QUANTUM MONTE CARLO TREE SEARCH WITH FIXED CONFIDENCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent advances in quantum computing are opening new opportunities for computationally intensive decision problems. This paper studies how quantum computing can improve Monte Carlo tree search (MCTS) in the fixed-confidence setting, where the goal is to identify a near-optimal move in a given game tree with high probability while minimizing query complexity. We first formulate MCTS under a quantum oracle model. We then develop a quantum MCTS algorithm (QMCTS) that combines threshold-based elimination on the search tree with quantum Monte Carlo estimation to reduce the cost of evaluating stochastic leaf values. We establish an instance-dependent lower bound on the query complexity and derive a corresponding upper bound for QMCTS. The lower-bound analysis introduces a new quantum phase-testing result that may also be useful beyond MCTS. We validate the theoretical results through simulation experiments and further demonstrate the feasibility of QMCTS on real quantum hardware.

1 INTRODUCTION

Monte Carlo three search (MCTS) is a search framework aiming to find the optimal decision based on the search tree built by random sampling of the decision space (Shah et al., 2022). It has achieved remarkable success in sequential decision making with a tree representation, such as the game of Go (Silver et al., 2017b), chess, and shogi (Silver et al., 2017a). The central statistical task is to compare candidate actions using noise simulation outcomes and take the optimal action. The effectiveness of MCTS, however, can be limited by the number of simulation required to identify the best move for a given node on the tree.

Quantum computing has advanced rapidly and has shown promise in areas such as combinatorial optimization (Abbas et al., 2024), simulation optimization (Hu et al., 2025; 2026), and machine learning (Biamonte et al., 2017). For certain computational tasks, quantum algorithms can achieve substantial speedups over their classical counterparts. This naturally raises the following question: *What can quantum computing offer to MCTS?* This question is relevant to a broad range of learning and decision-making problems in which MCTS serves as a key computational tool.

To answer this question, we adopt the fixed-confidence MCTS formulation of Kaufmann & Koolen (2017) and consider a two-player zero-sum game in which the possible sequences of play are represented by a maximin game tree. MAX nodes correspond to decisions made by the first player, while MIN nodes correspond to decisions made by the second player. The terminal payoffs are stochastic, and the objective is to identify a root move whose value is within $\epsilon > 0$ of the optimal root value with probability at least $1 - \delta$.

However, designing a quantum MCTS algorithm raises several fundamental technical challenges:

- **Quantum Information Extraction.** The advantage of quantum computing relies on processing information encoded in quantum superposition. However, extracting classical information (such as the mean estimate) requires *measurement*, which collapses the quantum state and prevents its further use. This constraints makes many classical sequential algorithms unsuitable in the quantum setting and requires new algorithmic designs.
- **Tree-Structured Exploration.** The recursive MAX-MIN structure couples the leaf values, and the effect of each leaf depends on its position in the tree. This structural dependence

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

makes it difficult to characterize the intrinsic difficulty of a problem instance and to obtain matching upper and lower bounds on the query complexity.

- **Algorithm Performance Analysis.** Classical query-complexity lower bounds often rely on change-of-measure arguments from sequential hypothesis testing (Lattimore & Szepesvári, 2020). These arguments do not directly extend to the quantum setting, so new tools are needed to establish instance-dependent lower bounds for quantum algorithms.

To aid the reader, we provide some intuitive interpretation of the quantum concepts used in this paper. Classical MCTS typically uses a simulation model to generate stochastic rollouts and estimate the values of left nodes and candidate moves. In the quantum setting, the simulation model is accessed through a quantum oracle model, which serves as the quantum counterpart of a classical simulator. A query to a classical simulator at a leaf node returns a random observation, while a query to a quantum oracle prepares a quantum sample, which is a superposition state that encodes the possible outcomes. Measuring this state produces a classical sample but collapses the quantum state and destroys the quantum advantage. Accordingly, classical sample complexity is replaced by quantum query complexity as the main measure of computational cost. This restriction on information extraction motivates the *lazy-measurement* principle used in our algorithm design.

In this work, we assume access to the quantum oracle model and focus on the design and analysis of MCTS algorithm. This assumption is natural when the underlying simulator is implemented by a classical program with accessible source code. Such a program can be represented as a Boolean circuit and implemented reversibly as a quantum circuit with polynomial overhead (Wang et al., 2021a). Similar assumptions are widely used in the quantum algorithm literature (Wang et al., 2021b; Zhong et al., 2023; Hu et al., 2025).

The main contributions of this paper are summarized as follows:

- We formulate quantum MCTS in the fixed-confidence setting and propose a quantum MCTS algorithm (QMCTS) that combines quantum Monte Carlo (QMC) estimation with threshold-based elimination under a lazy-measurement principle. We prove that QMCTS identifies an ϵ-optimal root move with probability at least $1 - \delta$.
- We establish an instance-dependent query-complexity upper bound for QMCTS and derive a lower bound for adaptive quantum algorithms. We characterize conditions under which the upper and lower bounds match up to logarithmic factors. Both analyses depend explicitly on the tree structure. The lower-bound analysis also introduces a new sequential quantum phase-testing result that may be useful beyond MCTS.
- We conduct numerical experiments to verify the theoretical results and evaluate the performance of the proposed algorithms. We also test the methods on a large tree constructed from the Lichess database (Lichess, 2026). Based on this experiment, we develop a classical-quantum hybrid algorithm that combines the strengths of classical and quantum MCTS algorithms. Finally, we implement a small experiment on IBM quantum hardware to demonstrate the feasibility of QMCTS.

Our study is related to two strands of literature:

Best Arm Identification. Best arm identification (BAI) is one of the most extensively studied problem in the bandit literature (Kaufmann et al., 2016; Garivier & Kaufmann, 2016). In the fixed-confidence setting, instance-dependent lower bounds and matching algorithms have been established (Degenne et al., 2019; Wang et al., 2021c). Recent work has also explored the use of quantum computing to improve BAI. Casalé et al. (2020) first formulated quantum BAI based on amplitude amplification. Wang et al. (2021b) established a quadratic speedup in query complexity under a strong quantum oracle model. More recently, Wang et al. (2025) studied BAI under quantum oracle models with different query structures. Our work differs from quantum BAI in two ways. First, we consider MCTS, where the goal is to identify the optimal move at the root, whose value is determined recursively by the MAX-MIN tree. This tree structure creates additional challenges for both algorithm design and query-complexity analysis. Second, we establish a new query-complexity lower bound for adaptive quantum MCTS algorithms. The BAI lower bound follows as a special case by considering a depth-one tree.

Monte Carlo Tree-Search. Many classical MCTS methods, including UCT, are built on regret-minimizing bandit algorithms (Kocsis & Szepesvári, 2006), while we consider a fixed-confidence formulation. Kaufmann & Koolen (2017) studied fixed-confidence MCTS and developed BAI-based algorithms with instance-dependent guarantees. Our work instead studies how quantum computing can reduce the query complexity of MCTS.

Research on quantum acceleration for MCTS and related tree-search methods remains limited. Peters (2021) applied quantum mean estimation to rollout values, while Sequeira et al. (2021) studied quantum Sparse Sampling with coherent access to environment transitions. Shukla (2026) constructed coherent rollout oracles to compare root actions through expected returns, and Finet et al. (2026) used a neutral-atom subroutine in the expansion step of MCTS. These methods do not directly address our setting, where the tree is fixed, uncertainty arises from stochastic leaf values, and root decisions are determined by recursive MAX-MIN backups rather than rollout expectations. We further provide a complete query-complexity analysis, with instance-dependent upper and lower bounds for identifying an ϵ-optimal root move.

2 Quantum Monte Carlo Tree Search

In this section, we first introduce the basic notation and concepts of quantum computing, and then formulate fixed-confidence MCTS under the quantum oracle model.

2.1 Preliminaries on Quantum Computing

Quantum computing exploits physical phenomena such as superposition, entanglement, and interference to process information. These mechanisms enable quantum parallelism and, when combined with suitable algorithmic procedures, can lead to significant speedups for certain computational tasks. We introduce the basic notation below and refer readers to (Nielsen & Chuang, 2010) for a comprehensive treatment.

In quantum computing, Dirac's bra-ket notation is used to represent vectors and linear operations in a complex Hilbert space. A quantum state is represented by a *ket* $|\psi\rangle \in \mathbb{C}^d$, and the corresponding *bra*, denoted by $\langle\psi|$, is defined as the conjugate transpose of the ket, i.e., $\langle\psi| = |\psi\rangle^\dagger$. For two quantum states $|\psi\rangle$ and $|\phi\rangle$, the quantity $\langle\phi|\psi\rangle$ denotes their inner product, while $|\psi\rangle\langle\phi|$ denotes their outer product and defines a linear operator. Quantum operations are represented by unitary operators U satisfying $U^\dagger U = UU^\dagger = I$, and their action on a quantum state is written as $|\psi\rangle \mapsto U|\psi\rangle$.

The basic unit of information in quantum computing is a *qubit* (quantum bit). A classical bit takes one of two values, 0 or 1, while a qubit is described by the corresponding computational basis states $|0\rangle$ and $|1\rangle$. Unlike a classical bit, a qubit can exist in a superposition of these basis states and is generally written as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ are the corresponding probability amplitudes satisfying $|\alpha|^2 + |\beta|^2 = 1$. Upon measurement, the qubit collapses to the state $|0\rangle$ with probability $|\alpha|^2$ and to the state $|1\rangle$ with probability $|\beta|^2$, respectively.

2.2 Problem Formulation

Consider a fixed game tree $\mathcal{T}$ with root node s_0. Each node is labeled as either MAX or MIN, with s_0 being a MAX node. For each node s, let $\mathcal{C}(s)$ denote the set of its children. Let $\mathcal{L}$ be the set of leaf nodes and $L := |\mathcal{L}|$. Each leaf $\ell \in \mathcal{L}$ is associated with a stochastic oracle that represents the leaf evaluation when ℓ is reached by the MCTS algorithm. For ease of presentation, we consider Bernoulli leaf oracles with unknown means $\mu_\ell \in (0, 1)$, where μ_ℓ represents the probability that the first player wins the game from leaf ℓ. Let $\mu := (\mu_\ell)_{\ell\in\mathcal{L}} \in (0, 1)^L$ denote the vector of leaf means. The proposed algorithms extend directly to leaf distributions supported on $[0, 1]$ (see Appendix D.2 for details).

The *value* of each node s (for the first player) is defined recursively as a function of μ

$$V_s(\mu) = \begin{cases} \mu_s, & s \in \mathcal{L}, \\ \max\limits_{c\in\mathcal{C}(s)} V_c(\mu), & s \text{ is a MAX node}, \\ \min\limits_{c\in\mathcal{C}(s)} V_c(\mu), & s \text{ is a MIN node}. \end{cases} \tag{1}$$

We suppress the dependence on μ when it is clear from the context. The optimal root move is the child of s_0 with the highest value

$$s^\star \in \arg\max_{s \in \mathcal{C}(s_0)} V_s. \tag{2}$$

with ties resolved according to a fixed ordering.

To identify $s^\star$ or an ϵ-optimal root move, an algorithm adaptively queries leaf nodes and observes their stochastic outcomes. After a random number τ of queries, the algorithm stops and recommends a move $\widehat{s} \in \mathcal{C}(s_0)$. Given a confidence level $\delta \in (0,1)$ and a precision parameter $\epsilon \in (0,1]$, the objective is to identify a move in $\mathcal{C}(s_0)$ whose value is within ϵ of the optimal root value with probability at least $1-\delta$, i.e.,

$$\mathbb{P}(V_{s_0} - V_{\widehat{s}} \le \epsilon) \ge 1 - \delta,$$

while minimizing the total number of queries. We formally introduce the quantum oracle model in Definition 1.

Definition 1 (Quantum oracle model)**.** *For each leaf $\ell \in \mathcal{L}$ with Bernoulli mean $\mu_\ell \in (0,1)$, we assume access to a unitary quantum oracle U_ℓ satisfying*

$$U_\ell|0\rangle = \sqrt{1-\mu_\ell}\,|0\rangle + \sqrt{\mu_\ell}\,|1\rangle. \tag{3}$$

The algorithm may query U_ℓ or its inverse $U_\ell^\dagger$, and each call is counted as one quantum query.

The quantum oracle model U_ℓ is the counterpart of a classical stochastic simulator. If $U_\ell|0\rangle$ is measured immediately, the outcome is 1 with probability μ_ℓ and 0 with probability $1-\mu_\ell$, which reproduces one classical Bernoulli sample from leaf ℓ. Before measurement, the quantum state can be processed coherently.

We next introduce several quantities that characterize the difficulty of the tree-search problem and will be used in the query-complexity analysis. Define the root performance gap

$$\Delta_\star = V_{s^\star} - \max_{s \in \mathcal{C}(s_0)\setminus\{s^\star\}} V_s \ge 0,$$

which measures the difference between the optimal root move and the best competing move. For an edge (s,c) directed from a node s to its child c, define the local edge gap

$$g(s,c) = |V_s - V_c| = \begin{cases} V_s - V_c, & s \text{ is a MAX node,} \\ V_c - V_s, & s \text{ is a MIN node.} \end{cases}$$

For a leaf $\ell \in \mathcal{L}$, let path(ℓ) denote the set of edges on the path from the root to ℓ, and define

$$\Delta_\ell = \max_{(s,c)\in \text{path}(\ell)} g(s,c),$$

which is the largest local edge gap along the path of ℓ.

3 Quantum Algorithm and Query Complexity Analysis

In this section, we first develop the quantum MCTS algorithm and its classical counterpart. We then provide a complete query-complexity analysis to characterize the resulting quantum advantage.

3.1 Quantum algorithm design

To develop an efficient quantum algorithm for MCTS, we consider three key issues: (a) using quantum methods to improve the accuracy of leaf-value evaluation; (b) designing the algorithm under the lazy-measurement principle; and (c) exploiting the tree structure to design an effective elimination rule. We first introduce the QMC estimator (Kothari & O'Donnell, 2023) in Lemma 1 and use it as a basic subroutine of QMCTS for leaf-value estimation. Compared with classical Monte Carlo method, QMC achieves a quadratic improvement in the dependence on the target accuracy while preserving the same confidence guarantee.

Lemma 1. *For each leaf $\ell \in \mathcal{L}$, there exists a quantum algorithm, denoted by* $\mathrm{QMC}(U_\ell, \alpha, \eta)$*, that uses $O(\frac{1}{\alpha}\log\frac{1}{\eta})$ queries to U_ℓ and returns an estimate $\widehat{\mu}_\ell$ satisfying*

$$\mathbb{P}\left(|\widehat{\mu}_\ell - \mu_\ell| > \alpha\right) \le \eta.$$

However, naively replacing classical Monte Carlo estimates with QMC estimates is not sufficient to obtain a quantum speedup for tree search. The main difficulty comes from the restriction on information extraction in quantum computing. Classical sequential MCTS methods (Kaufmann & Koolen, 2017) repeatedly use intermediate estimates to decide which nodes to sample or eliminate. In the quantum setting, obtaining such estimates requires measurement, which collapses the quantum state and may reduce the query advantage of QMC. On the other hand, delaying the elimination of clearly suboptimal moves can lead to many unnecessary queries. Therefore, a quantum MCTS algorithm needs to balance measurement frequency and adaptivity.

We design the quantum algorithm following a lazy-measurement principle. The algorithm proceeds in iterations, and the quantum state is measured only at the end of each iteration to obtain updated estimates. At iteration r, we first estimate the value of each active leaf and then propagate these estimates upward through the tree. Specifically, define

$$\widehat{V}_{s,r} = \begin{cases} \widehat{\mu}_{s,r}, & s \in \mathcal{L}_{r-1}, \\ \max\limits_{c \in \mathcal{C}_{r-1}(s)} \widehat{V}_{c,r}, & s \text{ is a MAX node}, \\ \min\limits_{c \in \mathcal{C}_{r-1}(s)} \widehat{V}_{c,r}, & s \text{ is a MIN node}, \end{cases} \tag{4}$$

where $\mathcal{L}_{r-1}$ is the set of active leaves and $\mathcal{C}_{r-1}(s)$ is the active children of node s at the beginning of iteration r.

For an active edge (s, c), define the empirical local gap by

$$\widehat{g}_r(s,c) = \begin{cases} \widehat{V}_{s,r} - \widehat{V}_{c,r}, & s \text{ is a MAX node}, \\ \widehat{V}_{c,r} - \widehat{V}_{s,r}, & s \text{ is a MIN node}, \end{cases} \tag{5}$$

which measures how far the child c is from the currently optimal child of node s.

We use the geometrically decreasing threshold $\gamma_r = 2^{-r}$ to eliminate suboptimal nodes. In each iteration, the internal nodes are processed in reverse topological order. For each active internal node s, any child c satisfying $\widehat{g}_r(s,c) > \gamma_r$ is removed together with all of its descendants. The remaining nodes then form the active subtree $\mathcal{T}_r$, with active leaf set $\mathcal{L}_r$. We set the QMC precision and confidence parameters to $\alpha_r = \gamma_r/2$ and $\eta_r = \delta/(2Lr^2)$, respectively. These choices allow clearly suboptimal nodes to be eliminated in early iterations without excessive query costs, while ensuring that all estimates used by the algorithm are accurate with probability at least $1 - \delta$ (see Appendix A for details). This high-probability event is used to establish the correctness of the algorithm.

After the elimination step, let $\widehat{s}_r$ denote the remaining root child with the largest estimated value. The algorithm stops when either only one root move remains or $\gamma_r \le \epsilon$, and returns $\widehat{s}_r$. We call this algorithm QMCTS and summarize it in Algorithm 1.

Algorithm 1 Quantum Monte Carlo Tree Search (QMCTS)

Require: Finite MAX–MIN tree $\mathcal{T}$, accuracy $\epsilon \in (0, 1]$, and confidence $\delta \in (0, 1/2)$
$\mathcal{T}_0 \leftarrow \mathcal{T}$ and $L \leftarrow |\mathcal{L}|$
for $r = 1, 2, \ldots$ **do**
  $\gamma_r \leftarrow 2^{-r}$, $\alpha_r \leftarrow \gamma_r/2$, and $\eta_r \leftarrow \delta/(2Lr^2)$
  for all $\ell \in \mathcal{L}_{r-1}$ **do**
    $\widehat{\mu}_{\ell,r} \leftarrow \mathrm{QMC}(U_\ell, \alpha_r, \eta_r)$
  end for
  Propagate $\widehat{V}_{s,r}$ upward using (4)
  for all internal nodes s, in reverse topological order **do**
    Remove each child c satisfying $\widehat{g}_r(s,c) > \gamma_r$
  end for
  Let $\mathcal{T}_r$ be the remaining connected subtree, with active leaf set $\mathcal{L}_r$
  $\widehat{s}_r \in \arg\max_{s \in \mathcal{C}_r(s_0)} \widehat{V}_{s,r}$
  if $|\mathcal{C}_r(s_0)| = 1$ **or** $\gamma_r \le \epsilon$ **then**
    return $\widehat{s}_r$
  end if
end for

Algorithm 2 Classical Monte Carlo Tree Search (CMCTS)

Require: Finite MAX-MIN tree $\mathcal{T}$, precision $\epsilon \in (0, 1]$, and confidence $\delta \in (0, 1/2)$

$\mathcal{T}_0 \leftarrow \mathcal{T}$ and $L \leftarrow |\mathcal{L}|$
$N_\ell \leftarrow 0$ for every $\ell \in \mathcal{L}$
for $r = 1, 2, \ldots$ **do**
  $\gamma_r \leftarrow 2^{-r}$, $\alpha_r \leftarrow \gamma_r/2$, and $\eta_r \leftarrow \delta/(2Lr^2)$
  $n_r \leftarrow \lceil \log(2/\eta_r)/(2\alpha_r^2) \rceil$
  for all $\ell \in \mathcal{L}_{r-1}$ **do**
    $\Delta_\ell \leftarrow \max\{0, n_r - N_\ell\}$
    Query leaf ℓ for Δ_ℓ additional observations and retain them
    $N_\ell \leftarrow N_\ell + \Delta_\ell$
    Construct $\widehat{\mu}^{\mathrm{C}}_{\ell,r}$ as the empirical mean of all N_ℓ retained observations
  end for
  Propagate $\widehat{V}^{\mathrm{C}}_{s,r}$ upward using (4)
  for all internal nodes s, in reverse topological order **do**
    Remove each child c satisfying $\widehat{g}^{\mathrm{C}}_r(s, c) > \gamma_r$
  end for
  Let $\mathcal{T}_r$ be the remaining connected subtree, with active leaf set $\mathcal{L}_r$
  $\widehat{s}^{\mathrm{C}}_r \in \arg\max_{s \in \mathcal{C}_r(s_0)} \widehat{V}^{\mathrm{C}}_{s,r}$
  if $|\mathcal{C}_r(s_0)| = 1$ **or** $\gamma_r \leq \epsilon$ **then**
    return $\widehat{s}^{\mathrm{C}}_r$
  end if
end for

For comparison, we also develop a classical counterpart of QMCTS, called CMCTS, summarized in Algorithm 2. CMCTS follows the same elimination framework as QMCTS. Its main advantage is that samples collected in previous iterations can be reused when updating the leaf estimates. At each iteration, given the same precision α_r and confidence η_r, CMCTS uses Hoeffding's inequality to determine the the number of additional samples required for each active leaf.

This motivates a Hybrid MCTS method that uses classical estimates until a fresh QMC estimate becomes more query-efficient than further classical sampling for each leaf. Since the query cost for each leaf depends only on the precision and confidence parameters, this comparison can be made before querying the oracles (see Algorithm 3 for details). The hybrid method avoids costly quantum estimation during coarse elimination while retaining the quantum query advantage at high precision.

3.2 Query Complexity Analysis

In this subsection, we analyze the query complexity of QMCTS and CMCTS. We first derive the upper bounds.

For each leaf $\ell \in \mathcal{L}$ and iteration $r \geq 1$, define the bad estimation event

$$F_{\ell,r} := \{\ell \in \mathcal{L}_{r-1} : |\widehat{\mu}_{\ell,r} - \mu_\ell| > \alpha_r\}.$$

If ℓ has already been eliminated, we set $F_{\ell,r} = \emptyset$. We then define the good event

$$\mathcal{G} = \bigcap_{r \geq 1} \bigcap_{\ell \in \mathcal{L}} F^c_{\ell,r},$$

under which every estimate used by the algorithm is within its prescribed precision α_r. Lemma 2 shows that the good event holds with probability at least $1 - \delta$.

Lemma 2. *The probability of the good event $\mathcal{G}$ is at least $1 - \delta$, that is, $\mathbb{P}(\mathcal{G}^c) \leq \delta$.*

On the good event $\mathcal{G}$, we can further establish two auxiliary results, as detailed in Appendix A. These results show that the estimation error remains controlled as values are propagated up the tree, and that the elimination step removes only suboptimal nodes while preserving at least one optimal child at every node.

Based on these results, Theorem 1 shows that QMCTS identifies an ϵ-optimal move with probability at least $1 - \delta$.

Theorem 1. *With probability at least $1-\delta$, QMCTS stops after finitely many iterations and returns an ϵ-optimal move for s_0, i.e.,*

$$\mathbb{P}(V_{s_0} - V_{\widehat{s}} \le \epsilon) \ge 1-\delta.$$

Theorem 2 establishes an upper bound on the query complexity of QMCTS.

Theorem 2. *With probability at least $1-\delta$, the total number of quantum oracle queries used by QMCTS satisfies*

$$Q = O\left(\sum_{\ell\in\mathcal{L}} \frac{1}{d_{\ell,\epsilon}} \left[\log\frac{2L}{\delta} + 2\log\left(2+\log_2\frac{1}{d_{\ell,\epsilon}}\right)\right]\right), \tag{6}$$

where $d_{\ell,\epsilon} = \Delta_\ell \vee \Delta_\star \vee \epsilon$.

The query complexity is determined by the effective difficulty $d_{\ell,\epsilon}$ of each leaf. QMCTS scales linearly with the inverse difficulty, up to logarithmic factors.

We also establishes the correctness of CMCTS and derive its query-complexity upper bound; see Theorems 5 and 6 in Appendix C. The classical query-complexity upper bound is

$$O\left(\sum_{\ell\in\mathcal{L}} \frac{1}{d_{\ell,\epsilon}^2} \left[\log\frac{4L}{\delta} + 2\log\left(2+\log_2\frac{1}{d_{\ell,\epsilon}}\right)\right]\right). \tag{7}$$

Comparing (6) and (7), QMCTS improves the dependence on the effective difficulty $d_{\ell,\epsilon}$ of each leaf from quadratic to linear. Thus, QMCTS achieves a quadratic speedup up to logarithmic factors.

We now turn to the query-complexity lower bound. Classical lower-bound analyses often rely on change-of-measure arguments from sequential hypothesis testing. These arguments do not directly extend to the quantum setting, so we develop a quantum counterpart.

Theorem 3 (Informal)**.** *Consider a problem instance μ for which the root decision depends on a collection of decision-relevant leaves. For any $\delta \in (0, 1/4]$, every adaptive quantum algorithm that identifies an ϵ-optimal root move with probability at least $1-\delta$ requires*

$$\mathbb{E}_\mu[\tau] = \Omega\left(\log\frac{1}{\delta}\sum_{\ell\in\mathcal{L}_{\mathrm{piv}}}\frac{1}{d_{\ell,\epsilon}}\right),$$

oracle queries. Here, $\mathcal{L}_{\mathrm{piv}}$ denotes the set of pivotal leaves; see Definition 2 in Appendix B.

The formal version of Theorem 3 is stated in Theorem 4 in Appendix B, where we specify the structural conditions required for the lower-bound analysis. The lower bound depends only on pivotal leaves because, in a tree-search problem, not every leaf can affect the root decision. Hence, the query complexity is determined by the parts of the tree that matter for distinguishing the best move. The dependence on the inverse difficulty $d_{\ell,\epsilon}$ is linear, which shows that the quadratic speedup for each pivotal leaf is optimal up to logarithmic factors.

Corollary 1 in the Appendix shows that, for uniformly pivotal instances with $\mathcal{L}_{\mathrm{piv}} = \mathcal{L}$, the upper and lower bounds match up to logarithmic factors. Thus, QMCTS is near-optimal in this case. This also highlights a structural difference between MCTS and BAI. In BAI, every arm can directly affect the final decision, which corresponds to the uniformly pivotal case. For a general MCTS problem instance, however, which leaves are pivotal is not known in advance. As a result, the algorithm may need to query leaves that eventually turn out to be nonpivotal. This extra cost comes from identifying the decision-relevant parts of the tree and makes the remaining gap between the upper and lower bounds difficult to close.

Technical Novelty. We conclude this section by highlighting the main technical ingredients in the query-complexity analysis. For the upper bound, we use induction on the tree to establish the validity of the elimination step and combine this argument with the QMC error guarantee to obtain the query-complexity bound. For the lower bound, we exploit the tree structure and introduce pivotal leaf blocks to characterize how individual leaves affect the root decision. We also develop a new sequential quantum phase-testing result (see Appendix B.1 for details) and reduce the MCTS identification problem to a collection of quantum phase-testing problems. This result is then used to derive the query-complexity lower bound and may be of independent interest for other quantum lower-bound analyses.

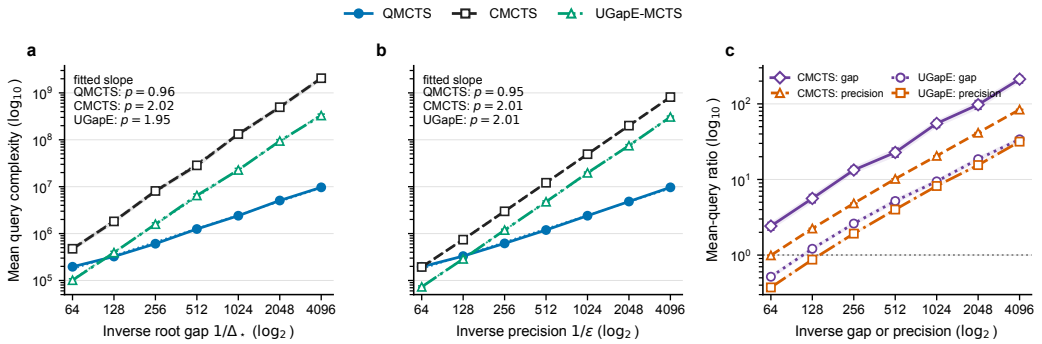


Figure 1: Query Complexity Scaling with Gap and Precision

4 Numerical Experiment

In this section, we conduct numerical experiments to evaluate QMCTS. We use synthetic experiments to verify the theoretical results, a large-scale chess experiment to access the scalability of the algorithm on a practical problem, and a real quantum computer experiment to examine the feasibility of implementing QMCTS on a quantum processor.

4.1 Synthetic Experiment

We first examine how the query complexity scales with the inverse root gap $1/\Delta_\star$ and the inverse precision $1/\epsilon$. We consider a fixed depth-two MAX-MIN tree in which the MAX root has eight actions, each leading to a MIN node with two leaves, for a total of 25 nodes and 16 Bernoulli leaves. We set $\delta = 0.05$ and run 100 independent replications for each setting. As a classical benchmark, we include UGapE-MCTS (Kaufmann & Koolen, 2017). In our implementation, QMCTS is simulated using standard quantum amplitude estimation as the QMC subroutine (Brassard et al., 2000).

Figure 1 reports the mean query complexity as the problem difficulty increases. In all tested settings, all three algorithms terminate and return an ϵ-optimal root action with a 100% empirical success rate. Panel (a) varies the inverse root gap with $\epsilon = 2^{-15}$ fixed, while panel (b) varies the inverse precision with $\Delta_\star = 2^{-15}$ fixed. The fitted log-log slopes are 0.96 and 0.95 for QMCTS, 2.02 and 2.01 for CMCTS, and 1.95 and 2.01 for UGapE-MCTS, respectively. Thus, QMCTS exhibits approximately linear scaling with the inverse gap and inverse precision, while both classical methods exhibit approximately quadratic scaling. These results are consistent with the theoretical analysis and support the quantum advantage over the classical benchmarks.

4.2 Large-Scale Chess Experiment

We next evaluate the algorithms on a large search tree constructed from the Lichess database (Lichess, 2026). Historical games that follow the same move sequence from the selected root position are merged into a common path in the tree. Each leaf represents games that reach the same terminal position in the constructed tree. From the root player's perspective, wins, draws, and losses are assigned scores of 1, $1/2$, and 0, respectively, and the leaf value is defined as the empirical mean score of the associated games. A leaf-oracle query returns an independent Bernoulli reward with this mean. The experiment can be viewed as an offline policy-improvement problem in which historical outcomes define the stochastic leaf rewards.

We first compare QMCTS and CMCTS on a depth-11 tree with 9509 nodes, 3072 leaves, and 10 root moves. The root performance gap is $\Delta_\star = 0.0020503$. We set $\delta = 0.05$, vary ϵ, and run 100 independent replications for each setting.

On this large tree, pure QMCTS is less query-efficient when the inverse precision is small. In the early iterations, thousands of leaves remain active, so QMCTS repeatedly applies QMC to many

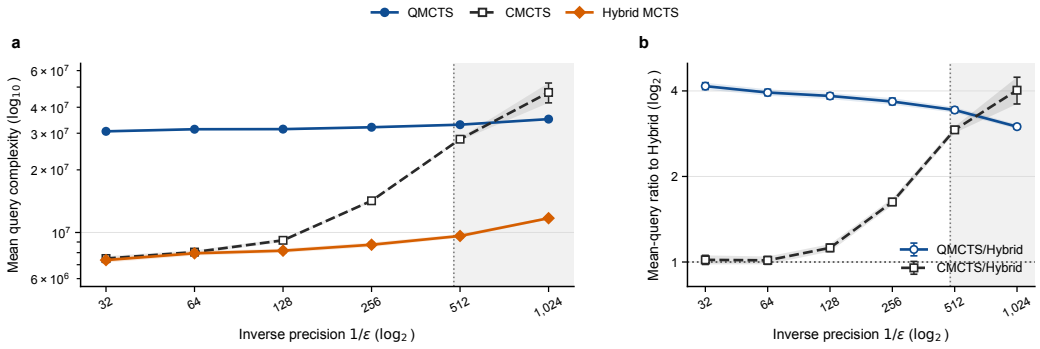


Figure 2: Large-scale chess query complexity.

leaves. In contrast, CMCTS requires only a small number of additional classical samples and reuses all previous observations. As a result, QMCTS incurs a large query cost before eliminating enough leaves.

Figure 2 reports the mean query complexity in the large-scale chess experiment. All methods terminate and return an ϵ-optimal root action with a 100% empirical success rate. Panel (a) shows that, at small inverse precision, CMCTS and Hybrid MCTS require substantially fewer queries than QMCTS. At $1/\epsilon = 32$, the mean query complexities are 30.64 million for QMCTS, 7.50 million for CMCTS, and 7.37 million for Hybrid MCTS. As the inverse precision increases, the query cost of CMCTS grows rapidly, while QMCTS benefits from its better scaling. At $1/\epsilon = 1024$, the corresponding query complexities are 35.07, 47.11, and 11.71 million. Panel (b) reports the ratios relative to Hybrid MCTS. At $1/\epsilon = 1024$, the QMCTS-to-Hybrid and CMCTS-to-Hybrid ratios are 2.994 and 4.022, respectively. These results show that Hybrid MCTS reduces the overall query cost by using classical sampling when the inverse precision is small and switching to quantum estimation as the inverse precision increases.

4.3 Real Experiment

We further evaluate QMCTS on an IBM Quantum processor using Qiskit (Javadi-Abhari et al., 2024). We consider a depth-two MAX–MIN tree with two root moves and four Bernoulli leaves, with $\Delta_\star = 0.01640625$, $\epsilon = 2^{-6}$, and $\delta = 0.05$.

We run QMCTS and CMCTS independently five times. Both algorithms terminate and return the unique ϵ-optimal root move in all replications. The mean query complexities are 68352 for QMCTS and 306444 for CMCTS. Thus, CMCTS uses about 4.48 times more oracle queries, corresponding to a 77.70% reduction for QMCTS. These results provide a small-scale hardware demonstration that QMCTS maintains correctness and a query-complexity advantage on a physical quantum processor.

5 Conclusion and Limitations

In this paper, we study how quantum computing can improve MCTS. We develop a quantum algorithm, QMCTS, and establish its correctness and query-complexity guarantees. Numerical experiments show a clear query advantage over classical benchmarks. The main limitations are twofold. First, the gap between the upper and lower query-complexity bounds remains open for general tree instances, and we conjecture that closing this gap may be difficult because of the tree structure of the problem. Second, query complexity does not fully capture the gate complexity and implementation cost on real quantum hardware. Future work may extend the framework to more general tree-search models, including broader leaf-value distributions, stochastic transitions, and additional sources of uncertainty, and develop hardware-level complexity analyses that account for circuit depth and gate complexity.

AI USE STATEMENT

In this work, we used generative AI tools to correct spelling and grammatical errors and improve readability, to assist in writing code to implement the methods and debug the implementations, and to search for related literature. We did not use generative AI tools for other research tasks requiring disclosure, including formulating the research ideas or hypotheses, developing the theoretical framework, formulating mathematical claims, providing critical ingredients for or writing proofs, designing the research methodology or experiments, or interpreting the results; these tasks were carried out by the authors. All AI-assisted outputs were reviewed and verified by the authors. We take responsibility for the final content of this work, including the text, claims, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We make several efforts to support the reproducibility of our results. The source code used to implement the proposed methods and reproduce the numerical experiments is provided in the online supplementary materials. For the theoretical results, all technical assumptions are stated explicitly in the main text, and complete proofs of the theoretical claims are provided in the appendix.

REFERENCES

Amira Abbas, Andris Ambainis, Brandon Augustino, Andreas Bärtschi, Harry Buhrman, Carleton Coffrin, Giorgio Cortiana, Vedran Dunjko, Daniel J Egger, Bruce G Elmegreen, et al. Challenges and opportunities in quantum optimization. *Nature Reviews Physics*, 6(12):718–735, 2024.

Robert Beals, Harry Buhrman, Richard Cleve, Michele Mosca, and Ronald De Wolf. Quantum lower bounds by polynomials. *Journal of the ACM (JACM)*, 48(4):778–797, 2001.

Jacob Biamonte, Peter Wittek, Nicola Pancotti, Patrick Rebentrost, Nathan Wiebe, and Seth Lloyd. Quantum machine learning. *Nature*, 549(7671):195–202, 2017.

Peter Borwein and Tamás Erdélyi. *Polynomials and polynomial inequalities*. Springer Science & Business Media, 2012.

Gilles Brassard, Peter Hoyer, Michele Mosca, and Alain Tapp. Quantum amplitude amplification and estimation. *arXiv preprint quant-ph/0005055*, 2000.

Balthazar Casalé, Giuseppe Di Molfetta, Hachem Kadri, and Liva Ralaivola. Quantum bandits. *Quantum Machine Intelligence*, 2(1):11, 2020.

Rémy Degenne, Wouter M Koolen, and Pierre Ménard. Non-asymptotic pure exploration by solving games. *Advances in Neural Information Processing Systems*, 32, 2019.

Yohan Finet, Yves Bérubé-Lauzière, and Victor Drouin-Touchette. Quantum-enhanced monte carlo tree search framework for combinatorial optimization problems. *arXiv preprint arXiv:2606.30415*, 2026.

Aurélien Garivier and Emilie Kaufmann. Optimal best arm identification with fixed confidence. In *Conference on Learning Theory*, pp. 998–1027. PMLR, 2016.

Mingjie Hu, Jianqiang Hu, and Tianxiang Wang. Quantum ranking and selection under stochastic constraints. 2025. Available at SSRN 5948136.

Mingjie Hu, Jian-qiang Hu, and Enlu Zhou. Quantum grover adaptive search for discrete simulation optimization. *arXiv preprint arXiv:2604.26277*, 2026.

Ali Javadi-Abhari, Matthew Treinish, Kevin Krsulich, Christopher J Wood, Jake Lishman, Julien Gacon, Simon Martiel, Paul D Nation, Lev S Bishop, Andrew W Cross, et al. Quantum computing with qiskit. *arXiv preprint arXiv:2405.08810*, 2024.

Emilie Kaufmann and Wouter Koolen. Monte-carlo tree search by best arm identification. *Advances in Neural Information Processing Systems*, 30, 2017.

Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. On the complexity of best-arm identification in multi-armed bandit models. *The Journal of Machine Learning Research*, 17(1):1–42, 2016.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *European conference on machine learning*, pp. 282–293. Springer, 2006.

Robin Kothari and Ryan O'Donnell. Mean estimation when you have the source code; or, quantum monte carlo methods. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pp. 1186–1215. SIAM, 2023.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Lichess. Lichess open database. `https://database.lichess.org/`, 2026.

Nikhil S Mande and Ronald de Wolf. Tight bounds for quantum phase estimation and related problems. *Quantum*, 10:2140, 2026.

Michael A Nielsen and Isaac L Chuang. *Quantum computation and quantum information*. Cambridge university press, 2010.

Orson R. L. Peters. Quantum speedups for monte carlo tree search. Master's thesis, Leiden University, 2021.

André Sequeira, Luis Paulo Santos, and Luis Soares Barbosa. Quantum tree-based planning. *IEEE Access*, 9:125416–125427, 2021.

Devavrat Shah, Qiaomin Xie, and Zhi Xu. Nonasymptotic analysis of monte carlo tree search. *Operations research*, 70(6):3234–3260, 2022.

Nishant Shukla. Coherent rollout oracles for finite-horizon sequential decision problems. *arXiv preprint arXiv:2604.25962*, 2026.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017a.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017b.

Daochen Wang, Aarthi Sundaram, Robin Kothari, Ashish Kapoor, and Martin Roetteler. Quantum algorithms for reinforcement learning with a generative model. In *International Conference on Machine Learning*, pp. 10916–10926. PMLR, 2021a.

Daochen Wang, Xuchen You, Tongyang Li, and Andrew M Childs. Quantum exploration algorithms for multi-armed bandits. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 10102–10110, 2021b.

Po-An Wang, Ruo-Chun Tzeng, and Alexandre Proutiere. Fast pure exploration via frank-wolfe. *Advances in Neural Information Processing Systems*, 34:5810–5821, 2021c.

Xuchuang Wang, Yu-Zhen Janice Chen, Matheus Guedes de Andrade, Jonathan Allcock, Mohammad Hajiesmaili, John CS Lui, and Don Towsley. Quantum best arm identification with quantum oracles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 21321–21329, 2025.

Han Zhong, Jiachen Hu, Yecheng Xue, Tongyang Li, and Liwei Wang. Provably efficient exploration in quantum reinforcement learning with logarithmic worst-case regret. *arXiv preprint arXiv:2302.10796*, 2023.

A Query Complexity Analysis

Proof of Lemma 2. Let $\mathcal{F}_{r-1}$ be the sigma-field generated by all operations and measurement outcomes before iteration r. Condition on $\mathcal{F}_{r-1}$, according to the Lemma 1, the estimators satisfy

$$\mathbb{P}(F_{\ell,r}|\mathcal{F}_{r-1}) \leq \mathbb{I}(\ell \in \mathcal{L}_{r-1})\eta_r \leq \eta_r.$$

Therefore, a union bound gives

$$\mathbb{P}(\mathcal{G}^c) \leq \sum_{r\geq 1}\sum_{\ell\in\mathcal{L}} \eta_r = \frac{\delta}{2}\sum_{r\geq 1}\sum_{\ell\in\mathcal{L}} \frac{1}{Lr^2} = \frac{\pi^2}{12}\delta < \delta.$$

□

For an active node before iteration r, let V_s^{r-1} be its true value computed on the active subtree $\mathcal{T}_{r-1}$.

Lemma 3. *On the good event $\mathcal{G}$, for each iteration r and every node s active before iteration r,*

$$\left|\widehat{V}_{s,r} - V_s^{(r-1)}\right| \leq \alpha_r.$$

Proof. Fix an iteration r. We prove the claim by induction on the height of s in the active tree $\mathcal{T}_{r-1}$, where the height of a node is the maximum number of edges from that node to an active descendant leaf.

For an active leaf $\ell \in \mathcal{L}_{r-1}$, we have $\widehat{V}_{\ell,r} = \widehat{\mu}_{\ell,r}$, and $V_\ell^{(r-1)} = \mu_\ell$. By the definition of the good event $\mathcal{G}$,

$$\left|\widehat{V}_{\ell,r} - V_\ell^{(r-1)}\right| \leq \alpha_r.$$

Now consider an active internal node s. Assume that the claim holds for every active child $c \in \mathcal{C}_{r-1}(s)$.

Suppose first that s is a MAX node. By the recursive definitions of the empirical and true values,

$$\widehat{V}_{s,r} = \max_{c\in\mathcal{C}_{r-1}(s)} \widehat{V}_{c,r}, \qquad V_s^{(r-1)} = \max_{c\in\mathcal{C}_{r-1}(s)} V_c^{(r-1)}.$$

The induction hypothesis implies that, for every active child c,

$$V_c^{(r-1)} - \alpha_r \leq \widehat{V}_{c,r} \leq V_c^{(r-1)} + \alpha_r.$$

Taking the maximum over $c \in \mathcal{C}_{r-1}(s)$ yields

$$V_s^{(r-1)} - \alpha_r \leq \widehat{V}_{s,r} \leq V_s^{(r-1)} + \alpha_r.$$

Therefore,

$$\left|\widehat{V}_{s,r} - V_s^{(r-1)}\right| \leq \alpha_r.$$

If s is a MIN node, then

$$\widehat{V}_{s,r} = \min_{c\in\mathcal{C}_{r-1}(s)} \widehat{V}_{c,r}, \qquad V_s^{(r-1)} = \min_{c\in\mathcal{C}_{r-1}(s)} V_c^{(r-1)}.$$

Applying the same inequalities and taking the minimum gives

$$V_s^{(r-1)} - \alpha_r \leq \widehat{V}_{s,r} \leq V_s^{(r-1)} + \alpha_r,$$

and thus

$$\left|\widehat{V}_{s,r} - V_s^{(r-1)}\right| \leq \alpha_r.$$

By induction, the claim holds for every node active before iteration r. □

Lemma 4. *On the good event $\mathcal{G}$, every deleted child is strictly suboptimal for its parent in the active true tree. Moreover, at least one optimal child of every active internal node is retained. Consequently, after every iteration, $V_s^{(r)} = V_s^{(r-1)} = V_s$ for every retained node s.*

Proof. We proceed by induction over the iterations. Before the first iteration, the active tree is the original tree, so the result holds.

Fix iteration r and suppose that $V_s^{(r-1)} = V_s$ for every node active before this iteration.

Consider first a MAX node s. If a child c is deleted, then $\widehat{V}_{s,r} - \widehat{V}_{c,r} > \gamma_r$. By Lemma 3 and $2\alpha_r = \gamma_r$,

$$V_s^{(r-1)} - V_c^{(r-1)} \geq \widehat{V}_{s,r} - \widehat{V}_{c,r} - 2\alpha_r > \gamma_r - 2\alpha_r = 0.$$

Hence, c is strictly suboptimal for s.

Let $c^\star$ be an optimal child of s, so that $V_{c^\star}^{(r-1)} = V_s^{(r-1)}$. Then

$$\widehat{V}_{s,r} - \widehat{V}_{c^\star,r} \leq 2\alpha_r = \gamma_r,$$

so $c^\star$ is not deleted.

The argument for a MIN node is identical, with the inequalities reversed. Therefore, every active internal node retains at least one child attaining its true MAX or MIN value.

Applying this argument from the deepest internal nodes toward the root, we have $V_s^{(r)} = V_s^{(r-1)} = V_s$. The conclusion follows by induction over r. □

A.1 Proof of Theorem 1

Proof. Because $\gamma_r = 2^{-r}$, we have $\gamma_r \to 0$ as $r \to \infty$. Since $\epsilon > 0$, there exists a finite iteration

$$R_\epsilon := \min\{r \geq 1 : \gamma_r \leq \epsilon\}.$$

The stopping condition of Algorithm 1 is satisfied no later than iteration R_ϵ. Hence, the algorithm terminates after finitely many iterations.

Condition on the good event $\mathcal{G}$. By Lemma 4, threshold elimination preserves the true value of every retained node. Moreover, at least one optimal root move remains active after every iteration. Consequently, for every iteration r, there exists an active root move $s_r^\star \in \mathcal{C}_r(s_0)$ such that $V_{s_r^\star} = V_{s_0}$.

Consider the two possible stopping conditions. First, suppose that $|\mathcal{C}_\tau(s_0)| = 1$. Since at least one optimal root move remains active, the unique remaining move must be optimal. Therefore,

$$V_{s_0} - V_{\widehat{s}} = 0 \leq \epsilon.$$

Now suppose that the algorithm stops because $\gamma_\tau \leq \epsilon$. The returned move satisfies

$$\widehat{s} \in \arg\max_{s \in \mathcal{C}_\tau(s_0)} \widehat{V}_{s,\tau}.$$

Let $s_\tau^\star$ be an optimal active root move. Then $\widehat{V}_{\widehat{s},\tau} \geq \widehat{V}_{s_\tau^\star,\tau}$. By Lemma 3, and because the true values of retained nodes are preserved by Lemma 4,

$$\left|\widehat{V}_{s,\tau} - V_s\right| \leq \alpha_\tau$$

for every root move s active at iteration τ. Hence,

$$\begin{aligned}
V_{s_0} - V_{\widehat{s}} &= V_{s_\tau^\star} - V_{\widehat{s}} \\
&= \left(V_{s_\tau^\star} - \widehat{V}_{s_\tau^\star,\tau}\right) + \left(\widehat{V}_{s_\tau^\star,\tau} - \widehat{V}_{\widehat{s},\tau}\right) + \left(\widehat{V}_{\widehat{s},\tau} - V_{\widehat{s}}\right) \\
&\leq \alpha_\tau + 0 + \alpha_\tau \\
&= 2\alpha_\tau \\
&\leq \gamma_\tau \\
&\leq \epsilon.
\end{aligned}$$

Thus, on the event $\mathcal{G}$, the algorithm returns an ϵ-optimal root move. Since $\mathbb{P}(\mathcal{G}) \geq 1-\delta$, we conclude that

$$\mathbb{P}(V_{s_0} - V_{\widehat{s}} \leq \epsilon) \geq \mathbb{P}(\mathcal{G}) \geq 1 - \delta.$$

□

A.2 Proof of Theorem 2

Proof. We prove the query bound on the good event $\mathcal{G}$.

For each leaf $\ell \in \mathcal{L}$, let R_ℓ denote the last iteration in which ℓ is queried. Suppose first that $R_\ell \geq 2$. Since ℓ is queried in iteration R_ℓ, it must remain active after iteration $R_\ell - 1$ and iteration $R_\ell - 1$ is nonterminal.

Consider any edge (s, c) on the active path from the root to ℓ. Since this edge survives iteration $R_\ell - 1$, its empirical gap satisfies $\widehat{g}_{R_\ell - 1}(s, c) \leq \gamma_{R_\ell - 1}$. By Lemma 3 and Lemma 4

$$\left|\widehat{V}_{u, R_\ell - 1} - V_u\right| \leq \alpha_{R_\ell - 1}$$

for every active node u. Therefore,

$$g(s, c) \leq \widehat{g}_{R_\ell - 1}(s, c) + 2\alpha_{R_\ell - 1}.$$

Since $2\alpha_r = \gamma_r$, it follows that $g(s, c) \leq 2\gamma_{R_\ell - 1}$. Taking the maximum over all edges on the path to ℓ gives $\Delta_\ell \leq 2\gamma_{R_\ell - 1}$.

Because iteration $R_\ell - 1$ is nonterminal, we have $\epsilon < \gamma_{R_\ell - 1} \leq 2\gamma_{R_\ell - 1}$.

Furthermore, at least two root moves remain after iteration $R_\ell - 1$. By Lemma 4, at least one optimal root move remains active. Let $s^\star$ be such an optimal move and let $s \neq s^\star$ be another active root move. Since the edge (s_0, s) has not been eliminated,

$$V_{s^\star} - V_s \leq 2\gamma_{R_\ell - 1}.$$

By the definition of the root gap,

$$\Delta_\star \leq V_{s^\star} - V_s \leq 2\gamma_{R_\ell - 1}.$$

Combining these inequalities yields

$$d_{\ell,\epsilon} = \Delta_\ell \vee \Delta_\star \vee \epsilon \leq 2\gamma_{R_\ell - 1}.$$

Since $\gamma_r = 2^{-r}$, $\gamma_{R_\ell - 1} = 2\gamma_{R_\ell}$, and thus

$$d_{\ell,\epsilon} \leq 4\gamma_{R_\ell}. \tag{8}$$

Equivalently,

$$\frac{1}{\gamma_{R_\ell}} \leq \frac{4}{d_{\ell,\epsilon}}. \tag{9}$$

If $R_\ell = 1$, then

$$\frac{1}{\gamma_{R_\ell}} = 2 \leq \frac{4}{d_{\ell,\epsilon}},$$

because $d_{\ell,\epsilon} \leq 1$. Hence, equation 9 holds for every leaf.

Equation equation 8 also implies $d_{\ell,\epsilon} \leq 4 \cdot 2^{-R_\ell}$, so

$$R_\ell \leq 2 + \log_2 \frac{1}{d_{\ell,\epsilon}}. \tag{10}$$

We now bound the queries allocated to leaf ℓ. In iteration r, according to Lemma 1, QMC uses at most $\frac{C_{\mathrm{Q}}}{\alpha_r} \log \frac{1}{\eta_r}$ queries to U_ℓ and $U_\ell^\dagger$, where $C_Q > 0$ is constant. Since $\alpha_r = \gamma_r/2, \eta_r = \delta/2Lr^2$, the total query complexity associated with ℓ satisfies

$$Q_\ell \leq C_{\mathrm{Q}} \sum_{r=1}^{R_\ell} \frac{1}{\alpha_r} \log \frac{1}{\eta_r} = 2C_{\mathrm{Q}} \sum_{r=1}^{R_\ell} \frac{1}{\gamma_r} \left[\log \frac{2L}{\delta} + 2 \log r\right].$$

Then, we have

$$Q_\ell \leq 2C_{\mathrm{Q}} \left[\log \frac{2L}{\delta} + 2 \log R_\ell\right] \sum_{r=1}^{R_\ell} \frac{1}{\gamma_r}.$$

Since $\gamma_r = 2^{-r}$,

$$\sum_{r=1}^{R_\ell} \frac{1}{\gamma_r} = \sum_{r=1}^{R_\ell} 2^r < 2^{R_\ell+1} = \frac{2}{\gamma_{R_\ell}}.$$

Consequently,

$$Q_\ell < \frac{4C_{\mathrm{Q}}}{\gamma_{R_\ell}} \left[\log \frac{2L}{\delta} + 2 \log R_\ell \right].$$

Applying equation 9 and equation 10 gives

$$Q_\ell \leq \frac{16C_{\mathrm{Q}}}{d_{\ell,\epsilon}} \left[\log \frac{2L}{\delta} + 2 \log \left(2 + \log_2 \frac{1}{d_{\ell,\epsilon}} \right) \right].$$

Finally, every oracle query is associated with exactly one active leaf. Summing the preceding inequality over $\ell \in \mathcal{L}$ proves equation 6. □

B Query Complexity Lower Bound

B.1 Sequential Quantum Phase Hypothesis Testing

In this subsection, we present the quantum-information-theoretic tool used in the lower-bound analysis. This is independent of the tree search problem and considers only a sequential hypothesis-testing problem for an unknown phase.

Let

$$P_\theta = |0\rangle\langle 0| + e^{i\theta}|1\rangle\langle 1|, \qquad \theta \in [-\pi, \pi),$$

be a one-qubit phase oracle. Fix a θ_{c} and a measurable set $\Theta \subseteq [-\pi, \pi)$ containing θ_{c}. We consider the testing problem

$$H_0 : \theta = \theta_{\mathrm{c}} \qquad \text{vs} \qquad H_1 : \theta \in [-\pi, \pi) \setminus \Theta.$$

The set Θ plays the role of an *indifference region*: when $\theta \in \Theta \setminus \{\theta_{\mathrm{c}}\}$, the testing procedure is not required to make either decision correctly. The procedure must distinguish θ_{c} from every phase outside Θ.

Let Leb denote the Lebesgue measure on $\mathbb{R}$ (i.e., length). Define $s := \mathrm{Leb}(\Theta)$. A smaller s places admissible alternatives closer to the reference phase and makes the two hypotheses harder to distinguish. For example, if $\Theta = [\theta_{\mathrm{c}} - h, \theta_{\mathrm{c}} + h]$, then the test is required to distinguish θ_{c} from all phases satisfying $|\theta - \theta_{\mathrm{c}}| > h$, and $s = 2h$.

Consider an arbitrary sequential quantum procedure. Between oracle queries, the procedure may apply arbitrary θ-independent quantum operations and measurements, and its subsequent operations may depend on previous measurement outcomes. The procedure may query P_θ, $P_\theta^\dagger$ and stops at a random time. Upon stopping, it outputs $D \in \{0, 1\}$, where $D = 0$ corresponds to accepting H_0 and $D = 1$ corresponds to accepting H_1.

For a prescribed error level $\zeta \in (0, 1/4]$, we require $\mathbb{P}_{\theta_{\mathrm{c}}}(D = 1) \leq \zeta$ and $\sup_{\theta \notin \Theta} \mathbb{P}_\theta(D = 0) \leq \zeta$.

Let N denote the random number of phase-oracle queries made by the procedure before it stops. The quantity $\mathbb{E}_{\theta_{\mathrm{c}}}[N]$ is therefore the average number of phase-oracle queries used when the true phase is θ_{c}.

Lemma 5. *Under the testing set above, if $s \in (0, \frac{\pi}{2}]$, then*

$$\mathbb{E}_{\theta_{\mathrm{c}}}[N] \geq \frac{1}{16s} \log \frac{1}{4\zeta} - \frac{1}{2}.$$

Proof. If $\zeta > 1/4$, then

$$\frac{1}{16s} \log \frac{1}{4\zeta} - \frac{1}{2} < 0,$$

and the claimed bound follows immediately.

For a fixed integer $n \geq 0$, define $a_n(\theta) := \mathbb{P}_\theta(D = 0,\ N \leq n)$, which is the acceptance probability of the test truncated after n queries.

By the polynomial method for quantum phase queries, $a_n(\theta)$ is a real trigonometric polynomial of degree at most $2n$; see, e.g., Beals et al. (2001); Mande & de Wolf (2026). Moreover, for every $\theta \notin \Theta$,

$$a_n(\theta) \leq \mathbb{P}_\theta(D = 0) \leq \zeta.$$

Since the set Θ has total angular length s, a trigonometric Remez inequality gives $a_n(\theta_c) \leq \zeta e^{8ns}$; see, e.g., Borwein & Erdélyi (2012); Mande & de Wolf (2026).

Choose

$$n = \left\lfloor \frac{1}{8s} \log \frac{1}{4\zeta} \right\rfloor.$$

Then $\zeta e^{8ns} \leq \frac{1}{4}$, and hence $a_n(\theta_c) \leq \frac{1}{4}$. On the other hand, correctness under the null hypothesis implies $\mathbb{P}_{\theta_c}(D = 0) \geq 1 - \zeta$. Therefore,

$$\begin{aligned}
\mathbb{P}_{\theta_c}(N > n) &\geq \mathbb{P}_{\theta_c}(D = 0,\ N > n) \\
&= \mathbb{P}_{\theta_c}(D = 0) - a_n(\theta_c) \\
&\geq 1 - \zeta - \frac{1}{4} \\
&\geq \frac{1}{2},
\end{aligned}$$

where the last inequality uses $\zeta \leq 1/4$.

Consequently,

$$\mathbb{E}_{\theta_c}[N] \geq n\, \mathbb{P}_{\theta_c}(N > n) \geq \frac{n}{2}.$$

Finally, using $\lfloor x \rfloor \geq x - 1$,

$$\mathbb{E}_{\theta_c}[N] \geq \frac{1}{16s} \log \frac{1}{4\zeta} - \frac{1}{2}.$$

□

B.2 Lower Bound Analysis

In this subection, we reduce the quantum MCTS identification problem to a sequential quantum phase hypothesis-testing problem and use the Lemma 5 to derive the query complexity lower bound.

For any leaf-mean vector $\nu \in (0,1)^L$, define the set of ϵ-optimal moves at the root s_0 by

$$\mathcal{M}_\epsilon(\nu) := \{s \in \mathcal{C}(s_0) : V_{s_0}(\nu) - V_s(\nu) \leq \epsilon\}. \tag{11}$$

The reduction starts by considering a family of leaf-mean vectors $\widetilde{\mu}(\theta), \theta \in [-\pi, \pi)$, constructed so that, for some reference phase θ_c,

$$\widetilde{\mu}(\theta_c) = \mu. \tag{12}$$

If

$$\mathcal{M}_\epsilon(\mu) \cap \mathcal{M}_\epsilon\big(\widetilde{\mu}(\theta)\big) = \emptyset, \tag{13}$$

then no root move is simultaneously a valid recommendation for the original instance and the perturbed instance. Therefore, any algorithm that is (ϵ, δ)-correct on both instances must distinguish between them.

Define

$$\Theta := \left\{\theta \in [-\pi, \pi) : \mathcal{M}_\epsilon(\mu) \cap \mathcal{M}_\epsilon\big(\widetilde{\mu}(\theta)\big) \neq \emptyset\right\}. \tag{14}$$

At $\theta = \theta_c$ the instance is μ, while every $\theta \notin \Theta$ produces an instance whose valid root recommendations are disjoint from those of μ. Consequently, a correct tree-search algorithm can be used to test

$$H_0 : \theta = \theta_c \qquad \text{vs} \qquad H_1 : \theta \notin \Theta.$$

In the Monte Carlo tree search problem, changing the mean of a single leaf does not necessarily change the optimal move at the root, because its effect may be masked by the MAX–MIN operations

along the path to the root. Therefore, we group leaf nodes into blocks in Definition 2 and consider joint perturbations of the leaves within each block, so that the perturbation can change the set of ϵ-optimal moves at the root.

Definition 2. *Fix a tree instance $\mu = (\mu_\ell)_{\ell\in\mathcal{L}}$, an accuracy level $\epsilon \in (0, 1]$, and a constant $C > 0$. A collection of nonempty leaf sets $\mathcal{B}_1, \dots, \mathcal{B}_m \subseteq \mathcal{L}$ is called a* disjoint C-pivotal-block family *at μ if the blocks are pairwise disjoint and there exists a common reference phase θ_{c} such that, for every block $\mathcal{B}_j$, there exists a valid leaf-mean vectors*

$$\mu^{(j)}(\theta) = \big(\mu_\ell^{(j)}(\theta)\big)_{\ell\in\mathcal{L}}, \qquad \theta \in [-\pi, \pi),$$

satisfying the following properties.

1. *Under θ_{c}, the perturbed instance is exactly the original instance: $\mu^{(j)}(\theta_{\mathrm{c}}) = \mu$. For an arbitrary phase θ, every leaf outside the block remains unchanged: $\mu_\ell^{(j)}(\theta) = \mu_\ell, \forall\, \ell \notin \mathcal{B}_j$. Hence, when $\theta \neq \theta_{\mathrm{c}}$, only the means of leaves in $\mathcal{B}_j$ are allowed to differ from those of the original instance.*

2. *For every $\ell \in \mathcal{B}_j$, let $U_\ell^{(j)}(\theta)$ denote the quantum leaf oracle corresponding to the Bernoulli mean $\mu_\ell^{(j)}(\theta)$. One call to $U_\ell^{(j)}(\theta)$, its inverse, or a controlled version can be implemented using at most one call to P_θ, its inverse, or a controlled version, together with quantum operations independent of θ.*

3. *Define the decision-preserving region*

$$\Theta_j := \Big\{\theta \in [-\pi, \pi) : \mathcal{M}_\epsilon(\mu) \cap \mathcal{M}_\epsilon\big(\mu^{(j)}(\theta)\big) \neq \emptyset\Big\},$$

and let $d_{\mathcal{B}_j} := \min_{\ell\in\mathcal{B}_j} d_{\ell,\epsilon}$. The perturbation family satisfies

$$0 < \mathrm{Leb}(\Theta_j) \le \min\Big\{\frac{\pi}{2}, C d_{\mathcal{B}_j}\Big\}.$$

Let

$$\mathcal{L}_{\mathrm{piv}} := \bigcup_{j=1}^{m} \mathcal{B}_j, \qquad b := \max_{1\le j\le m} |\mathcal{B}_j|.$$

For $x \in \mathbb{R}$, write $[x]_+ := \max\{x, 0\}$.

Theorem 4. *Suppose that μ admits a disjoint C-pivotal-block family with C and b bounded by universal constants. For $\delta \in (0, 1/4]$, the total number of leaf-oracle queries τ satisfies*

$$\mathbb{E}_\mu[\tau] = \Omega\left(\log\frac{1}{\delta}\sum_{\ell\in\mathcal{L}_{\mathrm{piv}}}\frac{1}{d_{\ell,\epsilon}}\right).$$

Proof. For each leaf ℓ, let N_ℓ denote the number of queries to its oracle before the algorithm stops. For block $\mathcal{B}_j$, define $N_j := \sum_{\ell\in\mathcal{B}_j} N_\ell$.

Fix one block $\mathcal{B}_j$ and run the tree algorithm on the family $\mu^{(j)}(\theta)$ from Definition 2. Convert its recommendation $\widehat{s}$ into a binary decision by setting

$$D_j = 0 \quad\Longleftrightarrow\quad \widehat{s} \in \mathcal{M}_\epsilon(\mu).$$

At $\theta = \theta_{\mathrm{c}}$, (ϵ, δ)-correctness gives $\mathbb{P}_{\theta_{\mathrm{c}}}(D_j = 1) \le \delta$.

For every $\theta \notin \Theta_j$, the sets $\mathcal{M}_\epsilon(\mu)$ and $\mathcal{M}_\epsilon(\mu^{(j)}(\theta))$ are disjoint. Therefore, whenever the algorithm is correct on the perturbed instance, its recommendation cannot belong to $\mathcal{M}_\epsilon(\mu)$, and $\sup_{\theta\notin\Theta_j}\mathbb{P}_\theta(D_j = 0) \le \delta$. Thus, the tree algorithm induces the phase test of Lemma 5.

By Definition 2, the induced test can be simulated using no more than N_j phase-oracle queries. Therefore, Lemma 5 and $\mathrm{Leb}(\Theta_j) \le C d_{\mathcal{B}_j}$ imply

$$\mathbb{E}_\mu[N_j] \ge \frac{1}{16 C d_{\mathcal{B}_j}} \log\frac{1}{4\delta} - \frac{1}{2}, \tag{15}$$

whenever $\delta \le 1/4$. If $\delta > 1/4$, then $\log \frac{1}{4\delta} < 0$. The theorem then follows immediately.

Since $d_{\mathcal{B}_j} = \min_{\ell \in \mathcal{B}_j} d_{\ell,\epsilon}$, we have

$$\frac{1}{d_{\mathcal{B}_j}} = \max_{\ell \in \mathcal{B}_j} \frac{1}{d_{\ell,\epsilon}} \ge \frac{1}{|\mathcal{B}_j|} \sum_{\ell \in \mathcal{B}_j} \frac{1}{d_{\ell,\epsilon}} \ge \frac{1}{b} \sum_{\ell \in \mathcal{B}_j} \frac{1}{d_{\ell,\epsilon}}.$$

Hence, $\tau \ge \sum_{j=1}^{m} N_j$. Taking expectations, summing equation 15 over the blocks, and applying the preceding inequality gives

$$\mathbb{E}_\mu[\tau] \ge \frac{1}{16bC} \log \frac{1}{4\delta} \sum_{\ell \in \mathcal{L}_{\mathrm{piv}}} \frac{1}{d_{\ell,\epsilon}} - \frac{m}{2}.$$

□

Define

$$H_{\mathrm{Q}}(\mu, \epsilon) := \sum_{\ell \in \mathcal{L}} \frac{1}{d_{\ell,\epsilon}}.$$

We call an tree instance *uniformly pivotal* if it admits a disjoint pivotal-block family that covers $\mathcal{L}$ and whose perturbation constant C and maximum block size b are bounded by universal constants. We use $\widetilde{O}$ to suppress logarithmic factors.

Corollary 1. *On every uniformly pivotal instance, any (ϵ, δ)-correct quantum algorithm satisfies*

$$\mathbb{E}_\mu[\tau] = \Omega\left(H_{\mathrm{Q}}(\mu, \epsilon) \log \frac{1}{\delta}\right),$$

whereas QMCTS uses

$$\widetilde{O}\left(H_{\mathrm{Q}}(\mu, \epsilon) \log \frac{1}{\delta}\right)$$

queries with probability at least $1 - \delta$. Therefore, the expected lower bound and the high-probability upper bound have the same leading instance dependence up to logarithmic factors.

Proof. Uniform pivotality gives $\mathcal{L}_{\mathrm{piv}} = \mathcal{L}$ with bounded block constants, so Theorem 4 yields the stated lower bound. Theorem 2 yields the upper bound. □

Exact identification, best arm identification, and general trees. Although QMCTS is presented for $\epsilon \in (0, 1]$, the formulation and lower-bound analysis also cover exact identification by setting $\epsilon = 0$, provided $\Delta_\star > 0$. For a depth-one MAX tree, this reduces to fixed-confidence exact BAI, while $\epsilon \in (0, 1]$ reduces to fixed-confidence ϵ-BAI. Thus, BAI is a special case of the tree-search problem studied here.

C Classical MCTS Benchmark

Theorem 5. *With probability at least $1 - \delta$, CMCTS stops after finitely many iterations and returns an ϵ-optimal root move for s_0, i.e.,*

$$\mathbb{P}\left(V_{s_0} - V_{\widehat{s}^{\mathrm{C}}} \le \epsilon\right) \ge 1 - \delta.$$

Proof. For each leaf ℓ, let $X_{\ell,1}, X_{\ell,2}, \ldots$ be an i.i.d. sequence of Bernoulli observations with mean μ_ℓ. For $n \ge 1$, define

$$\overline{X}_{\ell,n} := \frac{1}{n} \sum_{k=1}^{n} X_{\ell,k}.$$

Define the classical good event

$$\mathcal{G}_{\mathrm{C}} := \bigcap_{r \ge 1} \bigcap_{\ell \in \mathcal{L}} \left\{ \left| \overline{X}_{\ell,n_r} - \mu_\ell \right| \le \alpha_r \right\}.$$

By Hoeffding's inequality and the definition of n_r,

$$\mathbb{P}\left(\left|\overline{X}_{\ell,n_r} - \mu_\ell\right| > \alpha_r\right) \le 2\exp(-2n_r\alpha_r^2) \le \eta_r.$$

Consequently,

$$\mathbb{P}(\mathcal{G}_{\mathrm{C}}^{\mathrm{c}}) \le \sum_{r\ge 1}\sum_{\ell\in\mathcal{L}} \eta_r = \frac{\delta}{2}\sum_{r\ge 1}\frac{1}{r^2} < \delta.$$

Condition on $\mathcal{G}_{\mathrm{C}}$. Every active classical leaf estimate has error at most α_r. The proofs of Lemmas 3 and 4 therefore apply with $\widehat{\mu}_{\ell,r}^{\mathrm{C}}$ and $\widehat{V}_{s,r}^{\mathrm{C}}$ in place of their quantum counterparts.

Because $\gamma_r = 2^{-r}$ and $\epsilon > 0$, CMCTS stops no later than the first iteration r for which $\gamma_r \le \epsilon$. If it stops with only one root move remaining, that move is optimal. Otherwise, let $s_r^\star$ be an active optimal root move. Since CMCTS returns the active root move with the largest propagated estimate,

$$V_{s_0} - V_{\widehat{s}^{\mathrm{C}}} \le \left|V_{s_r^\star} - \widehat{V}_{s_r^\star,r}^{\mathrm{C}}\right| + \left|\widehat{V}_{\widehat{s}^{\mathrm{C}},r}^{\mathrm{C}} - V_{\widehat{s}^{\mathrm{C}}}\right| \le 2\alpha_r = \gamma_r \le \epsilon.$$

Since $\mathbb{P}(\mathcal{G}_{\mathrm{C}}) \ge 1-\delta$, the result follows. □

Theorem 6. *With probability at least $1-\delta$, the total number of leaf-oracle queries used by CMCTS is*

$$O\left(\sum_{\ell\in\mathcal{L}} \frac{1}{d_{\ell,\epsilon}^2}\left[\log\frac{4L}{\delta} + 2\log\left(2+\log_2\frac{1}{d_{\ell,\epsilon}}\right)\right]\right), \tag{16}$$

where $d_{\ell,\epsilon} = \Delta_\ell \vee \Delta_\star \vee \epsilon$.

Proof. Condition on the classical good event $\mathcal{G}_{\mathrm{C}}$, and let R_ℓ be the final iteration in which leaf ℓ is queried. The arguments in the proof of Theorem 2 give

$$\frac{1}{\gamma_{R_\ell}} \le \frac{4}{d_{\ell,\epsilon}}, \qquad R_\ell \le 2+\log_2\frac{1}{d_{\ell,\epsilon}}. \tag{17}$$

Using $\alpha_r = \frac{\gamma_r}{2}, \eta_r = \frac{\delta}{2Lr^2}$, we obtain

$$n_{R_\ell} \le 1 + \frac{1}{2\alpha_{R_\ell}^2}\log\frac{2}{\eta_{R_\ell}} = 1 + \frac{2}{\gamma_{R_\ell}^2}\left[\log\frac{4L}{\delta} + 2\log R_\ell\right].$$

Applying equation 17 yields

$$n_{R_\ell} \le 1 + \frac{32}{d_{\ell,\epsilon}^2}\left[\log\frac{4L}{\delta} + 2\log\left(2+\log_2\frac{1}{d_{\ell,\epsilon}}\right)\right].$$

Summing over all leaves proves (16). Finally, $\mathbb{P}(\mathcal{G}_{\mathrm{C}}) \ge 1-\delta$, so the bound holds with probability at least $1-\delta$. □

D Quantum Algorithm Design

D.1 Hybrid Algorithm

In this subsection, we provide implementation details for the Hybrid MCTS algorithm.

D.2 Query Complexity under Different Leaf Distributions

In this section, we provide additional numerical results to show that our algorithms also apply to more general leaf distributions.

We consider two additional distributional settings in the synthetic experiment. The first is a bounded three-point distribution supported on 0, $1/2$ and 1, with probabilities chosen to match the mean in the Bernoulli setting. The second is a bounded five-point discrete distribution with equally spaced support points chosen to preserve the same mean. All other experimental settings remain unchanged. As shown in Figures 3 and 4, the results are consistent with those obtained under Bernoulli distributions, which suggests that our methods are robust to the choice of output distribution.

Algorithm 3 Hybrid Monte Carlo Tree Search (Hybrid MCTS)

Require: Finite MAX–MIN tree $\mathcal{T}$, accuracy $\epsilon \in (0, 1]$, and confidence $\delta \in (0, 1/2)$

$\mathcal{T}_0 \leftarrow \mathcal{T}$ and $L \leftarrow |\mathcal{L}|$
$N_\ell \leftarrow 0$ for every $\ell \in \mathcal{L}$
$n_0 \leftarrow 0$ and $r \leftarrow 1$

▷ Classical phase

while true do
  $\gamma_r \leftarrow 2^{-r}$, $\alpha_r \leftarrow \gamma_r/2$, and $\eta_r \leftarrow \delta/(2Lr^2)$
  $n_r \leftarrow \lceil \log(2/\eta_r)/(2\alpha_r^2) \rceil$
  if a fresh QMC estimate with parameters α_r and η_r uses fewer than $n_r - n_{r-1}$ queries **then**
    break
  end if
  for all $\ell \in \mathcal{L}_{r-1}$ **do**
    $\Delta_\ell \leftarrow \max\{0, n_r - N_\ell\}$
    Query leaf ℓ for Δ_ℓ additional observations and retain them
    $N_\ell \leftarrow N_\ell + \Delta_\ell$
    Construct $\widehat{\mu}_{\ell,r}$ as the empirical mean of all N_ℓ retained observations
  end for
  Propagate $\widehat{V}_{s,r}$ upward using (4)
  for all internal nodes s, in reverse topological order **do**
    Remove each child c satisfying $\widehat{g}_r(s, c) > \gamma_r$
  end for
  Let $\mathcal{T}_r$ be the remaining connected subtree, with active leaf set $\mathcal{L}_r$
  $\widehat{s}_r \in \arg\max_{s \in \mathcal{C}_r(s_0)} \widehat{V}_{s,r}$
  if $|\mathcal{C}_r(s_0)| = 1$ **or** $\gamma_r \leq \epsilon$ **then**
    return $\widehat{s}_r$
  end if
  $r \leftarrow r + 1$
end while

▷ Quantum phase

while true do
  $\gamma_r \leftarrow 2^{-r}$, $\alpha_r \leftarrow \gamma_r/2$, and $\eta_r \leftarrow \delta/(2Lr^2)$
  for all $\ell \in \mathcal{L}_{r-1}$ **do**
    $\widehat{\mu}_{\ell,r} \leftarrow \mathrm{QMC}(U_\ell, \alpha_r, \eta_r)$
  end for
  Propagate $\widehat{V}_{s,r}$ upward (4)
  for all internal nodes s, in reverse topological order **do**
    Remove each child c satisfying $\widehat{g}_r(s, c) > \gamma_r$
  end for
  Let $\mathcal{T}_r$ be the remaining connected subtree, with active leaf set $\mathcal{L}_r$
  $\widehat{s}_r \in \arg\max_{s \in \mathcal{C}_r(s_0)} \widehat{V}_{s,r}$
  if $|\mathcal{C}_r(s_0)| = 1$ **or** $\gamma_r \leq \epsilon$ **then**
    return $\widehat{s}_r$
  end if
  $r \leftarrow r + 1$
end while

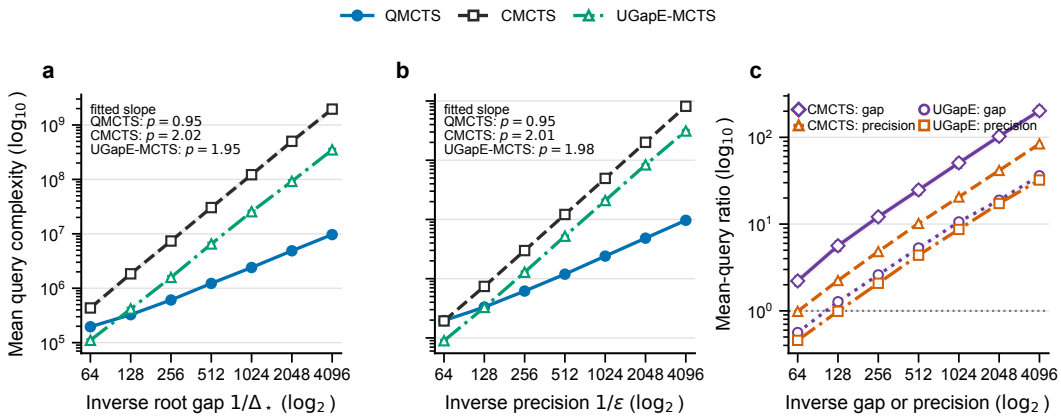


Figure 3: Query Complexity Scaling under A Bounded Three-point Distribution

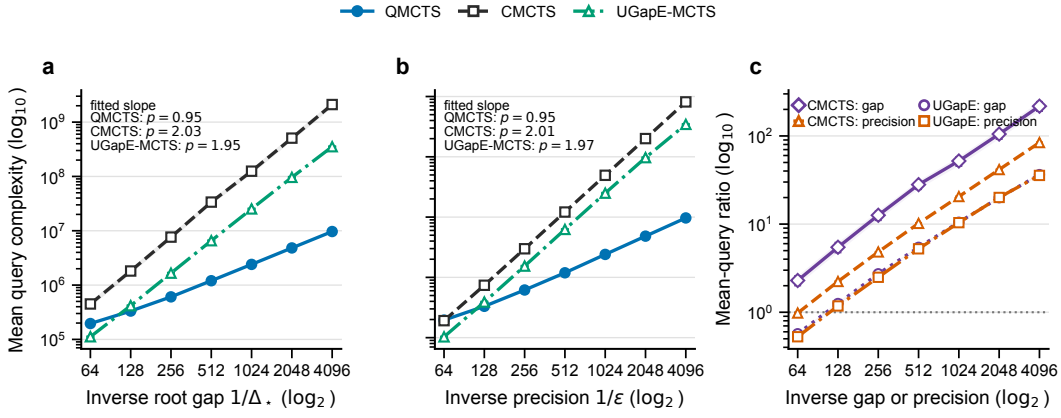


Figure 4: Query Complexity Scaling under A Bounded Five-point Discrete Uniform Distribution