Penalty-free approach to accelerating constrained quantum optimization

David Bucher,[1,2,*] Jonas Stein,[2] Sebastian Feld,[3] and Claudia Linnhoff-Popien[2]

[1]*Aqarios GmbH, Munich, Germany*
[2]*LMU Munich Department for Computer Science, Munich, Germany*
[3]*Delft University of Technology, Delft, The Netherlands*

(Received 23 May 2025; accepted 17 October 2025; published 2 December 2025)

Traditional methods for handling (inequality) constraints in the quantum approximate optimization *algorithm* (QAOA) typically rely on penalty terms and slack variables, which increase problem complexity and expand the search space. More sophisticated mixer-based QAOA variants restrict the search within the feasible assignments but often suffer from prohibitive circuit complexity. This paper presents a penalty-free formalism for incorporating inequality constraints into the cost function of QAOA using an oracle-based subroutine that evaluates constraint satisfaction in an additional register, subsequently called indicator function QAOA (IF-QAOA). Applied to the knapsack problem, we demonstrate in numerical simulations the superior performance of IF-QAOA over conventional penalty-based approaches. Using advanced QAOA simulation techniques, we find that IF-QAOA achieves significantly higher solution quality and a faster time to solution in 82% of our benchmark cases, even though circuit depth is approximately three times larger. Analysis of the scaling behavior shows favorable scaling of IF-QAOA compared to penalty-based methods. Also, benchmarks against the recently developed quantum tree generator QAOA for knapsack problems (P. Christiansen *et al.*, arXiv:2411.00518) demonstrated higher solution quality for circuits of similar depth. Additionally, this paper introduces a method for approximating the indicator function when the number of ancillary qubits is limited or the constraint function is noninteger. With a specialized simulation algorithm based on projective measurements, we empirically demonstrate that this formalism can encode general inequality constraints using a fixed number of ancillary qubits.

DOI: 10.1103/fb5m-cl9m

I. INTRODUCTION

Recent breakthroughs in quantum computing, from improved coherence times to larger qubit counts, have brought us closer to solving computational problems that remain intractable for classical computers [1,2]. Among the most promising applications of quantum computing (QC) is their potential to solve hard combinatorial optimization problems (COPs), which lie at the heart of many real-world challenges, ranging from portfolio optimization in finance [3–5] over resource allocation in logistics [6–8] to track reconstruction and jet clustering in high-energy physics [9–11]. While universal fault-tolerant quantum computers remain a future prospect, current noisy intermediate-scale quantum (NISQ) devices are already being explored for optimization tasks, making efficient quantum optimization algorithms increasingly relevant [12].

COPs represent a fundamental class of computational problems where we seek optimal solutions from a discrete search space. Formally, (binary) COPs can be defined as finding an optimal solution $x^* \in \{0,1\}^N$ that minimizes an objective function $f(x)$ while satisfying different types of constraints. Those can involve equality constraints, meaning a compound quantity $h_j(x)$ has to equal a fixed value, or inequality constraints, referring to hard lower (or upper) bounds of $g_i(x)$:

$$x^* = \arg\min_x f(x) \quad \text{such that} \quad g_i(x) \geqslant b_i \quad \forall i$$
$$h_j(x) = b_j \quad \forall j.$$

The objective and constraining functions f, g_i, h_j are typically of linear order but can generally take quadratic or even higher-order polynomial form.

Many significant real-world problems alongside Karp's 21 famous NP-complete problems can be formulated with this COP framework [13–15]. A particularly important subclass of COPs is the quadratic unconstrained binary optimization (QUBO), where the objective function is quadratic, and constraints are only present in the form of penalty terms. QUBOs have gained special attention in quantum computing due to their natural mapping to quantum systems through the isomorphism to quadratic Ising Hamiltonians [15].

This mapping and the computationally challenging task of solving QUBOs, i.e., finding the ground state of Ising Hamiltonians, motivated the development of two major quantum optimization approaches: quantum annealing and gate-based algorithms. Quantum annealers are special-purpose devices that exploit the adiabatic theorem of quantum mechanics

*Contact author: david.bucher@aqarios.com

[16,17] to find the ground state, restricted to solving QUBOs. In the gate-based paradigm, the primary focus of research is the quantum approximate optimization algorithm (QAOA), developed by Farhi *et al.* [18], which can be seen as a parametrized Trotterization of the annealing process. Parameterization and variational optimization help reduce the number of Trotterization steps, thereby lowering circuit depth, which is inevitable in the NISQ-era of QC.

Since most industry-relevant COPs feature constraints, they have to be transformed into an unconstrained optimization format to be solved on quantum devices. While a significant number of problems have been transformed to QUBO [14,15], this requires reformulating the constraints as (quadratic) penalty terms and adding them to the objective function. Despite being generally applicable, this method suffers from increasing the cost landscape's complexity [19]. Especially, inequality constraints pose a significant challenge since they also enlarge the search space by introducing slack variables into the optimization problem [15].

This paper introduces indicator function QAOA (IF-QAOA), an oracle-based approach for incorporating inequality constraints into QAOA. Our method employs circuit subroutines that encode constraint satisfaction information in an ancillary qubit. This enables the construction of an objective functions in which the (negative) cost $f(x)$ is applied conditionally based on constraint satisfaction ($g(x) \geqslant 0$), namely $\tilde{f}(x) = f(x)\Theta[g(x)]$, where Θ represents the Heaviside step function. A key advantage of this formulation is that it eliminates the need for a penalty parameter that is otherwise needed to tune the strength of the penalty term added to the objective function. The IF can be implemented efficiently regarding qubit and gate requirements through quantum phase estimation (QPE) of the constraint function $g(x)$.

While the concept of oracle-based routines in QAOA has already been proposed [20], previous implementations for inequality constraints relied on linear penalty functions, only applied to infeasible states, requiring an additional penalty factor [21,22]. Another implementation [23] measures the oracle qubit followed by a classically controlled application of one out of two cost operators. Consequently, the state collapses to either the feasible or infeasible subspace, potentially impairing the optimization performance of the QAOA algorithm.

Besides the definition of IF-QAOA, this work also presents a benchmark analysis of IF-QAOA compared to baseline QUBO approaches (slack variables with quadratic penalties) on the knapsack problem, a COP with a single inequality constraint, through a specialized simulation technique developed for IF-QAOA. The simulation results show significant performance improvements in solution quality and time to solution (TTS). Additionally, we demonstrate that our generally applicable IF-QAOA shows improved solution quality compared to the recently developed state-of-the-art approach, specialized to the knapsack problem, the quantum tree generator (QTG) QAOA [24].

Furthermore, we investigate the case where the oracle is approximate, i.e., the QPE bits cannot resolve all values of the constraining function g due to a limited number of ancillary qubits. Nevertheless, we show that a projective measurement helps to keep the algorithm well behaved at the expense of some failure probability. Furthermore, this technique also allows efficient simulation of different ancillary register sizes, giving us numerical results on the implications of the failure probability on TTS.

This paper makes the following contributions:

(i) A generally applicable framework for embedding inequality constraints into QAOA without requiring penalty terms, called IF-QAOA;

(ii) Efficient simulation techniques for IF-QAOA with ideal and approximate oracles;

(iii) Extensive numerical benchmarks on knapsack problems, demonstrating IF-QAOA's effectiveness compared to conventional quadratic penalty methods and recent state-of-the-art QTG-QAOA in ideal simulation.

The paper is organized in the following way. Section II introduces relevant background on quantum optimization, and Sec. III reviews previous approaches to constraint handling in quantum optimization. Section IV presents our indicator function methodology and theoretical framework. Section V provides the numerical simulations and analyzes the performance of our approach in both exact and approximate implementations. Finally, Sec. VI concludes with a discussion of practical implications and future directions.

II. BACKGROUND

A. Quantum approximate optimization algorithm

Farhi *et al.* first introduced the QAOA to find the maximum cut in three-regular graphs [18]. Nevertheless, QAOA can be applied to find the optimal solution to any QUBO or even higher-order optimization problem [25]. The algorithm is initialized in an equal superposition of N qubits $|+\rangle^{\otimes N}$ for a problem consisting of N binary decision variables. This is followed by p repetitions of two different unitary operations in alternating order, designed to progressively increase the probability of measuring an optimal bit string x^* upon final measurement:

$$|\boldsymbol{\beta}, \boldsymbol{\gamma}\rangle = U_M(\beta_p)U_f(\gamma_p)\cdots U_M(\beta_1)U_f(\gamma_1)|+\rangle^{\otimes N}, \quad (1)$$

where every layer is parameterized by the angles β_i and γ_i. The cost layer U_f is given by the evolution of the diagonal cost Hamiltonian $H_f = \sum_x f(x)|x\rangle\langle x|$

$$U_f(\gamma) = \exp(-i\gamma H_f). \quad (2)$$

The mixer unitary U_M is the evolution of the standard mixer Hamiltonian $H_M = \sum_i \sigma_x^{(i)}$:

$$U_M(\beta) = \exp(-i\beta H_M) = \prod_i R_X^{(i)}(2\beta), \quad (3)$$

where $R_X^{(i)}$ is the Pauli-X rotation on qubit i.

The QAOA circuit consists of the free rotation angles $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$, which must be explicitly selected for each problem instance to ensure effective optimization. This property classifies QAOA as a variational quantum algorithm. We aim to find parameters that lead to a high probability $P^* = |\langle x^*|\boldsymbol{\beta}, \boldsymbol{\gamma}\rangle|^2$ of measuring an optimal bit string x^* at the end of the algorithm execution. For cases where H_f has a degenerate ground state, we aggregate the probabilities of measuring any ground state into P^*. However, since x^* is not accessible *a priori*, optimizing for P^* is infeasible. Consequently, QAOA minimizes the expectation value of the cost Hamiltonian H_f instead:

$$C(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \langle \boldsymbol{\beta}, \boldsymbol{\gamma} | H_f | \boldsymbol{\beta}, \boldsymbol{\gamma} \rangle = \sum_{x=0}^{2^N-1} f(x) |\langle x | \boldsymbol{\beta}, \boldsymbol{\gamma} \rangle|^2. \quad (4)$$

The variational protocol embeds the quantum algorithm within a classical loop for optimizing the continuous parameters

$$\min_{\boldsymbol{\beta},\boldsymbol{\gamma}} C(\boldsymbol{\beta}, \boldsymbol{\gamma}). \quad (5)$$

When the type of classical optimizer chosen requires gradients to be computed, the parameter shift rule or adjoint differentiation are typically employed [26,27].

Since QAOA can be seen as a Trotterization of the adiabatic algorithm for optimization problems, we can also predefine the angles to growing γ_is and shrinking β_is. This initialization strategy has been successfully demonstrated for QAOA, even for a relatively low number of QAOA layers p [28,29].

B. Knapsack problem

The 0-1 knapsack problem is the most basic example of a combinatorial optimization problem with a single inequality constraint. The problem is defined as finding the optimal selection of items $I^* \subseteq I, |I| = N$ that maximizes its total *value* while adhering to an absolute weight capacity $W > 0$. Each item $(w_i, v_i) \in I$ is characterized by its individual weight $w_i \geqslant 0$ and a value $v_i \geqslant 0$. Generally, w_i, v_i, and W are considered integer valued. Formalized, this optimization can be expressed as follows:

$$\arg\max_{x\in\{0,1\}^N} \sum_i v_i x_i \quad \text{such that} \quad \sum_i w_i x_i \leqslant W, \quad (6)$$

where the set of binary variables x_i indicates whether the ith item is selected ($x_i = 1$) or not ($x_i = 0$) [30].

The knapsack problem belongs to the NP-hard complexity class, implying that it is at least as computationally difficult as the hardest problem within NP [31]. As a consequence, assuming P $\neq$ NP, no polynomial-time algorithm can exist to solve the knapsack problem optimally for all instances. Yet, in practice, many practically relevant knapsack instances can often be solved efficiently within polynomial time scaling using dynamic programming or branch-and-bound methods [32,33]. The problem finds numerous applications, for instance, in financial portfolio optimization [30], and serves as a foundation for various logistical and operational optimization problems, including the bin-packing problem [34].

We can generalize Eq. (6) into the following form:

$$\arg\min_{x\in\{0,1\}^N} f(x) \quad \text{s.t.} \quad g_i(x) \geqslant 0 \quad \forall i \in 1, \dots, k, \quad (7)$$

which allows us to consider any inequality-constrained binary programming problems. In the knapsack case, $k = 1$, $f(x) = -v^T x$ and $g_1(x) = W - w^T x$. We will subsequently call $\mathcal{D} = \{0, 1\}^N$ the domain space of the problem and $\mathcal{D} \supseteq \mathcal{F} = \{x \in \mathcal{D} \,|\, g_j(x) \geqslant 0 \,\forall j\}$ the feasible subspace, see Fig. 1 for an exemplary visualization. Let us also assume without loss of generalization that $f(x) \leqslant 0 \,\forall x \in \mathcal{D}$, which is naturally the case with knapsack and can generally be ensured by subtracting an upper bound of f from itself.

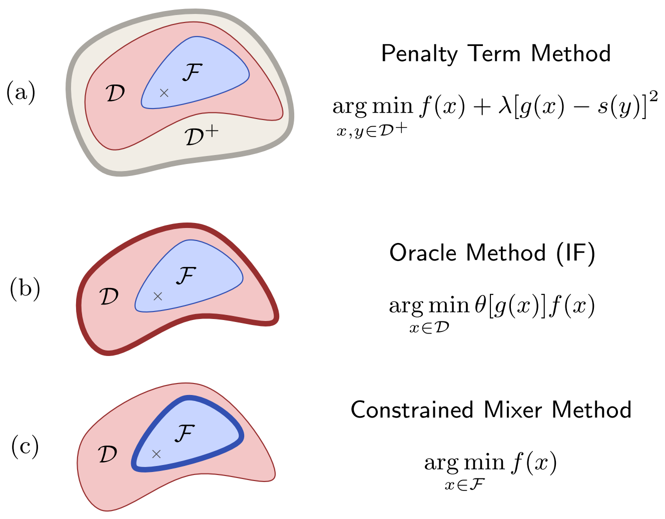


FIG. 1. Showcasing the search spaces of the various optimization strategies. In (a), the search space is enlarged to $\mathcal{D}^+$ through slack variables in the penalty term. In (b), the search space is exactly the space of the originally defined problem, and in (c), the search space is effectively reduced to the feasible subspace. The shapes of the domains are exemplary.

III. LITERATURE REVIEW

Before focusing on an in-depth review of constraint enforcement methods for QAOA, we briefly discuss alternative quantum optimization algorithms that are able to include constraints in the optimization. Based on one of the most prominent algorithms for theoretical quantum advantage, the Grover algorithm for unstructured search [35], the quantum minimum finding has emerged as a modification for solving general optimization problems [36]. The oracle-based algorithm was later extended to also feature general constraints [37]. Recently, quantum minimum finding has been successfully employed in combination with quantum tree generators (QTGs) to solve the knapsack problem specifically [38]. Other *Ansätze* feature hybrid methods that ensure constraint satisfaction through classical postprocessing [39] or an iterative quantum-classical hybrid loop [40,41].

Now, we systematically review the various methodologies developed in the literature for handling constraints in QAOA, focusing primarily on inequality constraints.

A. Penalty-based methods

The conventional approach to incorporating constraints into QAOA involves reformulating them into QUBO format and augmenting the objective function with a quadratic penalty [15].

Taking knapsack as an example, this reformulation requires introducing ancillary binary variables (slack variables) to encode the feasible region of the inequality constraint. Specifically, the interval until the maximum capacity $[0, W]$ must be represented through slack variable assignments, denoted by y. The logarithmic integer encoding [15] provides a qubit-efficient representation through a slack term $s(y) = \sum_j a_j y_j$, where $a_j = 2^j$ for $j < M$, and $a_M = W - 2^{M-1} + 1$ with $M = \lceil \log_2 W \rceil$. The resulting QUBO formulation becomes:

$$\min_{x,y \in \mathcal{D}^+} f(x) + \lambda[g(x) - s(y)]^2, \tag{8}$$

where $\lambda > 0$ is a penalty coefficient and $\mathcal{D}^+ = \mathcal{D} \times \{0, 1\}^M$ is the extended search space.

This penalty-based approach presents two significant challenges. First, it introduces an additional hyperparameter λ that needs to ensure no infeasible state has lower energy than a feasible one. With the square of the constraint violation, the penalty term can overshadow the initial problem and significantly skew the energy spectrum, leading to a more complex energy landscape to optimize [42]. Second, slack variables expand the search space from $\mathcal{D}$ to $\mathcal{D}^+$, as illustrated in Fig. 1(a), making the QUBO problem considerably larger and therefore harder to solve.

While these drawbacks are inherent to QUBO-based optimization methods, several techniques have been proposed to mitigate them. Cross-entropy methods can help determine appropriate penalty factors [43], although at the cost of increased computational overhead. Recent numerical studies suggest that incorporating the penalty term in the quantum routine while excluding slack variable assignments from the classical expectation value calculation yields promising results [44].

Alternatively, slack-variable-free approaches have emerged in literature: Gabbassov *et al.* [45] propose incorporating the knapsack constraint through Lagrangian duality, eliminating the need for slack variables. While Lagrangian methods typically require iterative penalty updates, the authors argue that in approximate optimization scenarios, a well-chosen initial penalty coefficient suffices for boosting the probability of sampling optimal or near-optimal solutions. Similarly, Montañez-Barrera *et al.* [46] propose unbalanced penalization that uses a linear and quadratic term without slack variables to approximate an exponential barrier penalty function. Van Dam *et al.* [47] present another slack-free approach, specifically for knapsack, that warm starts QAOA based on a distribution derived from the value-to-weight ratio of the items and uses specialized copula mixer Hamiltonians to bias the exploration towards that distribution. Numerically, all three approaches manage to demonstrate significant improvements over baseline methods.

B. Oracle methods

Generally, objective functions in QAOA are of QUBO or sometimes higher-order polynomial form [25]. However, it is possible to use the full capabilities of gate-based QC to construct more complex cost functions. Hadfield *et al.* [20] proposed using an oracle subroutine: a QPE as part of the cost unitary routine that synthesizes information about constraint satisfaction onto the state of a single qubit, which can be seen as an indicator qubit. While concrete constructions of this method are not further followed in Refs. [20–23,25–48] explore this path to solve the knapsack and EV charging problem. Reference [21] utilizes arithmetic adder circuits to compute $g(x)$ on separate registers, Ref. [48] employs a quantum digit (qudit) for implementation, and Ref. [22] uses the QPE/QFT-adder based technique [49]. All methods implement the following unconstrained nonlinear cost function

$$f(x) + \lambda(1 - \Theta[g(x)])g^a(x), \tag{9}$$

adding a linear penalty ($a = 1$) if the constraint is violated, where $\Theta[g(x)] = 1$ if $g(x) \geqslant 0$ and zero otherwise, also called the Heaviside step function. Refs. [21,22] only investigate $a = 1$, while Ref. [48] considers $a \in \{0, 1, 2\}$, with $a = 1$ working best. The method investigated by Ref. [22] is similar to IF-QAOA, but they require a penalty setting, which IF-QAOA overcomes, as explained in Sec. IV. While Refs. [21,22,48] demonstrate small-scale numerical success of their method, they lack an extensive numerical analysis towards more meaningful problem sizes.

In principle, oracle techniques search the entire domain space of the problem $\mathcal{D}$. They can, therefore, be considered more efficient than the slack-variable methods, see Fig. 1(b). A similar QPE-based oracle concept has been brought forward by Ref. [23] in the context of constrained qudit (quantum digit) optimization. However, the authors propose a measurement of the indicator qubit, leading to the state collapsing to either the feasible or infeasible subspace of the Hilbert space. A cost Hamiltonian with penalization is applied if the indicator qubit measurement reveals the infeasible region. Then, the state vector is in a superposition of only infeasible states, which the subsequent QAOA iterations must revert. This raises the question of whether restarting upon failed measurement is more efficient. IF-QAOA is entirely cost-function based and does not rely on measurements, as long as sufficient ancillary variables are employed and no approximation is required.

Constraint-preserving techniques represent a class of QAOA circuit architectures that operate entirely within the feasible Hilbert subspace

$$\mathcal{H}_{\mathcal{F}} = \text{span}\{|x\rangle,\ x \in \mathcal{F}\} \subseteq \mathcal{H} \tag{10}$$

with $\mathcal{F}$ being the feasible subspace.

While these methods explore the smallest possible search space by construction (see Fig. 1), the required circuit complexity to enforce constraints can be substantial. This creates a fundamental tradeoff between search space size and implementation cost that varies significantly with the type of constraint being enforced.

1. Quantum alternating operator Ansatz

Hadfield *et al.* [50] derived that QAOA mixers need not be Hamiltonian evolutions but can be arbitrary unitaries that enable transitions between all feasible states. This insight allows for the design of constraint-preserving mixers that prevent transitions from feasible to infeasible states, formally expressed as $\langle x|U_M(\beta)|y\rangle = 0, \forall \beta, x \in \mathcal{F}, y \notin \mathcal{F}$. The XY mixer exemplifies this approach, effectively enforcing one-hot constraints (Hamming weight 1 bit strings) [51].

Hamming weight constraints, i.e., max-k-vertex covering constraints [52], share a vital property, making the construction of mixer routines feasible. Namely, their solution structure is known beforehand. In contrast, inequality constraints lack inherent structure and depend heavily on constraint-specific data, leading to substantially more challenging construction of mixer operators. While methods exist for inequality constraints, they often incur prohibitive implementation costs, especially for nonsparse cases [53–55].

In the alternating operator context, Ref. [56] developed a mixer, as well as a hybrid algorithm, for the maximum independent set problem that iteratively checks the satisfaction of each constraint, and only applies the mixer if the resulting state is also satisfied.

Reference [19] introduced a constructive approach that maps the feasible subspace onto a reduced set of qubits with matching Hilbert space dimension. Each mixer operation combines the mapping unitary, standard X mixer on the reduced qubits, and the inverse mapping. While effective for structured constraints, inequality constraints require a machine learning method to determine the mapping unitary, which the authors also successfully demonstrated. However, the variational *Ansatz* is never guaranteed to produce a completely valid mapping unitary. Furthermore, this method introduces an additional variational optimization loop before QAOA optimization, increasing runtime overhead.

2. *Grover mixers*

Based on the works of Ref. [50], Bärtschi and Eidenbenz [57] developed a novel mixer method using a circuit for preparing the equal superposition of all feasible states. By combining this state preparation circuit with the Grover diffusion operator, they construct the following mixer:

$$U_M(\beta) = \exp\left(-\frac{i\beta}{|\mathcal{F}|}\sum_{x,y\in\mathcal{F}}|x\rangle\langle y|\right), \tag{11}$$

that restricts probability transfer between feasible states only. This *Ansatz* effectively shifts the complexity of synthesizing an appropriate mixer operator to the state preparation circuit. For constraints yielding regular state patterns, such as one-hot constraints [58], efficient implementations exist through short-depth state preparation circuits. However, generating an equal superposition of solely feasible solutions under an inequality state may be computationally hard [53] (e.g., the number of feasible solutions has to be known beforehand).

Quantum tree generator (QTG) offers a solution by constructing nonuniform superpositions of feasible states for a single linear inequality constraint, specifically the knapsack problem [38]. While initially intended to be used in the Grover adaptive search context, the superposition state can also be used to construct a Grover-like mixer for the QAOA, as demonstrated successfully in Ref. [24]. The authors coined the term AAM-QAOA because of the nonuniform superposition and proved the convergence of such mixers. Numerical simulations have shown significant performance improvements compared to the results from Ref. [47].

3. *Zeno dynamics*

Herman *et al.* [59] proposed a constraint mixer for inequality constraints using the default X mixer and quantum Zeno dynamics, where repeated projective measurements constrain the quantum state within the feasible subspace. While this method has been successfully demonstrated numerically and on trapped-ion hardware, solving the portfolio optimization problem, the Zeno approach requires dissecting the mixer operator into small time steps. In that way, Zeno dynamics increases the probability of successful projective measurements on indicator qubit. The measurement circuit is similar to the one explored in Ref. [23]. This dissection leads to lengthy circuits that may be obstructive in delivering a general performance advantage. There is a tradeoff between circuit depth and overall success probability.

IV. METHODS

A. Single constraint algorithm

The IF-QAOA is a constraint integration method that only applies the (polynomial) cost function $f(x) \leqslant 0\ \forall x \in \{0,1\}^N$ to the feasible solution candidates without including slack variables in the optimization problem, but using ancillary qubits for oracle construction. Let us consider an optimization problem with a single inequality constraint $g(x) \geqslant 0$ with an integer-valued image $g(\mathcal{D}) \in \{g^-,\dots,g^+\} \subset \mathbb{Z}$, contained between defined upper and lower bounds. In the case of linear f and g, it resembles the knapsack problem with integer weights. We define the minimization objective for the IF-QAOA as follows:

$$\tilde{f}(x) = f(x)\Theta[g(x)]. \tag{12}$$

Since $\Theta[g(x)] = 1$ if $g(x) \geqslant 0$ and therefore $x \in \mathcal{F}$, $\tilde{f}(x) \leqslant \tilde{f}(y) = 0\ \forall x \in \mathcal{F}, y \notin \mathcal{F}$, because $f(x) \leqslant 0\ \forall x \in \mathcal{D}$.

Therefore, the cost layer routine of the IF-QAOA circuit needs to have the following effect on the phase of a computational basis state $|x\rangle$

$$U_{\text{IF-}f}(\gamma)|x\rangle = e^{-i\gamma\tilde{f}(x)}|x\rangle. \tag{13}$$

To assemble the phase circuit, we construct the constraint Hamiltonian $H_g = \sum_x g(x)|x\rangle\langle x|$, diagonal in the computational basis, which can subsequently be used to perform QPE with M qubits in an ancillary register on the unitary $U_g = e^{2\pi i H_g/2^M}$. Gate-level implementation of the U_g is possible through multicontrolled phase gates, as long as g is polynomial. The number of ancillary qubits M is defined by the resolution required for g. Since $g(x) \in \mathbb{Z}$, we define $M = \max\{\lceil\log_2|g^-|\rceil, \lceil\log_2(g^+ + 1)\rceil\} + 1$ to fully resolve both the positive and negative part of g. Finally, we continue applying the QPE, denoted by $\mathcal{Q}$

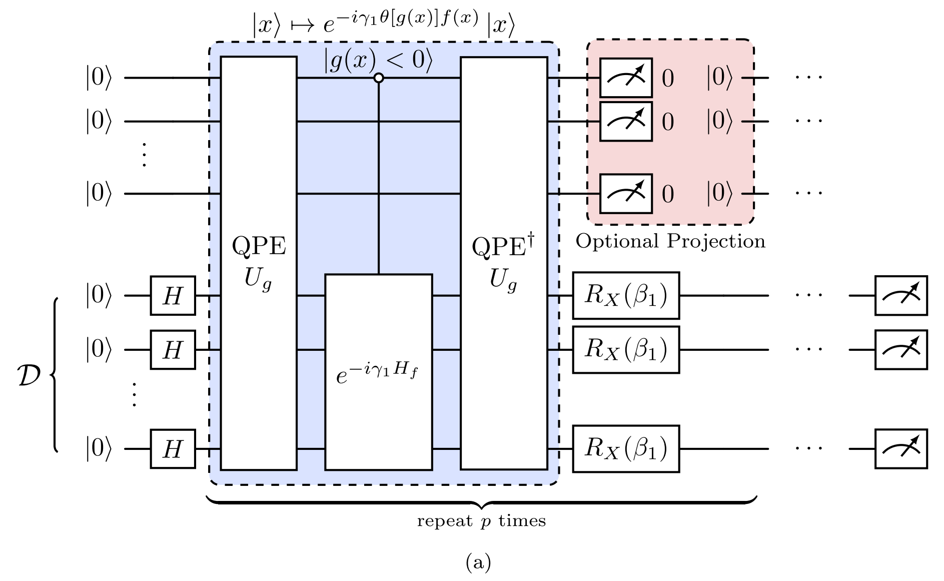
$$\mathcal{Q}_{U_g}|x\rangle|0\rangle = \mathcal{T}^\dagger\frac{1}{\sqrt{2^M}}\sum_k e^{\frac{2\pi i}{2^M}g(x)k}|x\rangle|k\rangle \tag{14}$$

$$= \frac{1}{2^M}\sum_k\sum_z e^{-\frac{2\pi i}{2^M}(z-g(x))k}|x\rangle|z\rangle \tag{15}$$

$$= \sum_z \delta_{g(x),z}|x\rangle|z\rangle = |x\rangle|g(x)\rangle, \tag{16}$$

where $\mathcal{T}$ denotes the QFT and $\delta_{g(x),z}$ denotes the Kronecker delta. Since the output register of the QPE is in two's complement [60], we define $|g(x)\rangle = |z_1\cdots z_M\rangle$, with $g(x) = 2^M(-z_1/2 + \sum_{i=2}2^{-i}z_i)$. Consequently, the sign of $g(x)$ is indicated by the first qubit z_1, i.e., $g(x) \geqslant 0 \Leftrightarrow z_1 = 0$.

Based on that state, we can apply the ordinary cost unitary inversely controlled by the indicator qubit z_1 and afterwards



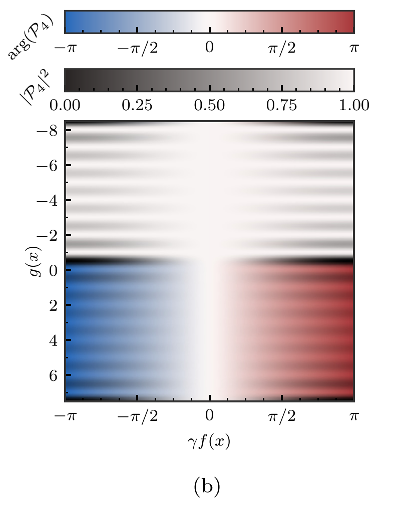


FIG. 2. (a) Shows the IF-QAOA circuit. The cost layer consists of the QPE, evaluating whether constraint g has been satisfied, and a controlled application of the cost Hamiltonian based on the indicator qubit. The projective measurement on the QPE register is only required when g is not resolvable by the QPE register. (b) IF-QAOA circuit presents the projection factor $\mathcal{P}$, defined in Eq. (21), in the approximate application of the indicator function, when g is not resolvable by the QPE register, here for $M = 4$. The shading of the plot shows the probability of successful projective measurements. Clearly, when g is an integer, this probability is always one. The hue indicates the applied phase to the compute register, showing that the application only happens for $g(x) \geqslant 0$.

uncompute the ancillary QPE register to be reused in the following QAOA iterations:

$$U_{\text{IF-}f}(\gamma) = \mathcal{Q}^\dagger_{U_g}\bar{C}_{z_1}[U_f(\gamma)]\mathcal{Q}_{U_g}, \tag{17}$$

where $\bar{C}_{z_1}$ denotes the negated control on z_1, as visualized in the circuit Fig. 2(a). Note that the circuit in Fig. 2(a) additionally features a projective measurement of the QPE register, which is needed when $g(x)$ will not be resolvable on the QPE register, e.g., when the image of g is noninteger, as detailed in Sec. IV C. In the resolvable case discussed in this section, the measurements will always be zero and can, therefore, be omitted.

1. Resource estimation

Let us compare the resources required to implement IF-QAOA against the quadratic penalty QUBO by assuming f and g are linear binary functions (knapsack problem). The number of ancillary qubits required differs: generally, $M^{\text{IF}} \geqslant M^{\text{QUBO}}$, since the IF-QAOA QPE register needs to resolve the whole image of the constraining function. In contrast, the QUBO method only requires encoding the feasible part of g. Recall,

$$M^{\text{IF}} = \left\lceil \log_2 \max\left\{\sum_i w_i - W, W\right\}\right\rceil + 1, \tag{18}$$

$$M^{\text{QUBO}} = \lceil \log_2 W\rceil. \tag{19}$$

Please refer to Appendix C for detailed resource estimations; here is a concise summary: The resources to implement the QUBO penalty approach are: $(N + M^{\text{QUBO}})$ layers with $(N + M^{\text{QUBO}})(N + M^{\text{QUBO}} - 1)/2$ controlled phase rotations plus a single layer of phase gates for the linear contributions. The IF-QAOA consists of the QPE part, requiring N layers and NM^{IF} controlled phase gates for the phase addition, followed by $2M^{\text{IF}} - 1$ QFT layers with $M^{\text{IF}}(M^{\text{IF}} - 1)/2$ two-qubit gates. The controlled cost unitary is implementable with roughly $3N$ gates on $\lceil \log_2 N\rceil$ layers (Appendix C provides an exact expression). The uncomputation then doubles the gate and layer resources of the QPE part. Altogether, both methods have an asymptotic $O(N + M)$ layer cost, but IF-QAOA only requires $O(NM + M^2)$ two-qubit gates compared to $O((N + M)^2)$ in the QUBO case. Numerical analysis on the knapsack problem in Appendix C shows that approximately three times more layers are necessary for IF-QAOA compared to QUBO.

2. Example

Consider a knapsack instance with 20 items, $W = 200$, $\sum_i w_i = 500$, thus $g^+ = 200$, $g^- = -300$. With $M^{\text{QUBO}} = 8$, $M^{\text{IF}} = 10$, QUBO requires 378 gates and 29 layers. The QPE part in IF-QAOA requires $20 + 19 = 39$ layers with $200 + 45 = 245$ two-qubit gates. Consequently, IF-QAOA totals 510 gates and 87 layers, clearly showing that IF-QAOA is more resource intensive in terms of layers. However, the gate count is still comparable.

B. Accelerated simulation

Classical simulation of QAOA can be accelerated using precomputation techniques, as explored in Refs. [61–63]. These methods are based on the observation that in ordinary state vector simulation, every simulated QAOA layer technically brute forces the objective function since the values can be read from the phase of the state vector entries. Therefore, brute forcing the objective function beforehand and reusing it in later iterations is more efficient [61–63]. Analogously, we can accelerate the simulation of IF-QAOA by implicitly applying the phase modification in Eq. (13) without technically simulating the QPE part: namely, we brute force and cache every element of the cost function $\tilde{f}(x)$ from Eq. (12) with the algorithm used in Ref. [63]. Then, we multiply $e^{-i\gamma\tilde{f}(x)}$ pointwise to the state vector of the simulator. This is followed by the gate-based application of the R_X mixer gates, similar to Refs. [61–63].

With the IF-QAOA, the simulation strategy has the advantage that we are not required to simulate the QPE register qubits, leaving us with a smaller Hilbert space and making larger problem sizes accessible to simulation. The enlarged Hilbert space was precisely why previous methods investigating similar oracle approaches omitted necessary numerical simulations [22].

Additionally, this technique allows the simulation of a penalty-based method that serves as an upper bound in performance to the slack variable-based quadratic penalty method from Eq. (8). Namely, we can simulate the $a = 2$ cost function from Eq. (9) using

$$f(x) + \lambda(1 - \Theta[g(x)])g^2(x), \tag{20}$$

subsequently called the virtual penalty approach. This cost function is precisely Eq. (8) if $s(y) = g(x)$ in the feasible part and $s(y) = 0$ in the infeasible part. Here, we again profit from the smaller Hilbert space required for simulation. This allows us to estimate how well the QUBO approach can ideally perform in larger instances, which are impossible to simulate due to the slack variables (and therefore qubits) in place.

C. Approximate indicator function

Now, we investigate the case when $g(x)$ is not resolvable by the ancillary qubit space, which may be the case if the coefficients are real-valued, i.e., $g(x) \in [g^-, g^+]$, or the number of total qubits is limited. In that case, the Kronecker-delta identity, used from Eq. (15) to Eq. (16) will not be exact but approximate. Consequently, there is a marginal probability that the cost function phase is applied even if the constraint is unsatisfied. Nevertheless, we can analytically compute that when we apply the inverse of the QPE and measure the QPE register to be zero, as shown in Fig. 2(a), only correctly evaluated feasible indicators will get a cost phase

$$\begin{aligned}\langle 0|\mathcal{Q}^\dagger_{U_g}\bar{C}_{z_1}[U_f(\gamma)]\mathcal{Q}_{U_g}|0\rangle|x\rangle\\ = \mathcal{P}_M[\gamma f(x), g(x)]|x\rangle,\end{aligned} \tag{21}$$

where $\mathcal{P}_M[\gamma f(x), g(x)]$ is a projection factor that depends on the number of ancillary variables M. It is visualized in Fig. 2(b) for $M = 4$. The projective measurement has a certain probability of failing, i.e., $|\mathcal{P}_M[\gamma f(x), g(x)]|^2 \leqslant 1$, making the operation non-unitary. Since the resolution of QPE grows with increasing ancillary qubits, the probability of a failed projective measurement can be reduced by increasing M.

We can analytically express $\mathcal{P}_M$ given by

$$\begin{aligned}&\mathcal{P}_M[\gamma f(x), g(x)]\\ &= \frac{e^{-i\gamma f(x)}+1}{2} + \frac{e^{-i\gamma f(x)}-1}{2}\vartheta_M[g(x)],\end{aligned} \tag{22}$$

where ϑ_M is the approximate sign function of $g(x)$, which can be numerically derived from a Fourier series, as detailed in Appendix A. For simulation, we compute $\vartheta_M[g(x)]$ beforehand through the brute-force result of the constraint function $g(x)$, as described in Sec. IV B, apply Eq. (22) in the optimization.

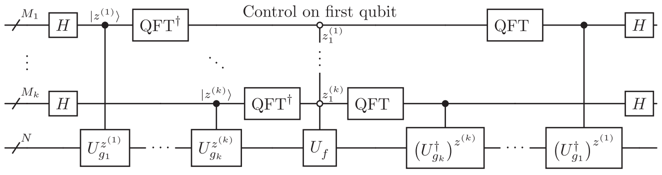


FIG. 3. Multiconstraint cost function subroutine. The first qubits of the QPE registers control the application of the cost operator.

When simulating the approximate evolution, we have to renormalize the state vector after each application of the approximate IF cost layer, denoted by the operator $\mathcal{N}$, yielding the combined layer operation

$$|\psi_i\rangle = U_M(\beta_i)\mathcal{N}\mathcal{P}(\gamma_i f, g)|\psi_{i-1}\rangle, \tag{23}$$

where $|\psi_i\rangle$ is the quantum state at the end of the ith QAOA iteration. The layer success probability is given by $q_i = \|\mathcal{P}[\gamma_i f(x), g(x)]|\psi_{i-1}\rangle\|^2$ and depends on the previous layer, γ_i, and both functions f, g. For multiple QAOA layers, this leads to an overall success probability that decays like $q_{\text{total}} = \prod_i q_i$.

D. Multiconstraint setup

So far, we have focused only on single constraints. For the general case with multiple (K) inequality constraints, we now propose the following procedure for generating corresponding IF-QAOA circuits, also depicted in Fig. 3:

(i) Ensure $f(x) \leqslant 0\,\forall x \in \mathcal{D}$. If this is not the case, subtract an upper bound from the cost function. Alternatively, assuring only the optimal solution $f(x^*) < 0$ may also suffice.

(ii) Compute upper g_k^+ and lower g_k^- bounds of all constraint functions g_k. If direct computation of the bounds is impossible, estimate the bounds using, e.g., roof duality [64] or semidefinite programming relaxations [65].

(iii) Derive the number of ancillary qubits M_k required for the QPE register of each constraint. If the number of qubits is limited or the constraining function is real valued, limit the maximal QPE register size to $M_k \leqslant M_{\max}$.

(iv) Apply the QPEs for each constraint, following Sec. IV A and IV C. The QFT part of the QPEs can be executed in parallel.

(v) Apply cost function unitary, multicontrolled by the first qubits of all QPE registers.

(vi) Uncompute QPEs and perform the projective measurements if required.

The entire QAOA circuit consists of $N + \sum_k M_k$ qubits and applies the cost function $f(x)\prod_k \Theta[g_k(x)]$. Each QPE requires the application of maximally $[\deg(g_k) + 1]$-qubit phase gates. The multicontrolled cost unitary can be most efficiently implemented through an additional ancillary qubit where the joint constraint satisfaction is computed (and uncomputed) via two $(k + 1)$-qubit Toffoli gates. Then, $[\deg(f) + 1]$-qubit phase gates can be employed to implement the conditional objective function. In total, $\sum_i[2|g_k|M_k + M_k(M_k - 1)] + |f| + 2$

non-single-qubit gates are required, where $|\cdot|$ denotes the number of terms in the respective polynomial. The number of circuit layers is problem dependent and greatly depends on the structure of f and g_k; no general statement can be made.

1. Alternative architectures

The multicontrolled gates or the number of qubits may be a bottleneck, therefore we propose two more NISQ-friendly alternatives here: first, instead of relying on the product of the step and cost function, in the no-penalty setting, we can go a step back and introduce constant penalty for every constraint violation, i.e., $f(x) + \sum_i \lambda_k(1 - \Theta[g_k(x)])$. This can be implemented by a single phase gate per constraint. Second, a sequential application of the QPEs can be employed to reduce the qubit count, allowing for the reuse of the QPE register. The no-penalty setting requires storing the indicator information in an additional qubit per constraint, requiring $k-1$ additional qubits. In the penalty setting, however, the sequential application is straightforward.

2. Example

To illustrate the pipeline, we sketch the procedure for a multi-K-knapsack problem here, defined as

$$\arg\min_x -\sum_{i=1}^{N}\sum_{k=1}^{K} v_i x_{i,k} \quad \text{s.t.} \sum_i w_{i,k}x_{i,k} \leqslant W_k \,\forall k$$
$$\sum_k x_{i,k} \leqslant 1\,\forall i.$$

Since the second type of constraint is a special set-packing constraint that can be enforced through either Grover mixers [57] or a special penalty without slack variables [44], we only focus on the first knapsack constraint here.

The knapsack objective naturally satisfies the first condition in the defined workflow. The second step can be easily evaluated, giving $g_k^- = W_k - \sum_i w_{i,k}$ and $g_k^+ = W_k$. Then the M's can be evaluated, depending on the preference of $M_{\max}$, and the circuit can be constructed. Notably, the cost function is separable per knapsack constraint, allowing us to implement the following effective cost:

$$\tilde{f}(x) = -\sum_k \Theta\left[W_j - \sum_j x_{i,j}w_{i,j}\right]\sum_i v_i x_{i,j},$$

where each knapsack is taken care of individually.

V. NUMERICAL SIMULATION

This section presents the numerical simulation experiments conducted to evaluate the performance of IF-QAOA. In our evaluation, we only focus on the single-constraint knapsack problem instances and rely on ideal statevector simulation with exact (shot-free) evaluation of all quantities, like the expectation value. All experiments have been conducted on an AMD Ryzen Threadripper PRO 5965WX.

A. Experimental setup

Our evaluation compares IF-QAOA against the quadratic penalty approaches as the baseline. We consider both the QUBO method with slack variables (slack penalty) and the virtual penalty from Eq. (20) in our experiments. The virtual penalty method serves as an upper performance bound to the QUBO method, helping to access problem instances where the extended Hilbert space is no longer simulatable.

1. Problem instances

For the problem sizes $N \in \{6, 8, \ldots, 22\}$, we generate two knapsack data sets, one with real-valued parameters and one with integer-valued items, both containing 128 instances for each item count. These bias-free instances help to gain insight into the scaling behavior of the algorithms. Starting with the real-valued data set, we sample w_i and $v_i \sim \mathcal{U}(0,1)$ from the uniform distribution. Afterwards, the capacity is sampled from $W \sim \mathcal{U}(0.2, 0.8)\sum_i w_i$ to guarantee a binding constraint.

The integer-valued instances are derived from the real-valued ones and use the fixed capacity $10N$. All weights and profits are scaled and rounded to the next integer $w_i \leftarrow \text{round}(10 \times w_i N/W)$, and finally, the capacity is set to $W \leftarrow 10N$.

Additionally, we normalize the objective function of each approach such that $\max_x \langle x|H_M|x\rangle - \min_x \langle x|H_M|x\rangle = 2N \overset{!}{=} \max_x \tilde{f}(x) - \min_x \tilde{f}(x)$, leading to a more even cost landscape in $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ direction [29], thereby enhancing optimizer convergence.

2. Penalty coefficient

For the penalty methods from Eq. (8) and Eq. (20), we must find a penalty coefficient for each problem instance individually. Even though setting $\lambda > \max_i v_i$ is a valid assignment that allows for no infeasible states to have lower energy than the best feasible one [15], its magnitude may overshadow the optimization objective, potentially leading to suboptimal optimization performance [66]. To ensure the most optimal performance of the penalty-based approaches in our comparison, we set the penalty such that the lowest-energy infeasible state has the same objective value as the second-to-lowest feasible state, i.e.,

$$\min_{x\notin\mathcal{F}} f(x) + \lambda(W - x^T w)^2 \overset{!}{=} \min_{x\in\mathcal{F}\setminus\{x^*\}} f(x). \tag{24}$$

Our proposed parameter setting is empirically validated in Appendix B. This approach reduces the magnitude by an average factor of 20 compared to $\max_i v_i$. As we brute force prior to QAOA simulation, determining λ with our method incurs no additional computational overhead. However, it should be noted that this technique cannot be applied outside the simulation context, requiring reverting to larger, suboptimal penalty factors.

3. Optimization

The optimization of the QAOA parameters is done with the L-BFGS algorithm [67], which has proven to be very efficient in the context of QAOA [68]. The gradients are evaluated exactly based on the adjoint differentiation method, similar to how gradients are computed in the accelerated QAOA framework of Ref. [63]. We optimize the parameters for each QAOA layer count p sequentially, starting with $p = 1$ and the initial parameter of $\beta_1 = \gamma_1 = 0.1$. After obtaining the optimal parameters, we use linear interpolation for both parameters $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$ to the next p as proposed in Ref. [68]. Repeating this method, we iteratively optimize for $p \in \{1, 2, 3, 4, 6, 8, 12, 16, 24, 32, 48, 64\}$, with maximally 100 L-BFGS iterations.

Whether IF-QAOA or penalty-based, all methods use the same indicator cost function defined in Eq. (12) for training and solution quality evaluation, similar to Ref. [44].

4. Metrics

In line with best practices on benchmarking quantum optimization [69], we will focus on two leading figures of merit to compare the performance of the algorithms. First, the solution quality is measured by computing the random-adjusted approximation ratio (RAAR), defined as follows:

$$\text{RAAR} = \frac{\langle \tilde{f} \rangle - \langle \psi | H_{\tilde{f}} | \psi \rangle}{\langle \tilde{f} \rangle - f^*}, \qquad (25)$$

where $\langle \tilde{f} \rangle$ is the uniform average of the indicator cost (corresponding to random sampling) and $f^* = \min_{x \in \mathcal{F}} f(x)$ is the optimal solution. Thereby, RAAR = 0 is as good as random sampling, while RAAR = 1 indicates always sampling a perfect solution.

The second metric, time to solution (TTS), is derived from the probability of sampling an optimal solution $P^* = |\langle x^* | \psi \rangle|^2$. We estimate the runtime as the number of shots required until measuring the optimal solution once with 99% certainty, scaled by the runtime of a single circuit execution. Formally, TTS is defined as follows [69,70]

$$\mathcal{T}_p = \mathbf{L}(p) \left\lceil \frac{\log 0.01}{\log(1 - P^*)} \right\rceil, \qquad (26)$$

where $\mathbf{L}(p)$ is the number of circuit layer operations (CLO) of the depth-p QAOA circuit, which is proportional to the circuit execution time. Please refer to Appendix C for the precise derivation of $\mathbf{L}(p)$ for both the IF-QAOA and QUBO penalty methods as well as assumptions on gate set and connectivity. The virtual penalty method receives the CLO of the slack penalty QAOA. Additionally, we also define the optimal TTS,

$$\mathcal{T}^* = \min_p \mathcal{T}_p \qquad (27)$$

as the minimum overall CLO required out of the various depth-p experiments.

B. Integer knapsack instances

We begin the analysis with the fully resolvable integer knapsack instances results, comparing IF-QAOA against the quadratic penalty approaches.

Figure 4 shows slices of the results for both problem sizes $N = 16$ and QAOA layer repetitions $p = 16$. It is unambiguously apparent that IF-QAOA produces the best solutions from the compared methods. Focusing on the top panel, showing the results for fixed QAOA repetitions $p = 16$ with respect

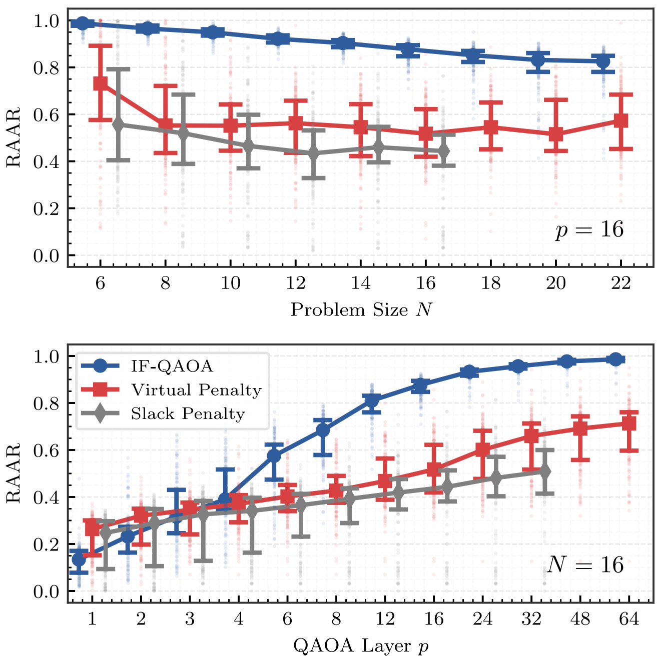


FIG. 4. The solution quality measured as RAAR displayed for a selection of results. The top plot shows RAAR depending on problem size N for fixed QAOA layer depth $p = 16$, and the bottom plot shows RAAR w.r.t. p at $N = 16$. Error bars indicate the 50% percentile interval.

to (w.r.t.) problem size, we can observe that IF-QAOA remains at a median RAAR > 0.8 for all problem sizes. In contrast, the penalty-based methods remain in RAAR ≈ 0.4–0.6. Furthermore, the slack penalty method performs slightly worse (but is similar overall) to the virtual penalty approach, justifying the use of the virtual penalty method to estimate the QUBO method behavior in larger instances. Besides the median values, it is also evident that IF-QAOA is less dependent on the problem instance itself, which is visible by the smaller interquartile ranges (IQRs) marked by the error bars.

When focusing on the RAAR w.r.t. the circuit layer depth, it is noteworthy that for low-depth QAOA, the penalty-based methods are outperforming IF-QAOA. We can see improved performance of IF-QAOA starting at depth $p \geqslant 3$. This could be explained by the heavily penalized infeasible states in the penalty methods, which focus the optimization on finding only feasible states, thereby increasing solution quality fast initially but slowing down once high feasibility has been achieved. IF-QAOA directly emphasizes the better solution candidates, reflected in the improved solution quality at higher ps. While RAAR continuously improves with increasing p for all methods, IF-QAOA is the only method that approaches perfect solution quality for almost all $N = 16$ instances within $p \leqslant 64$.

Figure 5(a) shows the median TTS for IF-QAOA and virtual penalty across all problem sizes and QAOA depths. Similar to the RAAR results in Fig. 4, IF-QAOA initially exhibits higher TTS than the virtual penalty counterparts. Yet, starting again with $p \geqslant 3$, IF-QAOA TTS monotonically decreases and drops below the penalty-based TTS. In contrast, the virtual penalty method shows TTS reduction only for

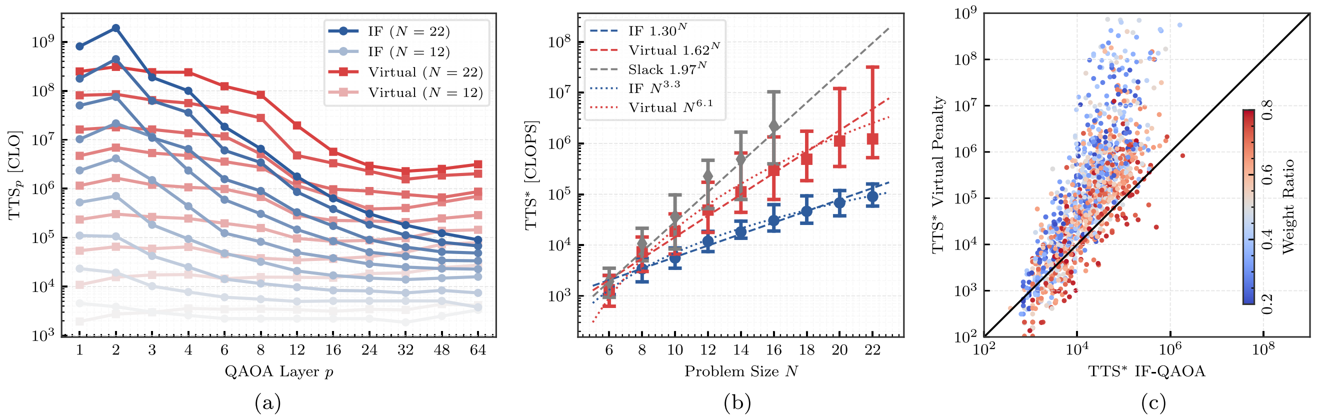


FIG. 5. (a) shows the median TTS for all problem instances in various color shades with increasing QAOA layer count p. As apparent, IF-QAOA starts with higher TTS than the baseline approach, but starts improving upon it starting with $p \geqslant 3$. (b) shows the optimal TTS* depending on the problem size. The dashed line shows the logarithmic regression, with the fitted bases displayed in the legend. (c) shows a scatter plot of all $9 \times 128 = 1152$ problem instances comparing TTS* of IF-QAOA against the virtual penalty. In 82% of the test cases, IF-QAOA arrives at a solution faster. The color shading of the scatter plots indicates the weight ratio $W/\sum_i w_i$ of the knapsack instance.

$N > 10$, but this improvement reverses after $p \approx 16$, with the exact threshold depending on the problem size. This reversal means the increased P^* fails to compensate for the longer circuit depth, leading to worse TTS runtime overall.

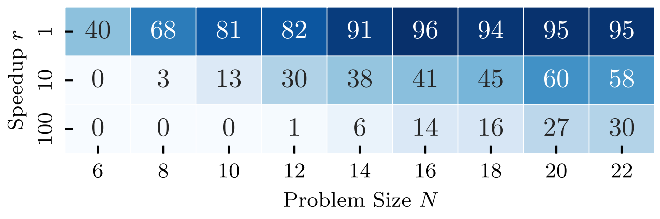
The optimal TTS* concerning problem size N is displayed for all approaches in Fig. 5(b), demonstrating the advantageous performance of IF-QAOA. When fitting an exponential to the TTS* data, IF-QAOA exhibits the most favorable scaling with a base of 1.30 compared to 1.63 of the virtual penalty. The statistical reliability of these fits is supported by the significantly lower variability in IF-QAOA results, which span only half an order of magnitude, versus nearly two orders of magnitude for the virtual penalty method. Interestingly, comparing the fit to the data suggests that both the virtual penalty and IF-QAOA may follow subexponential scaling. Further analysis with power-law models reveals that $\mathcal{T}^* \propto N^{3.3}$ for IF-QAOA versus $\mathcal{T}^* \propto N^{6.1}$ for the virtual penalty. However, additional experiments on larger problem instances and more carefully selected datasets are necessary to establish confident subexponential scaling claims.

Next, Fig. 5(c), shows a scatterplot of the TTS* from IF-QAOA against the virtual penalty method, with every dot located above the diagonal line meaning IF-QAOA solves the particular instance faster (TTS* is lower). Overall, this is the case for 82% of the instances. The coloring of the data points refers to the weight ratio of that specific instance, defined as $W/\sum_i w_i$. A tendency for cases with a lower weight ratio to be solved faster with IF-QAOA is qualitatively visible. This can be explained by the square of the penalty violation, i.e., $(W - \sum_i w_i)^2$, which can grow much higher for small weight ratio instances. This significantly skews the energy spectrum of the objective function, meaning that finding the optimal solution is less important than finding solutions that only minimize this penalty. IF-QAOA does not change the energy spectrum, making it easier for QAOA to amplify the optimal solution amplitude (which also explains the reduced variance between problem instances). The argument for the energy spectrum can also be made to explain the difference between the slack penalty and virtual penalty approaches. While the virtual penalty only applies correct penalties, the slack penalty introduces additional undesirable energy states. These problematic states arise when the slack variable assignment is inconsistent with the computational register assignment, creating considerable energy contributions that distort the optimization landscape even more.

Figure 6 gives a more detailed overview of how many instances are solved faster with IF-QAOA compared to the virtual penalty method. Here, we define a speed up factor r and count all instances where $r\mathcal{T}^*_{\text{IF-QAOA}} < \mathcal{T}^*_{\text{Virtual}}$. The results demonstrate that IF-QAOA outperforms the virtual penalty approach for nearly all (>90%) instances with $N \geqslant 14$. Remarkably, approximately one-third of these larger instances exhibit speedups exceeding two orders of magnitude (100×), highlighting the better scaling behavior of IF-QAOA.

C. Comparison with quantum tree generator mixer

To benchmark IF-QAOA against leading constrained preserving architectures, we selected the QTG-based AAM-QAOA (QTG-QAOA) as the most promising architecture for knapsack problems [24]. The authors showed a clear performance advantage of QTG-QAOA in their experiments, outperforming the copula mixer approach from Ref. [47]. We build our comparison on their results, featuring only depth $p = 1$ QTG-QAOA on hard knapsack instances generated by Ref. [71].



Speedup r \ Problem Size N	6	8	10	12	14	16	18	20	22
1	40	68	81	82	91	96	94	95	95
10	0	3	13	30	38	41	45	60	58
100	0	0	0	1	6	14	16	27	30

FIG. 6. Share of instances (in %) that IF-QAOA solves with speed up r compared to the virtual penalty approach, i.e., where $r\mathcal{T}^*_{\text{IF-QAOA}} < \mathcal{T}^*_{\text{Virtual}}$.

TABLE I. Number of IF-QAOA repetitions until CLO of QTG-QAOA at $p = 1$ is matched.

N	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
p	15	20	22	28	28	40	41	45	52	46	50	61	57	70	64	64	69	69

Solving the same knapsack instances [24], we set the number of IF-QAOA layers such that the total IF-QAOA circuit requires just as many CLO as $p = 1$ QTG-QAOA, i.e.,

$$\mathbf{L}_{\text{IF}}(p) \leqslant \mathbf{L}_{\text{QTG}}(1). \tag{28}$$

This way, we can ensure that a better solution quality of IF-QAOA also means better performance. The resulting p values are summarized in Table I, showing an increasing number of repetitions possible for IF-QAOA due to the more streamlined circuit architecture.

Since we instantly aim to optimize for large p values (Table I), we skip the previous interpolation scheme. Instead, we first optimize a linear schedule similar to Ref. [28]

$$\gamma_i = \Delta_\gamma \frac{i + 1/2}{p} \qquad \beta_i = \Delta_\beta \frac{p - i - 1/2}{p}, \tag{29}$$

where $\Delta_{\gamma,\beta}$ are the two parameters to be tuned. Fig. 7 shows the RAAR after 100 L-BFGS iterations, which is clearly above the RAAR found by the QTG approach. When fine tuning each of the β_i, γ_i starting with the optimized linear schedule, the RAAR can be improved even more. Yet, since we only simulate until $N = 22$, we cannot predict whether this pattern continues or QTG provides advantageous results for larger instances. Christiansen *et al.* [24] were able to simulate the larger problems since QTG restricts state vector simulation to the feasible subspace. These results indicate that simpler circuit architecture in the context of QAOA can lead to better results overall, since more repetitions of a single QAOA layer can be employed to find the optimal solution directly. However, there is a break-even point to be expected since the QTG-based approach only searches in the space of feasible assignments, while the IF-QAOA needs to find feasible results through optimization.

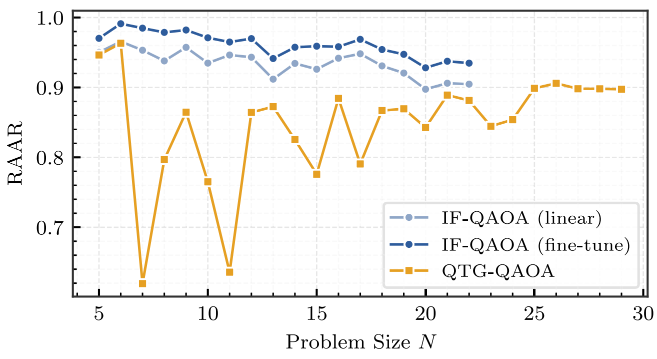


FIG. 7. RAAR of IF-QAOA (for p, see Table I) compared to QTG-QAOA at $p = 1$. IF-QAOA is separated between two results: linear refers to optimizing just the rates of the linear schedule. Fine tune refers to running 100 L-BFGS iterations on all parameters, starting with the optimized linear schedule. Results for QTG-QAOA and problem instances are taken from Ref. [24].

D. Approximated IF results

As the next and final analysis of the numerical results, we focus on the noninteger knapsack instances and approximative IF application. Since the approximate IF is not sharp like the ideal step function, see Fig. 2(b), we introduce a small offset of the constraining function, shifting the transition region more to the center, not occurring between $g(x) \in [-1, 0]$, but around $g(x) \approx 0$. Namely, we implement $\Theta[g(x) - \epsilon]$, where $\epsilon \in [0, 1]$. Running a linear search on a subset of the instances revealed that $\epsilon = 0.5$ empirically works best. The effect of the offset decreases with increasing resolution in the QPE register.

Since the projective measurement only has a certain probability of succeeding per QAOA repetition, here given as q_i, we expect the overall success probability $q_{\text{total}} = \prod_i q_i$ to deteriorate with an increasing number of repetitions and approximation degree. This can precisely be observed in Fig. 8(a), showing q_{total} w.r.t. p. The higher the number of ancillary variables is in the QPE register for IF computation, the lower the approximation degree of the sign function becomes and, therefore, also the lower any probability of failing.

Since we now have failure probability during circuit execution, TTS from Eq. (26) requires redefinition. Whenever the projective measurement fails, we can instantly start over, leaving us with an expected circuit layer depth of

$$\begin{aligned}\langle \mathbf{L} \rangle(q, p) &= \mathbf{L}_{\text{init}} + \mathbf{L}_{\text{layer}}(1 + q_1 + q_1 q_2 + q_1 q_2 q_3 + \cdots) \\ &= \mathbf{L}_{\text{init}} + \mathbf{L}_{\text{layer}} \left(1 + \sum_{i=1}^{p-1} \prod_{j=1}^{i} q_i \right).\end{aligned} \tag{30}$$

Together with the combined probability of successful indicator applications and finding the optimal solution $P^* q_{\text{total}}$, we can define the TTS for the approximative IF-QAOA as follows:

$$\mathcal{T}_p = \langle \mathbf{L} \rangle(q, p) \left\lceil \frac{\log 0.01}{\log(1 - P^* q_{\text{total}})} \right\rceil, \tag{31}$$

which is shown in Fig. 8(b) for problem size $N = 16$. Here, we can see that increased probability decay throughout the computation impacts TTS significantly with small QPE registers. For instance, $M = 4$ exhibits slower TTS for $p = 64$ compared to $p = 1$. The detrimental effect of projective measurements on TTS diminishes steadily as M increases. At $M \geqslant 12$, the performance impairment becomes negligible. Consequently, as M increases, TTS* becomes smaller and the optimal depth p shifts to the right.

The same can be observed when considering the scaling behavior of TTS*, as shown in Fig. 8(c). Higher resolution in the QPE register leads to more favorable scaling. The bases for the exponential fit decrease with increasing M from 1.54 at $M = 4$ to 1.26 at $M = 16$. Notably, even at $M = 4$, the TTS* scaling outperforms the penalty approaches discussed in Sec. V B. Also, scaling improves upon the exact integer cases, which can be explained by the nongrowing complexity of the oracle circuit due to fixed M. Furthermore, a subexponential behavior can also be observed here. A power-law fit reveals $\mathcal{T}^* \propto N^{2.9}$ at $M = 16$.

At $M = 12$ and $M = 16$, the exponential bases are very similar (1.28 and 1.26), affirming our prior statement that a resolution of $M \approx 12$ suffices for our benchmark instances. It

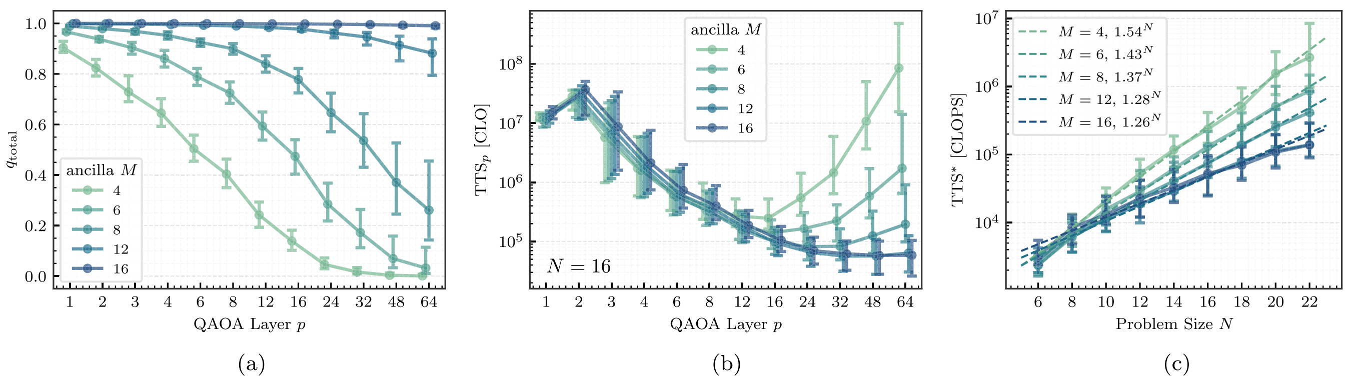


FIG. 8. The success probability q_{total} for different M is depicted in (a). (b) shows the TTS w.r.t. the QAOA repetitions for problem size $N = 16$. Clearly, fewer ancillary variables lead to a sudden increase in TTS after $p \approx 16$ due to vanishing success probability. TTS is relatively similar for shorter QAOA circuits. TTS* scaling is shown in (c), with more ancillary variables leading to more favorable scaling. However, we also see that the scaling saturates and is very comparable for $M \geqslant 12$.

is likely that when increasing N to sizes beyond our benchmark instances, M has to be further increased. To solve larger instances, p must grow beyond the tested depths, which subsequently causes the q_{total} to decay, ultimately requiring better resolution of the constraining function through more ancillary qubits.

VI. DISCUSSION AND CONCLUSION

We have provided a formalized description of the IF-QAOA for general inequality constraint encoding of COPs using an oracle circuit consisting of two QPEs. This circuit allows us to compute constraint satisfaction onto a single indicator qubit, which can subsequently be used to apply the cost function in a controlled fashion, not requiring penalty coefficients. We provided accelerated simulation techniques for both the exact and the approximate cases, where the image of the constraining function is not fully resolvable in the QPE register. Finally, we evaluated the performance of IF-QAOA compared to the default QUBO method, i.e., adding a slack variable and a quadratic penalty to the objective, using randomly generated knapsack instances.

We found that IF-QAOA produces more favorable results with a better approximation ratio and faster optimal TTS. The improved performance offsets the approximately three times higher circuit complexity in terms of CLO, through higher probability of sampling the optimal solution. Overall, in 82% of the tested instances, IF-QAOA arrived at the solution faster, despite slightly higher circuit layer complexity than the QUBO implementation due to oracle construction. Also, we found that the empirical scaling behavior of IF-QAOA has a lower exponential base (1.30) than the quadratic virtual penalty method (1.61). Investigating the instance properties, it was apparent that the weight ratio significantly impacted deciding which method solves a problem faster. Small weight ratios result in large penalties that significantly distort the energy spectrum. This distortion causes the quadratic penalty method to prioritize constraint satisfaction over finding the optimal solution within the feasible space. In contrast, IF-QAOA preserves the original energy spectrum and focuses on better-quality solutions.

Additionally, we compared IF-QAOA against the state-of-the-art AAM-QAOA based on QTG for knapsack [24,38]. Since the QTG-QAOA overhead allows for multiple IF-QAOA iterations to match circuit complexity, we found that IF-QAOA produces better results regarding RAAR when using a similar number of CLO. While our comparison with QTG-QAOA showed promising results for IF-QAOA at smaller problem sizes, we acknowledge the limited scope of this comparison. The generalizability of IF-QAOA's performance advantage remains an open question for larger problem instances, where QTG's restricted search space approach may ultimately prove more efficient. Future work should include more extensive benchmarking against ($p > 1$) QTG-QAOA across a broader range of problem sizes to establish more precise performance boundaries between these approaches. Nevertheless, it is worth noting that IF-QAOA is generally applicable compared to the knapsack-focused QTG.

Simulations of IF-QAOA in the approximate regime have proven that a small number of ancillary qubits suffice to find high-quality solutions even though each layer has some probability of failure. For $M \geqslant 12$, the resolution of the approximate sign function was satisfactory for the considered benchmarking instances. Increasing M did not improve performance anymore.

We can confidently conclude from our results that IF-QAOA provides a generally applicable framework with promising performance results for integrating inequality constraints into QAOA and solving general forms of COPs. However, this claim sustains itself solely on simulations of the knapsack problem, the simplest constrained optimization problem. An evaluation on a broader range of problem classes is required to reinforce this statement. The main limitation of IF-QAOA is that it is an entirely cost-function-based method, meaning that QAOA optimization is necessary to find feasible solutions. This can become detrimental when the ratio of feasible solutions compared to the search space diminishes. Another technical drawback is that the entire image of the constraining function needs to be resolved in the nonapproximative case, compared to only the feasible range of the QUBO method, leading to a potential overhead in qubit resources. Furthermore, the practical implementation of IF-QAOA on near-term quantum hardware remains an open challenge, as the oracle construction requires deeper circuits with high-fidelity controlled operations that may be beyond the capabilities of these devices.

For future work, IF-QAOA has to be tested in a multiconstraint setting with the multi-knapsack formulation already given in Sec. IV. Furthermore, IF-QAOA's flexibility can be taken advantage of by combining it with different constraint-preserving techniques, e.g., XY mixers [51] or Grover mixers [57]. For instance, the second constraint in the multi-knapsack problem can be enforced using Grover mixers. Experiments on hardware need to prove that the higher solution quality can remain in the presence of noise. The slightly higher circuit layer complexity can be obstructive in this case. Nevertheless, competitive results can still be expected since the gate count scales similarly to QUBO. Furthermore, different circuits can entirely replace the QPE when implementing the adder in the IF circuit. For instance, one could imagine quantum signal processing (QSP) [72] to construct the step function. Additionally, quantum singular value transform (QSVT) [73] constructions could implement more complex or specialized constraining functions in the ancillary register, e.g., constraints based on the solution of linear systems of equations, similar to simulation-based optimization [74]. Besides these suggested algorithmic improvements, hardware experiments have to be carried out to properly judge the performance of IF-QAOA on NISQ hardware.

ACKNOWLEDGMENTS

This work was supported by the Bundesministerium für Bildung und Forschung under the funding program "Förderprogramm Quantentechnologien–von den Grundlagen zum Markt" (funding program quantum technologies — from basic research to market), project QuCUN, 13N16199. J.S. acknowledges support from the Bundesministerium für Wirtschaft und Klimaschutz through the funding program "Quantum Computing–Applications for the industry" based on the allowance "Development of digital technologies" (Contract No. 01MQ22008A).

DATA AVAILABILITY

The data that support the findings of this article are openly available [75].

APPENDIX A: DERIVATION OF PROJECTION FACTOR IN APPROXIMATE IF-QAOA

To derive $\mathcal{P}_M$ and ϑ_M in Eq. (22), we begin with the expression in Eq. (15), where the QPE has already been applied:

$$\frac{1}{2^M}\sum_{z=0}\sum_{k=0} e^{-\frac{2\pi i}{2^M}(z-g(x))k}|x\rangle|z\rangle = |\psi(x)\rangle. \tag{A1}$$

Then we apply the controlled evolution of the cost Hamiltonian, which splits the term into two sections: one where $z_1 = 0$, the constraint is satisfied, and the cost function is applied, and one where $z_1 = 1$, the constraint is violated, and no phase is altered

$$\begin{aligned}|\psi'(x)\rangle &= \bar{C}_{z_1}(U_f)|\psi(x)\rangle \\ &= \frac{1}{2^M}\sum_{z=0}\sum_{k=0} e^{-\frac{2\pi i}{2^M}[z-g(x)]k} e^{-i\gamma f(x)(1-z_1)}|x\rangle|z\rangle. \end{aligned}\tag{A2}$$

As defined in Eq. (21), the projection factor $\mathcal{P}_M$ it is given by the overlap of $|\psi\rangle$ and $|\psi'\rangle$:

$$\mathcal{P}_M[\gamma f(x), g(x)] = \langle\psi(x)|\psi'(x)\rangle \tag{A3}$$

$$\begin{aligned} &= \frac{1}{2^{2M}}\sum_{k,k'} e^{\frac{2\pi i}{2^M}g(x)(k-k')}\sum_{z} e^{-\frac{2\pi i}{2^M}z(k-k')}e^{-i\gamma f(x)(1-z_1)} \\ &= \frac{1}{2^M}\sum_{k,k'} e^{\frac{2\pi i}{2^M}g(x)(k-k')}\phi(k-k'), \end{aligned}\tag{A4}$$

with helper function $\phi(k)$ defined for $-2^M < k < 2^M$. This function can be further simplified, leading us to

$$\phi(k) = \frac{1}{2^M}\sum_{z=0}^{2^M-1} e^{-i\gamma f(x)(1-z_1)}e^{-\frac{2\pi i}{2^M}zk} \tag{A5}$$

$$= \frac{1}{2^M}\sum_{z=0}^{2^{M-1}-1}\left(e^{-i\gamma f(x)} + e^{\frac{2\pi i}{2}k}\right)\left(e^{-\frac{2\pi i}{2^M}k}\right)^z \tag{A6}$$

$$= \frac{e^{-i\gamma f(x)} + (-1)^k}{2^M}\,\frac{1-e^{-\pi i k}}{1-e^{-\frac{2\pi i}{2^M}k}} \tag{A7}$$

$$= \begin{cases} \frac{1}{2}\left(e^{-i\gamma f(x)}+1\right) & \text{if } k=0 \\ \frac{1}{2^M}\frac{2}{1-e^{-\frac{2\pi i}{2^M}}}\left(e^{-i\gamma f(x)}-1\right) & \text{if } k \text{ odd} \\ 0 & \text{else,} \end{cases} \tag{A8}$$

using the geometric sum formula between Eq. (A6) and Eq. (A7). Since the summand in Eq. (A4) only depends on $k-k'$, we can substitute $k-k' \to k$ together with a weighting factor $(2^M - |k|)$, leading to

$$\begin{aligned} \mathcal{P}_M[\gamma f(x), g(x)] &= \phi(0) - \sum_{k=0}^{2^{M-1}-1}\left(1-\frac{2k+1}{2^M}\right) \\ &\quad\times\left[e^{\frac{2\pi i}{2^M}g(x)(2k+1)}\phi(2k+1)\right. \\ &\quad\left.+\,e^{-\frac{2\pi i}{2^M}g(x)(2k+1)}\phi(-2k-1)\right] \\ &= \frac{e^{-i\gamma f(x)}+1}{2} + \frac{e^{-i\gamma f(x)}-1}{2}\vartheta_M[g(x)], \end{aligned}\tag{A9}$$

with

$$\vartheta_M[g(x)] = \frac{2}{2^{M-2}}\mathrm{Re}\sum_{k=0}^{2^{M-1}-1}\left(1-\frac{2k+1}{2^M}\right)\frac{e^{\frac{2\pi i}{2^M}g(x)(2k+1)}}{1-e^{-\frac{2\pi i}{2^M}(2k+1)}}. \tag{A10}$$

The approximate sign function $\vartheta_M(\ell) = \mathrm{sign}(\ell)$ if $\ell \in \{-2^{M-1},\dots,2^{M-1}-1\}$. Since any $g(x) \in [g^-, g^+]$ can be expressed as $\ell + \delta$ with $\delta \in [0,1)$, we can identify (A10) with an inverse Fourier transform to compute unit spaced values offset by δ simultaneously, i.e.,

$$\vartheta_M(\ell+\delta) = 2\mathrm{Re}\,\mathcal{F}^{-1}_{k\to\ell}\left[(1-k/2^M)\tilde{\phi}(k)e^{\frac{2\pi i}{2^M}\delta k}\right], \tag{A11}$$

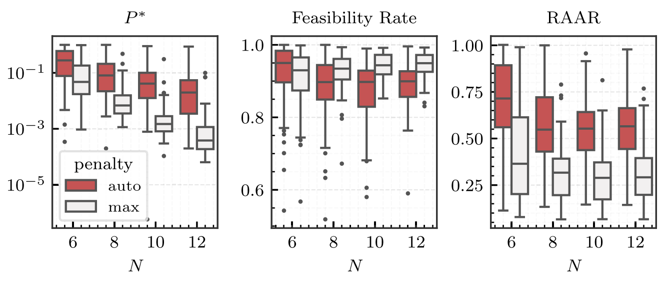


FIG. 9. Probability of sampling the optimal solution, feasibility rate $\sum_{x\in\mathcal{F}} |\langle x|\psi\rangle|^2$, and RAAR at $p = 16$ for different problem sizes w.r.t. penalty settings, where auto refers to Eq. (24).

with $\tilde{\phi}(0) = 0$ and $\tilde{\phi}(k) = \phi(k)$, otherwise. This allows fast and accurate numeric derivation of multiple $\vartheta_\ell[g(x)]$ terms since we can precompute various offset factors with fast Fourier transform (FFT) beforehand and interpolate them afterwards. From the $\mathcal{P}$ plot in Fig. 2(b), it is apparent that the oscillations of the ϑ are of length 1, therefore, $2^m = 4$ supersamples ($\delta \in \{0, 1/4, 1/2, 3/4\}$) already allow for accurate interpolation of any $g(x)$. With higher m, the numerical error decreases at the cost of more FFT calls. In total, the computational complexity for evaluating all support points is given by $O(M2^{M+m})$ with m being generally small. Throughout this work, we use a supersampling factor of $m = 3$.

Note that the geometric sum formula can also already be computed in Eq. (A1), which leads to a more manageable expression right away. However, this leads to an expression of $\mathcal{P}$ that does not allow for the Fourier transform trick, giving us inferior runtime complexity of $O(2^{2M+m})$ when interpolating, or $O(2^{M+N})$, when evaluating $\vartheta[g(x)]$ directly from the brute-forced $g(x)$ values.

APPENDIX B: EMPIRICAL PENALTY FACTOR VERFICATION

Figure 9 shows the performance of the virtual penalty approach with the penalty setting as explained in Eq. (24) (auto) and penalty set to $\lambda = \max_i v_i$ (max).

The probability of sampling the optimal solution P^* (left plot), which later determines TTS, is significantly higher with the autopenalty method, clearly showing that our approach, explained in Sec. V, is more favourable than the max approach known from literature.

However, as the penalty is set to a smaller value than $\max_i v_i$, infeasible states may have a smaller objective value than some feasible states, which means that a smaller fraction of feasible states will be measured overall. This is verified by the data in the center plot, where the max method exhibits a higher feasibility rate.

Conversely, due to the less overshadowed objective function, the auto method emphasizes on better solutions, resulting in overall better solution quality in terms of RAAR (right plot).

APPENDIX C: RESOURCE ESTIMATION

To reliably compute the circuit resources regarding circuit layer operations (CLO) and two-qubit gate counts, we must

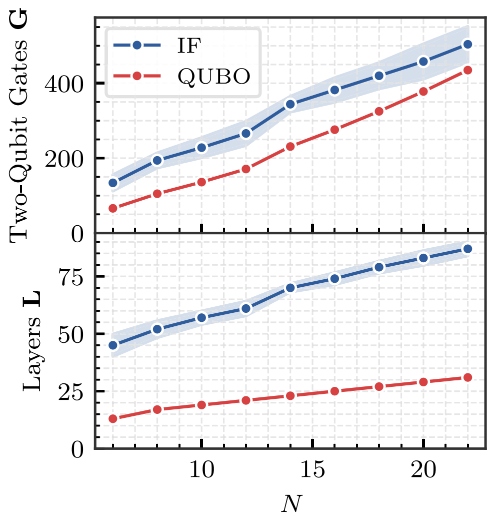


FIG. 10. Circuit resources of IF-QAOA and QUBO-QAOA for the considered knapsack instances for problem size N.

first define what gates are considered a single circuit layer operation. Like Ref. [38], we define single-controlled Pauli rotations and Phase gates, as well as cnot gates, as single-layer operations. Likewise, we also assume all-to-all connectivity, meaning that we can place gates between arbitrary qubits as long as one of the qubits is not yet used in that layer. The general QAOA circuit CLO ($\mathbf{L}$) and two-qubit gate resources ($\mathbf{G}$) can be expressed as follows:

$$\mathbf{L}_{\mathrm{QAOA}}(p) = \mathbf{L}_{\mathrm{init}} + p(\mathbf{L}_{\mathrm{cost}} + \mathbf{L}_{\mathrm{mixer}}) \qquad \text{(C1)}$$

$$\mathbf{G}_{\mathrm{QAOA}}(p) = \mathbf{G}_{\mathrm{init}} + p(\mathbf{G}_{\mathrm{cost}} + \mathbf{G}_{\mathrm{mixer}}), \qquad \text{(C2)}$$

where $\mathbf{L}_{\mathrm{mixer}} = \mathbf{L}_{\mathrm{init}} = 1$ in both the QUBO and IF case. Also, $\mathbf{G}_{\mathrm{mixer}} = \mathbf{G}_{\mathrm{init}} = 0$ for both cases.

That leaves us with finding the resources to implement the cost layer. Due to the fully connectedness of the QUBO with slack variable from Eq. (8),

$$\mathbf{G}^{\mathrm{QUBO}}_{\mathrm{cost}} = (N + M)(N + M - 1)/2 \qquad \text{(C3)}$$

$$\mathbf{L}^{\mathrm{QUBO}}_{\mathrm{cost}} = N + M\ (-1), \qquad \text{(C4)}$$

where 1 is subtracted if $N + M$ is even, given by the edge colorability of a complete graph [76].

In IF-QAOA, the cost layer is more complex to evaluate

$$\mathbf{L}^{\mathrm{IF}}_{\mathrm{cost}} = 2\mathbf{L}_{\mathrm{add}} + 2\mathbf{L}_{\mathrm{QFT}} + \mathbf{L}_{\mathrm{phase}}, \qquad \text{(C5)}$$

where $\mathbf{L}_{\mathrm{add}}$ is the CLO for adding the weight contributions in the phase of the QPE register. The QFT resource is given by $\mathbf{L}_{\mathrm{QFT}} = 2M - 1$ through parallel execution [38]. The gate resources are computed similarly, giving $\mathbf{G}_{\mathrm{QFT}} = M(M - 1)/2$.

The addition gates can be seen as an edge coloring problem of the graph connecting all compute qubits with all QPE qubits. The chromatic number subsequently gives the number of parallel rotation gate executions. As the connection graph is entirely bipartite, Kőnigs theorem tells that the chromatic number is the maximum degree in the graph [77], which yields in our case

$$\mathbf{L}_{\mathrm{add}} = \max\{N, M\}. \qquad \text{(C6)}$$

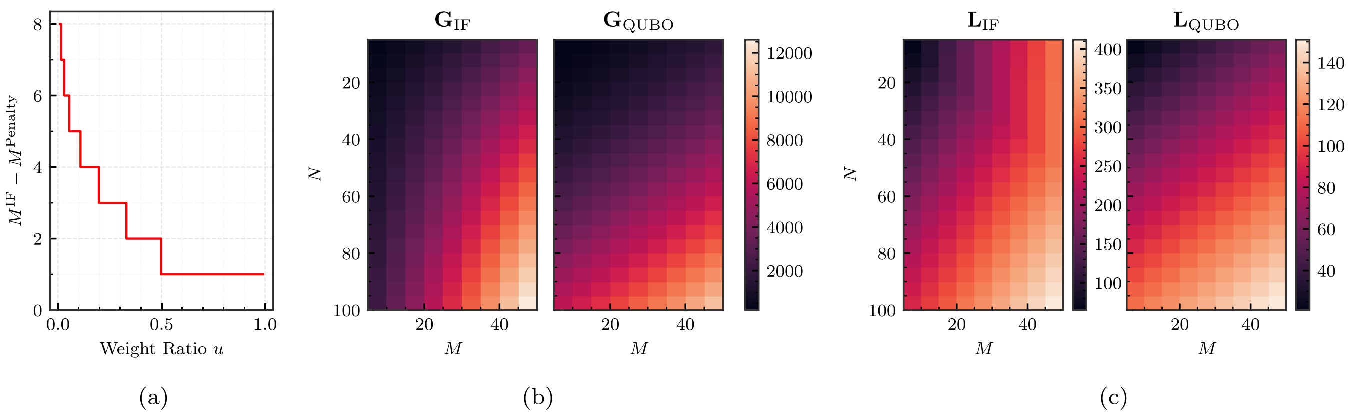


FIG. 11. (a) Shows the upper bound of $M^{\mathrm{IF}} - M^{\mathrm{QUBO}}$. This aligns with the exact expression if W is a power of two. Otherwise, the curve is slightly squeezed to the left. (b) and (c) show the gate and layer resources of a single QAOA layer of the two quantum circuit *Ansätze* as heatmaps with respect to problem size N and ancillary register size M.

The bipartite graph has $\mathbf{G}_{\mathrm{add}} = NM$ edges.

The QAOA phase part is given by $\mathbf{G}_{\mathrm{phase}} = N$ sequentially controlled phase gates applied on the compute register qubits

$$\mathbf{L}_{\mathrm{phase}} = N. \tag{C7}$$

Alternatively, the controlled rotation can be accelerated using $A < N$ ancillary qubits entangled with the control, similar to the method proposed in Ref. [38]. The number of sequential controlled phase gates is given by $\lceil N/(A+1)\rceil$, i.e., by how many parallel operations are possible. This parallel block is surrounded by a binary cnot tree that copies the state of the control qubit to the ancillaries, achievable in $[\log_2(A+1)]$ depth. The number of layers is then given by

$$\mathbf{L}_{\mathrm{phase}} = \min_{A}\left(2[\log_2(A+1)] + \lceil N/(A+1)\rceil\right), \tag{C8}$$

The number of two-qubit gates is increased by the $2A^*$ cnot gates, i.e., $G_{\mathrm{phase}} = N + 2A^*$, where A^* is the optimal number of ancillaries.

In total, the resources of the IF-QAOA-cost are defined

$$\begin{aligned}\mathbf{L}^{\mathrm{IF}}_{\mathrm{cost}} &= 2\max\{N, M\} + 4M + 2[\log_2(A^*+1)] \\ &\quad + \lceil N/(A^*+1)\rceil - 2,\end{aligned} \tag{C9}$$

$$\mathbf{G}^{\mathrm{IF}}_{\mathrm{cost}} = 2NM + M(M-1) + N + 2A^*. \tag{C10}$$

APPENDIX D: KNAPSACK RESOURCE COMPARISON

To compare the circuit resources required for the two methods, we compute $\mathbf{L}$ and $\mathbf{G}$ for a single QAOA layer for all instances considered in Sec. V. In Fig. 10, one can observe that in terms of two-qubit gate counts, IF-QAOA and QUBO have similar requirements, with IF-QAOA needing a slight, but constant overhead in gates. On average, this constant is approximately 110 two-qubit gates. However, focusing on the CLO, it is apparent that IF-QAOA is significantly more resource-intensive. This is due to the doubled QPE and controlled phase application required for the IF construction. On average, IF-QAOA circuits require 3.1× more CLO than QUBO circuits over the considered instances, with a slightly decreasing tendency as the problem size grows.

Figure 11 gives a more detailed view of the circuit requirements, additionally depending on the size of the ancillary register M. Recall that M differs depending on the number of QPE qubits required to resolve $g(x)$ in the case of IF-QAOA and the number of slack variables necessary to encode the range $\{0, \dots, W\}$ in the QUBO case, with $M^{\mathrm{IF}} > M^{\mathrm{QUBO}}$, see Eq. (19). M^{IF} can be reformulated through the weight ratio $u = W/\sum_i w_i$

$$\begin{aligned}M^{\mathrm{IF}} &= \lceil\max\{\log_2(1/u - 1), 0\} + \log_2 W\rceil + 1 \\ &\leqslant \max\{\lceil\log_2(1-u) - \log_2 u\rceil, 0\} + 1 + M^{\mathrm{QUBO}}.\end{aligned}$$

The difference between the upper bound of M^{IF} and M^{QUBO} is shown in Fig. 11(a). When $u \geqslant 1/2$, IF-QAOA requires just one additional qubit in the QPE register compared to the QUBO slack variables. However, when $u \to 0$, circuit resources for IF-QAOA significantly overgrow QUBO requirements. This is because $\{0, \dots, W\}$ are the only states that need to be covered by the slack variables, and the number of states is significantly fewer compared to the whole range of g if u is small.

Figure 11(b) shows the gate resources, which behave similarly in magnitude. $\mathbf{G}_{\mathrm{QUBO}}$ is quadratically dependent on $N + M$, thus, we can observe this regular diagonal pattern. $\mathbf{G}^{\mathrm{IF}}$ is only linearly dependent on N but quadratically on M, giving us a slightly different pattern. Most significantly, when M is small, but N is large, $\mathbf{G}^{\mathrm{IF}} < \mathbf{G}^{\mathrm{QUBO}}$.

We can identify a similar behavior for the circuit depth displayed in Fig. 11. $\mathbf{L}^{\mathrm{IF}}$ is significantly larger (≈3×), therefore, we display the heat map with separate scales. Again, QUBO only depends on $N + M$, exhibiting a regular diagonal pattern. For IF, we can distinguish two regimes. If $M > N$, $\mathbf{L}^{\mathrm{IF}}$ only depends on M and an inferior $\log_2 N$ dependence through $A < N$. In the other regime ($N > M$), we observe on $\mathbf{L}^{\mathrm{IF}} \propto N + 2M$, leading to a diagonal pattern with a steeper angle.

[1] Google Quantum AI and Collaborators, Quantum error correction below the surface code threshold, Nature(London) **638**, 920 (2025).

[2] M. Liu, R. Shaydulin, P. Niroula, M. DeCross, S.-H. Hung, W. Y. Kon, E. Cervero-Martín, K. Chakraborty, O. Amer, S. Aaronson, A. Acharya, Y. Alexeev, K. J. Berg, S. Chakrabarti, F. J. Curchod, J. M. Dreiling, N. Erickson, C. Foltz, M. Foss-Feig, D. Hayes *et al.*, Certified randomness using a trapped-ion quantum processor, Nature(London) **640** 343 (2025).

[3] H. Markowitz, Portfolio selection, J. Fin. **7**, 77 (1952).

[4] E. Grant, T. S. Humble, and B. Stump, Benchmarking quantum annealing controls with portfolio optimization, Phys. Rev. Appl. **15**, 014012 (2021).

[5] G. Buonaiuto, F. Gargiulo, G. De Pietro, M. Esposito, and M. Pota, Best practices for portfolio optimization by quantum computing, experimented on real quantum devices, Sci. Rep. **13**, 19434 (2023).

[6] D. Venturelli, D. J. J. Marchand, and G. Rojo, Quantum annealing implementation of job-shop scheduling, arXiv:1506.08479.

[7] S. Feld, C. Roch, T. Gabor, C. Seidel, F. Neukart, I. Galter, W. Mauerer, and C. Linnhoff-Popien, A hybrid solution method for the capacitated vehicle routing problem using a quantum annealer, Frontiers in ICT **6**, 13 (2019).

[8] S. Yarkoni, E. Raponi, T. Bäck, and S. Schmitt, Quantum annealing for industry applications: Introduction and review, Rep. Prog. Phys. **85**, 104001 (2022).

[9] A. Y. Wei, P. Naik, A. W. Harrow, and J. Thaler, Quantum algorithms for jet clustering, Phys. Rev. D **101**, 094015 (2020).

[10] Y. Zhu, W. Zhuang, C. Qian, Y. Ma, D. E. Liu, M. Ruan, and C. Zhou, A novel quantum realization of jet clustering in high-energy physics experiments, Sci. Bulletin **70**, 460 (2025).

[11] H. Okawa, X.-Z. Tao, Q.-G. Zeng, and M.-H. Yung, Quantum-annealing-inspired algorithms for multijet clustering, Phys. Lett. B **864**, 139393 (2025).

[12] A. Abbas, A. Ambainis, B. Augustino, A. Bärtschi, H. Buhrman, C. Coffrin, G. Cortiana, V. Dunjko, D. J. Egger, B. G. Elmegreen, N. Franco, F. Fratini, B. Fuller, J. Gacon, C. Gonciulea, S. Gribling, S. Gupta, S. Hadfield, R. Heese, G. Kircher, T. Kleinert *et al.*, Challenges and opportunities in quantum optimization, Nat. Rev. Phys. **6**, 718 (2024).

[13] R. M. Karp, Reducibility among combinatorial problems, in *Complexity of Computer Computations: Proceedings of a Symposium on the Complexity of Computer Computations, March 20–22, 1972, New York*, edited by R. E. Miller, J. W. Thatcher, and J. D. Bohlinger (Springer US, Boston, 1972), pp. 85–103.

[14] F. Glover, G. Kochenberger, R. Hennig, and Y. Du, Quantum bridge analytics I: A tutorial on formulating and using QUBO models, Ann. Oper. Res. **314**, 141 (2022).

[15] A. Lucas, Ising formulations of many NP problems, Front. Phys. **2**, (2014).

[16] M. Born and V. Fock, Beweis des adiabatensatzes, Z. Phys. **51**, 165 (1928).

[17] E. Farhi, J. Goldstone, S. Gutmann, and M. Sipser, Quantum computation by adiabatic evolution, arXiv:quant-ph/0001106.

[18] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv:1411.4028.

[19] T. Shirai and N. Togawa, Compressed space quantum approximate optimization algorithm for constrained combinatorial optimization, IEEE Trans. Quant. Eng. **6**, 3102214 (2025).

[20] S. Hadfield, Z. Wang, E. G. Rieffel, B. O'Gorman, D. Venturelli, and R. Biswas, Quantum approximate optimization with hard and soft constraints, *Proceedings of the Second International Workshop on Post Moores Era Supercomputing, PMES'17* (Association for Computing Machinery, New York, 2017), pp. 15–21.

[21] P. D. de la Grand'rive and J.-F. Hullo, Knapsack problem variants of QAOA for battery revenue optimisation, arXiv:1908.02210.

[22] K. Kea, C. Huot, and Y. Han, Leveraging knapsack QAOA approach for optimal electric vehicle charging, IEEE Access **11**, 109964 (2023).

[23] Y. Deller, S. Schmitt, M. Lewenstein, S. Lenk, M. Federer, F. Jendrzejewski, P. Hauke, and V. Kasper, Quantum approximate optimization algorithm for qudit systems, Phys. Rev. A **107**, 062410 (2023).

[24] P. Christiansen, L. Binkowski, D. Ramacciotti, and S. Wilkening, Quantum tree generator improves QAOA state-of-the-art for the knapsack problem, arXiv:2411.00518.

[25] A. Glos, A. Krawiec, and Z. Zimborás, Space-efficient binary optimization for variational quantum computing, npj Quantum Inf. **8**, 39 (2022).

[26] G. E. Crooks, Gradients of parameterized quantum gates using the parameter-shift rule and gate decomposition, arXiv:1905.13311.

[27] T. Jones and J. Gacon, Efficient calculation of gradients in classical simulations of variational quantum algorithms, arXiv:2009.02823.

[28] S. H. Sack and M. Serbyn, Quantum annealing initialization of the quantum approximate optimization algorithm, Quantum **5**, 491 (2021).

[29] J. A. Montañez-Barrera and K. Michielsen, Toward a linear-ramp QAOA protocol: Evidence of a scaling advantage in solving some combinatorial optimization problems, npj Quantum Inf. **11**, 131 (2025).

[30] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems* (Springer, Berlin, 2004).

[31] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness* (W. H. Freeman & Co., New York, 1990).

[32] S. Martello, D. Pisinger, and P. Toth, Dynamic programming and strong bounds for the 0-1 knapsack problem, Manag. Sci. **45**, 414 (1999).

[33] P. J. Kolesar, A branch and bound algorithm for the knapsack problem, Manag. Sci. **13**, 723 (1967).

[34] S. Martello and P. Toth, Knapsack *problems: Algorithms and Computer Implementations*, Wiley-Interscience Series in Discrete Mathematics and Optimization (J. Wiley, Chichester, 1990).

[35] L. K. Grover, A fast quantum mechanical algorithm for database search, *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing* (Association for Computing Machinery, New York, NY, 1996), pp. 212–219.

[36] C. Durr and P. Hoyer, A quantum algorithm for finding the minimum, arXiv:quant-ph/9607014.

[37] A. Gilliam, S. Woerner, and C. Gonciulea, Grover adaptive search for constrained polynomial binary optimization, Quantum **5**, 428 (2021).

[38] S. Wilkening, A.-I. Lefterovici, L. Binkowski, M. Perk, S. P. Fekete, and T. J. Osborne, A quantum algorithm for

solving 0-1 Knapsack problems, npj Quantum Inf. **11**, 146 (2025).
[39] T. Shirai and N. Togawa, Post-processing variationally scheduled quantum algorithm for constrained combinatorial optimization problems, IEEE Trans. Quant. Eng. **5**, 3101114 (2024).
[40] C. Gambella and A. Simonetto, Multiblock ADMM heuristics for mixed-binary optimization on classical and quantum computers, IEEE Trans. Quant. Eng. **1**, 3102022 (2020).
[41] Z. Zhao, L. Fan, and Z. Han, Hybrid quantum benders' Decomposition for mixed-integer linear programming, in *2022 IEEE Wireless Communications and Networking Conference (WCNC)* (IEEE Press, Austin, 2022), pp. 2536–2540.
[42] P. Mirkarimi, I. Shukla, D. C. Hoyle, R. Williams, and N. Chancellor, Quantum optimization with linear ising penalty functions for customer data science, Phys. Rev. Res. **6**, 043241 (2024).
[43] C. Roch, A. Impertro, and C. Linnhoff-Popien, Cross entropy optimization of constrained problem Hamiltonians for quantum annealing, in *Computational Science–ICCS 2021*, edited by M. Paszynski, D. Kranzlmüller, V. V. Krzhizhanovskaya, J. J. Dongarra, and P. M. A. Sloot (Springer International Publishing, Cham, 2021), Vol. 12747, pp. 60–73.
[44] M. Hess, L. Palackal, A. Awasthi, and K. Wintersperger, Effective embedding of integer linear inequalities for variational quantum algorithms, *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE Computer Society, Piscataway, NJ, 2024), pp. 221–231.
[45] E. Gabbassov, G. Rosenberg, and A. Scherer, Lagrangian duality in quantum optimization: Overcoming QUBO limitations for constrained problems, Phys. Rev. Res. **7**, 023305 (2025).
[46] J. A. Montañez-Barrera, D. Willsch, A. Maldonado-Romo, and K. Michielsen, Unbalanced penalization: A new approach to encode inequality constraints of combinatorial problems for quantum optimization algorithms, Quantum Sci. Technol. **9**, 025022 (2024).
[47] W. van Dam, K. Eldefrawy, N. Genise, and N. Parham, Quantum optimization heuristics with an application to knapsack problems, in *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE Computer Society, Piscataway, NJ, 2021).
[48] A. Bottarelli, S. Schmitt, and P. Hauke, Inequality constraints in variational quantum circuits with qudits, Phys. Rev. Res. **7**, 033202 (2025).
[49] T. G. Draper, Addition on a quantum computer, arXiv:quant-ph/0008033.
[50] S. Hadfield, Z. Wang, B. O'Gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, From the quantum approximate optimization algorithm to a quantum alternating operator ansatz, Algorithms **12**, 34 (2019).
[51] Z. Wang, N. C. Rubin, J. M. Dominy, and E. G. Rieffel, *XY*-mixers: Analytical and numerical results for QAOA, Phys. Rev. A **101**, 012320 (2020).
[52] J. Cook, S. Eidenbenz, and A. Bärtschi, The quantum alternating operator ansatz on maximum K-vertex cover, in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE Computer Society, Piscataway, NJ, 2020), pp. 83–92.
[53] N. P. Sawaya, A. T. Schmitz, and S. Hadfield, Encoding trade-offs and design toolkits in quantum algorithms for discrete optimization: Coloring, routing, scheduling, and other problems, Quantum **7**, 1111 (2023).
[54] F. G. Fuchs, K. O. Lye, H. M. Nilsen, A. J. Stasik, and G. Sartor, Constrained mixers for the quantum approximate optimization algorithm, Algorithms **15**, 202 (2022).
[55] Y. Ruan, Z. Yuan, X. Xue, and Z. Liu, Quantum approximate optimization for combinatorial problems with constraints, Inf. Sci. **619**, 98 (2023).
[56] Z. H. Saleem, T. Tomesh, B. Tariq, and M. Suchara, Approaches to constrained quantum approximate optimization, SN Comput. Sci. **4**, 183 (2023).
[57] A. Bärtschi and S. Eidenbenz, Grover mixers for QAOA: Shifting complexity from mixer design to state preparation, in *2020 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE Computer Society, Piscatway, NJ, 2020), pp. 72–82.
[58] D. Cruz, R. Fournier, F. Gremion, A. Jeannerot, K. Komagata, T. Tosic, J. Thiesbrummel, C. L. Chan, N. Macris, M.-A. Dupertuis, and C. Javerzac-Galy, Efficient quantum algorithms for GHZ and W states, and implementation on the IBM quantum computer, Adv. Quantum Technol. **2**, 1900015 (2019).
[59] D. Herman, R. Shaydulin, Y. Sun, S. Chakrabarti, S. Hu, P. Minssen, A. Rattew, R. Yalovetzky, and M. Pistoia, Constrained optimization via quantum zeno dynamics, Commun. Phys. **6**, 219 (2023).
[60] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition*, 1st ed. (Cambridge University Press, Cambridge, 2012).
[61] D. Lykov, R. Shaydulin, Y. Sun, Y. Alexeev, and M. Pistoia, Fast simulation of high-depth QAOA circuits, *Proceedings of the SC '23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis* (Association for Computing Machinery, New York, NY, 2023), pp. 1443–1451.
[62] J. Golden, A. Baertschi, D. O'Malley, E. Pelofske, and S. Eidenbenz, JuliQAOA: Fast, flexible QAOA simulation, *Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis, SC-W '23* (Association for Computing Machinery, New York, 2023), pp. 1454–1459.
[63] J. Stein, J. Blenninger, D. Bucher, P. J. Eder, E. Çetiner, M. Zorn, and C. Linnhoff-Popien, CUAOA: A novel cuda-accelerated simulation framework for the QAOA, *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE Computer Society, New York, 2024), pp. 280–285.
[64] P. L. Hammer, P. Hansen, and B. Simeone, Roof duality, complementation and persistency in quadratic 0–1 optimization, Math. Program. **28**, 121 (1984).
[65] M. X. Goemans and D. P. Williamson, Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming, J. ACM **42**, 1115 (1995).
[66] M. Ayodele, Penalty weights in QUBO formulations: Permutation problems, *Evolutionary Computation in Combinatorial Optimization* (Springer, Cham, 2022), pp. 159–174.
[67] D. C. Liu and J. Nocedal, On the limited memory BFGS method for large scale optimization, Math. Program. **45**, 503 (1989).
[68] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, Quantum approximate optimization algorithm: Performance,

mechanism, and implementation on near-term devices, Phys. Rev. X **10**, 021067 (2020).

[69] D. Bucher, N. Kraus, J. Blenninger, M. Lachner, J. Stein, and C. Linnhoff-Popien, Towards robust benchmarking of quantum optimization algorithms, *2024 IEEE International Conference on Quantum Computing and Engineering (QCE)* (IEEE Computer Society, Piscataway, NJ, 2024), pp. 159–170.

[70] T. Albash and D. A. Lidar, Adiabatic quantum computation, Rev. Mod. Phys. **90**, 015002 (2018).

[71] J. Jooken, P. Leyman, and P. De Causmaecker, A new class of hard problem instances for the 0–1 knapsack problem, Eur. J. Oper. Res. **301**, 841 (2022).

[72] G. H. Low and I. L. Chuang, Optimal Hamiltonian simulation by quantum signal processing, Phys. Rev. Lett. **118**, 010501 (2017).

[73] A. Gilyén, Y. Su, G. H. Low, and N. Wiebe, Quantum singular value transformation and beyond: Exponential improvements for quantum matrix arithmetics, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing* (Association for Computing Machinery, New York, NY, 2019), pp. 193–204.

[74] J. Stein, L. Müller, L. Hölscher, G. Chnitidis, J. Jojo, A. Farea, M. S. Çelebi, D. Bucher, J. Wulf, D. Fischer, P. Altmann, C. Linnhoff-Popien, and S. Feld, Exponential quantum speedup for simulation-based optimization applications, arXiv:2305.08482.

[75] D. Bucher, dbucher97/fast-qaoa-c: Fast QAOA Simulation Framework (2025), Zenodo, https://doi.org/10.5281/zenodo.17399560.

[76] J. H. van Lint and R. M. Wilson, *A Course in Combinatorics*, 2nd ed. (Cambridge University Press, Cambridge, 2006).

[77] N. Biggs, E. K. Lloyd, R. J. Wilson, and R. J. Wilson, *Graph Theory: 1736-1936*, 1st ed. (Clarendon Press, Oxford, 1998).