comparing methods in terms of obtaining BFS



C.2. Upper bound for QSAP

Akin to the heuristic in AQSAP and MOT, we use all of the entered columns in the QSAP-RMP to build our IP model to obtain a valid UB for QSAP. However, instead of solving the IP version of the last RMP directly, we solve an RMP with a slightly different objective function. In this model, we build a new quadratic cost function to reflect the interaction between each pair of stars (columns) of the RMP as below:

$$\begin{cases} Q_{s,s'} = 0, & \text{if } s \text{ and } s' \text{ have } i \text{ or } j \text{ in common} \\ Q_{s,s'} = \sum_{e \in s, f \in s'} q_{ef}, & \text{otherwise} \end{cases}$$

Therefore, the alternative model to solve is:

$$\min \quad \sum_{s \in S} C_s \lambda_s + \sum_{s \in S} \sum_{s' > s \in S} Q_{ss'} \lambda_s \lambda_{s'} \qquad \text{s.t.} \quad (30-32) \tag{58}$$

Through the new definition of the quadratic costs, we discard many impossible interactions among the stars from the beginning. This leads to reaching the same feasible solutions as with the previously mentioned heuristic in a much shorter time. Nonetheless, even with this reformulation the obtained feasible solutions are not promising. We do not report the details on the UB of the QSAP, but we would like to provide some remarks to direct future research on this topic:

Table 7 QSAP – 20% out-of-star quadratic matrix density – Comparing different methods – At least one of the methods stops within the time limit

Instance (m-n)	BFS	BQP		CG+BQPPricing		CG+ HeuristicBQPPricing		CG+ HeuristicSLTPricing		SLT		CG+SLTPricing		RLT		OuterApproximation	
		LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time
10-3	3213.4	**1**	**0**	0.96	5	0.96	3	0.96	3	1	1	0.96	1	**1**	**0**	**1**	**0**
10-4	3001.4	1	1	0.96	9	0.96	5	0.96	4	1	1	0.96	2	**1**	**0**	**1**	**0**
15-3	34726.1	1	2	0.93	60	0.93	11	0.93	10	1	6	0.93	21	**1**	**1**	**1**	**1**
15-4	23852.4	1	7	0.90	72	0.90	19	0.90	19	1	54	0.90	75	1	4	**1**	**3**
15-6	5132.6	1	5	0.87	52	0.87	21	0.87	21	1	33	0.87	66	**1**	**2**	1	4
18-3	9895.4	1	2	0.84	182	0.84	31	0.84	26	1	5	0.84	227	**1**	**1**	**1**	**1**
18-4	7357.1	1	6	0.92	174	0.92	44	0.92	35	1	19	0.92	174	**1**	**2**	1	5
18-6	6379.6	1	43	0.92	138	0.92	54	0.92	48	1	374	0.92	138	**1**	**21**	1	63
18-8	5446	1	228	0.90	116	0.90	55	**0.90**	**48**	1	3503	0.90	134	**1**	**201**	1	571
20-3	18140.5	1	8	0.87	533	0.87	189	0.87	13	1	39	0.87	627	1	4	**1**	**1**
20-4	15836	1	73	0.86	394	0.86	28	0.86	20	1	553	0.86	461	1	76	**1**	**14**
20-6	11201.2	1	1681	0.87	197	0.87	51	**0.87**	**26**	1	6065	0.87	286	1	1108	**1**	**315**
20-8	10102.4	**1**	**2891**	0.85	169	0.85	124	**0.85**	**40**	0.62	10800	0.85	219	1	6039	1	5047
20-10	5464.7	1	4997	0.88	242	0.88	167	**0.88**	**71**	0.69	10800	0.88	337	**1**	**4995**	1	5199
22-3	49327.2	1	248	0.87	1066	0.87	64	0.87	28	1	692	0.87	1693	1	153	**1**	**81**
22-4	28349	1	705	0.85	524	0.85	113	**0.85**	**45**	1	4661	0.85	659	1	832	**1**	**510**
22-6	21754.1	1	7202	0.84	214	0.84	115	**0.84**	**46**	0.61	10800	0.84	431	1	10800	**1**	**6822**
22-8	9714.9	0.71	10800	0.76	342	0.76	132	**0.76**	**87**	0.63	10800	0.76	470	**1**	**4536**	1	4872
22-10	7733.6	0.74	10800	0.78	322	**0.78**	**243**	0.78	263	0.61	10800	0.78	438	1	10800	**1**	**7526**
22-12	5884.6	0.66	10800	0.77	685	0.77	461	**0.77**	**687**	0.54	10800	0.77	818	0.78	10800	**1**	**10800**
25-3	51647	**1**	**88**	0.84	4549	0.84	218	0.84	149	1	321	0.84	4396	1	135	1	110
25-4	17712.6	**1**	**21**	0.80	1669	0.80	112	0.80	69	1	74	0.80	1812	1	51	1	71
25-6	12832.8	**1**	**2871**	0.74	2300	0.74	241	**0.74**	**196**	0.77	10800	0.74	2235	1	4243	1	6120
25-8	13192.7	0.55	10800	0.67	5744	0.67	542	**0.67**	**384**	0.41	10800	0.67	2463	**0.68**	**10800**	0.63	10800
25-10	8560.9	0.52	10800	0.74	2446	0.74	1284	**0.74**	**957**	0.31	10800	0.74	1967	0.57	10800	0.62	10800
25-12	7883.8	0.35	10800	0.60	1181	0.60	366	**0.60**	**295**	0.27	10800	0.60	1173	0.36	10800	0.44	10800
30-3	61853.4	**1**	**48**	0.81	10800	0.81	1105	0.81	237	1	257	0.81	10800	1	69	1	71
30-4	31188.4	**1**	**222**	0.65	10800	0.66	10800	0.66	1557	1	2317	0.66	10800	1	546	1	711
30-6	25956.1	0.52	10800	0.55	10800	**0.55**	**2629**	0.55	1965	0.34	10800	0.55	10800	**0.59**	**10800**	**0.59**	**10800**
30-8	19918.5	0.33	10800	0.52	10800	0.52	4722	**0.52**	**2499**	0.18	10800	0.52	7811	0.30	10800	0.38	10800
30-14	14062.6	0.19	10800	0.48	3044	0.48	1185	**0.48**	**1191**	0.15	10800	0.48	1461	0.17	10800	0.22	10800
50-3	71284.3	**1**	**4427**	0.75	10800	0.75	10800	0.69	10800	1	5015	0.37	10800	1	7429	0.86	10800

Table 8 QSAP – 20% out-of-star quadratic matrix density – Comparing different methods – None of the methods stops within the time limit

Instance (m-n)	BFS	BQP	CG+BQPPricing	CG+Heuristic BQPPricing	CG+Heuristic SLTPricing	SLT	CG+SLTPricing	RLT	Outer Approximation
		LB/BFS	LB/BFS	LB/BFS	LB/BFS	LB/BFS	LB/BFS	LB/BFS	LB/BFS
40-3	59514.1	**0.85**	0.71	0.65	0.76	0.74	0.71	0.82	0.80
40-4	53056.6	**0.70**	0.62	0.50	0.50	0.27	0.61	0.42	0.50
40-6	28760	0.27	**0.53**	0.29	0.29	0.14	0.50	0.24	0.26
40-8	39915	0.14	0.19	0.11	0.11	0.10	**0.21**	0.13	0.16
40-14	28785.3	0.10	0.18	0.17	0.10	0.10	**0.19**	0.10	0.12
50-4	77901.2	**0.74**	0.63	0.64	0.60	0.36	0.56	0.68	0.65
50-6	74931.2	0.29	0.53	**0.55**	0.22	0.14	0.33	0.18	0.23
50-8	70230.7	0.15	0.48	**0.49**	0.23	0.11	0.27	0.13	0.15
50-14	27759.7	0.12	**0.17**	0.01	0	0.12	0.15	0.11	0.14

REMARK 6. GUROBI methods, and in particular BQP, attain the BFS in the vast majority of the cases in the QSAP data sets. However, as the tables in Appendix B suggest, it could not find the best LB and close the optimality gap for many instances of the problems, while CG methods can prove optimality in many instances of adjacent-only problems.

REMARK 7. Interestingly, we observe that in the cases where CG methods terminate within the time limit, the obtained UB is fairly close to the BFS provided by GUROBI. In the contrast, when the CG methods reach the three-hour time limit, they suggest a weak UB, while their associated LB is still better than GUROBI in a relevant subset of instances.

Table 9 QSAP – 25% out-of-star quadratic matrix density – Comparing different methods – At least one of the methods stops within the time limit

Instance (m-n)	BFS	BQP		CG+BQPPricing		CG+ HeuristicBQPPricing		CG+ HeuristicSLTPricing		SLT		CG+SLTPricing		RLT		OuterApproximation	
		LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time	LB/BFS	Time
10-3	3456.3	1	1	0.89	7	0.89	5	0.89	4	**1**	**1**	0.89	1	**1**	**0**	**1**	**0**
10-4	3161	1	1	0.91	6	0.91	8	0.91	6	**1**	**0**	0.91	2	**1**	**0**	**1**	**0**
15-3	37682.9	1	2	0.86	66	0.86	11	0.86	8	1	6	0.86	20	**1**	**1**	**1**	**1**
15-4	25935.9	1	6	0.82	77	0.82	26	0.82	22	1	53	0.82	89	**1**	**3**	**1**	**3**
15-6	6285.4	1	3	0.71	52	0.71	24	0.71	23	1	50	0.71	65	**1**	**2**	1	4
18-3	10703.9	1	2	0.77	168	0.77	37	0.77	38	1	5	0.77	221	**1**	**0**	1	1
18-4	9346.3	1	7	0.72	203	0.72	32	0.72	29	1	23	0.72	222	**1**	**3**	1	10
18-6	7765.1	1	76	0.76	124	0.76	51	0.76	41	1	722	0.76	142	**1**	**27**	1	100
18-8	6158.9	1	187	0.80	116	0.80	80	0.80	71	1	2245	0.80	131	**1**	**156**	1	551
20-3	20576.4	1	10	0.77	519	0.77	52	0.77	16	1	33	0.77	704	1	4	**1**	**2**
20-4	16772.2	1	34	0.82	462	0.82	51	0.82	22	1	154	0.82	712	1	21	**1**	**10**
20-6	12269.6	1	960	0.80	239	0.80	77	**0.80**	**27**	1	4011	0.80	285	1	725	**1**	**165**
20-8	11915.7	0.75	10800	0.72	175	0.72	174	**0.72**	**43**	0.61	10800	0.72	264	1	6101	**1**	**2875**
20-10	6446.6	1	10800	0.75	238	0.75	140	**0.75**	**68**	0.66	10800	0.75	346	**1**	**3108**	1	8047
22-3	50813.7	1	126	0.84	1037	0.84	42	0.84	42	1	307	0.84	1644	1	79	**1**	**60**
22-4	30416.8	1	621	0.79	376	0.79	95	**0.79**	**54**	1	1807	0.79	729	1	545	**1**	**542**
22-6	24468.6	1	9098	0.75	249	0.75	98	**0.75**	**59**	0.70	10800	0.75	360	**1**	**6559**	1	10800
22-8	11778.1	0.73	10800	0.63	336	0.63	125	**0.63**	**90**	0.66	10800	0.63	443	**1**	**4523**	1	5447
22-10	10706.9	0.54	10800	0.56	334	0.56	257	**0.56**	**164**	0.50	10800	0.56	491	0.67	10800	**0.77**	**10800**
22-12	7346.2	0.59	10800	**0.61**	**590**	0.61	812	0.61	723	0.46	10800	0.61	973	0.76	10800	**1**	**10800**
25-3	54414.9	**1**	**59**	0.80	3991	0.80	368	0.80	283	1	150	0.80	4858	1	102	1	74
25-4	19097.4	**1**	**22**	0.75	2513	0.75	175	0.75	75	1	45	0.75	2209	1	39	1	59
25-6	15827.4	0.79	10800	0.60	3070	0.60	378	**0.60**	**131**	0.78	10800	0.60	1609	1	5183	**1**	**3281**
25-8	16652.1	0.47	10800	0.53	3526	0.53	969	**0.53**	**475**	0.37	10800	0.53	1659	**0.60**	**10800**	0.58	10800
25-10	13344.2	0.40	10800	0.47	1363	0.47	389	**0.47**	**271**	0.22	10800	0.47	1273	0.42	10800	0.46	10800
25-12	12815	0.21	10800	0.37	8340	0.37	251	**0.37**	**195**	0.19	10800	0.37	1505	0.24	10800	0.31	10800
30-3	65491.3	1	28	0.77	10800	0.77	904	0.77	272	1	190	0.77	10800	**1**	**18**	1	27
30-4	35820.9	**1**	**95**	0.57	10800	0.57	6451	0.57	6451	1	1277	0.57	10800	1	212	1	349
30-6	33203.3	0.51	10800	0.43	10800	0.43	7491	**0.43**	**3041**	0.29	10800	0.43	10800	**0.60**	**10800**	0.54	10800
30-8	27684.8	0.27	10800	0.37	8469	**0.37**	**1600**	0.37	2229	0.15	10800	0.37	6692	0.29	10800	0.31	10800
30-14	18797.5	0.15	10800	0.36	9217	0.36	1482	**0.36**	**1422**	0.12	10800	0.36	3186	0.14	10800	0.19	10800
40-14	43918.3	0.07	10800	**0.11**	**10800**	**0.1**	**1540**	0.07	10800	0.06	10800	**0.11**	**10800**	0.07	10800	0.09	10800
50-3	74787.6	**1**	**236**	0.71	10800	0.72	10800	0.71	10800	1	2195	0.66	10800	1	2001	0.94	10800
50-4	78766.4	**1**	**6021**	0.62	10800	0.63	10800	0.59	10800	0.44	10800	0.55	10800	0.79	10800	0.83	10800

Table 10 QSAP – 25% out-of-star quadratic matrix density – Comparing different methods – None of the methods stops within the time limit

Instance (m-n)	BFS	BQP LB/BFS	CG+BQPPricing LB/BFS	CG+Heuristic BQPPricing LB/BFS	CG+Heuristic SLTPricing LB/BFS	SLT LB/BFS	CG+SLTPricing LB/BFS	RLT LB/BFS	Outer Approximation LB/BFS
40-3	65784.3	**0.96**	0.66	0.58	0.69	0.86	0.64	0.85	0.82
40-4	59743.9	**0.72**	0.56	0.42	0.42	0.34	0.55	0.45	0.50
40-6	32330.7	0.3	**0.47**	0.26	0.26	0.15	0.44	0.25	0.31
40-8	47935.6	0.13	0.19	0.15	**0.33**	0.09	0.17	0.11	0.16
50-6	77643.1	0.3	**0.51**	0.52	0.23	0.17	0.39	0.22	0.26
50-8	74816.2	0.14	**0.46**	0.44	0.24	0.11	0.25	0.11	0.16
50-14	27759.7	0.13	**0.16**	0.01	0	0.13	0.13	0.12	**0.16**

Table 11 Data association on the MOT16-09 data set – Comparing GUROBI and CG – At least one of the methods stops within the time limit

Instance	BFS	BQP			CG+BQPPricing			SLT			CG+SLTPricing			RLT			CG+RLTPricing		
($\mathcal{T}$-h-d)		LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time
3-20-2	-266.4	-401.7	1.51	10800	-266.8	1	75	-266.4	1	1089	-266.8	1	64	-266.4	1	509	-266.8	**1**	**12**
4-25-2	-486.2	-1064.7	2.19	10800	-486.2	1	72	-486.2	1	6969	-486.2	1	950	-486.2	1	367	-486.2	**1**	**70**
4-25-3	-516.7	-1372.8	2.66	10800	-516.7	1	507	-551.9	1.07	10800	-538.2	1.04	10800	-516.7	1	5498	-516.7	**1**	**78**
5-25-2	-636.1	-1693.5	2.66	10800	-636.1	**1**	**200**	-665.3	1.05	10800	-636.1	1	4674	-636.1	1	5939	-636.1	1	466
5-25-3	-740.1	-1774.6	2.4	10800	-740.1	1	6522	-1477.4	2	10800	-740.1	1	6347	-754.8	1.02	10800	-740.1	**1**	**555**
5-25-4	-783.2	-2024.3	2.58	10800	-783.2	1	7291	-2177.8	2.78	10800	-783.2	1	5676	-806.7	1.03	10800	-783.2	**1**	**400**
6-25-3	-910.1	-2359	2.59	10800	-1365.8	1.5	10800	-2754.3	3.03	10800	-945.1	1.04	10800	-937.6	1.03	10800	-913.1	**1**	**733**
6-25-4	-1014.5	-2817	2.78	10800	-1898.7	1.87	10800	-3883	3.83	10800	-1070.2	1.05	10800	-1064.4	1.05	10800	-1014.5	**1**	**822**
6-25-5	-1110.1	-3087.7	2.78	10800	-2260.7	2.04	10800	-3493	3.15	10800	-2288.8	2.06	10800	-1137.2	1.02	10800	-1110.6	**1**	**821**
7-30-3	-1040.5	-2932.4	2.82	10800	-2509.7	2.41	10800	-3915.9	3.76	10800	-1135.1	1.09	10800	-1077.3	1.04	10800	-1040.8	**1**	**1304**
7-30-4	-1172.7	-3538.3	3.02	10800	-3038	2.59	10800	-4499	3.84	10800	-1267	1.08	10800	-1236.6	1.05	10800	-1172.8	**1**	**1374**
7-30-5	-1336.9	-3753	2.81	10800	-3496.2	2.62	10800	-5062.4	3.79	10800	-1456.3	1.09	10800	-1417.3	1.06	10800	-1336.9	**1**	**1718**
8-30-3	-1229.1	-3379.1	2.75	10800	-2261.9	1.84	10800	-4567.4	3.72	10800	-1741.7	1.42	10800	-1303.5	1.06	10800	-1229.7	**1**	**4293**
8-30-4	-1497.8	-4778.2	3.19	10800	-3806.2	2.54	10800	-5646.2	3.77	10800	-2002	1.34	10800	-1638.2	1.09	10800	-1497.8	**1**	**4252**
8-30-5	-1619.4	-5207.5	3.22	10800	-4373.9	2.7	10800	-6429.1	3.97	10800	-2902	1.79	10800	-1892.7	1.17	10800	-1632.4	**1.01**	**5088**

Table 12 Data association on the MOT16-09 data set – Comparing GUROBI and CG – None of the methods stops within the time limit

Instance	BFS	BQP		CG+BQPPricing		SLT		CG+SLTPricing		RLT		CG+RLTPricing	
($\mathcal{T}$-h-d)		LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS
9-35-4	-1790.8	-5601.5	3.13	-5377.9	3	-6569.6	3.67	-4230.2	2.36	-2026.3	1.13	-1792.1	**1.00**
9-35-5	-1974.7	-6624.1	3.35	NA	NA	-7740.1	3.92	-5085.4	2.58	-2358.4	1.19	-1988.9	**1.01**
9-35-6	-2065.4	-7647.6	3.7	NA	NA	-8519.1	4.12	-5134.7	2.49	-2535.4	1.23	-2150.4	**1.04**
10-35-4	-1971.7	-6472	3.28	NA	NA	-7357.6	3.73	-5469.1	2.77	-2288.2	1.16	-2073.7	**1.05**
10-35-5	-2150.6	-8116.3	3.77	NA	NA	-8663.3	4.03	-5700.6	2.65	-2665.1	1.24	-2346.1	**1.09**
10-35-6	-2405.1	-9273.5	3.86	NA	NA	-9792.8	4.07	-6824.3	2.84	-2988.5	1.24	-2621.2	**1.09**
10-35-7	-2569.7	-10134.5	3.94	NA	NA	-10681.3	4.16	-7544.9	2.94	-3252.3	1.27	-2833.3	**1.10**
11-35-4	-2167.5	-7494.8	3.46	-6969.1	3.22	-8306.2	3.83	-5996.4	2.77	-2650.4	1.22	-2412.8	**1.11**
11-35-6	-2687	-10773.8	4.01	-8849.4	3.29	-11098.4	4.13	-8436	3.14	-3440.9	1.28	-3074.5	**1.14**
11-35-9	-3141.3	-12752.7	4.06	-9512.7	3.03	-13306.3	4.24	-9496.8	3.02	-4107.9	1.31	-3540.1	**1.13**
12-40-4	-2467.1	-8257	3.35	-8606.6	3.49	-9107.1	3.69	-8168.3	3.31	-2915.7	1.18	-2862.4	**1.16**
12-40-6	-3044.7	-11709.1	3.85	-10676.2	3.51	-12016.3	3.95	-10281.8	3.38	-3762.6	1.24	-3520.2	**1.16**