Constraint-preserving quantum algorithm for the multi-frequency antenna placement problem

Matteo Vandelli[1], Francesco Ferrari[1], and Daniele Dragoni[1,2]

[1]Quantum Computing Solutions, Leonardo S.p.A., Via R. Pieragostini 80, Genova, 16151, Italy
[2]Hypercomputing Continuum Unit, Leonardo S.p.A., Via R. Pieragostini 80, Genova, 16151, Italy

***Abstract*—Quantum algorithms for combinatorial optimization typically encode constraints as soft penalties within the objective function, which can reduce efficiency and scalability compared to state-of-the-art classical methods that instead exploit constraints to guide the search toward high-quality solutions. Although solving this issue for an arbitrary problem is inherently a hard task, we address this challenge for a specific problem in the field of telecommunications, the *multi-frequency antenna placement problem*, by introducing a constraint-preserving quantum adiabatic algorithm (QAA). To this aim, we construct a quantum circuit that prepares an initial state comprising an equal superposition of all feasible solutions, and define a custom mixer that preserves both the one-hot encoding constraint for vertex coloring and the cardinality constraint on the number of antennas. This scheme can be extended to a broader range of applications characterized by similar constraints. We first benchmark the performance of this quantum algorithm against a basic version of QAA, demonstrating superior performance in terms of feasibility and success probability. We then apply this algorithm to large problem sizes with hundreds of variables using a constraint-aware decomposition method based on the SPLIT framework. Our results indicate competitive performance against other large-scale classical approaches, such as branch-and-bound and simulated annealing. This work supports previous claims that constraint-aware algorithms are crucial for the practical and efficient application of quantum methods in industrial settings.**

I. Introduction

Many combinatorial optimization problems arising in industrial applications [1], such as routing, scheduling, and resource allocation, are NP-hard and remain computationally challenging even for state-of-the-art classical algorithms like branch-and-bound [2]–[4] and simulated annealing [5]. These difficulties persist despite the enormous progress in high-performance computing (HPC).

Quantum computing approaches have been proposed as viable candidates for tackling these problems at scales currently intractable for classical digital computers, owing to their potential to significantly speed up solution space exploration [6]. In this respect, quantum computers are particularly well-suited for solving quadratic unconstrained binary optimization (QUBO) problems [7], a specific class of combinatorial optimization problems that can be directly mapped to Ising spin models [8] and natively encoded on quantum hardware. Several industrially relevant combinatorial problems can be mapped to QUBO form by encoding constraints as soft penalties in the objective function. However, this formulation often introduces significant drawbacks: it can substantially increase the number of variables and generally requires careful, problem-specific penalty tuning to balance constraint enforcement against the original objective function. As a result, penalty terms often make the optimization landscape rougher, complicating the search of feasible, high-quality solutions [9], [10].

To address these challenges, various techniques have been proposed to extend the applicability of quantum algorithms to specific classes of constrained problems beyond QUBO [11]–[13]. In the context of the popular quantum approximate optimization algorithm (QAOA) [14]–[17], this can be achieved, for instance, by using Grover-like mixers [18], [19]. General mixers satisfying linear constraints can be defined using the procedure outlined in Ref. [20], which nonetheless scales exponentially with the problem size in the case of general linear constraints. Some inequality constraints can be addressed using even more elaborate mixers based on quantum phase estimation [21].

In this work, we study the *multi-frequency antenna placement problem* (mAPP), an optimization problem with applications in Search-and-Rescue operations, and more broadly in the field of telecommunications. In mathematical terms, the mAPP translates into an unstructured quadratic program with binary variables and several equality constraints. We introduce a quantum adiabatic algorithm specifically tailored to ensure feasibility under mAPP constraints, thanks to suitable mixer operators. We first benchmark the performance of this method on small mAPP instances by comparing its results to those of a QUBO-based quantum algorithm. Then, we leverage the SPLIT decomposition scheme [22] to define a hybrid quantum-classical algorithm that can tackle intermediate and large scale instances, with hundreds of variables. An extended comparison with various classical methods, such as branch-and-bound and simulated annealing, is carried out and highlights the effectiveness of constraint-aware approaches.

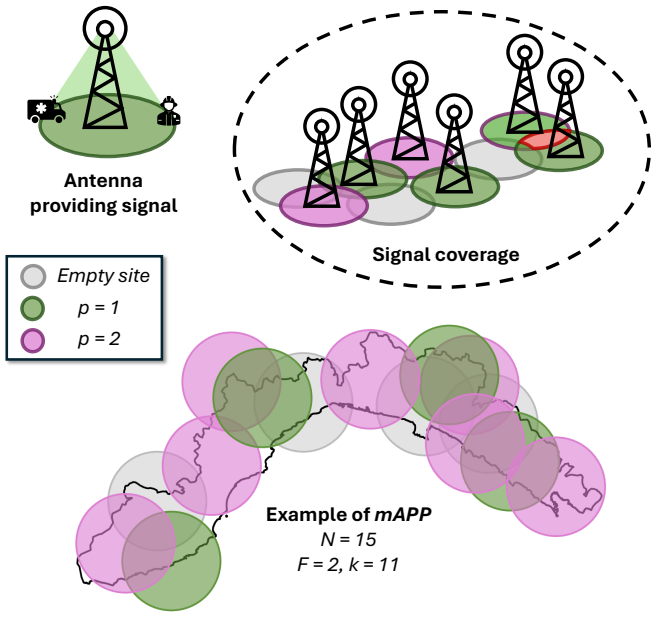

Fig. 1. Top: schematic illustration of the multi-frequency antenna placement problem. Grey areas represent empty sites ($p = 0$), while green/purple areas represent antennas operating at different frequencies ($p = 1, 2$). The red area indicates interfering signals. Bottom: solution of a mAPP instance on the Liguria region (Italy).

II. Mathematical formulation of the problem

The multi-frequency antenna placement problem is motivated by the need to create and manage an *ad-hoc* network of antennas on a territory hit by a natural disaster, providing signal coverage to the first responders on the ground (see Fig.1). In this scenario, a preliminary screening makes it possible to identify a set of potential locations at which to deploy a limited number of mobile antennas, that can be placed aboard vehicles. At that point, the available antennas have to be optimally placed on the territory in order to guarantee maximum coverage while avoiding interference. In order to achieve that, the operating frequency of the antennas can be chosen among different bands if there are overlaps between the antenna signals. The problem we formulate here aims to provide simultaneously the optimal placement of the antennas and the best frequency allocation in a given situation, providing a more realistic extension of the formulation given in Ref. [23]

In mathematical terms, we consider the problem of placing k antennas on N candidate locations, in such a way to maximize coverage and minimize interference. We assume that each antenna can be chosen to operate at one of F possible frequencies. The problem has variables x_{vp} where $v \in \{1, ..., N\}$ is the site index and $p \in \{0, 1, ..., F\}$ is the frequency index. The index $p = 0$ corresponds to an empty site, thus has to be treated differently from the other p indexes. We define $Q = N(F + 1)$ to be the total number of binary variables of the problem in this one-hot encoding scheme. In the quantum methods adopted here, Q qubits are used to encode these variables. The space of all the possible bitstrings is thus $S^Q = \{0, 1\}^{\otimes Q}$.

We define the cost function of the problem as

$$C(x) = \sum_{p,p'=1}^{F} \sum_{v<u} O_{vu}^{(p,p')} x_{vp} x_{up'} - \sum_{v=1}^{N} \sum_{p=1}^{F} A_v x_{vp} + \alpha \sum_{v=1}^{N} \sum_{p=2}^{F} p\, x_{vp} \tag{1}$$

where A_v is the area covered by site v, $O_{vu}^{(p,p')}$ represents the interference due to the signal overlap between sites v and u when occupied by antennas operating at frequency p and p', respectively. In the last term, a small coefficient α is introduced to favor solutions that use fewer frequencies. The binary quadratic program (QP) corresponding to this problem can be written as

$$\min_{x \in S^Q} C(x) \tag{2}$$

s.t.

$$\sum_{p=0}^{F} x_{vp} = 1, \quad \forall v \tag{3}$$

$$\sum_{v=1}^{N} \sum_{p=1}^{F} x_{vp} = k \tag{4}$$

The first set of constraints indicates that each site can be either assigned an antenna with frequency $p \in \{1, \ldots, F\}$ or be empty ($p = 0$). The second constraint is the *cardinality constraint* and enforces that the total number of antennas is k. The set of feasible solutions S_f^Q satisfying these constraints contains exactly $|S_f^Q| = \binom{N}{k} F^k$ elements, as opposed to the total number of possible bitstrings $|S^Q| = 2^{N(F+1)}$. This shows that the search space is decreased compared to an exhaustive search of all the possible bitstrings, but also that it can increase super-polynomially with the number of sites N if k grows with N. Note that, for given N and F, the largest feasible subspace is achieved when $k = \lfloor \frac{F}{F+1}(N + 1) \rfloor$.

III. Quantum Methodology

A. Quantum adiabatic algorithm

In the quantum approach, we solve the problem by encoding the variables x_{vp} into a set of qubits $q_{v,p}$. The quantum adiabatic algorithm (QAA) solves optimization problems by smoothly evolving the quantum system of Q qubits from an initial state $|\psi_{\text{in}}\rangle$, which is the ground state of a *mixer* Hamiltonian H_M, to the ground state of *problem* Hamiltonian H_P, which encodes the optimal solution. This slow evolution is governed by the Hamiltonian

$$H(t) = (1 - s(t))H_M + s(t)H_P. \tag{5}$$

Here, H_P can be, for instance, the Ising Hamiltonian obtained by quantization of the cost function of the problem (1), namely $H_P = C\left(\frac{1-Z}{2}\right)$, with Z being the third Pauli matrix [8].

The scheduling function $s(t)$ satisfies $s(0) = 0$ and $s(T) = 1$, where T is the total evolution time. In this work, we choose this function to be linear, namely $s(t) = t/T$. The QAA algorithm exploits the fact that, after a sufficiently slow unitary evolution U for a time T under the action of Hamiltonian $H(t)$, the final state $|\psi_{\text{out}}\rangle = U(T)\,|\psi_{\text{in}}\rangle$ has a high probability of being the ground state of H_P [24]–[27]. In the gate-based formulation of quantum computing, the continuous-time evolution of QAA is approximated by a discrete evolution, using the Trotter-Suzuki approximation, which yields a layered quantum circuit [28], [29]. Indeed, discretizing the total runtime T into L equal steps of length $\tau = T/L$, the Trotterized time evolution at first order is given by

$$U(T) \approx \prod_{l=1}^{L} \exp\Big(-i\,\tau\,\big(1-\tfrac{l}{L}\big)H_M\Big)\,\exp\Big(-i\,\tau\,\tfrac{l}{L}H_P\Big), \tag{6}$$

Different choices of the mixer and problem Hamiltonians are possible, depending on how constraints get incorporated into the QAA algorithm. This is discussed in the following two sections.

B. Basic QAA

The simplest version of the QAA (here denoted as QAA-BASIC) utilizes a local mixer, named the X-mixer, in the form $H_M = -\sum_{i=1}^{Q} X_i$, where X_i is the NOT gate acting on qubit i. Its ground state is $|\psi_{\text{in}}\rangle = |+\rangle^{\otimes Q}$, i.e. the uniform superposition of all Q-qubit bitstrings (written in the Z basis). The algorithm in this form does not support hard constraints, so the quadratic program (2) must be converted to the quadratic unconstrained binary optimization (QUBO) form [7]

$$\begin{aligned}\mathcal{Q}^{(\lambda)}(x) = C(x) &+ \lambda \sum_{v=1}^{N}\left(\sum_{p=0}^{F} x_{vp} - 1\right)^2 \\ &+ \lambda\left(\sum_{v=1}^{N}\sum_{p=1}^{F} x_{vp} - k\right)^2,\end{aligned} \tag{7}$$

In this formulation, the constraints of Eqs. (3) and (4) have been replaced by soft penalty terms with coefficient λ. The corresponding problem Hamiltonian in the QAA becomes $H_P = \mathcal{Q}^{(\lambda)}\big(\frac{1-Z}{2}\big)$. It is convenient to normalize H_P by dividing all its entries by the max norm.

C. Constraint-preserving QAA

For a constrained problem, the initial state $|\psi_{\text{in}}\rangle$ need not be the uniform superposition of all possible bitstrings, but may be suitably devised to enforce feasibility [16], [30]. In this case, the mixer H_M should be selected so that $|\psi_{\text{in}}\rangle$ is its ground state.

The problem we address in this work is particularly amenable to the application of a constraint-preserving QAA. Indeed, we can construct a quantum algorithm that exactly preserves the constraints of the Eqs. (3) and (4) by preparing an initial state $|\psi_{\text{in}}\rangle$ that is an equal superposition of all feasible states, and a mixer operator H_M that forces the system to remain in the feasible sector. This construction preserves both number and frequency constraints, so the problem Hamiltonian can simply be taken as $H_P = C(\frac{1-Z}{2})$ (rescaled by its max norm). The key ingredients of the constraint-preserving QAA, denoted in the following as QAA-APP, are:

1) Initial state preparation: The aim of state preparation is to construct a uniform superposition of feasible solutions. Here we choose to take the equal superposition of all feasible bitstrings with N sites, k antennas and F frequencies, i.e.

$$|\psi_{\text{in}}\rangle = \frac{1}{\sqrt{|S_f^Q|}} \sum_{x\in S_f^Q} |x\rangle \tag{8}$$

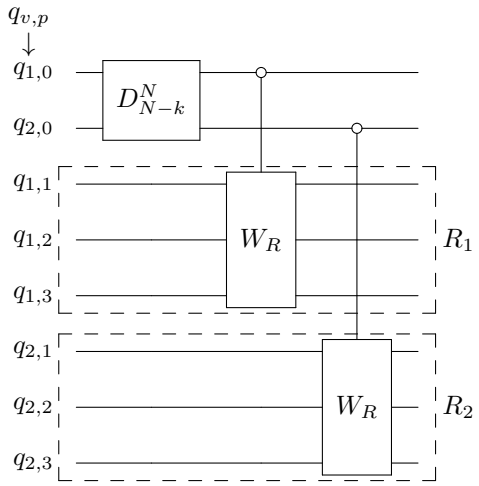

Fig. 2. Example of the state preparation circuit for the case of two sites with $F = 3$, which creates the initial mAPP state of Eq. (8). The block indicated with D^N_{N-k} creates the Dicke state with $N-k$ Hamming weight on the first $N = 2$ qubits. The W_R blocks prepare the corresponding W states on the target qubits in the R_1 and R_2 registers. The empty circle indicates an inversely controlled gate.

To explain the circuit that prepares $|\psi_{\text{in}}\rangle$, let us introduce a convenient ordering of the qubits, exemplified in Fig. 2. The first N qubits are taken to represent the variables corresponding to the *empty sites*, i.e. $q_{v,p=0}$ for $v \in \{1,\dots,N\}$. Then, N groups of F qubits are stacked consecutively in the register. Each of these groups, denoted by R_v, is formed by the qubits $\{q_{v,1},\dots,q_{v,F}\}$ for a certain site v.

Since k antennas need to be deployed in total, the feasible states are characterized by $N-k$ empty sites. Thus, the first step to prepare $|\psi_{\text{in}}\rangle$ consists of creating a Dicke state [31], [32] with Hamming weight $N-k$ for the first N qubits, i.e. $|D^N_{N-k}\rangle \otimes |0\rangle^{\otimes NF}$. This can be achieved by using the circuit proposed in Ref. [33] with gate complexity $\mathcal{O}((N-k)N) \approx \mathcal{O}(N^2)$, which is represented by the D^N_{N-k} block in Fig. 2. After that, we

apply inversely-controlled W gates, indicated as $\overline{CW}$, on the remaining qubits. This gate is defined as

$$\overline{CW}_{q,R} = |0\rangle\langle 0|_q \otimes W_R + |1\rangle\langle 1|_q \otimes \mathbb{1}_R, \tag{9}$$

where q is the control qubit and W_R is the operator that creates a W-state on the qubits in the register R. The circuit that implements W_R is given in Ref. [34], [35]. A series of $\overline{CW}$ gates is applied, each being inversely controlled on one of the first N qubits $q_{v,p=0}$ and targeting the corresponding register R_v. The resulting state

$$|\psi_{\text{in}}\rangle = \left(\prod_{v=1}^{N} \overline{CW}_{v,R_v}\right) |D^N_{N-k}\rangle \otimes |0\rangle^{\otimes NF}. \tag{10}$$

is exactly the equal superposition of all feasible states (8). An example of the state preparation circuit is depicted in Fig.2 for the case with $N = 2$ and $F = 3$. We use a version of $\overline{CW}$ that scales as $\mathcal{O}(F^2)$ in terms of gates, making the whole state preparation efficient with gate complexity $\mathcal{O}(N^2 + NF^2)$.

2) Constraint-preserving mixer for mAPP: The next important ingredient for the application of this algorithm is a constraint-preserving mixer. The mixer operator should enable exploration of the entire feasible subspace. This can be achieved by taking a two component mixer

$$H_M = H_{XY} + H_{\pm\pm}. \tag{11}$$

The first term is a regular XY mixer [13], [16] which, for each site v, acts on the $p > 0$ frequency indices:

$$H_{XY} = -\frac{\beta}{2} \sum_{v} \sum_{p=1}^{F} \big(X_{v,p}X_{v,(p \bmod F)+1} + Y_{v,p}Y_{v,(p \bmod F)+1}\big). \tag{12}$$

The second component has a 4-qubit form similar to the permutation matrix previously introduced for vehicle routing problems [36], namely

$$H_{\pm\pm} = -\beta \sum_{v<u} \sum_{p,p'=1}^{F} \left(S^+_{v,p}S^+_{u,0}S^-_{v,0}S^-_{u,p'} + \text{h.c.}\right) \tag{13}$$

where $S^{\pm} = \frac{1}{2}(X \pm iY)$ are ladder spin operators. The operator H_{XY} can increase or decrease the frequency index p at each site v, keeping $p \neq 0$. The 4-qubit term $H_{\pm\pm}$, instead, acts on pairs of sites and can swap an empty site ($p = 0$) with a site operating at finite frequency. The combination of these two terms allows to explore all feasible bitstrings. Moreover, the initial state $|\psi_{in}\rangle$ of Eq. (8) is the ground state of the Hamiltonian operator H_M within the Hilbert subspace spanned by the feasible states. Hence, this construction is a consistent application of QAA.

From a gate-based perspective, the mixer H_M presents two complications compared to the X-mixer. First, the operators in H_M do not commute with each other. As a result, the finite-time evolution under H_M is approximated using a Trotter–Suzuki decomposition (over M steps). Following the scheme outlined in Appendix A, the feasibility of the solution is always preserved, regardless of the number of Trotter steps. Additionally, it requires the application of 4-qubit gates, which typically involves compilation in terms of 2-qubit gates. Each of the terms in the decomposition can be compiled using 6 CNOT gates and a few single qubit gates. The overall gate complexity of the mixer can be estimated as $\mathcal{O}(N^2F^2M) = \mathcal{O}(Q^2M)$, where M is the number of Trotter steps. This increased circuit depth is partially balanced by the largely reduced search space S^Q_f, that allows for a reduced number of QAA layers L. We argue that these more complex state preparation and mixers result in circuit depths that will become practical only on fault-tolerant quantum hardware [37].

Method	Description
QAA-BASIC	Quantum adiabatic algorithm with local X-mixer and H_P based on the QUBO form of mAPP (7).
QAA-APP	Quantum adiabatic algorithm with the constraint-preserving mixer (11) and H_P based on the mAPP cost function (1).
CPLEX	*State-of-the-art* branch-and-bound algorithm, as implemented in the IBM ILOG CPLEX solver [38].
SA	Simulated annealing with local updates (single-variable flips), based on the QUBO formulation of mAPP (7).
CUSTOM-SA	Simulated annealing with constraint-preserving updates, based on the original mAPP cost function (1).
QAA-APP-SPLIT	SPLIT method using QAA-APP as subproblem solver, with constraint-preserving SweepUdpate and update of the subproblem number constraints (14).
CPLEX-APP-SPLIT	SPLIT method using CPLEX as subproblem solver, with constraint-preserving SweepUdpate and update of the subproblem number constraints (14).
CPLEX-APP-SPLIT -PLAIN	SPLIT method using CPLEX as subproblem solver as implemented in Ref. [22], i.e. with single-variable SweepUdpate and static subproblem number constraints (14).

TABLE I
List of quantum, classical and hybrid quantum-classical methods used in this work, with their abbreviations.

IV. Computational methods

A. Classical methods

To evaluate the performance of the above quantum algorithms, we compare the results against well-established classical methods. In Table I, we summarize the different methods employed in this work.

The first classical approach is the branch-and-bound algorithm as implemented in the IBM ILOG CPLEX solver [38], which guarantees optimality by systematically pruning the search space but has, in general, exponential worst-case complexity. In practice, when runtime becomes

prohibitive, CPLEX can also be used as an approximate solver by imposing a time limit on its execution.

The second classical method considered in this work is simulated annealing [5], a stochastic metaheuristic for local search that explores the energy landscape through probabilistic thermal fluctuations, often yielding high-quality solutions even for large-scale combinatorial problems. We evaluate the performance of two versions of simulated annealing. The first, referred to as SA, directly tackles the QUBO formulation of mAPP (2), in which constraints are implemented as soft-penalties. This variant relies on simple single-variable flips as update moves. On the other hand, the second version of simulated annealing, indicated as CUSTOM-SA, employs constraint-preserving updates that ensure the search always remains within the feasible subspace (once the system is initialized in a feasible state). The design of constraint-preserving moves essentially mimics the transitions between bitstrings allowed by the mixer Hamiltonian of Eq. (11). Specifically, we introduce two kinds of moves. The first update rule allows to change the operating frequency of an active site: we extract a random site v and identify the index p at which $x_{v,p} = 1$; if $p > 0$, we propose the move $x_{v,p} \leftrightarrow x_{v,p'}$, with $p' > 0$ a random frequency. The second update rule swaps the frequency indices of two sites: we randomly select two sites v, u and identify p_v, p_u giving $x_{v,p_v} = 1$ and $x_{u,p_u} = 1$; thus we propose the update $x_{v,p_v} \leftrightarrow x_{v,p_u}$ and $x_{u,p_u} \leftrightarrow x_{u,p_v}$. One possible effect of the latter move is to exchange $p = 0$ with $p > 0$ indices, effectively moving empty sites to different locations.

This benchmarking framework allows us to compare the performance of QAA not only with a heuristic classical baseline, but also with a state-of-the-art exact solver, thereby assessing both the quality of the solutions and the scaling behavior with problem size.

B. Problem decomposition (SPLIT)

Emulating quantum algorithm on large-scale instances exceeding 30-40 qubits remains computationally infeasible, even on modern HPC clusters. This effectively hinders the possibility of benchmarking these methods at industrially relevant scales. For this reason, in this work we combine the quantum methodology presented above with the SPLIT framework [22], which enables the decomposition of large-scale problems into smaller subproblems that can be handled more easily by emulated quantum methods. This reduction also potentially enables execution on real quantum hardware, allowing algorithms to run with a number of variables that would otherwise exceed the capacity of current devices. Notably, the advantage of adopting the SPLIT pipeline is not limited to quantum methods, as it can also extend the applicability of classical branch-and-bound algorithms to problem sizes that would otherwise be intractable within reasonable computational time. While a thorough description of SPLIT is provided in Ref. [22], here we outline the specifics and modifications applied in this work.

The first step of SPLIT consists of decomposing a mAPP instance into smaller mAPP subproblems. Let $\mathcal{C}$ denote the set of these subproblems. The variables $x_{v,p}$ of the original mAPP are assigned to the various subproblems $c \in \mathcal{C}$ via clustering. Specifically, we use *spectral clustering* based on Euclidean distances to group sites, with each resulting cluster corresponding to a subproblem. Then, for each site v, all the variables $x_{v,p=0}, \ldots, x_{v,p=F}$ are assigned to the corresponding subproblem c. This preserves the one-hot encoding structure of the mAPP variables in each subproblem.

Following subproblem decomposition, constraints are handled using the following strategy. One-hot constraints of Eq. (3) can be imposed directly on each site v, thanks to the above clustering scheme that ensures $x_{v,p=0}, \ldots, x_{v,p=F}$ belong to the same subproblem. On the other hand, the global number constraint (4) is split among subproblems by imposing a local (i.e., subproblem dependent) constraint on the number of antennas

$$\sum_{v \in c} \sum_{p>0} x_{v,p} = k_c, \quad \forall c \in \mathcal{C}. \tag{14}$$

Here $\{k_c\}_{c \in \mathcal{C}}$ are positive integer numbers that sum to the total number of antennas, i.e. $\sum_c k_c = k$. The initial choice of k_c is taken to be proportional to the fraction of sites belonging to the subproblem c, as suggested in the original formulation of SPLIT [22]. However, in this work, we introduce an improved scheme which updates the values of k_c at each iteration of the method.

To this end, we develop a routine for the SweepUpdate step of SPLIT [22] that preserves the original constraints of mAPP. We use a greedy local search method based on the two types of moves previously introduced for CUSTOM-SA: the onsite move $x_{v,p} \leftrightarrow x_{v,p'}$ is applied to all sites v, while the two-sites move is applied to all the combinations of sites belonging to different clusters. Moves are accepted if they lead to a reduction in the cost function. While both types of moves respect the mAPP constraints (3)-(4), the two-sites move does not respect the local number constraints of Eq. (14), as empty sites can be transferred between different subproblems. As a consequence, after the SweepUpdate step, the values of k_c are updated to match the newly found configuration, by computing the number of antennas assigned to each subproblem c in the current solution. This corresponds to a greedy update of the local number constraints (14) which reduces the dependence on the initial choice of k_c.

As summarized in Table I, we use SPLIT in combination with two different subproblem solvers, QAA-APP and CPLEX. In both cases, denoted by QAA-APP-SPLIT and CPLEX-APP-SPLIT (respectively), we adopt the constraint-preserving SweepUpdate routine and update the values of k_c at each iteration. To perform a comparison with the original formulation of SPLIT [22], we consider also the case of CPLEX-APP-SPLIT-PLAIN, in which k_c are kept static and the SweepUpdate routine is based on single-variable flip updates.

V. Results

A. Computational details

We proceed by describing the main metrics used to evaluate the numerical results. We generate batches of 20 mAPP instances for each problem size considered, with 'problem size' referring to the total number of variables $N(F+1)$ (i.e., the number of qubits Q in QAA methods). The different instances are constructed randomly generating candidate sites for antenna placement in the 20 regions of Italy. We assume that interference takes place only between antennas operating at the same frequencies, i.e. $O_{vu}^{(p,p')} = O_{vu}\delta_{p,p'}$ in Eq. (1).

For what concerns QAA, we consider two metrics to assess the performance:

- feasibility fraction, measuring the proportion of samples for which QAA is able to find a feasible solution to the problem

$$p_{\text{feasible}} = \frac{\#\text{ feasible counts}}{\#\text{ total counts}} \tag{15}$$

- success probability, measuring the proportion of samples of QAA that correspond to one of the exact solutions of the problem

$$p_{\text{success}} = \frac{\#\text{ exact solution counts}}{\#\text{ total counts}} \tag{16}$$

To compare the performance of different quantum and classical algorithms, we compute the normalized difference

$$\Delta\alpha = 1 - \frac{C(x_{\text{method}})}{C(x_{\text{CPLEX}})}, \tag{17}$$

which quantifies the solution quality relative to the reference value provided by CPLEX. Lower values of $\Delta\alpha$ correspond to higher quality solutions. Note that given the computational burden of large-scale instances, we set a time limit for CPLEX to 10 minutes. This limit is never reached for instances with fewer than 400 variables, which implies that, below this threshold, x_{CPLEX} coincides with the exact solution.

For methods based on CPLEX, the solution is obtained deterministically at the end of the calculation. In contrast, for stochastic methods, i.e. QAA and simulated annealing, we examine all generated samples and select the best feasible solution. Specifically, the QAA solution corresponds to the best feasible bitstring obtained from 5000 measurements. For simulated annealing approaches, the solution is taken as the best feasible result across multiple initializations: 1000 for SA and 100 for CUSTOM-SA.

As already mentioned, the reference against which we evaluate the performance of all methods is the CPLEX solver, through the ILOG CPLEX software package in Python [38]. Regarding the quantum approaches, the QAA circuits are implemented using the *Qiskit* package from IBM [39]. The decomposition of the mixer in terms of common gates is obtained using the `transpile` method contained in this library. The simulated annealing methods exploit a custom implementation in Python using *Numba* for just-in-time compilation [40]. Finally, the SPLIT framework uses the same Python implementation as in Ref. [22], with the aforementioned improvements. The solution of the subproblems is performed in parallel using the *mpi4py* package [41]. For the emulation of quantum circuits, we used a single node of our proprietary *davinci-1* cluster equipped with AMD EPYC 7402 24-Core CPUs and NVIDIA A100 GPUs. The classical calculations were performed using only the node CPUs.

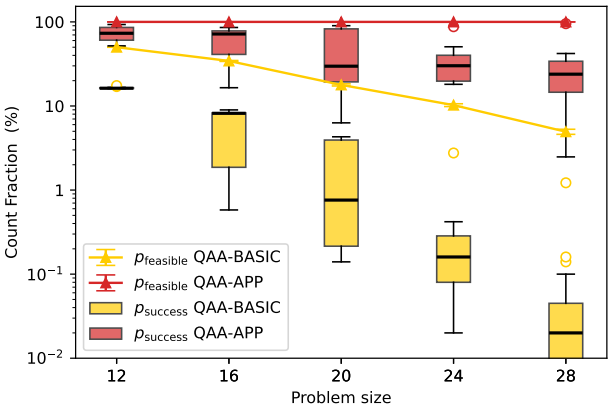

Fig. 3. Average percentage of feasibility fraction (p_{feasible}) of the various quantum methods tested on small instances (triangles and lines), superimposed to a boxplot showing the success probability (p_{success}), in percentage. Each box shows the median and interquartile range, with dots indicating outliers. Instances with different problem sizes $N(F+1)$ are considered, with $F=3$ frequencies and $N=3,4,5,6,7$ sites. The number of antennas is set to $k=\lfloor\frac{N}{2}\rfloor$.

B. Solution of small instances with QAA

We begin by investigating small problem instances with up to 28 variables, which can be solved directly using emulated QAA-BASIC and QAA-APP. For QAA-BASIC, we perform a Trotterization with $L = 100$ steps. By analyzing a handful of instances, we determine effective values for the soft penalty term λ, chosen proportional to the maximum entry of the quadratic program coefficients, and for the total evolution time. The parameters found are then used for all simulations. For QAA-APP, we use $L = 15$ Trotter steps and perform a similar analysis on a few instances to identify suitable values for the total evolution time and the mixer prefactor β. For what concerns the Trotterization of the mixer, we find that the value of M does not impact significantly the performance of QAA-APP. Hence, we settle for $M = 1$.

Our first objective is to examine the percentage of feasible solutions p_{feasible}, shown in Fig. 3. The plot highlights that the number of feasible bitstrings sampled by QAA-BASIC rapidly decreases as a function of N, while the QAA-APP retains by construction a 100% feasibility across all problem sizes. Specifically, for QAA-BASIC, we find a trend of p_{feasible} compatible with an exponential

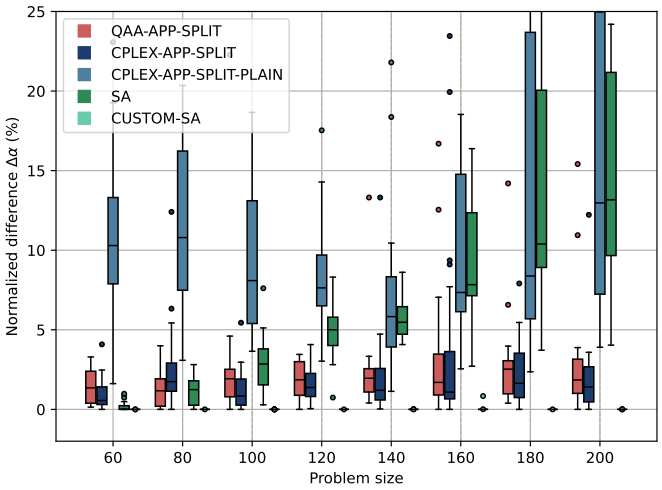

Fig. 4. Boxplots of the normalized difference $\Delta\alpha$ [Eq.(17)] for different optimization methods (indicated in the legend) across intermediate problem sizes using hybrid quantum-classical and purely classical approaches. Each box shows the median and interquartile range, with dots indicating outliers. Instances with different problem sizes $N(F+1)$ are considered, with $F=3$ frequencies and $N=15,20,25,30,35,40,45,50$ sites. The number of antennas is set to $k=\lfloor\frac{F}{F+1}(N+1)\rfloor$.

decrease as N (and thus Q) increases; hence, for a fixed number of Trotter steps L, an exponential number of measurements would be required to sample a feasible solution at least once as the problem size increases. This finding extends previous work by demonstrating that, for highly constrained problem instances, in addition to an exponential decrease in the optimal-solution probability [23], [42]–[45], also the cumulative probability of feasible solutions can likewise become negligibly small.

After analyzing feasibility, we turn to the quality of the solutions obtained using QAA-BASIC and QAA-APP. A boxplot summarizing the success probabilities obtained in the various instances is shown in Fig. 3. Our results show that p_{success} for QAA-BASIC decreases of several orders of magnitude in the range of problem sizes analyzed here, with a median trend compatible with an exponential function. This indicates that, in practice, identifying the exact solution with a limited number of measurements becomes extremely unlikely at large scales. On the other hand, the results for QAA-APP show a much slower decrease of p_{success}, with the median value remaining above 20% for the largest size considered here, compared to approximately 0.02% for QAA-BASIC. Overall, this analysis of small instances demonstrates how a QAA evolution that preserves the problem constraints can lead to considerably superior performance.

C. Addressing larger problem sizes with SPLIT

In this section, we address intermediate instances of size up to 200 variables. These instances cannot be directly solved with emulated QAA, but can be decomposed into smaller problems using SPLIT. This allows us to investigate the application of QAA-APP for larger problem sizes, which would otherwise be out of reach for emulation, and test them in combination with clustering approaches. Since QAA-BASIC can fail to provide feasible solutions already at $Q=28$, we consider here only QAA-APP in combination with SPLIT, which constitutes the hybrid method denoted as QAA-APP-SPLIT. We use a decomposition into $|\mathcal{C}|=6$ subproblems. We compare the results with different classical methods, namely SA and CUSTOM-SA, as well as CPLEX-APP-SPLIT and CPLEX-APP-SPLIT-PLAIN (see Table I).

All of the above methods provide feasible solutions for all instances tested in this section. The results in Fig. 4 show the quality of the solutions, as measured by the normalized difference $\Delta\alpha$ with respect to the exact result obtained by CPLEX. Focusing first on simulated annealing methods, we observe that the performance of SA degrades roughly linearly as the number of variables increases, with median $\Delta\alpha$ exceeding 5% at 140 variables. On the other hand, the constraint-preserving CUSTOM-SA always finds the optimal solution in the whole range of variables considered here, showing the best performance among all methods. For what concerns methods based on the SPLIT decomposition, we observe that CPLEX-APP-SPLIT consistently outperforms its plain counter-part (CPLEX-APP-SPLIT-PLAIN), further indicating that constraint-aware methods for this kind of problems produce higher quality solutions. Furthermore, we note that using QAA-APP in place of CPLEX as subproblem solver yields roughly comparable performances in the whole range of problem size considered here (see results for CPLEX-APP-SPLIT and QAA-APP-SPLIT). This further confirms the reliability of suitably designed quantum algorithms as a viable alternative to state-of-the-art classical solvers.

D. Scaling towards real-scale problem sizes

To validate the various algorithms on problems of industrially relevant size, we test them on larger instances with up to 800 variables. Note that in this regime, CPLEX also begins to struggle in both identifying and certifying the optimal solution within a reasonable computational time. Thus, to ensure consistency across experiments, we impose a time limit of 10-minutes on the branch-and-bound solver. Therefore, as optimality is no more guaranteed, other methods may outperform the reference by finding better solutions than CPLEX, hence exhibiting a negative $\Delta\alpha$. It is worth noting that none of the other classical approaches exceeds the 10-minutes time window for the instances considered here.

The results for the normalized difference $\Delta\alpha$ are collected in Fig. 5, for problem sizes ranging from 100 to 800 variables. We reiterate that, at this scale, a direct application of QAA is precluded by current emulation constraints. Consequently, we show results of QAA-APP-SPLIT, decomposing each instance into $|\mathcal{C}|=9$ subproblems, as a comparison with the classical methods. On problem

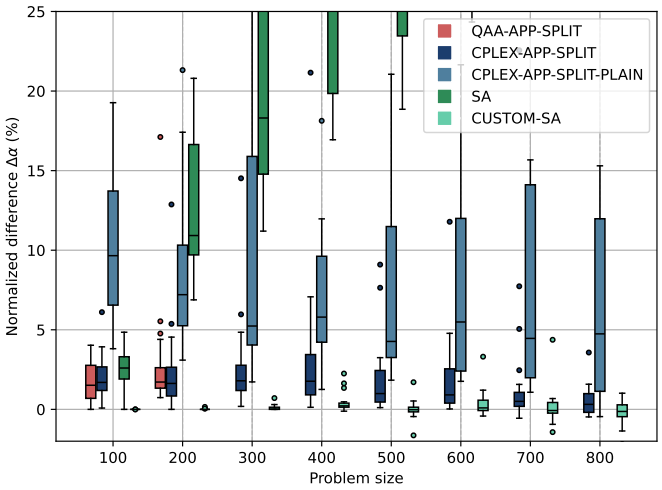

Fig. 5. Boxplots of the normalized difference $\Delta\alpha$ for different classical optimization methods (indicated in the legend) across large problem sizes. Each box shows the median and interquartile range, with dots indicating outliers. The methods shown here always find a feasible solution. Instances with different problem sizes $N(F+1)$ are considered, with $F = 4$ frequencies and $N = 20, 40, 60, 80, 100, 120, 140, 160$ sites. The number of antennas is set to $k = \lfloor \frac{F}{F+1}(N+1) \rfloor$.

instances with $Q = 100$ and $Q = 200$, QAA-APP-SPLIT displays similar quality of the solutions as CPLEX-APP-SPLIT. However, as the problem size increases, one needs to proportionally increase the number of subproblems $|\mathcal{C}|$, to ensure that the size of the subproblems remains within the classical emulation limits. This, in turn, results in performance degradation due to the approximate nature of the decomposition scheme [22]. Therefore, we assume that the performance of QAA-APP-SPLIT at larger problem sizes remains comparable to that of CPLEX-APP-SPLIT, and we proceed using only the latter method (still taking $|\mathcal{C}| = 9$).

For what concerns classical methods, constraint-aware solvers (CPLEX-APP-SPLIT and CUSTOM-SA) perform largely better than the corresponding general-purpose solvers (CPLEX-APP-SPLIT-PLAIN and SA). Indeed, while the $\Delta\alpha$ of CUSTOM-SA always lies within 1% difference from the CPLEX reference result, SA results consistently deteriorate with problem size. Analogously, CPLEX-APP-SPLIT-PLAIN exhibits a large variability in $\Delta\alpha$ on the instance pool (sometimes raising above 20%), signalling that a static distribution of the constraints k_c is in general inadequate for this kind of problems. CPLEX-APP-SPLIT, instead, achieves better performance and even finds higher-quality solutions than CPLEX for a number of instances above 600 variables. Overall, the trend of the data in Fig. 5 clearly indicates that constraint-aware approaches such as CPLEX-APP-SPLIT and CUSTOM-SA can outperform CPLEX at fixed runtime for large problem sizes.

VI. Conclusions

In this work, we presented mAPP, a discrete optimization problem that combines antenna placement and frequency allocation in a single quadratic program. This problem exhibits prototypical challenges of real-world industrial cases, such as irregular cost-function structures and coefficients, and several constraints that must be satisfied to achieve a meaningful solution. We applied a pool of quantum, classical and hybrid algorithms to the solution of mAPP, assessing their performance. Some of these methods are designed to inherently handle problem constraints, while others require conversion to a QUBO-based formulation with soft penalties.

We performed a numerical analysis on three different ranges of problem sizes. We first focused on small instances, where we tested the performance of two variants of the quantum adiabatic algorithm, showing how a constraint-preserving algorithm for mAPP (QAA-APP) consistently outperforms the standard QAA implementation based on the QUBO formulation.

We then moved to intermediate and large-scale instances, adopting the SPLIT metaheuristic [22] to extend the applicability of QAA-APP beyond the emulation limits on classical hardware. This yields the hybrid quantum-classical QAA-APP-SPLIT algorithm, which decomposes large problems into smaller subproblems, solved by QAA-APP. The numerical analysis on mAPP instances with hundreds of variables demonstrates that (i) the quality of the solutions obtained by QAA-APP-SPLIT mostly remains within 5% from the optimal result, (ii) QAA-APP-SPLIT matches the performance of its fully classical counterpart, CPLEX-APP-SPLIT, in which the CPLEX branch-and-bound algorithm [38] is used as subproblem solver. This indicates the viability of quantum algorithms as an alternative to state-of-the-art classical solvers. Furthermore, comparison with other numerical methods highlights the importance of carefully handling constraints, especially for large-scale problems, as demonstrated also by the different performance of two simulated annealing variants, one QUBO-based (SA) *vs* the other restricted to a local search within the feasible subspace (CUSTOM-SA).

Overall, our results challenge the widespread assumption in the quantum computing community that a straightforward deployment of QUBO-based quantum algorithms is sufficient to handle constraints in industrial applications. We emphasize that reformulating a quadratic program into QUBO form is often inadequate for problems with many constraints, as penalty tuning is instance-dependent and, even when done carefully, can still fail to deliver performance comparable to state-of-the-art classical methods.

The present work on mAPP, as well as previous studies on vehicle routing [36] and other problems [18], demonstrates that a problem-centric approach can achieve real-world utility, even before general-purpose approaches are developed to handle arbitrary constraints. Indeed, our QAA approach to the mAPP clearly indicates that the de-

sign of feasibility-preserving algorithms for specific problems can be formulated with relatively little effort compared to the case of generic constraints. As many combinatorial problems in the industrial sector are characterized by constraints similar to the ones discussed in this work, we argue that properly designed *ad-hoc* state preparation and mixer circuits could be implemented for certain classes of common constraints, and incorporated into existing quantum algorithms, such as QAA or QAOA, to largely improve their effectiveness.

Acknowledgments

This work was supported by the Next Generation EU program of the European Union through the Italian MUR National Recovery and Resilience Plan, Mission 4 Component 2 - Investment 1.4 - National Center for HPC, Big Data, and Quantum Computing (CN. 00000013 - SPOKE 10). We also gratefully acknowledge insightful discussions with the whole Quantum Computing Solutions team at Leonardo S.p.A., especially with Francesco Turro and Marco Maronese.

Disclaimer

The authors declare no competing interest.

Appendix A
Trotterization of the constraint-preserving mixer

The time evolution under the mixer of QAA-APP, Eq. (11), is implemented by performing a Trotter decomposition over M discrete steps, as follows

$$e^{-iH_M t} \approx \prod_{m=1}^{M} e^{-iH_{\pm\pm}\frac{t}{M}} e^{-iH_{XY}\frac{t}{M}} \tag{18}$$

Each term of the mixer is then further decomposed as

$$\begin{aligned} & e^{-iH_{XY}\frac{t}{M}} \\ & \approx \prod_{v}\prod_{p=1}^{F} e^{i\frac{\beta t}{2M}\left(X_{v,p}X_{v,(p \bmod F)+1}+Y_{v,p}Y_{v,(p \bmod F)+1}\right)} \\ & = \prod_{v}\prod_{p=1}^{F}\left[e^{i\frac{\beta t}{2M}X_{v,p}X_{v,(p \bmod F)+1}} \cdot e^{i\frac{\beta t}{2M}Y_{v,p}Y_{v,(p \bmod F)+1}}\right] \end{aligned} \tag{19}$$

and

$$\begin{aligned} & e^{-iH_{\pm\pm}\frac{t}{M}} \\ & \approx \prod_{v<u}\prod_{p,p'=1}^{F} e^{i\frac{\beta t}{M}\left(S^{+}_{v,p}S^{+}_{u,0}S^{-}_{v,0}S^{-}_{u,p'}+\text{h.c.}\right)} \\ & = \prod_{v<u}\prod_{p,p'=1}^{F}\Big[e^{i\frac{\beta t}{8M}X_{v,p}X_{u,0}X_{v,0}X_{u,p'}} \cdot e^{i\frac{\beta t}{8M}X_{v,p}Y_{u,0}X_{v,0}Y_{u,p'}} \\ & \qquad \cdot e^{i\frac{\beta t}{8M}Y_{v,p}X_{u,0}Y_{v,0}X_{u,p'}} \cdot e^{-i\frac{\beta t}{8M}X_{v,p}X_{u,0}Y_{v,0}Y_{u,p'}} \\ & \qquad \cdot e^{-i\frac{\beta t}{8M}Y_{v,p}Y_{u,0}X_{v,0}X_{u,p'}} \cdot e^{i\frac{\beta t}{8M}X_{v,p}Y_{u,0}Y_{v,0}X_{u,p'}} \\ & \qquad \cdot e^{i\frac{\beta t}{8M}Y_{v,p}X_{u,0}X_{v,0}Y_{u,p'}} \cdot e^{i\frac{\beta t}{8M}Y_{v,p}Y_{u,0}Y_{v,0}Y_{u,p'}}\Big] \end{aligned} \tag{20}$$

We observe that while a coarse Trotterization does not preserve the amplitudes of the bitstrings forming the superposition $|\psi_{\text{in}}\rangle$, the order of the operators in the present scheme always guarantees that feasibility is preserved, even for $M = 1$. Indeed the operators in the second rows of Eq. (19) and Eq. (20) are constraint-preserving. Their further decomposition (third rows of the above equations) is an exact equality, as all terms commute. Thus the final decomposition yields an operator that fully preserve mAPP constraints.

References

[1] Fotios Petropoulos, Gilbert Laporte, Emel Aktas, Sibel A. Alumur, Claudia Archetti, Hayriye Ayhan, Maria Battarra, Julia A. Bennell, Jean-Marie Bourjolly, John E. Boylan, Michèle Breton, David Canca, Laurent Charlin, Bo Chen, Cihan Tugrul Cicek, Louis Anthony Cox Jr, Christine S.M. Currie, Erik Demeulemeester, Li Ding, Stephen M. Disney, Matthias Ehrgott, Martin J. Eppler, Güneş Erdoğan, Bernard Fortz, L. Alberto Franco, Jens Frische, Salvatore Greco, Amanda J. Gregory, Raimo P. Hämäläinen, Willy Herroelen, Mike Hewitt, Jan Holmström, John N. Hooker, Tuğçe Işık, Jill Johnes, Bahar Y. Kara, Özlem Karsu, Katherine Kent, Charlotte Köhler, Martin Kunc, Yong-Hong Kuo, Adam N. Letchford, Janny Leung, Dong Li, Haitao Li, Judit Lienert, Ivana Ljubić, Andrea Lodi, Sebastián Lozano, Virginie Lurkin, Silvano Martello, Ian G. McHale, Gerald Midgley, John D.W. Morecroft, Akshay Mutha, Ceyda Oğuz, Sanja Petrovic, Ulrich Pferschy, Harilaos N. Psaraftis, Sam Rose, Lauri Saarinen, Said Salhi, Jing-Sheng Song, Dimitrios Sotiros, Kathryn E. Stecke, Arne K. Strauss, İstenç Tarhan, Clemens Thielen, Paolo Toth, Tom Van Woensel, Greet Vanden Berghe, Christos Vasilakis, Vikrant Vaze, Daniele Vigo, Kai Virtanen, Xun Wang, Rafał Weron, Leroy White, Mike Yearworth, E. Alper Yıldırım, Georges Zaccour, and Xuying Zhao. Operational research: methods and applications. *Journal of the Operational Research Society*, 75(3):423–617, 2024. doi:10.1080/01605682.2023.2253852.

[2] Ailsa H. Land and Alison G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):497–520, 1960. doi:10.2307/1910129.

[3] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial optimization: algorithms and complexity*. Prentice-Hall, Inc., USA, 1982.

[4] George L. Nemhauser and Laurence A. Wolsey. *Integer and combinatorial optimization*. Wiley-Interscience, USA, 1988. doi:10.1002/9781118627372.

[5] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. URL: https://www.science.org/doi/abs/10.1126/science.220.4598.671, arXiv:https://www.science.org/doi/pdf/10.1126/science.220.4598.671, doi:10.1126/science.220.4598.671.

[6] Amira Abbas, Andris Ambainis, Brandon Augustino, Andreas Bärtschi, Harry Buhrman, Carleton Coffrin, Giorgio Cortiana, Vedran Dunjko, Daniel J. Egger, Bruce G. Elmegreen, Nicola Franco, Filippo Fratini, Bryce Fuller, Julien Gacon, Constantin Gonciulea, Sander Gribling, Swati Gupta, Stuart Hadfield, Raoul Heese, Gerhard Kircher, Thomas Kleinert, Thorsten Koch, Georgios Korpas, Steve Lenk, Jakub Marecek, Vanio Markov, Guglielmo Mazzola, Stefano Mensa, Naeimeh Mohseni, Giacomo Nannicini, Corey O'Meara, Elena Peña Tapia, Sebastian Pokutta, Manuel Proissl, Patrick Rebentrost, Emre Sahin, Benjamin C. B. Symons, Sabine Tornow, Víctor Valls, Stefan Woerner, Mira L. Wolf-Bauwens, Jon Yard, Sheir Yarkoni, Dirk Zechiel, Sergiy Zhuk, and Christa Zoufal. Challenges and opportunities in quantum optimization. *Nature Reviews Physics*, 6(12):718–735, Dec 2024. doi:10.1038/s42254-024-00770-9.

[7] Fred Glover, Gary Kochenberger, Rick Hennig, and Yu Du. Quantum bridge analytics i: a tutorial on formulating and using qubo models. *Annals of Operations Research*, 314(1):141–183, Jul 2022. doi:10.1007/s10479-022-04634-2.

[8] Andrew Lucas. Ising formulations of many np problems. *Frontiers in Physics*, 2, 2014. URL: https://www.frontiersin.org/journals/physics/articles/10.3389/fphy.2014.00005, doi:10.3389/fphy.2014.00005.

[9] Einar Gabbassov, Gili Rosenberg, and Artur Scherer. Lagrangian duality in quantum optimization: Overcoming qubo limitations for constrained problems, 2024. URL: https://arxiv.org/abs/2310.04542, arXiv:2310.04542.

[10] Tatsuhiko Shirai and Nozomu Togawa. Compressed space quantum approximate optimization algorithm for constrained combinatorial optimization. *IEEE Transactions on Quantum Engineering*, 6:1–14, 2025. doi:10.1109/TQE.2025.3602404.

[11] Roman Martoňák, Giuseppe E. Santoro, and Erio Tosatti. Quantum annealing of the traveling-salesman problem. *Phys. Rev. E*, 70:057701, Nov 2004. URL: https://link.aps.org/doi/10.1103/PhysRevE.70.057701, doi:10.1103/PhysRevE.70.057701.

[12] Itay Hen and Marcelo S. Sarandy. Driver hamiltonians for constrained optimization in quantum annealing. *Phys. Rev. A*, 93:062312, Jun 2016. URL: https://link.aps.org/doi/10.1103/PhysRevA.93.062312, doi:10.1103/PhysRevA.93.062312.

[13] Itay Hen and Federico M. Spedalieri. Quantum annealing for constrained optimization. *Phys. Rev. Appl.*, 5:034007, Mar 2016. URL: https://link.aps.org/doi/10.1103/PhysRevApplied.5.034007, doi:10.1103/PhysRevApplied.5.034007.

[14] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm, 2014. URL: https://arxiv.org/abs/1411.4028, arXiv:1411.4028.

[15] Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D. Lukin. Quantum Approximate Optimization Algorithm: Performance, Mechanism, and Implementation on Near-Term Devices. *Phys. Rev. X*, 10:021067, Jun 2020. URL: https://link.aps.org/doi/10.1103/PhysRevX.10.021067, doi:10.1103/PhysRevX.10.021067.

[16] Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor G. Rieffel, Davide Venturelli, and Rupak Biswas. From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. *Algorithms*, 12(2), 2019. URL: https://www.mdpi.com/1999-4893/12/2/34, doi:10.3390/a12020034.

[17] Kostas Blekos, Dean Brand, Andrea Ceschini, Chiao-Hui Chou, Rui-Hao Li, Komal Pandya, and Alessandro Summer. A review on quantum approximate optimization algorithm and its variants. *Physics Reports*, 1068:1–66, June 2024. URL: http://dx.doi.org/10.1016/j.physrep.2024.03.002, doi:10.1016/j.physrep.2024.03.002.

[18] Andreas Bärtschi and Stephan Eidenbenz. Grover mixers for qaoa: Shifting complexity from mixer design to state preparation. In *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)*, pages 72–82, 2020. doi:10.1109/QCE49297.2020.00020.

[19] Ningyi Xie, Jiahua Xu, Tiejin Chen, Xinwei Lee, Yoshiyuki Saito, Nobuyoshi Asai, and Dongsheng Cai. Performance upper bound of a grover-mixer quantum alternating operator ansatz. *Phys. Rev. A*, 111:012401, Jan 2025. URL: https://link.aps.org/doi/10.1103/PhysRevA.111.012401, doi:10.1103/PhysRevA.111.012401.

[20] Hannes Leipold and Federico M Spedalieri. Constructing driver hamiltonians for optimization problems with linear constraints. *Quantum Science and Technology*, 7(1):015013, nov 2021. URL: https://dx.doi.org/10.1088/2058-9565/ac16b8, doi:10.1088/2058-9565/ac16b8.

[21] David Bucher, Jonas Stein, Sebastian Feld, and Claudia Linnhoff-Popien. If-qaoa: A penalty-free approach to accelerating constrained quantum optimization, 2025. URL: https://arxiv.org/abs/2504.08663, arXiv:2504.08663.

[22] Matteo Vandelli, Francesco Ferrari, and Daniele Dragoni. Parallel splitting method for large-scale quadratic programs, 2025. URL: https://arxiv.org/abs/2503.16977, arXiv:2503.16977.

[23] Matteo Vandelli, Alessandra Lignarolo, Carlo Cavazzoni, and Daniele Dragoni. Evaluating the practicality of quantum optimization algorithms for prototypical industrial applications. *Quantum Information Processing*, 23(10):344, Oct 2024. doi:10.1007/s11128-024-04560-1.

[24] Tosio Kato. On the adiabatic theorem of quantum mechanics. *Journal of the Physical Society of Japan*, 5(6):435–439, 1950. arXiv:https://doi.org/10.1143/JPSJ.5.435, doi:10.1143/JPSJ.5.435.

[25] Giuseppe E. Santoro, Roman Martoňák, Erio Tosatti, and Roberto Car. Theory of quantum annealing of an ising spin glass. *Science*, 295(5564):2427–2430, 2002. URL: https://www.science.org/doi/abs/10.1126/science.1068774, arXiv:https://www.science.org/doi/pdf/10.1126/science.1068774, doi:10.1126/science.1068774.

[26] Satoshi Morita and Hidetoshi Nishimori. Mathematical foundation of quantum annealing. *Journal of Mathematical Physics*, 49(12):125210, 12 2008. arXiv:https://pubs.aip.org/aip/jmp/article-pdf/doi/10.1063/1.2995837/13869474/125210_1_online.pdf, doi:10.1063/1.2995837.

[27] Tameem Albash and Daniel A. Lidar. Adiabatic quantum computation. *Rev. Mod. Phys.*, 90:015002, Jan 2018. URL: https://link.aps.org/doi/10.1103/RevModPhys.90.015002, doi:10.1103/RevModPhys.90.015002.

[28] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. Quantum computation by adiabatic evolution, 2000. arXiv:quant-ph/0001106.

[29] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, Joshua Lapan, Andrew Lundgren, and Daniel Preda. A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem. *Science*, 292(5516):472–475, 2001. URL: https://www.science.org/doi/abs/10.1126/science.1057726, arXiv:https://www.science.org/doi/pdf/10.1126/science.1057726, doi:10.1126/science.1057726.

[30] Zhihui Wang, Nicholas C. Rubin, Jason M. Dominy, and Eleanor G. Rieffel. *xy* mixers: Analytical and numerical results for the quantum alternating operator ansatz. *Phys. Rev. A*, 101:012320, Jan 2020. URL: https://link.aps.org/doi/10.1103/PhysRevA.101.012320, doi:10.1103/PhysRevA.101.012320.

[31] R. H. Dicke. Coherence in spontaneous radiation processes. *Phys. Rev.*, 93:99–110, Jan 1954. URL: https://link.aps.org/doi/10.1103/PhysRev.93.99, doi:10.1103/PhysRev.93.99.

[32] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[33] Andreas Bärtschi and Stephan Eidenbenz. Deterministic preparation of dicke states. In Leszek Antoni Gąsieniec, Jesper Jansson, and Christos Levcopoulos, editors, *Fundamentals of Computation Theory*, pages 126–139, Cham, 2019. Springer International Publishing.

[34] Diogo Cruz, Romain Fournier, Fabien Gremion, Alix Jeannerot, Kenichi Komagata, Tara Tosic, Jarla Thiesbrummel, Chun Lam Chan, Nicolas Macris, Marc-André Dupertuis, and Clément Javerzac-Galy. Efficient Quantum Algorithms for GHZ and W States, and Implementation on the IBM Quantum Computer. *Advanced Quantum Technologies*, 2(5–6), April 2019. URL: http://dx.doi.org/10.1002/qute.201900015, doi:10.1002/qute.201900015.

[35] Fırat Diker. Deterministic construction of arbitrary w states with quadratically increasing number of two-qubit gates. *AIP Advances*, 15(7), July 2025. URL: http://dx.doi.org/10.1063/5.0241266, doi:10.1063/5.0241266.

[36] Ningyi Xie, Xinwei Lee, Dongsheng Cai, Yoshiyuki Saito, Nobuyoshi Asai, and Hoong Chuin Lau. A feasibility-preserved quantum approximate solver for the capacitated vehicle routing problem. *Quantum Information Processing*, 23(8):291, Jul 2024. doi:10.1007/s11128-024-04497-5.

[37] Jens Eisert and John Preskill. Mind the gaps: The fraught road to quantum advantage, 2025. URL: https://arxiv.org/abs/2510.19928, arXiv:2510.19928.

[38] IBM ILOG Cplex. User's Manual for CPLEX V22.1. 2022. URL: https://www.ibm.com/docs/en/icos/22.1.1?topic=mc-what-is-cplex.

[39] Qiskit contributors. Qiskit: An open-source framework for quantum computing, 2023. doi:10.5281/zenodo.2573505.

[40] Siu Kwan Lam, Antoine Pitrou, and Stanley Seibert. Numba: a llvm-based python jit compiler. In *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, LLVM '15, New York, NY, USA, 2015. Association for Computing Machinery. doi:10.1145/2833157.2833162.

[41] Lisandro Dalcin and Yao-Lung L. Fang. mpi4py: Status update after 12 years of development. *Computing in Science & Engineering*, 23(4):47–54, 2021. doi:10.1109/MCSE.2021.3083216.

[42] Ajinkya Borle, Vincent E. Elfving, and Samuel J. Lomonaco. Quantum approximate optimization for hard problems in linear algebra. *SciPost Phys. Core*, 4:031, 2021. URL: https://scipost.org/10.21468/SciPostPhysCore.4.4.031, doi:10.21468/SciPostPhysCore.4.4.031.

[43] Dennis Willsch, Madita Willsch, Fengping Jin, Kristel Michielsen, and Hans De Raedt. Gpu-accelerated simulations of quantum annealing and the quantum approximate optimization algorithm. *Computer Physics Communications*, 278:108411, 2022. URL: https://www.sciencedirect.com/science/article/pii/S0010465522001308, doi:10.1016/j.cpc.2022.108411.

[44] Ruslan Shaydulin, Changhao Li, Shouvanik Chakrabarti, Matthew DeCross, Dylan Herman, Niraj Kumar, Jeffrey Larson, Danylo Lykov, Pierre Minssen, Yue Sun, Yuri Alexeev, Joan M. Dreiling, John P. Gaebler, Thomas M. Gatterman, Justin A. Gerber, Kevin Gilmore, Dan Gresh, Nathan Hewitt, Chandler V. Horst, Shaohan Hu, Jacob Johansen, Mitchell Matheny, Tanner Mengle, Michael Mills, Steven A. Moses, Brian Neyenhuis, Peter Siegfried, Romina Yalovetzky, and Marco Pistoia. Evidence of scaling advantage for the quantum approximate optimization algorithm on a classically intractable problem. *Science Advances*, 10(22):eadm6761, 2024. URL: https://www.science.org/doi/abs/10.1126/sciadv.adm6761, arXiv:https://www.science.org/doi/pdf/10.1126/sciadv.adm6761, doi:10.1126/sciadv.adm6761.

[45] J. A. Montañez-Barrera and Kristel Michielsen. Toward a linear-ramp qaoa protocol: evidence of a scaling advantage in solving some combinatorial optimization problems. *npj Quantum Information*, 11(1):131, Aug 2025. doi:10.1038/s41534-025-01082-1.