CHEMICAL ENGINEERING TRANSACTIONS

VOL. 32, 2013

Chief Editors: Sauro Pierucci, Jiří J. Klemeš
Copyright © 2013, AIDIC Servizi S.r.l.,
ISBN 978-88-95608-23-5; **ISSN** 1974-9791

A publication of
AIDIC
The Italian Association
of Chemical Engineering
Online at: www.aidic.it/cet

DOI: 10.3303/CET1332219

Tightening a Discrete Formulation of the Quadratic Assignment Problem

Axel Nyberg*, Tapio Westerlund

Center of Excellence in Optimization and Systems Engineering, Åbo Akademi University, Biskopsgatan 8, FI-20500 Åbo, Finland
axel.nyberg@abo.fi

The quadratic assignment problem is a well studied and notoriously difficult combinatorial problem. Recently, a discrete linear formulation of the quadratic assignment problem was presented that solved five previously unsolved instances from the quadratic assignment library, QAPLIB, to optimality. That formulation worked especially well on sparse instances. In this paper we show how to tighten that formulation by adding cuts to the auxiliary variables. The cuts are derived from solving linear programming problems before solving the main problem. The linear programming problems are easily solved even for larger instances and therefore many cuts can be added without any considerable change of computing time. With only a few cuts we can improve the root node bound considerably.

1. Introduction

The quadratic assignment problem (QAP) was originally presented by Koopmans and Beckmann (1957). QAPs arise in various fields including facility location, scheduling, manufacturing, statistical data analysis and economics to name a few. A tremendous amount of work has been done both on lower bounding and on finding solutions, but still, some instances of size $n = 30$ of the QAP are considered extremely difficult to solve to proven optimality (Loiola et al., 2007). Recently Nyberg and Westerlund (2012) solved four of the esc instances, of size $n = 32$ and $n = 64$, from the quadratic assignment problem library, QAPLIB, that had remained unsolved since 1990. Fischetti et al. (2012) solved two instances including the largest QAP solved so far ($n = 128$) and Nyberg et al. (2013b) solved the last unsolved instance from the esc family. The esc instances are about minimizing hardware when testing circuits and are all sparse and contain a lot of symmetries (Eschermann and Wunderlich, 1990). The instance tai30b (Taillard, 1995) is the latest QAP to be solved. It was solved using a similar code as in Hahn and Saltzmann (2010). The last remaining unsolved instance of size $n = 30$ from the QAPLIB is the tai30a (Taillard, 1991). In addition to solving a few instances, a lot of the recent work has been concentrated on calculating tight lower bounds for the larger instances (Peng et al., 2010).

Koopmans and Beckmann formulated the QAP in the following manner:

$$\min \sum_i^n \sum_j^n \sum_k^n \sum_l^n a_{ij} b_{kl}\, x_{ik} x_{jl} \tag{1}$$

s.t.

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

$$\sum_j^n x_{ij} = 1, \quad j = 1, 2, \ldots, n, \tag{3}$$

$$x_{ij} \in \{0,1\}, \quad i, j = 1,2, \ldots, n, \tag{4}$$

where a_{ij} and b_{kl} are the elements in the given flow and distance matrices A and B respectively.

The Koopmans-Beckmann formulation has $n^2(n-1)^2$ bilinear terms resulting in poor lower bounds if linearized directly using for example McCormick envelopes (McCormick, 1976). Instead of linearizing the above formulation, Nyberg and Westerlund (2012) recently proposed a discrete mixed-integer nonlinear programming formulation (MINLP) for the QAP with only n^2 discrete bilinear terms.

$$\min \sum_i^n \sum_j^n a'_{ij} b'_{ij} \tag{5}$$

s.t.

$$a'_{ij} = \sum_{k=1}^n a'_{kj} x_{ik} \quad \forall i,j, \tag{6}$$

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

$$x_{ij} \in X_n, \tag{8}$$

where X_n is the feasible set of the assignment constraints Eq (2), Eq (3) and Eq (4). We will use Eq (8) throughout this paper since these constraints are present in all our formulations. The above MINLP was reformulated into a discrete linear form (DLR) by which several previously unsolved instances (with sizes $n = 32$ and $n = 64$) from QAPLIB, were solved to proven optimality. The same idea was incorporated when optimizing a supply-chain design for specialty chemicals with good results (Nyberg et al., 2013a). This mixed-integer linear programming (MILP) formulation worked especially well on sparse instances and instances with few unique elements in one of the matrices. An interesting observation with the DLR formulation is that even though the root node bound is rather low compared to many other methods, the lower bound will increase very quickly after a few branch and bound iterations. In this paper we show how the root node bounds can be tightened by adding some simple cuts a priori.

2. Discrete linear reformulation

Below is the MILP formulation from Nyberg et al. (2013b).

$$\min \sum_{i=1}^n \sum_{j=1}^n \sum_{m=1}^{M_i} B_i^m z_{ij}^m \tag{9}$$

s.t. Eq(8) and

$$z_{ij}^m \leq \bar{A}_j \sum_{k \in K_i^m} x_{kj}, \; m = 1,2,\dots,M_i \quad \forall i,j, \tag{10}$$

$$\sum_{m=1}^{M_i} z_{ij}^m = \sum_{k=1}^n a_{kj} x_{ik}, \quad \forall i,j, \tag{11}$$

$$z_{ij}^m \geq \underline{A}_j \sum_{k \in K_i^m} x_{kj}, \quad m = 1,2,\dots,M_i \quad \forall i,j, \tag{12}$$

where

$$z_{ij}^m \in [0, \bar{A}_j] \qquad \forall i,j, \tag{13}$$

$$\bar{A}_j = \max_i a_{ij} \qquad \forall j, \tag{14}$$

$$\underline{A}_j = \min_i a_{ij} \qquad \forall j : j \neq i, \tag{15}$$

$$K_i^m = \{ j | b_{ij} = B_i^m \} \quad \forall i,j : i \neq j \wedge m = 1,\dots,M_i, \tag{16}$$

$$b_{ij} \in \{B_i^1, B_i^2, \dots, B_i^{M_i}\} \quad \forall i,j. \tag{17}$$

In the above formulations Eq(7) is discretized in Eq(9). Since $x_{ij} \in X_n$, for any feasible MILP solution, only one of the z_{ij}^m variables in Eq(11) is active ($\geq 0 \; \forall m$). From Eq(9) we observe that, for any row i, the constants B_i^m are the same despite the value of j. If we look at a complete row i of the matrix B, we observe that each active z_{ij}^m variable is preceded with a different element in that row. In the linear programming (LP) relaxations however, the assignment constraints in Eq(2), Eq(3) and Eq(4) are relaxed and therefore multiple z_{ij}^{m} variables are active. When minimizing over Eq(9) the LP relaxation will give as large values as possible to the variables z_{ij}^{m} preceded by the lowest constants B_i^1 while the z_{ij}^{m} variables that are preceded by the largest constants, $B_i^{M_i}$, will be as small as possible. In other words, the variables with the largest impact on the value of the objective function will be as close to zero as possible in the LP relaxation. Therefore, we will next show how to restrict these variables in order to tighten the whole formulation.

3. Bounding the auxiliary variables

In this section we show two different approaches for adding cuts to the original problem.

3.1 Largest elements

In order to increase the value of the z_{ij}^{m} variables preceded by large constants we can add lower bounds on the sum of those variables. By choosing the *p* largest values from every row in B and defining a new matrix B^{up_p} where:

$$b_{ij}^{up_p} = \begin{cases} 1, & \text{if } \mathrm{b_{ij}} \geq \text{the } p\text{th largest element in row } i, \\ 0, & \text{otherwise,} \end{cases} \tag{18}$$

we obtain a new QAP whose optimal solution equals the highest right-hand side value for the cut. Since we were changing all the largest elements in matrix B to one and removing the rest, the sum of the z_{ij}^{m} variables preceding the largest values has to be larger than the optimal solution of that problem. Therefore we can add the following cut:

$$\sum_j^n \sum_i^n z_{ij}^{M_p} \geq \min\left(\mathrm{QAP}\{A|B^{up_p}\}\right) \forall\, p = 1,2,\dots,U_B, \tag{19}$$

where M_p corresponds to all those variables that are preceded by a constant larger than or equal to the *p*th largest element in row i of the matrix B and U_b is the maximal amount of unique elements in a row of B. In other words we add cuts until we have all the z_{ij}^{m} variables on the left-hand side of Eq(19).

3.2 Smallest elements

In the same way as the lower bounds on the sum of the z_{ij}^{m} variables preceded by the largest constants are added, upper bounds on the sum of the the z_{ij}^{m} variables preceded by the smallest constants can be added. Now, by choosing the *p* smallest values from every row in B and defining a new matrix B^{lo_p} where:

$$b_{ij}^{lo_p} = \begin{cases} 1, & \text{if } \mathrm{b_{ij}} \leq \text{the } p\text{th smallest element in row } i, \\ 0, & \text{otherwise.} \end{cases} \tag{20}$$

Now, by maximizing instead of minimizing the new problem $\mathrm{QAP}\{A|B^{lo_p}\}$ we get the right-hand side for the following cut:

$$\sum_j^n \sum_i^n z_{ij}^{m_p} \leq \max(\mathrm{QAP}\{A|B^{lo_p}\} \forall\, p = 1,2,\dots,U_B\ ), \tag{21}$$

where m_p corresponds to the variables that are preceded by constants smaller than or equal to the pth smallest element of row i. By bounding the elements preceded by smaller elements, Eq(11) will force the other variables to take larger values and therefore the lower bound will also become higher.

3.3 Sub-optimal cuts

Unfortunately, finding the optimal cuts yields a new QAP that is no easier than the original QAP. Therefore we use a suboptimal solution for the new sub-problems, which can be found very quickly, to generate the cuts.

For the cuts in Section 3.1, the rows of matrix A are sorted in ascending order (with the diagonals excluded) while the rows of B^{up_p} are sorted in descending order, i.e. all elements equal to one are to the left of the matrix. By only changing the row order of matrix B we obtain a valid underestimating LP problem for the problem in Section 3.1. In the formulations below x_{kj} are continuous variables.

$$\min \sum_i^n \sum_j^n A_{\mathrm{SORT}_{ij}} B_{\mathrm{CUT}_{ij}} \tag{22}$$

s.t. Eq (8) and

$$B_{\mathrm{CUT}_{ij}} = \sum_{k=1}^n B_{\mathrm{SORT}_{ij}} x_{kj} \quad \forall i,j. \tag{23}$$

On the other hand, for the cuts in Section 3.2 the rows of both matrix A and B^{lo_p} are sorted in descending order.

$$\max \sum_i^n \sum_j^n A_{\mathrm{SORT}_{ij}} B_{\mathrm{CUT}_{ij}} \tag{24}$$

s.t. (8) and

$$B_{\mathrm{CUT_{ij}}} = \sum_{k=1}^n B_{\mathrm{SORT}_{ij}} x_{kj} \quad \forall i,j. \tag{25}$$

Eq (23) and Eq (25) are derived from applying Eq(7) on the sorted B matrices respectively. Therefore only the rows of the sorted B matrices are switched. Eq(22) gives a valid lower bound for $\min(\mathrm{QAP}\{A|B^{up_p}\})$ while the solution of Eq(24). Is a valid upper bound for $\max(\mathrm{QAP}\{A|B^{lo_p}\}$. The above LP problems are easily solvable and all *p* cuts can be calculated in a fraction of a second even for the largest instances from QAPLIB. However, the solution quality is considerably poorer than for the optimal cases.

3.4 Example

We will illustrate our examples with the following small QAP instance:

$$A = \begin{bmatrix} 0 & 5 & 3 & 7 \\ 5 & 0 & 4 & 9 \\ 3 & 4 & 0 & 8 \\ 7 & 9 & 8 & 0 \end{bmatrix} \quad \text{and} \quad B = \begin{bmatrix} 0 & 1 & 1 & 6 \\ 1 & 0 & 2 & 2 \\ 1 & 2 & 0 & 3 \\ 6 & 2 & 3 & 0 \end{bmatrix}$$

When discretizing the first row as in Eq (9) to Eq (11), of the above example, two auxiliary variables z_{ij}^m are needed for every b'_{i1} in that row. The objective function (Eq (9)) for the first row will now be $\sum_{i=1}^n (1z_{i1}^1 + 6z_{i1}^2)$. When solving the LP relaxation for the problem the sum for the variables $\sum_{i=1}^n 1z_{i1}^1$ will take up as much as possible of Eq (11) while the second part of the sum, $\sum_{i=1}^n 6z_{i1}^2$, will be as small as possible.

In the above instance, $U_B = 3$, i.e., the maximum number of unique elements in a row of the matrix B. Therefore two cuts will be added in both cases. The new matrices B^{up_p} and B^{lo_p} are shown below for both $p = 1$ and $p = 2$.

$$B^{up_1} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad B^{up_2} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}, \quad B^{lo_1} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}, \quad B^{lo_2} = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Eq(22) and Eq(23) are applied on the matrices below in order to calculate the sub-optimal cuts for Section 3.1. Since we have sorted the matrices and only change the order of the rows in the matrices $B_{\mathrm{SORT}}^{up_1}$ and $B_{\mathrm{SORT}}^{up_2}$ we obtain a poorer lower bound than by solving the QAP in section 3.1.

$$A_{\overline{\mathrm{SORT}}} = \begin{bmatrix} 3 & 5 & 7 & 0 \\ 4 & 5 & 9 & 0 \\ 3 & 4 & 8 & 0 \\ 7 & 8 & 9 & 0 \end{bmatrix}, \quad B_{\mathrm{SORT}}^{up_1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad B_{\mathrm{SORT}}^{up_2} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

And when calculating the sub-optimal cuts for Section 3.2 the following matrices are used.

$$A_{\overline{\text{SORT}}} = \begin{bmatrix} 7 & 5 & 3 & 0 \\ 9 & 5 & 4 & 0 \\ 8 & 4 & 3 & 0 \\ 9 & 8 & 7 & 0 \end{bmatrix}, \quad B_{\text{SORT}}^{lo_1} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad B_{\text{SORT}}^{lo_2} = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

As can be seen from above, changing the order of the rows in $B_{\text{SORT}}^{lo_1}$ and $B_{\text{SORT}}^{lo_2}$, as in Eq (25), will affect the solution value of Eq (24).

4. Results

All test runs were conducted on a 2.8 GHz Intel-i7(quad core) computer with 4 GB of RAM. CPLEX 12.5.0 was used as the MILP and LP solver. In Table 1, the root node bounds for some instances from the QAPLIB are shown. As can be seen, adding simple cuts can improve the root node bound considerably. These cuts don't require any computational effort since we are only solving a few LP problems.
In Table 2 we show the bounds if the sub-problems are solved to optimality. In the examples, the sub-problems require roughly the same amount of computational effort as the original problem. However, using a good lower bound technique, giving close to optimal lower bounds in a relatively short computational time, could make these cuts very effective.

Table 1: Lower bounds in the root node for some smaller instances from QAPLIB when adding the sub-optimal cuts.

Instance	Opt. Sol.	DLR-LB	Eq (22)	Eq (19)	Eq (19), Eq(22)
nug12	578	409	487	422	487
nug14	1,014	719	851	728	851
nug16b	1,240	869	1,019	890	1,019
nug30	6,124	3612	4,519	3,867	4,519
rou12	233,528	176,045	192,242	192,939	194,065
rou15	354,210	251,150	275,338	277,095	277,525
rou20	725,522	500,814	553,535	555,067	560,236
had12	1,652	1,416	1,526	1,469	1,526
had20	6,922	5,286	6,127	5,928	6,127
tai12a	224,416	172,852	183,166	182,542	184,157
tai20a	703,482	464,802	537,327	537,616	539,837
ste36a	9,526	5,780	7,102	5,782	7,102
ste36b	15,852	5,945	8,611	5,945	8,611
ste36c	8,239,110	5,518,062	6,381,100	5,518,062	6,381,100
esc16a	68	26	38	38	38
esc16b	292	196	220	220	220
esc16c	160	60	83	83	83
esc32a	130	11	35	35	35
esc32b	168	48	96	96	96
esc32c	642	309	350	350	350
esc32d	200	70	106	106	106
esc32h	438	156	257	257	257
esc64a	116	38	47	47	47
tho30	144,936	79,719	90,206	79,742	90,206
wil50	48,816	32,987	37,911	36,413	37,911

Table 2: Lower bounds for a few instances with optimal cuts.

Instance	Optimal value	DLR-LB	LB-subcuts	LB-optcuts
Nug12	578	409	487	501
esc16a	68	26	38	68
esc16b	292	196	220	292
esc16c	160	60	83	160
tai12a	224,416	172,852	184,157	198,165
had12	1,652	1,416	1,526	1,592
rou12	235,528	176,045	194,065	207,717
rou15	354,210	251,150	277,525	300,001

5. Conclusions

We have showed that we are able to improve the root node bound for many instances by adding a few rather simple cuts. For some of the sparse instances we have showed that adding the optimal cuts yields root node bounds that are equal to the optimal solution. However solving the sub-problems to optimality seems as difficult as solving the underlying QAP itself. If all the sub-problems were to be solved to optimality, better cuts could be derived. Future work should address how to calculate close to optimal lower bounds in less computational time for the sub-problems.

Acknowledgments

Financial support from the Foundation of Åbo Akademi University, as part of the grant for the Center of Excellence in Optimization and Systems Engineering are gratefully acknowledged.

References

Eschermann B., Wunderlich H.J. 1990, Optimized synthesis of self-testable finite state machines. Fault-Tolerant Computing, FTCS-20. 20th International Symposium. 390-397

Fischetti M., Monaci M., Salvagnin D., 2012, Three ideas for the Quadratic Assignment Problem. Operations Research 60, 954-964.

Hahn P.M., Saltzman M.J., 2010, A New Branch-and-Bound Solver for the Quadratic Assignment Problem Based on the Level-3 Reformulation-Linearization Technique, Studia Informatica Universalis, Combinatorial Optimization in Practice 8.2, 97-106.

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

Loiola E.M., Abreu N.M., Boaventura-Netto P.O., Hahn P., Querido T., 2007, A survey for the quadratic assignment problem, European Journal of Operational Research 176 (2), 657-690, DOI:10.1016/j.ejor.2005.09.032.

McCormick G.P., 1976, Computability of global solutions to factorable nonconvex programs: Part I - Convex underestimating problems. Mathematical Programming, Springer Berlin / Heidelberg, 147-175

Nyberg A., Grossmann I. E., Westerlund T., 2013a, An efficient reformulation of the multiechelon stochastic inventory system with uncertain demands, AIChE J 59(1), 23-28. DOI:10.1002/aic.13977

Nyberg A., Westerlund T., 2012, A new exact discrete linear reformulation of the quadratic assignment problem. European Journal of operational Research 220(2), 314-319.

Nyberg A., Westerlund T.,Lundell A., 2013b. Improved Discrete Reformulations for the Quadratic Assignment Problem, Lecture Notes in Computer Science 7874, 193-203. DOI: 10.1007/978-3-642-38171-3_13

Peng J., Mittelmann H., Li.X., 2010, A new relaxation framework for quadratic assignment problems based on matrix splitting, Mathematical Programming Computation 2, 59-77.

Taillard É.D., 1991, Robust tabu search for the quadratic assingnment problem, Parallel Computing 17, 443-455.

Taillard É.D., 1995, Comparison of iterative searches for the quadratic assignment problem, Location Science 3 (2), 87-105. DOI: 10.1016/0966-8349(95)00008-6