Quantum approximate stochastic combinatorial optimisation

Edric Matwiejew[1], Paulo E. Santos[2], Jingbo B. Wang[3], and Ugo Varetto[1]

[1]Pawsey Supercomputing Research Centre, Perth, WA 6152, Australia
[2]PrioriAnalytica, Adelaide, SA 5000, Australia
[3]Centre for Quantum Information, Simulation and Algorithms, The University of Western Australia, Perth, WA 6009, Australia

October 16, 2026

arXiv:2610.22876v1 [quant-ph] 16 Oct 2026

Two-stage stochastic combinatorial optimisation concerns discrete decisions made before all relevant information is known. An initial decision is made; once uncertainty is resolved, it can be adapted to the realised scenario, at a cost. The aim is to find the initial decision that minimises the upfront cost plus the expected cost of optimal adaptation. The first-stage decision problem is often NP-hard, and computing the expected minimum recourse cost can be #P-hard for a single first-stage candidate. We introduce Quantum Approximate Stochastic Combinatorial Optimisation (QASCO), a variational quantum-walk ansatz that coherently incorporates scenario-dependent recourse into its dynamics to search over first-stage decisions without explicitly evaluating expected minimum recourse costs. We demonstrate QASCO for MaxCut with stochastic edge weights, obtain a circuit with polynomial per-layer gate complexity, and study the ansatz via exact state-vector simulation. Across the instances considered, QASCO concentrates amplitude more effectively on stochastic-optimal first-stage decisions than a previously proposed two-stage QAOA construction. We further show that the ansatz can be calibrated using a depth-independent number of parameters and applied to unseen instances without further circuit-parameter optimisation, including joint decision–scenario–recourse spaces spanning more than 2^{32} configurations. Under a query-cost model, comparison with the tested classical heuristics using sample-average approximations of the expected minimum recourse cost shows that QASCO more reliably reaches globally optimal first-stage solutions while requiring fewer cut evaluations. These results show how to incorporate scenario-dependent recourse into the ansatz dynamics, providing a general framework for variational quantum walks in two-stage stochastic combinatorial optimisation.

1 Introduction

Combinatorial optimisation in domains such as vehicle routing, energy systems, telecommunications network design, and production scheduling often must be carried out before all relevant input data are known. A decision based on historical or modelled information

Edric Matwiejew: edric.matwiejew@pawsey.org.au

may need to be adapted once the remaining uncertainties are resolved. Evaluating a candidate decision therefore requires accounting not only for its immediate cost but also for the recourse actions it may demand across a range of possible scenarios [1, 2]. Two-stage stochastic optimisation with recourse provides a standard framework for this setting. An initial decision is made before uncertainty is resolved, incurring a deterministic first-stage cost, while a second-stage cost captures the optimal adaptation after an uncertain scenario is revealed [1–3].

Two-stage stochastic programs are computationally demanding because evaluating the expected minimum recourse cost for a given initial decision is often #P-hard, even when the stochastic parameters have independent probability distributions [4, 5]. This has motivated a long line of classical approximation strategies, ranging from sample average approximation (SAA), which replaces the exact expectation value by an empirical average over sampled scenarios [6], to machine-learning methods that approximate the recourse value function or guide candidate selection within a heuristic search [7–9]. This places an emphasis on the development of methods that avoid evaluating the expected minimum recourse cost exactly—either by *estimating* it more cheaply or by *generating* promising candidates without evaluating it at all.

In quantum computing, the Quantum Approximate Optimisation Algorithm (QAOA) [10, 11] and variants such as the Quantum Walk-based Optimisation Algorithm (QWOA) [12, 13] are prominent approaches to deterministic optimisation. These algorithms use a classically parameterised alternating ansatz comprised of a phase-separation unitary that encodes solution costs and a mixing unitary that drives transitions between candidate solutions. In QWOA, the mixing unitary is formulated as a continuous-time quantum walk (CTQW) on a graph tailored to the combinatorial structure of the search space. By exploiting correlations between the phase-encoded cost landscape and this graph structure [14], the quantum walk can improve convergence and substantially reduce the overhead of optimising the classical variational parameters [15].

Recent work has begun to address stochastic optimisation with recourse using QAOA-based methods. Rotello et al. [16] use digitised quantum annealing, implemented with a QAOA circuit, to approximately optimise second-stage decisions across scenarios, with an auxiliary register encoding the scenario distribution and controlling the second-stage variables. Quantum amplitude estimation is then used to estimate the expected recourse cost of the prepared state, providing a quadratic improvement in estimation-query complexity over classical Monte Carlo. Assuming the scenario-distribution state can be prepared efficiently, the runtime is independent of the number of scenarios, while representing the distribution as a quantum state provides an exponential reduction in space. However, the method evaluates the recourse objective for a classically supplied first-stage candidate rather than generating that candidate. More closely related to the present work, Kuroiwa et al. [17] combine a quantum generative adversarial network (qGAN) with QAOA in a single variational circuit. The qGAN encodes the scenario distribution, while QAOA jointly optimises the first- and second-stage variables by minimising the average measured first-stage and per-scenario recourse cost.

Here we introduce Quantum Approximate Stochastic Combinatorial Optimisation (QASCO), a variational quantum-walk ansatz for two-stage stochastic optimisation with recourse. The ansatz coherently aggregates amplitudes with phase-encoded scenario-dependent recourse information before applying a quantum walk over the aggregated first-stage solution space. Recourse information thereby influences the ansatz dynamics used to generate first-stage candidates without requiring explicit evaluation of the expected minimum recourse cost. The main contributions of this work are:

1. We introduce QASCO, a quantum-walk-based ansatz for two-stage stochastic combinatorial optimisation with recourse. Given an efficiently preparable scenario distribution, it uses scenario-dependent recourse to guide first-stage candidate generation without explicitly evaluating expected minimum recourse costs within the circuit.

2. We construct a QASCO quantum circuit for MaxCut with mutually independent edge-weight perturbations drawn from known finite distributions. Under the stated circuit assumptions, its per-layer gate complexity is polynomial.

3. We compare QASCO with an adaptation of the two-stage QAOA architecture of [17] using exact state-vector simulations. Across the tested problem ensembles, QASCO more consistently amplifies stochastic-optimal solutions with comparable parameter-optimisation effort.

4. We evaluate QASCO as a candidate generator under a query-cost model using the ansatz-depth-independent parameterisation scheme developed for QWOA in [15]. Calibrated parameters remain effective on new instances within each problem setting without per-instance optimisation. QASCO reaches a globally optimal solution in all 2400 test runs and has the lowest expected query cost in all eight settings against the tested classical uniform enumeration, greedy and tabu search methods using SAA estimates of the expected minimum recourse cost.

The remainder of the manuscript is structured as follows. Section 2 reviews the relevant background on stochastic combinatorial optimisation with recourse, introduces the stochastic MaxCut benchmark, and summarises the QWOA framework. We then introduce QASCO in Section 3. Section 4 details its circuit implementation for stochastic MaxCut, including recourse oracle construction and gate complexity analysis. Section 5 describes the numerical study of QASCO applied to the stochastic MaxCut problem, including benchmark ensemble generation, state-vector simulation, convergence metrics, parameter calibration, and the candidate-generator benchmark. In Section 6, we present our numerical results and discuss them in Section 7. We conclude in Section 8.

2 Background

2.1 Stochastic combinatorial optimisation with recourse

Combinatorial optimisation seeks solutions $x = (x_0, \ldots, x_{n-1}) \in X$ that minimise a real-valued cost function over a discrete decision space. Stochastic combinatorial optimisation addresses problems with uncertainty in parameters. One such approach models a stochastic problem in two stages. In the *first stage*, a solution $x \in X$ is selected based on known deterministic information. In the *second stage*, a scenario $\omega \in \Omega$ specifies the realised values of the uncertain parameters. A recourse action $y \in Y(x, \omega)$ may then be taken, incurring a cost that depends on x, y, and ω. The goal is to identify first-stage solutions that minimise the sum of the first-stage cost and the expected minimum recourse cost.

More concretely, let $\omega = (\omega_0, \ldots, \omega_{d_\omega - 1})$ denote a realisation of d_ω uncertain parameters, drawn from a finite scenario set Ω with joint probability mass function μ,

$$\mu(\omega) \geq 0, \qquad \sum_{\omega \in \Omega} \mu(\omega) = 1. \tag{1}$$

The first-stage cost $C(x)$ is a deterministic function that captures costs that can be evaluated before uncertainty is resolved, independent of μ [1, 2]. In the second stage, taking

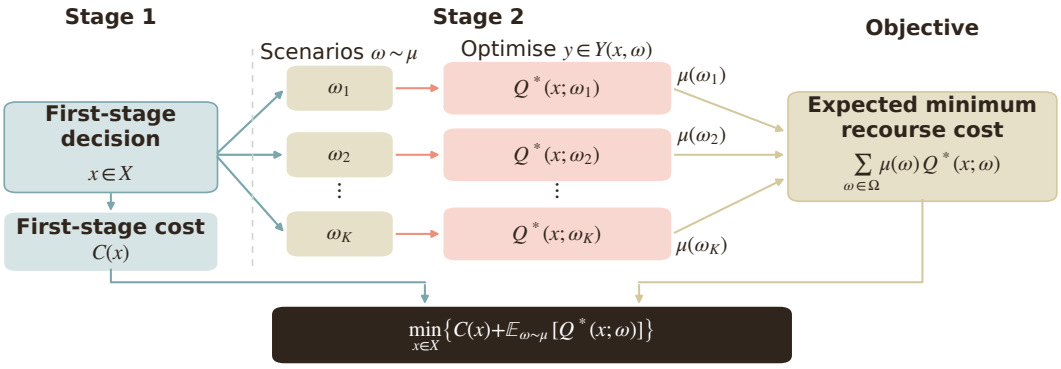


Figure 1: Two-stage stochastic optimisation with recourse. A first-stage decision x is selected before uncertainty is revealed, incurring the deterministic cost $C(x)$. For each scenario $\omega \in \Omega$, the recourse action is chosen from $Y(x,\omega)$ to minimise $Q(x,y;\omega)$, giving $Q^*(x;\omega)$. Averaging this minimum cost under μ and adding $C(x)$ gives the stochastic objective $J_\mu(x)$.

recourse action $y \in Y(x,\omega)$ incurs a recourse cost $Q(x,y;\omega)$ that depends on the first-stage decision, the recourse action, and the realised scenario. The minimum recourse cost for a given first-stage decision x and scenario ω is

$$Q^*(x;\omega) = \min_{y\in Y(x,\omega)} Q(x,y;\omega), \tag{2}$$

which, together with $C(x)$, defines the two-stage objective function

$$J_\mu(x) := C(x) + \mathbb{E}_{\omega\sim\mu}[Q^*(x;\omega)], \qquad \min_{x\in X} J_\mu(x), \tag{3}$$

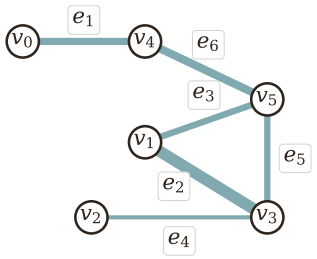
where $\mathbb{E}_{\omega\sim\mu}[Q^*(x;\omega)] = \sum_{\omega\in\Omega} \mu(\omega)\, Q^*(x;\omega)$ is the expectation of the minimum recourse cost over all possible scenarios [2, 3]. We illustrate the relationship between $J_\mu(x)$ and the various components of the two-stage stochastic optimisation problem in Figure 1.

2.1.1 Stochastic MaxCut

In this work, we consider the familiar weighted MaxCut problem with stochastic perturbations of the edge weights. The goal is to identify partitions of graph vertices into two sets such that the sum of the nominal edge weights between the sets (the "cut value") is large, while the expected minimum recourse cost under stochastic perturbations of the edge weights is small. A cut whose value falls below the target cut value can be repaired, when possible, by moving a minimal number of vertices to the other side of the cut. The resulting repair structure is closely related to recoverable robust optimisation, where an initial solution may be modified after uncertainty is realised [18], and to two-stage stochastic combinatorial optimisation problems with binary recourse [19]. Threshold-based models also arise in chance-constrained optimisation and reliable stochastic optimisation [20–22], where the resulting non-linearity generally prevents reduction to a deterministic problem instance built on the mean of the uncertain parameters.

Let $G = (V,E)$ be an undirected graph with $|V| = n$ vertices, where each edge $e \in E$ has a known nominal weight $\bar{w}_e > 0$. A partition $x = (x_0,\ldots,x_{n-1}) \in \{0,1\}^n$ corresponds to spin values $z_i := 1 - 2x_i \in \{+1,-1\}$, so $z_i \neq z_j$ when vertices i and j lie on opposite sides of the cut. The first-stage cost is the negative nominal cut value,

$$C(x) = -\sum_{e=\{u,v\}\in E} \bar{w}_e \frac{1 - z_u z_v}{2}. \tag{4}$$



Edge	$\bar{w}_e$	Δ_e	μ_e
e_1 (v_0, v_4)	18.22	{-9.3, -0.4, +15.1}	(0.26, 0.57, 0.17)
e_2 (v_1, v_3)	25.76	{-10.7, -8.7, +18.9}	(0.33, 0.33, 0.34)
e_3 (v_1, v_5)	16.85	{-14.2, +10.2, +32.7}	(0.66, 0.08, 0.26)
e_4 (v_2, v_3)	12.85	{-1.7, -0.0, +10.8}	(0.46, 0.47, 0.07)
e_5 (v_3, v_5)	16.56	{-5.2, -4.8, +3.9}	(0.20, 0.25, 0.56)
e_6 (v_4, v_5)	19.64	{-5.9, +2.0, +5.7}	(0.30, 0.59, 0.10)

Figure 2: Representative six-vertex stochastic MaxCut instance. **Left**: graph structure with edge thickness proportional to the nominal weight $\bar{w}_e$. **Right**: nominal weight $\bar{w}_e$, perturbation support Δ_e, and marginal distribution μ_e for each edge.

Each edge carries a random perturbation ω_e, drawn independently across edges from a zero-mean marginal distribution μ_e with finite support Δ_e. The scenario space factorises as $\Omega = \prod_{e \in E} \Delta_e$, with one perturbation per edge so that $d_\omega = |E|$. The realised cut value under scenario ω is

$$W(x;\omega) = \sum_{e=\{u,v\} \in E} (\bar{w}_e + \omega_e) \frac{1 - z_u z_v}{2}. \tag{5}$$

Under scenario ω, a first-stage cut x requires repair whenever $W(x;\omega) < T$, with a candidate repair y being successful whenever $W(y;\omega) \geq T$. The space of potential recourse actions is $Y(x,\omega) = \{0,1\}^n$. If the original cut falls below the threshold, a successful repair costs the Hamming distance $d_H(x,y)$ (number of bit flips) between the original and repaired partitions, while an unsuccessful repair costs $\Gamma = n+1$. The recourse cost for recourse action y under scenario ω, given first-stage cut x, is

$$Q(x,y;\omega) = \begin{cases} 0, & W(x;\omega) \geq T, \\ d_H(x,y), & W(x;\omega) < T \text{ and } W(y;\omega) \geq T, \\ \Gamma, & W(x;\omega) < T \text{ and } W(y;\omega) < T. \end{cases} \tag{6}$$

As a cut and its bitwise complement describe the same partition, we define the canonical domain $X_{\text{can}} := \{x \in \{0,1\}^n : x_0 = 0\}$. We denote the sets of canonical cuts minimising the nominal objective $C(x)$ and the stochastic objective $J_\mu(x)$ by

$$X_C^* := \arg\min_{x \in X_{\text{can}}} C(x), \qquad X_\mu^* := \arg\min_{x \in X_{\text{can}}} J_\mu(x), \tag{7}$$

respectively, and refer to these as the nominal and stochastic optima. We show an illustrative stochastic MaxCut instance in Figure 2.

2.2 Quantum approximate optimisation

The Quantum Approximate Optimisation Algorithm (QAOA) and related ansätze provide a standard near-term approach to discrete optimisation on quantum devices [10, 11, 23].

The QAOA alternates two parameterised unitaries: a *phase-separation* operator $U_C(\gamma) = e^{-i\gamma\hat{C}}$, where $\hat{C}$ is the diagonal operator with $\hat{C}\,|x\rangle = C(x)\,|x\rangle$, which imprints the objective function C as relative phases on computational basis states, and a *mixing* operator $U_M(t)$, which redistributes amplitude among candidate solutions. Starting from a uniform superposition, p layers of these operators produce the ansatz state

$$|\psi(\boldsymbol{\gamma}, \boldsymbol{t})\rangle = U_M(t_p)\, U_C(\gamma_p) \cdots U_M(t_1)\, U_C(\gamma_1)\, |s\rangle\,, \tag{8}$$

where $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_p)$ and $\boldsymbol{t} = (t_1, \ldots, t_p)$ are variational parameters optimised classically to minimise the expectation value of the measured solution costs, thereby amplifying low-cost solutions. In the original QAOA formulation, the mixer is the transverse-field unitary $U_M(t) = e^{-it\sum_j \hat{X}_j}$, where $\hat{X}_j$ is the Pauli-X operator on qubit j.

The Quantum Walk-based Optimisation Algorithm (QWOA) generalises the choice of mixing graph, using a continuous-time quantum walk (CTQW) $U_W(t) = e^{-itA}$ on a problem-specific graph with adjacency matrix A [24], whose vertices represent feasible solutions [12, 13]. The QAOA transverse-field mixer is exactly equivalent to a continuous-time quantum walk on the hypercube graph connecting $\{0,1\}^n$. The mixing graph can be chosen to reflect the combinatorial structure and constraints of the feasible solution space, subject to an efficient circuit implementation. Recent constructions define graph edges using the Hamming distance between solutions [15, 25].

Optimising QAOA or QWOA parameters can incur substantial classical overhead, particularly as the circuit depth and number of free parameters increase [11, 26]. This has motivated parameter-initialisation strategies such as INTERP [26], which warm-starts depth-p optimisation by linearly interpolating the optimised phase and mixer schedules from depth $p-1$ onto the new layer grid. Recent work on QWOA has introduced a compact linear-ramp parameterisation in which, given an appropriate problem-class-specific mixing graph, the entire depth-p schedule is generated from only three parameters, $\boldsymbol{\vartheta} := (\gamma_{\max}, t_{\max}, \lambda)$ [15, 25]. The resulting schedule can achieve convergence to low-cost solutions that is close to that achived by full variational optimisation of the $2p$ ansatsz parameters with fewer evaluations of the ansatz objective function [27]. Numerical simulations of random weighted-MaxCut instances found that it maintained an approximately constant optimal-solution probability while the required layer count grew quadratically with the number of vertices over the simulated range [28]. Under this parameterisation,

$$\gamma_k = [\lambda + (1-\lambda)s_k]\gamma_{\max}, \qquad t_k = [1 - (1-\lambda)s_k]t_{\max}, \tag{9}$$

where, for layer $k \in \{1, \ldots, p\}$, $s_k := (k-1)/(p-1)$ when $p > 1$, and $s_1 := 0$ when $p = 1$.

3 Variational ansatz for stochastic optimisation

We now introduce QASCO, a variational quantum ansatz for two-stage stochastic optimisation with recourse. The ansatz represents the problem configuration space using registers for the first-stage decision, scenario distribution, and recourse actions, coupling them so that scenario-weighted recourse costs guide amplitude transfer between first-stage candidates. We first define the joint configuration-space encoding and initialisation of the three registers. We then construct a single ansatz layer, specify the variational objective and the role of recourse-state preparation, and finally give a CTQW interpretation of the resulting mixing dynamics. The QASCO quantum circuit is illustrated in Figure 3.

3.1 Configuration-space encoding

A solution $x \in X$, a scenario $\omega \in \Omega$, and a recourse action $y \in Y$ are respectively encoded in three quantum registers in the computational basis, with Hilbert spaces $\mathcal{H}_X$, $\mathcal{H}_\Omega$, and $\mathcal{H}_Y$. The joint Hilbert space is

$$\mathcal{H} = \mathcal{H}_X \otimes \mathcal{H}_\Omega \otimes \mathcal{H}_Y,$$

with computational basis states $|x\rangle_X |\omega\rangle_\Omega |y\rangle_Y$.

The scenario Hilbert space decomposes as

$$\mathcal{H}_\Omega = \bigotimes_{i=0}^{d_\omega - 1} \mathcal{H}_{\Omega_i},$$

where $\mathcal{H}_{\Omega_i}$ is the Hilbert space of the subregister encoding Ω_i. For a scenario $\omega = (\omega_0, \ldots, \omega_{d_\omega - 1}) \in \Omega$, the corresponding scenario basis state is

$$|\omega\rangle_\Omega = |\omega_0\rangle_{\Omega_0} \cdots |\omega_{d_\omega - 1}\rangle_{\Omega_{d_\omega - 1}}.$$

3.2 State initialisation

The ansatz is initialised as a product state of the solution and scenario registers followed by the, potentially controlled, preparation of the recourse register. Let S_X and S_Ω denote the solution and scenario state-preparation unitaries, satisfying

$$S_X |0\rangle_X = |s\rangle_X := \frac{1}{\sqrt{|X|}} \sum_{x \in X} |x\rangle_X, \tag{10}$$

$$S_\Omega |0\rangle_\Omega = |\mu\rangle_\Omega := \sum_{\omega \in \Omega} \sqrt{\mu(\omega)}\, |\omega\rangle_\Omega, \tag{11}$$

where μ is the target scenario distribution defined in Equation (1). While the ansatz described in the rest of this section can use any efficiently preparable scenario distribution, including those with correlated uncertain parameters, we focus here on mutually independent parameters, as this allows a concrete analysis of the circuit components for a broad range of practical problems. In this case, we prepare the scenario state using the marginal distributions μ_i:

$$S_\Omega = \bigotimes_{i=0}^{d_\omega - 1} S_{\Omega_i}, S_{\Omega_i} |0\rangle_{\Omega_i} = |\mu_i\rangle_{\Omega_i} := \sum_{\omega_i \in \Omega_i} \sqrt{\mu_i(\omega_i)}\, |\omega_i\rangle_{\Omega_i},$$

where each unitary S_{Ω_i} acts on $\lceil \log_2 |\Omega_i| \rceil$ qubits (see Appendix A). Efficient state preparation is available for certain structured distributions [29], while variational approaches can approximate other target distributions [30]. Otherwise, marginal distributions with small discrete support can be prepared by direct unitary synthesis [31].

Subsequently, we apply the recourse preparation unitary

$$S_Y = \sum_{x \in X} \sum_{\omega \in \Omega} |x, \omega\rangle\langle x, \omega|_{X\Omega} \otimes S_Y^{(x,\omega)}, \tag{12}$$

where $S_Y^{(x,\omega)}$ denotes the conditional block acting on Y, with $S_Y^{(x,\omega)} |0\rangle_Y = |s_{x,\omega}\rangle_Y$. The normalised state $|s_{x,\omega}\rangle_Y$ defines the reference distribution over recourse actions associated with (x, ω) and, for recourse spaces with hard constraints, we require $\langle y | s_{x,\omega} \rangle = 0$ whenever

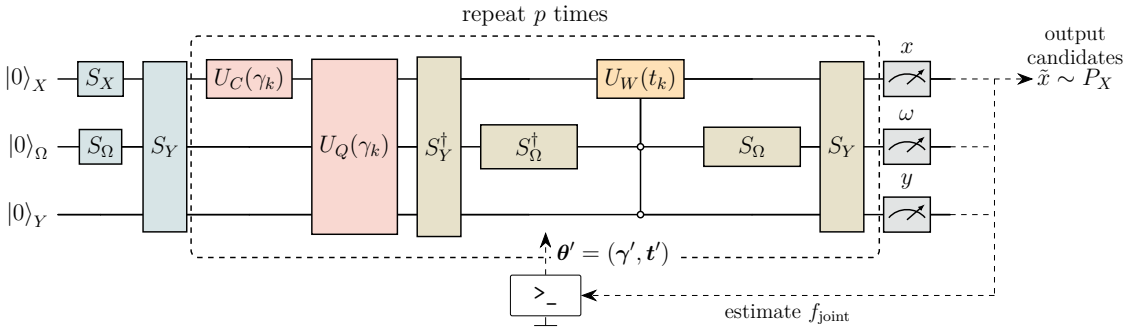


Figure 3: Schematic circuit for QASCO. S_X prepares the solution register in a uniform state over all first-stage solutions, and S_Ω prepares a probability-weighted superposition over the space of scenarios according to the target distribution μ. The recourse register is prepared conditionally on the solution and scenario registers by S_Y. U_C and U_Q phase-encode the first-stage and recourse costs, respectively. The adjoint preparation unitaries $S_Y^\dagger$ and $S_\Omega^\dagger$ aggregate amplitudes associated with the same first-stage choice in a component flagged by $|0\rangle_\Omega |0\rangle_Y$. The controlled mixer $\Lambda_0(U_W)$ acts on this component before the amplitudes are redistributed back to the full problem configuration space with S_Y and S_Ω. The ansatz objective function is estimated from measurement outcomes, and a classical optimiser updates the variational parameters. Measurements of the solution register produce first-stage candidates, whose stochastic objective values are estimated classically before final selection.

$y \notin Y(x, \omega)$. The challenges associated with recourse preparation and the preparation strategies explored in this work are discussed in Section 3.6.

Altogether, the initial ansatz state is

$$\left|\psi^{(0)}\right\rangle = S_Y\, S_\Omega\, S_X\, |0\rangle_X\, |0\rangle_\Omega\, |0\rangle_Y = \frac{1}{\sqrt{|X|}} \sum_{x \in X} \sum_{\omega \in \Omega} \sqrt{\mu(\omega)}\; |x\rangle_X\, |\omega\rangle_\Omega\, |s_{x,\omega}\rangle_Y \,. \tag{13}$$

3.3 Ansatz layer

For a single ansatz layer k, let the input state be $\left|\psi^{(k-1)}\right\rangle \in \mathcal{H}$. The layer begins with the first-stage phase-separation operator $U_C(\gamma) = e^{-i\gamma \hat{C}}$, which applies a cost-proportional phase rotation, where $\gamma \in \mathbb{R}$ is a variational parameter and $\hat{C}$ is the diagonal operator on the solution register with $\hat{C}\,|x\rangle = C(x)\,|x\rangle$. Applied in layer k, this gives $\left|\psi_C^{(k)}\right\rangle = U_C(\gamma_k)\left|\psi^{(k-1)}\right\rangle$, with amplitudes

$$(\psi_C^{(k)})_{x,\omega,y} = \psi_{x,\omega,y}^{(k-1)}\, e^{-i\gamma_k C(x)}.$$

The recourse oracle computes a cost $Q(x, y; \omega)$ for each encoded computational basis state. The recourse phase-separation operator $U_Q(\gamma)$ then imprints this cost as a phase,

$$U_Q(\gamma)\, |x\rangle_X\, |\omega\rangle_\Omega\, |y\rangle_Y = e^{-i\gamma Q(x,y;\omega)}\, |x\rangle_X\, |\omega\rangle_\Omega\, |y\rangle_Y \,. \tag{14}$$

After the recourse phase in layer k, the combined phase-separated state is

$$\left|\psi_{CQ}^{(k)}\right\rangle = \sum_{x,\omega,y} \psi_{x,\omega,y}^{(k-1)}\, e^{-i\gamma_k[C(x)+Q(x,y;\omega)]}\, |x\rangle_X\, |\omega\rangle_\Omega\, |y\rangle_Y \,. \tag{15}$$

The two phase-separation operators therefore encode the combined first-stage and recourse cost $C(x) + Q(x, y; \omega)$ on each computational basis state.

The ansatz next maps the prepared scenario–recourse component associated with each first-stage decision x to the all-zero state. Define the scenario–recourse reference state as

$$|r_x\rangle_{\Omega Y} := \sum_{\omega\in\Omega} \sqrt{\mu(\omega)}\ |\omega\rangle_\Omega\, |s_{x,\omega}\rangle_Y\,, \tag{16}$$

and the corresponding prepared reference subspace as

$$\mathcal{S} := \operatorname{span}\{|x\rangle_X\, |r_x\rangle_{\Omega Y} : x \in X\}\,.$$

Because $S_Y S_\Omega$ maps $|x\rangle_X\, |0\rangle_\Omega\, |0\rangle_Y$ to $|x\rangle_X\, |r_x\rangle_{\Omega Y}$, applying the inverse preparation unitaries maps $\mathcal{S}$ to the flagged subspace $\mathcal{H}_X \otimes |0\rangle_\Omega\, |0\rangle_Y$. Acting on the phase-separated state, this gives

$$S_\Omega^\dagger S_Y^\dagger \left|\psi_{CQ}^{(k)}\right\rangle = \sum_{x\in X} \phi_x^{(k,\mu)}\, |x\rangle_X\, |0\rangle_\Omega\, |0\rangle_Y + \left|\chi_\perp^{(k)}\right\rangle, \tag{17}$$

where $\left|\chi_\perp^{(k)}\right\rangle$ has no support on $|0\rangle_\Omega\, |0\rangle_Y$. Writing $a_s(y \mid x,\omega) := \langle y|s_{x,\omega}\rangle$, the amplitude of the flagged component associated with first-stage solution x is then

$$\begin{aligned}\phi_x^{(k,\mu)} &= \langle x|_X\, \langle r_x|_{\Omega Y} \left|\psi_{CQ}^{(k)}\right\rangle \\ &= e^{-i\gamma_k C(x)} \sum_{\omega\in\Omega} \sqrt{\mu(\omega)} \sum_{y\in Y} a_s^*(y \mid x,\omega)\, \psi_{x,\omega,y}^{(k-1)}\, e^{-i\gamma_k Q(x,y;\omega)}.\end{aligned} \tag{18}$$

The mixing unitary $U_W(t) = e^{-itA_X}$ on the first-stage register is then applied only to this component by conditioning its execution on $|0\rangle_\Omega\, |0\rangle_Y$. Defining $\Pi_0 := |0,0\rangle\langle 0,0|_{\Omega Y}$ on the scenario and recourse registers, the resulting unitary is

$$\Lambda_0(U_W(t)) := U_W(t) \otimes \Pi_0 + \mathbb{I}_X \otimes (\mathbb{I}_{\Omega Y} - \Pi_0). \tag{19}$$

The recourse and scenario preparation unitaries are then reapplied so that, altogether, one QASCO layer acts as

$$\left|\psi^{(k)}\right\rangle = S_Y S_\Omega\, \Lambda_0(U_W(t_k))\, S_\Omega^\dagger S_Y^\dagger\, U_Q(\gamma_k) U_C(\gamma_k) \left|\psi^{(k-1)}\right\rangle. \tag{20}$$

3.4 Quantum walk interpretation

To motivate the QASCO mixing mechanism, observe that, since $\Pi_0^2 = \Pi_0$, the zero-controlled walk has the exponential form

$$\Lambda_0(U_W(t)) = \exp[-it(A_X \otimes \Pi_0)]. \tag{21}$$

After conjugation by $S_Y S_\Omega$, the induced generator $A_{\text{mix}} = \sum_{x,x'\in X} (A_X)_{x,x'}\, |x, r_x\rangle\langle x', r_{x'}|$ admits a partite-graph interpretation. For each first-stage decision x, we define the partite set

$$\mathcal{V}_x = \{(x,\omega,y) : \omega \in \Omega,\, y \in Y\},$$

with size $|\Omega|\,|Y|$. Each solution-space edge $(A_X)_{x,x'} \neq 0$, with $x \neq x'$, induces connections between all elements with nonzero reference-state amplitudes in the corresponding partite sets $\mathcal{V}_x$ and $\mathcal{V}_{x'}$, with weights given by

$$\begin{aligned}&\langle x,\omega,y|\, A_{\text{mix}}\, |x',\omega',y'\rangle \\ &\quad = (A_X)_{x,x'}\, \langle \omega,y|r_x\rangle\, \langle r_{x'}|\omega',y'\rangle \\ &\quad = (A_X)_{x,x'} \sqrt{\mu(\omega)\mu(\omega')}\, a_s(y \mid x,\omega) a_s^*(y' \mid x',\omega').\end{aligned} \tag{22}$$

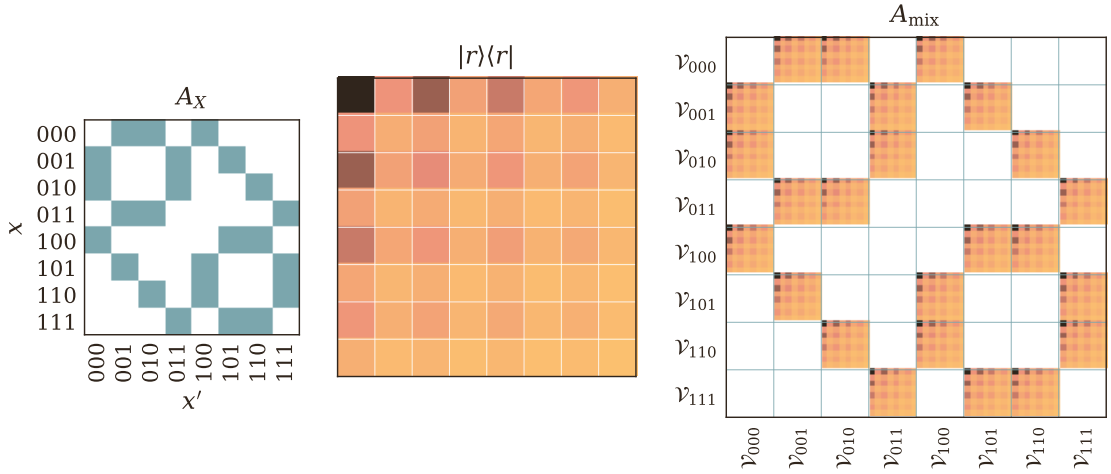


Figure 4: Induced partite-graph structure of the controlled mixer with a transverse-field, or hypercube, solution-space adjacency A_X with $|X| = 8$. Here S_Y prepares a uniform superposition over recourse actions, $|s_{x,\omega}\rangle_Y = |s\rangle_Y$. The scenario distribution is a product of $\mu_0 = (0.825, 0.175)$, $\mu_1 = (0.660, 0.340)$, and $\mu_2 = (0.739, 0.261)$. **Left**: the first-stage solution-space adjacency. **Middle**: the rank-one projector $|r\rangle\langle r|$, where $|r\rangle = |\mu\rangle_\Omega |s\rangle_Y$, with basis states ordered as (ω, y). **Right**: the induced adjacency matrix A_{mix}. Block rows and columns correspond to the partite sets $\mathcal{V}_x$ and $\mathcal{V}_{x'}$; each block is $(A_X)_{x,x'} |r\rangle\langle r|$. White entries are zero and darker entries have larger weights.

Thus, amplitude transferred into a first-stage decision is distributed across its scenario–recourse states according to the corresponding reference state. The mixer therefore acts on one aggregated amplitude per first-stage decision, incorporating scenario-weighted recourse phase information through $\phi_x^{(k,\mu)}$ (Equation (18)). Figure 4 illustrates the induced partite-graph structure for the QASCO MaxCut formulation explored later in this work when S_Y is uniform across all recourse choices.

3.5 Variational objective

Altogether, the ansatz prepares the state $|\psi(\boldsymbol{\theta})\rangle = \left|\psi^{(p)}\right\rangle$, where p is the number of ansatz layers and $\boldsymbol{\theta} = (\boldsymbol{\gamma}, \boldsymbol{t})$ are $2p$ variational parameters. The variational parameters are optimised to minimise the expectation value of the combined first-stage and recourse cost,

$$f_{\mathrm{joint}}(\boldsymbol{\theta}) = \langle \psi(\boldsymbol{\theta}) | \hat{H} | \psi(\boldsymbol{\theta}) \rangle, \tag{23}$$

where $\hat{H}$ is a diagonal cost operator with eigenvalues $C(x) + Q(x, y; \omega)$. This objective averages the combined first-stage and recourse cost under the measured ansatz distribution, whereas the problem objective J_μ (see Equation (3)) combines $C(x)$ with the minimum recourse cost averaged under the target scenario distribution. Minimising f_{joint} therefore does not by itself guarantee concentration on decisions that minimise J_μ.

Instead, the problem costs enter via the QASCO mixing process. Expanding the reference state in Equation (16) gives

$$\begin{aligned} F_s^\mu(x) &:= \langle x, r_x | \hat{H} | x, r_x \rangle \\ &= \sum_{\omega, y} \mu(\omega) |a_s(y \mid x, \omega)|^2 [C(x) + Q(x, y; \omega)] \\ &= C(x) + \sum_{\omega, y} \mu(\omega) |a_s(y \mid x, \omega)|^2 Q(x, y; \omega), \end{aligned} \tag{24}$$

which we term the *surrogate objective*. It averages the vertex costs within $\mathcal{V}_x$ according to the reference-state probabilities. For a given x, the recourse term averages $Q(x, y; \omega)$ over the target scenario distribution and the prepared recourse states. Since $\hat{H}$ is diagonal in x, the reference component $\sum_x b_x|x, r_x\rangle$ contributes $\sum_x |b_x|^2 F_s^\mu(x)$ to the measured objective. While the full ansatz objective also includes the orthogonal component and its interference with the reference component, numerical results in Section 6 suggest that changes in the reference contribution account for most of the observed reduction in the measured objective over the problem instances studied in this work.

3.6 Recourse preconditioning

The choice of recourse preparation S_Y determines how recourse costs are aggregated and, hence, how closely F_s^μ approximates J_μ. For any normalised recourse state supported on $Y(x, \omega)$,

$$\sum_{y \in Y} |a_s(y \mid x, \omega)|^2 Q(x, y; \omega) \geq Q^*(x; \omega),$$

hence $F_s^\mu(x) \geq J_\mu(x)$.

The ideal prepared state is supported only on feasible recourse actions minimising $Q(x, y; \omega)$, in which case $F_s^\mu(x) = J_\mu(x)$. However, preparing such a state is generally as hard as solving the recourse problem itself. For example, starting from a uniform recourse state over $Y(x, \omega)$, amplitude amplification requires $O(\sqrt{|Y(x, \omega)|/M})$ oracle calls to amplify the M recourse actions below a chosen cost threshold [32]. When few actions qualify and $|Y(x, \omega)| \propto |X|$, this can lead to prohibitively deep circuits.

Hence, we consider methods that prepare states merely biased toward low recourse costs within each (x, ω) block. In this work, we consider a *recourse preconditioner* based on a depth-p_R QWOA circuit with parameters optimised in advance to minimise $\langle Q|Q\rangle$, averaged over a uniform first-stage distribution and the target scenario distribution μ. We call these parameters the *recourse-preconditioner parameters*, and $\boldsymbol{\theta}$, which governs the p QASCO layers, the *ansatz parameters*. At $p_R = 0$, uniform recourse preparation gives

$$|s_{x,\omega}\rangle_Y := \frac{1}{\sqrt{|Y(x, \omega)|}} \sum_{y \in Y(x,\omega)} |y\rangle_Y ,$$

for which

$$F_s^\mu(x) = C(x) + \sum_{\omega \in \Omega} \mu(\omega) \frac{1}{|Y(x, \omega)|} \sum_{y \in Y(x,\omega)} Q(x, y; \omega),$$

Thus, uniform recourse preparation replaces the minimum recourse cost in J_μ with an average over candidate recourse actions.

4 QASCO for stochastic MaxCut

We now describe a circuit implementing QASCO (see Section 3) for the stochastic MaxCut problem defined in Section 2.1.1. The solution and recourse registers each require n qubits. The scenario register requires

$$n_\Omega = \sum_{e \in E} \lceil \log_2 |\Delta_e| \rceil$$

qubits. The configuration-space encoding therefore requires $2n + n_\Omega$ qubits.

We use b-bit fixed-point weights and assume each scenario subregister can be reversibly decoded into its numerical edge perturbation using $O(b + \log |E|)$ gates. This holds when each edge has at most a fixed number of possible perturbation values.

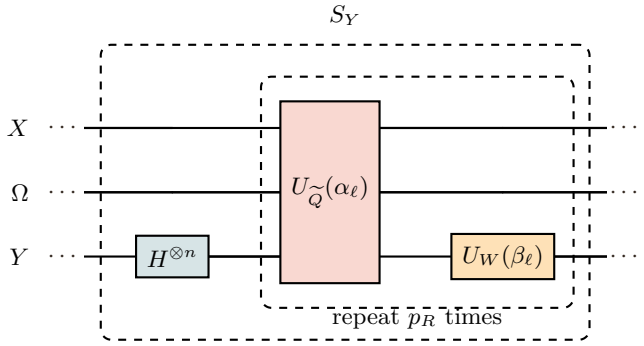


Figure 5: QWOA recourse preconditioner S_Y for stochastic MaxCut. Each of the p_R layers applies the modified recourse-phase oracle $U_{\widetilde{Q}}$, followed by transverse-field mixing on Y. The preconditioner parameters $(\boldsymbol{\alpha}, \boldsymbol{\beta})$ are optimised first and then held fixed during optimisation of the main ansatz parameters. For $p_R = 0$, the circuit reduces to $H^{\otimes n}$ on the recourse register.

For the mixing unitary $U_W(t)$, we use the transverse-field mixer on X [10]. This choice performs well for the weighted MaxCut problem and is compatible with the three-parameter schedule in Equation (9) [28]. The mixer requires n single-qubit Pauli-X rotations. In QASCO, these rotations are conditioned on the all-zero state of the scenario and recourse registers. This controlling state can be detected using a Toffoli ladder that computes the logical AND over the $n+n_\Omega$ controls, requiring $n+n_\Omega-2$ work ancillas and one flag qubit, which then controls the n rotations. After decomposition of the Toffoli gates into one- and two-qubit gates, the controlled mixer requires $O(n+n_\Omega)$ two-qubit gates [33].

Preparing the initial state on X and the first-stage phase-separation unitary also follows the standard QAOA form [10, 28]. We prepare the uniform superposition over X using n Hadamard gates, and $U_C(\gamma)$ decomposes into one two-qubit ZZ rotation for each weighted edge. Using the standard CNOT–R_Z–CNOT decomposition, this requires $2|E|$ CNOT gates and $|E|$ single-qubit rotations with no ancillary qubits.

To prepare the scenario reference state, we require unitaries S_{Ω_e} that prepare the amplitudes $\sqrt{\mu_e(\delta)}$ over $\delta \in \Delta_e$. We can realise each S_{Ω_e} using standard arbitrary state-preparation methods. For example, the uniformly controlled-rotation method of [31] requires $O(|\Delta_e|)$ single- and two-qubit gates on $\lceil \log_2 |\Delta_e| \rceil$ qubits and no additional ancillas, giving $O(\sum_{e\in E} |\Delta_e|)$ two-qubit gates for S_Ω.

For the coherent implementation of the piecewise recourse cost, we introduce indicators of whether the original and candidate repaired cuts fall below the threshold,

$$f_0 = \mathbf{1}\{W(x;\omega) < T\}, \qquad f_1 = \mathbf{1}\{W(y;\omega) < T\}.$$

We can then express the recourse cost of Equation (6) as

$$Q(x,y;\omega) = f_0(1-f_1)\, d_H(x,y) + f_0 f_1\, \Gamma. \tag{25}$$

The recourse preparation S_Y starts with n Hadamard gates, followed by p_R layers alternating phase separation under

$$\widetilde{Q}(x,y;\omega) = (1-f_1)\, d_H(x,y) + f_1\, \Gamma \tag{26}$$

with transverse-field mixing on Y (as illustrated in Figure 5). We denote this phase-separation unitary by $U_{\widetilde{Q}}$.

To compute f_0 and f_1, the realised cut values $W(x;\omega)$ and $W(y;\omega)$ are accumulated and compared with the cut threshold value T. The cut-value accumulator requires $b+\lceil \log_2 |E| \rceil$

qubits and can be reused to compute the two failure flags sequentially. Evaluating the cut values with standard reversible adders requires $O(|E|(b + \log|E|))$ two-qubit gates [34]. Including the failure flags and arithmetic workspace, this requires $O(b + \log|E|)$ reusable ancillary qubits.

The Hamming distance is $d_H(x, y) = \sum_{j=0}^{n-1}(x_j \oplus y_j)$, for which the first term of Equation (25) can be phase encoded using n commuting rotations conditioned on $f_0 = 1$, $f_1 = 0$, and $x_j \neq y_j$. The penalty term requires a single phase rotation conditioned on $f_0 = f_1 = 1$. After applying these phases, the cut-value calculations and failure flags are uncomputed. After decomposition into one- and two-qubit gates, the recourse phase oracle therefore requires $O(|E|(b + \log|E|) + n)$ two-qubit gates and $O(b + \log|E|)$ reusable ancillary qubits per application. Within S_Y, $U_{\widetilde{Q}}$ computes and uncomputes only $W(y; \omega)$. The depth-p_R recourse preparation therefore requires $O(p_R[|E|(b + \log|E|) + n])$ two-qubit gates per S_Y application, in addition to the single-qubit gates used for the initial and transverse-field mixing layers.

Combining these components, the circuit requires $2n + n_\Omega + O(\max\{n + n_\Omega, b + \log|E|\})$ qubits, where the final term accounts for reusable work ancillas required by the controlled mixer and recourse phase oracle. The one-time preparation of the initial state requires $O(\sum_{e\in E}|\Delta_e| + p_R[|E|(b + \log|E|) + n])$ two-qubit gates, in addition to $O(n(1 + p_R))$ single-qubit gates. Each QASCO layer then applies U_C and U_Q, followed by the adjoint scenario and recourse preparation unitaries, the controlled mixer, and the corresponding preparation unitaries. The resulting two-qubit gate complexity per layer is

$$O\left((1 + p_R)\left[|E|(b + \log|E|) + n\right] + n_\Omega + \sum_{e\in E}|\Delta_e|\right).$$

With at most a fixed number of possible perturbation values per edge, both n_Ω and $\sum_e|\Delta_e|$ are $O(|E|)$. The per-layer gate complexity then reduces to $O((1+p_R)[|E|(b+\log|E|)+n])$. For fixed p_R, the leading two-qubit gate cost is therefore the repeated implementation of the recourse phase oracle.

Since $|X| = 2^n$, this is $O(|E|(b + \log|E|) + \log|X|)$. If, in addition, each edge has at least two possible perturbation values, then $|E| = \Theta(\log|\Omega|)$ and this becomes

$$O(\log|\Omega|\,[b + \log\log|\Omega|] + \log|X|)\,.$$

Thus, for fixed p_R and fixed weight precision b, the per-layer two-qubit gate complexity is polylogarithmic in $|X|$ and $|\Omega|$, although the joint configuration space has size $|X|^2|\Omega|$.

5 Numerical methods

We conduct a numerical study in two parts. First, we examine QASCO's convergence behaviour and compare it with an adaptation of the QAOA-based method in [17], which we refer to as the *two-stage QAOA*, under full variational optimisation. The two-stage QAOA acts on the same registers X, Ω, and Y, and begins from the product state $|s\rangle_X|\mu\rangle_\Omega|s\rangle_Y$. In place of the recourse preconditioning and controlled mixing used by QASCO, it applies two sequential QAOA stages with transverse-field mixing. The first stage has p_1 layers acting on X, with phase separation under the first-stage cost C, while the second has p_2 layers with phase separation under the recourse cost $Q(x, y; \omega)$ and mixing over Y. Following [17], we set $p_1 = p_2 = p$. QASCO and the two-stage QAOA are optimised using the same ansatz objective function.

The second part of the numerical study evaluates QASCO as a fixed-schedule first-stage candidate generator against classical uniform enumeration of the solution space, random-restart greedy search, and tabu search. For each considered problem setting, we first calibrate the QASCO and configurable classical-search parameters on a calibration set and then evaluate performance on a separate test set. The stochastic objective is approximated using sample average approximation, and the methods are compared by the relative number of cut evaluations, or query cost, required to reach the stochastic optimum. This tests whether a schedule calibrated on representative instances can generate useful first-stage candidates without per-instance variational optimisation.

5.1 Benchmark ensembles

We performed all experiments using benchmark ensembles of randomly generated stochastic MaxCut problem instances. Graphs are sampled uniformly from connected simple graphs with n labelled vertices and $|E|$ edges. Each edge has an independent m-point weight distribution, giving $m^{|E|}$ scenarios. Given generation parameters $a < b$ and a fragile-edge fraction ρ, we select $\max\{1, \mathrm{round}(\rho|E|)\}$ edges uniformly as *fragile*. Their weights comprise $m-1$ independent values sampled uniformly from $[a, a+0.05(b-a)]$ and one sampled uniformly from $[3b, 8b]$. We sample the probability assigned to the high weight uniformly from $[0.05, 0.15]$; the remaining probability is divided among the low weights in proportion to independent values sampled uniformly from $[0, 1]$. For each remaining edge, a centre c is drawn uniformly from $[a+0.3(b-a), b-0.3(b-a)]$, with m weights drawn uniformly from $[c-0.05(b-a), c+0.05(b-a)]$ and probabilities obtained by normalising independent values sampled uniformly from $[0, 1]$.

Nominal edge weights are the probability-weighted means. Rare high weights on fragile edges can raise these means above the weights typically realised, encouraging disjoint nominal and stochastic optima. We accept only instances with unique, distinct nominal and stochastic optima in X_{can}.

The ansatz-convergence comparison (see Section 5.3) between QASCO and the two-stage QAOA uses three ensembles at $n = 6$, $m = 3$, and $|E| = 6$, generated with $\rho = 0.6$, $(a, b) = (1, 3)$, and recourse threshold $T = T_{\mathrm{med}}$, where T_{med} is the median expected cut value over the solution space, chosen as it was found to trigger recourse for a substantial fraction of solutions. To explore the extent to which the two ansätze can distinguish the nominal and stochastic optima, each ensemble consists of instances with the same number of canonical solutions ranked strictly between the stochastic optimum X^*_μ and the nominal optimum X^*_C under J_μ, which we denote as the *rank gap*. The three ensembles have $r_g = 1, 2, 3$ with 50 instances each.

The fixed-schedule calibration and test sets (see Section 5.4) contain instances with $n = 6, 7, 8$ at fixed $m = 3$ and $|E| = 10$; $m = 2, \ldots, 5$ at fixed $n = 7$ and $|E| = 8$; and $|E| = 8, 9, 10$ at fixed $n = 7$ and $m = 3$, corresponding to a total of eight unique problem settings. Instances are generated with $\rho = 0.2$, $(a, b) = (0.2, 5.0)$, and recourse threshold T given by the 85th percentile of the expected cut values over the per-instance solution space.

Each set contains equal numbers of instances in repair-depth bins $D \in \{0, 1, 2\}$. For the stochastic optimum $x \in X^*_\mu$, D is the largest value of $Q^*(x; \omega)$ over scenarios admitting a successful repair, with $D = 0$ if no repairable scenario requires a vertex move. For each problem setting $(n, m, |E|)$, the calibration and test sets each contain 60 accepted instances, with 20 instances in each repair-depth bin, giving 480 calibration and 480 test instances across the eight settings.

5.2 State-vector simulation

We use a GPU-accelerated, distributed-memory exact state-vector simulator that exploits the block structure of the phase operators and the reference-subspace structure of the QASCO mixer. The state is represented in the basis (x, ω, y), with $|X||\Omega||Y| = 2^{2n}m^{|E|}$ amplitudes when every edge has m perturbation values. The largest simulated state space, at $n = 7$, $|E| = 8$, and $m = 5$, contains $2^{14} \cdot 5^8$ complex amplitudes and requires approximately 95 GiB to store in double precision. Simulations were run on the NVIDIA GH200 nodes of Setonix at the Pawsey Supercomputing Research Centre.

During recourse-preconditioner optimisation, scenarios that induce equivalent recourse problems are grouped and represented by their combined probability weights. This preserves expectation values and normalisation over the original scenario distribution. For each QASCO mixing step, the simulator projects the joint state onto the prepared scenario–recourse reference states. Only the resulting $|X|$-dimensional vector undergoes transverse-field mixing; its change is expanded in the same reference states and added to the joint state. This implements the controlled mixer without constructing its full joint-space matrix.

For the two-stage QAOA, the first stage acts only on X, so we first evolve a $|X|$-dimensional state vector and then form $|\psi_X\rangle\,|\mu\rangle_\Omega\,|s\rangle_Y$. The second stage acts independently within each (x, ω) sector on the recourse register. The objective and marginal probabilities are obtained by summing the contributions from these sectors.

5.3 Ansatz convergence

We investigate the ability of the QASCO mixing unitary to drive convergence of the marginal distribution over the X register towards the stochastic optimum, such that solutions in X^*_μ are preferentially amplified. We assess this across the three $(n = 6, |E| = 6, m = 3)$ rank-gap ensembles described in Section 5.1, comparing the convergence of QASCO with that of the two-stage QAOA.

For QASCO, we set the recourse-preconditioner depth $p_R = p$. We use recourse-first optimisation for both ansätze: the $2p$ recourse-related parameters are first optimised to minimise the expected measured recourse cost $\langle Q \rangle$, with a uniform first-stage distribution and the target scenario distribution μ. These parameters are then held fixed while the remaining $2p$ parameters are optimised to minimise f_{joint}. For QASCO, these are the ansatz parameters; for the two-stage QAOA, they are the first-stage parameters. Both methods therefore optimise $4p$ parameters per instance and depth. The order of circuit operations is unchanged.

Optimisation is performed using the Nelder–Mead simplex method [35]. Each optimisation uses ten independent starting points at $p = 1$ and retains the best result. At each greater depth, one attempt is warm-started from the preceding depth's best parameters using the INTERP rule of [26]. We also test joint optimisation of all $4p$ two-stage QAOA parameters under f_{joint}, as originally formulated in [17], using the same depth-one restart count and INTERP initialisation. All protocols allow a maximum of 5000 iterations per optimisation attempt. For consistency with QASCO, the main comparison uses recourse-first optimisation.

Convergence is assessed from the marginal probabilities of the canonical first-stage cuts. For a canonical cut x, we denote by $P_p(x)$ the sum of the probabilities of measuring x and its bitwise complement at ansatz depth p. We first report rank-resolved probability by ordering the canonical cuts by increasing stochastic objective J_μ, so rank one corresponds to X^*_μ, and examine how probability redistributes across the best-ranked cuts as ansatz depth increases. We then compare the probability assigned to the stochastic optimum X^*_μ with

that assigned to the nominal optimum X_C^*. For each instance and depth, we first average probabilities over the optimisation repeats. We report the median of these per-instance probabilities across the 50 instances in each rank-gap ensemble, while the comparison between X_μ^* and X_C^* uses the averaged probabilities for each individual instance.

5.4 Candidate-generator benchmark

We compare fixed-schedule QASCO with uniform enumeration, random-restart greedy search, and fixed-tenure random-restart tabu search as first-stage candidate generators. For each problem setting, we configure the methods using the fixed-schedule calibration set described in Section 5.1 and then evaluate them on the corresponding test set.

Candidate costs are estimated using sample average approximation (SAA) [6]. Each run uses a fixed batch $\mathcal{B} = (\omega^{(1)}, \ldots, \omega^{(M_0)})$ of M_0 scenarios sampled with replacement from μ. For a candidate x, we use

$$\widehat{J}_{\mathcal{B}}(x) = C(x) + \frac{1}{M_0} \sum_{\ell=1}^{M_0} Q^*(x; \omega^{(\ell)}). \tag{27}$$

Each $Q^*(x;\omega)$ is evaluated exactly by searching the recourse actions in increasing Hamming distance from x until a successful repair is found or all recourse actions have been checked, with each cut evaluation counted as one query. We calibrate the SAA batch size M_0 independently for each problem setting by considering $M = 1, 2, 4, \ldots$ up to the full scenario-space size $m^{|E|}$. For each calibration instance and tested M, we use 200 independent SAA batches. For each instance, we determine the smallest M for which SAA selects a member of X_μ^* as the minimiser in at least 90% of the 200 batches, and maintains this success rate at all larger tested batch sizes. We denote this required batch size by M_{req}. Within each of the three repair-depth bins, we take the 90th percentile of these required batch sizes and set M_0 to the largest of the three. The query budget B_{q} for each problem setting is then set to the median query cost of evaluating every candidate in X_{can} once using the SAA evaluator with batch size M_0. Within a run of a candidate generator, we cache the SAA-evaluated cost for each observed candidate, so that repeated evaluations of the same candidate do not incur an additional query cost. Altogether, this design reduces the extent to which differences between candidate generators are confounded by stochastic variation in SAA accuracy or by setting-dependent candidate-evaluation cost.

We record when a stochastic-optimal candidate first completes its SAA evaluation. We compare candidate-generation performance using the expected running time (ERT) [36]. For run a, let N_a^{hit} denote the query count at which a member of X_μ^* first completes its SAA evaluation, and define the capped query cost $c_a = \min(N_a^{\mathrm{hit}}, B_{\mathrm{q}})$, such that an unsuccessful run incurs the full query budget. The ERT is then

$$\mathrm{ERT} = \frac{\sum_{a=1}^{A} c_a}{\#\{a : N_a^{\mathrm{hit}} \le B_{\mathrm{q}}\}}. \tag{28}$$

Here A is the number of runs. We use ERT when calibrating the candidate generators. To compare performance across problem settings with different query budgets, we report the hit rate h, the fraction of runs that fully evaluate a stochastic optimum, together with normalised ERT, $\mathrm{ERT}_{\mathrm{norm}}$, where normalised ERT is ERT divided by the setting's query budget B_{q}.

For each QASCO configuration (p, p_R) and problem setting, we treat each calibration instance independently, first optimising up to three linear-ramp parameters for the recourse preconditioner when $p_R > 0$ and then three for the ansatz. The phase and walk times

follow the linear-ramp parameterisation of Equation (9) throughout optimisation. We average the generated phase and walk schedules across calibration instances, then fit three common parameters for each schedule to this mean by unweighted least squares. These fitted parameters are held fixed when generating candidates on the calibration and test sets. With this schedule, first-stage candidates are generated by repeatedly sampling the X register of the prepared ansatz state, with each circuit shot producing one candidate that is mapped to its canonical representative x. For benchmark scoring, we record queries up to the first fully evaluated member of X^*_μ or exhaustion of B_{q}. Each depth-p circuit shot with recourse-preconditioner depth p_R is charged

$$N_{\mathrm{q}}^{\mathrm{shot}}(p, p_R) = 5p + 2p_R + 4pp_R, \tag{29}$$

We count one query for U_C and one for each realised-cut computation or uncomputation. Thus, $5p$ counts U_C and U_Q in the main layers, $2p_R$ counts the initial S_Y, and $4pp_R$ counts S_Y and $S_Y^\dagger$ within those layers. Each preconditioner layer uses two queries. SAA candidate evaluation is charged separately. We test $p \in \{1, \ldots, 8\}$ and $p_R \in \{0, 1, 2, 4, 8\}$ on the calibration set, selecting the configuration that minimises the ERT.

All methods use the same query budget B_{q} for each problem setting. Uniform enumeration evaluates the canonical solution space in random order without replacement. Greedy search selects the best strictly improving one-bit neighbour and restarts when no improvement is possible. Tabu search selects the best admissible one-bit neighbour, permitting worsening moves, and prohibits moves involving recently flipped bits for a fixed tenure τ, unless flipping them improves the best solution found so far [37, 38]. Both searches initialise each restart uniformly among unevaluated canonical solutions. We calibrate $\tau \in \{1, \ldots, n-2\}$ for each problem setting, choosing the tenure that minimises calibration ERT.

6 Results

We first compare QASCO and the two-stage QAOA on the three $n = 6$ rank-gap ensembles, each containing 50 instances, using recourse-first optimisation for both methods as described in Section 5.3. We average objective values and first-stage probabilities over optimisation repeats within each instance and depth, and report the median over instances with interquartile ranges where shown. We then compare QASCO with SAA-driven classical candidate generators on 480 test instances across eight problem settings. Each setting has 60 test instances and five runs per instance, giving 300 test runs per method per setting and 2400 per method overall. For hit rates, ERT and ERT ratios, we report 95% percentile bootstrap confidence intervals obtained by resampling whole instances within repair-depth bins, using the same resampled instances for all methods [39].

6.1 Comparison with two-stage QAOA

Figure 6 shows that the median optimised joint expectation value f_{joint} decreases at each simulated depth for both ansätze in all three ensembles, with QASCO consistently achieving lower values by a small margin. At $p = 5$, the QASCO medians are -10.961, -11.654, and -11.481 for $r_g = 1, 2, 3$, respectively, compared with -10.008, -11.353, and -10.955 for the two-stage QAOA.

In Figure 7, QASCO assigns the largest median probability among the six displayed ranks to the stochastic optimum at every tested depth in all three ensembles. Its median stochastic-optimum probability increases at each depth, from 0.119, 0.113, and 0.114 at

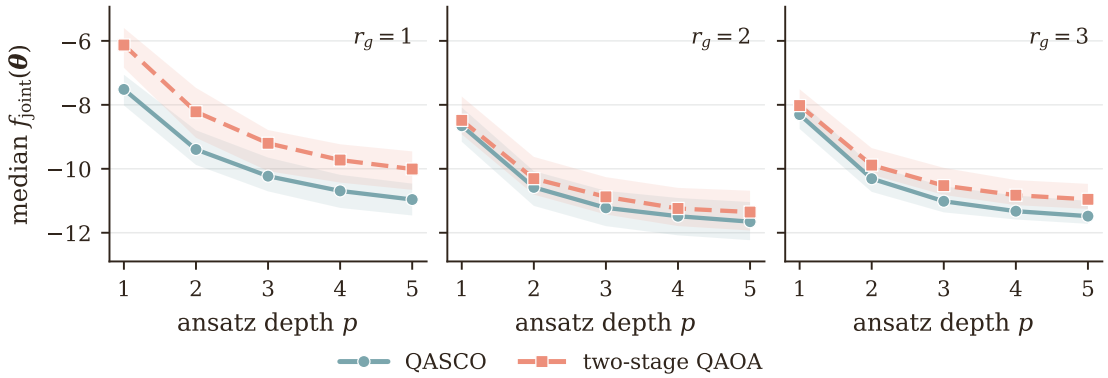


Figure 6: Median optimised joint expectation value $f_{\text{joint}}(\boldsymbol{\theta})$ versus ansatz depth p for QASCO and the two-stage QAOA.

$p = 1$ to 0.338, 0.263, and 0.283 at $p = 5$ for $r_g = 1, 2, 3$, respectively. The two-stage QAOA starts with slightly higher median stochastic-optimum probabilities of 0.131, 0.135, and 0.134, which reach 0.293, 0.235, and 0.237 at $p = 5$. However, its largest median probability remains at the nominal optimum at all considered depths.

Figure 8 compares the stochastic- and nominal-optimum probabilities within each instance at $p = 5$. QASCO assigns more probability to X^*_μ than to X^*_C in 46/50, 49/50, and 50/50 instances for $r_g = 1, 2, 3$, respectively. The corresponding counts for the two-stage QAOA are 16/50, 12/50, and 21/50. The QASCO counts increase across these ensembles, while the two-stage QAOA counts are nonmonotonic in the rank gap.

In Appendix B, we further examine the QASCO state in the reference subspace $\mathcal{S}$ and its orthogonal complement, together with their contributions to the change in f_{joint}. Across the three rank-gap ensembles, the median reference-subspace occupation P_R is 0.765 at $p = 1$ and 0.849 at $p = 5$, compared with $P_\perp = 0.235$ and 0.151, respectively. At $p = 5$, the median cumulative changes are $\Delta R_5 = -5.881$ and $\Delta I_5 = -0.434$. Overall, the state remains primarily in $\mathcal{S}$ at every tested depth, with changes in the reference contribution accounting for most of the observed reduction in f_{joint}. This is consistent with a substantial role for the surrogate objective F^μ_s in the measured convergence.

With recourse-first optimisation, the cumulative evaluation counts are similar for the two ansätze. The corresponding ensemble medians are 5310–5418.5 for QASCO and 5549–5835 for the two-stage QAOA. The classical optimiser reports successful termination for all instances. We found that joint optimisation of the two-stage QAOA variational parameters yields final median joint expectation values of -9.014, -10.382, and -9.710 for $r_g = 1, 2, 3$, respectively, all higher than those obtained with recourse-first optimisation. The corresponding median stochastic-optimum probabilities are 0.2947, 0.2354, and 0.2373, differing from the recourse-first values by less than 0.002 in each ensemble. However, recourse-first optimisation reduces cumulative objective-function evaluations through $p = 5$ by about 60% compared with the joint-optimisation ensemble medians of 13728–13948.5.

6.2 Candidate-generator benchmark

This fixed-schedule benchmark compares QASCO with classical uniform enumeration, greedy search, and tabu search on the eight $(n, m, |E|)$ problem settings described in Section 5.1. QASCO uses a calibration-selected (p, p_R) configuration for each setting, and tabu search uses a calibration-selected tenure τ.

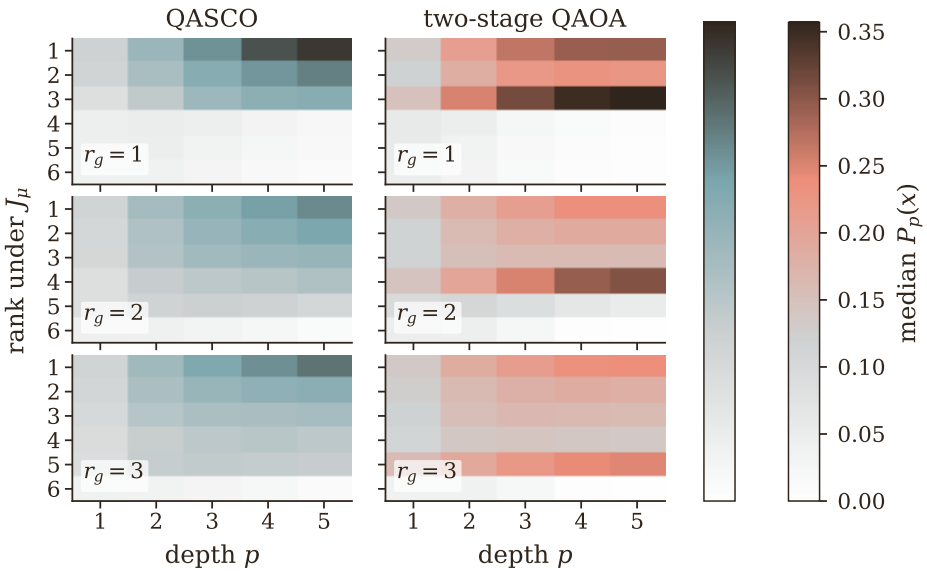


Figure 7: Median first-stage marginal probability $P_p(x)$ by rank under the stochastic objective J_μ and ansatz depth p, where rank one corresponds to the stochastic optimum. Rows are ordered by increasing rank gap r_g, with the problem instances in the first row having one solution between the stochastic optimum X^*_μ and the nominal optimum X^*_C, and the second and third rows having two and three such solutions, respectively.

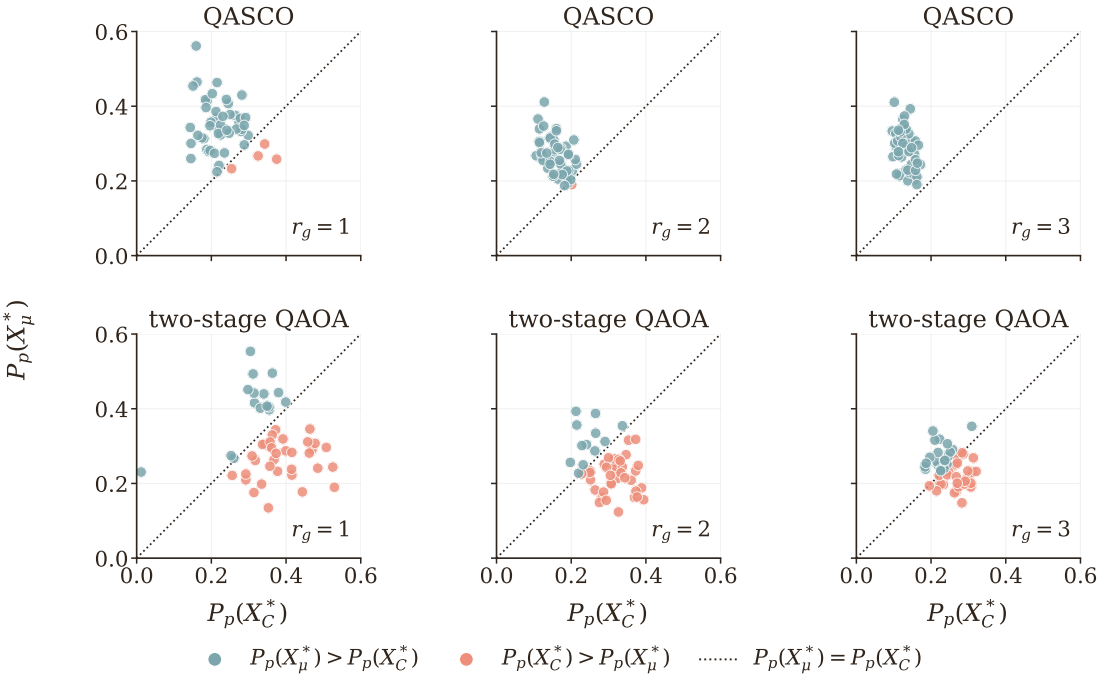


Figure 8: Probability of measuring the stochastic optimum X^*_μ versus the nominal optimum X^*_C at $p = 5$ for each of the rank-gap ensembles. Each point is one problem instance. Points above the dotted diagonal assign more probability to the stochastic optimum than to the nominal optimum.

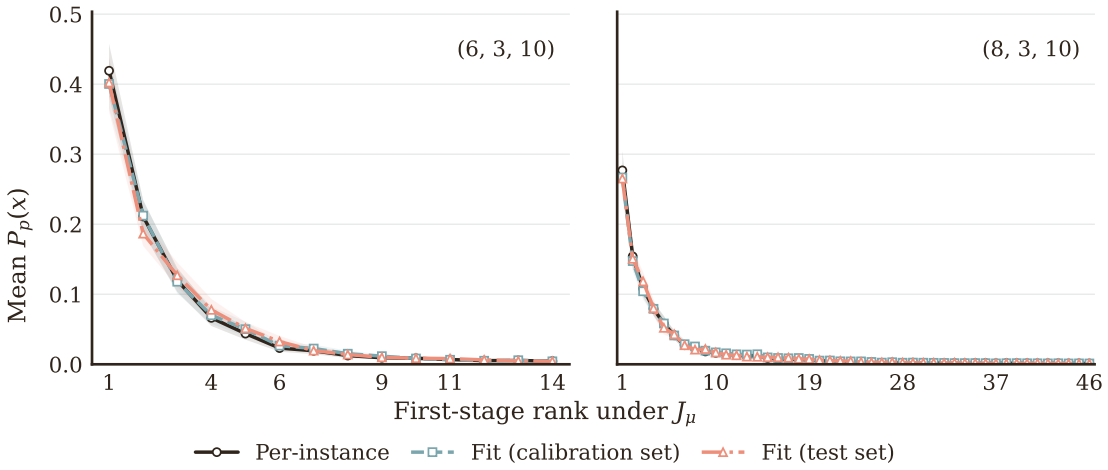


Figure 9: Mean QASCO first-stage marginal probability by rank under the stochastic objective J_μ for **(left)** $n = 6$, $m = 3$, $|E| = 10$, $(p, p_R) = (8, 0)$ and **(right)** $n = 8$, $m = 3$, $|E| = 10$, $(p, p_R) = (7, 1)$. Curves compare per-instance optimisation of the recourse-preconditioner and ansatz parameters on the calibration set with the calibrated schedule applied to the calibration and test sets. Each panel shows a range of ranks containing at least 95% of each mean distribution without renormalisation.

In Figure 9, we show representative probability distributions for the $n = 6$ and $n = 8$ problem settings with $m = 3$ and $|E| = 10$. In both cases, the calibrated schedule produces similar distributions on the calibration and test sets. For $n = 6$, mean stochastic-optimum probabilities are 0.419 with per-instance optimisation on the calibration set, 0.400 with the calibrated schedule on calibration instances, and 0.402 with the calibrated schedule on test instances. For $n = 8$, the corresponding probabilities are 0.277, 0.267, and 0.265. Across the three distributions, the eight lowest-cost cuts account for a mean total probability of 0.911–0.915 for $n = 6$ and 0.751–0.769 for $n = 8$. Across all eight settings, the test-minus-calibration difference in the ensemble mean of the stochastic objective J_μ, using the same calibrated schedule on both sets, ranges from -0.327 to $+0.587$, with all 95% confidence intervals containing zero.

Figure 10 shows that QASCO returns a stochastic optimum in all 2400 runs, with hit rate 1.000 in every problem setting. Classical hit rates range from 0.860 to 0.970 for tabu search, 0.853 to 0.950 for greedy search, and 0.780 to 0.873 for uniform enumeration. QASCO has the lowest ERT in every setting. Appendix D reports ERT values, query budgets, SAA batch sizes, and calibrated tabu tenures; Appendix C reports the QASCO depths and schedule parameters.

At fixed $m = 3$ and $|E| = 10$, the ERT ratios increase in favour of QASCO at each tested n, from 6.0 to 8.9 for tabu, 6.3 to 10.3 for greedy, and 7.8 to 22.9 for uniform enumeration between $n = 6$ and $n = 8$. Varying m with $n = 7$ and $|E| = 8$, all three ratios are largest at $m = 4$ and decrease at $m = 5$. With varying $|E|$ at fixed $n = 7$ and $m = 3$, the tabu and uniform ratios increase from 6.8 to 8.2 and 11.7 to 13.7, respectively, while the greedy ratio increases from 7.6 at $|E| = 8$ to 9.4 at $|E| = 9$, then decreases to 8.9 at $|E| = 10$.

Figure 11 groups the results by expected minimum recourse cost at the stochastic optimum. QASCO has a hit rate of 1.000 in all four groups. Tabu and greedy search also have hit rate 1.000 in the two lowest-cost groups, while uniform enumeration has hit rates 0.989 and 0.946. In the highest-cost group, $\bar{Q}^*_\mu > 2$, hit rates are 0.707 for tabu, 0.650 for greedy, and 0.432 for uniform enumeration. Normalised ERT increases across the

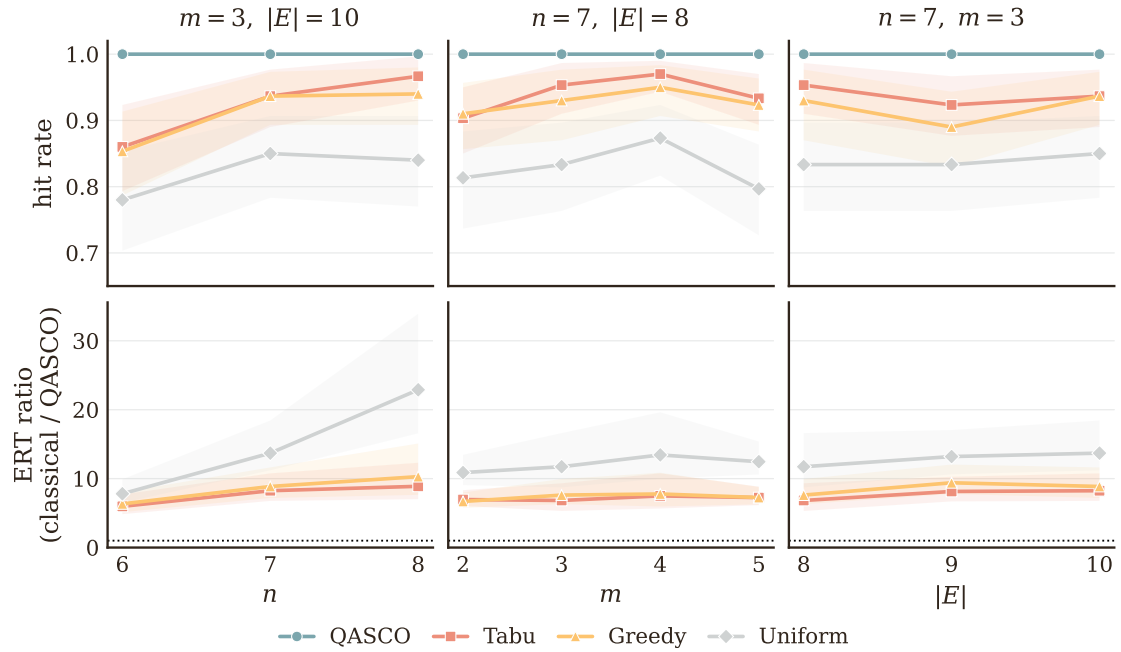


Figure 10: Hit rate h (top row) and classical-to-QASCO ERT ratio (bottom row) for the candidate-generator benchmark. Columns vary n at fixed $m = 3$, $|E| = 10$; m at fixed $n = 7$, $|E| = 8$; and $|E|$ at fixed $n = 7$, $m = 3$. ERT ratios greater than one favour QASCO, and the dotted line at one denotes equal ERT.

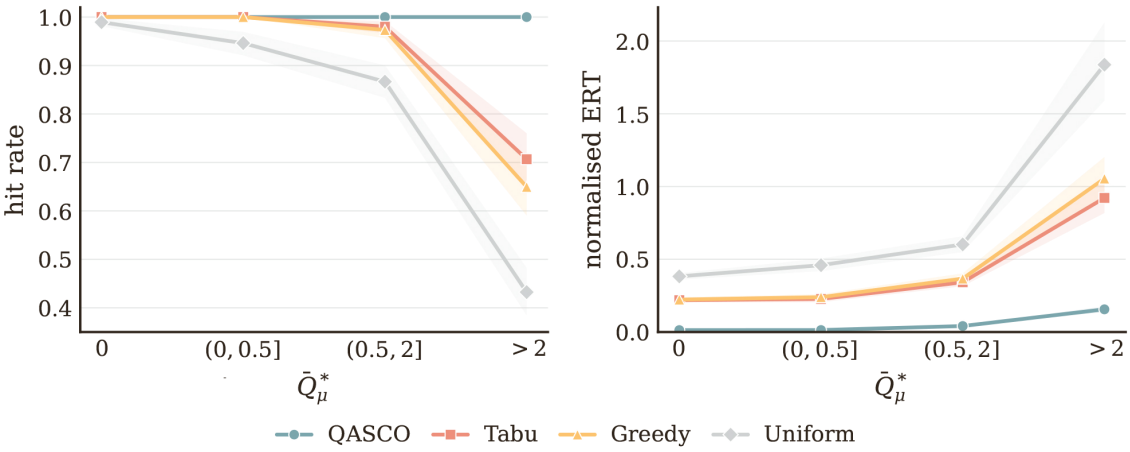


Figure 11: Candidate-generator performance grouped by expected minimum recourse cost at the stochastic optimum, $\bar{Q}^*_\mu := \mathbb{E}_{\omega\sim\mu}[Q^*(x;\omega)]$ for $x \in X^*_\mu$, pooled over the eight problem settings. The groups $\bar{Q}^*_\mu = 0$, $0 < \bar{Q}^*_\mu \le 0.5$, $0.5 < \bar{Q}^*_\mu \le 2$, and $\bar{Q}^*_\mu > 2$ contain 151, 104, 120, and 105 test instances, respectively. **Left:** Hit rate. **Right:** Normalised ERT.

four groups for all methods. Between the zero- and highest-cost groups, it increases from 0.013 to 0.156 for QASCO, 0.219 to 0.922 for tabu, 0.223 to 1.055 for greedy, and 0.382 to 1.837 for uniform enumeration. QASCO has the lowest normalised ERT in every group. We examine the query complexity of classical recourse evaluation and the SAA sampling requirements in Appendix E.

7 Discussion

The numerical results show that QASCO distinguishes stochastic-optimal first-stage decisions from the nominal optimum. In Figure 7 we see that, with variational optimisation of all $2p$ parameters, QASCO increasingly concentrates probability on the stochastic optimum under J_μ as depth increases and generally assigns it more probability than the nominal optimum. In contrast, the two-stage QAOA increasingly concentrates on the nominal optimum.

The observed preference of the two-stage QAOA for the nominal optimum may reflect the separation of its subcircuits over the first and second stages. The first-stage QAOA circuit redistributes probability using phase separation under $C(x)$, while the subsequent recourse QAOA circuit preserves the first-stage marginal [17]. Although recourse costs can still influence the optimised first-stage parameters through f_{joint}, they do not enter the first-stage phase separator, which may weaken the link between the ansatz dynamics and the stochastic optimum. In contrast, the QASCO mixing unitary aggregates amplitudes with recourse-dependent phases, so this information more directly influences the marginal distribution of the first-stage register (Equation (18)). The extra degrees of freedom enabled by the separate QAOA subcircuits of the two-stage QAOA may allow superior amplification of the stochastic optimum with appropriate parameters, but optimising the joint objective f_{joint} alone may make such parameters difficult to identify. With recourse-first optimisation, both ansätze optimise $4p$ parameters in two steps and were observed to converge consistently during classical parameter optimisation, with similar numbers of objective-function evaluations. QASCO's stronger preference for the stochastic optimum therefore persists under comparable parameter optimisation effort.

Parameter optimisation overhead and satisfying problem constraints are significant challenges for quantum variational algorithms in general [11, 40]. To this end, the modular design of the QASCO circuit lets existing QAOA and QWOA state-preparation and mixing unitaries serve as S_X and U_W, respectively, and, where appropriate, as the recourse preparation S_Y. In Section 6.2, we show that this enables a linear-ramp parameterisation scheme developed for deterministic optimisation in the QWOA framework [15, 25]. This uses three classically optimised ansatz parameters and at most three for the recourse preconditioner, requiring only the ansatz parameters when $p_R = 0$.

The distinction between the efficiently implementable joint objective f_{joint}, the stochastic objective J_μ and the surrogate objective F_s^μ remains an important consideration. Our numerical results show useful first-stage selection when optimising f_{joint}, despite the differences between these quantities. Three of the eight calibration-selected configurations use $p_R = 0$ (see Appendix C), indicating that recourse preconditioning is not always necessary for effective candidate generation when stochastic-optimal decisions remain favourable under the surrogate objective. Whether alternative recourse preconditioning strategies, such as threshold-based amplitude amplification, can improve convergence or yield a more efficient circuit realisation is an area for future investigation.

With the linear-ramp parameterisation, QASCO can prepare useful first-stage distributions without per-instance parameter optimisation on the test set. In the two settings

illustrated in Figure 9, calibrated schedules retain much of the stochastic-optimum probability and low-cost concentration achieved by per-instance optimisation on the calibration set, and produce similar cost-ranked distributions on calibration and test instances. These observations suggest that we may amortise parameter-optimisation overhead by calibrating schedule parameters offline on representative problem instances.

Under the stated query-cost model, QASCO with calibrated parameters attains the lowest ERT in every tested setting and reaches a stochastic optimum in every run. In Figure 10, the classical-to-QASCO ERT ratio increases with the size of the first-stage space, consistent with the increasing advantage reported for deterministic MaxCut over similar simple classical heuristics [28]. Classical hit rates also generally improve across this range; one possible explanation is that, as $|E|$ is held constant, the graph becomes sparser, resulting in a simpler first-stage landscape. This trend should therefore not be assumed to persist for the considered classical solvers in general.

Across the tested values of m and $|E|$, QASCO maintains its hit rate and a comparable ERT advantage over the classical methods, despite substantial variation in the calibrated SAA batch size and absolute query expenditure (see Appendix D). The QASCO mixing mechanism coherently aggregates scenario and recourse information so that its first-stage mixer acts on one reference amplitude per first-stage decision. Increasing either m or $|E|$ at fixed n enlarges the scenario space while leaving the first-stage and recourse decision spaces unchanged for stochastic MaxCut. The observed stability therefore provides evidence that QASCO's aggregation-based strategy remains effective as the scenario space grows.

We further observe in Figure 11 that the QASCO hit rate remains stable as expected minimum recourse cost at the stochastic optimum increases, while the hit rate for each classical solver decreases substantially over the same range. Higher expected minimum recourse cost can reflect both finite repair requirements and scenarios in which no repair reaches the threshold. The decline in classical hit rates may partly reflect the increased number of queries required to compute $Q^*(x;\omega)$ for each scenario in the SAA batch (Appendix E). The classical solvers rely on SAA evaluations of candidates, so this additional cost limits the search space within the query budget. For QASCO, the additional recourse evaluation cost is incurred when evaluating candidates returned by the circuit but, at fixed (p_R, p), it does not change the number of queries per circuit preparation.

Our circuit analysis identifies implementation of the recourse oracle as the primary resource challenge. Under the assumptions in Section 4, the per-layer two-qubit gate complexity is $O((1+p_R)[|E|(b+\log|E|)+n])$, with repeated recourse-oracle applications dominating the cost for fixed p_R. Explicit recourse encoding requires an n-qubit register, giving $2n+n_\Omega$ non-ancilla qubits in total. Nevertheless, this polynomial per-layer resource scaling is sufficient for the ansatz to act on a configuration space whose size grows exponentially with the number of encoded variables.

The two-stage QAOA uses $O(n)$ single-qubit gates for each transverse-field mixing step. The QASCO controlled mixer requires $O(n+n_\Omega)$ two-qubit gates, while its associated basis transformations additionally require the scenario and recourse preparation circuits and their inverses. With at most a fixed number of perturbation values per edge and $|E|=\Theta(n)$, the controlled mixer has $O(n)$ gate complexity; repeated recourse-oracle applications add $O((1+p_R)n(b+\log n))$ per layer. The additional controlled operations and oracle calls may make QASCO harder to implement on noisy intermediate-scale quantum devices. Nevertheless, for the problems studied here, our results suggest that QASCO is likely to be preferable to the two-stage QAOA when adequate quantum hardware is available. The original two-stage QAOA proposal used a trained quantum generative adversarial network (qGAN) to prepare the scenario distribution [17]. Such a method could also implement S_Ω

in QASCO when efficient for the target distribution [30].

While we optimise the schedule parameters by exact evaluation of f_{joint} here, the calibrated parameters show consistent trends with ansatz depth (see Appendix C). Parameter transfer from smaller to larger problem instances [41] or from shallower to deeper circuits [26] may therefore lower the cost of optimisation on hardware in practice. Fully shot-based parameter optimisation and optimisation in the presence of noise remain key areas for future investigation.

The successful use of the transverse-field mixer within QASCO suggests a route to stochastic optimisation problems whose first-stage search spaces admit existing QAOA or QWOA constructions. The first-stage mixing graph determines which first-stage decisions can couple, while the reference-state amplitudes determine the associated couplings over the scenario and recourse registers. Existing QAOA or QWOA state-preparation and mixing constructions that preserve feasibility [40, 42] could therefore support extensions to constrained first-stage search spaces.

8 Conclusion

We introduced Quantum Approximate Stochastic Combinatorial Optimisation (QASCO), a quantum approach to two-stage stochastic optimisation with recourse. QASCO coherently combines information about possible scenarios and recourse actions to increase the probability of measuring first-stage decisions with low cost under the stochastic objective, without explicitly evaluating their expected minimum recourse costs. We apply it to a MaxCut problem with stochastic edge weights, constructing a circuit whose gate count per layer grows polynomially with the number of problem variables.

In exact state-vector simulations, QASCO more consistently converges to first-stage decisions optimal under the stochastic objective than a previously proposed QAOA-based method under comparable parameter-optimisation effort. We apply a fixed-parameter-count schedule originally proposed for deterministic quantum approximate optimisation to QASCO and find that parameters calibrated on a set of representative instances can amplify low-cost states when applied to new problem instances without per-instance parameter optimisation. Compared to classical uniform enumeration, greedy search, and tabu search using sample average approximation of the stochastic objective, QASCO consistently reached a global optimum in the benchmarks, with a lower expected query cost under the stated model, and maintained its effectiveness across the tested increases in configuration-space size and expected minimum recourse cost at the stochastic optimum.

Studying QASCO under noise and developing end-to-end, shot-efficient parameter optimisation are natural next steps. An analytical investigation is also needed to identify the regimes in which QASCO is expected to converge effectively. Applying QASCO to additional stochastic combinatorial problems, particularly those with constrained first-stage and recourse decision spaces, will help establish its utility as a general framework for stochastic combinatorial optimisation.

Acknowledgements

This project was funded through the Department of Defence Advanced Strategic Capabilities Accelerator (ASCA). Computational resources were provided by the Setonix supercomputer system at the Pawsey Supercomputing Research Centre, which is funded by the Australian Government and the Government of Western Australia. Additional support came from Pawsey's Quantum Supercomputing Innovation Hub, made possible by a grant

from the Australian Government through the National Collaborative Research Infrastructure Strategy (NCRIS). EM also thanks the Pawsey quantum team for their support and helpful discussions throughout this work.

Author contributions

EM developed the proposed quantum walk-based optimisation method, devised and performed the numerical simulations, and prepared the initial manuscript draft. PS proposed the initial research direction, and provided expertise and guidance on problem formulation and solution approaches. JB supervised and guided the technical arguments, conclusions, and presentation of the manuscript. UV provided institutional support for the project and contributed to the manuscript's critical review. All authors contributed to writing, critically reviewing, and revising the manuscript. AI tools were used to assist with language editing, formatting and manuscript review.

References

[1] George B. Dantzig. "Linear programming under uncertainty". Management Science **1**, 197–206 (1955).

[2] John R. Birge and François Louveaux. "Introduction to stochastic programming". Springer. New York (1997).

[3] David W. Walkup and Roger J.-B. Wets. "Stochastic programs with recourse". SIAM Journal on Applied Mathematics **15**, 1299–1314 (1967).

[4] Martin Dyer and Leen Stougie. "Computational complexity of stochastic programming problems". Mathematical Programming **106**, 423–432 (2006).

[5] Grani A. Hanasusanto, Daniel Kuhn, and Wolfram Wiesemann. "A comment on 'Computational complexity of stochastic programming problems"'. Mathematical Programming **159**, 557–569 (2016).

[6] Anton J. Kleywegt, Alexander Shapiro, and Tito Homem-de-Mello. "The sample average approximation method for stochastic discrete optimization". SIAM Journal on Optimization **12**, 479–502 (2002).

[7] Justin Dumouchelle, Rahul Patel, Elias B. Khalil, and Merve Bodur. "Neur2SP: Neural two-stage stochastic programming". In Advances in Neural Information Processing Systems. Volume 35, pages 23992–24005. Curran Associates, Inc. (2022).

[8] Vinod Nair, Krishnamurthy Dvijotham, Iain Dunning, and Oriol Vinyals. "Learning fast optimizers for contextual stochastic integer programs". In Amir Globerson and Ricardo Silva, editors, Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence (UAI). Pages 591–600. AUAI Press (2018). url: `https://auai.org/uai2018/proceedings/papers/217.pdf`.

[9] Yoshua Bengio, Emma Frejinger, Andrea Lodi, Rahul Patel, and Sriram Sankaranarayanan. "A learning-based algorithm to quickly compute good primal solutions for stochastic integer programs". In Emmanuel Hebrard and Nysret Musliu, editors, Integration of Constraint Programming, Artificial Intelligence, and Operations Research. Volume 12296 of Lecture Notes in Computer Science, pages 99–111. Cham (2020). Springer.

[10] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. "A quantum approximate optimization algorithm" (2014). arXiv:1411.4028.

[11] M. Cerezo, Andrew Arrasmith, Ryan Babbush, Simon C. Benjamin, Suguru Endo, Keisuke Fujii, Jarrod R. McClean, Kosuke Mitarai, Xiao Yuan, Lukasz Cincio, and

Patrick J. Coles. "Variational quantum algorithms". Nature Reviews Physics **3**, 625–644 (2021).

[12] Samuel Marsh and J. B. Wang. "A quantum walk-assisted approximate algorithm for bounded NP optimisation problems". Quantum Information Processing **18**, 61 (2019).

[13] Samuel Marsh and Jingbo B. Wang. "Combinatorial optimization via highly efficient quantum walks". Physical Review Research **2**, 023302 (2020).

[14] Edric Matwiejew, Jason Pye, and Jingbo B. Wang. "Quantum optimisation for continuous multivariable functions by a structured search". Quantum Science and Technology **8**, 045013 (2023).

[15] Tavis Bennett, Lyle Noakes, and Jingbo B. Wang. "Analysis of the non-variational quantum walk-based optimisation algorithm" (2024). arXiv:2408.06368.

[16] Caleb Rotello, Peter Graf, Matthew Reynolds, Eric B. Jones, Cody James Winkleblack, and Wesley Jones. "Calculating the expected value function of a two-stage stochastic optimization program with a quantum algorithm" (2024). arXiv:2402.15029.

[17] Taihei Kuroiwa, Daiki Yamazaki, Keita Takahashi, Kodai Shiba, Chih-Chieh Chen, and Tomah Sogabe. "Quantum-circuit framework for two-stage stochastic programming via QAOA integrated with a quantum generative neural network" (2025). arXiv:2512.22434.

[18] Christoph Grüne. "The complexity classes of Hamming distance recoverable robust problems". In José A. Soto and Andreas Wiese, editors, LATIN 2024: Theoretical Informatics. Volume 14578 of Lecture Notes in Computer Science, pages 321–335. Cham (2024). Springer.

[19] Moira MacNeil and Merve Bodur. "Leveraging decision diagrams to solve two-stage stochastic programs with binary recourse and logical linking constraints". European Journal of Operational Research **315**, 228–241 (2024).

[20] Abraham Charnes and William W. Cooper. "Chance-constrained programming". Management Science **6**, 73–79 (1959).

[21] Evdokia Nikolova, Matthew Brand, and David R. Karger. "Optimal route planning under uncertainty". In Proceedings of the Sixteenth International Conference on Automated Planning and Scheduling (ICAPS). Pages 131–140. AAAI Press (2006). url: https://cdn.aaai.org/ICAPS/2006/ICAPS06-014.pdf.

[22] Evdokia Nikolova. "Approximation algorithms for reliable stochastic combinatorial optimization". In Maria Serna, Ronen Shaltiel, Klaus Jansen, and José Rolim, editors, Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques. Volume 6302 of Lecture Notes in Computer Science, pages 338–351. Berlin, Heidelberg (2010). Springer.

[23] John Preskill. "Quantum computing in the NISQ era and beyond". Quantum **2**, 79 (2018).

[24] Kia Manouchehri and Jingbo Wang. "Physical implementation of quantum walks". Springer. Berlin, Heidelberg (2014).

[25] Tavis Bennett, Lyle Noakes, and Jingbo Wang. "Non-variational quantum combinatorial optimisation". In 2024 IEEE International Conference on Quantum Computing and Engineering (QCE). Pages 31–41. IEEE (2024).

[26] Leo Zhou, Sheng-Tao Wang, Soonwon Choi, Hannes Pichler, and Mikhail D. Lukin. "Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices". Physical Review X **10**, 021067 (2020).

[27] Edric Matwiejew, Aidan Smith, Callum Neill, Paulo Santos, Ugo Varetto, and Jingbo Wang. "Quantum walk-based optimisation for capacitated vehicle routing with homogeneous and heterogeneous fleets" (2026). arXiv:2606.12856.

[28] Tavis Bennett, Aidan Smith, Edric Matwiejew, and Jingbo B. Wang. "Benchmarking quantum heuristics: Nonvariational quantum-walk-based optimization algorithm for the weighted MaxCut problem". Physical Review A **113**, 032603 (2026).
[29] Lov Grover and Terry Rudolph. "Creating superpositions that correspond to efficiently integrable probability distributions" (2002). arXiv:quant-ph/0208112.
[30] Christa Zoufal, Aurélien Lucchi, and Stefan Woerner. "Quantum generative adversarial networks for learning and loading random distributions". npj Quantum Information **5**, 103 (2019).
[31] Mikko Möttönen, Juha J. Vartiainen, Ville Bergholm, and Martti M. Salomaa. "Transformation of quantum states using uniformly controlled rotations". Quantum Information and Computation **5**, 467–473 (2005).
[32] Gilles Brassard, Peter Høyer, Michele Mosca, and Alain Tapp. "Quantum amplitude amplification and estimation". In Samuel J. Lomonaco, Jr. and Howard E. Brandt, editors, Quantum Computation and Information. Volume 305 of Contemporary Mathematics, pages 53–74. American Mathematical Society, Providence, RI (2002).
[33] Adriano Barenco, Charles H. Bennett, Richard Cleve, David P. DiVincenzo, Norman Margolus, Peter Shor, Tycho Sleator, John A. Smolin, and Harald Weinfurter. "Elementary gates for quantum computation". Physical Review A **52**, 3457–3467 (1995).
[34] Steven A. Cuccaro, Thomas G. Draper, Samuel A. Kutin, and David Petrie Moulton. "A new quantum ripple-carry addition circuit" (2004). arXiv:quant-ph/0410184.
[35] John A. Nelder and Roger Mead. "A simplex method for function minimization". The Computer Journal **7**, 308–313 (1965).
[36] Nikolaus Hansen, Anne Auger, Raymond Ros, Olaf Mersmann, Tea Tušar, and Dimo Brockhoff. "COCO: a platform for comparing continuous optimizers in a black-box setting". Optimization Methods and Software **36**, 114–144 (2021).
[37] Fred Glover. "Tabu search—part II". ORSA Journal on Computing **2**, 4–32 (1990).
[38] Ankur Nath and Alan Kuhnle. "MaxCutBench: Revisiting and benchmarking graph neural networks for maximum cut". Transactions on Machine Learning Research (2025). url: https://openreview.net/forum?id=322PpCGAX8.
[39] Bradley Efron and Robert J. Tibshirani. "An introduction to the bootstrap". Chapman & Hall/CRC. Boca Raton, FL (1994).
[40] Franz Georg Fuchs, Kjetil Olsen Lye, Halvor Møll Nilsen, Alexander Johannes Stasik, and Giorgio Sartor. "Constraint preserving mixers for the quantum approximate optimization algorithm". Algorithms **15**, 202 (2022).
[41] Alexey Galda, Xiaoyuan Liu, Danylo Lykov, Yuri Alexeev, and Ilya Safro. "Transferability of optimal QAOA parameters between random graphs". In 2021 IEEE International Conference on Quantum Computing and Engineering (QCE). Pages 171–180. IEEE (2021).
[42] Stuart Hadfield, Zhihui Wang, Bryan O'Gorman, Eleanor G. Rieffel, Davide Venturelli, and Rupak Biswas. "From the quantum approximate optimization algorithm to a quantum alternating operator ansatz". Algorithms **12**, 34 (2019).

A Independent uncertain parameters

Let $\omega_0, \ldots, \omega_{d_\omega - 1}$ be mutually independent uncertain parameters. For each $i \in \{0, \ldots, d_\omega - 1\}$, let Ω_i denote the support of ω_i and let $\mu_i(a) = \mathbb{P}[\omega_i = a]$, for $a \in \Omega_i$, be its marginal distribution. The scenario space factorises as $\Omega = \Omega_0 \times \cdots \times \Omega_{d_\omega - 1}$, and the joint distribution

satisfies

$$\mu(\omega) = \prod_{i=0}^{d_\omega - 1} \mu_i(\omega_i), \qquad \omega = (\omega_0, \ldots, \omega_{d_\omega - 1}) \in \Omega.$$

In particular, a general distribution on Ω requires $|\Omega| - 1$ free probability parameters, whereas the above product-form model requires only $\sum_{i=0}^{d_\omega - 1}(|\Omega_i| - 1)$. If each uncertain parameter has support satisfying $2 \le |\Omega_i| \le \kappa$ for a constant κ independent of d_ω, then $\log|\Omega| = \Theta(d_\omega)$ and the product model requires $O(d_\omega)$ probability parameters.

B Contributions to the ansatz objective

For the $n = 6$, $m = 3$, $|E| = 6$ instances considered in Section 6.1, we write $|\psi\rangle = |\psi_R\rangle + |\psi_\perp\rangle$, with reference component $|\psi_R\rangle = \sum_x b_x |x, r_x\rangle$ and $|\psi_\perp\rangle$ orthogonal to $\mathcal{S}$. The squared projection norms of each final depth-p state are $P_R = \sum_x |b_x|^2$ and $P_\perp = 1 - P_R$.

With respect to the cost operator $\hat{H}$ (see Section 3.5), the reference and interference contributions are

$$R := \langle\psi_R| \hat{H} |\psi_R\rangle = \sum_x |b_x|^2 F_s^\mu(x), \qquad I := 2\,\mathrm{Re}\,\langle\psi_R| \hat{H} |\psi_\perp\rangle .$$

We compute the cumulative changes ΔR_p and ΔI_p in these contributions over the p mixing steps. Figure 12 summarises these quantities across the three rank-gap ensembles. The median P_R increases from 0.765 at $p = 1$ to 0.865 at $p = 2$ and is 0.849 at $p = 5$.

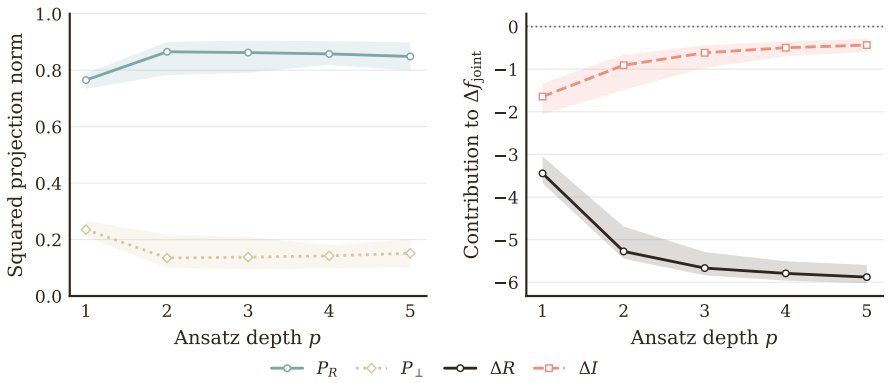


Figure 12: Reference-subspace occupation and objective contributions across the three rank-gap ensembles. **Left**: squared projection norms P_R and $P_\perp$ of the final state. **Right**: cumulative changes ΔR_p and ΔI_p in the reference and interference contributions, where negative values reduce f_{joint}. Lines show medians over the 150 instances and bands show interquartile ranges.

C Calibrated schedule parameters

Table 1 reports the selected QASCO depths and calibrated linear-ramp parameters for the eight problem settings in the candidate-generator benchmark of Section 6.2. Both schedules use Equation (9), except at $p_R = 1$, where the recourse phase and walk times are $\gamma_{R,\max}$ and $t_{R,\max}$ and λ_R is inactive. These parameters are fixed after calibration and used for all test instances in each setting.

Figure 13 shows the calibrated ansatz parameters across $p = 1, \ldots, 8$, holding p_R at its selected value for each setting. From $p = 2$ to $p = 8$, λ decreases monotonically in every

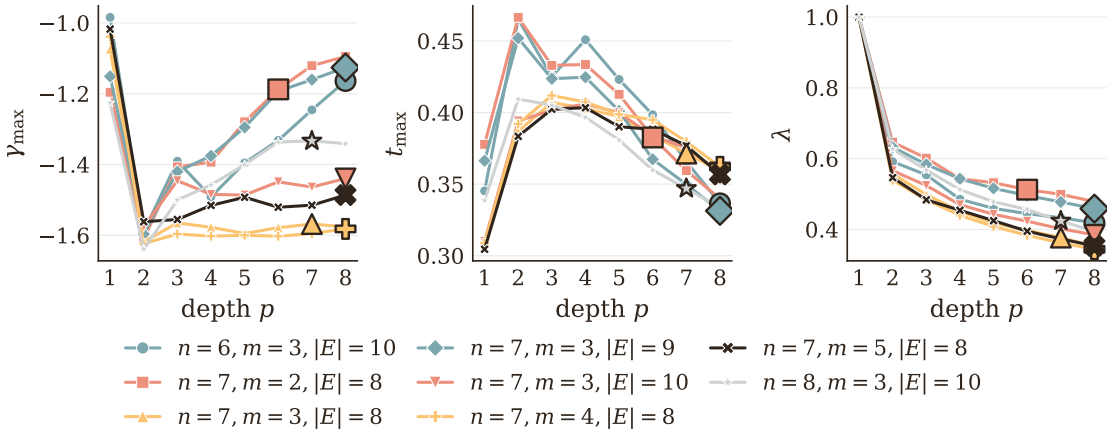


Figure 13: Calibrated ansatz linear-ramp parameters versus ansatz depth p for the eight problem settings. **Left**: $\gamma_{\max}$. **Middle**: $t_{\max}$. **Right**: λ. For each curve, the recourse-preconditioner depth p_R is fixed at the calibration-selected value (see Appendix C). Larger outlined markers indicate the selected ansatz depth.

Table 1: Calibration-selected QASCO configuration for each problem setting: ansatz depth p and schedule $(\gamma_{\max}, t_{\max}, \lambda)$, and recourse-preconditioner depth p_R and schedule $(\gamma_{R,\max}, t_{R,\max}, \lambda_R)$. Dashes indicate that no recourse preconditioner was selected ($p_R = 0$).

Problem setting	p	$\gamma_{\max}$	$t_{\max}$	λ	p_R	$\gamma_{R,\max}$	$t_{R,\max}$	λ_R
$n = 6, m = 3, \|E\| = 10$	8	-1.164	0.337	0.418	0	–	–	–
$n = 7, m = 2, \|E\| = 8$	6	-1.188	0.383	0.512	0	–	–	–
$n = 7, m = 3, \|E\| = 8$	7	-1.568	0.371	0.377	8	-1.376	0.305	0.658
$n = 7, m = 3, \|E\| = 9$	8	-1.126	0.331	0.459	0	–	–	–
$n = 7, m = 3, \|E\| = 10$	8	-1.440	0.358	0.385	4	-1.526	0.339	0.799
$n = 7, m = 4, \|E\| = 8$	8	-1.582	0.362	0.344	8	-1.330	0.305	0.641
$n = 7, m = 5, \|E\| = 8$	8	-1.486	0.357	0.353	8	-1.248	0.305	0.649
$n = 8, m = 3, \|E\| = 10$	7	-1.333	0.347	0.424	1	-1.456	0.298	0.999

setting, from 0.539–0.647 to 0.344–0.478. The mixer parameter $t_{\max}$ reaches its maximum at $p = 2$, 3, or 4 and decreases monotonically from $p = 4$ onwards in every setting, reaching 0.331–0.362 at $p = 8$. The phase parameter $\gamma_{\max}$ is most negative at $p = 2$ in every setting, but its subsequent variation differs between settings. At $p = 8$ it ranges from -1.582 to -1.095. The selected ansatz depths lie between 6 and 8.

D Candidate-generator results

Table 2 reports hit rate and ERT with 95% confidence intervals for the eight problem settings in the candidate-generator benchmark of Section 6.2, together with the query budget B_q, SAA batch size M_0 and selected tabu tenures. QASCO depths and parameter schedules are reported in Appendix C.

Table 2: Candidate-generator benchmark results. QASCO uses the calibration-selected depths and schedules in Appendix C; tabu search uses the calibration-selected tenure τ. Each method has 300 runs per setting (60 instances $\times$ five runs), or 2400 overall; hit rate and ERT include unsuccessful runs. The 95% percentile bootstrap confidence intervals are reported as relative percentage offsets. Hit-rate intervals are omitted when every run succeeded.

Setting $(n, m, \lvert E\rvert)$	Method	Hit rate	ERT (queries)	B_{q}	M_0
$(6, 3, 10)$	QASCO	1.000	127710 $[-29.2\%, +30.4\%]$	1416256	4096
	Tabu ($\tau = 3$)	0.860 $[-7.8\%, +7.3\%]$	763187 $[-17.2\%, +21.4\%]$		
	Greedy	0.853 $[-7.7\%, +7.0\%]$	809119 $[-17.0\%, +20.1\%]$		
	Uniform	0.780 $[-9.9\%, +9.9\%]$	997056 $[-17.5\%, +21.1\%]$		
$(7, 2, 8)$	QASCO	1.000	3498 $[-22.5\%, +25.9\%]$	58758	64
	Tabu ($\tau = 4$)	0.903 $[-5.9\%, +5.2\%]$	24362 $[-16.6\%, +19.1\%]$		
	Greedy	0.910 $[-5.8\%, +4.7\%]$	23390 $[-16.1\%, +21.1\%]$		
	Uniform	0.813 $[-8.6\%, +8.2\%]$	38049 $[-16.2\%, +19.9\%]$		
$(7, 3, 8)$	QASCO	1.000	53669 $[-33.9\%, +38.8\%]$	1043547	1024
	Tabu ($\tau = 3$)	0.953 $[-3.8\%, +3.6\%]$	366599 $[-15.0\%, +16.8\%]$		
	Greedy	0.930 $[-5.7\%, +5.1\%]$	408522 $[-17.1\%, +19.7\%]$		
	Uniform	0.833 $[-7.6\%, +7.2\%]$	629430 $[-14.6\%, +18.5\%]$		
$(7, 3, 9)$	QASCO	1.000	35569 $[-28.9\%, +34.1\%]$	777398	512
	Tabu ($\tau = 1$)	0.923 $[-5.0\%, +4.3\%]$	289528 $[-17.4\%, +21.0\%]$		
	Greedy	0.890 $[-6.4\%, +6.0\%]$	334326 $[-18.3\%, +23.0\%]$		
	Uniform	0.833 $[-7.6\%, +6.8\%]$	469639 $[-15.2\%, +19.5\%]$		
$(7, 3, 10)$	QASCO	1.000	61476 $[-30.1\%, +34.5\%]$	1467495	1024
	Tabu ($\tau = 1$)	0.937 $[-4.7\%, +3.5\%]$	506957 $[-16.2\%, +19.0\%]$		
	Greedy	0.937 $[-4.3\%, +3.8\%]$	544783 $[-16.0\%, +18.2\%]$		
	Uniform	0.850 $[-6.7\%, +6.7\%]$	842200 $[-14.9\%, +17.5\%]$		
$(7, 4, 8)$	QASCO	1.000	415740 $[-33.3\%, +38.0\%]$	9527076	8192
	Tabu ($\tau = 4$)	0.970 $[-2.8\%, +2.1\%]$	3115811 $[-13.4\%, +14.0\%]$		
	Greedy	0.950 $[-3.9\%, +3.5\%]$	3232464 $[-14.0\%, +15.3\%]$		
	Uniform	0.873 $[-5.7\%, +5.0\%]$	5588382 $[-11.5\%, +13.3\%]$		
$(7, 5, 8)$	QASCO	1.000	91350 $[-26.2\%, +31.4\%]$	1699004	1024
	Tabu ($\tau = 3$)	0.933 $[-4.6\%, +4.0\%]$	661872 $[-17.1\%, +20.2\%]$		
	Greedy	0.923 $[-4.3\%, +4.0\%]$	662992 $[-16.1\%, +19.2\%]$		
	Uniform	0.797 $[-8.4\%, +7.9\%]$	1137803 $[-15.1\%, +18.2\%]$		
$(8, 3, 10)$	QASCO	1.000	33762 $[-36.4\%, +46.3\%]$	1238403	256
	Tabu ($\tau = 4$)	0.967 $[-3.8\%, +2.7\%]$	300002 $[-18.3\%, +22.7\%]$		
	Greedy	0.940 $[-4.6\%, +3.9\%]$	347394 $[-19.4\%, +24.1\%]$		
	Uniform	0.840 $[-7.5\%, +7.5\%]$	772833 $[-15.0\%, +16.8\%]$		

E Recourse evaluation cost and SAA batch sizes

Figure 14 examines recourse evaluation cost and required SAA batch sizes for the calibration and test ensembles used in the candidate-generator benchmark of Section 6.2.

For each test instance, we use the benchmark evaluator of Section 5.4 to count the queries for threshold checks at the stochastic optimum, averaged over the scenarios in its five SAA batches. Each query evaluates a cut and tests whether its realised value meets or exceeds the threshold T. This measures per-scenario recourse evaluation cost before reuse of duplicate scenarios, excluding first-stage evaluation and candidate generation. Higher expected minimum recourse cost is associated with threshold-check queries, with mean

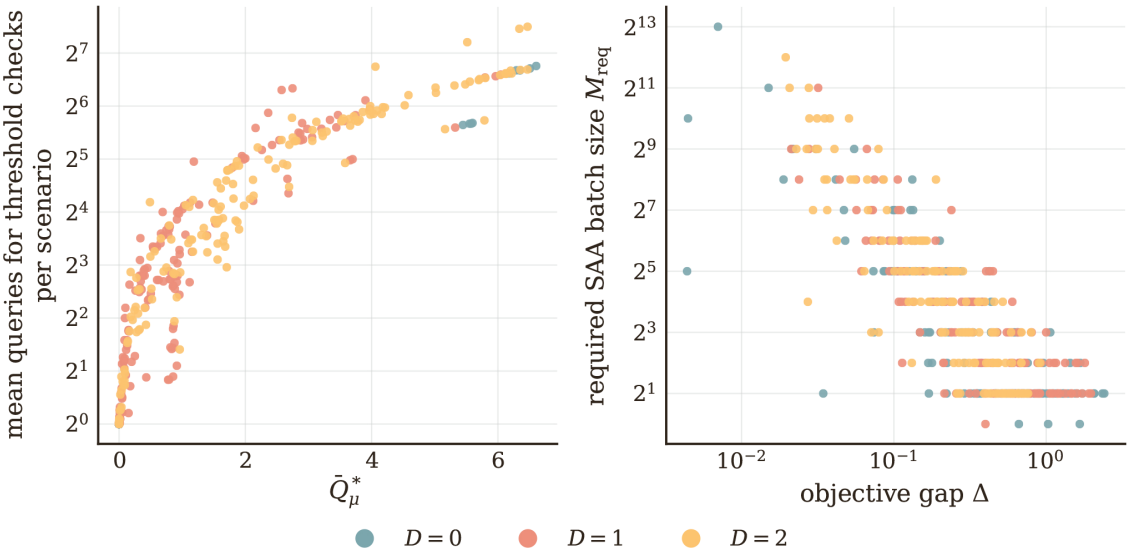


Figure 14: Recourse evaluation cost and required SAA batch sizes across the eight benchmark settings. **Left**: mean queries per sampled scenario for threshold checks at the stochastic optimum versus expected minimum recourse cost $\bar{Q}^*_\mu$, for the 480 test instances. **Right**: required SAA batch size M_{req} versus objective gap Δ, for the 480 calibration instances. Colours indicate the repair-depth bins $D = 0, 1, 2$ defined in Section 5.1.

counts ranging from 1 to 180.6. This relationship is consistent with the lower classical hit rates in the highest expected-cost group of Figure 11.

For each calibration instance, the objective gap Δ is the amount by which the stochastic objective J_μ of the best nonoptimal canonical solution exceeds the optimal value. We obtain the required SAA batch size M_{req} for each instance using the SAA calibration criterion in Section 5.4, then aggregate these into the calibrated batch size M_0 for each setting. Larger objective gaps require smaller SAA batch sizes.