Les Cahiers du GERAD ISSN: 0711–2440

A dual bounding framework for binary quadratic combinatorial optimization

M. Bayani, B. Rostami, Y. Adulyasak, L.-M. Rousseau

G–2021–43

August 2021

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

A dual bounding framework for binary quadratic combinatorial optimization

Mahdis Bayani [a, b, c]

Borzou Rostami [c, d, e]

Yossiri Adulyasak [a, f]

Louis-Martin Rousseau [c, b]

[a] *GERAD, Montréal (Qc), Canada, H3T 1J4*

[b] *Département de mathématiques et de génie industriel, Polytechnique Montréal, Montréal (Qc), Canada, H3C 3A7*

[c] *CIRRELT, Montréal (Qc), Canada, H3T 1J4*

[d] *Lazaridis School of Business and Economics, Wilfrid Laurier University, Waterloo, (On), Canada, N2L 3C5*

[e] *CERC–Data Science for Real-time Decision-making, Polytechnique Montréal, Montréal (Qc), Canada, H3C 3A7*

[f] *HEC Montréal, Montréal (Qc), Canada, H3T 2A7*

mahdis.bayani@polymtl.ca
brostami@wlu.ca
yossiri.adulyasak@hec.ca
louis-martin.rousseau@cirrelt.net

August 2021
Les Cahiers du GERAD
G–2021–43

Abstract : Binary quadratic programming (BQP) is a class of combinatorial optimization problems comprising binary variables, quadratic objective functions and linear/non-linear constraints. In this paper, we propose a unified framework to reformulate any BQP problem with linear constraints to a new BQP problem defined on a graph. This framework relies on the concept of stars in the graph and partitioning the quadratic costs into 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 in which the quadratic component of the problem is dealt with in the column generation master problem and in its subproblem. Results suggest that the framework outperforms the state-of-the-art solver in almost all the instances having zero out-of-star interactions both in terms of dual bound and computational time.

Keywords: Binary quadratic programming, combinatorial optimization, column generation, semi-assignment problem, multiple object tracking problem

Acknowledgements: We wish to gratefully thank Mitacs Accelerate Program for providing funding for this project. In addition, the second author gratefully acknowledges funding provided by the Canadian Natural Sciences and Engineering Research Council (NSERC) under the Discovery Grant RGPIN–2020–05395.

1 Introduction

Binary quadratic programming (BQP) is a large class of combinatorial optimization problems that arose 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:

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

where $X \subseteq \{0, 1\}^m$ is the set of feasible binary vectors and $\mathcal{E} = E \times E$.

Many quadratic combinatorial optimization problems can be naturally formulated in this fashion. Some important examples include the quadratic assignment problem (Cela 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 2018b, Rostami et al. 2018a), 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, the quadratic assignment problem, and the quadratic shortest path problem (Assad and Xu 1992, Rostami et al. 2018a), while their linear counterparts are polynomially solvable.

One way to deal with the challenges of BQP is to employ data-driven methods to identify patterns and structures in the objective function, constraint matrices, or instances (Punnen et al. 2017, Bettiol et al. 2020). In this paper, we employ such a technique to exploit the structure of the quadratic cost matrix $\boldsymbol{q}$ and propose a new reformulation for BQP with exponentially many variables. This new formulation is then used to distinguish an important category of BQP with a specific cost structure where the quadratic interaction between non-adjacent edges is zero. The relaxation of the obtained linear programming (LP) formulation can be efficiently tackled by the proposed column generation (CG) approach.

1.1 Literature review

One of the most natural ways to solve BQP problems with an exact method is to linearize the quadratic terms of the model and solve the resulting mixed-integer linear program (MILP) using state-of-the-art solvers. 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 Forrester (2005) and Sherali and Smith (2007) provide different reduced-size MILP reformulations of BQP, while Hahn et al. (2012) and Rostami and Malucelli (2015) develop the reformulation-linearization technique (RLT), which generally provides stronger MILP reformulations.

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 as an extension of MILP reformulations, non-negativity constraints are replaced by positive semi-definite

constraints (Helmberg et al. 2000, Oustry 2001). In quadratic reformulation, 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. 2018c). 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 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 BQP using a Lagrangian-based heuristic method. Examples of using decompositions to explore bounds for a specific BQP problem can be observed in the quadratic knapsack problem and 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 an 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 an MILP with an exponential number of cycles as variables and solve the relaxation using a CG. Another application that has been studied through column generation is the adjacent-only quadratic minimum spanning tree problem (AQMSTP). Pereira et al. (2015) suggest a reformulation of AQMSTP and employ row and column generation to evaluate the lower bounds. 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).

In recent years, employing data-driven methods to efficiently solve large-scale optimization problems has drawn researchers' attention. Such methods aim to tackle real-life problems by identifying the patterns and structures inherent in these problems. There are different extents of data-driven approaches. Exploring the constraint matrix structures, finding important patterns in a specific problem class, and revealing the structure of a data instance related to a large-scale problem, are among the notable techniques. 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 an MILP leads to Dantzig-Wolfe (DW) decomposition, Lagrangian relaxation, and branch-and-price (Bergner et al. 2015, Khaniyev et al. 2018). By exploiting and rearranging the structure of the constraint's matrix in general MILP, Bergner et al. (2015) provide a computational proof-of-concept to show that the DW reformulation can be automated and forced to all MILPs. To our knowledge, there are few methodological studies concerning structures in BQP problems. Bettiol et al. (2020) 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 which 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 literature explores the linearizability of the quadratic cost matrices of BQP. Linearizability is verified when there exists a linear cost matrix for the problem whose associated cost is always equal to the quadratic cost of that problem on all feasible space. 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 work in this area, Hu and Sotirov (2018a) represent 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.

1.2 Main contributions

Our main contributions are summarized as follows:

- We propose a unifying framework for general BQP which relies on detecting the patterns of data instances by exploiting the quadratic cost matrix $\boldsymbol{q}$.
- Based on the obtained specific structure, we present a CG algorithm to provide valid dual bounds for the reformulation.
- We extract a large class of BQP problems whose special quadratic matrix structure derives the greatest benefit from our dual bounding methodology.
- To demonstrate the flexibility of our modeling framework, we consider three BQP problems whose formulations have special structures that 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.
- To evaluate our solution method, we perform extensive computational experiments on instances of the quadratic semi-assignment problem, as well as a problem of data association for multiple object tracking (MOT) in computer vision.

The rest of the paper is organized as follows. In Section 2, we present our generic framework by means of star-based reformulation and introduce the class of BQP problems that motivated this study. In Section 3, we propose a column generation method to solve the reformulation. Section 4 is dedicated to presenting and reformulating two BQP problems, QSAP and MOT, as our illustrative examples. Finally, in Section 5, we explain our test sets and display the computational results from the introduced problems.

2 Proposed framework

Exploring data to find possible structures and patterns is a new strategy emerging to tackle difficult optimization problems (Khaniyev 2018). Inspired by this, in the following section we first develop a canonical form of the generic mathematical model of BQP and argue that several important applications in the literature are captured by this model. Second, we investigate a large class of BQP problems with special properties and elaborate on how this class takes advantage of our proposed framework. Note that the matrices and vectors are shown in bold text throughout the paper.

2.1 Reformulation

In order to tackle the complexity of BQP model in (1) using decomposition, we introduce the concept of star formulation (Pereira et al. 2013). Consider the BQP problem defined on graph $G = (V, E)$ in (1). Without loss of generality, we assume that $V = N \cup H$ where N can potentially be an empty set. 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 one endpoint in H, considering that when the set N is empty, both endpoints of all edges of the related graph are 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:

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

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

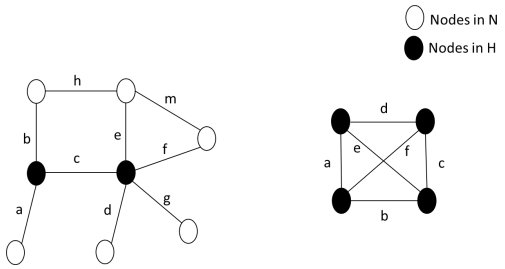


Figure 1: Example of graphs with adjacent edges in A. Graph on left: $A = \{a, b, d, e, f, g\}$, $\mathcal{A} = \{(a, b), (d, e), (d, f), (d, g), (e, f), (e, g), (f, g)\}$ Graph on right: $A = \{a, b, c, d, e, f\}$, $\mathcal{A} = \{(a, b), (a, d), (a, e), (a, f), (b, c), (b, e), (b, f), (c, d), (c, e), (c, f), (d, e), (d, f)\}$

We introduce star-shaped sub-graph s based on above definitions. For each $v \in H$ we define a star s centered at node v as any subset of $\delta(v)$ and let S^v be the set of all stars centered at node v. 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, we can consider $s = \{e, f, g\}$ as a star centered at the common endpoint of these edges.

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-stars linear costs, out-of-stars linear costs, in-stars quadratic costs, and out-of-stars quadratic costs. More precisely, given 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 $\bar{\boldsymbol{x}}^1 \subseteq \{0, 1\}^{m_1}$ and $\bar{\boldsymbol{x}}^2 \subseteq \{0, 1\}^{m-m_1}$ and where $\bar{\boldsymbol{x}}^1 \in A$ and $\bar{\boldsymbol{x}}^2 \in E \setminus A$. Therefore, the objective function of (1) at $\bar{\boldsymbol{x}}$ can be written as:

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

Based on this representation of the objective function and the fact that each star $s \in S$ consists of a subset of pairs in $\mathcal{A}$, we can reformulate this 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 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. By preserving the definition of variable $\boldsymbol{x} \in X$, we propose the following star-based reformulation:

$$\min \quad \sum_{s \in S} C_s \lambda_s + \sum_{e \in E \setminus A} c_e x_e + \sum_{(e,f) \in \mathcal{E} \setminus \mathcal{A}} q_{ef} x_e x_f \tag{3}$$

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

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

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

The objective function of the reformulation minimizes the total cost of the problem and consists of three different parts. The first part corresponds to the cost of the star, including the linear costs of the edges inside the star s 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, while the third quadratic cost is associated with the interaction between pairs of non-adjacent edges. 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 $e \in A$, there exist a parameter $b_{es} \in [0, 1]$ such that $x_e = \sum_{s \in S} b_{es} \lambda_s$ and $\sum_{s \in S} b_{es} = 1$ (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).

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. This will be further explained in Section 2.2.

As we observe in the right graph of Figure 1, a given set of star centers (H) can potentially comprise the whole node set of the graph (V) in some problems ($H = V$). Intuitively, for BQP problems with this property, set A contains all the graph's edges. To illustrate this, consider the quadratic minimum spanning tree (QMST) problem (Assad and Xu 1992), in which the quadratic cost comprises the interactions between all pairs of edges. In this problem, the set H is equal to all nodes of the graph ($H = V$). Thus, in the reformulated model of QMST, the objective function consists of 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, Rostami et al. 2018d) which is 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.

2.2 Motivation through adjacent-only class

Although the model presented in Section 2.1 is a generic reformulation of BQP, special structure assumptions for quadratic matrix $\boldsymbol{q}$ offer promising properties for solving the problem. This structure is our principal motivation behind the proposed reformulation scheme.

There are a variety of real-life applications that can be modeled as BQP problems, where interactions appear only between adjacent edges. In these 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. The third term of the objective function in (3) is therefore eliminated in the star-based reformulation, and the stars interact with one other only through linear costs. Thus, the new formulation for this particular class is an integer linear problem that is less complex to solve than the generic model. 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 matrix $\boldsymbol{q}$.

Numerous problems in the literature on BQP satisfy these assumptions. Some notable examples include the adjacent-only quadratic minimum spanning tree problem (Pereira et al. 2013, Pereira and da Cunha 2020), the quadratic travelling salesman problem (Fischer 2014, Punnen et al. 2017), the adjacent quadratic assignment problem (Fischer et al. 2009), the adjacent quadratic shortest path problem (Hu and Sotirov 2018b) and variants of the correlation clustering and the modularity maximization problems (Bonizzoni et al. 2008, Aloise et al. 2010, Yarkony et al. 2020). More details about this class of BQP problems, modeling and solution methodology are presented in Section 4.

3 Solution methodology

In this section, we propose the general idea of solving the presented generic reformulation of (3)–(6) to handle all BQP problems. It is observed that the use of star representation in the proposed reformulation results in an exponential number of variables $\lambda_s, s \in S$. Column generation algorithm has been introduced in the literature as a promising solution method for dealing with this problem. 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 suproblem will be added to the RMP and the updated RMP will be solved in the new iteration.

Since the generic reformulation (3)–(6) 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 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 (Rostami et al. 2018b). 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. 2020) 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. Here, we follow linearization techniques to transform the quadratic objective function to 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 feasible subset of stars to obtain the following RMP for the LP relaxation of the problem:

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

$$\text{s.t.} \quad (\boldsymbol{x}, \boldsymbol{\lambda}) \in \bar{\mathcal{F}}(\boldsymbol{x}, \boldsymbol{\lambda}) \tag{8}$$

$$(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{P}(\boldsymbol{x}, \boldsymbol{y}) \tag{9}$$

$$\boldsymbol{y} \in \mathbb{R}_+^{|(e,f) \in \mathcal{E} \setminus \mathcal{A}|} \tag{10}$$

$$\boldsymbol{\lambda} \in [0,1]^{|\bar{S}|} \tag{11}$$

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 (7). 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 to exact solution is NP-hard, intuitively adding columns with negative reduced cost without solving the subproblem to optimality can be a promising alternative when applying CG algorithm. However, to provide a valid dual bound, 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, a number of 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.

4 Applications of the model and the solution strategy

The objective of this section is to demonstrate how our modeling 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 have special structures that 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) in addition to two problems from the adjacent-only class of BQP to outline the computational advantages of our method 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 (3) to (6) 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_{ij} associated with the assignment of client $i \in N$ to server $j \in H$ and there is a quadratic cost q_{ijkl} 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. The QSAP has 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 QSAP (Saito et al. 2009). By transferring 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, and by recalling the concepts of Section 2.1, we rewrite the linear cost as $c_e, e \in A$, the quadratic cost as q_{ef} and the binary decision variable as x_e. Hence, denoting by variable x_e equals 1 if the client i is assigned to server j, and 0 otherwise, the BQP problem of semi-assignment on graphs is formulated as follows:

$$\min \quad \sum_{e \in A} c_e x_e + \sum_{(e,f) \in \mathcal{E}} q_{ef} x_e x_f \tag{12}$$

$$\text{s.t.} \quad \sum_{e \in \delta(i)} x_e = 1 \qquad \forall i \in N \tag{13}$$

$$x_e \in \{0, 1\} \qquad \forall e \in A. \tag{14}$$

The QSAP is known to be NP-hard, and solving it even for small size instances is very time-consuming (Sahni and Gonzalez 1976). 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 star-based model (3) to (6). 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 parameter $D_{is} \in \{0, 1\}$ 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 (7)–(11), the linearized RMP can be expressed as follows:

$$\text{[RMP-QSAP]:} \quad \min \quad \sum_{s\in\bar{S}} C_s\lambda_s + \sum_{(e,f)\in\mathcal{E}\setminus\mathcal{A}} q_{ef}y_{ef} \tag{15}$$

$$\text{s.t.} \quad \sum_{s\in\bar{S}} B_{js}\lambda_s \leq 1 \qquad \forall j \in H \tag{16}$$

$$\sum_{s\in\bar{S}} D_{is}\lambda_s = 1 \qquad \forall i \in N \tag{17}$$

$$\sum_{s\in\bar{S}} D_{es}\lambda_s = x_e \qquad \forall e \in A \tag{18}$$

$$(\boldsymbol{x},\boldsymbol{y}) \in \mathcal{P}(\boldsymbol{x},\boldsymbol{y}) \tag{19}$$

$$0 \leq x_e \leq 1 \qquad \forall e \in A \tag{20}$$

$$\boldsymbol{y} \in \mathbb{R}_+^{|(e,f)\in\mathcal{E}\setminus\mathcal{A}|} \tag{21}$$

$$\boldsymbol{\lambda} \in [0,1]^{|\bar{S}|} \tag{22}$$

where y_{ef} is the linearization variable. Constraints (16) impose that at most one star can be chosen among all the stars centered at j. Constraints (17) are the set partitioning linking constraints, which impose that each client must be included in exactly one star. Constraints (18) are the linking constraints, and enforce that if an edge is selected in the optimal solution, then it can be included in only one selected star.

4.1.2 Column generation

Starting from a feasible set of possible stars 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 per 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 negative reduced cost to be added.

Let π_j, ρ_i and γ_{ij} be the dual solutions corresponding to constraints (16)–(18), 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 based on edges. The star with minimum reduced cost can be found by solving the following pricing subproblems on graphs, one for each server j:

$$\min \quad \sum_{e\in\delta(j)} (c_e - \rho_e - \gamma_e)z_e + \sum_{e,f\in\delta(j):f>e} q_{ef}z_ez_f - \pi_j \tag{23}$$

$$\text{s.t.} \quad z_e \in \{0,1\} \qquad \forall e \in \delta(j) \tag{24}$$

where binary decision variable $z_e, e \in \delta(j)$ 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). In order to obtain an assignment to minimize the total completion time for all clients, $\sum_{i\in N} CT_i$, the objective function on graphs is:

$$\min \quad \sum_{e\in A} c_e x_e + \sum_{(e,f)\in\mathcal{A}} q_{ef} x_e x_f \tag{25}$$

where the constraints of the problem are exactly the same as (13) and (14) and the equation below holds:

$$q_{ef} = c_e + c_f. \qquad \forall (e,f) \in \mathcal{A} \tag{26}$$

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 lends it to the class of problems described in Section 2.2. We denote this problem by adjacent-only QSAP (AQSAP) in the rest of the paper. 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:

$$\min \quad \sum_{s\in S} C_s \lambda_s \tag{27}$$

$$\text{s.t.} \quad \sum_{s\in S} B_{js} \lambda_s \leq 1 \qquad \forall j \in H \tag{28}$$

$$\sum_{s\in S} D_{is} \lambda_s = 1 \qquad \forall i \in N \tag{29}$$

$$\boldsymbol{\lambda} \in [0,1]^{|S|} \tag{30}$$

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:

$$\min \quad \sum_{e\in\delta(j)} (c_e - \rho_e) z_e + \sum_{e,f\in\delta(j):f>e} q_{ef} z_e z_f - \pi_j \tag{31}$$

$$\text{s.t.} \quad z_e \in \{0,1\} \qquad \forall e \in \delta(j) \tag{32}$$

where $\pi_j, j \in H$ and $\rho_i, i \in N$ are the duals associated with constraints (28) and (29), 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 CG, 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. To further elaborate, MOT on a sequence of frames (from t_0 to t_3) is depicted in Figure 2. It shows that the output of the data association algorithm is recognizing four tracks (people) and two false detections, represented by crossmarks. Data association algorithms can be categorized as online or offline algorithms (Emami et al. 2018). Here we consider the offline data association case.

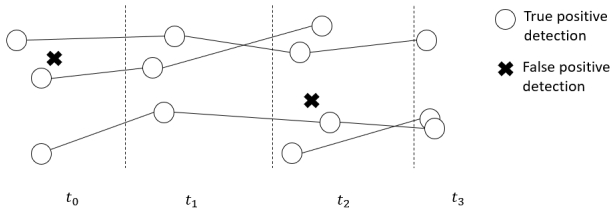


Figure 2: Graphical representation of data association in MOT

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 in a video sequence N and all possible tracks (target people) H. Similar to Section 2.1, 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 and k 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. Depending on the unary and pair-wise probabilities the cost can be negative or positive, 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:

$$\min \quad \sum_{e \in A} c_e x_e + \sum_{(e,f) \in \mathcal{A}} q_{ef} x_e x_f \tag{33}$$

$$\text{s.t.} \quad \sum_{e \in \delta(i)} x_e \leq 1 \qquad \forall i \in N \tag{34}$$

$$\sum_{e \in \delta^t(j)} x_e \leq 1 \qquad \forall j \in H, \quad \forall t \in T \tag{35}$$

$$x_e \in \{0, 1\} \qquad \forall e \in A \tag{36}$$

where constraints (34) are needed to mandate that more than one track assignment is not possible for every detection i. Constraints (35) 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 (34), any type of potential side constraint can be added to this primary model, depending on the structure of the problem (Dehghan and Shah 2017). Here, 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 (35) 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 the optimal one 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, (3) to (6), 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 bellow:

$$\text{[RMP-MOT]:} \quad \min \quad \sum_{s\in\bar{S}} C_s\lambda_s \tag{37}$$

$$\text{s.t.} \quad \sum_{s\in\bar{S}} \lambda_s \leq h \tag{38}$$

$$\sum_{s\in\bar{S}} D_{is}\lambda_s \leq 1 \qquad \forall i \in N \tag{39}$$

$$\boldsymbol{\lambda} \in [0,1]^{|\bar{S}|} \tag{40}$$

This model consists of one set of star-only constraints (38) enforcing the maximum number of tracks, and one set of coupling constraints (39) 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 dual variables associated with constraints (38) and (39), respectively. 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:

$$\min \quad \sum_{e\in\delta(j)} (c_e - \rho_e)z_e + \sum_{e,f\in\delta(j):f>e} q_{ef}z_ez_f - \pi \tag{41}$$

$$\text{s.t.} \quad \sum_{e\in\delta^t(j)} z_e \leq 1 \qquad \forall t \in T \tag{42}$$

$$z_e \in \{0,1\} \qquad \forall e \in \delta(j) \tag{43}$$

where constraint (42) 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 dependant on the star centers; the centers can be realized identical. 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 the quadratic interaction between $e_2 = \{i, l\}$ and $f_2 = \{k, l\}$, $q_{e_2 f_2}$.

Note that the pricing subproblem in this case is a constrained BQP problem that can be solved using both exact and approximation algorithms. 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 quadratic semi-assignment and MOT in terms of dual bound and computing time. We attempt to answer this fundamental question through our experiments: For which types of BQP problems it is worthwhile to perform star-based reformulation and the proposed CG alongside a state-of-the-art solver, instead of solving the original BQP formulation directly with the solver?

GUROBI version 9.0.1 is chosen as our benchmark mixed-integer programming (MIP) solver and through our experiments we deduce that for what types of BQP problems GUROBI alone can achieve better results and for what types using the proposed framework alongside the solver can improve the results. To this end, we first solve the primary model of BQP and its linearized variants using GUROBI. Then, we apply our reformulation framework and solve the pricing of CG with the same BQP model or linearized variants using GUROBI while the master problem is also handled by the solver. This leads to different strategies to solve the original BQP model and the reformulated model. We considered 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 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. These two criteria are described below:

Dual bound performance profile: The first set of graphs are based on the dual bound quality regardless of 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 are 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 could not converge in our desired time, we have information about the intermediate quality of dual bound in each iteration at the expense of slightly more computations (Lübbecke and Desrosiers 2005). Nonetheless, to attain a valid 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 in a branch-and-bound tree. However, we compute primal bounds by applying a trivial heuristic to the solutions of 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 details.

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 CG. Recent 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, although we implemented BoxPen, in-out separation, and interior-point stabilization techniques, the results were improved for some methods of CG, but not for all of them (Desaulniers et al. 2006, Rousseau et al. 2007). Hence, we have decided to keep the unstabilized results in our reports. In addition, since our main focus is on the application of the standard CG algorithm, further specialized CG enhancements were not performed. However, one can test different techniques to accelerate the CG to solve the proposed star-reformulation 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 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 experiments

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 our framework, we run the experiments on 5 randomly-generated data sets with different sparsity of 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 real-life applications of BQP, the quadratic matrix is mostly very sparse. 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 (23) and (31) 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, the pricing has to be solved to optimality. 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 standard linearization technique (SLT) to construct the set $\mathcal{P}(x, y)$ in constraints (19). The concept of standard linearization is presented in Appendix A.

We now provide a brief description of the different methods we used to solve instances of the QSAP and the AQSAP.

GUROBI+BQP: The BQP model (12)-(14) solved by GUROBI.

GUROBI+SLT: Linearized reformulation (using SLT) of the BQP model (12)–(14) solved by GUROBI.

CG+BQPPricing: CG algorithm for the model (15)–(21) where the UBQP pricing is solved by GUROBI.

CG+SLTPricing: CG algorithm for model (15)–(21) where the standard linearization of pricing subproblem is solved by GUROBI.

CG+HeuristicBQPPricing: CG algorithm for model (15)–(21) where the UBQP pricing is solved by using the hybrid heuristic method described above.

CG+HeuristicSLTPricing: CG algorithm for model (15)–(21) where the UBQP pricing is solved using the hybrid heuristic in which the exact iteration solves the standard linearized pricing subproblem by GUROBI.

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 and CG methods in terms of dual bounds 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 which 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.

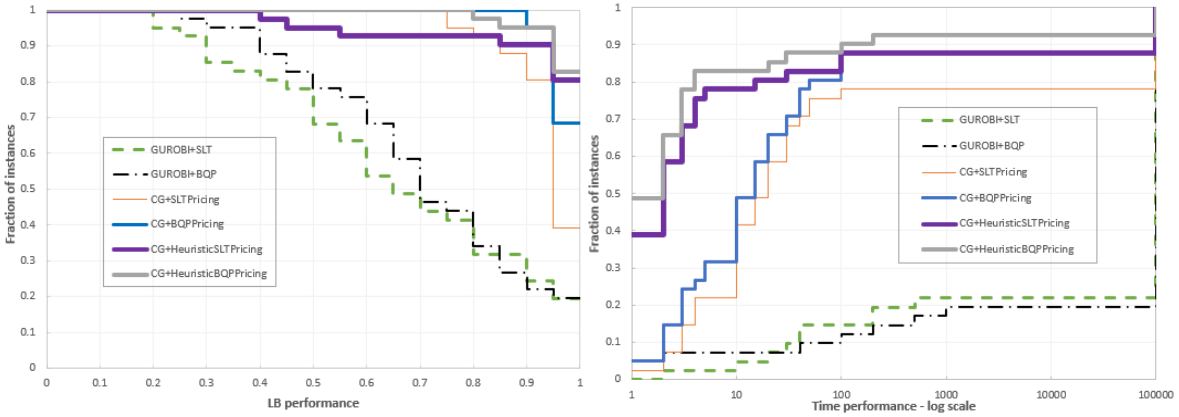
AQSAP results

The results for the AQSAP instances are given in Figure 3. According to LB performance analysis in Figure 3a, CG hybrid methods have the most wins in terms of providing the best LB (83% and 80% for CG+HeuristicBQPPricing and CG+HEURISTICBQPPricing, respectively) among all the methods, though CG+BQPPricing outperforms the hybrid methods in a few quantiles. For instance, all the instances are solved by CG+BQPPricing to 90% of the best LB, while only 95% of them could reach this ratio when solved by CG+HeuristicPricing. We also observe that in general, GUROBI solves the BQP model (GUROBI+BQP) slightly better than it solves the linearization model (GUROBI+SLT). If we seek a method that is able to 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 non-CG GUROBI methods. Finally, looking for a method with 100% performance, 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+Heuristic-

BQPPricing method. It also demonstrates the superiority of this method in all other quantiles. Comparing the GUROBI methods with CG, the GUROBI methods are inefficient at finding the best LBs, to such an extent that even the best GUROBI method obtains the best time performance in just 2% of cases. In addition, using the GUROBI+SLT method, only 15% of instances and using the GUROBI+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 non-CG GUROBI 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 in most instances.



(a) Dual bound performance profile

(b) Time performance profile

Figure 3: Performance profiles for AQSAP instances

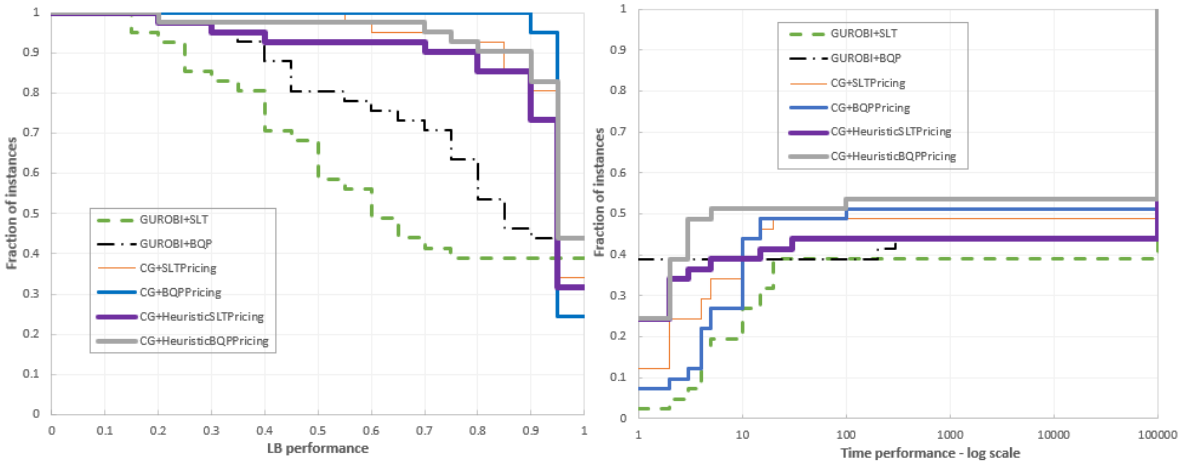
QSAP results

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. CG+HeuristicBQPPricing has the greatest number of instances with the best LB among all the methods (44%); however, this time CG+BQPPricing outperforms the hybrid counterpart in more intervals compared to the AQSAP. The performance of the GUROBI methods is almost the same as before. GUROBI+BQP is superior to GUROBI+SLT while it still has a huge LB performance gap compared to all the CG methods.

Nevertheless, in the next LB performance profile represented in Figure 5a, both GUROBI methods outperform CGs in the interval $[0.95, 1]$. Comparing the CG methods 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 both GUROBI 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 GUROBIs. However, in the most dense quadratic matrix in Figure 7a, the CG methods still outperform GUROBI 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. 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. For instance, in the case of 20% out-of-star density

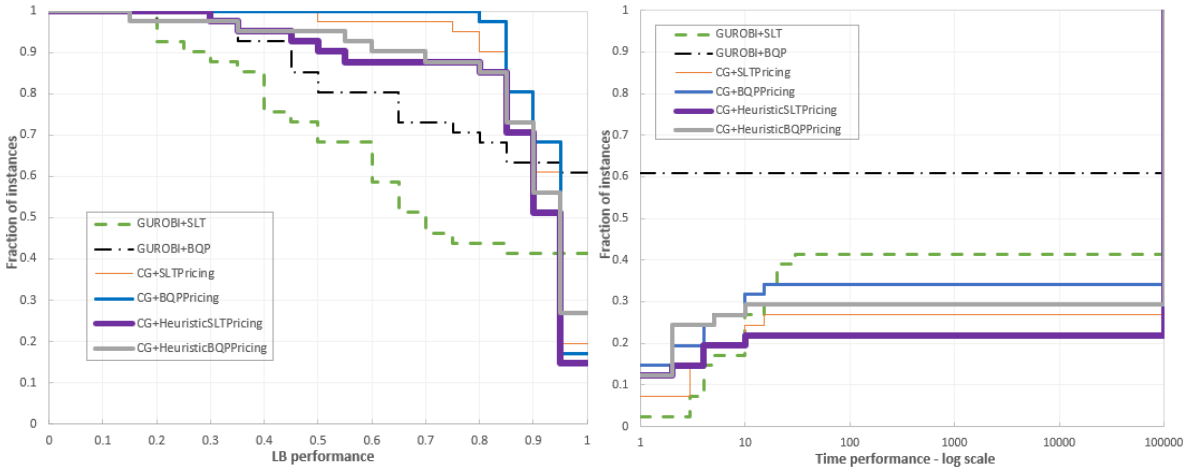
and for a minimum requirement of 60% LB performance, we would opt for CG+BQPPricing as all the instances reach this when they are solved using this method. In contrast, the best GUROBI method, GUROBI+BQP, satisfies this requirement in only 82% of problem instances.



(a) Dual bound performance profile

(b) Time performance profile

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



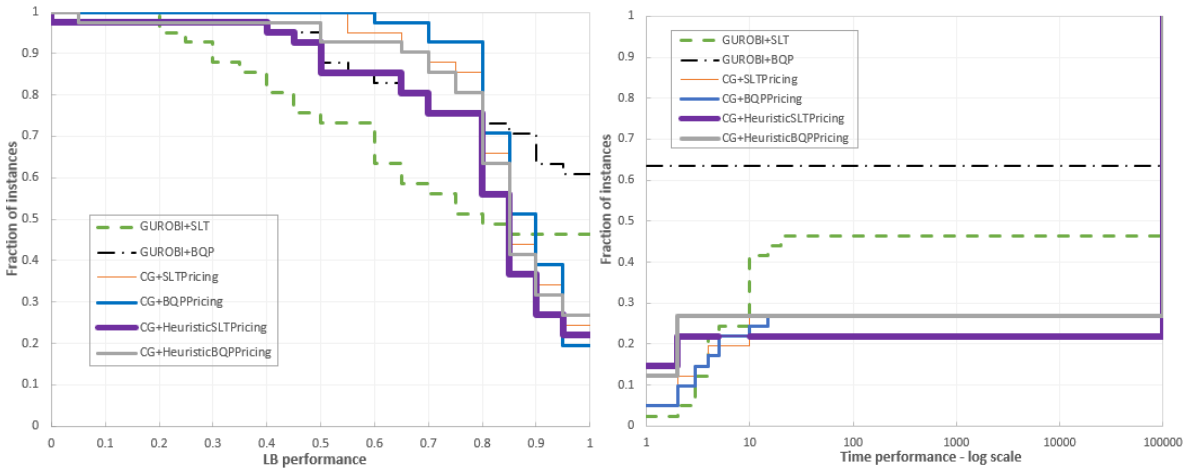
(a) Dual bound performance profile

(b) Time performance profile

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

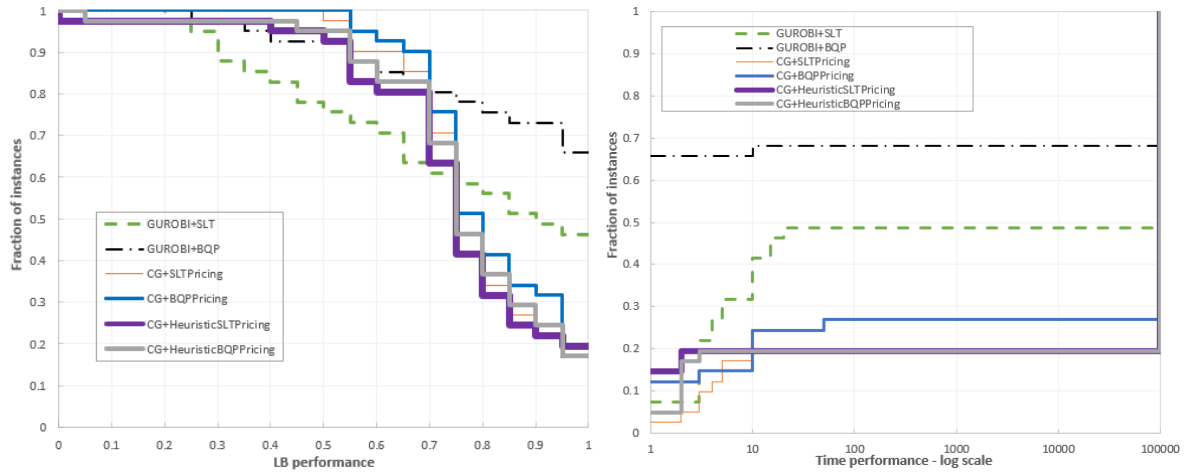
The second analysis is related to the time performance profiles. As shown in Figure 3b, all CG methods perform better than GUROBI methods in all quantiles when it comes to AQSAP. Starting from the plot associated with time performance in Figure 4, when we add out-of-star interactions, the GUROBI methods move upside of the figure and get closer to the CG methods. This is due to the fact that the speed of the GUROBI methods increases when instances consist of some out-of-star interactions in addition to the in-star interactions. Still, although the most wins in Figure 4b belong to the GUROBI+BQP method (39%), all CG algorithms perform better than GUROBIs in all other intervals. Comparing GUROBI methods together, similar to LB analysis, GUROBI+BQP outperforms GUROBI+SLT in terms of time and it is still 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 instances could not obtain the best LB when they are solved by the best CG method (CG+HeuristicBQPPricing), while this number is 57% for GUROBI+BQPPricing as the best GUROBI 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 more dense matrices, due to their speed improvement. Regarding the vertical lines for the GUROBI+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 40% of instances, whereas it obtained the best LB in the best time for the other 60% 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, their advantages are weakened in more dense quadratic matrices. As an example, Figure 7b shows that the CG+HeuristicBQPPricing method outperforms CG+BQPPrcing in only 8% of cases.



(a) Dual bound performance profile **(b) Time performance profile**

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



(a) Dual bound performance profile **(b) Time performance profile**

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

Looking at the trend in all of the time performance on Figures 4b to 7b, we observe that the GUROBI methods outperform CG methods when the quadratic cost matrix of QSAP is more dense. For instance, in Figure 7b we have 25% out-of-star quadratic costs and still in approximately 68% of the instances, the best GUROBI method (GUROBI+BQP) 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 25% 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 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 (100000) to its time performance.

Considering both performance profile analyses and the instance-by-instance details of Appendix B, it can be deduced that the proposed star-based reformulation and CG, even with a basic implementation, outperforms GUROBI in terms of computing time and LB for most of the instances of the AQSAP. Additionally, in a large number of 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 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 BQP consist of sparse quadratic matrices, 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.

5.2 MOT experiments

In addition, 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 which 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, detection of objects and the precision of estimating unary and pair-wise cost, 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 an MIP solver, computing real accurate costs are 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 distance between the detections.

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 (33) 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 C.11 and126C.12 of Appendix B.

Similar to the semi-assignment problem, here we evaluate the efficiency of 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:

GUROBI+BQP: The BQP model (33)–(36) solved by GUROBI.

GUROBI+SLT: Linearized reformulation (using SLT) of the BQP model (33)–(36) solved by GUROBI.

GUROBI+RLT: Linearized reformulation (using RLT) of the BQP model (33)–(36) solved by GUROBI.

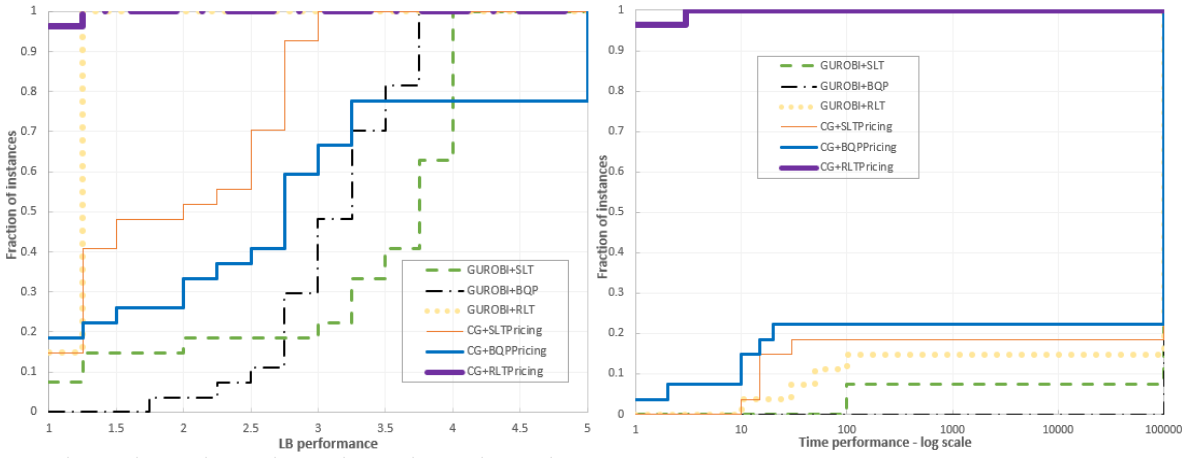
CG+BQPPricing: CG algorithm for model (37)–(40) where constrained BQP pricing is solved by GUROBI.

CG+SLTPricing: CG algorithm for the model (37)–(40) where the standard linearization of the pricing subproblem is solved by GUROBI.

CG+RLTPricing: CG algorithm for the model (37)–(40) where the linearized pricing subproblem using RLT is solved by GUROBI.

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 GUROBI+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 C.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 (GUROBI+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 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.



(a) Dual bound performance profile

(b) Time performance profile

Figure 8: Performance profiles for MOT problem

Although according to Figure 8a the ratio of obtained LB by GOROBI+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 GOROBI+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 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 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 star-based reformulation and CG methodology computationally outperform the GUROBI solver in obtaining LB for the data association problem in MOT. Moreover, similar to CG+RLTPricing, which outperforms GUROBI+RLT, the other CG methods outperform their GUROBI counterparts both in terms of LB and computational time. Evidently, 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 the optimal solution for the vast majority of instances.

6 Conclusion

In this study, we developed a reformulation framework and solution methodology for 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. Using our proposed framework, we studied the special structure of the quadratic matrix in a large class of BQP problems which resulted in a huge improvement in solving these problems. We proposed different column generation methods to provide dual bounds for BQP. To evaluate the efficiency of our framework, we perform extensive experiments on two BQP problems with different characteristics: the quadratic semi-assignment problem and the multiple object tracking problem. We compare the lower bound and computing time of the proposed star reformulations and CG methods with a state-of-the-art MIP solver on instances of 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 CG methods still obtain promising results in some instances. Particularly on larger instances, where all the tested methods meet the time limit, CG methods outperform the MIP solver in many cases.

Appendices

A Linearization

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

$$y_{ef} \geq x_e + x_f - 1 \qquad \forall (e,f) \in \mathcal{A} \tag{A.1}$$
$$y_{ef} \leq x_e \qquad \forall (e,f) \in \mathcal{A} \tag{A.2}$$
$$y_{ef} \leq x_f \qquad \forall (e,f) \in \mathcal{A} \tag{A.3}$$
$$y_{ef} \geq 0 \qquad \forall (e,f) \in \mathcal{A} \tag{A.4}$$

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

Reformulation linearization technique (RLT) is another type of linearization. For MOT BQP formulation and the pricing of its star-based reformulation, we could exploit a tighter linearization which is similar to RLT. Considering (35) and multiplying it once by $x_f, \forall f \in \delta^{t'}(j) : t' \neq t$ and then generating the same constraint for x_e, we obtain following strong valid inequalities instead of SLT constraints, (A.2) and (A.3):

$$\sum_{e \in \delta^t(j)} y_{ef} \leq x_f \qquad \forall t \in T, \forall j \in H, \forall f \in \delta^{t'}(j) : t' \neq t \tag{A.5}$$

$$\sum_{f \in \delta^t(j)} y_{ef} \leq x_e \qquad \forall t \in T, \forall j \in H, \forall e \in \delta^{t'}(j) : t' \neq t \tag{A.6}$$

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 B.1: AQSAP- Comparing GUROBI and CG- At least one of the methods stops within the time limit

Instance (n-h)	BFS	GUROBI+BQP			CG+BQPPricing			CG+HeuristicBQPPricing			CG+HeuristicSLTPricing			GUROBI+SLT			CG+SLTPricing		
		LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time
10-3	3038.8	3038.8	**1**	**0**	3038.8	1	2	3038.8	1	8	3038.8	1	6	3038.8	1	1	3038.8	1	2
10-4	2829.3	2829.3	**1**	**0**	2829.3	1	4	2829.3	1	11	2829.3	1	8	2829.3	1	2	2829.3	1	1
15-3	32202.4	32202.4	1	52	32202.4	1	56	32202.4	1	18	32202.4	**1**	**14**	32202.4	1	40	32202.4	1	16
15-4	21336.7	21336.7	1	524	21336.7	1	57	21336.7	1	28	21336.7	1	28	21335.4	1	314	21336.7	**1**	**17**
15-6	4472.7	4472.7	1	327	4456.6	1	41	4456.6	1	25	4456.6	1	16	4472.7	1	233	4456.6	**1**	**10**
18-3	8168.7	8168.7	1	43	8168.7	1	298	8168.7	1	35	8168.7	**1**	**24**	8168.7	1	25	8168.7	1	142
18-4	6747.3	6747.3	1	1852	6744.1	1	142	6744.1	**1**	**5**	6744.1	1	8	6747.3	1	909	6744.1	1	127
18-6	5906.2	4862.4	0.82	10800	5894.2	1	96	5894.2	1	53	5894.2	**1**	**43**	4769.9	0.81	10800	5894.2	1	147
18-8	4916	4176	0.85	10800	4916	1	96	4916	1	83	4916	**1**	**82**	3949	0.8	10800	4916	1	160
20-3	15549.8	15549.8	1	1099	15549.8	1	430	15549.7	**1**	**10**	15549.8	1	25	15549.8	1	345	15549.8	1	531
20-4	13657.3	12596.4	0.92	10800	13657.3	1	333	13657.3	1	58	13657.3	**1**	**25**	12835.8	0.94	10800	13657.3	1	331
20-6	9781.5	7037.9	0.72	10800	9781.5	1	227	9781.5	1	101	9781.5	**1**	**52**	6617	0.68	10800	9781.5	1	293
20-8	8559.1	5281.5	0.62	10800	8559.1	1	105	8559.1	**1**	**10**	8559.1	1	31	5292	0.62	10800	8559.1	1	176
20-10	4810.8	3978.4	0.83	10800	4810.8	1	151	4810.8	**1**	**18**	4810.8	1	59	3875	0.81	10800	4810.8	1	195
22-3	42873.5	41549.9	0.97	10800	42873.5	1	1744	42873.5	**1**	**51**	42873.5	1	74	42873.5	1	7360	42873.5	1	1612
22-4	24085.5	20842.4	0.87	10800	24082.7	1	500	24082.7	**1**	**27**	24082.7	1	76	20325.5	0.84	10800	24082.7	1	590
22-6	18316.3	13156.1	0.72	10800	18316.3	1	237	18316.3	**1**	**28**	18316.3	1	66	11492.1	0.63	10800	18316.3	1	404
22-8	7394.2	5446.3	0.74	10800	7384.8	1	205	7384.8	**1**	**13**	7384.8	1	31	4988.8	0.67	10800	7384.8	1	293
22-10	6017.8	4668.6	0.78	10800	6015.3	1	188	6015.3	1	95	6015.3	**1**	**49**	4406	0.73	10800	6015.3	1	284
22-12	4519.2	3768.7	0.83	10800	4507.3	1	158	4507.3	1	100	4507.3	**1**	**73**	3448.4	0.76	10800	4507.3	1	197
25-3	42794.9	40199.4	0.94	10800	42794.9	1	3127	42794.9	**1**	**51**	42794.9	1	72	41760	0.98	10800	42794.9	1	2417
25-4	14012.7	12582.1	0.9	10800	14012.7	1	598	14012.7	**1**	**16**	14012.7	1	79	12657.1	0.9	10800	14012.7	1	656
25-6	9478.1	6401.8	0.68	10800	9478.1	1	1937	9478.1	1	684	9478.1	**1**	**174**	5580	0.59	10800	9478.1	1	1477
25-8	8807.1	6016.4	0.68	10800	8796.6	1	581	8796.6	1	566	8796.6	**1**	**252**	5629.2	0.64	10800	8796.6	1	614
25-10	6318.8	4439.3	0.7	10800	6300	1	975	6300	**1**	**98**	6300	1	99	3880.8	0.61	10800	6300	1	591
25-12	4744	2882.8	0.61	10800	4736.2	1	265	4736.2	1	228	4736.2	**1**	**179**	2445.8	0.52	10800	4736.2	1	348
30-3	48847.1	43337.6	0.89	10800	48811.3	1	1741	48811.3	**1**	**70**	48811.3	1	98	45839.1	0.94	10800	48811.3	1	1863
30-4	20369.8	14991.8	0.74	10800	20210.9	0.99	10800	20349	1	3665	20349	**1**	**1143**	14924.2	0.73	10800	20349	1	9636
30-6	14167	7396.7	0.52	10800	14135.6	1	10800	14143.7	**1**	**2001**	14143.7	1	2287	7126.4	0.5	10800	14143.6	1	10800
30-8	10322.5	5967	0.58	10800	10310.2	1	6378	10310.2	1	2612	10310.2	**1**	**964**	5205.9	0.5	10800	10310.2	1	2953
30-14	6680.8	3151.4	0.47	10800	6679.7	1	1875	6679.7	**1**	**214**	6679.7	1	831	2710.4	0.41	10800	6679.7	1	818
40-14	10365.5	4145.3	0.4	10800	10163.6	0.98	10800	10345.1	1	10800	10355.7	**1**	**4062**	3573.3	0.34	10800	10346.6	1	10800
50-4	49468.9	21334.4	0.43	10800	49290.5	1	10800	49462.2	**1**	**6895**	49461.5	1	10800	18143.5	0.37	10800	47815.1	0.97	10800
50-6	40831	27484.4	0.67	10800	40753	1	10800	40820.8	**1**	**4156**	22875.6	0.56	10800	10049.1	0.25	10800	33571.4	0.82	10800
50-8	34960.5	17393.6	0.5	10800	34916.6	1	10800	34960.5	**1**	**7146**	15940.7	0.46	10800	8443.7	0.24	10800	27262	0.78	10800

Table B.2: AQSAP-Comparing GUROBI and CG- None of the methods stop within the time limit

Instance (n-h)	BFS	GUROBI+ BQP		CG+ BQPPricing		CG+Heuristic BQPPricing		CG+Heuristic SLTPricing		GUROBI+ SLTPricing		CG+ SLTPricing	
		LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS
40-3	46076.8	30559.7	0.66	43217.3	0.94	46059	**1**	46059	**1**	26953.3	0.58	42981.1	0.93
40-4	37297	16980.5	0.46	33348.9	**0.89**	29908.8	0.80	29908.8	0.80	15234.8	0.41	33305.3	**0.89**
40-6	17322.9	6164.7	0.36	14903.7	0.86	14903.7	0.86	14903.7	0.86	4854.7	0.28	15287.1	**0.88**
40-8	16018.9	5411.6	0.34	15025	**0.94**	13450.9	0.84	15991.2	1	4606.8	0.29	14450	0.90
50-3	52448.4	32173.2	0.61	52183.3	0.99	52442.3	**1**	52442.3	**1**	26415.1	0.50	49418.9	0.94
50-14	11696.4	3412.2	0.29	11601.7	0.99	11685.2	**1**	4748.7	0.41	3965.8	0.34	9139	0.78

A closer look at Tables B.3 to C.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 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 with better bounds. Interestingly, in a relevant subset of instances where none of the methods stop before the time limit, CG provides a better LB demonstrating there is still potential to improve our approach to be more competitive in solving 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 density of the quadratic matrix in the complexity of the problem. Table C.12 demonstrates that in many cases CG algorithms converge to the optimal solution of the problem. Specifically, when we use CG+RLTPricing and the problem can be solved within the time limit, optimality is proved in almost all the instances.

Table B.3: QSAP- 10% out-of-star quadratic matrix density- Comparing GUROBI and CG- At least one of the methods stops within the time limit

Instance (n-h)	BFS	GUROBI+BQP			CG+BQPPricing			CG+HeuristicBQPPricing			CG+HeuristicSLTPricing			GUROBI+SLT			CG+SLTPricing		
		LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time
10-3	3061.4	3061.4	**1**	**0**	3050.1	1	6	3050.1	1	6	3050.1	1	3	3061.4	1	2	3050.1	1	2
10-4	2829.3	2829.3	**1**	**1**	2829.3	1	5	2829.3	1	5	2829.3	1	5	2829.3	**1**	**1**	2829.3	1	2
15-3	33075.4	33075.4	**1**	**4**	32311	0.98	77	32311	0.98	10	32311	0.98	6	33075.4	1	11	32311	0.98	22
15-4	21665.4	21665.4	**1**	**21**	21372.1	0.99	71	21372.1	0.99	18	21372.1	**0.99**	**14**	21665.4	1	348	21372.1	0.99	12867
15-6	4497.2	4497.2	**1**	**7**	4466.2	0.99	40	4466.2	0.99	15	4466.2	0.99	15	4497.2	1	79	4466.2	0.99	64
18-3	8383	8383	**1**	**4**	8273	0.99	164	8273	0.99	41	8273	0.99	32	8383	1	13	8273	0.99	180
18-4	6884.3	6884.3	**1**	**23**	6758	0.98	144	6758	0.98	29	6758	0.98	24	6884.3	1	119	6758	0.98	169
18-6	5957.5	5957.5	**1**	**248**	5897.7	0.99	123	5897.7	0.99	47	5897.7	**0.99**	**39**	5957.5	1	4021	5897.7	0.99	135
18-8	4956.1	4956.1	**1**	**4762**	4924.1	0.99	125	4924.1	0.99	43	4924.1	**0.99**	**40**	3786.8	0.76	10800	4924.1	0.99	129
20-3	16245.3	16245.3	**1**	**20**	15736.8	0.97	533	15736.8	0.97	103	15736.8	0.97	55	16245.3	1	75	15736.8	0.97	561
20-4	13942.2	13942.2	**1**	**492**	13681.7	0.98	459	13681.7	0.98	53	13681.7	**0.98**	**25**	13942.2	1	1701	13681.7	0.98	502
20-6	9919.2	9919.2	**1**	**4867**	9787.3	0.99	224	9787.3	0.99	42	9787.3	**0.99**	**23**	5084.5	0.51	10800	9787.3	0.99	270
20-8	8706.3	6559.9	0.75	10800	8566.3	0.98	156	8566.3	0.98	76	8566.3	**0.98**	**31**	4588.4	0.53	10800	8566.3	0.98	250
20-10	4878.3	4298.2	0.88	10800	4827.6	0.99	194	4827.6	0.99	87	4827.6	**0.99**	**34**	3125.4	0.64	10800	4827.6	0.99	280
22-3	44517.4	44517.4	**1**	**444**	42919	0.96	1092	42919	0.96	34	42919	**0.96**	**21**	44517.4	1	1632	42919	0.96	1432
22-4	25071.6	25071.6	**1**	**4806**	24092.1	0.96	589	24092.1	0.96	89	24092.1	**0.96**	**36**	25071.6	1	9336	24092.1	0.96	768
22-6	18961.7	14956.3	**0.79**	**10800**	18341.3	0.97	312	18341.3	0.97	89	18341.3	**0.97**	**31**	10373.4	0.55	10800	18341.3	0.97	391
22-8	7587.9	6234.8	**0.82**	**10800**	7390.8	0.97	322	7392.3	0.97	109	7392.3	**0.97**	**59**	4715.4	0.62	10800	7392.3	0.97	511
22-10	6124.3	5123.1	0.84	10800	6031.4	0.98	336	6031.4	**0.98**	**84**	6031.4	0.98	107	3988.6	0.65	10800	6031.4	0.98	432
22-12	4583.5	4035.2	0.88	10800	4511.1	0.98	353	4511.1	**0.98**	**111**	4511.1	0.98	126	2821	0.62	10800	4511.1	0.98	478
25-3	44875.1	44875.1	**1**	**206**	43205.4	0.96	3207	43273.8	0.96	516	43273.7	**0.96**	**176**	44875.1	1	918	43273.7	0.96	4427
25-4	14811.7	14811.7	**1**	**83**	14137.4	0.95	1256	14137.4	0.95	122	14137.4	**0.95**	**66**	14811.7	1	567	14137.4	0.95	1298
25-6	9972.8	7807.7	**0.78**	**10800**	9490.4	0.95	2361	9494.8	0.95	376	9494.8	**0.95**	**195**	4679.2	0.47	10800	9494.8	0.95	2629
25-8	9088.3	6594.8	0.73	10800	8800.7	0.97	2051	8800.7	0.97	673	8800.7	**0.97**	**402**	4730.9	0.52	10800	8800.7	0.97	1492
25-10	6554.3	4794.3	0.73	10800	6305.8	0.96	2698	6305.9	**0.96**	**367**	6305.8	0.96	543	2535.2	0.39	10800	6305.8	0.96	1822
25-12	4966.7	2331.8	0.47	10800	4737.7	0.95	1130	4738.4	0.95	1024	4738.4	**0.95**	**916**	1706.5	0.34	10800	4738.4	0.95	1712
30-3	53502.1	53502.1	**1**	**203**	53502.1	0.94	8255	50132.5	0.94	412	50132.5	**0.94**	**79**	53502.1	1	2902	50132.5	0.94	9692
30-4	21896.3	21896.3	**1**	**1359**	20293.8	0.93	10800	20469.2	0.93	4229	20469.2	0.93	4350	21896.3	1	8366	20431.4	0.93	10800
30-6	15527.6	10779.1	0.69	10800	14073.2	0.91	10800	14175.1	**0.91**	**2324**	14170.6	0.91	3722	6271	0.40	10800	14163.5	0.91	10800
30-8	10807.5	4332.3	0.40	10800	10315.8	0.95	8560	10315.9	0.95	4120	10315.9	**0.95**	**2720**	2643.3	0.24	10800	10315.9	0.95	8120
30-14	6978.9	2537.3	0.36	10800	6681.2	0.96	7894	6681.4	**0.96**	**1063**	6681.4	0.96	10800	1781.4	0.26	10800	6681.4	0.96	10800

Table B.4: QSAP-10% out-of-star quadratic matrix density- Comparing GUROBI and CG- None of the methods stop within the time limit

Instance (n-h)	BFS	GUROBI+BQP		CG+BQPPricing		CG+Heuristic BQPPricing		CG+Heuristic SLTPricing		GUROBI+ SLTPricing		CG+SLTPricing	
		LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS
40-3	51693.8	38850.4	0.75	42972	0.83	35176.7	0.68	45140.8	**0.87**	32442.4	0.63	42972	0.83
40-4	42582.4	21564	0.51	32631.2	**0.77**	29474.6	0.69	29474.6	0.69	8965.4	0.21	32624.8	**0.77**
40-6	19870.8	6052	0.3	15025.5	**0.76**	3614.2	0.18	3614.2	0.18	3272.2	0.16	13398.2	0.67
40-8	18865.9	3981.4	0.21	8148	**0.43**	6590.5	0.35	6590.5	0.35	3289.8	0.17	8148.1	**0.43**
40-14	11635	2919.5	0.25	5890.6	**0.5**	5888	**0.5**	4373.7	0.37	2436.4	0.2	5921.8	**0.5**
50-3	60996.3	50548.7	0.83	53212.6	0.87	53929.1	**0.88**	50319.1	0.82	35160.1	0.58	48302.3	0.79
50-4	62141.8	28749.9	0.46	48821.6	0.79	50272.7	**0.81**	46924.5	0.76	16379.5	0.26	45135.3	0.73
50-6	51569.9	13775.9	0.27	39816.6	0.77	40662.1	**0.79**	13699	0.27	7959	0.15	25648.7	0.5
50-8	44208.2	8031.6	0.18	34076.6	0.77	34742.4	**0.79**	14741.6	0.33	6638.9	0.15	19770.9	0.45
50-14	14426.3	3238.6	0.22	5320.3	**0.37**	3734.4	0.26	4335.9	0.30	2830.9	0.20	4189.6	0.29

C Primal bounding

As mentioned before, primal bounding methodologies are beyond the scope of this project. However, to show the strength of proposed reformulation framework and CG for the adjacent-only class of BQPs, we obtain the primal bounds for GUROBI and all of the CG methods. In the next sections we first discuss the obtained UB in adjacent-only problems and then we point out some discussions on 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 CG methods terminate, in a relatively large fraction of instances in adjacent-only problems, the BFS is obtained through CG, meaning that the reformulation and CG outperform GUROBI in terms of both primal bound and dual bound for these problems.

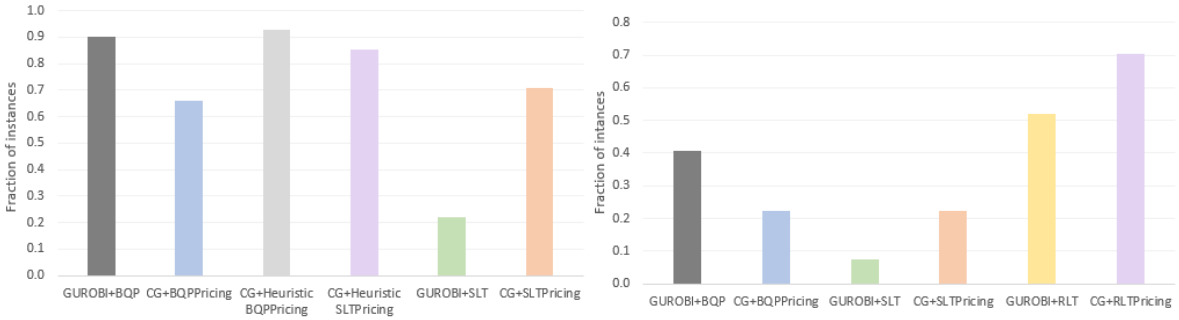
In Figure 1, we demonstrate a comparison of different methods for attaining the BFS for each instance. This figure proves the capabilities of CG methods to find the best upper bounds in both the AQSAP and MOT problem. According to Figure 1a, CG+HeuristicBQPPricing obtains the BFS for largest fraction of instances (93%) among all of the methods. For MOT, Figure 1b indicates that CG+RLTPricing is the best method, finding the BFS for 70% of instances.

Table C.5: QSAP- 15% out-of-star quadratic matrix density- Comparing GUROBI and CG in different methods - At least one of the methods stops within the time limit

Instance (n-h)	BFS	GUROBI+BQP			CG+BQPPricing			CG+HeuristicBQPPricing			CG+HeuristicSLTPricing			GUROBI+SLT			CG+SLTPricing		
		LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time
10-3	3172.5	3172.5	**1**	**0**	3067.9	0.97	4	3067.9	0.97	4	3067.9	0.97	4	3172.5	1	2	3067.9	0.97	1
10-4	2881.7	2881.7	**1**	**1**	2862.4	0.99	5	2862.4	0.99	5	2862.4	0.99	4	2881.7	**1**	**1**	2862.4	0.99	1
15-3	33324.7	33324.7	**1**	**2**	32313.8	0.97	54	32313.8	0.97	14	32313.8	0.97	10800	33324.7	1	8	32313.8	0.97	18
15-4	22318.5	22318.5	**1**	**6**	21376.6	0.96	63	21376.6	0.96	26	21376.6	0.96	20	22318.5	1	65	21376.6	0.96	65
15-6	4568.5	4568.5	**1**	**4**	4470.1	0.98	41	4470.1	0.98	16	4470.1	0.98	21	4568.5	1	38	4470.1	0.98	56
18-3	8669.3	8669.3	**1**	**2**	8285.5	0.96	182	8285.5	0.96	24	8285.5	0.96	22	8669.3	1	6	8285.5	0.96	212
18-4	6978.7	6978.7	**1**	**10**	6763.7	0.97	194	6763.7	0.97	29	6763.7	0.97	26	6978.7	1	38	6763.7	0.97	193
18-6	6126.4	6126.4	**1**	**130**	5898	0.96	122	5898	0.96	62	5898	0.96	59	6126.4	1	1544	5898	0.96	137
18-8	5034.8	5034.8	**1**	**419**	4925.8	0.98	97	4925.8	0.98	57	4925.8	**0.98**	**52**	5034.8	1	7584	4925.8	0.98	122
20-3	16434.4	16434.4	**1**	**9**	15760.5	0.96	600	15760.5	0.96	71	15760.5	0.96	45	16434.4	1	27	15760.5	0.96	550
20-4	14529.7	14529.7	**1**	**97**	13681.7	0.94	349	13681.7	0.94	32	13681.7	**0.94**	**19**	14529.7	1	788	13681.7	0.94	444
20-6	10257.6	10257.6	**1**	**1707**	9788	0.95	206	9788	0.95	57	9788	**0.95**	**25**	8734.4	0.85	10800	9788	0.95	283
20-8	9062.6	9062.6	**1**	**9703**	8566.4	0.95	227	8566.4	0.95	84	8566.4	**0.95**	**36**	5478.7	0.6	10800	8566.4	0.95	289
20-10	4965.2	4965.2	**1**	**4169**	4829.7	0.97	196	4829.7	0.97	110	4829.7	**0.97**	**50**	3239.9	0.65	10800	4829.7	0.97	315
22-3	45955.1	45955.1	**1**	**222**	42919	0.93	897	42919	0.93	52	42919	**0.93**	**37**	45955.1	1	1037	42919	0.93	1525
22-4	26366.2	26366.2	**1**	**899**	24093.1	0.91	464	24093.1	0.91	121	24093.1	**0.91**	**44**	26366.2	1	4682	24093.1	0.91	654
22-6	19512	19512	**1**	**8792**	18341.3	0.94	256	18341.3	0.94	73	18341.3	**0.94**	**29**	11861.2	0.61	10800	18341.3	0.94	433
22-8	8214.5	8214.5	**1**	**4946**	7393	0.9	525	7393	0.9	106	7393	**0.90**	**78**	5234.4	0.64	10800	7393	0.9	492
22-10	6349.8	5412.2	0.85	10800	6034.3	0.95	259	6034.3	**0.95**	**168**	6034.3	0.95	237	4141.8	0.65	10800	6034.3	0.95	368
22-12	4939.1	3403.4	0.69	10800	4511.1	0.91	433	4511.1	0.91	160	4511.1	**0.91**	**131**	3177.6	0.64	10800	4511.1	0.91	490
25-3	48713.8	48713.8	**1**	**117**	43288.9	0.89	4418	43349.3	0.89	287	43349.3	0.89	202	48713.8	1	377	43349.3	0.89	4186
25-4	16521.4	16521.4	**1**	**46**	14214.6	0.86	1685	14214.6	0.86	121	14214.6	0.86	75	16521.4	1	241	14214.6	0.86	1631
25-6	10827.9	10827.9	**1**	**7729**	9495.5	0.88	1932	9495.5	0.88	384	9495.5	**0.88**	**191**	5551.8	0.51	10800	9495.5	0.88	1895
25-8	10227.6	5978.8	0.58	10800	8801.9	0.86	4584	8801.9	0.86	952	8801.9	**0.86**	**511**	5319.3	0.52	10800	8801.9	0.86	2642
25-10	7063.3	4147.7	0.59	10800	6306.4	0.89	3336	6306.5	0.89	942	6306.5	**0.89**	**840**	2750.4	0.39	10800	6306.5	0.89	2130
25-12	5714.7	2494	0.44	10800	4738.4	0.83	1328	4738.4	0.83	10156	4738.4	0.83	10283	1804.4	0.32	10800	4738.4	0.83	10800
30-3	57082.6	57082.6	**1**	**93**	50126.8	0.88	10800	50207	0.88	1751	50207	0.88	264	57082.6	1	1025	50206.6	0.88	10800
30-4	24213	24213	**1**	**109**	20338.4	0.84	10800	20497.5	0.85	2085	20497.5	0.85	1132	24213	1	2207	20464.8	0.85	10800
30-6	18614.7	12465.7	0.67	10800	14146.5	0.76	10800	14178	**0.76**	**809**	14170.6	0.76	2739	7506.8	0.4	10800	14164.8	0.76	10800
30-8	14294	5066.5	0.35	10800	10315.4	0.72	5906	10316.2	0.72	5366	10316.1	**0.72**	**3479**	3269.6	0.23	10800	10316.1	0.72	8978
30-14	10284	2560.9	0.25	10800	6681	0.64	9593	6681.7	0.64	1371	6681.7	**0.64**	**1365**	1913.8	0.18	10800	6681.7	0.64	4691

Table C.6: QSAP- 15% out-of-star quadratic matrix density- Comparing GUROBI and CG- None of the methods stop within the time limit

Instance (n-h)	BFS	GUROBI+BQP		CG+BQPPricing		CG+Heuristic BQPPricing		CG+Heuristic SLTPricing		GUROBI+ SLTPricing		CG+SLTPricing	
		LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS
40-3	55430.1	47346.3	**0.85**	41261.2	0.74	34545.8	0.62	45138.8	0.81	36149.5	0.65	42274.4	0.76
40-4	47234.8	32290.2	0.68	32858.1	**0.7**	30129.9	0.64	30129.9	0.64	15269.6	0.32	32361.9	0.69
40-6	25063.4	6785.9	0.27	14519.2	**0.58**	5082.2	0.2	5082.2	0.2	3558	0.14	14122.5	0.56
40-8	27886.8	4382.9	0.16	8117.5	**0.29**	4652.8	0.17	4652.8	0.17	3512.4	0.13	8200.1	**0.29**
40-14	17160.5	2929.9	0.18	6253.1	0.36	3814.5	0.22	3263.6	0.19	2589.7	0.15	6274.5	**0.37**
50-3	65385.1	59427.3	**0.91**	53082.8	0.81	53776.8	0.82	50607.7	0.77	43112.9	0.66	49664.3	0.76
50-4	70572.9	40464.5	0.57	48807.7	0.69	49460.9	**0.7**	46713.3	0.66	20173.8	0.29	44474.4	0.63
50-6	62755.8	14993	0.24	39683.7	0.63	40812.9	**0.65**	14234.6	0.23	8856.8	0.14	32200.2	0.51
50-8	53815.1	8260.5	0.15	33595	0.62	34896.4	**0.65**	17103.3	0.32	7084.9	0.13	18782.7	0.35
50-14	26288.9	3124.8	0.12	4524.7	**0.17**	701.5	0.03	4107.3	0.16	3070.6	0.12	4326.9	0.16



(a) AQSAP methods **(b) MOT methods**

Figure 1: 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:

$$\min \quad \sum_{s \in S} C_s \lambda_s + \sum_{s \in S} \sum_{s' > s \in S} Q_{ss'} \lambda_s \lambda_{s'} \qquad \text{s.t.} \quad (28-30) \qquad \text{(C.7)}$$

Through the new definition of quadratic costs, we discard many impossible interactions among stars from the beginning. This leads to reaching the same feasible solution 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 UB of QSAP, but we would like to provide some remarks to direct future research on this topic:

Remark 6. GUROBI methods, and in particular GUROBI+BQP, attain the BFS in the vast majority of cases in 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. Evidently, GUROBI provides a high-quality feasible solution in a few seconds at the beginning of the branch-and-bound process based on strong heuristics, but it does not improve this primal bound hugely during the process.

Remark 8. 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 C.7: QSAP- 20% out-of-star quadratic matrix density- Comparing GUROBI and CG- At least one of the methods stops within the time limit

Instance (n-h)	BFS	GUROBI+BQP			CG+BQPPricing			CG+HeuristicBQPPricing			CG+HeuristicSLTPricing			GUROBI+SLT			CG+SLTPricing		
		LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time
10-3	3213.4	3213.4	**1**	**0**	3077.4	0.96	5	3077.4	0.96	3	3077.4	0.96	3	3213.4	1	1	3077.4	0.96	1
10-4	3001.4	3001.4	**1**	**1**	2869	0.96	9	2869	0.96	5	2869	0.96	4	3001.4	**1**	**1**	2869	0.96	2
15-3	34726.1	34726.1	**1**	**2**	32316.1	0.93	60	32316.1	0.93	11	32316.1	0.93	10	34726.1	1	6	32316.1	0.93	21
15-4	23852.4	23852.4	**1**	**7**	21377.4	0.9	72	21377.4	0.9	19	21377.4	0.9	19	23852.4	1	54	21377.4	0.9	75
15-6	5132.6	5132.6	**1**	**5**	4471.5	0.87	52	4471.5	0.87	21	4471.5	0.87	21	5132.6	1	33	4471.5	0.87	66
18-3	9895.4	9895.4	**1**	**2**	8291.7	0.84	182	8291.7	0.84	31	8291.7	0.84	26	9895.4	1	5	8291.7	0.84	227
18-4	7357.1	7357.1	**1**	**6**	6766.7	0.92	174	6766.7	0.92	44	6766.7	0.92	35	7357.1	1	19	6766.7	0.92	174
18-6	6379.6	6379.6	**1**	**43**	5898.1	0.92	138	5898.1	0.92	54	5898.1	0.92	48	6379.6	1	374	5898.1	0.92	138
18-8	5446	5446	**1**	**228**	4925.5	0.9	116	4925.5	0.9	55	4925.5	**0.9**	**48**	5446	1	3503	4925.5	0.9	134
20-3	18140.5	18140.5	**1**	**8**	15773.9	0.87	533	15773.9	0.87	189	15773.9	0.87	13	18140.5	**1**	**39**	15773.9	0.87	627
20-4	15836	15836	**1**	**73**	13681.7	0.86	394	13681.7	0.86	28	13681.7	**0.86**	**20**	15836	1	553	13681.7	0.86	461
20-6	11201.2	11201.2	**1**	**1681**	9788	0.87	197	9788	0.87	51	9788	**0.87**	**26**	11201.2	1	6065	9788	0.87	286
20-8	10102.4	10102.4	**1**	**2891**	8566.6	0.85	169	8566.6	0.85	124	8566.6	**0.85**	**40**	6217.2	0.62	10751	8566.6	0.85	219
20-10	5464.7	5464.7	**1**	**4997**	4831.7	0.88	242	4831.7	0.88	167	4831.7	**0.88**	**71**	3796.1	0.69	10751	4831.7	0.88	337
22-3	49327.2	49327.2	**1**	**248**	42917.7	0.87	1066	42919	0.87	64	42919	**0.87**	**28**	49327.2	1	692	42919	0.87	1693
22-4	28349	28349	**1**	**705**	24093.1	0.85	524	24093.1	0.85	113	24093.1	**0.85**	**45**	28349	1	4661	24093.1	0.85	659
22-6	21754.1	21754.1	**1**	**7202**	18337.9	0.84	214	18344	0.84	115	18344	**0.84**	**46**	13208.7	0.61	10800	18344	0.84	431
22-8	9714.9	6911.3	**0.71**	**10800**	7394.2	0.76	342	7394.2	0.76	132	7394.2	**0.76**	**87**	6078.8	0.63	10800	7394.2	0.76	470
22-10	7733.6	5698.1	0.74	10800	6034.5	0.78	322	6034.5	0.78	243	6034.5	**0.78**	**263**	4724.3	0.61	10800	6034.5	0.78	438
22-12	5974.1	3903.7	0.65	10800	4511.1	**0.76**	**685**	4511.1	0.76	461	4511.1	0.76	687	3156.2	0.53	10800	4511.1	0.76	818
25-3	51647	51647	**1**	**88**	43346	0.84	4549	43370.9	0.84	218	43370.9	0.84	149	51647	1	321	43370.9	0.84	4396
25-4	17712.6	17712.6	**1**	**21**	14221.2	0.8	1669	14228.2	0.8	112	14228.2	0.8	69	17712.6	1	74	14228.2	0.8	1812
25-6	12832.8	12832.8	**1**	**2871**	9495.5	0.74	2300	9495.5	0.74	241	9495.5	**0.74**	**196**	9895.7	0.77	10800	9495.5	0.74	2235
25-8	13672.4	7316.1	0.54	10800	8802.1	0.64	5744	8802.1	0.64	542	8802.1	**0.64**	**384**	5445.1	0.4	10800	8802.1	0.64	2463
25-10	9206.1	4485.7	0.49	10800	6311	0.69	2446	6311	0.69	1284	6311	**0.69**	**957**	2617.4	0.28	10800	6311	0.69	1967
25-12	8316.1	2788.9	0.34	10800	4738.4	0.57	1181	4738.4	0.57	366	4738.4	**0.57**	**295**	2092.4	0.25	10800	4738.4	0.57	1173
30-3	61853.4	61853.4	**1**	**48**	50149.5	0.81	10800	50268.5	0.81	1105	50268.5	0.81	237	61853.4	1	257	50260.7	0.81	10800
30-4	31188.4	31188.4	**1**	**222**	20271.2	0.65	10800	20534.9	0.66	10800	20535.1	0.66	1557	31188.4	1	2317	20485.1	0.66	10800
30-6	25956.1	13397.9	0.52	10800	14158.2	0.55	10800	14170.6	0.55	2629	14170.6	**0.55**	**1965**	8937	0.34	10752	14167.1	0.55	10800
30-8	22159.8	6551.6	0.3	10800	10316	0.47	10800	10316.2	0.47	4722	10316.2	**0.47**	**2499**	3578.7	0.16	10800	10316.2	0.47	7811
30-14	10000	2696.1	0.27	10800	6681.2	0.67	3044	6681.7	0.67	1185	6681.7	**0.67**	**1191**	2059.8	0.21	10800	6681.7	0.67	1461
50-3	71284.3	71281.1	**1**	**4427**	53119.7	0.75	10800	53649.2	0.75	10800	49488.7	0.69	10800	71284.3	1	5015	49430.2	0.69	10800

Table C.8: QSAP- 20% out-of-star quadratic matrix density- Comparing GUROBI and CG- None of the methods stop within the time limit

Instance (n-h)	BFS	GUROBI+BQP		CG+BQPPricing		CG+Heuristic BQPPricing		CG+Heuristic SLTPricing		GUROBI+ SLTPricing		CG+SLTPricing	
		LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS
40-3	59972.9	50584.6	**0.84**	42125.5	0.7	38673.8	0.64	45219.8	0.75	43825.9	0.73	42125.5	0.7
40-4	53056.6	37264.5	**0.7**	32994.7	0.62	26597.2	0.5	26597.2	0.5	14365.2	0.27	32254.8	0.61
40-6	29733.5	7813.6	0.26	15166.2	**0.51**	8320.1	0.28	8320.1	0.28	4022.8	0.14	14483.6	0.49
40-8	41010.5	5559.4	0.14	7742.1	0.19	4589.3	0.11	4589.3	0.11	3819.8	0.09	8439	**0.21**
40-14	28785.3	2945.6	0.1	5193.6	0.18	4788.3	0.17	2892.6	0.1	2773.3	0.1	5362.9	**0.19**
50-4	77901.2	57771.6	**0.74**	48799.5	0.63	49810.1	0.64	47129.8	0.6	27889.7	0.36	43984.5	0.56
50-6	74931.2	21463.3	0.29	39954.6	0.53	40953.9	**0.55**	16697.4	0.22	10209.2	0.14	24392.7	0.33
50-8	70230.7	10598.1	0.15	33909	0.48	34744.5	**0.49**	16170	0.23	7731.2	0.11	19306	0.27
50-14	27759.7	3352.6	0.12	4637	**0.17**	237.9	0.01	25.4	0	3280	0.12	4155.6	0.15

Table C.9: QSAP- 25% out-of-star quadratic matrix density- Comparing GUROBI and CG- At least one of the methods stops within the time limit

Instance (n-h)	BFS	GUROBI+BQP			CG+BQPPricing			CG+HeuristicBQPPricing			CG+HeuristicSLTPricing			GUROBI+SLT			CG+SLTPricing		
		LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time	LB	LB/BFS	Time
10-3	3456.3	3456.3	**1**	**1**	3090.6	0.89	7	3090.6	0.89	5	3090.6	0.89	4	3456.3	**1**	**1**	3090.6	0.89	1
10-4	3161	3161	1	1	2889	0.91	6	2889	0.91	8	2889	0.91	6	3161	**1**	**0.1**	2889	0.91	2
15-3	37682.9	37682.9	**1**	**2**	32316.1	0.86	66	32316.1	0.86	11	32316.1	0.86	8	37682.9	1	6	32316.1	0.86	20
15-4	25935.9	25935.9	**1**	**6**	21382.6	0.82	77	21382.6	0.82	26	21382.6	0.82	22	25935.9	1	53	21382.6	0.82	89
15-6	6285.4	6285.4	**1**	**3**	4472.8	0.71	52	4472.8	0.71	24	4472.8	0.71	23	6285.4	1	50	4472.8	0.71	65
18-3	10703.9	10703.9	**1**	**2**	8295.1	0.77	168	8295.1	0.77	37	8295.1	0.77	38	10703.9	1	5	8295.1	0.77	221
18-4	9346.3	9346.3	**1**	**7**	6768.2	0.72	203	6768.2	0.72	32	6768.2	0.72	29	9346.3	1	23	6768.2	0.72	222
18-6	7765.1	7765.1	**1**	**76**	5898.5	0.76	124	5898.5	0.76	51	5898.5	0.76	41	7765.1	1	722	5898.5	0.76	142
18-8	6158.9	6158.9	**1**	**187**	4926.2	0.8	116	4926.2	0.8	80	4926.2	0.8	71	6158.9	1	2245	4926.2	0.8	131
20-3	20576.4	20576.4	**1**	**10**	15777.7	0.77	519	15777.7	0.77	52	15777.7	0.77	16	20576.4	1	33	15777.7	0.77	704
20-4	16772.2	16772.2	**1**	**34**	13672.3	0.82	462	13681.7	0.82	51	13681.7	**0.82**	**22**	16772.2	1	154	13681.7	0.82	712
20-6	12269.6	12269.6	**1**	**960**	9788	0.8	239	9788	0.8	77	9788	**0.8**	**27**	12269.6	1	4011	9788	0.8	285
20-8	11938.8	8905.4	0.75	10800	8566.6	0.72	175	8566.6	0.72	174	8566.6	**0.72**	**43**	7221.3	0.6	10800	8566.6	0.72	264
20-10	6446.6	6446.2	**1**	**10800**	4831.7	0.75	238	4831.7	0.75	140	4831.7	**0.75**	**68**	4245	0.66	10800	4831.7	0.75	346
22-3	50813.7	50813.7	**1**	**126**	42911.5	0.84	1037	42919	**0.84**	**42**	42919	**0.84**	**42**	50813.7	1	307	42919	0.84	1644
22-4	30416.8	30416.8	**1**	**621**	24088.6	0.79	376	24093.2	0.79	95	24093.2	**0.79**	**54**	30416.8	1	1807	24093.2	0.79	729
22-6	24468.6	24468.6	**1**	**9098**	18344	0.75	249	18344	0.75	98	18344	**0.75**	**59**	17087.8	0.7	10800	18344	0.75	360
22-8	11778.1	8561.1	**0.73**	**10800**	7394.5	0.63	336	7394.5	0.63	125	7394.5	**0.63**	**90**	7731.5	0.66	10800	7394.5	0.63	443
22-10	11110	5756.4	0.52	10800	6031.4	0.54	334	6034.7	0.54	257	6034.7	**0.54**	**164**	5340.2	0.48	10800	6034.7	0.54	491
22-12	7911.4	4341.5	0.55	10800	4511.1	**0.57**	**590**	4511.1	0.57	812	4511.1	0.57	723	3369.7	0.43	10800	4511.1	0.57	973
25-3	54414.9	54414.9	**1**	**59**	43328.4	0.79	3991	43371.8	0.79	368	43371.8	0.79	283	54414.9	1	150	43371.8	0.79	4858
25-4	19097.4	19097.4	**1**	**22**	14228.2	0.75	2513	14228.2	0.75	175	14228.2	0.75	75	19097.4	1	45	14228.2	0.75	2209
25-6	15827.4	12459.5	**0.79**	**10800**	9495.5	0.6	3070	9495.5	0.6	378	9495.5	**0.6**	**131**	12407.8	0.78	10800	9495.5	0.6	1609
25-8	16732	7807.2	0.47	10800	8802.2	0.53	3526	8802.2	0.53	969	8802.2	**0.53**	**475**	6141.4	0.37	10800	8802.2	0.53	1659
25-10	13344.2	5355.8	0.4	10800	6311.2	0.47	1363	6311.2	0.47	389	6311.2	**0.47**	**271**	2873	0.22	10800	6311.2	0.47	1273
25-12	13607.4	2724.5	0.2	10800	4738.2	0.35	8340	4738.4	0.35	251	4738.4	**0.35**	**195**	2379.3	0.17	10800	4738.4	0.35	1505
30-3	65491.3	65491.3	**1**	**28**	50150.8	0.77	10800	50270.9	0.77	904	50270.9	0.77	272	65491.3	1	190	50263.2	0.77	10800
30-4	35820.9	35820.9	**1**	**95**	20266.7	0.57	10800	20537.4	0.57	6451	20537.4	0.57	6451	35820.9	1	1277	20509.3	0.57	10800
30-6	33203.3	17039.6	**0.51**	**10800**	14130.9	0.43	10800	14170.6	0.43	7491	14170.6	**0.43**	**3041**	9725.1	0.29	10800	14154.8	0.43	10800
30-8	29178.4	7536.3	0.26	10800	10316.1	0.35	8469	10315.9	**0.35**	**1600**	10316.2	0.35	2229	4109.8	0.14	10800	10316.2	0.35	6692
30-14	19586.9	2892.1	0.15	10800	6681	0.34	9217	6681.7	0.34	1482	6681.7	**0.34**	**1422**	2200.3	0.11	10800	6681.7	0.34	3186
40-14	43918.3	3333.9	0.07	10800	5011.2	**0.11**	**10800**	4716.2	**0.1**	**1540**	3076	0.07	10800	3023	0.06	10800	5011.2	**0.11**	**10800**
50-3	74787.6	74784.1	**1**	**236**	53079.8	0.71	10800	54068.7	0.72	10800	53284.7	0.71	10800	74787.6	1	2195	49006.6	0.66	10800
50-4	78766.4	78766.4	**1**	**6021**	49097.3	0.62	10800	49789.4	0.63	10800	46368.8	0.59	10800	34921.2	0.44	10800	43522.7	0.55	10800

Table C.10: QSAP- 25% out-of-star quadratic matrix density- Comparing GUROBI and CG- None of the methods stops within the time limit

Instance (n-h)	BFS	GUROBI+BQP		CG+BQPPricing		CG+Heuristic BQPPricing		CG+Heuristic SLTPricing		GUROBI+ SLTPricing		CG+SLTPricing	
		LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS	LB	LB/BFS
40-3	65784.3	62862.9	**0.96**	43109.1	0.66	37974.2	0.58	45150.2	0.69	56372.6	0.86	42429.9	0.64
40-4	59743.9	42785.1	**0.72**	33286.3	0.56	25012.1	0.42	25012.1	0.42	20339	0.34	32641	0.55
40-6	32760	9700.7	0.3	15083.3	**0.46**	8542.3	0.26	8542.3	0.26	4741.8	0.14	14156.1	0.43
40-8	47935.6	6240.7	0.13	8894.1	0.19	7232.1	0.15	15955.3	**0.33**	4242.9	0.09	8104.8	0.17
50-6	77643.1	23153.4	0.3	39887.4	**0.51**	40757.6	0.52	17943.3	0.23	12969	0.17	30068.9	0.39
50-8	74816.2	10268	0.14	34265.7	**0.46**	32813.6	0.44	17908.1	0.24	8371.8	0.11	18971.5	0.25
50-14	27759.7	3489	0.13	4515.4	**0.16**	404	0.01	50.7	0	3591.2	0.13	3668.6	0.13

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

Instance ($\mathcal{T}$-h-d)	BFS	GUROBI+BQP			CG+BQPPricing			GUROBI+SLT			CG+SLTPricing			GUROBI+RLT			CG+RLTPricing		
		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 C.12: Data association on MOT16-09 data set- Comparing GUROBI and CG- None of the methods stop within the time limit

Instance ($\mathcal{T}$-h-d)	BFS	GUROBI+BQP		CG+BQPPricing		GUROBI+SLT		CG+SLTPricing		GUROBI+RLT		CG+RLTPricing	
		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**

References

Adams WP, Forrester RJ (2005) A simple recipe for concise mixed 0-1 linearizations. Operations Research Letters 33(1):55–61, URL http://dx.doi.org/http://dx.doi.org/10.1016/j.orl.2004.05.001.

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.

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

Barahona F (1983) The max-cut problem on graphs not contractible to k5. Operations Research Letters 2(3):107–111.

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 (2020) Mining for diamonds—matrix generation algorithms for binary quadratically constrained quadratic problems URL http://www.optimization-online.org/DB_HTML/2020/04/7764.html.

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.

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

Chen WA, Zhu Z, Kong N (2017) A lagrangian decomposition approach to computing feasible solutions for quadratic binary programs. Optimization Letters 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.

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, Jäger G, Lau A, Molitor P (2009) Complexity and algorithms for the traveling salesman problem and the assignment problem of second order. Preprint 16.

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, 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 (2018a) The linearization problem of a binary quadratic problem and its applications. arXiv preprint arXiv:1802.02426 .

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

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.

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).

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

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, 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, 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 (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 (2018a) The quadratic shortest path problem: complexity, approximability, and solution methods. European Journal of Operational Research 268(2):473–485.

Rostami B, Desaulniers G, Errico F, Lodi A (2018b) A PCA-based approximation scheme for combinatorial optimization with uncertain and correlated data. Technical report, Les Cahiers du GERAD G–2018–61, GERAD, HEC Montreal, Canada.

Rostami B, Errico F, Lodi A (2018c) A convex reformulation and an outer approximation for a class of binary quadratic program. Technical report, Technical Report DS4DM–2018–002, CERC–Data Science for Real-Time Decision.

Rostami B, Kämmerling N, Buchheim C, Clausen U (2018d) Reliable single allocation hub location problem under hub breakdowns. Computers & Operations Research 96:15–29.

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, 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.

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, URL http://dx.doi.org/http://dx.doi.org/10.1016/j.disopt.2008.08.003.

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

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.