Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Adaptive Shot Allocation for Recursive QAOA via Reinforcement Learning

Euimin Lee

euimin97@yonsei.ac.kr

Yonsei University

Shiho Kim

Yonsei University

Research Article

Keywords: Recursive Quantum Approximate Optimization Algorithm; Reinforcement learning; Shot allocation; Measurement efficiency

Posted Date: October 27th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9313274/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Adaptive Shot Allocation for Recursive QAOA via Reinforcement Learning

Euimin Lee[1,2*] and Shiho Kim[1,2*]

[1]School of Integrated Technology, Yonsei University, 03722, Seoul, Korea.
[2]BK21 Graduate Program in Intelligent Semiconductor Technology, Yonsei University, 21983, Incheon, Korea.

*Corresponding author(s). E-mail(s): euimin97@yonsei.ac.kr; shiho@yonsei.ac.kr;

Abstract

Recursive QAOA (RQAOA) solves combinatorial optimization problems by using shallow quantum circuits to estimate pairwise correlations and recursively eliminate variables until a classical solver can handle the residual instance. Each elimination step requires measurement shots, and the total shot cost grows with the number of recursive stages. On near-term quantum devices, increasing shot counts can translate directly into greater exposure to hardware-level noise sources such as readout errors and decoherence, making shot-efficient execution not merely a cost-reduction measure but a factor with direct implications for solution reliability. While shot reduction has been studied broadly across NISQ algorithms, step-wise measurement control inside the recursive loop of RQAOA has received little attention. We formulate this step-wise allocation as a sequential decision problem and propose two complementary strategies for depth-1 RQAOA on weighted Max-Cut instances. A hand-crafted heuristic assigns shots based on local indicators of step difficulty, and a tabular Double Q-learning agent learns a residual policy that adjusts this baseline under a Lagrangian-constrained objective. Both methods are evaluated under a fixed-cap fairness protocol that equalizes the per-step budget across all strategies, and the elimination rule itself is kept unchanged so that the contribution of adaptive measurement control can be isolated. On a diverse set of weighted graph instances spanning a range of sizes and structures, the heuristic reduces total shots by approximately 23% relative to uniform allocation, and the RL policy achieves a 36% reduction with a lower effective shots per success ratio than both baselines. The improvement persists on problem sizes not seen during training, suggesting that reinforcement learning can serve as an effective tool for discovering efficient, instance-adaptive measurement strategies in recursive quantum optimization.

Keywords: Recursive Quantum Approximate Optimization Algorithm; Reinforcement learning; Shot allocation; Measurement efficiency

1 Introduction

Quantum computing has opened new possibilities for optimization[1, 2], simulation[3, 4], and learning[5] by exploiting quantum-mechanical features such as superposition, entanglement, and interference[6]. In the noisy intermediate-scale quantum (NISQ) regime[7], hybrid quantum–classical workflows—most prominently variational quantum algorithms[8, 9]—have become a central design paradigm because they combine shallow circuits with classical outer-loop control rather than relying on full fault tolerance[10–12].

Among these methods, the Quantum Approximate Optimization Algorithm (QAOA) has attracted sustained attention as a variational approach tailored to combinatorial optimization[13–18]. At the same time, its practical performance depends on circuit depth, parameter optimization, and the statistical quality of finite-shot expectation estimates. A substantial theoretical literature has also shown that low-depth QAOA can face locality [19–22] and trainability-induced limitations[23–26] on important instance families. This motivates hybrid variants that retain shallow circuits while changing the algorithmic structure around them.

Recursive QAOA (RQAOA)[27] is one such variant. Rather than attempting to solve the full optimization problem in one variational pass, RQAOA repeatedly estimates pairwise correlations from a shallow QAOA state, fixes a high-confidence relation between variables, contracts the instance, and then repeats the procedure on the reduced problem[28–31]. This recursive structure can partially bypass some locality limitations of low-depth QAOA, but it introduces a different resource bottleneck: each elimination step requires measurement shots, and the cumulative cost can become substantial.

Because each elimination step requires its own set of circuit executions, the total shot cost compounds across the recursive chain. On near-term quantum devices, this cumulative measurement burden increases exposure to hardware-level noise sources such as readout errors and decoherence[32, 33], making shot-efficient execution not merely a cost-reduction measure but a factor with direct implications for solution reliability. A broad literature has developed around shot-frugal variational optimization and adaptive gradient estimation[34–40], operator grouping and measurement reduction[41–46], and AI driven shot reduction strategies[47]. However, most of this work addresses parameter-update loops in VQE or gradient-based settings. In RQAOA, measurements support discrete elimination decisions, and each decision changes the reduced instance that determines the next state. Recursion-level shot control is therefore a distinct sequential decision problem.

The key observation motivating this work is that not all elimination steps are equally demanding. When the leading pairwise correlation is well separated from the next candidate, even a modest number of shots suffices to identify the correct edge to contract. Conversely, when two or more correlations are nearly tied, substantially

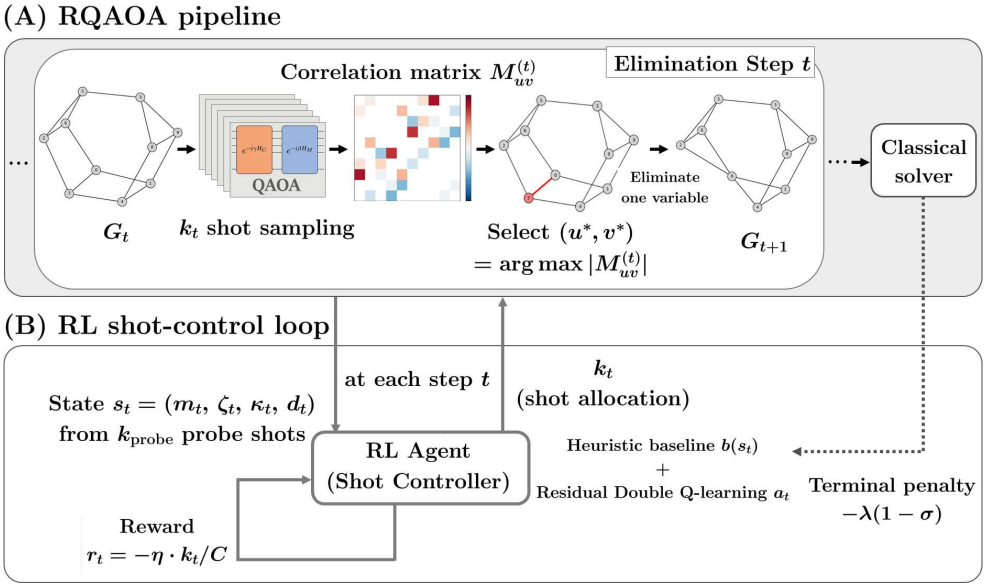


Fig. 1 Overview of the proposed shot-control framework for depth-1 RQAOA. **(A)** At each elimination step t, a depth-1 QAOA circuit is executed on the current reduced graph G_t using k_t measurement shots to estimate the pairwise correlation matrix $M_{uv}^{(t)}$. The edge (u^*, v^*) with the largest absolute correlation is selected, and the corresponding variable is eliminated to produce the next reduced graph G_{t+1}. This procedure repeats until the problem size falls below a classical-solver threshold n_c, at which point the residual instance is solved exactly. **(B)** The RL shot-control loop operates alongside the RQAOA pipeline. Before each elimination step, a small probe measurement of k_{probe} shots extracts a four-dimensional state descriptor $s_t = (m_t, \zeta_t, \kappa_t, d_t)$ that encodes the number of remaining variables, the z-gap between the two leading correlations, the conflict ratio among top candidate edges, and the graph distance between them. A tabular Double Q-learning agent takes this state as input and outputs a residual correction a_t to a hand-crafted heuristic baseline $b(s_t)$, which together determine the shot allocation k_t for that step. The agent receives a per-step cost $r_t = -\eta \cdot k_t/C$ proportional to the fraction of the shot cap used, and a terminal penalty $-\lambda(1-\sigma)$ that enforces a success-rate constraint via an adaptive Lagrangian multiplier.

more shots may be needed to resolve the ambiguity. This variation in per-step difficulty suggests that a uniform shot budget across all steps is wasteful and that an allocation strategy responsive to the local correlation structure can reduce the total measurement cost without degrading solution quality.

We address this problem by proposing two allocation strategies. A state-aware Heuristic assigns shots based on local indicators of step difficulty, providing an interpretable rule that already achieves meaningful savings on its own. Separately, a tabular Double Q-learning agent learns a residual policy anchored to the Heuristic output, adjusting it where the fixed thresholds are too conservative or too aggressive. Both strategies keep the elimination rule fixed and control only the measurement budget, unlike methods that learn the elimination rule itself[29] or the recursive structure[30]. This separation isolates the contribution of adaptive shot allocation and allows the

learned policy to be reused across instances without retraining the solver logic, building on the broader body of RL-based quantum optimization[48–50] and circuit-design work[51–53].

The contributions of this work are fourfold. First, we formulate step-wise shot allocation in depth-1 RQAOA as a measurement-control problem over the recursive chain. Second, we propose a state-aware Heuristic that provides an interpretable standalone allocation rule, together with a tabular residual Double Q-learning controller that refines its output. Third, on held-out test instances, the Heuristic reduces shot usage by approximately 23% relative to Uniform allocation, and the RL policy achieves about 36% reduction with a lower effective shots per success than both baselines. Fourth, the learned policy captures transferable step-level corrections that generalize to unseen problem sizes within the hard d-regular weighted Max-Cut family, indicating that the state representation encodes local step difficulty rather than a narrow training-size rule.

The remainder of this paper is organized as follows. Section 2 reviews the problem setting and related prior work. Section 3 presents the proposed shot-control framework. Section 4 describes the experimental setup. Section 5 reports the evaluation results, and Section 6 discusses interpretation and limitations. Finally, Section 7 concludes.

2 Background

2.1 Weighted Max-Cut and Ising formulation

We consider the weighted Max-Cut problem on a graph $G = (V, E, w)$. Let each vertex $u \in V$ be associated with a binary spin variable $z_u \in \{-1, +1\}$. Up to an additive constant, the weighted Max-Cut objective can be written in Ising form as

$$H_C = \sum_{(u,v)\in E} J_{uv} Z_u Z_v, \tag{1}$$

where Z_u denotes the Pauli-Z operator associated with vertex u, and J_{uv} is the coefficient induced by the edge weight[54, 55]. Max-Cut is NP-hard in general, and the best known polynomial-time approximation ratio is approximately 0.878[55]. In the weighted setting, the heterogeneity of edge coefficients J_{uv} amplifies the sensitivity of each elimination decision in recursive solvers, since a single misidentified correlation can propagate through edges of very different magnitudes. In this work, we focus on the weighted Ising setting without external field terms; the goal is to identify low-energy configurations of H_C, which correspond to high-quality cuts of the graph.

2.2 Depth-1 QAOA and recursive reduction in RQAOA

QAOA prepares a variational quantum state by alternating applications of the cost Hamiltonian H_C and a mixer Hamiltonian

$$H_M = \sum_{u\in V} X_u. \tag{2}$$

A depth-p QAOA state is

$$|\psi_p(\boldsymbol{\gamma},\boldsymbol{\beta})\rangle = \prod_{k=1}^{p} e^{-i\beta_k H_M} e^{-i\gamma_k H_C} |+\rangle^{\otimes n}. \tag{3}$$

In the present work we focus on the $p = 1$ setting. Depth-1 expectation values admit efficient classical simulation via closed-form expressions for the Ising cost function[27, 56], enabling large-scale benchmarking that would be infeasible at higher depths, and this restriction provides a controlled testbed for isolating recursion-level measurement effects from confounding factors such as deeper-circuit trainability[27, 29]. Accordingly, we intentionally restrict attention here to the noiseless $p = 1$ regime so that recursion-level shot allocation can be analyzed cleanly, without conflating it with hardware-noise effects or the additional performance variability introduced by optimizing deeper QAOA subroutines; these extensions are left to future work.

RQAOA uses this shallow QAOA state not as a one-shot solver, but as a subroutine inside a recursive elimination loop. Given the current reduced Hamiltonian H_t, we prepare $|\psi_1(\gamma_t,\beta_t)\rangle$ and estimate the pairwise correlations

$$M_{uv}^{(t)} = \langle\psi_1(\gamma_t,\beta_t)|Z_u Z_v|\psi_1(\gamma_t,\beta_t)\rangle. \tag{4}$$

We then select

$$(u^*, v^*) = \arg\max_{(u,v)\in E_t} \left|M_{uv}^{(t)}\right|, \tag{5}$$

and impose the constraint

$$Z_{u^*} = \operatorname{sign}\left(M_{u^*v^*}^{(t)}\right) Z_{v^*}, \tag{6}$$

thereby eliminating one variable and producing a reduced Hamiltonian H_{t+1}. This procedure is repeated until the number of remaining variables falls below a predefined threshold n_c, after which the residual problem is solved classically[27–29].

2.3 Related work

Finite-shot estimation is a central practical bottleneck in hybrid quantum algorithms. Since expectation values must be inferred from repeated circuit executions, higher precision generally demands more shots and therefore greater execution time and hardware cost[2, 9, 11]. A substantial literature has accordingly developed on shot reduction, adaptive shot use, operator grouping, and measurement optimization. Within the gradient-based optimizer family, adaptive shot heuristics such as iCANS, gCANS, and Rosalin adjust the per-iteration shot budget to reduce total measurement cost[34–36]; stochastic gradient and simultaneous-perturbation methods further reduce per-step overhead[38–40]; and latency-aware scheduling explicitly accounts for hardware turnaround time[37]. A parallel line of work addresses the measurement problem from the operator side through Pauli grouping[41, 42], commuting-family partitioning[43, 44], and clique-cover or finite-sampling-aware strategies[45, 46]. In the VQE domain, Liang *et al.* use RL to dynamically determine the total shot count across

optimization iterations, demonstrating transferability across molecular systems[47]. These efforts have been concentrated in VQE and gradient-based variational settings, and to our knowledge no prior work has addressed step-wise measurement-budget control within the recursive elimination chain of RQAOA.

Within the recursive quantum optimization literature, Bravyi *et al.* formalize hybrid recursive reduction for graph coloring[27], and subsequent work has extended recursive quantum optimization to complete graphs[28], quantum-informed recursive optimization[30], and maximum independent set[31]. Patel *et al.* learn the elimination policy itself using REINFORCE[29]. None of these works, however, study how the measurement budget should be allocated across elimination steps.

Reinforcement learning has proven effective for sequential resource-allocation problems in quantum computing more broadly, including QAOA parameter optimization[48, 49], ansatz and circuit design[51, 57, 58], and robust control under noise[52, 59–61]; recent surveys can be found in [50, 53]. Building on this line, we apply RL to the complementary problem of recursion-level shot allocation: the elimination rule is kept fixed, and the agent learns only how many shots to spend at each step, so that the effect of adaptive measurement control can be evaluated independently of changes to the solver itself.

3 Shot Control Framework for RQAOA

Each contraction in RQAOA alters the reduced graph and reshuffles the ranking of candidate edges, so the correlation structure that determines step difficulty is different at every elimination step. A fixed allocation rule cannot account for this variation and will inevitably over-allocate shots on easy steps or under-allocate on hard ones. This motivates a state-dependent allocation strategy that reads the local difficulty of each step and sets the measurement budget accordingly.

We formulate step-wise shot allocation in depth-1 RQAOA as a finite-horizon Markov decision process (MDP) whose episode corresponds to a single RQAOA run. At each elimination step the agent observes the current state of the reduced instance, allocates a measurement budget, and receives a reward signal that penalizes shot usage subject to a success constraint. The controller follows a two-level design: a hand-crafted Heuristic baseline encodes simple domain rules about when a step is likely to be easy or hard, and a residual RL policy learns step-by-step corrections to that baseline. The core design choices are therefore a lightweight tabular state, a baseline-guided residual action space, and a Lagrangian-constrained training objective[62–67].

3.1 MDP formulation

Let n denote the initial number of variables and n_c the classical-solver threshold. One RQAOA episode consists of $T = n - n_c$ elimination steps, indexed $t = 1, \ldots, T$. At step t, the agent observes a state s_t summarizing the reduced instance with m_t active variables, selects a shot allocation $k_t \leq C$ (where C is a per-instance shot cap), and observes a contraction outcome. The instance then transitions to $m_{t+1} = m_t - 1$. When $m_T = n_c$ the residual problem is solved classically and the episode terminates

with a binary success indicator:

$$\sigma = \mathbf{1}\left[\frac{E_{\text{out}}}{E_{\text{opt}}} \geq \rho^*\right], \tag{7}$$

where E_{opt} is the brute-force optimum and $\rho^* = 0.99$.

3.2 State representation

A lightweight four-dimensional state captures the difficulty of the current elimination step:

$$s_t = \big(m_t,\ \zeta_t,\ \kappa_t,\ d_t\big), \tag{8}$$

where $m_t \in [n_c{+}1,\ n]$ is the number of remaining variables, encoding position within the recursive chain. The *z-gap*

$$\zeta_t = \frac{|M^{(t)}_{(1)}|}{|M^{(t)}_{(2)}| + \varepsilon}, \qquad \varepsilon = 10^{-12}, \tag{9}$$

is the ratio of the two largest absolute pairwise correlations; $\zeta_t \gg 1$ indicates that the leading edge is easily identifiable even with few shots, while $\zeta_t \approx 1$ signals a near-tie vulnerable to sampling noise. The *conflict ratio* measures endpoint overlap among the top-k candidate edges ($k = 3$). Let $\mathcal{V}^{(t)}_k$ denote the set of unique endpoints of these k edges. Then

$$\kappa_t = 1 - \frac{|\mathcal{V}^{(t)}_k|}{2k}. \tag{10}$$

High κ_t means that eliminating one variable directly affects the next candidates, amplifying the downstream cost of a misidentified leading edge. Finally, the *edge distance*

$$d_t = \min_{u \in e_1,\, v \in e_2} \text{dist}_G(u, v), \tag{11}$$

is the minimum graph-theoretic distance between the endpoints of the two most strongly correlated edges e_1, e_2; small d_t indicates shared neighborhood structure.

State features are estimated from a probe measurement of $k_{\text{probe}} \in [16,\ 32]$ shots performed before the main allocation; all reported shot counts include these probe shots. Each feature is discretized into a small number of bins (m_t: $n{-}n_c$ levels; ζ_t: 7 levels; κ_t: 5 levels; d_t: 5 levels), yielding a compact tabular state space.

3.3 Action space and residual policy

Rather than selecting an absolute shot fraction from scratch, the controller begins from a hand-crafted Heuristic baseline and learns only residual corrections around it. This baseline is not merely an initialization device for RL; it is a standalone, state-aware allocation rule designed from local indicators of step difficulty. Six candidate fractions

of the cap C are available, $\mathcal{F} = \{0.20, 0.35, 0.50, 0.65, 0.80, 1.00\}$, indexed $0, \ldots, 5$. The baseline index $b(s_t)$ assigns an initial allocation level based on the state features:

$$b(s_t) = \begin{cases} 0 & \zeta_t \geq 4.0,\ \kappa_t < 0.10,\ d_t \geq 3 \quad \text{(use 20\% of } C\text{)}, \\ 1 & \zeta_t \geq 2.0,\ \kappa_t < 0.20,\ d_t \geq 2 \quad \text{(use 35\%)}, \\ 4 & \zeta_t < 0.9 \text{ and } (\kappa_t \geq 0.30 \text{ or } d_t \leq 1) \quad \text{(use 80\%)}, \\ 2 & \text{otherwise} \quad \text{(default, 50\%)}. \end{cases} \tag{12}$$

We evaluate this hand-crafted rule throughout as the standalone Heuristic baseline. This makes it possible to separate two questions experimentally: how far an interpretable domain-informed policy can go on its own, and how much additional benefit is obtained when RL learns state-dependent corrections to that policy.

The agent's action $a_t \in \mathcal{A} = \{-3, -2, -1, 0, +1, +2\}$ is a residual offset applied to this baseline:

$$f_t = \mathrm{clip}\big(b(s_t) + a_t,\ 0,\ 5\big), \qquad k_t = \max\big(k_{\text{probe}},\ \lfloor \mathcal{F}[f_t] \cdot C \rceil\big). \tag{13}$$

This residual design concentrates exploration near an interpretable and already competitive prior rather than sweeping over the full action range. It also guarantees that the hand-crafted rule remains exactly available through $a_t = 0$. The role of RL is therefore not to replace a weak comparator, but to learn when the baseline is too conservative or too aggressive, especially in regimes where fixed thresholds do not transfer perfectly across problem sizes.

3.4 Reward and Lagrangian-constrained objective

Per-step rewards penalize shot expenditure relative to the cap:

$$r_t = -\eta \cdot \frac{k_t}{C}, \qquad t = 1, \ldots, T, \tag{14}$$

with $\eta = 1.0$. At the terminal step, an additional failure penalty is incurred if the episode does not meet the success criterion:

$$r_T \ \mathrel{+}= \ -\lambda \cdot (1 - \sigma), \tag{15}$$

where σ is the success indicator defined in Eq. (7) and $\lambda \geq 0$ is a Lagrangian multiplier.

The multiplier enforces the constraint $\mathbb{P}_\pi[\sigma = 1] \geq p^*$ with target $p^* = 0.95$. It is adapted across episodes via

$$\lambda \leftarrow \mathrm{clip}\big(\lambda + \mu_\lambda \cdot (p^* - \hat{p}),\ 0,\ \lambda_{\max}\big), \tag{16}$$

where $\hat{p}$ is an exponential moving average of episode-success indicators with coefficient $\beta = 0.10$, and $\lambda_{\max} = 80$. During a warm-up phase of 100 episodes, λ is held at its initial value $\lambda_0 = 2.0$ to allow the policy to discover successful trajectories before the constraint tightens.

Algorithm 1 Lagrangian-constrained residual Double Q-learning for RQAOA shot allocation

Require: Instance $(\mathbf{J}, E_{\text{opt}}, C, n, n_c)$; hyperparameters $\alpha, \gamma, \lambda_0, \lambda_{\max}, p^*$

Initialize $Q_1, Q_2 \leftarrow \mathbf{0}$, $\lambda \leftarrow \lambda_0$, $\hat{p} \leftarrow p^*$, $\epsilon \leftarrow 1.0$
for episode $e = 1, \ldots, 1200$ **do**
  $m \leftarrow n$; $\mathbf{J}' \leftarrow \mathbf{J}$
  while $m > n_c$ **do**
    Probe with k_{probe} shots and extract $(\zeta,\ \kappa,\ d)$
    $s \leftarrow \text{encode}(m,\ \zeta,\ \kappa,\ d)$; $b \leftarrow \text{baseline}(\zeta,\ \kappa,\ d)$
    $a \leftarrow \epsilon\text{-greedy}(Q_1{+}Q_2,\ s)$; $f \leftarrow \text{clip}(b{+}a,\ 0,\ 5)$
    $k \leftarrow \max\big(k_{\text{probe}},\ \lfloor \mathcal{F}[f] \cdot C \rceil\big)$
    Execute RQAOA contraction with k shots; $r \leftarrow -k/C$; $m \leftarrow m{-}1$
    Update Q_1 or Q_2 via Eq. (18)
  end while
  Solve the residual problem classically and compute σ via Eq. (7)
  $r_T \mathrel{+}= -\lambda\,(1{-}\sigma)$; update terminal Q-value
  if $e > 100$ **then**
    $\hat{p} \leftarrow (1{-}\beta)\hat{p} + \beta\,\sigma$; update λ via Eq. (16)
  end if
  $\epsilon \leftarrow \max(0.02,\ \epsilon \times 0.995)$
end for
return best-checkpoint policy

The overall training objective is

$$\min_{\pi}\ \mathbb{E}_{\pi}\left[\sum_{t=1}^{T} k_t\right] \quad \text{subject to} \quad \mathbb{P}_{\pi}[\sigma = 1] \geq p^*. \tag{17}$$

3.5 Double Q-learning algorithm

We maintain two independent tabular Q-function estimates $Q_1, Q_2 : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, trained via Double Q-learning[64] to mitigate maximization bias. At each update, one table is selected uniformly at random; Q_1 is updated using the greedy action from Q_2 as the bootstrap target and vice versa:

$$Q_i(s,a) \leftarrow (1{-}\alpha)\,Q_i(s,a) + \alpha\big[r + \gamma\,Q_j\big(s',\ \arg\max_{a'} Q_i(s',a')\big)\big], \tag{18}$$

where (i, j) is a random permutation of $(1, 2)$.

An episode proceeds as follows. At each step t: (i) a probe measurement of k_{probe} shots is performed to compute ζ_t, κ_t, and d_t; (ii) the state s_t is encoded and discretized; (iii) the ϵ-greedy policy selects a_t and yields k_t total shots for the step; (iv) the contraction is executed; and (v) the transition (s_t, a_t, r_t, s_{t+1}) is used immediately to update Q_1 or Q_2.

Hyperparameters are fixed across all training runs: learning rate $\alpha = 0.15$, discount $\gamma = 0.97$, ϵ-decay from 1.0 to 0.02 at rate 0.995 per episode, and 1,200 training episodes

for each base-policy run. Each base policy is trained only on its designated training instance, and the learned policy is then evaluated, without further updating, on held-out and cross-size instances. Checkpoint selection retains the policy with the highest validation success rate, breaking ties by lower median total shots and then lower mean total shots.

4 Experimental Setup

All experiments are conducted on a noiseless simulator using Qiskit Aer (`AerSimulator` backend)[68]. Full implementation details, including angle optimization, probe-shot settings, and RL hyperparameters, are provided in Appendix A.

4.1 Training bases and evaluation pools

We evaluate two base policies, each trained on one designated weighted Max-Cut instance: **Base 0** (b_0), an 8-regular graph on $n = 14$ nodes, and **Base 1** (b_1), a 17-regular graph on $n = 20$ nodes. AAll graphs in the benchmark are d-regular with Gaussian edge weights. The benchmark combines designated hard training bases, reweighted variants of the base topologies, additional same-size hard instances from the pool, same-size random regular graphs without the hard-instance filter, and cross-size instances drawn from the hard-instance pool. Here, "hard" is used operationally only for instances whose Uniform-allocation approximation ratio is at most 0.95 relative to the brute-force optimum. All experiments use the same classical threshold $n_c = 8$ and circuit depth $p = 1$.

Each base policy is evaluated on a dedicated set of same-size test instances (reweighted variants of the training graph, different hard instances from the pool, and randomly generated regular graphs) as well as a shared **Cross-size** set spanning unseen node counts $n \in \{15, \ldots, 25\} \setminus \{14, 20\}$. The complete benchmark contains **104 unique graph instances** and **164 policy–instance evaluation pairs**.

4.2 Cap calibration and evaluation protocol

For each instance, the per-step shot cap C_{uniform} is determined using the Uniform baseline only. A two-stage search identifies the smallest cap such that Uniform achieves success rate ≥ 0.95 over $N_{\text{cal}} = 60$ calibration trials. The search first scans a coarse power-of-two grid $\{64, 128, \ldots, 4096\}$ and then refines the result within the identified bracket. All three methods are subsequently evaluated under this same cap, so that any observed difference in shot usage or success rate reflects the allocation strategy rather than unequal per-step budgets.

Some evaluation pairs remain strongly budget-limited even at the maximum cap. We retain only pairs for which $\text{SR}_{\text{Uniform}} \geq 0.90$, yielding an **Operational subset** of 135 pairs. Excluding the Training and Validation categories used for model selection leaves the **held-out operational subset** of 130 pairs over 81 unique instances, which is the basis for all main quantitative claims. The remaining 29 pairs are not discarded but reported separately as a robustness view (Complete benchmark, 164 pairs).

4.3 Baselines and metrics

We compare three allocation strategies, each evaluated over $N = 60$ independent trials per policy–instance pair. **Uniform** allocates exactly C_{uniform} shots at every elimination step. **Heuristic** applies the hand-crafted baseline $b(s_t)$ from Eq. (12) without RL adjustment ($a_t = 0$ throughout), serving both as a standalone policy and as the anchor for the residual RL controller. **RL** (b_0 or b_1) applies the learned Double Q-policy trained on the designated base instance.

For each pair, *shot reduction* is defined as

$$\text{Reduction} = 1 - \frac{\widetilde{S}_{\text{method}}}{\widetilde{S}_{\text{Uniform}}}, \tag{19}$$

where $\widetilde{S}$ denotes the median total shots over all trials. *Success rate* (SR) is the fraction of trials satisfying $E_{\text{out}}/E_{\text{opt}} \geq 0.99$. To combine resource usage and reliability into a single efficiency proxy, we define the *effective shots per success*:

$$\text{ESP} = \frac{\widetilde{S}_{\text{succ}}}{\text{SR}}, \tag{20}$$

where $\widetilde{S}_{\text{succ}}$ is the median total shots among successful trials. Lower ESP indicates better efficiency. For cross-method comparison we report the normalized ratio $\text{ESP}_{\text{method}}/\text{ESP}_{\text{Uniform}}$, for which values below 1.0 indicate improvement over Uniform. During training, the policy snapshot with the highest validation success rate is retained, breaking ties by lower median total shots.

5 Results

All three methods are evaluated under the fixed-cap fairness protocol of Sec. 4.2. The main quantitative claims below are based on the **held-out operational subset** (130 pairs, 81 unique instances) that excludes Training and Validation categories.

5.1 Overall and regime-level performance

Table 1 summarizes aggregate performance across the main held-out test set and robustness subsets.

On the held-out operational subset, the RL policy achieves a mean pairwise shot reduction of **36.1%** relative to Uniform, compared with **23.0%** for the Heuristic. The corresponding ESP ratios are **0.676** (RL) and **0.790** (Heuristic), indicating that RL attains a substantially lower effective cost per success. Including Training/Validation instances or the full Complete benchmark changes these figures only negligibly, confirming that the qualitative ranking is not driven by the filtering rule.

Breaking the held-out subset into evaluation regimes, RL remains effective in both: same-size held-out (34.8% reduction, 0.684 ESP ratio) and Cross-size (36.6%, 0.673). The gain is at least as strong on unseen sizes. The Heuristic's strong same-size performance (25.3% reduction) is itself informative: the hand-crafted rule captures

Table 1 Aggregate performance on the main held-out test set and robustness subsets. Reduction and ESP ratio are measured relative to Uniform.

Subset	N	RL Red.	RL ESP	Heu. Red.	Heu. ESP
Held-out operational pairs	130	36.1%	0.676	23.0%	0.790
Same-size held-out	32	34.8%	0.684	25.3%	0.766
Cross-size pairs	98	36.6%	0.673	22.2%	0.798
Operational pairs (incl. Train/Val)	135	36.1%	0.677	23.0%	0.790
Complete benchmark	164	36.5%	0.674	22.3%	0.798

Table 2 Comparison of the two RL base policies on the 51 matched cross-size instances evaluated under both policies.

Policy	N	Reduction	ESP ratio	Success rate
b_0 (trained at $n = 14$)	51	34.7%	0.691	0.915
b_1 (trained at $n = 20$)	51	38.4%	0.655	0.911

substantial allocation structure without any learning. The RL–Heuristic gap varies with instance complexity, being modest at n=14 (only six elimination steps) and consistently larger at n=20 and beyond.

5.2 Base policy comparison

The benchmark contains 51 *common* instances from the Operational subset that are evaluated under both b_0 and b_1. These all belong to the **Cross-size** regime, so the comparison isolates transfer quality on unseen sizes.

On this matched subset, b_1 achieves both higher reduction and lower ESP ratio than b_0, while the two policies have very similar success rates. One plausible explanation is that training on the larger $n = 20$ base exposes the agent to a broader range of elimination-step difficulties.

5.3 Generalization to unseen problem sizes

The Cross-size regime provides the most demanding transfer test, spanning $n \in \{15, \ldots, 25\} \setminus \{14, 20\}$ with 98 policy–instance pairs. The mean RL shot reduction on these unseen-size instances is **36.6%**, essentially matching the overall held-out average. Within the d-regular Gaussian-weighted benchmark family used in this study, this persistence suggests that the state representation captures local step difficulty in a transferable form rather than memorizing size-specific thresholds.

Figure 2 shows the shot reduction as a function of problem size, confirming that the RL advantage is stable across the full n range. The matched cross-size comparison in Table 2 further shows that both base policies transfer meaningfully, even though b_1 performs somewhat better than b_0.

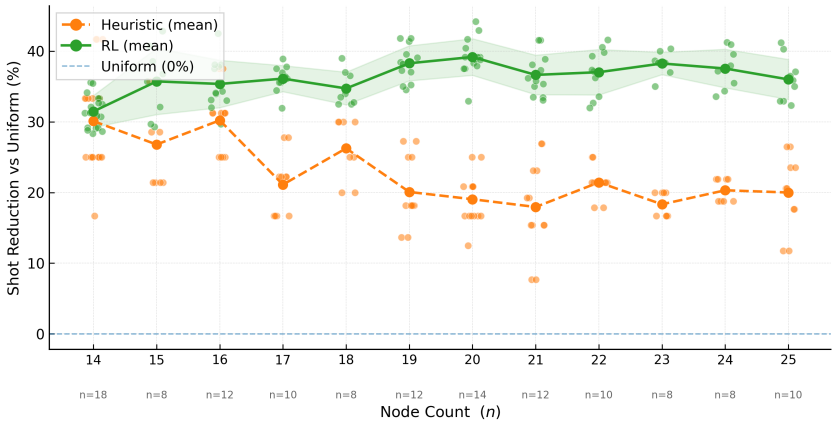


Fig. 2 Shot reduction (%) vs. problem size n for the held-out operational subset. Each point represents one policy–instance pair. The per-size mean (connected line) shows that RL achieves stable $\sim$36% reduction across the full range, while the Heuristic settles around $\sim$23%.

5.4 Reliability–efficiency trade-off

Figure 3 plots normalized shot usage against success rate for all pairs in the held-out operational subset. Relative to Uniform, the RL points are shifted toward lower shot usage while remaining in a high-success regime in the large majority of cases. The Heuristic occupies an intermediate position, with smaller shot savings and slightly higher success rates on average.

The RL policy is trained with a constraint target $p^* = 0.95$, yet the observed success rate on the held-out operational subset is only about 0.91. This gap is not contradictory: the constraint is enforced during training on a designated base instance through the adaptive multiplier λ, whereas test-time evaluation aggregates many held-out instances of varying difficulty. The held-out results suggest a favorable but nontrivial trade-off: RL gives up a modest amount of reliability relative to Uniform while achieving much larger shot savings, and the resulting ESP ratio remains the best among the tested methods.

To quantify the role of the adaptive constraint in maintaining reliability, we compare the default controller (residual action with adaptive λ) with an unconstrained variant ($\lambda = 0$, all other settings identical) on 22 matched cross-size instances. Table 3 reports coverage, defined as the number of matched instance pairs achieving SR $\geq \tau$, at several reliability floors.

At $\tau = 0.95$, the default controller remains feasible on 13 of 22 instances, compared with 5 for the unconstrained variant, indicating that the adaptive Lagrangian contributes meaningfully to high-reliability coverage. The coverage gap already appears at moderately strict floors and persists through tighter thresholds. This pattern suggests that the adaptive constraint primarily widens the set of instances on which the controller can satisfy strict reliability targets.

Table 3 SR-floor coverage comparison on 22 matched cross-size instances. Default denotes the residual controller with adaptive λ; No Constraint sets $\lambda = 0$ throughout training. Coverage is the number of pairs (out of 22) achieving SR $\geq \tau$.

τ (SR floor)	Default	No Constraint	Δ Coverage
0.90	18/22	16/22	+2
0.92	14/22	8/22	+6
0.94	13/22	5/22	+8
0.95	13/22	5/22	+8

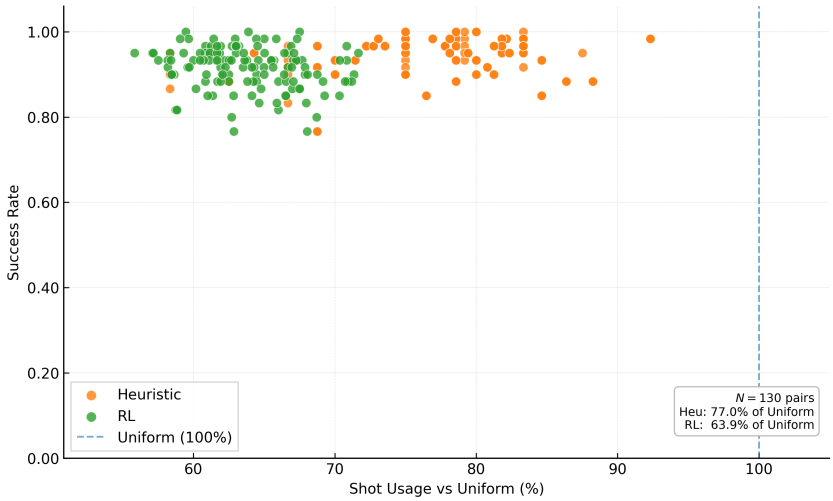


Fig. 3 Normalized shot usage ($\widetilde{S}_{\text{method}}/\widetilde{S}_{\text{Uniform}}$) vs. success rate for all pairs in the held-out operational subset. Uniform (blue), Heuristic (orange), and RL (green). RL consistently shifts toward lower shot usage while maintaining competitive success rates; the Heuristic occupies an intermediate position.

5.5 Sensitivity to filtering and metric choice

The main quantitative claims in this paper are based on the held-out operational subset filtered by $\text{SR}_{\text{Uniform}} \geq 0.90$ after cap calibration. A natural concern is whether this filtering inflates the reported gains. Table 4 shows that the qualitative ranking is stable as the inclusion criterion is relaxed: on the held-out operational subset (130 pairs), RL achieves 36.1% reduction with an ESP ratio of 0.676, compared with 23.0% and 0.790 for the Heuristic. Including all held-out pairs (158 pairs) and then the complete benchmark (164 pairs) changes these numbers only negligibly. The main conclusions are therefore not an artifact of the filtering boundary.

A separate concern is the choice of summary metric. Table 5 reports four shot-reduction metrics on the held-out operational subset ($N = 130$). The median-based reduction (36.1%) and mean-based reduction (36.0%) are nearly identical for RL,

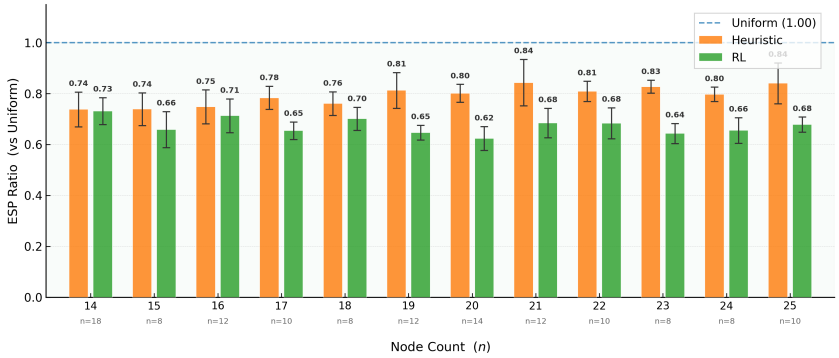


Fig. 4 Mean ESP ratio by problem size n for the held-out operational subset. Values below 1.0 indicate lower effective cost per success than Uniform. Error bars show the range across instances at each n. Both Heuristic and RL improve over Uniform, with RL achieving a consistently lower ESP ratio.

Table 4 Filtering sensitivity. The first row applies the primary evaluation protocol ($\mathrm{SR}_{\mathrm{Uniform}} \geq 0.90$, held-out); subsequent rows progressively relax the inclusion criterion. RL reduction and ESP ratio are stable across all subsets.

Subset	N	RL Red.	RL ESP	Heu. Red.	Heu. ESP
SR ≥ 0.90, held-out	130	36.1%	0.676	23.0%	0.790
All held-out pairs	158	36.5%	0.674	22.2%	0.800
Complete benchmark	164	36.5%	0.674	22.3%	0.798

Table 5 Metric robustness on the held-out operational subset ($N = 130$). All four metrics confirm RL's advantage. Success rates are 0.915 (RL) and 0.943 (Heuristic) throughout.

Metric	RL Reduction	Heu. Reduction
Median shots (primary)	36.1%	23.0%
Mean shots	36.0%	22.9%
P90 (worst-case tail)	27.2%	14.2%
Restart cost ($\widetilde{S}_{\mathrm{mean}}$/SR)	32.3%	20.9%

confirming that the savings are not driven by outlier trials. The tail-sensitive P90 metric still shows a 27.2% reduction for RL and 14.2% for the Heuristic, while the restart-cost metric S_{mean}/SR yields 32.3% and 20.9%, respectively. Success rates on this subset are 0.915 for RL and 0.943 for the Heuristic. Across all four metrics, RL retains its advantage over the Heuristic.

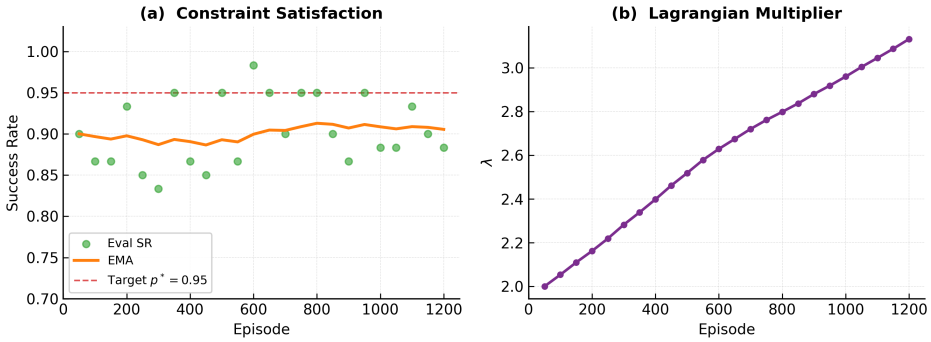


Fig. 5 Training diagnostics for the b_0 policy (n=14). (a) Evaluation success rate and its exponential moving average (EMA) remain near the target p^*=0.95 (dashed line). (b) The Lagrangian multiplier λ rises monotonically from 2.0 to ∼3.1, confirming that the policy sustains feasibility under a progressively tighter constraint.

5.6 Training dynamics and robustness checks

This subsection provides empirical evidence that the agent learns effectively under the Lagrangian-constrained objective and that the learned policy is robust to the penalty regime.

Training dynamics.

Figure 5 shows two training diagnostics for the b_0 policy (n=14). The evaluation success rate remains near the target p^*=0.95 throughout training (panel a), while the Lagrangian multiplier λ rises steadily from 2.0 to approximately 3.1 (panel b). Because the reward includes a term $-\lambda(1-\sigma)$, the rising λ continuously tightens the feasibility constraint; the fact that the success rate is maintained despite this increasingly stringent penalty indicates that the policy learns to reduce shot usage while keeping reliability near the training target. The resulting shot savings are quantified in the held-out evaluation (Table 1), where the best-checkpoint policy achieves 36.1% reduction across 130 unseen evaluation pairs.

Penalty regime sensitivity.

To assess whether the learned policy is sensitive to the Lagrangian hyperparameters, we trained an aggressive variant with λ_0 raised fourfold (from 2.0 to 8.0), $\lambda_{\max}$ nearly doubled (from 80 to 150), and the episode budget doubled (from 1,200 to 2,400), among other changes (full details in Appendix B). On a matched set of 20 cross-size instances, the aggressive variant achieved a mean shot reduction of 36.9% with a mean success rate of 0.911, compared with 37.0% and 0.926 for the standard setting. The near-identical shot reduction despite substantially different penalty regimes indicates that the learned allocation strategy is robust to the constraint-enforcement hyperparameters.

Role of residual structure and adaptive constraint.

The current controller combines a residual action parameterization with an adaptive Lagrangian constraint. During development, both choices were observed to improve training stability relative to alternatives explored at that stage. A systematic controlled comparison among controller architectures under the final codebase is deferred to future work; the penalty-regime robustness analysis above and the per-step allocation analysis in Section 6 provide partial support for the selected design.

6 Discussion

Effectiveness of the residual design.

The residual parameterization provides interpretability and baseline-guided exploration. The Heuristic baseline is not a strawman or a mere initialization device; it is a novel, state-aware allocation rule derived from local indicators of step difficulty. Its own strong performance—achieving approximately 23% reduction across the benchmark—shows that a compact domain-informed prior already captures a meaningful part of the budget-allocation structure. RL is useful because it learns when this prior should be tightened or relaxed on a step-by-step basis, rather than having to discover the entire allocation policy from scratch.

To characterize what the residual controller learns, we analyze the allocation pattern from the converged Q-tables. Figure 6 shows the mean shot fraction as a function of z-gap confidence and aggregated step difficulty. The residual controller allocates shots selectively: on easy steps it uses the minimum fraction (0.20), matching the direct-action baseline, but on hard steps it invests approximately twice the budget (0.41 vs. 0.20, selectivity +0.21). By contrast, a direct-action controller trained under identical conditions converges to near-uniform minimum allocation regardless of difficulty (91% of states at frac = 0.20, selectivity +0.10). This indiscriminate reduction is consistent with the lower success rates observed for direct-action policies and helps explain why the residual parameterization enables more effective difficulty-aware allocation. Its relative merit compared with alternative action-space designs warrants further investigation under the final codebase.

Training on hard instances.

Both base policies are trained on instances that are hard for depth-1 RQAOA (approximation ratio ≤ 0.95 under Uniform allocation). Hard instances expose a wider range of step difficulties during training—including near-ties in correlations, high conflict ratios, and close candidate edges—providing richer learning signal than easy instances where most steps are unambiguous. This choice is reflected in the base-policy comparison (Table 2): b_1, trained on the larger and structurally more complex n=20 instance, achieves higher shot reduction and lower ESP ratio than b_0 on the same matched cross-size set, suggesting that a more diverse training environment leads to a more effective allocation policy.

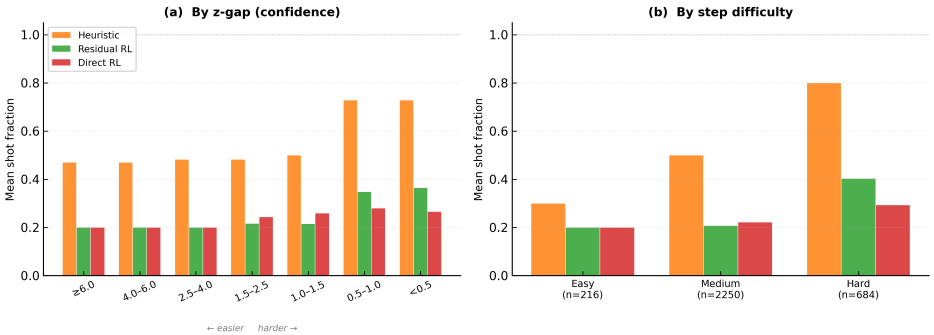


Fig. 6 Per-step allocation pattern by step difficulty. Left: mean shot fraction allocated by each controller (Heuristic, Residual RL, Direct RL) as a function of aggregated step difficulty. Right: action distribution across difficulty levels. The residual controller exhibits higher selectivity (hard − easy = +0.21) than the direct-action controller (+0.10), concentrating additional shots on hard steps while matching the minimum allocation on easy steps.

Transfer across problem sizes.

The cross-size results refer specifically to the held-out Cross-size set of 98 pairs, not to a benchmark that includes Training or Validation categories. The four-dimensional state $(m_t, \zeta_t, \kappa_t, d_t)$ appears to support this transfer because it uses mostly ratio-based or local structural features rather than absolute graph statistics. It is worth noting that the Heuristic itself also transfers to unseen sizes (22.2% mean reduction on the Cross-size subset), because its rules are defined in terms of the same state features. The contribution of RL is not to enable transfer *per se* but to learn corrections that consistently improve upon the Heuristic's fixed thresholds across the full tested size range. We note, however, that all evaluation instances belong to the same d-regular Gaussian-weighted benchmark family, even though not every category is hard-screened; whether these corrections remain effective on structurally different graph classes is an open empirical question.

When does RL help the least?

The regime in which RL offers the smallest incremental improvement over the Heuristic is same-size evaluation under b_0 (n=14). With only six elimination steps, these instances provide limited room for state-dependent optimization, and the Heuristic already captures most of the available allocation structure (ESP ratio 0.738 vs. 0.731 for RL). By contrast, under b_1 (n=20, twelve elimination steps) RL outperforms the Heuristic on all 14 same-size held-out instances (ESP ratio 0.624 vs. 0.802). We attribute this pattern to instance complexity rather than to a regime-specific advantage: longer recursion chains expose more steps at which the learned controller can identify and exploit state-dependent opportunities that fixed thresholds miss. From a practical standpoint, this scaling behavior is encouraging because the problems of greatest interest on near-term quantum devices are precisely those at larger sizes, where the RL advantage is most pronounced.

Separation from learned elimination policies.

Unlike Patel *et al.*[29], who learn the elimination policy itself, we keep the elimination rule fixed and optimize only the measurement budget across recursive steps. This isolates the contribution of adaptive shot allocation, preserves the interpretability of the recursive solver, and makes the learned controller easier to deploy as a plug-in replacement for uniform shot allocation. In this sense, our work is complementary to learned elimination-policy methods rather than a competing redesign of the recursive solver. This also distinguishes our setting from optimizer-side shot adaptation in VQE/VQA[34–38], where the controller typically allocates shots during parameter updates rather than along a recursive elimination chain whose measurements change the next reduced instance.

Limitations.

Several limitations constrain the scope of the present results. First, all experiments are conducted on a noiseless simulator. On real hardware, shot noise interacts with gate errors, readout noise, and decoherence, and the best allocation strategy may shift accordingly[32, 33, 69]. Second, the study is restricted to depth-1 QAOA. Higher depths may require a richer state representation and a more expressive controller. Third, each base policy is trained on a single designated instance. Although the transfer results are encouraging, multi-instance training may improve robustness. Fourth, the current controller is tabular; substantially larger problems may require function approximation. Fifth, controlled single-variable ablations comparing the residual structure and adaptive constraint against alternatives under the final codebase would strengthen the evidence for these design choices; such ablations are left to future work.

7 Conclusion

We presented a two-level framework for dynamic shot allocation in depth-1 RQAOA. A state-aware Heuristic baseline provides an interpretable allocation rule that already achieves approximately 23% shot reduction, and a residual Double Q-learning controller learns step-by-step corrections that bring the total reduction to 36% on a held-out benchmark of 130 evaluation pairs over 81 unique instances. The improvement scales with instance complexity and persists on unseen problem sizes, with the Cross-size regime retaining a 36.6% reduction. Training diagnostics, penalty-regime robustness checks, and per-step allocation analysis support the stability and interpretability of the selected controller configuration. From a practical standpoint, the learned controller acts as a plug-in replacement for uniform shot allocation in any existing RQAOA pipeline, without modifying the elimination rule or circuit construction. Because the RL advantage grows with problem size, the approach is well positioned for the larger instances that are of primary interest on near-term quantum hardware. Future directions include extending the framework to higher QAOA depths, where the richer correlation structure may offer additional room for adaptive allocation; validating on noisy hardware, where shot noise interacts with gate and readout errors; multi-instance training, which may improve robustness beyond single-instance policies; and evaluation on diverse graph families beyond the hard d-regular

weighted instances studied here, including non-regular topologies and broader weight distributions. More broadly, the present results suggest that recursion-level measurement control is a promising and largely unexplored complement to circuit-level and parameter-level optimization in shot-limited quantum algorithms.

A Implementation Details

All quantum circuits are executed with Qiskit Aer using the `AerSimulator` backend[68]. At each elimination step, the QAOA variational angles (γ, β) are optimized by a grid search over 48 candidate γ values in $[0, 2\pi]$, followed by a SciPy BFGS local refinement on both parameters. Probe shots used to estimate the state features are set to $k_{\text{probe}} = 16$ for $n \leq 16$ and $k_{\text{probe}} = 32$ for $n > 16$. Training hyperparameters are fixed across both base policies: learning rate $\alpha = 0.15$, discount factor $\gamma = 0.97$, ϵ-greedy exploration decaying from 1.0 to 0.02 at rate 0.995 per episode, and 1,200 training episodes. The Lagrangian multiplier is initialized at $\lambda_0 = 2.0$, held fixed for a 100-episode warm-up, and subsequently adapted with step size $\mu_\lambda = 1.0$ and ceiling $\lambda_{\max} = 80$. Checkpoint selection retains the policy with the highest validation success rate, breaking ties by lower median total shots and then lower mean total shots.

B Penalty Regime Sensitivity

To assess whether the learned allocation strategy is sensitive to the Lagrangian constraint hyperparameters, we compared the standard training configuration (v3) with an aggressive variant (v4) that simultaneously increases the penalty strength and the training budget. Table 6 summarizes the key differences; all other settings (residual action space, state features, action fractions) remain identical. Because multiple

Table 6 Hyperparameter comparison for the penalty regime sensitivity experiment.

Parameter	Standard (v3)	Aggressive (v4)
λ_0	2.0	8.0
$\lambda_{\max}$	80	150
μ_λ (step size)	1.0	2.0
Warm-up episodes	100	50
Additional fail penalty	0.0	5.0
Training episodes	1,200	2,400
Evaluation trials	60	100

hyperparameters were changed simultaneously, this comparison is best interpreted as a stress test of the penalty regime rather than a single-variable ablation. Table 7 reports test performance on a matched set of 20 cross-size instances evaluated under both configurations. The near-identical shot reduction (37.0% vs. 36.9%) and comparable success rates despite a fourfold increase in λ_0, doubled episode budget, and additional

Table 7 Performance comparison on 20 matched cross-size instances under the standard and aggressive penalty regimes.

Metric	Standard (v3)	Aggressive (v4)
Mean SR	0.926	0.911
Mean shot reduction (%)	37.0	36.9

failure penalty indicate that the learned policy is robust to substantial changes in the constraint-enforcement regime. During training, λ rose to approximately 3.1 under the standard setting and to approximately 12.0 under the aggressive setting, yet the two resulting policies selected similar allocation strategies. This suggests that the final policy is driven primarily by the state features and residual structure rather than by the specific penalty magnitude.

Acknowledgements. This research was supported by the Quantum Computing based on Quantum Advantage challenge research (RS-2025-08182968) through the National Research Foundation of Korea (NRF) funded by the Korean government (Ministry of Science and ICT (MSIT)).

Declarations

Ethics Approval

Not applicable.

Funding

This research was supported by the National Research Foundation of Korea (NRF), grant number RS-2025-08182968.

References

[1] Moll, N., Barkoutsos, P., Bishop, L.S., Chow, J.M., Cross, A., Egger, D.J., Filipp, S., Fuhrer, A., Gambetta, J.M., Ganzhorn, M., Mezzacapo, A., Salis, G., Smolin, J., Tavernelli, I., Temme, K.: Quantum optimization using variational algorithms on near-term quantum devices. Quantum Sci. Technol. **3**, 030503 (2018)

[2] Cerezo, M., Arrasmith, A., Babbush, R., Benjamin, S.C., Endo, S., Fujii, K., McClean, J.R., Mitarai, K., Yuan, X., Cincio, L., Coles, P.J.: Variational quantum algorithms. Nat. Rev. Phys. **3**, 625–644 (2021)

[3] Lee, E., Lee, S., Kim, S.: Quantum walk based Monte Carlo simulation for photon interaction cross sections. Phys. Rev. D **111**, 116001 (2025) https://doi.org/10.1103/PhysRevD.111.116001

[4] Bauer, B., Bravyi, S., Motta, M., Chan, G.K.-L.: Quantum algorithms for quantum chemistry and quantum materials science. Chem. Rev. **120**, 12685–12717 (2020)

[5] Biamonte, J., Wittek, P., Pancotti, N., Rebentrost, P., Wiebe, N., Lloyd, S.: Quantum machine learning. Nature **549**, 195–202 (2017)

[6] Nielsen, M.A., Chuang, I.L.: Quantum Computation and Quantum Information. Cambridge University Press, Cambridge (2010)

[7] Preskill, J.: Quantum computing in the NISQ era and beyond. Quantum **2**, 79 (2018)

[8] Peruzzo, A., McClean, J., Shadbolt, P., Yung, M.-H., Zhou, X.-Q., Love, P.J., Aspuru-Guzik, A., O'Brien, J.L.: A variational eigenvalue solver on a photonic quantum processor. Nat. Commun. **5**, 4213 (2014)

[9] McClean, J.R., Romero, J., Babbush, R., Aspuru-Guzik, A.: The theory of variational hybrid quantum-classical algorithms. New J. Phys. **18**, 023023 (2016)

[10] Kandala, A., Mezzacapo, A., Temme, K., Takita, M., Brink, M., Chow, J.M., Gambetta, J.M.: Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets. Nature **549**, 242–246 (2017)

[11] Bharti, K., Cervera-Lierta, A., Kyaw, T.H., Haug, T., Alperin-Lea, S., Anand, A., Degroote, M., Heimonen, H., Kottmann, J.S., Menke, T., Mok, W.-K., Sim, S., Kwek, L.-C., Aspuru-Guzik, A.: Noisy intermediate-scale quantum algorithms. Rev. Mod. Phys. **94**, 015004 (2022)

[12] Tilly, J., Chen, H., Cao, S., Picozzi, D., Setia, K., Li, Y., Grant, E., Wossnig, L., Rungger, I., Booth, G.H., Tennyson, J.: The variational quantum eigensolver: a review of methods and best practices. Phys. Rep. **986**, 1–128 (2022)

[13] Farhi, E., Goldstone, J., Gutmann, S.: A quantum approximate optimization algorithm (2014)

[14] Hadfield, S., Wang, Z., O'Gorman, B., Rieffel, E.G., Venturelli, D., Biswas, R.: From the quantum approximate optimization algorithm to a quantum alternating operator ansatz. Algorithms **12**, 34 (2019)

[15] Wang, Z., Hadfield, S., Jiang, Z., Rieffel, E.G.: Quantum approximate optimization algorithm for MaxCut: a fermionic view. Phys. Rev. A **97**, 022304 (2018)

[16] Zhou, L., Wang, S.-T., Choi, S., Pichler, H., Lukin, M.D.: Quantum approximate optimization algorithm: performance, mechanism, and implementation on near-term devices. Phys. Rev. X **10**, 021067 (2020)

[17] Crooks, G.E.: Performance of the quantum approximate optimization algorithm on the maximum cut problem (2018)

[18] Willsch, D., Willsch, M., Jin, F., De Raedt, H., Michielsen, K.: Benchmarking the quantum approximate optimization algorithm. Quantum Inf. Process. **19**, 197 (2020)

[19] Farhi, E., Gamarnik, D., Gutmann, S.: The quantum approximate optimization algorithm needs to see the whole graph: a typical case (2020)

[20] Farhi, E., Gamarnik, D., Gutmann, S.: The quantum approximate optimization algorithm needs to see the whole graph: worst case examples (2020)

[21] Marwaha, K.: Local classical MAX-CUT algorithm outperforms $p = 2$ QAOA on high-girth regular graphs. Quantum **5**, 437 (2021)

[22] Hastings, M.B.: Classical and quantum bounded depth approximation algorithms (2019)

[23] Bravyi, S., Kliesch, A., Koenig, R., Tang, E.: Obstacles to state preparation and variational optimization from symmetry protection. Phys. Rev. Lett. **125**, 260505 (2020)

[24] McClean, J.R., Boixo, S., Smelyanskiy, V.N., Babbush, R., Neven, H.: Barren plateaus in quantum neural network training landscapes. Nat. Commun. **9**, 4812 (2018)

[25] Cerezo, M., Sone, A., Volkoff, T., Cincio, L., Coles, P.J.: Cost function dependent barren plateaus in shallow parametrized quantum circuits. Nat. Commun. **12**, 1791 (2021)

[26] Bittel, L., Kliesch, M.: Training variational quantum algorithms is NP-hard. Phys. Rev. Lett. **127**, 120502 (2021)

[27] Bravyi, S., Kliesch, A., Koenig, R., Tang, E.: Hybrid quantum-classical algorithms for approximate graph coloring. Quantum **6**, 678 (2022)

[28] Bae, E., Lee, S.: Recursive QAOA outperforms the original QAOA for the MAX-CUT problem on complete graphs (2023)

[29] Patel, Y.J., Jerbi, S., Bäck, T., Dunjko, V.: Reinforcement learning assisted recursive QAOA. EPJ Quantum Technol. **11**, 1 (2024)

[30] Finžgar, J.R., Kerschbaumer, A., Schuetz, M.J.A., Mendl, C.B., Katzgraber, H.G.: Quantum-informed recursive optimization algorithms. PRX Quantum **5**, 020327 (2024)

[31] Brady, L.T., Hadfield, S.: Iterative quantum algorithms for maximum independent set. Phys. Rev. A **110**, 052435 (2024)

[32] Temme, K., Bravyi, S., Gambetta, J.M.: Error mitigation for short-depth quantum circuits. Phys. Rev. Lett. **119**, 180509 (2017)

[33] Endo, S., Cai, Z., Benjamin, S.C., Yuan, X.: Hybrid quantum-classical algorithms and quantum error mitigation. J. Phys. Soc. Jpn. **90**, 032001 (2021)

[34] Kübler, J.M., Arrasmith, A., Cincio, L., Coles, P.J.: An adaptive optimizer for measurement-frugal variational algorithms. Quantum **4**, 263 (2020)

[35] Arrasmith, A., Cincio, L., Somma, R.D., Coles, P.J.: Operator sampling for shot-frugal optimization in variational algorithms (2020)

[36] Gu, A., Lowe, A., Dub, P.A., Coles, P.J., Arrasmith, A.: Adaptive shot allocation for fast convergence in variational quantum algorithms (2021)

[37] Ito, K., Mizukami, W., Fujii, K.: Latency-aware adaptive shot allocation for variational quantum algorithms (2023)

[38] Sweke, R., Wilde, F., Meyer, J., Schuld, M., Faehrmann, P.K., Meynard-Piganeau, B., Eisert, J.: Stochastic gradient descent for hybrid quantum-classical optimization. Quantum **4**, 314 (2020)

[39] Stokes, J., Izaac, J., Killoran, N., Carleo, G.: Quantum natural gradient. Quantum **4**, 269 (2020)

[40] Spall, J.C.: Implementation of the simultaneous perturbation algorithm for stochastic optimization. IEEE Trans. Aerosp. Electron. Syst. **34**, 817–823 (1998)

[41] Zhao, A., Rubin, N.C., Miyake, A.: Measurement reduction in variational quantum algorithms. Phys. Rev. Lett. **127**, 110504 (2021)

[42] Huggins, W.J., McClean, J.R., Rubin, N., Jiang, Z., Wiebe, N., Whaley, K.B., Babbush, R.: Efficient and noise resilient measurements for quantum chemistry on near-term quantum computers. npj Quantum Inf. **7**, 23 (2021)

[43] Gokhale, P., Angiuli, O., Ding, Y., Gui, K., Tomesh, T., Suchara, M., Martonosi, M., Chong, F.T.: Minimizing state preparations in variational quantum eigensolver by partitioning into commuting families. npj Quantum Inf. **6**, 34 (2020)

[44] Izmaylov, A.F., Yen, T.-C., Ryabinkin, I.G.: Revising the measurement process in the variational quantum eigensolver: is it possible to reduce the number of separately measured operators? Chem. Sci. **10**, 3746–3755 (2019)

[45] Verteletskyi, V., Yen, T.-C., Izmaylov, A.F.: Measurement optimization in the variational quantum eigensolver using a minimum clique cover. J. Chem. Phys.

152, 124114 (2020)

[46] Crawford, O., Straaten, B., Wang, D., Parks, T., Campbell, E., Brierley, S.: Efficient quantum measurement of Pauli operators in the presence of finite sampling error. Quantum **5**, 385 (2021)

[47] Liang, S., Zhu, L., Liu, X., Yang, C., Li, X.: Artificial-intelligence-driven shot reduction in quantum measurement. Chem. Phys. Rev. **5**, 041403 (2024) https://doi.org/10.1063/5.0219663

[48] Wauters, M.M., El-Araby, E., Amin, M.H.: Reinforcement-learning-assisted quantum optimization. Phys. Rev. Res. **2**, 033446 (2020)

[49] Yao, J., Bukov, M., Lin, L.: Policy gradient based quantum approximate optimization algorithm. In: Proceedings of Machine Learning Research, vol. 107, pp. 605–634 (2020)

[50] Khairy, S., Shaydulin, R., Cincio, L., Alexeev, Y., Balaprakash, P.: Reinforcement-learning-based variational quantum circuits optimization for combinatorial problems (2020)

[51] Ostaszewski, M., Trenkwalder, L.M., Masarczyk, W., Scerri, E., Dunjko, V.: Reinforcement learning for optimization of variational quantum circuit architectures. In: Advances in Neural Information Processing Systems 34, pp. 18182–18194. Curran Associates, Red Hook, NY (2021)

[52] Skolik, A., Mangini, S., Bäck, T., Macchiavello, C., Dunjko, V.: Robustness of quantum reinforcement learning under hardware errors. EPJ Quantum Technol. **10**, 12 (2023)

[53] Altmann, P., Stein, J., Kölle, M., Bărligea, A., Gabor, T., Phan, T., Feld, S., Linnhoff-Popien, C.: Challenges for reinforcement learning in quantum circuit design (2024)

[54] Lucas, A.: Ising formulations of many NP problems. Front. Phys. **2**, 5 (2014)

[55] Goemans, M.X., Williamson, D.P.: Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. J. ACM **42**, 1115–1145 (1995)

[56] Ozaeta, A., Dam, W., McMahon, P.L.: Expectation values from the single-layer quantum approximate optimization algorithm on Ising problems. Quantum Science and Technology **7**(4), 045036 (2022) https://doi.org/10.1088/2058-9565/ac9f2b

[57] Kuo, E.-J., Fang, Y.-L.L., Chen, S.Y.-C.: Quantum architecture search via deep reinforcement learning (2021)

[58] Fösel, T., Niu, M.Y., Marquardt, F., Li, L.: Quantum circuit optimization with deep reinforcement learning (2021)

[59] Fösel, T., Tighineanu, P., Weiss, T., Marquardt, F.: Reinforcement learning with neural networks for quantum feedback. Phys. Rev. X **8**, 031084 (2018)

[60] Niu, M.Y., Boixo, S., Smelyanskiy, V.N., Neven, H.: Universal quantum control through deep reinforcement learning. npj Quantum Inf. **5**, 33 (2019)

[61] Bukov, M., Day, A.G.R., Sels, D., Weinberg, P., Polkovnikov, A., Mehta, P.: Reinforcement learning in different phases of quantum control. Phys. Rev. X **8**, 031086 (2018)

[62] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 2nd edn. MIT Press, Cambridge, MA (2018)

[63] Watkins, C.J.C.H., Dayan, P.: Q-learning. Mach. Learn. **8**, 279–292 (1992)

[64] Hasselt, H.: Double Q-learning. In: Lafferty, J.D., Williams, C.K.I., Shawe-Taylor, J., Zemel, R.S., Culotta, A. (eds.) Advances in Neural Information Processing Systems 23, pp. 2613–2621. Curran Associates, Red Hook, NY (2010)

[65] Altman, E.: Constrained Markov Decision Processes. Chapman & Hall/CRC, Boca Raton, FL (1999)

[66] Achiam, J., Held, D., Tamar, A., Abbeel, P.: Constrained policy optimization. In: Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, pp. 22–31. PMLR, Sydney, Australia (2017)

[67] Tessler, C., Mankowitz, D.J., Mannor, S.: Reward constrained policy optimization. In: International Conference on Learning Representations (2019)

[68] Aleksandrowicz, G., et al.: Qiskit: an open-source framework for quantum computing. Zenodo (2019). https://doi.org/10.5281/zenodo.2562111

[69] Kandala, A., Temme, K., Córcoles, A.D., Mezzacapo, A., Chow, J.M., Gambetta, J.M.: Error mitigation extends the computational reach of a noisy quantum processor. Nature **567**, 491–495 (2019)