Annals of Operations Research
https://doi.org/10.1007/s10479-026-07201-1

ORIGINAL RESEARCH

Quantum bridge analytics: a QUBO approach to the uncapacitated facility location problem

Haitao Li[1] · **Rick Hennig**[2] · **Gary Kochenberger**[2] · **Fred Glover**[2]

Received: 3 January 2025 / Accepted: 30 March 2026

Abstract
We examine the significant potential of quantum computing for solving combinatorial optimization problems in supply chains, with a focus on recent advances and challenges that lie ahead. We observe that the constructs in classical quantum physics pose three main gaps for exploiting quantum advantage in the supply chain area: optimization modeling, algorithm design, and solution assessment. To overcome these gaps, we elaborate how the Quadratic Unconstrained Binary Optimization (QUBO) model that lies at the heart of supply chain optimization with quantum computing can be fruitfully exploited within the framework of Quantum Bridge Analytics (QBA) to connect quantum computing and classical optimization. Our QBA/QUBO approach is implemented for the well-known uncapacitated facility location problem (UFLP) in supply chain network design, using Gurobi's QUBO solver, D-Wave's hybrid classical-quantum solver, and the proprietary next generation quantum (NGQ) solver. A comprehensive computational study is performed on the UFLP benchmark instances with comparisons to four recent algorithms for UFLP in the literature. The QBA/QUBO approach achieved high-quality results for problems of all sizes, solving all the small-size and some medium-size instances with up to 10,000 decision variables to optimality in less than 20 s, and achieving less than 3% optimality gap for large instances with more than 250,000 decision variables in less than 10 min on average. We also find that in addition to problem size, the polyhedral properties of a UFLP instance may significantly impact the efficiency of quantum annealing based algorithm.

Keywords Quantum computing · Quantum bridge analytics · Quadratic unconstrained binary optimization (QUBO) · Uncapacitated facility location problem (UFLP)

This paper is scheduled to appear in a special issue of the Annals of Operations Research in honor of Professor John Mulvey of Princeton University for his many and lasting contributions to the field of operations research and especially to optimization methods and modeling applications to business and government settings.

✉ Haitao Li
lihait@umsl.edu

[1] Supply Chain & Analytics Department, Ed G. Smith College of Business, University of Missouri – St. Louis, St. Louis, MO 63121, USA

[2] Entanglement Inc., Boulder, CO 80302, USA

1 Introduction

Quadratic unconstrained binary optimization (QUBO) is an extensively studied topic in mathematical programming and optimization, which can be traced back to the 1960s (Hammer et al., 1963). A typical QUBO model minimizes (or maximizes) a quadratic objective function by determining a set of binary (Boolean or 0–1) decision variables with no constraints. A general QUBO model is strongly NP-hard, although there are special cases of QUBO that are polynomially solvable (Punnen, 2022).

From the application perspective, a variety of combinatorial optimization problems in graph theory can be naturally modeled as QUBO problems, including the maximum cut problem, the maximum clique problem, and the maximum stable set problem, among others. Numerous other combinatorial optimization problems that are typically modeled as an integer program can be reformulated as a QUBO model, including quadratic assignment, capital budgeting, constraint satisfaction, multiple knapsack, set partitioning, and set packing problems. We refer to Kochenberger et al. (2014) and Glover et al. (2022a) for a comprehensive review on the modeling, solution methods and applications of QUBO, and a monograph by Punnen (2022) for a comprehensive treatment on history and mathematical derivations.

Perhaps the most exciting motivation for the growing interest in QUBO is its relevance to the emerging development of quantum computing. Different from the classical computing with information stored in a *binary bit* with two states (0 or 1), quantum computing is built upon the *quantum bit*, or *qubit*, which can be in a superposition of both states simultaneously. This property creates promise of superior computational efficiency for quantum computers to outperform classical computers in a wide range of applications. This excitement is tempered by an element of suspense, since predictions about when quantum advantage will become manifest vary from a few years to more than a decade, and a major issue is how to develop algorithms for QUBO problems and their variants that can meet the demand for solving them effectively in the meantime.

Finding an optimal QUBO solution is an NP-hard optimization problem analogous to finding the ground state of a spin glass system (Barahona, 1982), which can be described as the challenge of determining the alloy of two elements: Fe—a magnetic element being added to Au—a non-magnetic element (Bapst et al., 2013). The magnetic moments, called *spins*, interact with each other, which makes them align or disperse (become ferromagnetic or antiferromagnetic). The mixture starts in its liquid state at a high temperature. As the temperature cools down, the mixture gradually reaches its solid (ground) state with the positions of the magnetic elements becoming frozen in a random configuration. The magnetic moments in spin glasses can be modeled by Ising variables $s_i = \pm 1$, which interact via a quadratic energy function as in the Edwards and Anderson (EA) model (Edwards & Anderson, 1975): $-\sum_{<i,j>} J_{ij} s_i s_j$, where J_{ij} is the coupling between a pair of neighboring spins i and j to allow both positive (ferromagnetic) and negative (antiferromagnetic) interactions determined by a random parameter chosen from some probability distribution (such as a Gaussian).

A popular quantum algorithm for solving this type of optimization problem is known as quantum annealing (QA) or quantum adiabatic evolutionary algorithm (QAEA), which conceptually works as follows. For an instance of an optimization problem P, construct a Hamiltonian H_P as a function of the state $|\psi\rangle$ of the quantum system consisting of Ising variables, such that the ground state of H_P encodes the optimal solutions to P, and the energy of H_P at its ground state is the optimal or near-optimal objective value. Suppose the system has its initial state $|\psi(0)\rangle$ with the ground energy $H(0)$. At time t, the Hamiltonian $H(t)$ has

an instantaneous ground state $|\psi(t)\rangle$. During an adiabatic evolution, $H(t)$ is slowly varying following the Schrödinger equation. According to the adiabatic theorem, if $H(t)$ varies slowly enough, the evolving state $|\psi(t)\rangle$ will remain close to its instantaneous ground state (Messiah, 1962). Representing the running time of the algorithm by T, quantum annealing aims to find an interpolation between $H(0)$ and $H(T)$, such that when T is large enough and $H(t)$ varies slowly, the final state $|\psi(T)\rangle$ will be close to the corresponding ground state, which is, hopefully, an optimal (or near-optimal) solution to the problem P. We refer to de Wolf (2023) and Bapst et al. (2013) for a thorough coverage on quantum computing and the quantum adiabatic algorithm.

QA is related to but differs from the well-known simulated annealing process (Kirkpatrick et al., 1983) in the optimization community. In QA, the system first transits from one state to another in an adiabatic evolution process, which is analogous to moving from one solution to a neighborhood solution in a local search algorithm (Aarts & Lenstra, 2003) such as simulated annealing. In addition, the physical process of metal annealing in simulated annealing is similar to the annealing of spin glasses in QA by controlling the temperature. Moreover, both algorithms are probabilistic in nature in that a local move or state-transition follows certain probability distribution, such as the Gibbs-Boltzmann distribution, to avoid being trapped at local optima. However, QA has some unique features that are often viewed as advantageous to simulated annealing, at least in theory. For instance, quantum mechanics can ideally induce a "tunnel path" that avoids local optima efficiently, which is known as the quantum fluctuation as opposed to the thermal fluctuation in simulated annealing (Ray et al., 1989). The quantum mechanical wave function can also delocalize the solution space by smoothing the landscape, which makes it easier to escape from local optima (Das & Chakrabarti, 2008).

From the modeling perspective, while it is straightforward construct the Hamiltonian H_P when the problem P at hand has a natural spin glass formulation with no constraints, many real world applications in supply chain, operations management and engineering require constrained optimization formulations. Constraints can be treated as "frustration" in a classical spin glass model (Anderson, 1978), but there is no systematic approach in quantum mechanics to handle this situation. This has motivated the emergence of Quantum Bridge Analytics (QBA), as coined by Glover et al. (2022a), to bridge the gap between classical and quantum computing methods, including QUBO reformulation, unsupervised and supervised learning, and approximation algorithms.

QBA opens the door to developing quantum computing and quantum-inspired techniques for a wide range of optimization applications in supply chains (Li, 2023), including supply chain network design at the strategic level; production planning, supply chain configuration, and resource planning at the tactical level; and the operational-level scheduling and vehicle routing. These applications can be modeled as integer/binary programs, or mixed-integer programs, which are all good candidates for the QUBO and QBA approaches.

From the computational perspective, while quantum annealing has shown significant promises in computational efficiency, there have been debates on whether it can solve NP-hard combinatorial optimization problems more effectively (better solution quality) and more efficiently (faster) than traditional optimization techniques consisting either of exact methods of branch-and-bound and branch-and-cut in integer programming (cf. Nemhauser & Wolsey, 1988, Bertsimas & Weismantel, 2005, and Vanderbei, 2020), or of heuristic/metaheuristic methods including local search and evolutionary algorithms (cf. Gendreau & Potvin, 2019; Glover & Kochenberger, 2005). For instance, Dickson and Amin (2011) showed that it is possible to have a Hamiltonian for the NP-hard maximum independent set problem that will enable the problem to be solved in polynomial time, although finding such a Hamiltonian may not be trivial. Farhi et al. (2012) provided computational evidence that the quantum

adiabatic evolution algorithm fails to solve two NP-hard problems efficiently, namely, the 3-regular XORSAT and 3-regular Max-Cut. These studies approached NP-hard problems from the quantum physics perspective, but without considering the polyhedral properties of their integer programming formulation, which is the fundamental theoretical approach in mathematical programming to find and prove optimality of an NP-hard problem (Nemhauser & Wolsey, 1988). It remains an open question of how quantum annealing would perform on many NP-hard optimization problems in supply chains.

Most of the existing studies have focused on classical graph-based optimization problems, such as the Maximum-Cut problem and the Maximum Independent Set problem, but there is a lack of real world optimization problems treated in the quantum optimization literature and their mathematical programming formulations are usually not straightforwardly available in the graph setting. This paper aims to fill this gap.

We focus on the optimization component of QBA to delineate the technical relevance of applying quantum annealing, mathematical programming and QUBO methods to solve the uncapacitated facility location problem (UFLP, Drezner & Hamacher, 2004), a prominent NP-hard combinatorial optimization problem in supply chain network design. The UFLP is known to have polyhedral properties that significantly affect the computational effort to solve it using classical branch-and-bound and branch-and-cut methods. In our study, we have included benchmark instances with large duality gaps to provide insight into the factors impacting the difficulty of solving combinatorial optimization problems in this class.

We explore two ways to build the QUBO formulation for the UFLP: (i) by the algebraic QUBO reformulation of UFLP from its binary integer programming (BIP) model, and (ii) by explicitly specifying the Q matrix in the QUBO reformulation. We also study the effect of different penalty values for incorporating constraints in the QUBO formulation on solution quality and computational time. Computational results on benchmark instances with different characteristics, i.e., size and polyhedral properties, are compared with multiple algorithms including the quantum annealing algorithm and several benchmark metaheuristics.

The remainder of this paper is organized as follows. Section 2 reviews related work in the literature. Section 3 first presents the quantum annealing framework, then explains why and how mathematical programming and QUBO can be applied to bridge the gap. Section 4 presents an application of QBA/QUBO on the UFLP, with a comprehensive computational study on some benchmark instances. Section 5 draws conclusions and discusses future research opportunities.

2 Literature review

The early idea of using quantum annealing or quantum adiabatic evolution algorithm (QAEA) for solving NP-hard optimization problems can be found in Brooke et al. (1999). A traverse Ising model was presented by Kadowaki and Nishimori (1998) where quantum fluctuations were introduced to the classical simulated annealing framework, which aims to avoid local optima but in a different way than the thermal fluctuations in simulated annealing.

The theoretical promise of quantum annealing leads to an active stream of research on the question of whether NP-hard optimization problems can be solved in polynomial time via quantum computing, or in the language of quantum mechanics, how much time is needed for adiabatic evolution to reach the ground state of a system which encodes the optimal to solution to the original optimization problem. Farhi et al. (2000) studied the satisfiability problem and showed that a special case of a symmetric satisfiability problem can be solved

by the QAEA in polynomial time. Farhi et al. (2001) further showed that a QAEA worked well for small-size instances of the exact cover problem. In turn, Childs et al. (2002) found that for the small-size instances of the maximum clique problem, QAEA appeared to run in quadratic time.

Other computational investigations on this topic summarized by Dickson and Amin (2011) found that experiences with small-sized instances were misleading, and provided counterexamples to the polynomial scaling behavior of QAEA. The same authors analytically showed that it is possible to have a Hamiltonian that solves the maximum independent set problem in polynomial time although, again, identifying such a Hamiltonian was not trivial. They also pointed out that the QAEA can be viewed as an approximation (or heuristic) method which can reach the ground state fast, but it cannot be proved to solve *all* instances of an NP-hard problem in polynomial time. Dickson and Amin (2011)'s point highlights the important difference between *finding an optimal solution* (or reaching the ground state in QA and QAEA) and *proving a solution is optimal*. The latter is the core capability of the branch-and-bound and branch-and-cut method in integer programming (Nemhauser & Wolsey, 1988). The study of Farhi et al. (2012) showed that a QAEA failed to solve the 3-regular XORSAT and 3-regular Max-Cut problems efficiently. We refer to Bapst et al. (2013) for a systematic review on QA and QAEA and the related computational experiences.

The intrinsic link between Ising spin variables in quantum mechanics and binary decision variables in integer programming, as well as in the quadratic form of the Hamiltonian in QA and QAEA, has motivated the idea of applying the binary QUBO formulation to bridge the gaps between quantum computing and the classic optimization paradigms. These gaps are two-fold, encompassing both the modeling and algorithmic perspectives. From the modeling perspective, the Ising spin variables $\{-1, +1\}$ often do not work well to encode decision variables in customary optimization problems, which are often binary $\{0, 1\}$ in nature. While the translation between Ising spin variables and binary decision variables is straightforward, the modeling of constraints (or frustration in quantum physics) in the objective function (or the Hamiltonian in quantum computing) is not trivial, especially for the plethora of optimization applications in engineering and supply chains. An early treatment on a unified QUBO modeling framework for various optimization problems was provided by Kochenberger and Glover (2006), and a systematic approach to reformulate a BIP to a QUBO model, including the special cases of constraints for which there are known penalty functions, was presented in Glover et al. (2022a). The application of QUBO transformations for some selected optimization problems in supply chains was described in (Li, 2023), including the shortest path problem, the uncapacitated facility location problem, the job assignment problem, and the crew scheduling problem. An alternative approach called QUBO-Plus was introduced by Glover et al. (2022b), which keeps certain constraints in a model separate from the QUBO formulation so that they can be handled by an appropriate custom designed algorithm, with an illustration for an asset exchange problem motivated from the blockchain. From the algorithmic perspective, a wealth of solution techniques have been applied to the QUBO problem, including both exact and heuristic and metaheuristic algorithms, which are well-developed in the optimization community (cf. Kochenberger et al., 2014; Punnen, 2022). A proprietary metaheuristic, called the next generation quantum (NGQ) solver from Entanglement, has been investigated in a study by Du et al. (2023) and found to significantly outperform the classic integer programming solver by Cplex and the quantum–classical hybrid solver by D-Wave (2023) for solving large-scale set partitioning problems.

There has been a growing body of research into exploring *quantum utility* for optimization, which involves performing tasks effectively and efficiently by quantum computing compared to classical optimization methods, and into *quantum advantage*, which involves developing

new methods that outperform classical optimization methods for certain tasks. Venturelli et al. (2016) developed a QUBO formulation for the job shop scheduling problem solved by D-Wave's quantum annealers. Their computational study was limited to small instances with 6 jobs and 6 machines. Guillaume et al. (2016) presented an application of NASA's deep space network scheduling problem, which was formulated as QUBO and solved by a hybrid algorithmic framework that embeds quantum-inspired method to obtain quality solutions efficiently. Ikeda et al. (2019) applied quantum annealing for a nurse scheduling problem, which could find feasible solutions for small instances with 3–4 nurses over a short horizon. Geitz et al. (2022) developed a hybrid approach that combines constraint programming (CP, Marriott & Stuckey, 1998) and quantum annealing for a variant of job-shop scheduling problem with sequence-dependent setup times. Their computational results showed that for one instance with 4 jobs, 4 machines and 6 positions, the solutions found by D-Wave's quantum annealing and quantum-inspired algorithm by Fujitsu have an optimality gap of larger than 35%. Carugno et al. (2022) provided an evaluation of modeling and solving the job-shop scheduling problem using D-Wave quantum annealers, and found that quantum annealing can match with classical heuristics, simulated annealing, for very constrained instances. They also showed that building QUBO reformulation itself can take large amount of time. A comprehensive review and positioning paper by Koch et al. (2025) provided a systematic review on a suite of ten selected NP-hard combinatorial optimization problems as benchmarks for quantum optimization. None of the problems in the benchmark library have achieved either quantum advantage or quantum utility for reasonably large instances.

While quantum bridge analytics via QUBO and QUBO-Plus have paved the way for applying quantum-inspired approaches for a variety of combinatorial optimization problems, most of the existing studies either focus on simple stylized graph-based problems, e.g., maximum-cut, or applications with very small sizes. By contrast, our work aims to explore quantum utility and quantum advantage for solving a well-known optimization problem in logistics and supply chain with reasonably large size of up to 250,000 binary decision variables. We also investigate the questions of how the penalty value for constraints should be chosen in QUBO reformation, and how different choices affect the solution quality and computational time. Further, how would the polyhedral properties of a formulation affect the performance of quantum optimization.

3 Quantum bridge analytics via QUBO

In this section, we start with a formal description of the quantum annealing QAEA approach for an optimization problem, then identify the existing computational gaps between QAEA and classical optimization methods with a road map for real world applications. Next, we present a general QUBO reformulation approach and investigate how it bridges these computational gaps.

3.1 The QAEA approach

Consider a spin glass system described by N Ising spin variables $s_i \in \{-1, +1\}$ for $i = 1,2, \ldots, N$. A classical state of the system can be represented the vector $(s_1, s_2, \ldots, s_N)$ of Γ Ising spin variables with a total of $\Gamma = 2^N$ possible classical states denoted as $|1\rangle, |2\rangle, \ldots, |\Gamma\rangle$. Following the notation in (de Wolf, 2023), a *quantum state* (called state in the sequel) $|\phi\rangle$ is a *superposition* of the Γ classical states, which can written as $|\phi\rangle = \sum_{j=1}^{\Gamma} \alpha_j |j\rangle$, where

α_j is a complex number representing the *amplitude* of $|j\rangle$ in $|\phi\rangle$. A system in quantum state is in all classical states simultaneously, with each state having a different amplitude.

Since one can only observe the classical state, rather than directly observing the superposition, an attempted *measurement* of the system yields a classical state $|j\rangle$ with some probability. According to the Born's rule, the probability for the system to be in the classical state $|j\rangle$ is $|\alpha_j|^2$, i.e. the squared norm of the corresponding amplitude. And the vector of amplitudes has a Euclidean norm of 1, i.e., $\sum_{j=1}^{\Gamma}|\alpha_j|^2 = 1$.

After defining the Ising spin variables as decision variables for an optimization problem, a crucial step is to construct the Hamiltonian $H(T) = -\sum_{<i,j>} J_{ij}s_i s_j$, To do this, one needs to specify J_{ij}, the coupling between a pair of neighboring spins i and j, such that the ground state $|\phi_g(T)\rangle$ of the system at the end of executing QA with total running time T encodes the optimal solution to the problem at hand. While it is relatively straightforward to obtain J_{ij} for several graph-related combinatorial optimization problems, such as the coloring problem, XORSAT, k-SAT, and the exact cover problem as studied in the existing quantum computing field, getting the value for many real world optimization applications is not a trivial task—because a large number of constraints may be needed, and more importantly, the constraints must properly model the requirements and/or limitations on the decisions to make, whose intricacies are addressed in (Li, 2023). This is a gap that can be bridged by QUBO to be elaborated in the next subsection.

Let $|\phi(t)\rangle$ denote the state of the quantum system at time t, which evolves according to the Schrödinger equation:

$$i\frac{\mathrm{d}}{\mathrm{d}t}|\phi(t)\rangle = H(t)|\phi(t)\rangle \tag{1}$$

As noted earlier, the adiabatic theorem states that if $H(t)$ varies slowly enough, then $|\phi(t)\rangle$ will stay close to its corresponding instantaneous ground state $|\phi_g(t)\rangle$. This means that when the algorithm ends at time T, the ending ground state $|\phi_g(T)\rangle$ can be achieved and with an optimal or near-optimal solution found.

The design of a quantum annealing QAEA method involves: (i) choosing an initial state $|\phi(0)\rangle$ whose ground state is known and easy to compute; and (ii) controlling $H(t)$ such that it smoothly interpolates from $H(0)$ to $H(T)$, e.g., via a linear interpolation:

$$H(t) = \left(1 - \frac{t}{T}\right)H(0) + \frac{t}{T}H(T) \tag{2}$$

As T becomes large, and the change of $H(t)$ becomes slow, the relationship (2) implies $|\phi(t)\rangle$ will stay close to its ground state $|\phi_g(t)\rangle$ and when the algorithm ends at T, we will obtain $H(t) = H(T)$, with the ending ground state $|\phi_g(T)\rangle$ being the solution.

We refer to (Farhi et al., 2001) for an accessible description of QAEA with its implementation for an NP-hard problem. A commercial QA solver is available from the D-Wave's quantum–classical hybrid solver (D-Wave, 2023).

3.2 Gaps for QBA via QUBO

We now qualitatively identify the gaps between QAEA and classical optimization, and explain how QUBO can bridge them.

- *Gap 1: Optimization Modeling*. The modeling constructs in quantum mechanics, e.g., the Hamiltonian in the spin glass model, are not amenable for modeling the vast set of optimization problems in supply chain, operations management, and engineering, which have

been well-handled by various mathematical programming models, e.g., network optimization (Glover et al., 1992) and integer programming (Li, 2023). QUBO can serve as the bridge between quantum computing and mathematical programming to exploit quantum advantage for real world optimization applications.

- *Gap 2: Algorithm Design.* On one hand, classical optimization methods can provide guidance on how to choose the initial state (initial solution) in QAEA, a component that can significantly impact the performance of QAEA (Dickson & Amin, 2011). For instance, one choice of initial solution can be the solution to the linear programming (LP) relaxation (after a post-processing step to make the solution feasible), also called root relaxation of an integer programing model. The design of interpolation of $H(t)$ may also benefit from various advanced intensification and diversification schemes in tabu search (TS, Glover & Laguna, 1997), scatter search and path relinking (Glover, 1997), including those developed for QUBO (cf. Kochenberger et al. (2014), Punnen (2022)). On the other hand, classical metaheuristics may benefit by borrowing some ideas inspired by quantum mechanics phenomena, e.g., the tunneling effect, entanglement, etc., to improve computational performance.
- *Gap 3: Solution Assessment*. We also note that it's still an open question *whether QAEA is able to solve an NP-hard problem in polynomial time*. This means that although a QAEA algorithm may have found an optimal solution to an optimization problem when reaching the ground state at the end, it remains unable to verify that the solution is optimal. In other words, there is no means yet in quantum computing alone to either assess the quality of a solution or prove its optimality. This gap can be addressed by branch-and-bound and branch-and-cut methods in mixed-integer linear programming (MILP, Nemhauser & Wolsey, 1988), but while such methods converge to optimality in finite time, the amount of computation can theoretically be astronomical. Nevertheless, imperfect as it may be, the bounding information from MILP can be used to assess the quality of solutions obtained from QAEA, once the bridge is established between QAEA and an MILP model via QUBO.

Figure 1 provides a summary of how QUBO provides a bridge between the classical optimization and quantum computing. Note that the bridge for Gap 2 of the Algorithm Design is bidirectional, i.e., quantum computing and metaheuristics may benefit from each other through their unique features and complementary strengths.

Figure 1 also suggests a roadmap for exploiting the quantum advantage for a real-world optimization application. One opts to start with modeling an optimization problem at hand as a mathematical programming model, and ideally a BIP (Gap 1). Then both exact methods and heuristics or metaheuristics can be employed in an effort to solve the BIP (Gap 2), using bounds on the optimal objective value of the optimization problem obtained by the exact branch-and-bound and branch-and-cut methods, which can be used to assess the quality of the solution obtained by QAEA (Gap 3). A critical next step is to reformulate the BIP as a QUBO model, which makes it possible to implement a QA or QAEA algorithm (in addition to applying metaheuristic algorithms). Details on how to reformulate a BIP as a QUBO model are presented next.

3.3 QBA via QUBO

In this section, we elaborate how QUBO bridges Gap 1. We start with establishing the equivalence between the Ising spin variables in QAEA and the binary decision variables in QUBO. Define $x_i \in \{0, 1\}$ as a binary decision variable for $i = 1,2, \ldots, N$. The conversion between spin variables and binary variables is given by $x_i = \frac{S_i+1}{2}$ or $s_i = 2x_i - 1$, thus easily

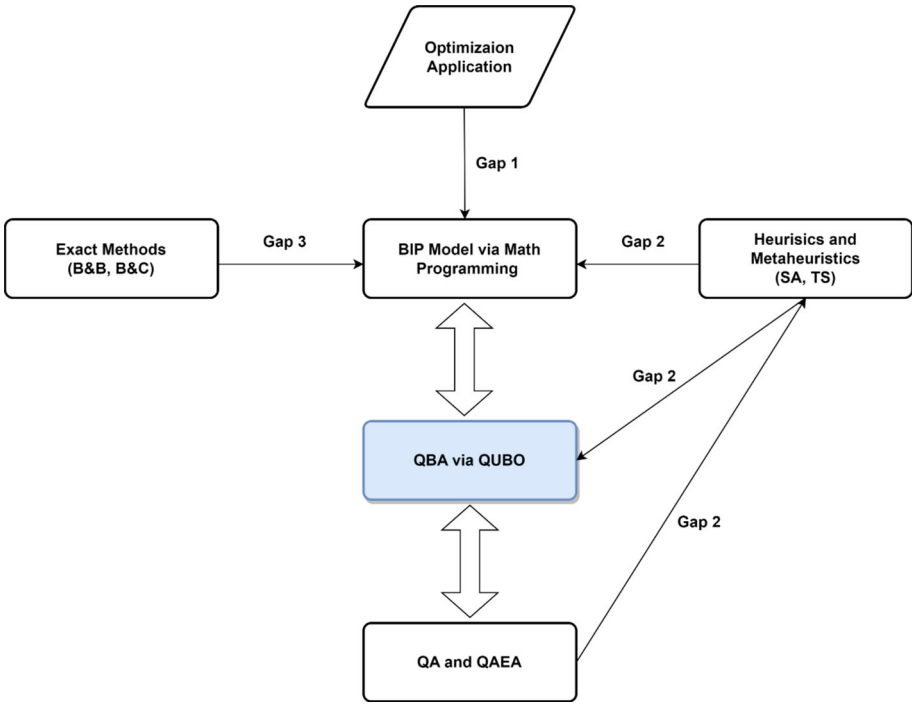

Fig. 1 A roadmap of applying QBA via QUBO for quantum advantage

establishing the link between the spin glass model and the QUBO model. The link between QUBO and the classical BIP model needs a systematic reformulation approach to establish, following the procedures described in Kochenberger and Glover (2006), Glover et al. (2022a) and Li (2023).

Specifically, consider a general BIP model with n binary decision variables x_j and objective function coefficients c_j, for $j = 1,2, \ldots, n$. We represent the three types of constraints as: m_1 equalities, m_2 inequalities with $\leq$, and m_3 inequalities with $\geq$. Let $[a_{ij}^1]_{m_1 \times n}$, $[a_{ij}^2]_{m_2 \times n}$, and $[a_{ij}^3]_{m_3 \times n}$ denote the entries in the left-hand-side coefficient matrix of the equalities, inequalities with $\leq$, and inequalities with $\geq$, respectively, and denote the right-hand-side constants of these constraints by b_i^1, b_i^2 and b_i^3, respectively, to give the BIP model formulation:

$$Minimize \sum_{j=1}^{n} c_j x_j \tag{3}$$

$$\text{Subject to}: \sum_{j=1}^{n} a_{ij}^1 x_j = b_i^1, \ \forall i = 1, 2, \ldots, m_1 \tag{4}$$

$$\sum_{j=1}^{n} a_{ij}^2 x_j \leq b_i^2, \ \forall i = 1, 2, \ldots, m_2 \tag{5}$$

$$\sum_{j=1}^{n} a_{ij}^3 x_j \geq b_i^3, \ \forall i = 1, 2, \ldots, m_3 \tag{6}$$

$$x_j \in \{0, 1\}, \ \forall j = 1, 2, \ldots, n \tag{7}$$

To reformulate this BIP as a QUBO model, we construct a penalty function for each constraint and add it to the objective function to penalize violation of the constraint. For the equality constraints of (4), we simply penalize the square of its violation, i.e., $P \cdot \sum_{i=1}^{m_1} (b_i^1 - \sum_{j=1}^{n} a_{ij}^1 x_j)^2$, where P is a sufficiently large positive constant whose value is typically known from past experience or readily determined by experiment. (There is generally a range of "good" values sufficiently large to assure the constraint will be satisfied at optimality, but neither too large nor too small, which can slow convergence to optimality, as we subsequently illustrate.)

The inequality constraints of (5) are first converted into equality constraints by adding slack variables $\omega_i^2 \in \{0\} \cup Z^+$ for i = 1,2, …, m_2, to yield:

$$\sum_{j=1}^{n} a_{ij}^2 x_j + \omega_i^2 = b_i^2, \forall i = 1, 2, \ldots, m_2, \tag{8}$$

Then we penalize (8) in the same way previously indicated for handling equality constraints by adding $P \cdot \sum_{i=1}^{m_2} (b_i^2 - \omega_i^2 - \sum_{j=1}^{n} a_{ij}^2 x_j)^2$ to the objective function.

Similarly, the inequality constraints of (6) are converted to equality constraints by subtracting surplus variables $\omega_i^3 \in \{0\} \cup Z^+$ and adding the penalty $P \cdot \sum_{i=1}^{m_3} (b_i^3 - \omega_i^3 - \sum_{j=1}^{n} a_{ij}^3 x_j)^2$ to the objective function.

These lead to the following QUBO reformulation for the original BIP of (3) through (6):

$$\begin{aligned} Minimize \sum_{j=1}^{n} c_j x_j + P \cdot \sum_{i=1}^{m_1} (b_i^1 - \sum_{j=1}^{n} a_{ij}^1 x_j)^2 + P \cdot \sum_{i=1}^{m_2} (b_i^2 - \omega_i^2 \\ - \sum_{j=1}^{n} a_{ij}^2 x_j)^2 + \sum_{i=1}^{m_3} (b_i^3 - \omega_i^3 - \sum_{j=1}^{n} a_{ij}^3 x_j)^2 \end{aligned} \tag{9}$$

The slack variables must be expressed as 0–1 variables, which is achieved with the use of binary expansions. This general QUBO reformulation can be improved by taking advantage of the known penalty functions of some special constraints as shown in Kochenberger and Glover (2006) and Glover et al. (2022a). As previously noted, the penalty constant P should be large enough to prevent violation of the constraints, but not too large to distort the information about the original objective function. An empirical rule-of-thumb that often works well is to choose P to lie between 75 to 150% of an estimated optimal objective value (Glover et al., 2022a).

4 QBA/QUBO for the uncapacitated facility location problem

We now describe the roadmap for implementing the uncapacitated facility location problem (UFLP, Drezner & Hamacher, 2004). The BIP formulation of UFLP goes beyond that of the graph-based optimization problems, and accordingly is not intuitive for the Ising spin model to handle directly. Due to its numerous practical applications, which motivate our

special focus on it, the UFLP has been extensively studied with various algorithms, including the recent artificial bee colony method (ABC, Kiran, 2015), modified binary vortex search (MBVS, Aslan & Pavone, 2023), and quantum annealing (Ding et al., 2021), making it a good candidate to gain computational experience for evaluating the merit of the QBA/QUBO approach.

4.1 The uncapacitated facility location problem (UFLP)

The general UFLP formulation arises by identifying a set I of candidate locations to build facilities to meet the demand of a set J of customers. Each customer must be served by one facility, and a facility can serve a customer only if it is selected. A facility $i \in I$, if selected, incurs a fixed cost of f_i. The cost for facility $i \in I$ to serve customer $j \in J$ is c_{ij}. The objective of the UFLP is to minimize the total operational costs consisting of the total fixed cost and the total cost of serving all customers. Define binary decision variable $y_i \in \{0, 1\}$ such that $y_i = 1$ if facility $i \in I$ is selected and $y_i = 0$ otherwise; also define binary decision variable $x_{ij} \in \{0, 1\}$ such that $x_{ij} = 1$ if customer $j \in J$ is assigned to facility $i \in I$ and $x_{ij} = 0$ otherwise. A BIP formulation of UFLP is then stated as follows.

$$Minimize \sum_{i \in I} f_i y_i + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} \tag{10}$$

$$\text{Subject to}: \sum_{i \in I} x_{ij} = 1, \ \forall j \in J \tag{11}$$

$$x_{ij} \leq y_i, \ \forall i \in I, \ j \in J \tag{12}$$

$$x_{ij}, \ \ y_i \in \{0, 1\} \tag{13}$$

The objective function (10) minimizes the total cost as the sum of total fixed cost and total service cost. Constraint (11) assigns exactly one facility to a customer, assuming single-sourcing. Constraint (12) ensures that no customer can be assigned to a facility if the facility is not selected. Constraint (13) completes the formulation by specifying the domain of the decision variables, to yield a BIP formulation with $|I| \times |J| + |I|$ decision variables and $|I| \times |J| + |J|$ constraints.

4.2 QUBO reformulation of UFLP

We observe that Constraint (11) is an equality, which can be handled in the same way as (4). Constraint (12) with the form of $u \leq v$ has a known penalty function (Glover et al., 2022a) given by:

$$u \leq v \rightarrow P \cdot (u - uv) \tag{14}$$

Thus, the QUBO reformulation of the UFLP can be written as:

$$\begin{aligned} &Minimize \sum_{i \in I} f_i y_i + \sum_{i \in I} \sum_{j \in J} c_{ij} x_{ij} + P \\ &\cdot \sum_{j \in J} \left(1 - \sum_{i \in I} x_{ij}\right)^2 + P \cdot \sum_{i \in I} \sum_{j \in J} (x_{ij} - x_{ij} y_i) \end{aligned} \tag{15}$$

The left-hand-side coefficient matrix of the QUBO formulation, called the Q matrix, can be obtained by expanding the third and fourth terms in (15), i.e., $P \cdot \sum_{j \in J} \left(1 - \sum_{i \in I} x_{ij}\right)^2$

and $P \cdot \sum_{i \in I} \sum_{j \in J} (x_{ij} - x_{ij} y_i)$, for each instance of UFLP. A numerical example is provided in the Appendix to demonstrate this.

Next, we present and prove some properties of the Q matrix in the QUBO formulation of UFLP, called Q-UFLP for convenience. Without loss of generality, we assume all the decision variables are organized in a row vector $\boldsymbol{z} = [y_1, y_2, \ldots, y_{|I|}, x_{11}, \ldots, x_{1|J|}, \ldots, x_{|I|1}, \ldots, x_{|I||J|}]$.

Property 1 The diagonal entries a_{ii} of Q-UFLP for $i = 1,2, \ldots, |I| \cdot |J|$ can be computed as: (i) $a_{ii} := f_i$ for $i \leq |I|$; and (ii) $a_{ii} := c_{kl}$ for $i > |I|$, where $k = \frac{i-|I|}{|J|}$ and $l = (i - |I|) mod |J|$.

Proof *Since $y_1, y_2, \ldots, y_{|I|}$ only appear in the first term of (15), and $y_i = y_i^2$, the entries of the first $|I|$ values in the diagonal are just the fixed costs of the facilities, thus (i) holds. For (ii), expanding the third and fourth terms of (15) and merging terms, we obtain the coefficient of x_{ij} (and x_{ij}^2) to be zero. Thus the entries of the remaining values in the diagonal are the service costs of the corresponding facility-customer pair. Q.E.D.*

Property 1 shows that the diagonal entries of Q-UFLP can be fully described by fixed costs f_i and variable costs c_{ij}, which are independent from the penalty value P. We then state and prove Property 2 for the non-diagonal entries of Q-UFLP.

Property 2 The entries of the upper-triangular part of Q-UFLP can be filled as follows. For $1 \leq i \leq |I|$ and $1 \leq j \leq |J|$: (i) fix $a_{i,i,(|I|+(i-1)\cdot|J|+j)} := -P$; and (ii) fix $a_{kl} := 2P$, where $k = |I| + (i-1) \cdot |J| + j$ and $l = \min\{|I| + m \cdot |J| + j, |I| \cdot (|J| + 1)\}$ for $m = i, \ldots, |I|$.

Proof *The upper-triangular part of Q-UFLP consists of the coefficients of cross-products of $y_i x_{ij}$ and the products of pairs of $x_{ij} x_{i'j}$ $(i' > i)$. Observe that $y_i x_{ij}$ only appears in the fourth term of (15), and that the coefficient of each product term $y_i x_{ij}$ is $-P$ for $1 \leq i \leq |I|$ and $1 \leq j \leq |J|$, thus (i) holds. Each term of $x_{ij} x_{i'j}$ $(i' > i)$ appears in the third term of (15) and has the coefficient of $2P$. For each pair of (i, j), the corresponding row in Q-UFLP is $|I| + (i-1) \cdot |J| + j$. Then in that row, we set the coefficients of all $x_{ij} x_{i'j}$ $(i' > i)$ to be $2P$, which corresponds to the columns of $|I| + m \cdot |J| + j$ for $m = i, \ldots, |I|$.*

Q.E.D.

Property 2 shows that the non-diagonal entries of Q-UFLP are either $-P$ or $2P$, and the Q-UFLP appears to be a sparse matrix.

4.3 Mapping between QUBO and Hamiltonian

Recall that the Ising spin variables take values in $\{-1, +1\}$, then there is a natural one-to-one mapping between the binary decision variables x_{ij} and y_i and their Ising spin counterparts: s_{ij} and s_i. Specifically, we have $x_{ij} = \frac{1+s_{ij}}{2}$ and $y_i = \frac{1+s_i}{2}$, resulting a total of $|I|(|J|+1)$ Ising spin variables. Substitute them into (15) yields the following Ising Hamiltonian, using a unified index for the Ising spin variables $k = 1,2, \ldots, |I|(|J|+1)$.

$$H(s) = \sum_{k=1}^{|I|(|J|+1)} h_k s_k + \sum_{l=1}^{|I|(|J|+1)} \sum_{k<l} J_{lk} s_l s_k + constant \tag{16}$$

where local fields h_k are the coefficients of the linear terms in the diagonal of the Q matrix, and J_{lk} are the couplers of the quadratic terms.

4.4 Implementation of QBA/QUBO for UFLP

The two properties suggest two different approaches to implement QUBO for UFLP, which differ in the way to generate the Q-UFLP matrix. The first approach, called the Penalty Function approach in the sequel, employs the algebraic formulation of (15) to populate the Q-UFLP matrix. The second approach, called the Q-Matrix approach in the sequel, utilizes Property 1 and Property 2 to directly specify the entries in the Q-UFLP matrix. Although the final Q-UFLP matrices generated are the same, the two approaches can differ in the efficiency of generating the Q-UFLP matrix. The Penalty Function approach relies on some optimization solver, e.g., Gurobi, Cplex, Xpress, to populate the Q-UFLP.

4.5 Computational study

We conduct a comprehensive computational study with a three-fold purpose: (i) to examine the effect of different penalty values P on the performance of the QBA/QUBO approach for UFLP; (ii) to examine the solution quality and efficiency of the QBA/QUBO approach for UFLP benchmark instances with different sizes and characteristics; and (iii) to compare the QBA/QUBO approach with benchmark metaheuristics and quantum annealing algorithms.

Table 1 summarizes the sizes and information about the benchmark instances used for our computational study. Class A includes two subsets of small-size instances from the ORLIB (Beasley, 1993). Two subsets of medium-size instances are included in Class B, where the set B-1 is from the ORLIB, and the set B-2 is from (Kochetov & Ivanenko, 2005) with large duality gap which are known to be hard to solve. Class C consists of large (asymmetric) instances from (Koerkel, 1989) and (Ghosh, 2003).

We implement the QBA/QUBO approach for UFLP via both the Q-Matrix and the Penalty Function approach, and use Gurobi 10's QUBO Solver (Gurobi, 2022) to solve the QUBO model. For each instance, we empirically choose the penalty value P from {25%, 50%, 75%, 100%} as the multiplier of the optimal objective value of the instance. All the computations using the Gurobi QUBO Solver were performed on a Windows Server with a 32-core 3.4 GHz CPU and 96 GB RAM with a limit of 3,600 s wall-clock time. The QUBO model makes it possible to implement D-Wave's quantum–classical hybrid solver (D-Wave, 2023), which we

Table 1 Summary of benchmark instances

Set Name		Number of Instances	Number of Facilities	Number of Customers	Number of Decision Variables	Number of Constraints	Additional Information
Class A	1	13	16	50	816	850	ORLIB
	2	12	25	50	1275	1300	ORLIB
Class B	1	12	50	50	2550	2550	ORLIB
	2	30	100	100	10,100	10,100	Large Duality Gap
Class C	1	15	250	250	62,750	62,750	Koerkel-Ghosh Asymmetric
	2	15	500	500	250,500	250,500	Koerkel-Ghosh Asymmetric

call QCHS for brevity, as an alternative way to solve the QUBO models for UFLP. We also report the results of the proprietary next generation quantum (NGQ) solver (Entanglement, 2023), with the default setting and a maximum time limit of 3600 s for the largest C-2 instances.

Comparisons are also made with four additional metaheuristic algorithms for a subset of the benchmark instances: the artificial bee colony (ABC, Kiran, 2015) algorithm, the artificial algae algorithm (AAA, Korkmaz et al., 2018), the modified binary vortex search (MBVS, Aslan & Pavone, 2023), and a custom designed hybrid classical-quantum annealer algorithm (Ding et al., 2021), which we call HCQA for short.

Solution quality is measured as the percentage of deviation of the best objective value ($Best$) from the optimal objective value (Opt), i.e., $(Best - Opt)/Opt \times 100$. For most of the benchmark instances in our computational study, their optimal objective values are known. For the large-size Class C instances, we use the best lower bound (LB) found by the branch-and-cut solver of Gurobi 10 within 3,600 s of wall-clock time to estimate the optimality gap by $(Best - LB)/LB \times 100$.

4.5.1 Results of the QBA/QUBO approach

Tables 2 and 3 show the effects of penalty P, a key parameter in the QBA/QUBO approach, on solution quality measured by average optimality gap (%) and average computational time to find the best solutions (in seconds), respectively. Note that no infeasibility is encountered even for small penalty coefficient 25% for all the instances. For the small-size Class A instances, all the coefficients of P values result in optimal solutions. The Q-Matrix approach takes more computational time on average, and has more variation than the Penalty Function approach; In medium-size Class B, the B-1 instances can be solved to optimality efficiently

Table 2 Effects of varying coefficient of penalty P on average optimality gap (%)

Set Name		Penalty Function				Q-Matrix			
		25%	50%	75%	100%	25%	50%	75%	100%
Class A	Set 1	0	0	0	0	0	0	0	0
	Set 2	0	0	0	0	0	0	0	0
Class B	Set 1	0	0	0	0	0	0	0	0
	Set 2	0	0	0	6.95	0	0	0	6.94

Table 3 Effects of varying coefficient of penalty P on average computational time (in seconds)

Set Name		Penalty Function				Q-Matrix			
		25%	50%	75%	100%	25%	50%	75%	100%
Class A	Set 1	1.2	1.45	1.23	1.3	5.26	18.89	11.56	28.46
	Set 2	2.75	4.15	3.73	2.12	2.82	10.68	28.4	3.35
Class B	Set 1	18.09	1.73	1.79	1.78	54.27	1.75	1.78	1.79
	Set 2	209.64	207.44	376.61	1030.73	131.93	320.87	413.21	769.93

although it has more than double of the size of those in Class A. But note that a low coefficient of 25% makes it significantly harder to solve the problem with both the Penalty Function and Q-Matrix approaches, i.e., with more than tenfold the time required when using other coefficient values. For the B-2 instances with large duality gaps, a large penalty coefficient of 100% leads to an average optimality gap of about 6.95% and requires significantly more time to reach optimality, while all the other coefficients yield optimality without difficulty.

The above observations corroborate the empirical premise that the penalty value P should not be too large or too small. They also provide empirical evidence of what an appropriate P would be for the UFLP. The 50% coefficient for penalty P seems to achieve the best overall performance in solution quality and efficiency, which will be chosen to present the remaining results in this section.

Given the stochastic nature of QCHS, multiple runs were performed on each instance in Class A. Table 4 reports the mean, standard deviation (std), minimum, maximum and coefficient of variation (cov) of the solution for each stance. The average cov is less than

Table 4 Statistics of QCHS solutions to Class A instances

Instance	Mean	Std	Min	Max	Cov
cap41	1,000,605	30,081.25	975,126.3	1,035,224	0.03
cap42	1,109,543	61,759.42	1,074,517	1,201,971	0.06
cap43	1,130,678	8542.125	1,118,860	1,138,325	0.00
cap44	1,262,713	35,156.68	1,240,853	1,314,961	0.03
cap51	1,142,892	28,364.11	1,108,819	1,174,206	0.02
cap61	1,045,929	55,769.99	1,003,978	1,127,914	0.05
cap62	1,101,897	74,360.72	1,045,962	1,205,659	0.07
cap63	1,181,420	54,238.65	1,109,892	1,241,016	0.05
cap64	1,256,746	50,102.71	1,213,593	1,306,454	0.04
cap71	947,699.1	4966.191	940,276.5	950,470.2	0.01
cap72	1,020,642	9071.871	1,012,970	1,030,987	0.01
cap73	1,093,589	11,356.45	1,083,949	1,105,987	0.01
cap74	1,208,661	13,351.46	1,188,949	1,218,487	0.01
cap81	901,532.5	35,660.41	850,090	932,101.6	0.04
cap82	1,026,141	60,949.59	976,910.8	1,113,964	0.06
cap83	1,151,934	35,137.97	1,119,013	1,189,750	0.03
cap84	1,351,237	129,337.4	1,261,383	1,538,959	0.10
cap91	848,745.9	25,198.25	832,291.2	885,471.6	0.03
cap92	1,065,271	87,378.23	952,291.2	1,159,865	0.08
cap93	1,179,956	175,140.3	1,072,291	1,438,709	0.15
cap94	1,326,538	148,493.6	1,252,291	1,549,278	0.11
cap101	819,166.2	7180.703	809,791.2	824,791.2	0.01
cap102	949,166.2	6250	939,791.2	952,291.2	0.01
cap103	1,063,541	10,103.63	1,054,791	1,072,291	0.01
cap104	1,233,541	12,500	1,227,291	1,252,291	0.01

Table 5 Results on set A-1 instances

Instance Name	Optimality Gap (%)				Computational Time (in seconds)			
	Penalty Function	Q-Matrix	QCHS	NGQ	Penalty Function	Q-Matrix	QCHS	NGQ
cap41	0.00	0.00	0.00	0.00	1.38	1.42	16.64	0.71
cap42	0.00	0.00	0.00	0.00	1.10	1.23	15.28	2.56
cap43	0.00	0.00	0.00	0.00	1.76	142.95	13.61	0.71
cap44	0.00	0.00	0.00	0.00	1.28	1.28	13.39	0.45
cap51	0.00	0.00	0.00	0.00	2.26	45.48	12.65	7.83
cap61	0.00	0.00	0.00	0.00	1.65	14.82	13.11	1.49
cap62	0.00	0.00	0.00	0.00	1.20	1.20	13.08	1.01
cap63	0.00	0.00	0.00	0.00	1.37	17.41	13.42	1.26
cap64	0.00	0.00	0.00	0.00	1.16	1.15	13.11	0.44
cap71	0.00	0.00	0.00	0.00	1.63	14.58	11.90	1.47
cap72	0.00	0.00	0.00	0.00	1.20	1.19	17.40	0.95
cap73	0.00	0.00	0.00	0.00	1.70	1.69	13.10	25.87
cap74	0.00	0.00	0.00	0.00	1.17	1.15	12.64	0.73
Average	**0.00**	**0.00**	**0.00**	**0.00**	**1.45**	**18.89**	**13.80**	**3.50**

5%. We also found that when giving sufficient time for QCHS, the variation of solutions diminishes.

Results of the three QBA/QUBO approaches on the small-size Class-A instances are presented in Tables 5 and 6. As shown in Table 4, all three approaches find optimal solutions to all the set A-1 instances. The Penalty Function approach clearly outperforms the other three in computational time. The NGQ solver performs better than the Q-Matrix approach and QCHS in computational time on average. For the set A-2 instances in Table 6, QCHS and NGQ find near-optimal solutions for all instances, with an average optimality gap of 0.02 and 0.01%, respectively. The Penalty Function and NGQ approaches outperform the other two in both solution quality and computational time.

Results on the medium-size Class-B instances are presented in Tables 7 and 8. As shown in Table 7, all the B-1 instances are solved to optimality by both the Penalty Function and Q-Matrix approaches with less than 2 s on average, although they are more than double the size of those in the Class A instances. This suggests that the efficiency of QUBO approach is affected not merely by the size of the problem/model. The QCHS has an average optimality gap of 0.12% and spends 17.55 s on average for the B-1 instances, while the average optimality gap of NGQ is 0.22% with 20.59 s.

The set B-2 instances with large duality gaps in Table 8 appear to be hard for both QCHS and NGQ to handle, with an average of more than 30% optimality gap. All the instances are solved to optimality by both the Penalty Function and Q-Matrix approaches. The Penalty Function approach is more efficient than Q-Matrix on average, although there are quite a few instances for which Q-Matrix spends less time than Penalty Function (highlighted in bold).

Tables 9 and 10 show the results on the large-size Class-C instances, for which we only report the better performed Penalty Function approach for QUBO. The QUBO solutions has an average optimality gap of 0.75% using 462.10 s on average. The QCHS has an average

Table 6 Results on set A-2 instances

Instance Name	Optimality Gap (%)				Computational Time (in seconds)			
	Penalty Function	Q-Matrix	QCHS	NGQ	Penalty Function	Q-Matrix	QCHS	NGQ
cap81	0.00	0.00	0.00	0.00	3.67	3.66	20.70	3.97
cap82	0.00	0.00	0.02	0.00	3.30	3.14	13.61	2.39
cap83	0.00	0.00	0.00	0.09	3.27	4.25	13.11	10.29
cap84	0.00	0.00	0.00	0.00	11.24	10.88	13.11	10.58
cap91	0.00	0.00	0.00	0.00	2.73	2.73	13.11	10.17
cap92	0.00	0.00	0.16	0.00	3.78	6.34	13.54	0.80
cap93	0.00	0.00	0.00	0.00	4.14	72.23	13.91	3.38
cap94	0.00	0.00	0.00	0.00	3.66	3.89	18.98	19.54
cap101	0.00	0.00	0.00	0.00	3.94	2.68	14.99	0.83
cap102	0.00	0.00	0.04	0.00	1.70	1.68	20.01	4.57
cap103	0.00	0.00	0.00	0.03	4.03	13.51	14.34	3.87
cap104	0.00	0.00	0.00	0.00	4.31	3.14	16.06	4.48
Average	**0.00**	**0.00**	**0.02**	**0.01**	**4.15**	**10.68**	**15.46**	**6.24**

Table 7 Results on the set B-1 instances

Instance Name	Optimality Gap (%)				Computational Time (in seconds)			
	Penalty Function	Q-Matrix	QCHS	NGQ	Penalty Function	Q-Matrix	QCHS	NGQ
cap111	0.00	0.00	0.11	0.14	1.61	1.59	21.13	19.74
cap112	0.00	0.00	0.58	0.31	1.64	1.72	16.48	16.43
cap113	0.00	0.00	0.10	0.30	1.70	1.72	18.87	24.12
cap114	0.00	0.00	0.00	0.08	1.75	1.75	17.19	28.69
cap121	0.00	0.00	0.00	0.18	1.75	1.76	21.83	21.25
cap122	0.00	0.00	0.14	0.45	1.77	1.83	16.43	28.32
cap123	0.00	0.00	0.10	0.15	1.83	1.81	20.00	18.03
cap124	0.00	0.00	0.00	0.07	1.81	1.86	15.12	8.01
cap131	0.00	0.00	0.16	0.44	1.59	1.58	16.29	17.82
cap132	0.00	0.00	0.22	0.28	1.72	1.72	17.73	19.81
cap133	0.00	0.00	0.00	0.26	1.80	1.73	15.24	27.97
cap134	0.00	0.00	0.00	0.00	1.84	1.89	14.32	16.92
Average	**0.00**	**0.00**	**0.12**	**0.22**	**1.73**	**1.75**	**17.55**	**20.59**

Table 8 Results on the set B-2 instances with large duality gaps

Instance Name	Optimality Gap (%)				Computational Time (in seconds)			
	Penalty Function	Q-Matrix	QCHS	NGQ	Penalty Function	Q-Matrix	QCHS	NGQ
1032GapAS	0.00	0.00	16.65	124.37	50.80	**11.67**	40.98	254.74
1132GapAS	0.00	0.00	41.37	115.83	225.09	**213.38**	50.73	292.02
1232GapAS	0.00	0.00	15.46	99.53	925.64	3063.49	36.85	116.07
1332GapAS	0.00	0.00	41.51	90.99	164.54	**160.59**	35.57	234.56
1432GapAS	0.00	0.00	24.94	99.36	3.75	3.76	34.28	292.94
1532GapAS	0.00	0.00	16.62	82.73	208.86	586.53	41.88	219.58
1632GapAS	0.00	0.00	24.93	91.06	195.05	**138.10**	54.15	186.49
1732GapAS	0.00	0.01	33.22	99.31	54.83	483.51	31.06	252.10
1832GapAS	0.00	0.00	41.50	91.08	14.84	452.63	36.03	232.38
1932GapAS	0.00	0.00	33.18	99.44	149.72	**13.49**	96.32	294.96
2032GapAS	0.00	0.00	33.18	107.63	336.71	**297.54**	131.32	125.67
2132GapAS	0.00	0.00	33.15	124.28	133.49	247.51	57.91	123.89
2232GapAS	0.00	0.00	33.20	107.74	210.69	**52.03**	34.06	211.39
2332GapAS	0.00	0.00	41.34	107.49	330.69	833.05	79.38	277.24
2432GapAS	0.00	0.00	24.96	91.10	492.30	**92.88**	73.72	194.56
2532GapAS	0.00	0.00	41.40	107.65	150.22	268.52	43.68	200.23
2632GapAS	0.00	0.00	41.41	132.42	200.64	206.47	105.18	197.40
2732GapAS	0.00	0.00	15.49	84.20	103.20	495.88	130.01	206.61
2832GapAS	0.00	0.00	8.33	107.67	61.91	383.08	35.79	119.73
2932GapAS	0.00	0.00	49.70	124.20	4.47	**4.44**	45.44	22.97
3032GapAS	0.00	0.00	16.64	99.50	288.33	485.77	88.20	122.62
3132GapAS	0.00	0.00	30.66	99.46	307.65	**83.24**	89.02	209.00
3232GapAS	0.00	0.00	33.09	124.15	195.74	**125.64**	43.80	215.81
332GapAS	0.00	0.00	24.90	124.21	105.75	**7.83**	43.78	279.05
432GapAS	0.00	0.00	41.45	91.01	248.44	**203.13**	112.43	290.96
532GapAS	0.00	0.00	24.85	99.41	396.42	**134.00**	35.84	180.26
632GapAS	0.00	0.00	33.10	107.59	84.14	**15.41**	34.94	169.03
732GapAS	0.00	0.00	18.07	126.38	285.06	445.33	34.44	277.20
832GapAS	0.00	0.00	33.16	82.86	25.93	34.40	82.49	255.85
932GapAS	0.00	0.00	33.17	99.38	268.18	**82.69**	76.70	149.40
Average	**0.00**	**0.00**	**30.02**	**104.73**	**207.44**	**320.87**	**61.20**	**206.82**

optimality gap of 2.45%, which is significantly lower than that for the set B-2 instances, although C-1 instances have more than six times of the size of the B-2 instances. This suggests that the performance of quantum annealer, in this current implementation by D-Wave, can be more susceptible to the polyhedral properties of a UFLP instance than to the size of an instance. NGQ spends about half of the time of QCHS with slightly higher average optimality gap of 3.85%.

Table 9 Results on the set C-1 instances

Instance Name	Optimality Gap (%)			Computational Time (in seconds)		
	QUBO	QCHS	NGQ	QUBO	QCHS	NGQ
ga250a-1	0.04	1.45	2.21	447.97	192.46	191.25
ga250a-2	0.03	1.27	2.26	447.27	167.25	147.57
ga250a-3	0.14	1.49	2.29	454.61	157.29	80.60
ga250a-4	0.07	1.63	2.16	454.06	149.13	170.87
ga250a-5	0.09	1.47	2.10	449.28	150.85	180.44
ga250b-1	1.01	4.47	4.25	462.77	194.84	196.11
ga250b-2	0.62	3.68	4.43	452.80	145.61	197.20
ga250b-3	0.84	3.89	3.15	461.78	139.00	195.42
ga250b-4	0.99	4.16	3.57	461.77	137.87	198.67
ga250b-5	0.74	3.41	4.04	459.58	143.26	190.32
ga250c-1	1.36	1.55	5.76	440.10	213.92	86.21
ga250c-2	0.37	3.17	4.80	485.36	2006.50	196.88
ga250c-3	2.38	1.12	6.42	472.60	478.03	190.00
ga250c-4	2.06	2.53	3.35	476.16	470.31	199.15
ga250c-5	1.17	0.62	6.94	483.33	486.58	199.18
Average	**0.75**	**2.45**	**3.85**	**462.10**	**358.50**	**174.66**

Table 10 Results on the set C-2 instances

Instance Name	Optimality Gap (%)			Computational Time (in seconds)		
	QUBO	QCHS	NGQ	QUBO	QCHS	NGQ
ga500a-1	7.07	10.79	3.08	267.44	45.50	3547.52
ga500a-2	7.20	10.74	1.54	278.49	45.87	3561.04
ga500a-3	6.47	10.41	3.25	278.96	44.57	3603.15
ga500a-4	7.13	10.85	1.47	293.49	47.51	3577.44
ga500a-5	6.87	10.63	1.91	287.42	46.36	3588.73
ga500b-1	4.08	95.34	3.39	297.02	45.93	3604.57
ga500b-2	4.48	95.78	2.72	344.84	47.37	3550.99
ga500b-3	3.53	94.58	3.93	360.50	48.02	3670.41
ga500b-4	4.10	92.69	2.99	379.11	48.65	3602.59
ga500b-5	4.06	92.30	2.74	389.18	47.39	3603.30
ga500c-1	10.52	497.40	6.56	280.38	52.86	2361.71
ga500c-2	12.88	495.84	9.47	279.16	53.32	3584.19
ga500c-3	11.08	508.22	9.20	280.06	54.14	3558.60
ga500c-4	11.10	508.46	11.64	280.69	53.19	3515.06
ga500c-5	11.48	497.17	6.52	281.17	54.55	3573.81
Average	**7.47**	**202.08**	**4.69**	**305.19**	**49.02**	**3500.21**

For even larger instances in set C-2 with 500 facilities, 500 customers, and more than 250,000 binary decision variables, Table 10 shows that the QUBO solutions have an average optimality gap of 7.47%. NGQ outperforms both QUBO and QCHS in solution quality with the same time limit of one hour. These instances are very challenging for QCHS to handle, which has an average optimality gap of over 200%.

4.5.2 Comparison with benchmark algorithms

We compare the QBA/QUBO solutions of 12 instances from the sets A-1, A-2 and B-1, with four recent benchmark algorithms in the literature: ABC (Kiran, 2015), AAA (Korkmaz et al., 2018)), MBVS (Aslan & Pavone, 2023)), and HCQA (Ding et al., 2021) in Tables 11 and 12. Since all four algorithms are probabilistic in nature, multiple runs on the same instance are needed for computation. The optimality gap and computational time of AAA and MBVS in Table 11 are averages, except that those of ABC are best solutions (optimality gaps) found. The percentage in a parenthesis shows the percentage of runs the algorithm reaches its best solution. ABC has low success rates for three instances in the B-1 set: cap131, cap132 and cap133. Both AAA and MBVS have high success rates for most of the instances. Note that HCQA, a custom designed hybrid classical-quantum annealing algorithm implemented on the D-Wave quantum computer, does not find optimal solutions for most of the instances. All four QBA/QUBO algorithms: QCHS, NGQ, Q-Matrix and Penalty Function, find optimal solutions to all the instances. As for computational time, the Penalty Function and Q-Matrix approaches use only fractions of that required by others.

5 Conclusions and future research

With the continuing development and growth of the quantum computing technology, the quantum annealing and quantum adiabatic evolution algorithms are generally anticipated to have significant potential for efficiently solving numerous NP-hard combinatorial optimization problems, including many in supply chains, although presently these approaches are not competitive with leading methods implemented on classical computers. Since the constructs in quantum physics, e.g., the Ising variable in a spin glass model, are not amenable to modeling complex real world optimization applications, gaps exit for exploiting quantum advantage. In this paper, we have identified three gaps: optimization modeling, algorithm design, and solution assessment, and elaborated how QUBO can be applied in the quantum bridge analytics (QBA) framework to bridge these gaps between quantum computing and classical optimization. We also provide a roadmap for modeling building and algorithm implementation to explore and exploit quantum computing for real world optimization applications.

We apply our QBA/QUBO approach for a well-known NP-hard network design problem in supply chain, i.e., the uncapacitated facility location problem (UFLP). Detailed steps are presented to reformulate the BIP model of UFLP to a QUBO model. Properties of the Q matrix are obtained to describe its structure and specification. The QUBO model allows one to apply optimization methods in both the classical optimization community and quantum computing for solving the UFLP.

A comprehensive computational experiment is performed on benchmark instances of UFLP with the following main findings. First, we find that a penalty value of 50% of the objective value appears to be a good choice for the QUBO approach for UFLP, while a penalty value that is appreciably smaller or larger is notably less effective. Secondly, our

Table 11 Comparison of optimality gap (%) with benchmark algorithms

Instance Name	ABC	AAA	MBVS	HCQA	QCHS	NGQ	Q-Matrix	Penalty Function
cap71	0.00	0.00	0.00	0.06	0.00	0.00	0.00	0.00
cap72	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.00
cap73	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
cap74	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
cap101	0.00	0.00	0.00	0.13	0.00	0.00	0.00	0.00
cap102	0.00	0.00	0.00	0.03	0.00	0.00	0.00	0.00
cap103	0.00 (83%)	0.00	0.00 (93%)	0.12	0.00	0.03	0.00	0.00
cap104	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
cap131	0.00 (20%)	0.00	0.00	0.33	0.00	0.44	0.00	0.00
cap132	0.00 (47%)	0.00	0.00	0.09	0.00	0.28	0.00	0.00
cap133	0.00 (17%)	0.00 (97%)	0.00	0.05	0.00	0.26	0.00	0.00
cap134	0.00 (100%)	0.00	0.00	0.00	0.00	0.00	0.00	0.00

1. Best objective value found in 30 runs on the same instance
2. Average objective value found in 30 runs on the same instance

Table 12 Comparison of computational time (in seconds) with benchmark algorithms

Instance Name	ABC	AAA	MBVS	HCQA	QCHS	NGQ	Q-Matrix	Penalty Function
cap71	9.09	25.09	–	–	11.90	1.47	14.58	1.63
cap72	8.98	25.23	–	–	17.40	0.95	1.19	1.20
cap73	8.99	25.24	–	–	13.10	25.87	1.69	1.70
cap74	8.95	25.18	–	–	12.64	0.73	1.15	1.17
cap101	14.45	26.35	–	–	14.99	0.83	2.68	3.94
cap102	14.48	26.12	–	–	20.01	4.57	1.68	1.70
cap103	14.38	26.23	–	–	14.34	3.87	13.51	4.03
cap104	14.33	25.80	–	–	16.06	4.48	3.14	4.31
cap131	33.85	26.76	–	–	16.29	17.82	1.58	1.59
cap132	33.83	26.83	–	–	17.73	19.81	1.72	1.72
cap133	33.62	26.68	–	–	15.24	27.97	1.73	1.80
cap134	33.51	26.82	–	–	14.32	16.92	1.89	1.84

1. No computational time was reported by the authors
2. No computational time was reported, but the authors mentioned "*one trial of solving a large QUBO matrix could be very time-consuming (about 15 min for a 900 by 900 QUBO matrix)*"

QBA-QUBO approach using Gurobi's QUBO solver, D-Wave's quantum–classical hybrid solver (QCHS), and the NGQ solver shows good performance for the selected benchmark instances: They solve all the small-size and some medium-size instances with up to 10,000 decision variables to optimality in less than 20 s, clearly outperforming some metaheuristics for UFLP in the literature; for large-size instances with more than 60,000 binary decision variables, they are able to find near-optimal solutions (less than 3% optimality gap) in less than 10 min. The largest instances with more than 250,000 binary decision variables turn out to be very challenging for QCHS with an average optimality gap of more than 200%, while NGQ is able to achieve less than 5% optimality gap on average with the same computational time limit. Another key finding is that the difficulty for quantum optimization to solve UFLP depends not only on the size of an instance, but also the polyhedral property of an instance. We find that UFLP instances with large duality gaps are much more difficult for both QCHS and NGQ to handle.

Our findings disclose the value for the optimization community of exploring quantum and quantum-inspired algorithms for other large-scale optimization applications in supply chains. While some existing methods of Quantum Bridge Analytics through QUBO reformulation are readily adaptable to supply chain optimization problems, we see that it is important to leverage theoretical development in mathematical programming on polyhedral analysis and relaxation techniques in this effort.

A limitation of our work is that the quantum classical hybrid solver tested in this paper does not provide the information needed to clearly differentiate the role and contribution of the quantum-based method versus those of the classical method in obtaining the solutions found. Future development of hybrid algorithms with transparency between the classical and quantum-based methods will be valuable. Future research efforts are also needed to explore quantum utility and quantum advantage for a variety of real world supply chain optimization problems, including production lot-sizing, resource planning, machine scheduling, project

scheduling, and vehicle routing, which have more complex mathematical programming formulations. First, they may have a mixture of binary, integer and continuous decision variables, requiring binary encoding that can expand the size of the QUBO reformulation. Secondly, they often have large number of general inequality constraints requiring the use of slack or surplus variables that produce QUBO formulations that are more difficult to solve. In addition, real world optimization applications can easily involve thousands of decision variables and constraints. Thus, the issue of scaling quantum optimization algorithms remains a challenge. Finally, while many existing studies report the use of quantum annealing, other types of quantum optimization algorithms exist, notably including the gate-based algorithms represented by the Quantum Approximation Optimization Algorithm and the Variational Quantum Eigensolver, that can run on universal quantum computers in the Noisy Intermediate-Scale Quantum category.

Appendix

A numerical example is provided to demonstrate the QUBO reformulation for the UFLP. Consider 3 candidate facilities to serve 4 customer locations. The fixed operating costs of facilities and the costs of facilities serving customer locations are provided in the following table (in $ thousand).

Facility	Fixed Cost	Customer 1	Customer 2	Customer 3	Customer 4
1	100	35	40	60	60
2	125	40	55	50	60
3	125	40	30	45	50

The BIP formulation (3) through (7) can be expanded as the follows.

$$\begin{aligned} Minimize\ & 100y_1 + 125y_2 + 125y_3 + 35x_{11} + 40x_{12} + 60x_{13} + 60x_{14} + 40x_{21} \\ & +55x_{22} + 50x_{23} + 60x_{24} + 40x_{31} + 30x_{32} + 45x_{33} + 50x_{34} \end{aligned}$$

$$\text{Subject to}: x_{11} + x_{21} + x_{31} = 1 \tag{17}$$

$$x_{12} + x_{22} + x_{32} = 1 \tag{18}$$

$$x_{13} + x_{23} + x_{33} = 1 \tag{19}$$

$$x_{14} + x_{24} + x_{34} = 1 \tag{20}$$

$$x_{11} \leq y_1 \tag{21}$$

$$x_{12} \leq y_1 \tag{22}$$

$$x_{13} \leq y_1 \tag{23}$$

$$x_{14} \leq y_1 \tag{24}$$

$$x_{21} \leq y_2 \tag{25}$$

$$x_{22} \leq y_2 \tag{26}$$

$$x_{23} \leq y_2 \tag{27}$$

$$x_{24} \leq y_2 \tag{28}$$

$$x_{31} \leq y_3 \tag{29}$$

$$x_{32} \leq y_3 \tag{30}$$

$$x_{33} \leq y_3 \tag{31}$$

$$x_{34} \leq y_3 \tag{32}$$

Penalizing (A1–A4) as in (4), the third term $P \cdot \sum_{j \in J} \left(1 - \sum_{i \in I} x_{ij}\right)^2$ in (15) becomes:

$$P \cdot (1 - x_{11} - x_{21} - x_{31})^2 + P \cdot (1 - x_{12} - x_{22} - x_{32})^2$$
$$+ P \cdot (1 - x_{13} - x_{23} - x_{33})^2 + P \cdot (1 - x_{14} - x_{24} - x_{34})^2$$

In general, the above is the summation of $|J|$ (the cardinality of set J) quadratic terms, as there are $|J|$ equality constraints in the BIP formulation.

Applying (14) penalize constraints (A5–A16), the fourth term $P \cdot \sum_{i \in I} \sum_{j \in J} (x_{ij} - x_{ij} y_i)$ in (15) yields:

$$P \cdot (x_{11} - x_{11} y_1) + P \cdot (x_{12} - x_{12} y_1) + P \cdot (x_{13} - x_{13} y_1) + P \cdot (x_{14} - x_{14} y_1)$$
$$+ P \cdot (x_{21} - x_{21} y_2) + P \cdot (x_{22} - x_{22} y_2) + P \cdot (x_{23} - x_{23} y_2) + P \cdot (x_{24} - x_{24} y_2) +$$

$$P \cdot (x_{31} - x_{31} y_3) + P \cdot (x_{32} - x_{32} y_3) + P \cdot (x_{33} - x_{33} y_3) + P \cdot (x_{34} - x_{34} y_3) +$$

In general, the above is the summation of $|I| \times |J|$ terms, as there are $|I| \times |J|$ inequality constraints in the BIP formulation.

Choosing $P = 250$, the entire QUBO formulation for this example has the following $\boldsymbol{Q}$ matrix:

$$\begin{bmatrix}
100 & 0 & 0 & -250 & -250 & -250 & -250 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 125 & 0 & 0 & 0 & 0 & 0 & -250 & -250 & -250 & -250 & 0 & 0 & 0 & 0 \\
0 & 0 & 125 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -250 & -250 & -250 & -250 \\
0 & 0 & 0 & 35 & 0 & 0 & 0 & 500 & 0 & 0 & 0 & 500 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 40 & 0 & 0 & 0 & 500 & 0 & 0 & 0 & 500 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 60 & 0 & 0 & 0 & 500 & 0 & 0 & 0 & 500 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 60 & 0 & 0 & 0 & 500 & 0 & 0 & 0 & 500 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 40 & 0 & 0 & 0 & 500 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 55 & 0 & 0 & 0 & 500 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 50 & 0 & 0 & 0 & 500 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 60 & 0 & 0 & 0 & 500 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 40 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 30 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 45 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 50
\end{bmatrix}$$

Solving it to optimality produces the solution $y_3 = 1$, $x_{31} = x_{32} = x_{33} = x_{34} = 1$, with an objective value of -710. Adding the constant of 1000, one obtains the optimal objective value of 290 to the original problem.

Acknowledgements Dr. Haitao Li's work was partially supported by the U.S. Department of Transportation through the University of Nebraska-Lincoln Mid-America Transportation Center (MATC) under contract number 69A3552348307. The findings and opinions expressed in this paper are those of the authors and do not reflect the views of the research sponsor. We thank Mr. Avinash Chaluvadi, graduate research assistant at the University of Missouri – St. Louis, for his work on the computational experiment.

Funding U.S. Department of Transportation, 69A3552348307, Haitao Li.

Declarations

Conflict of interest Haitao Li has received a research grant from the U.S. Department of Transportation. Gary Kochenberger declares that he has no conflict of interest. Fred Glover declares that he has no conflict of interest. Rick Hennig declares that he has no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

Aarts, E., & Lenstra, J. K. (2003). *Local search in combinatorial optimization*. Princeton University Press.

Anderson, P. W. (1978). The concept of frustration in spin glasses. *Journal of the Less Common Metals, 62*, 291–294.

Aslan, M., & Pavone, M. (2023). MBVS: A modified binary vortex search algorithm for solving uncapacitated facility location problem. *Neural Computing and Applications*. https://doi.org/10.1007/s00521-00023-09190-00529

Bapst, V., Foini, L., Krzakala, F., Semerjian, G., & Zamponi, F. (2013). The quantum adiabatic algorithm applied to random optimization problems: The quantum spin glass perspective. *Physics Reports, 523*(3), 127–205.

Barahona, F. (1982). On the computational complexity of Ising spin glass models. *Journal of Physics a: Mathematical and General, 15*(10), 3241.

Beasley, J. E. (1993). Lagrangean heurisics for location problems. *European Journal of Operational Research, 65*, 383–399.

Bertsimas, D. & Weismantel, R. (2005). *Optimization over Integers*, Dynamic Ideas.

Brooke, J., Bitko, D., Rosenbaum, T. F., & Aeppli, G. (1999). Quantumm annealing of a disordered magnet. *Science (New York, N.y.), 284*(5415), 779–781.

Carugno, C., Dacrema, M. F., & Cremonesi, P. (2022). Evaluating the job shop scheduling problem on a D-wave quantum annealer. *Scientific Reports, 12*, 6539.

Childs, A. M., Farhi, E., Goldstone, J., & Gutmann, S. (2002). Finding cliques by quantum adiabatic evolution. *Quantum Information & Computation, 2*(3), 181–191.

D-Wave. (2023). *D-Wave Quantum Advantage Hybrid Solver*.

Das, A., & Chakrabarti, B. K. (2008). Quantum annealing and analog quantum computation. *Reviews of Modern Physics, 80*, Article 1061.

de Wolf, R. (2023). *Quantum Computing: Lecture Notes*, University of Amsterdam.

Dickson, N. G., & Amin, M. H. S. (2011). Does adiabatic quantum optimization fail for NP-complete problems? *Physical Review Letters, 106*, Article 050502.

Ding, Y., Chen, X., Lamata, L., Solano, E., & Sanz, M. (2021). Implementation of a hybrid classical-quantum annealing algorithm for logistic network design. *SN Computer Science*. https://doi.org/10.1007/s42979-42021-00466-42972

Drezner, Z. & Hamacher, H. W. (2004). *Facility Location: Applications and Theory*.

Du, Y., Wang, H., Kennig, R., Hulandageri, A., Kochenberger, G., & Glover, F. (2023). New advances for quantum-inspired optimization. *International Transactions in Operational Research*. https://doi.org/10.1111/itor.13420

Edwards, S. F., & Anderson, P. W. (1975). Theory of spin glasses. *Journal of Physics f: Metal Physics, 5*, 965–974.

Entanglement (2023). https://www.entanglement.com.

Farhi, E., Goldstone, J., Gutmann, S., Lapan, J., Lundgren, A., & Preda, D. (2001). A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem. *Science, 292*(5516), 472–475.

Farhi, E., Goldstone, J., Gutmann, S. & Sipser, M. (2000*). Quantum computation by adiabatic evolution, Center for Theoretical Physics*. MIT**:** MIT-CTP-2936.

Farhi, E., Gosset, D., Hen, I., Sandvik, A. W., Shor, P., Young, A. P., & Zamponi, F. (2012). Performance of the quantum adiabatic algorithm on random instances of two optimization problems on regular hypergraphs. *Physical Review A, 86*, Article 052334.

Geitz, M., Grozea, C., Steigerwald, W., Stohr, R., & Wolf, A. (2022). Solving the extended job shop scheduling problem with AGVs - Classical and quantum approaches. *Lecture Notes in Computer Science., 13292*, 120–137.

Gendreau, M., & Potvin, J.-Y. (2019). *Handbook of metaheuristics*. Spinger.

Ghosh, D. (2003). Neighborhood search heuristics for the uncapacitated facility location problem. *European Journal of Operational Research, 150*, 150–162.

Glover, F. (1997). A template for scatter search and path relinking. *Lecture Notes in Computer Science*. J. K. Hao, E. Lutton, E. Ronald, M. Schoenaurer and D. Snyers. *1363***,** 13-54.

Glover, F., Klingman, D., & Phillips, N. V. (1992). *Network Models in Optimization and Their Applications in Practice*. Wiley-Interscience.

Glover, F., & Kochenberger, G. (2005). *Handbook of Metaheuristics*. Springer.

Glover, F., Kochenberger, G., Hennig, R., & Du, Y. (2022a). Quantum bridge analytics I: A tutorial on formulating and using QUBOmodels. *Annals of Operations Research, 314*, 141–183.

Glover, F., Kochenberger, G., Ma, M., & Du, Y. (2022b). Quantum bridge analytics II: QUBO-Plus, network optimization and combinatorial chaining for asset exchange. *Annals of Operations Research, 314*, 185–212.

Glover, F., & Laguna, M. (1997). *Tabu Search*. Kluwer Academic Publishers.

Guillaume, A., Goh, E. Y., Johnston, M. D., Wilson, B. D., Ramanan, A., Tibble, F., & Lackey, B. (2016). Deep Space Network scheduling using quantum annealing. *IEEE Transactions on Quantum Engineering, 3*, Article 3102413.

Gurobi (2022). *Gurobi Optimizer Reference Manual*, Gurobi Inc.

Hammer, P. L., Rosenberg, I., & Rudeanu, S. (1963). On the determination of the minima of pseudoboolean functions. *Studiisi Cercetari Matematice, 14*, 359–364.

Ikeda, K., Nakamura, Y., & Humble, T. S. (2019). Application of quantum annealing to nurse scheduling problem. *Scientific Reports, 9*, Article 12837.

Kadowaki, T., & Nishimori, H. (1998). Quantum annealing in the transverse Ising model. *Physical Review E, 58*(5), 5355–5363.

Kiran, M. S. (2015). The continuous Artificial Bee Colony algorithm for binary optimization. *Applied Soft Computing, 33*, 15–23.

Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science, 220*, 671–680.

Koch, T., Neira, D., Chen, Y., Cortiana, G., Egger, D., Heese, R., Hegade, N., Cadavid, A. G. (2025). *Quantum optimization benchmark library - The intractable decathlon, Quantum Optimization Working Group*.

Kochenberger, G. & Glover, F. (2006). A unified framework formodeling and solving combinatorial optimization problems: A tutorial. *Multiscal Optimization Methods and Applications*. W. Hager, S.-J. Huang, P. Pardalos and O. Prokopyev, Springer**:** 101–124.

Kochenberger, G., Hao, J.-K., Glover, F., Lewis, M., Lu, Z., Wang, H., & Wang, Y. (2014). The unconstrained binary quadratic programming problem: A survey. *Journal of Combinatorial Optimization, 28*, 58–81.

Kochetov, Y., & Ivanenko, D. (2005). Computationally difficult instances for the uncapacitated facility location problem. *Metaheuristics: Progress as Real Problem Solvers*. T. Ibaraki, K. Nonobe and M. Yagiura. *Boston, MA, Springer., 32*, 351–367.

Koerkel, M. (1989). On the exact solution of large-scale simple plant location problems. *European Journal of Operational Research, 39*, 157–173.

Korkmaz, S., Babalik, A., & Kiran, M. S. (2018). An artificial algae algorithm for solving binary optimization problems. *International Journal of Machine Learning and Cybernetics, 9*, 1233–1247.

Li, H. (2023). *Optimization Modeling for Supply Chain Applications*. World Scientific Publisher.

Marriott, K., & Stuckey, P. J. (1998). *Programming with Constraints*. Massachusetts, The MIT Press.

Messiah, A. (1962). *Quantum Mechanics* (Vol. 2). Norh-Holland.

Nemhauser, G., & Wolsey, L. (1988). *Integer and Combinatorial Optimization*. John Wiley.

Punnen, A. (2022). *The Quadratic Unconstrained Binary Optimization Problem: Theory, Algorithms and Applications*. Switzerland, Springer.
Ray, P., Chakrabarti, B. K., & Chakrabarti, A. (1989). Sherrington-Kirkpatrick model in a transverse field: Absence of replica symmetry breaking due to quantum fluctuations. *Physics Review B, 39*, 11828.
Vanderbei, R. J. (2020). *Linear Programming: Foundations and Extensions*. Springer.
Venturelli, D., Marchand, D. J. J. & Rojo, G. (2016). *Job shop scheduling solver based on quantum annealing*. arXiv:1506.08479.