European Journal of Operational Research 220 (2012) 314–319

Contents lists available at SciVerse ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

A new exact discrete linear reformulation of the quadratic assignment problem

Axel Nyberg *, Tapio Westerlund

Process Design and Systems Engineering, Åbo Akademi University, FIN-20500 Åbo, Finland

ARTICLE INFO

Article history:
Received 11 March 2011
Accepted 7 February 2012
Available online 16 February 2012

Keywords:
Combinatorial optimization
Quadratic assignment problem
Discrete linear reformulation
Mixed integer programming
Global Optimization

ABSTRACT

The quadratic assignment problem (QAP) is a challenging combinatorial problem. The problem is NP-hard and in addition, it is considered practically intractable to solve large QAP instances, to proven optimality, within reasonable time limits. In this paper we present an attractive mixed integer linear programming (MILP) formulation of the QAP. We first introduce a useful non-linear formulation of the problem and then a method of how to reformulate it to a new exact, compact discrete linear model. This reformulation is efficient for QAP instances with few unique elements in the flow or distance matrices. Finally, we present optimal results, obtained with the discrete linear reformulation, for some previously unsolved instances (with the size $n = 32$ and 64), from the quadratic assignment problem library, QAPLIB.

1. Introduction

The quadratic assignment problem was introduced by Koopmans and Beckmann (1957) in the basic form:

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{l=1}^{n} a_{ij}b_{kl}\cdot x_{ik}x_{jl} \tag{1}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1,\ldots,n; \tag{2}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad i = 1,\ldots,n; \tag{3}$$

$$x_{ij} \in \{0,1\}, \quad i,j = 1,\ldots,n; \tag{4}$$

a_{ij} are given distances between locations and b_{kl} flows between facilities defined in the matrices, **A** and **B**. The QAP is an NP-hard problem and many different formulations and methods for this problem have been suggested. These include heuristics (Taillard, 1991), linear reformulation techniques (Adams et al., 2007) and convex quadratic programming (Anstreicher and Brixius, 2001) to name a few. Computing lower bounds for QAPs using SDP relaxations has recently also been widely studied (Peng et al., 2010). In fact over one hundred papers have been published in the subject since 1999 according to Loiola et al. (2007). However, to this date only a few QAPs of size $n \geqslant 30$ from QAPLIB (Burkard et al., 1997; Hahn and Anjos, 2002) have been solved to proven optimality including the famous nug30 and the kra instances (Anstreicher et al., 2002). Most of these instances have been solved using computers connected in parallel and the CPU times required to solve the models, calculated for a single computer, are immense. It should however be mentioned that in addition to the instances in the QAPLIB Drezner et al. (2005) reported solutions of some special classes of QAPs as big as $n = 75$.[1]

This paper presents a compact exact discrete linear reformulation (DLR) of the QAP that can be solved using general MILP solvers. This formulation gives promising results on instances where one of the matrices has few unique elements per row. Various different MILP formulations have been presented through the years. The model with the smallest amount of variables and constraints, is to our knowledge, the model presented by Kaufman and Broeckx (1978). Their model contains n^2 binary variables, n^2 real variables and $2n + n^2$ constraints. The formulation gives, however, very poor lower bounds and the tightening of the formulation has recently been studied by Zhang et al. (2010). The MILP model in this paper contains $\alpha \cdot n^2$ binary variables, $\beta \cdot n^2$ real variables and $\gamma \cdot n + \delta \cdot n^2$ constraints, where α, β, γ and δ are problem specific parameters. In the worst case, our formulation has n^2 binary variables, n^3 real variables and $2n + n^3$ constraints. However, this is almost never the case. Values for these parameters for some instances in the QAPLIB are found in Sections 3–4. With respect to the notations, vectors and matrices are written in bold face throughout the paper.

* Corresponding author. Tel.: +358 407783015.
E-mail addresses: axnyberg@abo.fi (A. Nyberg), twesterl@abo.fi (T. Westerlund).

[1] In addition, after the submission of this paper, Fischetti, Monaci and Salvagin reported the solution of the instance esc128 in QAPLIB with $n = 128$.

0377-2217/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.ejor.2012.02.010

2. QAP-reformulations

Consider two permutation vectors $\mathbf{p}$ and $\tilde{\mathbf{p}}$ where $p_i = k$ if facility i is at location k and $\tilde{p}_i = k$ if facility k is at location i. The objective function in Eq. (1), can then be written as:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_{p_i p_j} b_{ij} = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij} b_{\tilde{p}_i \tilde{p}_j}. \tag{5}$$

In matrix form Eqs. (2) and (3) can be written as $\mathbf{Xe} = \mathbf{X}^T\mathbf{e} = \mathbf{e}$ where $\mathbf{e}$ is a vector with all elements equal to 1 and $\mathbf{X}$ is the so-called permutation matrix containing the binary variables, x_{ij}. The permutation vectors $\mathbf{p}$ and $\tilde{\mathbf{p}}$ are given by $\mathbf{p} = \mathbf{Xq}$ and $\tilde{\mathbf{p}} = \mathbf{X}^T\mathbf{q}$ where $\mathbf{q}^T = (1,2,\ldots,n)$. Using the properties of the permutation vectors, Eq. (5) can be rewritten in the form:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} a_{pij} b_{i\tilde{p}_j} = \sum_{i=1}^{n}\sum_{j=1}^{n} a'_{ij} b'_{ij}, \tag{6}$$

where a'_{ij} and b'_{ij} are given by:

$$a'_{ij} = \sum_{k=1}^{n} a_{kj} x_{ik} \quad \forall\, i,j, \tag{7}$$

$$b'_{ij} = \sum_{k=1}^{n} b_{ik} x_{kj} \quad \forall\, i,j. \tag{8}$$

The row sum and the column sum of the binary variables x are equal to 1 according to Eqs. (2) and (3).

Using more compact matrix notation, Eqs. (6)–(8) can also be written as $\mathbf{XA}\bullet\mathbf{BX}$ where $\bullet$stands for the scalar product of the matrices, defined as the sum of the product of the corresponding elements. This representation is considered later on in Section 3. From Eq. (6) we find that the objective function is a sum of bilinear terms, $a'_{ij}b'_{ij}$, of discrete variables, where each variable can obtain, at most, n different numerical values, defined by the elements of corresponding columns and rows of the matrices $\mathbf{A}$ and $\mathbf{B}$ respectively. We can formulate a linear relaxation, w_{ij}, of the bilinear term $a'_{ij}b'_{ij}$ by considering the discrete nature of the variable b'_{ij} as follows:

$$w_{ij} \geqslant \sum_{m=1}^{M_i} B_i^m z_{ij}^m \tag{9}$$

$$\sum_{m=1}^{M_i} z_{ij}^m = a'_{ij} \tag{10}$$

$$z_{ij}^m \leqslant \overline{A}_j \sum_{k\in K_i^m} x_{kj} \quad \forall\, m = 1,\ldots,M_i \tag{11}$$

B_i^m are the constant unique values of the elements b_{ij} in row i of the matrix $\mathbf{B}$ and M_i the number of unique elements in the row, where $M_i \leqslant n$. z_{ij}^m are nonnegative variables of which one will be equal to a'_{ij} while the others will be zero. $\overline{A}_j = \max_i a_{ij}$ (i.e. the largest element in column j of the matrix $\mathbf{A}$). The index sets K_i^m are connected to the unique values of the elements (with corresponding binary variables) of row i in the $\mathbf{B}$ matrix and are defined as follows:

$$K_i^m = \{j | b_{ij} = B_i^m\} \quad \forall\, j \quad \wedge m = 1,\ldots,M_i. \tag{12}$$

Since both a'_{ij} and b'_{ij} are nonnegative variables and the objective is to minimize a sum of such bilinear terms we will get an exact discrete linear relaxed reformulation of the QAP by including the constraints in Eqs. (10) and (11) and by replacing the bilinear terms $a'_{ij}b'_{ij}$ in the objective function with the corresponding variables w_{ij}. The complete formulation is given in the next section.

2.1. A simple example

To conclude this section we will illustrate the exact relaxation of one bilinear term as given in Eqs. (9)–(12). We consider the following QAP with $n = 5$ where:

$$\mathbf{A} = \begin{bmatrix} 0 & 3 & 5 & 9 & 6 \\ 3 & 0 & 2 & 6 & 9 \\ 5 & 2 & 0 & 8 & 10 \\ 9 & 6 & 8 & 0 & 2 \\ 6 & 9 & 10 & 2 & 0 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} 0 & 4 & 3 & 7 & 7 \\ 4 & 0 & 4 & 10 & 4 \\ 3 & 4 & 0 & 2 & 3 \\ 7 & 10 & 2 & 0 & 4 \\ 7 & 4 & 3 & 4 & 0 \end{bmatrix}$$

Now, as an illustrative example, consider the relaxation of the bilinear term $a'_{ij}b'_{ij}$, with $i = 2$ and $j = 3$, i.e. $a'_{23}b'_{23}$. According to Eqs. (7) and (8) a'_{23} and b'_{23} are given by:

$$a'_{23} = 5x_{21} + 2x_{22} + 0x_{23} + 8x_{24} + 10x_{25},$$
$$b'_{23} = 4x_{13} + 0x_{23} + 4x_{33} + 10x_{43} + 4x_{53},$$

where according to Eqs. (2) and (3):

$$x_{13} + x_{23} + x_{33} + x_{43} + x_{53} = 1,$$
$$x_{21} + x_{22} + x_{23} + x_{24} + x_{25} = 1.$$

Since row two ($i = 2$) in the matrix $\mathbf{B}$ has three unique values, namely 0, 4 and 10 and Eq. (2) holds, (i.e. $x_{13} + x_{23} + x_{33} + x_{43} + x_{53} = 1$), we get $b'_{23} \in \{0,4,10\} = \left\{B_2^1, B_2^2, B_2^3\right\}$ and $M_2 = 3$. Thus according to Eq. (9):

$$w_{23} \geqslant 0z_{23}^1 + 4z_{23}^2 + 10z_{23}^3.$$

Furthermore, according to Eq. (10):

$$z_{23}^1 + z_{23}^2 + z_{23}^3 = 5x_{21} + 2x_{22} + 0x_{23} + 8x_{24} + 10x_{25},$$

where the RHS is equal to a'_{23}. The index sets K_2^m (connected to the unique values of the elements of row two in the $\mathbf{B}$ matrix) are, according to Eq. (12), in this case $K_2^1 = \{2\}$; $K_2^2 = \{1,3,5\}$and$K_2^3 = \{4\}$. Furthermore, $\overline{A}_3 = 10$ (i.e. the largest value in column three of the $\mathbf{A}$ matrix). From Eq. (11) we, thus, obtain:

$$z_{23}^1 \leqslant 10x_{23},$$
$$z_{23}^2 \leqslant 10(x_{13} + x_{33} + x_{53}),$$
$$z_{23}^3 \leqslant 10x_{43}.$$

Since only one of the binary variables above will be equal to 1 and the others equal to 0 only one of the z_{23}^m variables can be nonzero and, thus, according to Eq. (10) equal to a'_{23}. Furthermore, observe that the binary variables above are connected to the parameters B_2^m, i.e. the unique values of b'_{23}.

As we minimize w_{23}, the inequality as in Eq. (9) will be active and we obtain:

$$w_{23} = 0z_{23}^1 + 4z_{23}^2 + 10z_{23}^3,$$

which result in an exact discrete linear relaxation of the bilinear term. In a similar way bilinear terms for all i, j in the QAP can be handled. Since, in a QAP, both a'_{ij} and b'_{ij} are discrete variables, we can do the discretization in either a'_{ij} or b'_{ij}. In Fig. 1 both discretizations are illustrated. The left figure correspond to the example above. From the figures we observe that the representation is a relaxation of the bilinear term for continuous values of the variables a'_{ij} and b'_{ij} but an exact reformulation of it if one of the variables is a discrete variable. A similar type of relaxation for continuous variables in pooling problems, using piecewise linear expressions, is given in Gounaris et al. (2009).

3. The discrete linear QAP reformulation

Now we reformulate every bilinear term in the QAP using the method described in Section 2. Thus we get an exact discrete linear reformulation (DLR) of the QAP. The discretization can be done in either a'_{ij} or b'_{ij}, as mentioned before. In the below reformulation, the discretization is done in the variables b'_{ij}.

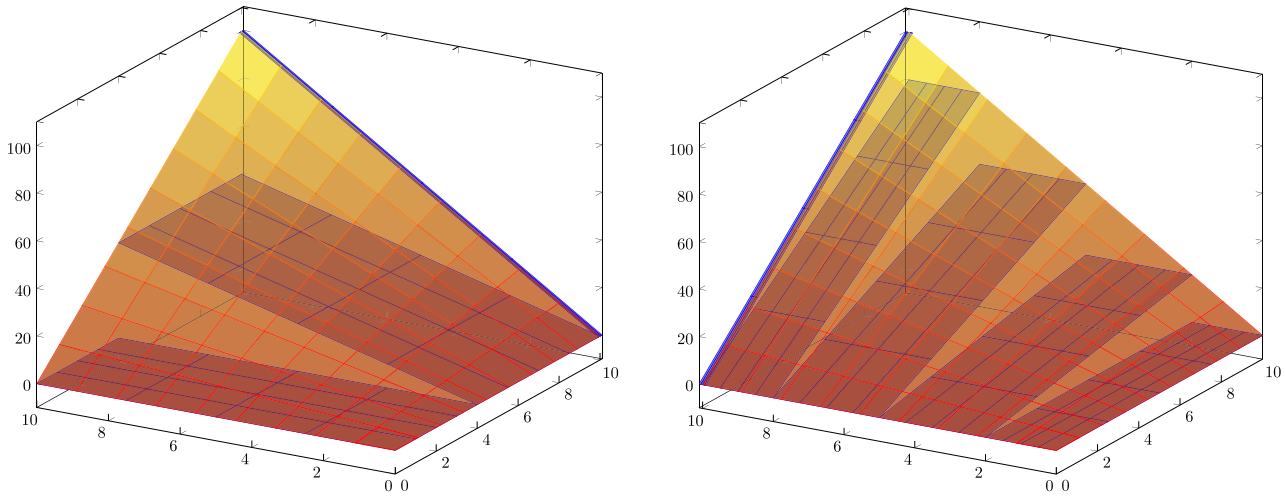

Fig. 1. Bilinear term $a'_{23}b'_{23}$ discretized in b'_{23} (to the left) and in a'_{23} (to the right).

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{m=1}^{M_i} B_i^m z_{ij}^m \tag{13}$$

subject to

$$\sum_{i=1}^{n} x_{ij} = 1 \quad \forall\, j, \tag{14}$$

$$\sum_{j=1}^{n} x_{ij} = 1 \quad \forall\, i, \tag{15}$$

$$\left.\begin{aligned} z_{ij}^m &\leqslant \overline{A}_j \sum_{k\in \mathbf{K}_i^m} x_{kj} \quad m = 1,\ldots,M_i, \\ \sum_{m=1}^{M_i} z_{ij}^m &= \sum_{k=1}^{n} a_{kj}x_{ik} \end{aligned}\right\} \quad \forall\, i,j, \tag{16}$$

$$x_{ij} \in \{0,1\} \quad z_{ij}^m \in [0,\overline{A}_j] \quad \forall\, i,j \wedge m = 1,\ldots,M_i, \tag{17}$$

where

$$\overline{A}_j = \max_i a_{ij} \quad \forall\, j, \tag{18}$$

$$\mathbf{K}_i^m = \{j | b_{ij} = B_i^m\} \quad \forall\, i,j \wedge m = 1,\ldots,M_i, \tag{19}$$

$$b_{ij} \in \left\{B_i^1, B_i^2, \ldots, B_i^{M_i}\right\} \quad \forall\, i,j. \tag{20}$$

The values of a_{ij} and b_{ij} are defined in the matrices **A** and **B**. The amount of variables and constraints in the model is dependent on the number of unique values in the rows or columns of the matrices (depending on which one is discretized). The number of binary variables is $\alpha \cdot n^2$ and the number of real variables $\beta \cdot n^2$ while the number of constraints is $\gamma \cdot n + \delta \cdot n^2$. In Table 1 the values of α, β, γ and δ are given for certain instances from QAPLIB when reformulated in this manner. In Table 1 the solution times when solving the instances to optimality in the different discretized forms are also given. As can be noted, fewer constraints and variables do not always result in shorter CPU times. Furthermore, it can be observed that the two discretization directions will result in highly different solution times. When the problem size increases, the choice of the direction will have a crucial importance on whether the problem can be solved or not. This can already be seen at the root node, since the value of the lower bound differs greatly depending on which way the model has been discretized.

Remark. The two optional discretization directions can algorithmically be implemented in different ways. In case the implementation is simply done by switching the matrices in the objective function, **XA•BX**, it should, however, be noted that the permutation matrix **X** must then be considered as its transpose as well (and the permutation vector p would be $\tilde{p}$) since **XA•BX = BX•XA = X^{T}BX•A = X^{T}B•AX^T**.

Table 1
Parameters α, β, γ and δ as well as solution times in seconds, when discretizing in A or B

Instance	in A					in B				
	α	β	γ	δ	time (s)	α	β	γ	δ	time (s)
chr12a	1	2.83	2	3.83	10	1	11.33	2	12.33	2
esc16a	1	3.12	2	2.75	33	1	2.5	2	3.12	11
nug12	1	6.17	2	5.17	59	1	5.33	2	6.33	1186

From the formulation in Eqs. (13)–(20) it may, further, be observed that if $\overline{A}_j = 0$ then the corresponding variables, $z_{ij}^m \;\forall\, i,m$ will be equal to zero. This reduces the number of variables, constraints in Eq. (16) and corresponding terms in the objective function Eq. (13). On the other hand if $B_i^m = 0$ it only reduces corresponding terms in the objective function, but does not affect the number of variables and constraints.

4. Special structures of the QAP

Many of the instances in the QAPLIB (e.g. esc32a-esc32h) have matrices where all elements in a row and the corresponding column are equal to zero. It is important to consider such special structures of QAP since they result in multiple equal solutions. We therefore, introduce a way to exclude such solutions in the DLR model. By first considering the **B** matrix and partitioning the matrices in the objective function (Eq. (6), written in matrix form) we obtain,

$$\mathrm{XA}\bullet\mathrm{BX} = \left[\begin{array}{c|c} \mathbf{X}_1 & \mathbf{X}_2 \\ \hline \mathbf{X}_3 & \mathbf{X}_4 \end{array}\right] \left[\begin{array}{c|c} \mathbf{A}_{B1} & \mathbf{A}_{B2} \\ \hline \mathbf{A}_{B3} & \mathbf{A}_{B4} \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{B}_1 & \mathbf{B}_2 \\ \hline \mathbf{B}_3 & \mathbf{B}_4 \end{array}\right] \left[\begin{array}{c|c} \mathbf{X}_1 & \mathbf{X}_2 \\ \hline \mathbf{X}_3 & \mathbf{X}_4 \end{array}\right] =$$

$$\left[\begin{array}{c|c} \mathbf{X}_1\mathbf{A}_{B1} + \mathbf{X}_2\mathbf{A}_{B3} & \mathbf{X}_1\mathbf{A}_{B2} + \mathbf{X}_2\mathbf{A}_{B4} \\ \hline \mathbf{X}_3\mathbf{A}_{B1} + \mathbf{X}_4\mathbf{A}_{B3} & \mathbf{X}_3\mathbf{A}_{B2} + \mathbf{X}_4\mathbf{A}_{B4} \end{array}\right] \bullet \left[\begin{array}{c|c} \mathbf{B}_1\mathbf{X}_1 + \mathbf{B}_2\mathbf{X}_3 & \mathbf{B}_1\mathbf{X}_2 + \mathbf{B}_2\mathbf{X}_4 \\ \hline \mathbf{B}_3\mathbf{X}_1 + \mathbf{B}_4\mathbf{X}_3 & \mathbf{B}_3\mathbf{X}_2 + \mathbf{B}_4\mathbf{X}_4 \end{array}\right]$$

When $\mathbf{B}_2$, $\mathbf{B}_3$, $\mathbf{B}_4 = \mathbf{0}$ we obtain:

$$\mathbf{XA}\bullet\mathbf{BX} = \left[\begin{array}{c|c} \mathbf{X}_1 & \mathbf{X}_2 \end{array}\right] \left[\begin{array}{c|c} \mathbf{A}_{B1} & \mathbf{A}_{B2} \\ \hline \mathbf{A}_{B3} & \mathbf{A}_{B4} \end{array}\right] \bullet \mathbf{B}_1 \left[\begin{array}{c|c} \mathbf{X}_1 & \mathbf{X}_2 \end{array}\right]$$

$$= \left[\begin{array}{c|c} \mathbf{X}_1 & \mathbf{X}_2 \end{array}\right] \mathbf{A} \bullet \mathbf{B}_1 \left[\begin{array}{c|c} \mathbf{X}_1 & \mathbf{X}_2 \end{array}\right]$$

$$= \mathbf{X}_B\mathbf{A}\bullet\mathbf{B}_1\mathbf{X}_B$$

If the number of remaining rows and columns in the B matrix is n_B then $n - n_B$ rows of the permutation matrix have been removed. The original QAP contains $n!$ permutations, but since $(n - n_B)$ rows of the permutation matrix have been eliminated, $(n - n_B)!$ permutations have as well. The number of permutations in the reduced problem is thus $n!/(n - n_B)!$. In Table 2 and 3 examples of the influence of the compression is illustrated by some instances from QAPLIB.

As we apply this method to certain instances in QAPLIB we can decrease the number of binary variables and constraints. The difference in the formulation after the removal of the unnecessary binary variables is that the index i should be summed up to only n_B and Eq. (14) should now be written as an inequality constraint,

$$\sum_{i=1}^{n_B} x_{ij} \leqslant 1, \quad j = 1, \ldots, n; \tag{21}$$

However, in Eq. (21) all binary variables may be equal to zero and forcing all other variables in Eq. (16) to be equal to zero as well. Then the equality constraint in Eq. (16) would not hold. Therefore, an additional variable, z_{ij}^0, with a corresponding relaxation constraint must be added. This constraint is given by:

$$z_{ij}^0 \leqslant \overline{A}_j \left(1 - \sum_{k=1}^{n_B} x_{kj}\right) \quad \forall\, i,j. \tag{22}$$

The variable, z_{ij}^0, is then added to the equality constraint in Eq. (16) by starting the summation in the LHS from $m = 0$. Also, in a similar way, rows and corresponding columns with all elements zero in **A** can be utilized to reduce the number of columns in the permutation matrix. In the latter case the index j should be summed up to only n_A and Eq. (15) must be written as an inequality constraint:

$$\sum_{j=1}^{n_A} x_{ij} \leqslant 1, \quad i = 1, \ldots, n; \tag{23}$$

where n_A is the number of remaining rows and columns in the A matrix. No additional relaxation constraints connected to the inequality constraint Eq. (23) need be added in this case, as long as we have discretized in B. On the other hand, if we discretize in A an additional variable and a corresponding relaxation constraint should be connected to Eq. (23) as well.

Additional remarks. If both the matrices **A** and **B** have rows (and columns) with all elements equal to zero we obtain:

$$\mathbf{XA} \bullet \mathbf{BX} = \mathbf{X}_B\mathbf{A} \bullet \mathbf{B}_1\mathbf{X}_B = \mathbf{X}_A\mathbf{A}_1 \bullet \mathbf{BX}_A = \widetilde{\mathbf{X}}\mathbf{A}_1 \bullet \mathbf{B}_1\widetilde{\mathbf{X}},$$

where

$$\mathbf{X} = \left[\begin{array}{c} \mathbf{X}_B \\ \hline - \end{array}\right] = \left[\begin{array}{c|c} \mathbf{X}_A & - \end{array}\right] = \left[\begin{array}{c|c} \widetilde{\mathbf{X}} & - \\ \hline - & - \end{array}\right]$$

8

Table 2
Variables, $(\alpha + \beta)n^2$, and constraints, $\gamma\, n + \delta n^2$, in the model discretized in A with and without compression.

Instance	No compression				Compressed				$(n - n_B)!$
	α	β	γ	δ	α	β	γ	δ	
esc32a	1	2.94	2	3.72	0.78	2.72	1.8	3.49	7!
esc32b	1	2.50	2	3.25	0.75	2.25	1.8	2.99	8!
esc32c	1	2.72	2	3.31	0.59	2.31	1.6	2.89	13!
esc32d	1	2.06	2	2.62	0.56	1.62	1.6	2.17	14!
esc32e	1	1.34	2	1.62	0.28	0.62	1.3	0.88	23!
esc32g	1	1.41	2	1.62	0.22	0.62	1.2	0.82	25!
esc32h	1	2.62	2	3.22	0.59	2.22	1.6	2.80	13!
esc64a	1	1.39	2	1.73	0.34	0.73	1.3	1.07	42!
esc128	1	1.26	2	1.50	0.24	0.50	1.2	0.74	97!
tai64c	1	1.20	2	1.41	0.20	0.41	1.2	0.60	51!
tai256c	1	1.36	2	1.72	0.36	0.72	1.4	1.08	164!

Table 3
Variables, $(\alpha + \beta)n^2$, and constraints, $\gamma\, n + \delta n^2$, in the model discretized in B with and without compression.

Instance	No compression				Compressed				$(n - n_B)!$
	α	β	γ	δ	α	β	γ	δ	
esc32a	1	3.91	2	4.69	0.78	3.91	1.8	4.68	7!
esc32b	1	3.75	2	4.50	0.75	3.75	1.8	4.49	8!
esc32c	1	2.97	2	3.56	0.59	2.97	1.6	3.55	13!
esc32d	1	2.81	2	3.38	0.56	2.81	1.6	3.36	14!
esc32e	1	1.41	2	1.69	0.28	1.41	1.3	1.67	23!
esc32g	1	1.09	2	1.31	0.22	1.09	1.2	1.29	25!
esc32h	1	2.97	2	3.56	0.59	2.97	1.6	3.55	13!
esc64a	1	2.06	2	2.41	0.34	2.06	1.3	2.40	42!
esc128	1	1.70	2	1.94	0.24	1.70	1.2	1.93	97!
tai64c	1	3.05	2	3.25	0.20	3.05	1.2	3.24	51!
tai256c	1	15.09	2	15.45	0.36	15.09	1.4	15.45	164!

and

$$\mathbf{A} = \left[\begin{array}{c|c} \mathbf{A}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right], \mathbf{B} = \left[\begin{array}{c|c} \mathbf{B}_1 & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} \end{array}\right]$$

$\widetilde{\mathbf{X}}$, $\mathbf{A}_1$ and $\mathbf{B}_1$ are $n_B \times n_A$, $n_A \times n_A$ and $n_B \times n_B$ matrices respectively. If both **A** and **B** are compressed an additional constraint to Eqs. (21)–(23) is needed to ensure the remaining number of permutations. This constraint is written as follows:

$$\sum_{i=1}^{n_B} \sum_{j=1}^{n_A} x_{ij} \geqslant n_B + n_A - n. \tag{24}$$

The permutations eliminated after compressing in both **A** and **B** are:

$$\min\left\{\frac{n!(n - n_B)!}{n_A!}, \frac{n!(n - n_A)!}{n_B!}\right\} \tag{25}$$

while the number of permutations left is:

$$\max\left\{\frac{n_A!}{(n - n_B)!}, \frac{n_B!}{(n - n_A)!}\right\}. \tag{26}$$

From Eq. (26) we find that only one permutation remains when $n_B + n_A - n = 0$. In this case the optimal value of the QAP is 0 and all $x_{ij} = 0$ in the compressed model. $n_B + n_A - n = 0$ when $n_B = n_A = \frac{n}{2}$.

Symmetries In addition to compressing the matrices, optional symmetry strategies can be utilized. Considering the objective function written as **XA•BX** we observe that when both **A** and **B** are

Table 4
Solution times (in seconds) for solving some instances to optimality utilizing symmetry (S) and compression (C) (when possible). DLR indicate the solution time when solving without S and C.

Instance	Disc in	DLR	S	C	C + S
nug12	A	59	20		
scr12	B	9.6	3.6		
chr12a	B	1.6	0.8		
tai12a	A	246	157		
rou12	B	1187	216		
esc16a	B	11.4	10.0	9.9	7.1
esc16b	B	158	>1200	71	>1200
esc16c	B	286	264	168	250

Table 5
Solution results when solving the instances esc32a, esc32c, esc32d, esc64a and tai64c from the QAPLIB

Instance	BKS	old LB	DLR	Nodes	Time (s)
esc32a	130	103	130	110365472	1618580
esc32c	642	616	642	1473284	24365
esc32d	200	191	200	2922791	36256
esc64a	116	98	116	62365	16370
tai64c	1855928	1855928	1855928	385103	182983

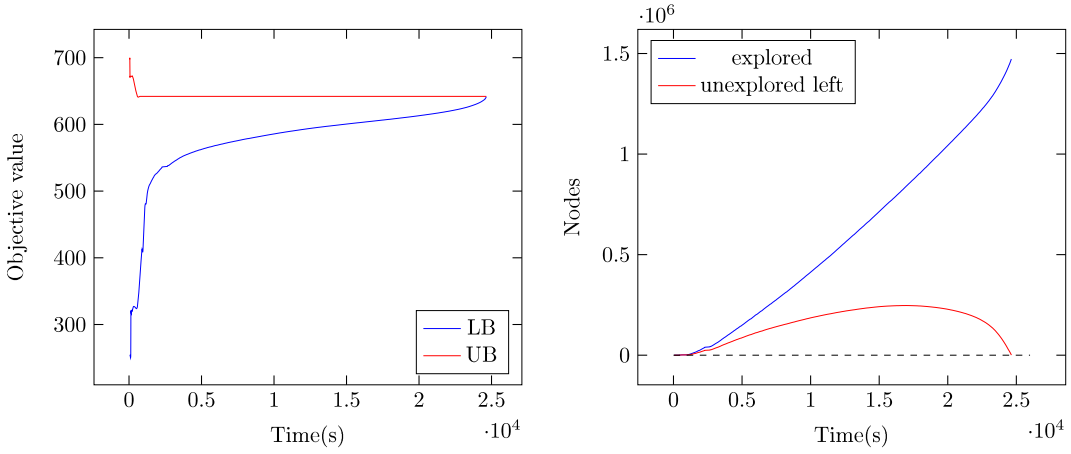

Fig. 2. Solution progress for the instance esc32c.

Table 6
Optimal permutation vectors obtained with the DLR model for the instances esc32a, esc32b, esc32c, esc32d, esc64a and tai64c in the QAPLIB.

Instance	
esc32a	11, 3, 7, 23, 19, 27, ∗, 14, 20, 17, 28, 9, 12, 4, 8, 2, 26, 24, 32, ∗, 22, 25, 6, 18, 29, 10, 30, ∗, ∗, ∗, ∗, ∗ $S\in$ {1,5,13,15,16,21,31}
esc32c	15, 12, ∗, 13, 22, 8, 24, 23, 20, 19, 4, 2, 1, 7, 6, 3, 5, 18, 17, 21, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗ $S\in$ {9,10,11,14, 16,25,26,27,28,29,30,31,32}
esc32d	18, 29, 10, 2, 25, 32, 22, 20, 24, 17, 30, 9, 1, 26, 31, 21, 19, 23,∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗ $S\in$ {3,4,5,6,7,8,11,12,13,14,15,16,27,28}
esc64a	18, 29, 10, 2, 25, 32, 22, 20, 24, 17, 30, 9, 1, 26, 31, 21, 19, 23,∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗, ∗ $S\in$ {3,4,5,6,7,8,11,12,13,14,15,16,27,28}
tai64c	1, 15, 32, 35, 61, 41, 29, 18, 12, 45, 63, 59, 47, ∗,. . .,∗ $S\in$ {2,3,4,5,6,7,8,9,10,11,13,14,16,17,19,20,21,22, 23,24,25,26,27,28,30,31,33,34, 36,37,38,39,40,42,43,44,46,48,49,50,51,52, 53,54,55,56,57,58,60,62,64}

symmetric we can write either $\mathbf{XA} \bullet \mathbf{BX} = \mathbf{A} \bullet \mathbf{X}^T\mathbf{BX} = \widetilde{\mathbf{A}} \bullet \mathbf{X}^T\mathbf{BX}$ or $\mathbf{XA} \bullet \mathbf{BX} = \mathbf{B} \bullet \mathbf{XAX}^T = \widetilde{\mathbf{B}} \bullet \mathbf{XAX}^T$ where $\widetilde{\mathbf{A}} = 2 \cdot \mathbf{U_A} - \mathbf{D_A}$ and $\widetilde{\mathbf{B}} = 2 \cdot \mathbf{U_B} - \mathbf{D_B}$. $\mathbf{U}_B$ and $\mathbf{U}_A$ are the upper (or lower) triangular parts and $\mathbf{D}_A$ and $\mathbf{D}_B$ the diagonal parts of the corresponding matrices. This reduces the number of variables and constraints. In Table 4 solution times for solving some smaller instances from the QAPLIB to optimality utilizing symmetry as well as compression (when possible) are given. When the symmetry option has been used it has been applied to the matrix not being discretized. Only the **A** matrix in the three last instances can be compressed. Table 4 shows that utilizing symmetry and compression reduces the solution time in many cases.

Some of the QAPLIB instances have persymmetric (symmetric w.r.t. the counter diagonal), centrosymmetric (symmetric w.r.t. the matrix center) or bisymmetric (symmetric and centrosymmetric) matrices. These properties can be utilized as well. For example, a centrosymmetric matrix **A** has the property **MA** = **AM**, where **M** is a counter-diagonal matrix, with elements equal to one in the counter diagonal and all entries off the counter-diagonal equal to zero. In this case two permutation vectors, $\mathbf{p}_1 = \mathbf{Xq}$ and $\mathbf{p}_2 = \mathbf{XMq}$ result in the same value of the objective function. In order to break this symmetry the first row of the permutation matrix can be partitioned to a left and a right half, $(\mathbf{x}_L|\mathbf{x}_R)$. Letting $\mathbf{x}_L = \mathbf{0}$ the symmetry will be broken and only one of the solutions will remain valid. In a similar way if the **B** matrix is centrosymmetric, a permutation vector $\mathbf{p}_3 = \mathbf{MXq}$ result in the same value of the objective function as with $\mathbf{p}_1$. This symmetry can be broken, for example, by including zero elements in the upper half of the first column in the permutation matrix. Breaking the centrosymmetry eliminates multiple solutions. There are several other symmetry options as well and several of them have been introduced in the literature (Anstreicher, 2003; de Klerk and Sotirov, 2010). However, we conclude this section only by stating that symmetries in the matrices have not been considered in greater detail in this paper, but we will introduce some alternative options for special structures of the **A** and **B** matrices in a forthcoming paper.

5. Results

By using the model formulation presented in this paper we have solved some previously unsolved instances from the QAPLIB to proven optimality. In Table 5 the results obtained with the DLR model are given. The first two columns show the best known solutions and the best lower bounds for the instances, reported in the QAPLIB.[2] The verified optimal solution obtained with the discrete linear reformulated MIP model, the total number of nodes visited and the solution time in seconds, as reported by Gurobi, are given in the last three columns. All instances in Table 5 have been discretized in **B** and compression was utilized in the $n = 64$ instances. All instances have been solved on a single PC with an Intel i7 4-core 2.8 GigaHertz processor and 6 GigaByte RAM, except for the esc64a instance where a PC with an Intel i7 6-core 3.2 GigaHertz processor was used. As a MILP solver Gurobi (4.0.1) with default parameter settings was used. The node files were stored on the hard drive, thus reducing the amount of RAM needed to solve the models. In Fig. 2, to the left, the best integer and best node solutions versus the solution time are illustrated when solving the instance esc32c. To the right, in the same figure, the number of nodes explored and nodes remaining versus the solution time are shown. The permutation vectors for the problems solved to optimality are given in Table 6, in the Appendix. Surprisingly the esc64a instance was solved in about 4.5 hours and tai64c in about 51 hours. Even though the problems with $n = 64$ are huge we were still able to solve them to proven optimality. This is both because of the efficiency of the discrete reformulation as well as the reduction of the permutation matrix by using compression. When solving the esc32a, esc32c and esc32d instances compression did, however, not result in a shorter solution time.

[2] The instance tai64c is a so-called grey pattern problem having a simplified QAP structure. The problem has previously been solved to proven optimality by Drezner (2006), using a special purpose algorithm utilizing the simpler structure. The problem is included in Table 5, since we have solved it in the QAP form and it was reported as unsolved in the QAPLIB.

Table 5 indicates that the solution times are quite moderate when comparing them to previously solved instances of $n \geqslant 30$ (Anstreicher et al., 2002; Adams et al., 2007). A good branching strategy is crucial for any large MILP problem as indicated in Achterberg et al. (2005). In all the test runs we, however, used default settings for Gurobi. A drawback with some recent methods is the amount of RAM required as indicated in (Adams et al., 2007; Hahn et al., 2008). With the DLR-model presented in this paper the amount of RAM is, on the other hand, not a crucial issue. Even on the largest problem from the QAPLIB, tai256c, the DLR can be written as a MILP model with $0.36n^2$ binary and $0.72n^2$ continuous variables as well as $1.4n + 1.08n^2$ constraints.

Having the opportunity to test the efficiency of new solution approaches on instances from the QAPLIB is a great resource. Results obtained with several different solvers on the same instances can easily be found in the QAPLIB from well documented references, thanks to the maintainers Hahn and Anjos (2002). The unsolved and recently solved problems in the library are however very challenging. This restricts the possibilities of doing comprehensive numerical comparisons, because of the CPU-time required to solve these problems. Already the CPU-time for a single instance might end up lasting weeks, even when solving it using parallel computation (Anstreicher et al., 2002). This practical limitation has also, in our case, limited the amount of comparisons done in this paper.

6. Conclusions

The exact discrete linear reformulation, DLR, presented in this paper is shown to be compact and from numerical experiments we have found that it gives very good results, especially on sparse instances with few unique elements per row. We have presented solutions to never before solved instances in the QAPLIB. The run times for problems solved with the DLR-model are reasonably short (taking into account the combinatorial complexity of the instances) and the RAM memory required is not a crucial issue with this model. However, we do not think that any QAP can be solved more efficiently with the given reformulation than with some other approach but we are of course happy that some of the instances in the QAPLIB that had never been solved to optimality before could be solved with this formulation. Default settings in the MILP solver Gurobi were used when conducting the experiments, whilst the node files were stored on a hard drive. A good branching strategy could probably still speed up the solution times. Symmetry options were not considered in greater detail in this paper, but will be considered in a forthcoming paper.

Acknowledgements

The financial support from the Academy of Finland project 127992 is gratefully acknowledged. We also want to acknowledge professor Peter Hahn at the University of Pennsylvania as well as Miguel Anjos at École Polytechnique Montreal for their kind support in connection with the QAPLIB.

Appendix A. Optimal permutations

Below are the optimal permutation vectors for the instances esc32a, esc32b, esc32c, esc32d, esc64a and tai64c obtained with the DLR QAP formulation. For each instance, the elements indicated by * can be replaced with any permutation of the elements in the corresponding set S.

References

Achterberg, T., Koch, T., Martin, A., 2005. Branching rules revisited. Operations Research Letters 33 (1), 42–54.

Adams, W.P., Guignard, M., Hahn, P.M., Hightower, W.L., 2007. A level-2 reformulation–linearization technique bound for the quadratic assignment problem. European Journal of Operational Research 180 (3), 983–996.

Anstreicher, K., Brixius, N., Goux, J.-P., Linderoth, J., 2002. Solving large quadratic assignment problems on computational grids. Mathematical Programming 91, 563–588.

Anstreicher, K.M., 2003. Recent advances in the solution of quadratic assignment problems. Mathematical Programming, Ser. B 97, 27–42.

Anstreicher, K.M., Brixius, N.W., 2001. A new bound for the quadratic assignment problem based on convex quadratic programming. Mathematical Programming 89, 341–357. doi:10.1007/PL00011402 <http://dx.doi.org/10.1007/PL00011402>.

Burkard, R., Çela, D.E., Karisch, S., Rendl, F., 1997. Qaplib – a quadratic assignment problem library. Journal of Global Optimization 10, 391–403.

de Klerk, E., Sotirov, R., 2010. Exploiting group symmetry in semidefinite programming relaxations of the quadratic assignment problem. Mathematical Programming 122, 225–246. doi:10.1007/s10107-008-0246-5 <http://dx.doi.org/10.1007/s10107-008-0246-5>.

Drezner, Z., 2006. Finding a cluster of points and the grey pattern quadratic assignment problem. OR Spectrum 28, 417–436. doi:10.1007/s00291-005-0010-7 <http://dx.doi.org/10.1007/s00291-005-0010-7>.

Drezner, Z., Hahn, P., Taillard, e., 2005. Recent advances for the quadratic assignment problem with special emphasis on instances that are difficult for meta-heuristic methods. Annals of Operations Research 139, 65–94. doi:10.1007/s10479-005-3444-z <http://dx.doi.org/10.1007/s10479-005-3444-z>.

Gounaris, C.E., Misener, R., Floudas, C.A., 2009. Computational comparison of piecewise linear relaxations for pooling problems. Industrial & Engineering Chemistry Research 48 (12), 5742–5766 <http://pubs.acs.org/doi/abs/10.1021/ie8016048>.

Hahn, P., Anjos, M., 2002. Qaplib - a quadratic assignment problem library online. University of Pennsylvania, School of Engineering and Applied Science, <http://www.seas.upenn.edu/qaplib/>.

Hahn, P.M., Zhu, Y.-R., Guignard, M., Hightower, W.L., 2008. A level-3 reformulation-linearization technique based bound for the quadratic assignment problem. Optimization-Online 09. URL <http://www.optimization-online.org/DB_HTML/2008/09/2094.html>.

Kaufman, L., Broeckx, F., 1978. An algorithm for the quadratic assignment problem using bender's decomposition. European Journal of Operational Research 2 (3), 207–211.

Koopmans, T., Beckmann, M., 1957. Assignment problems and location of economic activities. Econometrica 25, 53–76.

Loiola, E., Abreu, N., Boaventura-Netto, P., Hahn, P., Querido, T., 2007. A survey for the quadratic assignment problem. European Journal of Operational Research 176 (2), 657–690.

Peng, J., Mittelmann, H., Li, X., 2010. A new relaxation framework for quadratic assignment problems based on matrix splitting. Mathematical Programming Computation 2 (1), 59–77 <http://dx.doi.org/10.1007/s12532-010-0012-6>.

Taillard, E., 1991. Robust taboo search for the quadratic assignment problem. Parallel Computing 17 (4–5), 443–455.

Zhang, H., Beltran-Royo, C., Ma, L., 2010. Solving the quadratic assignment problem by means of general purpose mixed integer linear programming solvers. Statistics and Operations Research, Rey Juan Carlos University (URJC), C/ Tulipán s/n, 28933, M=stoles (Madrid), Spain 04, 1.