Decomposition Methods for Quadratic Zero-One Programming

Borzou Rostami

This dissertation is submitted for the degree of Doctor of Philosophy in

Information Technology

Advisor: Prof. Federico Malucelli

2014 - XXVI

Politecnico di Milano

To

My Family

Acknowledgements

I would like to thank many people who have supported and helped me on the path towards achieving this dissertation. First and foremost, my deepest gratitude and respect go to my supervisor Prof. Federico Malucelli for not only giving the opportunity to start my Ph.D. but also for his support, valuable comments and useful feedbacks through all my study period and for providing me insertion in to a friendly research group. Concerning different issues for a foreign student, he was also very supportive for my private life in Italy.

My sincere thanks go also to Dr. Pietro Belotti, for giving me the opportunity of doing research at Clemson University, leading me to work on diverse exciting problems and supporting my private life in the U.S.

I would also like to express my special appreciations and thanks to Dr. Stefano Gualandi who has been a tremendous mentor for me. I would like to thank him for encouraging my research and helping me in learning experimental issues in my research.

Living in Italy and studying in Politecnico di Milano was a valuable experience for me and I had apportunities to know many friends and colleagues. I wish to thank all of them, because their friendship helped me to enjoy the life in Italy and encouraged me in achieving my Ph.D.; especially Amir Ghalamzan, Firooz Salami, Ali Parichehreh, Salar Baybordi, Bayan Nasri, Ehsan Ranaie, and Giuseppe Carpentiri. My special thanks go out to my colleagues and people in "Dipartimento di Elettronica, Informazione e Bioingegneria" and particularly to Prof. Edoardo Amaldi, Prof. Paolo Cremonesi, Dr. Stefano Coniglio, and Leonardo Taccari.

Last but not least, I would also like to thank my beloved wife Sara. Thank you for supporting me for everything, and especially I can't thank you enough for encouraging me throughout this experience.

Abstract

The quadratic 0-1 programming with linear constraints (QP) is a general class of optimization problems that includes many combinatorial optimization problems. The Quadratic Assignment Problem (QAP), Quadratic Traveling Salesman Problem (QTSP), Graph Partitioning, Quadratic Knapsack Problem (QKP) and Quadratic Minimum Spanning Tree Problem (QMSTP) are among the well-known particular cases of the QP which arise in a variety of real-world applications. Since in general the QP is an NP-hard non-linear optimization problem, usually a problem reformulation is considered so that a computationally inexpensive bound on the optimal objective function value is obtained. In the first part of this thesis the most important solution methods for the QP are studied. These solution methods are divided into two main groups: Reformulation-Relaxation approaches (RR), and Reformulation-Decomposition approaches (RD). It is discussed how one can reformulate and solve the problem taking into account the problem structure. In the second part of the thesis, different RR and RD approaches are developed to four classical combinatorial optimization problems including QAP, QTSP, QMSTP, and Quadratic Shortest Path problem (QSP). As for the QAP, new compact reformulations are presented for the general case, and different decomposition strategies are developed for two special cases of the problem. The QTSP is another special case of QPs that is studied and analyzed in the spirit of the reformulation-decomposition. More precisely an extended Linear Programming formulation that contains a variable for each cycle in the graph is presented. Since the number of cycles is exponential in the graph size, the new formulation is solved via Column Generation approach. In the context of QMSTP, new bounding procedures based on a novel reformulation scheme and some new mixed 0-1 linear formulations are proposed. The efficiency of the proposed bounds is comprehensively discussed and some efficient dual-ascent algorithms to derive the new bounds are developed. Finally, quadratic shortest path problem is introduced as a novel QP. The NP-hardness of the problem is proved, and some bounding techniques for the problem are presented.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The Quadratic 0-1 program with linear constraints is a very general class of optimization problems and has a wide range of applications. Let n, m_1, m_2 be positive integers, $\mathbb{B} = \{0, 1\}$, $\mathbb{R}$ denote the set of reals, $\mathbb{R}_+$ denote the set of non-negative reals, $Q \in \mathbb{R}^{n\times n}$, $A \in \mathbb{R}^{m_1\times n}$, $D \in \mathbb{R}^{m_2\times n}$ be real matrices and $c \in \mathbb{R}^n_+$, $b \in \mathbb{R}^{m_1}$, and $g \in \mathbb{R}^{m_2}$ be three real vectors. A Quadratic 0-1 Programming with linear constraints is a problem of the following form:

$$\begin{aligned} \text{QP:} \quad \min \quad & x^TQx + c^Tx \\ \text{s.t.} \quad & x \in S \cap \mathbb{B}^n, \end{aligned}$$

where S is described as:

$$S = \{0 \leq x \leq 1 : \; Ax \leq b, \; Dx = g\}.$$

Many combinatorial optimization problems admit natural formulations as quadratic 0-1 programming problems. The Quadratic Assignment Problem (QAP), Quadratic Traveling Salesman Problem (QTSP), Graph Partitioning, Quadratic Knapsack Problem (QKP), and Quadratic Minimum Spanning Tree Problem (QMSTP) are among the well-known particular cases of the QP which arise in a variety of real-world applications. All of these problems are known to be NP-hard in general and therefore the QP is also NP-hard.

Branch-and-bound algorithms are the most successful procedure for solving the QP. The branch-and-bound algorithm is based on decomposition of the original problem into a series of smaller subproblems, then recursively solves each subproblem, and discards the non-optimal solutions by using the best obtained lower estimated bound. In fact this approach behaves well only if one gets tight lower bounds for the objective function. In general, the solution methods of finding a lower bound for the QP can be divided into two main groups: Reformulation-Relaxation approaches, and Reformulation-Decomposition approaches. Reformulation is the first step of the lower bounding procedure for the QP which tries to convert the problem into an equivalent one in such a way that solving the resulting problem be easier or more efficient than the original one. We say that two problems P and P' are equivalent if they admit the same set of solutions in the original variable space and the objective function values are equal at the corresponding solutions. Since in general the QP is a non-linear non convex problem, most of the proposed approaches try to reformulate the problem either as an equivalent Mixed Integer Linear Program (MILP) or as an equivalent quadratic 0-1 program and solve the resulting program by effective algorithms that take the problem structure into account. Based on the structure of the resulting problem different relaxation and/or decomposition methods may be applied to provide a lower bound. If the resulting problem is an MILP, then the most common method to obtain a lower bound is to drop the integrality requirement on the binary variables and solve the resulting linear programming relaxation of the problem. Usually a considerable gap between the optimal value of the MILP and its LP relaxation leads to apply some efficient decomposition methods to improve the LP bounds. *Dantzig-Wolf* decomposition, *Lagrangian relaxation*, and *Cutting plane* methods are among the most well-known decomposition methods in the literature which generate approximations to the convex hull of feasible solutions. If the resulting problem is a quadratic 0-1 Program, then unlike the MILP, using traditional decomposition methods may not be so trivial. Therefore, considering the problem structure, some different relaxations and decompositions approaches may be used.

The basic approach in optimizing a quadratic 0-1 program is to employ an initial linearization to transform the problem into an equivalent linear form. In order to achieve the linearity, auxiliary variables and constraints are added to the problem. However dealing with linear programming-based reformulations, two different issues must be considered: the increasing size of the problem in terms of number of the variables and constraints, and also tightness of the obtained lower bounds. More precisely, two different linearized reformulations P' and P'' of problem P may equivalently depict the same nonlinear program, but their size and continuous relaxations can be drastically differ depending on the manner in which the auxiliary variables and constraints are defined. The standard strategy to linearize the quadratic terms $x_i x_j$ for all $i, j = 1, 2, \dots, n$ is to introduce new binary variables $y_{ij} = x_i x_j$ which satisfy the following set of constraints:

$$y_{ij} \leq x_i, \quad y_{ij} \leq x_j, \text{ and } y_{ij} \geq x_i + x_j - 1.$$

The new formulation requires $O(n^2)$ additional variables and constraints and well-known from the literature (see [57, 62].) In order to improve the standard linearization technique, Glover [56] proposed a new strategy to linearize the quadratic terms $x_i x_j$ through the introduction of n unrestricted continuous variables and $4n$ linear inequalities. Adams et al. [1], Adams and Forrester [2], Chaovalitwonges [29], and Sherali and Smith [112] provide different $O(n)$ linearization approaches.

The reformulation-linearization technique (RLT) is an alternative and successful approach which was first introduced by Adams and Sherali [5] for 0-1 quadratic programming problems and then was generalized for a class of 0-1 mixed integer programming problems by the same authors [6]. The best description of the RLT with complete details can be found in Sherali and Adams [111]. The RLT is a technique for generating an n-level hierarchy of polyhedral representation for linear and polynomial 0-1 programming problems with the n-th level providing an explicit algebraic characterization of the convex hull of feasible solutions. The level of the hierarchy directly corresponds to the degree of the polynomial terms produced during the reformulation stage. Hence, in the reformulation phase, given a value of the level $d \in \{1, \dots, n\}$ the RLT constructs various polynomial factors of degree d comprised of the product of some d binary variables x_j or their complements $(1 - x_j)$. The RLT essentially consists of two steps: (i) a reformulation step, in which nonlinear valid inequalities are generated by combining constraints of the original problem, and (ii) a linearization step in which each product term is replaced by a single continuous variable. Applying RLT to the special cases of the QP (e.i., QAP) leads to tight linear relaxations [3, 4, 60].

Another relevant track of research on QP is to study the polyhedral structure of the set of feasible solutions to improve the strength of the LP-based reformulation bounds. Toward this end, one may construct a polyhedral approximation of the convex hull of the feasible solutions of the LP-based reformulation and then solve the QP over it to obtain a bound. One way to construct such polyhedral approximation is to generate some valid inequalities in a dynamic way by using Cutting-plane methods. Padberg [96] proposed a polytope, called Boolean quadratic polytope, associated to a linearized integer programming formulation of an unconstrained quadratic 0-1 programming and introduced three families of valid and facet-defining inequality for it. There are several papers devoted to study the polyhedral structure of the special cases of the QP [41, 43, 66, 69, 109].

Semidefinite programming (SDP) is another popular approach to generate strong relaxations of the QP. The SDP can be viewed as an extension of linear programming where the nonnegativity constraints are replaced by positive semidefinite (psd) constraints on matrix variables. More precisely for any vector $x \in \mathbb{B}^n$ of decision variables, we first introduce the new matrix $Y = xx^t$, which transform the quadratic function of x to a linear function of Y, and then impose a "rank one" non-convex constraint $Y = xx^t$ to the problem. Because of the non-convexity of the rank one constraint, a relaxation of this constraint is considered such that the resulting problem be a SDP. Applications of SDP for finding strong bounds for the various types of the QPs can be found in [49, 66, 95, 100, 105]. More recently, Billionnet et al. [19] proposed a reformulation approach to the QP based on converting the non-convex quadratic objective function to an equivalent convex quadratic function. The idea is to perturb the objective function with some specific multipliers in such a way that it is convex and the continuous relaxation of the QP is tighter. They showed that the optimal multipliers for the new convex program can be found by solving a SDP.

Decomposition methods provide a different approach to compute a lower bound for QP. Michelon and Maculan [89] applied a Lagrangian decomposition to obtain a lower bound for the integer non-linear programming with linear constraints. In fact they split the original variables into two groups and then joined these two groups of variables with a new "linking" constraint. Dualizing the new problem in the linking constraint, they could decompose the problem into two subproblems: a continuous nonlinear programming and an integer linear programming. Chardaire and Sutter [30] introduced a new decomposition method for unconstrained quadratic 0-1 programming that can be viewed as a more general Lagrangian decomposition where several copies of each variable are added. Following the same idea, Billionnet et al. [20] and [21] applied Lagrangian decomposition methods to the quadratic knapsack problem. Mauri and Lorena [87] introduced a new Lagrangian decomposition method for the unconstrained QP based on a graph partitioning. In fact they first linearized the unconstrained QP as an MILP, represented the resulting MILP by a graph, and then decomposed the MILP on some subproblems through the copy of the edges formed by vertices belonging to different clusters and of their respective vertices. Later, the same authors in [88] proposed a new alternative based on column generation to solve a Lagrangian decomposition approach reported in [87] to find lower bounds and feasible approximate solutions for QP.

The remaining of the thesis consists of two parts. Part I deals with the solution methods for the general case of the QP. We review the most important approaches from two different point of views. In Chapter 2 we present different reformulations and relaxation strategies based on linear and semidefinite programming. More precisely we start with classic linearization methods to obtain a lower bound, and then try to improve the reformulation so that its LP relaxation provides a stronger lower bound in a reasonable time. Moreover we propose the Semidefinite programming as a different approach which can be used either for generate a strong relaxation of the QP or to provide a convex reformulation of the problem. Chapter 3 describes different reformulations of the QP based on an equivalent convex or non-convex quadratic 0-1 programming. After introducing the different QP-based reformulation strategies, we use various decomposition techniques (including Lagrangian decomposition) to obtain a strong lower bound. Part II is concerned with the some special cases of the QPs related to quadratic version of some well-known combinatorial optimization problems. Among the most important classical combinatorial optimization problems, we study the quadratic assignment problem (QAP), quadratic minimum spanning tree problem (QMSTP), quadratic traveling salesman problem (QTSP) and finally, quadratic shortest path problem (QSP).

In Chapter 4 we consider the QAP as a classical difficult combinatorial optimization. Due to its wide verity of applications and its resistance to solution strategies, numerous researchers have studied the QAP and proposed both heuristic and exact solution methods. We review different reformulations and lower bounding procedure for the problem and then dedicate our efforts to analysis and the interpretation of lower bounds. Moreover, we present a novel revised version of the level-d RLT representation with a reduced number of constraints and provide a decomposition to solve the new MIP formulation.

In Chapter 5 we study two special cases of the QAP including the Adjacent QAP and QAP on reducible graphs. The Adjacent Quadratic Assignment Problem (AQAP) is a variant of the QAP where the cost coefficient matrix has a particular structure. Motivated by strong lower bounds obtained by applying RLT to the classical QAP, we propose two special RLT representations for the problem. The first is based on a "flow" formulation whose linear relaxation can be solved very efficiently for large instances while the second one has significantly more variables and constraints, but possesses some desirable properties relative to the constraint set. For the QAP on reducible graph we give a Lagrangian decomposition based on splitting the variables and then dualizing the copy constraint so that the resulting problem can be decompose to two quadratic semi-assignment problems.

Chapter 6 is concerned with the QTSP. The QTSP is as a variant of the classical Traveling Salesman Problem (TSP) whose costs are associated with each two edge that are traversed in succession. In this chapter we first present the problem statement and propose some linearized integer formulations for both symmetric and asymmetric version of the problem. In order to obtain a tight lower bound to the problem we provide a Linear Programming formulation for the general QTSP that has a variable for each cycle in the given graph. Then we apply a Column Generation approach to solve the new LP representation whose pricing sub-problem is considered as a resource-constrained elementary shortest path.

The QMSTP is a variant of the minimum spanning tree problem (MSTP) whose cost is not only depend on an edge-cost but also the interaction cost between every pairs of edge. In Chapter 7 we review different strategies found in the literature to compute the lower bound for the QMSTP. We develop new bounds based on a reformulation scheme and some new mixed 0-1 linear formulations that result from a reformulation-linearization technique (RLT). To derive new lower bounds we provide some efficient dual-ascent algorithms and compare the new bounds with the other bounding procedures in terms of both overall strength and computational effort.

Finally, in Chapter 8 we study the novel QSP which calls for the minimization of a quadratic objective function subject to shortest-path constraints. We first show that the problem is NP-hard. Then we propose and evaluate bounding techniques which exploit the structure of the problem. Finally, we present the possible polynomially solvable cases of the problem.

Part I

Quadratic 0-1 Programming

Chapter 2

Reformulation and Relaxations Strategies for the QP

The goal of this chapter is to show the numerous ways in which, given an initial formulation of the QP, problem structure can be used to obtain a reformulation which can be solved in an efficient way. In the first part of the chapter we present the LP-based reformulations which are the most frequently used approaches. These approaches consist in reformulating the problem as an equivalent MILP and solving the Linear programming relaxation of the latter problem to obtain a lower bound for the former. However, since solving the LP relaxation of the linearized formulation usually does not provide a strong lower bound, in the second part of the chapter we turn our attention to Semidefinite programming (SDP). SDP is a different approach which can be used either for generate a strong relaxation of the QP or to provide a convex reformulation of the QP.

2.1 LP-based reformulation relaxation

Several linearization strategies have been proposed in the literature for reformulating the quadratic 0-1 program as equivalent mixed integer programming. In the following, we present the most important strategies in terms of tightness of the bounds and also computational performance.

2.1.1 Standard linearization

The standard linearization (see [57, 62]) of the QP amounts to replace the quadratic terms $x_i x_j$ $\forall i, j = 1, 2, \ldots, n, i \neq j$, by a new binary variable y_{ij} such that $y_{ij} = 1$ if and only if both x_i and x_j are equal to one. The linearized model has the following form:

$$\begin{aligned}
\text{LQP1:} \quad \min \quad & \sum_i \sum_j q_{ij} y_{ij} + \sum_i c_i x_i \\
\text{s.t.} \quad & y_{ij} \le x_i \quad \forall i, j = 1, 2, \ldots, n, i \ne j & (2.1)\\
& y_{ij} \le x_j \quad \forall i, j = 1, 2, \ldots, n, i \ne j & (2.2)\\
& y_{ij} \ge x_i + x_j - 1 \quad \forall i, j = 1, 2, \ldots, n, i \ne j & (2.3)\\
& y \in \mathbb{B}^n & (2.4)\\
& x \in S \cap \mathbb{B}^n.
\end{aligned}$$

This model contains $O(n^2)$ additional constraints (2.1) to (2.4) to guarantee the equation $y_{ij} = x_i x_j$. Solving the Lp relaxation of LQP1 by a general-purpose solver provides a lower bound to the objective value of QP.

2.1.2 Glover linearization

Glover in [56] introduced an equivalent MILP formulation of QP by defining a new continuous variable $y_i = x_i f_i(x)$ for each $i = 1, 2, \ldots, n$, where $f_i(x) = \sum_j q_{ij} x_j$. By adding some additional constraints to enforce the equality $y_i = x_i f_i(x)$ for each i, the following problem implies:

LQP2:

$$\begin{aligned}
\min \quad & \sum_i y_i + \sum_i c_i x_i \\
\text{s.t.} \quad & L_i x_i \le y_i \le U_i x_i \quad \forall i = 1, 2, \ldots, n & (2.5)\\
& f_i(x) - U_i(1 - x_i) \le y_i \le f_i(x) - L_i(1 - x_i) \quad \forall i = 1, 2, \ldots, n & (2.6)\\
& x \in S \cap \mathbb{B}^n.
\end{aligned}$$

where for each $i \in \{1, 2, \ldots, n\}$, L_i and U_i are defined as follows:

$$L_i = \sum_j \min(q_{ij}, 0) \quad \text{and} \quad U_i = \sum_j \max(q_{ij}, 0). \tag{2.7}$$

Note that the problem QP and LQP2 are equivalent in a sense that for any feasible solution x of QP there exists a y such that (x, y) is a feasible solution to LQP2 with the same objective value. Conversely, for any feasible solution (x, y) to LQP2, the corresponding x is feasible to QP with the same objective value.

2.1.3 Revised Glover linearization

Motivated by desire to find an equivalent linearized model to the QP with less variables and constraints, Adams and Forrester [1] proposed a revised version of LQP2 with just n additional constraints. In this approach L_i and U_i are the lower and upper bounds on the linear functions $f_i(x)$ which are calculated as follows:

$$L_i = \min\{f_i(x) : x \in S\} \quad \text{and} \quad U_i = \max\{f_i(x) : x \in S\}.$$

The following theorem provides the reason of revising the Golver model.

Theorem 2.1.1 (Adams and Forrester [1]). *Removing the right hand side of constraints* (2.5) *and* (2.6) *does not have any affect on the minimum value of the LQP2.*

Proof. The proof follows from the fact that variable y has the non-negative coefficient in the objective function and just appears in constraints (2.5) and (2.6). Let the objective coefficient of y_i be positive, then by removing the right-hand sides of these two constraints, an optimal solution must be $y_i = \max\{L_i x_i, f_i(x) - U_i(1 - x_i)\}$. Considering $0 \leq x \leq 1$, we have

$$y_i = \max\{L_i x_i, f_i(x) - U_i(1 - x_i)\} \leq \min\{U_i x_i, f_i(x) - L_i(1 - x_i)\}. \tag{2.8}$$

Not that for variable y_i with zero coefficient, there exists an optimal y_i that satisfies (2.8). □

As a consequence of Theorem 2.1.1 and substituting $z_i = y_i - L_i x_i$ in the constraints and the objective function of LQP2 a revised form of the Glover model is obtained:

$$\begin{aligned} \text{LQP3:} \quad \min \quad & \sum_i (z_i + L_i x_i) + \sum_i c_i x_i \\ \text{s.t.} \quad & z_i \geq f_i(x) - U_i(1 - x_i) - L_i x_i \quad \forall i = 1, 2, \ldots, n \\ & z \geq 0, \; x \in S \cap \mathbb{B}^n. \end{aligned}$$

Although the strength of the lower bound obtained by the continuous relaxation of LQP3 is the same as the provided bound by LQP2, but the number of additional constraints from $4n$ have been reduced to n.

2.1.4 Enhanced Glover linearization

In order to improve the revised Glover model in sense of the quality of the lower bound let us consider the basic model of LQP3 with additional assumption $q_{ii} = 0$. Note that this assumption is not restrictive, since $q_{ii}x_i^2 = q_{ii}x_i$.

$$\begin{aligned} \min \quad & \sum_i (y_i + c_i x_i) \\ \text{s.t.} \quad & y_i \geq \overline{f}_i(x) - \overline{U}_i(1 - x_i) \quad \forall i = 1, 2, \ldots, n && (2.9) \\ & y_i \geq \overline{L}_i x_i \quad \forall i = 1, 2, \ldots, n && (2.10) \\ & x \in S \cap \mathbb{B}^n. && (2.11) \end{aligned}$$

where $\overline{f}_i(x) = \sum_{j \neq i} q_{ij}$, $\overline{U}_i = \max\{\overline{f}_i(x) : x \in X,\ x_i = 1\}$ and $\overline{L}_i = \min\{\overline{f}_i(x) : x \in X,\ x_i = 1\}$.

Let $c_i + \sum_{j \neq i} q_{ij} = \sum_j q_{ij}$. Then $\overline{U}_i = \max\{f_i(x) - c_i x_i : x \in X,\ x_i = 1\}$ and $\overline{L}_i = \min\{f_i(x) - c_i x_i : x \in X,\ x_i = 1\}$. Thus $U_i = c_i + \overline{U}_i$ and $L_i = c_i + \overline{L}_i$. By adding $c_i x_i$ to the both sides of (2.9) we have

$$\begin{aligned} y_i + c_i x_i \geq & \overline{f}_i(x) - \overline{U}_i(1 - x_i) + c_i x_i = (f_i(x) - c_i x_i) - (U_i - c_i)(1 - x_i) \\ = & f_i(x) - U_i(1 - x_i) + c_i(1 - x_i). && (2.12) \end{aligned}$$

Similarly, adding $c_i x_i$ to the both side of (2.10) implies:

$$y_i + c_i x_i \geq \overline{L}_i x_i + c_i x_i = (L_i - c_i)x_i + c_i x_i = L_i x_i \qquad (2.13)$$

Considering (2.12) and (2.13), and replacing $y_i + c_i x_i$ by $\tilde{z}_i$ we have

$$\begin{aligned} & \tilde{z}_i \geq f_i(x) - U_i(1 - x_i) + c_i(1 - x_i) && (2.14) \\ & \tilde{z}_i \geq L_i x_i. && (2.15) \end{aligned}$$

Substituting $z_i = \tilde{z}_i - L_i x_i$ in (2.14) and (2.15) we obtain a new linearized model with just n additional constraints as shown in LQP4.

$$\begin{aligned} \text{LQP4:} \quad \min \quad & \sum_i (z_i + L_i x_i) + \sum_i c_i x_i \\ \text{s.t.} \quad & z_i \geq f_i(x) - U_i(1 - x_i) - L_i x_i + c_i(1 - x_i) \quad \forall i = 1, 2, \ldots, n \\ & z \geq 0,\ x \in S \cap \mathbb{B}^n. \end{aligned}$$

Problems QP and LQP4 are equivalent in the sense that for each optimal solution to one problem, there exists an optimal solution to the other problem having the same optimal objective value.

Theorem 2.1.2. *Continuous relaxation of LQP4 provides a lower bound on QP at least as large as the continuous relaxation of LQP3.*

Proof. It is sufficient to show any feasible solution of LQP4 is also feasible in LQP3, which follows from the fact that $f_i(x)-U_i(1-x_i)-L_ix_i+c_i(1-x_i) \geq f_i(x)-U_i(1-x_i)-L_ix_i$ since $c \in \mathbb{R}_+{}^n$. $\square$

2.1.5 Some additional linearization models

Chaovalitwongse et al. [29] introduced a new linearization technique with $O(n)$ additional continuous variables and constraints. The new linearized model has the following form:

$$\begin{aligned}
\text{LQP5:} \quad \min \quad & \sum_i z_i + \sum_i (c_i - M)x_i \\
\text{s.t.} \quad & \sum_j q_{ij} - y_i - z_i + M = 0 \quad \forall i = 1,2,\dots,n \\
& y_i \leq 2M(1-x_i) \quad \forall i = 1,2,\dots,n \\
& z, y \ \geq 0, \ x \in S \cap \mathbb{B}^n.
\end{aligned}$$

where $M = \max_i\{\sum_j |q_{ij}| + |c_i|\}$.

Theorem 2.1.3 (Chaovalitwongse et al. [29]). *problems QP and LQP5 are equivalent,i.e., QP has an optimal solution x if and only if there exist y, z such that (x, y, z) is an optimal solution of LQP5.*

Sherali and Smith [112] linearized the QP in a more general form than LQP5 whose size is linear in terms of the number of variables. LQP6 gives the new linearization:

$$\begin{aligned}
\text{LQP6:} \quad \min \quad & \sum_i z_i + \sum_i c_i x_i \\
\text{s.t.} \quad & \sum_j q_{ij} - y_i - z_i = 0 \quad \forall i = 1,2,\dots,n \\
& y_i \leq U_i(1-x_i) \quad \forall i = 1,2,\dots,n \\
& z, y \ \geq 0, \ x \in S \cap \mathbb{B}^n.
\end{aligned}$$

where U_i is defined as (2.7). The authors proved that for a specific value of $U_i = \sum_j q_{ij}$ for each i, the lower bound obtained by solving the continuous relaxation of the standard linearized model, LQP1, is tighter than then one obtained by continuous relaxation of LQP6. Moreover they mentioned that by taking $U_i = M = \max_i\{\sum_j |q_{ij}| + |c_i|\} \ \forall i$, the LQP6 is exactly the same as LQP5, i.e., LQP5 is a special case of LQP6.

2.1.6 Reformulation-linearization technique

The reformulation-linearization technique (RLT) is a successful approach which was introduced by Adams and Sherali [5] for 0-1 quadratic programming problems. The application of RLT to QP, transforms the problem into an MILP via two basic steps of *reformulation* and *linearization*. In the reformulation step, the problem is reformulated by constructing redundant nonlinear restrictions, obtained by multiplying the constraints of S by product factors of the binary variables and their complements, while in the linearization step, the objective and constraints of the reformulated problem is linearized by substituting a continuous variable for each distinct nonlinear term. Depending on the product factors used to compute the nonlinear restrictions, different level of RLT can be obtained. Each level of the hierarchy provides a program whose continuous relaxation is at least as tight as the previous level, with the highest level giving a convex hull representation.

We concern ourselves in this section with the level-1 RLT formulation of the QP which is generated via the following two steps:

Reformulation. Multiply each of the m_1 inequities $Ax \leq b$ defining S by each binary variable x_j and its complement $(1 - x_j)$ and each of the m_2 equalities $Dx = g$ by each binary variable x_j, for all $j = 1, 2, \dots, n$. Append these new restrictions to S and Substitute $x_j^2 = x_j \ \forall j$ throughout the constraints and objective function.

Linearization. Linearize the resulting problem by substituting, for every occurrence of each product $x_i x_j$ with $i \neq j$ the continuous variable y_{ij}. Enforce constraints $y_{ij} = y_{ji}$ for all (i, j), with $i < j$.

The resulting level-1 RLT formulation is as follows:

$$
\begin{aligned}
\text{RLTQP:} \quad \min \quad & \sum_i \sum_{j \neq i} q_{ij} y_{ij} + \sum_i c_i x_i \\
\text{s.t.} \quad & \sum_{i \neq j} a_{hi} y_{ij} \leq (b_h - a_{hj}) x_j \quad \forall j = 1, 2, \dots, n, \ h = 1, 2, \dots, m_1 \\
& \sum_{i \neq j} a_{hi} (x_i - y_{ij}) \leq (b_h - a_{hj})(1 - x_j) \quad \forall j = 1, 2, \dots, n \\
& \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \forall h = 1, 2, \dots, m_1 \\
& \sum_{i \neq j} d_{hi} y_{ij} = g_h x_j \quad \forall j = 1, 2, \dots, n, \ h = 1, 2, \dots, m_2 \\
& y_{ij} = y_{ji} \quad \forall i, j = 1, 2, \dots, n, \ i < j \\
& y_{ij} \geq 0 \quad \forall i, j = 1, 2, \dots, n, \ i \neq j \\
& x \in S \cap \mathbb{B}^n.
\end{aligned}
$$

Problem QP and RLTQP are equivalent in the following sense. Given any feasible solution x to the QP, there exists a y such that (x, y) is feasible to the RLTQP with the same objective value. Conversely, for any feasible solution (x, y) to the RLTQP, the corresponding x is feasible to the QP with the same objective value. The proof is straightforward and follows the RLT theory [5, 6].

2.2 Semidefinte programming

Semidefinte programming (SDP) is a general form of linear programming which enables us to specify in addition to a set of linear constraints a special form of nonlinear constraint called “Semidefinte” constraint. In general a SDP has the following form:

$$\begin{aligned} \text{SDP:} \quad \min \quad & Q^0 \bullet X \\ \text{s.t.} \quad & Q^k \bullet X = b_k \quad \forall k = 1, 2, \ldots, m \\ & X \succcurlyeq 0. \end{aligned}$$

where Q^k and X are $n \times n$ matrices, X is symmetric, b_k are scalars, and $Q^k \bullet X = \sum_{i=1}^{n} \sum_{j=1}^{n} Q_{ij}^k X_{ij}$. The inequality sign in $X \succcurlyeq 0$ means that X is positive semidefinite, i.e., $y^T X y > 0$ for all $y \in \mathbb{R}^n$.

The SDP program is a convex optimization problem and can be solved very efficiently, both in theory and in practice. The most common approach to solve the SDP is the interior point method. Interior point methods were first introduced by Karmarkar in [72] for LPs and then were generalized by Alizadeh [8] and Kamath and Karmarkar [71] from linear programming to semidefinite programming. For more recent papers on interior-point methods for semidefinite programming we refer the reader to [10, 40, 47, 65, 68, 90].

Semidefinte programming has variety of applications in combinatorial optimization. Lovász in [81] applied the semidefinte programming to approximate the clique number and chromatic number of graphs. After the first application of SDP to graph optimization problem [81], Alizadeh was among the first researchers to realize the potential of SDP in connection with integer programming [9]. Since then, semidefinte programming has been applied to derive strong tractable convex relaxations of variety of NP-hard optimization problems including various quadratic optimization problems [49, 64, 66, 82, 95, 100]. More recently Billionnet et al. [19] applied SDP to reformulating the quadratic 0-1 programming with linear constraints into an equivalent 0-1 program with a convex quadratic objective function. The idea is to perturb the objective function in such a way that it is convex and the continuous relaxation of the 0-1 QP is tighter. Their method which called *quadratic convex relaxation* (QCR) has the advantage that it can be applied also for the convex instances and it improves the bound obtained by solving the continuous relaxation of the instance.

In the following subsections we present the applications of the SDP as a relaxations of the QP and also as a reformulation technique.

2.2.1 Semidefinte realaxation for the QP

Consider the following form of the QP

$$\begin{aligned} \text{QP:} \quad \min \quad & x^T Q x + c^T x \\ \text{s.t.} \quad & Ax \le b, \; Dx = g \\ & x_i^2 = x_i \quad \forall i. \end{aligned}$$

where constraints $x \in \mathbb{B}^n$ have been replaced by the non-convex quadratic constraints $x_i^2 = x_i$. By defining an $n \times n$ symmetric matrix $X = xx^T$ and denoting the diagonal of this matrix by x we can rewrite the QP as follows:

$$\begin{aligned} \min \quad & Q \bullet X + c^T x \\ \text{s.t.} \quad & Ax \le b, \; Dx = g \\ & X = xx^T \\ & diag(X) = x. \end{aligned}$$

Relaxing constraint $X - xx^T = 0$ to $X - xx^T \succcurlyeq 0$, i.e., $X - xx^T$ be positive semidefinite, implies the following SDP:

$$\text{SDR:} \quad \min \left\{ Q \bullet X + c^T x : \quad Ax \le b, \; Dx = g, \; diag(X) = x, \; Y \succcurlyeq 0 \right\} \tag{2.16}$$

where

$$Y = \begin{pmatrix} X & x \\ x & 1 \end{pmatrix}.$$

The solution of the simidefinite programming SDR provides a lower bound on the QP. In order to tightening the above simidefinite relaxation one can add some additional valid equalities and inequalities to the SDR by considering the general idea presented in [82, 111]. Consider constraints $a^h x \le b_h$ for each h and multiply either by x_j or $(1 - x_j)$, $\forall j$, to obtain the following inequalities:

$$\sum_i a_{hi} X_{ij} \le b_h x_j \tag{2.17}$$

$$\sum_i a_{hi}(x_i - X_{ij}) \le b_h (1 - x_j). \tag{2.18}$$

Replacing $a^h x \leq b_h$ in (2.16) by these two new inequities (2.17) and (2.18) a revised form of SDR with tighter bound is found.

$$\begin{aligned}
\text{RSDR1:} \quad \min \quad & Q \bullet X + c^T x \\
\text{s.t.} \quad & \sum_i a_{hi} X_{ij} \leq b_h x_j \quad \forall (j,h) \\
& \sum_i a_{hi} (x_i - X_{ij}) \leq b_h (1 - x_j) \quad \forall (j,h) \\
& Dx = g,\ diag(X) = x,\ Y \succcurlyeq 0.
\end{aligned}$$

In a similar way, any linear constraints $Dx = g$ can be multiplied by any variable x_j to yield a valid quadratic constraint. Imposing this new quadratic constraint to the SDR provides a different relaxation as follows:

$$\begin{aligned}
\text{RSDR2:} \quad \min \quad & Q \bullet X + c^T x \\
\text{s.t.} \quad & \sum_i d_{hi} X_{ij} - g_h x_j = 0 \quad \forall (j,h) \qquad (2.19) \\
& Ax \leq b,\ Dx = g,\ diag(X) = x,\ Y \succcurlyeq 0.
\end{aligned}$$

Another strategy to strengthen the SDR relaxation as proposed in [100], [95], is to include the following squared-norm in the model.

$$Dx = g \iff \|Dx - g\|^2 = 0 \iff DD^T \bullet X - 2g^T Dx + \|g\|^2 = 0$$

Faye and Roupin [39] proved that, replacing nm_2 constraints (2.19) by a single constraint $DD^T \bullet X - 2g^T Dx + \|g\|^2$ the feasible region of the resulting problem is equal to the feasible region of the RSDR2.

Roupin in [107] proposed a different quadratic constraint $DD^T \bullet xx^T = g^2$ with the assumption of $g \geq 0$ and has been proved that

$$\begin{aligned}
&\left\{(X,x) :\ X - xx^T \succcurlyeq 0,\ DD^T \bullet X - 2g^T Dx + g^2 = 0\right\} = \\
&\left\{(X,x) :\ X - xx^T \succcurlyeq 0, Dx = g,\ DD^T \bullet xx^T = g^2\right\}.
\end{aligned}$$

Anstreicher in [12] combined the RLT and SDP, to obtain a relaxation that dominates those obtained by using either technique alone. The author shows that this can yield significant benefits in terms of bound strength, though computational performance can be low.

2.2.2 SDP-based reformulation for the QP

In the previous section we explained how to relax the quadratic 0-1 programming into a semidefinte programming. In this section we present a different application of SDP to convert the QP with equality constraints into a quadratic convex formulation proposed by Billionnet et al. [19]. The general idea is to solve QP by reformulating the problem into an equivalent 0-1 program with a convex quadratic objective function, followed by the use of a standard mixed integer quadratic programming solver. This convexification method which is called Quadratic Convex Reformulation (QCR), uses the equality constraints of QP and requires the solution of a semidefinite program.

Given a QP, the QCR method perturbs the objective by adding terms of the form $\alpha_i(x_i^2 - x_i)$ and $\sum_j \beta_{kj}(\sum_i d_{ki}x_i - g_k)x_j$ where $\alpha \in \mathbb{R}^n$ and $\beta \in \mathbb{R}^{n\times n}$. The resulting problem $QP_{\alpha,\beta}$ is written as follows:

$$\begin{aligned} QP_{\alpha,\beta} \quad & \min \quad x^T Q_{\alpha,\beta} x + c_{\alpha,\beta}^T x \\ & \text{s.t.} \quad x \in S \cap \mathbb{B}^n. \end{aligned}$$

where

$$\begin{aligned} Q_{\alpha,\beta} &= Q + \frac{1}{2}(\beta^T D + D^T \beta) + Diag(\alpha) \\ c_{\alpha,\beta} &= c - \beta^T g - \alpha. \end{aligned}$$

$Diag(\alpha) \in \mathbb{R}^{n\times n}$ is a diagonal matrix with the elements of α on the diagonal. Considering $x \in \mathbb{B}^n$ and $Dx = g$ it is obvious that $QP_{\alpha,\beta}$ and QP are equivalent.

A natural question arises here about how to generate the values for α and β so that $QP_{\alpha,\beta}$ is convex and its continuous relaxation provides a lower bound as tight as possible. in doing so, we are interested in solving the following problem:

$$\text{CQP}: \max_{\substack{\alpha,\beta \\ Q_{\alpha,\beta}\succcurlyeq 0}} \quad \min\{x^t Q_{\alpha,\beta} x + c_{\alpha,\beta}^T x : x \in S\}.$$

Theorem 2.2.1 (Billionnet et al. [19]). *The optimum value of CQP is equal to the optimum value of the semidefinte programming RSDR2. Moreover the optimal value β^* is given by the optimal value of the dual variables associated with constraints $diag(X) = x$, and optimal values α^* are given by the optimal values of the dual variables associated with constraints* (2.19).

In general the QCR method find a lower bound on the QP in two steps: first solves a SDP to obtain the optimal values of α^* ad β^*, and then to by using a general-purpose (MIQP) solver, solves the continuous relaxation of the convex quadratic problem QP_{α^*,β^*}.

Chapter 3

Reformulation and Decomposition for the QP

In this chapter we present the general principle of reformulation-decomposition for the QP which consists of reformulating the QP as an equivalent quadratic programming, and then decomposing the resulting problem to obtain a strong bound for the original problem. Decomposition is a general approach of solving a problem by splitting it into smaller subproblems and solving each of the smaller subproblems separately, either in parallel or sequentially. In the first part of the chapter we present the Lagrangian decomposition approaches applied to both the QP and its MILP formulation, while in the second part of the chapter we introduced some QP-based reformulation strategies for the QP, and then provide a simple decomposition strategy for the resulting problem.

3.1 Reformulation and Lagrangian decomposition

Decomposition methods are general strategies to solve a problem by breaking it up into smaller ones and solving each of the smaller ones separately, either in parallel or sequentially. These methods can be applied for the problem with special structure called block-separable, i.e., the variables can be partitioned into p subsets such that the objective is a sum of functions over each subset, and each constraint involves only variables from one of the subsets. More precisely we have the following definition:

Definition 3.1.1 (Nowak [93]). *Problem* $P : \min\{F_0(x) : F_i(x) \leq 0, \forall i = 1, \ldots, m,\, l_j \leq x_j \leq u_j\, \forall j = 1, \ldots, n\}$ *is called block-separable, if there exists a partition* $\{I_1, \ldots, I_p\}$ *of*

$N = \{1, \ldots, n\}$ with $\cup_{k=i}^{p} I_k = N$ and $I_k \cap I_l = \emptyset \; \forall k, l, k \neq l$ such that

$$F_i(x) = \sum_{k=1}^{p} F_{i,k}(x_{I_k}), \quad i = 0, \ldots, m. \tag{3.1}$$

where x_{I_k} is a vector including all $x_i \in I_k$, and function $F_{i,k}(x_{I_k})$ is a function of x_{I_k}. If the size of all I_k is one, i.e., $I_k = \{k\}$, then for $k = 1, \ldots, p$, problem P is called separable. In this case we have

$$F_i(x) = \sum_{k=1}^{n} F_{i,k}(x_k), \quad i = 0, \ldots, m. \tag{3.2}$$

Since, in general, problem QP is not a block-separable problem, some extra efforts should be considered to reformulate it into a block-separable problem. In the following we present the decomposition strategies applied for both the original QP and its MILP formulation.

3.1.1 Lagrangian decomposition for the QP

Consider the extended version of problem QP as follows:

$$\begin{aligned} \min \quad & \sum_i \sum_j q_{ij} x_i x_j + \sum_i c_i x_i \\ \text{s.t.} \quad & \sum_j a_{hj} \leq b_i \quad h = 1, 2, \ldots, m_1 \\ & \sum_j d_{lj} = g_l \quad l = 1, 2, \ldots, m_2 \\ & x \in \mathbb{B}^n. \end{aligned}$$

In order to use a decomposition scheme, we first reformulate the problem into a block-separable problem and then apply Lagrangian relaxation to solve the resulting problem. Let $I_1, \ldots, I_p$ be an arbitrary partition of $N = \{1, \ldots, n\}$ into p subsets such that $I_k \cap I_l = \emptyset \; \forall k, l, k \neq l$ and $\cup_{k=i}^{p} I_k = N$. Creating a copy y of each variable x as

$$y_j^k = x_j \quad \forall j \in N \setminus I_k$$

and leting

$$f_k(x, y) = \sum_{i \in I_k} \sum_{j \in I_k} q_{ij} x_i x_j + \sum_{i \in I_k} \sum_{j \in N \setminus I_k} q_{ij} x_i y_j^k + \sum_{i \in I_k} c_i x_i.$$

the problem is rewritten as follows:

$$\begin{aligned}
\min \quad & \sum_k f_k(x,y) && (3.3)\\
\text{s.t.} \quad & \sum_{j\in I_k} a_{hj}x_j + \sum_{j\in N\setminus I_k} a_{hj}y_i^k \le b_h \quad k=1,\dots,p, h=1,2,\dots,m_1 && (3.4)\\
& \sum_{j\in I_k} d_{lj}x_j + \sum_{j\in N\setminus I_k} d_{lj}y_j^k = g_l \quad k=1,\dots,p, l=1,2,\dots,m_2 && (3.5)\\
& y_j^k = x_j \quad \forall k=1,\dots,p, j\in N\setminus I_k && (3.6)\\
& x\in \mathbb{B}^{|I_k|} \quad \forall k=1,\dots,p && (3.7)\\
& y^k \in \mathbb{B}^{|N\setminus I_k|} \quad \forall k=1,\dots,p. && (3.8)
\end{aligned}$$

Now we consider a relaxation of this problem by relaxing the copy constraints (3.6) using the multipliers $\lambda_j^k\ \forall k=1,\dots,p, j\in N\setminus I_k$. The resulting Lagrangian relaxation is:

$$\text{RQP1:} \quad \min\left\{ \sum_k \left(f_k(x,y) + \sum_{j\in N\setminus I_k} \lambda_j^k (x_j - y_i^k)\right) : (3.4),(3.5),(3.7),(3.8) \right\}.$$

Problem RQP1 can be decomposed into p subproblems $RQP1^k$. For each $k=1,\dots,p$, if variables x_i, $i\in I_k$ are fixed at 0 or 1, each subproblem $RQP1^k$ is an Integer linear programming on variables y^k. Since Integer linear programming is in general an NP-hard optimization problem, one can consider a continuous relaxation of the $RQP1^k$ by dropping the integrality restriction over the variables y^k, and then imposing the following constraints to strengthen the lower bound quality:

$$x_i y_j^k = x_j y_i^l \qquad \forall k,l=1,\dots,p,\ k\neq l,\ i\in I_k,\ j\in I_l.$$

Relaxing these new constraints with Lagrangian multipliers $\mu_{ij}\ \forall k,l=1,\dots,p,\ k\neq l,\ i\in I_k,\ j\in I_l$, we obtain the following Lagrangian dual problem:

$$\max_{\lambda,\mu} \sum_{k=1}^{p} \Theta_k^*(\lambda,\mu)$$

where for each $k, l = 1, \dots, p$, $\Theta_k^*(\lambda, \mu)$ is the optimal value of the below subproblem:

$$\begin{aligned}
\Theta_k(\lambda,\mu): \quad \min \quad & \sum_{i\in I_k}\sum_{j\in I_k} q_{ij}x_ix_j + \sum_{i\in I_k}\sum_{j\in N\setminus I_k}(q_{ij}+\mu_{ij}-\mu_{ji})x_iy_j^k \\
& + \sum_{i\in I_k}(c_i + \sum_{l\neq k}\lambda_i^k)x_i - \sum_{j\in N\setminus I_k} y_j^k \\
\text{s.t.} \quad & \sum_{j\in I_k} a_{hj}x_j + \sum_{j\in N\setminus I_k} a_{hj}y_i^k \le b_h \quad h = 1,2,\dots,m_1 \\
& \sum_{j\in I_k} d_{lj}x_j + \sum_{j\in N\setminus I_k} d_{lj}y_j^k = g_k \quad l = 1,2,\dots,m_2 \\
& x \in \mathbb{B}^{|I_k|} \\
& 0 \le y^k \le 1.
\end{aligned}$$

Note that for given λ and μ, and fixed values $(0$ or $1)$ of the $x_j \in I_k$ variables, problem $\Theta_k(\lambda, \mu)$ is a linear programming problem on variables y^k which can be solved in an efficient way.

We note that the quality of the bound depends not only on the size of the subsets and the quality of the Lagrangian multipliers, but also on how elements are assigned to the subsets. It is however, not clear which assignment of elements to groups that result in the tightest bound. The assignment of elements to the subsets can be done by considering the problem structure, but in general it can be done at random.

3.1.2 Lagrangian decomposition for the linearized QP

Mauri et al. [87] proposed a decomposition strategy for the unconstrained quadratic 0-1 programming which, unlike the reformulation-decomposition strategy proposed in the previous section, uses an MILP reformulation of the QP to perform the proposed decomposition. Consider a graph $G = (N, E)$ and adjacency matrix $\overline{E} \in \mathbb{R}^{n\times n}$ whose elements $e_{ij} = 1$ if $q_{ij} \neq 0$ and $e_{ij} = 0$ if $q_{ij} = 0$. This graph is partitioned into $p \le n$ clusters $V_1, \dots, V_p$, $V_k \cap V_l = \emptyset \ \forall k, l, k \neq l$; $G_k = (V_k, E_k) \ \forall k = 1, \dots, p$. Below is the MILP formulation of the QP obtained by standard linearization presented in Section 2.1.1 and the graph partitioning:

$$
\begin{aligned}
\min \quad & \sum_{k=1}^{p} \left(\sum_{i \in V_k} \sum_{\substack{j \in V_k \\ i \neq j}} q_{ij} w_{ij} + \sum_{i \in V_k} \sum_{\substack{j \in N \setminus V_k \\ i \neq j}} q_{ij} z_{ij} + \sum_{i \in V_k} c_i x_i \right) \\
\text{s.t.} \quad & w_{ij} \leq x_i \quad \forall k = 1, \ldots, p, i \in V_k, j \in V_k, i \neq j, q_{ij} > 0 && (3.9) \\
& w_{ij} \leq x_j \quad \forall k = 1, \ldots, p, i \in V_k, j \in V_k, i \neq j, q_{ij} > 0 && (3.10) \\
& w_{ij} \geq x_i + x_j - 1 \quad k = 1, \ldots, p, i \in V_k, j \in V_k, i \neq j, q_{ij} < 0 && (3.11) \\
& w_{ij} \geq 0 \quad k = 1, \ldots, p, i \in V_k, j \in V_k, i \neq j, q_{ij} < 0 && (3.12) \\
& z_{ij} \leq y_i^k \quad \forall k = 1, \ldots, p, i \in V_k, j \in N \setminus V_k, i \neq j, q_{ij} > 0 && (3.13) \\
& z_{ij} \leq y_j^k \quad \forall k = 1, \ldots, p, i \in V_k, j \in N \setminus V_k, i \neq j, q_{ij} > 0 && (3.14) \\
& z_{ij} \geq y_i^k + x_j - 1 \quad k = 1, \ldots, p, i \in V_k, j \in N \setminus V_k, i \neq j, q_{ij} < 0 && (3.15) \\
& z_{ij} \geq 0 \quad k = 1, \ldots, p, j \in N \setminus V_k && (3.16) \\
& x_j = y_j^k \quad \forall k = 1, \ldots, p, j \in N \setminus I_k && (3.17) \\
& z_{ij} = z_{ji} \quad k = 1, \ldots, p, i \in V_k, j \in N \setminus V_k, j > i && (3.18) \\
& x_i \in \mathbb{B} \quad k = 1, \ldots, p, i \in V_k. && (3.19)
\end{aligned}
$$

The variables w_{ij} represent the edges between the vertices i and j, while the variables z_{ij} shows edges between the vertex i and the copy of vertex j. Constraints (3.9) to (3.12) are related to the edge (i, j) inside the cluster k, and constraints (3.13) to (3.16) are related to the edge (i, j) with vertices in different clusters.

Dualizing constraints (3.17) and (3.18) with multiplier α and β in a Lagrangian fashion, the resulting problem can be partitioned into p independent subproblems, so the Lagrangian dual is presented as:

$$\max_{\alpha, \beta} \sum_{k=1}^{p} \Phi_k^*(\alpha, \beta)$$

where $\Phi_k^*(\alpha, \beta)$ is the optimal value of the subproblem k for each $k = 1, \ldots, p$. In order to solve the Lagrangian dual, the authors applied a subgradient algorithm, and used the graph partitioning scheme presented in [73] for the splitting procedure.

Using the same idea described above for the unconstrained quadratic 0-1 programming, we apply Lagrangian decomposition to a linearized version of the QP. For a given partition $I_1, \ldots, I_p$ of N, consider the following relaxation of QP:

$$\text{RQP2:} \quad \min \quad \sum_k \left(\sum_{i\in I_k} \sum_{j\in I_k} q_{ij}x_ix_j + \sum_{i\in I_k} \sum_{j\in N\setminus I_k} q_{ij}x_iy_j^k + \sum_{i\in I_k} c_ix_i \right) \tag{3.20}$$

$$\text{s.t.} \quad y_j^k = x_j \quad \forall k = 1,\ldots,p, j \in N \setminus I_k \tag{3.21}$$

$$x_iy_j^k = x_jy_i^l \quad \forall k,l = 1,\ldots,p; k \neq l; i \in I_k; j \in I_l \tag{3.22}$$

$$(3.4), (3.5), (3.7) \tag{3.23}$$

$$0 \leq y^k \leq 1 \quad \forall k = 1,\ldots,p. \tag{3.24}$$

where the binary restrictions on variables y have been relaxed. Relaxing constraints (3.21) with multipliers λ_j^k $\forall k = 1,\ldots,p, j \in N \setminus I_k$ and constraints (3.22) with multipliers μ_{ij} $\forall k,l = 1,\ldots,p; k \neq l; i \in I_k; j \in I_l$, we obtain the following Lagrangian function:

$$\Psi(x,y,\lambda,\mu) = \sum_k \Psi_k(x,y,\lambda,\mu), \tag{3.25}$$

where for each $k = 1,\ldots,p$, we have

$$\begin{aligned}\Psi_k(x,y,\lambda,\mu) = & \sum_{i\in I_k} \sum_{j\in I_k} q_{ij}x_ix_j + \sum_{i\in I_k} \sum_{j\in N\setminus I_k} (q_{ij} + \mu_{ij} - \mu_{ji})x_iy_j^k \\ & + \sum_{i\in I_k} (c_i + \sum_{l\neq k} \lambda_i^k)x_i - \sum_{j\in N\setminus I_k} y_j^k\end{aligned}$$

To linearize $\Psi_k(x,y,\lambda,\mu)$ for given λ and μ, we introduce new continuous variables $w_i^k = x_i \sum_{j\in I_k} q_{ij}x_j$ and $z_i^k = x_i \sum_{j\in N\setminus I_k} (q_{ij} + \mu_{ij} - \mu_{ji})y_j^k$ for each $i \in I_k$, and add some constraints to enforce these equalities. By doing so, the following Lagrangian relaxation problem implies:

$$\min \quad \sum_k \overline{\Psi}_k(x,y,\lambda,\mu) \tag{3.26}$$

$$\text{s.t.} \quad w_i^k \geq \sum_{j\in I_k} x_j - U_i^k(1 - x_i) - L_i^kx_i \quad \forall k = 1,\ldots,p, i \in I_k \tag{3.27}$$

$$z_i^k \geq \sum_{j\in N\setminus I_k} y_j^k - \bar{U}_i^k(1 - x_i) - \bar{L}_i^kx_i \quad \forall k = 1,\ldots,p, i \in I_k \tag{3.28}$$

$$(3.23), (3.24) \tag{3.29}$$

$$w_i^k, z_i^k \geq 0 \quad \forall k = 1,\ldots,p, i \in I_k. \tag{3.30}$$

where

$$\overline{\Psi}_k(x, y, \lambda, \mu) = \sum_{i \in I_k} \left(w_i^k + z_i^k + (L_i^k + \bar{L}_i^k + c_i + \sum_{l \neq k} \lambda_i^l \right) x_i - \sum_{i \in N \setminus I_k} y_i^k \tag{3.31}$$

and for each k and $i \in I_k$, the $L_i^k, \bar{L}_i^k, U_i^k$ and $\bar{U}_i^k$ are defined as:

$$L_i^k(U_i^k) = \min(\max)\{\sum_{j \in I_k} q_{ij} x_j : (3.4), (3.5), 0 \leq x \leq 1,\ 0 \leq y \leq 1\}$$
$$\bar{L}_i^k(\bar{U}_i^k) = \min(\max)\{\sum_{j \in N \setminus I_k} q_{ij} y_j^k : (3.4), (3.5), 0 \leq x \leq 1,\ 0 \leq y \leq 1\}.$$

To solve the problem we can partition it into p independent subproblems and solve the subproblems at each step of the subgradient algorithm.

3.2 QP-based reformulation and a simple decomposition

The general idea of the QP-based reformulation is to perturb the objective function to obtain an equivalent QP so that solving the resulting problem be more easier or provides a stronger bound for the original problem. Let us first define a reformulation for QP based on definition of reformulation proposed by Carraresi and Malucelli for the QAP [27]:

Definition 3.2.1. *Given an instances $P:\ \min\{x^T Q x + c^T x :\ x \in S \cap \mathbb{B}^n\}$ of the QP, the problem $P':\ \min\{x^T Q' x + c'^T x :\ x \in S \cap \mathbb{B}^n\}$ is called a reformulation of the P, if for each $x \in S \cap \mathbb{B}^n$ we have*

$$x^T Q x + c^T x = x^T Q' x + c'^T x. \tag{3.32}$$

It is easy to see that, if P'' is a reformulation of P' and P' is a reformulation of P, then P'' is also a reformulation of P.

The reformulation of the QP into an equivalent quadratic 0-1 programming with convex or non-convex objective function has already been considered for both the general and special cases of the QP in the literature. Hammer and Rubin [61] proposed the following reformulation which transform a QP into a convex one:

$$\min_{x \in S \cap \mathbb{B}^n} \left\{ x^T Q x + c^T x - \gamma(Q) \sum_{i=1}^{n} (x_i^2 - x_i) \right\} \tag{3.33}$$

where $\gamma(Q)$ is the smallest eigenvalue of Q. As we explained in Section 2.2.2, Billionnet et al. [19] proposed the QCR method to reformulate the QP into a convex quadratic programming. In [18], Billionnet and Elloumi provide a different reformulation strategy for convexifying the unconstraint QP which can be considered as a special case of the QCR method. As for the special cases of the QP, many QP-based reformulations have been proposed which among them we can refer to the reformulations approaches by Carraresi and Malucelli [27, 28] and Assad and Xu [14, 15]. In the following we generalized the reformulation strategy proposed for the QAP by Carraresi and Malucelli [27, 28] for the general QP.

3.2.1 A simple decomposition-based lower bound

Consider QP in its general form and rewrite it as $\min_{x\in S\cap\mathbb{B}^n}\sum_i\left(\sum_j q_{ij}x_j\right)x_i+\sum_i c_ix_i$. For each i, if $x_i = 1$, then replacing $\sum_j q_{ij}x_j$ with its minimum value over the set of possible feasible solutions provides a lower bound on the objective value of the original problem. More precisely we can solve a set of subproblems, P_i, with a linear objective function, one for each i, i.e.,

$$P_i: \quad w_i = \min\left\{\sum_j q_{ij}x_j \,:\, x\in S\cap\mathbb{B}^n,\ x_i = 1\right\}. \tag{3.34}$$

For several combinatorial optimization problems considered in this thesis, QAP, QMSTP and QSP, problem P_i can be solved in polynomial time. Once P_i has been computed for each i, one can solve the following problem to obtain a valid lower bound for the original problem:

$$P_0: \quad \min\left\{\sum_i (w_i + c_i)x_i \,:\, x\in S\cap\mathbb{B}^n\right\}. \tag{3.35}$$

This procedure was first introduced by Gilmore [54] and Lawler [78] for the quadratic assignment problem. The most advantageous property of this lower bounding procedure is that it can be computed efficiently if one can solve the minimization of a linear function over $S\cap\mathbb{B}^n$. However, the main drawback of the method is the fast deterioration of its quality with increasing problem size, as shown for some spacial cases including QAP [79] and QMSTP [94].

3.2.2 Improving the bound based on a reformulation

The lower bounding procedure described in previous section transfers part of the quadratic costs to the linear cost vector by solving each of the P_i subproblems. Nevertheless, the part of quadratic cost that is not included in the solution of P_i is simply ignored when computing P_0. In order to improve the bound we can shift out as much information as possible from the quadratic term. More precisely consider a reformulation of the QP by defining the following transformations:

$$q'_{ij} = q_{ij} - \sum_h a_{hj}\lambda^i_h - \sum_l d_{lj}\mu^i_l - \nu^i_j \quad \text{and} \quad c'_i = c_i + w_i. \tag{3.36}$$

where λ^i, μ^i and ν^i are the optimal dual solutions corresponding to constraints $Ax \leq b, Dx = g$ and $x \leq 1$ of the continuous relaxation of subproblem P_i.

Using these transformations, we can then define the following *Reformulated* QP in which the quadratic component has a lower overall impact:

$$\text{RQP3:} \quad \min \quad \sum_i \sum_{j \neq i} q'_{ij} x_i x_j + \sum_i c'_i x_i \tag{3.37}$$

$$\text{s.t.} \quad x \in S \cap \mathbb{B}^n. \tag{3.38}$$

In the below theorem we prove that problems QP and RQP3 are equivalent, i.e., RQP3 is a reformulation of the QP.

Theorem 3.2.1. *Problems QP and RQP3 are equivalent.*

Proof. For any feasible solution x we have

$$\begin{aligned}
&\sum_i \sum_j q'_{ij} x_i x_j + \sum_i c'_i x_i \\
&= \sum_i \sum_j (q_{ij} - \sum_h a_{hj}\lambda^i_h - \sum_l d_{lj}\mu^i_l - \nu^i_j) x_i x_j + \sum_i (c_i + w_i) x_i \\
&= \sum_i \sum_j q_{ij} x_i x_j + \sum_i c_i x_i \\
&- \sum_i \left(\sum_h \lambda^i_h \sum_j a_{hj} x_j + \sum_l \mu^i_l \sum_j d_{lj} x_j \right) x_i - \sum_i \sum_j \nu^i_j + \sum_i w_i x_i \\
&= \sum_i \sum_j q_{ij} x_i x_j + \sum_i c_i x_i.
\end{aligned}$$

Here, the last equality follows from $w_i = \sum_h \lambda^i_h b_h + \sum_l \mu^i_l g_l + \sum_j \nu^i_j$. □

Note that each components of the linear cost vector is updated to include the value of optimal solution of the corresponding P_i, thus by substituting matrix Q' in $RQP3$ with the null matrix, the resulting problem is exactly the same as problem P_0. In general we can iterate the reformulation process to find a sequence of equivalent QP instances ($QP_0, QP_1, \ldots, QP_i, QP_{i+1}$ with $QP_0 = QP$), each characterized by a stronger impact of linear cost than the previous.

Part II

Special Cases of the Quadratic 0-1 programming

Chapter 4

Quadratic Assignment Problem

The Quadratic Assignment Problem is a classical combinatorial optimization problem that has applications in many real-life problems in several areas. In this chapter after introducing the problem, we study the most important reformulations and lower bounding approaches including LP-based reformulations, QP-based reformulation and RLT representations. Although solving the RLT representations of the quadratic assignment problem provide a strong lower bound in high level RLT representations, these bounds require much computational effort, which can be problematic for branch-and-bound algorithms. In order to speed up the bound computation in the level-d RLT representation, we construct a new compact reformulation for each level of the RLT based on the structure of the problem.

4.1 Introduction

The quadratic assignment problem (QAP) was introduced by Koopmans and Beckmann in [76] in the context of facility location to deal with a one-to-one assignment of n facilities to n locations with the minimum cost. The assignment cost includes the quadratic cost of each possible assignment being the flow between each pair of facilities multiplied with the distance between their assigned locations and the total linear cost associated with allocating a facility to a certain location. More precisely, given n facilities, n locations, a flow f_{ij} from each facility i to each facility j, $i \neq j$, a distance d_{kl} from each location k to each location l, $k \neq l$, and a cost c_{ik} of assigning facility i to location k, the QAP is defined as

$$\text{QAP:} \quad \min \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{ij}d_{kl}x_{ik}x_{jl} + \sum_{i=1}^{n}\sum_{k=1}^{n} c_{ik}x_{ik} \tag{4.1}$$

$$\text{s.t.} \quad x \in X, \ x \text{ binary.} \tag{4.2}$$

where

$$X = \{x \geq 0 : \sum_{j=1}^{n} x_{ij} = 1 \quad \forall i = 1, \dots, n; \ \sum_{i=1}^{n} x_{ij} = 1 \quad \forall j = 1, \dots, n\}. \tag{4.3}$$

A more general QAP version was proposed by Lawler [78] and involves costs q_{ijkp} that do not necessarily correspond to products of flows and distances. The Lawler formulation is as follows:

$$\text{QAP:} \quad \min \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} q_{ijkl}x_{ij}x_{kl} + \sum_{i=1}^{n}\sum_{k=1}^{n} c_{ik}x_{ik} \tag{4.4}$$

$$\text{s.t.} \quad x \in X, \ x \text{ binary.} \tag{4.5}$$

The QAP has been used to model many applications including, among others, backboard wiring [114], coding of signals [17], typewriter keyboards and control panels design [101], image processing [116], scheduling [53], storage-and-retrieval [99]. It has been shown that the QAP is among the most difficult NP-hard combinatorial optimization problems and in general solving instances of size $n \geq 30$ in a reasonable time is impossible [80]. Many solution methods, exact or heuristics algorithm have been proposed for solving the QAP. Branch-and-Bound approaches and various reformulations and linearization of the QAP as MILPs are among the main exact solution methods for the QAP. One of the best attempts to date is [45] where the proposed solution method takes advantage of the symmetry present in known (and hard to solve) benchmark instances. Due to quadratic nature of the problem, many attempts have been made in the literature to linearize the objective function so that the resulting lower bound is strong enough to be used in a branch-and-bound algorithm. Among the best lower bounding approaches in the literature we can refer the reader to Frieze and Yadegar [48], Assad and Xu [15], Carraresi and Malucelli [27, 28], Adams and Johnson [4], level-1 RLT dual-ascent bound by Hahn and Grant [59], the convex quadratic programming bound by Anstreicher and Brixius [13], the level-2 RLT by Adams et al. [3], and level-3 RLT by Hahn et al. [60].

4.2 Lower bounding procedure

Lower bounds play an important role in success of the Branch-and-Bound type algorithms for the QAPs. The ideal lower bound should be sharp (i.e., "small" gap between the bound and the optimum solution) and enough fast to compute. In this section we present most important lower bounding schemes including the historical Gilmore and Lawler bound, bounds based on "reformulations", and finally bounds based on the MILP formulations.

4.2.1 Gilmore-Lawler lower bound

The Gilmore-Lawler bound (GLB) presented by Gilmore [54] and Lawler [78] is one of the best known lower bounds for QAP. However, it is not very tight even for small size problems and the gap from the optimal solution increases with the size of the problem. The GLB is given by the solution of the following linear assignment problem (LAP):

$$\begin{aligned} GLB = \min \quad & \sum_{i=1}^{n}\sum_{j=1}^{n}(c_{ij}+z_{ij})x_{ij} \\ \text{s.t.} \quad & x \in X,\ x \text{ binary}. \end{aligned}$$

where for each i, j the coefficient z_{ij} are found by solving the following problem:

$$\begin{aligned} SP_{ij}: \quad z_{ij} = \min \quad & \sum_{\substack{k=1\\k\neq i}}^{n}\sum_{\substack{l=1\\l\neq j}}^{n}(q_{ijkl})x_{kl} && (4.6)\\ \text{s.t.} \quad & x \in X,\ x \text{ binary} && (4.7)\\ & x_{ij} = 1. && (4.8) \end{aligned}$$

Since the GLB is depended on the idea of solving n^2+1 LAPs of size n, it is among the easiest approaches to compute.

4.2.2 Reformulation-based lower bounds

Carraressi and Malucelli [27] propose a way of reformulating the QAP so as to transfer the quadratic cost coefficients to linear cost coefficients. At the end of the transfer, they solve the linear part to obtain a lower bound. More precisely, they defined a reformulation for the QAP as follows:

$$\begin{aligned} P\colon \quad \min \quad & \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} q'_{ijkl}x_{ij}x_{kl} + \sum_{i=1}^{n}\sum_{k=1}^{n} c'_{ik}x_{ik} \\ \text{s.t.} \quad & x \in X, \ x \text{ binary.} \end{aligned}$$

where $c'_{ik} = c_{ik} + z_{ik}$ and $q'_{ijkl} = q_{ijkl} - \lambda_j^{ik} - \mu_l^{ik}$. The parameters λ^{ik} and μ^{ik} are the dual solutions corresponding to constraints defined in X when the facility i is assigned to the location k.

In order to strengthen the reformulation, the authors also proposed an iterative approach which leads to generate sharp bounds for the problem. This iterative procedure can be obtained by iteratively applying the reformulation presented above and thus defining a sequence of equivalent QAP instances $P_0, P_1, \dots, P_i, P_{i+1}$ with $P_0 = QAP$. Each reformulation is characterized by a stronger impact of linear cost than the previous one.

Assad and Xu in [15] proposed another kind of reformulation for the QAP that generates a monotonic sequence of lower bounds. Their method relies the following transformation:

$$\begin{aligned} c'_{ik} &= c_{ik} - (n-1)\theta_{ik} \forall (i,k) \\ q'_{ijkl} &= q_{ijkl} + \theta_{kl} \forall (i,j),(k,l), \ i \neq k, j \neq l. \end{aligned}$$

To compute the lower bound, they apply the Gilmore-Lawler procedure to the reformulated problem, i.e.,

$$AX(\theta) : \min \left\{ \sum_{i}\sum_{k} [z_{ik}(\theta) + c_{ik} - (n-1)\theta_{ik}]x_{ik} : x \in X \right\} \tag{4.9}$$

where

$$z_{ik}(\theta) = \min \left\{ \sum_{j}\sum_{l} q'_{ijkl}x_{jl} : x \in X, x_{ik} = 1 \right\} \tag{4.10}$$

Observe that the optimal value of the QAP is independent from vector θ. However, the approximated linear costs c' depend on θ and a better choice of θ yields a tighter bound. To determine the best value of parameter θ the authors proposed the following algorithm.

The Leveling Algorithm:

Step0: A tolerance limit ε is set for the stopping rule and iteration counter t is set to 1. Initially $\theta^0 = 0$.

Step1: Compute $z_{ik}(\theta^t)$ for all (i,k) using (4.10) and the compute the $AX(\theta^t)$ by using (4.9).

Step2: If $\max_{(i,k)}\{z_{ik}(\theta^t)\} - \min_{(i,k)}\{z_{ik}(\theta^t)\} \leq \varepsilon/(n-1)$ then stop, else set

$$\theta_{ik}^{t+1} = \theta_{ik}^t + \frac{1}{n-1} z_{ik}(\theta^t) \quad \forall (i,k), \quad t = t+1 \quad \text{and go to step1.}$$

Another reformulation based bound for the QAP has been proposed by Carraressi and Malucelli [28] which is a generalization of the two aforementioned schemes. For any value of θ the following transformation defines a reformulation for the QAP:

$$\begin{aligned} q'_{ijkl} &= q_{ijkl} + \gamma_{ijkl} - \alpha_{ikl} - \beta_{jkl} + \theta_{kl} \quad \forall (i,j),(k,l),\ i \neq k, j \neq l \\ c'_{ik} &= c_{ik} + z_{ik} - (n-1)\theta_{ik} \quad \forall (i,k) \end{aligned}$$

where $\gamma_{ijkl} = q_{klij} - q_{ijkl}$, and $\alpha_{ikl}, \beta_{jkl}$ are the dual solutions of subproblem SP_{kl}. Two update formulas for parameter θ are considered:

$$\begin{aligned} \theta_{ik} &= \theta_{ik} + \frac{1}{n-1} c_{ik} \quad \forall (i,k) \\ \theta_{ik} &= \frac{1}{n-1}(c_{ik} - \lambda_i - \mu_k) \quad \forall (i,k), \end{aligned}$$

where λ_i and μ_k are the dual solutions of the assignment problem with cost c'. Observe that, the first choice of updating θ does not guarantee monotonic sequence of lower bounds, while the second choice of θ yields a monotonic sequence of lower bounds as proved in [28].

4.2.3 MILP-based lower bounds

In this section we present the most important MILP formulations for the QAP. We will start with the smallest MILP formulation proposed by Kaufman and Broeckx [74] and provide its improved versions by Xia and Yuan [118] and Zhang et al. [119]. Then we describe the MILP formulation proposed by Frieze and Yadegar [48], and finally present the MIP formulations and the lower bounding approaches obtained by the reformulation-Linearization technique.

Given a general QAP formulation, Kaufman and Broeckx [74] proposed the following QAP linearization by introducing the continuous variable $w_{ij} = x_{ij} \sum_k \sum_l q_{ijkl} x_{kl}$ as contribution of each assignment variable x_{ij} to the overall cost.

$$\begin{aligned}
\text{KB:}\quad \min\quad & \sum_i \sum_j w_{ij} \\
\text{s.t.}\quad & w_{ij} \geq \sum_k \sum_l q_{ijkl} x_{kl} - U_{ij}(1 - x_{ij}) \quad \forall (i,j) \\
& w_{ij} \geq 0 \quad \forall (i,j) \\
& x \in X,\ x \text{ binary.}
\end{aligned}$$

Here, $U_{ij} = \sum_k \sum_l q_{ijkl}$. This formulation which employs n^2 continuous variables, n^2 binary variables and $n^2 + 2n$ constraints, is somehow less favored in the literature, due to its weake lower bound.

Theorem 4.2.1 (Zhang et al. [119]). *The optimal value of the LP relaxation of KB is equal to zero.*

Note that the KB formulation can be obtained by applying the general Glover linearization strategy. In order to strengthen the LP relaxation bound of KB, Xia and Yuan [118] and Zhang et al. [119] provide some alternative linearized models based on the revised version of the Glover technique explained in Chapter 2. The below program is the linearized formulation proposed by Zhang et al. [119]:

$$\begin{aligned}
\text{ZH:}\quad \min\quad & \sum_i \sum_j (w_{ij} + (c_{ij} + z_{ij})) \\
\text{s.t.}\quad & w_{ij} \geq \sum_k \sum_l q_{ijkl} x_{kl} - \bar{U}_{ij}(1 - x_{ij}) - (z_{ij} + c_{ij}) x_{ij} \quad \forall (i,j) \\
& w_{ij} \geq 0 \quad \forall (i,j) \\
& x \in X,\ x \text{ binary.}
\end{aligned}$$

where $\bar{U}_{ij} = \max\{\sum_k \sum_l q_{ijkl} x_{kl} \ :\ x \in X,\ x \text{ binary}\}$ and z_{ij} is the Gilmore-Lawler constant defined in (4.6).

In a different linearized model for the QAP, Frieze and Yadegar in [48] introduced n^4 continuous variable y_{ijkl}, implicity equal to the multiplication $x_{ij}x_{kl}$, and $O(n^3)$ new constraints to drive the following QAP linearization:

$$\begin{aligned}
\text{FY:} \quad \min \quad & \sum_i \sum_j \sum_k \sum_l q_{ijkl} y_{ijkl} + \sum_i \sum_j c_{ij} x_{ij} \\
\text{s.t.} \quad & \sum_i y_{ijkl} = x_{kl} \quad \forall (j,k,l) && (4.11) \\
& \sum_j y_{ijkl} = x_{kl} \quad \forall (i,k,l) && (4.12) \\
& \sum_k y_{ijkl} = x_{ij} \quad \forall (i,j,l) && (4.13) \\
& \sum_l y_{ijkl} = x_{ij} \quad \forall (i,j,k) && (4.14) \\
& y_{ijij} = x_{ij} \quad \forall (i,j) && (4.15) \\
& 0 \leq y_{ijij} \leq 1 \quad \forall (i,j,k,l) && (4.16) \\
& x \in X,\ x \text{ binary.}
\end{aligned}$$

In fact, the quadratic terms in the objective function have been eliminated at the cost of the additional decision variables y_{ijkl}. In order to compute a lower bound, the authors proposed a Lagrangian relaxation of FY formulation with multipliers α_{ikl} and β_{jkl} for constraints (4.11) and (4.12) respectively. The Lagrangian function $L(\alpha, \beta)$ is thus defined as

$$\begin{aligned}
L(\alpha,\beta) = \min \quad & \sum_i \sum_j \sum_k \sum_l \bar{q}_{ijkl} y_{ijkl} + \sum_i \sum_j \bar{c}_{ij} x_{ij} \\
\text{s.t.} \quad & (4.13), (4.14), (4.15), (4.16) \\
& x \in X,\ x \text{ binary.}
\end{aligned}$$

where

$$\bar{q}_{ijkl} = q_{ijkl} - \alpha_{ikl} - \beta_{jkl} \qquad \forall (i,j,k,l) \tag{4.17}$$

$$\bar{c}_{ij} = c_{ij} + \sum_l \alpha_{ikl} + \sum_l \beta_{jkl} \qquad \forall (i,j). \tag{4.18}$$

Note that the the objective function of the $L(\alpha, \beta)$ can be considered as a reformulation of the general QAP. More precisely by considering the transformation (4.17) and (4.18) one can obtain the GLB as follows:

$$GLB(\alpha,\beta) = \min\{\sum_i \sum_k [z_{ik}(\alpha,\beta) + c_{ik} - (n-1)\theta_{ik}] x_{ik} : x \in X\},$$

where

$$z_{ik}(\alpha,\beta) = \min\{\sum_j \sum_l q'_{ijkl} x_{jl} : x \in X, x_{ik} = 1\}$$

Frieze and Yadegar proved that the value of the Lagrangian function $L(\alpha, \beta)$ is equal to $GLB(\alpha, \beta)$; therefore, the solution of the Lagrangian dual $\max_{\alpha,\beta} L(\alpha, \beta)$ gives the optimal reformulation of the QAP.

4.3 Reformulation-linearization technique

In this section we present the Reformulation-linearization technique (RLT) applied to the QAP. Based on the RLT technique for general zero-one polynomial programs by Adams and Sherali [5, 6], the first RLT representation for the QAP was introduced by Adams and Johnson [4]. Consider problem QAP as presented in (4.4) and (4.5). The level-1 RLT representation is generated via the following two steps:
Reformulation: Multiply each of the $2n$ equations and each of the n^2 nonnegative constraints defining X by each of the n^2 binary variables x_{kl}, and append these new constraints to the formulation. When the variable x_{ij} in a given constraint is multiplied by x_{kl}, express the resulting product as $x_{ij}x_{kl}$ in that order. Substitute x^2_{kl} with x_{kl} throughout the constraints and set $x_{ij}x_{kl} = 0$ if $i = k$ and $j \neq l$ or $i \neq k$ and $j = l$.
Linearization: For all (i, j, k, l) with $i \neq k$ and $j \neq l$, substitute each product $x_{ij}x_{kl}$ with y_{ijkl}. Enforce the equality $y_{ijkl} = y_{klij}$ for all (i, j, k, l) with $i < k$ and $j \neq l$.

The level-1 RLT results as follows:

$$\text{RLT1: } \min \sum_i \sum_j \sum_{k\neq i} \sum_{l\neq j} q_{ijkl} y_{ijkl} + \sum_i \sum_j c_{ij} x_{ij}$$

$$\text{s.t.} \sum_{i\neq k} y_{ijkl} = x_{kl} \quad \forall (j,k,l),\ j \neq l \tag{4.19}$$

$$\sum_{j\neq l} y_{ijkl} = x_{kl} \quad \forall (i,k,l),\ i \neq k \tag{4.20}$$

$$y_{ijkl} = y_{klij} \quad \forall (i,j,k,l),\ i < k,\ j \neq l \tag{4.21}$$

$$y_{ijkl} \geq 0 \quad \forall (i,j,k,l),\ i \neq k,\ j \neq l \tag{4.22}$$

$$x \in X,\ x \text{ binary.}$$

Equation (4.21) is very important and says that if an element y_{ijkl}, $i \neq k, j \neq l$ is part of a solution (i.e., equal to 1) then it has a "complementary element" y_{klij} that is also in that solution. In general, the RLT1 representation has a large number of variables and constraints, which makes it computationally challenging, even for small QAP instances. Resende et al. [106] performed a computational test of the lower bounds generated by the LP relaxation of the RLT1. They reduced the numbers of variables and constraints in RLT1 by removing all variables y_{ijkl} with $i > k$ and $j \neq l$ and by making the substitutions suggested by (4.21) throughout the objective function and constraints. Then they solved the LP relaxation by using an experimental interior point method code, called ADP. To solve the RLT1, Adams and Johnson provide a Lagrangian relaxation which has a block-diagonal structure. More precisely they dualize constraints (4.21) on the complementary pairs and decompose the resulting problem into n^2 separate linear assignment problems of size $n-1$ and a linear assignment problem of size n. Hahn and Grant [59] gave a different interpretation of the same decomposition for lower bound calculation by using a dual-ascent strategy. Their dual-ascent procedure gives a bound very close to optimum of the LP relaxation of the RLT1, improving upon the computational results of Adams and Johnson [4], and requiring only a small fraction of the time of Resende et al. [106].

Based on the success of level-1 RLT representation to gain a tight bound for the QAP and also due to the block-diagonal structure of the problem which lends itself to efficient solution methods, the level-2 and level-3 RLT can be defined in the same way as the level-1 RLT via the reformulation and linearization steps. In the level-2 RLT representation, in addition to the operations done in the level-1, each binary variable in X is multiplied also by products $x_{kl}x_{pq}$ having $k \neq p$ and $l \neq q$. For more details concerning the reformulation and linearization step for the level-2 RLT we refer the reader to [3]. The level-2 RLT is called RLT2 and is written as follows:

$$\begin{aligned}
\min\quad & \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} q_{ijkl}y_{ijkl} + \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij} \\
\text{s.t.}\quad & \sum_{i\neq k,p} z_{ijklpq} = y_{klpq}\ \forall(j,k,l,p,q),\ j\neq l\neq q; k\neq p && (4.23)\\
& \sum_{j\neq l} z_{ijklpq} = y_{klpq} \quad \forall(i,k,l,p,q),\ i\neq k\neq p; j\neq q && (4.24)\\
& z_{ijklpq} = z_{ijpqkl} = \ldots = z_{pqklij}\ \forall(i,j,k,l,p,q), i<k<p; j\neq l\neq q && (4.25)\\
& z_{ijklpq} \geq 0\ \forall(i,j,k,l,p,q), i\neq k\neq p; j\neq l\neq q && (4.26)\\
& (4.19), (4.20), (4.21), (4.22) && (4.27)\\
& x\in X,\ x \text{ binary.}
\end{aligned}$$

The linear relaxation of the RLT2 is increasingly large and highly degenerate. Ramakrishnan et al. [103] enforced the constraints (4.25) to combine complementary variables and reduce the number of constraints and variables, then used commercial linear programming package CPLEX to solve the linear relaxation of the RLT2. However, because of the problem size and limitations of CPLEX, they were only able to solve instances up to size 12. Following the idea of Hahn and Grant [59] to solve the RLT1 in an efficient way, Adams et al. [3] have presented a dual-ascent strategy that exploits the block-diagonal structure of constraints in the RLT2 form.

In order to get even tighter bounds for the QAP, Zhu presented the level-3 RLT in his PhD dissertation [121]. The RLT3 formulation is significantly larger than the previous levels of RLT, but its continuous linear relaxation provides the tightest lower bound of all three RLT models. Hahn et al. [60] implemented a dual-ascent procedure similar to that employed in Adams et al. [3] for RLT2.

4.4 A Revised RLT

As far as the tightness of the bounds is concerned, the RLT representations of the QAP are among the most successful lower bounding approaches. However, in the high level RLT representation of the QAP these bounds require much computational effort, which can be problematic within a branch-and-bound algorithm. In order to speed up the bound computation in the level-d RLT representation of the QAP, we construct a smaller reformulation for each level of the RLT based on the structure of the problem. Let us start with the level-1 RLT formulation. The revised RLT1 (RRLT1) formulation is defined as follows:

$$\begin{aligned} \text{RRLT1:} \quad \min \quad & \sum_i \sum_j \sum_{k\neq i} \sum_{l\neq j} q_{ijkl} y_{ijkl} + \sum_i \sum_j c_{ij} x_{ij} \\ \text{s.t.} \quad & (4.19), (4.20), (4.21), (4.22) \qquad\qquad (4.28) \\ & x \in X', \ x \text{ binary.} \end{aligned}$$

where

$$X' = \{x \geq 0 : \sum_{j=1}^{n} x_{ij} = 1 \quad \forall i = 1, \dots, n\}. \qquad (4.29)$$

Note that the only difference with RLT1 is that constraints $\sum_{i=1}^{n} x_{ij} = 1, \forall j = 1, \dots, n$ are missing.

Theorem 4.4.1. *The problems RLT1 and RRLT1 are equivalent.*

Proof. To prove the theorm we use the idea of [2]. Consider any feasible solution $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ to Problem RRLT1. We first show that the following equations must hold.

$$\hat{y}_{pqst} = \hat{x}_{pq}\hat{x}_{st} \quad \forall\ (p, q, s, t);\ p \neq s;\ q \neq t. \qquad (4.30)$$

If $\hat{x}_{pq} = 0$, constraint $\sum_{i\neq k} \hat{y}_{ijpq} = \hat{x}_{pq}$ of (4.19), together with the nonnegativity restrictions $\hat{y}_{ijpq} \geq 0$ enforce that $\hat{y}_{ijpq} = 0$ for all $(i, j), i \neq p, j \neq q$. Consider the case $\hat{x}_{pq} = \hat{x}_{st} = 1$, and by contradiction assume that $\hat{y}_{pqst} = \hat{y}_{stpq} < 1$. The constraint $\sum_{j \neq q} \hat{y}_{sjpq} = \hat{x}_{pq}$ of (4.20), together with $\hat{y}_{stpq} < 1$ implies that there exists an index $l \neq t, q$ with $\hat{y}_{slpq} > 0$. By considering constraint (4.21), we have $\hat{y}_{pqsl} = \hat{y}_{slpq} > 0$, so that (4.20) implies $x_{sl} = 1$. The equalities $x_{sl} = x_{st} = 1$ for $l \neq t$, contradict the constraint $\sum_{j=1}^{n} x_{sj} = 1$ of X'. Consequently, $\hat{y}_{pqst} = \hat{x}_{pq}\hat{x}_{st}$ for binary $\hat{x}_{pq}$ and $\hat{x}_{qr}$. Now we show that $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ is a feasible solution for the RLT1. Since constraints (4.19), (4.20), (4.21), and (4.22) together with the binary restriction are precisely the same in both models, the proof is to show that $\sum_{i=1}^{n} \hat{x}_{ij} = 1, \forall j = 1, \dots, n$. Consider an index $s \in \{1, \dots, n\}$ such that $\sum_{i=1}^{n} \hat{x}_{is} = \epsilon \neq 1$. Multiplying this equation by binary variable $\hat{x}_{kl}$ with $l \neq s$ to obtain

$$\sum_{i \neq k} \hat{x}_{is}\hat{x}_{kl} = \epsilon x_{kl} \quad \forall (k, l), l \neq s \tag{4.31}$$

By (4.30) and (4.31), we have

$$\sum_{i \neq k} \hat{y}_{iskl} = \sum_{i \neq k} \hat{x}_{is}\hat{x}_{kl} = \epsilon x_{kl} \quad \forall (k, l), l \neq s \tag{4.32}$$

Since $\epsilon \neq 1$, the equations (4.32) contradict (4.20) and the proof is completed. □

The following theorem formally shows that the continuous relaxations of the RRLT1 (CRRLT1) is as tight as the continuous relaxation of the RLT1 (CRLT1), in that a linear combination of the constraints of CRRLT1 can implies constraint $\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j = 1, \dots, n$.

Theorem 4.4.2. *Problems CRLT1 and CRRLT1 are equivalent.*

Proof. Since constraints (4.19), (4.20), (4.21), and (4.22) appear in both CRLT1 and CRRLT1, the proof reduces to show that constraints $\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j = 1, \dots, n$ of X defined in (4.3) can be computed as linear combinations of the constraints of CRRLT1. Consider any $(j, k, l), j \neq l$, and observe that constraints (4.20) enforce that $\sum_{i \neq k} y_{ijkl} = x_{kl}$. Summing this equations over all $(k, l), l \neq j$, we obtain

$$\sum_{l \neq j} \sum_{k} \sum_{i \neq k} y_{ijkl} = \sum_{l \neq j} \sum_{k} x_{kl} \quad \forall j = 1, \dots, n. \tag{4.33}$$

By definition of X' in (4.29), the right hand side of (4.33) can be written as follows:

$$\sum_{l \neq j} \sum_{k} x_{kl} = \sum_{k} \sum_{l} x_{kl} - \sum_{k} x_{kj} = n - \sum_{k} x_{kj} \quad \forall j = 1, \dots, n. \tag{4.34}$$

Now consider the left hand side of (4.33). By using (4.19) and (4.21), it can written as:

$$\sum_{l\neq j}\sum_{k}\sum_{i\neq k} y_{ijkl} = \sum_{i\neq k}\sum_{k}\sum_{l\neq j} y_{klij} = \sum_{i\neq k}\sum_{k} x_{ij} = \sum_{k}\sum_{i\neq k} x_{ij}$$
$$= \sum_{k}\sum_{i} x_{ij} - \sum_{k} x_{kj} = n\sum_{i} x_{ij} - \sum_{k} x_{kj} \quad \forall j = 1,\ldots,n. \tag{4.35}$$

By (4.34) and (4.35), we have $\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j = 1,\ldots,n$. The proof is completed. □

We now turn to the level-2 RLT representation of the QAP. We give a smaller reformulation of the RLT2 called revised RLT2 (RRLT2) by substituting X defined in (4.3) with X' defined in (4.29) and removing constraints $\sum_{i\neq k} y_{ijkl} = x_{kl} \quad \forall (j,k,l),\ j\neq l$ from the RLT2 formulation. The RRLT2 has the following form:

$$\text{RRLT2:} \quad \min \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} q_{ijkl}y_{ijkl} + \sum_{i=1}^{n}\sum_{j=1}^{n} c_{ij}x_{ij}$$
$$\text{s.t.}(4.20)-(4.22),(4.23)-(4.26) \tag{4.36}$$
$$x \in X',\ x \text{ binary}.$$

Theorem 4.4.3. *Problems CRLT2 and CRRLT2 are equivalent.*

Proof. Following the same idea as the proof of Theorem 6.5.1 we show that constraints $\sum_{i=1}^{n} x_{ij} = 1 \quad \forall j = 1,\ldots,n$ of X defined in (4.3) and constraints (4.23) can be computed as linear combinations of the constraints of CRRLT1. Since the proof of the first part is the same as the one in Theorem 6.5.1, here we provide the proof for the second part. Consider any (j,k,l,p,q) with $j\neq l\neq q$, $k\neq p$, and observe that constraints (4.23) enforce that $\sum_{i\neq k} z_{ijklpq} = y_{klpq}$. Summing this equations over all (p,q) having $p\neq k$ and $q\neq j,l$, we obtain

$$\sum_{q\neq j,l}\sum_{p\neq k}\sum_{i\neq k,p} z_{ijklpq} = \sum_{q\neq j,l}\sum_{p\neq k} y_{klpq} \quad \forall (j,k,l) j\neq l. \tag{4.37}$$

By (4.20) and (4.21), the right hand side of (4.37) can be written as follows:

$$\sum_{q\neq j,l}\sum_{p\neq k} y_{klpq} = \sum_{p\neq k}\sum_{q\neq l} y_{klpq} - \sum_{p\neq k} y_{klpj} = (n-1)x_{kl} - \sum_{p\neq k} y_{pjkl} \quad \forall (j,k,l) j\neq l. \tag{4.38}$$

Now consider the left hand side of (4.33). By using (4.19) and (4.21), it can written as:

$$\sum_{q\neq j,l}\sum_{p\neq k}\sum_{i\neq k,p} z_{ijklpq} = \sum_{i\neq k,p}\sum_{p\neq k}\sum_{q\neq j,l} z_{pqijkl} = \sum_{p\neq k}\sum_{i\neq k,p} y_{ijkl}$$
$$= \sum_{p\neq k}\sum_{i\neq k} y_{ijkl} - \sum_{p\neq k} y_{pjkl} = (n-1)\sum_{i\neq k} y_{ijkl} - \sum_{p\neq k} y_{pjkl} \quad \forall (j,k,l) j \neq l. \tag{4.39}$$

By (4.38) and (4.39), we have $\sum_{i\neq k} y_{ijkl} = x_{kl}\ \forall (j,k,l), j \neq l$. The proof is completed. □

The idea of removing some set of constraints in level-1 and level-2 RLT representation of the QAP can be generalized for level-d RLT formulation with $1 \leq d \leq n$. In each level-d RLT representation we can eliminate one set of constraints generated in level-$(d-1)$ RLT, where level-0 RLT represent the original QAP. More precisely for each d, $2 \leq d \leq n-1$, the total number of constraints in the level-d RLT can be reduced by

$$f(d) = f(d-1) + (n-d+1)\prod_{i=0}^{d-2}(n-i)^2$$

where $f(1) = n$.

4.5 Computational Experiments

In this section we present the computational results of comparing the level-d RLT with the level-d RRLT for $d = 1, 2$, in terms of bound tightness and computational time. To obtain a lower bound for the QAP we solve the continuous relaxation of RRLT1 and RRLT2 by applying the Lagrangian relaxation. We implemented the algorithms in C++ language and run on an Intel Xeon CPU E5335 (2 quad core CPUs 2GH). Since solving both RRLT1 and RRLT2 are quite similar, we restrict our attention to explain the solving process to the RRLT2. We first place constraints (4.21) and (4.25) into the objective function. The Lagrangian function is then defined as:

$$K + \min\{\sum_i\sum_j\sum_k\sum_{p\neq i}\sum_{q\neq j}\sum_{l\neq k} \bar{D}_{ijkpql} z_{ijkpql} + \sum_i\sum_j\sum_{k\neq i}\sum_{l\neq j} \bar{B}_{ijkl} y_{ijkl}$$
$$+ \sum_i\sum_j \bar{c}_{ij} x_{ij} :\ (4.20), (4.22), (4.23), (4.24), (4.26),\ x \in X'\}. \tag{4.40}$$

where $\bar{B}_{ijkl}$ and $\bar{c}_{ij}$ are the adjusted values for q_{ijkl} and c_{ij} respectively, after placing constraints (4.21) into the objective function, $\bar{D}_{ijkpql}$ are the coefficients corresponding to the variables z_{ijkpql} after placing constraints (4.25) into the objective function, and $K = 0$.

Theorem 4.5.1. *An optimal solution* (x^*, y^*, z^*) *for problem* (4.40) *can be obtained via solving the semi-assignment problem*

$$K + \min\left\{\sum_p \sum_q (\bar{c}_{pq} + \rho_{pq})x_{pq} \,:\, x \in X'\right\}, \tag{4.41}$$

where for each (p, q)*,* ρ_{pq} *is obtained by solving the following semi-assignment problem* $Sub_1(p, q)$*:*

$$\rho_{pq} = \min \quad \sum_{k\neq p}\sum_{l\neq q}(\bar{B}_{klpq} + \gamma_{klpq})y_{klpq} \tag{4.42}$$

$$\text{s.t.} \quad \sum_{l\neq q} y_{klpq} = 1 \quad \forall k = 1, \ldots, n,\ k \neq p \tag{4.43}$$

$$y_{klpq} \geq 0 \quad \forall (k, l) \neq k \neq p, l \neq q. \tag{4.44}$$

and where for each (k, l, p, q) *with* $p \neq k$ *and* $q \neq l$

$$\gamma_{klpq} = \min \quad \sum_{i\neq p,k}\sum_{j\neq q,l} \bar{D}_{ijklpq}z_{ijklpq} \tag{4.45}$$

$$\text{s.t.} \quad \sum_{i\neq p,k} z_{ijklpq} = 1 \quad \forall j = 1, \ldots, n,\ j \neq q, l \tag{4.46}$$

$$\sum_{j\neq q,l} z_{ijklpq} = 1 \quad \forall i = 1, \ldots, n,\ i \neq p, k \tag{4.47}$$

$$z_{ijklpq} \geq 0 \quad \forall (i, j) \neq i \neq p, k,\ j \neq q, l. \tag{4.48}$$

We applied the dual ascent algorithm proposed in Adams et al. [3] to this new problem. The procedure consists of updating the constant term K and the cost matrices $\bar{D}$, $\bar{B}$ and $\bar{c}$, in such a way that the cost of any (integer) feasible solution with respect to the modified objective function remains unchanged, while maintaining nonnegative coefficients. As a consequence of this property, the value of K at any moment of the execution is a valid lower bound on the optimal solution cost for the QAP.

For computational testing of comparing the RLT and RRLT, we used 14 instances from the QAPLIB and 20 instances from the test set of Drugan [36]. This new test set currently was introduced by Drugan in [36] and are called composite QAPs (cQAPs). Tables 4.1 and 4.2 report the lower bounds and required CPU times of the dual ascent strategy applied to level-1, 2 RLT and RRLT for the instances of QAPLIB and cQAPs, respectively. The instance names and the corresponding dimensions are found in the first column. In the second column, there are the optimal values for each instance. The third and forth columns give the RLT1 lower bound and its CPU time, followed by the

Table 4.1 Comparison the level-1,2 RLT and RRLT lower bounds and CPU times for the instances of QAPLIB.

		RLT1		RRLT1		RLT2		RRLT2	
Instance	Opt.	Lb	Time	Lb	Time	Lb	Time	Lb	Time
nug12	578	511.45	6	511.59	6	578	774	578	364
nug15	1150	1001.4	15	1000.1	14	1140.05	11067	1141.58	11015
nug18	1930	1622.38	30	1620.04	31	1859.21	40480	1862.51	39379
nug20	2570	2239.20	47	2137.21	47	2449.41	64496	2454.36	63858
had16	3720	3525.22	19	3525.47	19	3720	407	3720	381
had18	5358	5036.65	30	5035.02	31	5358	19044	5358	14448
had20	6922	6504.74	47	6507.57	47	6922	56791	6922	48714
rou12	235528	219365	8	219329	6	235528	33	235528	33
rou15	354210	318496	14	318413	14	351106	11300	351537	12167
rou20	725520	632453	47	632346	47	687320	80052	688391	77251
tai15a	388214	346896	14	347086	14	377805	11244	378347	10037
tai15b	51765268	51459245	14	51443801	14	51765268	240	51765268	208
tai20a	703482	608846	46	608199	47	662750	88497	663855	78890
tai20b	122455319	88400570	54	87855727	57	122455319	3039	122455319	2791

Table 4.2 Comparison the level-1,2 RLT and RRLT lower bounds and CPU times for the cQAP data set.

		RLT1		RRLT1		RLT2		RRLT2	
Instance	Opt.	Lb	Time	Lb	Time	Lb	Time	Lb	Time
cqap20-0	238754	215117	48	215045	48	238754	541	238754	393
cqap20-1	233690	204845	46	204837	46	233690	707	233690	513
cqap20-2	230750	204996	47	204970	48	230750	668	230750	443
cqap20-3	235432	211780	47	211781	48	235432	452	235432	450
cqap20-4	242392	212641	47	212928	47	242392	584	242392	461
cqap20-5	236894	205933	47	205995	46	236894	748	236894	459
cqap20-6	241720	210392	48	210639	48	241720	651	241720	525
cqap20-7	242388	217115	48	217151	48	242388	600	242388	545
cqap20-8	236546	210846	47	210863	48	236546	726	236546	413
cqap20-9	239180	209644	47	209660	47	239180	884	239180	607
Average	237775	210331	47	210287	47	237775	656	237775	481
cqap24-0	312308	264043	98	263875	99	312308	1749	312308	1373
cqap24-1	305074	269989	99	270009	100	305074	1848	305074	1474
cqap24-2	310154	269105	97	268987	97	310154	1556	310154	1576
cqap24-3	307622	266878	98	266758	99	307622	1745	307622	1765
cqap24-4	313614	274491	98	274371	97	313614	1558	313614	1179
cqap24-5	308634	268792	98	268766	98	308634	1941	308634	1572
cqap24-6	301196	256687	98	256581	98	301196	2157	301196	1640
cqap24-7	301742	262322	98	262242	98	301742	2128	301742	1428
cqap24-8	303516	265041	99	265383	99	303516	2116	303516	1715
cqap24-9	309774	277585	99	277553	101	309774	1426	309774	1425
Average	307363	267493	98	267452	99	307363	1822	307363	1515

lower bound values and CPU times of RRLT1 in columns fifth and sixth, the RLT2 in columns seventh and eighth, and RRLT2 in columns ninth and tenth.

For the 14 tested instances from the QAPLIB, the dual ascent strategy applied to RLT1 and RRLT1 almost provide the same bounds with the same computational times, while the dual ascent strategy applied to RLT2 and RRLT2 provides the different results. More precisely, as we can observe from Table 4.1, the bounds from the dual ascent strategy applied to RLT2 and RRLT2 for problems nug12, had16, had18, had20, rou12, tai15b, and tai20b are exact, as they equal to the optimal objective values of the QAP. However the CPU time required by RLT2 to obtain these bounds is longer than the CPU time required by RRLT2. The bounds of the RRLT2 for the remaining problems are slightly tighter than the RLT2 bounds, but demand less computational effort except problem rou15 for which RLT2 requires less CPU time. It should be noted, however, that using RRLT2 the bound 351106 for rou15 achieved in less than 1500 seconds.

For the cQAP instances, as we can observe from Table 4.2, the RLT1 ,on average, is slightly better than the RRLT1 in both the bound tightness and required CPU time, but for all 20 instances, the lower bound obtained from the RLT2 and RRLT2 are equal to the objective value of the CQAPs. However, in terms of the required CPU time the RRLT2 outperforms RLT2 for all instances.

4.6 Conclusions

In this chapter we study the most important reformulations and lower bounding approaches including LP-based reformulations, QP-based reformulation and RLT representations. we proposed a revised form of the Reformulation Linearization Technique for the quadratic Assignment Problem without destroying the problem's structure. Our experimental results show that, by increasing the level of the RLT, solving the revised RLT representation provides a lower bound as strong as the bound obtained by the RLT, but with less computational effort.

Chapter 5

QAP with Special Structure

In this chapter we study two special cases of the QAP including the Adjacent QAP and QAP on reducible graphs. Adjacent QAP is defined on the Lawler form of the QAP whose objective coefficient matrix has a special structure, while QAP on reducible graph is defined on the QAP in Koopmans and Beckmann form where the flow and distance matrices are both reducible. Motivated by strong lower bounds obtained by applying Reformulation Linearization Technique (RLT) for the classical QAP, we propose two special RLT representations for the AQAP. The first is based on a "flow" formulation whose linear relaxation can be solved very efficiently for large instances while the second one has significantly more variables and constraints, but possesses some desirable properties relative to the constraint set. For the QAP on reducible graph we give a Lagrangian decomposition based on splitting the variables and then dualizing the copy constraint so that the resulting problem can be decompose into two quadratic semi assignment problems.

5.1 Introduction

Since solving a large scale QAP is extremely challenging in practice, some research has focused on the special cases of QAP. Christofides and Benavent [32] studied the QAP on a tree and proved that the problem is NP-hard even for this special case. They presented a branch-and-bound algorithm which uses the Lagrangian relaxation of an integer programming formulation of the tree QAP. Rendl in [104] generalized the approach presented by Christofides and Benavent to minimal series-parallel digraphs and showed that the subclass of series-parallel digraphs not containing bipartite subgraphs is solvable in polynomial time. Chen [31] proposed three special cases of the general form of the QAP that can be represented as parametric Linear assignment problems. The computational results have been reported for test problems up to size 50. Burkard et al. [23] provided three polynomial time solvable classes of the Koopmans-Beckmann form where one input matrix is monotone Anti-Monge while the other is either symmetric Toeplitz generated by a benevolent (or a k-benevolent) function, or symmetric with bandwidth one. Deineko and Woeginger [34] provided another polynomially solvable class for the Koopmans-Beckmann form with one matrix being Kalmanson and the other being symmetric decreasing circulant. They proved that identity permutation was the optimal solution for this case. Erdoğan and Tansel in [38] introduced two classes of QAPs namely, additively decomposable general costs and a subset of multiplicatively decomposable general costs and showed that both are solvable in polynomial time. More recently Çela et al. [24] studied two special family of the quadratic assignment problem; the first is QAP where the flow matrix is a monotone matrix and the distance matrix is a multi-cut matrix in normal form, while the second family are the so-called Product-Block QAPs. They showed that under some specific conditions these two families of QAPs can be solved in polynomial time. In the following sections we study and analyze two special cases of the QAP including the Adjacent QAP and QAP on reducible graphs.

5.2 Adjacent QAP

Consider a particular case of the Lawler form of the QAP called Adjacent QAP whose cost coefficient matrix has a special structure. Assume that for all (i, j, k, l) with $i \neq j$ and $k \neq l$, the quadratic terms q_{ijkl} get a nonzero coefficient if $i \neq l$ and $j = k$, or if $i = l$ and $j \neq k$, and get infinity if $i = l$ and $j = k$. By defining a new cost coefficient $D \in \mathcal{R}^{n\times n\times n}$ corresponding to the non zero elements of q, the AQAP can be written as:

$$\text{AQAP:}\quad \min \sum_{i}\sum_{j\neq i}\sum_{k\neq j,i} D_{ijk}x_{ij}x_{jk}$$

$$\sum_{j\neq i} x_{ij} = 1 \qquad \forall\, i = 1,\dots,n \tag{5.1}$$

$$\sum_{i\neq j} x_{ij} = 1 \qquad \forall\, j = 1,\dots,n \tag{5.2}$$

$$\boldsymbol{x} \text{ binary.} \tag{5.3}$$

This problem was introduced by Fischer et al. [44] as a subroutine for the quadratic traveling salesman problem (QTSP) where the binary variable x_{ij} is 1 if (i, j) belongs to the TSP tour. Thus the QAP differs from the AQAP in two key respects. First, since each binary variable x_{ij} in the latter represents whether or not travel is conducted along the route from i to j, no variables x_{ii} are found in the assignment set of this problem. Second, only those quadratic terms of the form $x_{ij}x_{kl}$ with $j = k$ and $i \neq l$ or with

$j \neq k$ and $i = l$ have nonzero objective function coefficients in the AQAP. Moreover the quadratic expressions of the form $x_{ij}x_{ji}$ are not allowed so as to eliminate subtours of two cities. It has been shown that the AQAP is NP-hard [44]. In Galbiati et al. [51] the authors introduce a new Combinatorial Optimization problem called *Minimum reload Cost Cycle Cover* which to best of our knowledge is the only related problem to AQAP. The MinRC3 consists of the problem of spanning the nodes of a given colored graph by a set of node-disjoint cycles at minimum reload cost, where a non-negative reload cost is paid whenever passing through a node where the two consecutive arcs have different colors.

Our main contribution consists in obtaining tight lower bounds for the AQAP. Our bounds are based on two mixed binary linear formulations that are derived by applying the RLT to AQAP. To derive the first model we apply a partial RLT to the AQAP, then using the special structure of the resulting mixed binary linear formulation, we extract a flow-based model which forms the basic frame of our study. By applying RLT to the basic model, we obtain another mixed binary linear program whose linear relaxation possesses some desirable properties relative to the constraint set. The second formulation has a large number of variables and constraints, and it is also highly degenerate. In order to develop a less computationally intensive lower bounding procedure, considerable attention is given to exploit the separable structure present in the LP relaxation of the problem and using Lagrangian relaxation techniques to obtain the LP bound.

5.3 Partial RLT Representation of the AQAP

To obtain a mixed binary formulation for the AQAP, we define $n(n-1)(n-2)$ continuous variables y_{ijk} such that

$$y_{ijk} = x_{ij}x_{jk} \text{ for all } (i,j,k) \text{ with } j \neq i \text{ and } k \neq i,j. \tag{5.4}$$

Then, a partial level-1 RLT representation of the AQAP can be obtained using two sets of operations. First, we multiply equations (5.1), for each $i \in \{1,\ldots,n\}$, by each of the $(n-1)$ variables x_{ki} for $k \neq i$. Then, for each $j \in \{1,\ldots,n\}$, we multiply equations (5.2) by each of the $(n-1)$ variables x_{jl} for $l \neq j$. All $2n(n-1)$ such quadratic equations are included within the formulation. We then set $x_{ij}x_{ji} = 0$ for all (i,j) with $i \neq j$. The linearization step makes the substitution of $y_{ijk} = x_{ij}x_{jk}$ for all (i,j,k) with $j \neq i$ and $k \neq i,j$, and enforces that all resulting variables y_{ijk} are nonnegative, to rewrite the problem as the mixed 0-1 linear program below.

$$\begin{aligned}
\text{PRLT1:} \quad & \min \quad \sum_{i}\sum_{j\neq i}\sum_{k\neq j,i} D_{ijk}y_{ijk} && && \\
& \text{s.t.} \quad \sum_{k\neq i,j} y_{kij} = x_{ij} && \forall\,(i,j), i\neq j && (5.5)\\
& \sum_{k\neq i,j} y_{ijk} = x_{ij} && \forall\,(i,j), j\neq i && (5.6)\\
& y_{ijk}\geq 0 && \forall\,(i,j,k),\ i\neq j\neq k,\ i\neq k && (5.7)\\
& (5.1),(5.2),(5.3). && &&
\end{aligned}$$

Lemma 5.3.1. *Problems AQAP and PRLT1 are equivalent.*

Proof. We prove that for any feasible solution $(\hat{\boldsymbol{x}},\hat{\boldsymbol{y}})$ of PRLT1

$$\hat{y}_{ijk} = \hat{x}_{ij}\hat{x}_{jk} \quad \forall\,(i,j,k);\ i\neq j\neq k;\ i\neq k. \tag{5.8}$$

Following the idea in [2], consider any $\hat{y}_{pqr}$. Constraint $\sum_{k\neq p,q}\hat{y}_{pqk} = \hat{x}_{pq}$ of (5.6), together with the nonnegativity restrictions $\hat{y}_{pqk}\geq 0$ for all $k\notin\{p,q,r\}$ of (5.7), enforces that $\hat{y}_{pqr}\leq\hat{x}_{pq}$. Similarly, the constraint $\sum_{k\neq q,r}\hat{y}_{kqr} = \hat{x}_{qr}$ of (5.5), together with the nonnegativity restrictions $\hat{y}_{kqr}\geq 0$ for all $k\notin\{p,q,r\}$ of (5.7), enforces that $\hat{y}_{pqr}\leq\hat{x}_{qr}$. Thus,

$$\hat{y}_{pqr}\leq\ \min\{\hat{x}_{pq},\hat{x}_{qr}\}\ \ \forall\,(p,q,r);\ p\neq q\neq r;\ p\neq r, \tag{5.9}$$

hence $\hat{y}_{pqr} = 0$ if either $\hat{x}_{pq} = 0$ or $\hat{x}_{qr} = 0$. In addition, if we subtract constraint $\sum_{k\neq q}\hat{x}_{qk} = 1$ of (5.1) from the equation $\sum_{k\neq p,q}\hat{y}_{pqk} = \hat{x}_{pq}$ of (5.6) to obtain

$$\sum_{k\neq p,q}(\hat{y}_{pqk}-\hat{x}_{qk}) = \hat{x}_{pq}+\hat{x}_{qp}-1. \tag{5.10}$$

By (5.9), we have $\sum_{k\neq p,q,r}(\hat{y}_{pqk}-\hat{x}_{qk})\leq 0$, hence (5.10) becomes

$$\hat{y}_{pqr}\geq\hat{x}_{pq}+\hat{x}_{qp}+\hat{x}_{qr}-1,$$

giving us that $\hat{y}_{pqr} = 1$ if $\hat{x}_{pq} = \hat{x}_{qr} = 1$. (Here, $\hat{x}_{pq}+\hat{x}_{qp}\leq 1$ since subtours of size 2 are not allowed.) Consequently, $\hat{y}_{pqr} = \hat{x}_{pq}\hat{x}_{qr}$ for binary $\hat{x}_{pq}$ and $\hat{x}_{qr}$, and the proof is complete. □

Considering the special structure of the PRLT1 formulation, one can obtain a compact representation for the problem as a consequence of the lemma and the theorem below.

Lemma 5.3.2. *The removal of constraints* (5.2) *from PRLT1 does not affect equivalence* (5.4).

Proof. If $x_{ij} = 0$ for $i \neq j$, constraints (5.5) imply $y_{ijk} = 0$ for all k such that $k \neq j$. If $x_{ij} = x_{jk} = 1$ and $y_{ijk} < 1$ then according to constraints (5.5) there exists an index $\ell \in \{1, \ldots, n\}$, $\ell \neq k$ such that $y_{ij\ell} > 0$. Constraints (5.6) together with $y_{ij\ell} > 0$ imply $x_{j\ell} = 1$, which contradicts (5.1). This contradiction completes the proof. □

We now prove that constraints (5.2) are redundant in PRLT1.

Theorem 5.3.3. *The optimal solution of PRLT1 does not change when removing constraints* (5.2).

Proof. Let PRLT1' represents the problem PRLT1 without constraints (5.2) and (x, y) be a feasible solution of PRLT1'. We show that (x, y) is also a feasible solution for the PRLT1. If not, there exists an index $t \in \{1, \ldots, n\}$ such that $\sum_{\substack{k=1\\k\neq t}}^{n} x_{kt} = \epsilon \neq 1$. Then according to Lemma 5.3.1, constraints (5.5) for index t and for all $j = 1, \ldots, n$, $j \neq t$ is written as follows

$$\sum_{k\neq t} y_{ktj} = \sum_{k\neq t} x_{kt}x_{tj} = x_{tj} \sum_{k\neq t} x_{kt} = \epsilon x_{tj}.$$

Since $\epsilon \neq 1$, the only possibility is that $x_{tj} = 0$ for all $j = 1, \ldots, n$, $j \neq t$, which contradicts equations (5.1). □

As a consequence of Theorem 7.2.1, the PRLT1 simplifies as follows:

$$\text{PRLT1':} \quad \min \left\{ \sum_{i} \sum_{j\neq i} \sum_{k\neq j,i} D_{ijk} y_{ijk} : (5.5), (5.6), (5.7), (5.1), (5.3) \right\}.$$

Observe that the continuous relaxations of the PRLT1' is as tight as the continuous relaxation of the PRLT1, in that a linear combination of the constraints of the latter problem implies constraint (5.2).

5.3.1 A flow-based solution method

In order to solve PRLT1' efficiently we introduce the following graph representation. Consider a directed graph $G = (V, E)$ where nodes in V correspond to feasible assignments in AQAP, that is $V = \{\langle i, j\rangle : i, j \in \{1, \ldots, n\}, i \neq j\}$, and arcs in E correspond to feasible pairs of assignments, that is, $E = \{(\langle i, j\rangle, \langle j, k\rangle) : \langle i, j\rangle, \langle j, k\rangle \in V, i \neq k\}$. Set V is partitioned into n clusters $V_1, V_2, \ldots, V_n$ such that $V_i = \{\langle i, j\rangle : j = 1, 2, ..., n, j \neq i\}$ for $i = 1, 2, ..., n$. Note that E only contains arcs between nodes from different clusters with a non negative cost, and there are no arcs between two nodes in the same cluster. A cost function $D : E \to \mathcal{R}^+$ is associated to the arcs of E and corresponds to the cost function of the AQAP associated with pairs of adjacent assignments.

A feasible solution of PRLT1' corresponds to a node disjoint cycle selection in G such that exactly one node of each cluster appears in one of the selected cycles. Let us call the problem of determining the cycle selection of minimum cost the *Generalized Minimum Cycle Cover* (GMCC). This problem can be formulated as follows:

$$\begin{aligned}
\text{GMCC:}\quad \min\quad & \sum_i \sum_{j\neq i} \sum_{k\neq i,j} D_{ijk} y_{ijk} & & \\
\text{s.t.}\quad & \sum_{i\neq j} \sum_{k\neq j,i} y_{ijk} = 1 & \forall j = 1,2,...,n \quad & (5.11)\\
& \sum_{j\neq i} \sum_{k\neq j,i} y_{ijk} = 1 & \forall i = 1,2,...,n \quad & (5.12)\\
& \sum_{k\neq i,j} y_{ijk} - \sum_{k\neq i,j} y_{kij} = 0 & \forall \langle i,j\rangle \in V \quad & (5.13)\\
& y_{ijk} \geq 0, \text{binary} & \forall (i,j,k), i\neq j\neq k; i\neq k, \quad & (5.14)
\end{aligned}$$

where variables y_{ijk} determine the arc selection, constraints (5.11) and (5.12) state that exactly one of the arcs entering the cluster and one of the arcs leaving the cluster must be selected. Constraints (5.13) are flow circulation constraints.

In the following theorem we show the possibility of obtaining a compact reformulation for the GMCC.

Theorem 5.3.4. *Constraints* (5.12) *in problem GMCC are redundant and can be removed from the model.*

Proof. Suppose y^* is a feasible solution of the GMCC after removing constraints (5.12) and ϵ_i be the value of $\sum_{j\neq i}\sum_{k\neq j,i} y^*_{ijk}$ for $i = 1,2,...,n$, We show that $\epsilon_i = 1$ for $i = 1,2,...,n$.

Let s be the index of the s-th cluster. Then according to (5.11), $\sum_{i\neq j}\sum_{k\neq j} y^*_{isk} = 1$. Therefore there is a node $\langle s,t\rangle$ in the cluster V_s with in-degree one, while the in-degree of all other nodes in the same cluster is zero. Hence by (5.11) the out-degree of the node $\langle s,t\rangle$ must be equal to one while out-degree of all the other nodes in the cluster V_s must be zero. This implies $\epsilon_s = 1$ and the proof is complete. □

As a consequence of Theorem 5.3.4, GMCC is simplified as:

$$\text{GMCC':}\quad \min\left\{ \sum_i \sum_{j\neq i} \sum_{k\neq i,j} D_{ijk} y_{ijk} \ :\ y \in Y,\ y \text{ binary} \right\}, \tag{5.15}$$

where $Y = \{y \geq 0 :\ (5.11), (5.13)\}$.

5.4 Bounds of the AQAP

The continuous relaxation of GMCC' provides a lower bound for the AQAP. In order to strengthen this lower bound, we apply RLT to GMCC' at the price of increased size. The key operation of the RLT applied to GMCC' is to multiply each constraints of the model by each binary variables x_{pql} with $p \neq q \neq l$ resulting the quadratic expression $x_{ijk}x_{pql}$. Since for all (i, j, k, p, q, l) with $i \neq j \neq k$ and $p \neq q \neq l$, the constraints defining Y do not allow to include all the products $x_{ijk}x_{pql}$ in the model, we introduce the following index sets:

$$
\begin{aligned}
&\mathcal{A} = \{(i, j, k) : i, j, k \in \{1, \ldots, n\}, i \neq j \neq k\},\\
&\mathcal{B} = \{(i, j, k, p, q, l) : (i, j, k), (p, q, l) \in A; p \neq i, j,\ q \neq i, j, k,\ l \neq j, k\},\\
&\mathcal{C}_1 = \{(i, j, k, p, i, j) : (i, j, k), (p, i, j) \in A\},\\
&\mathcal{C}_2 = \{(i, j, k, j, k, l) : (i, j, k), (j, k, l) \in A\},\\
&\mathcal{I} = \mathcal{B} \cup \mathcal{C}_1 \cup \mathcal{C}_2.
\end{aligned}
$$

where $\mathcal{A}$ represents the index set of the feasible pairs of assignments in the original problem, and $\mathcal{I}$ define an index set including all indices (i, j, k, p, q, l) for which the products $x_{ijk}x_{pql}$ are allowed in the formulation.

Considering the definition of the above index sets we apply the following reformulation and linearization steps to obtain an new mixed binary formulation for the GMCC':

Reformulation: Multiply each of the $n + n(n-1)$ equations (5.11), (5.13) and each of the $n(n-1)(n-2)$ nonnegativity constraints defining Y by each of the $n(n-1)(n-2)$ variables y_{pql}, $(p, q, l) \in \mathcal{A}$, and add these new constraints to the formulation. When a variable y_{ijk} in a given constraint is multiplied by y_{pql}, express the resulting product as $y_{ijk}y_{pql}$ in that order. Substitute y_{pql}^2 with y_{pql} in the objective function and in the constraints, and set $y_{ijk}y_{pql} = 0$ for all $(i, j, k, p, q, l) \notin \mathcal{I}$.

Linearization: Linearize the resulting problem by replacing the product $y_{ijk}y_{pql}$ by the continuous variable z_{ijkpql} for all $(i, j, k, p, q, l) \in \mathcal{I}$. Enforce the equality $z_{ijkpql} = z_{pqlijk}$ for all $(i, j, k, p, q, l) \in \mathcal{I}$ and $i < p$.

Once these steps have been applied to GMCC', the following mixed binary programming model is obtained, where the coefficients E_{ijkpql} found in the objective are all 0.

$$\text{NewRLT:} \quad \min \sum_{(i,j,k)\in\mathcal{A}} D_{ijk}y_{ijk} + \sum_{(i,j,k,l,p,q)\in\mathcal{I}} E_{ijkpql}z_{ijkpql} \tag{5.16}$$

$$\text{s.t.} \quad \sum_{p,l:(i,j,k,p,q,l)\in\mathcal{I}} z_{ijkpql} = y_{ijk} \quad \forall(q,i,j,k);\ (i,j,k)\in\mathcal{A},\ q\neq j \tag{5.17}$$

$$\sum_{l:(i,j,k,p,q,l)\in\mathcal{I}} z_{ijkpql} - \sum_{l:(i,j,k,l,p,q)\in\mathcal{I}} z_{ijklpq} = 0$$

$$\forall(i,j,k,p,q);\ (i,j,k)\in\mathcal{A},\ p\neq i,j,\ q\neq j,k \tag{5.18}$$

$$z_{ijkpql} = z_{pqlijk} \quad \forall(i,j,k,p,q,l)\in\mathcal{I};\ i<p \tag{5.19}$$

$$z_{ijkpql} \geq 0 \quad \forall(i,j,k,p,q,l)\in\mathcal{I} \tag{5.20}$$

$$y\in Y,\ y \text{ binary.}$$

Note that an optimal solution to the NewRLT will yield an optimal solution to the GMCC' problem. However, if the binary restrictions on variables y are relaxed in NewRLT, the problem is no longer equivalent to GMCC', providing a lower bound on the optimal objective value.

5.4.1 Lagrangian Relaxation Scheme

Consider the continuous relaxation of NewRLT (CNewRLT). Due to the large number of variables and constraints, and also degeneracy of the problem, solving CNewRLT to obtain a lower bound for AQAP is too time demanding. Therefore we apply a Lagrangian dual by relaxing constraints (5.19) using a set of Lagrangian multipliers λ_{ijkpql} for all $(i,j,k,p,q,l)\in\mathcal{I},\ i<p$. For convenience we assume that $\lambda_{pqlijk} = -\lambda_{ijkpql}$ for all $(p,q,l,i,j,k)\in\mathcal{I},\ i<p$. Let $\overline{D}_{ijk}$ and $\overline{E}_{ijkpql}$ denote the adjusted values for D_{ijk} and E_{ijkpql} respectively, after placing constraints (5.19) into the objective function. The resulting Lagrangian relaxation is:

$$\min\left\{K + \sum_{(i,j,k)\in\mathcal{A}} \overline{D}_{ijk}y_{ijk} + \sum_{(i,j,k,p,q,l)\in\mathcal{I}} \overline{E}_{ijkpql}z_{ijkpql} : y\in Y,\ (5.17),(5.18),(5.20)\right\}, \tag{5.21}$$

where

$$\begin{aligned}
&\overline{E}_{ijkpql} = E_{ijkpql} + \lambda_{ijkpql} \quad \forall (p,q,l,i,j,k) \in \mathcal{I}, && (5.22)\\
&\overline{D}_{ijk} = D_{ijk} \quad \forall (i,j,k) \in \mathcal{A},\\
&K = 0.
\end{aligned}$$

In order to solve (8.18) efficiently, we first solve $n(n-1)(n-2)$ independent sub problems $SP(i,j,k)$, one over each $(i,j,k) \in \mathcal{A}$:

$$\varphi_{ijk} = \min \sum_{\substack{(p,q,l)\in\mathcal{A}:\\(i,j,k,p,q,l)\in\mathcal{I}}} \overline{E}_{ijkpql} z_{ijkpql} \tag{5.23}$$

$$\text{s.t.} \sum_{\substack{p,l:\\(i,j,k,p,q,l)\in\mathcal{I}}} z_{ijkpql} = 1 \quad \forall q;\ q \neq j \tag{5.24}$$

$$\sum_{\substack{l:\\(i,j,k,p,q,l)\in\mathcal{I}}} z_{ijkpql} - \sum_{\substack{l:\\(i,j,k,l,p,q)\in\mathcal{I}}} z_{ijklpq} = 0 \quad \forall p,q;\ p \neq i,j,\ q \neq j,k \tag{5.25}$$

$$z_{ijkpql} \geq 0 \quad (i,j,k,p,q,l) \in \mathcal{I}. \tag{5.26}$$

then the optimal objective value of (8.18) is given by the solution of the following GMCC':

$$MP: \quad \Delta = \min \quad \{ \sum_{(i,j,k)\in\mathcal{A}} (\overline{D}_{ijk} + \varphi_{ijk}) y_{ijk} :\ y \in Y \}, \tag{5.27}$$

The decomposition result is stated in the following theorem.

Theorem 5.4.1. *An optimal solution* $(\boldsymbol{y}^*, \boldsymbol{z}^*)$ *to the problem* (8.18) *is given by*

$$\begin{aligned}
&\boldsymbol{y}^* = \hat{\boldsymbol{y}},\\
&z^*_{ijkpql} = \hat{z}_{ijkpql} \hat{y}_{pql} \quad \forall (i,j,k,p,q,l) \in \mathcal{I},
\end{aligned}$$

where $\hat{\boldsymbol{z}}$ *and* $\hat{\boldsymbol{y}}$ *are optimal solutions of sub problem* $SP(i,j,k)$, $(i,j,k) \in \mathcal{A}$ *and* MP, *respectively.*

Proof. We propose the proof similar to the one used in [3] for the RLT2 representation of the QAP. We first show that $(\boldsymbol{y}^*, \boldsymbol{z}^*)$ is feasible to problem (8.18). Since $\hat{\boldsymbol{y}} \in Y$ and $\hat{\boldsymbol{z}}$ satisfies (8.27)–(5.26), it follows that:

$$\sum_{\substack{p,l:\\(i,j,k,p,q,l)\in\mathcal{I}}} z^*_{ijkpql} = \sum_{\substack{p,l:\\(i,j,k,p,q,l)\in\mathcal{I}}} \hat{z}_{ijkpql} y^*_{ijk} = y^*_{ijk},$$

and

$$\sum_{\substack{l:\\(i,j,k,p,q,l)\in\mathcal{I}}} z^*_{ijkpql} - \sum_{\substack{l:\\(i,j,k,l,p,q)\in\mathcal{I}}} z^*_{ijklpq} = \sum_{\substack{l:\\(i,j,k,p,q,l)\in\mathcal{I}}} \hat{z}_{ijkpql} y^*_{ijk} - \sum_{\substack{l:\\(i,j,k,l,p,q)\in\mathcal{I}}} \hat{z}_{ijklpq} y^*_{ijk} = y^*_{ijk}\Big(\sum_{\substack{l:\\(i,j,k,p,q,l)\in\mathcal{I}}} \hat{z}_{ijkpql} - \sum_{\substack{l:\\(i,j,k,p,q,l)\in\mathcal{I}}} \hat{z}_{ijklpq}\Big) = 0.$$

Next, we should find a dual solution to (8.18) that, together with $(\boldsymbol{y}^*, \boldsymbol{z}^*)$, satisfies complementary slackness conditions. To this end, let $(\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\nu}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\gamma}})$ be the optimal dual solution corresponding to constraints (8.27) to (5.26), two equality constraints and nonnegativity restriction in Y respectively. For each $(i,j,k) \in \mathcal{A}$, dual feasibility of (8.26)–(5.26) $\forall (p,q,l) \in \mathcal{A}$ such that $(i,j,k,p,q,l) \in \mathcal{I}$ gives:

$$\hat{\mu}_{ijkq} + \hat{\nu}_{ijkpq} - \hat{\nu}_{ijkql} + \hat{\pi}_{ijkpql} = \overline{E}_{ijkpql},$$

Moreover, dual feasibility of $(\hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\gamma}})$ to (8.25) for each $(i,j,k) \in \mathcal{A}$, and dual optimality of $\hat{\mu}_{ijkq}$ $\forall q \neq j$ to (8.26)–(5.26) provide:

$$\hat{\alpha}_j + \hat{\beta}_{ij} - \hat{\beta}_{jk} + \hat{\gamma}_{ijk} = \overline{D}_{ijk} + \varphi_{ijk} = \overline{D}_{ijk} + \sum_{q\neq j} \hat{\mu}_{ijkq}.$$

Therefore $(\boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{\pi}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) = (\hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\nu}}, \hat{\boldsymbol{\pi}}, \hat{\boldsymbol{\alpha}}, \hat{\boldsymbol{\beta}}, \hat{\boldsymbol{\gamma}})$ is a dual feasible solution to (8.18), where the variables $\boldsymbol{\mu}, \boldsymbol{\nu}$ and $\boldsymbol{\pi}$ correspond to constraints (5.17), (5.18) and (5.20) respectively and the variables $\boldsymbol{\alpha}, \boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ correspond to the two equality constraints and nonnegativity restriction defining Y respectively.

Next, we will show that the optimal value of problem (8.18) is equal to Δ. Dual optimality of $(\hat{\alpha}, \hat{\beta}, \hat{\gamma})$ to (8.25) implies:

$$\sum_j \hat{\alpha}_j = \sum_{(i,j,k)\in\mathcal{A}} (D_{ijk} + \varphi_{ijk}) y^*_{ijk} = \sum_{(i,j,k)\in\mathcal{A}} \Big(D_{ijk} + \sum_{\substack{(p,q,l)\in\mathcal{A}:\\(i,j,k,p,q,l)\in\mathcal{I}}} \overline{E}_{ijkpql} \hat{z}_{ijkpql}\Big) y^*_{ijk}$$
$$= \sum_{(i,j,k)\in\mathcal{A}} D_{ijk} y^*_{ijk} + \sum_{(i,j,k,p,q,l)\in\mathcal{I}} \overline{E}_{ijkpql} z^*_{ijkpq\ell}.$$

Note that the second equality follows from the definition of φ_{ijk} $\forall$ $(i,j,k) \in \mathcal{A}$, while the last equality results from the definition of $\boldsymbol{z}^*$. □

Using this theorem one can easily check that, if $y^*_{ijk} = 1$ for some $(i,j,k) \in \mathcal{A}$, then

$\hat{\boldsymbol{w}} = \hat{\boldsymbol{z}}_{ijk}$ is a vector that minimize the following problem:

$$\varphi_{ijk} = \left\{ \sum_{\substack{(p,q,l)\in\mathcal{A}:\\ (i,j,k,p,q,l)\in\mathcal{I}}} \overline{E}_{ijkpql} w_{pql} \;:\; \boldsymbol{w} \in Y, \text{and } w_{ijk} = 1 \right\}$$

which is the same as the continuous relaxation of GMCC' (CGMCC) with an extra constraint $w_{ijk} = 1$. Thus, the decomposition strategy proposed in Theorem 5.4.1 provides a way of solving problem (8.18) by solving $n(n-1)(n-2)+1$ $CGMCC$ problems.

In order to develop a procedure to find the optimal dual multipliers, we use the idea of splitting the costs between symmetrical entries of E so as to maximize the value of Δ. This idea has been proposed in [25] and already used to solve the RLT representations of the QAP in [3, 59, 60]. Using this idea, it is not necessary to explicitly find the value of the dual multipliers λ_{ijkpql} in each iteration of the dual ascent algorithm. In fact one can adjust the coefficient $\overline{E}_{ijkpql}$ on variables z_{ijkpql} every time multiples of (5.19) are placed into the objective function. More precisely, for any $(i,j,k,p,q,l) \in \mathcal{I}$ with $i < p$, we can increase $\overline{E}_{ijkpql}$ by the same quantity by which $\overline{E}_{pqlijk}$ is decreased.

Dual-ascent Strategy

In the following we present a dual-ascent algorithm similar to that used in [3] for the RLT2 representation of the QAP. The procedure consists of updating the constant term K and the cost matrices $\overline{D}$ and $\overline{E}$ in such a way that the cost of any feasible solution with respect to the modified objective function remains unchanged while maintaining nonnegative coefficients. As a consequence of this property, the value of K at any iteration of the algorithm is a valid lower bound on the optimal solution cost for the AQAP. The main steps of the dual-ascent algorithm are summarizes as follows:

1 Initialization: Set $\overline{E}_{ijkpql} = 0$ for all $(i,j,k,p,q,l) \in \mathcal{I}$, $\overline{D}_{ijk} = D_{ijk}$ for all $(i,j,k) \in \mathcal{A}$, $K = 0$, and an iteration counter $I = 0$.

2 Spreading: For each $(i,j,k) \in \mathcal{A}$, spread the coefficient $\overline{D}_{ijk}$ amongst the coefficient $\overline{E}_{ijkpql}$ for all $(i,j,k,p,q,l) \in \mathcal{I}$; i.e, $\overline{E}_{ijkpql} = \frac{\overline{D}_{ijk}}{n-1}$. Then update $\overline{D}_{ijk}$ to 0 for each $(i,j,k) \in \mathcal{A}$.

3 Solve the Lagrangian relaxation problem: Use Theorem 5.4.1 to solve (8.18) as follows:

(a) Solve the sub problems: solve $n(n-1)(n-2)$ problems CGMCC defined in (8.26)–(5.26) to obtain $\hat{\boldsymbol{z}}$. For a selected $(i,j,k) \in \mathcal{A}$, change the coefficients $\overline{E}_{ijkpql}$ to a percentage of the sum of $\overline{E}_{ijkpql}$ and $\overline{E}_{pqlijk}$ and adjust $\overline{E}_{pqlijk}$

so that the sum stays constant. (For $i < p$ we set $\overline{E}_{ijkpql} = \overline{E}_{pqlijk} = \frac{\overline{E}_{ijkpql}+\overline{E}_{pqlijk}}{2}$, and for $i > p$ we set $\overline{E}_{ijkpql} = \overline{E}_{ijkpql} + \overline{E}_{pqlijk}$, $\overline{E}_{pqlijk} = 0$). Solve the subproblem and place the associated equations (5.17) and (5.18) into the objective function with the optimal dual multipliers, readjust $\overline{E}_{ijkpql}$ and increase $\overline{D}_{ijk}$ by φ_{ijk}.

(b) Solve the problem (8.25) to obtain $\hat{\boldsymbol{y}}$ and place the equality constraints of Y into the objective function with the optimal dual multipliers, readjusting $\overline{D}_{ijk}$ and increasing the scalar K by the nonnegative value of Δ.

4 If $I \geq MaxIteration$ or $\Delta =$ Optimal objective value, stop, otherwise set $I = I + 1$ and return to 2.

5.5 QAP on Reducible graphs

Consider a special QAP in Koopmans and Beckmann form where the graphs associated to flow and the distance coefficient matrices is belong to a class of graphs with a special structure namely the Reducible graphs. Malucelli and Pretolani in [84] studied quadratic semi assignment problem (QSAP) on reducible graphs and have presented a polynomial time algorithm for the problem. In this section we introduce the class of reducible graphs, review the algorithm presented by Malucelli and Pretolani for the QSAP, and provide a lower bounding procedure for the QAP on reducible graphs based on a Lagrangian decomposition whose subproblems are QSAPs.

5.5.1 Reducible graphs

Consider undirected graph $G(V, E)$ with $|V| = n$ and $|E| = m$. The graph G is called reducible if and only if it can be reduced to a single node by the following operations:

- Tail reduction: Let i be a node of degree 1 (i.e. there is only one arc incident with node i) and (j, i) be the arc connecting node i to the rest of the graph G. The graph G can be reduced to a new graph G' where node i and arc (j, i) have been deleted. This reduction operation is denoted by $Tail(i)$.

- Series reduction: Let i be a node of degree 2 and let (i, j) and (i, h), $j \neq h$, the two arcs incident with i; the graph G can be reduced to a new graph G' obtained from G by eliminating node i, arc (j, i), arc (i, h) and adding a new arc (j, h). This operation is denoted by $Series(h, i, j)$.

- Parallel reduction: Let $a = (i, j)$ and $a' = (i, j)$ be two “parallel” arcs of graph G. The graph can be reduced to a new graph G' with a single arc between nodes i and j. This operation is denoted by $Parallel(i, j)$.

The reducible graphs have some important properties including and not limit to

- By applying repeatedly reductions to a graph G as far as possible; the resulting graph is independent of the sequence of reductions.
- If G is a connected graph, any graph G', obtained from G by reduction, is also connected.
- Any simple reducible graph (i.e. without parallel arcs) contains at most $2n-3$ arcs.

The class of reducible graphs can be considered as an extension of an another calls of graphs called *series-parallel* graphs. Applying a sequence of series and parallel reduction on the *series-parallel* graphs imply a single arc. More precisely any *series-parallel* graph is reducible while there exist reducible graphs which are not *series-parallel* graphs. As an example, trees belong to the class of reducible graphs, but they are not series-parallel in general. More details concerning the reducible graphs can be found in [84].

5.5.2 Quadratic Semi-Assignment Problem on reducible graph

Consider the QAP in the Koopmans and Beckmann form. Dropping constraints $\sum_{i=1}^{n} x_{ij} = 1\ \forall j = 1, \ldots, n$ from the set X implies the following relaxed problem which is called Quadratic Semi-Assignment Problem (QSAP).

$$\text{SQAP:} \quad \min \quad \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} f_{ij} d_{kl} x_{ik} x_{jl} + \sum_{i=1}^{n}\sum_{k=1}^{n} c_{ik} x_{ik} \tag{5.28}$$

$$\text{s.t.} \quad \sum_{j=1}^{n} x_{ij} = 1 \quad \forall i = 1, \ldots, n \tag{5.29}$$

$$x \text{ binary.} \tag{5.30}$$

This problem is well known to belong to the class of the NP-hard optimization problems [108]. Malucelli and Pretolani [84] introduced a class of the QSAP where the graph G_f corresponding to the flow matrix is reducible. They proposed a polynomial time algorithm to solve this special problem and provide lower bounds and solution techniques for the general case of the QSAP. In the following we present their approach in more details.

A polynomial algorithm for the QSAP on reducible graphs:

Consider undirected connected graph $G_f(V_f, E_f)$ where V_f represents the set of facilities $\{1, 2, \ldots, n\}$ and E_f is determined by the positive coefficients f_{ij}, that is $E_f = \{(i,j) \in V_f \times V_f : f_{ij} > 0 \text{ or } f_{ji} > 0\}$. Let V_d represents the set of location. For each $i \in V_f$ define the labels $u_{ik} \forall k \in V_d$ and for each $(i,j) \in E_f$ define the labels $w_{ikjl} \forall k,l \in V_d$. The labels are initialized as follows:

$$u_{ik} = c_{ik} \quad \forall i \in V_f, k \in V_d$$
$$w_{ikjl} = f_{ij}d_{kl} + f_{ji}d_{lk} \quad \forall i,j \in V_f, k,l \in V_d$$

Since in the QSAP any two arcs in the solution can be incident in the same location, the solution method consists of updating the labels according to the reduction operations performed on G_f. At the end, when G_f has been reduced to a single node, the minimum label of that node gives the optimal solution value. The process of updating the labels is described as follows:

- Updating the labels u by tail reduction: Consider the node $i \in V_f$ with $deg(i) = 1$ and let $(j,i) \in E_f$. First do the Tail reduction $Tail(i)$ on G_f to obtain the $G_f(T)$, set $G_f = G_f(T)$, and for each $k \in V_d$ update the labels u_{jk} as follows:

$$u_{jk} = \min\{u_{il} + w_{iljk}, l \in V_d\} \tag{5.31}$$

  Note that u_{jk} is modified in order to take into account the best possible assignment for i once j has been assigned to k. This operation can be carried out in $O(n^2)$ time.

- Updating the labels w by Series reduction: Consider the node $i \in V_f$ with $deg(i) = 2$ and let $(i,j),(i,h) \in E_f$. First do the series reduction $Series(h,i,j)$ on G_f to obtain the $G_f(S)$, set $G_f = G_f(S)$, and for each $k,l \in V_d$ update the labels w_{jlhk} as follows:

$$w_{jlhk} = \min\{w_{itjl} + w_{ithk} + u_{it}, t \in V_d\} \tag{5.32}$$

  The label w_{jlhk} takes into account the best possible assignment for i once j and h have been assigned to l and k, respectively. This operation can be carried out in $O(n^3)$ time.

- Updating the labels w by Parallel reduction: Consider two parallel arcs $a' = a'' = (i,j)$ with labels w'_{ikjl} and w''_{ikjl} respectively. By doing the parallel reduction $Parallel(i,j)$ on G_f obtain the $G_f(P)$, set $G_f = G_f(P)$, and for each $k,l \in V_d$ update the labels w_{ikjl} as $w_{ikjl} = w'_{ikjl} + w''_{ikjl}$.

Since applying reduction operations on a reducible graph needs at most $O(n)$, the overall complexity of the transformations is $O(nn^3)$.

The process of updating the labels is terminated when G_f is reduced to a single node. In order to obtain the optimal solution one can store in a stack the local choices made in each Series or Tail reduction operation. Let node i and arc (i, j) eliminated from G_f by tail reduction. For each $k \in V_d$ let $i(k)$ denotes the index $l \in V_d$ giving the minimum in (5.31). We put on the stack a label Tail, the nodes i and j, and the set $\{i(k) : k = 1, 2, \ldots, n\}$. For the Series reduction, let (j, h) be the new arc introduced by the reduction and let i be the eliminated node; for each pair $k, l \in V_d$, denote by $i(k, l)$ the index $t \in V_d$ giving the minimum in (5.32). We put on the stack a label Series, the nodes j, h and i, and the set $\{i(k, l) : k, l = 1, 2, \ldots, n\}$.

By keeping track of the choices made during the reduction process we are able to obtain the optimal assignment ρ going backwards, once G_f has been reduced to a single node. More precisely let $u_i k$ be the minimum label of the remaining single node i; set $\rho(i) = k$. Then, repeatedly remove elements from the stack and, according to the label Tail or Series, perform the following operations:

$$Tail: \quad \text{let } k = \rho(j);\ \text{set } \rho(i) = i(k);$$
$$Series: \quad \text{let } k = \rho(h) \text{ and } l = \rho(j);\ \text{set } \rho(h) = i(k, l).$$

5.5.3 A Lagrangain Decomposition approach

The QAP on reducible graphs is a generalization of the QAP on series-parallel digraph. Rendl in [104] studied the QAP on series-parallel digraphs and showed that the problem is NP-hard. He also reported that the class of series-parallel digraphs not containing bipartite subgraphs is solvable in polynomial time. Since the class of series-parallel graphs is included in the class of reducible graphs, and QAP on series-parallel digraphs is NP-hard, the QAP on reducible graphs is also NP-hard. Considering the NP-hardness of the problem we focus our attention to find a lower bound for the problem.

Given a QAP in Koopmans and Beckmann form, let $G_f(V_f, E_f)$ and $G_d(V_d, E_d)$ are two connected reducible graphs called flow and distance graphs corresponding to the flow and distance matrices of the QAP where V_f and V_d represent the set of facilities and the set of locations, and E_f and E_d are determined by the coefficients f_{ij} and d_{kl} respectively, i.e.,

$$E_f = \{(i, j) \in V_f \times V_f : \ f_{ij} \geq 0 \text{ or } f_{ji} \geq 0\}$$
$$E_d = \{(k, l) \in V_d \times V_d : \ d_{kl} \geq 0 \text{ or } f_{lk} \geq 0\}.$$

The QAP structure along with reducibility of the flow and distance graphs can be used to obtain improved problem formulations and more effective algorithm that take the structure into account. One typical way to obtain a reformulation for QAP is to split the variables x into two groups x and y and then join the two groups of variables with a new “linking” constraint $x = y$. The resulting problem has the following form:

$$\text{P:} \quad \min\left\{\, f(x) + g(y) : \; x \in X_f, \; y \in Y_d, \; x = y^t \right\},$$

where

$$f(x) = \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n}(1/2f_{ij})d_{kl}x_{ik}x_{jl} + \sum_{i=1}^{n}\sum_{k=1}^{n} 1/2c_{ik}x_{ik}$$

$$g(y) = \sum_{k=1}^{n}\sum_{l=1}^{n}\sum_{i=1}^{n}\sum_{j=1}^{n} d_{kl}(1/2f_{ij})y_{ki}x_{lj} + \sum_{k=1}^{n}\sum_{i=1}^{n} c_{ik}y_{ki}$$

$$X_f = \{x : \sum_{k=1}^{n} x_{ik} = 1 \quad \forall i = 1,\ldots,n; \; x \text{ binary}\} \tag{5.33}$$

$$Y_d = \{y : \sum_{i=1}^{n} y_{ki} = 1 \quad \forall k = 1,\ldots,n; \; y \text{ binary}\}. \tag{5.34}$$

Problem P is clearly equivalent to the QAP in the sense that they have equal optimal values, but having different variable spaces. In addition if x^* is an optimal solution of QAP, then the solution $(x, y) = (x^*, x^*)$ is the optimal solution for problem P, and if (x^*, x^*) is an optimal solution of P, then $x^* = y^*$ is the optimal solution of the QAP.

In order to solve problem P we apply Lagrangian relaxation by dualizing the linking constraint $x = y$ with multipliers λ. This separates the problem into an x-problem and an y-problem. the Lagrangian dual is thus defined as:

$$\begin{aligned} LD(\lambda, x, y) : \quad & \min_{x,y}\{f(x) + g(y) + \lambda(y - x) \mid x \in X_f, \; y \in Y_d\} \\ & = \min_{x}\{f(x) - \lambda x \mid x \in X_f\} + \min_{y}\{f(y) + \lambda y \mid y \in Y_d\}. \end{aligned}$$

Observe that both subproblems are quadratic semi assignment problems on reducible graphs, thus are solvable by using the polynomial time algorithm present by Malucelli and Pretolani for QSAP on reducible graphs.

5.6 Computational experiments

In this section we report the computational experiments of lower bounding approaches for both the Adjacent QAP and the QAP on reducible graphs. In the following we first present the computational details for AQAP and then present the results for the QAP on reducible graphs.

5.6.1 Computational results for AQAP

In the following subsections we first introduce the test instances and then provide the computational details of applying the dual-ascent procedure to solve NewRLT formulation. We compare the NewRLT with the classical RLT2 representations of the QAP for evaluating both the strength of the resulting bounds and computational effort.

Test instances of AQAP

We use a class of randomly generated instances with size ranging from $n = 10$ to $n = 20$ introduced in [42]. All instances consist of complete graphs with n vertices and $m = n(n-1)/2$ edges. The cost function $D : E \times E \to \mathcal{R}^+$ is defined as an integer number which is chosen from the set $\{0, 1, ..., 10000\}$ uniformly for all $(i, j), (j, k) \in E$ such that $i \neq k$.

These instances can be transformed to the standard QAP instances by defining a new interaction cost C_{ijkl} for all (i, j, k, l) with $i \neq k, j \neq l$ as follows:

$$q_{ijkl} = \begin{cases} \frac{1}{2}D_{ijl} & (i,j,l) \in \mathcal{A},\ j = k \\ \frac{1}{2}D_{klj} & (k,l,j) \in \mathcal{A},\ i = l \\ M & (i = j \text{ or } k = l) \text{ or } (i \neq j, k \neq l, l = i, k = j) \\ 0, & \text{otherwise} \end{cases}$$

where M is a large number. It is not difficult to see that, the optimum value of the standard QAP instance obtained with this transformation is equal to the optimum value of the AQAP instance.

Lower bounds computation for AQAP

In this section we present the computational experiments comparing the lower bounds obtained with CGMCC, classical RLT2 [3], and NewRLT2. We implemented all algorithms in C++ language and run on an Intel Xeon CPU E5335 (2 quad core CPUs 2GH). To solve the standard RLT2 we applied the dual-ascent algorithm presented in [3] and used the Hungarian algorithm to solve the assignment sub problems. To solve NewRLT we applied the dual-ascent algorithm presented in Section 5.4.1, while foe solving the CGMCC problems arise each sub-problem of NewRLT formulation we used CPLEX 12.5 with default setting. Due to the tradeoff between bound strength and CPU execution time we terminated the dual-ascent algorithms for RLT2 and NewRLT in 1000 and 25 iterations, respectively, if the optimal solution has been not found yet.

Tables 6.1-6.3 report the lower bounds and CPU times obtained by using different approaches. To do a fair comparison between RLT2 and NewRLT we reported the lower

Table 5.1 Comparison of different lower bounding approaches for the AQAP instances.

				RLT2						NewRLT2					
Instance		PRLT1		Itr(100)		Itr(500)		Itr(1000)		Itr(5)		Itr(15)		Itr(25)	
n	Opt.	Lb	time	Lb	time	Lb	time	Lb	time	Lb	time	Lb	time	Lb	time
10	12899	10849.9	0	12899	4	-	-	-	-	12899	10	-	-	-	-
10	11477	8213.5	0	11477	4	-	-	-	-	11477	10	-	-	-	-
10	13434	8791.1	0	13434	8	-	-	-	-	13434	13	-	-	-	-
10	10317	8024.1	0	10317	5	-	-	-	-	10317	13	-	-	-	-
10	10936	9632.8	0	10936	5	-	-	-	-	10936	6	-	-	-	-
10	14361	10761	0	14361	6	-	-	-	-	14361	16	-	-	-	-
10	10783	7317.1	0	10783	6	-	-	-	-	10783	16	-	-	-	-
10	13811	10486.9	0	13811	7	-	-	-	-	13811	13	-	-	-	-
10	11937	9416.1	0	11937	5	-	-	-	-	11937	10	-	-	-	-
10	15053	10777.4	0	15053	10	-	-	-	-	15053	10	-	-	-	-
11	13056	10299.8	0	13056	13	-	-	-	-	13056	31	-	-	-	-
11	11463	9416.2	0	11463	12	-	-	-	-	11463	15	-	-	-	-
11	11146	8350.1	0	11146	13	-	-	-	-	11146	20	-	-	-	-
11	12257	9301.7	0	12257	25	-	-	-	-	12248.7	25	12257	36	-	-
11	13744	10084.1	0	13744	24	-	-	-	-	13601.6	25	13744	46	-	-
11	11171	8965.1	0	11171	9	-	-	-	-	11171	15	-	-	-	-
11	11186	8234.8	0	11186	18	-	-	-	-	11186	30	-	-	-	-
11	12809	9035.0	0	12809	17	-	-	-	-	12809	30	-	-	-	-
11	11347	8906.0	0	11347	17	-	-	-	-	11347	31	-	-	-	-
11	12720	7134.9	0	12720	21	-	-	-	-	12680.3	25	12720	35	-	-
12	10961	9200.6	0	10961	18	-	-	-	-	10961	33	-	-	-	-
12	8280	8066.5	0	8280	13	-	-	-	-	8280	17	-	-	-	-
12	10286	8079.0	0	10286	35	-	-	-	-	10286	25	-	-	-	-
12	10627	7685.5	0	10627	42	-	-	-	-	10627	32	-	-	-	-
12	9845	8469.1	0	9845	18	-	-	-	-	9845	25	-	-	-	-
12	12459	9664.1	0	12459	30	-	-	-	-	12459	40	-	-	-	-
12	9414	9237.9	0	9414	17	-	-	-	-	9414	17	-	-	-	-
12	12625	8764.2	0	12625	43	-	-	-	-	12625	33	-	-	-	-
12	12466	9172.5	0	12466	43	-	-	-	-	12466	48	-	-	-	-
12	12016	8894.1	0	12016	42	-	-	-	-	12016	48	-	-	-	-
13	11001	9285.5	0	11001	37	-	-	-	-	11001	39	-	-	-	-
13	11459	8674.1	0	11459	115	-	-	-	-	11459	51	-	-	-	-
13	11319	6680.9	0	10994.4	119	11319	222	-	-	11189.1	64	11319	101	-	-
13	11688	8011.0	0	11661.3	120	11688	143	-	-	11688	75	-	-	-	-
13	11434	8243.7	0	11434	56	-	-	-	-	11434	77	-	-	-	-
13	12604	8911.6	0	12574.9	122	12604	144	-	-	12604	76	-	-	-	-
13	9620	8516.5	0	9620	30	-	-	-	-	9620	40	-	-	-	-
13	12015	8104.6	0	12015	100	-	-	-	-	12015	64	-	-	-	-
13	12250	8936.	0	11843.9	120	12250	278	-	-	12250	101	-	-	-	-
13	12453	8852.6	0	12396.7	119	12453	150	-	-	12453	77	-	-	-	-
14	12668	9077.7	0	12300.8	196	12668	362	-	-	12264	99	12668	217	-	-
14	9708	7608.5	0	9708	80	-	-	-	-	9708	62	-	-	-	-
14	10611	7100.7	0	10495.7	196	10611	284	-	-	10611	99	-	-	-	-
14	11086	8391.2	0	11086	148	-	-	-	-	11086	99	-	-	-	-
14	9264	8582.0	0	9264	91	-	-	-	-	9264	43	-	-	-	-
14	11615	8746.5	0	11615	114	-	-	-	-	11615	99	-	-	-	-
14	11073	7316.4	0	10860.9	206	11073	361	-	-	10885.7	100	11073	159	-	-
14	11289	7612.6	0	10770.6	195	11289	399	-	-	11188.5	100	11289	161	-	-
14	10979	8861.3	0	10979	121	-	-	-	-	10979	98	-	-	-	-
14	10704	8530.6	0	10704	131	-	-	-	-	10704	80	-	-	-	-
15	12183	8856.6	0	11916	305	12183	607	-	-	12183	155	-	-	-	-
15	11061	8201.2	0	10864.8	307	11061	428	-	-	11061	157	-	-	-	-
15	10516	7055.5	0	10024.9	310	10516	741	-	-	10509.5	157	10516	216	-	-
15	12886	7603.8	0	11004	307	11796.9	1519	11864.6	3015	11928.5	158	12286	308	-	-
15	11867	8133.1	0	10569.3	307	11357.1	1552	11413.4	3090	11067.4	154	11867	303	-	-
15	13432	8904.6	0	11673.7	303	12273.6	1504	12337.4	3008	12313.6	155	13326.5	446	13432	598
15	9335	7992.8	0	9335	216	-	-	-	-	9335	101	-	-	-	-
15	12416	7973.5	0	10992.3	311	11693	1508	11745.7	2996	11928.6	157	12416	280	-	-
15	12140	8279.0	0	11480.8	309	12137.8	1526	12140	1556	12119.6	158	12140	218	-	-
15	12252	7946.3	0	10898.7	307	11600.2	1520	11662.6	3030	11741.4	163	12252	321	-	-

Table 5.2 Comparison of different lower bounding approaches for the AQAP instances.

				RLT2						NewRLT2					
Instance		PRLT1		Itr(100)		Itr(500)		Itr(1000)		Itr(5)		Itr(15)		Itr(25)	
n	Opt.	Lb	time	Lb	time	Lb	time	Lb	time	Lb	time	Lb	time	Lb	time
16	13334	8693.2	0	11246.6	618	11917.3	3076	11967.9	6128	12144	231	13187.2	696	1334	1197
16	10166	8084.4	0	10166	622	-	-	-	-	10166	254	-	-	-	-
16	10306	7202.4	0	9513.9	620	10152.3	3076	10209.4	6179	10289.5	268	10306	424	-	-
16	10684	7224.5	0	10280.4	608	10684	1520	-	-	10683.8	254	10684	349	-	-
16	11634	7881.1	0	10385.7	609	11032.8	3099	11091	6336	11158.2	270	11634	591	-	-
16	13049	8699.5	0	11091.8	655	11776.1	3236	11853.2	6458	11939.9	253	12907.6	483	13023	1225
16	10453	7188.0	0	9654.1	562	10387.9	3295	10453	5888	10211.3	263	10453	631	-	-
16	9956	6571.1	0	9106.4	648	9838.1	3346	9956	6697	9869.8	248	9956	384	-	-
16	10383	8136.9	0	10026.2	572	10383	1165	-	-	10383	246	-	-	-	-
16	11233	7086.8	0	9477	661	10054.6	3325	10110.6	6648	10221.6	259	11233	771	-	-
17	11823	7799.2	0	9882.6	737	10589.6	3460	-10655	6799	10801.4	398	11777.8	1162	11823	1405
17	11072	7283.9	0	9370.6	691	10117.5	3421	10212.6	6805	10186.8	396	11046.7	1161	11072	1406
17	9801	7107.4	0	9475.3	684	9801	1501	-	-	9801	403	-	-	-	-
17	12004	7315.0	0	9917.4	682	10702.6	3390	10771.3	7106	10802.7	398	12004	1190	-	-
17	11314	7730.7	0	9729.8	709	10374.1	3521	10435.6	6992	10540.6	394	11314	1073	-	-
17	11682	6694.8	0	9748.6	711	10413.9	3507	10496.2	6972	10804.1	391	11562.3	1125	11682	1888
17	10248	7286.7	0	9533.8	698	10181.5	3437	10248	6173	10220.5	399	10248	619	-	-
17	9009	6707.0	0	8761.1	693	9009	1376	-	-	9009	390	-	-	-	-
17	10866	7021.4	0	9338.7	688	9940.7	3551	10019.8	7083	10702.2	394	10866	608	-	-
17	11637	7467.1	0	9661.8	672	10270.1	3391	10343.5	6729	10874.8	397	11621.9	1125	11637	1357
18	12302	8096.2	0	10099.4	1151	10855	5544	10935.7	10681	11452	2001	12302	5625	-	-
18	8814	6873.9	0	8744	1094	8814	1522	-	-	8814	2018	-	-	-	-
18	12460	7930.9	0	9684	1061	10234.5	5388	10296.3	10720	10772.9	1998	11744.1	5496	11981.8	9533
18	11702	7181.0	0	9263.3	1055	9954.8	5341	10041.4	10583	10573.6	1907	11479.4	5348	11700.3	9356
18	10555	6670.5	0	8474	1034	9133.7	5098	9213	10260	9445.4	1921	10417.6	5325	10555	7563
18	12578	8051.8	0	9787.9	926	10640.4	5085	10718.3	10049	11009.1	1987	12132.9	5820	12427.9	10063
18	11054	7502.5	0	9129.5	1002	9700.8	4958	9751.9	9999	10268.7	2041	11054	5954	-	-
18	10900	6993.9	0	8909.1	1050	9542.1	5059	9628.5	10001	10113.9	1954	10900	5159	-	-
18	10626	6749.5	0	8647.2	1010	9347.6	5068	9417.2	9954	9856.4	1928	10626	5051	-	-
18	10055	7759.1	0	9199.7	1043	9864.8	5308	9984.1	10740	10028.7	1937	10055	2909	-	-
19	11415	7752.9	0	9348.9	1618	9929.8	7926	9995.2	16050	10423.5	2958	11231.2	8175	11309.8	14272
19	11651	7037.9	0	8523.2	1521	9250	7638	9309.5	15329	9893.6	3049	10968.9	8936	11240.9	15754
19	10587	6877.0	0	8312.2	1396	8959.1	7457	9037.3	14765	9599.7	2962	104449.2	8436	10587	12739
19	11843	6831.9	0	8680	1576	9370.4	8072	9460.8	16148	9963.6	2968	10959.2	8241	11195.1	14081
19	11006	7881.1	0	8967.5	1477	9565.6	7790	9635.7	15111	10134.6	3012	10959.8	8124	11006	10934
19	11256	7411.9	0	9286.3	1519	9957.8	7882	10042.3	15304	10470.1	2988	11256	7162	-	-
19	9622	7068.4	0	8691.4	1424	9251.9	7267	9346.1	15050	9605.8	2951	9622	3901	-	-
19	12386	8215.9	0	9583.2	1558	10135.9	7696	10205.3	15217	10767	2991	11635.2	8536	11875	15135
19	11030	7783.5	0	9137.3	1489	9810.4	7482	9894.6	15017	10677	2922	11030	4778	-	-
19	11282	6834.7	0	8908.3	1529	9526.3	7503	9602.2	14721	10080.3	2877	11071.3	7878	11146.3	13730
20	11627	7086.6	0	8806.1	2489	9488.1	12187	9579.2	24794	10153	4386	11178.6	12088	11443.6	21375
20	11148	7758.9	0	9172.9	2482	9689.1	12445	9757.8	24546	10379.9	4524	11069.1	12092	11148	18849
20	10287	6446.6	0	7827.4	2205	8418.5	10748	8485.8	21770	8923.5	4335	9941.3	11715	10174.6	20306
20	11660	6417.3	0	8370.9	2414	9123.8	12244	9211	24671	9938.7	4490	11078.5	12324	11372	21114
20	11132	7134.6	0	8359.4	2553	9030.7	12759	9114.6	24920	9826	4587	10670.9	12814	10928.8	22215
20	11446	7360.1	0	8725.4	2207	9361.7	10682	9426.5	22247	10115.2	4468	10964.1	13246	11190.1	24181
20	9811	6918.6	0	8136.6	2566	8783.7	12362	8869.3	25173	9543.2	4485	9811	9496	-	-
20	10787	6550.1	0	7954.18	2217	8527.5	11638	8603.5	23234	9166.3	4353	10144.6	11703	10403.9	20356
20	11890	6944.4	0	8454.1	2478	9082.2	12991	9168.5	24953	9776.3	4447	10711.1	11962	10943.9	20519
20	10961	6892.9	0	8393.5	2188	9130.5	11785	9195.8	24194	9669.4	4627	10581.5	14755	10829	26940

Table 5.3 AQAP: Comparison of NewRLT and RLT2 in terms of CPU execution time and the gap for AQAP instances of size $15 \leq n \leq 20$. Each row gives the average values for the respective 10 instances.

	RLT2						NewRLT2					
	Itr(100)		Itr(500)		Itr(1000)		Itr(5)		Itr(15)		Itr(25)	
size	Gap(%)	time	Gap(%)	time	Gap(%)	time	Gap(%)	time	Gap(%)	time	Gap(%)	time
15	8.45	298	3.49	1112	3.20	1868	3.25	151	0.00	250	0.00	265
16	10.00	617	4.29	2576	3.83	4761	3.68	254	0.22	482	0.02	557
17	14.51	696	6.50	3055	5.16	5653	5.32	396	0.52	885	0.00	1033
18	20.59	1042	12.98	4827	12.14	9450	8.28	1969	1.28	4870	0.51	6323
19	25.36	1510	17.10	7671	16.12	15271	10.43	2967	1.98	7416	1.48	11248
20	31.56	2379	22.20	11990	21.13	24050	12.90	4470	4.30	12219	2.03	20535

bounds and CPU execution times at different intervals; $\{100, 500, 1000\}$ for RLT2 and $\{5, 15, 25\}$ for NewRLT. In all tables, the first two columns give the problem sizes and the optimal objective values. From left to right, the third and forth columns give the lower bound and CPU time of CGMCC, followed by the lower bound values and CPU times of the RLT2 for three intervals in columns fifth to tenth, and NewRLT for three intervals in columns eleventh to sixteenth. For both the RLT2 and NewRLT, if the optimal objective values are verified in previous intervals, the current lower bounds and CPU times are shown by dash marks.

As we observe from these tables, it is clear that the bounds implied by GMCC are the weakest bounds for all 110 instances. The bounds obtained from the dual ascent strategies applied to RLT2 and NewRLT for 50 instances of sizes $10 - 14$ are exact, as they equal to the optimal objective values of the AQAP. However, the dual-ascent algorithm takes 254.2 seconds to calculate the NewRLT bounds of all these 50 instances while the dual-ascent algorithm takes 380 seconds to calculate the RLT2 bounds. Thus, the total average CPU time spent by NewRLT is 33% less than the total average CPU time spent by RLT2 to solve all instances of size $10 \leq n \leq 14$.

For the instances of size $n \geq 15$, the RLT2 with $Itr = 100$ provides lower bounds that are much stronger than those provided by GMCC, but need a high computational effort. The bounds of RLT2 become stronger as Itr grows, but at the cost of increased CPU time. From 60 instances with size $n \geq 15$, the total number of instances solved to optimality by RLT2 are 14 from which, 2 instances in at most 100 iterations, 8 instances in at most 500 iterations, and 4 instances in at most 1000 iterations. Lower bounds yielded by NewRLT for these instances outperform both the GMCC and RLT2 bounds. More precisely, the lower bounds obtained by NewRLT even with $Itr = 5$ are almost always stronger than those provided by RLT with $Itr = 1000$. Moreover, from the 60 instances, the NewRLT was able to solve 43 instances to optimality; 8 instances in at most 5 iterations, 24 instances in at most 15 iterations , and 9 instances in at most 25 iterations.

To make a better comparison in terms of CPU execution times and also bounds tightness, we report in Table 5.3 the average CPU time and also the average gap between the lower bounds and the optimal objective values. Each row of the table gives the average values of the respective 10 instances of size $15 \leq n \leq 20$. The formula we used to compute the gaps is $100 \times (Opt - Lb)/Lb$, where Opt and Lb stand for the optimal objective value and the lower bound, respectively. As we can observe, the bounds of RLT2 and NewRLT get stronger as Itr grows, but this progress for NewRLT is more significant. For problem sizes $n = 15, 16, 17$, the bounds of NewRLT with $Itr = 25$ are almost exact (only 0.02% of gap for instances of size $n = 16$) with the maximum CPU execution time of 1033 for $n = 17$, while there are modest gaps between the RLT2 bounds and the corresponding optimal objective values, i.e., the gap of $3.20\%, 3.83\%, 5.16\%$ for $n = 15, 16, 17$, respectively, with more computational effort. The significant reduced gaps of NewRLT for problem sizes $n = 18, 19, 20$, even with $Itr = 5, 15$ against the gaps of RLT2 indicate that NewRLT outperforms the RLT2.

5.6.2 Computational results for QAP on reducible graphs

In this section,we report the computational experiments of lower bounding approaches for QAP on reducible graphs. In the following subsection we present the generation of test instances. Then, we will discuss in detail the results of our lower bounding procedure.

Generation of instances

In this section we generate a class of random instances for the QAP on reducible graphs. In order to generate QAP instances whose flow and distance graphs are reducible, we first randomly generated flow, distance and Linear cost matrices F, D and C uniformly from the set $\{1, 2, \dots, 10\}$ (i.e., $f_{ij}, d_{ij}, c_{ij} \in \{1, 2, \dots, 10\}$). Then we transform G_f and G_d to reducible graphs $\overline{G_f} = (\overline{V_f}, \overline{E_f})$ and $\overline{G_d} = (\overline{V_d}, \overline{E_d})$ with $\overline{V_f} = V_f$, $\overline{V_d} = V_d$, $\overline{E_f} \subseteq E_f$, and $\overline{E_d} \subseteq E_d$. To obtain the reducible graphs $\overline{G_f}$ ($\overline{G_d}$) we removed $\overline{E_f}$ ($\overline{E_d}$) from E_f (E_d) such that no arcs in $E_f \setminus \overline{E_f}$ ($E_d \setminus \overline{E_d}$) can be added to $\overline{E_f}$ ($\overline{E_d}$) that the reducibility will be maintained. A trivial algorithm to find such reducible subgraphs has an $O(nm)$ complexity [85]. We generated randomly generated problems with size $n \in \{10, 15, 20\}$.

Lower bound Computation

Table 5.4 presents computational results for 30 randomly generated QAP instances. The computational times are obtained on an Intel Core i5-2410M CPU with 2.30 GHz and 6 GB RAM in single processor mode. All the algorithms are coded in C language. We compare four different lower bounding approaches in sense of tightness of the bounds

Table 5.4 Comparison of four different lower bonding approaches for QAP on reducible graphs.

	Gilmore-Lawler		Kaufman-Broeckx			RLT1 dual ascent			Lagrangian Dec.		
size	Lb	time	Lb	time	Imp. (%)	Lb	time	Imp. (%)	Lb	time	Imp. (%)
10	26	0.87	24.67	3.62	-5.11	35.00	44.58	34.61	27.36	0.90	52.30
10	23	0.87	23.02	3.74	0.08	37.60	42.56	63.47	26.69	0.91	16.04
10	19	0.86	22.78	3.61	19.89	34.59	40.25	82.05	28.91	0.91	52.15
10	20	0.85	21.24	3.50	6.20	33.00	40.26	65.00	27.05	0.95	35.25
10	25	0.88	23.02	3.67	-7.92	42.11	44.26	68.44	26.82	0.63	7.28
10	24	0.86	23.00	3.55	-4.16	35.00	41.72	45.83	27.58	0.81	14.91
10	20	0.89	20.34	3.47	1.70	41.76	44.08	108.8	25.03	0.80	25.15
10	20	0.84	19.94	3.59	-0.30	30.50	42.91	52.50	27.86	0.90	39.30
10	16	0.87	20.80	3.62	30.00	29.97	44.16	87.31	25.35	0.61	58.43
10	20	0.86	21.77	3.66	8.85	35.19	43.35	75.95	25.83	0.96	29.15
Ave.	**21.60**	**0.86**	**22.05**	**3.60**	**4.92**	**35.47**	**44.82**	**68.39**	**26.84**	**0.83**	**32.99**
15	26	3.76	27.24	8.08	4.76	29.55	138.26	13.65	30.74	1.75	18.23
15	26	3.71	27.48	8.29	5.69	28.87	140.98	11.03	32.07	1.71	23.34
15	26	3.76	26.25	8.08	0.96	30.79	141.88	18.42	29.34	1.45	12.84
15	26	3.88	26.52	8.34	2.00	29.87	142.22	14.88	28.80	1.65	10.76
15	26	3.45	26.16	8.22	0.61	31.43	142.15	20.88	27.49	1.54	5.73
15	26	3.55	26.06	8.29	0.23	29.38	142.38	13.00	28.57	1.46	9.88
15	26	3.45	26.45	8.07	1.73	29.66	143.60	14.07	30.53	1.51	17.42
15	26	3.63	26.57	8.32	2.19	31.77	144.92	22.19	30.10	1.38	15.76
15	26	3.41	26.91	8.42	3.50	30.79	146.27	18.42	31.51	1.38	21.19
15	26	3.56	26.08	8.40	0.30	31.08	146.22	19.53	29.84	1.33	14.76
Ave.	**26**	**3.61**	**26.57**	**8.25**	**2.19**	**30.31**	**142.88**	**16.60**	**29.89**	**1.51**	**14.99**
20	28	7.21	29.01	20.39	3.60	31.46	515.63	12.35	31.93	6.73	14.03
20	28	7.20	29.84	19.51	6.57	29.65	503.44	5.89	32.82	7.56	17.21
20	28	7.36	28.00	19.38	0.00	30.69	504.11	9.60	28.58	8.70	2.07
20	28	7.18	28.95	19.47	3.46	30.16	505.90	7.71	29.00	8.24	3.57
20	28	7.21	28.07	19.51	0.25	29.28	505.49	4.57	29.61	8.50	5.75
20	28	7.04	29.76	19.78	6.28	29.55	506.87	5.53	31.13	8.48	11.17
20	28	7.00	28.71	19.59	2.53	30.63	507.14	9.39	30.83	8.23	10.10
20	28	7.40	29.62	19.33	5.78	29.31	506.26	4.67	30.49	8.50	8.89
20	28	7.30	28.88	19.66	3.14	31.12	504.10	11.14	31.08	8.27	11.00
20	28	7.32	28.67	19.62	2.39	30.86	512.31	10.21	29.09	8.26	3.89
Ave.	**28**	**7.22**	**28.95**	**19.64**	**3.39**	**30.27**	**507.12**	**8.10**	**30.45**	**8.14**	**8.75**

and also computational performance. From the second to the twelfth, we present the lower bound values and CPU times obtained with the classical lower bounds of Gilmore and Lawler, the most recent version of the Kaufman-Broeckx lower bound of Zhang et al. [119], level-1 RLT dual-ascent bound by Hahn and Grant [59], and our Lagrangian decomposition approach. We have only reported results for instances with size up to 20 vertices because of excessive CPU time requirements in solving large sized level-1 RLT. Table is divided into three parts where we separately report results for problems with dimension $10, 15$ and 20.The last row of each part provides the average values of the corresponding columns.

Considering instances with dimension 10 and 15, the level-1 RLT dual-ascent bound by Hahn and Grant and the Kaufman-Broeckx type lower bound have the largest (68.39%), and lowest (4.92%) overall average percent improvement over the Gilmore and Lawler lower bound respectively, while for the instances with dimension 20 Lagrangian decomposition has the best overall average (8.75) percent improvement over the Gilmore and Lawler lower bound. From these results we can conclude that the dual ascent strategy RLT1 yields tighter lower bound at the expense of the increase in CPU time requirement, but the bound weakens as n increases, while the Lagrangian decomposition provides the best overall average lower bounds with a reasonable computational effort as n increases. Moreover Lagrangian decomposition outperforms Kaufman-Broeckx type approach in both computational time and bound tightness.

5.7 Conclusions

In this chapter we studied the lower bounding approaches for two special cases of the QAP including the Adjacent QAP and QAP on reducible graphs. As for the AQAP, we proposed two mixed binary linear formulations for the problem using reformulation linearization technique. To obtain the first formulation we apply a partial RLT to the AQAP, then using the special structure of the resulting mixed binary linear formulation, we extract a flow-based model. Applying the level-1 RLT for the flow-based model, we derived the second mixed binary linear program, NewRLT, whose linear relaxation possesses some desirable properties relative to the constraint set. To solve the continuous relaxation of the latter, we developed a dual-ascent algorithm similar to the one in [3]. We compared the NewRLT bounds with those from the classical level-2 RLT [3] in terms of the bounds strength and also CPU execution time. Our computational experiments indicate that the NewRLT outperforms the classical level-2 RLT, as it obtains lower bounds close to the optimal solutions for all instances with less computational effort. For the QAP on reducible graph we give a Lagrangian decomposition based on splitting the variables and then dualizing the copy constraint in Lagrangian fashion. We applied a polynomial algorithm to solve the quadratic semi assignment problems arise as subproblems of the Lagrangian decomposition.

Chapter 6

Quadratic Traveling Salesman Problem

Given a graph and a function that maps every consecutive pair of edges to a cost, the Quadratic Traveling Salesman Problem (QTSP) consists in finding a cycle that visits every vertex exactly once and such that the sum of the costs over all pairs of consecutive edges of the cycle is minimum. In this chapter we study both the Symmetric and the Asymmetric version of the problem and propose an extended Linear Programming formulation that contains a variable for each cycle in the graph. Since the number of cycles is exponential in the graph size, we solve our formulation via column generation.

6.1 Introduction

The Traveling Salesman Problem (TSP) is one of the most studied optimization problems. Given an undirected graph $G = (V, E)$ with costs $c_e, e \in E$, the problem consists in finding a cycle C that visits each vertex in V exactly once (i.e., an Hamiltonian), such that the sum of the costs c_e of each edge in C is minimum. In its most common form, the TSP has a linear cost function. Now, consider a variant of the TSP having a quadratic cost function, the so-called Quadratic TSP (QTSP). The input of this problem is an undirected (or directed) graph $G = (V, E)$ and a cost function $q : E \times E \to \mathbb{R}^+$ that maps every consecutive pair of edges (or arcs) to a non-negative cost. The QTSP consists in finding a cycle C of minimum overall cost that visits every vertex of G exactly once. This problem is NP-hard [42]. We distinguish between Symmetric QTSP (SQTSP) and Asymmetric QTSP (AQTSP), depending on the fact that the direction of the cycle matters.

The QTSP was introduced with an application to bioinformatics in [67] and also has application in robotics and telecommunications. In robotics, a variant of QTSP called *Angle-TSP* can be used for the optimization of robot paths with respect to energetic aspects. The Angle-TSP problem seeks to minimize the total angle of a TSP tour for a set of points in Euclidean space where the angle of a tour is the sum of the direction changes at all points [7]. The QTSP also can be viewed as a generalization of the Reload Cost TSP (RTSP) introduced in [11]. In the RTSP, one is given a graph whose every edge is assigned a color and there is a *reload* cost when passing through a node on two edges that have different colors.

The QTSP has been introduced quite recently and its literature is limited. In particular, the QTSP has been tackled in [42] with heuristic algorithms based on well-known heuristics for the TSP, with an ad-hoc branch-and-bound solver, and with a branch-and-cut approach based on a linearization of a 0-1 Quadratic Programming formulation of the problem. In [43] and [41] the polyhedral structure of a linearized integer programming formulation has been used to develop a branch-and-cut algorithm for the SQTSP and AQTSP respectively that are the current state-of-the-art for QTSP.

In this chapter we present an extended Linear Programming formulation of the QTSP with an exponential number of variables that is solved via Column Generation (CG). The basic idea is to have a variable for each cycle of G. This yields a pricing subproblem that consists in finding a cycle of minimum quadratic cost. We formulate the pricing subproblem as a 0-1 Quadratic Program, which is linearized and solved with standard techniques. We resort to stabilization techniques to overcome the tailing-off effect of CG approach.

6.2 The Symmetric QTSP

Consider a complete undirected graph G on a vertex set $V = \{1, 2, \dots, n\}$ and let $q : E \times E \rightarrow \mathbb{R}^+$ be a cost function that maps every consecutive pair of edges to nonnegative costs. The cost of a subgraph in G is equal to the sum of the costs of the pairs of edges incident in the same vertex. The SQTSP seeks a tour (i.e., a cycle passing through each vertex exactly once) of minimum cost. The problem can be formulated in a straightforward way by means of a quadratic 0-1 model. Let us define the binary variable x_{ij} that is equal to 1 if edge $\{i, j\}$ belongs to the minimum cost tour, and 0 otherwise. The model is:

$$
\begin{aligned}
\min \quad & \sum_{\{i,j\}\in E}\sum_{\{j,k\}\in E} q_{ijk}x_{ij}x_{jk} \\
\text{s.t.} \quad & \sum_{j:\{i,j\}\in E} x_{ij} = 2 \quad \forall i \in V && (6.1)\\
& \sum_{\substack{\{i,j\}\in E:\\ i\in S, j\in S}} x_{ij} \le |S| - 1 \quad \emptyset \neq S \subsetneq V && (6.2)\\
& x_{ij} \in \{0,1\} \quad \forall \{i,j\} \in E.
\end{aligned}
$$

The objective function considers all costs between edges incident on the same vertices. Constraints (6.1) ensure that each vertex has degree two in the tour, and constraints (6.2) ensure that no subtour is formed among the subsets of vertices.

We can linearize all products $x_{ij}x_{jk}$ using the standard linearization proposed in [57, 62]. We introduce binary variables u_{ijk} and link them to the original variables using the following additional linear inequalities:

$$u_{ijk} \le x_{ij}, \quad u_{ijk} \le x_{jk}, \text{ and } u_{ijk} \ge x_{ij} + x_{jk} - 1. \tag{6.3}$$

6.2.1 MILP representation for the SQTSP

In this section we develop a linearized formulation to the SQTSP which yields a tighter continuous relaxation than the continuous relaxation of the standard linearization seen in (6.3). The new MILP formulation is obtained by introducing a new graph $\tilde{G} = (\tilde{V}, \tilde{E})$, called the *Gadget graph* constructed as follows:

- For each edge in the graph G, create two nodes $\langle i,j\rangle$ and $\langle j,i\rangle \in \tilde{V}$.
- For each node $i \in V$ in G, define a super node $S_i = \{\langle i,j\rangle : j = 1,2,\ldots,n, j \neq i\}$. This is only an aggregation of nodes of $\tilde{V}$.
- For any $i,j,k \in V$, create an edge between nodes $\langle i,j\rangle$ and $\langle i,k\rangle$ in the super node S_i with weight q_{jik}.
- For any $i,j \in V$, create an edge between nodes $\langle i,j\rangle$ and $\langle j,i\rangle$ in super nodes S_i and S_j respectively with weight zero.

In Figure 6.1 we present an example of a graph G and the corresponding *Gadget* graph $\tilde{G}$.

Defining decision variables u_{jik} to indicate whether edge $\{\langle i,j\rangle, \langle i,k\rangle\}$ in super node S_i is contained in the solution, we can rewrite the SQTSP on graph G as the following

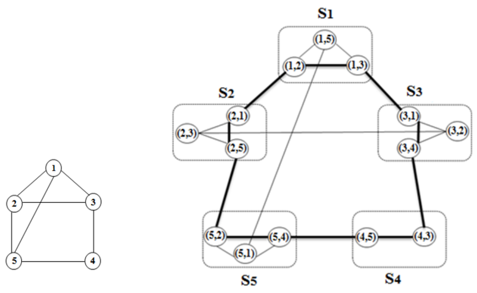


Fig. 6.1 Graph G and its corresponding *Gadget* graph $\tilde{G}$. Note that we assume G to be complete, but for simplicity this example shows the connections in the gadget graph when G is not complete, or when some of its edges have infinite cost

integer linear problem on graph $\tilde{G}$:

$$\text{Gadget:} \quad \min \sum_{i \in V} \sum_{\substack{j,k \in V: \\ \{\langle i,j\rangle,\langle i,k\rangle\} \in S_i}} q_{jik} u_{jik} \tag{6.4}$$

$$\text{s.t.} \quad \sum_{\substack{j,k \in V: \\ \{\langle i,j\rangle,\langle i,k\rangle\} \in S_i}} u_{jik} = 1 \quad \forall i \in V \tag{6.5}$$

$$\sum_{\substack{\langle i,k\rangle \in S_i: \\ k \neq j}} u_{jik} - \sum_{\substack{\langle j,k\rangle \in S_j: \\ k \neq i}} u_{ijk} = 0 \quad \forall \langle i,j\rangle \in \tilde{V} \tag{6.6}$$

$$\sum_{i \in S} \sum_{\substack{\{\langle i,j\rangle,\langle i,k\rangle\} \in \tilde{E}: \\ j,k \in S}} u_{jik} \leq |S| - 1 \quad S \subsetneq V \tag{6.7}$$

$$u_{jik} \in \{0,1\} \quad \forall \{\langle i,j\rangle,\langle i,k\rangle\} \in \tilde{E}. \tag{6.8}$$

Constraints (6.5) guarantee that within every super node we select exactly one edge, constraints (6.6) indicate that the number of the edges incident on node $\langle i,j\rangle$ is equal to the number of incident edges on node $\langle j,i\rangle$, and constraints (6.7) are the subtour elimination constraints. Note that in terms of feasible solutions, in the *Gadget graph* a tour is called *feasible* if it is simple and passes through each super node S_i by visiting exactly one edge of S_i. In the example shown in Figure 6.1, the bold lines define a feasible tour (a subtour of $\tilde{G}$) corresponding to the feasible tour $\{(1,2),(2,5),(5,4),(4,3),(3,1)\}$ in the original graph G.

An alternative model in the case of reload costs spanning tree was studied in [52], where it is assumed that the triangle inequality holds for reload costs at each node of the graph.

6.3 The Asymmetric QTSP

Consider a complete directed graph G with vertex set $V = \{1, 2, \dots, n\}$, and cost function $q : A \times A \to \mathbb{R}^+$ that maps every consecutive pair of arcs to a non negative integer cost. The AQTSP seeks a directed tour (i.e., a directed cycle passing through each vertex exactly once) of minimum cost and is formulated as follows:

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} \sum_{(j,k)\in A} q_{ijk} x_{ij} x_{jk} \\
\text{s.t.} \quad & \sum_{(i,j)\in A} x_{ij} = 1 \quad \forall i \in V && (6.9)\\
& \sum_{(i,j)\in A} x_{ij} = 1 \quad \forall j \in V && (6.10)\\
& \sum_{i\in S, j\notin S : (i,j)\in A} x_{ij} \geq 1 \quad S \subsetneq V && (6.11)\\
& x_{ij} \in \{0,1\} \quad \forall (i,j) \in A.
\end{aligned}
$$

The binary variable x_{ij} is equal to 1 if the arc (i,j) belongs to the minimum cost tour. Constraints (6.9) and (6.10) force to select a single outgoing arc and a single incoming arc for each node, respectively, and constraints (6.11) are the well known subtour elimination constraints.

A relaxation of this problem has been studied in [50] where the authors were looking for a *minimum spanning arborescence* with changeover costs, that are a particular type of quadratic costs. The 0-1 formulation of the minimum spanning arborescence, which could be used for lower bounding the ATSP, does not have constraints (6.10) and fixes a given vertex as a root.

6.3.1 MILP formulation for the AQTSP

In order to linearize the AQTSP, we follow the idea of constructing an auxiliary graph as in the SQTSP. We call this auxiliary graph the *extended graph* and denote it as $\overline{G} = (\overline{V}, \overline{A})$. For each arc (i,j) in G we create a node $\langle i,j \rangle$, and for each triple i, j, k, we introduce arc $(\langle i,j \rangle, \langle j,k \rangle)$ with weight q_{ijk}. If we partition the set of nodes of $\overline{G}$ into n clusters $\mathcal{V}_1, \mathcal{V}_2, \dots, \mathcal{V}_n$ such that $\mathcal{V}_i = \{\langle i,j \rangle : j = 1, 2, \dots, n, j \neq i\}$ for $i = 1, 2, \dots, n$, then all arcs are defined between nodes $\langle i,j \rangle$ and $\langle j,k \rangle$ from different clusters such that $q_{ijk} > 0$; therefore there are no intra-set arcs.

Proposition 6.3.1. *Any feasible tour in G corresponds to a tour in $\overline{G}$ that goes through each cluster exactly once.*

Figure 6.2 represents the extended graph, $\overline{K}_4$, of a directed complete graph, K_4. The bold lines illustrate a feasible tour.

Definition 6.3.1. *Given a directed graph $G = (V, A)$, a cost function $c : A \to \mathbb{R}^+$, and a partition $\{\mathcal{V}_i : i = 1, \dots, k\}$ of V such that $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset$ for $i = j = 1, 2, \dots, k$, $i \neq j$ and $\bigcup_{i=1}^{k} \mathcal{V}_i = V$, the* Asymmetric Generalized *TSP (AGTSP) can be stated as the problem of finding a feasible cycle $T \subset A$ which includes exactly one node from each cluster $\mathcal{V}_i$, $i = 1, \dots, k$, and whose global cost $\sum_{e \in T} c_e$ is minimum. Therefore the AGTSP involves two related decisions: (i) choosing a node subset $S \subsetneq V$ such that $|S \cap \mathcal{V}_i| = 1$ for all $i = 1, 2, \dots, n$ and (ii) finding a minimum cost Hamiltonian cycle in the subgraph of G induced by S.*

Corollary 6.3.2. *Solving the AQTSP on graph G is equivalent to solving the AGTSP on $\overline{G}$.*

Using Corollary 6.3.2, instead of solving the original AQTSP one can solve an AGTSP on graph $\overline{G}$ which is again NP-hard, as it can be reduced to an Asymmetric TSP [77, 92].

Now we turn our attention to the integer linear programming formulation for the AGTSP on $\overline{G}$. Defining variables u_{ijk} indicate whether arc $(\langle i,j\rangle, \langle j,k\rangle)$ is selected or not in the solution, and variables y_{ij} indicate whether node $\langle i,j\rangle$ is visited or not, the problem is:

$$
\begin{aligned}
\text{GTSP1:} \quad \min \quad & \sum_{(\langle i,j\rangle,\langle j,k\rangle)\in\overline{A}} q_{ijk} u_{ijk} \\
\text{s.t.} \quad & \sum_{\substack{j\in V:\\ \langle i,j\rangle\in\mathcal{V}_i}} y_{ij} = 1 && \forall i \in V && (6.12)\\
& \sum_{\substack{k\in V:\\ \langle j,k\rangle\in\overline{V}}} u_{ijk} = y_{ij} && \forall \langle i,j\rangle \in \overline{V} && (6.13)\\
& \sum_{\substack{k\in V:\\ \langle k,i\rangle\in\overline{V}}} u_{kij} = y_{ij} && \forall \langle i,j\rangle \in \overline{V} && (6.14)\\
& \sum_{i\in S} \sum_{\substack{j\notin S:\\ \langle i,j\rangle\in\mathcal{V}_i}} \sum_{\substack{k\in V:\\ \langle j,k\rangle\in\mathcal{V}_j}} u_{ijk} \geq 1 && S \subsetneq V && (6.15)\\
& u_{ijk} \in \{0,1\} && \forall(\langle i,j\rangle, \langle j,k\rangle) \in \overline{A} && (6.16)\\
& y_{ij} \in \{0,1\} && \forall \langle i,j\rangle \in \overline{V}.
\end{aligned}
$$

Constraints (6.12) guarantee that we select exactly one node from each cluster. Constraints (6.13) and (6.14) require a solution to include exactly one of the arcs entering and

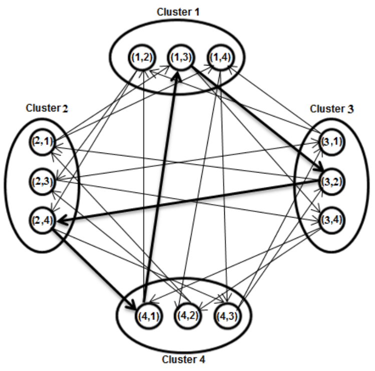


Fig. 6.2 Extended graph $\overline{K}_4$ of the complete graph K_4

exactly one of the arcs leaving the node $\langle i, j\rangle$ if this node is visited. Finally, constraints (6.15) eliminate all subtours.

Note that the integrality of variables u_{ijk} for all $(\langle i, j\rangle, \langle j, k\rangle) \in \overline{A}$ is implied by the other constraints, thus it can be relaxed.

By eliminating variables y_{ij} we can write the problem as follows:

$$\text{GTSP2:} \quad \min \sum_{(\langle i,j\rangle,\langle j,k\rangle)\in\overline{A}} q_{ijk}u_{ijk} \tag{6.17}$$

$$\text{s.t.} \quad \sum_{j\in V} \sum_{\substack{k\in V:\\ (\langle i,j\rangle,\langle j,k\rangle)\in\overline{A}}} u_{ijk} = 1 \qquad \forall i \in V \tag{6.18}$$

$$\sum_{k\in V} \sum_{\substack{j\in V:\\ (\langle k,i\rangle,\langle i,j\rangle)\in\overline{A}}} u_{kij} = 1 \qquad \forall i \in V \tag{6.19}$$

$$\sum_{\substack{k\in V:\\ \langle j,k\rangle\in\overline{V}}} u_{ijk} - \sum_{\substack{k\in V:\\ \langle k,i\rangle\in\overline{V}}} u_{kij} = 0 \qquad \forall \langle i, j\rangle \in \overline{V} \tag{6.20}$$

$$(6.15), (6.16). \tag{6.21}$$

Constraints (6.18) require a solution to include exactly one of the arcs entering a cluster, while constraints (6.19) require a solution to include exactly one of the arcs leaving a cluster. Constraint (6.20) is equivalent to network flow conservation constraints and ensure that a solution tour is uninterrupted and continuous.

Theorem 6.3.3. *Constraints* (6.18) *in problem GTSP2 are redundant and can be removed from the model.*

Proof. Suppose u^* is a feasible solution for problem GTSP2 after removing constraint

(6.18) and define, for each $i \in V$,

$$\epsilon_i = \sum_{j \in V} \sum_{\substack{k \in V: \\ (\langle i,j \rangle, \langle j,k \rangle) \in \overline{A}}} u^*_{ijk}.$$

We show that $\epsilon_i = 1$ for all $i \in V$.

Consider cluster $\mathcal{V}_s$. Then constraint (6.19) is satisfied for cluster $\mathcal{V}_s$ by u^*, i.e.,

$$\sum_{k \in V} \sum_{\substack{j \in V \\ (\langle k,s \rangle, \langle s,j \rangle) \in \overline{A}}} u^*_{ksj} = 1.$$

Therefore there exists a node $\langle s, t \rangle \in \mathcal{V}_s$ with in-degree one in the tour, while the in-degree of all other nodes in cluster $\mathcal{V}_s$ is zero. Hence by (6.20) the out-degree of node $\langle s, t \rangle$ must be equal to one while the out-degree of all the other nodes in the same cluster must be zero. □

As a consequence of Theorem 7.2.1 GTSP2 simplifies as follows:

$$\begin{aligned} \text{GTSP3:} \quad \min \quad & \sum_{(\langle i,j \rangle, \langle j,k \rangle) \in \overline{A}} q_{ijk} u_{ijk} \\ \text{s.t.} \quad & (6.19), (6.20), (6.21). \end{aligned}$$

6.4 Cycle Reformulation of the General QTSP

In this section we present a new formulation of the QTSP which is based on a cycle generation approach in the given graph. Let C be a cycle of G represented by the set of edges (arcs) that appear in the cycle. The cost of cycle C is the sum of the costs of the pairs of consecutive edges (arcs) contained in the cycle, i.e.

$$q(C) = \sum_{i,j,k \in V : (i,j),(j,k) \in C} q_{ijk}.$$

Let $\mathcal{C}$ and $\mathcal{T}$ denote the collection of all cycles and all tours of G, respectively. Clearly, we have that $\mathcal{T} \subseteq \mathcal{C}$, and hence

$$\min_{C \in \mathcal{C}} q(C) \leq \min_{T \in \mathcal{T}} q(T).$$

Following the approach of Held and Karp to the TSP [63], we associate a penalty π_i with every vertex i in V, and define $\pi(C) = \sum_{i \in C} \pi_i$. Let us consider a new cost function defined as follows: $d(C) = q(C) + \pi(C)$. Let T^* denote an optimal tour of G, i.e. $q(T^*) = \min_{T \in \mathcal{T}} q(T)$. Then, the following relations hold:

$$\min_{C \in \mathcal{C}} d(C) = \min_{C \in \mathcal{C}} \{q(C) + \sum_{i \in C} \pi_i\} \leq d(T^*) = q(T^*) + \sum_{i \in V} \pi_i$$

$$\min_{C \in \mathcal{C}} \{q(C) - \sum_{i \in V \setminus C} \pi_i\} \leq q(T^*).$$

For any vector of penalty terms π we get a lower bound. However, we are interested in finding π that maximizes the lower bound:

$$\max_{\pi \in \mathbb{R}^n} \min_{C \in \mathcal{C}} \{q(C) - \sum_{i \in V \setminus C} \pi_i\}. \tag{6.22}$$

We describe an LP equivalent to (6.22) by introducing a variable z as follows:

$$\begin{aligned} \text{P: max} \quad & z \\ \text{s.t.} \quad & z + \sum_{i \in V \setminus C} \pi_i \leq q(C) & \forall C \in \mathcal{C} \qquad (6.23) \\ & z, \pi \text{ unrestricted.} \end{aligned}$$

Let λ_C be the dual multipliers of constraints (6.23). We can write the dual of problem P as follows:

$$\begin{aligned} \text{D1: min} \quad & \sum_{C \in \mathcal{C}} q(C) \lambda_C & & (6.24) \\ \text{s.t.} \quad & \sum_{C \in \mathcal{C}: i \notin C} \lambda_C = 0 & \forall i \in V \quad & (6.25) \\ & \sum_{C \in \mathcal{C}} \lambda_C = 1 & & (6.26) \\ & \lambda_C \geq 0 & \forall C \in \mathcal{C}. \quad & (6.27) \end{aligned}$$

Since all multipliers in (6.25) are non-negative, (and, in each iteration of a column generation approach, exactly one column is generated as we explain in Section 6.5), an optimal solution to the problem must satisfy $\lambda^*_C = 1$ for some $C^* \in \mathcal{C}$ and $\lambda^*_C = 0$ for all $C \in \mathcal{C} \setminus C^*$. It follows that the cycle C^* (single or multiple) is optimal and $q(C^*)$ provides a lower bound for the original QTSP.

By subtracting each constraint (6.25) from (6.26) and removing (6.26) from D1, one can find a relaxation of the problem D1 as follows:

$$\begin{aligned} \text{D2:} \min \quad & \sum_{C \in \mathcal{C}} q(C)\lambda_C \\ \text{s.t.} \quad & \sum_{C \in \mathcal{C}: i \in C} \lambda_C = 1 && \forall i \in V \qquad (6.28) \\ & \lambda_C \geq 0 && \forall C \in \mathcal{C}. \end{aligned}$$

Problem D2 seeks a minimum-weight "combination of cycles" such that each vertex appears, on average, in one cycle.

6.5 A Column Generation Approach

In this section we develop a column generation approach to solve problems D1 and D2. Since the number of cycles in $\mathcal{C}$ is exponential with respect to the number of vertices, we first consider a restricted version of the problem with a feasible subset of cycles, $\bar{\mathcal{C}} \subseteq \mathcal{C}$. Note that the subset $\bar{\mathcal{C}}$ for problem D1 must include at least one tour, while an initial set $\bar{\mathcal{C}}$ for problem D2 must satisfy constraints (6.28). Let us first start with problem D1 and suppose that $\pi = (\pi_1, \pi_2, \ldots, \pi_n)$ and z are the dual variables corresponding to constraints (6.25) and (6.26) respectively. The reduced cost of the variable λ_C for each $C \in \mathcal{C}$ is

$$\overline{q(C)} = q(C) - (\pi(C))' - z$$

where $(\pi(C))' = \sum_{i \notin C} \pi_i$.

A column entering the basis can be found by computing a minimum cost cycle with respect to $q_{ijk} + \pi_j$ for each $(i,j) \in E, (j,k) \in E$. Let $\overline{q(C^p)} = \min_{C \in \mathcal{C}} \overline{q(C)}$. If $\overline{q(C^p)} \geq 0$ then the current solution is optimal. Otherwise we select column C^p to enter the basis.

Theorem 6.5.1. *If the column C^p entering the basis corresponds to a tour, then it is an optimal tour.*

Proof. Consider any cycle C. Since column C^p is the selected column to enter the basis we have

$$\overline{q(C^p)} = q(C^p) - (\pi(C^p))' - z \leq q(C) - (\pi(C))' - z.$$

If cycle C is also a tour, then $q(C^p) \leq q(C)$. □

Note that for problem D2, the reduced cost of the variable λ_C for each $C \in \mathcal{C}$ is modified to:

$$\overline{q(C)} = q(C) - \pi(C).$$

6.5.1 Pricing subproblems

A column entering the basis can be found by computing a minimum cost cycle in the original graph G with respect to $q_{ijk} + \pi_j$. Looking for a cycle having minimum negative cost with respect to a quadratic objective is itself an interesting combinatorial optimization problem, which is NP-hard [7, 51]. In this section we explain how to update the linearized models, presented in Sections 6.2 and 6.3, to solve the pricing problems. In order to define a suitable model for the pricing subproblem of the restricted master problem D1, we consider the SQTSP and AQTSP separately.

Symmetric case

Consider the pricing problem of D1 in the symmetric case. As we mention in Section 6.2, instead of looking for a cycle in the original graph one can easily find a cycle in $\hat{G}$; i.e., finding a negative reduced cost of problem D1 is the same as finding a negative cost feasible cycle in $\hat{G}$. Defining a binary variable w_i to indicate whether the super node S_i is on the cycle or not, the minimum negative cost cycle is then found by solving the following problem:

$$\begin{aligned}
\min \quad & \sum_{i \in V} \sum_{\{\langle i,j\rangle,\langle i,k\rangle\} \in S_i} q_{jik} u_{jik} - \sum_{i \in V} \pi_i (1 - w_i) - z && \\
\text{s.t.} \quad & \sum_{\substack{j,k \in V: \\ \{\langle i,j\rangle,\langle i,k\rangle\} \in S_i}} u_{jik} = w_i && i \in V \qquad (6.29) \\
& (6.6) - (6.8) && \\
& w_i \in \{0,1\} && \forall i \in V.
\end{aligned}$$

Considering the definition of the w_i variables, one can obtain an equivalent formulation for the pricing problem by replacing constraint (6.29) with the so-called resource constraint (6.30) and removing variables w_i from the model.

$$\sum_{\substack{j,k \in V \\ \{\langle i,j\rangle,\langle i,k\rangle\} \in S_i}} u_{jik} \le 1, \qquad (6.30)$$

It should be observed that by forcing the solution of the pricing problem to be a cycle with negative cost one can easily remove constraints (6.7) from the pricing model. Therefore, finding a cycle with negative reduced cost is simply a matter of finding a path between each node $\langle i, j\rangle$ in $\hat{G}$ and itself that has a negative cost and satisfies resource constraint (6.30). Since dual variables π are defined on each super node of $\hat{G}$, the problem of finding a negative reduced cost cycle can be formulated as resource-constrained elementary shortest path problem. Let $\bar{q}_{st}$ denote the cost of the resource-constrained elementary shortest path from origin node $\langle s,t\rangle$ to itself in $\hat{G}$. The pricing problem can be written as: $\min_{\langle s,t\rangle\in\hat{V}}\{\bar{q}_{st}\}$, where $\bar{q}_{st}$ is the optimal value of the following problem.

$$\begin{aligned}
\bar{q}_{st} = \min \quad & \sum_{i\in V}\sum_{\{\langle i,j\rangle,\langle i,k\rangle\}\in S_i} (q_{jik}+\pi_i)u_{jik} - \sum_{i\in V}\pi_i - z \\
\text{s.t.} \quad & \sum_{\substack{j\in V:\\ \{\langle s,t\rangle,\langle s,j\rangle\}\in S_i}} u_{tsj} = 1 && (6.31)\\
& \sum_{\substack{j\in V:\\ \{\langle s,j\rangle,\langle s,t\rangle\}\in S_i}} u_{jst} = 1 && (6.32)\\
& \sum_{\substack{\langle i,k\rangle\in S_i:\\ k\neq j}} u_{jik} - \sum_{\substack{\langle j,k\rangle\in S_j:\\ k\neq i}} u_{ijk} = 0 \quad \forall\langle i,j\rangle\neq\langle s,t\rangle && (6.33)\\
& \sum_{\substack{j,k\in V\\ \{\langle i,j\rangle,\langle i,k\rangle\}\in S_i}} u_{jik} \le 1 \qquad i\in V && (6.34)\\
& u_{jik}\in\{0,1\} \quad \forall\{\langle i,j\rangle,\langle i,k\rangle\}\in\hat{E}. && (6.35)
\end{aligned}$$

Constraints (6.31), (6.32), (6.33) and (6.35) find a path from the source node $\langle s,t\rangle$ to itself. The resource constraint (6.34) guarantee that each super node S_i is visited at most once.

Asymmetric case

In the asymmetric case, finding a negative reduced cost directed cycle for problem D1 is equivalent to solving the modified version of the GTSP1 or GTSP3 explained in Section 6.3. Here we provide a version of the latter so that the resulting problem gives the most negative directed cycle.

$$\begin{aligned}
\min \quad & \sum_{(\langle i,j\rangle,\langle j,k\rangle)\in\overline{A}} q_{ijk}u_{ijk} - \sum_{i\in V}\pi_i(1-t_i) - z \\
\text{s.t.} \quad & \sum_{k\in V}\sum_{\substack{j\in V\\ (\langle k,i\rangle,\langle i,j\rangle)\in\overline{A}}} u_{kij} = t_i && \forall i\in V\\
& (6.16), (6.20), (6.19) \\
& t_i\in\{0,1\} && \forall i\in V.
\end{aligned}$$

Binary variable t_i is equal to 1 if cluster $\mathcal{V}_i$ is visited, otherwise it is zero. Following the same process as in the symmetric case, one can obtain an equivalent formulation based on resource-constrained elementary shortest path problem in the extended graph.

The solution of the subproblems provides either a certificate of optimality of the current solutions (λ, π, z) or a new column C^p that will be added to the master problem. It is worth pointing out that solving the pricing subproblem to optimality is only needed to prove optimality of the current primal and dual solutions; one can stop solving the subproblem whenever a negative reduced cost column is found [16]. This happens because adding this column to $\bar{\mathcal{C}}$ ensures that the new dual solution (π, z) will be different, and therefore the termination of the algorithm.

6.5.2 The Stabilized version of the column generation

Column generation methods usually suffer from slow convergence to the optimal solution and tailing off effects. Primal degeneracy, dual degeneracy and instability in the behavior of dual variables are well known to be the main causes of this behavior [55, 70].

To control the dual variables during the solution process, we use the stabilized column generation approach proposed in [37]. This approach combines the box step method [86] with a kind of descent method proposed in [75]. The box step method introduces a box around the previous dual vector and modifies the master problem such that the feasible dual space is limited to the area defined by these boxes, while the latter tries to adapt the master problem so that the distance separating a dual solution from the previous optimal dual solution is linearly penalized.

In order to present the idea, let us rewrite the restricted version of the master problem D1, for $\bar{\mathcal{C}} \in \mathcal{C}$, as the following model:

$$
\begin{aligned}
\text{RD1: min} \quad & \sum_{C \in \bar{\mathcal{C}}} q(C)\lambda_C && && (6.36)\\
\text{s.t.} \quad & \sum_{C \in \bar{\mathcal{C}} : i \in C} \lambda_C \geq 1 && \forall i \in V && (6.37)\\
& (6.26), (6.27).
\end{aligned}
$$

Note that since the set partitioning constraints admit negative dual values which can be problematic for the sub-problem, we used a relaxed version of the problem as the first step of the stabilization approach.

Consider the dual variables π associated with the constraints (6.37) and bound each π_i in the interval $[\delta_i^-, \delta_i^+]$. These bounds are first given as parameters to the model and then automatically updated during the process. The dual variable π_i can take values outside the given bounds, but the dual objective is then penalized by $\varepsilon_i^-(\delta_i^- - \pi_i)$ if $\pi_i < \delta_i^-$ and by $\varepsilon_i^+(\delta_i^+ - \pi_i)$ if $\pi_i > \delta_i^+$. The dual of the problem RD1 then becomes:

$$
\begin{aligned}
\text{SP: max} \quad & z + \sum_{i\in V} \pi_i - \varepsilon_i^- w_i^- - \varepsilon_i^+ w_i^+ & \\
\text{s.t.} \quad & z + \sum_{i\in C} \pi_i \leq q(C) & \forall C \in \bar{\mathcal{C}} \\
& \pi_i + w_i^- \geq \delta_i^- & \forall i \in V \\
& \pi_i - w_i^+ \leq \delta_i^+ & \forall i \in V \\
& \pi, w^-, w^+ \geq 0,\ z \text{ unrestricted.} &
\end{aligned}
$$

The primal of the stabilized restricted master problem, and hence the dual of SP, is:

$$
\begin{aligned}
\text{SD1: min} \quad & \sum_{C\in\bar{\mathcal{C}}} q(C)\lambda_C + \sum_{i\in V} -\delta_i^- \mu_i^- + \delta_i^+ \mu_i^+ & \\
\text{s.t.} \quad & \sum_{C\in\bar{\mathcal{C}}: i\in C} \lambda_C - \mu_i^- + \mu_i^+ \geq 1 & \forall i \in V \\
& \sum_{C\in\bar{\mathcal{C}}} \lambda_C = 1 & (6.38) \\
& \mu_i^- \leq \varepsilon_i^- & \forall i \in V \\
& \mu_i^+ \leq \varepsilon_i^+ & \forall i \in V \\
& \lambda, \mu^-, \mu^+ \geq 0. &
\end{aligned}
$$

This method is referred to as BoxPen stabilization since the bounds (δ^-, δ^+) on the original dual variables π can be represented by a bounding box containing the current dual solution. Note that the stabilized version of the problem D2 is the same as SD1 without the convexity constraint (6.38) and is called SD2. In order to use the stabilized models efficiently, one must initialize and update the parameters correctly. In order to reduce the dual variables' variations, we select $[\delta^-, \delta^+]$ to form a small box containing the current dual solution, and solve the problem SD1 (SD2). At the first iteration, when no solution is available to the problem, the dual variables π can be simply estimated. If the new π lies in the box $[\delta^-, \delta^+]$, reduce its width and augment the penalty given by ε^- and ε^+. Otherwise, enlarge the box and decrease the penalty. The update could be performed in each iteration, or alternatively, each time a dual solution of currently best value is obtained.

6.6 Computational Experiments

In this section we present our computational experiments on a class of randomly generated instances with size ranging from $n = 5$ to $n = 25$ introduced in [42]. All instances consist of complete graphs with n vertices and $m = n(n-1)/2$ edges. The cost function $q : E \times E \to \mathbb{R}_0^+$ for both symmetric and asymmetric instances is defined as an integer number which is chosen from the set $\{0, 1, ..., 10000\}$ uniformly for all $(i, j), (j, k) \in E$ such that $i \neq k$ and set to infinity for all $(i, j), (j, i) \in E$. We used the AMPL modeling language [46] with GUROBI 5.0.0 [58] as linear solver for the RMP and as a mixed integer linear solver for the pricing problem on an Intel Core i5-2410M CPU with 2.30 GHz and 6 GB RAM in single processor mode.

6.6.1 Stabilization

We implemented the stabilized column generation approach using different sets of initial values. In the following we present the results of some preliminary experiments whose purpose was to initialize and update the parameters for both SD1 and SD2.

For the problem SD1, we initialized δ^- and δ^+ at -1000 and 1000 respectively. The vector parameter ε^- and ε^+ were selected as -5 and 5 respectively and were kept fixed throughout the solution process. We updated the parameter (δ^-, δ^+) from $(-1000, 1000)$ to $(\tilde{\pi} - 100, \tilde{\pi} + 100)$, where $\tilde{\pi}$ is the current dual solution, only if the column returned by the subproblem had a non-negative reduced cost and $(\mu^-, \mu^+) \neq (0, 0)$. The stopping criteria of the stabilized column generation algorithm is $\overline{r(C)} \geq 0$ and $(\mu^-, \mu^+) = (0, 0)$.

In order to find potentially good initial values of (δ^-, δ^+) for problem SD2, we first solved the problem D2 with a feasible subset of cycles. By using the dual variables $\tilde{\pi}$ of problem D2, we initialized (δ^-, δ^+) with $(\tilde{\pi} - 10, \tilde{\pi} + 10)$. The vector parameters ε^- and ε^+ were initially set to 0.0001. If the subproblem is able to find a negative reduced cost cycle, then the values of ε^- and ε^+ were increased by 10%. However, when there was no more such column and $(\mu^-, \mu^+) \neq (0, 0)$, the values of ε^- and ε^+ were decreased by dividing each one by 100 and the parameters (δ^-, δ^+) were updated to $(\tilde{\pi} - 100, \tilde{\pi} + 100)$, where $\tilde{\pi}$ is the current dual solution of the SD2. The stopping criterion of the stabilized column generation algorithm is the same as that of problem SD1.

In order to show the effectiveness of stabilization, we compare the computational time of column generation to its stabilized version on two instances of different dimensions in Table 6.1. Each row of the table reports the average computational time over ten instances

Table 6.1 Computational time of Column Generation (CG) and Stabilized CG approaches for both The SQTSP and AQTSP instances

	CPU time (Symmetric)				CPU time (Asymmetric)			
size	CG2	SCG2	CG1	SCG1	CG2	SCG2	CG1	SCG1
10	2.15	1.91	4.36	4.12	3.91	3.26	8.93	7.35
20	43.57	33.34	205.94	60.70	97.43	34.92	508.53	271.81

Table 6.2 Comparison of three different lower bounding approaches for symmetric QTSP instances.

			$SCG2$				$SCG1$			
$size$	$Opt.$	$LB(LP)$	LB	$time$	$iter$	Gap	LB	$time$	$iter$	Gap
5	14922.2	13839.5	13606.3	0.62	6	1.21	14922.2	3.89	65	0.0
6	12217.5	11817.1	11816.2	0.53	7	0.99	12160.4	3.35	43	0.14
7	14648.6	13200.4	13747.9	0.85	11	0.62	14356.5	3.51	41	0.20
8	15055.7	12544.1	12623.9	1.12	14	0.96	14542.6	4.06	43	0.20
9	14562.2	11909.4	12710.7	1.91	16	0.73	14225.7	4.12	37	0.12
10	15018.6	11939.9	13104.5	1.77	18	0.62	14718.7	4.86	45	0.09
11	13819.6	10367.8	11175.9	1.75	17	0.76	12958.9	6.37	33	0.24
12	14719.4	10896.2	12036.6	3.15	21	0.70	13740.2	7.96	31	0.25
13	13174.3	9826.6	10954.4	4.90	23	0.66	12705.5	12.11	32	0.14
14	13548.7	9823.7	11089.7	9.34	25	0.66	12974.4	13.83	32	0.15
15	12531.0	9354.7	10697.8	12.38	28	0.57	12219.3	14.20	25	0.09
16	13426.1	9065.7	10253.3	13.24	26	0.72	12573.4	20.56	28	0.19
17	13022.5	9324.8	10420.6	18.55	30	0.70	12735.9	27.35	32	0.07
18	12388.6	8522.8	9831.7	22.35	34	0.66	12137.6	28.38	29	0.06
19	12697.5	8630.6	10036.8	29.98	39	0.65	12394.9	45.92	34	0.07
20	13246.0	8797.3	10092.8	33.34	40	0.70	12491.8	60.70	35	0.16
21	12699.4	8352.0	9868.1	45.71	45	0.65	12260.4	75.16	34	0.10
22	12032.2	8078.2	9519.4	50.81	47	0.63	11859.1	98.45	32	0.04
23	12378.4	8123.9	9376.2	53.98	46	0.70	11911.7	154.01	37	0.10
24	11871.1	7996.5	9250.5	65.78	51	0.67	11480.9	203.40	41	0.10
25	11673.5	7494.2	8717.8	81.18	54	0.70	11187.1	172.77	28	0.11

of the same size. CG1 and CG2 stand for Column Generation approach to the problem SD1 and SD2 respectively. SCG1 and SCG2 are for the stabilized version of the CG1 and CG2 respectively. The results show that stabilization is effective for both symmetric and asymmetric instances.

6.6.2 Lower bound computation

A solution of the pricing problem, which may contain a single or multiple cycles, is optimal for the master problem $RD1$ if it covers all the nodes $i \in V$. Since looking for a single cycle in the pricing problem requires some kind of subtour elimination constraint, we restrict the search to find a cycle (single or multiple) with negative cost. In other words, we allow subtours in the optimal solution of the original QTSP, which is in fact a relaxation of the problem. Therefore, when no more new columns can be priced out, a solution of the master problem RD1 gives a lower bound on the original problem. Note that the optimal value of the problem RD2 always gives a lower bound on the original problem, regardless of the solution being a single cycle or multiple ones.

In Tables 6.2 and 6.3 we present computational results of the lower bounding schemes for the symmetric and the asymmetric QTSPs respectively. Each row of these tables reports the average results over ten instances of the same size. The problem size is found in the first column of the tables. The second column shows the average optimal values (opt) of the 10 instances for each dimension. We compare three different average lower bounds (LB) on the optimal objective values, their computing time (time), number of iterations (iter), and the average gap. The first lower bound LB(LP) in column three is the lower bound obtained with the linear relaxation of problem (6.4)–(6.8) and the linear relaxation of problem GTSP3 for the SQTSP and the AQTSP respectively. The computation time of the LP relaxation is less than two seconds for all instances; therefore we did not mention it in the table. The second lower bound is obtained via the Stabilized Column Generation approach to the problem SD2 (SCG2); and the third lower bound is obtained via the Stabilized Column Generation approach apply to the problem SD1 (SCG1). Columns four to seven and eight to eleven represent the optimal value, computation time, number of iterations needed to identify the optimal solution and the ratio of gap between the lower bound obtained by SCG and the optimal solution, and the gap between the lower bound obtained by the linear relaxation and the optimal solution. The formula we used to compute these ratio of gaps is $(opt - LB(SCG))/(opt - LB(LP))$, where opt and $LB()$ stand for the optimal value and the lower bound, respectively. We can see in these tables that the bounds obtained by SCG1 outperforms the other two in all instances and are close to the optimal values in both the SQTSP and the AQTSP. Also, the lower bound obtained by SCG2 is better than the one obtained with LP relaxation except for the instances of dimension five, for which the LP relaxation gives on average a tighter bound. On average the ratio of gap between the lower bound obtained by SCG2 and the optimal solution, and the gap between the lower bound obtained by the linear relaxation and the optimal solution for the symmetric instances is 0.72, while this ratio for the asymmetric instances is 0.73. As we see, on average, the improvement of lower bound by applying the SCG2 for both symmetric and the asymmetric cases is almost the same, while the computational time for the asymmetric instances is larger than the computational time for the symmetric ones. According to Table 6.2, applying SCG1 yields a considerable improvement of the lower bounds in both the symmetric and the asymmetric cases, i.e., on average the ratio of gap between the lower bound obtained by SCG1 and the optimal solution over the gap between the lower bound obtained by linear relaxation and the optimal solution for the symmetric instances is 0.09, and for the asymmetric instances is 0.20. These gaps show the improvement of lower bounds in both the symmetric and asymmetric cases in comparison to the SCG2. However, it should be noted that the improvement of the lower bounds and also of the computational time in the symmetric case is more attractive than in the asymmetric case.

Table 6.3 Comparison of three different lower bounding approaches for the asymmetric QTSP instances.

			$SCG2$				$SCG1$			
size	*Opt.*	$LB(LP)$	LB	*time*	*iter*	*Gap*	LB	*time*	*iter*	*Gap*
5	12372.2	10758.2	11126.8	1.80	46	0.77	12373.2	2.10	56	0.0
6	11883.1	10291.6	10596.9	1.79	38	0.80	10684.6	1.51	37	0.75
7	13204.9	10486.3	11369.9	1.95	42	0.67	12746.3	2.98	55	0.16
8	13363.4	10116.3	11194.4	2.08	38	0.66	12878.5	3.24	45	0.14
9	13063.2	9610.1	10546.5	2.52	37	0.72	12135.1	5.24	48	0.26
10	12921.5	9427.1	10461.9	3.26	33	0.70	12500.8	7.35	43	0.12
11	12997.5	8965.8	9891.20	4.41	34	0.77	12089.9	11.99	49	0.22
12	11434.8	8723.3	9682.4	5.79	29	0.64	10897.9	10.87	35	0.19
13	12171.5	8421.6	9608.1	8.55	33	0.68	11584.3	21.11	41	0.15
14	11838.3	8182.7	9005.6	9.95	37	0.77	10635.6	20.66	38	0.32
15	12428.8	8094.6	9564.7	16.83	35	0.66	11748.8	37.05	45	0.15
16	12135.7	7726.8	8764.6	16.64	35	0.76	11119.8	47.34	51	0.23
17	11832.2	7241.3	8682.7	21.69	35	0.68	10094.7	60.02	50	0.37
18	11662.2	7380.9	8544.8	24.40	36	0.72	11104.6	119.24	59	0.13
19	12095.9	7264.6	8560.4	31.22	38	0.73	11207.8	157.51	57	0.18
20	11802.8	6951.1	8200.1	34.92	37	0.74	11162.4	271.81	63	0.13
21	11288.2	6948.5	8140.5	51.44	40	0.72	10819.1	435.246	58	0.10
22	11741.0	6906.8	7950.1	51.61	43	0.78	10517.2	480.15	58	0.25
23	11549.7	6821.3	7688.3	59.05	45	0.81	10549.3	671.15	68	0.21
24	11239.1	6580.4	7716.1	84.03	43	0.75	10180.1	877.83	69	0.22
25	11434.8	6663.1	7785.6	105.36	44	0.76	10422.3	1698.31	70	0.21

6.7 Conclusions

In this chapter we proposed two different linearization models to the SQTSP and the AQTSP. Moreover, we presented a cycle formulation for the QTSP (in general) and solved the resulting LP problem by a Column Generation approach. We have shown how the linearized formulations can be applied to find the negative reduced cost in the pricing problem. To overcome the problems of instability in the behavior of dual variables of the presented master problem, we used a stabilized column generation approach. Our experiments show that our column generation approach outperforms the LP relaxation of the QTSP in both the symmetric and the asymmetric cases.

Chapter 7

Quadratic Minimum Spanning Tree Problem

The Minimum Spanning Tree Problem (MSTP) is one of the most known combinatorial optimization problems. It concerns the determination of a minimum edge-cost subgraph spanning all the vertices of a given connected graph. The Quadratic Minimum Spanning Tree Problem (QMSTP) is a variant of the MST whose cost considers also the interaction between every pair of edges of the tree. In this chapter we review different strategies found in the literature to compute a lower bound for the QMSTP and develop new bounds based on a reformulation scheme and some new mixed 0-1 linear formulations that result from a reformulation-linearization technique (RLT). We develop some efficient dual-ascent algorithms to derive new lower bounds and compare the new bounds with the other bounding procedures in terms of both overall strength and computational effort.

7.1 Introduction

Given an undirected graph $G = (V, E)$, with $|V| = n$ and $|E| = m$, a matrix of quadratic costs C with $C_{ef} \geq 0$, $\forall e, f \in E$, and linear costs $d_e \geq 0$, $\forall e \in E$, the quadratic minimum spanning tree problem (QMSTP) consists of finding a spanning tree $T \in G$ with minimum overall cost $\sum_{e,f \in T} C_{ef} + \sum_{e \in T} d_e$.

The QMSTP has been used to model many applications arising in transportation, telecommunication, and energy networks, where linear costs account for the use or construction of edges while the quadratic costs represent the interference between the edges [14, 97]. When the interference refers only to pairs of adjacent edges the problem is named adjacent QMSTP (AQMSTP). Both the general QMSTP and the AQMSTP are NP-hard as proved by Assad and Xu [14].

Many exact and heuristic algorithms have been proposed for solving both the QMSTP and the AQMSTP. Assad and Xu in [14] proposed a lower bounding procedure and two heuristic approaches. They also described the branch-and-bound algorithm based on a linearized formulation. Öncan and Punnen [94] introduced a Lagrangian relaxation procedure to obtain an improved lower bound and an efficient local search algorithm. Cordone and Passeri [33] have developed two heuristics and an exact approach. Buchheim and Klein [22] proposed complete polyhedral descriptions of the QMSTP with one quadratic term and provide an improved version of the standard linearization by means of cutting planes. Perrira et al. [98] proposed some new formulations using a particular partitioning of the spanning trees, and provided a new mixed binary formulation for the problem by applying the first level of the reformulation-linearization technique. The most effective heuristic approaches for the QMSTP can be found in [83, 97, 115, 120].

Lower bounds constitute a fundamental component of the branch-and-bound technique, and are a basic tool for the evaluation of the quality of the solutions obtained from heuristic algorithms. There are several branch-and bound based solution methods proposed in the literature for solving the QMSTP [14, 33, 98]. In practice, the lack of efficiently computable, tight lower bounds for the QMSTP can be key factor in the problem's difficulty. In the other hand, the bounding procedure used in branch-and-bound algorithms must tradeoff between tightness and computational efficiency. The first lower bounding procedure for the QMSTP, proposed by Assad and Xu [14] iteratively applies an adaptation of the Gilmore-Lawler procedure to a sequence of equivalent QMSTPs. Öncan and Punnen in [94] introduced an extended formulation based on the addition of two sets of valid inequalities to the linearized formulation of [14]. Their lower bounding approach applies a Lagrangian relaxation where the Gilmore-Lawler procedure is used to solve the resulting Lagrangian subproblem. Pereira et al. [98] proposed a new mixed binary formulation for the problem and developed a Lagrangian relaxation approach to obtain a linear programming based lower bound.

In order to find a common framework for description of the individual bounds we review and analyze different bounding procedure proposed in the literature and compare them in terms of continuous relaxation of an MILP. We describe new bounds for the problem by considering a reformulation of the problem based on dual information retrieved from the continuous relaxation of the MILP. The basic idea is to solve the continuous relaxation of the given MILP, and then use the reduced costs as the objective coefficients of the reformulated problem. Moreover, for generating superior bounds, we develop a mixed 0-1 linear formulation based on the second level of the reformulation-linearization technique and devise an efficient dual-ascent algorithm to solve the continuous relaxation of the proposed model based on reduced costs.

7.2 Problem formulation and lower bounds review

In order to present the mathematical formulation of the QMSTP, let us first introduce some notation used in the sequel. We denote by $E(S)$ the set of all edges with both endpoints in S for any $S \subset V$, and $\delta(i)$ as the set of all edges incident in node i. We define the binary variable x_e to indicate the presence of edge $e \in E$ in the optimal spanning tree. The QMSTP has the following Integer formulation:

$$\begin{aligned}
\text{QMSTP:} \quad \min \quad & \sum_{\substack{e,f\in E\\ e\neq f}} C_{ef}x_e x_f + \sum_{e\in E} d_e x_e && \\
\text{s.t.} \quad & \sum_{e\in E} x_e = n-1 && && (7.1)\\
& \sum_{e\in E(S)} x_e \leq |S|-1 && \forall\, \emptyset \neq S \subset V && (7.2)\\
& x_e \text{ integer} && \forall e \in E. && (7.3)
\end{aligned}$$

where the objective function considers the linear cost of the selected edges and also the interaction costs between pairs of edges. Constraints (7.2) are the subtour elimination constraints and ensure that no subgraph induced by the nonempty subset $S \subset V$ contains a cycle. These subtour elimination constraints together with the cardinality constraint (7.1) guarantee the connectivity of the induced subgraph. Constraints (7.1) to (7.3) define the set of spanning trees in G and thereafter is denoted by X, i.e.,

$$X = \{x \geq 0 : \text{(7.1), (7.2), (7.3)}\}.$$

7.2.1 Gilmore-Lawler type bound

The Gilmore-Lawler procedure, shortly denoted by GL, has been adapted to many other quadratic $0-1$ problems [e.g. 25, 26]. The popularity of this approach for computing lower bounds of the linearly constrained quadratic $0-1$ problems stems from the fact that it is computationally inexpensive. For each edge e, potentially in the solution, we consider the best cumulation providing the minimum interaction cost with e. Let P_e be such a subproblem for a given edge $e \in E$:

$$P_e : \quad z_e = \min \left\{ \sum_{\substack{f\in E\\ f\neq e}} C_{ef}x_f : x \in X,\ x_e = 1 \right\} \quad \forall e \in E. \qquad (7.4)$$

The value z_e is the best quadratic contribution to the QMSTP objective function where edge e is in the solution. Once z_e has been computed for each $e \in E$, the GL type bound is given by the solution of the following MSTP:

$$LB_{GL} = \min \left\{ \sum_{e \in E} (z_e + d_e) x_e : \; x \in X \right\}. \tag{7.5}$$

Although the GL bound that we just described is a pure combinatorial bound of the QMSTP, it can also be obtained in the terms of a linear programming. More precisely, consider the following MILP formulation where the decision variables y_{ef} equal to 1 if and only if both edges e and f are present in the solution of the problem:

$$\begin{aligned}
\text{P:} \quad \min \quad & \sum_{\substack{e,f \in E \\ e \neq f}} C_{ef} y_{ef} + \sum_{e \in E} d_e x_e \\
\text{s.t.} \quad & \sum_{f \in E} y_{ef} = (n-1) x_e && \forall e \in E && (7.6) \\
& \sum_{f \in E(S)} y_{ef} \leq (|S| - 1) x_e && \forall\, \emptyset \neq S \subset V,\ e \in E && (7.7) \\
& y_{ee} = x_e && \forall e \in E && (7.8) \\
& y_{ef} \geq 0 && \forall e, f \in E && (7.9) \\
& x \in X.
\end{aligned}$$

Constraints (7.6) guarantee that whenever an edge $e \in E$ is selected, the total number of selected edges interacting with e must be equal to $(n-1)$, including e itself. Overall, constraints (7.6) to (7.9) enforce $y_{e_1 e}, \ldots, y_{e_m e}$ to be a spanning tree containing e, in case $x_e = 1$, or to be the null vector, in case $x_e = 0$.

Let us consider now the lower bound computations taking advantage of the MILP formulation introduced above. Consider the continuous relaxation of problem P (CP) by replacing the boolean constraints on variable x by nonnegativity constraint. The problem CP is computationally interesting since, its optimal objective value gives the GL bound as stated in the following theorem.

Theorem 7.2.1. *The continuous relaxation of problem P gives the GL bound.*

Proof. Let λ, μ_S, α_e, and γ_{Se} denote the dual variables corresponding to constraints (7.1), (7.2), (7.6), and (7.7), respectively. Consider the dual of CP

DCP:

$$
\begin{aligned}
\max \quad & -\lambda(n-1) - \sum_{S\subset V} \mu_S(|S|-1) \\
\text{s.t.} \quad & -\lambda - \sum_{\substack{S\subset V:\\ e\in E(S)}} \mu_S + (n-1)\alpha_e + \sum_{S\subset V}(|S|-1)\gamma_{Se} \le d_e \quad \forall e \in E & (7.10)\\
& -\alpha_e - \sum_{\substack{S\subset V:\\ f\in E(S)}} \gamma_{Se} \le C_{ef} \quad \forall e,f \in E & (7.11)\\
& \gamma_{Se} \ge 0 \quad \forall S \subset V,\ e \in E & (7.12)\\
& \mu_S \ge 0 \quad \forall S \subset V.
\end{aligned}
$$

Considering constraints (7.10), one can maximize the objective function by solving the following m independent subproblems; one for each $e \in E$:

$$
Sub_e: \quad \overline{z}_e = \max \quad \left\{ -(n-1)\alpha_e - \sum_{S\subset V}(|S|-1)\gamma_{Se} : \text{ (7.11), (7.12)} \right\}. \tag{7.13}
$$

For each $e \in E$, subproblem Sub_e is precisely the dual of the subproblem (7.4) used in the GL bound. Thus $\overline{z}_e = z_e$, and DCP can be rewritten as follows:

$$
\begin{aligned}
\text{DCP} \quad \max \quad & -\lambda(n-1) - \sum_{S\subset V} \mu_S(|S|-1) \\
\text{s.t.} \quad & -\lambda - \sum_{\substack{S\subset V:\\ e\in E(S)}} \mu_S \le d_e + z_e & \forall e \in E\\
& \mu_S \ge 0 & \forall S \subset V.
\end{aligned}
$$

which is exactly the dual of the final MSTP (7.5) of the GL bound. □

Theorem 7.2.1 shows that the optimal solution of problem CP can be computed by solving m MSTP containing a fixed edge e, and a single MSTP. Although the LB_{GL} provides a lower bound for the QMSTP that can be computed very efficiently, as in the QAP case, the obtained bounds are not so close to the optimal solution values even for small size instances.

7.2.2 Assad and Xu bound

In order to improve the GL bound for the QMSTP, Assad and Xu[14] proposed a method that generates a monotonic sequence of lower bounds. Their method relies on an equivalent reformulation of the QMSTP with coefficients C'_{ef} and d'_e where:

$$d'_e = d_e - (n-2)\theta_e \quad \forall e \in E, \quad \text{and} \quad C'_{ef} = C_{ef} + \theta_f \quad \forall e, f \in E. \tag{7.14}$$

The authors proved that, for any value of vector θ, transformation (7.14) defines an equivalent QMSTP, i.e., for each $x \in X$:

$$\sum_{\substack{e,f\in E\\ e\neq f}} C'_{ef}x_e x_f + \sum_{e\in E} d'_e x_e = \sum_{\substack{e,f\in E\\ e\neq f}} C_{ef}x_e x_f + \sum_{e\in E} d_e x_e.$$

Although the optimum of the QMSTP is independent of vector θ, the approximated linear cost d'_e depends on θ. Therefore, a better choice of θ yields a tighter bound.

For a given θ, Assad and Xu apply the GL procedure to compute the following lower bound:

$$LB_{AX}(\theta) = \min\left\{\sum_{e\in E}(z_e(\theta) + c'_e)x_e : x \in X\right\},$$

where

$$z_e(\theta) = \min\left\{\sum_{\substack{f\in E\\ f\neq e}} C'_{ef}x_f : x_e = 1 \text{ and } x \in X\right\} \quad \forall e \in E.$$

Observe that $LB_{AX}(\theta)$ is a piecewise concave linear function, and for $\theta = 0$ it gives the GL bound. To find the best value of parameter θ the authors provide a leveling procedure to find an $\varepsilon - optimal$ solution. More precisely, by updating $\theta_e^{i+1} = \theta_e^i + \frac{1}{n-1}z_e(\theta^i)$ at each iteration i, the $LB_{AX}(\theta)$ is iteratively improved until the stopping criterion $\max_e\{z_e(\theta^i)\} - \min_e\{z_e(\theta^i)\} < \varepsilon/(n-1)$ is satisfied.

We now turn to problem CP and review the AX bound in the context of linear programming. To this end, consider problem P and add the following set of constraints to the formulation

$$\sum_{e\in E} y_{ef} = (n-1)x_f \quad \forall f \in E \tag{7.15}$$

denoting the resulting formulation as QAX. Assad and Xu showed that QAX and QMSTP are equivalent. The following theorem shows the relationship between the AX bounds and the continuous relaxation of QAX.

Theorem 7.2.2. *The continuous relaxation of problem QAX gives the AX bound.*

Proof. The relaxation of (7.15) in QAX formulation using Lagrangian multipliers θ yields the Lagrangian dual:

$$\text{LD}: \quad \max_{\theta\in R^m} L(\theta)$$

where $L(\theta)$ is the optimal value of the below problem:
LR(θ):

$$L(\theta) = \min \left\{ \sum_{\substack{e,f \in E \\ e \neq f}} (C_{ef} + \theta_f) y_{ef} + \sum_{e \in E} (d_e - (n-2)\theta_e) x_e \, : \, x \in X, \, (7.6) - (7.9) \right\}.$$

Since LR(θ) satisfies the integrality property, the optimal solution value of problem LD is equal to the optimal solution value of the continuous relaxation of problem QAX. On the other hand, according to Theorem 7.2.1, the value of $L(\theta)$ is equal to the GL bound of problem LR(θ). This completes the proof. □

7.2.3 Öncan and Punnen bound

Öncan and Punnen in [94] proposed an extended formulation for the QMSTP based on the addition of the following two sets of valid inequalities to QAX:

$$\sum_{e \in \delta(i)} y_{ef} \geq x_f \qquad \forall i \in V, \; f \in E \qquad (7.16)$$

$$\sum_{f \in \delta(i)} y_{ef} \geq x_e \qquad \forall i \in V, \; e \in E \qquad (7.17)$$

The formulation is presented below as problem QOP:

$$\text{QOP:} \quad \min \left\{ \sum_{\substack{e,f \in E \\ e \neq f}} C_{ef} y_{ef} + \sum_{e \in E} d_e x_e \, : \, x \in X, \, (7.6) - (7.9), (7.16), (7.17) \right\}.$$

Consider now the Lagrangian relaxation of (7.16) with multipliers λ_{if}. The Lagrangian function has the following form:

$$L(\lambda) = \min \left\{ \sum_{e,f \in E} C''_{ef} y_{ef} + \sum_{e \in E} d''_e x_e \, : \, x \in X, \, (7.6) - (7.9), (7.17) \right\}$$

where

$$C''_{ef} = C_{ef} - \sum_{\substack{i \in V: \\ e \in \delta(i)}} \lambda_{if} \qquad \forall e, f \in E, e \neq f \qquad (7.18)$$

$$d''_e = d_e + \sum_{i \in V} \lambda_{ie} \qquad \forall e \in E. \qquad (7.19)$$

To compute the Lagrangian function for a given λ, the authors use the decomposition approach presented in Theorem 7.2.1. In so doing, they first sequentially fix one edge at a time and find the minimum spanning tree containing it, then they solve a MSTP with the optimal objective values of the subproblems as edge weights.

It should be noted that the actual formula used in [94] for computing coefficients C'' is

$$C''_{ef} = C_{ef} - \lambda_{if} \qquad\qquad \forall e, f \in E, i \in V.$$

which is not correct. However, the general form of the Lagrangian function and consequently the decomposition procedure is still correct. Our computational experiments (see section 7.5) show that in spite of the correct decomposition structure with easily solvable subproblems, the produced bounds are not significantly stronger than the AX bounds and in some benchmark instances are even worse.

7.3 Bounds based on a new reformulation

In order to develop a new bound for the QMSTP, we consider a modified problem based on dual information retrieved from the solution of the CP. The basic idea is to solve CP, and then use the reduced costs as the coefficients of a reformulation adapting the approach proposed in [27, 28] for the QAP. To this end, we introduce the following problem transformation:

$$\begin{cases} \tilde{C}_{ef} = C_{ef} + \alpha_e + \sum_{\substack{S \subset V: \\ f \in E(S)}} \gamma_{Se} + \delta_{ef} + \theta_f & \forall e, f \in E \\ \tilde{d}_e = d_e - (n-1)\alpha_e - \sum_{S \subset V} (|S| - 1)\gamma_{Se} - (n-2)\theta_e & \forall e \in E. \end{cases} \tag{7.20}$$

where α_e, $e \in E$, θ_f, $f \in E$, δ_{ef}, for $e, f \in E$, are vectors of real parameters, and γ_{Se}, for $e \in E, S \subset V$ is a vector of nonnegative real parameters. This transformation, in general, does not provide a reformulation of the QMSTP for any value of parameters α, θ, δ, and γ, but for some particular values it guarantees the definition of an equivalent QMSTP. In the following we present the general framework of our reformulation scheme.

Reformulation framework

For any given value of θ^0, we initialize the reformulation process by setting $\alpha = \gamma = \delta = 0$, and $\theta = \theta^0$. Noting that this parameter setting guarantees that the transformation (7.20) provides a reformulation for the QMSTP as proved in [14]. Next, we apply the GL procedure to obtain a the lower bound. More precisely, we sequentially consider the following m subproblems $SP(e)$, $e \in E$:

$$\begin{aligned} SP(e): \quad z(e) = \min & \sum_{f \in E} \tilde{C}_{ef} y_{ef} \\ \text{s.t.} \quad & \sum_{f \in E} y_{ef} = (n-1) && && (7.21) \\ & \sum_{f \in E(S)} y_{ef} \leq |S| - 1 && \forall\, \emptyset \neq S \subset V && (7.22) \\ & y_{ef} \geq 0 && \forall f \in E \\ & y_{ee} = 1. \end{aligned}$$

The solution of the following minimum spanning tree problem gives the GL bound for the transformed problem:

$$W' = \left\{ \sum_{e} \tilde{d}_e x_e \,:\, x \in X \right\}.$$

Upon solving $SP(e)$, for all $e \in E$, we retrieve the dual solutions α'_e and γ'_{Se} corresponding to constraints (7.21) and (7.22), respectively, and modify parameters α, and γ as

$$\begin{aligned} \alpha_e &= \alpha'_e \quad \forall e \in E, \\ \gamma_{Se} &= \gamma'_{Se} \quad \forall e \in E,\ S \subset V. \end{aligned}$$

Moreover, we define δ_{ef} for all $e, f \in ord(E)$, and set $\delta_{fe} = -\delta_{ef}$ for all $e, f \in ord(E)$, where $ord(E) = \{e = (i,j), f = (k,l) \in E : i < k \text{ or } i = k, j < l\}$.

With the parameter setting just described we have the following:

Theorem 7.3.1. *For any value of θ and any value of δ with $\delta_{fe} = -\delta_{ef}, \forall e, f \in ord(E)$, if $\alpha_e = \alpha'_e$, $e \in E$ and $\gamma_{Se} = \gamma'_{Se}$, $e \in E, S \subset V$, the transformation* (7.20) *provides a reformulation of the QMSTP.*

Proof. Consider problem QAX with the following extra constraints:

$$y_{ef} = y_{fe} \quad \forall e, f \in ord(E), \tag{7.23}$$

Associate Lagrangian multipliers α_e, θ_f, and δ_{ef} with constraints (7.6), (7.15), and (7.23), respectively, and nonnegative Lagrange multipliers γ_{Se} with constraints (7.7). The Lagrangian function can be computed by solving the following MILP:

MP$(\alpha, \theta, \delta, \gamma)$:

$$\begin{aligned}\min \quad & \sum_{\substack{e,f\in E\\ e\neq f}} C_{ef}y_{ef} + \sum_{e\in E} d_e x_e + \sum_{\substack{e,f\in E\\ e\neq f}} \delta_{ef}y_{ef} + \sum_{f\in E}\theta_f[\sum_{\substack{e\in E\\ e\neq f}} y_{ef} - (n-2)x_f] \\ & + \sum_{e\in E}\alpha_e[\sum_{\substack{f\in E\\ f\neq e}} y_{ef} - (n-2)x_e] + \sum_{e\in E}\sum_{\emptyset\subset S\subset V}\gamma_{Se}[\sum_{f\in E(S)} y_{ef} - (|S|-1)x_e] \\ \text{s.t.} \quad & (7.6) - (7.9) \\ & x \in X.\end{aligned}$$

For any feasible solution (x, y) and any value of the multipliers α_e, and θ_f, the terms $\sum_f \theta_f[\sum_{e\in E} y_{ef} - (n-2)x_f]$ and $\sum_e \alpha_e[\sum_{f\in E} y_{ef} - (n-1)x_e]$ are all zero. Moreover, for each $e \in E$ and $S \subset V$ complementary slackness properties imply:

$$\sum_{e\in E}\sum_{\emptyset\subset S\subset V}\gamma_{Se}[\sum_{f\in E(S)} y_{ef} - (|S|-1)x_e] = 0.$$

Therefore, the objective function of MP$(\alpha, \theta, \delta, \gamma)$ reduces to

$$\begin{aligned}\min \quad & \sum_{\substack{e,f\in E\\ e\neq f}} (C_{ef}+\delta_{ef})y_{ef} + \sum_{e\in E} d_e x_e = \\ \min \quad & \sum_{e,f\in ord(E)} (C_{ef}+\delta_{ef})y_{ef} + \sum_{e,f\notin ord(E)} (C_{fe}-\delta_{ef})y_{fe} + \sum_{e\in E} d_e x_e. \quad (7.24)\end{aligned}$$

Since the optimum of (7.24) achieved if $y_{ef} = y_{fe}$, it is simplified as

$$\min \quad \sum_{e,f\in ord(E)} (C_{ef})y_{ef} + \sum_{e,f\notin ord(E)} (C_{fe})y_{fe} + \sum_{e\in E} d_e x_e.$$

and proof is completed. □

Let us make some observations concerning the way we can choose the multipliers in (7.20).

Observation 1.
The MP$(\alpha, \theta, \delta, \gamma)$ formulation contains an exponential number of constraints with respect to the number of nodes. Therefore, explicitly keeping track of the dual values for (7.22) is not practical. However, for any given value of θ and δ, transformation (7.20) can be interpreted by considering the meaning of reduced costs in (7.11), and the optimal value of subproblem $SP(e)$. More preciesly, we can simplify the transformation (7.20) as

$$\begin{cases} \tilde{C}_{ef} = \widetilde{rC}_{ef} + \delta_{ef} + \theta_f & \forall e, f \in E \\ \tilde{d}_e = d_e + z(e) - (n-2)\theta_e & \forall e \in E. \end{cases} \tag{7.25}$$

where $\widetilde{rC}_{ef}$ is the reduced cost of edge f with respect to the optimal spanning tree containing the fixed edge e. In this case, the reduced cost $\widetilde{rC}_{ef}$ can be computed without using the dual variables α and γ. More specifically, the reduced cost $\widetilde{rC}_{ef}$ of any edge f not in the minimum spanning tree containing the fixed edge e is the difference between the cost of f and the largest cost of any edge, different from e, in the cycle induced by f. Note that the reduced costs computed this way are obviously non negative.

Observation 2.
In order to develop a procedure to find the optimal dual multipliers δ, we can split the costs between symmetrical entries of $\tilde{C}$ so as to maximize the value of W'. Using this idea, it is not necessary to explicitly find the value of the multipliers δ_{ef} in each iteration of the dual ascent algorithm. In fact one can adjust the coefficient $\tilde{C}_{ef}$ on variables y_{ef} every time multiples of (7.23) are placed into the objective function. More precisely, for any $e, f \in ord(E)$, we can increase $\tilde{C}_{ef}$ by the same quantity we decrease $\tilde{C}_{fe}$.

Observation 3.
By stopping the procedure at the first iteration, with $\theta = 0$, the value of W' gives the GL bound. To obtain tighter lower bounds we can maximize the function W' to obtain the best value of θ for example through subgradient techniques. However, we follow the idea of Assad and Xu to update the parameter θ. More precisely, we propose the following two update rules:

$$\theta_e^{i+1} = \frac{1}{(n-2)^{1/2}}\theta_e^i + \frac{1}{(n-2)^{3/2}}(\tilde{d}_e + \widetilde{rd}_e) \quad \forall e \in E \tag{7.26}$$

$$\theta_e^{i+1} = \frac{1}{n-2}\widetilde{d}_e \quad \forall e \in E. \tag{7.27}$$

where $\widetilde{rd}_e$ for all $e \in E$ is the reduced cost of edge e with respect to the optimal spanning tree. The first choice of updating θ does not guarantee a monotonically increasing sequence of lower bounds, while for the second choice our computational experiments show that it yields a monotonic sequence of lower bounds.

7.4 Reformulation-linearization technique applied to the QMSTP

In this section we present the level-1 and level-2 of the RLT representations of the QMSTP. The level-1 RLT representation of the QMSTP proposed in [98] applies two sets of operations. First, each constraint defining X is multiplied by each of the m variables x_e. All such quadratic constraints are included in the formulation. Note that when a variable x_e in a given constraint is multiplied by x_f the resulting product is expressed as $x_e x_f$ in that order. The linearization step makes the substitution of $y_{ef} = x_e x_f$ for all $e, f \in E$, and imposes the restrictions $y_{ef} = y_{fe}$ for all $e, f \in ord(E)$, to rewrite the problem as the MILP below:

$$\begin{aligned} \text{QRLT1:} \quad \min \quad & \sum_{\substack{e,f\in E\\ e\neq f}} C_{ef} y_{ef} + \sum_{e\in E} d_e x_e \\ \text{s.t.} \quad & (7.6), (7.7), (7.8), (7.9), (7.23) \\ & x \in X. \end{aligned}$$

Observe that QRLT1 is exactly the same as problem P with the addition of constraints (7.23). However, the continuous relaxation of QRLT1 (CRLT1) provides a lower bound at least as large as the continuous relaxations of QAX (CAX) and QOP (COP) as proved in [98]. In fact the authors showed that the feasible region to CRLT1 (P_{RLT1}) is contained within the feasible region to CAX (P_{AX}) and COP (P_{OP}), i.e,

$$P_{AX} \supseteq P_{OP} \supseteq P_{RLT1}. \tag{7.28}$$

A natural question concerning the relation in (7.28) is whether these inclusions are strict or not. In the following we show that $P_{OP} \supset P_{RLT1}$. To show the tight inclusion, consider a graph $G = (V, E)$ with $V = \{a, b, c, d\}$ and $E = \{e_1, e_2, e_3, e_4\}$ where $e_1 = \{a, b\}$, $e_2 = \{b, c\}$, $e_3 = \{b, d\}$, and $\{c, d\}$. Suppose that the linear cost d_i for any $e_i \in E$ is equal to zero, and the quadratic costs C_{ij} of edges $e_i, e_j \in E$ is also equal to zero except for (e_1, e_2) and (e_4, e_1) which $C_{12} = C_{41} = 1$. Consider the point $(\overline{\boldsymbol{x}}, \overline{\boldsymbol{y}})$ with $\overline{x}_1 = 1, \overline{x}_2 = \overline{x}_3 = \overline{x}_4 = \frac{2}{3}$, $\overline{y}_{13} = \overline{y}_{14} = \overline{y}_{21} = \overline{y}_{31} = \overline{y}_{42} = 1, \overline{y}_{24} = \overline{y}_{32} = \overline{y}_{43} = \frac{1}{3}$ and all other $\overline{y}_{ij} = 0$ with $i \neq j$. This point is a feasible and optimal solution with objective value 0 for problem COP, but it is not feasible for CRLT1, as $1 = \overline{y}_{21} \neq \overline{y}_{12} = 0$. For the given objective function coefficients, the point $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}})$ with $\hat{x}_1 = \hat{x}_3 = \hat{x}_4 = 1, \hat{x}_2 = 0, \hat{y}_{13} = \hat{y}_{14} = \hat{y}_{31} = \hat{y}_{34} = \hat{y}_{41} = \hat{y}_{43} = 1$ and $\hat{y}_{ij} = 0$ for all other $i \neq j$, with objective value of 1 is in fact the optimal solution for problem CRLT1.

Thus we have the following:

Proposition 7.4.1.

$$P_{AX} \supseteq P_{OP} \supset P_{RLT1}.$$

In order to solve the CRLT1, the authors proposed constructing a Lagrangain dual by placing constraints (7.23) into the objective function and applied the GL procedure to solve the Lagrangian function. They used subgradient method to compute an optimal (or near-optimal) set of Lagrangian multipliers.

7.4.1 Level-2 RLT on the QMSTP

Based on the success of level-1 RLT representation to gain tighter bounds for the QMSTP and also due to the block-diagonal structure of the problem which lends to efficient solution methods, we turn attention to the level-2 representation of the problem. We first present the level-2 RLT formulation of the problem and then show how to handle it via a Lagrangian relaxation approach to obtain a Lagrangian function with block-diagonal structure. Since the dualized constraints are indeed much more than those in the level-1 RLT, finding the near-optimal dual multipliers using the classical subgradient is not viable. Therefore, we devise an efficient dual-ascent procedure to solve the continuous relaxation of the level-2 RLT.

The level-2 RLT representation of QMSTP can be obtained in the same way as the level-1 RLT via the following reformulation and linearization steps.

Reformulation: Multiply each constraints defining X by each binary variable x_e, $e \in E$, and also by each pair-wise product of variables $x_e x_f$, $e, f \in E$. Add these new restrictions to the problem formulation. When a variable x_e is multiplied by x_f, the resulting product is expressed as $x_e x_f$, and when it is multiplied by $x_f x_h$, the resulting product is expressed as $x_e x_f x_h$, preserving the order in both cases.

Linearization: Linearize the resulting problem by substituting, $x_e x_f$ and $x_e x_f x_h$ with continuous variables y_{ef} and u_{efh}, respectively. Enforce the equality $y_{ef} = y_{fe}$ for all $e, f \in E$, and also enforce the equalities $u_{efh} = u_{ehf} = u_{feh} = u_{fhe} = u_{hef} = u_{hfe}$ for all $e, f, h \in E$. For convenience we set $u_{eef} = u_{efe} = u_{fee} = y_{fe}$ for all $e, f \in E$ and $y_{ee} = x_e$ for all $e \in E$.

The resulting formulation is presented below where the coefficients D_{efh} found in the objective function are all zero.

QRLT2:

$$\min \quad \sum_{e,f,h\in E} D_{efh}u_{efh} + \sum_{\substack{e,f\in E\\ e\neq f}} C_{ef}y_{ef} + \sum_{e\in E} d_e x_e$$

$$\sum_{f\in E} u_{ehf} = (n-1)y_{ef} \quad \forall e,h\in E \tag{7.29}$$

$$\sum_{f\in E(S)} u_{ehf} \le (|S|-1)y_{ef} \quad \forall\, \emptyset \neq S\subset V,\ e,h\in E \tag{7.30}$$

$$u_{eeh} = u_{ehe} = u_{hee} = y_{he} \quad \forall e,h\in E \tag{7.31}$$

$$u_{efh} = u_{ehf} = u_{feh} = u_{fhe} = u_{hef} = u_{hfe} \quad \forall e,f,h\in E \tag{7.32}$$

$$u_{efh} \ge 0 \quad \forall e,f,h\in E \tag{7.33}$$

$$(7.6), (7.7), (7.8), (7.9), (7.23) \tag{7.34}$$

$$x\in X.$$

Note that an optimal solution of the QRLT2 will yield an optimal solution also for the QMSTP problem. However, if the binary restrictions on variables x are relaxed in QRLT2, the problem is no longer equivalent to QMSTP, providing a lower bound.

Lagrangian Relaxation Scheme of QRLT2

In order to solve the continuous relaxation of QRLT2 efficiently, we apply a Lagrangian relaxation to constraints (7.23) and (7.32). Let $\overline{D}_{efh}$, $\overline{C}_{ef}$ and $\overline{d}_e$ denote the adjusted values for D_{efh}, C_{ef} and d_e respectively, after placing constraints (7.23) and (7.32) into the objective function. The resulting Lagrangian relaxation is:

$$K + \min \quad \sum_{e,f,h\in E} \overline{D}_{efh}u_{efh} + \sum_{\substack{e,f\in E\\ e\neq f}} \overline{C}_{ef}y_{ef} + \sum_{e\in E} \overline{d}_e x_e \tag{7.35}$$

$$(7.29), (7.30), (7.31), (7.33), (7.34) \tag{7.36}$$

$$x\in X. \tag{7.37}$$

where K is a scalar equal to 0.

To solve the Lagrangian function (7.35) we introduce the following subproblems. For any two edges $e,f\in E, e\neq f$, let φ_{ef} denote the cost of the optimal spanning tree containing the two fixed edges e,f, and for any edge $e\in e$, let ρ_e denotes the optimal spanning tree containing the fixed edge e; i.e.,

$$\varphi_{ef} = \min\left\{\sum_h \bar{D}_{efh}u_{efh} : \boldsymbol{u}_{ef}\in X, u_{efe}=1, u_{eff}=1\right\}, \tag{7.38}$$

$$\rho_e = \min\left\{\sum_f (\bar{C}_{ef} + \varphi_{ef}) y_{ef} : \boldsymbol{y}_e \in X, y_{ee} = 1\right\}. \tag{7.39}$$

The following theorem formally shows the decomposition framework of (7.35).

Theorem 7.4.2. *An optimal solution of problem* (7.35) *can be obtained by solving the following minimum spanning tree problem*

$$K + \min\left\{\sum_e (\bar{d}_e + \rho_e) x_e : \ x \in X\right\}. \tag{7.40}$$

Proof. The proof is a trivial extension of the proof of Theorem 7.2.1. □

In the following we present a dual-ascent algorithm similar to that used in [3] for the RLT2 representation of the QAP. The most important part of the algorithm is to use the meaning of reduced costs to readjusting $\overline{D}$, $\overline{C}$ and $\overline{d}$. The main steps of our dual-ascent algorithm are summarized as follows:

1. Initialization.
   Set $\overline{D}_{efh} = 0$ for all $e, f, h \in E, \overline{C}_{ef} = C_{ef}$ for all $e, f \in E, \overline{d}_e = d_e$ for all $e \in E$, $K = 0$, and an iteration counter $I = 0$.

2. Spreading $\overline{d}$ on $\overline{C}$.
   For each $e \in E$, spread the coefficient $\overline{d}_e$ on coefficients $\overline{C}_{ef}$ for all $f \in E, f \neq e$; i.e, $\overline{C}_{ef} = \overline{C}_{ef} + \frac{\overline{d}_e}{n-2}$. Then update $\overline{d}_e$ to 0 for each $e \in E$.

3. Spreading $\overline{C}$ on $\overline{D}$.
   For each $e, f \in E, e \neq f$, spread the coefficient $\overline{D}_{ef}$ on coefficients $\overline{D}_{efh}$ for all $h \in E, h \neq e, f$; i.e, $\overline{D}_{efh} = \overline{D}_{efh} + \frac{\overline{C}_{ef}}{n-3}$. Then update $\overline{C}_{ef}$ to 0 for each $e, f \in E, e \neq f$.

4. Solving the Lagrangian relaxation problem.
   Use Theorem 7.4.2 to solve (7.35) as follows:

$4-a)$ For a selected $e, f \in E, e \neq f$, change coefficients $\overline{D}_{efh} = (\overline{D}_{efh} + \overline{D}_{ehf} + \overline{D}_{feh} + \overline{D}_{fhe} + \overline{D}_{hef} + \overline{D}_{hfe})/6$ for each $h \in E, h \neq e, f$. Solve subproblem (7.38) and compute the reduced cost of edge h with respect to the optimal spanning tree containing the fixed edges e, f. The reduced cost $\overline{rD}_{efh}$ of any edge h not in the minimum spanning tree containing the fixed edges e, f, is the difference between the cost of h and the largest cost of any edge, different from edges e, f, in the cycle induced by h. Note that if the cycle contains only

Table 7.1 Comparison of different lower bounding approaches on TestSet OPvsym.

Instance			Gap(%)						CPU Time					
n	d(%)	Ub	AX	MR	NMR	OP	RLT1	RLT2	AX	MR	NMR	OP	RLT1	RLT2
6	100	16273.9	1.2	0.8	1.8	4.8	0.0	0.00	0	0	0	0	0	0
7	100	19625.7	1.0	1.1	2.2	5.33	0.0	0.0	0	0	0	0	0	0
8	100	27039.4	2.0	1.5	1.4	4.65	0.0	0.0	0	0	0	0	0	0
9	100	22769.9	0.7	1.1	1.1	2.6	0.0	0.0	0	0	0	0	0	0
10	100	25743.8	0.9	2.1	2.0	6.1	0.0	0.0	0	0	0	0	0	0
11	100	29325.6	1.0	2.1	1.6	4.5	0.0	0.1	0	0	0	0	0	0
12	100	32577.8	0.7	1.2	1.1	3.5	0.0	0.0	0	0	0	0	0	1
13	100	40488.5	1.0	1.1	1.0	3.6	0.0	1.0	0	0	0	0	0	2
14	100	44240.4	0.6	1.1	1.0	2.2	0.0	1.0	0	0	0	0	1	5
15	100	50821.6	0.7	1.1	0.9	2.3	0.0	0.0	0	0	0	1	1	7
16	100	41940.2	1.1	2.3	2.2	2.1	0.0	1.0	0	0	0	2	2	10
17	100	41819.0	1.0	0.8	0.7	1.2	0.0	1.0	0	0	0	2	2	12
18	100	46130.2	1.3	1.5	1.3	2.0	0.0	1.0	0	0	0	3	3	24
20	100	55326.2	0.9	1.1	1.0	1.8	0.0	0.0	0	0	0	5	7	42
30	100	78999.9	0.5	1.10	1.0	0.7	0.0	0.0	0	1	1	23	3	445
50	100	165419.6	0.5	0.5	0.4	0.3	0.0	-	2	17	18	366	371	-

h and the two fixed edges e, f, then the reduced cost of h set to infinity. Set $\overline{D}_{efh} = \overline{rD}_{efh}$ and increase $\overline{C}_{ef}$ by φ_{ef}.

$4-b$) For a selected $e \in E$, change the coefficients $\overline{C}_{ef} = (\overline{C}_{ef} + \overline{C}_{fe})/2$ for each $f \in E, f \neq e$. Solve subproblem (7.39) and compute the reduced cost of edge f with respect to the optimal spanning tree containing the fixed edges e. The reduced cost $\overline{rC}_{ef}$ of any edge f not in the minimum spanning tree containing the fixed edge e, is the difference between the cost of f and the largest cost of any edge, different from edges e, in the cycle induced by f. Set $\overline{C}_{ef} = \overline{rC}_{ef}$ and increase $\overline{d}_e$ by ρ_e.

$4-c$) Solve problem (7.40) and compute the reduced cost of edge e with respect to the optimal spanning tree, readjusting $\overline{d}_e$ and increasing the scalar K by the weight of the minimum spanning tree.

4. If $I \geq MaxIteration$, stop; otherwise set $I = I + 1$ and return to 2.

In our implementation, each spanning tree problem is solved with Prim's algorithm [102]. Moreover, we start steps $4-a$, and $4-b$ with edges having reduced costs equal to zero, to increase the possibility of obtaining a tight lower bound.

7.5 Computational experiments

In this section we present our computational experiments on the lower bound computations for the QMSTP. We implemented the algorithms in C++ language and run on an

Table 7.2 Comparison of different lower bounding approaches on TestSet OPesym.

Instance			Gap(%)						CPU Time					
n	d(%)	Ub	AX	MR	NMR	OP	RLT1	RLT2	AX	MR	NMR	OP	RLT1	RLT2
6	100	541.2	2.9	2.6	3.2	3.0	0.4	0.5	0	0	0	0	0	0
7	100	783.7	4.8	4.5	4.2	4.7	0.3	1.2	0	0	0	0	0	0
8	100	1020.1	5.2	5.7	5.3	4.4	0.4	1.9	0	0	0	0	0	0
9	100	1356	6.5	5.5	4.8	5.6	0.6	2.2	0	0	0	0	0	0
10	100	1427.1	6.7	5.7	5.2	3.5	0.5	2.3	0	0	0	0	0	0
11	100	1545.1	6.4	5.6	4.7	6.0	0.3	2.4	0	0	0	0	0	0
12	100	1901.6	8.0	6.7	5.5	6.5	0.4	2.8	0	0	0	0	1	2
13	100	2175.3	7.9	6.6	5.7	6.8	0.3	2.9	0	0	0	0	1	4
14	100	2527.9	8.2	6.2	5.4	6.8	0.2	2.9	0	0	0	1	2	6
15	100	2588.8	8.2	6.5	5.6	7.8	0.4	2.9	0	0	0	1	3	11
16	100	2980.1	9.2	7.6	6.4	8.6	0.6	3.3	0	0	0	1	4	19
17	100	3372.2	9.8	7.8	6.9	8.2	0.9	3.7	0	0	0	2	5	24
18	100	3569.0	10.7	7.8	6.8	8.7	0.5	3.9	0	0	0	3	6	34
30	100	8056.7	13.2	11.0	9.6	12.1	1.7	6.3	1	4	9	21	59	493
50	100	15788.8	14.9	12.1	10.5	13.7	1.91	-	9	37	93	392	658	-

Intel Xeon CPU E5335 (2 quad core CPUs 2GH). In the following we first present the test instances and provide the comparison of the lower bonding approaches in details.

7.5.1 Test instances

We considered the benchmark sets CP and OP introduced in [33] and [94], respectively. The OP consists of three classes of random instances denoted as OPsym, OPvsym, and OPesym with size ranging from $n = 6$ to $n = 18$, $n = 20$, $n = 30$, and $n = 50$. This benchmark consists of 450 instances of complete graphs with n vertices and $m = n(n-1)/2$ edges. For each problem size, 10 random instances have been generated. The CP consists of four classes of random instances denoted as CP1, CP2, CP3, and CP4 with size $n \in \{10, 15, 20, 25, 30, 35, 40, 45, 50\}$. This benchmark consists of 108 instances with different densities $d \in \{33\%, 67\%, 100\%\}$. These two benchmarks have been generated as follows:

1. OPsym: The linear and the quadratic costs are chosen at random according to uniformly distributed from the sets $\{1, \ldots, 100\}$ and $\{1, \ldots, 20\}$, respectively.
2. OPvsym: The linear costs are uniformly distributed at random in $\{1, \ldots, 10000\}$ while the quadratic costs C_{ef} are obtained associating to the vertices random values uniformly distributed in $\{1, \ldots, 10\}$ and multiplying the four values associated to the end vertices of edges e and f, i.e., for two edges $e_p = (1, 2)$ and $e_f = (3, 4)$ the value of C_{ef} is computed by multiplying the weight of vertices $1, 2, 3$, and 4.
3. OPesym: The vertices uniformly have spread at random in a rectangle with coordinates $(0, 0), (0, 100), (100, 0)$ and $(100, 100)$. The linear costs are chosen as

Table 7.3 Comparison of different lower bounding approaches on TestSet OPsym.

Instance			Gap(%)						CPU Time					
n	d(%)	Ub	AX	MR	NMR	OP	RLT1	RLT2	AX	MR	NMR	OP	RLT1	RLT2
6	100	258.4	13.2	14.8	13.1	25.6	1.92	2551.4	0	0	0	0	0	0
7	100	326.8	24.3	22.1	19.0	38.1	5.3	2.9	0	0	0	0	0	0
8	100	438.5	24.2	25.7	22.5	39.5	9.7	5.8	0	0	0	0	0	0
9	100	534.9	31.5	33.9	28.3	44.2	14.7	7.7	0	0	0	0	0	0
10	100	653.9	41.4	45.5	39.3	57.9	23.4	11.89	0	0	0	0	0	1
11	100	785.9	55.5	57.8	51.3	71.8	34.5	17.6	0	0	0	0	1	2
12	100	918.5	64.9	69.5	60.5	84.4	42.0	21.6	0	0	0	0	1	3
13	100	1067.1	75.9	79.9	71.9	92.2	51.0	25.7	0	0	0	0	2	5
14	100	1249.8	82.0	85.0	77.1	98.4	56.4	29.0	0	0	0	1	2	8
15	100	1390.2	101.2	104.8	94.6	118.9	70.4	34.7	0	0	0	1	3	12
16	100	1629.3	115.4	119.6	110.6	134.7	83.7	28.2	0	0	0	2	4	20
17	100	1823.8	123.2	127.1	116.5	142.0	89.6	44.9	0	0	0	2	5	26
18	100	2981	137.3	142.5	131.8	157.3	103.2	49.2	0	0	0	3	6	37
20	100	2572.7	167.9	173.2	160.3	189.5	127.1	59.9	0	0	0	8	10	70
30	100	6015.9	329.9	332.2	315.7	356.3	261.5	111.6	0	2	6	32	59	503
50	100	17616.9	774.2	719.9	699.6	793.2	618.1	-	3	37	76	584	749	-

the Euclidean distances between the end vertices of each edge, while the quadratic costs are selected as the Euclidean distances between the midpoints of the edges.

4. CP1: The linear and the quadratic costs are chosen at random according to uniformly distributed from the set $\{1, \dots, 10\}$.

5. CP2: The linear and the quadratic costs are chosen at random according to uniformly distributed from the sets $\{1, \dots, 10\}$ and $\{1, \dots, 100\}$, respectively.

6. CP3: The linear and the quadratic costs are chosen at random according to uniformly distributed from the sets $\{1, \dots, 100\}$ and $\{1, \dots, 10\}$, respectively.

7. CP4: The linear and the quadratic costs are chosen at random according to uniformly distributed from the set $\{1, \dots, 100\}$.

7.5.2 Lower bound computation

We compare the lower bounds effectiveness and computational efficiency using six alternative approaches discussed in this paper. Due to the tradeoff between bound strength and CPU execution time we terminated the dual-ascent algorithms for RLT2 in 20 iterations for $n < 30$, and 10 iterations for $n \geq 30$. Moreover, due to the exceeded memory limit we are not able to solve the RLT2 for instances with size $n \geq 40$. In subgradient implementation, we allow 1500 iterations with an initial step size of 2.

Tables 7.1 to 7.3 present the percent gap between the lower bounds and the objective value of the best known solutions, and CPU execution times of using different approaches

Table 7.4 Comparison of different lower bounding approaches on TestSet CP1.

Instance			Gap(%)						CPU Time					
n	d(%)	Ub	AX	MR	NMR	OP	RLT1	RLT2	AX	MR	NMR	OP	RLT1	RLT2
10	33	350	12.2	5.1	5.1	7.4	0.0	1.7	0	0	0	1	1	1
10	67	255	49.1	42.5	37.1	44.9	25.6	12.8	0	0	0	1	1	1
10	100	239	67.1	64.8	60.4	64.8	48.4	20.1	0.0	0	0	1	1	1
Ave.		281.3	42.8	37.4	32.5	39.1	24.6	11.5	0.0	0.0	0.0	1.0	1.0	1.0
15	33	745	46.6	42.2	38.2	41.1	28.4	16.8	0	0	0	2	30	13
15	67	659	95.0	87.8	82.0	86.7	71.2	34.8	0	0	0	2	3	13
15	100	620	112.3	104.6	102.6	105.3	93.2	40.0	0	0	0	2	3	13
Ave.		674.6	84.6	78.2	74.2	77.7	64.2	30.5	0.0	0.0	0.0	2.0	3.0	13.0
20	33	1379	76.3	71.3	65.6	69.2	55.3	30.5	0	0	1	7	10	78
20	67	1252	132.3	126.8	120.1	120.8	107.6	48.5	0	0	1	6	10	82
20	100	1174	154.7	145.6	141.1	141.1	131.6	59.1	0	0	1	8	10	78
Ave.		1268.3	121.1	114.5	108.9	110.3	98.1	46.0	0.0	0.0	1.0	7.0	10.0	79.3
25	33	2185	95.8	87.1	82.1	83.5	70.1	37.0	0	1	1	15	26	328
25	67	2023	167.6	157.7	152.9	153.5	142.3	63.2	0	1	1	18	27	334
25	100	1943	190.4	182.4	178.8	176.8	170.2	78.1	0	1	1	18	28	332
Ave.		2050.3	151.2	142.4	137.9	137.9	127.5	59.4	0.0	1.0	1.0	17.0	27.0	331.3
30	33	3205	124.1	115.2	110.7	111.4	98.1	49.1	0	3	3	33	54	544
30	67	2998	193.1	182.5	177.8	176.5	168.1	78.3	0	3	3	42	61	521
30	100	2874	208.3	201.5	199.1	195.3	191.4	92.2	0	3	3	46	68	491
Ave.		3025.6	175.1	166.4	162.5	166.1	152.5	73.2	0.0	3.0	3.0	40.3	61.0	518.6
35	33	4474	149.6	139.7	134.9	134.2	122.1	59.1	1	8	8	74	111	1475
35	67	4147	211.5	200.2	198.1	196.0	190.4	87.8	1	6	8	95	118	1393
35	100	4000	222.5	216.9	214.9	210.8	206.8	104.8	1	6	8	92	143	1393
Ave.		4207.0	194.5	185.5	182,6	180.3	174.7	83.9	1.0	6.7	8.0	87.0	124.0	1420.3
40	33	5945	173.3	162.1	156.8	155.4	143.8	-	1	15	16	142	200	-
40	67	5567	229.6	220.6	217.7	214.7	210.0	-	1	12	12	163	244	-
40	100	5368	235.7	231.1	229.7	225.5	222.1	-	1	14	18	189	268	-
Ave.		5626.6	212.8	204.6	201.4	198.5	191.9	-	1.0	3.6	15.3	164.6	237.3	-
45	33	7521	188.9	177.1	172.1	170.8	161.2	-	2	21	26	279	329	-
45	67	7161	241.4	235.1	232.3	228.6	225.2	-	2	22	33	307	395	-
45	100	6944	244.7	241.2	239.7	236.2	234.5	-	2	24	37	351	489	-
Ave.		7208.6	225.0	217.8	214.7	211.8	206.9	-	2.0	22.3	32.0	312.3	404.3	-
50	33	9393	207.9	194.8	190.0	188.3	179.7	-	2	36	51	501	534	-
50	67	8958	251.2	245.6	243.6	239.4	236.8	-	3	42	60	563	712	-
50	100	8713	252.1	248.6	247.5	243.9	243.4	-	3	46	66	649	835	-
Ave.		9021.3	237.1	229.6	227.0	223.8	219.9	-	2.6	41.3	59.0	571.0	693.6	-

Table 7.5 Comparison of different lower bounding approaches on TestSet CP2.

Instance			Gap(%)						CPU Time					
n	d(%)	Ub	AX	MR	NMR	OP	RLT1	RLT2	AX	MR	NMR	OP	RLT1	RLT2
10	33	3122	17.2	7.2	7.3	9.2	0.0	2.5	0	0	0	1	1	1
10	67	2042	91.7	78.1	67.5	81.3	45.9	19.4	0	0	0	1	1	1
10	100	1815	137.2	125.7	116.1	127.1	93.7	31.1	0	0	0	1	1	1
Ave.		2326.3	82.0	70.3	63.6	72.5	46.5	17.6	0.0	0.0	0.0	1.0	1.0	1.0
15	33	6539	67.9	59.5	54.2	59.1	39.6	22.7	0	0	0	2	3	13
15	67	5573	170.1	146.1	140.0	148.0	115.2	48.2	0	0	0	2	3	13
15	100	5184	243.5	225.2	210.9	222.9	183.3	60.2	0	0	0	2	3	13
Ave.		5765.3	160.5	143.6	135.0	143.3	112.7	43.7	0.0	0.0	0.0	2.0	3.0	13.0
20	33	12425	113.2	106.4	93.7	100.7	76.6	40.4	0	0	1	6	10	81
20	67	10893	247.4	225.7	214.1	218.4	185.4	65.7	0	0	1	7	10	83
20	100	10215	356.0	321.9	307.4	313.0	271.7	87.7	0	0	1	7	9	83
Ave.		11177.6	238.8	217.6	204.9	210.7	177.9	64.4	0.0	0.0	1.0	6.7	9.7	82.3
25	33	19976	144.4	127.5	119.2	122.5	98.4	48.4	0	1	2	16	24	326
25	67	18251	339.2	307.2	297.1	300.3	261.1	88.8	0	1	2	20	24	314
25	100	17411	504.5	473.4	446.4	451.8	401.9	119.8	0	1	2	20	25	302
Ave.		11177.6	329.3	302.7	287.4	291.5	253.8	85.6	0.0	1.0	2.0	18.6	24.3	314.0
30	33	29731	198.2	184.5	171.2	174.3	146.8	65.7	0	5	6	36	54	507
30	67	27581	427.7	389.7	374.9	377.3	336.4	114.8	0	8	9	47	56	528
30	100	26146	616.7	570.5	550.4	552.5	503.5	146.1	0	6	6	50	66	490
Ave.		27819.3	414.2	381.5	365.5	368.1	328.9	108.8	0.0	6.3	7.0	44.3	58.7	508.3
35	33	42305	248.1	221.3	214.0	215.2	188.1	79.5	0	19	21	77	122	1421
35	67	38490	522.0	474.1	462.6	464.7	419.5	136.5	0	11	14	97	114	1394
35	100	36723	728.7	669.8	653.7	653.4	600.6	168.3	0	10	13	102	117	1394
Ave.		39172.7	499.3	454.7	443.2	444.2	402.3	128.1	0.0	13.3	16.0	92.0	117.6	1403.0
40	33	56237	300.4	275.0	260.3	260.4	229.6	-	1	30	40	150	214	-
40	67	51851	610.6	567.4	542.9	542.6	496.1	-	1	22	30	183	205	-
40	100	49817	861.3	798.9	775.9	774.2	717.4	-	1	22	25	192	232	-
Ave.		52635.0	590.7	546.6	526.2	525.6	480.8	-	1.0	24.7	31.7	175.0	217.0	-
45	33	70603	340.5	311.0	297.1	296.7	266.2	-	1	40	57	294	348	-
45	67	66889	691.8	639.4	619.2	618.0	570.3	-	1	34	46	341	374	-
45	100	64840	974.7	896.0	878.1	875.9	817.8	-	1	35	45	377	382	-
Ave.		67444.0	668.9	615.4	598.1	596.8	551.4	-	1.0	36.3	49.3	337.3	368.0	-
50	33	88942	394.2	363.0	345.5	345.1	310.8	-	2	80	75	516	543	-
50	67	84020	783.3	721.0	702.6	699.3	649.9	-	2	66	81	621	614	-
50	100	81858	1090.8	1008.1	987.3	983.0	925.0	-	2	51	72	611	656	-
Ave.		84940.0	756.1	697.3	678.4	675.8	628.5	-	2.0	65.7	76.0	582.7	604.3	-

Table 7.6 Comparison of different lower bounding approaches on TestSet CP3.

Instance			Gap(%)						CPU Time					
n	d(%)	Ub	AX	MR	NMR	OP	RLT1	RLT2	AX	MR	NMR	OP	RLT1	RLT2
10	33	646	0.6	1.8	1.8	3.1	0.0	0.0	0	0	0	1	1	1
10	67	488	11.4	14.2	10.1	22.6	0.0	2.5	0	0	0	1	1	1
10	100	426	18.9	24.2	20.6	35.2	10.3	6.2	0	0	0	1	1	1
Ave.		520.7	10.3	13.4	10.8	20.3	3.4	2.9	0.0	0.0	0.0	1.0	1.0	1.0
15	33	1236	14.0	19.4	14.8	23.9	4.7	5.1	0	0	0	2	4	13
15	67	966	25.7	32.3	24.6	37.0	13.9	9.0	0	0	0	2	4	13
15	100	975	36.1	41.3	36.5	50.4	25.0	14.8	0	0	0	2	3	13
Ave.		1059.0	25.2	31.0	25.3	37.1	14.5	9.6	0.0	0.0	0.0	2.0	3.7	13.0
20	33	1972	29.5	33.4	27.8	38.0	17.9	12.1	0	1	1	6	9	86
20	67	1792	50.9	56.6	48.8	62.7	37.1	21.9	0	1	1	6	10	80
20	100	1544	62.3	61.0	57.0	74.0	46.2	26.5	0	1	1	6	10	81
Ave.		1769.3	47.5	50.3	44.5	58.2	33.7	20.1	0.0	1.0	1.0	6.0	10.0	82.3
25	33	2976	43.7	47.3	42.3	51.9	30.0	19.6	0	1	1	14	28	323
25	67	2546	73.3	75.1	70.0	84.2	56.2	32.8	0	2	2	16	26	324
25	100	2471	93.0	88.7	85.6	104.0	75.3	42.0	0	2	2	15	28	305
Ave.		2664.3	70.0	70.3	65.9	80.0	53.8	31.4	0.0	1.7	1.7	15.0	27.3	319.3
30	33	4070	56.1	59.8	54.5	61.1	42.5	25.8	0	3	4	33	57	507
30	67	3649	95.8	94.7	89.6	104.6	77.7	45.0	0	4	5	34	66	502
30	100	3483	114.8	109.3	105.9	122.9	96.1	54.3	0	4	5	42	68	490
Ave.		3734.0	88.9	87.9	83.3	96.2	72.1	41.7	0.0	3.7	4.7	36.3	63.7	499.7
35	33	5423	78.9	84.8	76.6	84.9	61.4	37.4	0	8	12	69	125	1416
35	67	4981	116.4	112.6	109.2	126.2	98.0	55.9	0	8	12	81	126	1482
35	100	4770	138.0	130.8	127.0	144.1	117.8	66.4	0	8	11	92	146	1457
Ave.		5058.0	111.1	109.4	104.2	118.4	92.4	53.2	0.0	8.0	11.7	80.7	132.3	1451.7
40	33	6925	94.9	95.9	91.2	99.1	76.1	-	1	17	23	135	220	-
40	67	6456	133.6	127.7	123.9	140.0	113.3	-	1	19	25	152	272	-
40	100	6208	158.1	151.7	145.6	159.5	135.9	-	1	17	23	171	272	-
Ave.		6529.7	128.8	125.1	120.2	133.0	108.4	-	1.0	17.7	23.7	152.7	254.7	-
45	33	8720	108.3	110.0	103.5	111.3	89.2	-	1	35	45	285	410	-
45	67	8225	150.6	144.5	139.8	154.1	130.0	-	1	30	45	284	486	-
45	100	7827	171.9	163.6	158.7	170.0	149.9	-	1	30	45	331	502	-
Ave.		8257.3	143.6	139.3	134.0	145.1	122.7	-	1.0	31.7	45.0	300.0	466.0	-
50	33	10717	124.2	122.8	117.7	126.5	103.9	-	2	72	76	421	664	-
50	67	10100	167.9	159.5	155.2	168.6	145.5	-	2	58	81	509	816	-
50	100	9836	191.3	182.8	176.6	186.2	167.5	-	2	53	82	593	873	-
Ave.		10217.7	161.1	155.0	149.8	160.4	138.9	-	2.0	61.0	79.7	507.7	784.3	-

Table 7.7 Comparison of different lower bounding approaches on TestSet CP4.

Instance			Gap(%)						CPU Time					
n	d(%)	Ub	AX	MR	NMR	OP	RLT1	RLT2	AX	MR	NMR	OP	RLT1	RLT2
10	33	3486	14.3	6.1	6.0	8.7	0.0	1.8	0	0	0	1	1	1
10	67	2404	67.4	58.8	49.5	60.8	34.0	15.8	0	0	0	1	1	1
10	100	2197	93.2	87.6	83.5	91.2	66.3	26.0	0	0	0	1	1	1
Ave.		2695.7	58.3	50.8	46.3	53.5	33.4	14.5	0.0	0.0	0.0	1.0	1.0	1.0
15	33	7245	58.5	52.1	47.2	50.9	35.3	20.4	0	0	0	2	3	13
15	67	6188	130.8	119.0	111.4	119.5	92.5	42.6	0	0	0	2	3	13
15	100	5879	172.8	160.5	154.7	162.6	136.1	52.1	0	0	0	2	4	13
Ave.		6437.3	120.7	110.5	104.5	111.0	88.1	38.3	0.0	0.0	0.0	2.0	3.3	13.0
20	33	13288	98.3	90.7	82.3	88.5	67.9	36.4	0	1	1	7	10	78
20	67	11893	198.7	187.3	176.2	180.0	153.4	59.8	0	1	1	7	10	83
20	100	11101	265.2	246.5	237.7	243.3	210.6	78.5	0	1	1	8	11	83
Ave.		12094.0	187.4	174.8	165.4	170.6	143.9	58.2	0.0	1.0	1.0	7.7	10.3	81.3
25	33	21176	127.7	114.1	107.1	110.0	89.3	45.2	0	4	3	18	26	337
25	67	19207	270.5	251.9	242.3	245.8	215.1	80.3	0	2	3	18	27	325
25	100	18370	375.1	359.0	342.6	350.2	309.1	107.0	0	3	3	20	26	350
Ave.		19584.3	257.7	241.6	230.6	235.3	204.5	77.5	0.0	3.0	3.0	18.7	26.3	337.3
30	33	31077	174.3	161.6	152.4	154.6	131.9	61.2	0	5	7	37	56	537
30	67	28777	346.9	326.0	311.1	313.9	280.3	105.3	0	8	7	45	55	504
30	100	27314	465.5	442.9	427.2	433.1	391.7	131.4	0	6	6	46	61	534
Ave.		29056.0	328.9	310.1	296.9	300.5	267.9	99.3	0.0	6.3	6.7	42.7	57.3	525.0
35	33	43629	220.2	200.3	192.3	193.5	169.8	74.6	0	18	21	86	108	1393
35	67	39660	419.7	389.1	381.8	384.8	346.4	120.7	0	12	15	88	113	1394
35	100	38049	552.4	525.8	509.3	514.5	467.3	153.0	0	12	16	93	122	1394
Ave.		40446.0	397.4	371.7	361.1	364.6	327.8	116.1	0 .0	14.0	17.3	89.0	114.3	1393.7
40	33	58874	274.9	254.5	241.4	241.4	214.5	-	1	40	35	158	194	-
40	67	53592	498.6	469.7	453.4	455.1	415.5	-	1	22	29	212	209	-
40	100	51229	652.5	626.6	605.2	609.1	558.1	-	1	22	28	197	232	-
Ave.		54565.0	475.3	450.2	433.3	435.2	396.1	-	1.0	28.0	30.6	189.0	211.7	-
45	33	72676	305.3	282.1	269.7	269.7	242.8	-	1	43	46	286	354	-
45	67	68737	565.0	534.1	516.7	516.9	477.6	-	1	35	52	355	369	-
45	100	66508	744.7	707.1	690.2	692.9	640.3	-	1	39	57	366	398	-
Ave.		69307.0	538.3	507.7	492.2	493.1	453.5	-	1.0	39.0	51.7	335.7	373.7	-
50	33	91009	353.4	328.2	313.4	313.4	282.9	-	2	80	75	541	582	-
50	67	86231	642.8	608.5	588.8	588.4	546.0	-	2	69	80	612	606	-
50	100	83838	838.1	796.5	779.3	781.7	726.7	-	2	69	80	657	690	-
Ave.		87026.0	611.4	577.7	560.5	561.1	518.5	-	2.0	72.7	78.3	603.3	626.0	-

for OPvsym, OPesym and OPsym, respectively. Each row of the Tables 7.1 to 7.3 gives the average values of the respective 10 randomly generated test instances. Tables 7.4 to 7.7 report the results for CP1 to CP4 in terms of the gap and CPU execution times. In all tables the first three columns indicate the problem size (n), the density (d), and the objective value of the best known solutions (Ub) obtained from [33]. The next six columns from left to right, give the percent gaps obtained using the Assad and Xu's procedure (AX), the Monotonic Reformulation (MR) scheme, the Non Monotonic Reformulation (NMR) scheme, the revised form of the Öncan and Punnen Lagrangian relaxation approach by considering the correct formula (7.18) and (7.19) (OP), the subgradient implementation of QRLT1 (RLT1), and the dual-ascent implementation of QRLT2 (RLT2). The CPU execution times of each approach are given in the last six columns of the tables. The formula we used to compute the percent gaps is $100 \times (Ub - Lb)/Lb$, where Lb stands for the value of the lower bound.

As we can observe in tables 7.1 to 7.3, the bounds obtained by NMR are almost always stronger than the AX, MR, and OP bounds for all three data sets, but still weaker than the RLT1 and RLT2. For the OPVSYM, which seems to be the easiest test set among the OP instances, the RLT1 and RLT2 almost always give the optimal solutions with more CPU execution times. For this data set, the AX, MR, NMR and OP approaches provide tight bounds with short execution times with the exception of OP whose computational times for instances with $n = 50$ are high. For the OPESYM, which seems to be more difficult than the OPVSYM, the best approach is the RLT1 and the worst one is the AX. For this data set NMR provides good bounds in reasonable times, and always outperform the AX, MR and OP. The RLT2 provides the bounds slightly worse than RLT1 and requires more computational effort. For the OPSYM which seems to be the most difficult data set amongst the OP test set, the RLT2 provides lower bounds that are significantly stronger than those provided by the other approaches (especially those by RLT1.) For this test set, the bound obtained by NMR are still stronger than the AX, MR, and OP. Note that, the revised form of the Öncan and Punnen Lagrangian relaxation approach does not always yield tighter lower bound values for all the data sets than the ones obtained by the Assad and Xu's leveling procedure, as claimed in their paper [94].

The results of comparing different lower bounding approaches for the CP data set are given in Tables 7.4 to 7.7. For all four CP data sets, the RLT2 outperforms the other approaches in terms of the bound tightness, but it always needs more computational effort. After the RLT2, the RLT1 is the one that provides the best bounds. Considering the significant gaps obtained for this data set, we can conclude that the NMR, OP, and RLT1 almost provide the same bounds, but NMR outperforms the other two in terms of execution times. In order to give a better explanation of the comparison we reported the average results over the 12 different instance of each dimension in Table 7.8. Overall,

Table 7.8 Comparison of different lower bounding approaches on TestSet CP. Each row reports the average on 12 instances.

Instance		Gap(%)						CPU Time					
n	Ub	AX	MR	NMR	OP	RLT1	RLT2	AX	MR	NMR	OP	RLT1	RLT2
10	1455.8	48.3	43.0	38.7	46.3	27.0	11.6	0.0	0.0	0.0	1.0	1.0	1.0
15	3484.1	97.8	90.8	84.8	92.3	69.8	30.5	0.0	0.0	0.0	2.0	3.2	13.0
20	6577.3	148.7	139.4	131.0	137.5	113.4	47.3	0.0	1.0	1.0	6.7	10.0	81.3
25	10711.2	202.1	189.3	180.5	186.2	159.9	63.5	0.0	1.7	1.9	17.3	26.2	325.0
30	15908.7	251.8	236.5	227.1	231.5	205.4	80.8	0.0	4.8	5.3	40.9	60.2	512.9
35	22220.9	300.7	280.5	272.8	276.9	249.1	95.0	0.3	10.5	13.2	87.2	122.1	1417.2
40	29839.1	363.0	331.7	320.3	323.1	294.4	-	1.0	21.0	24.8	170.3	230.2	-
45	38054.2	394.04	370.1	359.7	361.8	333.7	-	1.3	32.3	44.5	321.3	403.0	-
50	47801.2	441.4	414.9	404.0	405.3	376.5	-	2.2	60.2	73.2	506.2	631.5	-

the table indicates that the dual-ascent strategy applied for RLT2 provides a significant improvement over all the other approaches reducing the gaps between the linear programming and the best known integer solution values. For the size $n = 35$, which was the biggest size that we could solve by RLT2, the overall gap reduction of the RLT2 over the AX is 68%, while the overall gap reduction of RLT1, NMR, OP, and MR over the AX are 17%, 9.3%, 8%, and 6.6%, respectively. From these results we can conclude that the RLT2 yields the tightest bound but in the cost of increased CPU execution time. Moreover, the NMR seems to be a good candidate when the tradeoff between the bounds tightness and the CPU execution times matters.

7.6 Conclusion

In this chapter we study the QMSTP. We reviewed and analyzed the Gilmore-Lawler type bound, the Assad and Xu reformulation scheme, and the Lagrangian relaxation of Öncan and Punnen, and compared them in terms of continuous relaxation of a proposed MILP. To improve the bounds for the QMSTP, we develop a mixed 0-1 linear formulation based on using the second level of the reformulation-linearization technique and devised an efficient dual-ascent algorithm to solve the continuous relaxation of the proposed model based on reduced costs. The basic idea proposed in the paper is to use the meaning of the reduced cost to retrieve the dual information of the continuous relaxation of the MILP. In Computational experiments we compared the tightness and speed of computation of the new bounding techniques with those of literature. The results indicated that the dual-ascent procedure applied to QRLT2 provides the best bounds at the price of large computational effort, while the bound obtained by the non-monotonic reformulation scheme seems to tradeoff between the bound tightness and computational effort.

Chapter 8

Quadratic Shortest Path Problem

Finding the shortest path in a directed graph is one of the most important optimization problems which has applications in a wide range of fields. In its basic version, however, the problem fails to represent situations in which value of the objective function is determined not only by the choice of each single arc, but also by the combined presence of pairs of arcs in the solution. In this chapter we model these situations as a Quadratic Shortest Path Problem (QSPP) which calls for the minimization of a quadratic objective function subject to shortest-path constraints. We first show that the problem is NP-hard. Then we propose and evaluate bounding techniques which exploit the structure of the problem. Finally, we present the possible polynomially solvable cases of the problem.

8.1 Introduction

Shortest Path Problem (SPP) is among the most studied optimization problems in graph theory. They arise frequently in practice in a variety of settings and often appear as subproblems in many combinatorial optimization algorithms. Given a directed graph $G(N, A)$, where N is the set of nodes and $A \subset N \times N$ is the set of directed arcs, let $c : A \to \mathbb{R}^+$ be a cost function associated to each arc. The SPP problem calls for finding a path of minimal cost from a source node $s \in N$ to a target node $t \in N$.

The SPP can be solved efficiently in polynomial time as long as the graph does not contain a negative cost cycle, i.e., a cycle in which the sum of the arc costs is less than zero. Fore example, Dijkstra's algorithm [35] takes $O(|A| \ln |N|)$ time to provide a shortest-path tree from a source node s to all other nodes. Even better, Thorup's algorithm [117] in undirected graphs with integer weights, finds a path between a source and a target node in $O(|A|)$ time and space.

The basic SPP fails to model situations in which the value of a linear objective function is not the only interesting parameter in the choice of the optimal solution. Such problems include situations in which the choice of the shortest path is constrained by parameters such as the variance of the cost of the path, or cases in which the objective function takes into account not only the cost of each selected arc but also the cost of the interactions among the arcs in the solution.

In this chapter we study a variant of the SPP that has a quadratic cost function, the so-called Quadratic SPP (QSPP). In the QSPP, the value to optimize is a combination of a linear and a quadratic components, i.e., the cost of the path is determined not only by the presence of each single arc in the solution, but also by the combined presence of pairs of arcs. The input of this problem is a directed graph $G(N, A)$, a source node $s \in N$, a target node $t \in N$, a cost function $c : A \to \mathbb{R}^+$, which maps every arc to a non-negative cost, and a cost function $q : A \times A \to \mathbb{R}^+$ that maps every pair of arcs to a non-negative real cost. The QSPP consists in finding a path from node s to node t with minimum overall cost.

8.2 Motivations and Related work

The shortest-path problem is important in numerous application and research areas including operation research, transportation sciences, artificial intelligence, communication networks and many others. Even harder problems that cannot be expressed as a genuine shortest path often involve solving a SPP sub-problem. Nevertheless, its inherent simplicity does not allow the SPP to address more complex situations in which the "length" of the path is not the only variable of interest.

The first shortest path problem that is directly related to QSPP is probably that of Variance Constrained Shortest Path [113]. The problem seeks to locate the path with the minimum expected cost subject to the constraint that the variance of the cost is less than a specified threshold. The problem arises for example in the transportation of hazardous materials. In such cases a path must be short but it must also be subject to a constraint that the variance of the risk associated to the route is less than a specified threshold. In addition the problem may arise in all situations in which the costs associated to each arc consist of stochastic variables. Possible approaches to solving the Variance Constrained Shortest Path problem involve a relaxation in which the quadratic variance constraint is incorporated in the objective function yielding a QSPP.

A different body of related work comes from research on network protocols. In [91], authors study different restoration schemes for self-healing ATM networks. In particular, the authors examine line and end-to-end restoration schemes. In the former, link failures are addressed by routing traffic around failed link, in the latter instead traffic is rerouted by computing an alternate path between source and destination. Within their analysis, the authors point out the need to solve a QSPP to address rerouting in the latter scheme. Nevertheless, they do not provide details about the algorithm used to obtain a QSPP solution.

A more recent work [110] has instead tackled the problem of stochastic costs from a slightly different perspective. The authors develop a multi-objective model for route planning to minimize both the expected travel time of a path and its variance. The problem arises in a route planning system which allows travelers to choose not only the shortest route but also routes with minimal variance with respect to their expected travel times. The paper solves the multi-objective optimization problem by combining the linear and quadratic objective function into a single quadratic shortest path problem. Specifically, the authors model the travel time associated to the arcs, $a \in A$, in a graph representing a road network as a multivariate normal random variable, with possible correlation between travel times on different arcs. The model is represented by a vector of mean travel times c and a covariance matrix Q representing the correlation between travel times on different links. Based on this model, the cost of a path between two nodes is also represented by a random variable where mean and variance are obtained by combining the mean and variance of the travel times of each arc in the path. Given a path, let $x_a \in \{0, 1\}$ be a binary variable such that $x_a = 1$ if and only if arc a belongs to the path. Then the mean and variance of the path's travel time are expressed by the following two equations.

$$E(x) = c^T x = \sum_{a \in A} c_a x_a$$

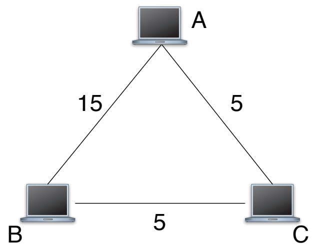
$$V(x) = x^T Q x = \sum_{a,b \in A} q_{ab} x_a x_b$$

The multi-objective optimization model can then be solved by optimizing some combination of mean and variance, which lead to a "shortest path problem" whose objective function is a combination of a linear and a quadratic component.

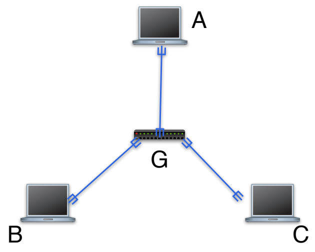
$$\min \quad E(x) + \alpha V(x) = \sum_{a \in A} c_a x_a + \alpha \sum_{a,b \in A} q_{ab} x_a x_b$$

The authors in [110] first solve a linear relaxation and then enumerate some selected paths. An interesting further development of our research is to compare the approach used by [110] to ours to identify reciprocal strengths and weaknesses.

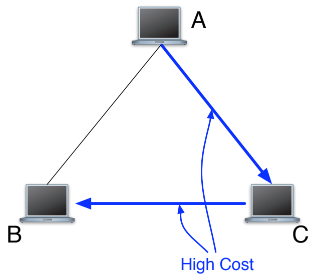
Another application of the QSPP is related to the design of self-repairing telecommu-



(a) Logical Topology



(b) Physical Topology



(c) Discouraged Pair

Fig. 8.1 A logical network and the underlying physical infrastructure.

nication networks. The growing importance of electronic communication in the modern society has fostered increasing interest in networks that are able to recovery automatically from failures in the underlying infrastructure, for example by rerouting traffic along alternative paths. These networks are often based on a logical network topology deployed on top of the underlying physical infrastructure. Communication paths are established on the logical topology, but they correspond to possibly more complex paths in the physical network. Let us consider the simple fully connected network depicted in Figure 8.1a. With the arc costs shown in the figure, the shortest path from node A to node B is the one going through node C in the physical topology of Figure 8.1b. Nevertheless, the choice of ACB as a path may be undesirable in that the path traverses the same gateway G and the same links - GC and CG - multiple times. Discouraging the choice of such pairs of links involves assigning a high cost to the selection of pairs of logical arcs corresponding to the same links (or nodes) in the physical network as shown in Figure 8.1c.

All the problem described above involve variants of the shortest-path problem in which the cost associated to each arc is integrated by a contribution associated to the presence of pairs of arcs in the solution. Such a contribution can be expressed by a quadratic objective function on binary variables associated to each arc, and leads to the definition of a QSPP.

8.3 Problem formulation and complexity

The scenarios described in Section 8.2 highlighted the significance of a quadratic extension to the shortest path problem and provided a motivation for the study presented in this chapter. In this section we present the problem formulation and discusses the complexity of the problem by proving its NP-hardness. We define the binary variable x_{ij} that is equal to 1 if arc (i,j) is on the minimum cost path from the source node s to the target node t, and 0 otherwise. The QSPP can be modeled as follows:

$$\text{QSPP:} \quad z^* = \min \quad \sum_i \sum_j \sum_k \sum_l q_{ijkl} x_{ij} x_{kl} + \sum_i \sum_j c_{ij} x_{ij} \tag{8.1}$$

$$\text{s.t.} \quad x \in X_{st}, \; x \text{ binary}. \tag{8.2}$$

where the feasible region, X_{st}, is exactly the same as that associated to the standard shortest-path problem and can be defined as follows.

$$\begin{aligned} X_{st} = \{0 \leq x \leq 1 : \sum_j x_{ij} + \sum_j x_{ji} = -1 \text{ if } i = s, \\ \sum_j x_{ij} + \sum_j x_{ji} = 1 \text{ if } i = t, \sum_j x_{ij} + \sum_j x_{ji} = 0 \; \forall i : i \neq s, t\}. \end{aligned} \tag{8.3}$$

Theorem 8.3.1. *QSPP is strongly NP-hard.*

Proof. Let us consider the general form of the QAP on bipartite graph $\mathcal{G} = (U, V, E)$ with nodes $U \cup V$, undirected arcs E, a linear cost c which maps every arc to a non-negative real cost, and a quadratic cost q which maps every pairs of arcs to a non-negative real cost. We show that this generic instance of the QAP can be reduced to a corresponding instance of QSPP in polynomial time. We define the QSPP instance on a graph $\tilde{G} = (\tilde{N}, \tilde{A})$ and map each feasible QAP assignment onto a feasible path in $\tilde{G}$. The sets $\tilde{N}$ and $\tilde{A}$ are defines as follows:

- The set of nodes $\tilde{N}$ consists of a source node s, target node t, and a node for each pair of nodes from U and V in the QAP, that is:

$$\tilde{N} = (U \times V) \cup \{s, t\}.$$

  Each node $n \in (U \times V)$ corresponds to an edge in the original QAP and is expressed

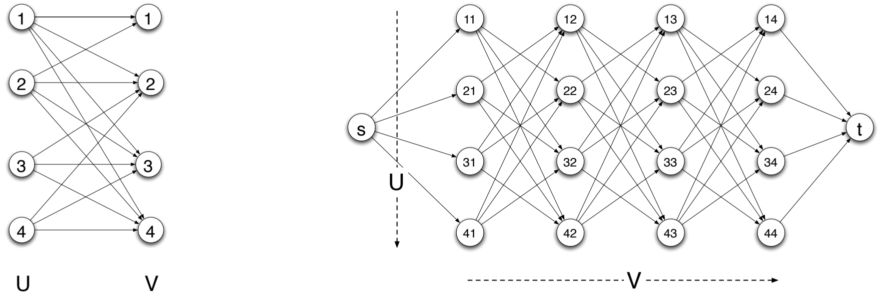


Fig. 8.2 The graphs G and $\tilde{G}$.

by the following two functions:

$$\text{Given a node } n = (i, j) : \quad \begin{aligned} \mathsf{U}(n) &= i \\ \mathsf{V}(n) &= j \end{aligned}$$

- The set of arcs $\tilde{A}$ is defined as $A_s \cup A \cup A_t$, where

$$A_s = \{(s, n_2) : n_2 = (i, 1)\}, A_t = \{(n_1, t) : n_1 = (i, |V|)\},$$

and

$$A = \{(n_1, n_2) : n_1 = (i, j), n_2 = (h, k), i \neq h, \ k = j + 1\}.$$

Figure 8.2 shows the graphs G and $\tilde{G}$ with $|V| = |U| = 4$. With reference to this figure, $\mathsf{U}(n)$ represents the row of node n in the grid arranged graph on the right. Moreover, it represents the index of the first of the two QAP nodes corresponding to n in the bipartite graph on the left ($i \in U$). Analogously, $\mathsf{V}(n)$ represents the column in the grid and the index of the second QAP node in the bipartite graph ($j \in V$).

It is worth observing that there are no arcs between nodes with the same U value as these would correspond to infeasible solutions for the original QAP instance, in that the same node in V would be assigned to two nodes in U.

The graph structure resulting from the above transformation has a number of nodes equal to $|U|^2 + 2$ and a number of arcs equal to $|U|^3 + 2|U|$. Each arc in $\tilde{G}$ corresponds to the arc of G determined by the values of the functions U and V on its target endpoint. This enables the mapping of a given assignment configuration in the original graph G onto a path from s to t in $\tilde{G}$. The first arc in the path determines the node in V which is assigned to the first node in U, the second arc in the path determines the node in V associated to the second node in U and so on. The last arc in the path simply terminates all possible paths with the target node t.

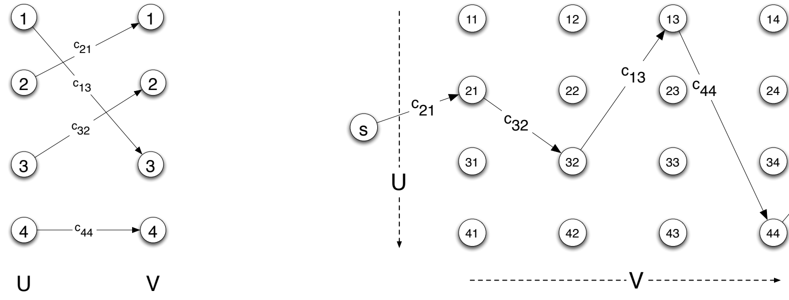


Fig. 8.3 Assignment of linear costs in the QAP to QSPP reduction.

This mechanism maps a given feasible assignment in G into a unique feasible path in $\tilde{G}$. Nevertheless, there are still some infeasible QAP assignments that have a corresponding feasible path in $\tilde{G}$. This apparent problem is easily addressed by correctly generating the cost matrix as we show next.

The linear cost vector is defined in Equation (8.4). The cost for an arc pointing to node j is given by the cost of the arc from $\mathsf{U}(j)$ to $\mathsf{V}(j)$ in the QAP.

$$\tilde{c}_{ij} = \begin{cases} c_{\mathsf{U}(j)\mathsf{V}(j)} & j \neq t \\ 0 & j = t. \end{cases} \tag{8.4}$$

The assignment of quadratic costs to pairs of arcs in $\tilde{G}$ is defined according to Equation (8.5). In general, the cost $\tilde{q}_{ijhk}$ is equal to the cost $q_{\mathsf{U}(j)\mathsf{V}(j)\mathsf{U}(k)\mathsf{V}(k)}$ in the original problem. However, Equation (8.5) includes an additional constraint to prevent the creation of paths corresponding to unfeasible QAP solutions, where two distinct nodes in U are assigned to the same node in V[1].

$$\tilde{q}_{ijhk} = \begin{cases} q_{\mathsf{U}(j)\mathsf{V}(j)\mathsf{U}(k)\mathsf{V}(k)} & j \neq t \wedge k \neq t \wedge \mathsf{U}(j) \neq \mathsf{U}(k) \\ 0 & j = t \vee k = t \\ \infty & \text{otherwise.} \end{cases} \tag{8.5}$$

This additional constraint prevents situations where two arcs are chosen whose target endpoints have the same value for function U. For example, the cost corresponding to the pair of arcs $(1,1) - (3,2)$ and $(3,2) - (1,3)$ is set to ∞ by Equation (8.5) preventing their choice in the solution value.

This definition of costs allows a QAP instance to be transformed into a corresponding QSP instance in a polynomial number of steps. The number of elements in the $\tilde{Q}$ matrix

[1]Distinct nodes in V are always assigned to distinct nodes in U thanks to structure of $\tilde{G}$

is the square of the number of arcs in $\tilde{G}$ and is therefore $O(|U|^6)$.

Any algorithm to solve QSPP can be applied on the transformed instance to obtain an optimal solution. The solution to the original QAP problem is then obtained by applying the U and V functions to the nodes in the optimal QSP path. A node n in the optimal path corresponds to an arc from node $\mathsf{U}(n)$ to node $\mathsf{V}(n)$ in the QAP optimal solution. $\square$

8.4 Lower bounding procedure for the QSPP

In this section we consider four possible lower bounds for the QSPP. The first bound is trivially obtained by considering the linear shortest path problem associated to the QSPP instance. The second bound goes a step forward and builds a new linear shortest path problem from the quadratic cost matrix. The third bound applies to a sequence of reformulations to the original problem to shift the contribution of quadratic costs into the linear cost vector. Finally, the last bound is obtained by applying the reformulation-linearization technique.

8.4.1 Trivial Lower Bound

The simplest lower bound we consider is obtained by considering only the linear portion of the quadratic objective function in (8.1). The value obtained by solving the resulting shortest path problem (8.6) is clearly a lower bound for the QSPP in that $q_{ijkl} \geq 0 \; \forall i, j, k, l$.

$$\text{TB:} \quad \min \quad \{\sum_i \sum_j c_{ij} x_{ij} : x \in X_{st}\} \tag{8.6}$$

Clearly, the performance of this lower bound becomes increasingly poor as we increase the ratio of quadratic to linear costs in the QSPP instance. Nevertheless, its simplicity still makes it a good candidate for the integration in a branch-and-bound algorithm. Finally, the s-t path obtained by the trivial lower bound constitutes a heuristic solution whose cost can be obtained by computing the quadratic contributions associated to the arcs in the path.

8.4.2 A Simple lower bound

The second bound we consider derives from a simple observation on the structure of the quadratic shortest path problem. According to (8.1), the cost of a path from the source node s to the target node t is the sum of the linear costs c_{ij} of each arc (i, j) and the quadratic costs q_{ijhk} associated to each pair of arcs $(i, j), (h, k)$ in the path. A different way to view this sum is to associate each arc (i, j) to an overall cost c_{ij}^{TOT} that depends on the arcs that are present in the solution.

$$\min \quad \sum_i \sum_j c_{ij}^{TOT} x_{ij} \tag{8.7}$$

A lower bound for the problem can therefore be obtained if we replace each c_{ij}^{TOT} with its minimum value over the set of possible feasible solutions which contain arc (i, j). This can be done by solving a set of subproblems with a linear objective function, one for each arc in the graph. Let P_{ij} represents such a subproblem for a given arc $(i, j) \in A$:

$$P_{ij}: \quad w_{ij} = \min\{ \sum_k \sum_l q_{ijkl} x_{kl} : \; x \in X_{st}, \; x_{ij} = 1\}. \tag{8.8}$$

The idea is for each P_{ij} to compute the shortest among the paths from s to t which contain arc (i, j), using the ij-th column of the quadratic cost matrix as the cost vector. This yields a lower bound for the value of c_{ij}^{TOT} in any feasible solution containing (i, j). In reality, problem P_{ij} represents a relaxation of this shortest path: in particular, it admits solutions that consist of the union of a path from s to t that does not contain arc (i, j) and a cycle containing (i, j). More precisely, problem P_{ij} can be viewed as a minimum cost flow problem with two origins s and j and two destinations i and t in a network without arc (i, j). Thanks to the integrality property of minimum-cost flow, P_{ij} can be rewritten as:

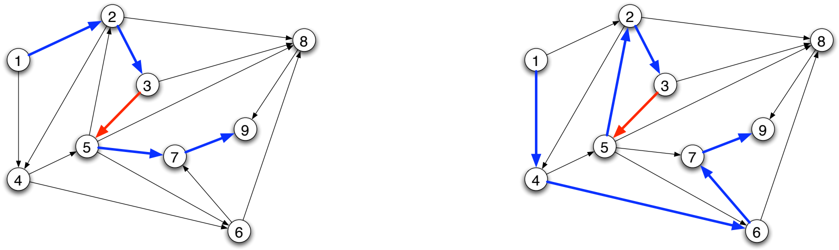
$$P_{ij}: \quad w_{ij} = \min\{ \sum_l \sum_k q_{ijlk} x_{lk} : x \in X_{st}, \; 0 \le x \le 1, \; x_{ij} = 1\}. \tag{8.9}$$

Thus the solution to P_{ij} can easily be found by solving a minimum-cost-flow problem with two units of cost to be transferred between two sources s and j and two destinations i and t in a graph $G' = (N, A \setminus \{(i, j)\})$. The resulting solution will then have either of the two forms depicted in Figure 8.4.

The value of the relaxation w_{ij}, combined with the linear cost c_{ij} of arc (i, j), yields a lower bound for c_{ij}^{TOT}, which can then be integrated into (8.7) as follows:

$$\overline{P}: \quad \overline{z} = \min\{ \sum_i \sum_j (c_{ij} + w_{ij}) x_{ij} : \; x \in X_{st}\}. \tag{8.10}$$

It is worth observing that problem P_{ij} computes a minimum cost flow without capacity constraints and thus allows the same arc to appear multiple times in the solution. This possibility can be avoided, and the bound can be made tighter, by adding an additional constraint, $x_{kl} \le 1$, to the problem P_{ij}. Regardless of the presence of this additional constraint, the solution of $\overline{P}$ yields a lower bound for the original QSPP as confirmed by



(a) Path containing arc (i, j) (b) Path and cycle

Fig. 8.4 Possible solutions to P_{ij}.

the following theorem.

Theorem 8.4.1. *$\overline{P}$ is a lower bound for P; that is $\overline{z} \leq z^*$*

Proof. Let x^*, $\overline{x}$, and x^{ij} be the optimal solutions of the problems P, $\overline{P}$ and P_{ij} respectively. Then

$$\begin{aligned}\overline{z} &= \sum_i \sum_j w_{ij}\overline{x}_{ij} + \sum_i \sum_j c_{ij}\overline{x}_{ij} \leq \sum_i \sum_j w_{ij}x^*_{ij} + \sum_i \sum_j c_{ij}x^*_{ij} \\ &= \sum_i \sum_j (\sum_k \sum_l q_{ijkl}x^{ij}_{kl})x^*_{ij} + \sum_i \sum_j c_{ij}x^*_{ij} \\ &\leq \sum_i \sum_j \sum_k \sum_l q_{ijkl}x^*_{kl}x^*_{ij} + \sum_i \sum_j c_{ij}x^*_{ij} = z^*.\end{aligned}$$

Note that the first and second inequalities follow from the optimality of $\overline{x}$ and x^* for the problems $\overline{P}$ and P respectively, while the second equality results from the definition of w. □

The proof of Theorem 8.4.1 allows us to derive two simple optimality tests.

Corollary 8.4.2. *For all k, l, if $x^{ij}_{kl} = x^*_{kl}$, $\forall (i, j)$ such that $x^*_{ij} = 1$, then $\overline{z} = z^*$.*

Corollary 8.4.3. *For all k, l, if $x^{ij}_{kl} = x^{i'j'}_{kl}$, $\forall (i, j), (i', j')$ such that $(i, j) \neq (i', j')$, then $\overline{z} = z^*$.*

8.4.3 The reformulation lower bound

The lower bound described in Section 8.4.2 transfers part of the quadratic costs to the linear cost vector by solving each of the P_{ij} subproblems. Nevertheless, the part of quadratic cost that is not included in the solution to P_{ij} is simply ignored when computing $\overline{P}$.

Following the *Reformulation* idea of Carraresi and Malucelli for the QAP [27], we present our next lower bound for the QSPP. The new bound captures the aforementioned left-over part, by means of the reduced costs associated to the optimal solution of each P_{ij} instance. More precisely, let us consider the dual of problem P_{ij} in its continuous formulation (8.9) as follows:

$$\overline{D_{ij}} \quad w_{ij} = \max \quad \lambda_t^{ij} - \lambda_s^{ij} + \sum_k \sum_l \mu_{hk}^{ij} \tag{8.11}$$

$$\text{s.t.} \quad -\lambda_k^{ij} + \lambda_l^{ij} + \mu_{hk}^{ij} \leq q_{ijkl} \qquad \forall (k,l) \in A \tag{8.12}$$

$$\mu \leq 0, \ \lambda \text{ unrestricted.} \tag{8.13}$$

where each of the λ^{ij} is associated to one of the constraints in the feasible region defined by X_{st} and μ^{ij} is associated to constraint $x \leq 1$ in (8.9). Let $\lambda_k^{*ij}, \forall k \in N$ and $\mu_{kl}^{ij*}, \forall k, l \in N$ represent the values of the optimal dual variables of P_{ij}. We can define a matrix comprising the reduced costs of all P_{ij} instances as follows:

$$\overline{q}_{ijkl} = q_{ijkl} + \lambda_k^{ij} - \lambda_l^{ij} - \mu_{kl}^{ij} \tag{8.14}$$

Using this matrix, we define the below *Reformulated* QSPP (RQSPP) in which the quadratic component has a lower overall impact.

$$\text{RQSPP:} \quad z^* = \min \quad \sum_i \sum_j \sum_k \sum_l \overline{q}_{ijkl} x_{ij} x_{kl} + \sum_i \sum_j (c_{ij} + w_{ij}) x_{ij} \tag{8.15}$$

$$\text{s.t.} \quad x \in X_{st}. \tag{8.16}$$

Each components of the linear cost vector is updated to include the value of optimal solution of the corresponding P_{ij}; as a result, we can apply the trivial lower bound to RQSPP to compute the value of the bound described in Section 8.4.2. Moreover, the use of the reduced costs as the quadratic-cost matrix balances the increased linear costs making $RQSPP$ equivalent to $QSPP$ as shown by the following theorem.

Theorem 8.4.4. *Problems QSPP and RQSPP are equivalent.*

Proof. Consider any feasible solution x for the problem. Since all feasible solutions consist of a connected path possibly combined with a connected cycle, we have

$$\sum_{kl} (\lambda_k^{ij} - \lambda_l^{ij} - \mu_{kl}^{ij}) = \lambda_s^{ij} - \lambda_t^{ij} - \sum_{kl} \mu_{kl}^{ij}. \tag{8.17}$$

Thus for any feasible solution x, it follows:

$$
\begin{aligned}
&\sum_{ijkl} \overline{q}_{ijkl} x_{ij} x_{kl} + \sum_{ij} (c_{ij} + w_{ij}) x_{ij} \\
=& \sum_{ijkl} q_{ijkl} x_{ij} x_{kl} + \sum_{ij} c_{ij} x_{ij} + \sum_{ij} \sum_{kl} (\lambda_k^{ij} - \lambda_l^{ij} - \mu_{kl}) x_{ij} x_{kl} + \sum_{ij} w_{ij} x_{ij} \\
=& \sum_{ijkl} q_{ijkl} x_{ij} x_{kl} + \sum_{ij} c_{ij} x_{ij}.
\end{aligned}
$$

Note that the last equality follows from Equation (8.17) and the definition of w. □

The strength of reformulating a QSPP instance according to (8.15) is the ability to iterate the described lower bound procedure. This can be obtained by iteratively applying the reformulation (8.15) and thus defining a sequence of equivalent QSPP instances ($P_0, P_1, \dots, P_i, P_{k+1}$ with $P_0 = RQSPP$), each characterized by a stronger impact of linear cost than the previous. To improve the bound quality we must allow each iteration to capture a different part of the quadratic costs of the original instance. Specifically, complementary slackness implies that the reduced costs corresponding to the optimal solution of a given P_p^{ij} (i.e. the P^{ij} subproblem of the p-th reformulation) are all zero. Iterating the reformulation using these quadratic costs as the cost vector for P_{p+1}^{ij} would then lead to the choice of the same path (of cost 0) without changing either the linear or quadratic costs.

8.4.4 Reformulation-linearization technique applied to the QSPP

In this section we first present an MILP formulation for the QSPP based on an application of the first level of RLT. Then we develop an effective Lagrangian relaxation scheme to obtain its LP relaxation bound. The level-1 RLT formulation is generated via the following steps:

- *Reformulation*: Multiply each of the n equations and each of the $n(n-1)$ restrictions $0 \le x \le 1$ defining X_{st} in (8.3) by each binary variable x_{kl}, and append these new restrictions to X_{st}. When a variable x_{ij} in a given constraint is multiplied by x_{kl} express the resulting product as $x_{ij}x_{kl}$ in that order. Substitute $x_{kl}^2 = x_{kl}$ throughout the objective function and constraints. Set $x_{ij}x_{kl} = 0$ if $i = k$ or $j = l$, but not both.

- *Linearization*: Linearize the resulting problem by replacing the product $x_{ij}x_{kl}$ by the continuous variable y_{ijkl}, for all (i, j, k, l) such that $i \ne k$ and $j \ne l$. Enforce the restrictions that $y_{ijkl} = y_{klij}$ for all (i, j, k, l) with $i < k$ and $j \ne l$.

Once these steps have been applied, the following MILP formulation is obtained:

RLT:

$$z^* = \min \quad \sum_i \sum_j \sum_{k \neq i} \sum_{l \neq j} q_{ijkl} y_{ijkl} + \sum_i \sum_j c_{ij} x_{ij} \tag{8.18}$$

$$\text{s.t.} \quad -\sum_{j \neq l} y_{ijkl} + \sum_{j \neq k} y_{jikl} = \begin{cases} -x_{kl} & \forall (i,k,l) : i = s, i \neq k, l \\ x_{kl} & \forall (i,k,l) : i = t, i \neq k, l \\ 0 & \forall (i,k,l) : i \neq s, t, k, l \end{cases} \tag{8.19}$$

$$y_{ijkl} = y_{klij} \quad \forall (i,j,k,l) : i < k, j \neq l \tag{8.20}$$

$$0 \leq y_{ijkl} \leq x_{kl} \quad \forall (i,j,k,l) : i \neq k, j \neq l \tag{8.21}$$

$$x \in X_{st} \tag{8.22}$$

$$x \text{ binary.} \tag{8.23}$$

Problem QSPP and RLT are equivalent, i.e, given any feasible solution x to the QSPP, there exist a y such that (x, y) is feasible to the PRLT with the same objective value and, conversely, for any feasible solution (x, y) to the PRlT, the corresponding x is feasible to the QSPP with the same objective value.

Solving the RLT

To address the solution of the RLT formulation, we develop a Lagrangian dual procedure which is in the same spirit as that used in [3] to solve the level-1 RLT for the QAP. Consider the Lagrangian relaxation of the continuous relaxation of the RLT (CRLT) with multipliers u_{ijkl} for constraints (8.20). The Lagrangian function $L(u)$ is thus defined by

$$\begin{aligned} L(u) = \min \quad \{ & \sum_i \sum_j \sum_{k>i} \sum_{l \neq j} (q_{ijkl} - u_{ijkl}) y_{ijkl} \\ & + \sum_i \sum_j \sum_{k<i} \sum_{l \neq j} (q_{ijkl} - u_{klij}) y_{ijkl} + \sum_i \sum_j c_{ij} x_{ij} \\ & : \quad (8.19), (8.20), (8.21), (8.22)\}. \end{aligned} \tag{8.24}$$

The Lagrangian function is computationally interesting in that, for any given value u, the $L(u)$ can be computed by solving $n(n-1)$ minimum-cost-flow type problems and a single shortest path problem of size n. The decomposition structure is stated in the following theorem.

Theorem 8.4.5. *For a given value u, an optimal solution (x^*, y^*) to the Lagrangian function $L(u)$ can be obtained by solving*

$$\Delta = \min \quad \{\sum_k \sum_{l \neq k} (c_{kl} + \varphi_{kl}) x_{kl} : x \in X_{st}\} \tag{8.25}$$

where for each (k, l) with $k \neq l$, the value φ_{kl} is computed as:

$$\varphi_{kl} = \min \quad \sum_i \sum_j \sum_{k>i} \sum_{l \neq j} (q_{ijkl} - u_{ijkl}) y_{ijkl} + \sum_i \sum_j \sum_{k<i} \sum_{l \neq j} (q_{ijkl} + u_{klij}) y_{ijkl} \tag{8.26}$$

$$\text{s.t.} \quad -\sum_{j \neq l} y_{ijkl} + \sum_{j \neq k} y_{jikl} = \begin{cases} -1 & i = s, i \neq k, l \\ 1 & i = t, i \neq k, l \\ 0 & \forall i, i \neq s, t, k, l \end{cases} \tag{8.27}$$

$$0 \leq y_{ijkl} \leq 1 \quad \forall (i, j), i \neq k, j \neq l. \tag{8.28}$$

Here, x^* is an optimal solution to (8.25) *and $y^*_{ijkl} = \hat{y}_{ijkl} x^*_{kl} \forall (i, j, k, l), i \neq k, j \neq l$, where for each (k, l), $\hat{y}_{ijkl} \forall (i, j), i \neq k, j \neq l$, is an optimal solution to the problem defined in* (8.26)–(8.28).

Proof. The solution (x^*, y^*) is primal feasible to $L(u)$. Since x^* is a feasible solution for (8.25), it satisfies (8.22). Also for each (k, l), $\hat{y}_{ijkl}$ $\forall i \neq k, j \neq l$, satisfies (8.27) and (8.28), it follows that $0 \leq y^*_{ijkl} = \hat{y}_{ijkl} x^*_{kl} \leq x^*_{kl}$ and

$$\begin{aligned} -\sum_{j \neq l} y^*_{ijkl} + \sum_{j \neq k} y^*_{jikl} &= -\sum_{j \neq l} \hat{y}_{ijkl} x^*_{kl} + \sum_{j \neq k} \hat{y}_{jikl} x^*_{kl} \\ &= x^*_{kl} (-\sum_{j \neq l} \hat{y}_{ijkl} + \sum_{j \neq k} \hat{y}_{jikl}) \begin{cases} -x^*_{kl} & i = s, i \neq k, l \\ x^*_{kl} & i = t, i \neq k, l \\ 0 & \forall i, i \neq s, t, k, l \end{cases} \end{aligned}$$

Thus the proof reduces to finding a dual solution to (8.18) that together with (x^*, y^*), satisfy complementary slackness conditions. Toward this end, define $(\hat{\pi}(u), \hat{\delta}(u), \hat{\alpha}(u), \hat{\beta}(u))$ to be optimal dual solutions to constraints (8.27) to (8.28) and flow constraint and $0 \leq x \leq 1$ restriction in X_{st} in (8.25) respectively. For ease of reading we show these variables as $(\hat{\pi}, \hat{\delta}, \hat{\alpha}, \hat{\beta})$. For each (k, l), dual feasibility of (8.26) – (8.28) gives:

$$\begin{aligned} \hat{\pi}_{jkl} - \hat{\pi}_{ikl} + \hat{\delta}_{ijkl} &= q_{ijkl} - u_{ijkl} \quad \forall (i, j, k, l), k > i, l \neq j, \\ \hat{\pi}_{jkl} - \hat{\pi}_{ikl} + \hat{\delta}_{ijkl} &= q_{ijkl} - u_{klij} \quad \forall (i, j, k, l), k < i, l \neq j. \end{aligned}$$

Moreover, dual feasibility of $(\hat{\alpha}, \hat{\beta})$ to (8.25), and dual optimality of $\hat{\pi}$ and $\hat{\delta}$ to (8.26) – (8.28) for each (k, l) provide:

$$\hat{\alpha}_l - \hat{\alpha}_k + \hat{\beta}_{kl} = c_{kl} + \varphi_{kl} = c_{kl} + \hat{\pi}_{tkl} - \hat{\pi}_{skl} + \sum_i \sum_j \hat{\delta}_{ijkl} \quad \forall (i, j), i \neq k, j \neq l.$$

Therefore $(\pi, \delta, \alpha, \beta) = (\hat{\pi}, \hat{\delta}, \hat{\alpha}, \hat{\beta})$ is a dual feasible solution to (8.24), where the variables π, δ correspond to constraints (8.19), and (8.21) respectively, and the variables α, β correspond to the flow constraint and $0 \leq x \leq 1$ restriction respectively.

Now we show that $L^*(u) = \Delta^*$. Dual optimality of $(\hat{\alpha}, \hat{\beta})$ to (8.25) implies:

$$\begin{aligned}
\Delta^* =& \hat{\alpha}_t - \hat{\alpha}_s + \sum_k \sum_l \hat{\beta}_k l \\
=& \sum_k \sum_l [c_{kl} + \varphi_{kl}] x^*_{kl} \\
=& \sum_k \sum_l [c_{kl} + \sum_{i<k} \sum_{j \neq l} (q_{ijkl} - u_{ijkl}) \hat{y}_{ijkl} + \sum_{i>k} \sum_{j \neq l} (q_{ijkl} + u_{klij}) \hat{y}_{ijkl}] x^*_{kl} \\
=& \sum_k \sum_l \sum_{i<k} \sum_{j \neq l} (q_{ijkl} - u_{ijkl}) y^*_{ijkl} + \sum_k \sum_l \sum_{i>k} \sum_{j \neq l} (q_{ijkl} + u_{klij}) y^*_{ijkl} \\
+& \sum_k \sum_l c_{kl} x^*_{kl} = L^*(u).
\end{aligned}$$

Note that the third equality follows from the definition of φ_{kl}, while the forth equality results from the definition of y^*. □

8.5 A polynomially solvable case

In this section we consider a special case of the QSPP, namely Adjacent QSPP (AQSPP), where the linear costs of all arcs and the interaction costs of all non-adjacent pair of arcs are assumed to be zero. Therefore, the quadratic terms of the form $x_{ij}x_{kl}$ with $j = k$ and $i \neq l$ or with $j \neq k$ and $i = l$ have nonzero objective function coefficients in the AQSPP. Also the quadratic expressions of the form $x_{ij}x_{ji}$ are not considered as a means of eliminating subtours on two nodes. The AQSPP can be viewed as a generalization of the Reload Cost path introduced in [11]. In the reload cost problems, one is given a graph whose every edge is assigned a color and there is a *reload* cost when passing through a node on two edges that have different colors.

An Integer linear programming formulation of the AQSPP that takes advantage of the sparsity of nonzero objective function coefficient can be written as follows:

$$\begin{aligned}
\text{AQSPP:} \quad z^* = \min \quad & \sum_i \sum_{j \neq i} \sum_{k \neq i,j} \bar{q}_{ijk} x_{ij} x_{jk} \\
\text{s.t.} \quad & x \in X_{st}, \ x \text{ binary}.
\end{aligned}$$

where $\bar{q}_{ijk} = q_{ijkl} \forall (i,j,k,l), i \neq j, k \neq l, i \neq l,$ and $j = k$.

Theorem 8.5.1. *The AQSPP is solvable in polynomial time.*

Proof. Consider directed graph $G' = (N', E')$ obtained from G as follows: for each arc

(i,j) in the graph G create a node $\langle i,j\rangle$ in G', a link between each two nodes $\langle i,j\rangle$ and $\langle j,k\rangle$ with $\bar{q}_{ijk} > 0$ in G', and assign a weight $\bar{q}_{ijk}$ to each arc $(\langle i,j\rangle, \langle j,k\rangle)$. Link the source node s to all nodes $\langle s,j\rangle$ by zero cost arc $(s, \langle s,j\rangle)$ and link all nodes $\langle i,t\rangle$ to the target node t by zero cost arc $(\langle i,t\rangle, t)$. If in G' we partition the set of nodes into $|N|(= n+2)$ clusters $V_s, V_1, V_2, \dots V_n, V_t$ with $V_i = \{\langle i,j\rangle : j = 1,2,\dots,n, j \neq i\} \forall i = 1,2,\dots,n$, and $V_i = \{i\}$ with $i = s,t$, then all arcs are defined between the nodes from different clusters and there are no intra-set arcs. Since G' contains at most $(n+1)^2 + 2$ nodes, it can be constructed in polynomial time.

Now we show that a minimum cost $s-t$ simple path in G' corresponds to a minimum quadratic cost $s-t$ simple path in G, and vice versa. Toward this end, let $P = (s = a_0, a_1, \dots, a_m, a_{m+1} = t)$ be the minimum quadratic cost $s-t$ simple path in G with cost C_P. Then the corresponding path $P' = (s, \langle s,a_1\rangle, \langle a_1,a_2\rangle, \dots, \langle a_m, a_{m+1}\rangle, t)$ in G' with $C'_P = C_P$ is also simple and optimal in G'. Because P' is simple, so it is enough to show that P' is optimal, if not, there is a different simple path $T' = (s = b_0, \langle s,b_1\rangle, \langle b_1,b_2\rangle, \dots, \langle b_l, b_{l+1}\rangle, b_{l+1} = t)$ in G' with $C'_T < C'_P$ which is optimal. We claim that the corresponding path $T = (s, b_1, b_2, \dots, b_l, t)$ in G is simple, otherwise; contains at least one cycle T_0 with positive cost C_{T_0}. Thus $C_T = C_{T_0} + C_{T\setminus T_0}$, where $T \setminus T_0$ is acyclic part of the T. Projecting the simple path $T \setminus T_0$ on G' implies a simple path $(T\setminus T_0)'$ with $C_{(T\setminus T_0)'} < C'_T$ which contradict the optimality of T'. Therefore, T is a simple path in G with $C_T = C_{T'} < C'_P = C_P$, which is contradict the optimality of the path P in G.

Consider the optimal $s-t$ simple path $P' = (s, \langle s,a_1\rangle, \langle a_1,a_2\rangle, \dots, \langle a_m, a_{m+1}\rangle, t)$ in G'. We show that the corresponding path $P = (s = a_0, a_1, \dots, a_m, a_{m+1} = t)$ is the minimum $s-t$ simple path in G. P is a simple path in G, otherwise; P' can't be the optimal $s-t$ simple path in G'. Also P is the minimum $s-t$ path in G, otherwise there exists a path R in G with $C_R < C_P$ which is optimal. The projection of the path R on G' implies a simple path R' with $C'_R < C_P = C'_p$ which is a contradiction, and the proof is complete. □

We can generalize the adjacent QSPP to h-Adjacent QSPP (h-QSPP) by defining the concept of h-Adjacency.

Definition 8.5.1. *Given a fixed positive integer number h, the graph $G = (V,E)$ and two arcs (i,j) and (k,l) in E. We say that two arcs (i,j) and (k,l) are h-adjacent in G, if there exist a path P_{il} of length at most h from node i to node l.*

In the h-AQSPP, in addition to the linear costs, all the quadratic costs of non-h-

adjacency arcs $(i,j),(k,l) \in E$ are assumed to be zero, i.e.,

$$\bar{q}_{ijkl} = \begin{cases} q_{ijkl} & (i,j) \text{ and } (k,l) \text{ are } h\text{-adjacent} \\ 0 & otherwise. \end{cases} \tag{8.29}$$

Note that the AQSPP is a special case of the h-Adjacent QSPP with $h = 2$.

8.6 Computational experiments

In this section, we describe the results of the numerical evaluation we carried out to assess the performance of the lower bounds described in Sections 8.4. The section is structured as follows. In the following subsection we present the generation of test instances.Then,we will discuss in detail the results of our computational results.

8.6.1 Problem Instances

Our evaluation consists of tests carried out in three sets of Quadratic Shortest Path instances, obtained from three different groups of graphs. We generated a number of instances in each graph by varying the source and target node of the desired path. The first group of graphs will be referred to as **30NODES** and consists of a set of 30 nodes connected by 300 directed arcs. This is a fairly connected topology in which the length of paths ranges from 1 to 3 arcs. The second group, referred to as **50NODES**, is also characterized by a fairly high degree of connectivity and consists of 50 nodes connected by 500 directed arcs. As in the case of 30NODES, two distance between two random nodes ranges from 1 to 3 directed arcs. Finally, **30SPARSE** is a less connected graph consisting of 30 nodes but only 100 arcs. This yields a longer path length with the longest path between two nodes being 7 arcs long.

8.6.2 Lower Bound Performance

We evaluate the performance of our lower bounds on random set of instances by considering only those yielding a path that is at least three arcs long. We compute the optimal solution and thus consider the percentage gap between the bound and the optimal solution as well as the time required by the algorithm to compute the lower bound. For each set of the instances we compare the lower bounds obtained by different approaches.

Tables 8.1 and 8.2 show the results for paths that are at least three arcs long in 30NODES and 50NODES respectively; while Table 8.3 shows the results for paths consisting of six or more arcs in 30SPARSE.

In all tables the trivial lower bound is unable to obtain reasonable results. However, its inexpensive computation time still makes it a valid candidate for use in Branch-and-Bound. What is more interesting is the comparison between the two improved bounds and the bound obtained with the RLT formulation.

In the 30NODES instance set, the best values are obtained with the iterated bound, which yields an average percentage gap of 2.37%. The RLT formulation also achieves good results with an average gap of 2.84%. The non-iterate bound is only slightly worse with an average gap of 5.44%. In comparison between computation times, the non-iterated lower bound runs in an average of only $280ms$; the iterated bound runs for an average of $6.20s$; while the bound obtained by CPLEX requires an average solve time of $9.83s$

Considering the larger graph of the 50NODES instance set, the reformulated-based procedure performs better. The iterated bound achieves a percentage gap of only 1.61% in an average of $23.1s$, while the non-iterated bound yields a gap of 5.56% but runs in as little as $1.19s$. The RLT bound computed by CPLEX, on the other hand, runs in an average of $56.23s$ with a percentage gap (5.47%) which is only slightly better than that of the non-iterated bound. Finally, all lower bounds perform better in the 30SPARSE instance set, thanks to the lower number of arcs with average solve times that are well below $1s$. The value of the bounds is also better, with percentage gaps ranging from around 0.5% for RLT and the iterated bound and 2.15% for the non iterated bound.

8.7 conclusion

In this chapter, we presented our study of a quadratic optimization problem which seeks to minimize a quadratic objective function in the presence of shortest path constraints. This new Quadratic Shortest Path problem extends traditional shortest paths to address situations in which the value of the objective function is determined not only by the choice of each single arc, but also by the combined presence of pairs of arcs in the solution. This problem arises in application areas such as route guidance and robust network design. We first provided a definition of the QSP problem proving that it is NP-hard. Second, we proposed different bounding techniques which improve the trivial lower bound. Finally, we evaluated these techniques in sense of bound tightness and computational performance. Computational experiments demonstrate that the iterative QP-based reformulation approach is an effective method to address this type of optimization problem by achieving the same or better performance than the RLT (using Cplex) in a shorter computation time. A future research is to apply the explained decomposition approach for solving the RLT formulation.

Table 8.1 Lower-bound performance in the 30NODES set of instances.

	Trivial Bound			Non-Iterated Bound			Iterated Bound			CPLEX RLT			
Inst.	Value	Gap%	Time	Value	Gap%	Time	Value	Gap%	Time	Value	Gap%	Time	Opt.
2 − 20	484	84.12	0	3013.5	1.13	0.19	3048	0.00	3.22	3020.00	0.92	15.39	3048
3 − 10	624	79.19	0	2674.5	10.79	0.23	2958.22	1.33	3.90	2998.00	0.00	16.32	2998
4 − 10	671	77.77	0	2849.5	5.61	0.24	2989.88	0.96	5.15	2927.00	3.05	19.38	3019
5 − 20	606	79.16	0	2736.5	5.90	0.24	2902.64	0.18	6.00	2908.00	0.00	11.61	2908
8 − 1	360	84.69	0	2351	0.00	0.25	2351	0.00	5.80	2337.75	0.56	19.15	2351
10 − 20	781	76.95	0	3388	0.00	0.25	3388	0.00	6.16	3156.50	6.836	9.61	3388
12 − 3	370	75.48	0	1509	0.00	0.25	1509	0.00	6.13	1509.00	0.00	3.07	1509
12 − 20	351	86.94	0	2687	0.00	0.27	2687	0.00	7.20	2687.00	0.00	13.00	2687
14 − 8	417	80.67	0	1724	20.07	0.28	1724	20.07	6.27	2013.00	6.68	3.75	2157
15 − 0	347	82.33	0	1964	0.00	0.28	1964	0.00	6.25	1964.00	0.00	4.06	1964
15 − 17	209	88.64	0	1839	0.00	0.31	1839	0.00	6.35	1839.00	0.00	4.50	1839
17 − 4	374	81.06	0	1975	0.00	0.291	1975	0.00	7.39	1975.00	0.00	4.84	1975
19 − 13	229	92.26	0	2490.5	15.83	0.291	2605.75	11.94	6.36	2386.17	19.36	13.87	2959
19 − 20	190	88.13	0	1601	0.00	0.311	1601	0.00	6.32	1601.00	0.00	1.79	1601
19 − 24	231	90.33	0	2087.5	21.66	0.30	2244.78	6.08	7.48	2390.00	0.00	14.90	2390
19 − 27	323	87.49	0	2298	11.00	0.30	2561.48	0.79	6.36	2568.25	0.53	18.38	2582
23 − 26	536	74.16	0	2074	0.00	0.31	2074	0.00	6.40	2074.00	0.00	3.25	2074
23 − 28	265	82.26	0	1494	0.00	0.31	1494	0.00	6.43	1494.00	0.00	6.60	1494
25 − 3	346	84.26	0	2198	0.00	0.30	2198	0.00	6.44	2198.00	0.00	3.79	2198
26 − 0	291	89.45	0	2546.5	7.70	0.30	2652.81	3.85	6.47	2534.00	8.16	14.68	2759
26 − 2	412	82.94	0	2335.5	3.29	0.31	2414.08	0.04	6.43	2285.75	5.35	3.84	2415
26 − 16	429	83.34	0	2183	15.22	0.31	2295	10.87	7.51	2495.50	3.09	4.56	2575
28 − 0	500	83.16	0	2799	5.73	0.33	2967.03	0.07	6.44	2739.25	7.74	17.06	2969
28 − 4	220	92.48	0	2350	19.69	0.31	2784	4.85	6.52	2587.25	11.58	3.90	2926
29 − 4	398	85.29	0	2522.5	6.78	0.32	2691.72	0.53	6.48	2706.00	0.00	11.95	2706
Average	–	83.32	0	–	5.44	0.28	–	2.37	6.20	–	2.84	9.83	–

Table 8.2 Lower-bound performance in the 50nodes set of instances.

	Trivial Bound			Non-Iterated Bound			Iterated Bound			CPLEX RLT			
Inst.	Value	Gap%	Time	Value	Gap%	Time	Value	Gap%	Time	Value	Gap%	Time	Opt.
0 – 5	612	70.45	0	2032.5	1.86	0.92	2071	0.00	14.88	2043.00	1.35	18.26	2071
0 – 17	724	72.22	0	2606	0.00	1.08	2606	0.00	17.06	2606	0.00	15.72	2606
0 – 18	390	83.80	0	2408	0.00	1.12	2408	0.00	24.27	2408.00	0.00	9.56	2408
0 – 23	785	79.39	0	3435	9.80	1.09	3808	0.00	25.52	3452.00	9.35	62.12	3808
0 – 29	969	69.14	0	3140	0.00	1.08	3140	0.00	22.90	3140	0.00	38.34	3140
1 – 13	382	81.34	0	2047	0.00	1.16	2047	0.00	23.36	2047.00	0.00	76.21	2047
1 – 23	503	82.46	0	2867	0.00	1.21	2867	0.00	23.54	2867.00	0.00	65.49	2867
1 – 26	529	86.29	0	2879.5	25.36	1.21	3268.5	15.28	23.84	3071.25	20.39	56.12	3858
1 – 38	536	78.10	0	2345	4.17	1.21	2447	0.00	23.97	2386.50	2.47	119.20	2447
1 – 47	524	80.63	0	2611	3.48	1.21	2703.36	0.00	24.22	2574.00	4.84	69.29	2705
1 – 48	384	84.86	0	2463.5	2.86	1.24	2536	0.00	24.05	2438.00	3.86	66.42	2536
2 – 5	680	65.24	0	1956	0.00	1.26	1956	0.00	24.06	1956.00	0.00	14.50	1956
2 – 10	628	78.54	0	2926	0.00	1.28	2926	0.00	24.35	2783.00	4.89	81.09	2926
2 – 14	482	82.61	0	2281.5	17.69	1.26	2645.41	4.57	24.19	2383.00	14.03	52.64	2772
2 – 20	452	81.51	0	2445	0.00	1.28	2445	0.00	24.17	2445.00	0.00	59.64	2445
2 – 24	534	82.77	0	2595.5	16.27	1.26	2595.5	16.27	23.50	2633.50	15.05	13.32	3100
2 – 26	380	88.59	0	3124.5	6.20	1.29	3317.52	0.40	27.60	3124.25	6.21	54.48	3331
2 – 29	976	61.08	0	24.83	1.00	1.32	2508	0.00	24.05	2495.50	0.50	9.28	2508
2 – 42	483	84.11	0	2749	9.57	1.3	3040	0.00	23.94	2795.50	8.04	66.35	3040
3 – 4	318	86.70	0.01	2133	10.79	1.31	2391	0.00	27.80	2135.75	10.68	108.79	2391
3 – 23	596	82.77	0	3042.5	12.04	1.35	3262	5.70	23.97	2966.25	14.25	43.17	3459
3 – 40	386	81.42	0.01	2046	1.54	1.32	2078	0.00	27.82	2078.00	0.00	81.43	2078
3 – 45	516	83.97	0.01	2932	8.89	1.31	3199.62	0.57	24.09	2732.50	15.09	87.78	3218
4 – 15	511	80.77	0	2574	3.12	1.32	2644.14	0.48	24.17	2479.25	6.69	71.18	2657
4 – 23	635	76.99	0	2760	0.00	1.37	2760	0.00	24.37	2760.00	0.00	74.67	2760
4 – 43	342	80.30	0	1736	0.00	0.86	1736	0.00	14.52	1736.00	0.00	21.72	1736
5 – 41	446	77.84	0	2013	0.00	1.02	2013	0.00	19.57	2013.00	0.00	13.85	2013
6 – 4	288	88.92	0	2371.5	8.75	1.04	2599	0.00	19.35	2485.25	4.38	76.72	2599
6 – 13	350	89.10	0	2995	6.73	1.03	3185.03	0.81	20.99	3123.00	2.80	80.42	3211
6 – 15	532	83.73	0	3148.5	3.72	1.05	3261.59	0.26	22.33	2844.17	13.02	58.35	3270
6 – 20	312	89.25	0	2368.5	18.38	1.12	2743.48	5.46	23.42	2564.17	11.64	77.04	2902
Average	–	80.48	0	–	5.56	1.19	–	1.61	23.1	–	5.47	56.23	–

Table 8.3 Lower-bound performance in the 30SPARSE set of instances set of instances.

	Trivial Bound			Non-Iterated Bound			Iterated Bound			CPLEX RLT			
Inst.	Value	Gap%	Time	Value	Gap%	Time	Value	Gap%	Time	Value	Gap%	Time	Opt.
13 – 24	3270	80.93	0	16614.5	3.09	0.03	17127.5	0.10	0.31	17144	0.00	1.07	17144
15 – 0	2690	86.44	0	19712.5	0.62	0.03	19831.2	0.02	0.33	19835	0.00	0.68	19835
15 – 1	2583	88.61	0	21799	3.90	0.02	22528.1	0.69	0.45	22684	0.00	1.25	22684
15 – 4	3455	86.68	0	24572.5	5.25	0.02	25359.4	2.22	0.43	25651.8	1.09	2.33	25934
15 – 6	2799	84.60	0	17491.5	3.76	0.03	17911.1	1.45	0.54	18175	0.00	0.70	18175
15 – 9	2952	84.49	0	18453	3.06	0.03	19017.8	0.10	0.49	19036	0.00	1.04	19036
15 – 10	2692	87.05	0	19979.5	3.87	0.02	20225.8	2.69	0.5	20774	0.00	0.88	20784
15 – 11	2773	83.22	0	16524	0.00	0.03	16524	0.00	0.48	16524	0.00	0.48	16524
15 – 12	2798	83.78	0	17255	0.00	0.03	17255	0.00	0.5	17255	0.00	0.49	17255
15 – 14	3448	81.22	0	18359	0.00	0.03	18359	0.00	0.51	18359	0.00	0.58	18359
15 – 16	2676	84.61	0	16918	2.70	0.02	17372.3	0.08	0.5	17387	0.00	0.67	17387
15 – 17	2703	84.57	0.01	17518	0.00	0.03	17518	0.00	0.52	17518	0.00	0.63	17518
15 – 20	3111	86.54	0	22570.5	2.36	0.03	23033.9	0.36	0.55	22350.2	3.32	1.00	23117
15 – 22	2552	84.67	0	16570	0.45	0.03	13642.7	0.01	0.54	16645	0.00	0.51	16645
15 – 23	3795	79.36	0	18391	0.00	0.03	18391	0.00	0.55	18391	0.00	0.34	18391
15 – 24	4645	80.80	0	24192	0.00	0.03	24192	0.00	0.54	24192	0.00	0.36	24192
15 – 26	2633	85.51	0	17840	1.83	0.03	18140.5	0.17	0.54	18172	0.00	0.69	18172
15 – 27	2596	85.57	0	17981	0.08	0.02	17994.6	0.00	0.56	17995	0.00	0.72	17995
14 – 24	3713	78.18	0	16547	2.74	0.03	1699.4	0.09	0.55	17014	0.00	0.51	17014
19 – 1	2310	85.69	0	15359.5	4.86	0.03	15991.2	0.95	0.54	16144	0.00	1.14	16144
19 – 4	3182	83.34	0	17541.5	8.15	0.03	18523.4	3.01	0.54	18790.8	1.61	2.65	19098
19 – 20	2838	83.29	0	16507	2.80	0.03	16913.3	0.40	0.66	16286.8	4.09	1.07	16982
19 – 24	4372	76.17	0	18346	0.00	0.03	18346	0.00	0.67	18346	0.00	0.38	18346
Average	–	83.71	0	–	2.15	0.03	–	0.54	0.51	–	0.44	0.9	–

References

[1] Adams, W. P. and Forrester, R. J. (2005). A simple recipe for concise mixed 0-1 linearizations. *Operations Research Letters*, 33(1):55 – 61.

[2] Adams, W. P., Forrester, R. J., and Glover, F. W. (2004). Comparisons and enhancement strategies for linearizing mixed 0-1 quadratic programs. *Discrete Optimization*, 1(2):99–120.

[3] Adams, W. P., Guignard, M., Hahn, P. M., and Hightower, W. L. (2007). A level-2 reformulation-linearization technique bound for the quadratic assignment problem. *European Journal of Operational Research*, 180(3):983–996.

[4] Adams, W. P. and Johnson, T. A. (1994). Improved linear programming-based lower bounds for the quadratic assignment problem. *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 16:43–77.

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

[6] Adams, W. P. and Sherali, H. D. (1990). Linearization strategies for a class of zero-one mixed integer programming problems. *Operations Research*, 38(2):217–226.

[7] Aggarwal, A., Coppersmith, D., Khanna, S., Motwani, R., and Schieber, B. (2000). The angular-metric traveling salesman problem. *SIAM Journal on Computing*, 29(3):697–711.

[8] Alizadeh, F. (1992). Optimization over positive semi-definite cone; interior-point methods and combinatorial applications. *In P. Pardalos, editor, Advances in Optimization and Paral lel Computing.*

[9] Alizadeh, F. (1995). Interior point methods in semidefinite programming with applications to combinatorial optimization. *SIAM Journal on Optimization*, 5(1):13–51.

[10] Alizadeh, F., Haeberly, J.-P. A., and Overton, M. L. (1998). Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results. *SIAM Journal on Optimization*, 8(3):746–768.

[11] Amaldi, E., Galbiati, G., and Maffioli, F. (2011). On minimum reload cost paths, tours, and flows. *Networks*, 57(3):254–260.

[12] Anstreicher, K. M. (2009). Semidefinite programming versus the reformulation-linearization technique for nonconvex quadratically constrained quadratic programming. *Journal of Global Optimization*, 43(2-3):471–484.

[13] Anstreicher, K. M. and Brixius, N. W. (2001). A new bound for the quadratic assignment problem based on convex quadratic programming. *Mathematical Programming*, 89(3):341–357.

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

[15] Assad, A. A. and Xu, W. (1985). On lower bounds for a class of quadratic 0, 1 programs. *Operations Research Letters*, 4(4):175–180.

[16] Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W., and Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329.

[17] Ben-David, G. and Malah, D. (2005). Bounds on the performance of vector-quantizers under channel errors. *Information Theory, IEEE Transactions on*, 51(6):2227–2235.

[18] Billionnet, A. and Elloumi, S. (2007). Using a mixed integer quadratic programming solver for the unconstrained quadratic 0-1 problem. *Mathematical Programming*, 109(1):55–68.

[19] Billionnet, A., Elloumi, S., and Plateau, M.-C. (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.

[20] Billionnet, A., Faye, A., and Soutif, É. (1999). A new upper bound for the 0-1 quadratic knapsack problem. *European Journal of Operational Research*, 112(3):664–672.

[21] Billionnet, A. and 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.

[22] Buchheim, C. and Klein, L. (2014). Combinatorial optimization with one quadratic term: Spanning trees and forests. *Discrete Applied Mathematics*.

[23] Burkard, R. E., Çela, E., Rote, G., and Woeginger, G. J. (1998). The quadratic assignment problem with a monotone anti-monge and a symmetric toeplitz matrix: Easy and hard cases. *Mathematical Programming*, 82(1-2):125–158.

[24] Çela, E., Deineko, V. G., and Woeginger, G. J. (2014). Well-solvable cases of the QAP with block-structured matrices. *ArXiv e-prints*.

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

[26] Caprara, A., Pisinger, D., and Toth, P. (1999). Exact solution of the quadratic knapsack problem. *INFORMS Journal on Computing*, 11(2):125–137.

[27] Carraresi, P. and Malucelli, F. (1992). A new lower bound for the quadratic assignment problem. *Operations Research*, 40(1-Supplement-1):S22–S27.

[28] Carraresi, P. and Malucelli, F. (1994). A reformulation scheme and new lower bounds for the QAP. *Quadratic Assignment and Related Problems, DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, 16:147–160.

[29] Chaovalitwongse, W., Pardalos, P. M., and Prokopyev, O. A. (2004). A new linearization technique for multi-quadratic 0–1 programming problems. *Operations Research Letters*, 32(6):517–522.

[30] Chardaire, P. and Sutter, A. (1995). A decomposition method for quadratic zero-one programming. *Management Science*, 41(4):704 – 712.

[31] Chen, B. (1995). Special cases of the quadratic assignment problem. *European Journal of Operational Research*, 81(2):410–419.

[32] Christofides, N. and Benavent, E. (1989). An exact algorithm for the quadratic assignment problem on a tree. *Operations Research*, 37(5):760–768.

[33] Cordone, R. and Passeri, G. (2012). Solving the quadratic minimum spanning tree problem. *Applied Mathematics and Computation*, 218(23):11597–11612.

[34] DeIneko, V. G. and Woeginger, G. J. (1998). A solvable case of the quadratic assignment problem1. *Operations Research Letters*, 22(1):13–17.

[35] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, 1(1):269–271.

[36] Drugan, M. M. (2014). Generating QAP instances with known optimum solution and additively decomposable cost function. *Journal of Combinatorial Optimization*, pages 1–35.

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

[38] Erdoğan, G. and Tansel, B. . (2011). Two classes of quadratic assignment problems that are solvable as linear assignment problems. *Discrete Optimization*, 8(3):446 – 451.

[39] Faye, A. and Roupin, F. (2007). Partial lagrangian relaxation for general quadratic programming. *4OR*, 5(1):75–88.

[40] Feng, Z. and Fang, L. (2014). A new o(nL)-iteration predictor-corrector algorithm with wide neighborhood for semidefinite programming. *Journal of Computational and Applied Mathematics*, 256(0):65 – 76.

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

[42] Fischer, A., Fischer, F., Jäger, G., Keilwagen, J., Molitor, P., and Grosse, I. (2014). Exact algorithms and heuristics for the quadratic traveling salesman problem with an application in bioinformatics. *Discrete Applied Mathematics*, 166:97–114.

[43] Fischer, A. and Helmberg, C. (2013). The symmetric quadratic traveling salesman problem. *Mathematical Programming*, 142(1-2):205–254.

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

[45] Fischetti, M., Monaci, M., and Salvagnin, D. (2012). Three ideas for the quadratic assignment problem. *Operations Research*, 60(4):954–964.

[46] Fourer, R., Gay, D. M., and Kernighan, B. W. (1987). *AMPL: A mathematical programming language*. AT&T Bell Laboratories Murray Hill, NJ 07974.

[47] Freund, R. M. (1994). Complexity of an algorithm for finding an approximate solution of a semi-definite program with no regularity assumption. Technical report, Technical Report OR-302-94, Operations Research Center, MIT.

[48] Frieze, A. and Yadegar, J. (1983). On the quadratic assignment problem. *Discrete applied mathematics*, 5(1):89–98.

[49] Fujie, T. and Kojima, M. (1997). Semidefinite programming relaxation for nonconvex quadratic programs. *Journal of Global Optimization*, 10(4):367–380.

[50] Galbiati, G., Gualandi, S., and Maffioli, F. (2011). On minimum changeover cost arborescences. In *Experimental Algorithms*, Lecture Notes in Computer Science, 6630, pages 112–123. Springer.

[51] Galbiati, G., Gualandi, S., and Maffioli, F. (2014). On minimum reload cost cycle cover. *Discrete Applied Mathematics*, 164, Part 1(0):112 – 120.

[52] Gamvros, I., Gouveia, L., and Raghavan, S. (2012). Reload cost trees and network design. *Networks*, 59(4):365–379.

[53] Geoffrion, A. and Graves, G. (1976). Scheduling parallel production lines with changeover costs: Practical application of a quadratic assignment/LP approach. *Operations Research*, 24(4):595–610.

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

[55] Gilmore, P. C. and Gomory, R. E. (1961). A linear programming approach to the cutting-stock problem. *Operations research*, 9(6):849–859.

[56] Glover, F. (1975). Improved linear integer programming formulations of nonlinear integer problems. *Management Science*, 22(4):455–460.

[57] Glover, F. and Woolsey, E. (1974). Technical Note–Converting the 0-1 polynomial programming problem to a 0-1 linear program. *Operations Research*, 22(1):180–182.

[58] Gurobi Optimization, I. (2013). Gurobi optimizer reference manual.

[59] Hahn, P. and Grant, T. (1998). Lower bounds for the quadratic assignment problem based upon a dual formulation. *Operations Research*, 46(6):912–922.

[60] Hahn, P. M., Zhu, Y.-R., Guignard, M., Hightower, W. L., and Saltzman, M. J. (2012). A level-3 reformulation-linearization technique-based bound for the quadratic assignment problem. *INFORMS Journal on Computing*, 24(2):202–209.

[61] Hammer, P. L. and Rubin, A. A. (1970). Some remarks on quadratic programming with 0-1 variables. *RAIRO-Operations Research-Recherche Opérationnelle*, 4(V3):67–79.

[62] Hansen, P. (1979). Methods of nonlinear 0-1 programming. *Annals of Discrete Mathematics*, 5:53–70.

[63] Held, M. and Karp, R. M. (1970). The traveling-salesman problem and minimum spanning trees. *Operations Research*, 18(6):1138–1162.

[64] Helmberg, C. and Rendl, F. (1998). Solving quadratic (0, 1)-problems by semidefinite programs and cutting planes. *Mathematical Programming*, 82(3):291–315.

[65] Helmberg, C., Rendl, F., Vanderbei, R. J., and Wolkowicz, H. (1996). An interior-point method for semidefinite programming. *SIAM Journal on Optimization*, 6(2):342–361.

[66] Helmberg, C., Rendl, F., and Weismantel, R. (2000). A semidefinite programming approach to the Quadratic Knapsack Problem. *Journal of Combinatorial Optimization*, 4(2):197–215.

[67] Jäger, G. and Molitor, P. (2008). Algorithms and experimental study for the traveling salesman problem of second order. In *Combinatorial Optimization and Applications*, pages 211–224. Springer.

[68] Jarre, F. (1993). An interior-point method for minimizing the maximum eigenvalue of a linear combination of matrices. *SIAM Journal on Control and Optimization*, 31(5):1360–1377.

[69] Jünger, M. and Kaibel, V. (2001). Box-inequalities for quadratic assignment polytopes. *Mathematical Programming*, 91(1):175–197.

[70] Kallehauge, B., Larsen, J., and Madsen, O. B. (2006). Lagrangian duality applied to the vehicle routing problem with time windows. *Computers & Operations Research*, 33(5):1464–1487.

[71] Kamath, A. and Karmarkar, N. (1992). Recent advances in global optimization. chapter A continuous approach to compute upper bounds in quadratic maximization problems with integer constraints, pages 125–140. Princeton University Press, Princeton, NJ, USA.

[72] Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311. ACM.

[73] Karypis, G., Kumar, V., and Kumar, V. (1998). Multilevel k-way partitioning scheme for irregular graphs. *Journal of Parallel and Distributed Computing*, 48:96–129.

[74] Kaufman, L. and Broeckx, F. (1978). An algorithm for the quadratic assignment problem using bender's decomposition. *European Journal of Operational Research*, 2(3):207 – 211.

[75] Kim, S., Chang, K.-N., and Lee, J.-Y. (1995). A descent method with linear programming subproblems for nondifferentiable convex optimization. *Mathematical Programming*, 71(1):17–28.

[76] Koopmans, T. C. and Beckmann, M. (1957). Assignment problems and the location of economic activities. *Econometrica: Journal of the Econometric Society*, pages 53–76.

[77] Laporte, G., Mercure, H., and Nobert, Y. (1987). Generalized travelling salesman problem through n sets of nodes: the asymmetrical case. *Discrete Applied Mathematics*, 18(2):185–197.

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

[79] Li, Y., Pardalos, P., Ramakrishnan, K., and Resende, M. (1994). Lower bounds for the quadratic assignment problem. *Annals of Operations Research*, 50(1):387–410.

[80] Loiola, E. M., De Abreu, N. M. M., Boaventura-Netto, P. O., Hahn, P., and Querido, T. (2007). A survey for the quadratic assignment problem. *European Journal of Operational Research*, 176(2):657–690.

[81] Lovász, L. (1979). On the Shannon capacity of a graph. *Information Theory, IEEE Transactions on*, 25(1):1–7.

[82] Lovász, L. and Schrijver, A. (1991). Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1(2):166–190.

[83] Lozano, M., Glover, F., García-Martínez, C., Rodríguez, F. J., and Martí, R. (2014). Tabu search with strategic oscillation for the quadratic minimum spanning tree. *IIE Transactions*, 46(4):414–428.

[84] Malucelli, F. and Pretolani, D. (1994). Quadratic semi-assignment problems on structured graphs. *Ricerca Operativa*, 69:57–78.

[85] Malucelli, F. and Pretolani, D. (1995). Lower bounds for the quadratic semi-assignment problem. *European Journal of Operational Research*, 83(2):365–375.

[86] Marsten, R., Hogan, W., and Blankenship, J. W. (1975). The boxstep method for large-scale optimization. *Operations Research*, 23(3):389–405.

[87] Mauri, G. R. and Lorena, L. A. N. (2011). Lagrangean decompositions for the unconstrained binary quadratic programming problem. *International Transactions in Operational Research*, 18(2):257–270.

[88] Mauri, G. R. and Lorena, L. A. N. (2012). A column generation approach for the unconstrained binary quadratic programming problem. *European Journal of Operational Research*, 217(1):69 – 74.

[89] Michelon, P. and Maculan, N. (1991). Lagrangian decomposition for integer nonlinear programming with linear constraints. *Mathematical Programming*, 52(2):303–313.

[90] Monteiro, R. D. and Zhang, Y. (1998). A unified analysis for a class of long-step primal-dual path-following interior-point algorithms for semidefinite programming. *Mathematical Programming*, 81(3):281–299.

[91] Murakami, K. and Kim, H. S. (1997). Comparative study on restoration schemes of survivable atm networks. In *INFOCOM'97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, volume 1, pages 345–352. IEEE.

[92] Noon, C. E. and Bean, J. C. (1991). A lagrangian based approach for the asymmetric generalized traveling salesman problem. *Operations Research*, 39(4):623–632.

[93] Nowak, I. (2005). Lagrangian decomposition of block-separable mixed-integer all-quadratic programs. *Mathematical programming*, 102(2):295–312.

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

[95] Oustry, C. L. F. (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.

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

[97] Palubeckis, G., Rubliauskas, D., and Targamadze, A. (2010). Metaheuristic approaches for the quadratic minimum spanning tree problem. *Information Technology and Control*, 39(4):257–268.

[98] Pereira, D., Gendreau, M., and Cunha, A. (2013). Lower bounds and exact algorithms for the quadratic minimum spanning tree problem. *Online Optimization.*

[99] Polak, G. G. (2005). On a special case of the quadratic assignment problem with an application to storage-and-retrieval devices. *Annals of Operations Research*, 138(1):223–233.

[100] Poljak, S., Rendl, F., and Wolkowicz, H. (1995). A recipe for semidefinite relaxation for (0, 1)-quadratic programming. *Journal of Global Optimization*, 7(1):51–73.

[101] Pollatschek, M., Gershoni, N., and Radday, Y. (1976). Optimization of the typewriter keyboard by simulation. *Angewandte Informatik*, 17(0):438–439.

[102] Prim, R. C. (1957). Shortest connection networks and some generalizations. *Bell System Technical Journal*, 36(6):1389–1401.

[103] Ramakrishnan, K. G., Resende, M. G. C., Ramachandran, B., and Pekny, J. F. (2002). Tight QAP bounds via linear programming. In Pardalos, P. M., Migdalas, A., and Burkard, R. E., editors, *Combinatorial and Global Optimization*, chapter 19, pages 297–303. World Scientific Publishing, Singapore.

[104] Rendl, F. (1986). Quadratic assignment problems on series-parallel digraphs. *Zeitschrift für Operations Research*, 30(3):A161–A173.

[105] Rendl, F. (1999). Semidefinite programming and combinatorial optimization. *Applied Numerical Mathematics*, 29(3):255 – 281.

[106] Resende, M. G., Ramakrishnan, K., and Drezner, Z. (1995). Computing lower bounds for the quadratic assignment problem with an interior point algorithm for linear programming. *Operations Research*, 43(5):781–791.

[107] Roupin, F. (2004). From linear to semidefinite programming: an algorithm to obtain semidefinite relaxations for bivalent quadratic problems. *Journal of Combinatorial Optimization*, 8(4):469–493.

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

[109] Saito, H., Fujie, T., Matsui, T., and Matuura, S. (2009). A study of the quadratic semi-assignment polytope. *Discrete Optimization*, 6(1):37 – 50.

[110] Sen, S., Pillai, R., Joshi, S., and Rathi, A. K. (2001). A mean-variance model for route guidance in advanced traveler information systems. *Transportation Science*, 35(1):37–49.

[111] Sherali, H. D. and Adams, W. P. (1998). *A reformulation-linearization technique for solving discrete and continuous nonconvex problems*, volume 31. Springer.

[112] Sherali, H. D. and Smith, J. C. (2007). An improved linearization strategy for zero-one quadratic programming problems. *Optimization Letters*, 1(1):33–47.

[113] Sivakumar, R. A. and Batta, R. (1994). The variance-constrained shortest path problem. *Transportation Science*, 28(4):309–316.

[114] Steinberg, L. (1961). The backboard wiring problem: A placement algorithm. *Siam Review*, 3(1):37–50.

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

[116] Taillard, E. D. (1995). Comparison of iterative searches for the quadratic assignment problem. *Location Science*, 3(2):87–105.

[117] Thorup, M. (1997). Undirected single source shortest paths in linear time. In *Foundations of Computer Science, 1997. Proceedings., 38th Annual Symposium on*, pages 12–21. IEEE.

[118] Xia, Y. and Yuan, Y.-X. (2006). A new linearization method for quadratic assignment problems. *Optimisation Methods and Software*, 21(5):805–818.

[119] Zhang, H., Beltran-Royo, C., and Ma, L. (2013). Solving the quadratic assignment problem by means of general purpose mixed integer linear programming solvers. *Annals of Operations Research*, 207(1):261–278.

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

[121] Zhu, Y.-R. (2007). *Recent advances and challenges in quadratic assignment and related problems*. PhD thesis, University of Pennsylvania.