

# An Exceptionally Difficult Binary Quadratic Optimization Problem with Symmetry: a Challenge for The Largest Unsolved QAP Instance Tai256c

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

December 18, 2023

## Abstract

Tai256c is the largest unsolved quadratic assignment problem (QAP) instance in QAPLIB. It is known that QAP tai256c 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. As the BQOP is much simpler than the original QAP, the conversion increases the possibility to solve the QAP. Solving exactly the BQOP, however, is still very difficult. Indeed, a 1.48% gap remains between the best known upper bound (UB) and lower bound (LB) of the unknown optimal value. This paper shows that the BQOP admits a nontrivial symmetry, a property that makes the BQOP very hard to solve. The symmetry induces equivalent subproblems in branch and bound (BB) methods. To effectively improve the LB, we propose an efficient BB method that incorporates a doubly nonnegative relaxation, the standard orbit branching and a technique to prune equivalent subproblems. With this BB method, a new LB with 1.25% gap is successfully obtained, and computing an LB with 1.0% gap is shown to be still quite difficult.

## 1 Introduction

For a positive integer  $n$ , we let  $N = \{1, \dots, n\}$  represent a set of locations and also a set of facilities. Given  $n \times n$  symmetric matrices  $\mathbf{A} = [a_{ik}]$  and  $\mathbf{B} = [b_{j\ell}]$ , the quadratic assignment problem (QAP) is stated as

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

---

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

†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<sup>2</sup>IM), Zuse Institute Berlin, Takustrasse 7,14195 Berlin, Germany (shinano@zib.de).

where  $a_{ik}$  denotes the flow between facilities  $i$  and  $k$ ,  $b_{j\ell} = b_{\ell j}$  the distance between locations  $j$  and  $\ell$ , and  $(\pi(1), \dots, \pi(n))$  a permutation of  $1, \dots, n$  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 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 heuristic solutions. Heuristic methods such as tabu search, genetic method and simulated annealing have been developed for the QAP [10, 15, 37, 38]. Those methods frequently attain a near-optimal solution, which happens to be an exact optimal solution. 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  $\zeta^*$ . The second technique is to provide a lower bound (LB)  $\underline{\zeta}$  for  $\zeta^*$ . If  $\underline{\zeta} = \bar{\zeta}$  holds, then we can conclude that  $\underline{\zeta} = \zeta^* = \bar{\zeta}$ . Various relaxation methods [2, 16, 23, 35, 40] have been proposed for computing LBs. The two techniques mentioned above play as essential tools in the branch and bound (BB) method for QAPs [1, 9, 17, 23, 32, 36].

In this paper, we focus on the largest unsolved instance tai256c in QAPLIB [8]. The main purpose of this paper is to investigate the challenge to solve the instance and provide an improved lower bound. Nissofolk et al. [29, 30] showed that tai256c can be converted into a 256-dimensional binary quadratic optimization problem (BQOP) with a single cardinality constraint  $\sum_{i=1}^{256} x_i = 92$ .

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

See Section 2.2. Here each feasible solution  $\pi$  of QAP (1) is converted to a feasible solution  $\mathbf{x}$  of BQOP (2). They further transformed the BQOP (2) to a mixed integer convex quadratic program (MIQP) by the non-diagonal quadratic convex reformulation technique (NDQCR) developed in [20] for the quadratic knapsack problems. An LB = 44,095,032 (1.48% gap with respect to the best known UB 44,759,294) was obtained by applying CPLEX to the resulting MIQP, where CPLEX terminated in almost 8.5 days since the node limit exceeded. It demonstrated that the simple BQOP equivalent to tai256c remained still difficult to solve.

We show that BQOP (2) admits a nontrivial symmetry property, inherited from tai256c:

(a) The matrix  $\mathbf{B}$  satisfies

$$\mathbf{x}_\sigma^T \mathbf{B} \mathbf{x}_\sigma = \mathbf{x}^T \mathbf{B} \mathbf{x} \text{ for every } \mathbf{x} \in \{0, 1\}^n \text{ and } \sigma \in \mathcal{G}, \quad (3)$$

where  $\mathcal{G}$  denotes a subgroup of the symmetry group  $\mathcal{S}_n$  on  $\{1, \dots, n\}$  with  $|\mathcal{G}| = 2,048$ .

(b) BQOP (2) has at least 1,024 distinct feasible solutions with the best known UB 44,759,294.

The size of BQOP (2), 256, is not larger than quadratic unconstrained binary problem (QUBO) instances whose optimal solutions are known in the benchmark problem sets [39, 13, 27]. In fact, all QUBO instances with dimension less than 500 in the sets were solved exactly. BQOP (2) involves a single cardinality constraint  $\sum_{i=1}^{256} x_i = 92$  in binary variables  $x_i$  ( $i = 1, \dots, 256$ ), which is expected to make solving BQOP (2) easier in comparison to

QUBOs, since it considerably reduces the number of binary feasible solutions. Moreover, it is straightforward to transform BQOP (2) into a QUBO by adding a penalty term  $\lambda(\sum_{i=1}^{256} x_i - 92)^2$  to the objective quadratic form  $\mathbf{x}^T \mathbf{B} \mathbf{x}$  with a sufficiently large  $\lambda > 0$ .

Suppose that a BB method is applied to the BQOP with the best known UB. Then we (implicitly) construct an enumeration tree of its subproblems, where a subproblem is pruned whenever an LB of the subproblem not less than the best known UB of the BQOP (or an optimal solution of the subproblem) is obtained. In general, as the size of a subproblem involving a feasible solution with the best known UB becomes larger, the subproblem is harder to be pruned. As a result of the feature (b), the BB method cannot terminate in earlier stages since at least 1,024 distinct feasible solutions with the best known UB are distributed over subproblems.

To address the difficulty mentioned above in numerically solving BQOP (2) and to compute a new LB better than the known ones, this paper proposes

(c) a BB method to show that the unknown optimal value  $\zeta^*$  is not less than a given  $\hat{\zeta}$ .

Here a *target* LB  $\hat{\zeta}$  is chosen in the interval of the best known LB =  $\underline{\zeta}$  = 44,095,032 and UB =  $\bar{\zeta}$  = 44,759,294. We fix  $\hat{\zeta}$  and  $\bar{\zeta}$  before starting the BB method. The BB method terminates immediately after an LB not less than  $\hat{\zeta}$  is obtained. The proposed BB method implements the Lagrangian doubly nonnegative (Lag-DNN) relaxation [21, 22] as a lower bounding procedure for subproblems.

Using this method, we compute a new LB 44,200,000 (1.25% gap) in 39.2 days on a Mac Studio (20 cpu), and provide estimates on the amount of work (the number of subproblems to be solved and the execution time) for larger LBs. If we chose  $\hat{\zeta}$  to be the best known UB =  $\bar{\zeta}$ , then  $\bar{\zeta}$  would be proved to be the optimal value. In this case,  $2.6 \cdot 10^{12}$  days would be required to solve  $6.7 \cdot 10^{16}$  Lag-DNN relaxation subproblems of BQOP (2). This is not an accurate estimate and the execution time certainly depends on a BB method including a lower bounding procedure, a branching rule and the computer used. Nevertheless, it illustrates the extreme difficulty of solving the BQOP.

## Contribution of the paper and existing results

Our first contribution is to show and analyze the nontrivial symmetry property in BQOP (2) induced from tai256c. This BQOP is a simple, low-dimensional, and extremely difficult BQOP instance. As mentioned above, the symmetry property makes BQOP (2) hard to solve. As far as the authors are aware of, such a BQOP is not known.

The second contribution of this paper is a BB method to prove that the unknown optimal value  $\zeta^*$  is not less than a given target LB =  $\bar{\zeta}$ . A BB method with a target LB was originally developed for large scale QAPs, which was successful to obtain improved lower bounds for some of the QAP instances in QAPLIB including sko100a, . . . , sko100f, tai80b, tai100b and tai150b. See [26]. The size of QAP tai256c, however, was too large to handle by the original BB method for the QAPs. In the proposed method, we employ three effective techniques: The first one is the Lag-DNN relaxation [21, 22] subproblems of BQOP (2) for the lower bounding procedure. This relaxation is (almost) equivalent to a DNN relaxation [3, Theorem 2.6], which is known as one of the strongest (numerically tractable) conic relaxations for combinatorial optimization problems [19]. The second one is the standard orbit branching [31, 34] for reducing the size of the enumeration tree of subproblems to be solved. The third

one is a new technique for pruning equivalent subproblems of BQOP (2). This technique works effectively to improve the computational efficiency since the equivalence of some distinct subproblems occurs in the enumeration tree even after the orbit branching is applied. With this BB method, we have updated the known LB with 1.48% gap to a new LB with 1.25% gap in 39.2 days on a Mac Studio (20 cpu), and showed that an LB with 1.01% gap is still quite difficult to attain. This can also be regarded as an important contribution.

Exploiting symmetries of QAPs in their SDP relaxation was discussed in [11, 12, 33] (also [6] in their DNN relaxation). However, those results are not relevant to the subsequent discussion of this paper.

This paper is a revised version of an unpublished technical report [14].

## Outline of the paper

In Section 2, we introduce key components that will be utilized in the subsequent sections, including the conversion of QAP (1) into BQOP (2) satisfying the symmetry property (3), the Lag-DNN relaxation and the Newton-bracketing (NB) method for solving the relaxation. In Section 3, we present computational results using DABS (Diverse Adaptive Bulk Search, a genetic algorithm-based search algorithm) [28] and Gurobi Optimizer (version 11.0.0) [18]. We show that state-of-the-art BQOP solver Gurobi could not improve the known LB  $\zeta$ , demonstrating the difficulty of the problem. In Section 4, we describe the BB method (c) in detail and report numerical results. We conclude the paper in Section 5.

## 2 Preliminaries

### 2.1 Notation and symbols

Let  $N = \{1, \dots, n\}$ . We are mainly concerned with the BQOP (2) induced from tai256c. In that case,  $n = 256$ . Let  $\mathbb{R}^n$  denotes the  $n$ -dimensional Euclidean space of column vectors  $\mathbf{x} = (x_1, \dots, x_n)$ , and  $\mathbb{R}_+^n$  its nonnegative orthant  $\{\mathbf{x} \in \mathbb{R}^n : x_i \geq 0 \ (i \in N)\}$ . For  $\mathbf{x} \in \mathbb{R}^n$ ,  $\mathbf{x}^T$  is the transpose of  $\mathbf{x}$ . For each permutation  $\sigma$  of  $N$  and each  $\mathbf{x} \in \{0, 1\}^n \subset \mathbb{R}_+^n$ ,  $\mathbf{x}_\sigma$  denotes  $\mathbf{x}' \in \{0, 1\}^n$  such that  $x'_j = x_{\sigma(i)}$  ( $i \in N$ ).  $\mathbb{R}^{m \times n}$  denotes the linear space of  $m \times n$  matrices.  $\mathbb{S}^n$  denotes the linear space of  $n \times n$  symmetric matrices with the inner product  $\mathbf{A} \bullet \mathbf{B} = \sum_{i \in N} \sum_{j \in N} A_{ij} B_{ij}$  for  $\mathbf{A}, \mathbf{B} \in \mathbb{S}^n$ ,  $\mathbb{S}_+^n$  the convex cone of positive semidefinite matrices in  $\mathbb{S}^n$ , and  $\mathbb{N}^n$  the convex cone of matrices with nonnegative elements in  $\mathbb{S}^n$ .

### 2.2 Conversion from tai256c to BQOP (2)

QAP (1) can be rewritten with an  $n \times n$  matrix variable  $\mathbf{X}$  as a quadratic optimization problem:

$$\zeta^* = \inf \{(\mathbf{A}\mathbf{X}\mathbf{B}) \bullet \mathbf{X} : \mathbf{X} \in \Pi\}, \quad (4)$$

where  $\Pi$  denotes the set of  $n \times n$  permutation matrix. We note that each feasible solution  $\pi$  of QAP (1), which is a permutation of  $N$ , corresponds to an  $\mathbf{X} \in \Pi$  such that  $X_{ij} = 1$  iff  $\pi(i) = j$  for every  $(i, j) \in N \times N$ . In case of tai256c,  $n = 256$  and  $\mathbf{A}$  can be represented

as  $\mathbf{A} = \mathbf{f}\mathbf{f}^T$  for  $\mathbf{f} = \begin{pmatrix} \mathbf{e} \\ \mathbf{0} \end{pmatrix}$ , where  $\mathbf{e}$  denotes the 92-dimensional column vector of 1's. Hence, the objective function  $(\mathbf{A}\mathbf{X}\mathbf{B}) \bullet \mathbf{X}$  of QAP (4) can be rewritten as

$$(\mathbf{A}\mathbf{X}\mathbf{B}) \bullet \mathbf{X} = (\mathbf{f}\mathbf{f}^T\mathbf{X}\mathbf{B}) \bullet \mathbf{X}^T = (\mathbf{X}^T\mathbf{f})^T\mathbf{B}(\mathbf{X}^T\mathbf{f}).$$

We then see that  $\mathbf{x} = \mathbf{X}^T\mathbf{f} \in \{0, 1\}^n$  and  $\sum_{i \in N} x_i = 92$  for every  $\mathbf{X} \in \Pi$ . Conversely, if  $\mathbf{x} \in \{0, 1\}^n$  and  $\sum_{i \in N} x_i = 92$ , then  $\mathbf{x} = \mathbf{X}^T\mathbf{f}$  for some  $\mathbf{X} \in \Pi$ . Therefore, QAP (4) (hence QAP (1)) is equivalent to BQOP (2).

### 2.3 Symmetry of the matrix $\mathbf{B}$

We computed  $\mathcal{G}$  by a simple implicit enumeration of permutations  $\sigma$  satisfying (3), and found:

- $|\mathcal{G}| = 2,048$
- The best known feasible solution  $\mathbf{x}^*$  of BQOP (2) with the objective value = the best known UB 44,759,294 for tai256c is expanded to the set of feasible solutions  $\{(\mathbf{x}^*)_\sigma : \sigma \in \mathcal{G}\}$  with the common objective value, where  $|\{(\mathbf{x}^*)_\sigma : \sigma \in \mathcal{G}\}| = 1024$ ;  $(\mathbf{x}^*)_\sigma = (\mathbf{x}^*)_{\sigma'}$  can occur for distinct  $\sigma \in \mathcal{G}$  and  $\sigma' \in \mathcal{G}$ .

Computing the group  $\mathcal{G}$  of permutations can also be carried out with a software called Nauty [24]. The symmetry of  $\mathbf{B}$  is utilized in orbit branching (Section 4.2) and eliminating equivalent subproblems (Section 4.5) which are implemented in the BB method for solving BQOP (2).

### 2.4 A Lagrangian doubly nonnegative (Lag-DNN) relaxation of a linearly constrained QOP in binary variables

We briefly describe a Lag-DNN relaxation, which was originally proposed in [21] combined with the the bisection-projection (BP) method for computing LBs of linearly constrained QOPs in binary variables. More recently, the BP method was further developed to the Newton-bracketing (NB) method [22]. In our proposed BB method, the NB method is incorporated for computing LBs of BQOP (2). BQOP (2) as well as its subproblem BQOP( $I_0, I_1$ ) presented in Section 4.1, are special cases of a linearly constrained QOP in binary variables.

$$\zeta = \inf \{ \mathbf{u}^T \mathbf{C} \mathbf{u} : \mathbf{u} \in \{0, 1\}^n, \mathbf{F} \mathbf{u} - \mathbf{b} s = \mathbf{0}, s = 1 \}, \quad (5)$$

where  $\mathbf{C} \in \mathbb{S}^n$ ,  $\mathbf{F} \in \mathbb{R}^{m \times n}$  and  $\mathbf{b} \in \mathbb{R}^m$ .

BQOP (5) is rewritten to strengthen the DNN relaxation by introducing slack variable vector  $\mathbf{v} \in \{0, 1\}^n$  for  $\mathbf{u} \in \{0, 1\}^n$ :

$$\zeta = \inf \left\{ \mathbf{u}^T \mathbf{C} \mathbf{u} : \begin{array}{l} (\mathbf{u}, \mathbf{v}, s) \geq \mathbf{0}, (u_j + v_j - s)^2 = 0 \ (j \in N), \\ u_j v_j = 0 \ (j \in N), (\mathbf{F} \mathbf{u} - \mathbf{b} s)^T (\mathbf{F} \mathbf{u} - \mathbf{b} s) = 0, s^2 = 1 \end{array} \right\} \quad (6)$$

Introducing a penalty function (or a Lagrange function)

$$\begin{aligned} L(\mathbf{u}, \mathbf{v}, s, \lambda) = & \mathbf{u}^T \mathbf{C} \mathbf{u} + \lambda \left( \sum_{j \in N} (u_j + v_j - s)^2 + \sum_{j \in N} u_j v_j \right. \\ & \left. + (\mathbf{F} \mathbf{u} - \mathbf{b} s)^T (\mathbf{F} \mathbf{u} - \mathbf{b} s) \right) \text{ for every } (\mathbf{u}, \mathbf{v}, s, \lambda) \geq \mathbf{0}, \end{aligned}$$

we consider a simple QOP

$$\zeta(\lambda) = \inf \{ L(\mathbf{u}, \mathbf{v}, s, \lambda) : (\mathbf{u}, \mathbf{v}, s) \geq \mathbf{0}, s^2 = 1 \},$$

where  $\lambda \geq 0$  denotes a penalty parameter (or a Lagrangian multiplier). We can prove that  $\zeta(\lambda)$  converges to  $\zeta$  as  $\lambda \rightarrow \infty$ . See [21, Lemma 3]. Since  $L(\mathbf{u}, \mathbf{v}, s, \lambda)$  is a quadratic form in  $(\mathbf{u}, \mathbf{v}, s) \in \mathbb{R}^{2n+1}$  for each fixed  $\lambda$  and linear in  $\lambda$ , we can represent  $L(\mathbf{u}, \mathbf{v}, s, \lambda)$  as

$$L(\mathbf{u}, \mathbf{v}, s, \lambda) = \mathbf{Q}(\lambda) \bullet \left( \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ s \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ s \end{pmatrix}^T \right) \text{ for every } (\mathbf{u}, \mathbf{v}, s, \lambda) \geq \mathbf{0},$$

where  $\mathbf{Q}_\lambda = (\mathbf{Q}^1 + \lambda \mathbf{Q}^2)$  and  $\mathbf{Q}^1 \in \mathbb{S}^{2n+1}$  and  $\mathbf{Q}^2 \in \mathbb{S}_+^{2n+1} + \mathbb{N}^{2n+1}$ . Thus

$$\zeta(\lambda) = \inf \left\{ \mathbf{Q}_\lambda \bullet \left( \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ s \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ s \end{pmatrix}^T \right) : \begin{array}{l} (\mathbf{u}, \mathbf{v}, s) \geq \mathbf{0}, \\ \mathbf{H} \bullet \left( \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ s \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ s \end{pmatrix}^T \right) = 1 \end{array} \right\},$$

where  $\mathbf{H}$  denotes the  $(2n+1) \times (2n+1)$  matrix with elements  $H_{ij} = 0$  ( $(i, j) \neq (2n+1, 2n+1)$ ) and  $H_{2n+1, 2n+1} = 1$ . By replacing  $\begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ s \end{pmatrix} \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \\ s \end{pmatrix}^T$  by a matrix variable  $\mathbf{W} \in \mathbb{S}^{2n+1}$ , we obtain a Lag-DNN relaxation of BQOP (5).

$$\eta(\lambda) = \inf \{ \mathbf{Q}_\lambda \bullet \mathbf{W} : \mathbf{W} \in \mathbb{K}, \mathbf{H} \bullet \mathbf{W} = 1 \}, \quad (7)$$

where  $\mathbb{K} = \mathbb{S}_+^{2n+1} \cap \mathbb{N}^{2n+1}$  denotes the  $(2n+1)$ -dimensional DNN cone. We note that a DNN relaxation of BQOP (6) can be written as

$$\eta = \inf \{ \mathbf{Q}^1 \bullet \mathbf{W} : \mathbf{W} \in \mathbb{K}, \mathbf{Q}^2 \bullet \mathbf{W} = 0, \mathbf{H} \bullet \mathbf{W} = 1 \}.$$

The Lag-DNN relaxation (7) is almost as strong as the DNN relaxation above in the sense that  $\eta \geq \eta(\lambda)$  converges monotonically to  $\eta$  as  $\lambda \rightarrow \infty$ . See [3, Theorem 2.6]. In Section 4, which presents numerical results for the BB method applied to BQOP (2), a value of  $\lambda = 10^8 / \|\mathbf{Q}^1\|$  is used.

## 2.5 The Newton-bracketing (NB) Method [3, 22]

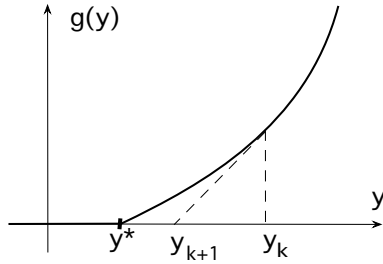
Given  $b_0 > \eta(\lambda)$ , the NB Method applied to (7) generates a sequence of intervals  $[a_k, b_k]$  ( $k = 0, 1, \dots$ ) which converges to  $\eta(\lambda)$  monotonically. In this section, we present how the sequence is generated briefly. For more details, we refer to [4, Section 3], [3, Section 4] and [22]. Throughout this section,  $\lambda > 0$  is fixed. The dual of DNN problem (7) can be written as

$$y^* = \sup \{ y : \mathbf{Q}_\lambda - \mathbf{H}y = \mathbf{Y} \in \mathbb{K}^* \}, \quad (8)$$

where  $\mathbb{K}^* = \mathbb{S}_+^{2n+1} + \mathbb{N}^{2n+1}$  (the dual of the  $(2n+1)$ -dim. DNN cone  $\mathbb{K} = \mathbb{S}_+^{2n+1} \cap \mathbb{N}^{2n+1}$ ). By the strong duality (see [3, Lemma 2.3]),  $y^* = \eta(\lambda)$  holds. Define

$$g(y) = \inf \{ \|\mathbf{Q}_\lambda - \mathbf{H}y - \mathbf{Y}\| : \mathbf{Y} \in \mathbb{K}^* \} \text{ for every } y_0 \in \mathbb{R}.$$

Figure 1: The convex function  $g : \mathbb{R} \rightarrow \mathbb{R}_+$  and the NB method.



Obviously,  $g(y) \geq 0$  for every  $y \in \mathbb{R}$ ,  $\mathbf{Y} = \mathbf{Q}_\lambda - \mathbf{H}y \in \mathbb{K}^*$  (i.e.,  $(y, \mathbf{Y})$  is a feasible solution of (8)) if and only if  $g(y) = 0$ , and  $g(y^*) = 0$ . Since  $\mathbf{H} \in \mathbb{K}^*$ , we also know that  $g(y) = 0$  if  $y \leq y^*$ . Hence,  $y^*$  corresponds to the maximum zero of  $g$ . Furthermore,  $g : (y^*, \infty) \rightarrow \mathbb{R}$  is convex and continuously differentiable ([3, Lemma 4.1]). Therefore, we can generate a sequence  $\{b_k = y_k\}$  ( $k = 0, 1, \dots$ ) converging monotonically to  $\eta(\lambda)$  by applying the Newton method with a given initial point  $y_0 = b_0 > y^*$ . See Figure 1. The function value  $g(y)$  and the derivative  $g'(y)$  at  $y = y_k > y^*$  is not given explicitly but can be computed by the accelerated proximal gradient (APG) method [5]. This method also computes  $(\mathbf{Y}_k^1, \mathbf{Y}_k^2) \in \mathbb{S}^{2n+1} \times \mathbb{N}^{2n+1}$  which (approximately) satisfies  $\mathbf{Q}_\lambda - \mathbf{H}y_k = \mathbf{Y}^1 + \mathbf{Y}^2$ . and  $\mathbf{Y}^1 \in \mathbb{S}_+^{2n+1}$ . We obtain  $a_k \leq y^*$  by letting

$$a_k = y_k + (n + 1) \min\{0, \text{the minimum eigenvalue of } \mathbf{Y}_k^1\}.$$

See [4, Lemma 3.1]. In each iteration of the NB method, most of its execution time is consumed to evaluate  $g(y_k)$ ,  $g'(y_k)$ ,  $\mathbf{Y}_k^1$  and  $\mathbf{Y}_k^2$  by the APG method.

### 3 Numerical experiments using DABS and Gurobi

We report computational results obtained by DABS (Diverse Adaptive Bulk Search), a genetic algorithm-based search algorithm for QUBO [28], and a general BQOP solver Gurobi Optimizer (version 11.0.0) [18], for BQOP (2). Numerical experiments were conducted on Intel(R) Xeon(R) Gold 6246 (3.30 GHz) processors using 48 threads with 1.5TB of RAM.

BQOP (2) with a single cardinality constraint  $\sum_{i=1}^n x_i = 92$  can be transformed to a simple QUBO by adding the penalty term  $\lambda(\sum_{i=1}^n x_i - 92)^2$  to the objective function  $\mathbf{x}^T \mathbf{Q} \mathbf{x}$  and removing the constraint, where  $\lambda > 0$  is a penalty parameter. We applied DABS to the resulting QUBO with  $\lambda = 10^7$ . DABS attained a feasible solution with the objective value 44,759,294, which coincides with the best known upper bound [7] for tai256c, within a few seconds. Moreover, the feasible solution computed is contained in the 1,024 known ones with the same objective value 44,759,294 (see Section 2.3).

We also applied Gurobi to the BQOP reformulation (2) of tai256c. Gurobi is state-of-the-art as a solver for general BQOPs. In a benchmark project conducted by Mittelmann [25], several solvers are tested for BQOPs. The results there indicate that Gurobi is the fastest in solving those BQOPs. Gurobi has been enhanced as a BQOP solver, and its performance and efficiency in this specific area has been constantly improved. In Section 4.2, we will see

from the symmetry of  $\mathbf{B}$  that BQOP (2) has an optimal solution  $\mathbf{x}$  with  $x_1 = 1$ . When Gurobi was applied to BQOP (2) with  $x_1$  fixed to 1, any information on the symmetry of  $\mathbf{B}$  could not be utilized.

To experiment with Gurobi, some parameters were needed to be decided, in particular `MIPFocus`, and `PreQLinearize`. The parameter `MIPFocus` controls solution strategy of branch-and-bound. We chose `MIPFocus=3` which focuses on computing the LB. The parameter `PreQLinearize` controls presolving for BQOPs. More precisely, the parameter `PreQLinearize=0` adds neither any variables nor constraints, but it performs adjustments on quadratic objective functions to make them positive semidefinite. The parameter values `PreQLinearize=1` and `PreQLinearize=2` attempt to linearize quadratic constraints or a quadratic objective, replacing quadratic terms with linear terms with additional variables and linear constraints.

In the first step, we examined the value of `PreQLinearize` by comparing the LB obtained at the root node. `PreQLinearize=0` provided the best LB 41,172,797, `PreQLinearize=1` the LB 10,759,778, and `PreQLinearize=2` the LB 3,987,504. As the best LB 41,172,797 was obtained by `PreQLinearize=0`, we adopted this setting for our experiments.

In the second step, we examined 1,024 potentially best solutions generated by the symmetry of BQOP (2). We determine the best initial solution among them by executing branch-and-bound with a time limit of 1 hour and comparing the LBs.

In the final step, we executed branch-and-bound with `MIPFocus=3`, `PreQLinearize=0` and the initial solution chosen in the second step. We had to set a time limit of 60,000 seconds due to limitations in computational resources. We finally obtained an LB of 41,669,052 using Gurobi, which generated 12,518,148 nodes.

The LB 41,669,052 obtained is significantly lower than the LBs we will present in Section 4. This suggests that, despite its rapid improvement in solving BQOP problems, the state-of-the-art general BQOP solver is still considerably less effective in solving BQOP (2) or improving its LB.

There are two primary factors for Gurobi’s poor performance in solving BQOP (2). Recall that the LB 41,172,797 (8.01% gap) was obtained at the root node problem. We applied the Lag-DNN relaxation to the same problem, and computed an LB 43,881,304 (1.96% gap) of the problem by the NB method in less than 10 minutes. This shows that the LB procedure incorporated in Gurobi is much weaker than the Lag-DNN relaxation. Another noteworthy factor contributing to Gurobi’s poor performance is its inability to utilize the symmetry property of BQOP (2). This also played a role in the generation of a huge number of nodes, reaching 12,518,148, by Gurobi to compute the LB 41,669,052 (6.90%) which is still much smaller than the LB 43,881,304 (1.96%) of the root node computed by the Lag-DNN relaxation. This sharply contrasts with our improved BB method, which generated 1,077,353 nodes to compute the LB 44,200,000 (1.25%) in 39.2 days. See Section 4.5.

## 4 A branch and bound method for a given target lower bound

The optimal value  $\zeta^*$  of BQOP (2) remains unknown, with an LB  $\underline{\zeta} = 44,095,032 \leq \zeta^* \leq$  a UB  $\bar{\zeta} = 44,759,294$  exhibiting a 1.48% gap between them. We need to improve the UB and/or the LB to compute the optimal value  $\zeta^*$ . To improve the LB, we propose



a 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. See Sections 2.4 and 2.5 for the Lag-DNN relaxation and the NB method, respectively. Any upper bounding procedure is not incorporated. Before the start of the BB method, a target LB,  $\hat{\zeta}$ , is first set such that  $\underline{\zeta} = 44,095,032 < \hat{\zeta} \leq \bar{\zeta} = 44,759,294$ . A target LB,  $\hat{\zeta}$ , is the desired value to obtain. Ideally, we want to set  $\hat{\zeta} = \bar{\zeta}$  to confirm whether  $\bar{\zeta}$  is the optimal value. But such a setting may be too ambitious, and 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 Section 4.3.

We describe a class of subproblems of BQOP (2) which appear in the enumeration tree generated by the BB method in Section 4.1, and the branching procedure used in the BB method in Section 4.2. Before presenting numerical results on the BB method in Section 4.4, we provide a preliminary estimate for the amount of work (the number of nodes to generate and execution time) to attain given target LBs by the BB method in Section 4.3. Based on this estimation, we choose some reasonable target LBs for the numerical experiment on the BB method whose results are reported in Section 4.4. In Section 4.5, we propose a new technique using the equivalence of subproblems of BQOP (2) to improve the performance of the BB method, and show that the improved BB method is twice efficient than the original BB method. A new LB  $\hat{\zeta} = 44,200,000$  (1.25% gap) is also attained by the improved BB method in 39.2 days.

## 4.1 A class of subproblems of BQOP (2)

Let

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

Obviously,  $F$  is uniquely determined by  $I_0$  and  $I_1$  as  $F = N \setminus (I_0 \cup I_1)$  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)

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

where

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

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

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

For example, if  $F = \{1, \dots, \ell\}$  and  $I_1 \cup I_0 = \{\ell + 1, \dots, n\}$ , then  $\mathbf{B}(I_0, I_1)$  is an  $\ell \times \ell$  matrix with elements  $B(I_0, I_1)_{ij}$  ( $i = 1, \dots, \ell, j = 1, \dots, \ell$ ) such that

$$B(I_0, I_1)_{ij} = \begin{cases} B_{ij} & \text{if } i \neq j, \\ B_{ii} + 2 \sum_{k \in I_1} B_{ki} & \text{if } i = j. \end{cases}$$

For computing an LB of  $\text{BQOP}(I_0, I_1)$  in the BB method, we applied the NB method to the Lag-DNN relaxation of  $\text{BQOP}(I_0, I_1)$  with  $\lambda = 10^8 / \|\mathbf{B}(I_0, I_1)\|$ .

## 4.2 Orbit branching

We discuss the orbit branching technique as in [31, 34]. As mentioned in Section 1,  $\mathbf{B} = \mathbf{B}(\emptyset, \emptyset)$  satisfies the symmetry property (3). This property is partially inherited by many  $\mathbf{B}(I_0, I_1)$  ( $(I_0, I_1, F) \in \mathcal{S}$ ). Let  $(I_0, I_1, F) \in \mathcal{S}$  be fixed. Assume in general that

$$\mathbf{y}_\sigma^T \mathbf{B}(I_0, I_1) \mathbf{y}_\sigma = \mathbf{y}^T \mathbf{B}(I_0, I_1) \mathbf{y} \text{ for every } \mathbf{y} \in \{0, 1\}^{|F|} \text{ and } \sigma \in \mathcal{G}(I_0, I_1) \quad (9)$$

holds, where  $\mathcal{G}(I_0, I_1)$  is a group of permutations of  $F$ . Let  $\omega(i) = \{j \in F : j = \sigma_i \text{ for some } \sigma \in \mathcal{G}(I_0, I_1)\}$  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 group  $\mathcal{G}(I_0, I_1)$ . Let  $\min(o)$  denote the minimum index of orbit  $o$ , which serves as a representative for  $o$ . Assume that  $o \in \mathcal{O}(I_0, I_1)$ . Then we know that all  $\text{BQOP}(I_0, I_1 \cup \{j\})$  ( $j \in o$ ) are equivalent in the sense that they share a common optimal value  $\zeta(I_0, I_1 \cup \min(o))$ . Therefore, we can branch  $\text{BQOP}(I_0, I_1)$  to two sub BQOPs,  $\text{BQOP}(I_0 \cup o, I_1)$  and  $\text{BQOP}(I_0, I_1 \cup \min(o))$ .

In general,  $\mathcal{O}(I_0, I_1)$  consists of multiple orbits. Selecting an appropriate  $o$  from  $\mathcal{O}(I_0, I_1)$  for branching of  $\text{BQOP}(I_0, I_1)$  to  $\text{BQOP}(I_0 \cup o, I_1)$  and  $\text{BQOP}(I_0, I_1 \cup \min(o))$  is an important issue to design an efficient branch and bound method. In our numerical experiment presented in Section 4.4,

- an orbit  $o$  is chosen from  $\mathcal{O}(I_0, I_1)$  according to the average objective value of  $\text{BQOP}(I_0, I_1 \cup \min(o))$  over all feasible solutions, so that the chosen orbit,  $o^*$ , attains the largest value. Then, we apply the branching of  $\text{BQOP}(I_0, I_1)$  to two subproblems  $\text{BQOP}(I_0 \cup o^*, I_1)$  and  $\text{BQOP}(I_0, I_1 \cup \{\min(o^*)\})$ . Here the average objective value of  $\text{BQOP}(I_0, I_1)$  over all feasible solutions is computed as the objective value  $\mathbf{x}^T \mathbf{B} \mathbf{x}$  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  $(I_0, I_1, F) \in \mathcal{S}$ .

$\mathcal{G}(\emptyset, \emptyset) = \mathcal{G}$  has the single orbit  $o = N = \{1, \dots, n\}$ . We branch  $\text{BQOP}(\emptyset, \emptyset)$  into two subproblems  $\text{BQOP}(N, \emptyset)$  and  $\text{BQOP}(\emptyset, \{1\})$ . Obviously, the former  $\text{BQOP}(N, \emptyset)$  is infeasible. Table 1 summarizes the branching of the node  $\text{BQOP}(\emptyset, \{1\})$  to  $\text{BQOP}(\{2, 16, 17, 241\}, \{1\})$  and  $\text{BQOP}(\emptyset, \{1, 2\})$ , 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 bounding procedure is not applied. At each node  $\text{BQOP}(I_0, I_1)$  of the enumeration tree, the NB method generates a sequence of intervals  $[a_p, b_p]$  ( $p = 1, 2, \dots$ ) satisfying a monotonicity property: (1)  $a_p$  converges monotonically to an LB  $\nu$  of  $\text{BQOP}(I_0, I_1)$  from below, and (2)  $b_p$  converges monotonically to  $\nu$  from above. Thus, if  $\hat{\zeta} \leq a_q$  holds for some  $q$ , we know that the LB  $\nu$  to which the interval  $[a_p, b_p]$  converges is not smaller than  $\hat{\zeta}$ , and  $\text{BQOP}(I_0, I_1)$  can

Table 1: A summary of branching of  $\text{BQOP}(I_0, I_1)$  with  $I_0 = \emptyset$ ,  $I_1 = \{1\}$  and  $F = \{2, 3, \dots, 256\}$  to  $\text{BQOP}(\{2, 16, 17, 241\}, \{1\})$  and  $\text{BQOP}(\emptyset, \{1, 2\})$ . Here  $F = \{2, 3, \dots, 256\}$  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  $\text{BQOP}(\emptyset, \{1, \min(o)\})$  over all feasible solutions.

Orbit number	Orbit	The size of orbit	The average objective value of $\text{BQOP}(\emptyset, \{1, \min(o)\})$
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

be pruned. On the other hand, if  $b_q < \hat{\zeta}$  holds for some  $q$ , we know the LB  $\nu$  of  $\text{BQOP}(I_0, I_1)$  is smaller than  $\hat{\zeta}$ ; hence the iteration can be stopped and branching to  $\text{BQOP}(I_0, I_1)$  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.

### 4.3 Estimating the total number of nodes generated by the BB method

All the computations for numerical results reported in this section and the next two sections were performed using MATLAB 2022a on a Mac Studio with Apple M1 Ultra CPU, 20 cores and 128 GB memory. For the parallel computation, we solved Lag-DNN relaxations of 20 subproblems  $\text{BQOP}(I_0, I_1)$  in parallel by the NB method with the ‘parfor’ loop of MATLAB.

To choose a reasonable target LB  $\hat{\zeta}$  which can be attained by the BB method, we performed preliminary numerical experiments to estimate the computational work. Given a target LB  $\hat{\zeta}$ , we construct an 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  $t_0, t_1, \dots, t_{\ell-1} < 1000 \leq t_\ell$ ; hence the full enumeration tree has been constructed up to the depth  $\ell$  by the BB method. We start sampling at the depth  $\ell$  and construct a random subtree to estimate the total number of nodes in 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

Table 2: Estimation of the work of the BB method described in Sections 4.1 and 4.2. For each target LB  $\hat{\zeta}$ , we applied 5 different random sampling of  $s_k$  nodes from  $\bar{t}_k$  active nodes at the depth  $k$  with  $k \geq \ell$  for the next depth  $k + 1$ . Here min, mean and max denote the minimum, the mean and the maximum of those 5 estimations of the number of nodes in the enumeration tree to generate and the execution time (day), respectively. A Lag-DNN subproblem at each node was solved in about  $30 \sim 150$  seconds.

target LB		no. of nodes			exec. timed (day)		
$\hat{\zeta}$	gap	min	mean	max	min	mean	max
44,100,000	1.46%	22,175	27,278	29,692	0.9	1.1	1.2
44,120,000	1.43%	53,625	72,574	91,944	2.1	2.8	3.6
44,130,000	1.41%	64,084	133,417	275,264	2.5	5.2	10.7
44,150,000	1.36%	241,827	293,696	339,245	9.4	11.5	13.2
44,200,000	1.25%	827,791	1,983,516	2,891,498	32.3	77.4	112.8
44,300,000	1.03%	$5.4 \cdot 10^7$	$3.7 \cdot 10^8$	$8.1 \cdot 10^8$	$2.1 \cdot 10^3$	$1.4 \cdot 10^4$	$3.1 \cdot 10^4$
44,500,000	0.58%	$1.3 \cdot 10^{11}$	$5.5 \cdot 10^{12}$	$2.6 \cdot 10^{13}$	$5.1 \cdot 10^6$	$2.2 \cdot 10^8$	$1.0 \cdot 10^9$
44,759,294	0.00%	$1.2 \cdot 10^{14}$	$6.7 \cdot 10^{16}$	$3.3 \cdot 10^{17}$	$4.6 \cdot 10^9$	$2.6 \cdot 10^{12}$	$1.3 \cdot 10^{13}$

nodes in the full enumeration tree at the depth  $(k + 1)$ . Next, we let  $\bar{t}_{k+1} = 2r_k$ , 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 may regard  $2r_k/s_k = \bar{t}_{k+1}/s_k$  as the increasing rate of the nodes from the depth  $k$  to the depth  $k + 1$ , and the total number of nodes in the full enumeration tree is estimated by

$$\sum_{k=1}^{\ell} t_k + \sum_{k>\ell} \hat{t}_k, \text{ where } \hat{t}_\ell = t_\ell, \hat{t}_{k+1} = (2r_k/s_k)\hat{t}_k \text{ (} k \geq \ell \text{)}. \quad (10)$$

We continue this process till  $r_k$  attains 0. Table 2 shows the estimation of computational work (the number of nodes to generate and the execution time) for 8 cases  $\hat{\zeta} = 44,100,000, \dots, 44,759,294$ . In spite of the simplicity of this unrefined method, it provides useful information on whether a given target LB can be attained by the BB method on the computer used.

## 4.4 Numerical results

We see from Table 2 that the cases with the target LB  $\hat{\zeta} = 44,759,294, 44,500,000$  and  $44,300,000$  are very challenging. The case  $\hat{\zeta} = 44,200,000$  could be processed but might take more than a few months. Table 3 shows numerical results for the other 4 cases with  $\hat{\zeta} = 44,100,000, 44,120,000, 44,130,000$  and  $44,150,000$ . We observe that the estimation of the number of nodes and execution time described in the previous section are useful.

Figure 2 (A) displays the change in the number of nodes as the depth  $k$  increases, and Figure 2 (B) the change in the number of nodes with size 2 orbit. All other nodes are of the trivial single orbit  $N$ , except the root node having size 256 orbit as shown in Section 2.2 and the depth 1 node having sizes 1 through 8 orbit as observed in Table 1.

Recall that the NB method applied to a node  $\text{BQOP}(I_0, I_1)$  generates a sequence of intervals  $[a_p, b_p]$  ( $p = 1, \dots$ ) which monotonically converges to an LB,  $\nu$ , of  $\text{BQOP}(I_0, I_1)$ . Hence, the iteration terminates either when  $b_q < \hat{\zeta}$  occurs —  $\text{BQOP}(I_0, I_1)$  is turned out to be active in this case — or when  $\hat{\zeta} \leq a_q$  occurs —  $\text{BQOP}(I_0, I_1)$  is pruned in this case.

Table 3: Numerical results on the BB method described in Sections 4.1 and 4.2. The 3 numbers in the parenthesis  $(\cdot, \cdot, \cdot)$  denote the minimum, mean and maximum estimation from Table 2, respectively.

target LB		no. of nodes		exec. time (day)	
$\hat{\zeta}$	gap	estimation (min,mean,max)	estimation (min,mean,max)	estimation (min,mean,max)	estimation (min,mean,max)
44,100,000	1.46%	23,510	(22,175, 27,278, 29,692)	1.0	(0.9, 1.1, 1.2)
44,120,000	1.43%	63,554	(53,625, 72,574, 91,944)	2.5	(2.1, 2.5, 3.6)
44,130,000	1.41%	102,310	(64,084, 133,417, 275,264)	4.1	(2.5, 5.2, 10.7)
44,150,000	1.36%	277,304	(241,827, 293,696, 339, 245)	10.7	(9.4, 11.5, 13.2)

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  $k$ .

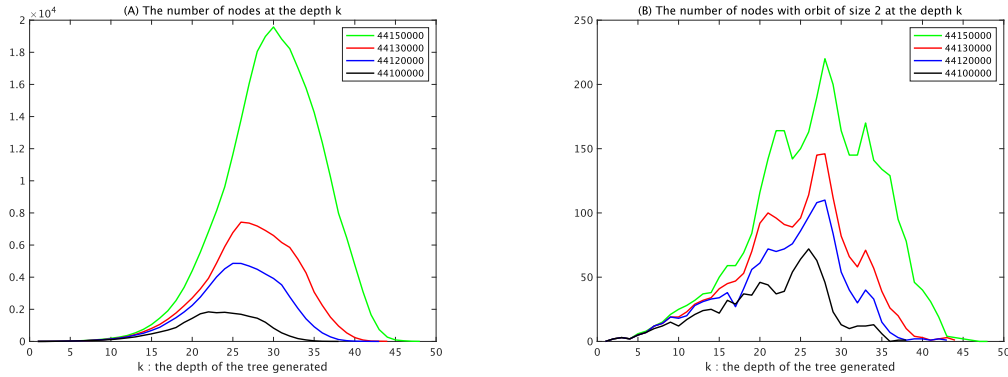
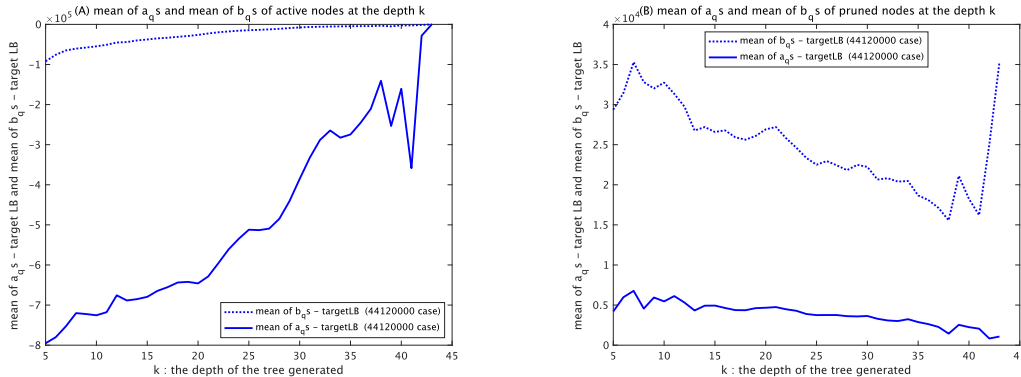


Figure 3: The mean of  $a_q$  (the blue solid line) and  $b_q$  (the blue dotted line) when the NB terminated at iteration  $q$  as  $b_q < \hat{\zeta} = 44,120,000$  (*i.e.*, active node) — Case (A) or at iteration  $q$  as  $\hat{\zeta} = 44,120,000 \leq a_q$  (*i.e.*, pruned node) — Case (B).



We see from Figure 3 that any tight LB is not necessary to decide whether  $\text{BQOP}(I_0, I_1)$  is active or to be pruned in most cases, particularly, in an early 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.

## 4.5 Improving the BB method with the use of equivalence of subproblems

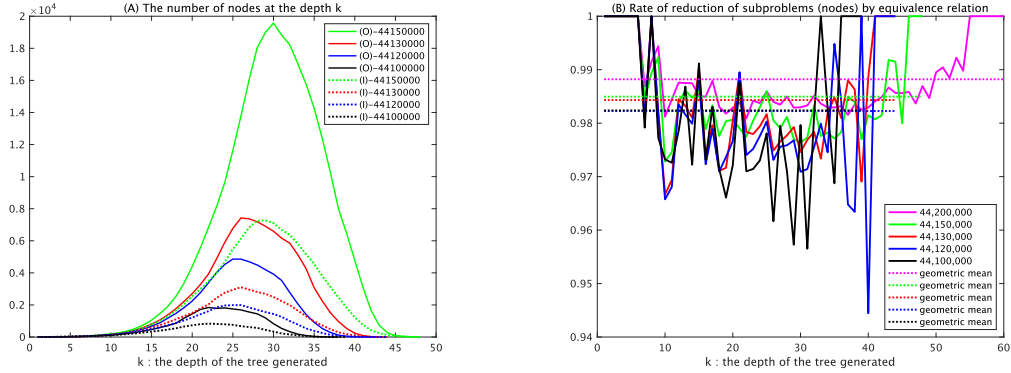
Through numerical results reported in Section 4.4, we found that even with the orbit branching, multiple subproblems appeared in the enumeration tree turned out to be equivalent to each other. Here, two subproblems  $\text{BQOP}(I_0, I_1)$  and  $\text{BQOP}(I'_0, I'_1)$  are called equivalent if

$$|I_0| = |I'_0|, |I_1| = |I'_1| \text{ and } \mathbf{B}(I'_0, I'_1) = \mathbf{P}^T \mathbf{B}(I_0, I_1) \mathbf{P} \quad (11)$$

for some permutation matrix  $\mathbf{P}$ .

The equivalent subproblems,  $\text{BQOP}(I_0, I_1)$  and  $\text{BQOP}(I'_0, I'_1)$ , share not only a common optimal value, but also a common LB, which is obtained as an optimal value of their equivalent Lag-DNN relaxations. Therefore, one of them can be pruned even when both of them are active. For a given pair of subproblems,  $\text{BQOP}(I_0, I_1)$  and  $\text{BQOP}(I'_0, I'_1)$ , checking (11) requires significantly less CPU time than computing their lower bounds. Moreover, various necessary conditions that are easy to implement can be used for verifying (11) with some permutation matrix  $\mathbf{P}$ . Some of the conditions are, for instance,  $\sum_i [B(I'_0, I'_1)]_{ii} = \sum_i [B(I_0, I_1)]_{ii}$ ,  $\sum_{i,j} [B(I'_0, I'_1)]_{ij} = \sum_{i,j} [B(I_0, I_1)]_{ij}$ ,  $\max_i [B(I'_0, I'_1)]_{ii} = \max_i [B(I_0, I_1)]_{ii}$  and  $\min_{i,j} [B(I'_0, I'_1)]_{ij} = \min_{i,j} [B(I_0, I_1)]_{ij}$ . By applying those necessary conditions, the number of the candidates for pairs of subproblems  $\text{BQOP}(I_0, I_1)$  and  $\text{BQOP}(I'_0, I'_1)$  for which (11)

Figure 4: (A) Comparison of the numbers of nodes of the enumeration trees in the original BB method (O) and the improved BB method (I) at the depth  $k$  of their enumeration trees. (B) Reduction rate of subproblems at the depth  $k$  by the equivalence relation.



is tested can be reduced, before verifying (11) with some permutation matrix  $\mathbf{P}$ . When a pair of subproblems  $\text{BQOP}(I_0, I_1)$  and  $\text{BQOP}(I'_0, I'_1)$  satisfies those conditions, an implicit enumeration procedure is applied to generate all possible permutation matrix  $\mathbf{P}$  satisfying (11). If such a permutation matrix  $\mathbf{P}$  is identified, then subproblems  $\text{BQOP}(I_0, I_1)$  and  $\text{BQOP}(I'_0, I'_1)$  are equivalent and one of them is pruned. Otherwise, they are not equivalent.

We improved the BB method by incorporating the technique mentioned above for pruning equivalent subproblems. Table 4 shows numerical results on the improved BB method in comparison to the original BB method whose numerical results have been reported in Section 4.4. We observe that the total number of nodes generated in the **improved BB** method is less than half of the one in the original BB method in all target LB cases, and that a larger target LB 44,200,000 (1.25% gap) is newly computed.

Table 4: Numerical results on the improved BB method in comparison to the original BB method.

target LB		no. of nodes		exec. time (day)	
$\hat{\zeta}$	gap	<b>improved BB</b>	original BB	<b>improved BB</b>	original BB
44,100,000	1.46%	<b>11,594</b>	23,5100	<b>0.6</b>	1.0
44,120,000	1.43%	<b>29,050</b>	63,554	<b>1.2</b>	2.5
44,130,000	1.41%	<b>43,904</b>	102,310	<b>1.8</b>	4.1
44,150,000	1.36%	<b>109,284</b>	277,304	<b>4.3</b>	10.7
44,200,000	1.25%	<b>1,077,353</b>	-	<b>39.2</b>	-

Figure 4 (A) compares the numbers of nodes at the depth  $k$  in the original and **improved**

BB methods for the target LB = 44,100,000, 44,120,000, 44,130,000 and 44,150,000 cases. We can confirm that there exist significant differences between the numbers of nodes generated by them. Figure 4 (B) demonstrates the effectiveness of the technique for pruning equivalent subproblems at each depth  $k$  of the enumeration tree. Let  $v_k$  denote the number of active subproblems determined by the LB procedure at the depth  $k$ . Their  $2v_k$  subproblems are generated by the orbit branching. By applying the technique, we try to reduce the  $2v_k$  subproblems to which the LB procedure is applied at the depth  $k + 1$ . Suppose that some  $w_k$  nodes are pruned by the equivalence relation, where  $w_k$  could be 0. Figure 4 (B) shows the changes of  $(2v_k - w_k)/(2v_k)$  as  $k$  increases ( $k = 1, \dots, \ell$ ) and their geometric mean  $r = (\prod_{k=1}^{\ell} (2v_k - w_k)/(2v_k))^{1/\ell}$ , where  $\ell$  denotes the depth of the enumeration tree when the improved BB method terminated. In all target LB cases, we see that  $r \in [0.98, 0.99]$  in Figure 4 (B). It can be summarized that the technique reduces the number  $v_k$  of subproblems generated by the orbit branching to  $rv_k$  at the depth  $k$  on average, and the modified BB method can reduce the total number of nodes generated by the original BB method by the factor  $r^\ell$ .

From the discussions above, we can conclude that the technique proposed for pruning equivalent subproblems is indeed effective in accelerating the original BB method. We must say, however, that computing an LB with 1.1% gap remains very difficult since the technique would reduce the number of nodes generated by at most  $1/8 \in [0.98, 0.99]^{100}$ , where the improved BB method is assumed to terminate in the enumeration tree at the depth  $\ell = 100$ .

## 5 Concluding remarks

We have investigated the 256-dimensional BQOP with a single cardinality constraint, BQOP (2), which is converted from the largest unsolved QAP instance tai256c. The converted BQOP with dimension 256 is much simpler than the original QAP tai256c involving  $256 \times 256 = 65536$  binary variables, and its dimension 256 is not so large. While one might expect the converted BQOP to be notably easier to solve compared to the original QAP tai256c, our findings indicate that it still presents a significant challenge. The challenge primarily stems from the symmetry property (3) exhibited in the coefficient matrix  $\mathbf{B}$ , which is inherited from tai256c. For future development toward solving the BQOP, we need

- an efficient and much stronger lower bounding procedure than the DNN relaxation,
- additional techniques to enhance the exploitation of the symmetry property (3), and
- more powerful computer systems.

While we have focused on BQOP (2) 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 (3).

## Acknowledgements

The authors are grateful to Professor Koji Nakano for providing them with numerical results on DABS (Diverse Adaptive Bulk Search, a genetic algorithm-based search algorithm for solving QUBO [28]), which have been included in Section 4. The work of Sunyoung Kim



was supported by NRF 2021-R1A2C1003810. The work for this article has been partially conducted within the Research Campus MODAL funded by the German Federal Ministry of Education and Research (BMBF grant numbers 05M14ZAM, 05M20ZBM).

## References

- [1] K. Anstreicher, N. Brixius, Goux J-P., Linderoth, and J. Solving large quadratic assignment problems on computational grids. *Math. Program.*, 91:563–588, 2002.
- [2] K. Anstreicher and H. Wolkowicz. On Lagrangian relaxation of quadratic matrix constraints. *SIAM J. Matrix Anal. Appl.*, 22(41-55), 2000.
- [3] N. Arima, S. Kim, M. Kojima, and K. C. Toh. Lagrangian-conic relaxations, Part I: A unified framework and its applications to quadratic optimization problems. *Pacific J. of Optim.*, 14(1):161–192, 2018.
- [4] N. Arima, S. Kim, M. Kojima, and K.C. Toh. A robust Lagrangian-DNN method for a class of quadratic optimization problems. *Comput. Optim. Appl.*, 66(3):453–479, 2017.
- [5] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2:183–202, 2009.
- [6] 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.
- [7] E. Burkard, R. E. Cela, S.E. Karisch, and F. Rendl. QAPLIB – a quadratic assignment problem library. *J. Global Optim.*, 10:391–403, 1997.
- [8] R.E. BURKARD, E. ÇELA, S.E. KARISCH, and F. RENDL. QAPLIB, <https://www.miguelanjos.com/qaplib>.
- [9] J. Clausen and M. Perregaard. Solving large quadratic assignment problems in parallel. *Comput. Optim. Appl.*, 8:111–127, 1997.
- [10] D. T. Connolly. An improved annealing scheme for the QAP. *Eur. J. Oper. Res.*, 46(93-100), 1990.
- [11] Etine de Klerk and R. Sotirov. Exploiting group symmetry in semidefinite programming relaxations of the quadratic assignment problem. *Math. Program.*, 122:225–246, 2010.
- [12] Etine de Klerk and R. Sotirov. Improved semidefinite programming bounds for quadratic assignment problems with suitable symmetry. *Math. Program.*, 133:75–91, 2012.
- [13] F. Fabio et al. QPLIB: a library of quadratic programming instances. *Mathematical Programming Computation*, 11:237–265, 2019.

- [14] K. Fujii, S. Kim, M. Kojima, H. D. Mittelmann, and Y. Shinano. The largest unsolved QAP instance tai256c can be converted into a 256-dimensional simple BQOP with a single cardinality constraint. Technical Report arXiv:2210.15963, ZIB-Report 22-18, October 2022.
- [15] L. M. Gambardella, E. D. Taillard, and M. Dorigo. Ant colonies for the QAP. Technical report idsia-4-97, IDSIA, Lugano, Switzerland, 1997.
- [16] P. C. Gilmore. Optimal and suboptimal algorithms for the quadratic assignment problem. *SIAM J. Appl. Math.*, 10:305–313, 1962.
- [17] A. D. 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.
- [18] Gurobi Optimization. Gurobi 11.0: Every Solution, Globally Optimized. <https://www.gurobi.com>, 2023.
- [19] 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.
- [20] S. Ji, X. Zheng, and X Sun. An improved convex 0-1 quadratic program reformulation for quadratic knapsack problems. *Pacific J. of Optim.*, 8(1):75–87, 2012.
- [21] 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.
- [22] 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.
- [23] E. L. Lawler. The quadratic assignment problem. *Management Sci.*, 19:586–590, 1963.
- [24] B. D. McKay. Nauty users guide (version 2:4). Technical report, Dept. Comp. Sci., Australian National University, 2010.
- [25] H. Mittelmann. Benchmarks for optimization software. <http://plato.asu.edu/bench.html>, 2012. Accessed: Nov 2023.
- [26] H. Mittelmann. Improved QAPLIB lower bounds using BBCPOP and Newt-Bracket, [https://plato.asu.edu/ftp/qaplib\\_bounds.html](https://plato.asu.edu/ftp/qaplib_bounds.html), December 2023.
- [27] H. Mittelmann. Nonconvex QUBO-QPLIB benchmark, September 2023.
- [28] K. Nakano, D. Takafuji, Y. Ito, T. Yazane, S. Yano, J. an, R. Katsuki, and R. Mori. Diverse adaptive bulk search: a framework for solving qubo problems on multiple gpus. In *Proc. of 2023 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 314–325, May, 2023.

- [29] O Nissofolk, R. Pörn, and T. Westerlund. Testing a non-diagonal convex reformulation technique for 0-1 quadratic programs. Presentation at ESXAP 26, Slides, <https://blogs.abo.fi/ose/files/2017/02/Westerlund-ESCAPE26.pdf>, June 2016.
- [30] O Nissofolk, R. Pörn, and T. Westerlund. Testing the non-diagonal quadratic convex reformulation technique. In *Proceedings of the 26th European Symposium on Computer Aided Process Engineering-ESCAPE 26*, pages 331–336. Elsevier, June 2016.
- [31] J. Ostrowski, J. Linderoth, Rossi F., and S. Smriglio. Orbital branching. *Math. Program.*, 126:147–178, 2011.
- [32] 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.
- [33] F.N. Permenter and P. A. Parrilo. Dimension reduction for semidefinite programs via jordan algebras. *Math. Program.*, 181:51–84, 2020.
- [34] M. E. Pfetsch and Rehn T. A computational comparison of symmetry handling methods for mixed integer programs. *Math. Program. Comput.*, 11:37–93, 2019.
- [35] J. Povh and F. Rendl. Copositive and semidefinite relaxations of the quadratic assignment problem. *Discrete Optim.*, 6:231–241, 2009.
- [36] C. Roucairol. A parallel branch and bound algorithm for the quadratic assignment problem. *Discret. Appl. Math.*, 18:211–255, 1987.
- [37] J. Skorin-Kapov. Tabu search applied to the quadratic assignment problem. *ORSA J. Comput.*, 2:33–45, 1990.
- [38] E. Tailard. Robust taboo search for the quadratic assignment problem. *Parallel Comput.*, 17:443–455, 1991.
- [39] A. Wiegele. Biq mac library. <http://www.biqmac.uni-klu.ac.at/biqmaclib.html>, 2007.
- [40] 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.