Mallach, Sven

Article — Published Version

Inductive linearization for binary quadratic programs with linear constraints: a computational study

4OR

Provided in Cooperation with:
Springer Nature

Suggested Citation: Mallach, Sven (2023) : Inductive linearization for binary quadratic programs with linear constraints: a computational study, 4OR, ISSN 1614-2411, Springer, Berlin, Heidelberg, Vol. 22, Iss. 1, pp. 47-87, https://doi.org/10.1007/s10288-023-00537-5

This Version is available at:
https://hdl.handle.net/10419/312515

4OR (2024) 22:47–87
https://doi.org/10.1007/s10288-023-00537-5

RESEARCH PAPER

Inductive linearization for binary quadratic programs with linear constraints: a computational study

Sven Mallach[1]

Received: 18 March 2022 / Revised: 21 October 2022 / Accepted: 7 February 2023 /
Published online: 23 March 2023

Abstract
The computational utility of inductive linearizations for binary quadratic programs when combined with a mixed-integer programming solver is investigated for several combinatorial optimization problems and established benchmark instances.

Keywords Non-linear programming · Binary quadratic programming · Mixed-integer programming · Linearization

Mathematics Subject Classification 68R01 · 90C05 · 90C09 · 90C10 · 90C11 · 90C20 · 90C30

1 Introduction

Given a binary quadratic program (BQP) comprising linear (and possibly quadratic) constraints, the inductive linearization technique (Mallach 2021) may serve as a computationally attractive compromise between the well-known "standard" linearization as of Glover and Woolsey (1974), and a complete application of the Reformulation Linearization Technique [RLT, see e.g. Adams and Sherali (1999, 1986)]. In several relevant cases, inductive linearizations are more constraint-side compact than the "standard" linearization and provide a continuous relaxation that is at least as tight. Prominent combinatorial optimization problems where this applies are for instance the Quadratic Assignment, the Quadratic Matching, and the Quadratic Traveling Salesman Problem.

Given this theoretical basis, the central contribution of this paper is a systematic computational study in order to address a number of research questions: Does the mentioned symbiosis of compactness and continuous relaxation strength translate

✉ Sven Mallach
sven.mallach@cs.uni-bonn.de

[1] High Performance Computing and Analytics Lab, University of Bonn, Friedrich-Hirzebruch-Allee 8, 53115 Bonn, Germany

into a faster solution time of the corresponding mixed-integer programs? How do the obtained relaxation bounds compare with the "standard" linearization and the RLT at a broader scope? For which problem structures is the inductive linearization technique (not) well-suited and why? Can an inductive linearization be obtained quickly in practice, and if so, how?

To this end, we investigate the performance of the inductive as well as the "standard" linearization in combination with (and in comparison to) a professional mixed-integer programming solver on various BQPs with linear constraints. More precisely, we look at the Quadratic Assignment Problem, the Quadratic Knapsack Problem, the Quadratic Matching Problem, the Quadratic Shortest Path Problem, and on further instances of the well-established QPLIB[1] and MINLPLib.[2] Besides that, an algorithmic frame to derive inductive linearizations in practice, and thus complementing the mixed-integer programming (MIP) approach from Mallach (2021), is presented.

The outline of this paper is as follows: In the beginning of Sect. 2, we briefly review the main concepts of the inductive linearization technique and then strongly emphasize on its practical application. In Sect. 3, we present the aforementioned applications along with the respective problem formulations as well as the benchmark instances used for the computational study, and a detailed discussion of the respective results. Finally, a conclusion is given in Sect. 4.

2 Inductive linearization

The inductive linearization technique addresses optimization problems adhering to or comprising the structure

$$\begin{aligned} &\min\ x^\mathsf{T} C_0 x + c^\mathsf{T} x \\ &\text{s.t. } x^\mathsf{T} C_k x + g_k^\mathsf{T} x \leq \beta_k && k = 1, \ldots, m_Q \\ &\quad Ax \leqq b \\ &\quad x \in \{0,1\}^n, \end{aligned}$$

where, for $n, m_L \in \mathbb{N}\backslash\{0\}$ and $m_Q \in \mathbb{N}$, the matrices $A \in \mathbb{R}^{m_L \times n}$ as well as $C_k \in \mathbb{R}^{n \times n}$ for $k \in \{0, \ldots, m_Q\}$, the vectors $g_k \in \mathbb{R}^n$ for $k \in \{1, \ldots, m_Q\}$ and $c \in \mathbb{R}^n$, and finally the scalars b as well as $\beta_k \in \mathbb{R}$ for $k \in \{1, \ldots, m_Q\}$ are given as input data. Throughout this paper, we will denote by $N := \{1, \ldots, n\}$ the index set of the binary variables $x \in \{0,1\}^n$. Moreover, the set of products P truly present in the problem is determined by the the objective function and the quadratic restrictions as follows:

$$P := \{(i,j) \subseteq N \times N \mid \exists\, k \in \{0, \ldots, m_Q\} : C_{k_{ij}} \neq 0\}$$

Naturally, we assume that P is non-empty and, since we have for binary solutions that $x_i x_j = x_j x_i$ for $i, j \in N$, $i \neq j$, and $x_i = x_i^2$ for all $i \in N$, we assume further that the matrices C_k, $k \in \{0, \ldots, m_Q\}$, are strictly upper triangular so that we have $i < j$ for each $(i,j) \in P$.

[1] https://qplib.zib.de.

[2] https://www.minlplib.org.

While quadratic constraints may or may not be present, some linear constraints on the binary variables are actually necessary to apply the inductive linearization technique. We assume w.l.o.g. that these are given as equations and less-or-equal inequalities, hence denoted $Ax \leqq b$. More precisely, the requirement is that each binary variable x_i, $i \in N$, being a factor of a product in P (to be linearized with the technique proposed), appears in at least one of these constraints (with a non-zero coefficient). Clearly, this can be assumed (or established) for a binary problem without loss of generality. Indeed, if there is a factor x_i, $i \in N$, that is entirely free w.r.t. the linear constraints then in principle any linear equation or less-or-equal inequality may be employed that is valid for its feasible set (even, though rather not so desirable, $x_i \leq 1$). Moreover, if some factor is left free w.r.t. the linear constraints, then this does not affect a successful inductive linearization of other products that do have linearly constrained factors. For simplicity, we thus assume from now on that all factors appearing in P are linearly constrained.

The inductive linearization technique is a generalization of a principle proposed by Liberti (2007) for the special case of equations with right hand side and left hand side coefficients equal to one, and of its later revision (cf. Mallach (2018)). In his original article, Liberti coined the name "compact linearization" because it typically adds fewer constraints to the mentioned problems than the "standard" linearization as of Glover and Woolsey (1974). For the problem under consideration, a general form of the latter reads:

$$
\begin{aligned}
\min\ & d^\mathsf{T} y + c^\mathsf{T} x \\
\text{s.t. } & h_k^\mathsf{T} y + g_k^\mathsf{T} x \leq \beta_k \qquad k = 1, \ldots, m_Q \\
& Ax \leqq b && (1)\\
& y_{ij} - x_i \leq 0 \qquad (i,j) \in P && (2)\\
& y_{ij} - x_j \leq 0 \qquad (i,j) \in P && (3)\\
& x_i + x_j - y_{ij} \leq 1 \qquad (i,j) \in P && (4)\\
& y \geq 0 \\
& x \in \{0,1\}^n
\end{aligned}
$$

With $m := |P|$, we here use $y \in \mathbb{R}^m$ to denote the linearized products, $d \in \mathbb{R}^m$ to denote their corresponding objective coefficients, and $h_k \in \mathbb{R}^m$, $k = 1, \ldots, m_Q$, to express their coefficients in the original quadratic constraints. Thereby, we use the subscript notation y_{ij} for $i < j$ and analogously define $d_{ij} = (C_0)_{ij}$ and $h_{kij} = (C_k)_{ij}$. Of course, if $d_{ij} < 0$ (4) can be omitted, and if $d_{ij} > 0$ (2) and (3) can be omitted for $(i,j) \in P$.

As we will see, in many cases, inductive linearizations are constraint-side compact as well. However, this cannot be guaranteed for any kind of BQP with linear constraints. Moreover, depending on how the method is applied, more than $|P|$ linearization variables may be induced (although this can, in principle, always be circumvented as described in Sect. 2.3). Therefore, and to have a clear distinction from other linearizations being called "compact", as well as to emphasize that the proposed method aims at "inducing" the products associated to the set P by multiplying original constraints

with original variables, the technique is referred to as "inductive linearization" since its generalization first presented in Mallach (2021).

2.1 Mathematical derivation of inductive linearizations

Given a problem as introduced at the beginning of this section, suppose that we identify a working (sub-)set of the linear constraints $Ax \leqq b$ to actually induce the linearization with. Let us denote the index set of the selected equations and inequalities with K_E and K_I, respectively. That is, we consider the constraints

$$\sum_{i \in I_k} a_k^i x_i = b_k \quad \text{for all } k \in K_E \tag{5}$$

$$\sum_{i \in I_k} a_k^i x_i \leq b_k \quad \text{for all } k \in K_I \tag{6}$$

where $I_k := \{i \in N \mid a_k^i \neq 0\}$ denotes the respective support index set for each $k \in K_E$ or $k \in K_I$.

As already mentioned, we require w.l.o.g. a choice of $K := K_E \cup K_I$ such that there exist indices $k, \ell \in K$ with $i \in I_k$ and $j \in I_\ell$ for all $(i, j) \in P$. To refer to the respective constraints, we will use the notation $K(i) := \{k \in K : i \in I_k\}$, as well as $K_E(i)$ and $K_I(i)$ analogously defined if more preciseness is in order. Moreover, although we do not require this for the original problem, let us temporarily assume in addition that $b_k > 0$ for all $k \in K$ and $a_k^i > 0$ for all $i \in I_k, k \in K$. We will elaborate in Sect. 2.4 on how to handle constraints not fulfilling these prerequisites.

The first step of the inductive linearization approach now associates to each equation $k \in K_E$ another index set $M_k^E \subseteq N$ that is supposed to specify original variables used as multipliers. To each inequality $k \in K_I$, two such index sets $M_k^+, M_k^- \subseteq N$ are associated. The corresponding interpretation is as follows: If $j \in M_k^E$ ($j \in M_k^+$) the equation $k \in K_E$ (inequality $k \in K_I$) is multiplied by x_j, and if $j \in M_k^-$, the inequality $k \in K_I$ is multiplied by $(1 - x_j)$.

This leads to the following set of first-level RLT constraints:

$$\sum_{i \in I_k} a_k^i x_i x_j = b_k x_j \qquad \text{for all } j \in M_k^E,\ k \in K_E \tag{7}$$

$$\sum_{i \in I_k} a_k^i x_i x_j \leq b_k x_j \qquad \text{for all } j \in M_k^+,\ k \in K_I \tag{8}$$

$$\sum_{i \in I_k} a_k^i x_i (1 - x_j) \leq b_k (1 - x_j) \qquad \text{for all } j \in M_k^-,\ k \in K_I \tag{9}$$

Let $M_k := M_k^E$ if $k \in K_E$, and $M_k := M_k^+ \cup M_k^-$ if $k \in K_I$. Then

$$Q := \{(i, j) \mid i \leq j \text{ and } \exists k \in K : i \in I_k \text{ and } j \in M_k, \text{ or } j \in I_k \text{ and } i \in M_k\}$$

is the index set of the products induced by (7)–(9).

For ease of reference, we also define

$$M := \bigcup_{k \in K} M_k$$

which is to be regarded as a multiset of multiplier indices.

Remark 1 In the general approach described above, the induced set Q may contain tuples that correspond to squares. It is a simple and worthwhile optimization to replace these by their linear counterparts before actually deriving the respective linearization constraint (see also Sect. 2.2 and Mallach (2021)).

If we now rewrite (7)–(9) by substituting for each $(i, j) \in Q$ the product $x_i x_j$ by a continuous linearization variable y_{ij} that has explicit lower and upper bounds, i.e., $0 \leq y_{ij} \leq 1$, we obtain the *linearization constraints*:

$$\sum_{i \in I_k, (i,j) \in Q} a_k^i y_{ij} + \sum_{i \in I_k, (j,i) \in Q} a_k^i y_{ji} = b_k x_j \qquad \text{for all } j \in M_k^E,\ k \in K_E \tag{10}$$

$$\sum_{i \in I_k, (i,j) \in Q} a_k^i y_{ij} + \sum_{i \in I_k, (j,i) \in Q} a_k^i y_{ji} \leq b_k x_j \qquad \text{for all } j \in M_k^+,\ k \in K_I \tag{11}$$

$$\sum_{i \in I_k, (i,j) \in Q} a_k^i (x_i - y_{ij}) + \sum_{i \in I_k, (j,i) \in Q} a_k^i (x_i - y_{ji}) \leq b_k (1 - x_j) \qquad \text{for all } j \in M_k^-,\ k \in K_I \tag{12}$$

Now, as is expressed by the following theorem, for all the induced $(i, j) \in Q$ and binary x_i, x_j, one has $y_{ij} = x_i x_j$ if the following three consistency conditions are met:

Condition 1 *There is a $k \in K(i)$ such that $j \in M_k^E$ or $j \in M_k^+$, respectively.*

Condition 2 *There is a $k \in K(j)$ such that $i \in M_k^E$ or $i \in M_k^+$, respectively.*

Condition 3 *There is a $k \in K(i)$ such that $j \in M_k^E$ or $j \in M_k^-$, respectively, **or** a $k \in K(j)$ such that $i \in M_k^E$ or $i \in M_k^-$, respectively.*

Theorem 4 [Mallach (2021)] *For any integer solution $x \in \{0, 1\}^n$, the linearization constraints (10)–(12) imply $y_{ij} = x_i x_j$ for all $(i, j) \in Q$ if and only if Conditions 1–3 are satisfied.*

So altogether, if we choose M consistently in terms of the three conditions and such that Q contains P, we obtain a linearization for our original problem. In fact, at the potential expense of losing some continuous relaxation strength, it is always possible to have $Q = P$ as is described in Sect. 2.3.

2.2 Linear relaxation strength of inductive linearizations

The case where the constraint set K employed satisfies $b_k = 1$ for all $k \in K$, and $a_k^i = 1$ for all $i \in I_k$ is an example of practical relevance where the linear programming relaxation obtained from an inductive linearization is provably at least as tight as the one obtained from the "standard" linearization. The same is true for equations with a right hand side of two and zero–one left hand side coefficients if these equations are multiplied by all variables on their left hand sides (i.e., $M_k = I_k$) and squares are ruled out.

More generally, the corresponding proofs in Mallach (2021) make apparent that the tightness of the relaxations of inductively linearized BQPs relates (besides other criteria) to the ratio between the right hand side and the left hand side coefficients. It also provides an example case where an inductive linearization provably has a *strictly* stronger continuous relaxation than the "standard" linearization.

In our computational study, we will identify experimentally further cases where this relation is achieved and further investigate how far the bounds deviate in practice.

2.3 Practical derivation of inductive linearizations and their compactness

Given a problem formulation on sheet, a multiset M that induces a set $Q \supseteq P$ and establishes a consistent linearization can often be derived by inspection once the necessary implications imposed by Conditions 1–3 are understood (see also Sect. 2.4). For instance if the factor pairs in P or the constraints at hand have a complicated structure, it may nevertheless be non-trivial to find a consistent combination of constraints and multipliers, especially if the outcome is supposed to be compact or to meet other objectives. Moreover, an automated derivation is desirable for larger problem instances and allows for a linearization framework to be coupled with a mixed-integer programming solver.

Concerning the computation of an inductive linearization, it has been shown in Mallach (2021) that the associated optimization problem (allowing e.g. to derive a linearization that is as compact as possible in terms of additional variables and constraints) is NP-hard. This result is however rather of theoretic prominence as it considers the general case of any possible input program whereas common inputs are usually structured and the associated "covering problems" turn out to be often simple to solve. Indeed they can frequently be solved even exactly, using e.g. a mixed-integer program or combinatorial polynomial-time algorithms for more specifically structured BQPs. Typically, the number of candidate constraints to induce a certain product, respectively to satisfy one of the three conditions, is anyway rather small. Likewise, for many applications, the complexity can also be reduced by carefully preselecting the set K of original constraints considered for inductions.

It is further apparent from the hardness proof as well as the mixed-integer program in Mallach (2021) that the compactness of the resulting linearization is influenced by the support of the employed constraint set K and its relation to the factor pairs given by P. In this context, it is also worth to mention that the presolve routines of a MIP solver may well eliminate some of the variables and constraints imposed by an inductive

linearization that is not "most compact". Moreover, a few additional constraints may sometimes improve the relaxation strength, so compactness need not necessarily be an ultimate goal.

Notwithstanding, it is also possible to eliminate (all) variables in $Q \setminus P$ if (all of) these have been generated from inequalities (possibly obtained from equations in a preprocessing step). As is clear from inequalities (11) and (12), removing summands on their left hand sides will neither harm their validity nor their necessary implications on the respective remaining linearization variables. In general, however, the feasible region of the continuous relaxation may of course be enlarged by this procedure (referred to as weakened inductive linearization in Sect. 3).

As a particularly practical approach to derive inductive linearizations, we present (the heuristic) Algorithm 1 which is an extension of a special-case combinatorial algorithm from Mallach (2018) to the general case. It also serves to derive the inductive linearizations during the computational study in Sect. 3.

Algorithm 1 Main routine of the heuristic to construct an inductive linearization.

function ConstructSets(Sets P, K, w)
  for all $k \in K_E$ **do**
    $M_k^E \leftarrow \emptyset$
  for all $k \in K_I$ **do**
    $M_k^+ \leftarrow \emptyset$; $M_k^- \leftarrow \emptyset$
  $Q \leftarrow P$
  $Q_{\text{new}} \leftarrow P$
  while $\emptyset \neq Q_{\text{add}} \leftarrow$ Append(Q, Q_{new}, K, M) **do**
    $Q \leftarrow Q \cup Q_{\text{add}}$
    $Q_{\text{new}} \leftarrow Q_{\text{add}}$
return Q and M

Algorithm 1 consists of two simple components. First, the major routine ConstructSets which is to be supplied with the input sets P and K as well as weights w_{kj}^E, respectively w_{kj}^+ and w_{kj}^-, indicating a "cost" of creating a linearization constraint by multiplying constraint $k \in K$ with original variable x_j, respectively $1 - x_j$, just as in the mixed-integer program in Mallach (2021). The major routine then invokes the subroutine Append (listed as Algorithm 2) which extends a partial inductive linearization represented by Q and M. It does so by inducing additional linearization constraints in order to satisfy Conditions 1–3 for the variables Q_{new} added in the previous iteration (initially, $Q_{new} = P$), and by appending the variables Q_{add} newly induced when creating these linearization constraints. Algorithm 1 terminates if a steady state is reached, i.e., if no further constraint-factor multiplications are necessary to satisfy Conditions 1–3 for the current Q, which is then, together with the constraint multiplier set M, returned.

Algorithm 2 Subroutine used by the heuristic to construct an inductive linearization.

function APPEND(Sets Q, Q_{new}, K, M, w)
  $Q_{\text{add}} \leftarrow \emptyset$
  for all $(i, j) \in Q_{\text{new}}$ **do**
    if $j \notin M_k^E \; \forall\, k \in K_E(i) \wedge j \notin M_k^+ \; \forall\, k \in K_I(i)$ **then** ▷ If Condition 1 is not satisfied
      $\bar{k} \leftarrow nil$; $\bar{w} \leftarrow 0$
      if $K_E(i) \neq \emptyset$ **then**
        $\bar{k} \leftarrow \arg\min_{k \in K_E(i)} w_{kj}^E$; $\bar{w} \leftarrow w_{\bar{k}j}^E$
      if $K_I(i) \neq \emptyset \wedge (\bar{k} = nil \vee \min_{k \in K_I(i)} w_{kj}^+ < \bar{w})$ **then**
        $\bar{k} \leftarrow \arg\min_{k \in K_I(i)} w_{kj}^+$
        $M_{\bar{k}}^+ \leftarrow M_{\bar{k}}^+ \cup \{j\}$
      else
        $M_{\bar{k}}^E \leftarrow M_{\bar{k}}^E \cup \{j\}$
      for all $a \in I_{\bar{k}}$ **do**
        if $a \leq j \wedge (a, j) \notin Q \cup Q_{\text{add}}$ **then** $Q_{\text{add}} \leftarrow Q_{\text{add}} \cup \{(a, j)\}$
        else if $a > j \wedge (j, a) \notin Q \cup Q_{\text{add}}$ **then** $Q_{\text{add}} \leftarrow Q_{\text{add}} \cup \{(j, a)\}$
    if $i \notin M_\ell^E \; \forall\, \ell \in K_E(j) \wedge i \notin M_\ell^+ \; \forall\, \ell \in K_I(j)$ **then** ▷ If Condition 2 is not satisfied
      $\bar{\ell} \leftarrow nil$; $\bar{w} \leftarrow 0$
      if $K_E(j) \neq \emptyset$ **then**
        $\bar{\ell} \leftarrow \arg\min_{\ell \in K_E(j)} w_{\ell i}^E$; $\bar{w} \leftarrow w_{\bar{\ell} i}^E$
      if $K_I(j) \neq \emptyset \wedge (\bar{\ell} = nil \vee \min_{\ell \in K_I(j)} w_{\ell i}^+ < \bar{w})$ **then**
        $\bar{\ell} \leftarrow \arg\min_{\ell \in K_I(j)} w_{\ell i}^+$
        $M_{\bar{\ell}}^+ \leftarrow M_{\bar{\ell}}^+ \cup \{i\}$
      else
        $M_{\bar{\ell}}^E \leftarrow M_{\bar{\ell}}^E \cup \{i\}$
      for all $a \in I_{\bar{\ell}}$ **do**
        if $a \leq i \wedge (a, i) \notin Q \cup Q_{\text{add}}$ **then** $Q_{\text{add}} \leftarrow Q_{\text{add}} \cup \{(a, i)\}$
        else if $a > i \wedge (i, a) \notin Q \cup Q_{\text{add}}$ **then** $Q_{\text{add}} \leftarrow Q_{\text{add}} \cup \{(i, a)\}$
    if $j \notin M_k \; \forall\, k \in K_E(i) \wedge j \notin M_k^- \; \forall\, k \in K_I(i) \wedge i \notin M_\ell \; \forall\, \ell \in K_E(j) \wedge i \notin M_\ell^- \; \forall\, \ell \in K_I(j)$ **then**
      ▷ If Condition 3 is not satisfied
      $\bar{k} \leftarrow nil$; $\bar{\ell} \leftarrow nil$; $\bar{w}_1 \leftarrow 0$; $\bar{w}_2 \leftarrow 0$
      if $K_E(i) \neq \emptyset$ **then**
        $\bar{k} \leftarrow \arg\min_{k \in K_E(i)} w_{kj}^E$; $\bar{w}_1 \leftarrow w_{\bar{k}j}^E$
      if $K_I(i) \neq \emptyset \wedge (\bar{k} = nil \vee \min_{k \in K_I(i)} w_{kj}^- < \bar{w}_1)$ **then**
        $\bar{k} \leftarrow \arg\min_{k \in K_I(i)} w_{kj}^-$; $\bar{w}_1 \leftarrow w_{\bar{k}j}^-$
      if $K_E(j) \neq \emptyset$ **then**
        $\bar{\ell} \leftarrow \arg\min_{\ell \in K_E(j)} w_{\ell i}^E$; $\bar{w}_2 \leftarrow w_{\bar{\ell} i}^E$
      if $K_I(j) \neq \emptyset \wedge (\bar{\ell} = nil \vee \min_{\ell \in K_I(j)} w_{\ell i}^- < \bar{w}_2)$ **then**
        $\bar{\ell} \leftarrow \arg\min_{\ell \in K_I(j)} w_{\ell i}^-$; $\bar{w}_2 \leftarrow w_{\bar{\ell} i}^-$
      if $\bar{w}_1 \leq \bar{w}_2$ **then**
        if $\bar{k} \in K_E(i)$ **then**
          $M_{\bar{k}}^E \leftarrow M_{\bar{k}}^E \cup \{j\}$
        else
          $M_{\bar{k}}^- \leftarrow M_{\bar{k}}^- \cup \{j\}$
        for all $a \in I_{\bar{k}}$ **do**
          if $a \leq j \wedge (a, j) \notin Q \cup Q_{\text{add}}$ **then** $Q_{\text{add}} \leftarrow Q_{\text{add}} \cup \{(a, j)\}$
          else if $a > j \wedge (j, a) \notin Q \cup Q_{\text{add}}$ **then** $Q_{\text{add}} \leftarrow Q_{\text{add}} \cup \{(j, a)\}$

else
if $\bar{\ell} \in K_E(j)$ **then**
$M^E_{\bar{\ell}} \leftarrow M^E_{\bar{\ell}} \cup \{i\}$
else
$M^-_{\bar{\ell}} \leftarrow M^-_{\bar{\ell}} \cup \{i\}$
for all $a \in I_{\bar{\ell}}$ **do**
if $a \leq i \wedge (a,i) \notin Q \cup Q_{\text{add}}$ **then** $Q_{\text{add}} \leftarrow Q_{\text{add}} \cup \{(a,i)\}$
else if $a > i \wedge (i,a) \notin Q \cup Q_{\text{add}}$ **then** $Q_{\text{add}} \leftarrow Q_{\text{add}} \cup \{(i,a)\}$
return Q_{add}

2.4 Normalization, inductive linearizations with general linear constraint sets

We shall now discuss how to deal with the case that some (or all) of the original constraints $k \in K$ to be employed, i.e., (5) and (6), do not satisfy $b_k > 0$ and $a^i_k > 0$ for all $i \in I_k$. To this end, suppose that

$$\sum_{i \in I_k} a^i_k x_i \leqq b_k$$

is an equation or $\leq$-inequality in K, and let $I^-_k \subseteq I_k$ be the set of variable indices such that $a^i_k < 0$ for each $i \in I^-_k$. For ease of notation, define also $I^+_k = I_k \backslash I^-_k$.

The *explicit* approach (see also, e.g. Hammer et al. (1984)) to deal with such constraints is to define a new *complement* variable $\bar{x}_i$ for each $i \in I^-_k$, $k \in K$, along with the corresponding equation:

$$x_i + \bar{x}_i = 1 \tag{13}$$

Apparently, the equations (13) have only positive coefficients on the left hand side and a positive right hand side. Moreover, we may now replace any of the original constraints with

$$\begin{aligned} &\sum_{i \in I^+_k} a^i_k x_i + \sum_{i \in I^-_k} a^i_k (1 - \bar{x}_i) \leqq b_k \\ \Leftrightarrow &\sum_{i \in I^+_k} a^i_k x_i + \sum_{i \in I^-_k} -a^i_k \bar{x}_i \leqq b_k + \sum_{i \in I^-_k} -a^i_k \end{aligned}$$

where the term $-\sum_{i \in I^-_k} a^i_k$ on the left and on the right hand side is non-negative as well.

Carrying out this procedure for every equation or inequality with negative coefficients on the left hand side clearly gives a normalized system with only non-negative coefficients on the left. Now if any of the resulting right hand sides is negative, the system is obviously infeasible. Furthermore, if any of them is zero then the values of all the variables on the respective left hand side can be fixed, and these variables can thus be removed from the formulation. We conclude that the prerequisites $b_k > 0$ for

all $k \in K$ and $a_k^i > 0$ for all $i \in I_k$, $k \in K$, can therefore be satisfied without loss of generality.

From a computational point of view, the explicit approach however has some drawbacks. The first is clearly that it may add up to n variables and equations while it is not clear a priori which of the resulting normalized constraints are eventually at all employed for multiplications. Moreover, the additional equations (13) are rather undesirable candidates for multiplications as the resulting linearization constraints provide a linkage of linearization and original variables that is similarly poor as in case of the "standard" linearization.

A more economical strategy is to keep the original constraints as they are, and to consider them for multiplication with each x_j (and $(1 - x_j)$ for case (9)) and each $\bar{x}_j$ (and $(1 - \bar{x}_j)$ for case (9)) without ever truly introducing the complement variables. Instead, the idea of this *implicit* approach is to choose, for each $i \in I_k$, the "right" of the four possible combinations $x_i x_j$, $\bar{x}_i x_j$, $x_i \bar{x}_j$, and $\bar{x}_i \bar{x}_j$ to be induced, such that the respective linearization constraint imposes the necessary implications on their value.

More precisely, as can be verified from the proof of Theorem 4 in Mallach (2021), the inequalities (8) (respectively their linearized counterparts (11)) have the effect of enforcing all the products (respectively, linearization variables) on the left hand side to be zero if the multiplier x_j is zero. Similarly, the constraints (9) (respectively, (12)) enforce any $x_i x_j$ (y_{ij}) on the left hand side to coincide with x_i if x_j is one. The equations (7) respectively (10) directly impose both relationships at once. These implications are exactly what is established by Conditions 1–3 if the coefficients and right hand sides of the original constraints are non-negative respectively positive.

To achieve the same in the general case, one has to replace (8) by

$$\begin{aligned} &\sum_{i \in I_k^+} a_k^i x_i x_j + \sum_{i \in I_k^-} a_k^i (1 - \bar{x}_i) x_j \leq b_k x_j \\ \Leftrightarrow &\sum_{i \in I_k^+} a_k^i x_i x_j + \sum_{i \in I_k^-} -a_k^i \bar{x}_i x_j \leq \left(b_k + \sum_{i \in I_k^-} -a_k^i \right) x_j \end{aligned}$$

respectively by

$$\begin{aligned} &\sum_{i \in I_k^+} a_k^i x_i \bar{x}_j + \sum_{i \in I_k^-} a_k^i (1 - \bar{x}_i) \bar{x}_j \leq b_k \bar{x}_j \\ \Leftrightarrow &\sum_{i \in I_k^+} a_k^i x_i \bar{x}_j + \sum_{i \in I_k^-} -a_k^i \bar{x}_i \bar{x}_j \leq \left(b_k + \sum_{i \in I_k^-} -a_k^i \right) - \left(b_k + \sum_{i \in I_k^-} -a_k^i \right) x_j. \end{aligned}$$

The Eq. (7) are to be replaced analogously. It is easy to see that, as desired, the linearization variables to be substituted for the products are forced to zero whenever the multiplier x_j respectively $\bar{x}_j$ is zero, and in particular that this effect is preserved when re-substituting the conceptual $\bar{x}_j$ by $(1 - x_j)$ in *linear* terms.

Further, the inequalities (9) need to be replaced by

$$\sum_{i\in I_k^+} a_k^i x_i(1-x_j) + \sum_{i\in I_k^-} a_k^i(1-\bar{x}_i)(1-x_j) \;\leq\; b_k(1-x_j)$$

$$\Leftrightarrow \sum_{i\in I_k^+} a_k^i(x_i - x_i x_j) + \sum_{i\in I_k^-} a_k^i(x_i + \bar{x}_i x_j) \leq\; b_k - \left(b_k + \sum_{i\in I_k^-} -a_k^i\right) x_j$$

respectively by

$$\sum_{i\in I_k^+} a_k^i x_i(1-\bar{x}_j) + \sum_{i\in I_k^-} (a_k^i(1-\bar{x}_i))(1-\bar{x}_j) \leq\; b_k(1-\bar{x}_j)$$

$$\Leftrightarrow \sum_{i\in I_k^+} a_k^i(x_i - x_i \bar{x}_j) + \sum_{i\in I_k^-} a_k^i(x_i + \bar{x}_i \bar{x}_j) \leq \left(b_k + \sum_{i\in I_k^-} -a_k^i\right) x_j + \sum_{i\in I_k^-} a_k^i.$$

Here, one may again verify that, if x_j respectively $\bar{x}_j$ is equal to one, then the inequalities enforce the linearization variables to be substituted for the products to equal x_i if $i \in I_k^+$ and to equal $(1-x_i)$ if $i \in I_k^-$, just as desired. Again, this effect is preserved when re-substituting the conceptual $\bar{x}_i$ or $\bar{x}_j$ by respectively $(1-x_i)$ and $(1-x_j)$ in linear expressions.

As a result, we achieved that complement variables and the associated equations need not be introduced. Instead, it now suffices to enforce for each $(i, j) \in Q$ that a linearization variable is induced for at least one of the products $x_i x_j$, $\bar{x}_i x_j$, $x_i \bar{x}_j$, and $\bar{x}_i \bar{x}_j$, and that its associated Conditions 1–3 are satisfied by means of the adapted linearization constraints above. Given this situation, original terms involving *other* products that refer to the same index pair (such as products $x_i x_j$ present in the objective function or in quadratic constraints but without an associated linearization variable induced) may then be substituted for based on the following relations:

$$\begin{aligned}
x_i x_j &= x_j - \bar{x}_i x_j && \Leftrightarrow \bar{x}_i x_j = x_j - x_i x_j \\
x_i x_j &= x_i - x_i \bar{x}_j && \Leftrightarrow x_i \bar{x}_j = x_i - x_i x_j \\
x_i x_j &= \bar{x}_i \bar{x}_j + x_i + x_j - 1 && \Leftrightarrow \bar{x}_i \bar{x}_j = x_i x_j - x_i - x_j + 1
\end{aligned}$$

Remark 2 By contrast, back-substitution into the adapted linearization constraints may invalidate the construction derived above and thus the linearization. It is crucial that the satisfaction of the consistency conditions of at least one linearization variable w.r.t. an index pair remains unaffected by such substitutions, as only this ensures the validity of the latter in arrears.

We conclude that although the handling of negative factor coefficients becomes almost oblivious with the implicit approach, the complexity of deriving an inductive linearization still increases with their presence in practice, and the resulting linearizations may be less compact as multiple linearization variables w.r.t. the same index pair

may be induced along with further corresponding linearization constraints. Nevertheless, the opportunity to eliminate any linearization variable in $Q \setminus P$ as described in Sect. 2.3 is retained. Moreover, besides splitting equations, one may consider both an original equation and its equivalent resulting from a multiplication with -1 as candidates to create linearization constraints (and to induce linearization variables) if the original one has positive and negative coefficients on its left hand side.

3 A computational study

Some evidence for the computational utility of inductive linearizations is already available in the literature for specific applications [e.g. (Davidović et al. 2007; Liberti 2007; Mallach 2017, 2018, 2019)]. Here, we provide a systematic study and a structured overview of computational results for various well-known and well-suited as well as not so well-suited binary quadratic programs with linear constraints.

To this end, we identified a number of prominent combinatorial optimization problems and benchmark instances commonly used by the community in order to evaluate inductive linearizations in comparison with "standard" linearizations. More precisely, we consider and distinguish the following linearizations:

- An inductive linearization (IL), if necessary with implicit normalization.
- A weakened inductive linearization (ILW), with $Q = P$ enforced as described in Sect. 2.3.
- The complete "standard" linearization (SLC) involving $|P|$ additional variables, $3|P|$ additional inequalities, and $7|P|$ additional non-zeros.
- The reduced "standard" linearization (SLR), which only comprises those of the inequalities (2)–(4) that are required due to the objective coefficients.

On the one hand, we evaluate the performance in terms of the running times achieved when passing these linearizations to a mixed-integer programming solver. Here, we emphasize that the purpose of these experiments is to support an assessment towards the question which kind of problem structures (respectively classes of constraints) the inductive linearization technique appears particularly suited or rather not suited for. Clearly, the displayed (wall clock) running times can only serve as an indicator for this assessment, especially as they depend on various influences (as e.g. seeds, parameters, branching decisions, solver versions) whose possible combinations would justify a computational study on their own.

On the other hand, we compare the optimality gaps obtained with the linear relaxations that are associated with the different linearizations. The corresponding figures depict this gap on the ordinate axis in percent, computed as $100(1 - \frac{z_{LP}}{z_{OPT}})$ (respectively $100(1 - \frac{z_{OPT}}{z_{LP}})$), where z_{LP} is the bound obtained by solving the relaxation and z_{OPT} is the optimum value for a minimization (maximization) problem. Where appropriate, we will also put the corresponding results into relation with a first-level RLT. We emphasize that however the bound computed by a MIP solver after solving the linear relaxation may still be better due to presolve mechanisms.

Of course, the different linearizations were created based on the same input formulation in which, as a preprocessing, all linear greater-or-equal inequalities were turned

into less-or-equal ones, and, if a left hand side has only integer coefficients, right hand sides were rounded down if fractional.

To actually derive the inductive linearizations, we employed Algorithm 1. In each iteration of the function APPEND, the cost coefficients associated to the constraint-multiplier combinations (see Sect. 2.3) were recomputed as the negated number of products in Q_{add} for which one of Conditions 1–3 would be satisfied if the respective combination was realized. In contrast, coefficient ranges and right hand sides as well as the number of non-zero coefficients were not taken into account. Further, due to the order of looking up constraints being candidates for multiplications, a slight implicit preference of equations over inequalities (in case of equal costs) is imposed by the implementation. In the subsequent subsections, the table columns titled "A1 [s]" display the running time of this algorithm (i.e., the wall clock time to derive the respective inductive linearization) in seconds.

In order to finally solve the resulting MIPs, we employed Gurobi[3] in version 9.11 with its seed parameter set to one, and its `MIPGap` parameter set to 10^{-6} which was sometimes necessary to ensure a solution process until proven optimality.

All computations were carried out using a single thread on a Debian Linux system equipped with an Intel Xeon E5-2690 CPU (3 GHz) and 128 GB RAM. Each run had a time limit of 48 h. If it was exceeded, this is indicated by "–" in the respective table columns.

3.1 The quadratic assignment problem

Given $T, D \in \mathbb{R}^{n\times n}$ and $c \in \mathbb{R}^n$, a quadratic assignment problem (QAP) in the form by Koopmans and Beckmann (1957) can be written as follows.

$$\min \sum_{i=1}^{n}\sum_{p=1}^{n}\sum_{j=1}^{n}\sum_{q=1}^{n} t_{ij}d_{pq}\, x_{ip}x_{jq} + \sum_{i=1}^{n}\sum_{p=1}^{n} c_{ip}x_{ip}$$

$$\text{s.t.} \sum_{i=1}^{n} x_{ip} = 1 \quad \text{for all } p \in \{1,\dots,n\} \tag{14}$$

$$\sum_{p=1}^{n} x_{ip} = 1 \quad \text{for all } i \in \{1,\dots,n\} \tag{15}$$

$$x_{ip} \geq 0 \quad \text{for all } i, p \in \{1,\dots,n\}$$

$$x_{ip} \in \mathbb{Z} \quad \text{for all } i, p \in \{1,\dots,n\}$$

Frieze and Yadegar (1983) derived a linearization of the QAP that is actually an inductive one, but that is not yet most compact. To characterize a most compact inductive linearization for the case where all (meaningfully) possible products are of interest, observe first that each of the variables $X := \{x_{ip} \mid i, p \in \{1,\dots,n\}\}$ occurs exactly once in the equation set (14) and exactly once in the equation set (15). Thus, in order to induce all products and to satisfy Conditions 1 and 2 for them, it

[3] https://www.gurobi.com/.

would suffice to *either* multiply all of the constraints (14) with X, *or* to multiply all of the constraints (15) with X. Moreover, since objective coefficients for x_{ip}^2 may be moved to the linear part, and since the variables y_{ipiq} for all $p, q \in \{1, \ldots, n\}$ as well as all variables y_{ipjp} for all $i, j \in \{1, \ldots, n\}$ can be eliminated, it suffices to formulate the resulting constraints only for $i \neq j$, or $p \neq q$, respectively. If one further identifies y_{jqip} with y_{ipjq} whenever $i < j$, a most compact inductive linearization of all meaningfully possible products is:

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} \sum_{p=1}^{n} \sum_{j=i+1}^{n} \sum_{p \neq q=1}^{n} (t_{ij} d_{pq} + t_{ji} d_{qp}) y_{ipjq} + \sum_{i=1}^{n} \sum_{p=1}^{n} (c_{ip} + t_{ii} d_{pp}) x_{ip} \\
\text{s.t.} \quad & \sum_{i=1}^{n} x_{ip} = 1 && \text{for all } p \in \{1, \ldots, n\} \\
& \sum_{p=1}^{n} x_{ip} = 1 && \text{for all } i \in \{1, \ldots, n\} \\
& \sum_{i=1}^{j-1} y_{ipjq} + \sum_{i=j+1}^{n} y_{jqip} = x_{jq} && \text{for all } p, j, q \in \{1, \ldots, n\},\ p \neq q \qquad (16) \\
& y_{ipjq} \geq 0 && \text{for all } i, j, p, q \in \{1, \ldots, n\},\ i < j,\ p \neq q \\
& x_{ip} \in \{0, 1\} && \text{for all } i, p \in \{1, \ldots, n\}
\end{aligned}
$$

In this formulation, (16) could also be replaced by the constraints

$$
\begin{aligned}
& \sum_{p=1}^{q-1} y_{ipjq} + \sum_{p=q+1}^{n} y_{ipjq} = x_{jq} && \text{for all } i, j, q \in \{1, \ldots, n\},\ i < j \\
& \sum_{p=1}^{q-1} y_{jqip} + \sum_{p=q+1}^{n} y_{jqip} = x_{jq} && \text{for all } i, j, q \in \{1, \ldots, n\},\ j < i
\end{aligned}
$$

which resembles again the freedom to choose one of (14) and (15) as the basis for inductions.

The total number of additional equations thus amounts to only $n^3 - n^2$ instead of $3 \cdot \left(\frac{1}{2}(n^2 - n)(n^2 - n)\right) = \frac{3}{2}(n^4 - 2n^3 + n^2)$ inequalities when using the complete "standard" linearization and creating y_{ipjq} only for $i < j$ and $p \neq q$ as well. However, these most compact formulations have a weaker linear programming relaxation than the ones by Frieze and Yadegar (1983) and Adams and Johnson (1994) that comprise more constraints.

If P does not contain all meaningfully possible products, several linearization constraints, i.e., constraint-factor combinations, can be saved. The best possible reduction then depends on the actual factor pairs in P and their "distribution" over the assignment constraints.

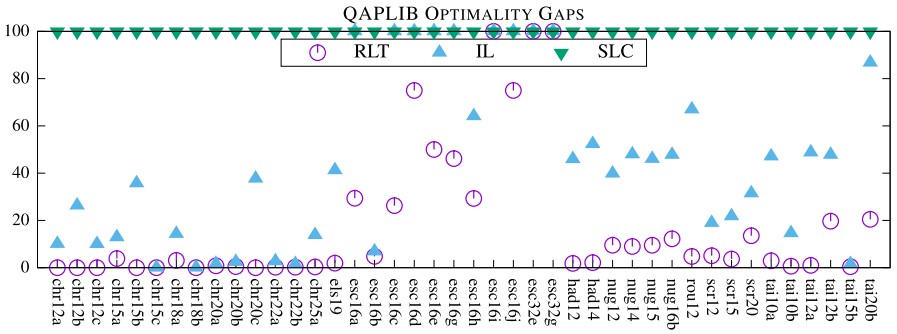

Fig. 1 Optimality gaps obtained with continuous relaxations (in percent) for the QAPLIB instances solved by using at least one of the linearizations (time limit 48 h)

Instance Description

With the only exception of `esc16f` (that has $T = 0$), we report on those of the established QAPLIB (Burkard et al. 1997) instances that could be solved within the 48 h limit using either IL or one of the standard linearizations SLC and SLR. Thereby, the density $100|P|/\binom{|N|}{2}$ of the 42 instances solved spreads between 1 and 87%.

LP Relaxation Bounds and MIP Performance

As shown in Fig. 1, for all considered instances, the "standard" linearization SLC provides a lower bound of zero which translates into an optimality gap of 100%. For some `esc` instances, the computed inductive linearization IL and even a first-level RLT (here derived using square reductions as of Remark 1 in addition) do not improve on this bound. More typically, a strong first-level RLT bound is obtained and IL delivers an optimality gap that is closer to the one of the RLT than to the one of SLC.

As is displayed in Table 1, using Gurobi with IL clearly outperforms its combination with each of the two "standard" linearizations SLC and SLR. With the only exception of `esc32e`, the running times with IL are faster in all the remaining 41 cases, frequently by orders of magnitude. Supported further by the compactness of the derived inductive linearizations, their usually better relaxation strength translates into a superior performance compared with the "standard" linearizations.

We find that assignment constraints prove typically (but not always) suitable for inductive linearizations, even though they are of course not competitive to state-of-the-art methods for the pure QAP. While we here observe that only those QAPLIB instances with up to about 20000 products can be handled within the time limit, it is observed in Sects. 3.5 and 3.6 that inductive linearizations turn out to be frequently attractive for related problems with additional structure in terms of further variables and constraints. This is also in line with results that have been obtained earlier for e.g. quadratic semi-assignment problems (Billionnet and Elloumi 2001), graph partitioning (Mallach 2018), multiprocessor scheduling (Mallach 2017), or graph layering (Mallach 2019).

Table 1 further shows that the task to find a good combination satisfying Conditions 1 and 2 for all induced products could be solved very quickly with Algorithm 1 for the considered QAPLIB instances. In some further experiments, the mixed-integer program from Mallach (2021) could also be solved by Gurobi at the root of its branch-and-bound tree.

Table 1 MIP results for QAPLIB instances solved within 48 h by using at least one of the linearizations (except `esc16f`)

Instance	$\|N\|$	$\|P\|$	SLC	SLR	IL				
			Solve [s]	Solve [s]	$\|Q\|$	$\|M\|$	NZ+	A1 [s]	Solve [s]
esc32e	1024	4992	1.08	**0.51**	5952	384	12,288	0.29	0.90
esc32g	1024	7488	83.06	50.58	8928	576	18,432	0.25	**18.68**
esc16j	256	2112	2.47	0.94	2880	384	6144	0.02	**0.85**
chr25a	625	14,400	–	–	14,400	1200	30,000	0.11	**43.00**
esc16i	256	2640	3.95	3.22	3600	480	7680	0.02	**2.07**
chr22a	484	9702	–	–	9702	924	20,328	0.07	**3.84**
chr22b	484	9702	–	–	9702	924	20,328	0.07	**2.47**
chr20a	400	7220	15,265.27	13,445.08	7220	760	15,200	0.05	**3.66**
chr20b	400	7220	–	–	7220	760	15,200	0.05	**7.58**
chr20c	400	7220	461.58	1178.36	7220	760	15,200	0.05	**10.14**
chr18a	324	5202	3020.19	3993.45	5202	612	11,016	0.03	**1.50**
chr18b	324	5202	–	–	5202	612	11,016	0.03	**0.64**
esc16d	256	3696	19.56	80.49	5040	672	10,752	0.02	**14.13**
esc16e	256	3696	20.61	4.75	5040	672	10,752	0.02	**3.74**
esc16g	256	3696	10.77	12.11	5040	672	10,752	0.02	**4.80**
chr15a	225	2940	238.84	245.43	2940	420	6300	0.02	**0.81**
chr15b	225	2940	38.39	33.93	2940	420	6300	0.02	**0.97**
chr15c	225	2940	476.93	1178.79	2940	420	6300	0.02	**0.05**
chr12a	144	1430	9.36	8.83	1452	264	3168	0.02	**0.31**
chr12b	144	1430	5.30	5.61	1452	264	3168	0.01	**0.28**
chr12c	144	1430	53.51	46.86	1452	264	3168	0.01	**0.32**

Table 1 continued

Instance	$\|N\|$	$\|P\|$	SLC	SLR	IL				
			Solve [s]	Solve [s]	$\|Q\|$	$\|M\|$	NZ+	A1 [s]	Solve [s]
esc16a	256	6688	116.90	60.53	9120	1216	19,456	0.02	**31.93**
esc16c	256	8976	13,0344.85	94,465.73	12,240	1632	26,112	0.02	**3804.05**
els19	361	19,152	625.85	1014.66	19,152	2128	40,432	0.04	**19.94**
scr20	400	23,560	–	–	23,560	2480	49,600	0.05	**4443.16**
scr15	225	8820	1805.14	3565.79	8820	1260	18,900	0.02	**80.31**
scr12	144	3696	34.04	104.92	3696	672	8064	0.01	**6.64**
esc16b	256	16,192	–	–	21,184	2848	45,568	0.02	**25.25**
nug12	144	5940	20,650.09	21,190.65	5940	1080	12,960	0.01	**105.32**
nug16b	256	20,160	–	–	20,160	2688	43,008	0.03	**29,747.55**
esc16h	256	20,240	10,0567.63	67,570.01	22,080	3120	49,920	0.02	**675.30**
nug15	225	15,750	–	–	15,750	2250	33,750	0.03	**12,409.31**
tai10b	100	3150	55.71	59.47	3150	700	7000	0.01	**1.52**
nug14	196	12,376	–	–	12,376	1904	26,656	0.02	**12,910.87**
tai15b	225	17,010	54,407.42	16,9093.38	17,010	2430	36,450	0.02	**245.14**
tai12b	144	7040	1376.05	3431.63	7934	1517	18,204	0.02	**198.62**
tai20b	400	60,040	–	–	60,040	6320	126,400	0.06	**1900.86**
tai10a	100	3870	353.71	695.93	3870	860	8600	0.00	**39.97**
tai12a	144	8448	33,467.24	10,188.60	8448	1536	18,432	0.02	**744.46**
rou12	144	8580	59,224.34	32,873.04	8580	1560	18,720	0.02	**3784.72**
had12	144	8712	14,5050.52	11,3320.70	8712	1584	19,008	0.01	**3078.47**
had14	196	16,562	–	–	16,562	2548	35,672	0.02	**59,730.09**

The instances are sorted in ascending order w.r.t. their density. Each of them has the value of n (here $|N| = n^2$) in its name and positive objective coefficients only. Thus, SLR has $|P|$ additional inequalities and $3|P|$ additional non-zeros. The additional non-zeros for IL are displayed in column "NZ+" The fastest solution times are printed in bold

3.2 The quadratic 0–1 knapsack problem

The certainly simplest inequality-only application for the inductive linearization technique is the quadratic 0–1 knapsack problem (QKP). It is particularly interesting for the computational study because here the left hand side coefficients (item sizes) and the right hand sides (knapsack capacity) vary and they relate to each other at different, and sometimes large, ratios.

The canonical formulation with a capacity $b \in \mathbb{R}$, and a variable x_j for each item j of a ground set J with size $a_j \in \mathbb{R}$ reads:

$$\begin{aligned} \max \quad & \sum_{i,j \in J, i<j} q_{ij} x_i x_j + \sum_{i \in J} c_i x_i \\ \text{s.t.} \quad & \sum_{i \in J} a_i x_i \leq b \\ & x_i \in \{0,1\} \quad \text{for all } i \in J \end{aligned} \tag{17}$$

It has then been observed in the literature that inequalities of type (8) could be used in combination with the "standard" linearization (see e.g., Billionnet and Calmels (1996)), and also inequalities of type (9) have been applied in the context of semidefinite relaxations to improve the obtained dual bounds (Helmberg et al. 2000). A corresponding square-reduced inductive linearization is the following mixed-integer program:

$$\begin{aligned} \max \quad & \sum_{i,j \in J, i<j} q_{ij} y_{ij} + \sum_{i \in J} c_i x_i \\ \text{s.t.} \quad & \sum_{i \in J} a_i x_i \leq b \\ & \sum_{i \in J, i \neq j} a_i y_{ij} \leq (b - a_j) x_j && \text{for all } j \in J \\ & \sum_{i \in J, i \neq j} a_i (x_i - y_{ij}) \leq b(1 - x_j) && \text{for all } j \in J \\ & y_{ij} \in [0,1] && \text{for all } i, j \in J, i < j \\ & x_i \in \{0,1\} && \text{for all } i \in J \end{aligned}$$

Irrespective of the cardinality of P, the Conditions 1–3 must be established using (17) for any product which implies that all possible products are induced as soon as P is non-empty. Thus, this formulation is constraint-side compact, and more and more superior over "standard" linearizations in this respect with increasing density. Assuming $|J| = n$, even a QKP involving all $\binom{n}{2}$ possible products could be linearized using only $2n - 1$ constraints ($n - 1$ inequalities of type (9) suffice to satisfy Condition 3 for all of them while n of (8) are needed for Conditions 1 and 2). However, in the general case of arbitrary a_j, $j \in J$, and b, an implication of the "standard" linearization inequalities (2)–(4) cannot be expected.

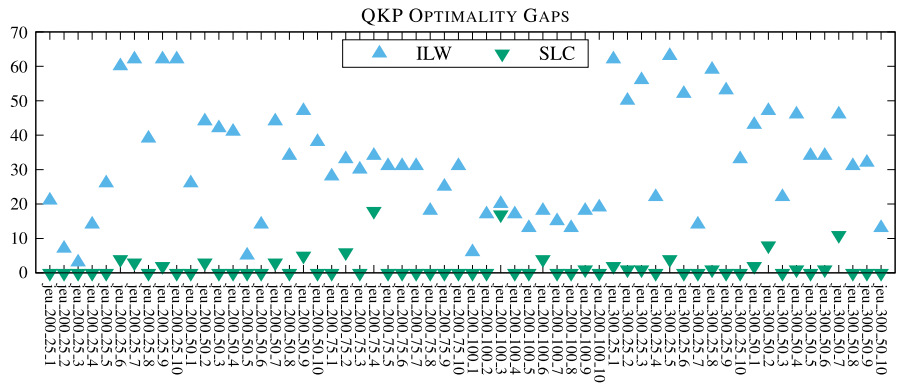

Fig. 2 Optimality gaps obtained with continuous relaxations (in percent) for the sixty selected QKP instances

Instance Description

As a benchmark set, we selected the 60 largest of the randomly generated instances by Billionnet and Soutif (2004) for presentation while the remaining ones with 100 elements could be solved more routinely using all methods. Their naming scheme follows the pattern `jeu_n_d_i` where `n` is the number of items, `d` indicates the approximate product density, and `i` is a running index.

LP Relaxation Bounds and MIP Performance

The optimality gaps obtained with ILW and SLC are shown in Fig. 2. They confirm what could be expected from theory: Due to the comparably large ratios between the constraint coefficients and right hand sides, the upper bounds obtained with ILW (the gap is 31.65% on average) are typically significantly weaker than with SLC (2.10%). While being considerably less compact in size, IL achieves the same bounds as ILW. Since there is only a single inequality comprising all original variables that serves as a basis for inductions, a first-level RLT basically coincides with IL, but involves a "standard" linearization in addition. While the latter is here already able to almost close the gaps standalone, side experiments showed that the much larger full first-level RLT did not suffice as well to close it entirely.

The MIP results are displayed in Table 2. Only in two of the considered cases, it is slightly faster to use ILW than to pass a "standard" linearization to the MIP solver while considerably more timeouts are observed. As it turns out, the improved linearization variable-constraint linkage does not suffice in order to compensate for the weaker relaxation bounds of ILW. Moreover, whereas the numbers of linearization constraints are smaller with ILW than with the "standard" linearizations, the converse is true for the induced numbers of non-zeros. However, despite these preconditions apply broadly, one still observes a considerable variance in the solution times even within each size and product density group, with positive and negative outliers. Moreover, for the densest instances with 200 items in Table 2, only those with a comparably small knapsack capacity remain solvable using ILW while the left hand side coefficient ranges do (almost) not vary over the entire instance set. Even though there is less correlation for the other density groups, the theory-predicted sensitivity of the strength

Table 2 MIP results for sixty QKP instances with 200 and 300 items (time limit 48 h)

Instance	$\|P\|$	LHS	RHS	SLC	SLR	ILW			
				Solve [s]	Solve [s]	$\|M\|$	NZ+	A1 [s]	Solve [s]
jeu_200_25_1	5142	1–50	4027	68.64	**24.79**	399	31,125	0.01	99.28
jeu_200_25_2	5013	1–50	4271	1.45	**1.16**	399	30,351	0.02	8.44
jeu_200_25_3	4984	1–50	4763	2.33	**0.99**	399	30,173	0.01	10.12
jeu_200_25_4	5105	1–50	4518	**12.76**	27.15	399	30,895	0.01	22.76
jeu_200_25_5	4965	1–50	3688	**6.07**	14.19	399	30,063	0.01	19.47
jeu_200_25_6	5053	1–50	1700	102.65	**40.33**	399	30,591	0.01	48.35
jeu_200_25_7	4930	1–50	1242	**11.40**	27.46	399	29,849	0.01	88.51
jeu_200_25_8	4880	1–50	2861	**5.92**	15.14	399	29,555	0.01	15.86
jeu_200_25_9	4846	1–50	1012	10.17	**7.43**	399	29,353	0.01	78.14
jeu_200_25_10	4885	1–50	990	17.07	**5.93**	399	29,579	0.02	27.29
jeu_200_50_1	10,028	1–50	3547	1.76	**1.37**	399	60,321	0.01	10.03
jeu_200_50_2	10,047	1–50	2245	3526.24	**1021.62**	399	60,445	0.01	–
jeu_200_50_3	9906	1–50	2120	175.45	**46.06**	399	59,609	0.01	17,308.21
jeu_200_50_4	10,002	1–50	2132	1.56	**1.26**	399	60,159	0.02	20.34
jeu_200_50_5	9910	1–50	4833	**35.85**	201.36	399	59,617	0.01	454.80
jeu_200_50_6	9930	1–50	4354	**2183.73**	5363.41	399	59,745	0.02	–
jeu_200_50_7	9995	1–50	2430	2323.07	**457.21**	399	60,141	0.02	3460.83
jeu_200_50_8	9877	1–50	3159	**5.49**	11.05	399	59,419	0.02	84.13
jeu_200_50_9	9857	2–50	1105	246.87	**71.67**	399	59,313	0.02	–
jeu_200_50_10	9881	1–50	2756	59.96	**41.37**	399	59,443	0.01	144.48

Table 2 continued

Instance	$\|P\|$	LHS	RHS	SLC	SLR	ILW			
				Solve [s]	Solve [s]	$\|M\|$	NZ+	A1 [s]	Solve [s]
jeu_200_75_1	14,894	1–50	2866	**152.85**	282.07	399	89,425	0.01	–
jeu_200_75_2	14,935	1–50	2139	5528.92	**1407.65**	399	89,685	0.01	–
jeu_200_75_3	14,869	1–50	408	134.08	51.88	399	89,283	0.02	**51.23**
jeu_200_75_4	14,839	1–50	1220	**72.01**	168.76	399	89,105	0.02	–
jeu_200_75_5	14,920	1–50	859	475.26	**53.63**	399	89,589	0.02	–
jeu_200_75_6	14,898	1–50	1349	**11.77**	12.26	399	89,457	0.02	–
jeu_200_75_7	15,051	1–50	1736	**13.33**	103.06	399	90,373	0.02	–
jeu_200_75_8	14,836	1–50	3412	**5.70**	5.99	399	89,079	0.02	144.83
jeu_200_75_9	14,890	1–50	3412	**2159.61**	2397.77	399	89,407	0.01	–
jeu_200_75_10	14,863	1–50	955	41.58	**17.16**	399	89,237	0.02	–
jeu_200_100_1	19,900	1–50	4785	–	–	399	119,401	0.02	–
jeu_200_100_2	19,900	1–50	1325	1359.39	**941.21**	399	119,401	0.01	–
jeu_200_100_3	19,900	1–50	173	15.63	13.16	399	119,401	0.02	**6.17**
jeu_200_100_4	19,900	1–50	498	15.32	**10.68**	399	119,401	0.01	34.79
jeu_200_100_5	19,900	1–50	4005	**43.10**	2557.07	399	119,401	0.01	–
jeu_200_100_6	19,900	1–50	195	6.14	**4.42**	399	119,401	0.02	5.77
jeu_200_100_7	19,900	1–50	3583	**127.29**	2084.33	399	119,401	0.01	–
jeu_200_100_8	19,900	1–50	3677	**7122.79**	16,832.06	399	119,401	0.02	–
jeu_200_100_9	19,900	1–50	3256	**395.66**	1278.81	399	119,401	0.02	–
jeu_200_100_10	19,900	1–50	1859	48,407.79	**1853.81**	399	119,401	0.02	–

Table 2 continued

Instance	$\|P\|$	LHS	RHS	SLC	SLR	ILW			
				Solve [s]	Solve [s]	$\|M\|$	NZ+	A1 [s]	Solve [s]
jeu_300_25_1	11,339	1–50	376	12.99	**4.30**	599	68,447	0.03	70.88
jeu_300_25_2	11,201	1–50	3878	207.73	**130.67**	599	67,623	0.03	133.93
jeu_300_25_3	11,157	1–50	3128	**52.16**	60.23	599	67,341	0.03	79.04
jeu_300_25_4	11,256	1–50	5759	69.01	**61.69**	599	67,951	0.03	125.63
jeu_300_25_5	11,403	1–50	197	**7.70**	15.95	599	68,833	0.03	42.70
jeu_300_25_6	11,406	1–50	3451	43.79	**25.59**	599	68,861	0.03	57.04
jeu_300_25_7	11,192	1–50	6282	**31.04**	94.61	599	67,569	0.03	103.40
jeu_300_25_8	11,157	1–50	114	2.61	**1.74**	599	67,351	0.03	7.15
jeu_300_25_9	11,212	1–50	3150	64.47	**61.51**	599	67,689	0.03	77.52
jeu_300_25_10	11,340	1–50	5322	**111.64**	190.87	599	68,431	0.03	146.62
jeu_300_50_1	22,534	1–50	3550	45,101.55	**4177.61**	599	135,457	0.03	–
jeu_300_50_2	22,436	1–50	794	368.21	**69.41**	599	134,873	0.03	–
jeu_300_50_3	22,351	1–50	5946	**165.15**	1385.97	599	134,359	0.03	4094.58
jeu_300_50_4	22,403	1–50	1957	1278.35	**566.32**	599	134,655	0.03	–
jeu_300_50_5	22,400	1–50	4912	**1092.21**	1399.43	599	134,645	0.03	1122.80
jeu_300_50_6	22,444	1–50	5106	**1768.96**	6667.74	599	134,905	0.03	5813.68
jeu_300_50_7	22,580	1–50	328	43.23	**41.20**	599	135,725	0.03	69.51
jeu_300_50_8	22,253	1–50	5069	3717.43	**2915.27**	599	133,765	0.03	3103.92
jeu_300_50_9	22,321	1–50	4907	**105.61**	319.16	599	134,177	0.03	327.98
jeu_300_50_10	22,476	1–50	7040	**74.54**	3590.43	599	135,099	0.03	3189.32

The third column gives the interval of left hand side coefficients and the fourth column specifies the value of the right hand side of the knapsack constraint. Concerning the inductive linearization, the presentation is restricted to ILW that clearly outperformed IL. SLR has here $2|P|$ additional inequalities, and $4|P|$ additional non-zeros as all the instances have positive objective coefficients only. The additional non-zeros for ILW are displayed in column "NZ+"
The fastest solution times are printed in bold

of inductive linearizations to the "ratio" between the right hand side and the left hand side coefficients is here apparent.

The suitability of using inductive linearizations for more complex problems with knapsack constraints may as well be influenced by the mentioned ratio and of course by the additional inherent structure. As mentioned above, deriving an inductive linearization is trivial when based on the knapsack inequality and the unique solution was also computed fast with Algorithm 1 in our experiments.

3.3 The quadratic matching problem

Given an undirected graph $G = (V, E)$, a canonical BQP that models the quadratic matching problem (QMP) on G [see e.g. (Hupp et al. 2015)] can be expressed as

$$\begin{aligned} \max \quad & \sum_{e,f \in E, e \neq f} q_{ef} x_e x_f + \sum_{e \in E} c_e x_e \\ \text{s.t.} \quad & \sum_{e:\{i,j\} \in E} x_e \leq 1 && \text{for all } i \in V \\ & x_e \in \{0, 1\} && \text{for all } e \in E. \end{aligned} \tag{18}$$

It is easily observed that each edge (factor) occurs in exactly two constraints, namely those inequalities (18) associated with its endpoints. In this sense, the situation is similar as in case of the QAP. However, the constraints need not be as regular as they depend on the structure of G. Notwithstanding, their coefficients and right hand sides ensure that (the relaxation of) an inductive linearization will imply the inequalities of a "standard linearization".

Instance Description

For our experiments, we first employed all the instances used by Hupp et al. (2015), and finally selected a representative subset of those called "BM" in this reference, as the other ones with less variables or products could be solved quickly using all linearization approaches, and the other instances of about the same size produced similar results as has been the case also in the reference. The product densities of the selected instances lie between 63% and 83%, increasing from suffixes `_80` to `_100`.

LP Relaxation Bounds and MIP Performance

As can be seen from Fig. 3, the optimality gaps obtained when solving the relaxations of the 50 representative instances with ILW are close to the strong RLT bounds while the upper bounds obtained with SLC are weak. Over the entire "BM" set consisting of 150 instances, the bounds are always within 13% and 32% for ILW, within 5% and 30% using the RLT, and between 84% and 92% for SLC.

Concerning the MIP solver performance, there is a clear picture: For all the "BM" instances (including the representative ones displayed in Table 3), Gurobi could solve the problems better using one of the inductive linearizations than when using SLC or SLR. In 101 of the entire 150 cases, the best results are obtained with ILW that, on average, took about 36% and 42% of the running time that was required with SLC and SLR, respectively. Besides the stronger relaxations, this performance of the

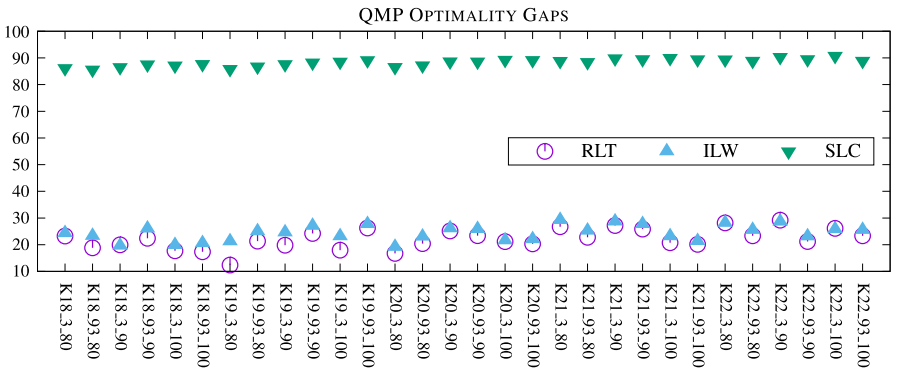

Fig. 3 Optimality gaps obtained with continuous relaxations (in percent) for the fifty selected QMP instances

inductive linearizations is further supported by the constraint-side compactness while the number of induced non-zeros is typically higher. Although using ILW turned out to be more effective than using IL, both are displayed in Table 3 in order to support an assessment of the model size reduction obtained by the weakening. Roughly, the number of linearization constraints compared to IL was reduced by the weakening procedure by about 21%, 12%, and 2% for the instances with the suffixes `_80`, `_90`, and `_100`, respectively. The running times obtained indicate only a slight and inconsistent correlation to the three density levels, and with IL and ILW the numbers of additional constraints and the additional non-zeros generated by Algorithm 1 are typically even smaller for the denser instances.

We conclude that the matching constraints appear as rather attractive candidates for inductive linearizations. In additional experiments, computing an inductive linearization with a mixed-integer programs turned out to be comparably more difficult for the MIP solver employed. On the contrary, Algorithm 1 delivered good linearizations routinely and quickly.

3.4 The quadratic shortest path problem

Given a directed graph $G = (V, A)$, a source node $s \in V$ and a target node $t \in V$, a canonical BQP that models the quadratic shortest s-t-path problem (QSPP) on G (see e.g. Rostami et al. (2018)) can be expressed as

$$\begin{aligned} \min \quad & \sum_{a,b \in A, a \neq b} q_{ab} x_a x_b + \sum_{a \in A} c_a x_a \\ \text{s.t.} \quad & \sum_{a:(i,j) \in A} x_a - \sum_{a:(j,i) \in A} x_a = b(i) \quad \text{for all } i \in V \\ & x_a \in \{0, 1\} \quad \text{for all } a \in A, \end{aligned} \tag{19}$$

where $b(i) = 0$ for all $i \in V \backslash \{s, t\}$, $b(s) = 1$, and $b(t) = -1$.

Table 3 MIP results for QMP instances

Instance	$\|N\|$	$\|P\|$	SLC	SLR			IL					ILW			
			Solve [s]	Ineq	NZ+	Solve [s]	$\|Q\|$	$\|M\|$	NZ+	A1 [s]	Solve [s]	$\|M\|$	NZ+	A1 [s]	Solve [s]
K18_3_80	153	7344	712.9	11,057	25,745	1226.12	9173	3701	83,356	0.01	417.48	3701	65,832	0.01	**337.63**
K18_93_80	153	7344	820.47	11,016	25,704	1196.44	9174	3716	83,817	0.01	405.96	3716	66,203	0.02	**364.33**
K18_3_90	153	8262	781.72	12,450	28,974	1417.91	9174	3610	80,319	0.01	348.62	3610	70,775	0.02	**323.96**
K18_93_90	153	8262	2799.3	12,394	28,918	1818.34	9177	3637	81,108	0.01	571.00	3637	71,409	0.01	**507.00**
K18_3_100	153	9180	1148.17	13,806	32,166	1986.93	9180	3110	63,615	0.01	**368.43**	3110	61,985	0.01	387.08
K18_93_100	153	9180	1099.73	13,770	32,130	1819.45	9180	3110	63,615	0.01	365.24	3110	61,985	0.02	**329.49**
K19_3_80	171	9303	1210.87	13,991	32,597	2134.02	11,625	4469	107,275	0.01	902.62	4469	84,823	0.02	**660.93**
K19_93_80	171	9303	1451.92	13,948	32,554	1033.92	11,623	4456	106,892	0.01	831.30	4456	84,588	0.02	**720.52**
K19_3_90	171	10,466	2161.09	15,725	36,657	2812.64	11,623	4323	102,237	0.01	**891.23**	4323	90,097	0.02	935.09
K19_93_90	171	10,466	2498.86	15,697	36,629	3580.45	11,619	4322	102,346	0.01	1097.33	4322	90,177	0.02	**933.85**
K19_3_100	171	11,628	2801.27	17,460	40,716	3605.27	11,628	3704	80,392	0.01	**938.71**	3704	78,456	0.02	959.27
K19_93_100	171	11,628	4828.07	17,420	40,676	5371.73	11,628	3704	80,392	0.01	2748.25	3704	78,456	0.02	**1654.43**
K20_3_80	190	11,628	2900.87	17,468	40,724	5018.20	14,530	5270	133,810	0.01	1539.10	5270	105,921	0.02	**1226.82**
K20_93_80	190	11,628	8433.19	17,424	40,680	6646.60	14,530	5252	133,144	0.01	**2372.57**	5252	105,307	0.02	3450.33
K20_3_90	190	13,082	14,971.2	19,631	45,795	10, 067.05	14,533	5148	129,182	0.02	6140.68	5148	113,921	0.02	**3525.25**
K20_93_90	190	13,082	14,153.77	19,609	45,773	8607.70	14,533	5169	129,959	0.01	3374.80	5169	114,785	0.02	**3095.12**
K20_3_100	190	14,535	14,389.77	21,839	50,909	10, 777.90	14,535	4369	100,283	0.01	**2623.22**	4369	98,005	0.02	5673.50
K20_93_100	190	14,535	17,472.3	21,761	50,831	13, 232.15	14,535	4369	100,283	0.01	**2594.51**	4369	98,005	0.02	4593.54

Table 3 continued

Instance	$\|N\|$	$\|P\|$	SLC	SLR			IL					ILW			
			Solve [s]	Ineq	NZ+	Solve [s]	$\|Q\|$	$\|M\|$	NZ+	A1 [s]	Solve [s]	$\|M\|$	NZ+	A1 [s]	Solve [s]
K21_3_80	210	14,364	20,896.08	21,570	50,298	14, 241.93	17,949	6231	167,649	0.02	10,329.48	6231	132,584	0.02	**9170.29**
K21_93_80	210	14,364	8088.23	21,508	50,236	12, 995.96	17,951	6208	166,672	0.02	6892.78	6208	131,842	0.02	**6056.94**
K21_3_90	210	16,160	28,160.35	24,254	56,574	16, 921.53	17,953	6094	162,146	0.02	10,309.10	6094	143,142	0.02	**9808.24**
K21_93_90	210	16,160	26,248.96	24,185	56,505	18, 076.83	17,949	6002	158,718	0.02	13,745.99	6002	140,278	0.02	**9395.55**
K21_3_100	210	17,955	15,908.77	26,943	62,853	22, 677.88	17,955	5109	123,651	0.02	4477.94	5109	120,993	0.02	**4366.52**
K21_93_100	210	17,955	12,342.29	26,907	62,817	17175.62	17,955	5109	123,651	0.02	4895.42	5109	120,993	0.02	**3805.73**
K22_3_80	231	17,556	61,509.99	26,354	61,466	43,650.05	21,939	7294	207,137	0.02	**25,657.52**	7294	164,123	0.02	33851.39
K22_93_80	231	17,556	45,138.04	26,310	61,422	29,255.38	21,940	7231	204,512	0.02	**13,909.77**	7231	161,770	0.02	16,650.54
K22_3_90	231	19,751	10,7845.78	29,605	69,107	53,720.78	21,940	7089	198,690	0.02	35,845.89	7089	175,698	0.02	**35,254.61**
K22_93_90	231	19,751	63,734.22	29,597	69,099	35,114.34	21,943	7092	198,687	0.02	**20,895.07**	7092	175,552	0.02	21,459.32
K22_3_100	231	21,945	10,5069.34	32,892	76,782	76,238.32	21,945	5928	150,879	0.02	**27,527.84**	5928	147,801	0.02	28,042.02
K22_93_100	231	21,945	78,275.46	32,914	76,804	57210.90	21,945	5928	150,879	0.02	**21,124.73**	5928	147,801	0.02	21634.03

Since these have products with positive and negative objective coefficients, the exact number of additional inequalities ("Ineq") and non-zeros ("NZ+") for SLR vary. They are given explicitly as it is the case for IL and ILW (having $Q = P$) as well

The fastest solution times are printed in bold

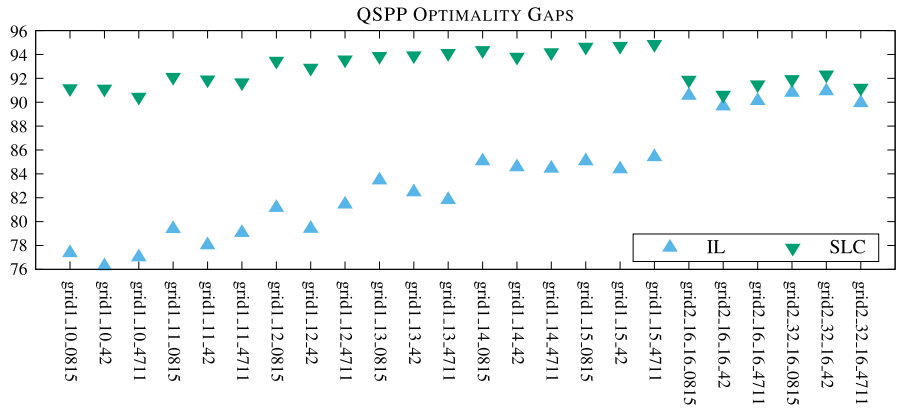

Fig. 4 Optimality gaps obtained with continuous relaxations (in percent) for the QSPP instances solved by using at least one of the linearizations (time limit 48 h)

Since the left-hand side coefficients of the only constraints (19) are from the set $\{-1, 1\}$, normalization is inevitably required when creating an inductive linearization of this BQP.

Instance Description

For the experiments, we self-generated grid instances as described by Rostami et al. (2018), three for each proposed type and size. Their density is about 50%.

LP Relaxation Bounds and MIP Performance

The given density and instance sizes lead to a relatively large number of products so that only 24 out of the 39 generated instances could be solved within an 48 h time limit when passing one of the linearizations to the MIP solver.

Looking at the optimality gaps obtained for these instances as shown in Fig. 4, the bounds computed with IL tend to get weaker with increasing size while the bounds derived with the "standard" linearization reside on an even weaker level. More precisely, for the solved `grid1` instances, the gaps are between 76% and 85% with IL and between 90% and 95% with SLC. In case of the solved `grid2` instances the optimality gaps are closer to each other and between 89% and 93%. The better but still rather weak IL bounds are attenuated by effects of the normalization as is further described below. The discussion indicates a direction to achieve potential bound improvements as well as an explanation why the further addition of constraint-variable multiplications is not auspicious, while especially the model size of a first-level RLT would here be anyway to excessive.

As one can see from Table 4, SLC and especially SLR work better with Gurobi for the considered instances. Despite that the LP bounds obtained with the "standard" linearizations are even worse than with IL, the obtained running times are orders of magnitudes faster than with IL. The negative coefficients respectively the model size play here an important role: Influenced by the required normalization, the number of induced products and non-zeros is significantly larger using IL while many linearization variables referring to the same pair of original variables are generated. Moreover, the constraint linkage among these linearization variables is poor and so

Table 4 MIP results for QSPP instances solved by using at least one of the linearizations (time limit 48 h)

Instance	$\|A\|$	$\|V\|$	$\|P\|$	SLC	SLR	IL				
				Solve [s]	Solve [s]	$\|Q\|$	$\|M\|$	NZ+	A1 [s]	Solve [s]
grid1_10_0815	180	100	8010	96.92	**64.87**	33,360	19,216	88,502	0.05	1106.19
grid1_10_42	180	100	8010	**60.61**	75.71	33,360	19,216	88,502	0.05	972.28
grid1_10_4711	180	100	8010	128.64	**34.85**	33,360	19,216	88,502	0.05	1178.54
grid1_11_0815	220	121	11,990	310.34	**185.40**	49,982	28,426	131,962	0.08	6886.59
grid1_11_42	220	121	11,990	317.42	**181.26**	49,982	28,426	131,962	0.08	4457.41
grid1_11_4711	220	121	11,990	**317.20**	358.32	49,982	28,426	131,962	0.09	4714.31
grid1_12_0815	264	144	17,292	1710.15	**756.30**	72,090	40,568	189,558	0.12	30,972.13
grid1_12_42	264	144	17,292	1527.37	**569.71**	72,090	40,568	189,558	0.12	28,713.18
grid1_12_4711	264	144	17,292	1117.61	**588.56**	72,090	40,568	189,558	0.12	32,184.93
grid1_13_0815	312	169	24,180	5246.88	**2734.81**	100,764	56,200	264,032	0.17	13,2808.27
grid1_13_42	312	169	24,180	4547.93	**2474.63**	100,764	56,200	264,032	0.16	13,1224.59
grid1_13_4711	312	169	24,180	4232.15	**2572.58**	100,764	56,200	264,032	0.16	12,6878.77

Table 4 continued

Instance	$\|A\|$	$\|V\|$	$\|P\|$	SLC	SLR	IL				
				Solve [s]	Solve [s]	$\|Q\|$	$\|M\|$	NZ+	A1 [s]	Solve [s]
grid1_14_0815	364	196	32,942	18,302.35	**10,619.37**	137,180	75,928	358,366	0.23	–
grid1_14_42	364	196	32,942	23,212.84	**14,277.08**	137,180	75,928	358,366	0.22	–
grid1_14_4711	364	196	32,942	18,005.20	**12,659.37**	137,180	75,928	358,366	0.23	–
grid1_15_0815	420	225	43,890	**68,029.07**	97,204.98	182,610	100,406	475,782	0.30	–
grid1_15_42	420	225	43,890	50,130.75	**49592.79**	182,610	100,406	475,782	0.29	–
grid1_15_4711	420	225	43,890	71,814.46	**44,802.88**	182,610	100,406	475,782	0.30	–
grid2_16_16_0815	512	258	65,416	5144.08	**509.03**	284,856	144,642	728,528	0.40	10,2981.29
grid2_16_16_42	512	258	65,416	3092.66	**408.14**	284,856	144,642	728,528	0.40	92,653.53
grid2_16_16_4711	512	258	65,416	3078.51	**675.97**	284,856	144,642	728,528	0.41	10,6852.08
grid2_32_16_0815	1040	514	269,960	27,591.82	**21260.33**	1,174,432	579,914	2,974,504	1.75	–
grid2_32_16_42	1040	514	269,960	**15,871.29**	11,1965.82	1,174,432	579,914	2,974,504	1.71	–
grid2_32_16_4711	1040	514	269,960	**15,622.74**	14,1909.80	1,174,432	579,914	2,974,504	1.70	–

The second column gives the number of arcs (variables) and the third column gives the number of original equations. Since the instances have positive objective coefficients only, SLR has $|P|$ additional inequalities and $3|P|$ additional non-zeros. The additional non-zeros for IL are displayed in column "NZ+"
The fastest solution times are printed in bold

is consequently the impact on the bound. Therefore, although one could reduce the total number of linearization variables by a factor of about three when investing more time to derive an inductive linearization using the MIP from Mallach (2021), which improves the solution times considerably, the latter are still not competitive to those obtained with the "standard" linearization, and especially not to those obtained with specialized methods for the QSPP as in Rostami et al. (2018). Moreover, for some instances, even the root linear program of IL turned out to be comparably challenging to solve with Gurobi.

The suitability of typical "flow conservation" constraints like (19) to derive inductive linearizations thus appears to be limited so far. However, side experiments indicate that strong bounds could be achieved with an inductive linearization that resolves the indicated poor linearization variable-constraint linkage while preserving consistency (e.g. using careful back-substitutions as indicated in Sect. 2.4) which is a subject for further research.

Currently, the structure of the present constraints where each potential factor (arc) occurs on the left hand sides of two constraints (those associated with its endpoints), but with opposite signs, impairs the MIP solution times to derive an inductive linearization. Algorithm 1 is also but less affected. Not surprisingly, for the largest instances with $|P| > 10^5$, a noticeable increase in the derivation time occurs.

3.5 MINLPLib

To further broaden the experimental setting, we incorporated library collections such as the MINLPLib. In addition, they provide instance formats that one may pass on directly to Gurobi (we used the `.lp` format) giving rise to another meaningful comparison.

Instance Description

We employed all linearly constrained BQPs from MINLPLib except for six QSPP instances and one QAP instance. All except ten of these instances have equations only, however also for these exceptions only equations were employed to create the inductive linearizations. The first two of these exceptions are `crossdock_15x7` and `crossdock_15x8` which have 30 equations (with only zero–one coefficients and a right hand side of one) and 14 respectively 16 inequalities (with larger coefficients and right hand sides). The same applies to the eight generalized assignment problem instances with prefix `pb` whose first (second) two digits refer to the number of equations (inequalities). Stemming from Cordeau et al. (2006), they have been contributed to the MINLPLib by Monique Guignard. In Pessoa et al. (2010), the instance `pb351575` still remained unsolved due to a missing optimality certificate for the objective value of 6301723. In all conscience, its solution has not been reported until present. By investing more computational resources and time in addition to the main experiments presented, we could solve all these instances to optimality using IL and thus confirm their optimal values as listed in Table 5.

Table 5 Confirmed optimal solution values for some generalized assignment problem instances from the MINLPLib

Instance	Optimal Value	Instance	Optimal value
pb302035	3379359	pb351535	4456670
pb302055	3593105	pb351555	4639128
pb302075	4050938	pb351575	6301723
pb302095	5710645	pb351595	6670264

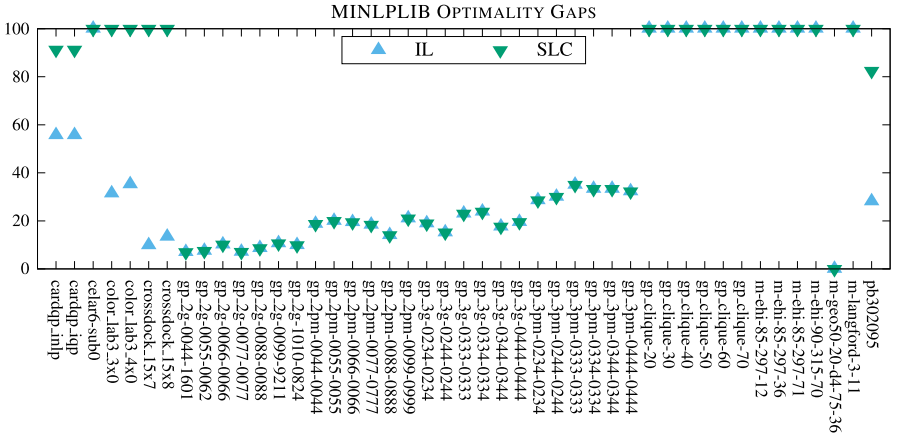

Fig. 5 Optimality gaps obtained with continuous relaxations (in percent) for the MINLPLib instances solved by using at least one of the linearizations (time limit 48 h). Some instance names where shortened, `gp-` stands for `graphpart-` and `m-` for `maxcsp-`

LP Relaxation Bounds and MIP Performance

Figure 5 displays the sustained relaxation optimality gaps for the instances that could be solved within 48 h using at least one method. The MIP results are shown in Tables 6 and 7.

Besides the already mentioned `crossdock` and `pb` instances, the two `color_lab` instances are the only ones that do not have their non-zero left hand side coefficients and right hand sides all equal to one. As forward referenced from Sect. 3.1, an inductive linearization is frequently (but not always) suited and superior to a "standard" linearization for instances with this structure. In several cases, this is also true when compared with Gurobi as a standalone solver.

On the one hand, for the `crossdock` and the `pb` instances, the improved bounds with IL translate into a better MIP performance, supported further by the ideal situation that IL is here also very effective in terms of the number of linearization equations *and* additional non-zeros. Moreover, for `celar6-sub0`, the inductive linearization is orders of magnitude faster even though there are many products and induced non-zeros, and even though there is no according indication in terms of the relaxation optimality gaps.

On the other hand, for the maximum constraint satisfiability problems, SLR performs clearly best with Gurobi while IL requires less linearization constraints but induces more non-zeros and linearization variables, and even its root linear program

Table 6 MIP results: Linearly constrained BQPs from the MINLPLib (1st part)

Instance	$\|N\|$	m_L	$\|P\|$	LHS	RHS	SLC	SLR			IL					GRB
						Solve [s]	Ineq	NZ+	Solve [s]	$\|Q\|$	$\|M\|$	NZ+	A1 [s]	Solve [s]	Solve [s]
gp_2g-0044-1601	48	16	96	1–1	1–1	0.03	141	333	0.02	288	192	768	0.00	0.06	**0.01**
gp_2g-0055-0062	75	25	150	1–1	1–1	0.08	219	519	**0.06**	450	300	1200	0.00	0.11	**0.06**
gp_2g-0066-0066	108	36	216	1–1	1–1	0.27	342	774	0.15	648	432	1728	0.01	0.19	**0.13**
gp_2g-0077-0077	147	49	294	1–1	1–1	0.29	426	1014	0.27	882	588	2352	0.01	0.32	**0.13**
gp_2g-0088-0088	192	64	384	1–1	1–1	0.34	612	1380	0.24	1152	768	3072	0.02	0.50	**0.18**
gp_2g-0099-9211	243	81	486	1–1	1–1	0.50	711	1683	0.38	1458	972	3888	0.03	1.99	**0.31**
gp_2g-1010-0824	300	100	600	1–1	1–1	0.57	891	2091	0.36	1800	1200	4800	0.04	2.95	**0.23**
gp_2pm-0044-0044	48	16	96	1–1	1–1	0.03	144	336	0.03	288	192	768	0.00	0.07	**0.01**
gp_2pm-0055-0055	75	25	150	1–1	1–1	0.09	225	525	**0.07**	450	300	1200	0.01	0.11	0.08
gp_2pm-0066-0066	108	36	216	1–1	1–1	0.12	324	756	0.15	648	432	1728	0.01	0.29	**0.08**
gp_2pm-0077-0777	147	49	294	1–1	1–1	0.20	441	1029	0.22	882	588	2352	0.01	0.93	**0.14**
gp_2pm-0088-0888	192	64	384	1–1	1–1	0.22	576	1344	0.20	1152	768	3072	0.02	0.54	**0.11**

Table 6 continued

Instance	$\|N\|$	m_L	$\|P\|$	LHS	RHS	SLC	SLR			IL					GRB
						Solve [s]	Ineq	NZ+	Solve [s]	$\|Q\|$	$\|M\|$	NZ+	A1 [s]	Solve [s]	Solve [s]
gp_2pm-0099-0999	243	81	486	1–1	1–1	0.69	729	1701	0.50	1458	972	3888	0.03	17.93	**0.31**
gp_3g-0234-0234	72	24	180	1–1	1–1	0.17	264	624	0.13	540	360	1440	0.00	**0.12**	**0.12**
gp_3g-0244-0244	96	32	240	1–1	1–1	0.32	354	834	0.28	720	480	1920	0.01	**0.19**	0.20
gp_3g-0333-0333	81	27	243	1–1	1–1	0.16	345	831	0.14	729	486	1944	0.00	0.20	**0.09**
gp_3g-0334-0334	108	36	324	1–1	1–1	0.33	471	1119	0.21	972	648	2592	0.01	0.82	**0.15**
gp_3g-0344-0344	144	48	432	1–1	1–1	0.90	648	1512	0.46	1296	864	3456	0.01	0.61	**0.32**
gp_3g-0444-0444	192	64	576	1–1	1–1	1.56	900	2052	**1.18**	1728	1152	4608	0.02	5.17	3.35
gp_3pm-0234-0234	72	24	180	1–1	1–1	0.19	264	624	0.11	540	360	1440	0.00	0.24	**0.08**
gp_3pm-0244-0244	96	32	240	1–1	1–1	0.34	360	840	0.23	720	480	1920	0.01	0.66	**0.14**
gp_3pm-0333-0333	81	27	243	1–1	1–1	0.12	363	849	**0.07**	729	486	1944	0.00	0.67	**0.07**
gp_3pm-0334-0334	108	36	324	1–1	1–1	0.66	486	1134	0.50	972	648	2592	0.01	2.57	**0.12**
gp_3pm-0344-0344	144	48	432	1–1	1–1	**0.84**	648	1512	0.93	1296	864	3456	0.02	21.77	1.50
gp_3pm-0444-0444	192	64	576	1–1	1–1	21.40	864	2016	20.61	1728	1152	4608	0.02	259.74	**11.72**

The fifth column gives the interval of left hand side coefficients and the sixth column specifies the ranges of the right hand sides. For SLR, the varying exact number of additional inequalities ("Ineq") and non-zeros ("NZ+") are displayed explicitly. The same is true for IL. The columns labeled "GRB" display the wall clock time in seconds when passing the `.lp`-file directly to Gurobi. The prefix "gp" abbreviates "graphpart" The fastest solution times are printed in bold

Table 7 MIP results: Linearly constrained BQPs from the MINLPLib (time limit 48 h, 2nd part)

Instance	$\|N\|$	m_L	$\|P\|$	LHS	RHS	SLC	SLR	IL					GRB
						Solve [s]	Solve [s]	$\|Q\|$	$\|M\|$	NZ+	A1 [s]	Solve [s]	Solve [s]
cardqp_inlp / _iqp	50	1	1225	1–1	10–10	25.37	144.49	1225	50	2500	0.00	270.37	**18.51**
celar6-sub0	640	16	64,032	1–1	1–1	2515.80	1937.70	93,744	4624	192,112	0.11	**31.78**	142.34
color_lab3_3x0	316	80	2241	1–1	0–1	266.47	**57.05**	3984	1992	9960	0.04	14,349.98	70.78
color_lab3_4x0	395	80	3984	1–1	0–1	**60,106.29**	–	6225	2490	14,940	0.05	–	–
crossdock_15x7	210	44	2793	1–233	1–302	953.87	770.44	2793	798	6384	0.02	**156.66**	1562.87
crossdock_15x8	240	46	3648	1–209	1–233	1732.60	1141.33	3648	912	8208	0.02	**401.92**	29,883.15
gp_clique-20	60	20	570	1–1	1–1	1.01	**0.44**	1710	1140	4560	0.00	0.57	0.89
gp_clique-30	90	30	1305	1–1	1–1	9.09	**6.49**	3915	2610	10,440	0.01	8.07	6.74
gp_clique-40	120	40	2340	1–1	1–1	58.63	**39.13**	7020	4680	18,720	0.01	75.34	40.52
gp_clique-50	150	50	3675	1–1	1–1	746.27	766.82	11,025	7350	29,400	0.02	986.70	**357.88**
gp_clique-60	180	60	5310	1–1	1–1	**5428.22**	19,639.69	15,930	10,620	42,480	0.02	5524.39	18,400.19
gp_clique-70	210	70	7245	1–1	1–1	15,646.96	**6931.17**	21,735	14,490	57,960	0.02	82,306.97	–
m-ehi-85-297-12	2071	297	101,333	1–1	1–1	64,617.37	**1742.66**	199,509	57,192	456,210	1.38	44,245.78	10,7791.48
m-ehi-85-297-36	2046	297	99,286	1–1	1–1	41,759.38	**3147.43**	196,266	56,782	449,314	1.36	61,696.60	78,597.00
m-ehi-85-297-71	2075	297	101,923	1–1	1–1	11,434.65	**11,104.69**	200,794	57,454	459,042	1.43	11,9508.18	40,835.13
m-ehi-90-315-70	2203	315	108,662	1–1	1–1	–	**8260.88**	214,502	61,338	490,342	1.88	90,374.67	–
m-geo*	1000	50	33,837	1–1	1–1	5.98	**0.33**	135,600	13,560	284,760	0.28	1856.88	4.17
m-langford-3-11	627	33	14,196	1–1	1–1	**48,399.07**	63,878.86	182,604	18,908	384,116	0.13	–	–
pb302095	600	50	165,300	1–98	1–129	–	–	173,884	17,400	365,168	0.15	**136,298.34**	–

The fifth column gives the interval of left hand side coefficients and the sixth column specifies the ranges of the right hand sides. For SLR, the number of additional inequalities is equal to $|P|$ (only inequalities (4)), and the number of non-zeros is thus equal to $3|P|$. For IL, the number non-zeros are displayed explicitly ("NZ+"). The columns labeled "GRB" display the wall clock time in seconds when passing the `.lp`-file directly to Gurobi (up to timeouts). The prefix `gp-` stands for `graphpart-` while "m" abbreviates "maxcsp", and the full name of m-geo* is maxcsp-geo50-20-d4–75-36 The fastest solution times are printed in bold

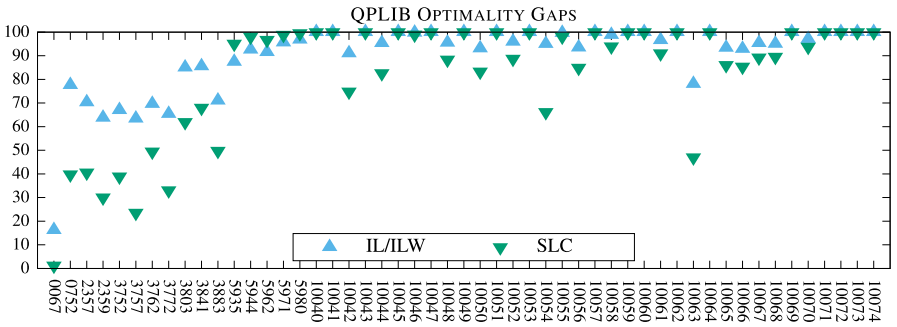

Fig. 6 Optimality gaps obtained with continuous relaxations (in percent) for the QPLIB instances with linear inequality constraints that were solved by using at least one of the linearizations (time limit 48 h)

proved difficult to solve with Gurobi. Moreover, the `color_lab` instances turn out to be difficult when using IL, even though the bound obtained with its continuous relaxation is much better than with SLC and also the model size is auspicious.

For the "semi-assignment"-like graph partitioning instances (see also a more comprehensive study in Mallach (2018) with an additional normalization constraint), we see a diverse picture as well: While the `graphpart-2` and `graphpart-3` instances prove rather simple for all methods (which also achieve the same relatively strong relaxation optimality gaps), the bounds obtained for the `graphpart-clique-` instances are equal to zero and they become especially challenging for IL with increasing size. In the latter instances, each node must be assigned to one of three partitions and the costs of placing any pair of nodes in the same partition are equal for all partitions. The initial number of products P as well as the computed inductive linearizations are significantly larger than for the other `graphpart` instances mentioned before. Additional experiments revealed that the bounds for most of the instances, in particular the `graphpart` and the `maxcsp` instances, could not be improved even when multiplying all constraints with all variables which here leads to a (square-reduced) first-level RLT since the "standard" linearization inequalities are then implied for these instances.

3.6 QPLIB

Finally, we look at the QPLIB (Furini et al. 2019), and we again chose the `.lp` format to pass the selected instances directly to Gurobi.

Instance Description

Here, we selected all BQPs with linear inequality constraints to be part of our testbed, especially since the MINLPLib experiments only involved instances with linear equation constraints, and since most instances of this kind in QPLIB are also part of the MINLPLib and QAPLIB experiments.

LP Relaxation Bounds and MIP Performance

For those selected instances that could be solved within 48 h using at least one method, Fig. 6 shows the relaxation optimality gaps. For the majority of the instances,

Table 8 MIP results for the (linear) inequality-constrained BQPs from the QPLIB solved by using at least one of the linearizations (time limit 48 h)

Instance	$\|N\|$	m_L	$\|P\|$	LHS	RHS	SLC	SLR			ILW				GRB
						Solve [s]	Ineq	NZ+	Solve [s]	$\|M\|$	NZ+	A1 [s]	Solve [s]	Solve [s]
0067	80	1	2844	2–50	1555–1555	146.33	5688	11,376	**103.97**	159	17,067	0.00	162.18	141.61
0752	250	1	3114	− 1–1	− 1–1	87,163.98	4686	10,914	–	499	19,113	0.02	–	**9450.56**
2357	240	2240	2254	− 1–1	0–1	1653.46	3418	7926	996.49	6687	22,661	1.23	5230.46	**502.98**
2359	306	3264	1740	− 1–1	0–1	169.64	2553	6033	**127.99**	4752	15,909	2.30	1544.24	529.78
3752	462	6160	3988	− 1–1	0–1	–	5836	13,812	–	11,666	39,601	6.32	–	**63,118.56**
3757	552	8096	1463	− 1–1	0–1	1004.99	2189	5115	874.74	3887	12,039	10.12	2955.59	**803.89**
3762	90	480	1133	− 1–1	0–1	47.19	1680	3946	213.76	3554	12,271	0.10	1387.61	**39.31**
3772	380	4560	2761	− 1–1	0–1	19,452.10	4117	9639	5789.64	7887	26,577	3.84	31,669.27	**3137.08**
3803	190	2280	2538	− 1–1	0–1	454.61	4250	9326	**294.35**	8833	28,905	1.00	52,979.55	405.58
3841	300	4600	4600	− 1–1	0–1	97,036.34	7603	16,803	–	15,367	51,290	3.03	–	**33,631.62**
3883	182	1456	2947	− 1–1	0–1	41,069.49	4324	10,218	**7924.69**	9068	31,406	0.58	–	19,768.73
5935	100	1237	4950	1–1	1–1	2860.17	8725	18,625	2365.87	7115	25,867	0.13	707.32	**259.42**
5944	100	2475	4950	1–1	1–1	131.00	8730	18,630	91.32	4757	17,301	0.21	75.36	**16.99**
5962	150	2793	11,175	1–1	1–1	70,407.89	19,705	42,055	94,666.80	16,447	59,671	0.41	20,114.83	**11,003.64**
5971	150	5587	11,175	1–1	1–1	3819.98	19,672	42,022	2741.02	10,974	39,902	0.65	585.06	**112.71**
5980	150	8381	11,175	1–1	1–1	157.74	19,649	41,999	109.84	5449	19,851	0.64	**40.34**	46.67
10040	125	6	7174	1–998	62–31,468	1315.91	10,822	25,170	771.56	247	43,045	0.01	–	**0.01**

Table 8 continued

Instance	$\|N\|$	m_L	$\|P\|$	LHS	RHS	SLC	SLR			ILW				GRB
						Solve [s]	Ineq	NZ+	Solve [s]	$\|M\|$	NZ+	A1 [s]	Solve [s]	Solve [s]
10041	125	6	7615	1–998	62–31,468	–	11,395	26,625	–	247	45,691	0.01	–	**0.30**
10042	125	5	7619	2–1000	47,135–50,209	66,995.42	11,420	26,658	30,723.40	247	45,715	0.01	–	**0.17**
10043	150	10	10,513	1–1000	54,036–58,799	499.84	15,703	36,729	154.32	297	63,121	0.01	–	**0.01**
10044	150	6	10,549	1–999	50–57,547	953.01	15,852	36,950	297.66	291	63,295	0.01	–	**2.55**
10045	150	10	10,091	1–1000	54,036–58,799	–	15,202	35,384	–	285	60,547	0.01	–	**0.02**
10046	150	6	9875	1–999	50–57,547	344.86	14,894	34,644	161.52	287	59,253	0.01	–	**0.16**
10047	150	10	11,168	1–1000	54,036–58,799	–	16,811	39,147	–	299	67,009	0.01	–	**0.01**
10048	150	5	11,018	2–1000	55,564–60,822	–	16,475	38,511	–	297	66,109	0.01	–	**138.17**
10049	150	10	11,018	1–1000	54,036–58,799	–	16,528	38,564	–	297	66,109	0.01	–	**0.01**
10050	150	5	10,878	2–1000	55,564–60,822	–	16,287	38,043	–	295	65,269	0.01	–	**0.63**
10051	150	10	11,015	1–1000	54,036–58,799	–	16,438	38,468	–	297	66,091	0.01	–	**4.76**
10052	150	6	11,165	1–999	50–57,547	–	16,798	39,128	–	299	66,991	0.01	–	**3.26**
10053	150	10	11,019	1–1000	54,036–58,799	–	16,634	38,672	–	297	66,115	0.01	–	**0.01**
10054	175	11	13,398	1–1000	116–22,886	516.18	20,162	46,958	128.34	339	80,389	0.01	–	**43.57**
10055	175	5	13,918	2–1000	66,009–70,533	492.83	20,924	48,760	177.06	347	83,511	0.01	–	**1.47**
10056	175	5	14,535	2–1000	66,009–70,533	–	21,912	50,982	–	341	87,211	0.01	–	**0.23**
10057	200	11	15,056	1–1000	66–77,923	495.82	22,601	52,713	93.97	371	90,347	0.02	–	**0.01**
10058	200	11	17,491	1–1000	100–54,119	907.07	26,231	61,213	**219.88**	395	105,337	0.02	–	264.54
10059	200	10	14,135	1–1000	45,545–53,253	–	21,165	49,435	–	341	84,811	0.02	–	**0.02**
10060	200	10	17,474	1–1000	45,545–53,253	798.39	26,114	61,062	390.81	389	104,896	0.02	–	**0.11**

Table 8 continued

Instance	$\|N\|$	m_L	$\|P\|$	LHS	RHS	SLC	SLR			ILW				GRB
						Solve [s]	Ineq	NZ+	Solve [s]	$\|M\|$	NZ+	A1 [s]	Solve [s]	Solve [s]
10061	200	5	17,892	2–1000	73,590–80,431	–	26,912	62,696	–	381	107,353	0.01	–	**0.97**
10062	200	10	18,162	1–1000	45,545–53,253	–	27,215	63,539	–	383	108,973	0.02	–	**0.06**
10063	200	5	19,413	2–1000	73,590–80,431	5895.00	29,189	68,015	2174.58	395	116,479	0.01	–	**0.09**
10064	200	11	19,272	1–1000	66–77,923	–	28,885	67,429	–	393	115,633	0.02	–	**0.02**
10065	200	11	19,302	1–1000	100–50,591	–	28,925	67,529	–	393	115,813	0.02	–	**2.19**
10066	200	11	19,301	1–1000	100–54,119	–	28,747	67,349	–	393	115,807	0.02	–	**0.45**
10067	200	5	19,642	2–1000	73,590–80,431	–	29,462	68,746	–	397	117,853	0.02	–	**0.39**
10068	200	11	19,879	1–1000	100–54,119	–	29,898	69,656	–	399	119,275	0.02	–	**1.26**
10069	200	10	19,264	1–1000	45,545–53,253	1042.20	28,787	67,315	380.48	405	117,012	0.02	–	**0.01**
10070	200	11	19,478	1–1000	100–54,119	–	29,215	68,171	–	395	116,869	0.02	–	**32.63**
10071	200	11	19,698	1–1000	66–77,923	–	29,552	68,948	–	399	118,189	0.02	–	**0.03**
10072	75	10	2049	1–1000	23,837–30,414	5842.82	3104	7202	2566.77	133	12,295	0.01	–	**0.00**
10073	75	6	1928	1–999	37–10,480	8692.58	2889	6745	2172.87	133	11,697	0.01	–	**0.00**
10074	75	10	2774	1–1000	23,837–30,414	10,672.72	4141	9689	5733.37	149	16,645	0.01	–	**0.00**

The fifth column gives the interval of left hand side coefficients and the sixth column specifies the ranges of the right hand sides. For SLR, the varying exact number of additional inequalities ("Ineq") and non-zeros ("NZ+") are displayed explicitly. The same is true for ILW. The columns labeled "GRB" display the wall clock time in seconds when passing the `.lp`-file directly to Gurobi

The fastest solution times are printed in bold

these gaps are large irrespective of the method used. Zero lower bounds (100% gaps) are frequently observed for minimization problems like e.g. most instances with prefix `100` that have large ratios between the right hand sides and the left hand side coefficients. For a few of these with slightly better gaps, the SLC bound turns out to be superior. Also for instance `0067` (actually, a QKP), the ILW gap of 16.25% is weaker than the 1.26% gap obtained with SLC which is in line with Sect. 3.2. Another example is instance `0752` where the ILW gap of 77.60% is considerably worse than the 39.83% gap obtained with SLC. Due to negative coefficients this instance requires normalization, and so do the instances with first digits `2` and `3`. Here, a similar effect as in Sect. 3.4 concerning poorly linked linearization variables referring to the same pair of original variables applies. Only for the instances with first prefix `5`, the ILW bounds are slightly superior even though on a weak level.

The MIP results are presented in Table 8. After the above discussion regarding the relaxation bounds and the impact of normalization, it could be expected that better results are obtained when using Gurobi as a standalone solver or sometimes also when supplying it with one of the "standard" linearization than when using ILW (which still worked better than IL). Indeed, while the instances with prefix `100` and their special coefficient ranges turn out to be unsuited for an inductive linearization, also the size of the linearizations for the instances starting with the digits `2` and `3` turns out to be unfavorable. Only for the instances with first digit `5`, competitive model sizes and solution times are achieved with IL.

4 Conclusion

A framework to derive inductive linearizations in practice has been outlined and it has been demonstrated that it can be effectively applied to a variety of binary quadratic programs with linear constraints. The experiments covered the Quadratic Assignment, Knapsack, Matching and Shortest Path Problems, as well as instances from the MINLPLib and QPLIB.

Addressing the research questions mentioned in the introduction, the best results are typically obtained with an inductive linearization if its continuous relaxation is comparably tight and the number of linearization constraints and induced non-zero coefficients keep moderate at the same time. These ideal conditions are for example observed for Quadratic Assignment and Quadratic Matching Problems. If only the first property is fulfilled, for instance because the original constraints employed have right hand sides equal to one and zero–one left hand sides but the structure of the factor pairs or constraints impedes a compact model extension, or if the formulation is more compact without providing a strong relaxation, the results show that an inductive linearization may or may not compare well with a "standard" linearization. Whether one or the other case applies may then depend on further parameters such as the density of the given product terms and the actual distribution of their factors over the set of (employed) constraints, which in turn again also influence the achievable compactness and relaxation strength of inductive linearizations. Moreover, if the ratios between right hand sides and (non-negative) left hand side coefficients are large, the relaxations of inductive linearizations tend to become weaker, and "standard" linearizations may

lead to superior running times with a mixed-integer programming solver as has been observed with the Quadratic Knapsack Problem. Until now, negative constraint coefficients may also impair the performance sustained with inductive linearizations. One reason is that the corresponding normalization step typically counteracts a compact reformulation, another is that often a poor linkage between linearization variables referring to the same pair of original variables is established by the induced constraints. Improvements in this respect are a natural subject for further research. The explanations for positive and negative observations at hand, it should be emphasized that they can only provide a partial image while exceptions from the respective rules of thumb exist naturally, especially since a MIP solution process depends on various further influences such as e.g. branching decisions or cutting place effectiveness. Across the instance sets considered, the inductive linearization technique frequently led to a continuous relaxation that is at least as tight as the one provided by a "standard" linearization. In several cases, the bounds also turned out to be equal or almost as good as those provided by a full first-level RLT even though being typically much more compact. The broad computational experiments further revealed that inductive linearizations can be computed quickly in an automated fashion for the various different application instances, and that this can typically be done well even with a simple combinatorial heuristic.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Conflict of interest The author declares that there is no conflict of interest.

References

Adams WP, Johnson TA (1994) Improved linear programming-based lower bounds for the quadratic assignment problem. In: Pardalos PM, Wolkowicz H (eds) Quadratic assignment and related problems, vol 16. DIMACS Series on Discrete Mathematics and Computer Science. AMS, Providence, pp 43–75

Adams WP, Sherali HD (1986) A tight linearization and an algorithm for zero-one quadratic programming problems. Manag Sci 32(10):1274–1290. https://doi.org/10.1287/mnsc.32.10.1274

Adams WP, Sherali HD (1999) A reformulation-linearization technique for solving discrete and continuous nonconvex problems. In: Nonconvex optimization and its applications, vol 31. Springer. https://doi.org/10.1007/978-1-4757-4388-3

Billionnet A, Calmels F (1996) Linear programming for the 0–1 quadratic knapsack problem. Eur J Oper Res 92(2):310–325. https://doi.org/10.1016/0377-2217(94)00229-0

Billionnet A, Elloumi S (2001) Best reduction of the quadratic semi-assignment problem. Discrete Appl Math 109(3):197–213. https://doi.org/10.1016/S0166-218X(00)00257-2

Billionnet A, Soutif E (2004) An exact method based on Lagrangian decomposition for the 0–1 quadratic knapsack problem. Eur J Oper Res 157(3):565–575. https://doi.org/10.1016/S0377-2217(03)00244-3

Burkard RE, Karisch SE, Rendl F (1997) QAPLIB—a quadratic assignment problem library. J Glob Optim 10(4):391–403. https://doi.org/10.1023/A:1008293323270

Cordeau JF, Gaudioso M, Laporte G, Moccia L (2006) A memetic heuristic for the generalized quadratic assignment problem. INFORMS J Comput 18(4):433–443. https://doi.org/10.1287/ijoc.1040.0128

Davidović T, Liberti L, Maculan N, Mladenović N (2007) Towards the optimal solution of the multiprocessor scheduling problem with communication delays. In: Baptiste P, Kendall G, Munier-Kordon A, Sourd F (eds) Proceedings of the 3rd multidisciplinary international conference on scheduling: theory and applications, École Polytechnique, Paris, pp 128–135

Frieze AM, Yadegar J (1983) On the quadratic assignment problem. Discrete Appl Math 5(1):89–98. https://doi.org/10.1016/0166-218X(83)90018-5

Furini F, Traversi E, Belotti P, Frangioni A, Gleixner A, Gould N, Liberti L, Lodi A, Misener R, Mittelmann H, Sahinidis N, Vigerske S, Wiegele A (2019) QPLIB: a library of quadratic programming instances. Math Program Comput 11:237–265. https://doi.org/10.1007/s12532-018-0147-4

Glover F, Woolsey E (1974) Converting the 0–1 polynomial programming problem to a 0–1 linear program. Oper Res 22(1):180–182. https://doi.org/10.1287/opre.22.1.180

Hammer PL, Hansen P, Simeone B (1984) Roof duality, complementation and persistency in quadratic 0–1 optimization. Math Program 28(2):121–155. https://doi.org/10.1007/BF02612354

Helmberg C, Rendl F, Weismantel R (2000) A semidefinite programming approach to the quadratic knapsack problem. J Comb Optim 4(2):197–215. https://doi.org/10.1023/A:1009898604624

Hupp L, Klein L, Liers F (2015) An exact solution method for quadratic matching: the one-quadratic-term technique and generalisations. Discrete Optim 18:193–216. https://doi.org/10.1016/j.disopt.2015.10.002

Koopmans TC, Beckmann M (1957) Assignment problems and the location of economic activities. Econometrica 25(1):53–76

Liberti L (2007) Compact linearization for binary quadratic problems. 4OR 5(3):231–245. https://doi.org/10.1007/s10288-006-0015-3

Mallach S (2017) Improved mixed-integer programming models for the multiprocessor scheduling problem with communication delays. J Comb Optim. https://doi.org/10.1007/s10878-017-0199-9

Mallach S (2018) Compact linearization for binary quadratic problems subject to assignment constraints. 4OR 16:295–309. https://doi.org/10.1007/s10288-017-0364-0

Mallach S (2019) A natural quadratic approach to the generalized graph layering problem. In: Archambault D, Tóth CD (eds) Graph drawing and network visualization. Springer, Cham, pp 532–544. https://doi.org/10.1007/978-3-030-35802-0_40

Mallach S (2021) Inductive linearization for binary quadratic programs with linear constraints. 4OR 19(4):549–570. https://doi.org/10.1007/s10288-020-00460-z

Pessoa AA, Hahn PM, Guignard M, Zhu YR (2010) Algorithms for the generalized quadratic assignment problem combining Lagrangean decomposition and the reformulation-linearization technique. Eur J Oper Res 206(1):54–63. https://doi.org/10.1016/j.ejor.2010.02.006

Rostami B, Chassein A, Hopf M, Frey D, Buchheim C, Malucelli F, Goerigk M (2018) The quadratic shortest path problem: complexity, approximability, and solution methods. Eur J Oper Res 268(2):473–485. https://doi.org/10.1016/j.ejor.2018.01.054