POLITECNICO DI TORINO
Repository ISTITUZIONALE

Improving the Solving of Optimization Problems: A Comprehensive Review of Quantum Approaches

Original
Improving the Solving of Optimization Problems: A Comprehensive Review of Quantum Approaches / Volpe, D., Orlandi, G., Turvani, G.. - In: QUANTUM REPORTS. - ISSN 2624-960X. - 7:1(2025), pp. 1-19. [10.3390/quantum7010003]

Availability:
This version is available at: 11583/2996287 since: 2025-01-07T09:55:39Z

Publisher:
MDPI

Published
DOI:10.3390/quantum7010003

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright

(Article begins on next page)

18 November 2026

quantum reports

MDPI

Review

Improving the Solving of Optimization Problems: A Comprehensive Review of Quantum Approaches

Deborah Volpe *,†, Giacomo Orlandi *,† and Giovanna Turvani

Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Turin, Italy; giovanna.turvani@polito.it

* Correspondence: deborah.volpe@polito.it (D.V.); giacomo.orlandi@polito.it (G.O.)

† These authors contributed equally to this work.

Abstract: Optimization is a crucial challenge across various domains, including finance, resource allocation, and mobility. Quantum computing has the potential to redefine the way we handle complex problems by reducing computational complexity and enhancing solution quality. Optimization, particularly of objective functions, stands to benefit significantly from quantum solvers, which leverage principles of quantum mechanics like superposition, entanglement, and tunneling. The Ising and Quadratic Unconstrained Binary Optimization (QUBO) models are the most suitable formulations for these solvers, involving binary variables and constraints treated as penalties within the overall objective function. To harness quantum approaches for optimization, two primary strategies are employed: exploiting quantum annealers—special-purpose optimization devices—and designing algorithms based on quantum circuits. This review provides a comprehensive overview of quantum optimization methods, examining their advantages, challenges, and limitations. It demonstrates their application to real-world scenarios and outlines the steps to convert generic optimization problems into quantum-compliant models. Lastly, it discusses available tools and frameworks that facilitate the exploration of quantum solutions for optimization tasks.

Keywords: QUBO; quantum computing; design automation; quantum optimization; quantum annealer; quantum circuit model; quantum-inspired optimization

check for updates

Academic Editor: Lev Vaidman

Received: 31 October 2024
Revised: 31 December 2024
Accepted: 2 January 2025
Published: 7 January 2025

Citation: Volpe, D.; Orlandi, G.; Turvani, G. Improving the Solving of Optimization Problems: A Comprehensive Review of Quantum Approaches. *Quantum Rep.* **2025**, *7*, 3. https://doi.org/10.3390/quantum7010003

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Solving optimization problems is a fundamental challenge across various research and industrial domains, including finance [1], automatic control systems [2], machine learning [3], and telecommunications [4].

Many optimization problems belong to the NP (Nondeterministic Polynomial time) complexity class, which includes decision problems that, although not solvable in polynomial time, can be verified in polynomial time using classical strategies [5]. Particularly, NP-complete problems are significant since every problem in NP can be reduced to them in polynomial time. Solving any NP-complete problem in polynomial time would imply P = NP. In contrast, NP-hard problems are at least as difficult as NP-complete problems but may not be verifiable in polynomial time and are not limited to decision problems, encompassing many optimization tasks.

The growing need for efficient solutions to large-scale problems has driven continuous innovation, fostering new approaches that better explore solution spaces [6].

With advances in quantum computing [7], interest has risen in evaluating its potential for optimization. Quantum computing could reduce computational complexity and

improve solution quality, addressing challenges in diverse fields such as healthcare, energy systems, and finance. Notable examples include grid management [8] and portfolio optimization [9]. By enhancing optimization techniques, quantum approaches can bridge the gap between computational methods and real-world applications.

Currently, two main methods are employed: quantum annealers (QAs) [10,11], special-purpose devices for optimization tasks, and algorithms based on quantum circuits or quantum gate arrays (QGAs) [7], which run on general-purpose quantum hardware or classical simulators. To leverage quantum solvers, optimization problems must be reformulated into quantum-compliant models. The Ising [12] and Quadratic Unconstrained Binary Optimization (QUBO) [13] models are widely used for this purpose. However, translating problems into these formulations often demands specialized expertise. To simplify this process, recent years have seen the development of tools and frameworks to assist users.

This review provides a comprehensive overview of quantum approaches to optimization, emphasizing their advantages, challenges, and limitations. It demonstrates how these methods apply to real-world scenarios, detailing the required steps and offering practical examples. Furthermore, it examines tools, frameworks, and solvers that support users in adopting quantum techniques for optimization tasks.

The article is organized as follows: Section 2 reviews optimization problems and conventional solvers. Section 3 discusses quantum-compliant formulations and tools for problem translation. Sections 4 and 5 focus on quantum annealers and circuit-based solvers, respectively. Finally, Section 6 concludes the article.

2. Solving Optimization Problems

The goal of optimization problems is to determine the configuration of parameters, inputs, or variables that minimizes or maximizes an objective function. For example, optimization minimizes delivery times in logistics and reduces costs in energy systems. These problems arise in fields like engineering, finance, and healthcare, and solvers explore solutions in an intelligent and systematic manner.

Optimization problems are classified based on variable types:

- Combinatorial Optimization (CO): this involves discrete variables.
- Continuous Optimization: this involves continuous variables.

When constraints limit feasible solutions, the problem is termed constrained optimization. Constraints can be integrated into the objective function via penalty functions or handled by generating only feasible solutions during exploration. The former is widely adopted in quantum optimization.

Objective functions are called cost functions when minimized and fitness functions when maximized. Many real-world problems involve multi-objective optimization (MO), requiring trade-offs among objectives. Solutions can be identified using Pareto optimal sets, enabling preference evaluation during (interactive methods) or after (a posteriori methods) optimization. Alternatively, objectives can be aggregated into a single scalar function (a priori methods).

Exploration strategies depend on problem characteristics. Brute-force approaches guarantee optimal solutions but become impractical as the problem size grows due to exponential complexity.

Deterministic methods suit convex problems with single minima but face challenges in multimodal or non-convex scenarios. In contrast, heuristic approaches are preferred for large-scale problems, balancing the exploration of feasible spaces and the exploitation of prior solutions.

Population-based algorithms explore solutions in parallel, inspired by natural and social systems. Examples include the following:

- Cooperative algorithms: these mimic swarm behavior (e.g., particle swarm optimization [14]).
- Genetic algorithms (GAs): these model biological evolution [15].
- Artificial immune algorithms: these simulate immune responses [16].

Algorithms inspired by physical systems are also prominent. For instance, simulated annealing (SA) [17] mimics annealing in materials science. SA probabilistically accepts solutions that may worsen objective values based on a temperature parameter, enabling exploration early and exploitation as temperature decreases. Modern hardware accelerators, such as probabilistic computers [18] and Coherent Ising Machines (CIMs) [19], exploit physical processes to solve optimization problems.

Quantum solvers leverage quantum phenomena such as superposition, entanglement, and tunneling. Quantum annealing (QA) parallels simulated annealing (see Section 4), while quantum gate arrays (QGAs) use methods like Grover Adaptive Search and Quantum Approximate Optimization Algorithms (QAOAs) (see Section 5).

However, quantum devices remain in the Noisy Intermediate-Scale Quantum (NISQ) era, with limited qubits, connectivity, and susceptibility to errors, posing challenges for real-world applications.

3. Quantum-Compliant Problem Formulations

3.1. Ising

The Ising formulation [20] is a physical–mathematical model used in statistical mechanics to describe phase transitions in ferromagnetic systems. It represents magnetic dipoles as binary variables, $+1$ (spin-up) and -1 (spin-down), arranged in a lattice structure with interactions. Initially proposed for a one-dimensional chain, it was extended to two dimensions by Lars Onsager in 1944.

Its energy function, known as Hamiltonian, includes the following:

- An external field term h, whose sign determines the preferred spin orientation and the magnitude reflects the strength of the spin's contribution to the total energy.
- An interaction term between neighboring spins J, due to the influence of the magnetic field generated by each magnetic dipole on its neighbors, which can be either aligned or counter-aligned, with the strength depending on its magnitude.

The Ising Hamiltonian is expressed as follows:

$$H(\mathbf{s}) = \sum_{i=1}^{N} h_i s_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1, j \neq i}^{N} J_{ij} s_i s_j \,, \tag{1}$$

where N is the total number of spins, s_i is ith spin, h_i is the external field coefficient of the ith spin, and J_{ij} is the interaction coefficient between the ith and jth spins.

Minimizing this Hamiltonian is generally NP-hard, except for special cases, such as planar 2D graphs without external fields [21].

The Ising model, extensively studied with computational methods like Monte Carlo simulations, is widely used for solving CO problems, where the ground state represents the optimal solution. Any CO problem can be recast into the Ising framework, as any binary-variable objective function can be reduced to a second-order polynomial, matching the Ising Hamiltonian, at the expense of introducing auxiliary variables.

3.2. QUBO

Quadratic Unconstrained Binary Optimization (QUBO) [13] models CO problems using a quadratic pseudo-Boolean objective function, similar to the Ising model but

with unipolar variables $(0, 1)$ instead of bipolar $(-1, 1)$. A QUBO problem is typically expressed as follows:

$$\text{minimize } f(\mathbf{x}) = q_0 + \sum_{i=1}^{N} q_i x_i + \sum_{i=1}^{N} \sum_{j=i+1}^{N} q_{ij} x_i x_j \,, \tag{2}$$

where N is the number of variables, equivalent to the number of spins in the Ising case; x_i is the ith binary variable; q_i is the bias associated with the single ith variable; q_{ij} is the coupling coefficient between the ith and jth variables; and q_0 is an offset, which can be neglected during optimization, as it does not influence the optimal configuration of variables. Alternatively, it can be written in matrix form:

$$\text{minimize } f(\mathbf{x}) = \mathbf{x}^t \mathbf{Q} \mathbf{x} \,, \tag{3}$$

where $\mathbf{Q}$ is a square matrix with linear coefficients on the diagonal and coupling coefficients off-diagonal, either in upper triangular or symmetric form.

In both QUBO and Ising formulations, the constraints can be taken into account only through the quadratic penalty functions:

$$\text{minimize } y = f(\mathbf{x}) + \lambda g(\mathbf{x}) \,, \tag{4}$$

where λ is a positive penalty parameter and $g(\mathbf{x})$ is the penalty function. Multi-objective optimization can also be handled in QUBO using an aggregation approach (Objective Weighting), which combines multiple objectives into a single one [22].

The QUBO formulation can be translated in the Ising model by applying a simple change of variable:

$$s_i = 2x_i - 1 \,. \tag{5}$$

From this equation, transformations to switch from QUBO to Ising can be straightforwardly performed.

3.3. HUBO/PUBO

The QUBO formulation can be extended to handle higher-degree polynomials while preserving its features, resulting in Higher-Order Unconstrained Binary Optimization (HUBO) or Polynomial Unconstrained Binary Optimization (PUBO). These models minimize polynomial pseudo-Boolean functions:

$$\text{minimize, } f(\mathbf{x}) = \sum_{V \subseteq 1,2,\ldots,N} q_V \prod_{i \in V} x_i, \tag{6}$$

where V represents subsets of indices from $1, 2, \ldots, N$. The Ising model's higher-order extension is known as the Higher-Order Ising model.

Formulating higher-order polynomials in QUBO/Ising models requires polynomial reduction, introducing auxiliary variables. This expands the search space and reshapes the energy landscape, adding small barriers that make exploration more challenging [23].

Therefore, higher-order models for problems that are naturally represented by polynomial cost functions scale more efficiently with the problem size and offer a smoother search space, simplifying exploration compared to reduced formulations.

3.4. PCBO

The Polynomial Constrained Binary Optimization (PCBO) extends the PUBO model by optimizing a polynomial function with binary variables while explicitly including polynomial constraints, unlike QUBO or PUBO, which handle constraints via penalties.

3.5. Writing an Optimization Problem as QUBO

Most quantum solvers support QUBO/Ising formulations, while only a few handle PUBO or PCBO. Due to the intuitive use of unipolar variables and straightforward conversion, PCBO, PUBO, and QUBO are preferred over Ising. This work focuses on reducing problems to the more widely supported QUBO formulation.

The first step involves identifying the problem variables, i.e., the elements to optimize. Two situations can occur:

- The optimization subjects can be easily described through binary variables since the target is to optimize decisional elements;
- The optimization subjects can assume several continuous or discrete values.

In the first case, a suitable data structure is needed to reference variables in the cost function. While some cases are straightforward, more complex scenarios, such as associating one set of elements with another, may require a binary variable grid. Columns and rows represent different objects (e.g., data points and classes), with a value of 1 indicating an association. Constraints are typically added to prevent multiple associations.

The second case is more complex, as continuous variables must be converted into binary form. This typically begins with discretization, which reduces solution accuracy. Therefore, precision must balance variable count and representation accuracy. Several encoding mechanisms can be considered for describing the discretized variables.

In both cases, converting the function to a polynomial form may be required. This can be challenging and may reduce accuracy, impacting reliability.

Afterwards, constraints have to be included in the formulation, either directly (PCBO) or via penalty functions. Creating a penalty function may frequently require the addition of auxiliary variables. Moreover, an appropriate penalty weight has to be chosen to correctly penalize the configurations violating the constraints. This task is crucial since a low weight may neglect constraints, while a high weight can flatten the objective function, hindering solution evaluation. Some techniques can be applied to effectively estimate its value.

At this point, if the solver cannot handle the PUBO formulation, a polynomial reduction step, introducing further auxiliary variables, has to be applied to obtain the QUBO formulation. Finally, the QUBO formulation can be moved to Ising if needed by using the relation of Equation (5).

The stages below summarize the process of transforming an optimization problem into a form compatible with a quantum solver:

- Define the variables, objective functions, and eventual constraints.
- Choose the solver.
- Convert the problem into a solver-compliant format (e.g., QUBO) by executing the following steps:
    1. Describe variables as binary.
    2. Compose the cost function.
    3. Write constraints as penalties.
    4. Estimate penalty weights to integrate constraints into the cost function.
    5. Perform polynomial reduction if necessary.

3.6. Frameworks and Tools

All the steps discussed above require substantial expertise in QUBO formulation, which many users may lack. This knowledge gap limits access to the potential benefits of quantum computation. Moreover, tasks such as variable encoding, solution decoding, and function rewriting can be time-consuming and lead to errors when performed manually. Managing penalty functions is equally important, especially the operation of translating

a constraint into a suitable penalty function. This process can be even more challenging when auxiliary variables are needed for inequality constraints. Additionally, selecting an appropriate penalty weight is crucial for obtaining satisfactory results. For these reasons, libraries and frameworks that automate the translation of optimization problems into solver-compatible formats are essential in assisting end-users.

The main libraries include pyqubo [24,25], qubovert [26], dimod [27], Fixstarts Amplify [28], Qiskit-optimization [29], and openQAOA Entropica [30]. Additionally, several key frameworks have emerged in the last two years to address user demands for tools that automate the entire process, including AutoQUBO [31–33], QUBO.jl [34], and mqt-qao [35–37]. Refs. [34,35] offer a detailed overview of these tools, along with a comparison of their features. In addition to these tools, a framework for formulating graph-based optimization problems into QUBO format has been proposed in [38], as part of the Munich Quantum Toolkit [37]. Furthermore, in [39], a similar tool is introduced for formulating discrete optimization problems such as QUBO.

Finally, D-Wave has released a QUBO preprocessing toolchain [40] optimized for their Quantum Annealer platform to reduce problem complexity. This toolchain includes methods for reducing the number of QUBO variables and estimating the lower bound of the objective function, based on techniques from [41]. Alternatively, the tool Qoolchain, proposed in [42], offers further methods to reduce QUBO size, decompose it, and estimate function bounds especially focusing on GAS.

4. Quantum Annealer

Quantum annealers [10,11] are special-purpose quantum computers designed to solve QUBO problems. They are defined as analog quantum computers because they exploit quantum systems' continuous, analog properties to explore the problem solution space. Unlike digital quantum computers, which rely on discrete operations and gates, quantum annealers use smooth, gradual changes in physical parameters (such as magnetic fields) to steer the system toward a low-energy solution. They are the physical implementation of the quantum annealing optimization process (Section 4.2), which is based on Adiabatic Quantum Computing (Section 4.1).

4.1. Adiabatic Quantum Computing

Adiabatic Quantum Computing [43] (AQC) exploits the adiabatic theorem—asserting that a quantum system remains in the ground state if its Hamiltonian changes sufficiently slowly and if there is a significant energy difference between the ground energy level and the next highest energy state during the process—to perform optimization. The form of the considered time-dependent Hamiltonian is as follows:

$$H(t) = A(t)H_1 + B(t)H_0\,, \tag{7}$$

where H_1 is the final Hamiltonian encoding the optimization problem to solve; H_0 is the initial Hamiltonian, whose ground state should be well known and easy to prepare; and $A(t)$ and $B(t)$ are the field strengths, respectively. They have a complementary behavior with time so that $H(0) = H_0$ and $H(\infty) = H_1$. In this way, the final state of the system, i.e., the ground state of the final Hamiltonian, should be the minimum of the encoded optimization problem function.

The success of the AQC depends on the evolution speed, i.e., the rate of change of $H(t)$, which should be slower than the inverse square of the minimum energy gap [44]:

$$T \gg \frac{\hbar \cdot \max_s \left| \langle \psi_1(s) \left| \frac{dH(s)}{ds} \right| \psi_0(s) \rangle \right|}{\Delta E(t)^2}, \tag{8}$$

where

- T is the required evolution time.
- $\Delta E(t)$ is the energy gap between the two lowest energy levels at time t.
- $\psi_0(s)$ and $\psi_1(s)$ are the instantaneous ground and first excited states of the time-dependent Hamiltonian $H(s)$, respectively.
- $H(s)$ is the Hamiltonian of the system, parametrized by $s = t/T$ with $s \in [0, 1]$ as the normalized time.
- $\frac{dH(s)}{ds}$ is the derivative of the Hamiltonian with respect to the normalized time, representing the rate of change in the Hamiltonian during evolution.
- $\langle \psi_1(s) | \frac{dH(s)}{ds} | \psi_0(s) \rangle$ measures the coupling between the ground and excited states induced by the evolution of the Hamiltonian.

4.2. Quantum Annealing

Quantum annealing is the practical implementation of the theoretical and ideal AQC. It is based on AQC principles, but to make the physical implementation feasible, the possible final Hamiltonian is limited to the Ising form, which is reformulated as a quantum Hamiltonian formulation:

$$H_1 = \left(-\sum_{ij} J_{ij} \sigma_i^z \sigma_j^z - h \sum_i \sigma_i^z \right), \tag{9}$$

where σ_i^z is the Pauli matrix associated with the z component, allowing the optimization of only Ising/QUBO problems. On the other hand, the initial Hamiltonian is usually the transverse field $H_0 = -\sum_i \sigma_i^x$, where σ_i^x is the Pauli matrix associated with the x component. Moreover, in quantum annealing, the adiabaticity is no longer guaranteed since the estimation of the time T_A is harder than the optimization problem to solve [45], and it could be impractical to run the system evolution for such a long time. Therefore, the quantum annealing runs for a certain time and, even if it does not satisfy the adiabaticity, it is expected to be able to find a good approximation of the optimal solution. In fact, it is not strictly needed to remain in the ground state of $H(t)$. Ultimately, upon measuring the state, it is sufficient for the amplitude of an optimal or adequately good solution in the final state to be large enough. This ensures that the probability of obtaining a useful result remains high.

To better understand the quantum annealing evolution, a qualitative idea can be given considering a hydraulic model in Figure 7b of [46]:

- In the beginning, the transverse field makes the bottom of the tank flat; thus, the water is uniformly distributed. This corresponds to having all the spins aligned along the x-axis (Figure 7a of [46]), i.e., state superposition.
- By increasing the longitudinal field and decreasing the strength of the transverse one, the bottom of the tank is gradually deformed and the water begins to flow towards the lowest points. This corresponds to having the spins that gradually modify their orientation to follow the instantaneous ground state.
- In the end, only the longitudinal field is present, which has deformed the bottom of the tank as a function of the problem, and the water is concentrated in the lowest point, i.e., that with minimum energy. Therefore, the final spin configuration corresponds to the optimal solution of the optimization problem expressed as the Ising Hamiltonian.

Quantum annealing can be seen as an improvement of the simulated annealing [17], in which thermal fluctuations are replaced with quantum fluctuations [11]. The trans-

verse Ising model form, proposed in [11], is the basis of the Hamiltonian used in quantum annealing:

$$H(t) = -\sum_{ij} J_{ij}\sigma_i^z\sigma_j^z - h\sum_i \sigma_i^z - \Gamma(t)\sum_i \sigma_i^x\,, \tag{10}$$

where $\Gamma(t)$ is the transverse field strength (equivalent of B(t) of the previous section).

The transverse field $\Gamma(t)$ causes quantum tunneling between system eigenstates. The advantage of quantum annealing exploration is that the probability of overcoming an energy barrier with height Δ is proportional to $e^{-\frac{\sqrt{\Delta}w}{\Gamma}}$ (where w is the width of the energy barrier and Γ the strength of the transverse field). On the other hand, in the simulated annealing case, it is equal to $e^{-\frac{\Delta}{k_B T}}$ (where T is the temperature parameter and k_B is the Boltzmann constant). Consequently, quantum exploration is significantly more effective than classical one in case of problems with the energy landscape with a high amount of perturbation with many high and thin barriers ($w \ll \Delta$) [47], as shown in Figure 7c of [46].

4.3. Hardware Implementations

A leading company in the field of QAs is D-Wave Systems [48], which, since 2011, has been developing QAs with a growing number of qubits (more than 5000), enabling the solution of increasingly complex optimization problems. These devices are implemented in superconductive technology, which requires very strict operating conditions, e.g., 15 mK, and presents limited connectivity. Therefore, to map a generic fully connected problem, minor embedding [49] is needed. This technique consists of splitting a logical bit into multiple physical bits of the target architecture, each one belonging to a different subgraph. The physical bits associated with the same logical bit are strongly ferromagnetically coupled, so that discordant states are penalized. Therefore, the physical bits are expected to collapse to the same value at the end of the annealing process. It is necessary to note that using a minor embedding configuration rapidly increases the number of involved physical bits, which adds a time overhead in the overall execution.

D-Wave has proposed three minor embedding topologies: Chimera at the beginning, and Pegasus and Zephyr, recently. The Chimera structure is a connected network of bits with groups of densely connected nodes sparsely connected to other groups of densely connected nodes. In this topology, variables have degree 6, which means that each bit is connected through couplers to six others. The instructions for obtaining a Chimera mapping for some important problems are reported in [50]. It is the topology of D-Wave 2000Q. Pegasus [51] is able to provide a more efficient embedding compared to Chimera and requires a shorter embedding time. This topology has a degree of 15, so a lower number of bits is required to obtain an equivalent configuration of the Chimera one. The additional connectivity of the Pegasus graph can be also used to construct a simple error correction scheme for quantum annealing. It is the topology of Advantage devices. Zephyr represents a further advantage, with a qubit degree of 20.

It is possible to use the real quantum annealer by creating a free account on Dwave Leap, a cloud service providing one minute of free computing access per month.

4.4. Recent Evolutions

Classical and quantum solver performance is strongly dependent on the energy profile. For example, SA and other local search-based methods are effective in exploring wide and smooth barriers in the energy landscape, but they cannot efficiently overcome high and narrow barriers. In contrast, QA is particularly well suited for problems where the energy landscape is characterized by high and narrow peaks, leveraging quantum tunneling to bypass these barriers. However, QA is expected to be less effective in wide and flat

regions [52]. Real-world problems, however, often exhibit heterogeneous energy profiles, as shown in Figure 10 of [46]. A solver effectiveness depends on the size and features of the region in the energy profile that matches its exploration mechanism. For instance, in a heterogeneous landscape, significant improvements can be achieved by combining a local approach for exploring wide and smooth regions with QA to handle rough, peak-filled regions.

For these reasons, interest in the development of hybrid solvers, which are designed to effectively alternate between local and global search strategies, arises. Several proposals for hybrid approaches have been introduced in recent works [53]. Nowadays, D-Wave has also proposed hybrid solvers [54], which divide the problem, assigning different parts to classical solvers and quantum annealers, and then reconstructing a global solution from the local ones. These solvers support constrained problems in addition to QUBO.

In addition, the recently proposed reverse annealing [9] begins from a known candidate solution (classical solution), gradually reintroducing quantum fluctuations (superposition) to explore nearby states in the solution space, allowing for refinement and local search within a specific region of the energy landscape.

5. Quantum Circuit Model-Based Solvers

This section discusses the most popular quantum circuit-based algorithms for solving optimization problems. Due to the limitations of quantum hardware, these algorithms are all hybrid, i.e., involving both classical and quantum computers. QAOA and VQE (discussed in Sections 5.1 and 5.2) exploit variational circuits whose parameter optimization helps address noise problems of NISQ hardware. GAS (explored in Section 5.3) leverages the Quantum Fourier Transform (QFT), which is designed to be effective in future Fault-Tolerant Quantum Computers (FTQCs).

5.1. QAOA

The Quantum Approximate Optimization Algorithm (QAOA) [55] is a hybrid quantum–classical algorithm introduced in [56], designed to solve optimization problems approximating the quantum AQC on gate-based quantum computers (discussed in Section 4.1). Starting from an Ising Hamiltonian, it has to be reformulated as a quantum Hamiltonian formulation by replacing the binary variables with Pauli Z matrices, since the eigenvalues of this matrix are $+1$ and -1.

$$H_1 = \sum_i h_i \sigma_i^z + \sum_{i,j} J_{ij} \sigma_i^z \otimes \sigma_j^z. \tag{11}$$

Quantum circuit model computers nature operate by applying discrete quantum gates to modify the qubit's state, while AQC depends on a continuous evolution of the Hamiltonian. This evolution can be discretized through the Trotterization technique, which approximates it in discrete steps. The AQC time-dependent Hamiltonian (Equation (7)) can be discretized for the execution of quantum gates by expressing it as a product of unitary operators performing a small evolution step. Since $H(t)$ is diagonal in the quantum system computational space, i.e., a $2^n \times 2^n$ Hilbert space, it can be made unitary by exponentiating $H(t)$, which corresponds to taking the exponential of its elements. The discretized operator can be obtained by considering the evolution only in a small time instant Δt:

$$e^{i\Delta t(A(t_c)H_0 + B(t_c)H_1)}, \tag{12}$$

where t_C is a fixed timestep within the range $[0, T]$, with T being the total duration of the process. In a small time interval Δt, the Hamiltonian can be assumed to be constant. By discretizing the process into p steps, the Hamiltonian becomes

$$\prod_{m=0}^{p} e^{i\Delta t(A(t_m)H_0 + B(t_m)H_1)}, \tag{13}$$

where $t_m = m\Delta t/T$ and $p = T/\Delta t$. According to the Lie–Trotter product formula, if Δt is very small, $e^{A+B} \approx e^A e^B$ with A, B being two square complex matrices. Furthermore, by selecting $A(t_m)$ and $B(t_m)$ as small increments of time, the quantum circuit simulates the adiabatic evolution of the Hamiltonian. However, instead of $A(t_m)$ and $B(t_m)$, two angles, γ and β, can be chosen at each iteration of the parametric circuit according to a classical optimization algorithm to obtain the best values for reaching the ground state of the Ising Hamiltonian. The quantum part of QAOA is performed by a variational quantum circuit—i.e., a circuit including parametric gates, whose value has to be optimized during algorithm execution—preparing the following state:

$$|x\rangle = \prod_{m=p}^{0} e^{i\beta_m H_0} e^{i\gamma_m H_1} |\Psi_0\rangle = \prod_{m=p}^{0} U_M(\beta_m) U_c(\gamma_m) |\Psi_0\rangle. \tag{14}$$

To keep the analogy with the quantum adiabatic algorithm and ensure the system evolves while keeping the quantum state as the ground state of the evolving Hamiltonian, Ψ_0 must be the ground state of H_0. A suitable choice for the mixed Hamiltonian is the tensor product of n Pauli X operators (σ_j^x) with a negative sign, with n being the number of qubits. Given a quantum state $|\Psi\rangle$, the expectation value of the operator σ_j^x in this state is as follows:

$$\langle \Psi | X_j | \Psi \rangle = 2c_0 c_1 \leq 1, \quad \text{with} |\Psi\rangle = c_0|0\rangle + c_1|1\rangle, \tag{15}$$

which is equal to 1 only when $c_0 = c_1 = 1/\sqrt{2}$; thus, the expectation value is maximum when $|\Psi\rangle = |+\rangle$. Therefore, H_0 is specifically chosen because its ground state is $|\Psi_0\rangle = \bigotimes_{j=0}^{n-1} |+\rangle$, which is straightforward to prepare, as it can be obtained by applying n Hadamard gates and is completely unentangled. To represent the mixed Hamiltonian with unitary quantum gates, it is transformed into the following operator, since the σ_j^x matrices commute:

$$e^{i\beta_m H_0} = e^{-i\beta_m \bigotimes_{j=0}^{n-1} \sigma_j^x} = \bigotimes_{j=0}^{n-1} e^{-i\beta_m \sigma_j^x} = R_X(2\beta_m). \tag{16}$$

Therefore, the operator $U_M(\beta_m)$ is implemented just by applying the R_X gate to every qubit in the circuit (Figure 1).

Concerning the Ising Hamiltonian operator, H_1 comprises only diagonal matrices; thus, they commute as well. Consequently, the linear and quadratic terms can be separated into two distinct operators:

$$U_c(\gamma_m) = e^{i\gamma_m H_1} = e^{i\gamma_m \sum_j h_j \sigma_j^z + \sum_{j,k} J_{jk} \sigma_j^z \otimes \sigma_k^z} = \prod_j e^{i\gamma_m h_j \sigma_j^z} \prod_{j,k} e^{i\gamma_m J_{jk} \sigma_j^z \otimes \sigma_k^z}. \tag{17}$$

The operator associated with the linear Ising term has the same form as the mixed Hamiltonian one—a product of operators applied on different qubits is equivalent to the tensor product of these operators. Therefore, it can be implemented by applying the $R_Z(-2\gamma_m h_j)$ quantum gate on the j^{th} qubit. The operator associated with the quadratic term

includes a tensor product between two diagonal matrices; hence, it should be performed by a two-qubit gate. Two different cases can be discerned:

- If qubits j and k have the same value, the operator becomes $e^{i\gamma_m J_{jk}}$,
- If qubits j and k have different value, the operator becomes $e^{-i\gamma_m J_{jk}}$.

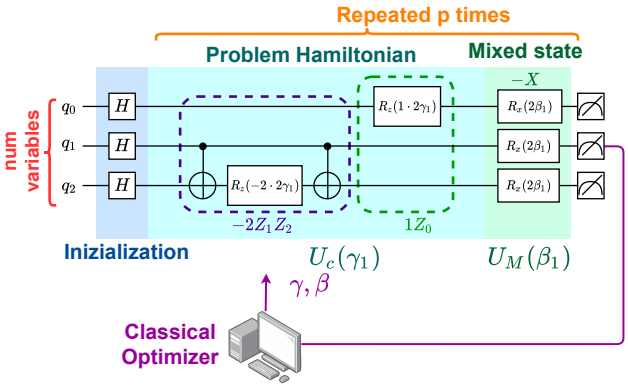


Figure 1. Example of QAOA quantum circuit to optimize the Ising Hamiltonian $-2s_1s_2 + 1s_0$.

Therefore, this transformation is equivalent to applying the $R_Z(-2\gamma_m J_{jk})$ gate on the k^{th} between two controlled not gates (Figure 1). In this way, if $|j\rangle = |1\rangle$ and $|k\rangle = |0\rangle$, $|k\rangle$ is inverted and its phase is flipped. If $|j\rangle = |0\rangle$ and $|k\rangle = |1\rangle$, the CNOT gate does not affect $|k\rangle$ and its phase is flipped. Hence, the operator flips the phase only when the two qubits have different values. The sequence of these gates produces a quantum circuit implementing the operators $U_M(\beta_m)U_c(\gamma_m)$, which is also called ansatz since it is a parametric circuit repeated many times (p) in the algorithm to approximate AQC as discrete evolution.

Since QAOA is a variational circuit (Figure 1), a classical optimization part is needed to properly select the circuit parameters and obtain the ground state of the problem Hamiltonian. Therefore, the algorithm steps are reported below [57]:

1. The effective quantum circuit is defined by selecting a proper value for the number of discretization steps p.
2. Initial values for the parameters $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$ are chosen.
3. The quantum circuit is executed multiple times and the expectation value with respect to the problem Hamiltonian is calculated.
4. The circuit parameters $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$ are updated by the classical optimizer exploration approach exploiting the obtained expectation value.
5. Repeat from step 2 until the algorithm's stop condition—e.g., a selected number of iterations or the convergence of the classical optimizer—is met.

The classical optimizer can be any algorithm that does not require an explicit expression for the objective function, but only a cost evaluation based on the selected parameters. The results produced by QAOA may significantly vary according to the optimizer used to tune the angle parameters. However, the execution time of these algorithms is crucial; since a quantum algorithm is exploited as a heuristic, increasing the number of iterations may be more favorable than relying on more accurate classical optimizers. An overview and comparison of the best classical optimizers for QAOA is provided in [58].

Regarding the number of time steps p, a larger value should better approximate the adiabatic evolution. However, this also increases the quantum circuit depth, which can negatively impact the accuracy and reliability of NISQ hardware [59].

QAOA can directly solve PUBO problems, as a monomial of any degree can be encoded in a quantum state. The polynomial Hamiltonian can be reformulated as a quantum Hamiltonian, as shown in Equation (11). Calling L the set of qubits involved, a

generic polynomial term is composed of the tensor product of $|L|$ σ^z matrices. Therefore, two different cases can still be distinguished:

- If the number of qubits in the state $|1\rangle$ is even, the operator is $e^{i\gamma_m J_L}$;
- If the number of qubits in the state $|1\rangle$ is odd, the operator is $e^{-i\gamma_m J_L}$.

This transformation is then a parity check on the qubits in L, followed by a $R_Z(-2\gamma_m J_L)$ gate, ensuring that the phase shift is negative only when the parity is even. Parity can be determined by applying a CNOT gate for each qubit l in L except for the last one, using l as the control qubit and the last as the target (an example is reported in Figure 2). For formulations that naturally belong to the PUBO category, results in [60] evidence that using a PUBO function not only avoids the need for auxiliary variables and improves the cost function profile, but also improves the accuracy of results and reduces the optimization time of angle parameters in QAOA, when using the same depth and hyperparameters for the classical optimization algorithm. However, this comes at the expense of increasing the depth of the quantum circuit, which grows linearly compared to the QUBO case.

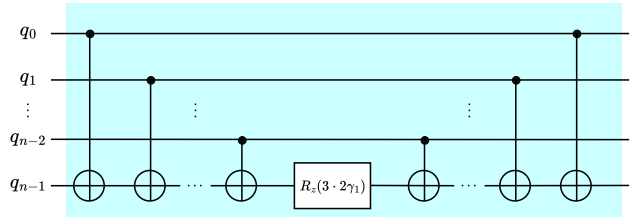


Figure 2. Example of the quantum operator realizing the $3s_0s_1 \dots s_{n-1}$ higher-order Ising term.

Recently, improvements to the algorithm have been proposed in the literature. Among them, the so-called Warm-Start QAOA [61] is worth mentioning, which exploits as the initialization state the solution of the continuous relaxation of the QUBO problem—which can be easily obtained since it is a convex problem—instead of the uniform superposition. Moreover, the mixed Hamiltonian is modified to be coherent with the initialization state.

5.2. VQE

The Variational Quantum Eigensolver (VQE), introduced in [55,62], was originally conceived for finding the lowest eigenvalue of any type of Hamiltonian, in particular focusing on quantum chemistry applications, but it is also applicable for optimization problems in general. It exploits a parametric guess quantum circuit (ansatz), whose parameters can be classically optimized, to make the state converge toward an upper bound of the ground state of an input Hamiltonian thanks to the variational principle (Figure 3). The variational principle states that the smallest expectation value of an observable A, which is hence described by a Hermitian operator (for which $A^\dagger = A$), is always found at an eigenvector of the observable [55]. The expectation value of an Hermitian operator A in a state $|\psi\rangle$ is given by the following:

$$\langle A \rangle_\psi = \sum_{j,k} |\langle \lambda_j^k | \psi \rangle|^2 \lambda_j = \langle \psi | A | \psi \rangle, \tag{18}$$

where λ_j is an eigenvalue of the operator A, and $|\lambda_j^k\rangle$ is the kth eigenvector associated with the eigenvalue λ_j, as multiple eigenvectors can correspond to the same eigenvalue. Supposing that λ_0 is the smallest among the eigenvectors of A, then

$$\langle A \rangle_\psi = \sum_{j,k} |\langle \lambda_j^k | \psi \rangle|^2 \lambda_j \geq \sum_{j,k} |\langle \lambda_j^k | \psi \rangle|^2 \lambda_0 = \lambda_0. \tag{19}$$

This expectation value, which is the smallest of the observable A, can be achieved in a state that is either one of the eigenvectors associated with λ_0 or a normalized linear combination of these eigenvectors. Indeed, from the definition of eigenvalue, the following is obtained:

$$\langle \lambda_0^k | A | \lambda_0^k \rangle = \lambda_0 \langle \lambda_0^k | \lambda_0^k \rangle = \lambda_0. \tag{20}$$

Therefore, minimizing the expectation value means finding the lowest eigenvalue. In the case of an Ising Hamiltonian—which is a diagonal operator and hence its eigenvectors are the computational basis states—this also coincides to identifying the lowest energy state since

$$\langle \psi | H | \psi \rangle = \sum_x |a_x|^2 f(x), \tag{21}$$

where $|a_x|^2$ represents the probability of observing the input configuration $|x\rangle$ when $|\psi\rangle$ is measured in the computational basis. However, a general Hamiltonian cannot be assumed to be diagonal. For any operator with eigenvectors different from the computational basis, computing the minimum of its expectation value on a quantum computer requires measuring in the basis defined by its eigenvectors. In a quantum circuit, it is always possible to apply a change of basis so that measurement in the computational basis becomes equivalent to measurement in the eigenvector basis.

After the introduction of the principle and objectives of VQE, we can now examine each step of the algorithm, which are listed below:

1. An ansatz and an initial set of values for the parameters are chosen.
2. The quantum state is prepared by the parametric circuit.
3. The quantum state is measured, so the expectation value, namely the energy of the Hamiltonian, is estimated.
4. A classical optimizer is used to determine the parameter values for the next iteration to decrease the estimated expectation value.
5. Repeat from step 2 until the algorithm's stop condition—e.g., a selected number of iterations or the convergence of the classical optimizer—is met.

From the algorithm steps, a remarkable similarity with QAOA can be noticed. In fact, QAOA can be regarded as a special case of VQE, employing a specific ansatz based on the Ising Hamiltonian.

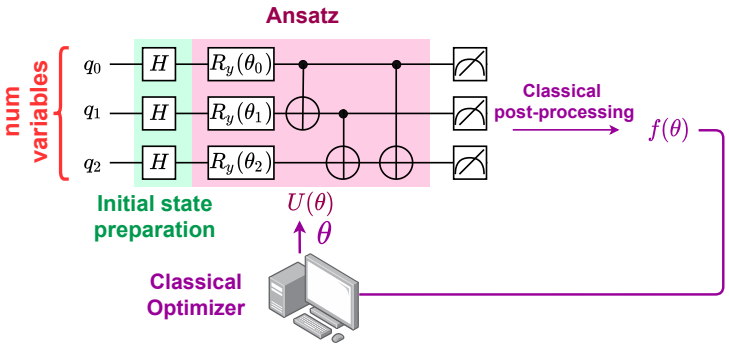


Figure 3. VQE example.

The ansatz is a parametric quantum circuit with a structure chosen to produce a trial wavefunction to be measured, which, after optimizing the ansatz's parameters, should approximate the Hamiltonian minimum expectation value. Many ansatz structures are suitable for this goal, and the quality of the approximation also depends on this choice. Therefore, the structure employed can be considered as a hyperparameter of the problem. The optimal choice typically depends on the problem characteristics. However, some features of an ansatz can be evaluated to identify the best fit for the problem at hand [63]:

- Expressibility: It is the span of possible quantum states that the ansatz can reach in the Hilbert space, or how well it can explore the space of the possible states. It can be quantitatively assessed by the distance between the distributions of the unitary operators generated by the ansatz and the maximally expressive uniform distribution of operators (Haar measure) [64].
- Trainability: It is the ability to find the best parameter values to optimize the ansatz according to the Hamiltonian expectation value within a feasible time frame. An ansatz is not considered trainable when the cost function gradients vanish exponentially as a function of the optimization parameters, which is referred to as the barren plateau problem. The trainability depends on the number of parameters, their mutual dependence, and the optimization landscape. This concept is related to expressibility, as a good ansatz should accurately approximate the quantum state of the best expectation value but an overly accurate one could make the training of its parameters intractable.
- Circuit depth: The depth of a quantum circuit can influence the final estimation due to noise and decoherence problems; hence, it should be carefully managed in NISQ hardware applications.

A further classification of ansatz can be based on whether their structure changes during optimization:

- Fixed structures remain unchanged during optimization, except for their parameters.
- Adaptive structures evolve during the optimization process by incorporating additional operators. Some gradually grow at each optimization step, others embed part of the ansatz into the optimization alongside the Hamiltonian, while some modify the existing structure to learn an optimal configuration.

For both the ansatz categories, ref. [63] provides various examples highlighting their structures and discussing pros and cons.

The classical optimization of the quantum circuit parameters must be sufficiently fast to maintain the effectiveness of quantum optimization, while still providing enough accuracy to converge to the Hamiltonian ground state. The training of the ansatz parameters is NP-hard [65]. However, several heuristic methods, especially those based on gradient calculation, are employed for this purpose. An overview of classical optimization approaches suitable for VQE is presented in [63].

The adaptive nature of VQE, whose parameters are optimized within the different iterations of the algorithm, allows balancing the systematic errors caused by the imperfection of control systems [66]. This makes VQE particularly well suited for NISQ hardware, as it can mitigate the noise problems commonly affecting these systems, whereas algorithms relying on error correction may provide more accurate results in future FTQCs.

5.3. GAS

The Grover Adaptive Search is a successive approximation hybrid quantum–classical algorithm designed to find the minimum of a cost function $f(x)$, which represents a combinatorial optimization problem [55,67,68]. It is composed of the following steps:

1. Initialize the iteration index i and the variable y_i to zero.
2. Adjust the cost function $f(x)$ by vertically shifting it based on y_i (Figure 4).
3. Increment the iteration index i.
4. Randomly sample a value y_i from the range of $f(x)$ such that $y_i < 0$.
5. Consider y_i as the new optimal value, as a negative value of the adjusted cost function consistently indicates a lower value than the previous offset y_{i-1}.
6. Repeat steps 2–5 until no further negative values are sampled. This process identifies the variable combination where $f(x) = 0$, solving the optimization problem.

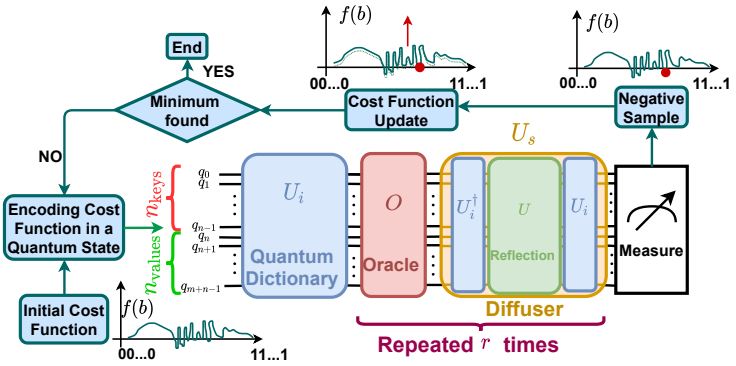


Figure 4. GAS intuitive scheme.

To fully leverage this approach, an efficient negative sampling method is crucial. The best option is a quantum routine that accelerates the process of finding solutions at each iteration. Grover Search (GS) [69] is a quantum algorithm offering a quadratic speedup over classical methods when searching for items in an unordered dataset. Initially, the objective function, shifted by y_i, must be encoded into a quantum state as a uniform superposition of the domain–image obtained through a quantum dictionary [68], the quantum counterpart of a classical dictionary, which allows the description of key–value pair data structures. The image of $f(\mathbf{x})$ is represented using the two's complement binary format for signed integers.

Encoding an n-variable problem requires n qubits for the keys and m—sufficiently large to avoid overflow, based on the function bounds—for the values. Unfortunately, estimating the exact bounds has the same computational complexity as solving the problem via brute force, growing as 2^n. Thus, preprocessing is necessary to select an appropriate value for m. Setting m too low compromises the mechanism's effectiveness, while setting it too high wastes limited resources. Several methods in the literature provide lower bounds for the minimum and, by addressing the negated problem, upper bounds for the maximum, typically requiring no more than a couple of additional qubits.

The following equation describes the quantum dictionary operator U_i:

$$U_i|0\rangle_n|0\rangle_m = \frac{1}{\sqrt{2^n}} \sum_{x=0}^{2^n-1} |x\rangle_n |f(x) - y_i\rangle_m. \tag{22}$$

At this point, the Grover phase oracle is applied to label the desired items of the dataset—in this case, the negative samples of the objective function—and the diffuser operator is employed to amplify the probability amplitude of the corresponding states. For implementing an effective GS, phase oracle and diffuser have to be repeated r times, where r is also called the number of GS rotations, and its optimal value depends on the dataset characteristics. It can be estimated through the following equation [70]:

$$r \approx \frac{\pi}{4}\sqrt{\frac{2^n}{M}} - \frac{1}{2}, \tag{23}$$

where M is the number of desired items, i.e., the negative samples, which is unknown a priori. This number of rotations ensures finding one of the M states with the maximum probability [7]; hence, either a too-high or too-low r may lead to an incorrect result. Its estimation in each GS execution is crucial, and some techniques were proposed in the literature for this purpose [68,71–73]. The quantum circuit employed for GS execution in the GAS algorithm is reported in Figure 4.

Once a negative value has been acquired, the function can be classically shifted by that amount and re-encoded in the quantum dictionary to search for a new negative. The

samples obtained are, with each iteration, closer to the function's minimum until the last negative is obtained, corresponding to the optimal solution. However, determining whether additional negative values need to be sampled is a crucial task. Theoretically, a characteristic of GS can be used for this goal: when no item meets the conditions of the GS, any possible configuration can be sampled according to a uniform probability distribution. If only non-negative function values are available, the GS outcomes are positive or null samples. Unfortunately, there are other situations where a non-negative value can be obtained, since it can also emerge when an incorrect number of Grover rotations is chosen. Consequently, proper techniques, principally based on the counts of consequently positive samples and the employment of thresholds, were defined in the state of the art for stopping the algorithm [71].

Similarly to QAOA, GAS can directly address the PUBO problem by employing multi-controlled gates in the quantum dictionary [68]. A method to reduce the required number of gates of this GAS implementation was proposed in [74]. Moreover, PCBO can be explored with GAS by incorporating constraints directly within the oracle rather than embedding them in the cost function through penalties [75].

6. Conclusions

Optimization influences several fields of research and industry, like finance, telecommunications, and mobility. As the complexity of the problem grows, the search for efficient and scalable solutions has led to significant interest in quantum approaches. This review article has outlined the main advancements in these fields, focusing on quantum annealers and quantum circuit-based algorithms solvers.

The potential of quantum solutions in handling NP-hard problems is promising, even if quantum devices are in their early stages. Even though quantum annealers and circuit-based models could provide meaningful benefits, they face hardware limitations and challenges related to coherence and scalability.

A key insight is the importance of effective problem formulation. Translating conventional optimization problems into quantum-compliant formats, such as QUBO or Ising models, is a critical step in exploiting quantum solvers. Tools and frameworks available in the state of the art for assisting end-users in this challenging task have been presented.

Emerging trends in this field suggest exciting opportunities for future research. Advancements in hybrid quantum–classical approaches are particularly meaningful as they leverage the strengths of both paradigms to address increasingly complex problems. Additionally, exploring enhanced heuristic methods and machine learning techniques for guiding optimization processes in the quantum context is a promising direction.

In conclusion, the quantum approaches show great potential and are expected to provide significant advantages once quantum technologies reach full maturity. Moreover, future improvements in exploration mechanisms can be achieved through synergy between quantum and classical approaches, offering a new frontier for solving increasingly complex problems across diverse domains. The integration of these two methods presents an exciting opportunity to advance optimization and address the increasing complexity of real-world challenges across various fields.

Author Contributions: Conceptualization, D.V. and G.O.; methodology, D.V. and G.O.; formal analysis, D.V. and G.O.; investigation, D.V. and G.O.; resources, D.V. and G.O.; data curation, D.V. and G.O.; writing—original draft preparation, D.V. and G.O.; writing—review and editing, D.V., G.O. and G.T.; visualization, D.V. and G.O.; supervision, G.T.; project administration, G.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data generated during and/or analyzed during the current study are available from the corresponding author on reasonable request

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Mattesi, M.; Asproni, L.; Mattia, C.; Tufano, S.; Ranieri, G.; Caputo, D.; Corbelletto, D. Diversifying Investments and Maximizing Sharpe Ratio: A Novel Quadratic Unconstrained Binary Optimization Formulation. *Quantum Rep.* **2024**, *6*, 244–262. [CrossRef]
2. Marchioli, V.; Boggio, M.; Volpe, D.; Massotti, L.; Novara, C. Scheduling of Satellite Constellation Operations in EO Missions using Quantum Optimization. In Proceedings of the International Conference on Optimization, Learning Algorithms and Applications 2024, Tenerife, Spain, 24–26 July 2024; Springer: Berlin/Heidelberg, Germany, 2024. [CrossRef]
3. Date, P.; Arthur, D.; Pusey-Nazzaro, L. QUBO formulations for training machine learning models. *Sci. Rep.* **2021**, *11*, 10029. [CrossRef] [PubMed]
4. Volpe, D.; Cirillo, G.A.; Fantini, R.; Boella, A.; Mondo, G.; Graziano, M.; Turvani, G. Quantum-compliant users scheduling optimization in joint transmission mobile access networks. *Quantum Inf. Process.* **2024**, *23*, 262. [CrossRef]
5. Garey, M.R.; Johnson, D.S. *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*, 1st ed.; Freeman, W.H., Ed.; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 1979.
6. Qian, C.; Chételat, D.; Morris, C. Exploring the power of graph neural networks in solving linear optimization problems. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Valencia, Spain, 2–4 May 2024; Proceedings of Machine Learning Research: Brookline, MA, USA, 2024; pp. 1432–1440. Available online: https://proceedings.mlr.press/v238/qian24a.html (accessed on 31 December 2024).
7. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information: 10th Anniversary Edition*; Cambridge University Press: Cambridge, UK, 2010. [CrossRef]
8. Blenninger, J.; Bucher, D.; Cortiana, G.; Ghosh, K.; Mohseni, N.; Nüßlein, J.; O'Meara, C.; Porawski, D.; Wimmer, B. Q-GRID: Quantum Optimization for the Future Energy Grid. In *KI-Künstliche Intelligenz*; Springer: Berlin/Heidelberg, Germany, 2024; pp. 1–11. [CrossRef]
9. Venturelli, D.; Kondratyev, A. Reverse quantum annealing approach to portfolio optimization problems. *Quantum Mach. Intell.* **2019**, *1*, 17–30. [CrossRef]
10. Johnson, M.W.; Amin, M.H.; Gildert, S.; Lanting, T.; Hamze, F.; Dickson, N.; Harris, R.; Berkley, A.J.; Johansson, J.; Bunyk, P.; et al. Quantum annealing with manufactured spins. *Nature* **2011**, *473*, 194–198. [CrossRef] [PubMed]
11. Kadowaki, T.; Nishimori, H. Quantum annealing in the transverse Ising model. *Phys. Rev.* **1998**, *58*, 5355. [CrossRef]
12. Lucas, A. Ising formulations of many NP problems. *Front. Phys.* **2014**, *2*, 74887. [CrossRef]
13. Glover, F.; Kochenberger, G.; Du, Y. A tutorial on formulating and using QUBO models. *arXiv* **2018**, arXiv:1811.11538. [CrossRef]
14. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the ICNN'95-International Conference on Neural Networks, Perth, WA, Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948. [CrossRef]
15. Frenzel, J.F. Genetic algorithms. *IEEE Potentials* **1993**, *12*, 21–24. [CrossRef]
16. Wang, X.; Gao, X.Z.; Ovaska, S.J. Artificial immune optimization methods and applications—A survey. In Proceedings of the 2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583), The Hague, The Netherlands, 10–13 October 2004; IEEE: Piscataway, NJ, USA, 2004; Volume 4, pp. 3415–3420. [CrossRef]
17. Kirkpatrick, S.; Gelatt, C.D.; Vecchi, M.P. Optimization by Simulated Annealing. *Science* **1983**, *220*, 671–680. [CrossRef] [PubMed]
18. Camsari, K.Y.; Faria, R.; Sutton, B.M.; Datta, S. Stochastic *p*-Bits for Invertible Logic. *Phys. Rev. X* **2017**, *7*, 031014. [CrossRef]
19. McMahon, P.L.; Marandi, A.; Haribara, Y.; Hamerly, R.; Langrock, C.; Tamate, S.; Inagaki, T.; Takesue, H.; Utsunomiya, S.; Aihara, K.; et al. A fully programmable 100-spin coherent Ising machine with all-to-all connections. *Science* **2016**, *354*, 614–617. [CrossRef]
20. Tanahashi, K.; Takayanagi, S.; Motohashi, T.; Tanaka, S. Application of Ising Machines and a Software Development for Ising Machines. *J. Phys. Soc. Jpn.* **2019**, *88*, 061010. [CrossRef]
21. Barahona, F. On the computational complexity of Ising spin glass models. *J. Phys. Math. Gen.* **1982**, *15*, 3241. [CrossRef]
22. Ayodele, M.; Allmendinger, R.; López-Ibáñez, M.; Parizy, M. Multi-Objective QUBO Solver: Bi-Objective Quadratic Assignment Problem. In Proceedings of the Genetic and Evolutionary Computation Conference, GECCO '22, New York, NY, USA, 9–13 July 2022; pp. 467–475. [CrossRef]
23. Dobrynin, D.; Renaudineau, A.; Hizzani, M.; Strukov, D.; Mohseni, M.; Strachan, J.P. Energy landscapes of combinatorial optimization in Ising machines. *Phys. Rev. E* **2024**, *110*, 045308 . [CrossRef]
24. PyQUBO Documentation. PyQUBO Documentation-Getting Started. Available online: https://pyqubo.readthedocs.io/en/latest/getting_started.html (accessed on 15 December 2023).
25. Zaman, M.; Tanahashi, K.; Tanaka, S. PyQUBO: Python library for mapping combinatorial optimization problems to QUBO form. *IEEE Trans. Comput.* **2021**, *71*, 838–850. [CrossRef]

26. Qubovert Documentation. Qubovert Documentation-Getting Started. Available online: https://qubovert.readthedocs.io/en/latest/ (accessed on 15 December 2023).
27. Dimod Documentation. Dimod Documentation-Getting Started. Available online: https://docs.ocean.dwavesys.com/en/stable/docs_dimod/ (accessed on 15 December 2023).
28. Fixstarts Documentation. Fixstarts Documentation-Getting Started. Available online: https://amplify.fixstars.com/en/docs/ (accessed on 15 December 2023).
29. Qiskit-Optimization Documentation. Qiskit-Optimization Documentation-Getting Started. Available online: https://qiskit.org/ecosystem/optimization/index.html (accessed on 15 December 2023).
30. openQAOA Entropica Labs Documentation. openQAOA Entropica Labs Documentation-Getting Started. Available online: https://el-openqaoa.readthedocs.io/en/latest/ (accessed on 15 December 2023).
31. Autoqubo GitHub Repository. Available online: https://github.com/FujitsuResearch/autoqubo (accessed on 15 December 2023).
32. Moraglio, A.; Georgescu, S.; Sadowski, P. AutoQubo: Data-driven automatic QUBO generation. In Proceedings of the Genetic and Evolutionary Computation Conference Companion, Boston, MA, USA, 9–13 July 2022; pp. 2232–2239. [CrossRef]
33. Pauckert, J.; Ayodele, M.; García, M.D.; Georgescu, S.; Parizy, M. AutoQUBO v2: Towards Efficient and Effective QUBO Formulations for Ising Machines. In Proceedings of the Companion Conference on Genetic and Evolutionary Computation, Lisbon, Portugal, 15–19 July 2023; pp. 227–230. [CrossRef]
34. Xavier, P.M.; Ripper, P.; Andrade, T.; Garcia, J.D.; Maculan, N.; Neira, D.E.B. Qubo.jl: A julia ecosystem for quadratic unconstrained binary optimization. *arXiv* **2023**, arXiv:2307.02577. [CrossRef]
35. Volpe, D.; Quetschlich, N.; Graziano, M.; Turvani, G.; Wille, R. Towards an Automatic Framework for Solving Optimization Problems with Quantum Computers. In Proceedings of the 2024 IEEE International Conference on Quantum Software (QSW), Shenzhen, China, 7–13 July 2024; pp. 46–57. [CrossRef]
36. Volpe, D.; Quetschlich, N.; Graziano, M.; Turvani, G.; Wille, R. A Predictive Approach for Selecting the Best Quantum Solver for an Optimization Problem. *arXiv* **2024**, arXiv:2408.03613. [CrossRef]
37. Wille, R.; Berent, L.; Forster, T.; Kunasaikaran, J.; Mato, K.; Peham, T.; Quetschlich, N.; Rovara, D.; Sander, A.; Schmid, L.; et al. The MQT Handbook: A Summary of Design Automation Tools and Software for Quantum Computing. *arXiv* **2024**, arXiv:2405.17543. [CrossRef]
38. Rovara, D.; Quetschlich, N.; Wille, R. A Framework to Formulate Pathfinding Problems for Quantum Computing. *arXiv* **2024**, arXiv:2404.10820. [CrossRef]
39. Lobe, E. *quark: QUantum Application Reformulation Kernel*; GI Quantum Computing Workshop: Berlin, Germany, 2023. [CrossRef]
40. Dwave Preprocessing Toolchain. Available online: https://github.com/dwavesystems/dwave-preprocessing (accessed on 31 October 2024).
41. Boros, E.; Hammer, P.L.; Tavares, G. *Preprocessing of Unconstrained Quadratic Binary Optimization*; Technical Report; Technical Report RRR 10-2006; RUTCOR: New Brunswick, NJ, USA, 2006.
42. Orlandi, G.; Volpe, D.; Graziano, M.; Turvani, G. Qoolchain: A QUBO Preprocessing Toolchain for Enhancing Quantum Optimization. *Adv. Quantum Technol.* **2024**, 2400384. [CrossRef]
43. Albash, T.; Lidar, D.A. Adiabatic quantum computation. *Rev. Mod. Phys.* **2018**, *90*, 015002. [CrossRef]
44. Jansen, S.; Ruskai, M.B.; Seiler, R. Bounds for the adiabatic approximation with applications to quantum computation. *J. Math. Phys.* **2007**, *48*, 102111 . [CrossRef]
45. Cubitt, T.S.; Perez-Garcia, D.; Wolf, M.M. Undecidability of the spectral gap. *Nature* **2015**, *528*, 207–211. [CrossRef]
46. Volpe, D.; Cirillo, G.A.; Zamboni, M.; Turvani, G. Integration of simulated quantum annealing in parallel tempering and population annealing for heterogeneous-profile QUBO exploration. *IEEE Access* **2023**, *11*, 30390–30441. [CrossRef]
47. Forno, E.; Acquaviva, A.; Kobayashi, Y.; Macii, E.; Urgese, G. A Parallel Hardware Architecture For Quantum Annealing Algorithm Acceleration. In Proceedings of the 2018 IFIP/IEEE International Conference on Very Large Scale Integration (VLSI-SoC), Verona, Italy, 8–10 October 2018; pp. 31–36. [CrossRef]
48. D-Wave Systems. Available online: https://www.dwavesys.com/ (accessed on 31 October 2024).
49. Venturelli, D.; Mandrà, S.; Knysh, S.; O'Gorman, B.; Biswas, R.; Smelyanskiy, V. Quantum optimization of fully connected spin glasses. *Phys. Rev.* **2015**, *5*, 031040. [CrossRef]
50. Lewis, M.; Glover, F. Quadratic unconstrained binary optimization problem preprocessing: Theory and empirical analysis. *Networks* **2017**, *70*, 79–97. [CrossRef]
51. Boothby, K.; Bunyk, P.; Raymond, J.; Roy, A. Next-generation topology of d-wave quantum processors. *arXiv* **2020**, arXiv:2003.00133. [CrossRef]
52. Denchev, V.S.; Boixo, S.; Isakov, S.V.; Ding, N.; Babbush, R.; Smelyanskiy, V.; Martinis, J.; Neven, H. What is the computational value of finite-range tunneling? *Phys. Rev.* **2016**, *6*, 031015. [CrossRef]
53. Irie, H.; Liang, H.; Gongyo, S.; Hatsuda, T. Hybrid quantum annealing via molecular dynamics. *Sci. Rep.* **2021**, *11*, 8426. [CrossRef]

54. Dwave. Dwave-Hybrid. Available online: https://github.com/dwavesystems/dwave-hybrid (accessed on 31 October 2024).
55. Combarro, E.; Gonzalez-Castillo, S.; Di Meglio, A. *A Practical Guide to Quantum Machine Learning and Quantum Optimization: Hands-on Approach to Modern Quantum Algorithms*; Packt Publishing Ltd.: Birmingham, UK, 2023.
56. Farhi, E.; Goldstone, J.; Gutmann, S. A quantum approximate optimization algorithm. *arXiv* **2014**, arXiv:1411.4028. [CrossRef]
57. Blekos, K.; Brand, D.; Ceschini, A.; Chou, C.H.; Li, R.H.; Pandya, K.; Summer, A. A Review on Quantum Approximate Optimization Algorithm and its Variants. *arXiv* **2023**, arXiv:2306.09198. [CrossRef]
58. A study of the performance of classical minimizers in the Quantum Approximate Optimization Algorithm. *J. Comput. Appl. Math.* **2022**, *404*, 113388. [CrossRef]
59. Pellow-Jarman, A.; McFarthing, S.; Sinayskiy, I.; Park, D.K.; Pillay, A.; Petruccione, F. The effect of classical optimizers and Ansatz depth on QAOA performance in noisy devices. *Sci. Rep.* **2024**, *14*, 16011. [CrossRef]
60. Stein, J.; Chamanian, F.; Zorn, M.; Nüßlein, J.; Zielinski, S.; Kölle, M.; Linnhoff-Popien, C. Evidence that PUBO outperforms QUBO when solving continuous optimization problems with the QAOA. *arXiv* **2023**, arXiv:2305.03390. [CrossRef]
61. Egger, D.J.; Mareček, J.; Woerner, S. Warm-starting quantum optimization. *Quantum* **2021**, *5*, 479. [CrossRef]
62. Peruzzo, A.; McClean, J.; Shadbolt, P.; Yung, M.H.; Zhou, X.Q.; Love, P.J.; Aspuru-Guzik, A.; O'brien, J.L. A variational eigenvalue solver on a photonic quantum processor. *Nat. Commun.* **2014**, *5*, 4213. [CrossRef]
63. Tilly, J.; Chen, H.; Cao, S.; Picozzi, D.; Setia, K.; Li, Y.; Grant, E.; Wossnig, L.; Rungger, I.; Booth, G.H.; et al. The variational quantum eigensolver: a review of methods and best practices. *Phys. Rep.* **2022**, *986*, 1–128. [CrossRef]
64. Holmes, Z.; Sharma, K.; Cerezo, M.; Coles, P.J. Connecting Ansatz Expressibility to Gradient Magnitudes and Barren Plateaus. *Prx Quantum* **2022**, *3*, 010313. [CrossRef]
65. Bittel, L.; Kliesch, M. Training Variational Quantum Algorithms Is NP-Hard. *Phys. Rev. Lett.* **2021**, *127*, 120502. [CrossRef] [PubMed]
66. O'Malley, P.J.J.; Babbush, R.; Kivlichan, I.D.; Romero, J.; McClean, J.R.; Barends, R.; Kelly, J.; Roushan, P.; Tranter, A.; Ding, N.; et al. Scalable Quantum Simulation of Molecular Energies. *Phys. Rev. X* **2016**, *6*, 031007. [CrossRef]
67. Durr, C.; Hoyer, P. A Quantum Algorithm for Finding the Minimum. *arXiv* **1996**, arXiv:9607014. [CrossRef]
68. Gilliam, A.; Woerner, S.; Gonciulea, C. Grover adaptive search for constrained polynomial binary optimization. *Quantum* **2021**, *5*, 428. [CrossRef]
69. Grover, L.K. Quantum Mechanics Helps in Searching for a Needle in a Haystack. *Phys. Rev. Lett.* **1997**, *79*, 325–328. [CrossRef]
70. Sadana, S. Grover's search algorithm for n qubits with optimal number of iterations. *arXiv* **2020**, arXiv:2011.04051. [CrossRef]
71. Giuffrida, L.; Volpe, D.; Cirillo, G.A.; Zamboni, M.; Turvani, G. Engineering Grover adaptive search: Exploring the degrees of freedom for efficient QUBO solving. *IEEE J. Emerg. Sel. Top. Circuits Syst.* **2022**, *12*, 614–623. [CrossRef]
72. Baritompa, W.P.; Bulger, D.W.; Wood, G.R. Grover's Quantum Algorithm Applied to Global Optimization. *SIAM J. Optim.* **2005**, *15*, 1170–1184. [CrossRef]
73. Ominato, H.; Ohyama, T.; Yamaguchi, K. Grover Adaptive Search with Fewer Queries. *IEEE Access* **2024**, *12*, 74619–74632. [CrossRef]
74. Sano, Y.; Mitarai, K.; Yamamoto, N.; Ishikawa, N. Accelerating grover adaptive search: Qubit and gate count reduction strategies with higher-order formulations. *IEEE Trans. Quantum Eng.* **2024**, *5*, 3101712. [CrossRef]
75. Gilliam, A.; Venci, C.; Muralidharan, S.; Dorum, V.; May, E.; Narasimhan, R.; Gonciulea, C. Foundational Patterns for Efficient Quantum Computing. *arXiv* **2021**, arXiv:1907.11513. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.