On MABs and Separation of Concerns in Monte-Carlo Planning for MDPs

Zohar Feldman and **Carmel Domshlak**
Technion—Israel Institute of Technology
Haifa, Israel
{zoharf@tx,dcarmel@ie}.technion.ac.il

Abstract

Linking online planning for MDPs with their special case of stochastic multi-armed bandit problems, we analyze three state-of-the-art Monte-Carlo tree search algorithms: UCT, BRUE, and MaxUCT. Using the outcome, we (i) introduce two new MCTS algorithms, MaxBRUE, which combines uniform sampling with Bellman backups, and MpaUCT, which combines UCB1 with a novel backup procedure, (ii) analyze them formally and empirically, and (iii) show how MCTS algorithms can be further stratified by an exploration control mechanism that improves their empirical performance without harming the formal guarantees.

Introduction

In online planning for MDPs, the agent focuses on its current state only, deliberates about the set of possible policies from that state onwards and, when interrupted, chooses what action to perform next. In formal analysis of algorithms for online MDP planning, the quality of the action a, chosen for state s with H steps-to-go, is assessed in terms of the *simple regret* measure, capturing the performance loss that results from taking a and then following an optimal policy π^* for the remaining $H-1$ steps, instead of following π^* from the beginning (Bubeck and Munos 2010).

Most algorithms for online MDP planning constitute variants of what is called Monte-Carlo tree search (MCTS) (Sutton and Barto 1998; Péret and Garcia 2004; Kocsis and Szepesvári 2006; Coquelin and Munos 2007; Cazenave 2009; Rosin 2011; Tolpin and Shimony 2012). When the MDP is specified declaratively, that is, when all its parameters are provided explicitly, the palette of algorithmic choices is wider (Bonet and Geffner 2012; Kolobov, Mausam, and Weld 2012; Busoniu and Munos 2012; Keller and Helmert 2013). However, when only a generative model of MDP is available, that is, when the actions of the MDP are given only by their "black box" simulators, MCTS algorithms are basically the only choice. In MCTS, agent deliberation is based on simulated sequential sampling of the state space. MCTS algorithms have also become popular in other settings of sequential decision making, including those with partial state observability and adversarial effects (Gelly and Silver 2011; Sturtevant 2008; Bjarnason, Fern, and Tadepalli 2009; Balla and Fern 2009; Eyerich, Keller, and Helmert 2010; Browne et al. 2012).

The popularity of MCTS methods is due in part to their ability to their ability to deal with generative problem representations, but they have other desirable features as well. First, while MCTS algorithms can natively exploit problem-specific heuristic functions, their correctness is independent of the heuristic's properties, and they can as well be applied without any heuristic information whatsoever. Second, numerous MCTS algorithms exhibit strong anytimeness: not only can a meaningful action recommendation be provided at any interruption point instantly, in time $O(1)$, but the quality of the recommendation also improves very smoothly, in time steps that are independent of the size of the explored state space.

Fundamental developments in the area of MCTS algorithms can all be traced back to stochastic multi-armed bandit (MAB) problems (Robbins 1952). Here we take a closer look at three state-of-the-art MCTS algorithms, UCT (Kocsis and Szepesvári 2006), BRUE (Feldman and Domshlak 2012; 2013), and MaxUCT (Keller and Helmert 2013), linking them to algorithms for online planning in MABs. This analysis leads to certain interesting realizations about the examined MCTS algorithms. Taking these realizations as our point of departure, we:

- Introduce two new MCTS algorithms, MaxBRUE, which combines uniform sampling with Bellman backups, and MpaUCT which combines UCB1 with a novel backup procedure;
- Establish formal guarantees of exponential-rate convergence for MaxBRUE (that turn out to be even stronger than those known to be provided by BRUE), and a hint about the polynomial-rate convergence of MpaUCT;
- Demonstrate empirically that, in line with the empirical analysis of pure exploration in MAB (Bubeck and Munos 2010), MaxBRUE performs better than MpaUCT and MaxUCT under a permissive planning-time allowance, while the opposite holds under short planning times;
- Show how MaxBRUE (and probably other algorithms) can be stratified by an exploration control mechanism that substantially improves the empirical performance without harming the formal guarantees.

Background

Henceforth, the operation of drawing a sample from a distribution $\mathcal{D}$ over set $\aleph$ is denoted by $\sim \mathcal{D}[\aleph]$, $\mathcal{U}$ denotes uniform distribution, and $[\![n]\!]$ for $n \in \mathbb{N}$ denotes the set $\{1, \ldots, n\}$. For a sequence of tuples ρ, $\rho[i]$ denotes the i-th tuple along ρ, and $\rho[i].x$ denotes the value of the field x in that tuple.

Markov Decision Processes. MDP is a standard model for planning under uncertainty (Puterman 1994). An MDP $\langle S, A, \mathbb{P}, R\rangle$ is defined by a set of states S, a set of state transforming actions A, a stochastic transition function $\mathbb{P} : S\times A\times S \to [0, 1]$, and a reward function $R : S\times A\times S \to \mathbb{R}$. The states are fully observable and, in the finite horizon setting considered here, the rewards are accumulated over some predefined number of steps H. In what follows, $s\langle h\rangle$ denotes an MDP state s with h steps-to-go, and $A(s) \subseteq A$ denotes the actions applicable in state s. The objective of planning in MDPs is to sequentially choose actions so as to maximize the accumulated reward. The representation of large-scale MDPs can be either declarative or generative, but anyway concise, and allowing for simulated execution of all feasible action sequences, from any state of the MDP. Henceforce, the state and action branching factors of the MDP in question are denoted by $K = \max_s |A(s)|$ and $B = \max_{s,a} |\{s' \mid \mathbb{P}(s'|s, a) > 0\}|$ respectively.

Simple Regret Minimization in MAB. A stochastic multi-armed bandit (MAB) problem is an MDP defined over a single state s. The actions in MABs do not affect the state, but are associated with stochastic rewards. Most research on MABs has been devoted to the setup of reinforcement learning-while-acting, where the cumulative regret is of interest and exploration must be intertwined with exploitation. For this setup, an action selection strategy called UCB1 was shown to attain the optimal logarithmic cumulative regret by balancing the empirical attractiveness of the actions with the potential of less sampled actions. Specifically, UCB1 samples each action once, and then iteratively selects actions as

$$\operatorname*{argmax}_a \left[\widehat{\mu}_a + \alpha\sqrt{\frac{\log n}{n_a}}\right],$$

where n is the total number of samples so far, n_a is the number of samples that went to action a, and $\widehat{\mu}_a$ is the average reward of these samples of a. The parameter α is an exploration factor that balances the two components of the UCB1 formula.

In contrast to learning-while-acting, in online planning for MAB the agent is provided with a simulator that can be used "free of charge" to evaluate the alternative actions by drawing samples from their reward distributions. An algorithm for online planning for MAB is defined by an exploration strategy, used to sample the actions, and a recommendation strategy, used at the end of the planning to select an action that is believed to minimize simple regret. Recently, Bubeck et al. (2010) investigated worst-case convergence-rate guarantees provided by various MAB planning algorithms; their key findings are depicted in Table 1. Two exploration strategies, the uniform one and a generalization of UCB1, have been examined in the context of "the empirical best action" (EBA) and "the most played action" (MPA) recommendation strategies. The table provides upper bounds on the expected simple regret of the considered pairs of exploration (rows) and recommendation (columns) strategies, whereas the $\bigcirc$ symbols are distribution-dependent constants. Bubeck et al. (2010) also examined these algorithms empirically, and showed that, in line with the details of their formal analysis, the UCB1-based exploration strategy outperforms uniform+EBA under a moderate number of samples, while the opposite holds under more permissive exploration budgets.

	EBA	MPA
uniform	$\bigcirc e^{-\bigcirc n}$	—
UCB(α)	$\bigcirc n^{-\bigcirc}$	$\bigcirc n^{-2(\alpha-1)}$

Table 1: Upper bounds on the expected simple regret of some online planning algorithms for MAB (Bubeck and Munos 2010)

Monte-Carlo Tree Search. MCTS, a canonical scheme underlying various MCTS algorithms for online MDP planning, is depicted in Figure 1. MCTS explores the state space in the radius of H steps from the initial state s_0 by iteratively issuing simulated rollouts from s_0. Each such rollout ρ comprises a sequence of simulated steps $\langle s, a, r, s'\rangle$, where s is a state, a is an action applicable in s, r is an immediate reward collected from issuing the action a, and s' is the resulting state. In particular, $\rho[0].s = s_0$ and $\rho[t].s' = \rho[t+1].s$ for all t.

Each generated rollout is used to update some variables of interest. These variables typically include at least the action value estimators $\widehat{Q}(s\langle h\rangle, a)$, as well as the counters $n(s\langle h\rangle, a)$ that record the number of times the corresponding estimators $\widehat{Q}(s\langle h\rangle, a)$ have been updated. Instances of MCTS vary mostly along the different implementation of the strategies STOP-ROLLOUT, specifying when to stop a rollout; SELECT-ACTION, prescribing the action to apply in the current state of the rollout; and UPDATE, specifying how a rollout should update the maintained variables.

Once interrupted, MCTS uses the information collected throughout the exploration to recommend an action to perform at state s_0. The rollout-based exploration of MCTS is especially appealing in the setup of online planning because it allows smooth improvement of the intermediate quality of recommendation by propagating to the root information from states at deeper levels in iterations of low complexity of $O(H)$.

MCTS algorithms: UCT, BRUE, and MaxUCT. UCT, one of the most popular algorithms for online MDP planning to date, is depicted in Figure 2 as a particular instantiation of MCTS. In UCT, the rollouts end at terminal states, i.e., at depth H or at states with no applicable actions.[1] Each rollout updates all value estimators $\widehat{Q}(s\langle h\rangle, a)$ of the $(s\langle h\rangle, a)$ pairs encountered along the rollout. The estimators are updated via the MC-BACKUP procedure, which averages the accumulated reward of the rollouts from $s\langle h\rangle$ to terminal states.

[1] In a more popular version of UCT, a rollout ends at a newly encountered node, but this is secondary to our discussion.

MCTS: [input: $\langle S, A, \mathbb{P}, R\rangle$; $s_0 \in S$]

while time permits **do**
  $\rho \leftarrow$ ROLLOUT *// generate rollout*
  UPDATE(ρ)
return $\arg\max_a \widehat{Q}(s_0\langle H\rangle, a)$

procedure ROLLOUT
  $\rho \leftarrow \langle\rangle$; $s \leftarrow s_0$; $t \leftarrow 0$
  while not STOP-ROLLOUT(ρ) **do**
    $a \leftarrow$ SELECT-ACTION(s,t)
    $s' \leftarrow$ SAMPLE-OUTCOME(s,a,t)
    $r \leftarrow R(s,a,s')$
    $\rho[t] \leftarrow \langle s,a,r,s'\rangle$
    $s \leftarrow s'$; $t \leftarrow t+1$
  return ρ

Figure 1: Monte-Carlo tree search

Under this flow, a necessary condition for the value estimators to converge to their true values is that the portion of samples that correspond to selections of optimal actions must tend to 1 as the number of samples increases. At the same time, in order to increase the confidence that the optimal actions will be recognized at the nodes, all the applicable actions must be sampled infinitely often. The sampling strategy of UCT, UCB1, aims at achieving precisely that: UCB1 ensures that each action is selected at least a logarithmic number of times, and that suboptimal actions are selected at most a logarithmic number of times; thus, the proportion of best-action selections indeed tends to 1.

UCT has many success stories and much of this success is accounted for by the exploitative property of UCT. This property results in skewing towards more attractive actions right from the beginning of exploration, a protocol that presumably enables fast homing on "good" actions. However, Table 1 shows that exploitation may considerably slow down the reduction of simple regret over time. Indeed, much like UCB1 for MABs, UCT achieves only polynomial-rate reduction of simple regret over time (Bubeck, Munos, and Stoltz 2011), and the number of samples after which the bounds of UCT on simple regret become meaningful might be as high as hyper-exponential in H (Coquelin and Munos 2007).

Using this observation and following the findings of Bubeck et al. (Bubeck and Munos 2010), in our earlier work we introduced the concept of "separation of concerns," whereby the first part of each rollout is devoted solely to the purpose of selecting particular nodes, whereas the second part is devoted to estimating their value (Feldman and Domshlak 2012). We showed a specific algorithm, BRUE, that implements this concept by always updating value estimators with samples that activate currently best actions only, but the estimators to be updated are chosen by rolling out actions uniformly at random. It turns out that, in contrast to UCT, BRUE achieves an exponential-rate reduction of simple regret over time, with the bounds on simple regret becoming meaningful after only exponential in H^2 number of samples. Moreover, BRUE was also shown to be very effective in practice.

procedure UPDATE(ρ)
  $\bar{r} \leftarrow 0$
  for $d \leftarrow |\rho|, \ldots, 1$ **do**
    $h \leftarrow H - d$
    $a \leftarrow \rho[d].a$
    $n(s\langle h\rangle) \leftarrow n(s\langle h\rangle) + 1$
    $n(s\langle h\rangle, a) \leftarrow n(s\langle h\rangle, a) + 1$
    $\bar{r} \leftarrow \bar{r} + \rho[d].r$
    MC-BACKUP$(s\langle h\rangle, a, \bar{r})$

procedure MC-BACKUP($s\langle h\rangle, a, \bar{r}$)
  $\widehat{Q}(s\langle h\rangle, a) \leftarrow \frac{n(s\langle h\rangle,a)-1}{n(s\langle h\rangle,a)}\widehat{Q}(s\langle h\rangle, a) + \frac{1}{n(s\langle h\rangle,a)}\bar{r}$

procedure STOP-ROLLOUT(ρ)
  $t \leftarrow |\rho|$
  return $t = H$ **or** $A(\rho[t].s') = \emptyset$

procedure SELECT-ACTION(s,d) *// UCB*
  $h \leftarrow H - d$
  if $\exists a : n(s\langle h\rangle, a) = 0$ **then**
    return a
  return $\text{argmax}_a \left[\widehat{Q}(s\langle h\rangle, a) + c\sqrt{\frac{\log n(s\langle h\rangle)}{n(s\langle h\rangle,a)}}\right]$

procedure SAMPLE-OUTCOME(s,a,t)
  return $s' \sim \mathbb{P}(S \mid s,a)$

Figure 2: UCT algorithm as a specific set of sub-routines for MCTS

Finally, BRUE was not the only successful attempt to improve over UCT. In particular, Keller & Helmert (2013) recently introduced MaxUCT, a modification of UCT in which MC backups are replaced with Bellman backups using approximate transition probabilities, and demonstrated that MaxUCT substantially outperforms UCT empirically. In terms of formal guarantees, however, there is no dramatic difference between the convergence rates of the two algorithms.

From MAB to MDP

Relating between online planning for MAB and for more general MDPs, we begin by drawing ties between

(i) MCTS rollout sampling strategies and arm exploration strategies in MAB, and

(ii) MCTS selection of actions used to update search nodes and arm recommendation strategies in MAB.

Considering the three algorithms discussed above in that perspective, the picture appears to be as follows.

- UCT combines MC backups with a rollout sampling driven by the UCB1 action-selection strategy. Interestingly, there is no perfect analogy between UCT and a reasonable algorithm for pure exploration in MAB. This is because, at all nodes but the root, UCB1 *de facto* drives *both* UCT's rollout sampling and node updates, yet recommending an arm in MAB according to UCB1 does not have well justified semantics.

	EBA	MPA	UCB1
Uniform	BRUE, MaxBRUE	—	—
UCB1	MaxUCT	MpaUCT	UCT

Table 2: MCTS algorithms for MDPs through the lens of the MAB exploration topology. Rows are exploration strategies, and columns are recommendation strategies.

- BRUE is analogous to uniform exploration with empirical best action recommendation: Applying the principle of separation of concerns, nodes are reached by selecting actions uniformly, and the samples used in the MC backups are generated by selecting the empirical best actions.
- MaxUCT is analogous to UCB(α) exploration with best empirical action recommendation: With Bellman backups, the updated value corresponds to the value of the action with the best empirical value. Interestingly, this perspective reveals that switching from MC backups to Bellman backups in MaxUCT essentially constitutes another way to separate concerns in the sense discussed above.

Building on this link between online planning for MAB and general MDPs, in what follows we present and analyze two new MCTS algorithms for MDP planning. The union of the known and new algorithms is depicted in Table 2, with the names of the new algorithms underscored.

The first algorithm, MpaUCT, is analogous to UCB(α) exploration with "most played action" recommendations in MAB. This algorithm is inspired by the findings of Bubeck et al. (Bubeck, Munos, and Stoltz 2011) that, unlike all other bounds shown in Table 1, the convergence rate of UCB(α)+MPA planning on MAB can be bounded independently of the problem parameters.

A simple adaptation of the MPA recommendation strategy to MDPs is a modification of the Bellman backup: Instead of folding up the value of the empirically best action, we propagate the value of the action that was updated the most. Ties are broken in favor of actions with better empirical value. The resulting algorithm is depicted in Figure 3, and later on we present our empirical findings with it.

The second algorithm, MaxBRUE, is—like BRUE—analogous to uniform exploration with EBA recommendations, but it employs Bellman backups rather than MC backups. MaxBRUE is depicted in Figure 4. As we show in the proof of Theorem 1 below, not only does MaxBRUE achieve exponential-rate reduction of simple regret similarly to BRUE, but the particular parameters of the convergence bounds are more attractive than those currently known for BRUE. Basically, Theorem 1 positions MaxBRUE as the worst-case most efficient MCTS algorithm for online MDP planning to date.

Theorem 1 *Let* MaxBRUE *be called on a state s_0 of an MDP $\langle S, A, \mathbb{P}, R\rangle$ with rewards in $[0,1]$, and finite horizon H. After $n \geq 1$ iterations of* MaxBRUE*, we have the probability p_{err} of sub-optimal action choice being bounded as $p_{err} \leq \alpha e^{-\beta n}$, and the expected simple regret Δ being bounded as $\Delta \leq H\alpha e^{-\beta n}$, where $\alpha = 3K\,(3BK)^H$, $\beta = \frac{\varepsilon^2}{4K(4BK)^H H^2}$, and ε is the simple regret of the second-best action at $s_0\langle H\rangle$.*

procedure UPDATE(ρ)
  for $d \leftarrow |\rho|, \ldots, 1$ **do**
    $h \leftarrow H - d$
    $a \leftarrow \rho[d].a$
    $s' \leftarrow \rho[d].s'$
    $n(s\langle h\rangle) \leftarrow n(s\langle h\rangle) + 1$
    $n(s\langle h\rangle, a) \leftarrow n(s\langle h\rangle, a) + 1$
    $n(s\langle h\rangle, a, s') \leftarrow n(s\langle h\rangle, a, s') + 1$
    $\widehat{R}(s\langle h\rangle, a) = \widehat{R}(s\langle h\rangle, a) + \rho[d].r$
    MPA-BACKUP($s\langle h\rangle, a$)

procedure MPA-BACKUP($s\langle h\rangle, a$)
  $\widehat{Q}(s\langle h\rangle, a) \leftarrow \frac{\widehat{R}(s\langle h\rangle, a)}{n(s\langle h\rangle, a)}$
  $v \leftarrow 0$
  for $s' \in \{s' \mid n(s\langle h\rangle, a, s') > 0\}$ **do**
    $A \leftarrow \operatorname{argmax}_{a'} n(s'\langle h-1\rangle, a')$
    $a^* \leftarrow \operatorname{argmax}_{a' \in A} \widehat{Q}(s'\langle h-1\rangle, a')$
    $v \leftarrow v + \frac{n(s\langle h\rangle, a, s')}{n(s\langle h\rangle, a)} \widehat{Q}(s'\langle h-1\rangle, a^*)$
  $\widehat{Q}(s\langle h\rangle, a) \leftarrow \widehat{Q}(s\langle h\rangle, a) + v$

Figure 3: MpaUCT as UCT with a modified UPDATE procedure

Proof: A key sub-claim we prove first is that, at any iteration of the algorithm, for all $h \in [\![H]\!]$, all states s reachable from s_0 in $H - h$ steps, all actions $a \in A(s)$, and any $\delta > 0$, it holds that

$$\mathbb{P}\left\{\left|\widehat{Q}\left(s\langle h\rangle, a\right) - Q\left(s\langle h\rangle, a\right)\right| \geq \delta\right\} \leq 2\,(3KB)^h\, e^{-\frac{2\delta^2 n(s\langle h\rangle, a)}{(4BK)^h h^2}}. \quad (1)$$

The proof of this sub-claim is by induction on h. Starting with $h = 1$, by Hoeffding concentration inequality, we have that

$$\mathbb{P}\left\{\left|\widehat{Q}\left(s\langle 1\rangle, a\right) - Q\left(s\langle 1\rangle, a\right)\right| \geq \delta\right\} \leq 2e^{-2\delta^2 n(s\langle 1\rangle, a)}.$$

Now, assuming Eq. 1 holds for $h' \leq h$, we prove it holds for $h + 1$. From the induction hypothesis, we have

$$\begin{aligned}
&\mathbb{P}\left\{\left|\widehat{Q}\left(s\langle h\rangle, a\right) - Q\left(s\langle h\rangle, a\right)\right| \geq \delta\right\} \\
&\leq \mathbb{P}\left\{n(s\langle h\rangle, a) \leq \frac{n(s\langle h\rangle)}{2K}\right\} + \\
&\quad \mathbb{P}\left\{\left|\widehat{Q}\left(s\langle h\rangle, a\right) - Q\left(s\langle h\rangle, a\right)\right| \geq \delta \;\middle|\; n(s\langle h\rangle, a) > \frac{n(s\langle h\rangle)}{2K}\right\} \\
&\leq e^{-\frac{n(s\langle h\rangle)}{2K^2}} + 2\,(3BK)^h\, e^{-\frac{2\delta^2 n(s\langle h\rangle)}{2K(4BK)^h h^2}} \\
&\leq \left(1 + 2\,(3BK)^h\right) e^{-\frac{\delta^2 n(s\langle h\rangle)}{K(4BK)^h h^2}},
\end{aligned}$$

which implies

$$\begin{aligned}
&\mathbb{P}\left\{\left|\max_a \widehat{Q}\left(s\langle h\rangle, a\right) - Q\left(s\langle h\rangle, \pi^*(s)\right)\right| \geq \delta\right\} \\
&\leq K\left(1 + 2\,(3BK)^h\right) e^{-\frac{\delta^2 n(s\langle h\rangle)}{K(4BK)^h h^2}}.
\end{aligned} \quad (2)$$

procedure UPDATE(ρ)
  for $d \leftarrow |\rho|, \ldots, 1$ **do**
    $h \leftarrow H - d$
    $a \leftarrow \rho[d].a$
    $s' \leftarrow \rho[d].s'$
    $n(s\langle h\rangle) \leftarrow n(s\langle h\rangle) + 1$
    $n(s\langle h\rangle, a) \leftarrow n(s\langle h\rangle, a) + 1$
    $n(s\langle h\rangle, a, s') \leftarrow n(s\langle h\rangle, a, s') + 1$
    $\widehat{R}(s\langle h\rangle, a) = \widehat{R}(s\langle h\rangle, a) + \rho[d].r$
    BELLMAN-BACKUP($s\langle h\rangle, a$)

procedure BELLMAN-BACKUP($s\langle h\rangle, a$)
  $\widehat{Q}(s\langle h\rangle, a) \leftarrow \frac{\widehat{R}(s\langle h\rangle, a)}{n(s\langle h\rangle, a)}$
  $\upsilon \leftarrow 0$
  for $s' \in \{s' \mid n(s\langle h\rangle, a, s') > 0\}$ **do**
    $\upsilon \leftarrow \upsilon + \frac{n(s\langle h\rangle, a, s')}{n(s\langle h\rangle, a)} \max_{a'} \widehat{Q}(s'\langle h-1\rangle, a')$
  $\widehat{Q}(s\langle h\rangle, a) \leftarrow \widehat{Q}(s\langle h\rangle, a) + \upsilon$

procedure STOP-ROLLOUT(ρ)
  $t \leftarrow |\rho|$
  return $t = H$ **or** $A(\rho[t].s') = \emptyset$

procedure SELECT-ACTION(s,t) *// uniform*
  return $a \sim \mathcal{U}[A(s)]$

procedure SAMPLE-OUTCOME(s,a,t)
  return $s' \sim \mathbb{P}(S \mid s, a)$

Figure 4: MaxBRUE algorithm as a specific set of subroutines for MCTS

Denoting $\widehat{p}_{s,a,s'} = \frac{n(s\langle h+1\rangle, a, s')}{n(s\langle h+1\rangle, a)}$ and $\widehat{p}^{\text{diff}}_{s,a,s'} = \widehat{p}_{s,a,s'} - \mathbb{P}(s'|s,a)$, it holds that

$$
\begin{aligned}
&\left|\widehat{Q}\left(s\langle h+1\rangle, a\right) - Q\left(s\langle h+1\rangle, a\right)\right| \\
&= \left|\sum_{s'} \widehat{p}_{s,a,s'} \left[R\left(s,a,s'\right) + \max_{a'} \widehat{Q}\left(s'\langle h\rangle, a'\right)\right]\right. \\
&\quad \left. - \sum_{s'} \mathbb{P}(s'|s,a) \left[R\left(s,a,s'\right) + Q\left(s'\langle h\rangle, \pi^*(s')\right)\right]\right| \\
&\leq \left|\sum_{s'} \left(\widehat{p}_{s,a,s'} - \mathbb{P}(s'|s,a)\right) \left[R\left(s,a,s'\right) + Q\left(s'\langle h\rangle, \pi^*(s')\right)\right]\right| \\
&+ \left|\sum_{s'} \widehat{p}_{s,a,s'} \left|\max_{a'} \widehat{Q}\left(s'\langle h\rangle, a'\right) - Q\left(s'\langle h\rangle, \pi^*(s')\right)\right|\right|,
\end{aligned}
$$

and thus

$$
\begin{aligned}
&\mathbb{P}\left\{\left|\widehat{Q}\left(s\langle h+1\rangle, a\right) - Q\left(s\langle h+1\rangle, a\right)\right| \geq \delta\right\} \\
&\leq \mathbb{P}\left\{\left|\sum_{s'} \widehat{p}^{\text{diff}}_{s,a,s'} \left[R\left(s,a,s'\right) + Q\left(s'\langle h\rangle, \pi^*(s')\right)\right]\right| \geq \frac{\delta}{2}\right\} \\
&+ \mathbb{P}\left\{\sum_{s'} \widehat{p}_{s,a,s'} \left|\max_{a'} \widehat{Q}\left(s'\langle h\rangle, a'\right) - Q\left(s'\langle h\rangle, \pi^*(s')\right)\right| \geq \frac{\delta}{2}\right\} \\
&\leq 2e^{-\frac{\delta^2 n(s\langle h+1\rangle, a)}{2(h+1)^2}} \\
&+ \sum_{s'} \mathbb{P}\left\{\left|\max_{a'} \widehat{Q}\left(s'\langle h\rangle, a'\right) - Q\left(s'\langle h\rangle, \pi^*(s')\right)\right| \geq \frac{\delta}{2\sqrt{B\widehat{p}_{s,a,s'}}}\right\}.
\end{aligned} \tag{3}
$$

In the last bounding of Eq. 3, the first term is due to Hoeffding and the second term is justified by noting that the solution to the problem

$$
\underset{p}{\text{maximize}} \quad \sum_{i=1}^{B} \sqrt{cp_i} \quad \text{subject to} \quad \sum_{i=1}^{B} p_i = 1
$$

is $p = \left(\frac{1}{B}, \ldots, \frac{1}{B}\right)$, with value $\sum_{i=1}^{B} \sqrt{\frac{c}{B}} = \sqrt{cB}$. From Eq. 2 we have

$$
\begin{aligned}
&\sum_{s'} \mathbb{P}\left\{\left|\max_{a'} \widehat{Q}\left(s'\langle h\rangle, a'\right) - Q\left(s'\langle h\rangle, \pi^*(s')\right)\right| \geq \frac{\delta}{2\sqrt{B\widehat{p}_{s,a,s'}}}\right\} \\
&\leq \sum_{s'} K\left(1 + 2\left(3BK\right)^h\right) e^{-\frac{\delta^2 n(s'\langle h\rangle)}{(4BK)^{h+1} h^2 \widehat{p}_{s,a,s'}}} \\
&\leq BK\left(1 + 2\left(3BK\right)^h\right) e^{-\frac{\delta^2 n(s\langle h+1\rangle, a)}{(4BK)^{h+1} h^2}}.
\end{aligned} \tag{4}
$$

Returning now to Eq. 3, we have

$$
\begin{aligned}
&\mathbb{P}\left\{\left|\widehat{Q}\left(s\langle h+1\rangle, a\right) - Q\left(s\langle h+1\rangle, a\right)\right| \geq \delta\right\} \\
&\leq 2e^{-\frac{\delta^2 n(s\langle h+1\rangle, a)}{2(h+1)^2}} + BK\left(1 + 2(3BK)^h\right) e^{-\frac{\delta^2 n(s\langle h+1\rangle, a)}{(4BK)^{h+1} h^2}} \\
&\leq (3BK)^{h+1} e^{-\frac{\delta^2 n(s\langle h+1\rangle, a)}{(4BK)^{h+1}(h+1)^2}},
\end{aligned} \tag{5}
$$

finalizing the proof of the induction step. Now, given the subclaim around Eq. 1 and denoting $Q^{\text{diff}}_a = |\widehat{Q}(s_0\langle H\rangle, a) - Q(s_0\langle H\rangle, a)|$, the first part of Theorem 1 is concluded by

$$
\begin{aligned}
p_{\text{err}} &\leq \sum_a \mathbb{P}\left\{Q^{\text{diff}}_a \geq \frac{\varepsilon}{2}\right\} \\
&\leq \sum_a \left[\mathbb{P}\left\{n(s_0\langle H\rangle, a) \leq \frac{n(s_0\langle H\rangle)}{2K}\right\} + \right. \\
&\qquad \left. \mathbb{P}\left\{Q^{\text{diff}}_a \geq \frac{\varepsilon}{2} \;\middle|\; n(s_0\langle H\rangle, a) > \frac{n}{2K}\right\}\right] \\
&\leq \sum_a \left[e^{-\frac{n}{2K^2}} + 2\left(3BK\right)^H e^{-\frac{\varepsilon^2 n}{4K(4BK)^H H^2}}\right] \\
&\leq 3K\left(3BK\right)^H e^{-\frac{\varepsilon^2 n}{4K(4BK)^H H^2}}
\end{aligned} \tag{6}
$$

The second part of Theorem 1 is obtained from the fact that the maximal loss from choosing a sub-optimal action at $s_0\langle H\rangle$ is H. ∎

MaxBRUE+: Controlled Exploration

At any non-terminal node $s\langle h\rangle$, the inaccuracy of the Q-value estimators in Bellman-based MaxUCT and MaxBRUE stems from two sources: The inaccuracy in the estimation of the action transition probabilities, and the inaccuracy of the value estimators that are folded up to $s\langle h\rangle$ from its immediate successors. The former is reduced with sampling actions at $s\langle h\rangle$, and the latter is reduced with sampling $s\langle h\rangle$'s successors.

If we wish to optimize the formal guarantees on the convergence rate, it is ideal to have these two sources of inaccuracy balanced. This can be achieved by equating the number of samples of the node with the number of samples of its immediate successors. In fact, this is precisely what is done by the seminal sparse sampling algorithm of Kerns et al. (2002) for PAC (probably approximately correct) MDP planning. However, the PAC setting does not require smooth improvement of the quality of recommendation over time, and thus the nodes can be sampled in a systematic manner. In contrast, in the online setup, smooth convergence is critical, and rollout-based exploration seems to serve this purpose quite effectively. At the same time, rollout-based exploration of MCTS leads to unbalanced node sampling.

While we might see unbalanced node sampling as an inevitable cost for a justified cause, this is not entirely so. The overall convergence rate with rollout-based exploration is dictated by the accuracy of the estimates at the nodes that are farther towards the horizon from $s_0\langle H\rangle$. Due to branching, these nodes are expected to be sampled less frequently. However, since the search spaces induced by MDPs most typically form DAGs (and not just trees), a node with multiple ancestors might possibly be sampled more than each of its ancestors. In terms of the formal guarantees, the latter type of imbalance is worthless, and samples are better be diverted to nodes that are sampled less than their immediate ancestors.

A rather natural way to control this type of imbalance is by modifying the protocol for stopping a rollout. Given the last sample $\langle s, a, r, s'\rangle$ along the ongoing rollout, if the number of samples $n(s\langle h\rangle, a)$ is smaller than the number of samples $n(s'\langle h-1\rangle, a')$ of some action $a' \in A(s')$ applicable at the resulting state s', the rollout is stopped. Supported by the formal analysis of MaxBRUE in the proof of Theorem 1, we slightly modify the latter condition as follows:

- We replace the requirement depicted above with a weaker one whereby the overall number of updates $n\left(s'\langle h-1\rangle\right)$ of the resulting state s' is at least K times larger than $n\left(s\langle h\rangle, a\right)$.
- We multiply the counter $n\left(s\langle h\rangle, a\right)$ by $B \cdot \mathbb{P}(s'|a, s')$. If $\mathbb{P}(S|a, s')$ induces a uniform distribution over the plausible outcomes of a at s, this modification changes nothing. At the same time, for more/less probable outcomes s', this modification implies a stronger/weaker condition, respectively.

The substitution of the sub-procedure STOP-PROBE of MaxBRUE with that depicted in Figure 5 constitutes the stratified algorithm MaxBRUE+.

Note that, in the proof of Theorem 1, we make use of the fact that $n(s'\langle h-1\rangle) \geq n(s\langle h\rangle, a, s')$ to replace the former with the latter in the bounding in Eq. 4. Therefore, enforcing the more conservative $n(s'\langle h-1\rangle) \leq K \cdot |S(s,a)| \cdot n(s\langle h\rangle, a, s')$ does not affect the bound. At the same time, it is easy to see that (i) the two sources of inaccuracy are balanced when $n(s'\langle h-1\rangle) = K \cdot |S(s,a)| \cdot n(s\langle h\rangle, a, s')$, and (ii) beyond this point, the node accuracy is surpassed by the accuracy of its children.

procedure STOP-ROLLOUT(ρ)
  $d \leftarrow |\rho|$
  $h \leftarrow H - d$
  $s \leftarrow \rho[d].s$
  $a \leftarrow \rho[d].a$
  $s' \leftarrow \rho[d].s'$
  $S(s,a) \leftarrow \{s' \mid n(s\langle h\rangle, a, s') > 0\}$
  if $n\left(s'\langle h-1\rangle\right) > K \cdot |S(s,a)| \cdot n\left(s\langle h\rangle, a, s'\right)$ **then**
    return true
  return ($d = H$ **or** $A(\rho[d].s') = \emptyset$)

Figure 5: MaxBRUE+ as MaxBRUE with a modified STOP-PROBE procedure

Experimental Evaluation

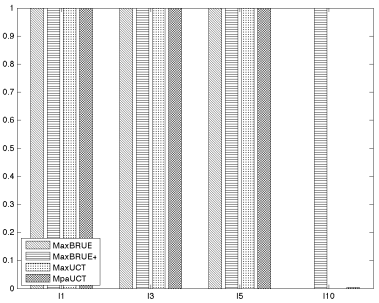
Our empirical study comprises two sets of experiments, comparing four algorithms: MaxUCT (Keller and Helmert 2013), MpaUCT, MaxBRUE, and MaxBRUE+. In the first set, we evaluate the four algorithms in terms of their reduction of simple regret in the $Sailing$ domain (Péret and Garcia 2004). In this domain, a sailboat navigates to a goal location on an 8-connected grid, under fluctuating wind conditions. At a high level, the goal is to reach a concrete destination as quickly as possible, by choosing at each grid location a neighbor location to move to. The duration of each such move depends on the direction of the move (*ceteris paribus*, diagonal moves take $\sqrt{2}$ more time than straight moves), the direction of the wind relative to the sailing direction (the sailboat cannot sail against the wind and moves fastest with a tail wind), and the tack. In Figure 6, we plot the empirical simple regret for increasing deliberation times for two grid sizes, 20×20 and 40×40, averaged over 2000 runs with varying origin and goal locations. It is interesting to see that MaxBRUE clearly dominates MaxUCT (and MpaUCT), right from the beginning, unlike the case of BRUE and UCT whereby UCT performs better until some point.[2] Figure 6 also demonstrates the benefit of the exploration control of MaxBRUE+, as well as the benefit of using the MPA-backup of MpaUCT.

The second set of experiments compares between the empirical reward collected by the four algorithms on five IPPC-2011 domains, *Game-of-Life*, *SysAdmin*, *Traffic*, *Crossing*, and *Navigation*. (Almost all of the tasks in these domains are simply too large for us to examine simple regret.) From each domain, we chose four tasks, I_1, I_3, I_5, I_{10}, with higher indexes corresponding to tasks of higher complexity. Each algorithm was given a deliberation budget that decreased linearly from 10 seconds in the first step to 1 second in the

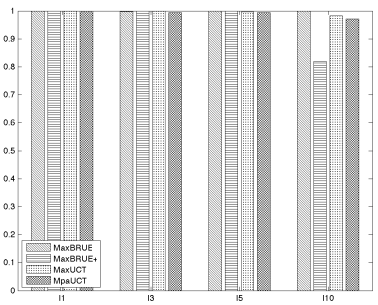
[2]For simple regret analysis on the *Sailing* domain with longer deliberation times, we refer the reader to Feldman & Domshlak (2013).

	K	B	H
Sailing	8	3	$4p$
Navigation	5	2	40
Crossing	5	2^p	40
SysAdmin	p	2^p	10
Game-of-Life	p^2	2^{p^2}	10
Traffic	2^{p^2}	2^{2p}	10

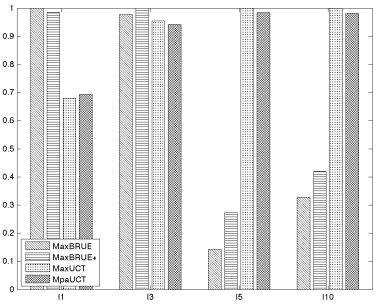
(a) Domain parameters



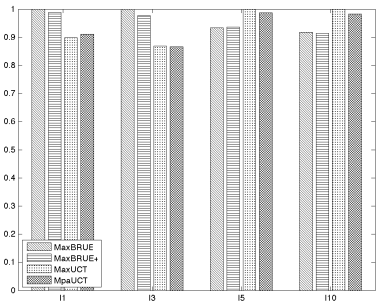
(b) *Navigation*



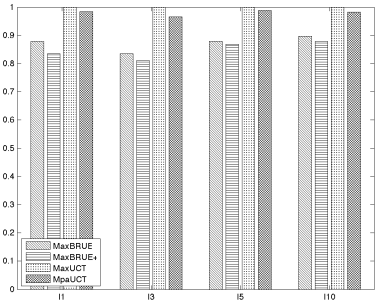
(c) *Crossing*



(d) *SysAdmin*



(e) *Game-of-Life*



(f) *Traffic*

Figure 7: (a) Structural and experimental parameters of the $Sailing$ and IPPC-2011 domains, and (b-f) scores for the different MCTS algorithms as their normalized improvement over a trivial baseline

last step. For each domain, Figure 7(a) depicts the general bound on the state branching factor K and the action branching factor B, as well as the specific horizon H used in the experiments, all as functions of a domain-specific parameter p that scales linearly with the instance index. ($H > 10$ was used in goal-oriented domains.)

Figures 7(b-f) plot the score of the four algorithms based on 700 samples, normalized between 0 and 1 as their average improvement over a trivial baseline of random action selection. With the exception of the *SysAdmin* domain[3], it appears that the results are quite similar for all algorithms. However, the relative performance differences seem to comply with the analysis of Bubeck et al. (2010) for online planning in MABs. Specifically, Bubeck et al. (2010) observed that, despite the superior convergence rate of uniform sampling in general, the bounds provided by the UCB(α)-based exploration can be more attractive if the sample allowance is not permissive enough with respect to the number of actions K. In case of more general MDPs, the "structural complexity" of the problem is determined not only by K, but also by the action branching factor B and the horizon H. In that respect, considering the "structural complexity" of the domains depicted in Figure 7(a), the results in Figures 7(b-f) are in line with the relative pros and cons of the uniform and UCB(α) explorations.

[3]At this stage, the dramatically superior performance of MaxBRUE+ instance I_10 of the *Navigation* domain should be considered a positive anomaly, and not given any deep generalizing explanations.

- In *Sailing*, *Navigation*, and *Crossing*, both K and B grow reasonably slowly with the size of the problem. This makes our fixed time budget reasonably permissive (and thus gives the advantage to uniform exploration) across the instances.
- In the domains of intermediate structural complexity, *Game-of-Life* and *SysAdmin*, the same time budget appears to be reasonably permissive on the smaller instances, giving an advantage to uniform exploration, but then the instances grow rather fast, giving an advantage to the UCB1-based algorithms.
- In *Traffic*, both K and B grow exponentially fast with the size of the problem, making the time budget we fixed to be too small for the uniform exploration to shine even on the smallest instance.

Summary

By drawing ties between online planning in MDPs and MABs, we have shown that the state-of-art MC planning algorithms UCT, BRUE and MaxUCT, as well as the two newly introduced MaxBRUE and MpaUCT, borrow the theoretical and empirical properties of their MAB counterparts. In particular, we have proven that the exponential convergence of uniform exploration with recommendation of the empirically best action in MABs applies also to MaxBRUE, resulting in the best known convergence rates among online MCTS algorithms to date. Moreover, in line with MAB results, the superiority of MaxBRUE with a *permissive budget* as well as the superiority of the UCB(α)-based exploration

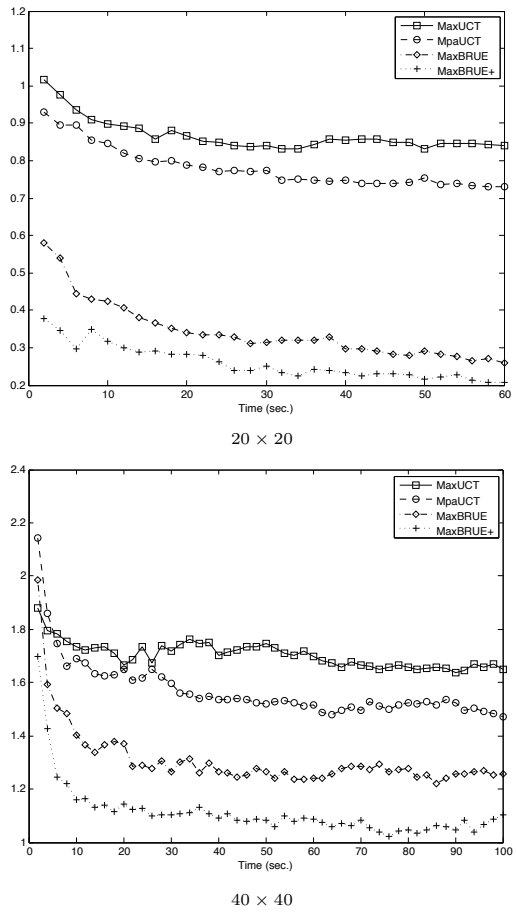


Figure 6: Simple regret reduction over time in the *Sailing* domain by different MCTS algorithms

algorithms MaxUCT and MpaUCT with a *moderate budget* has been demonstrated empirically. We have also shown that a particular exploration control mechanism applied to MaxBRUE substantially improves its performance. We believe that this mechanism and variations of it can be valuable to other online planning algorithms as well. Finally, other exploration strategies that are found appealing in the context of MABs can also be "converted" to MDPs following the lines of this work.

Acknowledgements This work was partially supported by the EOARD grant FA8655-12-1-2096, and the ISF grant 1045/12.

References

Balla, R., and Fern, A. 2009. UCT for tactical assault planning in real-time strategy games. In *IJCAI*, 40–45.

Bjarnason, R.; Fern, A.; and Tadepalli, P. 2009. Lower bounding Klondike Solitaire with Monte-Carlo planning. In *ICAPS*.

Bonet, B., and Geffner, H. 2012. Action selection for MDPs: Anytime AO* vs. UCT. In *AAAI*.

Browne, C.; Powley, E. J.; Whitehouse, D.; Lucas, S. M.; Cowling, P. I.; Rohlfshagen, P.; Tavener, S.; Perez, D.; Samothrakis, S.; and Colton, S. 2012. A survey of Monte-Carlo tree search methods. *IEEE Trans. on Comp. Intell. and AI in Games* 143.

Bubeck, S., and Munos, R. 2010. Open loop optimistic planning. In *COLT*, 477–489.

Bubeck, S.; Munos, R.; and Stoltz, G. 2011. Pure exploration in finitely-armed and continuous-armed bandits. *Theor. Comput. Sci.* 412(19):1832–1852.

Busoniu, L., and Munos, R. 2012. Optimistic planning for Markov decision processes. In *AISTATS*, number 22 in JMLR (Proceedings Track), 182–189.

Cazenave, T. 2009. Nested Monte-Carlo search. In *IJCAI*, 456–461.

Coquelin, P.-A., and Munos, R. 2007. Bandit algorithms for tree search. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI)*, 67–74.

Eyerich, P.; Keller, T.; and Helmert, M. 2010. High-quality policies for the Canadian Traveler's problem. In *AAAI*.

Feldman, Z., and Domshlak, C. 2012. Simple regret optimization in online planning for Markov decision processes. *CoRR* arXiv:1206.3382v2 [cs.AI].

Feldman, Z., and Domshlak, C. 2013. Monte-Carlo planning: Theoretically fast convergence meets practical efficiency. In *UAI*.

Gelly, S., and Silver, D. 2011. Monte-Carlo tree search and rapid action value estimation in computer Go. *AIJ* 175(11):1856–1875.

Kearns, M. J.; Mansour, Y.; and Ng, A. Y. 2002. A sparse sampling algorithm for near-optimal planning in large Markov decision processes. *Machine Learning* 49(2-3):193–208.

Keller, T., and Helmert, M. 2013. Trial-based heuristic tree search for finite horizon MDPs. In *ICAPS*, 135–143.

Kocsis, L., and Szepesvári, C. 2006. Bandit based Monte-Carlo planning. In *ECML*, 282–293.

Kolobov, A.; Mausam; and Weld, D. 2012. LRTDP vs. UCT for online probabilistic planning. In *AAAI*.

Péret, L., and Garcia, F. 2004. On-line search for solving Markov decision processes via heuristic sampling. In *ECAI*, 530–534.

Puterman, M. 1994. *Markov Decision Processes*. Wiley.

Robbins, H. 1952. Some aspects of the sequential design of experiments. *Bull. Amer. Math. Soc.* 58(5):527535.

Rosin, C. D. 2011. Nested rollout policy adaptation for Monte Carlo tree search. In *IJCAI*, 649–654.

Sturtevant, N. 2008. An analysis of UCT in multi-player games. In *CCG*, 3749.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

Tolpin, D., and Shimony, S. E. 2012. MCTS based on simple regret. In *AAAI*.