The Largest Unsolved QAP Instance Tai256c Can Be Converted into A 256-dimensional Simple BQOP with A Single Cardinality Constraint

Koichi Fujii[*] Sunyoung Kim[†] Masakazu Kojima[‡] Hans D. Mittelmann[§] Yuji Shinano[¶]

October 28, 2022

Abstract

Tai256c is the largest unsolved quadratic assignment problem (QAP) instance in QAPLIB; a 1.48% gap remains between the best known feasible objective value and lower bound of the unknown optimal value. This paper shows that the instance can be converted into a 256 dimensional binary quadratic optimization problem (BQOP) with a single cardinality constraint which requires the sum of the binary variables to be 92. The converted BQOP is much simpler than the original QAP tai256c and it also inherits some of the symmetry properties. However, it is still very difficult to solve. We present an efficient branch and bound method for improving the lower bound effectively. A new lower bound with 1.36% gap is also provided.

Key words. Quadratic assignment problems, largest unsolved instance, exploiting symmetry, Lagrangian-DNN relaxation, Newton-bracketing method, branch-and-bound methods.

AMS Classification. 90C20, 90C22, 90C25, 90C26.

1 Introduction

For a positive integer n, we let $N = \{1, \ldots, n\}$ represent a set of locations and also a set of facilities. Given $n \times n$ symmetric matrices $\boldsymbol{A} = [a_{ik}]$ and $\boldsymbol{B} = [b_{j\ell}]$, the quadratic assignment

[*]NTT DATA Mathematical Systems Inc., Tokyo 160-00016, Japan (fujii@msi.co.jp).

[†]Corresponding author, Department of Mathematics, Ewha W. University, Seoul, 52 Ewhayeodae-gil, Sudaemoon-gu, Seoul 03760, Korea (skim@ewha.ac.kr). The research was supported by NRF 2021-R1A2C1003810.

[‡]Department of Industrial and Systems Engineering, Chuo University, Tokyo 192-0393, Japan (kojima@is.titech.ac.jp).

[§]School of Mathematical and Statistical Sciences, Arizona State University, Tempe, Arizona 85287-1804, U.S.A. (mittelma@asu.edu).

[¶]Department of Applied Algorithmic Intelligence Methods (A^2IM), Zuse Institute Berlin, Takustrasse 7,14195 Berlin, Germany (shinano@zib.de).

problem (QAP) is stated as

$$\zeta^* = \min_{\pi} \sum_{i \in N}^{n} \sum_{k \in N}^{n} a_{ik} b_{\pi(i)\pi(k)}, \tag{1}$$

where a_{ik} denotes the flow between facilities i and k, $b_{j\ell}$ the distance between locations j and ℓ, and $(\pi(1), \ldots, \pi(n))$ a permutation of $1, \ldots, n$ such that $\pi(i) = j$ if facility i is assigned to location j. We assume that the distance b_{jj} from $j \in N$ to itself and the flow a_{ii} from $i \in N$ to itself are both zero.

The QAP is known as NP-hard in theory, and solving exactly large scale instances (*e.g.*, $n \geq 40$) is very difficult in practice. To obtain an exact optimal solution, we basically need two types of techniques. The first one is for computing approximate optimal solutions and objective values. Heuristic methods such as tabu search, genetic method and simulated annealing have been developed for the QAP [7, 12, 27, 28]. Those methods frequently attain near-optimal solutions, which also happen to be the exact optimal solution in some cases. The exactness is, however, not guaranteed in general. The objective value $\bar{\zeta}$ obtained by those methods serves as an upper bound (UB) for the unknown optimal value ζ^*. The second technique is to provide a lower bound (LB) $\underline{\zeta}$ for ζ^*. If $\underline{\zeta} = \bar{\zeta}$ holds, then we can conclude that $\underline{\zeta} = \zeta^* = \bar{\zeta}$. Various relaxation methods [3, 13, 19, 25, 29] have been proposed for computing LBs. The two techniques mentioned above play as essential tools in the branch and bound (BB) method for QAPs [2, 6, 14, 19, 22, 26].

In this paper, we focus our attention on the largest unsolved instance tai256c in QAPLIB, and show:

- Tai256c can be converted into a 256-dimensional binary quadratic optimization problem (BQOP) with a single cardinality constraint $\sum_{i=1}^{256} x_i = 92$.
- The converted BQOP satisfies a certain nice symmetry property inherited from tai256c.
- The proposed BB method can improve LBs for the unknown optimal value.

Moreover, we provide a new LB.

In case of the instance tai256c [5], $n = 256$ and both matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ satisfy certain symmetries. Using the symmetry of the matrix $\boldsymbol{A}$ presented in Section 2.1 of [10], we can convert QAP (1) to the following BQOP with a single cardinality constraint:

$$\zeta^* \;=\; \min \left\{ \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} : \boldsymbol{x} \in \{0,1\}^n \text{ and } \sum_{i=1}^{n} x_i = 92 \right\}. \tag{2}$$

We note that the best known feasible solution π^* of tai256c with the objective value $\bar{\zeta} = 44759294$ is converted to a feasible solution $\boldsymbol{x}^*$ of BQOP (2) with the same objective value $\bar{\zeta}$ (see [5]). Also the best known LB, $\underline{\zeta} = 44095032$, for tai256c serves as an LB of BQOP (2) (see [1]). More details of this conversion is described in Section 2.

Furthermore, the matrix $\boldsymbol{B}$ satisfies a symmetry such that

$$\boldsymbol{P}^T \boldsymbol{B} \boldsymbol{P} = \boldsymbol{B} \text{ for every } \boldsymbol{P} \in \mathcal{G}, \tag{3}$$

where $\mathcal{G}$ is a group of permutation matrices satisfying

$$\left.\begin{array}{l} \boldsymbol{P}_1 \boldsymbol{P}_2 \in \mathcal{G} \text{ if } \boldsymbol{P}_1 \in \mathcal{G} \text{ and } \boldsymbol{P}_2 \in \mathcal{G}, \\ \boldsymbol{P}^{-1} = \boldsymbol{P}^T \in \mathcal{G} \text{ if } \boldsymbol{P} \in \mathcal{G}. \end{array}\right\} \tag{4}$$

We describe the symmetry including the orbit branching [21, 24] in Section 3.

Exploiting symmetries of QAPs in their SDP relaxation was discussed in [8, 9, 23] (also [4] in their DNN relaxation). However, those results are not relevant to the subsequent discussion of this paper.

In Section 2, we show how to convert QAP (1) into BQOP (2). The conversion can be obtained using the result in Section 2.1 of [10]. The symmetry in the converted BQOP inherited from QAP (1) is discussed in Section 3. In Section 4, we present computational results using Gurobi Optimizer (version 9.5.2). We show that Gurobi could improve neither of the known LB $\underline{\zeta}$ and UB $\bar{\zeta}$. In Section 5, we propose a branch-and-bound method with the use of the Newton-bracketing (NB) method [18] to prove that the optimal value is not less than a given $\hat{\zeta}$. Here $\hat{\zeta}$ is a target LB chosen in the interval of the best known LB $\underline{\zeta}$ and UB $\bar{\zeta}$ before starting the BB method. If we chose $\hat{\zeta}$ to be the best known objective value $\bar{\zeta} = 44759294$, then $\bar{\zeta}$ would be proved to be the optimal value. In that case, however, the number of nodes which the BB method needs to generate is estimated at more than 4.1e14; hence the target $\hat{\zeta} = \bar{\zeta}$ is too difficult to attain. A target LB 44150000 (1.36% gap) was attained on Mac Studio (20 cpu).

2 Conversion from QAP (1) to BQOP (2)

We present a conversion of QAP (1) into BQOP (2) using Theorem 1 in Section 2.1 of [10]. Recall that $a_{ii} = 0$ for every $i \in N$. We say that two facilities $i \in N$ and $k \in N$ are *clones* if

$$a_{ik} = a_{ki},\ a_{ih} = a_{kh} \text{ and } a_{hi} = a_{hk} \text{ for every } h \in N \backslash \{i, k\}.$$

We write $i \sim k$ if two facilities $i \in N$ and $k \in N$ are clones. Then $\sim$ becomes an equivalence relation. We denote $\theta(i)$ as the class of facilities k which is equivalent to i, *i.e.*, $\theta(i) = \{k \in N : i \sim k\}$. Define

$$\begin{aligned} M &= \{\theta(i) : i \in N\}, \\ \tilde{a}_{\theta(i)\theta(k)} &= a_{ik} \text{ for every } i \in N \text{ and } k \in N \text{ with } i \neq k, \text{ or equivalently,} \\ \tilde{a}_{uv} &= a_{ik} \text{ if } u = \theta(i),\ v = \theta(k),\ i \in N \text{ and } k \in N \text{ with } i \neq k, \\ \mu_u &= |\{i : \theta(i) = u\}| \text{ for every } u \in M. \end{aligned}$$

By [10, Theorem 1], QAP (1) is equivalent to

$$\zeta^* = \min \left\{ \sum_{i \in N} \sum_{u \in M} \sum_{j \in N} \sum_{v \in M} \tilde{a}_{uv} b_{ij} x_{iu} x_{jv} : \begin{array}{l} x_{iu} \in \{0,1\}\ ((i,u) \in N \times M), \\ \sum_{i \in N} x_{iu} = \mu_u\ (u \in M), \\ \sum_{u \in M} x_{iu} = 1\ (i \in N) \end{array} \right\}. \quad (5)$$

We note that $\tilde{a}_{uu}$ may not be zero ($u \in M$).

In the QAP instance tai256c, the matrix A satisfies that

$$a_{ik} = 1\ (1 \leq i,\ k \leq 92,\ i \neq k) \text{ and } a_{ik} = 0\ (93 \leq i \leq 256 \text{ or } 93 \leq k \leq 256).$$

It is straightforward to verify that M consists of two classes

$$\{1, \ldots, 92\} \text{ and } \{93, \ldots, 256\},$$

and that

$$\widetilde{\boldsymbol{A}} = \begin{pmatrix} \tilde{a}_{11} & \tilde{a}_{12} \\ \tilde{a}_{21} & \tilde{a}_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}.$$

Therefore, QAP (5) turns out to be

$$\begin{aligned} \zeta^* &= \min \left\{ \sum_{i\in N}\sum_{u=1}^{2}\sum_{j\in N}\sum_{v=1}^{2} \tilde{a}_{uv} b_{ij} x_{iu} x_{jv} : \begin{array}{l} x_{iu} \in \{0,1\}\ ((i,u) \in N \times M), \\ \sum_{i\in N} x_{iu} = \mu_u\ (u \in M), \\ \sum_{u\in M} x_{iu} = 1\ (i \in n) \end{array} \right\} \\ &= \min \left\{ \sum_{i\in N}\sum_{j\in N} b_{ij} x_{i1} x_{j1} : \begin{array}{l} x_{iu} \in \{0,1\}\ ((i,u) \in N \times \{1,2\}), \\ \sum_{i\in N} x_{i1} = 92, \\ \sum_{i\in N} x_{i2} = 164, \\ x_{i1} + x_{i2} = 1\ (i \in N) \end{array} \right\} \\ &= \min \left\{ \sum_{i\in N}\sum_{j\in N} b_{ij} x_{i1} x_{j1} : \begin{array}{l} x_{i1} \in \{0,1\}\ (i \in N), \\ \sum_{i\in N} x_{i1} = 92 \end{array} \right\}, \end{aligned} \tag{6}$$

which coincides with BQOP (2). We note that x_{i2} serves as a slack variable for each binary variable x_{i1} $(i \in N)$ in (6).

3 Symmetry of the matrix $\boldsymbol{B}$

As mentioned in Section 1, the coefficient matrix $\boldsymbol{B}$ of the objective function of BQOP (2) satisfies (3), where $\mathcal{G}$ is a group of $n \times n$ permutation matrices satisfying (4). Each $n \times n$ permutation matrix $\boldsymbol{P}$ corresponds uniquely to a permutation π of N such that

$$\boldsymbol{P}\boldsymbol{x} = \boldsymbol{x}_\pi \text{ for every } \boldsymbol{x} \in \{0,1\}^n,$$

where $\boldsymbol{x}_\pi = (x_{\pi(1)}, \ldots, x_{\pi(n)})^T$. We denote this correspondence $\boldsymbol{P} \to \pi$ by $\pi = \varphi(\boldsymbol{P})$, which forms an isomorphism from the group of all $n \times n$ permutation matrices onto the symmetry group $\mathcal{S}_n$ of the permutations of N. We identify the group $\mathcal{G}$ with the permutation group $\{\varphi(\boldsymbol{P}) : \boldsymbol{P} \in \mathcal{G}\}$ since they are isomorphic. With this notation, we can rewrite the condition (3) as

$$\boldsymbol{x}_\pi^T \boldsymbol{B} \boldsymbol{x}_\pi = \boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x} \text{ for every } \boldsymbol{x} \in \{0,1\}^n \text{ and } \pi \in \mathcal{G}. \tag{7}$$

While software called Nauty [20] could be used to compute $\mathcal{G}$, we instead have computed $\mathcal{G}$ by a simple implicit enumeration of permutation matrices $\boldsymbol{P}$ satisfying (3), and found:

(a) $|\mathcal{G}| = 2048$

(b) The best known feasible solution $\boldsymbol{x}^*$ of BQOP (2) with the objective value $\bar{\zeta}$ is expanded to the set of feasible solutions $\{\boldsymbol{P}\boldsymbol{x}^* : \boldsymbol{P} \in \mathcal{G}\}$ with the common objective value $\bar{\zeta}$, where $|\{\boldsymbol{P}\boldsymbol{x}^* : \boldsymbol{P} \in \mathcal{G}\}| = 1028$; $\boldsymbol{P}_1\boldsymbol{x}^* = \boldsymbol{P}_2\boldsymbol{x}^*$ can occur for distinct $\boldsymbol{P}_1 \in \mathcal{G}$ and $\boldsymbol{P}_2 \in \mathcal{G}$.

4 Numerical experiments using Gurobi

We report computational results obtained by a general BQOP solver, Gurobi Optimizer (version 9.5.2) [15], for tai256c. We used the formulation (2) with x_1 fixed to 1. We will see in Section 5.2 that (2) has an optimal solution $\boldsymbol{x}$ with $x_1 = 1$. Numerical experiments were conducted on Intel(R) Xeon(R) CPU E7-8880 v4 (2.20GHz) processors using 32 threads with 2TB of RAM.

Computational experiments were performed iteratively with a starting value at each run initialized as the last updated incumbent solution, till the size of the brand-and-bound tree exceeded 2TB memory. The Gurobi parameter `Heuristics=0.8` was used in order to accelerate the heuristic search at the final run. We tested Gurobi parameter `PreQLinearize`, which controls a linearization of the model. At the root node, `PreQLinearize=1` provided the lower bound 1202900 and `PreQLinearize=2` the lower bound 384430.663. Since these values are lower than the lower bound 4117200 obtained with `PreQLinearize=-1`, the default value of `PreQLinearize` was used. Finally, we obtained a lower bound LB = 4181317 and an upper bound UB = 44780594 with 6 runs.

5 A branch and bound method for a given target lower bound

The optimal value ζ^* of BQOP (2) is currently unknown. Only a UB $\bar{\zeta} = 44759294 \geq \zeta^*$ and an LB $\underline{\zeta} = 44095032$ (with 1.48% gap from $\bar{\zeta}$) $\leq \zeta^*$ are known. To compute the optimal value ζ^*, we need to improve the UB and/or the LB.

Focusing on improving the LB, we present a BB method with the use of the Newton-bracketing (NB) method [18] for solving a Lagrangian doubly nonnegative (Lag-DNN) relaxation of subproblems of BQOP (2). The Lag-DNN relaxation is known to provide LBs with higher quality for BQOPs than the LP and the SDP relaxation [16, 17]. Before starting the BB method, a target LB, $\hat{\zeta}$, is first set such that $\underline{\zeta} = 44095032 < \hat{\zeta} \leq \bar{\zeta} = 44759294$. A target LB, $\hat{\zeta}$, is the desired value to attain. Ideally, we want set $\hat{\zeta} = \bar{\zeta}$ to confirm whether $\bar{\zeta}$ is the optimal value. But such a setting may be too ambitious, which requires much stronger computing power than the machine currently used. As a larger $\hat{\zeta}$ is set, the computational cost rapidly increases as we will see in Figure 1.

In Section 5.1, we describe a class of subproblems of BQOP (2) which appear in the enumeration tree generated by the BB method. For the lower bounding procedure, we use the Lag-DNN relaxation of a subproblem and the NB method for computing its optimal value which serves as an LB of the subproblem. We refer to [18] for details on them. Any upper bounding procedure is not incorporated. The branching procedure used in the BB method is presented in Section 5.2, and numerical results in Section 5.3.

Remark 5.1. The BB method with a fixed target LB above originally developed for large scale QAPs was successful to obtain improved lower bounds for some of the QAP instances in QAPLIB including sko100a,. . . , sko100f, tai80b, tai100b and tai150b. See [1]. The size of QAP tai256c, however, was too large to handle by the original BB method for the QAPs.

5.1 A class of subproblems of BQOP (2)

Let

$$\mathcal{S} \;=\; \left\{(I_0, I_1, F) : \begin{array}{l}\text{a partition of } N, \textit{ i.e., } I_0 \bigcup I_1 \bigcup F = N, \\ I_0,\ I_1 \text{ and } F \text{ are disjoint with each other}\end{array}\right\}.$$

Obviously, F is uniquely determined as $F = N\backslash\big(I_0 \bigcup I_1\big)$ for each $(I_0, I_1, F) \in \mathcal{S}$. Hence, F in the triplet (I_0, I_1, F) is redundant, and we frequently omit F for the simplicity of notation. For each $(I_0, I_1, F) \in \mathcal{S}$, we consider a subproblem of BQOP (2)

$$\begin{aligned}\mathrm{BQOP}(I_0, I_1) \;:\; \zeta(I_0, I_1) \;&=\; \min\left\{\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} : \begin{array}{l}\boldsymbol{x} \in \{0,1\}^n,\ \sum_{i=1}^n x_i = 92, \\ x_i = 0\ (i \in I_0),\ x_j = 1\ (j \in I_1)\end{array}\right\} \\ &=\; \min\left\{\boldsymbol{y}^T\boldsymbol{B}(I_0, I_1)\boldsymbol{y} : \begin{array}{l}\boldsymbol{y} \in \{0,1\}^F, \\ \sum_{i\in F} y_i = 92 - |I_1|\end{array}\right\},\end{aligned}$$

where

$\boldsymbol{y} \in \mathbb{R}^F$ denotes the subvector of $\boldsymbol{x}$ with elements x_i $(i \in F)$,

$\boldsymbol{B}(I_0, I_1) = \boldsymbol{B}_{FF} + 2 \times$ diagonal matrix of $\left(\sum_{k\in I_1} \boldsymbol{B}_{kF}\right)$,

$\boldsymbol{B}_{EF}$ = the $|E| \times |F|$ submatrix of $\boldsymbol{B}$ consisting of elements B_{ij} $(i \in E,\ j \in F)$.

Applying a penalty function method, we can convert $\mathrm{BQOP}(I_0, I_1)$ into a simple quadratic unconstrained binary optimization (QUBO)

$$\mathrm{QUBO}(I_0, I_1, \lambda) \;:\; \zeta(I_0, I_1, \lambda) \;=\; \min\left\{\boldsymbol{y}^T\boldsymbol{B}(I_0, I_1, \lambda)\boldsymbol{y} : \boldsymbol{y} \in \{0,1\}^F\right\},$$

where

$$\boldsymbol{y}^T\boldsymbol{B}(I_0, I_1, \lambda)\boldsymbol{y} \;=\; \boldsymbol{y}^T\boldsymbol{B}(I_0, I_1)\boldsymbol{y} + \lambda\big(\sum_{i\in F} y_i - 92 + |I_1|\big)^2 \text{ for every } \boldsymbol{y} \in \{0,1\}^F,$$

and $\lambda > 0$ is a Lagrangian or penalty parameter. It is straightforward to show that $\zeta(I_0, I_1, \lambda)$ converges monotonically to $\zeta(I_0, I_1)$ from below as $\lambda \to \infty$. For computing an LB of $\mathrm{BQOP}(I_0, I_1)$ in the BB method presented in this section, we applied the NB method to the DNN relaxation of $\mathrm{BQOP}(I_0, I_1, \lambda)$ with $\lambda = 1.0\mathrm{e}8/\ \|\ \boldsymbol{B}(I_0, I_1)\ \|$. See [18] for more details. We note that the matrix $\boldsymbol{B}(I_0, I_1, \lambda)$ satisfies the same symmetry property (3) presented in Section 5.2. In particular, BQOP (2) is converted into $\mathrm{QUBO}(\emptyset, \emptyset, \lambda)$.

5.2 The orbit branching

We discuss the orbit branching in [21, 24]. As mentioned in Section 3, $\boldsymbol{B} = \boldsymbol{B}(\emptyset, \emptyset)$ satisfies the symmetry property (7). This property is partially inherited to many $\boldsymbol{B}(I_0, I_1)$ $((I_0, I_1, F) \in \mathcal{S})$. Let $(I_0, I_1, F) \in \mathcal{S}$ be fixed. Assume in general that

$$\boldsymbol{y}_\pi^T\boldsymbol{B}(I_0, I_1)\boldsymbol{y}_\pi = \boldsymbol{y}^T\boldsymbol{B}(I_0, I_1)\boldsymbol{y} \text{ for every } \boldsymbol{y} \in \{0,1\}^F \text{ and } \pi \in \mathcal{G}(I_0, I_1) \tag{8}$$

holds, where $\mathcal{G}(I_0, I_1)$ is a permutation group. Then, we can define the equivalence relation $\sim$ on F by $i \sim j$ if $j = \pi_i$ for some $\pi \in \mathcal{G}(I_0, I_1)$. Let $\omega(i) = \{j \in F : j \sim i\}$ for every $i \in F$, and $\mathcal{O}(I_0, I_1) = \{\omega(i) : i \in F\}$. Each $o \in \mathcal{O}(I_0, I_1)$ is called an *orbit* of the permutation group $\mathcal{G}(I_0, I_1)$. We note that $\omega(i) = \omega(j)$ if $i \sim j$, and that $\mathcal{O}(I_0, I_1)$ can be the trivial identity permutation π such that $\boldsymbol{y}_\pi = \boldsymbol{y}$ for every $\boldsymbol{y} \in \{0,1\}^F$; hence $\mathcal{O}(I_0, I_1)$ can be defined consistently even if $\boldsymbol{B}(I_0, I_1)$ does not satisfy any symmetry. Let $\min(o)$ denote the minimum index of orbit o, which serves as a representative from o.

Assume that $o \in \mathcal{O}(I_0, I_1)$. Obviously, the feasible region of BQOP(I_0, I_1) is partitioned disjointly into the subset

$$\left\{ \boldsymbol{y} \in \{0,1\}^F : \sum_{i \in F} y_i = 92 - |I_1|\,,\ y_j = 0 \text{ for every } j \in o \right\},$$

which forms the feasible region of BQOP$(I_0 \bigcup o, I_1)$, and the subset

$$\left\{ \boldsymbol{y} \in \{0,1\}^F : \sum_{i \in F} y_i = 92 - |I_1|\,,\ y_j = 1 \text{ for some } j \in o \right\},$$

which forms the feasible region of BQOP$(I_0, I_1 \bigcup \{j\})$. We also see that

$$\min\{\boldsymbol{y}^T \boldsymbol{B}(I_0, I_1)\boldsymbol{y} : \boldsymbol{y} \in \text{ the latter subset}\}$$

is equivalent to $\min_{j \in o} \zeta(I_0, I_1 \bigcup \{j\})$. By the construction of orbit o, we know that all BQOP$(I_0, I_1 \bigcup \{j\})$ $(j \in o)$ are equivalent in the sense that they share a common optimal value $\zeta(I_0, I_1 \bigcup \min(o))$. Therefore, we can branch BQOP(I_0, I_1) into two sub BQOPs, BQOP$(I_0 \bigcup o, I_1)$ and BQOP$(I_0, I_1 \bigcup \min(o))$.

In general, $\mathcal{O}(I_0, I_1)$ consists of multiple orbits. How o is chosen from $\mathcal{O}(I_0, I_1)$ for branching of BQOP(I_0, I_1) into BQOP$(I_0 \bigcup o, I_1)$ and BQOP$(I_0, I_1 \bigcup \min(o))$ is an important issue to design an efficient branch and bound method. In our numerical experiment presented in Section 5.3,

- an orbit o is chosen from $\mathcal{O}(I_0, I_1)$ according to the average objective value of BQOP$(I_0, I_1 \bigcup \min(o))$ over all feasible solutions, so that the chosen orbit, o^*, attains the largest value. Then, we apply the branching of BQOP(I_0, I_1) into two subproblems BQOP$(I_0 \bigcup o^*, I_1)$ and BQOP$(I_0, I_1 \bigcup \{\min(o^*)\})$. Here the average objective value of BQOP(I_0, I_1) over all feasible solutions is computed as the objective value $\boldsymbol{x}^T \boldsymbol{B} \boldsymbol{x}$ with $x_i = 0$ $(i \in I_0)$, $x_j = 1$ $(j \in I_1)$ and $x_k = (92 - |I_1|)/|F|$ $(k \in F)$ for every $(I_0, I_1, F) \in \mathcal{S}$.

See [11, Section 5] for some other branching rules which can be combined with the NB method.

$\mathcal{G}(\emptyset, \emptyset) = \mathcal{G}$ has the single orbit $o = N = \{1, \ldots, n\}$. We branch BQOP$(\emptyset, \emptyset)$ into two subproblems BQOP$(N, \emptyset)$ and BQOP$(\emptyset, \{1\})$. Obviously, the former BQOP$(N, \emptyset)$ is infeasible. Table 1 summarizes the branching of the node BQOP$(\emptyset, \{1\})$ into BQOP$(\{2, 16, 17, 241\}, \{1\})$ and BQOP$(\emptyset, \{1, 2\})$, where orbit $\{2, 16, 17, 241\}$ is chosen from $\mathcal{O}(\emptyset, \{1\})$.

In addition to the branching rule mentioned above, we employ the simple breadth first search; the method to search the enumeration tree is not relevant to the computational efficiency since the incumbent objective value is fixed to the target LB $\hat{\zeta}$ and any upper

Table 1: A summary of branching of BQOP(I_0, I_1) with $I_0 = \emptyset$, $I_1 = \{1\}$ and $F = \{2, 3, \ldots, 256\}$ into BQOP($\{2, 16, 17, 241\}, \{1\}$) and BQOP($\emptyset, \{1, 2\}$). Here $F = \{2, 3, \ldots, 256\}$ is partitioned into 44 orbits, which consist of 21 orbits with size 8, 21 orbits with size 4, 1 orbit with size 2 and 1 orbit with size 1. The 44 orbits are listed according to the decreasing order of the average objective value of BQOP($\emptyset, \{1, \min(o)\}$) over all feasible solutions.

Orbit number	orbit	the size of orbit	the average objective value of BQOP($\emptyset, \{1, \min(o)\}$)
1	2 16 17 241	4	52655297.0
2	18 32 242 256	4	52567852.0
3	3 15 33 225	4	52524130.0
4	19 31 34 48 226 240 243 255	8	52515385.0
5	35 47 227 239	4	52502268.0
	...		...
30	87 91 102 108 166 172 183 187	8	52483274.0
31	9 129	2	52483139.0
32	72 74 117 125 149 157 200 202	8	52483097.0
33	25 130 144 249	4	52483097.0
	...		...
43	121 136 138 153	4	52481955.0
44	137	1	52481773.0

bounding procedure is not applied. At each node BQOP(I_0, I_1) of the enumeration tree, the NB method generates a sequence of intervals $[a_p, b_p]$ $(p = 1, 2, \ldots)$ satisfying the monotonicity: (1) a_p converges monotonically to an LB ν of BQOP(I_0, I_1) from below, and (2) b_p converges monotonically to ν from above. Thus, if $\hat{\zeta} \leq a_q$ holds for some q, we know that the LB ν to which the interval $[a_p, b_p]$ converges is not smaller than $\hat{\zeta}$, and BQOP(I_0, I_1) can be pruned. On the other hand, if $b_q < \hat{\zeta}$ holds for some q, we know the LB ν of BQOP(I_0, I_1) is smaller than $\hat{\zeta}$; hence the iteration can be stopped and branching to BQOP(I_0, I_1) can be applied. Therefore, the above properties (1) and (2) of the NB method work very effectively to increase the computational efficiency of the BB method. See Figure 3.

5.3 Numerical results

All the computations for numerical results reported in this section were performed using MATLAB 2022a on Mac Studio with Apple M1 Ultra CPU, 20 cores and 128 GB memory.

To choose a reasonable target LB $\hat{\zeta}$ which can be attained, we performed some preliminary numerical experiments to estimate the computational work. Given a target LB $\hat{\zeta}$, we construct the enumeration tree by the breadth first search as long as the number t_k of nodes at the depth k of the tree is smaller than 1000. Suppose that $t_0, t_1, \ldots, t_{\ell-1} < 1000 \leq t_\ell$; hence the full enumeration tree has been constructed up to the depth ℓ by the BB method. We start sampling at the depth ℓ and construct a random subtree to estimate the total number of nodes of the full enumeration tree. Let $\bar{t}_\ell = t_\ell$. At each depth $k \geq \ell$, we choose s_k nodes randomly from $\bar{t}_k$ active nodes for the next depth $(k + 1)$, where

$$s_k = \begin{cases} 100 & \text{if } \bar{t}_k \geq 500, \\ \bar{t}_k & \text{otherwise.} \end{cases}$$

Then, we apply the lower bounding procedure using the NB method to the selected s_k nodes and the branching procedure to the resulting r_k active nodes to generate a subset of the

nodes of the full enumeration tree at the depth $(k+1)$. Next, we let $\bar{t}_{k+1} = 2 * r_k$, which is the cardinality of the subset (the number of nodes in the subset) as each active node is branched into two child nodes. We continue this process till r_k attains 0. We may regard $2 * r_k/s_k = \bar{t}_{k+1}/s_k$ as the increasing rate of the nodes from the depth k to the depth $k+1$, and the total number of nodes of the full enumeration tree is estimated by

$$\sum_{k=1}^{\ell} t_k + \sum_{k>\ell} \hat{t}_k, \text{ where } \hat{t}_\ell = t_\ell, \ \hat{t}_{k+1} = (2r_k/s_k)\hat{t}_k \ (k \geq \ell). \tag{9}$$

Although this unrefined method seems simple, it provides useful information on whether a given target LB can be attained by the BB method on the computer used.

Figure 1 (A) illustrates how the estimated increasing rate changes in the eight cases where the target LB $\hat{\zeta} = 44759294$ (the best known objective value $\bar{\zeta}$), 44600000 (with 0.36% gap from $\bar{\zeta}$), 44400000 (0.80% gap), 44200000 (1.25% gap), 44150000 (1.36% gap), 44130000 (1.41% gap), 44120000 (1.43% gap) and 44100000 (1.47% gap) are taken. By applying the formula (9), the total number of nodes in the estimated tree is computed as in Table 2. The mean execution time to process one node, which mainly consists of execution time to solve the Lag-DNN relaxation of $\mathrm{B}(I_0, I_1)$ by the NB method, is about $50 \sim 60$ seconds, depending on the target LB. Obviously, the cases with $\hat{\zeta} = 44759294$ and 44600000 are very challenging. The cases with $\hat{\zeta} = 44200000$ and 44150000 could be easily processed by a supercomputer or a large scale clustered computer system. We show more details of the numerical results for $\hat{\zeta} = 44150000,\ 44130000,\ 44120000,\ 44100000$ cases below.

Figure 1 (B) compares the increasing rate of nodes of the full enumeration tree with its rough estimation described above for the four cases. Table 2 shows the total number of nodes of the enumeration tree in these cases. Figure 2 (A) displays how the number of nodes changes as the depth k increases, and Figure 2 (B) how the number of nodes with size 2 orbit changes. All other nodes are of the trivial single orbit N, except the root node having size 256 orbit as shown in Section 5.2 and the depth 1 node having size 4 orbit as observed in Table 1.

Recall that the NB method applied to a node $\mathrm{BQOP}(I_0, I_1)$ generates a sequence of interval $[a_p, b_p]$ $(p = 1, \ldots)$ which monotonically converges to an LB, ν, of $\mathrm{BQOP}(I_0, I_1)$. Hence, the iteration terminates either when $b_q < \hat{\zeta}$ occurs — $\mathrm{BQOP}(I_0, I_1)$ is turned out to be active in this case — or when $\hat{\zeta} \leq a_q$ occurs — $\mathrm{BQOP}(I_0, I_1)$ is pruned in this case. We see from Figure 3 that any tight LB is not necessary to decide whether $\mathrm{BQOP}(I_0, I_1)$ is active or to be pruned in most cases, particularly, in earlier stage of the BB method. This is an important feature of the NB method, which works effectively to increase the computational efficiency, when it is incorporated in the BB method.

Table 2: Total number (B) of nodes of the full enumeration tree and its estimation (A).

	Target LB							
	44759294	44600000	44400000	44200000	44150000	44130000	44120000	44100000
(A) Estimated tree	4.1e14	2.8e13	1.4e10	9.2e5	3.2e5	1.2e5	9.2e4	2.2e4
(B) Full tree					277304	102310	63554	23510

Figure 1: (A) Estimated increasing rate of nodes of the full enumeration tree. (B) Comparison of increasing rate of nodes of the full enumeration tree and its estimation.

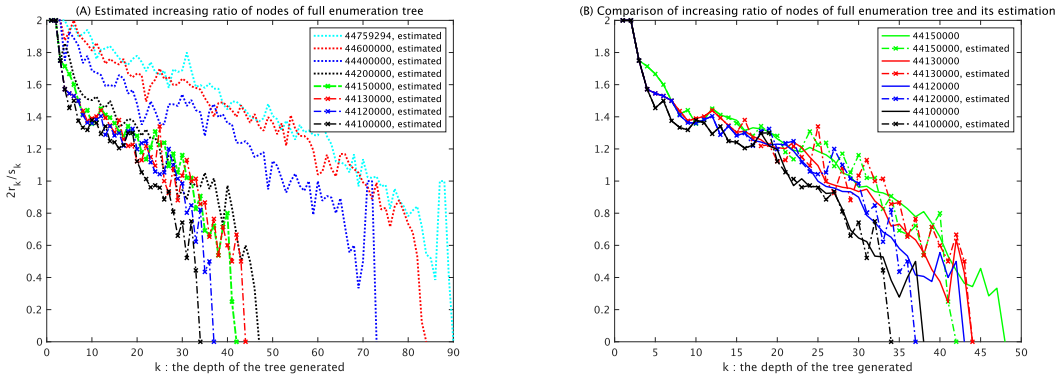


Figure 2: (A) The number of nodes of the enumeration tree at the depth k. (B) The number of nodes of the enumeration tree with size 2 orbit at the depth k.

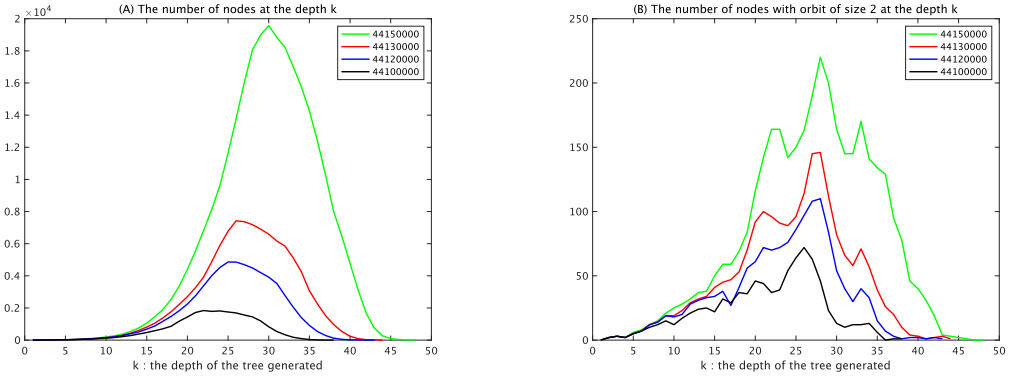
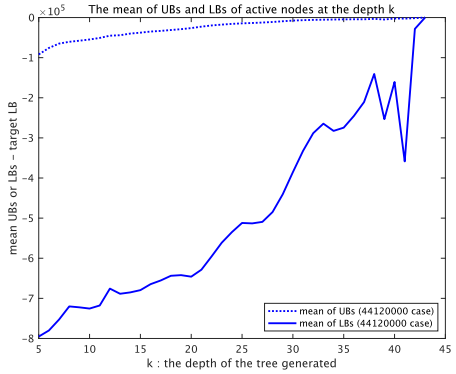
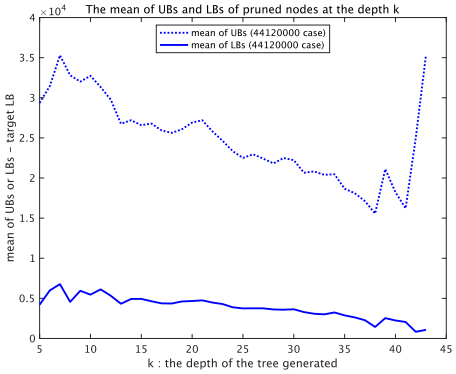


Figure 3: The mean of a_q (the blue solid line) and b_q (the blue dotted line) when the NB terminated at iteration q as $b_q < \hat{\zeta} = 441200000$ (*i.e.*, active node) — Case (A) or at iteration q as $\hat{\zeta} = 441200000 \leq a_q$ (*i.e.*, pruned node) — Case (B).





6 Concluding remarks

While we have focused on the BQOP converted from the QAP tai256c in this paper, it is straightforward to adapt the discussion of the paper to general BQOPs with a single cardinality constraint and general QUBOs which satisfy the symmetry property (4).

We have shown that the largest unsolved QAP instance tai256c can be converted into a simple BQOP with a single cardinality constraint, and further into a 256-dimensional QUBO with a penalty parameter λ whose optimal value converges to the optimal value of tai256c monotonically from below as $\lambda \to \infty$. Since the converted QUBO with dimension 256 is much simpler than the original tai256c that involves $256 \times 256 = 65536$ binary variables, and its dimension 256 is not so large, it might be natural to expect to solve it much easier than tai256c. We have found, however, that the QUBO is still quite difficult to solve. The difficulty essentially arises from the symmetry property (7) on the coefficient matrix $\boldsymbol{B}$ inherited from tai256c. We recall that the best known solution of tai256c is expanded to 1024 different feasible solutions of the QUBO with the best known objective value. When a branch and bound method is applied, all subproblems involving either of those feasible solutions need to be processed.

References

[1] M. Anjos. QAPLIB, https://www.miguelanjos.com/qaplib.

[2] K. Anstreicher, N. Brixius, Goux J-P., Linderoth, and J. Solving large quadratic assignment problems on computational grids. *Math. Program.*, 91:563–588, 2002.

[3] K. Anstreicher and H. Wolkowicz. On Lagrangian relaxation of quadratic matrix constraints. *SIAM J. Matrix Anal. Appl.*, 22(41-55), 2000.

[4] D. Brosch and E. de Klerk. Jordan symmetry reduction for conic optimization over the doubly nonnegative cone: theory and software. *Optim. Methods and Softw.*, page DOI: 10.1080/10556788.2021.2022146, 2022.

[5] E. Burkard, R, E. Cela, S.E. Karisch, and F. Rendl. QAPLIB – a quadratic assignment problem library. *J. Global Optim.*, 10:391–403, 1997.

[6] J. Clausen and M. Perregaard. Solving large quadratic assignment problems in parallel. *Comput. Optim. Appl.*, 8:111–127, 1997.

[7] D. T. Connolly. An improved annealing scheme for the qap. *Eur. J. Oper. Res.*, 46(93-100), 1990.

[8] Etine de Klerk and R. Sotirov. Exploiting group symmetry in semidefinite programming relaxations of the quadratic assignment problem. *Math. Program.*, 122:225–246, 2010.

[9] Etine de Klerk and R. Sotirov. Improved semidefinite programming bounds for quadratic assignment problems with suitable symmetry. *Math. Program.*, 133:75–91, 2012.

[10] M. Fischetti, M. Monaci, and D. Salvagnin. Three ideas for the quadratic assignment problem. *Oper. Res.*, 60(4):Published Online:1 Aug 2012, 2012.

[11] K. Fujii, N. Ito, S. Kim, M. Kojima, Y. Shinano, and K.C. Toh. Solving challenging large qaps. Technical Report arXiv:2101.09629, ZIB Report 21-20, 2021.

[12] L. M. Gambardella, E. D. Taillard, and M. Dorigo. Ant colonies for the qap. Technical report idsia-4-97, IDSIA, Lugano, Switzerland, 1997.

[13] P. C. Gilmore. Optimal and suboptimal algorithms for the quadratic assignment problem. *SIAM J. Appl. Math.*, 10:305–313, 1962.

[14] D. C. Goncalves, A. A. Pessoa, L. M. de A. Drummond, C. Bentes, and R. Farias. Solving the quadratic assignment problem on heterogeneous environment (cpus and gpus) with the application of level 2 reformulation and linearization technique. Technical report arxiv:1510.02065v1, 2015.

[15] Gurobi Optimization, LLC. Gurobi optimizer reference manual. https://www.gurobi.com, 2022.

[16] N. Ito, S. Kim, M. Kojima, A. Takeda, and K.C. Toh. Equivalences and differences in conic relaxations of combinatorial quadratic optimization problems. *J. Global Optim.*, 72(4):619–653, 2018.

[17] S. Kim, M. Kojima, and K. C. Toh. A Lagrangian-DNN relaxation: a fast method for computing tight lower bounds for a class of quadratic optimization problems. *Math. Program.*, 156:161–187, 2016.

[18] S. Kim, M. Kojima, and K. C. Toh. A Newton-bracketing method for a simple conic optimization problem. *Optim. Methods and Softw.*, 36:371–388, 2021.

[19] E. L. Lawler. The quadratic assignment problem. *Management Sci.*, 19:586–590, 1963.

[20] B. D. McKay. Nauty users guide (version 2:4). Technical report, Dept. Comp. Sci., Australian National University, 2010.

[21] J. Ostrowski, J. Linderoth, Rossi F., and S. Smriglio. Orbital branching. *Math. Program.*, 126:147–178, 2011.

[22] P. M. Pardalos, K. G. Ramakrishnan, M. G. C. Resende, and Y. Li. Implementation of a variance reduction-based lower bound in a branch-and-bound algorithm for the quadratic assignment problem. *SIAM J. Optim.*, 7:281–294, 1997.

[23] F.N. Permenter and P. A. Parrilo. Dimension reduction for semidefinite programs via jordan algebras. *Math. Program.*, 181:51–84, 2020.

[24] M. E. Pfetsch and Rehn T. A computational comparison of symmetry handling methods for mixed integer programs. *Math. Program. Comput.*, 11:37–93, 2019.

[25] J. Povh and F. Rendl. Copositive and semidefinite relaxations of the quadratic assignment problem. *Discrete Optim.*, 6:231–241, 2009.

[26] C. Roucairol. A parallel branch and bound algorithm for the quadratic assignment problem. *Discret. Appl. Math.*, 18:211–255, 1987.

[27] J. Skorin-Kapov. Tabu search applied to the quadratic assignment problem. . *ORSA J. Comput.*, 2:33–45, 1990.

[28] E. Tailard. Robust taboo search for the quadratic assignment problem. *Parallel Comput.*, 17:443–455, 1991.

[29] Q. Zhao, S.E. Karisch, F. Rendl, and H. Wolkowicz. Semidefinite programming relaxations for the quadratic assignment problem. *J. Comb. Optim.*, 2(71-109), 1998.