A polynomial time algorithm for the linearization problem of the QSPP and its applications

Hao Hu * Renata Sotirov †

Abstract

Given an instance of the quadratic shortest path problem (QSPP) on a digraph G, the linearization problem for the QSPP asks whether there exists an instance of the linear shortest path problem on G such that the associated costs for both problems are equal for every s-t path in G. We prove here that the linearization problem for the QSPP on directed acyclic graphs can be solved in $\mathcal{O}(nm^3)$ time, where n is the number of vertices and m is the number of arcs in G.

By exploiting this linearization result, we introduce a family of lower bounds for the QSPP on acyclic digraphs. The strongest lower bound from this family of bounds is the optimal solution of a linear programming problem. To the best of our knowledge, this is the first study in which the linearization problem is exploited to compute bounds for the corresponding optimization problem. Numerical results show that our approach provides the best known linear programming bound for the QSPP.

We also present a lower bound for the QSPP that is derived from a sequence of problem reformulations, and prove finite convergence of that sequence. This lower bound belongs to our family of linear bounds, and requires less computational effort than the best bound from the family.

Keywords: quadratic shortest path problem, linearization problem, directed acyclic graph, lower bounds

1 Introduction

The quadratic shortest path problem is the problem of finding a s-t path in a directed graph such that the sum of interaction costs over all pairs of arcs on the path is minimized. The QSPP is an NP-hard optimization problem, see [12, 21]. The adjacent QSPP is a special case of the QSPP, where the interaction cost between two arcs is assumed to be

*CentER, Department of Econometrics and OR, Tilburg University, The Netherlands, h.hu@uvt.nl
†Department of Econometrics and OR, Tilburg University, The Netherlands, r.sotirov@uvt.nl

zero if they are not adjacent. It is proven in [12, 20] that the adjacent QSPP on cyclic digraphs is NP-hard. However, the adjacent QSPP on a directed acyclic graph (DAG) is solvable in polynomial time [21].

The QSPP has many applications including forming network protocols [16], route-planning model [22], transportation of hazardous materials [23]. A detailed overview of the QSPP and its applications is given in the paper by Rostami et al. [20].

There are several approaches for solving the QSPP. Buchheim and Traversi [2] provide a generic method for solving binary quadratic programming problems, including the QSPP. Rostami et al. [20] present a reformulation-based lower bound which is incorporated into a branch and bound algorithm to obtain an optimal solution for the QSPP. Hu and Sotirov [13] derive several semidefinite programming relaxations with a matrix variable of order $m + 1$, where m is the number of arcs in the graph. Their strongest relaxation is solved by the alternating direction method of multipliers, and used within a branch and bound algorithm.

We study here the linearization problem for the QSPP and related linear programming bounds. An instance of the quadratic shortest path problem is said to be linearizable if there exists an instance of the linear shortest path problem (SPP) on the same graph such that the QSPP and SPP objective values are the same for every s-t path. The linearization problem for the QSPP asks whether an instance of the QSPP is linearizable. Since there exist efficient algorithms for solving the shortest path problem, e.g., the Dijkstra algorithm [8, 12] and the Floyd-Warshall algorithm [9, 24], a linearizable QSPP instance is solvable in polynomial time. The authors in [12] provide a polynomial time algorithm that verifies if a QSPP instance on a directed grid graph is linearizable. Hu and Sotirov [12] also present necessary conditions for a QSPP instance on complete digraphs to be linearizable. These conditions are also sufficient when the complete digraph has only four vertices.

The linearization problem is studied in context of other combinatorial optimization problems. Kabadi and Punnen [14] give a necessary and sufficient condition for an instance of the quadratic assignment problem (QAP) to be linearizable, and develop a polynomial-time algorithm to solve the corresponding linearization problem. The linearization problem for the Koopmans-Beckmann QAP is studied in [19]. Linearizable special cases of the QAP are studied in [1, 6, 17]. In [5] it is shown that the linearization problem for the bilinear assignment problem can be verified in polynomial time. The linearization problem for the quadratic minimum spanning tree problem was considered in [4]. Punnen and Woods [18] provide necessary and sufficient conditions for which a cost matrix of the quadratic traveling salesman problem is linearizable.

Main results and outline.
In this paper, we present necessary and sufficient conditions for an instance of the QSPP on a directed acyclic graph to be linearizable. Those conditions can be verified in $\mathcal{O}(nm^3)$ time, where n is the number of vertices and m is the number of arcs in the graph. In the

other words, we can provide an answer to the linearization problem on DAGs in polynomial time. Our linearization result is a generalization of the corresponding result for the QSPP on directed grid graphs from [12]. By exploiting our linearization result, we introduce a family of lower bounds for the QSPP on DAGs. We show that the best bound from the family of bounds can be obtained by solving a linear programming problem. The numerical results show that our strongest linear programming bound outperforms existing linear programming bounds. To the best of our knowledge, we are the first to exploit the linearization problem of a combinatorial optimization problem in order to compute its bound.

We also present a lower bound for the QSPP based on a sequence of the problem reformulations. The idea of successive bound improvements was introduced in the context of the quadratic assignment problem in [3], and implemented for the QSPP in [20]. However, we use different cost lifting than the authors in [20], which enables us to prove finite convergence of the sequence of the bounds when the cost matrix is rational. Another interesting result is that our reformulation-based bound belongs to the family of linear bounds that is introduced in this paper.

The paper is organized as follows. In Section 2, we formally introduce the quadratic shortest path problem, the linear shortest path problem, and the linearization problem for the QSPP. In Section 3, we present a polynomial-time algorithm that verifies whether a QSPP instance on a directed ayclic graph is linearizable. In Section 4, we introduce a family of lower bounds for the QSPP on DAGs by exploiting the linearization result from Section 3. We also show how to compute the strongest bound from the family in the same section. A variant of the reformulation-based bound for the QSPP is proposed in Section 5. Numerical result are given in Section 6.

2 The quadratic shortest path problem

In this section, we formally introduce the (quadratic) shortest path problem, and define the QSPP linearization problem.

Let $G = (V, A)$ be a directed graph with n vertices and m arcs, and s, t two distinguished vertices in G. A path is a sequence of distinct vertices $(v_1, \ldots, v_k)$ such that $(v_i, v_{i+1}) \in A$ for $i = 1, \ldots, k-1$. A s-t path is a path $P = (v_1, v_2, \ldots, v_k)$ from the source vertex $s = v_1$ to the target vertex $t = v_k$. Let $\mathcal{P}$ be the set of all s-t paths in G.

A cost matrix $Q = (q_{ef}) \in \mathbb{R}^{m \times m}$ is a symmetric matrix such that $q_{ef} + q_{fe}$ is the interaction cost between arcs e and f for each pair of distinct arcs $(e, f) \in A \times A$, and $q_{e,e}$ is the linear cost of the arc e. Note that we allow here that Q has also negative elements. The quadratic cost of a path $P \in \mathcal{P}$ is given by $C(P, Q) = \sum_{e,f \in P} q_{e,f}$. Here $e \in P$ means that the arc e is on the path P. Given a cost matrix Q, the quadratic shortest path problem

is:

$$\text{minimize}\Big\{ \sum_{e,f \in P} q_{e,f} \mid P \in \mathcal{P} \Big\}. \tag{1}$$

For simplicity, we refer to a QSPP instance only by its cost matrix Q, if it is clear from the context what are the source and target vertices.

In the case that there are no interaction costs between arcs, we collect linear costs of the arcs into the cost vector $c \in \mathbb{R}^m$. For a path $P \in \mathcal{P}$, its linear cost is given by $C(P, c) = \sum_{e \in P} c_e$. Therefore, the shortest path problem with the cost vector c is:

$$\text{minimize}\Big\{ \sum_{e \in P} c_e \mid P \in \mathcal{P} \Big\}. \tag{2}$$

For a cost matrix Q, the quadratic shortest path *linearization problem on a DAG* asks whether there exists a cost vector $c \in \mathbb{R}^m$ such that $C(P, Q) = C(P, c)$ for every s-t paths $P \in \mathcal{P}$. If such a cost vector c exists, then we call it a *linearization vector* of Q. The cost matrix Q is said to be linearizable if its *linearization vector* c exists.

3 The QSPP linearization problem on DAGs

In this section, we first introduce several assumptions and definitions. Then, we derive necessary and sufficient conditions for an instance of the QSPP on a directed acyclic graph to be linearizable, see Theorem 3.8. We also show that those conditions can be verified in $\mathcal{O}(nm^3)$ time. This result is a generalization of the corresponding results for the QSPP on directed grid graphs from [12].

We have the following assumptions in this section:

(i) G is a directed acyclic graph;

(ii) the vertices $v_1, \ldots, v_n$ are topologically sorted, that is, $(v_i, v_j) \in A$ implies $i < j$;

(iii) for each vertex v, there exists at least one s-t path containing v;

(iv) the diagonal entries of the cost matrix Q are zeros.

Note that these assumptions do not restrict the generality. For instance, assumption (iv) is not restrictive as Q is linearizable if and only if $Q + \mathrm{Diag}(c)$ is linearizable for any cost vector c, see [12].

Here, we choose and fix an arbitrary labeling of the arcs. The vertices $v_2, \ldots, v_{n-1}$, e.g., those between the source vertex v_1 and the target vertex v_n, are called the *transshipment vertices*. For each transshipment vertex, we pick the outgoing arc with the smallest index and call it a *non-basic arc*. The remaining arcs are *basic*. Thus there are $n - 2$ non-basic arcs and $m - n + 2$ basic arcs.

Note that for a linearizable cost matrix its linearization vector is not necessarily unique. However, we would like to restrict our analysis to the linearization vectors that are in a unique, reduced form. For this purpose we introduce the following definitions, see also [12].

Definition 3.1. *We say that the cost vectors c_1 and c_2 are* equivalent *if $C(P, c_1) = C(P, c_2)$ for all $P \in \mathcal{P}$.*

Definition 3.2. *The* reduced form *of a cost vector c is an equivalent cost vector $R(c)$ such that $(R(c))_e = 0$ for every non-basic arc e.*

The existence of the reduced form of the cost vector c follows from the following transformation. Let v be a transshipment vertex, and f the non-basic arc going from v. Define $\hat{c}$ as follows:

$$\hat{c}_e := \begin{cases} c_e - c_f & \text{if } e \text{ is an outgoing arc from vertex } v, \\ c_e + c_f & \text{if } e \text{ is an incoming arc to } v, \\ c_e & \text{otherwise.} \end{cases} \tag{3}$$

It is not difficult to verify that $\hat{c}$ and c are equivalent. Furthermore, if we apply this transformation at each transshipment vertex v in the reverse topological order i.e., $v_{n-1}, \ldots, v_2$, then the obtained cost vector, after $n-2$ transformations, is in the reduced form. Moreover the resulting cost vector is equivalent to c. Let us define now critical paths, see also [12].

Definition 3.3. *For a basic arc $e = (u, v)$, the associated* critical path *P_e is a s-t path containing arc e and determined as follows. Choose an arbitrary s-u path P_1, and take for P_2 the unique v-t path with only non-basic arcs. Then, the critical path $P_e = (P_1, P_2)$ is the concatenation of the paths P_1 and P_2.*

The uniqueness of P_2 in Definition 3.3 follows from the fact that each transshipment vertex has exactly one outgoing arc that is non-basic and G is acyclic. Clearly, to each basic arc e we can associate one critical path P_e as given above.

The following result shows that for a linearizable cost matrix Q there exists a unique linearization vector in reduced form. The uniqueness is up to the choice of non-basic arcs and critical paths.

Proposition 3.4. *Let $Q \in \mathbb{R}^{m \times m}$ be a linearizable cost matrix for the QSPP, and $c \in \mathbb{R}^m$ its linearization vector. Then, the reduced form of c, $R(c) \in \mathbb{R}^m$, is uniquely determined by the costs of the critical paths in the underlying graph G.*

Proof. Let M be a binary matrix whose rows correspond to the s-t paths in G and columns correspond to the basic arcs. In particular, $M_{P,e} = 1$ if and only if the path P contains the basic arc e. Let b be the vector whose elements contain quadratic costs of s-t paths.

Let $\hat{c}_B \in \mathbb{R}^{m-n+2}$ is the subvector of $R(c)$ composed of the elements corresponding to the basic arcs. Then, $\hat{c}_B$ satisfies the linear system $M\hat{c}_B = b$. In order to show the uniqueness of $\hat{c}_B$ it suffices to prove that the rank of M equals $m - n + 2$, which is the number of the basic arcs.

Let $\bar{M}$ be a square submatrix of M of size $(m - n + 2) \times (m - n + 2)$ whose rows correspond to the critical paths. Let $\mathcal{C}_i$ be the set of the basic arcs emanating from vertex v_i for $i = 1, \ldots, n - 1$. Since the sets $\mathcal{C}_1, \ldots, \mathcal{C}_{n-1}$ partition the set of the basic arcs, they can be used to index the matrix $\bar{M}$. Upon rearrangement, $\bar{M}$ is a block matrix such that the (i, j)th block $\bar{M}^{ij}$ is the submatrix whose rows and columns correspond to $\mathcal{C}_i$ and $\mathcal{C}_j$, respectively. It is readily to see that every diagonal block $\bar{M}^{ii}$ is an identity-matrix. Furthermore, the block $\bar{M}^{ij}$ is a zero matrix for $i < j$. To see this, we first recall that the vertices are topologically ordered. Then, note that for the critical path that is associated to the arc $e = (i, j)$, all arcs visited after e are non-basic by construction. Thus, the rank of $\bar{M}$ is $m - n + 2$, and this finishes the proof. □

From the previous proposition, it follows that for a linearizable cost matrix Q its linearization vector can be computed easily from the costs of the critical paths. However, the above calculation of the unique linear cost vector in reduced form can be performed even when the linearizability of Q is not known. Since the resulting vector does not have to be a linearization vector, we call it pseudo-linearization vector. In particular, we have the following definition, see also [12].

Definition 3.5. *The pseudo-linearization vector of the cost matrix $Q \in \mathbb{R}^{m\times m}$ is the unique cost vector $p \in \mathbb{R}^m$ in reduced form such that $C(P, Q) = C(P, p)$ for every critical path P.*

Here, the uniqueness is up to the choice of non-basic arcs and critical paths. Recall that the pseudo-linearization vector can be computed also for a non-linearizable cost matrix. The following lemma shows that for linearizable Q its pseudo-linearization vector coincides with the linearization vector in reduced form.

Lemma 3.6. *Let Q be linearizable. Then the corresponding linearization vector in reduced form and the pseudo-linearization vector are equal.*

Proof. Let c be the linearization vector of Q in reduced form, and p the pseudo-linearization vector of Q. Then, $C(P, c) = C(P, p)$ for each critical path P. From Proposition 3.4 it follows that $c = p$ since both cost vectors are in the reduced form. □

If we change the input target vertex from t to another vertex v, some arcs and vertices have to be removed from G in order to satisfy assumption (iii). This results in a reduced QSPP instance. To simplify the presentation, we introduce the following notation.

Notation	Definition
$G_v = (V_v, A_v)$	an induced subgraph of G for which assumption (iii) is satisfied with target vertex v
Q_v	an $\lvert A_v\rvert \times \lvert A_v\rvert$ submatrix of Q whose rows and columns correspond to the arcs in A_v
$R_v(\cdot)$	the linear operator that maps a cost vector on G_v to its reduced form
p_v	the pseudo-linearization vector of Q_v

Table 1: Notation with respect to target vertex v.

The next result from [12] establishes a relationship between the linearization vector of Q_t and the linearization vector of Q_v where $(v,t) \in A$.

Lemma 3.7. [12] *The cost vector $c \in \mathbb{R}^m$ is a linearization of $Q_t \in \mathbb{R}^{m\times m}$ if and only if the cost vector $T_e(c) \in \mathbb{R}^{|A_v|}$ given by*

$$\big(T_e(c)\big)_{e'} = \begin{cases} c_{e'} - 2\cdot q_{e,e'} & \text{if } e' = (u,w) \in A_v \text{ and } u \neq s \\ c_{e'} - 2\cdot q_{e,e'} + c_e & \text{if } e' = (u,w) \in A_v \text{ and } u = s \end{cases}, \tag{4}$$

is a linearization of Q_v for every vertex v such that $e = (v,t) \in A$.

Now, we are ready to prove the main result in this section.

Theorem 3.8. *Let $R_v(\cdot)$ and p_v be defined as in Table 1, and T_e given as in Lemma 3.7. Then, the pseudo-linearization vector p_t is a linearization of Q_t if and only if $(R_u \circ T_e)(p_v) = p_u$ for every $e = (u,v) \in A$.*

Proof. If Q_t is linearizable, then it follows from Lemma 3.6 and Lemma 3.7 that for every vertex v, the cost matrix Q_v is linearizable with the linearization vector p_v. Thus, $(R_u \circ T_e)(p_v) = p_u$ for every arc $e = (u,v) \in A$.

Conversely, assume that Q_t is not linearizable. Then, from Lemma 3.7, it follows that there exists an arc $e = (v,t) \in A$ such that the vector $T_e(p_t)$ is not a linearization vector of Q_v. Let us distinguish the following two cases:

(i) If Q_v is linearizable, then p_v is its linearization vector and $(R_v \circ T_e)(p_t) \neq p_v$ by Lemma 3.6.

(ii) If Q_v is not linearizable, then we again distinguish two cases. Thus, we repeat the whole argument to Q_v.

This recursive process must eventually end up with case (i), as the number of vertices in the underlying graph decreases in each recursion step, and every cost matrix on a graph with at most three vertices is linearizable. Thus, we obtain $(R_u \circ T_e)(p_v) \neq p_u$ for some $e = (u,v) \in A$. □

Note that the iterative procedure from Theorem 3.8 provides an answer to the QSPP linearization problem. Moreover, it returns the linearization vector in reduced form if such exists.

The quadratic cost of a s-t path can be computed in $\mathcal{O}(m^2)$ steps, and thus we need $\mathcal{O}(m^3)$ steps for the $m-n+2$ critical paths. The pseudo-linearization of Q can be obtained in $\mathcal{O}(m^2)$ steps by solving a linear system whose left-hand-side is a lower triangular square matrix $\bar{M}$ of order $m-n+2$, see Proposition 3.4. Since there are n vertices, computing all pseudo-linearizations requires $\mathcal{O}(nm^3)$ steps. The rest of the computation takes at most $\mathcal{O}(m^3)$ steps. Thus the complexity of the algorithm given in Theorem 3.8 is $\mathcal{O}(nm^3)$.

4 The linearization-based bounds on DAGs

In this section, we first introduce a family of lower bounds based on the linearization problem for the QSPP on DAGs from the previous section. Then, we show that the strongest bound in this family is an optimal solution of a linear programming problem. Our numerical results in Section 6 verify the quality of so obtained linear bound.

Before we proceed, let us recall the linear shortest path problem. Let $\mathcal{I}_G \in \mathbb{R}^{n\times m}$ be the incidence matrix of a digraph G. The optimal value $OPT(c)$ of the shortest path problem on G with the cost vector $c \in \mathbb{R}^m$ can be obtained by solving the following linear program:

$$OPT(c) = \max_{y\in\mathbb{R}^n}\{y_s - y_t \mid \mathcal{I}_G^{\mathrm{T}} y \leq c\}. \tag{5}$$

If a cost matrix Q of the QSPP is linearizable with the linearization vector c, then the optimal solution for the QSPP can be obtained by solving the linear programming problem (5). However, if Q is not linearizable then one might look for a linearizable matrix that is "close" to Q in order to obtain a bound for the problem. For that purpose, and for a given cost matrix $Q \in \mathbb{R}^{m\times m}$, we introduce the set $\mathcal{S}(Q)$ of cost vectors $c' \in \mathbb{R}^m$ such that

$$\text{(i) } c' \text{ is a linearization vector of } Q'; \tag{6}$$

$$\text{(ii) } Q - Q' \text{ is elementwise non-negative.} \tag{7}$$

For a cost vector $c' \in \mathcal{S}(Q)$ it is not difficult to verify that $C(P,c') \leq C(P,Q)$ for every s-t path P. Therefore $v_{LBB}(c') := OPT(c')$ with $c' \in \mathcal{S}(Q)$ is a lower bound for the QSPP. The set of all bounds obtained in this way forms a new family of lower bounds for the QSPP on acyclic digraphs. We call $v_{LBB}(c')$ the *linearization-based bound* (LBB) with respect to c'.

The strongest linearization-based bound (LBB_*) on a DAG is given as follows:

$$v_{LBB_*} := \max_{c'\in\mathbb{R}^m}\{v_{LBB}(c') \mid c' \in \mathcal{S}(Q)\} = \max_{\substack{c'\in\mathbb{R}^m \\ y\in\mathbb{R}^n}}\{y_s - y_t \mid c' \in \mathcal{S}(Q),\ \mathcal{I}_G^{\mathrm{T}} y \leq c'\}, \tag{8}$$

where the second equality exploits (5). Let us show that $\mathcal{S}(Q)$ is a polyhedron for the QSPP is on a DAG. Therefore, v_{LBB_*} can be computed by solving the linear program (8).

Proposition 4.1. *Let G be an acyclic digraph. Then $\mathcal{S}(Q)$ is a polyhedron.*

Proof. To show that $\mathcal{S}(Q)$ is a polyhedron, we derive linear equalities and inequalities to describe $\mathcal{S}(Q)$. The constraints (7) are clearly linear inequalities. To verify the constraint (6), we need to show that $(R_u \circ T_e)(p_v) = p_u$ for every $e = (u, v) \in A$ from Theorem 3.8 can be obtained by using linear equalities. Namely, for a given cost vector c, its reduced form can be obtained by applying a number of linear operations, see Section 3. In particular, each operation corresponds to an elementary row operation to c. Thus, there exist elementary matrices $M_k, \ldots, M_1$ such that $R_u(c) = M_k M_{k-1} \cdots M_1 c$ is the reduced form of c. Similarly, the linear operator T_e from Lemma 3.7, and the pseudo-linearization vector p_v can be obtained by a number of linear operations. To conclude, the constraint (6) can be described by linear equalities.

Putting all together, $\mathcal{S}(Q)$ can be described by a linear system of size $\mathcal{O}(m^2) \times \mathcal{O}(m^2)$ in variables Q' and c'. □

It is not clear whether $\mathcal{S}(Q)$ is a polyhedron in general. Thus, optimization over $\mathcal{S}(Q)$ might be difficult if the considered graph is not a DAG. However, even in the case that the digraph is not a DAG a cost vector $c' \in \mathcal{S}(Q)$ yields a lower bound $OPT(c')$ for the QSPP.

5 A reformulation-based bound

Carraresi et al. [3] present a bounding scheme for the quadratic assignment problem that uses a sequence of equivalent formulations of the problem. Inspired by [3], Rostami et al. [20] proposed an iterative procedure to compute a lower bound for the QSPP. Based on the ideas from [3, 20], we propose another iterative procedure to compute a sequence of improving bounds for the QSPP. We prove that this sequence converges in a finite number of steps when elements of Q are rational. We also show that the sequence of the bounds obtained by our procedure belongs to the family of the linearization-based bounds introduced in Section 4.

We first describe the first step of the iterative, bounding procedure. The first step is the same as the first step in [20]. Let us fix an arc $e \in A$. Denote by $Q_{:,e}$ the eth column of Q, and I_e the eth column of the $m \times m$ identity matrix. Let $b \in \mathbb{R}^n$ be a vector such that

$b_s = 1, b_t = -1$ and $b_v = 0$ if $v \in V \backslash \{s,t\}$. Consider the following primal and dual pair:

$$\begin{aligned} (P) \qquad & \min_{x \in \mathbb{R}^m} \{\ Q_{:,e}^{\mathrm{T}} x \mid \mathcal{I}_G x = b,\ x_e = 1,\ x \geq 0\ \} \\ (D) \qquad & \max_{\substack{y_e \in \mathbb{R}^n, \\ z_e \in \mathbb{R}}} \{\ b^{\mathrm{T}} y_e + z_e \mid \mathcal{I}_G^{\mathrm{T}} y_e + I_e z_e \leq Q_{:,e}\ \}. \end{aligned} \tag{9}$$

Let c'_e be the optimal value of the above primal-dual pair. The optimal value c'_e can be interpreted as a lower bound for the cost of the shortest s-t path containing arc e with respect to the cost matrix Q. Let $c' = (c'_e) \in \mathbb{R}^m$ be a cost vector whose m elements are obtained by solving m optimization problems of the form (9). Define the cost matrix Q' such that its eth column is given by $Q'_{:,e} := \mathcal{I}_G^{\mathrm{T}} y_e^* + I_e z_e^*$, where (y_e^*, z_e^*) is the optimal solution for the dual problem (D) with $e \in A$.

In [20], the authors prove that the cost vector c' is a linearization vector of Q', and that the cost matrix $Q - Q'$ is nonnegative. Therefore, the cost vector c' yields a lower bound $OPT(c')$ for the QSPP. In the view of the previous section we have that $OPT(c')$ equals the linearization-based bound with respect to c', i.e., $v_{LBB}(c')$. Moreover, this bound is known as the Gilmore-Lawler type bound (GLT) (see [10, 15, 20]) and we denote it from now on by v_{GLT}.

By applying the same procedure to the reformulated problem with objective $Q - Q'$, one would not obtain any further improvement of the lower bound, see [20]. However, the above described procedure can be applied repetitively if $Q - Q'$ is "shifted" properly. To be more precise, let $sym : \mathbb{R}^{m \times m} \to \mathbb{R}^{m \times m}$ be given by $sym(M) = \frac{1}{2}(M + M^{\mathrm{T}})$ and $shift : \mathbb{R}^{m \times m} \to \mathbb{R}^{m \times m}$ by

$$(shift(M))_{e,f} = \begin{cases} \lceil sym(M)_{e,f} \rceil & \text{if } e > f \\ \lfloor sym(M)_{f,e} \rfloor & \text{otherwise} \end{cases}, \tag{10}$$

for every pair of arcs $e \neq f$. Note that this shifting is proper only if the matrix that needs to be shifted is integer. However, we can assume here that the elements of Q are rational and then re-scale them to integer values, see step 1 in Algorithm 1. Applying the shifting operation (10) after each reformulation of the problem, results in a sequence of improving bounds, see Algorithm 1. The bound obtained at the end of the algorithm is our reformulation-based bound denoted by RBB. The authors from [20] apply different kind of shifting, which results in a different lower bound.

The next result establishes the finite convergence property of Algorithm 1.

Proposition 5.1. *Algorithm 1 terminates in a finite number of iterations if Q is rational.*

Proof. For simplicity, we assume that Q is integer. Assume that Q is nonnegative. Since the coefficient matrices of the primal and dual linear programs (9) are totally unimodular,

Algorithm 1: Reformulation-based bound

Input : A QSPP instance with rational Q, counter $k \leftarrow 0$;
Output: The reformulation-based bound v_{RBB};

$Q_0 \leftarrow \alpha Q$ (α is the least common denominator of the entries in Q) ;
repeat
  $k \leftarrow k+1$;
  Compute c'_k and Q'_k from the dual program (9) for Q_{k-1};
  $Q_k \leftarrow shift(Q_{k-1} - Q'_k)$;
until $||c'_k|| = 0$;
$v_{RBB} \leftarrow \frac{1}{\alpha} OPT(\sum_{i=1}^{k} c'_k)$;

it follows that c'_1 and Q'_1 are integral. Since $Q \geq 0$, we have that the vector c'_1 is also nonnegative. Now, if there exists an arc e such that $(c'_1)_e \geq 1$, the algorithm continues to iterate. After a finite number of iterations, we have that $||c'_k|| = 0$ for some k since $\sum_{i=1}^{k} c'_k$ is element-wise upper bounded by the sum of entries of Q.

If Q contains negative entries, then we use the fact that $Q - Q'_1 \geq 0$, and thus $shift(Q_- Q'_1)$, is also nonnegative. Therefore we are in the previous setting. □

Let us show that the RBB is a member of the family of linearization-based bounds.

Proposition 5.2. *Let $Q_0, \ldots, Q_k$ be the sequence of the reformulated cost matrices obtained from Algorithm 1. Let $Q'_1, \ldots, Q'_k$ be the sequence of linearizable matrices, and $c'_1, \ldots, c'_k$ the corresponding linearization vectors. Then, $\frac{1}{\alpha} \sum_{i=1}^{k} c'_k \in \mathcal{S}(Q)$, where Q is the rational input cost matrix and $Q_0 = \alpha Q$.*

Proof. For simplicity, we assume that Q is integer and thus $\alpha = 1$. To prove $\sum_{i=1}^{k} c'_k \in \mathcal{S}(Q)$, we need to verify constraints (6) and (7). Constraint (6) holds as c'_i is a linearization vector of Q'_i and therefore also of $sym(Q'_i)$, for every $i \in \{1, \ldots, k\}$. To verify constraint (7), we note that by applying $Q_i = shift(Q_{i-1} - Q'_i)$ $(i = 1, \ldots, k)$ recursively, it follows

$$sym(Q_k) = sym(Q_{k-1} - Q'_k) = sym(Q_0 - \sum_{i=1}^{k} Q'_i) = Q - \sum_{i=1}^{k} sym(Q'_i).$$

Here, the first and second equalities use the fact that $sym(shift(Q_i)) = sym(Q_i)$ $(i = 0, \ldots, k)$, and the last equality uses the fact that $Q = Q_0$. Since $Q_k \geq 0$, we have $sym(Q_k) \geq 0$ and the required inequality follows. In the case that Q is not integer, the proof follows after re-scaling by α. □

As a direct consequence of Proposition 5.2, we have the following inequality.

Corollary 5.3. *If G is acyclic, then $v_{RBB} \leq v_{LBB_*}$.*

In [20], the authors use sym function instead of $shift$ function to compute a lower bound for the QSPP by the here described iterative procedure. Using a similar argument as in Lemma 5.2, one can show that the bound from [20] is also a member of the family of linearization-based bounds introduced in Section 4. However, in their case the finite convergence result may not hold, see Lemma 5.1.

To sum up, one can find an element from $\mathcal{S}(Q)$ by solving m linear programs (9). The resulting vector can be exploited to reformulate the objective of the QSPP, and then again solve m linear programs. This iterative procedure results in a reformulation-based bound. Reformulation-based bounds depend on the type of the shifting. A reformulation-based bound does not, in general, yield the strongest linearization-based bound (see Section 6). However, one can compute a reformulation-based bound for the QSPP even when there is not known description of $\mathcal{S}(Q)$. Therefore, a reformulation-based bound can be computed also for the QSPP on a cyclic digraph when the cost matrix is nonnegative.

6 Numerical experiments

In this section we compare several lower bounding approaches including the Gilmore-Lawler type (GLT) bound, the strongest linearization-based bound (LBB_*) from Section 4, and the reformulation-based bound (RBB) from Section 5. We also provide numerical results for the linear programming relaxation of the mixed integer linear programming (MILP) formulation of the QSPP from [20]. More precisely, we compute the MILP formulation with relaxed binary constraints. The obtained lower bound is denoted by LP_{milp}. The optimal values of the test instances are obtained by using the branch-and-bound algorithm from [13] or Cplex MIQP solver.

All lower bounds are implemented in Matlab on the machine with an Intel(R) Core(TM) i7-7700 CPU, 3.60GHz and 16 GB memory. The bounds LP_{milp}, GLT, RBB and LBB_* are computed by Cplex [7].

We consider the following test instances.

(i) GRID1 is a QSPP instance on the directed grid graph, see [13]. The directed grid graph of size $p \times q$ has vertices $v_{i,j}$ for $i = 1, \ldots, p$, $j = 1, \ldots, q$ and arcs $(v_{i,j}, v_{k,l})$ if $|i-k| + |j-l| = 1$ and $i \leq k$ and $j \leq l$. The source vertex is $s = v_{1,1}$ and the target vertex is $t = v_{p,q}$.

(ii) GRID3 is a QSPP instance on the directed flow grid graph, see [13]. The directed flow grid graph of size $p \times q$ contains all vertices and arcs as the directed grid graph of size $p \times q$, as well as two extra vertices; a source vertex s and a target vertex t. Additionally, there are p arcs from s to the vertices $v_{i,1}$ $(i = 1, \ldots, p)$, and p arcs

from vertices $v_{i,q}$ $(i = 1, \ldots, p)$ to t.

(iii) PAR-K is a QSPP instance on the incomplete K-partite graph, see [13]. The incomplete K-partite graph is a directed acyclic graph whose vertex set is partitioned into K disjoint sets $V_1 \cup V_2 \ldots \cup V_K$ such that (u, v) is an arc if $u \in V_i$ and $v \in V_{i+1}$ for $i = 1, \ldots, K-1$. We set $V_1 = \{s\}, V_2 = \{t\}$, and $|V_i| = K$ for $i = 2, \ldots, K-1$.

(iv) TOUR is a QSPP instance on the tournament graph. The tournament graph of size n is a directed acyclic graph on the vertex set $\{1, \ldots, n\}$ such that (i, j) is an arc if $i < j$. The source vertex is $s = 1$ and the target vertex is $t = n$.

We generate the cost matrix Q in the following way. Let d be a density parameter. For each pair of arcs e and f, the random variable $W_{(e,f)}$ is uniformly distributed on the support $\{1, \ldots, 5\}$. The realization of $W_{(e,f)}$ is $w_{e,f}$. The cost matrices are generated in two steps. In the first step, we initialize the cost matrix as follows. For GRID1 and PAR-K instances, we set $q_{e,f} = w_{e,f}$ for every pair of arcs e, f. For GRID3 instance, we set $q_{e,f} = w_{e,f}$ if neither e nor f has the form $(v_{i,j}, v_{i+1,j})$, and $q_{e,f} = 0$ otherwise. For TOUR instance, we set $q_{ef} = |i - j|^2$ for every pair of arcs $e = (i, j)$ and $f = (k, l)$ such that $|i - j| = |k - l|$, and $q_{e,f} = 0$ otherwise. In the second step, we set $q_{e,f} = 0$ with probability $1 - d$ for every pair of arcs e, f.

We also provide numerical results for instances whose cost matrices contain negative values. To generate those instances we first use the procedure as described above, and then multiply $q_{e,f}$ and $q_{f,e}$ by -1 with probability 0.5 for every pair of arcs $e, f \in A$. The numerical results with so generated data are presented in the last table.

Tables 2–6 read as follows. In the first three columns, we list the number of vertices n, the number of arcs m, and the density parameter d, respectively. In the fourth column, we present optimal values obtained by the branch-and-bound algorithm from [13] or Cplex. In the fifth to eighth column we include the lower bounds LBB_*, RBB, GLT and LP_{milp}, respectively. In the ninth to the last column, we list computational times required to compute LBB_*, RBB, LP_{milp}, respectively. Numbers in the brackets in the second to last column provide the number of iterations needed to compute the reformulation-based bound.

We round up all lower bounds. However, computational times (in seconds) are rounded to the nearest integer. Since computational times for computing the GLT bounds are small, we omit those results from the table. The computational time for computing the LBB_* consists of generating the linear system for $\mathcal{S}(Q)$ and solving the linear programming problem (8).

Table 2 presents computational results for GRID1 instances where $p = q = 12, \ldots, 15$. The results show that for dense instances the linearization-based bounds are at least 96% of the optimal value. The LBB_* is tight for one instance of size $n = 144$. We observe that the reformulation-based bounds are considerably weaker than the linearization-based

bounds, but require less computational effort. For the sparse instances with $d = 0.2$, the reformulation-based bounds and LP_{milp} are very weak, while the linearization-based bounds perform reasonably well. For dense instances, we observed that RBB/GLT is approximately 40%.

Table 3 shows the computational results for GRID3 instances where $p = q = 12, \ldots, 15$. For sparse instances the RBB, GLB and LP_{milp} bounds are trivial, while the LBB_* is not. For dense instances our linearization-based bound is about 90% of the optimal value.

Table 4 presents the computational results for PAR-K instances where $K = 5, \ldots, 8$. Here $n = (K-2)K + 2$. Our linearization-based bounds are equal to the optimal values for all PAR-K instances. The strength of the RBB decreases as the instance size grows. We do not present results for low density cost matrices for PAR-K instances as the optimal values for those instances are almost always zero.

Table 5 presents the computational results for TOUR instances where $n = 10, \ldots, 25$. We observe that the quality of our LBB_* decreases as the instance size grows. More precisely, the ratio LBB_*/OPT goes from 72% down to 45%. Finally, we observed that the bound $RBB = GLT = n + 1$ for all test instance, which gives a rather weak bound. Note that LP_{milp} is zero for all tested instances.

Table 6 reports bounds for instances whose cost matrices contain also negative elements. The first two rows correspond to GRID1 instances with negative cost elements. Here, it follows a similar observation as for the results in Table 2. Namely, the LBB_* provides the strongest lower bound, while other bounds are much weaker than the LBB_*. Bounds for GRID3 instances (resp. PAR-K instances) with negative costs are listed in rows three and four (resp. five and six) of Table 6. The results show that also for those instances the best linearization-based bound is superior to others. Thus the LBB_* retains its quality even for instances with negative interaction costs.

7 Conclusion

In this paper, we provide necessary and sufficient conditions for an instance of the QSPP on a directed acyclic graph to be linearizable, see Theorem 3.8. These conditions can be verified in $\mathcal{O}(nm^3)$ time, where n is the number of vertices and m is the number of arcs.

We present a family of lower bounds based on the linearization problem for the QSPP on DAGs. The strongest bound in this family is the optimal value of a linear programming problem (8). We also propose a variant of the reformulation-based bound for the QSPP, and prove that it can be computed in a finite number of steps for rational data matrices, see Propositions 5.1. Both bounding approaches can be exploited also for cost matrices with negative elements.

Our numerical results show that the strongest linearization-based bound is superior

to other linear bounds for the QSPP on DAGs. The interested reader can download our algorithm that solves the linearization problem and MATLAB codes that compute here presented lower bounds, from the following link:

`https://www.huhao.org`

To conclude, we show how to exploit the linearization problem for the quadratic shortest path problem in order to compute strong bounds for the QSPP. Despite the fact that the linearization problem is studied for many quadratic optimization problems, these are the first known results on the linearization problem based bounds. We expect that similar results can be obtained for other optimization problems, for example the quadratic assignment problem, the quadratic minimum spanning tree problem, and the quadratic traveling salesman problem. However, this is a subject of our future research.

n	m	d	OPT	LBB_*	RBB	GLT	LP_{milp}	LBB^s_*	$RBB^{s,it}$	LP^s_{milp}
144	264	0.2	113	109	7	1	1	3	3(12)	1
144	264	0.2	115	101	8	5	1	3	3(13)	1
144	264	0.8	851	851	653	499	434	3	4(17)	1
144	264	0.8	923	909	673	517	455	3	4(17)	1
169	312	0.2	159	122	2	0	1	6	4(12)	1
169	312	0.2	149	138	11	3	3	6	4(12)	1
169	312	0.8	1110	1103	807	631	555	5	7(19)	1
169	312	0.8	1150	1121	830	635	559	6	6(18)	1
196	364	0.2	188	153	13	3	0	11	7(14)	1
196	364	0.2	184	152	9	1	0	11	7(14)	1
196	364	0.8	1288	1255	902	708	631	12	9(19)	1
196	364	0.8	1311	1253	881	685	625	11	9(19)	1
225	420	0.2	237	179	22	6	1	23	10(15)	2
225	420	0.2	227	176	10	2	0	23	9(14)	2
225	420	0.8	1539	1481	1048	808	727	23	13(20)	2
225	420	0.8	1527	1477	1093	839	734	24	14(21)	2

Table 2: GRID1 INSTANCES: bounds, optimal values and time (s)

n	m	d	OPT	LBB_*	RBB	GLT	LP_{milp}	LBB^s_*	$RBB^{s,it}$	LP^s_{milp}
146	288	0.2	19	13	0	0	0	6	2(8)	1
146	288	0.2	12	9	0	0	0	7	2(8)	1
146	288	0.8	273	256	127	110	88	7	3(9)	1
146	288	0.8	277	252	128	114	90	7	3(11)	1
171	338	0.2	22	13	0	0	0	13	3(7)	1
171	338	0.2	15	11	0	0	0	13	3(7)	1
171	338	0.8	339	309	165	148	102	15	6(15)	1
171	338	0.8	323	295	145	131	97	16	6(14)	1
198	392	0.2	24	17	0	0	0	28	4(8)	1
198	392	0.2	25	13	0	0	0	28	4(7)	2
198	392	0.8	370	330	155	137	105	33	8(14)	1
198	392	0.8	367	339	167	140	104	33	9(15)	2
227	450	0.2	18	10	0	0	0	54	5(6)	2
227	450	0.2	24	17	0	0	0	50	5(7)	2
227	450	0.8	400	374	181	160	124	59	14(17)	2
227	450	0.8	426	386	173	150	121	55	14(17)	2

Table 3: GRID3 INSTANCES: bounds, optimal values and time (s)

n	m	d	OPT	LBB_*	RBB	GLT	LP_{milp}	LBB^s_*	$RBB^{s,it}$	LP^s_{milp}
17	60	0.8	20	20	17	9	3	0	0(7)	0
17	60	0.8	9	9	9	9	4	0	0(7)	0
17	60	0.8	13	13	13	7	4	0	0(7)	0
17	60	0.8	19	19	18	11	4	0	0(8)	0
26	120	0.8	22	22	16	10	2	0	0(8)	0
26	120	0.8	26	26	20	10	3	0	0(9)	0
26	120	0.8	19	19	16	8	2	0	0(9)	0
26	120	0.8	23	23	16	12	1	0	0(9)	0
37	210	0.8	33	33	20	9	0	3	1(10)	0
37	210	0.8	34	34	16	7	1	3	1(10)	0
37	210	0.8	39	39	26	13	1	3	1(9)	0
37	210	0.8	31	31	21	6	0	2	1(10)	0
50	336	0.8	51	51	23	8	0	249	3(11)	1
50	336	0.8	46	46	22	5	0	246	2(10)	1
50	336	0.8	40	40	21	8	1	241	2(10)	1
50	336	0.8	55	55	22	7	0	218	3(11)	1

Table 4: PAR-K INSTANCES: bounds, optimal values and time (s)

n	m	d	OPT	LBB_*	RBB	GLT	LP_{milp}	LBB^s_*	$RBB^{s,it}$	LP^s_{milp}
10	45	1.0	29	21	11	11	0	0	0(2)	0
11	55	1.0	30	20	12	12	0	0	0(2)	0
12	66	1.0	33	25	13	13	0	0	0(2)	0
13	78	1.0	38	24	14	14	0	0	0(2)	0
14	91	1.0	45	29	15	15	0	0	0(2)	0
15	105	1.0	50	28	16	16	0	0	0(2)	0
16	120	1.0	55	33	17	17	0	0	0(2)	0
17	136	1.0	58	32	18	18	0	1	0(2)	0
18	153	1.0	63	37	19	19	0	1	0(2)	0
19	171	1.0	70	36	20	20	0	2	0(2)	0
20	190	1.0	75	41	21	21	0	3	0(2)	0
21	210	1.0	82	40	22	22	0	6	0(2)	0
22	231	1.0	91	45	23	23	0	11	0(2)	1
23	253	1.0	94	44	24	24	0	16	0(2)	1
24	276	1.0	99	49	25	25	0	38	0(2)	1
25	300	1.0	106	48	26	26	0	45	0(2)	1

Table 5: TOUR INSTANCES: bounds, optimal values and time (s)

n	m	d	OPT	LBB_*	RBB	GLT	LP_{milp}	LBB^s_*	$RBB^{s,it}$	LP^s_{milp}
225	420	0.2	-305	-353	-718	-930	-1030	23	12(16)	2
225	420	0.8	-545	-679	-1358	-1815	-2013	25	18(24)	3
227	450	0.2	-167	-191	-403	-432	-456	55	11(13)	2
227	450	0.8	-351	-399	-721	-796	-883	57	23(27)	2
50	336	0.8	-132	-132	-162	-195	-223	232	3(12)	1
50	336	0.8	-130	-130	-170	-203	-227	237	3(13)	1

Table 6: instances with negative weights: bounds, optimal values and time (s)

References

[1] W. Adams, L. Waddell. Linear programming insights into solvable cases of the quadratic assignment problem, Discrete Optimization 14, 46–60, 2014.

[2] C. Buchheim, E. Traversi. Quadratic 0-1 optimization using separable underestimators, Technical Report, Optimization Online, 2015.

[3] P. Carraresi, F. Malucelli. A new lower bound for the quadratic assignment problem, Operations Research 40(1), S22-S27, 1992.

[4] A. Ćustić, A. P. Punnen. A characterization of linearizable instances of the quadratic minimum spanning tree problem, arXiv:1510.02197, 2015.

[5] A. Ćustić, V. Sokol, A. P. Punnen, B. Bhattacharya. The bilinear assignment problem: Complexity and polynomially solvable special cases, Mathematical Programming, 166 (1-2), 185–205, 2017.

[6] E. Çela, V. G. Deineko, G. J. Woeginger. Linearizable special cases of the QAP, Journal of Combinatorial Optimization 31(3), 1269-1279, 2016.

[7] Cplex, ILOG. 7.0 Reference Manual. ILOG CPLEX Division, Incline Village, NV. 2000.

[8] E. W. Dijkstra. A note on two problems in connexion with graphs, Numerische mathematik 1(1), 269–271, 1959.

[9] R. W. Floyd. Algorithm 97: Shortest path, Communications of the ACM 5(6), 345, 1962. Computer Science, Springer, 621–632, 2009.

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

[11] F. Glover. Improved linear integer programming formulations of nonlinear integer problems. Management Science 22(4), 455–460, 1975.

[12] H. Hu, R. Sotirov. Special cases of the quadratic shortest path problem, Journal of Combinatorial Optimization (accepted), 2017.

[13] H. Hu, R. Sotirov. On solving the quadratic shortest path problem, arXiv:1708.06580 [math.OC].

[14] S. N. Kabadi, A. P. Punnen. An $O(n^4)$ algorithm for the QAP linearization problem, Mathematics of Operations Research 36, 754–761, 2011.

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

[16] K. Murakami, H. S. Kim. Comparative study on restoration schemes of survivable ATM networks, INFOCOM'97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Driving the Information Revolution., Proceedings IEEE, 1, 345–352, 1997.

[17] A. P. Punnen. Combinatorial optimization with multiplicative objective function, International Journal of Operations and Quantitative Management 7, 205–209, 2001.

[18] A. P. Punnen, B. D. Woods. A characterization of linearizable instances of the quadratic traveling salesman problem, arXiv:1708.07217 [cs.DM].

[19] A. P. Punnen, S. N. Kabadi, A linear time algorithm for the Koopmans–Beckman QAP linearization and related problems, Discrete Optimization 10, 200–209, 2013.

[20] B. Rostami, A. Chassein, M. Hopf, D. Frey, C. Buchheim, F. Malucelli, M. Goerigk. The quadratic shortest path problem: complexity, approximability, and solution methods, Optimization Online, 2016.

[21] B. Rostami, F. Malucelli, D. Frey, C. Buchheim. On the quadratic shortest path problem. In: E. Bampis (ed.) Experimental Algorithms, Lecture Notes in Computer Science, Springer International Publishing, 9125, 379–390, 2015.

[22] S. Sen, R. Pillai, S. Joshi, A. K. Rathi. A mean-variance model for route guidance in advanced traveler information systems, Transportation Science 35(1), 37–49, 2001.

[23] R. A. Sivakumar, R. Batta. The variance-constrained shortest path problem, Transportation Science 28(4), 309–316, 1994.

[24] S. Warshall. A theorem on Boolean matrices, Journal of the ACM 9(1), 11-12, 1962.