Dam, Tuan; D'Eramo, Carlo; Peters, Jan; Pajarinen, Joni

A Unified Perspective on Value Backup and Exploration in Monte-Carlo Tree Search

Published in:
Journal of Artificial Intelligence Research

DOI:
10.1613/jair.1.16019

Published: 01/01/2024

Document Version
Publisher's PDF, also known as Version of record

Journal of Artificial Intelligence Research 81 (2024) 511-577 Submitted 02/2024; published 11/2024

A Unified Perspective on Value Backup and Exploration in Monte-Carlo Tree Search

Tuan Dam TUANDQ@SOICT.HUST.EDU.VN
Hanoi University of Science and Technology, Vietnam

Carlo D'Eramo CARLO.DERAMO@UNI-WUERZBURG.DE
Center for Artificial Intelligence and Data Science, University of Würzburg, Germany
Hessian.ai, Germany
Department of Computer Science, Technical University of Darmstadt, Germany

Jan Peters MAIL@JAN-PETERS.NET
Hessian.ai, Germany
Department of Computer Science, Technical University of Darmstadt, Germany

Joni Pajarinen JONI.PAJARINEN@AALTO.FI
Department of Electrical Engineering and Automation, Aalto University, Finland

Abstract

Monte-Carlo Tree Search (MCTS) is a class of methods for solving complex decision-making problems through the synergy of Monte-Carlo planning and Reinforcement Learning (RL). The highly combinatorial nature of the problems commonly addressed by MCTS requires the use of efficient exploration strategies for navigating the planning tree and quickly convergent value backup methods. These crucial problems are particularly evident in recent advances that combine MCTS with deep neural networks for function approximation. In this work, we propose two methods for improving the convergence rate and exploration based on a newly introduced backup operator and entropy regularization. We provide strong theoretical guarantees to bound convergence rate, approximation error, and regret of our methods. Moreover, we introduce a mathematical framework based on the use of the α-divergence for backup and exploration in MCTS. We show that this theoretical formulation unifies different approaches, including our newly introduced ones, under the same mathematical framework, allowing to obtain different methods by simply changing the value of α. In practice, our unified perspective offers a flexible way to balance between exploration and exploitation by tuning the single α parameter according to the problem at hand. We validate our methods through a rigorous empirical study from basic toy problems to the complex Atari games, and including both MDP and POMDP problems.

1. Introduction

Monte-Carlo Tree Search (MCTS) is an effective method that combines a random sampling strategy with tree search to determine the optimal decision for on-the-fly planning tasks. MCTS has yielded impressive results in Go (Silver et al., 2016) (AlphaGo), Chess (Silver et al., 2017a) (AlphaZero), or video games (Osband et al., 2016), and it has been further exploited successfully in motion planning (Nguyen et al., 2017; Sukkar et al., 2019), autonomous car driving (Volpi et al., 2017; Chen et al., 2020), and autonomous robotic assembly tasks (Funk et al., 2021). Many of the MCTS successes (Silver et al., 2016, 2017a,b) rely on coupling MCTS with neural networks trained using Reinforcement Learning (RL) (Sutton and Barto, 1998) methods such as Deep Q-Learning (Mnih et al., 2015), to speed up learning of large scale problems. Despite AlphaGo and AlphaZero achieving state-of-the-art performance in games with high branching factor-like Go (Silver et al., 2016) and Chess (Silver et al., 2017a), both methods suffer from poor sample efficiency, mostly due to the inefficiency of the average reward backup operator (Coulom, 2006), which is well-known for the issue of underestimating the optimum, and due to the polynomial convergence rate of UCT (Kocsis et al., 2006) or PUCT (Xiao et al., 2019). These issues pose the open research problem of finding effective backup operators and efficient exploration strategies in the tree search.

In this paper, we answer this open research problem by proposing two principal approaches. First, we introduce a novel backup operator based on a power mean (Bullen, 2013) that, through the tuning of a single coefficient, computes a value between the average reward and the maximum one. This allows for balancing between the negatively biased estimate of the average reward, and the positively biased estimate of the maximum reward; in practice, this translates to balancing between a safe but slow update, and a greedy but misleading one. We propose a variant of UCT based on the power mean operator, which we call Power-UCT. We theoretically prove the convergence of Power-UCT, based on the consideration that the algorithm converges for all values between the range computed by the power mean. We empirically evaluate Power-UCT w.r.t. UCT, POMCP Silver and Veness (2010) which is a well known POMDP variant of UCT, and the MENTS algorithm (Xiao et al., 2019) in classic MDP and POMDP benchmarks. Remarkably, we show how Power-UCT outperforms the baselines in terms of quality and speed of learning.

Second, we provide a theory of the use of convex regularization in MCTS, which has proven to be an efficient solution for driving exploration and stabilizing learning in RL (Schulman et al., 2015, 2017a; Haarnoja et al., 2018; Buesing et al., 2020). In particular, we show how a regularized objective function in MCTS can be seen as an instance of the Legendre-Fenchel transform, similar to previous findings on the use of duality in RL (Mensch and Blondel, 2018; Geist et al., 2019; Nachum and Dai, 2020a) and game theory (Shalev-Shwartz and Singer, 2006; Pavel, 2007). Establishing our theoretical framework, we derive the first regret analysis of regularized MCTS, and prove that a generic convex regularizer guarantees an exponential convergence rate to the solution of the regularized objective function, which improves on the polynomial rate of PUCT. These results provide a theoretical ground for the use of arbitrary entropy-based regularizers in MCTS until now limited to maximum entropy (Xiao et al., 2019), among which we specifically study the relative entropy of policy updates, drawing on similarities with trust-region and proximal methods in RL (Schul-

man et al., 2015, 2017b), and the Tsallis entropy, used for enforcing the learning of sparse policies (Lee et al., 2018). Moreover, we provide an empirical analysis of a toy problem introduced in Xiao et al. (2019) to evince the practical consequences of our theoretical results for each regularizer. We empirically evaluate the proposed operators in AlphaGo on several Atari games, confirming the benefit of convex regularization in MCTS, and in particular, the superiority of Tsallis entropy w.r.t. other regularizers.

Finally, we provide a theory of the use of α-divergence in MCTS for backup and exploration. Remarkably, we show that our theoretical framework unifies our two proposed methods Power-UCT (Dam et al., 2019) and entropy regularization (Dam et al., 2021), that can be obtained for particular choices of the value of α. In the general case where α is considered a real number greater than 0, we show that tuning α directly influences the navigation and backup phases of the tree search, providing a unique powerful mathematical formulation to effectively balance between exploration and exploitation in MCTS.

2. Related Work

We want to improve the efficiency and performance of MCTS by addressing the two crucial problems of value backup and exploration. Our contribution follows on from a plethora of previous works that we briefly summarize in the following.

Backup operators. To improve upon the UCT algorithm in MCTS, Khandelwal et al. (2016) formalize and analyze different on-policy and off-policy complex backup approaches for MCTS planning based on techniques in the RL literature. Khandelwal et al. (2016) propose four complex backup strategies: MCTS(λ), MaxMCTS(λ), MCTS$_\gamma$, MaxMCTS$_\gamma$, and report that MaxMCTS(λ) and MaxMCTS$_\gamma$ perform better than UCT for certain parameter setups. Vodopivec et al. (2017) propose an approach called SARSA-UCT, which performs the dynamic programming backups using SARSA (Rummery, 1995). Both Khandelwal et al. (2016) and Vodopivec et al. (2017) directly borrow value backup ideas from RL in order to estimate the value at each tree node. However, they do not provide any proof of convergence. The recently introduced MENTS algorithm (Xiao et al., 2019), uses softmax backup operator at each node in combination with an entropy-based exploration policy, and shows a better convergence rate w.r.t. UCT.

Exploration. Entropy regularization is a common tool for controlling exploration in RL and has led to several successful methods (Schulman et al., 2015; Haarnoja et al., 2018; Schulman et al., 2017a; Mnih et al., 2016). Typically specific forms of entropy are utilized such as maximum entropy (Haarnoja et al., 2018) or relative entropy (Schulman et al., 2015). This approach is an instance of the more generic duality framework, commonly used in convex optimization theory. Duality has been extensively studied in game theory (Shalev-Shwartz and Singer, 2006; Pavel, 2007) and more recently in RL, for instance considering mirror descent optimization (Montgomery and Levine, 2016; Mei et al., 2019), drawing the connection between MCTS and regularized policy optimization (Grill et al., 2020), or formalizing the RL objective via Legendre-Rockafellar duality (Nachum and Dai, 2020a). Recently (Geist et al., 2019) introduced regularized Markov Decision Processes, formalizing the RL objective with a generalized form of convex regularization, based on the Legendre-Fenchel transform. Several works focus on modifying classical MCTS to improve

exploration. For instance, Tesauro et al. (2012) propose a Bayesian version of UCT to improve estimation of node values and uncertainties given limited experience.

α-divergence. The use of α-divergence in RL has been widely explored, particularly by Belousov and Peters (2019), who proposed using it to measure the divergence in policy search. Their work generalizes relative entropy policy search to constrain policy updates. Belousov and Peters (2019) studied a particular class of f-divergence, known as α-divergence, which resulted in compatible policy update and value function improvement in actor-critic methods. Another study by Lee et al. (2019) analyzed α-divergence as a generalized Tsallis Entropy regularizer in MDP. By scaling the α parameter as an entropic index, Lee et al. (2019) controlled the generalized Tsallis Entropy regularizer and derived Shannon-Gibbs entropy and Tsallis Entropy as special cases.

3. Preliminaries

3.1 Markov Decision Process

In the context of Reinforcement Learning (RL), an agent's goal is to determine how to interact with the environment modeled as a Markov Decision Process (MDP), which is a well-known mathematical framework for sequential decision-making. Our focus is on an infinite-horizon discount MDP that can be represented as a 5-tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the finite discrete action space with $|\mathcal{A}|$ representing the number of actions, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the probability distribution over the next state s' given the current state s and action a, and $\gamma \in [0, 1)$ is the discount factor. A policy $\pi \in \Pi : \mathcal{S} \to \mathcal{A}$ is a probability distribution over possible actions a given the current state s.

A policy π induces a $\mathcal{Q}$ value function: $Q^\pi(s, a) \triangleq \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r_{i+k+1} | s_i = s, a_i = a, \pi\right]$, where r_{i+1} is the reward obtained after the i-th transition, respectively defining the value function under the policy π as $V^\pi(s) \triangleq \max_{a \in \mathcal{A}} Q^\pi(s, a)$.

The Bellman operator under the policy π is defined as

$$\mathcal{T}_\pi Q(s, a) \triangleq \sum_{s'} \mathcal{P}(s'|s, a) \left[\mathcal{R}(s, a, s') + \gamma \sum_{a'} \pi(a'|s') Q(s', a') \right], \tag{1}$$

The goal is to find the optimal policy that satisfies the optimal Bellman equation (Bellman, 1954)

$$Q^*(s, a) \triangleq \sum_{s'} \mathcal{P}(s'|s, a) \left[\mathcal{R}(s, a, s') + \gamma \max_{a'} Q^*(s', a') \right], \tag{2}$$

which is the fixed point of the optimal Bellman operator

$$\mathcal{T}^* Q(s, a) \triangleq \sum_{s'} \mathcal{P}(s'|s, a) \left[\mathcal{R}(s, a, s') + \gamma \max_{a'} Q(s', a') \right]. \tag{3}$$

The optimal value function is defined $V^*(s) \triangleq \max_{a \in \mathcal{A}} Q^*(s, a)$.

3.2 Monte-Carlo Tree Search

Monte-Carlo Tree Search (MCTS) is a tree search method for MDPs that combines Monte-Carlo sampling, tree search, and multi-armed bandits to make optimal decisions efficiently. The MCTS tree is composed of nodes and edges that represent visited states and actions taken in each state, respectively. As shown in Figure 1, the algorithm consists of four main steps:

1. **Selection:** where a *tree-policy* is used to traverse the tree from the root node to a leaf node.
2. **Expansion:** where the new node is added to the tree according to the tree policy;
3. **Simulation:** where a Monte-Carlo rollout or a neural network is used to estimate the value of the new node.
4. **Backup:** where the collected reward is used to update the action-values along the path from the leaf node to the root node.

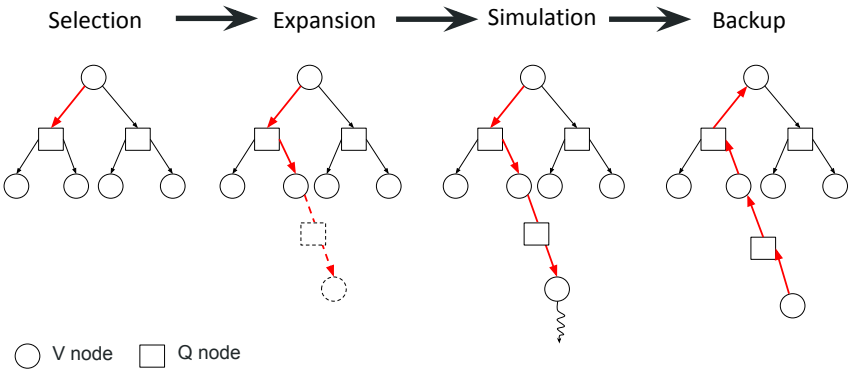


Figure 1: Four basic steps of MCTS

Pseudocode of a general MCTS approach is shown in Algorithm 1. The tree-policy used to select the action to execute in each node needs to balance the use of already known good actions, and the visitation of unknown states.

3.3 Upper Confidence bound for Trees

In this section, we present the MCTS algorithm UCT (Upper Confidence bounds for Trees) Kocsis et al. (2006), an extension of the well-known UCB1 Auer et al. (2002) multi-armed bandit algorithm. UCB1 chooses the arm (action a) using

$$a = \underset{i \in \{1 \ldots K\}}{\arg\max}\, \overline{X}_{i,T_i(n-1)} + C\sqrt{\frac{\log n}{T_i(n-1)}}. \tag{4}$$

where $T_i(n) = \sum_{t=1}^{n} \mathbf{1}\{t = i\}$ is the number of times arm i is played up to time n. $\overline{X}_{i,T_i(n-1)}$ denotes the average reward of arm i up to time $n-1$ and $C = \sqrt{2}$ is an exploration constant.

Algorithm 1: Pseudocode of General MCTS.

s, s': states
a: action
r: reward
$\mathcal{P}(\cdot|s,a)$: transition function
$\mathcal{R}(s,a,s')$: reward function

R = `Rollout`(s, *depth*)
  Use a random policy/behavior policy to collect accumulated rewards from environment.

a = `SelectAction`(s)
  return action selection from tree policy.

`Expand`(s)
  Expand the tree by adding a new node at state s.

`Backup`(s, r)
  Update the collected reward along the visited path

R= `Simulation`(s, *depth*)
  return `Rollout`(s, *depth*)

a = `Search`(s)
  $s_0 = s$
  depth $= 0$
  while *Time remaining* **do**
    a = `SelectAction`(s) // Selection
    $s' = \mathcal{P}(\cdot|s,a)$
    $r = \mathcal{R}(s,a,s')$
    `Expand`(*s'*) //Expansion
    `Simulation` (s', *depth*) //Simulation
    $s = s'$
    `Backup`(s, r) // Backpropagation
    depth $=$ *depth* $+ 1$
  return the best action at state s_0

`MainLoop`
  a = `Search`(s)

In UCT, each node is a separate bandit, where the arms correspond to the actions, and the payoff is the reward of the episodes starting from them. In the backup phase, value is backed up recursively from the leaf node to the root as

$$\overline{X}_n = \sum_{i=1}^{K} \Big(\frac{T_i(n)}{n}\Big) \overline{X}_{i,T_i(n)}. \tag{5}$$

Pseudocode of UCT is shown in Algorithm 2. Kocsis et al. (2006) proved that UCT converges in the limit to the optimal policy.

3.4 α-divergence

The f-divergence(Csiszár, 1964) generalizes the definition of the distance between two probabilistic distributions P and Q on a finite set $\mathcal{A}$ as

$$D_f\left(P\|Q\right) = \sum_{a\in\mathcal{A}} Q(a) f\left(\frac{P(a)}{Q(a)}\right), \tag{6}$$

where f is a convex function on $(0,\infty)$ such as $f(1) = 0$. For example, the KL-divergence corresponds to $f_{KL} = x\log x - (x-1)$. The $\alpha-$divergence is a subclass of f-divergence generated by $\alpha-$function with $\alpha \in \mathbb{R}$. $\alpha-$function is defined as

$$f_\alpha(x) = \frac{(x^\alpha - 1) - \alpha(x-1)}{\alpha(\alpha-1)}. \tag{7}$$

The $\alpha-$divergence between two probabilistic distributions P and Q on a finite set $\mathcal{A}$ is defined as

$$D_\alpha\left(P\|Q\right) = \sum_{a\in\mathcal{A}} Q(a) f_\alpha\left(\frac{P(a)}{Q(a)}\right), \tag{8}$$

Algorithm 2: Pseudocode of UCT.

s, s': states
a: action
$N(s)$: number of simulations of V_Node of state s
$n(s,a)$: number of simulations of Q_Node of state s and action a
$S_{s,a}$: is the set of next states after taking action a from state s
$r^{(i)}_{s,a}$: the i-th instantaneous reward collected after executing action a in state s
$V(s)$: Value of V_Node at state s. Default is 0
$Q(s,a)$: Value of Q_Node at state s, action a. Default is 0
$\mathcal{P}(\cdot|s,a)$: transition function
$\mathcal{R}(s,a,s')$: reward function
γ: discount factor
$\epsilon > 0$:

R = `Rollout`$(s, depth)$
  if $\gamma^{depth} < \epsilon$ **then**
    return 0
  $a \sim \pi_{\text{Rollout}}(.)$
  $s' \sim \mathcal{P}(\cdot|s,a)$
  $r \sim \mathcal{R}(s,a,s')$
  return $r + \gamma$Rollout $(s', depth+1)$

a = `SelectAction`(s)
  return $\arg\max_a Q(s,a) + C\sqrt{\frac{\log N(s)}{n(s,a)}}$

a = `Search`(s)
  while *Time remaining* **do**
    `SimulateV` $(s, 0)$
  return `SelectAction`(s)

`SimulateV`$(s, depth)$
  $a =$`SelectAction` (s)
  `SimulateQ` (s, a, depth)
  $N(s) \leftarrow N(s) + 1$
  $V(s) \leftarrow \sum_a \frac{n(s,a)}{N(s)} Q(s,a)$

`SimulateQ`$(s, a, depth)$
  $s' \sim \mathcal{P}(\cdot|s,a)$
  $r \sim \mathcal{R}(s,a,s')$
  if *Node* s' *not expanded* **then**
    `Rollout`(s', depth)
  else
    `SimulateV` $(s', \text{depth}+1)$
  $n(s,a) \leftarrow n(s,a) + 1$
  $Q(s,a) \leftarrow \frac{\sum_{i=1}^{n(s,a)} r^{(i)}_{s,a} + \gamma \sum_{s' \in S_{s,a}} N(s')V(s')}{n(s,a)}$

`MainLoop`
  $a =$ `Search`(s)

where $\sum_{a\in\mathcal{A}} Q(a) = \sum_{a\in\mathcal{A}} P(a) = 1$.
Furthermore, given the $\alpha-$function, we can derive the generalization of Tsallis entropy of a policy π as

$$H_\alpha(s) = \frac{1}{\alpha(1-\alpha)}\left(1 - \sum_{a\in\mathcal{A}} \pi(s,a)^\alpha\right) \tag{9}$$

In addition, we have

$$\lim_{\alpha\to 1} H_\alpha(s) = -\sum_{a\in\mathcal{A}} \pi(s,a)\log\pi(s,a) \tag{10}$$

$$H_2(s) = \frac{1}{2}\left(1 - \sum_{a\in\mathcal{A}} \pi(s,a)^2\right), \tag{11}$$

respectively, the Shannon entropy (10) and the Tsallis entropy (11) functions.

4. Generalized Mean Estimation

In this section, we present a novel approach to estimating the expected value of a bandit arm in MCTS using the *power mean* (Mitrinovic and Vasic, 1970). The power mean is a

function for aggregating sets of numbers, which includes the Pythagorean means (arithmetic, geometric, and harmonic means) as special cases. For a sequence of positive numbers $X = (X_1, ..., X_n)$ and positive weights $w = (w_1, ..., w_n)$, the power mean with exponent p (where p is an extended real number) is defined as

$$\mathrm{M}^{[p]}n(X, w) = \left(\frac{\sum_{i=1}^{n} w_i X_i^p}{\sum_{i=1}^{n} w_i} \right)^{\frac{1}{p}}. \tag{12}$$

When $p = 1$, it becomes the weighted arithmetic mean. In the limit as p approaches 0, it becomes the geometric mean, and when $p = -1$, it becomes the harmonic mean (Mitrinovic and Vasic, 1970). We will use the power mean to estimate the expected value of bandit arms in our proposed approach for MCTS.

Furthermore, we get (Mitrinovic and Vasic, 1970)

$$\mathrm{M}_n^{[-\infty]}(X, w) = \lim_{p \to -\infty} \mathrm{M}_n^{[p]}(X, w) = \mathrm{Min}(X_1, ..., X_n), \tag{13}$$

$$\mathrm{M}_n^{[+\infty]}(X, w) = \lim_{p \to +\infty} \mathrm{M}_n^{[p]}(X, w) = \mathrm{Max}(X_1, ..., X_n), \tag{14}$$

The weighted arithmetic mean lies between $\mathrm{Min}(X_1, ..., X_n)$ and $\mathrm{Max}(X_1, ..., X_n)$. Moreover, the following lemma shows that $\mathrm{M}_n^{[p]}(X, w)$ is an increasing function.

Lemma 1. *$M_n^{[p]}(X, w)$ is an increasing function meaning that*

$$M_n^{[1]}(X, w) \leq M_n^{[q]}(X, w) \leq M_n^{[p]}(X, w), \forall p \geq q \geq 1 \tag{15}$$

Proof. For the proof, see Mitrinovic and Vasic (1970). □

The following lemma shows that Power Mean can be upper bound by Average Mean plus with a constant.

Lemma 2. *Let $0 < l \leq X_i \leq U, C = \frac{U}{l}, \forall i \in (1, ..., n)$ and $p > q$. We define:*

$$Q(X, w, p, q) = \frac{M_n^{[p]}(X, w)}{M_n^{[q]}(X, w)} \tag{16}$$

$$D(X, w, p, q) = M_n^{[p]}(X, w) - M_n^{[q]}(X, w). \tag{17}$$

Then we have:

$$Q(X, w, p, q) \leq L_{p,q} D(X, w, p, q) \leq H_{p,q}$$

$$L_{p,q} = \left(\frac{q(C^p - C^q)}{(p - q)(C^q - 1)} \right)^{\frac{1}{p}} \left(\frac{p(C^q - C^p)}{(q - p)(C^p - 1)} \right)^{-\frac{1}{q}}$$

$$H_{p,q} = (\theta U^p + (1 - \theta) l^p)^{\frac{1}{p}} - (\theta U^q + (1 - \theta) l^q)^{1/q},$$

where θ is defined in the following way. Let

$$h(x) = x^{\frac{1}{p}} - (ax + b)^{1/q}$$

where:

$$a = \frac{U^q - l^q}{U^p - l^p}; b = \frac{U^p l^q - U^q l^p}{U^p - l^p} \tag{18}$$

$$x^{'} = \arg\max\{h(x), x \in (l^p, U^p)\} \tag{19}$$

then:

$$\theta = \frac{x' - l^p}{U^p - l^p}.$$

Proof. Refer to Mitrinovic and Vasic (1970). □

From power mean, we derive power mean backup operator and propose our novel method, Power-UCT

4.1 Power Mean Backup

It has been studied that when performing backups with the average mean backup operator, the resulting value estimate of the node typically underestimates the actual value, while using the maximum backup operator leads to overestimation (Coulom, 2006). Typically, the average backup operator is used in situations where the number of simulations is low to provide a conservative update of the nodes due to the lack of samples, whereas the maximum operator is favored when the number of simulations is high. We propose a new backup operator for UCT based on the power mean, as shown in Equation 12, to address this issue:

$$\overline{X}_n(p) = \left(\sum_{i=1}^{K} \left(\frac{T_i(n)}{n} \right) \overline{X}^p_{i,T_i(n)} \right)^{\frac{1}{p}}. \tag{20}$$

We aim to bridge the gap between the average and maximum estimators by proposing a novel approach called Power-UCT which offers the advantages of both. In the following, we provide a more detailed description of our approach.

4.2 Power-UCT

MCTS has two types of nodes, V-nodes for state-values and Q-nodes for state-action values. When an action is taken from the V-node of the current state, it leads to the corresponding Q-node, which then leads to the V-node of the reached state. Our proposed backup operator in UCT, called Power-UCT, does not require major changes to the algorithm. In Power-UCT, the expansion of nodes and rollouts are done in the same way as UCT, and the only difference is the method of computing the backup of returns from Q-nodes to V-nodes. Unlike UCT, which computes the average of returns, Power-UCT uses a power mean of them. Note that our algorithm can be applied to several bandit-based enhancements of UCT, but for simplicity, we focus only on UCT. The backup value for each state s corresponds to its V-node is

$$V(s) \leftarrow \left(\sum_a \frac{n(s,a)}{N(s)} Q(s,a)^p \right)^{\frac{1}{p}} \tag{21}$$

SimulateV *(s, depth)*	**SimulateV** *(s, depth)*
$a =$`SelectAction` (s)	$a =$`SelectAction` (s)
`SimulateQ` (s, a, depth)	`SimulateQ` (s, a, depth)
$N(s) \leftarrow N(s) + 1$	$N(s) \leftarrow N(s) + 1$
$V(s) \leftarrow \sum_a \frac{n(s,a)}{N(s)} Q(s, a)$	$V(s) \leftarrow \left(\sum_a \frac{n(s,a)}{N(s)} Q(s, a)^p\right)^{1/p}$

Figure 2: Comparing UCT (left) and Power-UCT (right) in the SimulateV procedure. The only distinction is in the V value function backup; the two methods are identical in other procedures.

where $N(s)$ is the number of visits to state s, $n(s, a)$ is the number of visits of action a in state s. On the other hand, the backup value of Q_nodes is

$$Q(s, a) \leftarrow \frac{(\sum_{i=1}^{n(s,a)} r_{s,a}^{(i)}) + \gamma \sum_{s' \in S_{s,a}} N(s')V(s')}{n(s, a)} \tag{22}$$

where γ is the discount factor, $S_{s,a}$ is the set of next states after taking action a from state s, and $r_{s,a}^{(i)}$ is the i-th instantaneous reward collected after executing action a in state s. Power-UCT and UCT follow the same steps, differing only in the V value backup. Power-UCT employs a power mean backup operator, while UCT uses an average mean backup operator. See Fig. 2 for details.

4.3 Theoretical Analysis

This section presents how Power-UCT can adapt to all UCT Kocsis et al. (2006) theorems using a generalized mean as the backup operator. The main contribution of this work is Theorem 6 and Theorem 7, which prove the convergence of the failure probability and derive the bias of power mean estimated payoff. To prove them, we first introduce Theorem 1, which establishes the concentration of power mean with respect to i.i.d random variables X. Then, Theorem upper bounds the expected number of times a suboptimal arm is played, while Theorem 3 bounds the expected error of power mean estimation. Furthermore, Theorem 4 provides a lower bound on the number of times any arm is played, and Theorem 5 demonstrates the concentration of power mean backup around its mean value at each node in the tree.

Theorem 1. *If* $X_1, X_2, ..., X_n$ *are independent with* $\Pr(0 \leq X_i \leq 1) = 1$ *then for any* $\epsilon > 0$, $p \geq 1, \exists C_p > 0$ *that*

$$\Pr\left(\left|\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}} - \mathbb{E}\left[\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}}\right]\right| > \epsilon\right) \leq 2\exp\left(-C_p n\epsilon^2\right)$$

The derivation of Theorem 1 involves the estimation of the variance of the power mean operator and the application of Chernoff's inequality. It is worth noting that this result

can be viewed as an extension of the well-known Hoeffding inequality to the power mean. Considering a sequence of i.i.d. random variables X_{it} as the payoff received at any internal leaf node of the tree, we assume the existence of the expected payoff and let $\mu_{in} = \mathbb{E}[\overline{X_{in}}]$. Additionally, we assume that the power mean reward varies over time and only converges in the limit. This assumption implies that

$$\mu_i = \lim_{n\to\infty} \mu_{in}.$$

Let $\delta_{in} = \mu_i - \mu_{in}$ which also means that

$$\lim_{n\to\infty} \delta_{in} = 0.$$

From now on, let $*$ be the upper index for all quantities related to the optimal arm. By assumption, the rewards lie between 0 and 1.

$\overline{X}_n(p) = \left(\sum_{i=1}^{K} \left(\frac{T_i(n)}{n}\right) \overline{X}^p_{i,T_i(n)}\right)^{1/p}$ is defined as the power mean value backup at the root node. Here $T_i(n)$ is the number of visitations of the arm i, $\overline{X}_{i,T_i(n)}$ is the average rewards collected at arm i, $\overline{X}_{i,n}(p)$ is the estimated power mean backup of all the collected rewards at arm i at the time step n. We make an assumption regarding the concentration of the power mean backup operator.

Assumption 1. *Fix $1 \leq i \leq K$. Let $\{F_{it}\}_t$ be a filtration such that$\{X_{it}\}_t$ is $\{F_{it}\}$-adapted and $X_{i,t}$ is conditionally independent of $F_{i,t+1}, F_{i,t+2}, \ldots$ given $F_{i,t-1}$. Then $0 \leq X_{it} \leq 1$ and the limit of $\mu_{in} = \mathbb{E}[\overline{X_{in}}(p)]$ exists, Further, we assume that there exists a constant $C > 0$ and an integer N_c such that for $n > N_c$, for any $\delta > 0$, $\triangle_n(\delta) = C\sqrt{n \log(1/\delta)}$, the following bounds hold*

$$\Pr(\overline{X}_{in}(p) \geq \mathbb{E}[\overline{X}_{in}(p)] + \triangle_n(\delta)/n) \leq \delta, \tag{23}$$

$$\Pr(\overline{X}_{in}(p) \leq \mathbb{E}[\overline{X}_{in}(p)] - \triangle_n(\delta)/n) \leq \delta. \tag{24}$$

Under Assumption 1, a suitable choice for the bias sequence $c_{t,s}$ is given by

$$c_{t,s} = 2C\sqrt{\frac{\log t}{s}}. \tag{25}$$

where C is an exploration constant.

Next, we derive Theorems 2, 3, and 4 following the derivations in Kocsis et al. (2006). First, from Assumption 1, we derive an upper bound on the error for the expected number of times suboptimal arms are played.

Theorem 2. *Consider UCB1 (using power mean estimator) applied to a non-stationary problem where the pay-off sequence satisfies Assumption 1 and where the bias sequence, $c_{t,s}$ defined in (25). Fix $\epsilon \geq 0$. Let $T_k(n)$ denote the number of plays of arm k. Then if k is the index of a suboptimal arm then Each sub-optimal arm k is played in expectation at most*

$$\mathbb{E}[T_k(n)] \leq \frac{16C^2 \ln n}{(1-\epsilon)^2 \triangle_k^2} + A(\epsilon) + N_c + \frac{\pi^2}{3} + 1. \tag{26}$$

Next, we derive our version of Theorem 3 in Kocsis et al. (2006), which computes the upper bound of the difference between the value backup of an arm with μ^* up to time n.

Theorem 3. *Under the assumptions of Theorem 2,*

$$\left|\mathbb{E}\left[\overline{X}_n(p)\right] - \mu^*\right| \leq |\delta_n^*| + \mathcal{O}\left(\frac{K(C^2 \log n + N_0)}{n}\right)^{\frac{1}{p}}.$$

A lower bound for the times choosing any arm follows

Theorem 4. *(**Lower Bound**) Under the assumptions of Theorem 2, there exists some positive constant ρ such that for all arms k and n, $T_k(n) \geq \lceil \rho \log(n) \rceil$.*

For deriving the concentration of estimated payoff around its mean, we modify Lemma 14 in Kocsis et al. (2006) for power mean: in the proof, we first replace the partial sums term with a partial mean term and modify the following equations accordingly. The partial mean term can then be easily replaced by a partial power mean term and we get

Theorem 5. *Fix an arbitrary $\delta \leq 0$ and fix $p \geq 1$, let $\triangle_n = (\frac{9}{4})^{p-1}(9\sqrt{\frac{1}{C_p} n \log(2/\delta)})$. Let n_0 be such that*

$$\sqrt{n_0} \leq \mathcal{O}(K(C^2 \log n_0 + N_0(1/2))). \tag{27}$$

Then for any $n \geq n_0$, under the assumptions of Theorem 2, the following bounds hold true

$$\Pr(\overline{X}_n(p) \geq \mathbb{E}[\overline{X}_n(p)] + (\triangle_n/n)^{\frac{1}{p}}) \leq \delta \tag{28}$$

$$\Pr(\overline{X}_n(p) \leq \mathbb{E}[\overline{X}_n(p)] - (\triangle_n/n)^{\frac{1}{p}}) \leq \delta \tag{29}$$

Using The Hoeffding-Azuma inequality for Stopped Martingales Inequality (Lemma 10 in Kocsis et al. (2006)), under Assumption 1 and the result from Theorem 4 we get

Theorem 6. *(**Convergence of Failure Probability**) Under the assumptions of Theorem 2, it holds that*

$$\lim_{t \to \infty} \Pr(I_t \neq i^*) = 0. \tag{30}$$

And finally, the following is our main result showing the expected payoff of our Power-UCT.

Theorem 7. *Consider algorithm Power-UCT running on a game tree of depth D, branching factor K with stochastic payoff at the leaves. Assume that the payoffs lie in the interval [0,1]. Then the bias of the estimated expected payoff, $\overline{X_n}$, is $\mathcal{O}(KD(\log(n)/n)^{\frac{1}{p}} + K^D(1/n)^{\frac{1}{p}})$. Further, the failure probability at the root convergences to zero as the number of samples grows to infinity.*

Proof. (Sketch) As for UCT Kocsis et al. (2006), the proof is done by induction on D. When $D = 1$, Power-UCT corresponds to UCB1 with average mean backup at the leaf node, and the proof of convergence follows as the result of Hoeffding's inequality, the expected payoff is guaranteed directly from Theorem 1, Theorem 3 and Theorem 6. Now we assume that the

result holds up to depth $D-1$ and consider the tree of depth D. Running Power-UCT on root node is equivalent to UCB1 on non-stationary bandit settings, but with power mean backup. The error bound of running Power-UCT for the whole tree is the sum of payoff at root node with payoff starting from any node i after the first action chosen from root node until the end. This payoff by induction at depth $D-1$ in addition to the bound from Theorem 3 when the drift-conditions are satisfied, and with straightforward algebra, we can compute the payoff at the depth D, in combination with Theorem 6. Since our induction hypothesis holds for all nodes at a distance of one node from the root, the proof is finished by observing that Theorem 3 and Theorem 5 do indeed ensure that the drift conditions are satisfied. This completes our proof of the convergence of Power-UCT. Interestingly, the proof guarantees the convergence for any finite value of p. □

5. Convex Regularization in Monte-Carlo Tree Search

Consider an MDP $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P}, \gamma \rangle$, as previously defined. Let $\Omega : \Pi \to \mathbb{R}$ be a strongly convex function. For a policy $\pi_s = \pi(\cdot|s)$ and $Q_s = Q(s,\cdot) \in \mathbb{R}^{\mathcal{A}}$, we observe that the Bellman operator $\mathcal{T}_{\pi_s} Q_s = \langle \pi(\cdot|s), Q(s,\cdot) \rangle = \langle \pi_s, Q_s \rangle$. The Legendre-Fenchel transform (or convex conjugate) of Ω is $\Omega^* : \mathbb{R}^{\mathcal{A}} \to \mathbb{R}$, defined as:

$$\Omega^*(Q_s) \triangleq \max_{\pi_s \in \Pi_s} \left\{ \langle \pi_s, Q_s \rangle - \tau \Omega(\pi_s) \right\}, \tag{31}$$

where the temperature τ specifies the strength of regularization. Among the several properties of the Legendre-Fenchel transform, we use the following (Mensch and Blondel, 2018; Geist et al., 2019).

Proposition 1. *Let Ω be strongly convex.*

- *Unique maximizing argument: $\nabla \Omega^*$ is Lipschitz and satisfies*

$$\nabla \Omega^*(Q_s) = \arg\max_{\pi_s \in \Pi_s} \left\{ \langle \pi_s, Q_s \rangle - \tau \Omega(\pi_s) \right\}. \tag{32}$$

- *Boundedness: if there are constants L_Ω and U_Ω such that for all $\pi_s \in \Pi_s$, we have $L_\Omega \leq \Omega(\pi_s) \leq U_\Omega$, then*

$$\max_{a \in \mathcal{A}} Q_s(a) - \tau U_\Omega \leq \Omega^*(Q_s) \leq \max_{a \in \mathcal{A}} Q_s(a) - \tau L_\Omega. \tag{33}$$

- *Contraction: for any $Q_1, Q_2 \in \mathbb{R}^{\mathcal{S} \times \mathcal{A}}$*

$$\| \Omega^*(Q_1) - \Omega^*(Q_2) \|_\infty \leq \gamma \| Q_1 - Q_2 \|_\infty . \tag{34}$$

Note that if $\Omega(\cdot)$ is strongly convex, $\tau\Omega(\cdot)$ is also strongly convex; thus all the properties shown in Proposition 1 still hold[1].
Solving equation (31) leads to the solution of the optimal primal policy function $\nabla \Omega^*(\cdot)$.

1. Other works use the same formula, e.g. Equation (31) in Niculae and Blondel (2017).

$a =$**SelectAction** *(s)*
  return
    $\arg\max_a Q(s,a) + C\sqrt{\frac{\log N(s)}{n(s,a)}}$

SimulateV *(s, depth)*
  $a =$`SelectAction` (s)
  `SimulateQ` (s, a, depth)
  $N(s) \leftarrow N(s) + 1$
  $V(s) \leftarrow \left(\sum_a \frac{n(s,a)}{N(s)} Q(s,a)^p\right)^{1/p}$

$a =$**SelectAction** *(s)*
  $\lambda_s = \epsilon|\mathcal{A}| / \log(\sum_a N(s,a)+1)$
  $\pi(a|s) = \nabla\Omega^*(Q(s)/\tau)(a)$
  $\epsilon_0 \sim$ uniform distribution
  if $\epsilon_0 > \lambda_s$ **then**
    $a \sim \pi(\cdot|s)$
  else
    $a \sim$ randomly from action set
  return a

SimulateV *(s, depth)*
  $a =$`SelectAction` (s)
  `SimulateQ` (s, a, depth)
  $N(s) \leftarrow N(s) + 1$
  $V(s) \leftarrow \tau\Omega^*(Q(s|\cdot)/\tau)$

Figure 3: Comparing Power-UCT (left) and E3W (right) in the SimulateV procedure. The only distinction is in the V value function backup; the two methods are identical in other procedures.

Since $\Omega(\cdot)$ is strongly convex, the dual function $\Omega^*(\cdot)$ is also convex. One can solve the optimization problem (31) in the dual space Nachum and Dai (2020b) as

$$\Omega(\pi_s) = \max_{Q_s \in \mathbb{R}^{\mathcal{A}}} \{\langle \pi_s, Q_s \rangle - \tau\Omega^*(Q_s)\} \tag{35}$$

and find the solution of the optimal dual value function as $\Omega^*(\cdot)$. The Legendre-Fenchel transform of the value conjugate function is the convex function Ω, i.e. $\Omega^{**} = \Omega$. In the subsequent section, we exploit this primal-dual connection based on the Legendre-Fenchel transform, which uses both the conjugate value function and policy function, to establish the regularized MCTS backup and tree policy.

We first study the general framework of convex regularization in MCTS, and then investigate the α-divergence function as a particular form of the convex regularizer with a specific value of a constant α to derive the entropy-based regularization methods in MCTS.

5.1 Regularized Backup and Tree Policy

As discussed in section 3.2, the MCTS tree contains nodes that represent state, action $s \in \mathcal{S}, a \in \mathcal{A}$ and have two attributes: a visitation count $N(s,a)$ and a state-action function $Q_\Omega(s,a)$. Additionally, MCTS constructs a look-ahead tree search $\mathcal{T}$ in real-time during simulation to determine the optimal action at the root node. When a trajectory of state, action $s_0, a_0, s_1, a_1 ..., s_T$ is obtained, s_T is defined as the leaf node corresponding to the reached state s_T. Assuming s_T is expanded, its corresponding value function $V(s_T)$ is initialized as an estimate returned from an evaluation function computed in s_T, e.g. a discounted cumulative reward averaged over multiple rollouts, or the value-function of node

s_T returned by a value-function approximator, e.g. a neural network pretrained with deep Q-learning (Mnih et al., 2015), as done in (Silver et al., 2016; Xiao et al., 2019). Its respective state action values are initialized as $Q_\Omega(s_T, a) = 0$, and $N(s_T, a) = 0$ for all $a \in \mathcal{A}$. Unlike UCT, which uses the average mean to backpropagate the value function of each node in the tree (as mentioned in section 3.3), statistical information is backpropagated along the trajectory in MCTS by updating the visitation count using $N(s_t, a_t) = N(s_t, a_t) + 1$, and the action-value functions by

$$Q_\Omega(s_t, a_t) = \begin{cases} r(s_t, a_t) + \gamma V_\Omega(s_T) & \text{if } t = T, \\ r(s_t, a_t) + \gamma\tau\Omega^*(Q_\Omega(s_{t+1})/\tau) & \text{if } t < T, \end{cases} \tag{36}$$

where $Q_\Omega(s_t) \in \mathbb{R}^{\mathcal{A}}$ with $Q_\Omega(s_t, a), \forall a \in \mathcal{A}$, $r(s_t, a_t)$ is an average of collected rewards up to time t at state action s_t, a_t, $V_\Omega(s_t)$ is the regularized value function. Through the use of the convex conjugate in Equation (36), we extent the E2W sampling strategy which is limited to maximum entropy regularization (Xiao et al., 2019) and derive a novel sampling strategy that generalizes to any convex regularizer

$$\pi_t(a_t|s_t) = (1 - \lambda_{s_t})\nabla\Omega^*(Q_\Omega(s_t)/\tau)(a_t) + \frac{\lambda_{s_t}}{|\mathcal{A}|}, \tag{37}$$

where $\lambda_{s_t} = \epsilon|\mathcal{A}|/\log(\sum_a N(s_t,a)+1)$ with $\epsilon > 0$ as an exploration parameter, and $\nabla\Omega^*$ depends on the measure in use (see Table 1 for maximum, relative, and Tsallis entropy). The use of the convex conjugate in this equation results in a novel sampling strategy that generalizes to any convex regularizer. This strategy is called the *Extended Empirical Exponential Weight* (E3W) sampling strategy, which extends the previous E2W sampling strategy by introducing a connection with the duality representation through the Legendre-Fenchel transform. This transformation provides a basic connection for a theory of several state-of-the-art algorithms in reinforcement learning, such as TRPO, SAC, A3C (Geist and Scherrer, 2011). Our result is the first to introduce the connection with MCTS.

5.2 Convergence Rate to Regularized Objective

Our findings demonstrate that by assuming a σ^2-subgaussian distribution for each node in the tree, the regularized value V_Ω can be estimated effectively at the root state $s \in \mathcal{S}$. This result expands upon the analysis presented in (Xiao et al., 2019), which only considers the use of maximum entropy. The primary outcomes of this study are presented here, while a comprehensive proof of all the theorems can be found in the Appendix section.

Theorem 8. *At the root node s where $N(s)$ is the number of visitations, with $\epsilon > 0$, $V_\Omega(s)$ is the estimated value, with constant C and $\hat{C}$, we have*

$$\mathbb{P}(|V_\Omega(s) - V_\Omega^*(s)| > \epsilon) \leq C\exp\{\frac{-N(s)\epsilon}{\hat{C}\sigma\log^2(2 + N(s))}\}, \tag{38}$$

where $V_\Omega(s) = \Omega^*(Q_s)$ and $V_\Omega^*(s) = \Omega^*(Q_s^*)$.

From this theorem, we obtain that E3W ensures the exponential convergence rate of choosing the best action a^* at the root node.

Theorem 9. *Let a_t be the action returned by E3W at step t. For large enough t and constants $C, \hat{C}$*

$$\mathbb{P}(a_t \neq a^*) \leq Ct \exp\{-\frac{t}{\hat{C}\sigma(\log(t))^3}\}. \tag{39}$$

For any strongly convex regularizer, this outcome demonstrates that the exponential convergence rate of selecting the optimal action at the root node holds true, which has been previously demonstrated for the maximum entropy case in a study by Xiao et al. (2019).

5.3 Entropy-Regularization Backup Operators

This section focuses on entropic-based regularizers as backup operators and sampling strategies in MCTS, rather than generic strongly convex regularizers. Table 1 shows the Legendre-Fenchel transform and maximizing argument of entropic-based regularizers, which can respectively replace the backup operation (Equation 36) and sampling strategy E3W (Equation 37). The maximum entropy regularization is used in the MENTS algorithm (Xiao et al., 2019). This approach is similar to the maximum entropy RL framework that encourages exploration (Haarnoja et al., 2018; Schulman et al., 2017a). We introduce two other entropic-based regularizer algorithms in MCTS: *Relative Entropy Monte-Carlo Planning* (RENTS), which is inspired by trust-region (Schulman et al., 2015; Belousov and Peters, 2019) and proximal optimization methods (Schulman et al., 2017b) in RL and uses the relative entropy of the policy update, and Tsallis entropy, which has been recently used in RL to encourage the learning of sparse policies (Lee et al., 2018). We call this algorithm: *Tsallis Entropy Monte-Carlo Planning* (TENTS). Contrary to maximum and relative entropy, the definition of the Legendre-Fenchel and maximizing argument of Tsallis entropy is non-trivial, being

$$\Omega^*(Q_t) = \tau \cdot \text{spmax}(Q_t(s,\cdot)/\tau), \tag{40}$$

$$\nabla\Omega^*(Q_t) = \max\{\frac{Q_t(s,a)}{\tau} - \frac{\sum_{a\in\mathcal{K}} Q_t(s,a)/\tau - 1}{|\mathcal{K}|}, 0\}, \tag{41}$$

where spmax is defined for any function $f : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ as

$$\text{spmax}(f(s,\cdot)) \triangleq \sum_{a\in\mathcal{K}} \left(\frac{f(s,a)^2}{2} - \frac{(\sum_{a\in\mathcal{K}} f(s,a) - 1)^2}{2|\mathcal{K}|^2} \right) + \frac{1}{2},$$

and $\mathcal{K}$ is the set of actions that satisfy $1 + if(s,a_i) > \sum_{j=1}^{i} f(s,a_j)$, with a_i indicating the action with the i-th largest value of $f(s,a)$ (Lee et al., 2018). It is worth noting that implementing the Tsallis entropy does not pose a significant challenge. While it does require additional computation that may take up to $O(|\mathcal{A}|\log(|\mathcal{A}|))$ time in the worst-case scenario, the order of Q-values remains unchanged between rollouts, thus reducing computational complexity in practice.

5.4 Regret Analysis

At the root node, let each children node i be assigned with a random variable X_i, with mean value V_i, while the quantities related to the optimal branch are denoted by $*$, e.g.

Table 1: List of entropy regularizers with Legendre-Fenchel transforms (regularized value functions) and maximizing arguments (regularized policies).

Entropy	**Regularizer** $\Omega(\pi_s)$	**Value** $\Omega^*(Q_s)$	**Policy** $\nabla\Omega^*(Q_s)$
Maximum	$\sum_a \pi(a\|s)\log\pi(a\|s)$	$\tau\log\sum_a e^{\frac{Q(s,a)}{\tau}}$	$\frac{e^{\frac{Q(s,a)}{\tau}}}{\sum_b e^{\frac{Q(s,b)}{\tau}}}$
Relative	$\mathrm{D}_{\mathrm{KL}}(\pi_t(a\|s)\|\|\pi_{t-1}(a\|s))$	$\tau\log\sum_a \pi_{t-1}(a\|s)e^{\frac{Q_t(s,a)}{\tau}}$	$\frac{\pi_{t-1}(a\|s)e^{\frac{Q_t(s,a)}{\tau}}}{\sum_b \pi_{t-1}(b\|s)e^{\frac{Q_t(s,b)}{\tau}}}$
Tsallis	$\frac{1}{2}(\parallel \pi(a\|s) \parallel_2^2 -1)$	Equation (40)	Equation (41)

mean value V^*. At each timestep n, the mean value of variable X_i is V_{i_n}. Even though there are some relevant works in literature focus on simple regret for MCTS Danihelka et al. (2021); Shperberg et al. (2017); Hay et al. (2014); Feldman and Domshlak (2013); Tolpin and Shimony (2012). In this work, we study the psudo-regret (Coquelin and Munos, 2007). The pseudo-regret (Coquelin and Munos, 2007) at the root node, at timestep n, is defined as $R_n^{\mathrm{UCT}} = nV^* - \sum_{t=1}^n V_{i_t}$. Similarly, we define the regret of E3W at the root node of the tree as

$$R_n = nV^* - \sum_{t=1}^n V_{i_t} = nV^* - \sum_{t=1}^n \mathbb{I}(i_t = i)V_{i_t} = nV^* - \sum_i V_i \sum_{t=1}^n \hat{\pi}_t(a_i|s), \tag{42}$$

where $\hat{\pi}_t(\cdot)$ is the policy at time step t, and $\mathbb{I}(\cdot)$ is the indicator function.
The expected regret is defined as

$$\mathbb{E}[R_n] = nV^* - \sum_{t=1}^n \langle \hat{\pi}_t(\cdot), V(\cdot)\rangle\,. \tag{43}$$

Theorem 10. *Consider an E3W policy applied to the tree. Let define* $\mathcal{D}_{\Omega^*}(x,y) = \Omega^*(x) - \Omega^*(y) - \nabla\Omega^*(y)(x-y)$ *as the Bregman divergence between* x *and* y*, The expected pseudo regret* R_n *satisfies*

$$\mathbb{E}[R_n] \leq -\tau\Omega(\hat{\pi}) + \sum_{t=1}^n \mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot) + V(\cdot), \hat{V}_t(\cdot)) + \mathcal{O}(\frac{n}{\log n}).$$

This theorem bounds the regret of E3W for a generic convex regularizer Ω; the regret bounds for each entropy regularizer can be easily derived from it. Let $m = \min_a \nabla\Omega^*(a|s)$.

Corollary 1. *Maximum entropy regret:* $\mathbb{E}[R_n] \leq \tau(\log|\mathcal{A}|) + \frac{n|\mathcal{A}|}{\tau} + \mathcal{O}(\frac{n}{\log n})$.

Corollary 2. *Relative entropy regret:* $\mathbb{E}[R_n] \leq \tau(\log|\mathcal{A}| - \frac{1}{m}) + \frac{n|\mathcal{A}|}{\tau} + \mathcal{O}(\frac{n}{\log n})$.

Corollary 3. *Tsallis entropy regret:* $\mathbb{E}[R_n] \leq \tau(\frac{|\mathcal{A}|-1}{2|\mathcal{A}|}) + \frac{n|\mathcal{K}|}{2} + \mathcal{O}(\frac{n}{\log n})$.

Remarks. The regret bound of UCT and its variance have been studied for non-regularized MCTS with a binary tree in previous work (Coquelin and Munos, 2007). In contrast, our regret bound analysis in Theorem 10 is applicable to general regularized MCTS. The corollaries suggest that the maximum and relative entropy have similar bounds, with the bounds for relative entropy being slightly smaller due to $\frac{1}{m}$. Interestingly, the bounds for Tsallis entropy become tighter as the number of actions increases, resulting in smaller regret for problems with a high branching factor. This result demonstrates the advantage of Tsallis entropy over other entropy regularizers in complex problems, as empirically verified in Section 7.

5.5 Error Analysis

We analyse the error of the regularized value estimate at the root node $n(s)$ w.r.t. the optimal value: $\varepsilon_\Omega = V_\Omega(s) - V^*(s)$.

Theorem 11. *For any $\delta > 0$ and generic convex regularizer Ω, with some constant $C, \hat{C}$, with probability at least $1-\delta$, ε_Ω satisfies*

$$-\sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} - \frac{\tau(U_\Omega - L_\Omega)}{1-\gamma} \leq \varepsilon_\Omega \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}}. \tag{44}$$

As far as we are aware, this theorem is the first to provide an error analysis of value estimation at the root node in MCTS with convex regularization. In Table 1, we provide a better understanding of the impact of each entropy regularizer by specializing the bound in Equation (44) to each regularizer. From (Lee et al., 2018), we know that for maximum entropy $\Omega(\pi_t) = \sum_a \pi_t \log \pi_t$, we have $-\log|\mathcal{A}| \leq \Omega(\pi_t) \leq 0$; for relative entropy $\Omega(\pi_t) = \mathrm{KL}(\pi_t||\pi_{t-1})$, if we define $m = \min_a \pi_{t-1}(a|s)$, then we can derive $0 \leq \Omega(\pi_t) \leq -\log|\mathcal{A}| + \log\frac{1}{m}$; and for Tsallis entropy $\Omega(\pi_t) = \frac{1}{2}(\| \pi_t \|_2^2 - 1)$, we have $-\frac{|\mathcal{A}|-1}{2|\mathcal{A}|} \leq \Omega(\pi_t) \leq 0$. Defining $\Psi = \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}}$ yields the following corollaries.

Corollary 4. *Maximum entropy error:* $-\Psi - \dfrac{\tau \log|\mathcal{A}|}{1-\gamma} \leq \varepsilon_\Omega \leq \Psi.$

Corollary 5. *Relative entropy error:* $-\Psi - \dfrac{\tau(\log|\mathcal{A}| - \log\frac{1}{m})}{1-\gamma} \leq \varepsilon_\Omega \leq \Psi.$

Corollary 6. *Tsallis entropy error:* $-\Psi - \dfrac{|\mathcal{A}|-1}{2|\mathcal{A}|}\dfrac{\tau}{1-\gamma} \leq \varepsilon_\Omega \leq \Psi.$

Corollaries 4-6 demonstrate that in scenarios where the number of actions $|\mathcal{A}|$ is large, the error of TENTS is the smallest and that the lower bound of RENTS is smaller than the lower bound of MENTS.

6. α-divergence in MCTS

In this section, we show how to use α-divergence as a convex regularization function to generalize the entropy regularization in MCTS and respectively derive MENTS, RENTS and

TENTS. Additionally, we show how to derive power mean (which is used as the backup operator in Power-UCT) using α-divergence as the distance function to replace the Euclidean distance in the definition of the empirical average mean value. Finally, we study the regret bound and error analysis of the α-divergence regularization in MCTS.

6.1 α-divergence Regularization in MCTS

We introduce α-divergence regularization to MCTS. Denote the Legendre-Fenchel transform (or convex conjugate) of α-divergence regularization with $\Omega^* : \mathbb{R}^{\mathcal{A}} \to \mathbb{R}$, defined as:

$$\Omega^*(Q_s) \triangleq \max_{\pi_s \in \Pi_s} \langle \pi_s, Q_s \rangle - \tau H_\alpha(\pi_s), \tag{45}$$

where the temperature τ specifies the strength of regularization, and $H_\alpha(\pi_s)$ is the generalized Tsallis entropy derived from α function defined in (9). Note that α-divergence of the current policy π_s and the uniform policy has the same form as $H_\alpha(\pi_s)$. It is known that:

- when $\alpha = 1$, we have the regularizer $H_1(\pi_s) = \pi_s \log \pi_s$, which is the Shannon entropy, getting MENTS. Note that if we apply the α-divergence with $\alpha = 1$, we get RENTS;
- when $\alpha = 2$, we have the regularizer $H_2(\pi_s) = \frac{1}{2}(\pi_s - 1)^2$, and derive Tsallis entropy, getting TENTS.

For $\alpha > 1, \alpha \neq 2$ we can derive (Chen et al., 2018)

$$\nabla\Omega^*(Q_t) = \left(\max\left\{ \frac{Q^{\pi^*_\tau(s,a)}}{\tau} - \frac{c(s)}{\tau}, 0 \right\}(\alpha - 1) \right)^{\frac{1}{\alpha - 1}} \tag{46}$$

where

$$c(s) = \tau \frac{\sum_{a \in \mathcal{K}(s)} \frac{Q^{\pi^*_\tau(s,a)}}{\tau} - 1}{\|\mathcal{K}(s)\|} + \tau\left(1 - \frac{1}{\alpha - 1}\right), \tag{47}$$

with $\mathcal{K}(s)$ representing the set of actions with non-zero chance of exploration in state s, as determined below

$$\mathcal{K}(s) = \left\{ a_i \middle| 1 + i\frac{Q^{\pi^*_\tau(s,a_i)}}{\tau} > \sum_{j=1}^{i} \frac{Q^{\pi^*_\tau(s,a_j)}}{\tau} + i(1 - \frac{1}{\alpha - 1}) \right\}, \tag{48}$$

where a_i denotes the action with the $i-$th highest Q-value in state s. and the regularized value function

$$\Omega^*(Q_t) = \left\langle \nabla\Omega^*(Q_t), Q^{\pi^*_\tau(s,a)} \right\rangle - \tau H_\alpha(\nabla\Omega^*(Q_t)). \tag{49}$$

6.2 Connecting Power Mean with α-divergence

In order to connect the Power-UCT approach that we introduced in Section 4.2 with α-divergence, we study here the entropic mean (Ben-Tal et al., 1989) which uses f-divergence, of which α-divergence is a special case, as the distance measure. Since power mean is a

special case of the entropic mean, the entropic mean allows us to connect the geometric properties of the power mean used in Power-UCT with α-divergence.

In more detail, let $a = (a_1, a_2, ...a_n)$ be given strictly positive numbers and let $w = (w_1, w_2, ..., w_n)$ be given weights and $\sum_{i=1}^{n} w_i = 1, w_i > 0, i = 1...n$. Let's define $dist(\alpha, \beta)$ as the distance measure between $\alpha, \beta > 0$ that satisfies

$$dist(\alpha, \beta) = \begin{cases} 0 \text{ if } \alpha = \beta \\ > 0 \text{ if } \alpha \neq \beta \end{cases} \tag{50}$$

When we consider the distance as f-divergence between the two distributions, we get the entropic mean of $a = (a_1, a_2, ...a_n)$ with weights $w = (w_1, w_2, ..., w_n)$ as

$$\text{mean}_w(a) = \min_{x>0} \left\{ \sum_{i=1}^{n} w_i a_i f\left(\frac{x}{a_i}\right) \right\}. \tag{51}$$

When applying $f_\alpha(x) = \frac{x^{1-p}-p}{p(p-1)} + \frac{x}{p}$, with $p = 1 - \alpha$, we get

$$\text{mean}_w(a) = \left(\sum_{i=1}^{n} w_i a_i^p \right)^{\frac{1}{p}}, \tag{52}$$

which is equal to the power mean.

6.3 Regret and Error Analysis of α-divergence in Monte-Carlo Tree Search

We measure how different values of α in the α-divergence function affect the regret in MCTS.

Theorem 12. *When $\alpha \in (0, 1)$, the regret of E3W is*

$$\mathbb{E}[R_n] \leq \frac{\tau}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha} - 1) + n(2\tau)^{-1}|\mathcal{A}|^{\alpha} + \mathcal{O}(\frac{n}{\log n}).$$

For $\alpha \in (1, \infty)$, we derive the following results

Theorem 13. *When $\alpha \in (1, \infty)$, the regret of E3W is*

$$\mathbb{E}[R_n] \leq \frac{\tau}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha} - 1) + \frac{n|\mathcal{K}|}{2} + \mathcal{O}(\frac{n}{\log n}).$$

where $|\mathcal{K}|$ is the number of actions that are assigned non-zero probability in the policy at the root node. Note that when $\alpha = 1, 2$, please refer to Corollary 1, 2, 3.
We analyse the error of the regularized value estimate at the root node $n(s)$ w.r.t. the optimal value: $\varepsilon_\Omega = V_\Omega(s) - V^*(s)$. where Ω is the α-divergence regularizer f_α.

Theorem 14. *For any $\delta > 0$ and α-divergence regularizer f_α ($\alpha \neq 1, 2$), with some constant $C, \hat{C}$, with probability at least $1 - \delta$, ε_Ω satisfies*

$$-\sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} - \frac{\tau}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha} - 1) \leq \varepsilon_\Omega \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}}. \tag{53}$$

For $\alpha = 1, 2$, please refer to Corollary 4, Corollary 5, Corollary 6. We can see that when α increases, the error bound decreases.

7. Empirical Evaluation

The empirical evaluation consists of three parts. Firstly, we evaluate Power-UCT using the *FrozenLake*, *Copy*, *Rocksample*, and *Pocman* environments. This evaluation demonstrates the advantages of using the power mean backup operator over the average mean and maximum backup operators in UCT. We aim to answer the following empirical questions: (1) Does the power mean offer superior performance in MDP and POMDP MCTS tasks compared to the regular mean operator? (2) How does the choice of p value affect the overall performance? (3) How does Power-UCT compare to state-of-the-art methods in tree-search? We choose the recent MENTS algorithm (Xiao et al., 2019) as a representative state-of-the-art method. For MENTS we find the best combination of the two hyper-parameters (temperature τ and exploration ϵ) by grid search. In MDP tasks, we find the UCT exploration constant using grid search. For Power-UCT, we find the p-value by increasing it until performance starts to decrease. In the *Rocksample* and *Pocman* environments, we utilize the source code provided by the authors of the POMCP paper (Silver et al., 2010) to execute the POMCP algorithm. Subsequently, we develop our Power-UCT method based on this implementation.

In the second part, we evaluate the benefits of our proposed entropy-based MCTS regularizers (MENTS, RENTS, and TENTS) in different tasks. First, we use Synthetic Tree to complement our theoretical analysis and provide an interpretable demonstration of the effects of our proposed regularizers. Second, we apply our entropy-based regularizers to various Atari games using a simplified version of the AlphaGo algorithm to highlight the benefits of our approach. Our implementation is a simplified version of the original algorithm, where we remove various tricks in favor of better interpretability. For the same reason, we do not compare with the most recent and state-of-the-art MuZero (Schrittwieser et al., 2019), as this is a slightly different solution highly tuned to maximize performance, and a detailed description of its implementation is not available.

The learning time of AlphaZero can be slow in problems with high branching factor, due to the need of a large number of MCTS simulations for obtaining good estimates of the randomly initialized action-values. To overcome this problem, AlphaGo (Silver et al., 2016) initializes the action-values using the values retrieved from a pretrained state-action value network, which is kept fixed during the training.

Finally, we use the Synthetic Tree environment to show how α-divergence help to balance between exploration and exploitation in MCTS effectively. We measure the error of value estimate and the regret at the root node with different values of α to show that the empirical results match our theoretical analysis. We release the source code to reproceduce the experimental results[2].

7.1 *FrozenLake*

In the OpenAI Gym (Brockman et al., 2016), the *FrozenLake* problem is a well-known empirical MDP environment. The task is to navigate an agent in an 8x8 ice-grid world to reach a target position while avoiding unstable locations that lead to a fall into the water. The stochastic nature of the environment makes the task challenging, as the agent moves

2. https://github.com/damquangtuan/UnifiedMCTS.git

Table 2: Mean and two times standard deviation of the success rate, over 500 evaluation runs, of UCT, Power-UCT and MENTS in *FrozenLake* from OpenAI Gym. The top row of each table shows the number of simulations used for tree-search at each time step.

Algorithm	4096	16384	65536	262144
UCT	0.08 ± 0.02	0.23 ± 0.04	0.54 ± 0.05	0.69 ± 0.04
p=2.2	0.12 ± 0.03	0.32 ± 0.04	$\mathbf{0.62 \pm 0.04}$	$\mathbf{0.81 \pm 0.03}$
p=max	0.10 ± 0.03	0.36 ± 0.04	0.55 ± 0.04	0.69 ± 0.04
MENTS	$\mathbf{0.28 \pm 0.04}$	$\mathbf{0.46 \pm 0.04}$	$\mathbf{0.62 \pm 0.04}$	0.74 ± 0.04

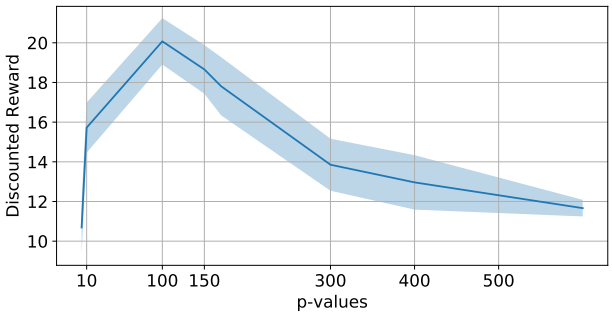


Figure 4: Evaluating Power-UCT w.r.t. different p-values: The mean discounted total reward at 65536 simulations (shaded area denotes standard error) over 100 evaluation runs.

only one-third of the time in the intended direction and the rest of the time in one of the two tangential directions. Reaching the target position yields the agent a reward of +1, while all other outcomes yield a reward of zero. Table 2 shows that Power-UCT performs better than UCT, and its performance outperforms MENTS when increasing the number of simulations.

7.2 Copy Environment

Our aim is to investigate Power-UCT and MENTS, RENTS's performance in environments with high branching factors. We conduct our study using the Copy environment in OpenAI gym, where the agent's task is to copy characters from an input band to an output band. The agent can move, read the input band, and write a character from the alphabet to the output band at each time step. Thus, the number of available actions scales with the size of the alphabet. The agent receives a reward of +1 for each correct character copied. If the agent copies an incorrect character or runs out of time, the task ends. The total accumulated rewards are equal to the size of the input band.

In contrast to our previous experiments, the tree-search runs only once at the start, and there is no re-planning between actions. As a result, all actions are selected based on value estimates from the initial search. The input band size is fixed to 40 characters, while we vary the alphabet size to test different branching factors. Table 3 demonstrates that

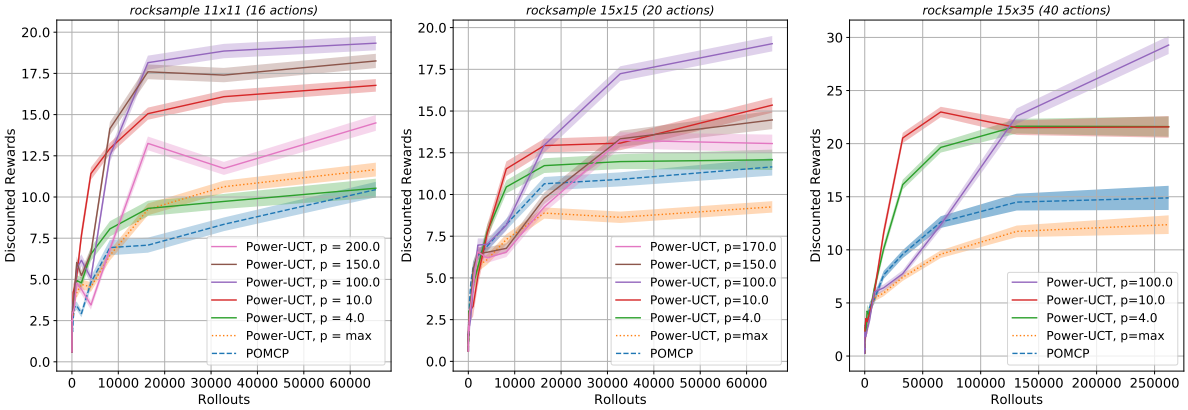


Figure 5: Performance of Power-UCT compared to POMCP in *rocksample*. The mean of total discounted reward over 1000 evaluation runs is shown by thick lines while the shaded area shows standard error.

Table 3: Mean and two times standard deviation of discounted total reward, over 100 evaluation runs, of UCT, Power-UCT and MENTS in the copy environment with 144 actions (top) and 200 actions (bottom). Top row: number of simulations at each time step.

Algorithm	512	2048	8192	32768
UCT	2.6 ± 0.98	$9. \pm 1.17$	34.66 ± 1.68	$\mathbf{40. \pm 0.}$
$p = 3$	3.24 ± 1.17	$\mathbf{12.35 \pm 1.14}$	$\mathbf{40. \pm 0.}$	$\mathbf{40. \pm 0.}$
$p = \max$	2.56 ± 1.48	9.55 ± 3.06	37.52 ± 5.11	39.77 ± 0.84
MENTS	$\mathbf{3.26 \pm 1.32}$	11.96 ± 2.94	39.37 ± 1.15	39.35 ± 0.95
RENTS	3.21 ± 1.42	11.71 ± 2.94	39.96 ± 0.56	$\mathbf{40.0 \pm 0.0}$

(a) 144 Actions

Algorithm	512	2048	8192	32768
UCT	1.98 ± 0.63	6.43 ± 1.36	24.5 ± 1.56	$\mathbf{40. \pm 0.}$
$p = 3$	$\mathbf{2.55 \pm 0.99}$	$\mathbf{9.11 \pm 1.41}$	$\mathbf{36.02 \pm 1.72}$	$\mathbf{40. \pm 0.}$
$p = \max$	2.03 ± 1.37	6.99 ± 2.51	27.89 ± 4.12	39.93 ± 0.51
MENTS	2.44 ± 1.34	8.86 ± 2.65	34.63 ± 5.6	39.42 ± 0.99
RENTS	2.41 ± 1.32	8.78 ± 2.68	34.76 ± 4.89	$\mathbf{40.0 \pm 0.0}$

(b) 200 Actions

Algorithm	512	2048	8192	32768
UCT	1.65 ± 0.95	3.45 ± 1.07	13.9 ± 1.43	$\mathbf{40. \pm 0.}$
$p = 3$	$\mathbf{2.55 \pm 0.99}$	$\mathbf{9.11 \pm 1.41}$	$\mathbf{36.02 \pm 1.72}$	$\mathbf{40. \pm 0.}$
$p = \max$	1.48 ± 1.04	4.49 ± 2.09	16.95 ± 3.49	39.94 ± 0.47
MENTS	1.71 ± 0.90	5.28 ± 1.70	21.08 ± 2.98	39.71 ± 0.91
RENTS	1.71 ± 0.90	5.28 ± 1.70	21.03 ± 2.93	$\mathbf{40.0 \pm 0.0}$

(c) 300 Actions

Power-UCT is much more efficient than the baseline UCT in solving the task. Additionally, we observe that MENTS, RENTS and Power-UCT with $p = \infty$ exhibit more significant variance than Power-UCT with a finite value of p. They cannot solve the task reliably as they do not reach the maximum reward of 40 with zero standard deviation.

7.3 Rocksample and PocMan

We evaluate the Power-UCT's performance in POMDP problems against the standard POMCP algorithm (Silver and Veness, 2010). Since the state is not fully observable in POMDPs, POMCP assigns a unique action-observation history to each tree node, which is a sufficient statistic for optimal decision-making in POMDPs instead of the full state. Similar to fully observable UCT, POMCP selects actions using the UCB1 bandit strategy (Auer et al., 2002). To conduct a fair comparison, we modify POMCP by replacing the backup operator with the power mean backup operator used in fully observable UCT. This modification results in a POMDP version of Power-UCT. Similarly, we modify POMCP to implement the MENTS approach. Next, we discuss the evaluation of the POMDP based Power-UCT, MENTS, and POMCP, in *rocksample* and *pocman* environments (Silver and Veness, 2010).

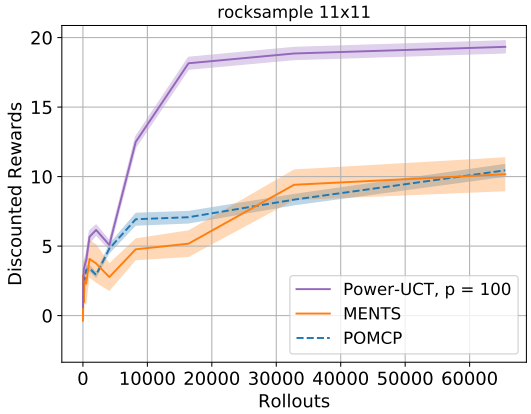


Figure 6: Performance of Power-UCT compared to UCT and MENTS in *rocksample* 11x11. The mean of discounted total reward over 1000 evaluation runs is shown by thick lines while the shaded area shows standard error.

Rocksample. The benchmark *rocksample* (n,k) (Smith and Simmons, 2004) emulates a Mars Explorer robot in a n x n grid containing k rocks. The goal is to identify valuable rocks with a long-range sensor, sample valuable rocks, and then exit the map to the east. The agent can perform $k+5$ actions, such as moving in one of the four directions (north, south, east, west), tracking down one of the k rocks, or taking a rock sample. *rock sampling* requires robust exploration to discover informative actions that may not provide immediate rewards but may provide high long-term rewards. We examine three versions of *rocksample* with different grid sizes and rock counts: *rocksample* (11,11), *rocksample* (15,15), and *rocksample* (15,35). We set the value of the exploration constant to the difference between the maximum and minimum immediate rewards ($R_{max} - R_{min}$), as in (Silver and Veness, 2010). The superiority of Power-UCT over POMCP is evident in Fig. 5 for almost all values of p. We perform a sensitivity analysis for Power-UCT in *rocksample* (11x11) with different p values in 65536 simulations, and the results are shown in Fig. 4. The figure shows that it is easy to find a suitable p-value in *rocksample*. Additionally, Fig. 6 demonstrates that Power-UCT significantly outperforms MENTS in *rocksample* (11,11). It is possible that MENTS does

Table 4: Discounted total reward in *pocman* for the comparison methods. Mean ± standard error are computed from 1000 simulations except in MENTS where we ran 100 simulations.

	1024	4096	16384	65536
POMCP	30.89 ± 1.4	33.47 ± 1.4	33.44 ± 1.39	32.36 ± 1.6
$p = max$	14.82 ± 1.52	14.91 ± 1.52	14.34 ± 1.52	14.98 ± 1.76
$p = 10$	29.14 ± 1.61	35.26 ± 1.56	44.14 ± 1.60	$\mathbf{53.30 \pm 1.46}$
$p = 30$	28.78 ± 1.44	33.92 ± 1.56	42.45 ± 1.54	49.66 ± 1.70
MENTS	$\mathbf{54.08 \pm 3.20}$	$\mathbf{55.37 \pm 3.0}$	$\mathbf{53.90 \pm 2.86}$	51.03 ± 3.36

not explore sufficiently in this task, but further analysis of MENTS is required to confirm this.

Pocman. In addition to the previous experiment, we evaluate the performance of Power-UCT in another POMDP domain, namely the *pocman* problem (Silver and Veness, 2010). The *pocman* problem requires an agent called PocMan to navigate in a maze of size (17x19) using only local observations while trying to eat as many food pellets as possible and avoiding being caught by four ghosts. PocMan receives different rewards such as -1 at each step, $+10$ for eating each food pellet, $+25$ for eating a ghost, and -100 for dying. The results in Table 4 indicate that both Power-UCT and MENTS perform better than POMCP in terms of discounted total rewards. Additionally, with 65536 simulations, Power-UCT outperforms MENTS.

7.4 Synthetic Tree

In the paper by Xiao et al. (2019), a synthetic tree toy problem is used to compare MENTS and UCT. The problem involves a tree with depth d and branching factor k. Each edge of the tree has a random value between 0 and 1, and at each leaf, a Gaussian distribution is used as an evaluation function resembling the return of random rollouts. The mean of the Gaussian distribution is the sum of the values assigned to the edges connecting the root node to the leaf, while the standard deviation is $\sigma = 0.05$[3] To ensure stability, the means are normalized between 0 and 1. Similar to Xiao et al. (2019), we conduct 25 experiments on five trees with five runs each, covering all combinations of branching factors $k = \{2, 4, 6, 8, 10, 12, 14, 16\}$ and depths $d = \{1, 2, 3, 4, 5\}$. We compute the value estimation error at the root node compared to the regularized optimal value, $\varepsilon_\Omega = V_\Omega - V^*_\Omega$, the value estimation error at the root node compared to the unregularized optimal value, $\varepsilon_{\text{UCT}} = V_\Omega - V^*_{\text{UCT}}$, and the regret R as described in Equation (42). For fair comparison, we set $\tau = 0.1$ and $\epsilon = 0.1$ for all algorithms. The behavior of UCT, Power-UCT, and each regularizer for different tree configurations are shown in Figure 9 and 7. In our experiments, we found that TENTS is a robust method that converges quickly to the optimal value and is largely unaffected by increasing tree size, unlike RENTS and MENTS, which converge more slowly. The optimal value obtained by TENTS is very close to the optimal value of UCT and converges faster than the value estimated by UCT. Power-UCT, on the other hand, converges faster than UCT and very closed to TENTS. Regarding regret, UCT is

3. The value of the standard deviation is not provided in Xiao et al. (2019). After trying different values, we observed that our results match the one in Xiao et al. (2019) when using $\sigma = 0.05$.

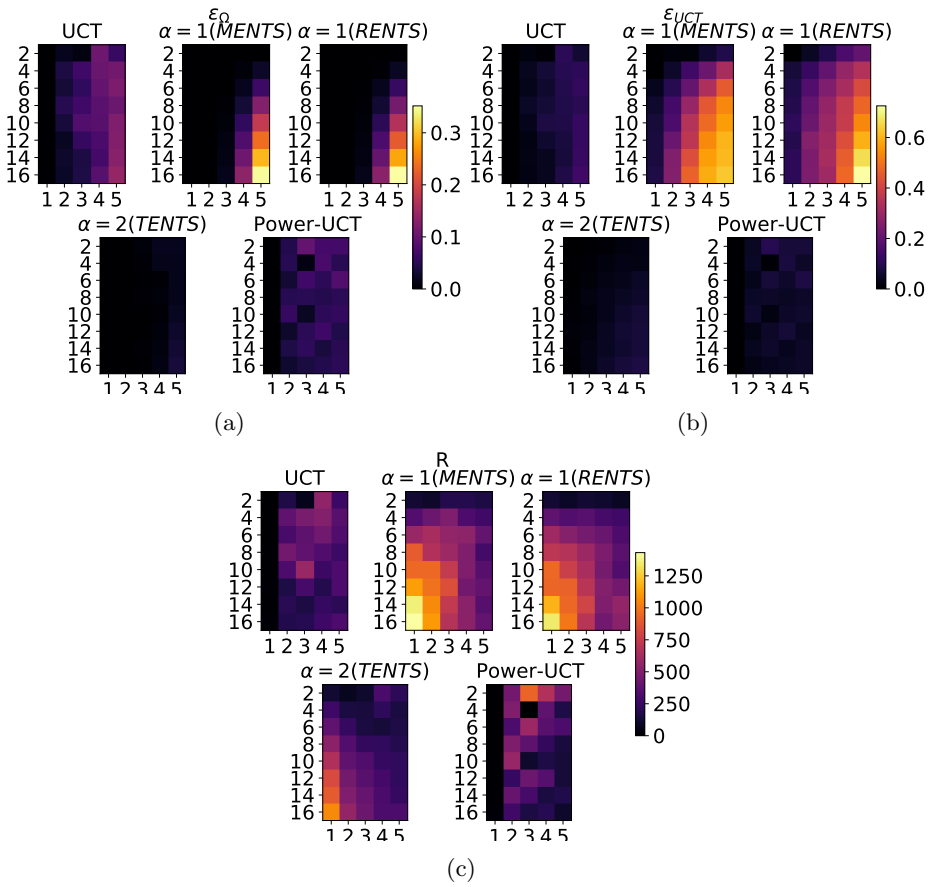
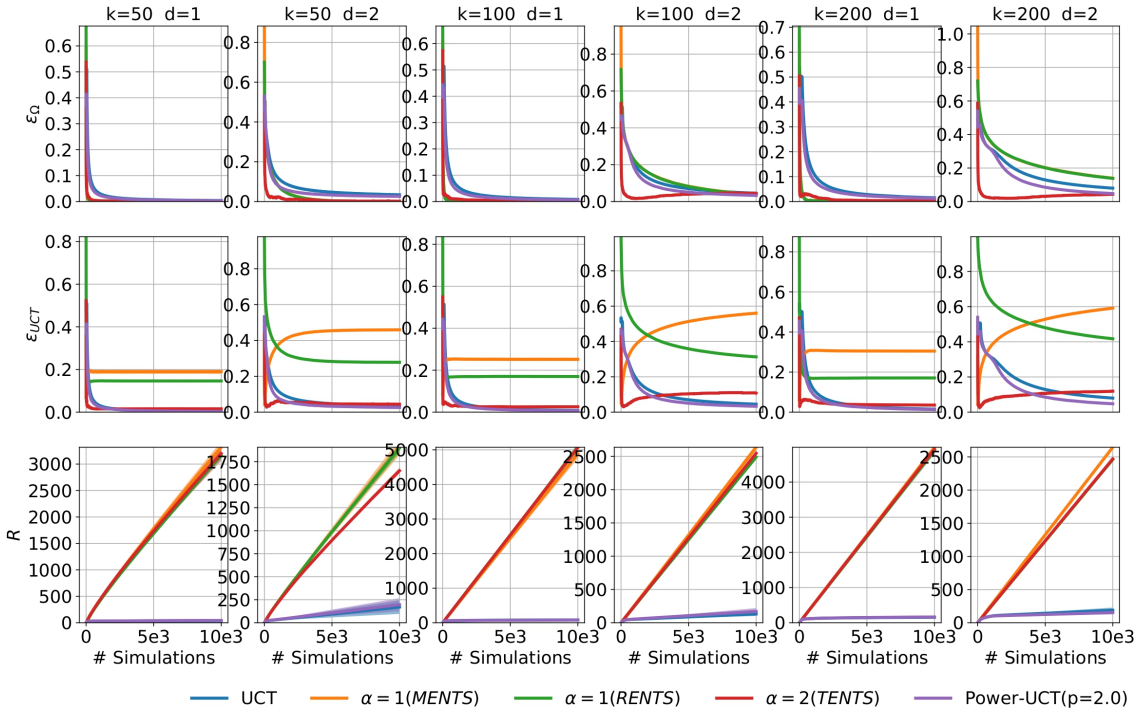


Figure 7: For different branching factor k (rows) and depth d (columns), the heatmaps show: the absolute error of the value estimate at the root node after the last simulation of each algorithm w.r.t. the respective optimal value (a), and w.r.t. the optimal value of UCT (b); regret at the root node (c).

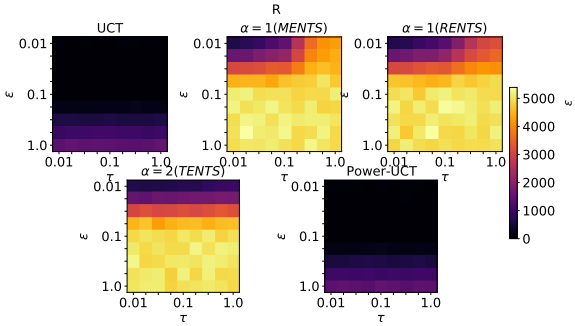
less exploratory than regularized methods, resulting in lower regret but slower convergence. Power-UCT shows lowest regret. However, TENTS has the smallest regret among the regularizers and explores more efficiently than other regularizers. We show in Figure 8(a) that TENTS outperforms other regularized methods in high branching factor problems in terms of approximation error and regret. Additionally, we conduct a sensitivity analysis of each algorithm w.r.t. the values of the exploration coefficient ε and τ in two different trees in Figures 8(b) and 8(c). Our results demonstrate the superiority of TENTS in this toy problem, confirming our theoretical findings about the advantages of TENTS in problems with many actions in terms of approximation error (Corollary 6) and regret (Corollary 3).

7.5 Entropy-regularized AlphaGo

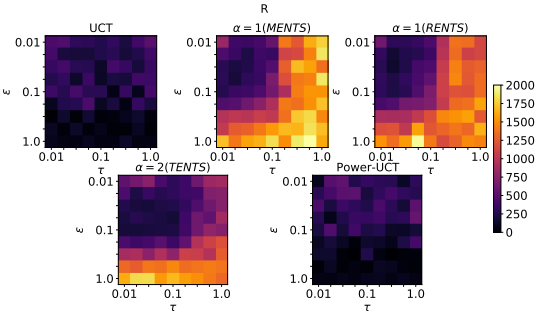
Atari. We measure our entropy-based regularization MCTS algorithms in Atari 2600 (Bellemare et al., 2013) Games. Atari 2600 (Bellemare et al., 2013) is a popular benchmark for



(a) Results in trees with high branching factor.



(b) $k = 100$, $d = 1$.



(c) $k = 8$, $d = 3$.

Figure 8: High branching factor trees (a), regret sensitivity study w.r.t. ε and τ (b, c).

testing Deep RL methods (Mnih et al., 2015; Van Hasselt et al., 2016; Bellemare et al., 2017), but has not been relatively studied in MCTS. In this experiment, we modify the standard AlphaGo algorithm, PUCT, using our regularized value-backup operator and policy selection. We use a pre-trained deep Q-network, using the same experimental setting of Mnih et al. (2015) as prior, to initialize the action value function of each node after the expansion step in the tree as $Q_{\text{init}}(s,a) = (Q(s,a) - V(s))/\tau$, for MENTS and TENTS, as in Xiao et al. (2019). For RENTS, we initialize $Q_{\text{init}}(s,a) = \log P_{\text{prior}}(a|s)) + (Q(s,a) - V(s))/\tau$, where P_{prior} is the Boltzmann distribution induced by the action values $Q(s,.)$ computed from the network. Each experimental run consists of 512 MCTS simulations. To find hyper-

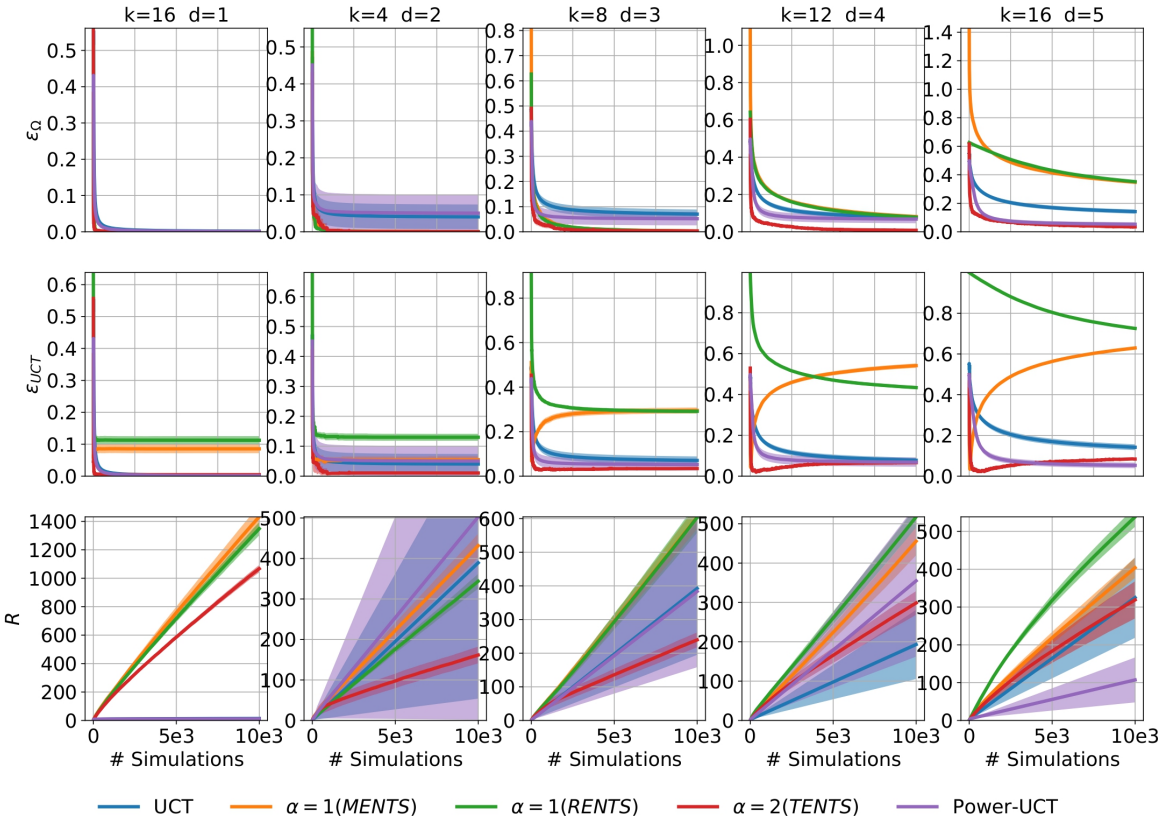


Figure 9: For each algorithm, we show the convergence of the value estimate at the root node to the respective optimal value (top), to the UCT optimal value (middle), and the regret (bottom).

parameters, for each of our regularized MCTS algorithms, we perform a grid search over the temperature parameter τ with a range from 0.01 to 1. In addition, the discount factor is set to $\gamma = 0.99$, and for the PUCT algorithm, we use an exploration constant of $c = 0.1$. The performance of our regularized MCTS algorithms and the standard PUCT and MaxMCTS baselines is evaluated using 22-Atari games in terms of cumulative reward. The results in Table 5 show that our regularized methods outperform the baselines, with TENTS scoring the highest in all games. In particular, TENTS performs significantly better in games with high branching factors, such as Asteroids and Phoenix, confirming the results of our experiment with synthetic trees and the theoretical advantages of TENTS in corollary 3 and 6.

7.6 Power-UCT in Synthetic Tree

We evaluate Power-UCT using the synthetic tree toy problem within the same experimental framework as in the previous section, setting the variance at each node to $\sigma = 0.05$, and normalizing the mean values between 0 and 1. Figure 10 shows the convergence of the value estimations and regret of Power-UCT at the root node in the Synthetic Tree environment. It

Table 5: Average score in Atari over 100 seeds per game. Bold denotes no statistically significant difference to the highest mean (t-test, $p < 0.05$). Bottom row shows # no difference to highest mean.

	UCT	MaxMCTS	$\alpha = 1$(MENTS)	$\alpha = 1$(RENTS)	$\alpha = 2$(TENTS)
Alien	**1,486.80**	**1,461.10**	**1,508.60**	**1,547.80**	**1,568.60**
Amidar	115.62	**124.92**	**123.30**	**125.58**	**121.84**
Asterix	4,855.00	**5,484.50**	**5,576.00**	**5,743.50**	**5,647.00**
Asteroids	873.40	899.60	1,414.70	1,486.40	**1,642.10**
Atlantis	35,182.00	**35,720.00**	**36,277.00**	35,314.00	**35,756.00**
BankHeist	475.50	458.60	**622.30**	**636.70**	**631.40**
BeamRider	**2,616.72**	**2,661.30**	**2,822.18**	2,558.94	**2,804.88**
Breakout	**303.04**	296.14	**309.03**	300.35	**316.68**
Centipede	1,782.18	1,728.69	**2,012.86**	**2,253.42**	**2,258.89**
DemonAttack	579.90	640.80	**1,044.50**	**1,124.70**	**1,113.30**
Enduro	**129.28**	124.20	128.79	**134.88**	**132.05**
Frostbite	1,244.00	1,332.10	**2,388.20**	**2,369.80**	**2,260.60**
Gopher	3,348.40	3,303.00	**3,536.40**	**3,372.80**	**3,447.80**
Hero	3,009.95	3,010.55	**3,044.55**	**3,077.20**	**3,074.00**
MsPacman	1,940.20	1,907.10	2,018.30	**2,190.30**	**2,094.40**
Phoenix	2,747.30	2,626.60	3,098.30	2,582.30	**3,975.30**
Qbert	7,987.25	8,033.50	8,051.25	8,254.00	**8,437.75**
Robotank	**11.43**	11.00	**11.59**	**11.51**	**11.47**
Seaquest	**3,276.40**	**3,217.20**	**3,312.40**	**3,345.20**	**3,324.40**
Solaris	895.00	923.20	**1,118.20**	**1,115.00**	**1,127.60**
SpaceInvaders	778.45	**835.90**	**832.55**	**867.35**	**822.95**
WizardOfWor	685.00	666.00	**1,211.00**	**1,241.00**	**1,231.00**
# Highest mean	6/22	7/22	17/22	16/22	**22/22**

shows that the error in the value estimate at the root node varies with different values of p in different settings. For example, for $k = 10, d = 1$, the errors ϵ_Ω and ϵ_{UCT} converge similarly for $p = 1, 2, 4, 8, 10, 16$, but the regret is smallest for $p = 16$. In the case of $k = 14, d = 3$, $p = 2$, Power-UCT achieves the fastest convergence, but the performance decreases when we increase the value of p with $p = 4, 8, 10, 16$. Furthermore, we evaluate the performance of Power-UCT in high branching factor scenarios as shown in Fig 11. The results show the effectiveness of Power-UCT as depending on the problem settings, we can always find the value of p that Power-UCT outperforms UCT. For example, for $k = 100, d = 2$, Power-UCT performs best with $p = 16$, and its performance decreases as p decreases.

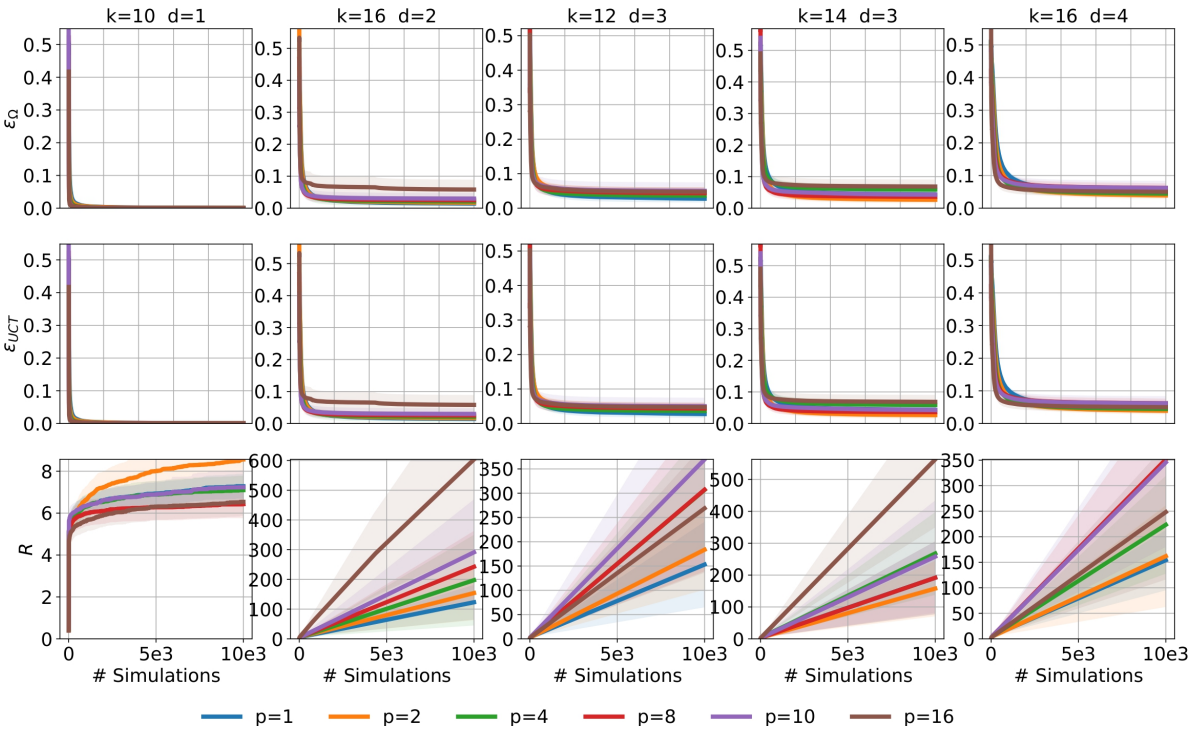


Figure 10: We show the convergence of the value estimate at the root node to the respective optimal value (top), to the UCT optimal value (middle), and the regret (bottom) with different p parameter of Power-UCT in Synthetic tree environment with $p = 1.0$(UCT), 2.0, 4.0, 6.0, 8.0, 10.0, 16.0.

7.7 α-divergence in Synthetic Tree

We further use the toy problem Synthetic Tree to measure how the α-divergence help to balance between exploration and exploitation in MCTS. We use the same experimental settings as in the last section with the variance of the distributions at each node of the Synthetic Tree is set to $\sigma = 0.05$. The mean value of each distribution at each node of the toy problem is normalized between 0 and 1 for stabilizing. We set the temperature $\tau = 0.1$ and the exploration $\epsilon = 0.1$. Figure 13 illustrates the heatmap of the absolute error of the value estimate at the root node after the last simulation of each algorithm w.r.t. the respective optimal regularized value, the optimal value of UCT and regret at the root node with $\alpha = 1.0$ (MENTS), 1.5, 2.0 (TENTS), 4.0, 8.0, 16.0. Figure 12 shows the convergence of the value estimate and regret at the root node of α-divergence in Synthetic Tree environment. It shows that the error of the value estimate at the root node with respect to the optimal UCT value and the regularized value decrease when α increase which matches our theoretically results in Theorem 14. Regarding the regret, the performance is different depend on different settings of branching factor k and depth d, which illustrate that the value of α helps to trade off between exploration and exploitation depending on each environment. For example, with $k = 16, d = 2$, the regret is smaller when we increase

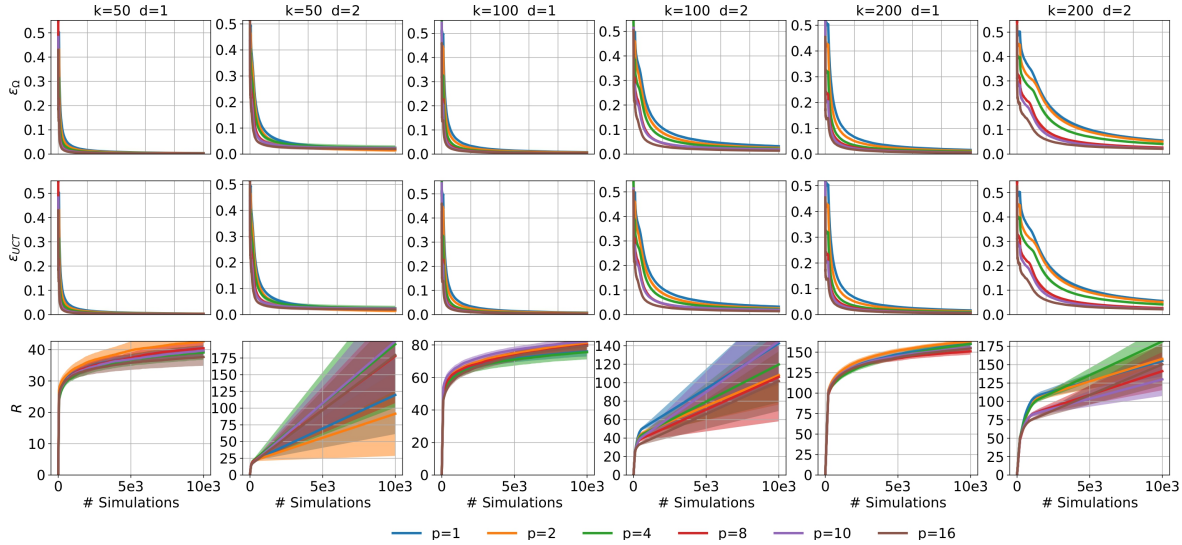


Figure 11: We show the convergence of the value estimate at the root node to the respective optimal value (top), to the UCT optimal value (middle), and the regret (bottom) with different p parameter of Power-UCT in high branching factor Synthetic tree environments with $p = 1.0$(UCT), 2.0, 4.0, 6.0, 8.0, 10.0, 16.0.

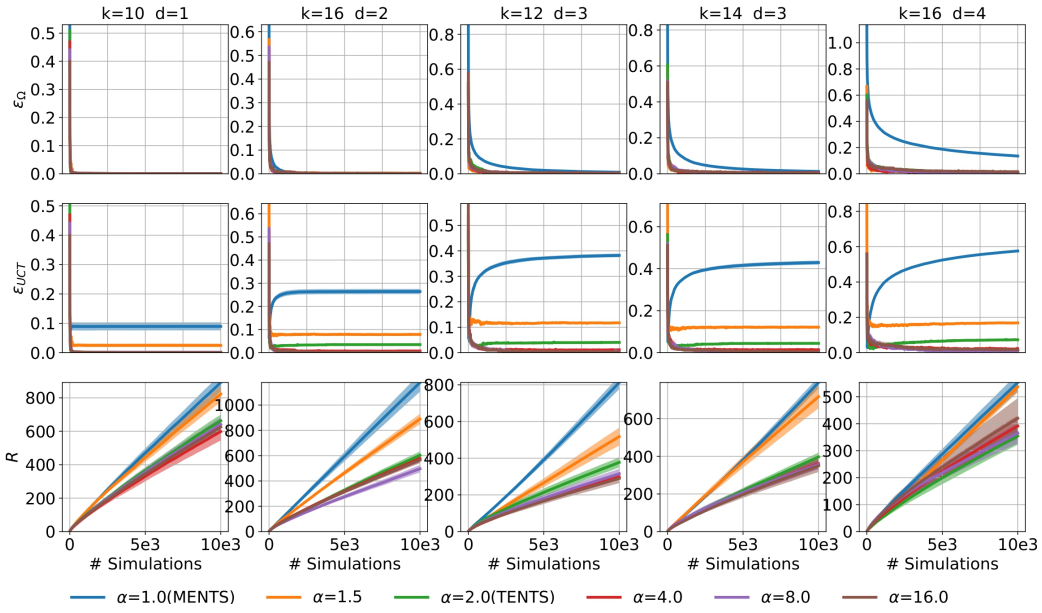


Figure 12: We show the convergence of the value estimate at the root node to the respective optimal value (top), to the UCT optimal value (middle), and the regret (bottom) with different α parameter of α-divergence in Synthetic tree environment with $\alpha = 1.0$ (MENTS), 1.5, 2.0 (TENTS), 4.0, 8.0, 16.0.

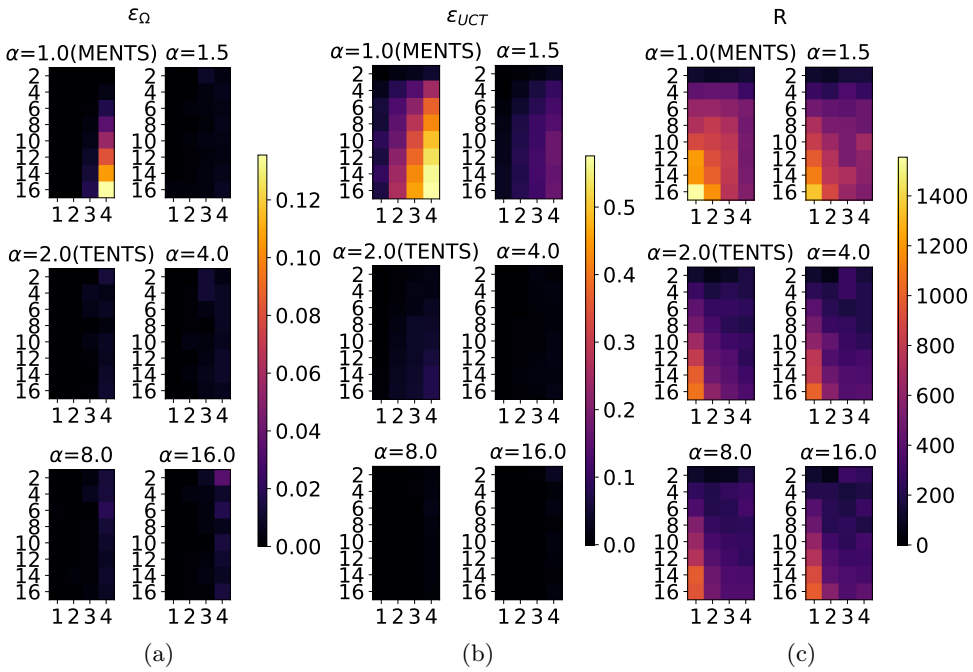


Figure 13: We show the effectiveness of α-divergence in Synthetic Tree environment with different branching factor k (rows) and depth d (columns). The heatmaps show: the absolute error of the value estimate at the root node after the last simulation of each algorithm w.r.t. the respective optimal value (a), and w.r.t. the optimal value of UCT (b); regret at the root node (c).

the value of α, and the regret is smallest with $\alpha = 8.0$. When $k = 14, d = 3$, the regret is smaller when we increase the value of α and the regret performance is the best with $\alpha = 16.0$, and when $k = 16, d = 4$, the regret enjoys the best performance with $\alpha = 2.0$ (TENTS)

8. Discusstion

Based on the above results across diverse environments, we can identify specific scenarios where each method excels and provide guidelines for selecting the appropriate algorithm depending on the domain characteristics.

Power-UCT vs. UCT: Power-UCT consistently outperforms the standard UCT algorithm in both MDP and POMDP settings. This superiority is particularly noticeable as the complexity of the environment increases, such as in high branching factor scenarios (e.g., Copy environment) and partially observable domains (e.g., Rocksample, PocMan). The power mean backup operator used in Power-UCT helps alleviate the bias problem as the average mean underestimates the optimal value, and the maximum backup operators overestimate it, allowing Power-UCT to adapt better to different reward structures and decision-making complexities.

Effect of the p-Value in Power-UCT: The choice of the p-value in Power-UCT is crucial and should be carefully tuned according to the specific characteristics of the problem domain. The results show that the optimal p-value can vary significantly depending on the task.

Entropy-Based Regularizers (MENTS, RENTS, TENTS) in MCTS: Entropy-based regularizers offer distinct advantages in environments where balancing exploration and exploitation is challenging. TENTS, in particular, demonstrates strong performance in both synthetic and real-world domains, such as Atari games. It is shown to converge quickly, especially in high branching factor problems, by maintaining a fine balance between exploration and exploitation. On the other hand, MENTS and RENTS exhibit more variance in performance and may be less consistent across different problem settings.

α-divergence Regularizers in MCTS: The use of α-divergence in MCTS provides a versatile mechanism to balance exploration and exploitation based on the specific characteristics of the environment. In the Synthetic Tree experiments, varying the α parameter revealed crucial insights into how this method adjusts the behavior of MCTS. The results showed that adjusting α allows for fine-tuning the level of exploration versus exploitation, directly impacting the accuracy of value estimates and regret minimization at the root node.

Domain Characteristics and Algorithm Selection: The properties of the environment play a significant role in determining the effectiveness of each algorithm:

High Branching Factor: In environments like the Synthetic Tree and certain Atari games, methods that prioritize efficient exploitation, such as TENTS or Power-UCT, are more effective. In addition, using a larger α value encourages broader exploration, helping to cover more action possibilities. This prevents premature convergence to suboptimal paths and ensures a thorough search of the space.

Stochasticity and Partial Observability: In POMDPs, where the environment is uncertain and only partially observable (e.g., Rocksample, PocMan), Power-UCT with an appropriately tuned p-value offers significant advantages. Additionally, incorporating entropy regularization (TENTS) can further improve robustness in these scenarios.

Reward Structure: Domains with sparse or highly variable rewards benefit from Power-UCT's ability to adjust the backup operator, allowing the algorithm to focus on promising branches without being misled by noise. Environments like FrozenLake and Copy show how adjusting the backup strategy helps in stabilizing performance.

Final Recommendation: The choice of algorithm should be guided by the specific characteristics of the task at hand. Power-UCT, with its tunable p-value, is recommended for most scenarios, especially where exploration needs to be carefully balanced with exploitation. TENTS stands out as a robust regularizer, particularly in complex, high branching factor domains. For tasks with simpler dynamics, standard UCT may still suffice, but Power-UCT and entropy-regularized methods generally offer superior performance and adaptability across a broader range of challenges.

Furthermore, the empirical findings highlight the crucial role of α-divergence in tuning MCTS algorithms. By adjusting α, one can navigate the trade-offs between exploration and exploitation, thereby impacting the accuracy of value estimates and regret. Selecting the appropriate α value is context-dependent, with higher values favoring extensive exploration in high-branching environments, while moderate values (like TENTS) are effective for balancing exploration and exploitation in typical scenarios. These insights provide valu-

able guidelines for practitioners applying MCTS in various domains, enabling more effective search strategies tailored to the specific challenges of the environment.

9. Conclusion

Our research presented three main contributions. First, we proposed a novel backup operator in MCTS called Power-UCT, which is derived from power mean with theoretical convergence guarantee to the optimum and was shown to outperform other baselines in experiments on highly complex MDPs and POMDPs.

Second, we developed a theory of convex regularization in MCTS using the Legendre-Fenchel transform framework, which proved that a generic strongly convex regularizer converges exponentially to the optimal action at the root node. Additionally, our results provided theoretical support for previous findings on maximum entropy regularization, and we conducted the first study of MCTS regret using a strongly convex regularizer. Our empirical results on AlphaGo demonstrated the superiority of Tsallis entropy over other entropy-regularizers in convex regularization.

Finally, we present a new perspective on the use of α-divergence in Monte-Carlo Tree Search (MCTS). We establish a connection between Power-UCT and convex regularization in MCTS by demonstrating that the Power Mean backup operator in Power-UCT can be obtained as a solution using an α-function as the probabilistic distance, which replaces the Euclidean distance used to calculate the average mean. The generalized power mean is the closed-form solution. Additionally, we show that entropic regularization in MCTS can be derived using α-function regularization. We analyze the regret bound of Power-UCT and E3W with respect to the α parameter, and we provide an error-bound analysis between the regularized value estimate and the optimal regularized value at the root node. Finally, our empirical results in a synthetic tree experiment demonstrate the effective balance between exploration and exploitation of α-divergence in MCTS with different values of α.

While the presented methods demonstrate strong performance across various domains, further research is necessary to explore their limitations and potential improvements. For instance, combining Power-UCT with entropy-based regularizers might offer even greater flexibility and robustness. Additionally, extending these methods to more complex real-world tasks or integrating them with learning-based approaches like MuZero could unlock new levels of performance.

Acknowledgments

This project has received funding from the German Research Foundation project PA 3179/1-1 (ROBOLEAP), and the cluster project 'The Third Wave of Artificial Intelligence – 3AI' funded by the Ministry for Science and Arts of the state of Hessen, Germany.

Appendix A. Generalized Mean Estimation in Monte-Carlo Tree Search

We derive here the proof of convergence for Power-UCT. The proof is based on the proof of the UCT Kocsis et al. (2006) method but differs in several key places. In this section, we show that Power-UCT can smoothly adapt to all theorems of UCT Kocsis et al. (2006).

The following results can be seen as a generalization of the results for UCT, as we consider a generalized mean instead of a standard mean as the backup operator. Our main results are Theorem 6 and Theorem 7, which respectively prove the convergence of failure probability at the root node, and derive the bias of power mean estimated payoff. In order to prove them, we start with Theorem 1 to show the concentration of power mean with respect to i.i.d random variable X. Subsequently, Theorem 2 shows the upper bound of the expected number of times when a suboptimal arm is played. Theorem 3 bounds the expected error of the power mean estimation. Theorem 4 shows the lower bound of the number of times any arm is played. Theorem 5 shows the concentration of power mean backup around its mean value at each node in the tree.
We start with well-known lemmas and respective proofs: The following lemma shows that Power Mean can be upper bound by Average Mean plus with a constant

Lemma 2. *Let* $0 < l \leq X_i \leq U, C = \frac{U}{l}, \forall i \in (1, ..., n)$ *and* $p > q$. *We define*

$$Q(X, w, p, q) = \frac{M_n^{[p]}(X, w)}{M_n^{[q]}(X, w)}$$
$$D(X, w, p, q) = M_n^{[p]}(X, w) - M_n^{[q]}(X, w).$$

Then we have

$$Q(X, w, p, q) \leq L_{p,q}$$
$$D(X, w, p, q) \leq H_{p,q}$$
$$L_{p,q} = \left(\frac{q(C^p - C^q)}{(p-q)(C^q - 1)}\right)^{\frac{1}{p}} \left(\frac{p(C^q - C^p)}{(q-p)(C^p - 1)}\right)^{-\frac{1}{q}}$$
$$H_{p,q} = (\theta U^p + (1-\theta)l^p)^{\frac{1}{p}} - (\theta U^q + (1-\theta)l^q)^{1/q},$$

where θ *is defined in the following way. Let*

$$h(x) = x^{\frac{1}{p}} - (ax + b)^{1/q}$$

where

$$a = \frac{U^q - l^q}{U^p - l^p}$$
$$b = \frac{U^p l^q - U^q l^p}{U^p - l^p}$$
$$x^{'} = \arg\max\{h(x), x \in (l^p, U^p)\}$$

then

$$\theta = \frac{x' - l^p}{U^p - l^p}.$$

Proof. Refer to Mitrinovic and Vasic (1970). □

Lemma 3. *Let X be an independent random variable with common mean μ and $a \leq X \leq b$. Then for any t*

$$\mathbb{E}[\exp(tX)] \leq \exp\left(t\mu + t^2 \frac{(b-a)^2}{8}\right) \tag{54}$$

Proof. Refer to Wasserman (2004) page 67. □

Lemma 4. *Chernoff's inequality $t > 0$,*

$$\Pr(X > \epsilon) \leq \exp(-t\epsilon)\mathbb{E}[\exp(tX)] \tag{55}$$

Proof. This is a well-known result. □

The next result show the concentration Inequality of Power Mean estimation

Theorem 1. *If $X_1, X_2, ..., X_n$ are independent with $\Pr(0 \leq X_i \leq 1) = 1$ then for any $\epsilon > 0$, $p \geq 1, \exists C_p > 0$ that*

$$\Pr\left(\left|\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}} - \mathbb{E}\left[\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}}\right]\right| > \epsilon\right) \leq 2\exp\left(-C_p n\epsilon^2\right)$$

Proof. Apply the Theorem 6 Francois et al. (2007), we have

$$\frac{\sqrt{\mathrm{Var}(\parallel X \parallel_p)}}{\mathbb{E}(\parallel X \parallel_p)} \simeq \frac{1}{\sqrt{n}}\left(\frac{1}{p}\frac{\sigma_p}{\mu_p}\right) \tag{56}$$

where $\parallel X \parallel_p$ is p-norm. $\mu_p = \mathbb{E}[X^p], \sigma_p^2 = \mathrm{Var}(X^p)$
Then

$$\mathrm{Var}\left(\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}}\right) \simeq \frac{1}{n}\left(\frac{1}{p}\frac{\sigma_p}{\mu_p}\mathbb{E}\left[\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}}\right]\right)^2 \tag{57}$$

Let define $\frac{1}{C_p} = \left(\frac{1}{p}\frac{\sigma_p}{\mu_p}\mathbb{E}\left[\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}}\right]\right)^2$ We have

$$\mathrm{Var}\left(\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}}\right) \simeq \frac{1}{nC_p} \tag{58}$$

With $\sigma^2 = \mathrm{Var}(X)$, Applying Lemma 3 and Lemma 4 we have

$$\Pr(|X - \mathbb{E}[X]| > \epsilon) \leq 2\exp\left(-\frac{\epsilon^2}{\sigma^2}\right) \tag{59}$$

Apply (59) for the power mean of $X_1, X_2, ..., X_n$, and based on the result of 58, we get the result of Theorem 1. □

With $\triangle_n = 9\sqrt{\frac{1}{C_p} n \log(\frac{2}{\delta})}$, ($\delta > 0$ are constant), and $\mu_p = \mathbb{E}\left[\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}}\right]$, apply Theorem 1, we get

$$\Pr\left(\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}} - \mu_p > \frac{\triangle_n}{9n}\right) \leq \exp\left(-C_p n \left(\frac{\triangle_n}{9n}\right)^2\right).$$

Therefore,

$$\Pr\left(\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}} - \mu_p > \frac{\triangle_n}{9n}\right) \leq \exp\left(-C_p \frac{1}{C_p} \log\left(\frac{2}{\delta}\right)\right) = \frac{\delta}{2}. \tag{60}$$

we have

$$\Pr\left(\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}} - \mu_p > \frac{\triangle_n}{9n}\right) \leq \frac{\delta}{2}. \tag{61}$$

Let's start with an assumption

Assumption 1. *Fix $1 \leq i \leq K$. Let $\{F_{it}\}_t$ be a filtration such that$\{X_{it}\}_t$ is $\{F_{it}\}$-adapted and $X_{i,t}$ is conditionally independent of $F_{i,t+1}, F_{i,t+2}, ...$ given $F_{i,t-1}$. Then $0 \leq X_{it} \leq 1$ and the limit of $\mu_{in} = \mathbb{E}[\overline{X_{in}}(p)]$ exists, Further, we assume that there exists a constant $C > 0$ and an integer N_c such that for $n > N_c$, for any $\delta > 0$, $\triangle_n(\delta) = C\sqrt{n \log(1/\delta)}$, the following bounds hold:*

$$\Pr(\overline{X}_{in}(p) \geq \mathbb{E}[\overline{X}_{in}(p)] + \triangle_n(\delta)/n) \leq \delta, \tag{62}$$

$$\Pr(\overline{X}_{in}(p) \leq \mathbb{E}[\overline{X}_{in}(p)] - \triangle_n(\delta)/n) \leq \delta. \tag{63}$$

Under Assumption 1, For any internal node arm k, at time step t, let define $\mu_{kt} = \mathbb{E}[\overline{X}_{kt}(p)]$, a suitable choice for bias sequence is that $c_{t,s} = 2C\sqrt{\frac{\log t}{s}}$ (C is an exploration constant) used in UCB1 (using power mean estimator), we get

$$\Pr\left(\left(\frac{\sum_{i=1}^s X_{ki}^p}{s}\right)^{\frac{1}{p}} - \mu_{kt} > 2C\sqrt{\frac{\log t}{s}}\right) \leq t^{-4} \tag{64}$$

$$\Pr\left(\left(\frac{\sum_{i=1}^s X_{ki}^p}{s}\right)^{\frac{1}{p}} - \mu_{kt} < -2C\sqrt{\frac{\log t}{s}}\right) \leq t^{-4}. \tag{65}$$

From Assumption 1, we derive the upper bound for the expectation of the number of plays a sub-optimal arm

Theorem 2. *Consider UCB1 (using power mean estimator) applied to a non-stationary problem where the pay-off sequence satisfies Assumption 1 and where the bias sequence, $c_{t,s} = 2C\sqrt{\log t/s}$ (C is an exploration constant). Fix $\epsilon \geq 0$. Let $T_k(n)$ denote the number of plays of arm k. Then if k is the index of a suboptimal arm then Each sub-optimal arm k is played in expectation at most*

$$\mathbb{E}[T_k(n)] \leq \frac{16C^2 \ln n}{(1-\epsilon)^2 \triangle_k^2} + A(\epsilon) + N_c + \frac{\pi^2}{3} + 1. \tag{66}$$

Proof. When a sub-optimal arm k is pulled at time t we get

$$\left(\frac{\sum_{i=1}^{T_k(t-1)} X_{k,i}^p}{T_k(t-1)}\right)^{\frac{1}{p}} + 2C\sqrt{\frac{\ln t}{T_k(t-1)}} \geq \left(\frac{\sum_{i=1}^{T_{k^*}(t-1)} X_{k^*,i}^p}{T_{k^*}(t-1)}\right)^{\frac{1}{p}} + 2C\sqrt{\frac{\ln t}{T_{k^*}(t-1)}} \tag{67}$$

Now, consider the following two inequalities:

- The empirical mean of the optimal arm is not within its confidence interval

$$\left(\frac{\sum_{i=1}^{T_{k^*}(t-1)} X_{k^*,i}^p}{T_{k^*}(t-1)}\right)^{\frac{1}{p}} + 2C\sqrt{\frac{\ln t}{T_{k^*}(t-1)}} \leq \mu_t^* \tag{68}$$

- The empirical mean of the arm k is not within its confidence interval

$$\left(\frac{\sum_{i=1}^{T_k(t-1)} X_{k,i}^p}{T_k(t-1)}\right)^{\frac{1}{p}} \geq \mu_{kt} + 2C\sqrt{\frac{\ln t}{T_k(t-1)}} \tag{69}$$

If both previous inequalities (68), (69) do not hold, and if a sub-optimal arm k is pulled, then we deduce that

$$\mu_{kt} + 2C\sqrt{\frac{\ln t}{T_k(t-1)}} \geq \left(\frac{\sum_{i=1}^{T_k(t-1)} X_{k,i}^p}{T_k(t-1)}\right)^{\frac{1}{p}} \text{ see (69)} \tag{70}$$

and

$$\left(\frac{\sum_{i=1}^{T_k(t-1)} X_{k,i}^p}{T_k(t-1)}\right)^{\frac{1}{p}} \geq \left(\frac{\sum_{i=1}^{T_{k^*}(t-1)} X_{k^*,i}^p}{T_{k^*}(t-1)}\right)^{\frac{1}{p}} + 2C\sqrt{\frac{\ln t}{T_{k^*}(t-1)}} - 2C\sqrt{\frac{\ln t}{T_k(t-1)}} \text{ see (67)} \tag{71}$$

and

$$\left(\frac{\sum_{i=1}^{T_{k^*}(t-1)} X_{k^*,i}^p}{T_{k^*}(t-1)}\right)^{\frac{1}{p}} + 2C\sqrt{\frac{\ln t}{T_{k^*}(t-1)}} \geq \mu_t^* \text{ see (68).} \tag{72}$$

So that

$$\mu_{kt} + 4C\sqrt{\frac{\ln t}{T_k(t-1)}} \geq \mu_t^*. \tag{73}$$

$\mu_{kt} = \mu_k + \delta_{kt}$, $\mu_t^* = \mu^* + \delta_t^*$ and we have an assumption that $\lim_{t\to\infty} \mu_{kt} = \mu_k$ for any $k \in [1, 2, ...K]$ yields $\lim_{t\to\infty} \delta_{kt} = 0$ Therefore, for any $\epsilon > 0$, we can find an index $A(\epsilon)$ such that for any $t > A(\epsilon)$: $\delta_{kt} \leq \epsilon\triangle_k$ with $\triangle_k = \mu^* - \mu_k$. Which means that

$$4C\sqrt{\frac{\ln t}{T_k(t-1)}} \geq \triangle_k - \delta_{kt} + \delta_t^* \geq (1-\epsilon)\triangle_k \tag{74}$$

which implies $T_k(t-1) \leq \frac{16C^2 \ln t}{(1-\epsilon)^2 \triangle_k^2}$.
This says that whenever $T_k(t-1) \geq \frac{16C^2 \ln t}{(1-\epsilon)^2 \triangle_k^2} + A(\epsilon) + N_c$, either arm k is not pulled at time t or one of the two following events (68), (69) holds. Thus if we define $u = \frac{16C^2 \ln t}{(1-\epsilon)^2 \triangle_k^2} + A(\epsilon) + N_c$, we have

$$\begin{aligned} T_k(n) &\leq u + \sum_{t=u+1}^{n} \mathbf{1}\{I_t = k; T_k(t) \geq u\} \\ &\leq u + \sum_{t=u+1}^{n} \mathbf{1}\{(68), \text{ or } (69) \text{ holds } \} \end{aligned}$$

We have from (64),(65)

$$\Pr\left(\Big(\frac{\sum_{i=1}^{T_{k^*}(t-1)} X_{k^*,i}^p}{T_{k^*}(t-1)} \Big)^{\frac{1}{p}} + 2C\sqrt{\frac{\ln t}{T_{k^*}(t-1)}} \leq \mu_t^* \right) \leq \sum_{s=1}^{t} \frac{1}{t^4} = \frac{1}{t^3} \tag{75}$$

and

$$\Pr\left(\Big(\frac{\sum_{i=1}^{T_k(t-1)} X_{k,i}^p}{T_k(t-1)} \Big)^{\frac{1}{p}} \geq \mu_{kt} + 2C\sqrt{\frac{\ln t}{T_k(t-1)}} \right) \leq \sum_{s=1}^{t} \frac{1}{t^4} = \frac{1}{t^3} \tag{76}$$

so that from (75), we have

$$\begin{aligned} \mathbb{E}[T_k(n)] &\leq \frac{16C^2 \ln t}{(1-\epsilon)^2 \triangle_k^2} + A(\epsilon) + N_c + \sum_{t=u+1}^{n} \frac{2}{t^{8C^2-1}} = \frac{16C^2 \ln t}{(1-\epsilon)^2 \triangle_k^2} + A(\epsilon) + N_c \\ &\quad + \sum_{t=u+1}^{n} \frac{2}{t^3} \\ &\leq \frac{16C^2 \ln t}{(1-\epsilon)^2 \triangle_k^2} + A(\epsilon) + N_c + \frac{\pi^2}{3} \end{aligned}$$

□

Based on this result we derive an upper bound for the expectation of power mean in the next theorem as follows.

Theorem 3. *Under the assumptions of Theorem 2,*

$$\left| \mathbb{E}\left[\overline{X}_n(p)\right] - \mu^* \right| \leq |\delta_n^*| + \mathcal{O}\left(\frac{K(C^2 \log n + N_0)}{n} \right)^{\frac{1}{p}}.$$

Proof. In UCT, the value of each node is used for backup as $\overline{X}_n = \sum_{i=1}^{K} \left(\frac{T_i(n)}{n} \right) \overline{X}_{i,T_i(n)}$, and the authors show that

$$\begin{aligned} &\left| \mathbb{E}\left[\overline{X}_n\right] - \mu^* \right| \leq \left| \mathbb{E}\left[\overline{X}_n\right] - \mu_n^* \right| + \left| \mu_n^* - \mu^* \right| \\ &= \left| \delta_n^* \right| + \left| \mathbb{E}\left[\overline{X}_n\right] - \mu_n^* \right| \\ &\leq \left| \delta_n^* \right| + \mathcal{O}\left(\frac{K(C^2 \log n + N_0)}{n} \right) \end{aligned} \tag{77}$$

We derive the same results replacing the average with the power mean. First, we have

$$\mathbb{E}\left[\overline{X}_n(p)\right] - \mu_n^* = \mathbb{E}\left[\left(\sum_{i=1}^{K} \frac{T_i(n)}{n} \overline{X}^p_{i,T_i(n)}\right)^{\frac{1}{p}}\right] - \mu_n^*. \tag{78}$$

In the proof, we will make use of the following inequalities:

$$0 \leq X_i \leq 1, \tag{79}$$

$$x^{\frac{1}{p}} \leq y^{\frac{1}{p}} \text{ when } 0 \ \leq x \leq y, \tag{80}$$

$$(x+y)^m \leq x^m + y^m (0 \leq m \leq 1), \tag{81}$$

$$\mathbb{E}[f(X)] \leq f(\mathbb{E}[X]) \ \textit{(f(X) is concave)}. \tag{82}$$

With i^* being the index of the optimal arm, we can derive an upper bound on the difference between the value backup and the true average reward

$$\begin{aligned}
&\mathbb{E}\left[\left(\sum_{i=1}^{K} \frac{T_i(n)}{n} \overline{X}^p_{i,T_i(n)}\right)^{\frac{1}{p}}\right] - \mu_n^* \\
&\leq \mathbb{E}\left[\left(\left(\sum_{i=1;i\neq i^*}^{K} \frac{T_i(n)}{n}\right) + \overline{X}^p_{i*,T_i*(n)}\right)^{\frac{1}{p}}\right] - \mu_n^* (\text{see (79)}) \\
&\leq \mathbb{E}\left[\left(\sum_{i=1;i\neq i^*}^{K} \frac{T_i(n)}{n}\right)^{\frac{1}{p}} + \overline{X}_{i*,T_i*(n)}\right] - \mu_n^* (\text{see (81)}) \\
&= \mathbb{E}\left[\left(\sum_{i=1;i\neq i^*}^{K} \frac{T_i(n)}{n}\right)^{\frac{1}{p}}\right] + \mathbb{E}\left[\overline{X}_{i*,T_i*(n)}\right] - \mu_n^* \\
&= \mathbb{E}\left[\left(\sum_{i=1;i\neq i^*}^{K} \frac{T_i(n)}{n}\right)^{\frac{1}{p}}\right] \\
&\leq \left(\sum_{i=1;i\neq i^*}^{K} \mathbb{E}\left[\frac{T_i(n)}{n}\right]\right)^{\frac{1}{p}} (\text{see (82)}) \\
&\leq ((K-1)\mathcal{O}\left(\frac{K(C^2 \log n + N_0)}{n}\right))^{\frac{1}{p}} (\text{Theorem 2 \& (80)})
\end{aligned} \tag{83}$$

According to Lemma 1, it holds that

$$\mathbb{E}\left[\overline{X}_n(p)\right] \geq \mathbb{E}\left[\overline{X}_n\right]$$

for $p \geq 1$. Because of this, we can reuse the lower bound given by (77)

$$-\mathcal{O}\left(\frac{K(C^2 \log n + N_0)}{n}\right) \leq \mathbb{E}\left[\overline{X}_n\right] - \mu_n^*,$$

so that

$$-\mathcal{O}\left(\frac{K(C^2\log n+N_0)}{n}\right)\leq \mathbb{E}\big[\overline{X}_n\big]-\mu_n^*$$
$$\leq \mathbb{E}\big[\overline{X}_n(p)\big]-\mu_n^*. \tag{84}$$

Combining (83) and (84) concludes our prove

$$\big|\mathbb{E}\big[\overline{X}_n(p)\big]-\mu^*\big|\leq|\delta_n^*|+O\left(\frac{K(C^2\log n+N_0)}{n}\right)^{\frac{1}{p}}.$$

□

The following theorem shows lower bound of choosing all the arms

Theorem 4. ***(Lower Bound)*** *Under the assumptions of Theorem 2, there exists some positive constant ρ such that for all arms k and n, $T_k(n)\geq\lceil\rho\log(n)\rceil$*

Proof. There should exist a constant S that

$$\left(\frac{\sum_{i=1}^{T_k(t-1)}X_{k,i}^p}{T_k(t-1)}\right)^{\frac{1}{p}}+2C\sqrt{\frac{\ln t}{T_k(t-1)}}\leq S$$

for all arm k so

$$\mu_k+\delta_{kt}+2C\sqrt{\frac{\log t}{T_k(t-1)}}\leq S$$

because

$$\lim_{t\to\infty}\delta_{kt}=0$$

so there exists a positive constant ρ that $T_k(t-1)\geq\lceil\rho\log(t)\rceil$ □

The next result shows the estimated optimal payoff concentration around its mean (Theorem 5). In order to prove that, we now reproduce here Lemma 5, 6 Kocsis et al. (2006) that we use for our proof:

Lemma 5. ***Hoeffding-Azuma inequality for Stopped Martingales*** *(Lemma 10 in Kocsis et al. (2006)). Assume that S_t is a centered martingale such that the corresponding martingale difference process is uniformly bounded by C. Then, for any fixed $\epsilon\geq 0$, integers $0\leq a\leq b$, the following inequalities hold*

$$\Pr(S_N\geq\epsilon N)\leq(b-a+1)\exp\left(\frac{-2a^2\epsilon^2}{C^2}\right)+\Pr(N\notin[a,b]), \tag{85}$$

$$\Pr(S_N\leq\epsilon N)\leq(b-a+1)\exp\left(\frac{-2a^2\epsilon^2}{C^2}\right)+\Pr(N\notin[a,b]), \tag{86}$$

Lemma 6. *(Lemma 13 in Kocsis et al. (2006)) Let (Z_i), i=1,...,n be a sequence of random variables such that Z_i is conditionally independent of $Z_{i+1}, ..., Z_n$ given $Z_1, ..., Z_{i-1}$. Let define $N_n = \sum_{i=1}^{n} Z_i$, and let a_n is an upper bound on $\mathbb{E}[N_n]$. Then for all $\triangle \geq 0$, if n is such that $a_n \leq \triangle/2$ then*

$$\Pr(N_n \geq \triangle) \leq \exp(-\triangle^2/(8n)). \tag{87}$$

The next lemma is our core result for propagating confidence bounds upward in the tree, and it is used for the prove of Theorem 5 about the concentration of power mean estimator.

Lemma 7. *let Z_i, a_i be as in Lemma 6. Let F_i denotes a filtration over some probability space. Y_i be an F_i-adapted real valued martingale-difference sequence. Let X_i be an i.i.d. sequence with mean μ. We assume that both X_i and Y_i lie in the [0,1] interval. Consider the partial sums*

$$S_n = \left(\frac{\sum_{i=1}^{n} (1 - Z_i) X_i^p + Z_i Y_i^p}{n} \right)^{\frac{1}{p}}. \tag{88}$$

Fix an arbitrary $\delta > 0$, and fix $p \geq 1$. Let $\triangle_n = 9\sqrt{\frac{1}{C_p} n \log(2/\delta)}$, and $\triangle = (9/4)^{p-1} \triangle_n$ let

$$R_n = \mathbb{E}\left[\left(\frac{\sum_{i=1}^{n} X_i^p}{n} \right)^{\frac{1}{p}} \right] - \mathbb{E}[S_n]. \tag{89}$$

Then for n such that $a_n \leq (1/9)\triangle_n$ and $|R_n| \leq (4/9)(\triangle/n)^{\frac{1}{p}}$

$$\Pr(S_n \geq \mathbb{E}[S_n] + (\triangle/n)^{\frac{1}{p}}) \leq \delta \tag{90}$$

$$\Pr(S_n \leq \mathbb{E}[S_n] - (\triangle/n)^{\frac{1}{p}}) \leq \delta \tag{91}$$

Proof. We have a very fundamental probability inequality:
Consider two events: A, B. If $A \in B$, then $\Pr(A) \leq \Pr(B)$.
Therefore, if we have three random variables X, Y, Z and if we are sure that

$$Y \geq Z, \text{ then } \Pr(X \geq Y) \leq \Pr(X \geq Z) \tag{92}$$

We have

$$\begin{aligned}
&\left(\frac{\sum_{i=1}^{n} (1 - Z_i) X_i^p + Z_i Y_i^p}{n} \right)^{\frac{1}{p}} \\
&= \left(\frac{\sum_{i=1}^{n} X_i^p}{n} + \frac{Z_i (Y_i^p - X_i^p)}{n} \right)^{\frac{1}{p}} \leq \left(\frac{\sum_{i=1}^{n} X_i^p}{n} + \frac{2\sum_{i=1}^{n} Z_i}{n} \right)^{\frac{1}{p}} (X_i, Y_i \in [0,1]) \\
&\leq \left(\frac{\sum_{i=1}^{n} X_i^p}{n} \right)^{\frac{1}{p}} + \left(\frac{2\sum_{i=1}^{n} Z_i}{n} \right)^{\frac{1}{p}} (\text{see (80)})
\end{aligned} \tag{93}$$

Therefore,

$$T = \Pr\left(S_n \geq \mathbb{E}[S_n] + (\triangle/n)^{\frac{1}{p}}\right) \tag{94}$$

$$= \Pr\left(\Big(\frac{\sum_{i=1}^n (1-Z_i)X_i^p + Z_i Y_i^p}{n}\Big)^{\frac{1}{p}} \geq \mathbb{E}[\Big(\frac{\sum_{i=1}^n X_i^p}{n}\Big)^{\frac{1}{p}}] - R_n + (\triangle/n)^{\frac{1}{p}}\right) \text{(see (89))} \tag{95}$$

$$\leq \Pr\left(\Big(\frac{\sum_{i=1}^n X_i^p}{n}\Big)^{\frac{1}{p}} + \Big(\frac{2\sum_{i=1}^n Z_i}{n}\Big)^{\frac{1}{p}} \geq \mathbb{E}\left[\Big(\frac{\sum_{i=1}^n X_i^p}{n}\Big)^{\frac{1}{p}}\right] - R_n + (\triangle/n)^{\frac{1}{p}}\right)$$
$$\text{(see (92), (93)))} \tag{96}$$

Using the elementary inequality $I(A + B \geq \triangle/n) \leq I(A \geq \alpha\triangle/n) + I(B \geq (1-\alpha)\triangle/n)$ that holds for any $A, B \geq 0; 0 \leq \alpha \leq 1$, we get

$$\begin{aligned} T \leq & \Pr\left(\Big(\frac{\sum_{i=1}^n X_i^p}{n}\Big)^{\frac{1}{p}} \geq \mathbb{E}\left[\Big(\frac{\sum_{i=1}^n X_i^p}{n}\Big)^{\frac{1}{p}}\right] + 1/9(\triangle/n)^{\frac{1}{p}}\right) \\ & + \Pr\left(\Big(\frac{2\sum_{i=1}^n Z_i}{n}\Big)^{\frac{1}{p}} \geq 8/9(\triangle/n)^{\frac{1}{p}} - R_n\right) \end{aligned} \tag{97}$$

Define $\mu_p = \mathbb{E}\left[\Big(\frac{\sum_{i=1}^n X_i^p}{n}\Big)^{\frac{1}{p}}\right]$, we have

$$\leq \Pr\left(\Big(\frac{\sum_{i=1}^n X_i^p}{n}\Big)^{\frac{1}{p}} \geq \mu_p + 1/9(\triangle/n)^{\frac{1}{p}}\right) + \Pr\left(\Big(\frac{2\sum_{i=1}^n Z_i}{n}\Big)^{\frac{1}{p}} \geq 4/9(\triangle/n)^{\frac{1}{p}}\right)$$
$$(\text{ see } R_n \leq (4/9)(\triangle/n)^{\frac{1}{p}}) \tag{98}$$

$$= \Pr\left(\Big(\frac{\sum_{i=1}^n X_i^p}{n}\Big)^{\frac{1}{p}} \geq \mu_p + \frac{1}{9}\frac{9}{4}(\frac{4}{9}\triangle_n/n)^{\frac{1}{p}}\right) + \Pr\left(\Big(\frac{2\sum_{i=1}^n Z_i}{n}\Big)^{\frac{1}{p}} \geq (\frac{(4/9)^p\triangle}{n})^{\frac{1}{p}}\right)$$
$$(\text{ definition of } \triangle) \tag{99}$$

$$\leq \Pr\left(\Big(\frac{\sum_{i=1}^n X_i^p}{n}\Big)^{\frac{1}{p}} \geq \mu_p + \triangle_n/9n\right) + \Pr\left(\Big(\frac{\sum_{i=1}^n Z_i}{n}\Big) \geq 2\triangle_n/9n\right)$$
$$\text{(see (80) and } f(x) = a^x \text{ is decrease when } a < 1) \tag{100}$$

The first term is bounded by $\delta/2$ according to (61) and the second term is bounded by $\delta/2$ according to Lemma 6 (the condition of Lemma 6 is satisfied because $a_n \leq (1/9)\triangle_n$). This finishes the proof of the first part (90). The second part (91) can be proved in an analogous manner. □

Theorem 5. *Fix an arbitrary $\delta \leq 0$ and fix $p \geq 1$, let $\triangle_n = (\frac{9}{4})^{p-1}(9\sqrt{\frac{1}{C_p} n \log(2/\delta)})$. Let n_0 be such that*

$$\sqrt{n_0} \leq \mathcal{O}(K(C^2 \log n_0 + N_0(1/2))). \tag{101}$$

Then for any $n \geq n_0$, under the assumptions of Theorem 2, the following bounds hold true

$$\Pr(\overline{X}_n(p) \geq \mathbb{E}[\overline{X}_n(p)] + (\triangle_n/n)^{\frac{1}{p}}) \leq \delta \tag{102}$$

$$\Pr(\overline{X}_n(p) \leq \mathbb{E}[\overline{X}_n(p)] - (\triangle_n/n)^{\frac{1}{p}}) \leq \delta \tag{103}$$

Proof. Let X_t is the payoff sequence of the best arm. Y_t is the payoff at time t. Both X_t, Y_t lies in [0,1] interval, and $\overline{X}_n(p) = \left(\frac{\sum_{i=1}^n (1-Z_i)X_i^p + Z_i Y_i^p}{n}\right)^{\frac{1}{p}}$ Apply Lemma 6 and remember that $X^{\frac{1}{p}} - Y^{\frac{1}{p}} \leq (X-Y)^{\frac{1}{p}}$ we have

$$\begin{aligned}
R_n &= \mathbb{E}\left[\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}}\right] - \mathbb{E}\left[\left(\frac{\sum_{i=1}^n (1-Z_i)X_i^p + Z_i Y_i^p}{n}\right)^{\frac{1}{p}}\right]. \\
&= \mathbb{E}\left[\left(\frac{\sum_{i=1}^n X_i^p}{n}\right)^{\frac{1}{p}} - \left(\frac{\sum_{i=1}^n (1-Z_i)X_i^p + Z_i Y_i^p}{n}\right)^{\frac{1}{p}}\right]. \\
&\leq \mathbb{E}\left[\left(\frac{\sum_{i=1}^n X_i^p - \sum_{i=1}^n (1-Z_i)X_i^p - Z_i Y_i^p}{n}\right)^{\frac{1}{p}}\right]. \\
&= \mathbb{E}\left[\left(\frac{\sum_{i=1}^n Z_i(X_i^p - Y_i^p)}{n}\right)^{\frac{1}{p}}\right]. \\
&\leq \mathbb{E}\left[\left(\sum_{i=1}^K \frac{T_i(n)}{n}\right)^{\frac{1}{p}}\right]. \\
&\leq \left(\frac{\sum_{i=1}^K \mathbb{E}[T_i(n)]}{n}\right)^{\frac{1}{p}}. \text{ see Jensen inequality} \\
&= \left((K-1)O\left(\frac{K(C^2 \log n + N_0(1/2))}{n}\right)\right)^{\frac{1}{p}}.
\end{aligned}$$

So that let n_0 be an index such that if $n \geq n_0$ then $a_n \leq \triangle_n/9$ and $R_n \leq 4/9(\triangle_n/n)^{\frac{1}{p}}$. Such an index exists since $\triangle_n = \mathcal{O}(\sqrt{n})$ and $a_n, R_n = \mathcal{O}((\log n/n)^{\frac{1}{p}})$. Hence, for $n \geq n_0$, the conditions of lemma 6 are satisfied and the desired tail-inequalities hold for $\overline{X_n}(p)$. □

In the next theorem, we show that Power-UCT can ensure the convergence of choosing the best arm at the root node.

Theorem 6. *(**Convergence of Failure Probability**) Under the assumptions of Theorem 2, it holds that*

$$\lim_{t \to \infty} \Pr(I_t \neq i^*) = 0 \tag{104}$$

Proof. We show that Power-UCT can smoothly adapt to UCT's prove. Let i be the index of a suboptimal arm and let $p_{it} = \Pr(\overline{X}_{i,T_i(t)}(p) \geq \overline{X}^*_{T^*(t)}(p))$ from above. Clearly,

$\Pr(I_t \neq i*) \leq \sum_{i \neq i*} p_{it}$. Hence, it suffices to show that $p_{it} \leq \epsilon/K$ holds for all suboptimal arms for t sufficiently large.
Clearly, if $\overline{X}_{i,T_i(t)}(p) \leq \mu_i + \triangle_i/2$ and $\overline{X}^*_{T^*(t)}(p) \geq \mu^* - \triangle_i/2$ then $\overline{X}_{i,T_i(t)}(p) < \overline{X}^*_{T^*(t)}(p)$. Hence,

$$p_t \leq \Pr(\overline{X}_{i,T_i(t)}(p) \leq \mu_i + \triangle_i/2) + \Pr(\overline{X}^*_{T^*(t)}(p) \geq \mu^* - \triangle_i/2)$$

The first probability can be expected to be converging much slower since $T_i(t)$ converges slowly. Hence, we bound it first.
In fact,

$$\Pr(\overline{X}_{i,T_i(t)}(p) \leq \mu_i + \triangle_i/2) \leq \Pr(\overline{X}_{i,T_i(t)}(p) \leq \overline{\mu}_{i,T_i(t)} - |\delta_{i,T_i(t)}| + \triangle_i/2).$$

Without the loss of generality, we may assume that $|\delta_{i,T_i(t)}| \leq \triangle_i/4$. Therefore

$$\Pr(\overline{X}_{i,T_i(t)}(p) \leq \mu_i + \triangle_i/2) \leq \Pr(\overline{X}_{i,T_i(t)}(p) \leq \overline{\mu}_{i,T_i(t)} + \triangle_i/4).$$

Now let a be an index such that if $t \geq a$ then $(t+1)\Pr(\overline{X}_{i,T_i(t)}(p) \leq \overline{\mu}_{i,T_i(t)} + \triangle_i/4) \leq \epsilon/(2K)$. Such an index exist by our assumptions on the concentration properties of the average payoffs. Then, for $t \geq a$

$$\begin{aligned}\Pr(\overline{X}_{i,T_i(t)}(p) \leq \overline{\mu}_{i,T_i(t)} + \triangle_i/4) \leq \Pr(\overline{X}_{i,T_i(t)}(p) \leq \overline{\mu}_{i,T_i(t)} + \triangle_i/4, T_i(t) \geq a) \\ + \Pr(T_i(t) \leq a)\end{aligned}$$

Since the lower-bound on $T_i(t)$ grows to infinity as $t \to \infty$, the second term becomes zero when t is sufficiently large. The first term is bounded using the method of Lemma 5. By choosing $b = 2a$, we get

$$\begin{aligned}\Pr(\overline{X}_{i,T_i(t)}(p) \leq \overline{\mu}_{i,T_i(t)} + \triangle_i/4, T_i(t) \geq a) \leq (a+1)\Pr(\overline{X}_{i,a}(p) \leq \overline{\mu}_{i,a} + \triangle_i/4, T_i(t) \geq a) \\ + \Pr(T_i(t) \geq 2b) \leq \epsilon/(2K),\end{aligned}$$

where we have assumed that t is large enough so that $P(T_i(t) \geq 2b) = 0$.
Bounding $\Pr(\overline{X}^*_{T^*(t)}(p) \geq \mu^* - \triangle_i/2)$ by $\epsilon/(2K)$ can be done in an analogous manner. Collecting the bound yields that $p_{it} \leq \epsilon/K$ for t sufficiently large which complete the prove. □

Now is our result to show the bias of expected payoff $\overline{X_n}(p)$

Theorem 7. *Consider algorithm Power-UCT running on a game tree of depth D, branching factor K with stochastic payoff at the leaves. Assume that the payoffs lie in the interval [0,1]. Then the bias of the estimated expected payoff, $\overline{X_n}$, is $\mathcal{O}(KD(\log(n)/n)^{\frac{1}{p}} + K^D(1/n)^{\frac{1}{p}})$. Further, the failure probability at the root convergences to zero as the number of samples grows to infinity.*

Proof. The proof is done by induction on D. When $D = 1$, Power-UCT becomes UCB1 problem and as the result of Hoeffding's inequality, the convergence is guaranteed directly from Theorem 1, Theorem 3 and Theorem 6.

Now we assume that the result holds up to depth $D-1$ and consider the tree of Depth D. Running Power-UCT on root node is equivalence as UCB1 on non-stationary bandit settings. The error bound of running Power-UCT for the whole tree is the sum of payoff at root node with payoff starting from any node i after the first action chosen from root node until the end. This payoff by induction at depth $(D-1)$ is

$$\mathcal{O}(K(D-1)(\log(n)/n)^{\frac{1}{p}} + K^{D-1}(1/n)^{\frac{1}{p}}).$$

According to the Theorem 3, the payoff at the root node is

$$|\delta_n^*| + \mathcal{O}\left(\frac{K(\log n + N_0)}{n}\right)^{\frac{1}{p}}.$$

The payoff of the whole tree with depth D

$$\begin{aligned}
&|\delta_n^*| + \mathcal{O}\left(\frac{K(\log n + N_0)}{n}\right)^{\frac{1}{p}} \\
&= \mathcal{O}(K(D-1)(\log(n)/n)^{\frac{1}{p}} + K^{D-1}(1/n)^{\frac{1}{p}}) \\
&+ \mathcal{O}\left(\frac{K(\log n + N_0)}{n}\right)^{\frac{1}{p}} \\
&\leq \mathcal{O}(K(D-1)(\log(n)/n)^{\frac{1}{p}} + K^{D-1}(1/n)^{\frac{1}{p}}) \\
&+ \mathcal{O}\left(K\left(\frac{\log n}{n}\right)^{\frac{1}{p}} + KN_0\left(\frac{1}{n}\right)^{\frac{1}{p}}\right) \\
&= \mathcal{O}(KD(\log(n)/n)^{\frac{1}{p}} + K^{D}(1/n)^{\frac{1}{p}})
\end{aligned}$$

with $N_0 = \mathcal{O}((K-1)K^{D-1})$, which completes our proof of the convergence of Power-UCT. Since by our induction hypothesis this holds for all nodes at a distance of one node from the root, the proof is finished by observing that Theorem 3 and Theorem 5 do indeed ensure that the drift conditions are satisfied. Interestingly, the proof guarantees the convergence for any finite value of p. □

Appendix B. Convex Regularization in Monte-Carlo Tree Search

In this section, we describe how to derive the theoretical results presented for the Convex Regularization in Monte-Carlo Tree Search.

First, the exponential convergence rate of the estimated value function to the conjugate regularized value function at the root node (Theorem 1) is derived based on induction with respect to the depth D of the tree. When $D = 1$, we derive the concentration of the average reward at the leaf node with respect to the ∞-norm (as shown in Lemma 1) based on the result from Theorem 2.19 in Wainwright (2019), and the induction is done over the tree by additionally exploiting the contraction property of the convex regularized value function. Second, based on Theorem 1, we prove the exponential convergence rate of choosing the

best action at the root node (Theorem 2). Third, the pseudo-regret analysis of E3W is derived based on the Bregman divergence properties and the contraction properties of the Legendre-Fenchel transform (Proposition 1). Finally, the bias error of estimated value at the root node is derived based on results of Theorem 1, and the boundedness property of the Legendre-Fenchel transform (Proposition 1).

Let $\hat{r}$ and r be respectively the average and the the expected reward at the leaf node, and the reward distribution at the leaf node be σ^2-sub-Gaussian.

Lemma 1. *For the stochastic bandit problem E3W guarantees that, for $t \geq 4$,*

$$\Pr\big(\parallel r - \hat{r}_t \parallel_\infty \geq \frac{2\sigma}{\log(2+t)}\big) \leq 4|\mathcal{A}| \exp\Big(- \frac{t}{(\log(2+t))^3}\Big).$$

Proof. Let us define $N_t(a)$ as the number of times action a have been chosen until time t, and $\hat{N}_t(a) = \sum_{s=1}^{t} \pi_s(a)$, where $\pi_s(a)$ is the E3W policy at time step s. By choosing $\lambda_s = \frac{|\mathcal{A}|}{\log(1+s)}$, it follows that for all a and $t \geq 4$,

$$\begin{aligned}\hat{N}_t(a) &= \sum_{s=1}^{t} \pi_s(a) \geq \sum_{s=1}^{t} \frac{1}{\log(1+s)} \geq \sum_{s=1}^{t} \frac{1}{\log(1+s)} - \frac{s/(s+1)}{(\log(1+s))^2} \\ &\geq \int_1^{1+t} \frac{1}{\log(1+s)} - \frac{s/(s+1)}{(\log(1+s))^2} ds = \frac{1+t}{\log(2+t)} - \frac{1}{\log 2} \geq \frac{t}{2\log(2+t)}.\end{aligned}$$

From Theorem 2.19 in Wainwright (2019), we have the following concentration inequality

$$\Pr(|N_t(a) - \hat{N}_t(a)| > \epsilon) \leq 2\exp\{-\frac{\epsilon^2}{2\sum_{s=1}^{t}\sigma_s^2}\} \leq 2\exp\{-\frac{2\epsilon^2}{t}\},$$

where $\sigma_s^2 \leq 1/4$ is the variance of a Bernoulli distribution with $p = \pi_s(k)$ at time step s. We define the event

$$E_\epsilon = \{\forall a \in \mathcal{A}, |\hat{N}_t(a) - N_t(a)| \leq \epsilon\},$$

and consequently

$$\Pr(|\hat{N}_t(a) - N_t(a)| \geq \epsilon) \leq 2|\mathcal{A}| \exp(-\frac{2\epsilon^2}{t}). \tag{105}$$

Conditioned on the event E_ϵ, for $\epsilon = \frac{t}{4\log(2+t)}$, we have $N_t(a) \geq \frac{t}{4\log(2+t)}$. For any action a by the definition of sub-gaussian,

$$\Pr\left(|r(a) - \hat{r}_t(a)| > \sqrt{\frac{8\sigma^2 \log(\frac{2}{\delta})\log(2+t)}{t}}\right) \leq \Pr\left(|r(a) - \hat{r}_t(a)| > \sqrt{\frac{2\sigma^2\log(\frac{2}{\delta})}{N_t(a)}}\right) \leq \delta$$

by choosing a δ satisfying $\log(\frac{2}{\delta}) = \frac{1}{(\log(2+t))^3}$, we have

$$\Pr\left(|r(a) - \hat{r}_t(a)| > \sqrt{\frac{2\sigma^2\log(\frac{2}{\delta})}{N_t(a)}}\right) \leq 2\exp\left(-\frac{1}{(\log(2+t))^3}\right).$$

Therefore, for $t \geq 2$

$$\Pr\left(\parallel r - \hat{r}_t \parallel_\infty > \frac{2\sigma}{\log(2+t)} \right) \leq \Pr\left(\parallel r - \hat{r}_t \parallel_\infty > \frac{2\sigma}{\log(2+t)} \middle| E_\epsilon \right) + \Pr(E_\epsilon^C)$$
$$\leq \sum_k \left(\Pr\left(|r(a) - \hat{r}_t(a)| > \frac{2\sigma}{\log(2+t)} \right) + \Pr(E_\epsilon^C) \leq 2|\mathcal{A}| \exp\left(- \frac{1}{(\log(2+t))^3} \right) \right)$$
$$+ 2|\mathcal{A}| \exp\left(- \frac{t}{(\log(2+t))^3} \right) = 4|\mathcal{A}| \exp\left(- \frac{t}{(\log(2+t))^3} \right).$$

□

Lemma 2. *Given two policies $\pi^{(1)} = \nabla\Omega^*(r^{(1)})$ and $\pi^{(2)} = \nabla\Omega^*(r^{(2)}), \exists L$, such that*

$$\parallel \pi^{(1)} - \pi^{(2)} \parallel_p \leq L \parallel r^{(1)} - r^{(2)} \parallel_p .$$

Proof. This comes directly from the fact that $\pi = \nabla\Omega^*(r)$ is Lipschitz continuous with ℓ^p-norm. Note that p has different values according to the choice of regularizer. Refer to Niculae and Blondel (2017) for a discussion of each norm using maximum entropy and Tsallis entropy regularizer. Relative entropy shares the same properties with maximum Entropy. □

Lemma 3. *Consider the E3W policy applied to a tree. At any node s of the tree with depth d, Let us define $N_t^*(s,a) = \pi^*(a|s).t$, and $\hat{N}_t(s,a) = \sum_{s=1}^t \pi_s(a|s)$, where $\pi_k(a|s)$ is the policy at time step k. There exists some C and $\hat{C}$ such that*

$$\Pr\left(|\hat{N}_t(s,a) - N_t^*(s,a)| > \frac{Ct}{\log t} \right) \leq \hat{C}|\mathcal{A}|t \exp\{-\frac{t}{(\log t)^3}\}.$$

Proof. We denote the following event,

$$E_{r_k} = \{\parallel r(s',\cdot) - \hat{r}_k(s',\cdot) \parallel_\infty < \frac{2\sigma}{\log(2+k)}\}.$$

Thus, conditioned on the event $\bigcap_{i=1}^{t} E_{r_t}$ and for $t \geq 4$, we bound $|\hat{N}_t(s,a) - N_t^*(s,a)|$ as

$$\begin{aligned}
|\hat{N}_t(s,a) - N_t^*(s,a)| &\leq \sum_{k=1}^{t} |\hat{\pi}_k(a|s) - \pi^*(a|s)| + \sum_{k=1}^{t} \lambda_k \\
&\leq \sum_{k=1}^{t} \parallel \hat{\pi}_k(\cdot|s) - \pi^*(\cdot|s) \parallel_\infty + \sum_{k=1}^{t} \lambda_k \\
&\leq \sum_{k=1}^{t} \parallel \hat{\pi}_k(\cdot|s) - \pi^*(\cdot|s) \parallel_p + \sum_{k=1}^{t} \lambda_k \\
&\leq L \sum_{k=1}^{t} \parallel \hat{Q}_k(s',\cdot) - Q(s',\cdot) \parallel_p + \sum_{k=1}^{t} \lambda_k (\text{Lemma 2}) \\
&\leq L|\mathcal{A}|^{\frac{1}{p}} \sum_{k=1}^{t} \parallel \hat{Q}_k(s',\cdot) - Q(s',\cdot) \parallel_\infty + \sum_{k=1}^{t} \lambda_k (\text{ Property of } p\text{-norm}) \\
&\leq L|\mathcal{A}|^{\frac{1}{p}} \gamma^d \sum_{k=1}^{t} \parallel \hat{r}_k(s'',\cdot) - r(s'',\cdot) \parallel_\infty + \sum_{k=1}^{t} \lambda_k (\text{Contraction 1}) \\
&\leq L|\mathcal{A}|^{\frac{1}{p}} \gamma^d \sum_{k=1}^{t} \frac{2\sigma}{\log(2+k)} + \sum_{k=1}^{t} \lambda_k \\
&\leq L|\mathcal{A}|^{\frac{1}{p}} \gamma^d \int_{k=0}^{t} \frac{2\sigma}{\log(2+k)} dk + \int_{k=0}^{t} \frac{|\mathcal{A}|}{\log(1+k)} dk \\
&\leq \frac{Ct}{\log t}.
\end{aligned}$$

for some constant C depending on $|\mathcal{A}|, p, d, \sigma, L$, and γ . Finally,

$$\begin{aligned}
\Pr(|\hat{N}_t(s,a) - N_t^*(s,a)| \geq \frac{Ct}{\log t}) &\leq \sum_{i=1}^{t} \Pr(E_{r_t}^c) = \sum_{i=1}^{t} 4|\mathcal{A}| \exp(-\frac{t}{(\log(2+t))^3}) \\
&\leq 4|\mathcal{A}| t \exp(-\frac{t}{(\log(2+t))^3}) \\
&= O(t \exp(-\frac{t}{(\log(t))^3})).
\end{aligned}$$

□

Lemma 4. *Consider the E3W policy applied to a tree. At any node s of the tree, Let us define $N_t^*(s,a) = \pi^*(a|s).t$, and $N_t(s,a)$ as the number of times action a have been chosen until time step t. There exists some C and $\hat{C}$ such that*

$$\Pr\left(|N_t(s,a) - N_t^*(s,a)| > \frac{Ct}{\log t}\right) \leq \hat{C} t \exp\{-\frac{t}{(\log t)^3}\}.$$

Proof. Based on the result from Lemma 3, we have

$$\begin{aligned}
&\Pr\left(|N_t(s,a) - N_t^*(s,a)| > (1+C)\frac{t}{\log t}\right) \leq Ct\exp\{-\frac{t}{(\log t)^3}\} \\
&\leq \Pr\left(|\hat{N}_t(s,a) - N_t^*(s,a)| > \frac{Ct}{\log t}\right) + \Pr\left(|N_t(s,a) - \hat{N}_t(s,a)| > \frac{t}{\log t}\right) \\
&\leq 4|\mathcal{A}|t\exp\{-\frac{t}{(\log(2+t))^3}\} + 2|\mathcal{A}|\exp\{-\frac{t}{(\log(2+t))^2}\} \text{(Lemma 3 and (105))} \\
&\leq O(t\exp(-\frac{t}{(\log t)^3})).
\end{aligned}$$

□

Theorem 8. *At the root node s of the tree, defining $N(s)$ as the number of visitations and $V_{\Omega^*}(s)$ as the estimated value at node s, for $\epsilon > 0$, we have*

$$\Pr(|V_\Omega(s) - V_\Omega^*(s)| > \epsilon) \leq C\exp\{-\frac{N(s)\epsilon}{\hat{C}(\log(2+N(s)))^2}\}.$$

Proof. We prove this concentration inequality by induction. When the depth of the tree is $D = 1$, from Proposition 1, we get

$$|V_\Omega(s) - V_\Omega^*(s)| = \| \Omega^*(Q_\Omega(s,\cdot)) - \Omega^*(Q_\Omega^*(s,\cdot)) \|_\infty \leq \gamma \| \hat{r} - r^* \|_\infty \text{ (Contraction)}$$

where $\hat{r}$ is the average rewards and r^* is the mean reward. So that

$$\Pr(|V_\Omega(s) - V_\Omega^*(s)| > \epsilon) \leq \Pr(\gamma \| \hat{r} - r^* \|_\infty > \epsilon).$$

From Lemma 1, with $\epsilon = \frac{2\sigma\gamma}{\log(2+N(s))}$, we have

$$\begin{aligned}
\Pr(|V_\Omega(s) - V_\Omega^*(s)| > \epsilon) \leq \Pr(\gamma \| \hat{r} - r^* \|_\infty > \epsilon) &\leq 4|\mathcal{A}|\exp\{-\frac{N(s)\epsilon}{2\sigma\gamma(\log(2+N(s)))^2}\} \\
&= C\exp\{-\frac{N(s)\epsilon}{\hat{C}(\log(2+N(s)))^2}\}.
\end{aligned}$$

Let assume we have the concentration bound at the depth $D-1$, Let us define $V_\Omega(s_a) = Q_\Omega(s,a)$, where s_a is the state reached taking action a from state s. then at depth $D-1$

$$\Pr(|V_\Omega(s_a) - V_\Omega^*(s_a)| > \epsilon) \leq C\exp\{-\frac{N(s_a)\epsilon}{\hat{C}(\log(2+N(s_a)))^2}\}. \tag{106}$$

Now at the depth D, because of the Contraction Property, we have

$$\begin{aligned}
|V_\Omega(s) - V_\Omega^*(s)| &\leq \gamma \| Q_\Omega(s,\cdot) - Q_\Omega^*(s,\cdot) \|_\infty \\
&= \gamma|Q_\Omega(s,a) - Q_\Omega^*(s,a)|.
\end{aligned}$$

So that

$$\begin{aligned}\Pr(|V_\Omega(s) - V^*_\Omega(s)| > \epsilon) &\leq \Pr(\gamma \parallel Q_\Omega(s,a) - Q^*_\Omega(s,a) \parallel > \epsilon) \\ &\leq C_a \exp\{-\frac{N(s_a)\epsilon}{\hat{C}_a(\log(2+N(s_a)))^2}\} \\ &\leq C_a \exp\{-\frac{N(s_a)\epsilon}{\hat{C}_a(\log(2+N(s)))^2}\}.\end{aligned}$$

From (106), we can have $\lim_{t\to\infty} N(s_a) = \infty$ because if $\exists L, N(s_a) < L$, we can find $\epsilon > 0$ for which (106) is not satisfied. From Lemma 4, when $N(s)$ is large enough, we have $N(s_a) \to \pi^*(a|s)N(s)$ (for example $N(s_a) > \frac{1}{2}\pi^*(a|s)N(s)$), that means we can find C and $\hat{C}$ that satisfy

$$\Pr(|V_\Omega(s) - V^*_\Omega(s)| > \epsilon) \leq C \exp\{-\frac{N(s)\epsilon}{\hat{C}(\log(2+N(s)))^2}\}.$$

□

Lemma 5. *At any node s of the tree, $N(s)$ is the number of visitations. We define the event*

$$E_s = \{\forall a \in \mathcal{A}, |N(s,a) - N^*(s,a)| < \frac{N^*(s,a)}{2}\} \text{ where } N^*(s,a) = \pi^*(a|s)N(s),$$

where $\epsilon > 0$ and $V_{\Omega^*}(s)$ is the estimated value at node s. We have

$$\Pr(|V_\Omega(s) - V^*_\Omega(s)| > \epsilon | E_s) \leq C \exp\{-\frac{N(s)\epsilon}{\hat{C}(\log(2+N(s)))^2}\}.$$

Proof. The proof is the same as in Theorem 2. We prove the concentration inequality by induction. When the depth of the tree is $D = 1$, from Proposition 1, we get

$$|V_\Omega(s) - V^*_\Omega(s)| = \parallel \Omega^*(Q_\Omega(s,\cdot)) - \Omega^*(Q^*_\Omega(s,\cdot)) \parallel \leq \gamma \parallel \hat{r} - r^* \parallel_\infty \text{ (Contraction Property)}$$

where $\hat{r}$ is the average rewards and r^* is the mean rewards. So that

$$\Pr(|V_\Omega(s) - V^*_\Omega(s)| > \epsilon) \leq \Pr(\gamma \parallel \hat{r} - r^* \parallel_\infty > \epsilon).$$

From Lemma 1, with $\epsilon = \frac{2\sigma\gamma}{\log(2+N(s))}$ and given E_s, we have

$$\begin{aligned}\Pr(|V_\Omega(s) - V^*_\Omega(s)| > \epsilon) \leq \Pr(\gamma \parallel \hat{r} - r^* \parallel_\infty > \epsilon) &\leq 4|\mathcal{A}| \exp\{-\frac{N(s)\epsilon}{2\sigma\gamma(\log(2+N(s)))^2}\} \\ &= C \exp\{-\frac{N(s)\epsilon}{\hat{C}(\log(2+N(s)))^2}\}.\end{aligned}$$

Let assume we have the concentration bound at the depth $D-1$, Let us define $V_\Omega(s_a) = Q_\Omega(s,a)$, where s_a is the state reached taking action a from state s, then at depth $D-1$

$$\Pr(|V_\Omega(s_a) - V^*_\Omega(s_a)| > \epsilon) \leq C \exp\{-\frac{N(s_a)\epsilon}{\hat{C}(\log(2+N(s_a)))^2}\}.$$

Now at depth D, because of the Contraction Property and given E_s, we have

$$\begin{aligned}|V_\Omega(s) - V^*_\Omega(s)| &\leq \gamma \parallel Q_\Omega(s,\cdot) - Q^*_\Omega(s,\cdot) \parallel_\infty \\ &= \gamma |Q_\Omega(s,a) - Q^*_\Omega(s,a)| (\exists a, \text{ satisfied}).\end{aligned}$$

So that

$$\begin{aligned}\Pr(|V_\Omega(s) - V^*_\Omega(s)| > \epsilon) &\leq \Pr(\gamma \parallel Q_\Omega(s,a) - Q^*_\Omega(s,a) \parallel > \epsilon) \\ &\leq C_a \exp\{-\frac{N(s_a)\epsilon}{\hat{C}_a(\log(2+N(s_a)))^2}\} \\ &\leq C_a \exp\{-\frac{N(s_a)\epsilon}{\hat{C}_a(\log(2+N(s)))^2}\} \\ &\leq C \exp\{-\frac{N(s)\epsilon}{\hat{C}(\log(2+N(s)))^2}\} (\text{because of } E_s)\end{aligned}$$

. □

Theorem 9. *Let a_t be the action returned by algorithm E3W at iteration t. Then for t large enough, with some constants $C, \hat{C}$,*

$$\Pr(a_t \neq a^*) \leq Ct \exp\{-\frac{t}{\hat{C}\sigma(\log(t))^3}\}.$$

Proof. Let us define event E_s as in Lemma 5. Let a^* be the action with largest value estimate at the root node state s. The probability that E3W selects a sub-optimal arm at s is

$$\begin{aligned}&\Pr(a_t \neq a^*) \leq \sum_a \Pr(V_\Omega(s_a)) > V_\Omega(s_{a^*}) | E_s) + \Pr(E^c_s) \\ &= \sum_a \Pr((V_\Omega(s_a) - V^*_\Omega(s_a)) - (V_\Omega(s_{a^*}) - V^*_\Omega(s_{a^*})) \geq V^*_\Omega(s_{a^*}) - V^*_\Omega(s_a) | E_s) + \Pr(E^c_s).\end{aligned}$$

Let us define $\Delta = V^*_\Omega(s_{a^*}) - V^*_\Omega(s_a)$, therefore for $\Delta > 0$, we have

$$\begin{aligned}&\Pr(a_t \neq a^*) \leq \sum_a \Pr((V_\Omega(s_a) - V^*_\Omega(s_a)) - (V_\Omega(s_{a^*}) - V^*_\Omega(s_{a^*})) \geq \Delta | E_s) + + \Pr(E^c_s) \\ &\leq \sum_a \Pr(|V_\Omega(s_a) - V^*_\Omega(s_a)| \geq \alpha\Delta | E_s) + \Pr(|V_\Omega(s_{a^*}) - V^*_\Omega(s_{a^*})| \geq \beta\Delta | E_s) + \Pr(E^c_s) \\ &\leq \sum_a C_a \exp\{-\frac{N(s)(\alpha\Delta)}{\hat{C}_a(\log(2+N(s)))^2}\} + C_{a^*} \exp\{-\frac{N(s)(\beta\Delta)}{\hat{C}_{a^*}(\log(2+N(s)))^2}\} + \Pr(E^c_s),\end{aligned}$$

where $\alpha + \beta = 1$, $\alpha > 0$, $\beta > 0$, and $N(s)$ is the number of visitations the root node s. Let us define $\frac{1}{\hat{C}} = \min\{\frac{(\alpha\Delta)}{C_a}, \frac{(\beta\Delta)}{C_{a^*}}\}$, and $C = \frac{1}{|\mathcal{A}|}\max\{C_a, C_{a^*}\}$ we have

$$\Pr(a \neq a^*) \leq C \exp\{-\frac{t}{\hat{C}\sigma(\log(2+t))^2}\} + \Pr(E^c_s).$$

From Lemma 4, $\exists C^{'}, \hat{C}'$ for which

$$\Pr(E_s^c) \leq C^{'} t \exp\{-\frac{t}{\hat{C}'(\log(t))^3}\},$$

so that

$$\Pr(a \neq a^*) \leq O(t \exp\{-\frac{t}{(\log(t))^3}\}).$$

□

Theorem 10. *Consider an E3W policy applied to the tree. Let define* $\mathcal{D}_{\Omega^*}(x, y) = \Omega^*(x) - \Omega^*(y) - \nabla\Omega^*(y)(x - y)$ *as the Bregman divergence between* x *and* y*, The expected pseudo regret* R_n *satisfies*

$$\mathbb{E}[R_n] \leq -\tau\Omega(\hat{\pi}) + \sum_{t=1}^{n} \mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot) + V(\cdot), \hat{V}_t(\cdot)) + \mathcal{O}(\frac{n}{\log n}).$$

Proof. Without loss of generality, we can assume that $V_i \in [-1, 0], \forall i \in [1, |A|]$. as the definition of regret, we have

$$\mathbb{E}[R_n] = nV^* - \sum_{t=1}^{n} \langle \hat{\pi}_t(\cdot), V(\cdot) \rangle \leq \hat{V}_1(0) - \sum_{t=1}^{n} \langle \hat{\pi}_t(\cdot), V(\cdot) \rangle \leq -\tau\Omega(\hat{\pi}) - \sum_{t=1}^{n} \langle \hat{\pi}_t(\cdot), V(\cdot) \rangle .$$

By the definition of the tree policy, we can obtain

$$\begin{aligned}
-\sum_{t=1}^{n} \langle \hat{\pi}_t(\cdot), V(\cdot) \rangle &= -\sum_{t=1}^{n} \left\langle (1 - \lambda_t)\nabla\Omega^*(\hat{V}_t(\cdot)), V(\cdot) \right\rangle - \sum_{t=1}^{n} \left\langle \frac{\lambda_t(\cdot)}{|A|}, V(\cdot) \right\rangle \\
&= -\sum_{t=1}^{n} \left\langle (1 - \lambda_t)\nabla\Omega^*(\hat{V}_t(\cdot)), V(\cdot) \right\rangle - \sum_{t=1}^{n} \left\langle \frac{\lambda_t(\cdot)}{|A|}, V(\cdot) \right\rangle \\
&\leq -\sum_{t=1}^{n} \left\langle \nabla\Omega^*(\hat{V}_t(\cdot)), V(\cdot) \right\rangle - \sum_{t=1}^{n} \left\langle \frac{\lambda_t(\cdot)}{|A|}, V(\cdot) \right\rangle .
\end{aligned}$$

with

$$
\begin{aligned}
-\sum_{t=1}^{n}\left\langle \nabla\Omega^*(\hat{V}_t(\cdot)), V(\cdot)\right\rangle &= \sum_{t=1}^{n}\Omega^*(\hat{V}_t(\cdot)+V(\cdot)) - \sum_{t=1}^{n}\Omega^*(\hat{V}_t(\cdot)) - \sum_{t=1}^{n}\left\langle \nabla\Omega^*(\hat{V}_t(\cdot)), V(\cdot)\right\rangle \\
&\quad - (\sum_{t=1}^{n}\Omega^*(\hat{V}_t(\cdot)+V(\cdot)) - \sum_{t=1}^{n}\Omega^*(\hat{V}_t(\cdot))) \\
&= \sum_{t=1}^{n}\mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot)+V(\cdot), \hat{V}_t(\cdot)) \\
&\quad - (\sum_{t=1}^{n}\Omega^*(\hat{V}_t(\cdot)+V(\cdot)) - \sum_{t=1}^{n}\Omega^*(\hat{V}_t(\cdot))) \\
&\leq \sum_{t=1}^{n}\mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot)+V(\cdot), \hat{V}_t(\cdot)) + n \parallel V(\cdot) \parallel_\infty \\
&\quad \text{(Contraction property, Proposition 1)} \\
&\leq \sum_{t=1}^{n}\mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot)+V(\cdot), \hat{V}_t(\cdot)).(\text{ because } V_i \leq 0)
\end{aligned}
$$

And

$$
-\sum_{t=1}^{n}\left\langle \frac{\lambda_t(\cdot)}{|A|}, V(\cdot)\right\rangle \leq \mathcal{O}(\frac{n}{\log n}), (\text{Because} \sum_{k=1}^{n}\frac{1}{\log(k+1)} \to \mathcal{O}(\frac{n}{\log n}))
$$

So that

$$
\mathbb{E}[R_n] \leq -\tau\Omega(\hat{\pi}) + \sum_{t=1}^{n}\mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot)+V(\cdot), \hat{V}_t(\cdot)) + \mathcal{O}(\frac{n}{\log n}).
$$

We consider the generalized Tsallis Entropy $\Omega(\pi) = \mathcal{S}_\alpha(\pi) = \frac{1}{1-\alpha}(1 - \sum_i \pi^\alpha(a_i|s))$. According to (Abernethy et al., 2015), when $\alpha \in (0,1)$

$$
\begin{aligned}
\mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot)+V(\cdot), \hat{V}_t(\cdot)) &\leq (\tau\alpha)^{-1}|\mathcal{A}|^\alpha \\
-\Omega(\hat{\pi}_n) &\leq \frac{1}{1-\alpha}(|\mathcal{A}|^{1-\alpha} - 1).
\end{aligned}
$$

Then, for the generalized Tsallis Entropy, when $\alpha \in (0,1)$, the regret is

$$
\mathbb{E}[R_n] \leq \frac{\tau}{1-\alpha}(|\mathcal{A}|^{1-\alpha} - 1) + n(\tau\alpha)^{-1}|\mathcal{A}|^\alpha + \mathcal{O}(\frac{n}{\log n}),
$$

when $\alpha = 2$, which is the Tsallis entropy case we consider, according to Zimmert and Seldin (2019), By Taylor's theorem $\exists z \in \text{conv}(\hat{V}_t, \hat{V}_t + V)$, we have

$$
\mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot)+V(\cdot), \hat{V}_t(\cdot)) \leq \frac{1}{2}\left\langle V(\cdot), \nabla^2\Omega^*(z)V(\cdot)\right\rangle \leq \frac{|\mathcal{K}|}{2}.
$$

So that when $\alpha = 2$, we have

$$\mathbb{E}[R_n] \leq \tau(\frac{|\mathcal{A}| - 1}{|\mathcal{A}|}) + \frac{n|\mathcal{K}|}{2} + \mathcal{O}(\frac{n}{\log n}).$$

when $\alpha = 1$, which is the maximum entropy case in our work, we derive.

$$\mathbb{E}[R_n] \leq \tau(\log |\mathcal{A}|) + \frac{n|\mathcal{A}|}{\tau} + \mathcal{O}(\frac{n}{\log n})$$

Finally, when the convex regularizer is relative entropy, One can simply write $KL(\pi_t||\pi_{t-1}) = -H(\pi_t) - \mathbb{E}_{\pi_t} \log \pi_{t-1}$, let $m = \min_a \pi_{t-1}(a|s)$, we have

$$\mathbb{E}[R_n] \leq \tau(\log |\mathcal{A}| - \frac{1}{m}) + \frac{n|\mathcal{A}|}{\tau} + \mathcal{O}(\frac{n}{\log n}).$$

□

Before derive the next theorem, we state the Theorem 2 in Geist et al. (2019)

- Boundedness: for two constants L_Ω and U_Ω such that for all $\pi \in \Pi$, we have $L_\Omega \leq \Omega(\pi) \leq U_\Omega$, then

$$V^*(s) - \frac{\tau(U_\Omega - L_\Omega)}{1 - \gamma} \leq V^*_\Omega(s) \leq V^*(s). \tag{107}$$

Where τ is the temperature and γ is the discount constant.

Theorem 11. *For any $\delta > 0$, with probability at least $1 - \delta$, the ε_Ω satisfies*

$$-\sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} - \frac{\tau(U_\Omega - L_\Omega)}{1 - \gamma} \leq \varepsilon_\Omega \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}}.$$

Proof. From Theorem 2, let us define $\delta = C \exp\{-\frac{2N(s)\epsilon^2}{\hat{C}\sigma^2}\}$, so that $\epsilon = \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}}$ then for any $\delta > 0$, we have

$$\Pr(|V_\Omega(s) - V^*_\Omega(s)| \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}}) \geq 1 - \delta.$$

Then, for any $\delta > 0$, with probability at least $1 - \delta$, we have

$$\begin{aligned}
&|V_\Omega(s) - V^*_\Omega(s)| \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} \\
&-\sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} \leq V_\Omega(s) - V^*_\Omega(s) \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} \\
&-\sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} + V^*_\Omega(s) \leq V_\Omega(s) \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} + V^*_\Omega(s).
\end{aligned}$$

From Proposition 1, we have

$$-\sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} + V^*(s) - \frac{\tau(U_\Omega - L_\Omega)}{1 - \gamma} \leq V_\Omega(s) \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} + V^*(s).$$

□

Appendix C. A Unified Perspective on Value Backup and Exploration in Monte-Carlo Tree Search

Theorem 12. *When $\alpha \in (0,1)$, the regret of E3W (Dam et al., 2021) with the regularizer f_α is*

$$\mathbb{E}[R_n] \leq \frac{\tau}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha}-1) + n(2\tau)^{-1}|\mathcal{A}|^{\alpha} + \mathcal{O}(\frac{n}{\log n}).$$

Proof. Please refer to equation (107) in Theorem 10 □

Theorem 13. *When $\alpha \in (1,\infty)$, $\alpha \neq 2$, the regret of E3W (Dam et al., 2021) with the regularizer f_α is*

$$\mathbb{E}[R_n] \leq \frac{\tau}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha}-1) + \frac{n|\mathcal{K}|}{2} + \mathcal{O}(\frac{n}{\log n}).$$

where $|\mathcal{K}|$ is the number of actions that are assigned non-zero probability in the policy at the root node.

Proof. From Theorem 10, we have

$$\mathbb{E}[R_n] \leq -\tau\Omega(\hat{\pi}) + \sum_{t=1}^{n} \mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot) + V(\cdot), \hat{V}_t(\cdot)) + \mathcal{O}(\frac{n}{\log n}).$$

Here, $\Omega(\hat{\pi}) = f_\alpha(\hat{\pi}) = \frac{1}{\alpha(1-\alpha)}(1 - \sum_i \hat{\pi}^\alpha(a_i|s))$. So as the result from Theorem 10, we have

$$-\Omega(\hat{\pi}_n) \leq \frac{1}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha}-1).$$

By Taylor's theorem $\exists z \in \text{conv}(\hat{V}_t, \hat{V}_t + V)$, we have

$$\mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot) + V(\cdot), \hat{V}_t(\cdot)) \leq \frac{1}{2}\left\langle V(\cdot), \nabla^2\Omega^*(z)V(\cdot)\right\rangle.$$

So that according to Equations (46), (47), (48), (49), we have

$$\mathcal{D}_{\Omega^*}(\hat{V}_t(\cdot) + V(\cdot), \hat{V}_t(\cdot)) \leq \frac{1}{2}\left\langle V(\cdot), \nabla^2\Omega^*(z)V(\cdot)\right\rangle \leq \frac{|\mathcal{K}|}{2}.$$

so that

$$\mathbb{E}[R_n] \leq \frac{\tau}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha}-1) + \frac{n|\mathcal{K}|}{2} + \mathcal{O}(\frac{n}{\log n}).$$

□

We analyse the error of the regularized value estimate at the root node $n(s)$ w.r.t. the optimal value: $\varepsilon_\Omega = V_\Omega(s) - V^*(s)$. where Ω is the α-divergence regularizer f_α.

Theorem 14. *For any $\delta > 0$ and α-divergence regularizer f_α ($\alpha \neq 1, 2$), with some constant $C, \hat{C}$, with probability at least $1 - \delta$, ε_Ω satisfies*

$$-\sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} - \frac{\tau}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha} - 1) \leq \varepsilon_\Omega \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}}. \tag{108}$$

Proof. We have

$$0 \leq -\Omega(\hat{\pi}_n) \leq \frac{1}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha} - 1).$$

combine with Theorem 11 we will have

$$-\sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}} - \frac{\tau}{\alpha(1-\alpha)}(|\mathcal{A}|^{1-\alpha} - 1) \leq \varepsilon_\Omega \leq \sqrt{\frac{\hat{C}\sigma^2 \log \frac{C}{\delta}}{2N(s)}}. \tag{109}$$

□

Appendix D. Experimental Hyperparameter Search

Table 6: The hyperparameter τ (temperature) for MENTS and TENTS in Atari.

	MENTS	RENTS	TENTS
Alien	0.1	0.01	0.03
Asterix	0.02	0.09	0.1
Asteroids	0.08	0.1	0.2
Atlantis	0.08	0.01	0.03
BankHeist	0.0	0.01	0.0
BeamRider	0.02	0.02	0.03
Breakout	0.02	0.01	0.04
Centipede	0.0	0.5	0.0
DemonAttack	0.0	0.6	0.0
Enduro	0.02	0.08	0.1
Frostbite	0.01	0.01	0.02
Gopher	0.0	0.07	0.0
Hero	0.4	0.04	0.03
MsPacman	0.09	0.08	0.03
Phoenix	0.07	0.03	0.6
Qbert	0.02	0.06	0.4
Robotank	0.01	0.03	0.05
Seaquest	0.02	0.02	0.03
Solaris	0.03	0.05	0.06
SpaceInvaders	0.02	0.01	0.06
WizardOfWor	0.1	0.2	0.01

To compare the performance of UCT, Power-UCT, E3W to other state-of-the-art planning algorithms, we run several experiments on standard MDP as well as POMDP environments. The hyperparameters are tuned using grid-search. Except for the case of PocMan environment, we scale the rewards into the range [0, 1]. We set the discount factor $\gamma = 0.99$ in all

MDPs (except for *FrozenLake*(8×8) where we set the discount factor=1.0) and $\gamma = 0.95$ in all POMDPs.

FrozenLake:In *FrozenLake*(8×8) environment, since this is a challenging problem, we do Bayesian optimization to find the best hyperparameter for MENTS with $\tau = 0.046, \epsilon = 0.17$. We perform hyper parameter search for the exploration constant C in UCT with C $= \{0.01, 0.02, ..., 1.2, 1.3, 1.41\}$ and find that UCT works best for $C = 1.41$. We set $C = 1.41$ for Power-UCT also.

Atari: We run E3W in Atari with 100 random seeds, where each seed with 512 samples and collect the average score. We found that only 512 simulations were necessary due to the utilization of a pretrained neural network. The temperature parameter τ of MENTS, RENTS and TENTS is tuned from {0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0}. The selected parameter τ are shown in Table 6. The exploration constant ϵ for MENTS, RENTS and TENTS are set to 0.01. For MaxMCTS, we find the best hyperparameter $\lambda = 1.0$

Rocksample and ***pocman***: In all *Rocksample* and *Pocman* environments, we set the heuristic for rollouts as $treeknowledge = 0, rolloutknowledge = 1$. We use gridsearch $(\tau = 0.01, 0.02, ..., 0.1, 0.2, ..., 1.0)$, $(\epsilon = 1.0, 0.5, 0.25, 0.125, 0.01, 0, 025)$ to find the best hyperparameter $\tau = 0.5, \epsilon = 0.125$ for MENTS in *Pocman* (Table. 4). The exploration constant for UCT and Power-UCT is set to $c = \sqrt{2}$.

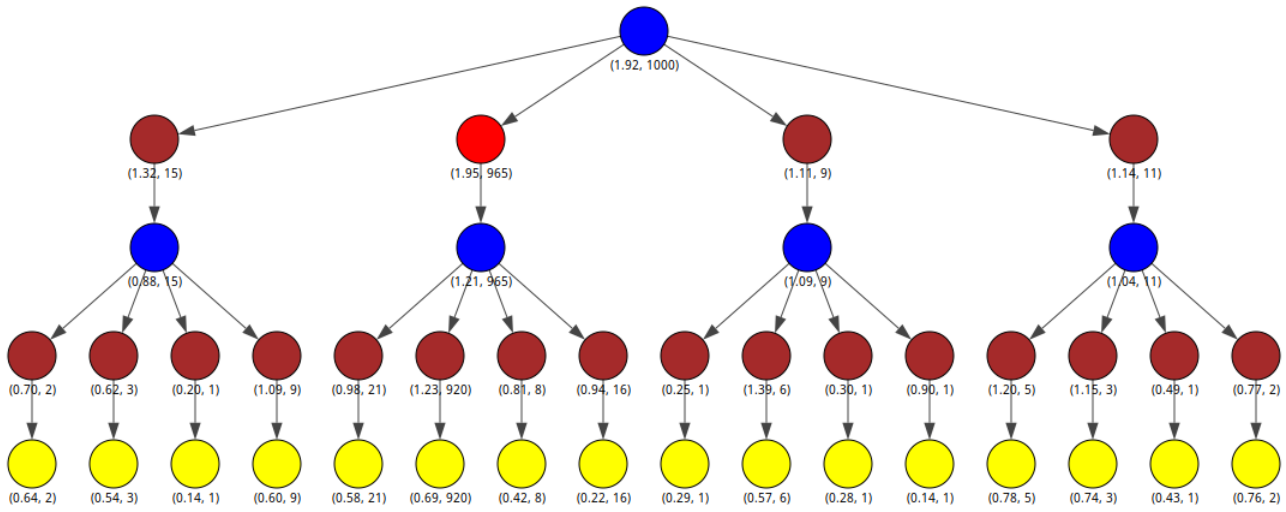
Copy: For *Copy* environments (Table. 3), We find the best hyperparameter after a grid-search for MENTS with $\tau = 0.1, 1.0, 0.08$ in *Copy*-144,*Copy*-200 and *Copy*-300 respectively. For RENT, the performance does not change much and the best hyperparameter after a gridsearch for RENTS is $\tau = 0.08$ in all *Copy*-144,*Copy*-200 and *Copy*-300. We set $\epsilon = 0$ for both MENTS and RENTS since the softmax policy itself does the exploration and the *Copy* environment does not need much exploration.

We find the exploration constant works best for UCT and Power-UCT with $c = 0.25$ by doing grid search over {0.1, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75}.

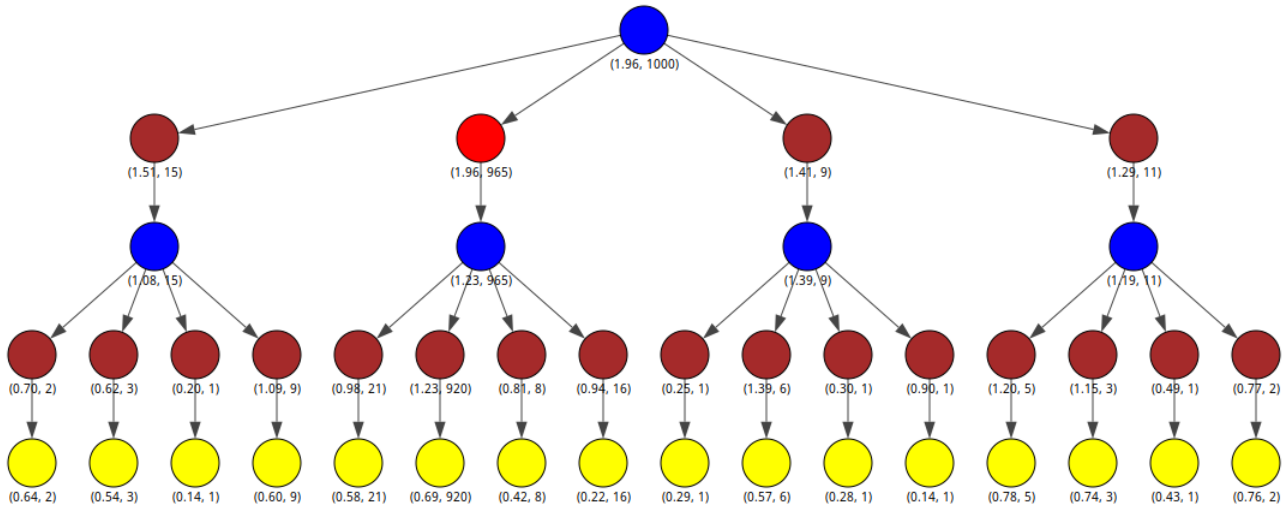
Appendix E. Additional experiments

We additionally include a concrete running example of UCT, Power-UCT and MENTS, TENTS, α-divergence($\alpha = 4.0, 8.0, 10.0, 16.0$) approaches on a small synthetic tree to significantly show the benefits of each method. We run each algorithm in a synthetic tree and show the whole planning tree of each method after a certain number of samples. Due to the limitation space, we plot four Figures. Fig. 14 shows the whole MCTS tree comparing UCT and Power-UCT. Fig. 15 compares MENTS and TENTS. Fig. 16 shows compares $\alpha-$divergence with $\alpha = 4$ and $\alpha = 8$, while Fig. 17 compares $\alpha-$divergence with $\alpha = 10$ and $\alpha = 16$. As shown in Fig. 14, Power-UCT and UCT share nearly the same visitation count of each node, and both methods show that we can find the optional node in red color. However, due to the power mean estimator, the estimated value of Power-UCT is higher than the average mean estimator of UCT. We further compare the detail intermediate steps of MENTS and TENTS in Fig. 15. Both methods can find the optimal action (shown in red color). However, it shows that TENTS converges faster than MENTS (as the number of visitation count at the optimal node is 706 compared to 695 of MENTS after 1000 samples.)

We take a further step to compare α-divergence with difference value of $\alpha = 4, 8, 10, 16$. As



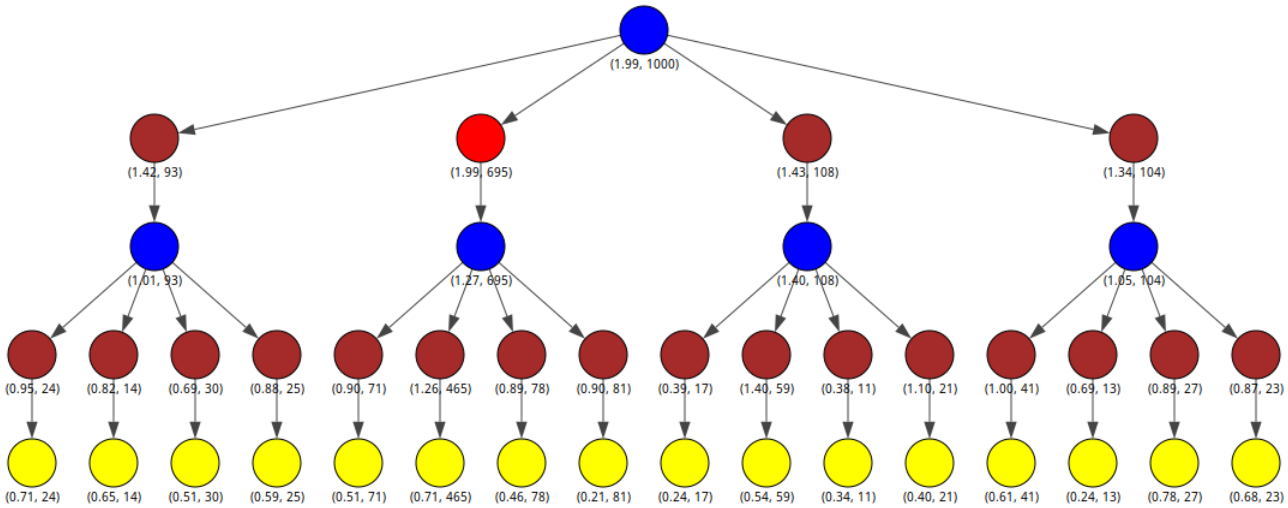
(a) UCT.



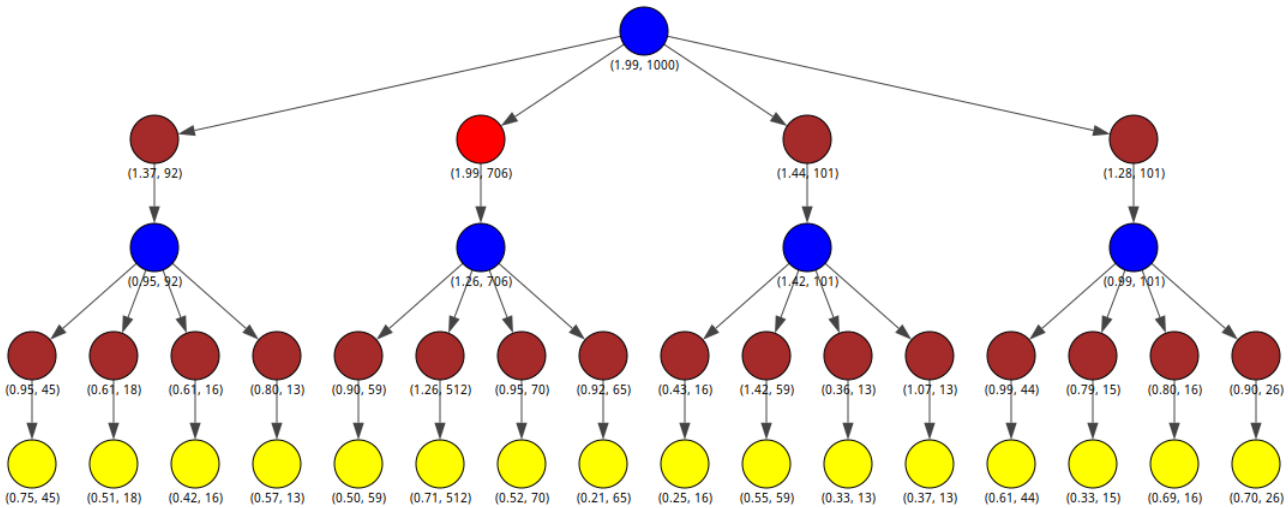
(b) Power-UCT.

Figure 14: Monte-Carlo Tree Search of UCT and Power-UCT after 1000 samples. Blue color circle is Vnode, Brown color circle is Qnode. Yellow color circle is the leaf node. Red color circle is the node with the optimal Qvalue function. Labels below the circles are (empirical mean value, visitation count). The two methods share nearly the same visitation count of each node. However, due to the power mean estimator, the estimated value in Power-UCT is higher than the average mean estimator of UCT.

shown in Fig 16, Fig 17, the convergence rate of choosing the optimal node (in red color) increase when we increase the value of $\alpha = 4, 8, 10$. However, the performance does not increase when we increase $\alpha = 10$, and $\alpha = 16$.

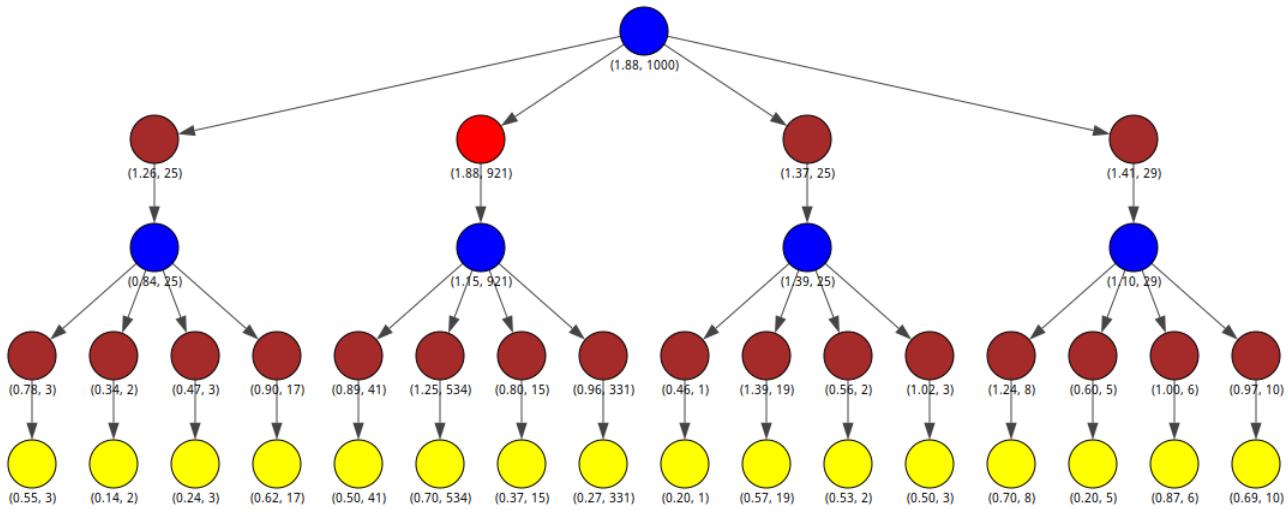


(a) MENTS.

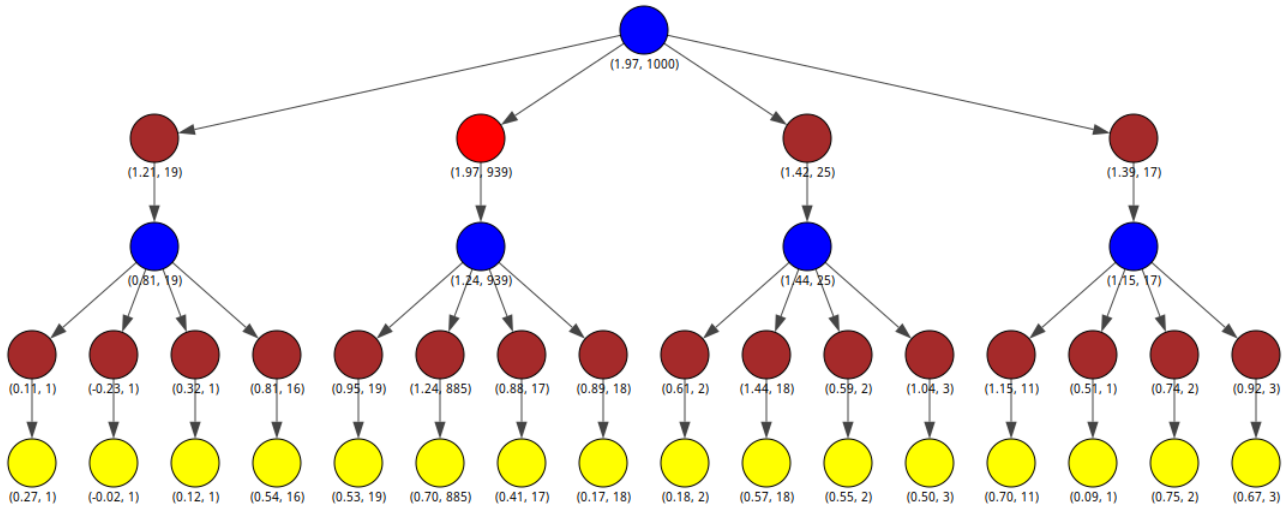


(b) TENTS.

Figure 15: Monte-Carlo Tree Search of MENTS and TENTS after 1000 samples. Blue color circle is Vnode, Brown color circle is Qnode. Yellow color circle is the leaf node. Red color circle is the node with the optimal Qvalue function. Labels below the circles are (empirical mean value, visitation count). Due to the sparsity of action selection policy, optimal action selection in TENTS converges faster than MENTS as we can see at the optimal node in red color, the number of visitation in the tree of TENTS is 706 compared to 695 in MENTS.

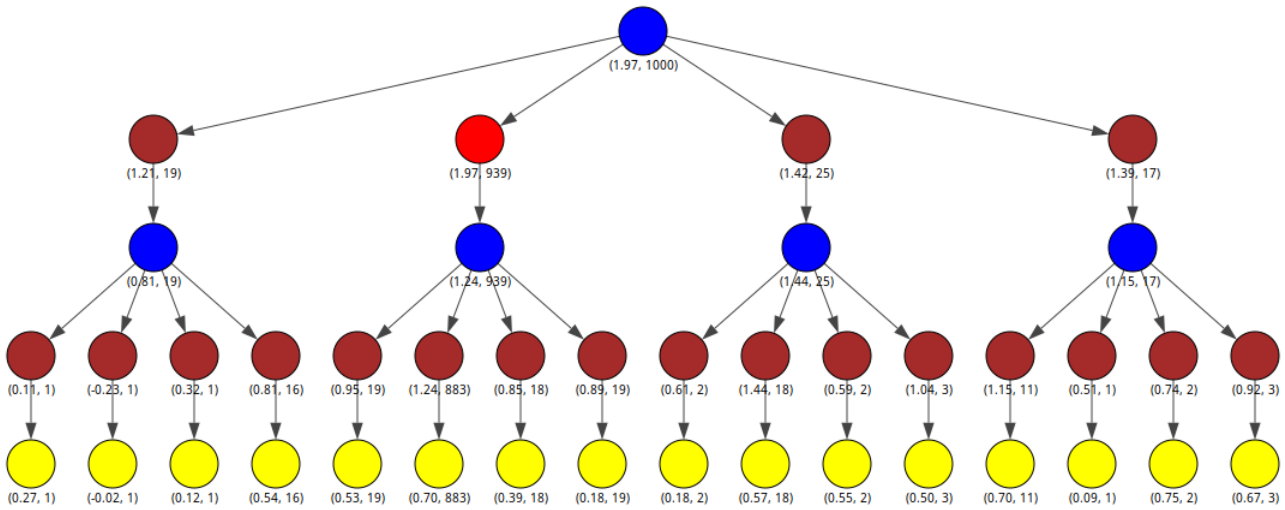


(a) α-divergence($\alpha = 4.0$).

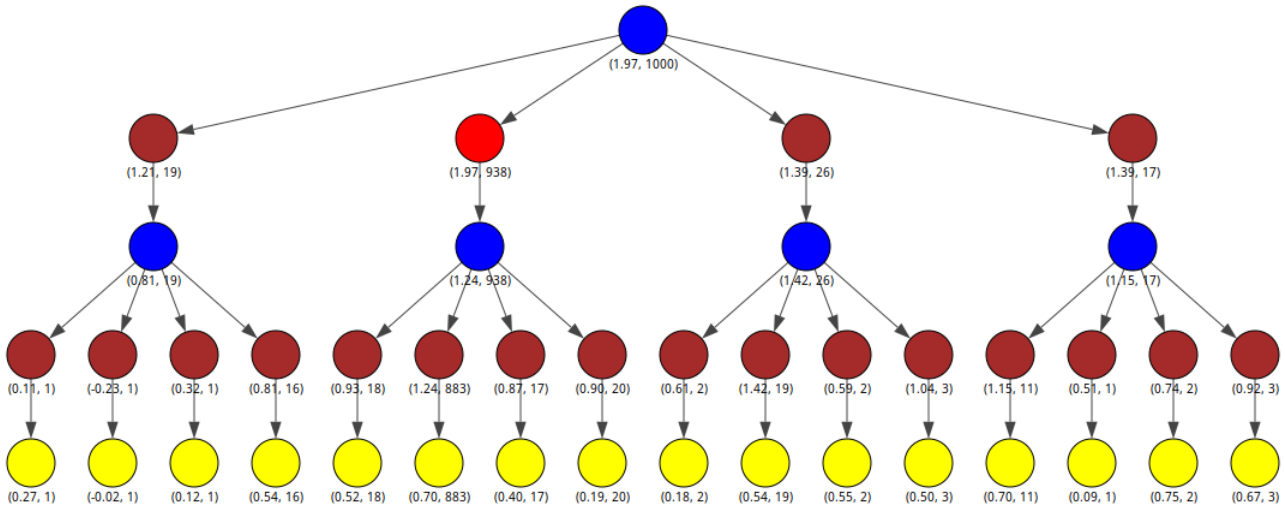


(b) α-divergence($\alpha = 8.0$).

Figure 16: Monte-Carlo Tree Search of α-divergence($\alpha = 4.0$) and α-divergence($\alpha = 8.0$) after 1000 samples. Blue color circle is Vnode, Brown color circle is Qnode. Yellow color circle is the leaf node. Red color circle is the node with the optimal Qvalue function. Labels below the circles are (empirical mean value, visitation count). The optimal action selection in α-divergence($\alpha = 8.0$) converges faster than α-divergence($\alpha = 4.0$) as we can see at the optimal node in red color, the number of visitation in the tree of $\alpha = 4.0$ is 921 compared to 939 of $\alpha = 8.0$.



(a) α-divergence($\alpha = 10.0$).



(b) α-divergence($\alpha = 16.0$).

Figure 17: Monte-Carlo Tree Search of α-divergence($\alpha = 10.0$) and α-divergence($\alpha = 16.0$) after 1000 samples. Blue color circle is Vnode, Brown color circle is Qnode. Yellow color circle is the leaf node. Red color circle is the node with the optimal Qvalue function. Labels below the circles are (empirical mean value, visitation count). The optimal action selection in $\alpha = 10.0$, and $\alpha = 16.0$ share the same converges rate as we can see at the optimal node in red color, the number of visitation in the tree of $\alpha = 10.0$ is 939 compared to 938 of $\alpha = 16.0$.

References

Abernethy, J. D., Lee, C., and Tewari, A. (2015). Fighting bandits with a new kind of smoothness. *Advances in Neural Information Processing Systems*, 28.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256.

Bellemare, M. G., Dabney, W., and Munos, R. (2017). A distributional perspective on reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 449–458. JMLR. org.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. (2013). The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279.

Bellman, R. (1954). The theory of dynamic programming. Technical report, Rand corp santa monica ca.

Belousov, B. and Peters, J. (2019). Entropic regularization of markov decision processes. *Entropy*, 21(7):674.

Ben-Tal, A., Charnes, A., and Teboulle, M. (1989). Entropic means. *Journal of Mathematical Analysis and Applications*, 139(2):537–551.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). Openai gym.

Buesing, L., Heess, N., and Weber, T. (2020). Approximate inference in discrete distributions with monte carlo tree search and value functions. In *International Conference on Artificial Intelligence and Statistics*, pages 624–634. PMLR.

Bullen, P. S. (2013). *Handbook of means and their inequalities*. Springer Science & Business Media.

Chen, G., Peng, Y., and Zhang, M. (2018). Effective exploration for deep reinforcement learning via bootstrapped q-ensembles under tsallis entropy regularization.

Chen, J., Zhang, C., Luo, J., Xie, J., and Wan, Y. (2020). Driving maneuvers prediction based autonomous driving control by deep monte carlo tree search. *IEEE transactions on vehicular technology*, 69(7):7146–7158.

Coquelin, P.-A. and Munos, R. (2007). Bandit algorithms for tree search.

Coulom, R. (2006). Efficient selectivity and backup operators in monte-carlo tree search. In *International conference on computers and games*. Springer.

Csiszár, I. (1964). Eine informationstheoretische ungleichung und ihre anwendung auf beweis der ergodizitaet von markoffschen ketten. *Magyer Tud. Akad. Mat. Kutato Int. Koezl.*, 8:85–108.

Dam, T., Klink, P., D'Eramo, C., Peters, J., and Pajarinen, J. (2019). Generalized mean estimation in monte-carlo tree search.

Dam, T. Q., D'Eramo, C., Peters, J., and Pajarinen, J. (2021). Convex regularization in monte-carlo tree search. In *International Conference on Machine Learning*, pages 2365–2375. PMLR.

Danihelka, I., Guez, A., Schrittwieser, J., and Silver, D. (2021). Policy improvement by planning with gumbel. In *International Conference on Learning Representations*.

Feldman, Z. and Domshlak, C. (2013). Monte-carlo planning: Theoretically fast convergence meets practical efficiency.

Francois, D., Wertz, V., and Verleysen, M. (2007). The concentration of fractional distances. *IEEE Transactions on Knowledge and Data Engineering*, 19(7):873–886.

Funk, N., Chalvatzaki, G., Belousov, B., and Peters, J. (2021). Learn2assemble with structured representations and search for robotic architectural construction. In *5th Annual Conference on Robot Learning*.

Geist, M. and Scherrer, B. (2011). L1-penalized projected bellman residual. In *Proceedings of the European Workshop on Reinforcement Learning (EWRL 2011)*, Lecture Notes in Computer Science (LNCS). Springer Verlag - Heidelberg Berlin.

Geist, M., Scherrer, B., and Pietquin, O. (2019). A theory of regularized markov decision processes. In *International Conference on Machine Learning*, pages 2160–2169.

Grill, J.-B., Altché, F., Tang, Y., Hubert, T., Valko, M., Antonoglou, I., and Munos, R. (2020). Monte-carlo tree search as regularized policy optimization.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870.

Hay, N., Russell, S., Tolpin, D., and Shimony, S. E. (2014). Selecting computations: Theory and applications.

Khandelwal, P., Liebman, E., Niekum, S., and Stone, P. (2016). On the analysis of complex backup strategies in monte carlo tree search. In *International Conference on Machine Learning*.

Kocsis, L., Szepesvári, C., and Willemson, J. (2006). Improved monte-carlo search. *Univ. Tartu, Estonia, Tech. Rep*, 1.

Lee, K., Choi, S., and Oh, S. (2018). Sparse markov decision processes with causal sparse tsallis entropy regularization for reinforcement learning. *IEEE Robotics and Automation Letters*, 3(3):1466–1473.

Lee, K., Kim, S., Lim, S., Choi, S., and Oh, S. (2019). Tsallis reinforcement learning: A unified framework for maximum entropy reinforcement learning.

Mei, J., Xiao, C., Huang, R., Schuurmans, D., and Müller, M. (2019). On principled entropy exploration in policy optimization. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, pages 3130–3136. AAAI Press.

Mensch, A. and Blondel, M. (2018). Differentiable dynamic programming for structured prediction and attention. In *International Conference on Machine Learning*, pages 3462–3471.

Mitrinovic, D. S. and Vasic, P. M. (1970). *Analytic inequalities*. Springer.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533.

Montgomery, W. H. and Levine, S. (2016). Guided policy search via approximate mirror descent. In *Advances in Neural Information Processing Systems*, pages 4008–4016.

Nachum, O. and Dai, B. (2020a). Reinforcement learning via fenchel-rockafellar duality. *CoRR*, abs/2001.01866.

Nachum, O. and Dai, B. (2020b). Reinforcement learning via fenchel-rockafellar duality.

Nguyen, Q. V., Colas, F., Vincent, E., and Charpillet, F. (2017). Long-term robot motion planning for active sound source localization with monte carlo tree search. In *2017 Hands-free Speech Communications and Microphone Arrays (HSCMA)*, pages 61–65. IEEE.

Niculae, V. and Blondel, M. (2017). A regularized framework for sparse and structured neural attention.

Osband, I., Blundell, C., Pritzel, A., and Van Roy, B. (2016). Deep exploration via bootstrapped dqn. *Advances in neural information processing systems*, 29:4026–4034.

Pavel, L. (2007). An extension of duality to a game-theoretic framework. *Automatica*, 43(2):226 – 237.

Rummery, G. A. (1995). *Problem solving with reinforcement learning*. PhD thesis, University of Cambridge Ph. D. dissertation.

Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., Lillicrap, T., and Silver, D. (2019). Mastering atari, go, chess and shogi by planning with a learned model.

Schulman, J., Chen, X., and Abbeel, P. (2017a). Equivalence between policy gradients and soft q-learning.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017b). Proximal policy optimization algorithms.

Shalev-Shwartz, S. and Singer, Y. (2006). Convex repeated games and fenchel duality. *Advances in neural information processing systems*, 19:1265–1272.

Shperberg, S. S., Shimony, S. E., and Felner, A. (2017). Monte-carlo tree search using batch value of perfect information. In *UAI*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. (2017a). Mastering chess and shogi by self-play with a general reinforcement learning algorithm.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017b). Mastering the game of go without human knowledge. *Nature*, 550(7676):354–359.

Silver, D. and Veness, J. (2010). Monte-carlo planning in large pomdps. In *Advances in neural information processing systems*.

Smith, T. and Simmons, R. (2004). Heuristic search value iteration for pomdps. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 520–527. AUAI Press.

Sukkar, F., Best, G., Yoo, C., and Fitch, R. (2019). Multi-robot region-of-interest reconstruction with dec-mcts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9101–9107. IEEE.

Sutton, R. S. and Barto, A. G. (1998). *Introduction to reinforcement learning*, volume 135. MIT press Cambridge.

Tesauro, G., Rajan, V. T., and Segal, R. (2012). Bayesian inference in monte-carlo tree search.

Tolpin, D. and Shimony, S. (2012). Mcts based on simple regret.

Van Hasselt, H., Guez, A., and Silver, D. (2016). Deep reinforcement learning with double q-learning. In *Thirtieth AAAI conference on artificial intelligence*.

Vodopivec, T., Samothrakis, S., and Ster, B. (2017). On monte carlo tree search and reinforcement learning. *Journal of Artificial Intelligence Research*, 60:881–936.

Volpi, N. C., Wu, Y., and Ognibene, D. (2017). Towards event-based mcts for autonomous cars. In *2017 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pages 420–427. IEEE.

Wainwright, M. J. (2019). *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press.

Wasserman, L. (2004). All of statistics: a concise course in statistical inference. 2004.

Xiao, C., Huang, R., Mei, J., Schuurmans, D., and Müller, M. (2019). Maximum entropy monte-carlo planning. In *Advances in Neural Information Processing Systems*, pages 9516–9524.

Zimmert, J. and Seldin, Y. (2019). An optimal algorithm for stochastic and adversarial bandits. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 467–475. PMLR.