Metareasoning for Monte Carlo Tree Search

Nicholas Hay
Stuart J. Russell

Electrical Engineering and Computer Sciences
University of California at Berkeley

Technical Report No. UCB/EECS-2011-119
http://www.eecs.berkeley.edu/Pubs/TechRpts/2011/EECS-2011-119.html

November 20, 2011

Metareasoning for Monte Carlo Tree Search

Nicholas Hay Stuart Russell

November 20, 2011

Abstract

Sequential decision problems are often approximately solvable by simulating possible future action sequences; such methods are a staple of game-playing algorithms, robot path planners, model-predictive control systems, and logistical planners in operations research. Since the 1960s, researchers have sought effective *metareasoning* methods for selecting *which* action sequences to simulate, basing their approach on some estimate of the expected improvement in decision quality resulting from any particular simulation. Recently, this approach has been applied successfully in the context of *Monte Carlo tree search*, where each simulation takes the form of a randomized sequence of actions leading to a terminal state. In particular, the UCT algorithm borrows asymptotically optimal selection rules from the theory of bandit problems and has led to a new generation of master-level Go programs such as MoGo. We argue that, despite this success, the bandit framework is inappropriate as a basis for selecting computations. We propose instead a theoretical framework for metareasoning that is isomorphic to the statistical framework of *ranking and selection*. In this framework, we describe two apparently distinct conceptual approaches to the forward search metareasoning problem and prove them to be equivalent. We derive a number of basic results applicable to simple Monte Carlo selection problems, including asymptotic regret bounds, and discuss prospects for their extension to combinatorial settings.

Contents

1 Introduction

The broad family of sequential decision problems includes combinatorial search problems, game playing, robotic path planning, model-predictive control problems, Markov decision processes (fully or partially observable), and a huge range of applications. In almost all realistic instances, exact solution is intractable and approximate methods are sought. Perhaps the most popular approach is to simulate a limited number of possible future action sequences. For example, a typical game-playing algorithm explores a tree or graph of action sequences with some limit on depth, using pruning methods to avoid irrelevant subtrees; based on what it finds in the explored portion of the state space, the algorithm then selects a move.

Clearly, it is desirable to select the best possible move from the least possible amount of exploration. For a given amount of exploration, decision quality can be improved by directing exploration towards those actions sequences whose outcomes are helpful in selecting a good move. Thus, the *metalevel* decision problem is to choose what future action sequences to explore (or, more generally, what deliberative computations to do), while the *object-level* decision problem is to choose an action to execute in the real world.

That the metalevel decision problem can itself be formulated and solved decision-theoretically was noted by Matheson (1968), borrowing directly from the related concept of *information value theory* (Howard, 1966). In essence, computations can be selected according to the expected improvement in decision quality re-

sulting from their execution. I. J. Good (1968) independently proposed using this idea to control search in chess, and later defined "Type II rationality" to refer to agents that optimally solve the metalevel decision problem before acting. As interest in probabilistic and decision-theoretic approaches in AI grew during the 1980s, several authors explored these ideas further (Dean and Boddy, 1988; Doyle, 1988; Fehling and Breese, 1988; Horvitz, 1987). Work by Russell and Wefald (2008, 1988, 1989, 1991); andd Eric Wefald (1991) formulated the metalevel sequential decision problem, employing an explicit model of the results of computational actions, and applied this to the control of game-playing search in Othello with encouraging results.

An independent thread of research on metalevel control began with work by Kocsis and Szepesvári (2006) on the UCT algorithm, which operates in the context of *Monte Carlo tree search* (MCTS) algorithms. In MCTS, each computation takes the form of a simulating a randomized sequence of actions leading from a leaf of the current tree to a terminal state. UCT is primarily a method for selecting a leaf from which to conduct the next simulation. It forms the core of the successful MoGo algorithm for Go playing (Gelly and Silver, 2011). It is situated within the theory of bandit problems (Berry and Fristedt, 1985), applying the bandit algorithm UCB1 (Auer et al., 2002) recursively to select actions to perform within simulations. Kocsis and Szepesvári (2006) show UCT's estimates of the utility of the best action converges at rate $O(\frac{\log n}{n})$ in the number of simulations n, and that the probability of simulating a suboptimal action at the root converges to zero polynomially.

We argue in this technical report that the problem of choosing the next simulation computation in MCTS is, like all metalevel decision problems, fundamentally distinct from the class of bandit problems:

- The goal of any metareasoning process is to choose simulations in order to maximize the utility of the action *eventually* taken (less the cost of time used to find it). The goal is not to avoid simulating suboptimal actions; indeed, it may be useful to simulate suboptimal actions to eliminate them.
- The goal is not even to accurately estimate action utilities, for, if the metareasoning process has determined one action is likely to be significantly better than the rest, it will stop and take it rather than trying to determine exactly how much better it is.
- Bandit algorithms pay a higher cost for choosing a worse bandit arm to sample—in medical trials, for example, there is a real cost to trying an ineffective or toxic drug on a patient. In metareasoning, on the other hand, the costs of simulating good and bad moves are the same, regardless of move quality. Thus, one expects UCT to be inappropriately biased away from exploring actions whose current utility estimates are low.
- Bandit algorithms are designed to optimize over infinite sequences of trials, with no notion of "stopping." The metareasoning problem requires trading

off the expected gains from computation against the cost of time, leading to a natural stopping rule.

In order to develop better alternatives to UCT it pays to have a theoretical understanding of MCTS as a metareasoning problem. We develop such an understanding in the following, then sketch prospects for improved algorithms for MCTS.

Section 2 sets the scene, precisely specifying MCTS and UCT, then outlining some problematic aspects of UCT.

Section 3 presents a general theory of probabilistic metareasoning and derives general results. We give two equivalent ways to define a metalevel decision problem, explore the properties of myopic policies and their relation to optimal policies, and consider how adding context affects the optimal policy.

Section 4 gives results specific to metareasoning via Monte Carlo methods, i.e., by drawing noisy samples of action values. We consider two sampling models, Bernoulli and normal, and consider myopic policies and optimal for solving them. We compare these polices to the UCB algorithm. We proceed give some regret bounds of our own for the Monte Carlo sampling problem.

Section 5 sketches how the above results can be applied to the design of improved MCTS algorithms.

2 Monte Carlo tree search

2.1 Ranking and selection versus bandits

Suppose there are k actions available. Each action's consequences are uncertain, but an accurate generative model is available to stochastically simulate the consequences of any action at a cost. Eventually one of the actions must be selected for actual execution. Which strategy for sequentially simulating actions maximizes the expected net utility of the actual consequences less the cost of simulation?

Problems of this form have been studied under various headings. In statistics, *ranking and selection* problems have been studied since the 1950s; Frazier (2009) provides a useful survey. In machine learning, Madani et al. (2004) studied *budgeted learning* problems. Here we'll call them ranking and selection problems, although it is important to note that the emphasis in statistics is on ranking populations based on samples, with no underlying sequential decision problem.

Bandit problems (Robbins, 1952; Berry and Fristedt, 1985) are similar but importantly different. In a bandit problem there are k actions, one of which can be taken at each time step; each action's reward at each time step is drawn from a fixed distribution for that action and is independent of actions taken

previously. The canonical example is a medical trial: given k different treatments, how should they be allocated to each of a line of patients to maximize the expected total number patients successfully treated? On the one hand, it is good to experiment to learn which treatments are best, but on the other hand, these are real patients so it is good to give them the best known treatment. This conflict is an instance of the exploration vs. exploitation tradeoff that is also seen in reinforcement learning (Sutton and Barto, 1998).

The essential difference between these two problems is that ranking and selection algorithms simulate whereas bandit algorithms act. All other things equal, in a bandit problem it is worse to take a bad action than a good one. In a ranking and selection problem, on the other hand, both are equally good. Put differently, ranking and selection problems are pure exploration.

For example, one can show quite generally in bandits that actions with higher means and higher uncertainty are preferred (Yu, 2011). However, in the two-action, normally distributed ranking and selection problem (see section 4.4) the optimal policy selects the action with highest uncertainty, regardless of its mean (Frazier, 2009).

2.2 Monte Carlo tree search

Consider the problem of deciding which action to take in a particular state of a Markov Decision Process (MDP) given a generative model of that MDP. This is similar to the ranking and selection problem in the previous section, except the consequences of choosing an action are not immediate but depend on further action choices the agent will make in the future. The generative model can be used to perform Monte Carlo simulations of the consequences of each action, and these simulations can be integrated together into a lookahead tree rooted at the present state. *Monte Carlo tree search* (MCTS) must decide how best to perform Monte Carlo simulations to build up such a tree, and how to use it to decide which action to take.

To make this more precise, a generic framework for MCTS is given in Algorithm 1. MCTS performs a sequence of simulations of potential futures from the given state, building up a tree of all states visited in these simulations to record the simulations' results. Each node represents a state and keeps track of certain statistics (updated by UpdateNode). Until a terminal state is reached, actions are selected to be simulated in some fashion by ChooseAction. SampleNextNode then uses the generative model to sample a new state and reward, returning the node corresponding to the new state (created if this state hasn't yet been visited) with the reward. Finally, where Finished decides to stop simulating, BestAction returns the apparently best action at the root, given the information in the tree.

Specific MCTS algorithms specify the workings of UpdateNode, ChooseAction, BestAction, and Finished.

Algorithm 1 Monte Carlo Tree Search

```
function MCTS(state)
    root = NewNode(state=state)
    while not Finished(root) do
        Simulate(root)
    end while
    return BestAction(root)
end function

function Simulate(node)
    if Terminal(node) return 0
    action ← SelectAction(node)
    (next-node, reward) ← SampleNextNode(node, action)
    q ← reward + γ Simulate(next-node)
    UpdateNode(node, action, next-node, reward, q)
    return q
end function
```

UCT (Kocsis and Szepesvári, 2006) is a special case of Algorithm 1 where:

- UpdateNode keeps track for each state s the number of times n_s that state has been visited in a simulation, the number of times n_{sa} each action a has been tried in that state and the average future reward r_{sa} received in those simulations.
- SelectAction implements the UCB algorithm (Auer et al., 2002) from the theory of multi-armed bandits (Berry and Fristedt, 1985): if there are any untried actions it selects one at random, otherwise selecting the action a that maximizes
$$r_{sa} + k\sqrt{\frac{\log n_s}{n_{sa}}}.$$
The first term favors actions that have performed well so far, the second term favors actions that have been tried fewer times (so their value is more uncertain), and the constant k tunes the balance between these two goals.
- Finished terminates either when a time limit is reached, or after a given number of simulations.
- BestAction returns the action a at the root state s_0 which maximizes r_{s_0a} the average reward of all simulations which tried a as their first action.

As noted in the Introduction, there are a few problematic aspects of UCT:

- Choosing which simulation to perform is analogous to a ranking and selection problem, not a bandit problem, i.e., it is pure exploration without

exploitation. Indeed, the special case where the MDP terminates after the next action is exactly the ranking and selection problem.

- UCT uses the same decision process at each node recursively down the tree. Except at the root, however, the decision as to which successor of a given node to simulate next is not taken in a vacuum; instead, it is taken in the *context* of the information obtained about the ancestors of the node and about all their other descendants. As we show in Section 3.5, varying the a node's context can change the optimal metalevel policy below that node.

- In UCT, the values of the actions are estimated by the average reward of all simulations that take that action. Not all simulations use the same future policy, however, and the correct estimate of an action's value should use the future policy the agent will actually take. A more accurate estimate would take this into account, either by estimating an action's value using the apparently best future policy found so far (this corresponds to expectimax backup of simulation values at the leaves up the tree) or a more sophisticated estimate—e.g., those suggested by Baum and Smith (1999).

In order to improve upon UCT, Section 3 below gives a general probabilistic theory of metareasoning as an information collection problem, then Section 4 applies it to the study of controlling Monte Carlo sampling. Section 5 sketches how this can be applied to the design of improved MCTS algorithms.

3 Theory of metareasoning

Metareasoning treats computation as evidence about the uncertain utility of actions, choosing computations to find the best action in the least time. This begs a formalization within decision theory, where uncertainty is modeled probabilistically, and computations are selected to maximize expected utility.

This perspective places metareasoning within the wider class of information collection problems (Frazier, 2009). Information collection problems are found in many literatures under many different names, including ranking and selection problems, multi-armed bandit problems, sequential analysis, sequential design of experiments, foraging theory, consumer search theory, and information value theory (see (Frazier, 2009) for review).

This section gives the formal definition of metalevel decision problems within decision theory, investigating their general properties. The next section will apply this theory to choosing Monte Carlo samples.

U_i	The random variable equal to utility of object level action i for $i = 1, \ldots, k$.
$\mathcal{E}$	The set of all possible computations.
$E \in \mathcal{E}$	A random variable denoting a computation whose random value, e, is the result of that computation.
c	Cost of performing a computation; $c > 0$.
$\mathcal{S}$	The set of all possible belief states, i.e., the set of all possible sequences of computational results.
$S_t \in \mathcal{S}$	State at time $t = 0, 1, \ldots$, equal to the sequence of results of computations performed up to this time.
$\mathcal{E}_\perp$	$\mathcal{E} \cup \{\perp\}$, where $\perp$ denotes the decision to stop computing and select an action $1, \ldots, k$.
$E_t \in \mathcal{E}_\perp$	The decision at time t: either to compute ($E_t \in \mathcal{E}$) or to stop ($E_t = \perp$).
$\pi \colon \mathcal{S} \to \mathcal{E}_\perp$	A metalevel policy, which prescribes what to do in each possible state $s \in \mathcal{S}$.
$u_i \colon \mathcal{S} \to \mathbb{R}$	The expected utility of action i in state s, defined by $u_i(s) = E[U_i \mid S_0 = s]$.

Table 1: Metalevel probability model notation.

3.1 Metalevel probability models

Definition 1. A **metalevel probability model** $M = (U_1, \ldots, U_k, \mathcal{E})$ consists of jointly distributed random variables:

- $U_1, \ldots, U_k$, where U_i is the utility of performing action i, and
- $E_1, E_2, \cdots \in \mathcal{E}$, where E_j is a computation that can be performed and whose value is the result of that computation.

A metalevel probability model, when combined with a cost $c > 0$ of computation,[1] defines a metalevel decision problem: what is the optimal way to choose a finite sequence of computations $E \in \mathcal{E}$ to observe in order to maximize the expected utility of the action taken after that sequence less the cost of computation?

To get a precise answer to this question we need to give it mathematical form, and to do that we need notation (see Table 1 for an overview).

After performing a sequence of computations $E_1, \ldots, E_n \in \mathcal{E}$ we have a sequence of results $e_1, \ldots, e_n$, namely the (random) values of these variables. Let $s = \langle e_1, \ldots, e_n \rangle$ denote the sequence of results, and $\mathcal{S}$ the set of all such possible sequences. Such a sequence is the **state** of knowledge of the agent after performing these computations.

A **metalevel policy** $\pi \colon \mathcal{S} \to \mathcal{E}_\perp$, where $\mathcal{E}_\perp = \mathcal{E} \cup \{\perp\}$, chooses either a computation $\pi(s) \in \mathcal{E}$ to perform in any given state $s \in \mathcal{S}$, or decides to stop $\pi(s) = \perp$

[1] The assumption of a fixed cost of computation is a simplification; precise conditions for its validity are given by (Harada, 1997).

and take one of the object-level actions $1, \ldots, k$. Given such a policy, the random sequence of states it visits and the sequence of computations performed when executing is defined by

$$\begin{aligned} S_0 &= \langle\rangle \\ E_t &= \pi(S_t) \\ S_{t+1} &= S_t \cdot [E_t] \qquad \text{if } E_t \in \mathcal{E} \end{aligned} \tag{1}$$

where $S_i \cdot [E_t]$ denotes the sequence S_i extended by the result of the computation (i.e., the value of the random variable) E_t. If $E_t = \perp$ the states are defined only up to S_t. Denote by N^π the (random) time of the final state S_t.[2]

Once the policy decides to stop, an object-level action must be taken. This will be selected in order to maximize expected utility. If the final state S_{N^π} is s, the expected utility of performing N^π computations and choosing the action $i \in \{1, \ldots, k\}$ is

$$-c\, N^\pi + \mathrm{E}[U_i | S_{N^\pi} = s].$$

As the cost of computation is sunk, $\operatorname{argmax}_i \mathrm{E}_\pi[U_i | S_{N^\pi} = s]$ is the optimal choice, yielding an expected utility of

$$-c\, N^\pi + \max_i \mathrm{E}[U_i | S_{N^\pi} = s].$$

If we define $u_i(s) = \mathrm{E}[U_i | S_0 = s]$, the expected utility of a policy π without knowing the results S_{N^π} of its computations is

$$V_M^\pi = \mathrm{E}_\pi[-c\, N^\pi + \max_i u_i(S_{N^\pi})] \tag{2}$$

where the notation $\mathrm{E}_\pi[\ldots]$ makes explicit the dependence of the outside expectation on π, and where we make explicit the dependence of V_M^π on the metalevel probability model M.

Now we can formally state our question: *given a metalevel probability model M defining V_M^π through equation* (2)*, find* $\operatorname{argmax}_\pi V_M^\pi$.

It will be useful later to generalize the above notation to cover starting in an arbitrary state $s \in \mathcal{S}$:

$$V_M^\pi(s) = \mathrm{E}_\pi[-c\, N_s^\pi + \max_i u_i(S_{N_s^\pi}) | S_0 = s] \tag{3}$$

where N_s^π denotes the number of computations performed if the the policy is started in state s. Note that $V_M^\pi(\langle\rangle) = V_M^\pi$ and $N_{\langle\rangle}^\pi = N^\pi$.

Example 1. Suppose the computer is trying to decide between two actions. The first is safe, achieving an outcome of known utility u. The second is risky, achieving of an outcome of utility 1 (success) with probability Θ, and otherwise an outcome of utility 0 (failure). This probability Θ is itself unknown, with

[2] We'll assume all policies π are **proper**, i.e., that they halt with probability 1.

value uniformly distributed between 0 and 1. The computer has an accurate simulator of the risky action to aid in its choice. For cost c this simulator will generate an outcome, which will be success with probability Θ and failure otherwise. This problem is represented by the metalevel probability model with utilities U_1, U_2 and computational variables $\mathcal{E} = \{E_1, \dots\}$ under the following generative model:

$$\begin{aligned} \Theta &\sim \text{Uniform}(0,1) \\ U_1 &= u \\ U_2 \mid \Theta &\sim \text{Bernoulli}(\Theta) \\ E_i \mid \Theta &\overset{iid}{\sim} \text{Bernoulli}(\Theta) \qquad \text{for } i = 1, \dots \end{aligned}$$

See section 4.1 below for more on this model.

3.2 Metalevel MDP

In this section we show that metalevel decision processes are equivalent to a particular kind of Markov Decision Process (MDP). This will allow us to apply general results on MDPs to the metareasoning problem.

Definition 2. A **metalevel MDP** $M = (\mathcal{S}_\perp, \mathcal{E}_\perp, T, R)$ is an undiscounted MDP with state space $\mathcal{S}_\perp$, action set $\mathcal{E}_\perp$, transition model T, and reward function R, together with a constant $c > 0$ and functions $u_i \colon \mathcal{S} \to \mathbb{R}$ for $i = 1, \dots, k$ such that:

- $\mathcal{S}_\perp = \mathcal{S} \cup \{\perp\}$ where $\perp$ is the terminal state.
- $\mathcal{E}_\perp = \mathcal{E} \cup \{\perp\}$ where $\perp$ transitions to $\perp$ with probability 1.
- The utility estimates $u_i(s)$ are coherent:[3]

  $$u_i(s) = \sum_{s'} T(s, E, s')\, u_i(s')$$

  for all $s \in \mathcal{S}$, $E \in \mathcal{E}$, and $i = 1, \dots, k$.
- The reward function is defined by

  $$\begin{aligned} R(s, E, s') &= -c && \text{for } s, s' \in \mathcal{S} \text{ and } E \in \mathcal{E}, \\ R(s, \perp, \perp) &= \max_i u_i(s). \end{aligned}$$

[3] This formulation assumes a discrete transition distribution, but can be generalized straightforwardly to transition kernels in general.

In a metalevel MDP M, policies are given by functions $\pi\colon \mathcal{S} \to \mathcal{E}_\perp$, which have value functions defined in the usual way as the expected sum of rewards r_t received until termination at time T:

$$V_M^\pi(s) = \mathrm{E}_\pi \left[\sum_{t=0}^{T} r_t \right] . \tag{4}$$

Theorem 1. *Metalevel probability models and metalevel MDPs are equivalent ways of specifying a metalevel decision problem: given a metalevel probability model M with state space $\mathcal{S}$ and computation set $\mathcal{E}$ there is a metalevel MDP M' with state space $\mathcal{S} \cup \{\perp\}$ and action space $\mathcal{E} \cup \{\perp\}$ such that*

$$V_M^\pi(s) = V_{M'}^\pi(s)$$

for all policies π and states $s \in \mathcal{S}$, and conversely.

Proof. Given a metalevel probability model $M = (U_1, \ldots, U_k, \mathcal{E})$ with state space $\mathcal{S}$ and a cost $c > 0$ of computation, define a metalevel MDP by:

$$\begin{aligned}
\mathcal{S}_\perp &= \mathcal{S} \cup \{\perp\} \\
\mathcal{E}_\perp &= \mathcal{E} \cup \{\perp\} \\
u_i(s) &= E[U_i | S_0 = s] \\
T(s, E, s') &= P(S_1 = s' | S_0 = s, E_0 = E) \\
T(s, \perp, s') &= \begin{cases} 1 & \text{if } s' = \perp, \\ 0 & \text{otherwise.} \end{cases} \\
R(s, E, s') &= -c \\
R(s, \perp, \perp) &= \max_i u_i(s)
\end{aligned}$$

To see that this is a metalevel MDP we need only check coherence:

$$\begin{aligned}
\sum_{s'} T(s, E, s') u_i(s') &= \sum_{s'} P(S_1 = s' | S_0 = s, E_0 = E) E[U_i | S_0 = s'] \\
&= \sum_{s'} P(S_1 = s' | S_0 = s, E_0 = E) E[U_i | S_1 = s'] \\
&= E[U_i | S_0 = s, E_0 = E] \\
&= E[U_i | S_0 = s] \\
&= u_i(s).
\end{aligned}$$

We'll show that $V_M^\pi(s)$ for the original metalevel probability model satisfies the Bellman equations for the metalevel MDP, which implies that $V_M^\pi = V_{M'}^\pi$. Take

any $s \in \mathcal{S}$ and observe that

$$
\begin{aligned}
V_M^\pi(s) &= \mathrm{E}_\pi[-c\, N_s^\pi + \max_i u_i(S_{N_s^\pi}) \mid S_0 = s] \\
&= \begin{cases} \max_i u_i(s) & \text{if } \pi(s) = \bot, \\ \mathrm{E}_\pi[-c\,(1 + N_{S_1}^\pi) + \max_i u_i(S_{1+N_{S_1}^\pi}) \mid S_0 = s] & \text{otherwise.} \end{cases} \\
&= \begin{cases} \max_i u_i(s) + V_M^\pi(\bot) & \text{if } \pi(s) = \bot, \\ -c + \sum_{s'} P(S_1 = s' | S_0 = s, E_0 = \pi(s)) & \\ \qquad \mathrm{E}_\pi[-c\, N_{S_1}^\pi + \max_i u_i(S_{1+N_{S_1}^\pi}) \mid S_1 = s'] & \text{otherwise.} \end{cases} \\
&= \begin{cases} \max_i u_i(s) + V_M^\pi(\bot) & \text{if } \pi(s) = \bot, \\ -c + \sum_{s'} T(s, \pi(s), s') \mathrm{E}_\pi[-c\, N_{s'}^\pi + \max_i u_i(S_{N_{s'}^\pi}) \mid S_0 = s'] & \text{otherwise.} \end{cases} \\
&= \begin{cases} \max_i u_i(s) + V_M^\pi(\bot) & \text{if } \pi(s) = \bot, \\ -c + \sum_{s'} T(s, \pi(s), s') V_M^\pi(s') & \text{otherwise.} \end{cases} \\
&= \sum_{s'} T(s, \pi(s), s')[R(s, \pi(s), s') + V_M^\pi(s')]
\end{aligned}
$$

The converse is straightforward. □

In the following we'll freely use both forms as convenient.

3.3 The myopic policy

In general it may be difficult to find the optimal policy for a metalevel MDP, so approximations or simplifications are often made. The metagreedy assumption (Russell and Wefald, 1991) is that there is at most one computation remaining before action will be taken. A **myopic policy** (also known as a knowledge gradient algorithm (Frazier, 2009)) selects computations based on this assumption.

The value of immediately stopping in a state s is:

$$\max_i u_i(s),$$

and the value of performing the computation E then immediately stopping is:

$$-c + \sum_{s'} T(s, E, s') \max_i u_i(s').$$

Thus, the myopic policy will continue computing if there is some computation E such that:

$$\sum_{s'} T(s, E, s') \max_i u_i(s') - \max_i u_i(s) \geq c\ ; \tag{5}$$

otherwise, it will stop. It will select the E that maximizes the excess of computing over stopping. Notice that for the myopic policy the cost of computation c only affects the threshold for stopping, not which computation is selected if it is optimal to continue. This is not true for optimal policies in general.

3.4 Relation between myopic and optimal policies

Definition 3. In a metalevel MDP, a state s' is **reachable** from a state s if there exists a policy π such that there is positive probability of transitioning to state s' starting from state s under policy π.

Theorem 2. *In a metalevel MDP, if the optimal policy stops in a state s then the myopic policy stops too. As a partial converse, if the myopic policy stops in all states reachable from a given state s, then the optimal policy stops at s.*

Proof. The optimal policy stops in a state s if

$$V^{\pi}(s) \leq \max_i u_i(s)$$

for all policies π, but this includes the policies which perform exactly one computation before acting, so the myopic policy stops too.

For the partial converse, suppose that the myopic policy stops for all states s' reachable from a state s. Then for all such s' and all $E \in \mathcal{E}$

$$-c + \left(\sum_{s''} T(s', E, s'') \max_i u_i(s'') \right) - \max_i u_i(s') \leq 0. \tag{6}$$

The results of Ng et al. (1999) show we can transform an MDP's reward function by shaping without affecting which policies are optimal. If we use the shaping function

$$\varphi(s) = \begin{cases} \max_i u_i(s) & \text{if } s \in \mathcal{S}, \\ 0 & \text{if } s = \bot, \end{cases}$$

to transform the reward function of the metalevel MDP, we get:

$$\begin{aligned} R'(s, E, s') &= R(s, E, s') + \varphi(s') - \varphi(s) \\ &= \begin{cases} -c + \max_i u_i(s') - \max_i u_i(s) & \text{if } E \in \mathcal{E}, \\ 0 & \text{if } E = \bot \text{ and } s' = \bot. \end{cases} \end{aligned}$$

Then equation (6) shows that the expected reward of any computation E is non-positive for all s' reachable from s:

$$R'(s', E) = \sum_{s''} T(s', E, s'') R'(s', E, s'') \leq 0.$$

Thus, the utility of the optimal policy is no greater than stopping at s, so the optimal policy stops. □

3.5 Context effects

Suppose we place a metalevel MDP M within a wider context, where there is a new external action of fixed utility u, but the computations and other actions are as before. Let M_u be the metalevel MDP M augmented with this new external action; the only change is that the terminal reward becomes

$$R_u(s, \bot, \bot) = \max(u, R(s, \bot, \bot)) = \max(u, \max_i u_i(s)).$$

How does the optimal policy for M_u relate to that of M? How does it depend on u?

If L is a lower bound on the utility of all actions in M, then M_L is isomorphic to M, and thus the utility of all polices are the same. This is because there is no advantage to ever taking the augmenting action, i.e.,

$$\max(L, \max_i u_i(s)) = \max_i u_i(s)$$

for any state s.

If U is an upper bound on the utility of all actions in M, then in M_U it is optimal to stop in all states, as the terminal reward equals U in all states, so not stopping will only incur additional costs $-c$ for computing to no benefit.

What happens for intermediate values $L \leq u \leq U$?

Theorem 3. *Suppose we take a metalevel MDP M and add to it an external action of fixed utility u yielding an augmented metalevel MDP M_u. Then there is a function $I(s)$ assigning an interval to each state $s \in \mathcal{S}$ such that it is optimal to continue computing from state s in M_u iff the value of the fixed action u lies within this interval $I(s)$. Furthermore, whenever $I(s)$ is non-empty it contains $\max_i u_i(s)$.*

Proof. The utility of policy π starting in state s of M_u is

$$V^\pi_{M_u}(s) = \mathrm{E}_\pi[-c\, N^\pi_s + \max(u, \max_i u_i(S_{N^\pi_s})) \mid S_0 = s],$$

and the utility of stopping in state s is

$$Q^\pi_{M_u}(s, \bot) = \max(u, \max_i u_i(s)).$$

We need to show that the set $I(s)$ of u such that

$$\max_\pi \left(V^\pi_{M_u}(s) - Q^\pi_{M_u}(s, \bot)\right) \geq 0$$

forms an interval.

Notice that the above expression has a simple form as a function of u:

$$\begin{aligned} g(u) &\equiv \max_\pi \left(V^\pi_{M_u}(s) - Q^\pi_{M_u}(s, \bot)\right) \\ &= -w + \max_\pi \mathrm{E}_\pi[\max(X^\pi, u) \mid S_0 = s] - \max(v, u) \end{aligned}$$

where

$$\begin{aligned} w &= c\,\mathrm{E}_\pi[N_s^\pi \mid S_0 = s] \\ X^\pi &= \max_i u_i(S_{N_s^\pi}) \\ v &= \max_i u_i(s)\ . \end{aligned}$$

Functions of this form increase monotonically in u to a maximum at $u = v$, then decrease monotonically there after, approaching $-w$ in the limit. This is because $\mathrm{E}_\pi[\max(X, u) \mid S_0 = s]$ is convex in u by Jensen's inequality, with subderivative at most one everywhere. This property is preserved by maximization over π and translation by $-w$. The term $-\max(v, u)$ is constant before u, so the function is nondecreasing, and has derivative -1 there after, so the function is nonincreasing.

Therefore, the set of u for which such a function is non-negative forms an interval containing v if non-empty. □

This result shows context can affect whether it is optimal to stop or not. Can it affect which computation it is optimal to take? Yes, as the following example shows.

Example 2. Suppose there are two actions:

- Action A has low expected utility and high variance: it is equally likely to have values -1.5 or 1.5, so has prior mean 0.
- Action B has high expected utility and low variance: it is equally likely to have values 0.25 or 1.75, so has prior mean 1.

There are two computations which can be performed at cost 0.2: observe action A's value exactly, and observe action B's value exactly. This forms a metalevel MDP with $1 + 2 * 2 + 4 = 9$ states, so it straightforward to solve by backwards induction:

$$\begin{aligned} Q_u^*(\langle\rangle, \bot) &= \max(u, 1) \\ Q_u^*(\langle\rangle, A) &= 0.5\max(u, 1.5, -0.2 + 0.5\max(u, 1.5) + 0.5\max(u, 1.75)) \\ &\quad + 0.5\max(u, 1, -0.2 + 0.5\max(u, 0.25) + 0.5\max(u, 1.75)) - 0.2 \\ Q_u^*(\langle\rangle, B) &= 0.5\max(u, 0.25, -0.2 + 0.5\max(u, 1.5) + 0.5\max(u, 0.25)) \\ &\quad + 0.5\max(u, 1.75, -0.2 + 0.5\max(u, 1.75) + 0.5\max(u, 1.75)) - 0.2 \end{aligned}$$

Figure 1 plots these functions less $Q_u^*(\langle\rangle, \bot)$. Observe that for $u = 0$ it is optimal to observe B but for $u = 1$ it is optimal to observe A, showing that context changes which computation is optimal.

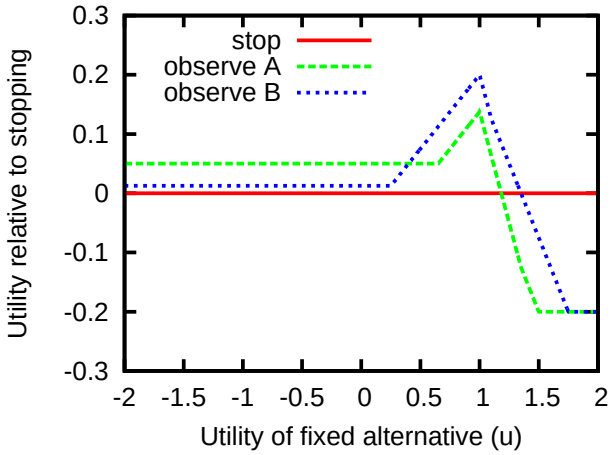

Figure 1: Utility of stopping, observing A, or observing B, relative to stopping, as a function of the value of a fixed alternative u. Notice that as u increases it is optimal to observe B, then to observe A, then to stop.

4 Monte Carlo Metareasoning

One important way to gain information about the expected utility of an action is through Monte Carlo simulation of its consequences. This is more complex in the sequential case, where the consequences include as yet undetermined future actions by the agent, but it is interesting even in the one-shot case where they do not.

In this section we apply the general theory above to controlling Monte Carlo simulation of actions. This problem has been studied before in the ranking and selection literature (Swisher et al., 2003) as Bayesian ranking and selection, including the formulation as an MDP and myopic policies (called knowledge gradient algorithms); see (Frazier, 2009) for review.

A Monte Carlo simulator stochastically simulates one possible sequence of consequences following from a given action, returning the total utility of these consequences. Statistically, this corresponds to sampling from the distribution over the utility of taking a given action. Equivalently, this corresponds to making a noisy measurement of the *expected* utility of the action.

Given such a simulator, the corresponding metareasoning problem is to choose which actions to simulate and when to stop. These choices can depend on the results of the simulations made so far.

Mathematically, to define such a problem we need to specify the prior distribution over action utilities $U_1, \ldots, U_k$, a cost $c > 0$ of sampling, and a distribution for a sample E of action i's utility conditional on action i having utility $U_i = u$.

4.1 Bernoulli sampling

Perhaps the simplest case is the **Bernoulli sampling model** for a single action with unknown utility U_1, The model supposes that there is some latent probability Θ, assumed uniformly distributed, of the action leading to success (utility 1) rather than failure (utility 0). The samples (simulations) are assumed well-calibrated, so they share this same probability of success. Formally:

$$\begin{aligned} \Theta &\sim \text{Beta}(1,1) \\ U_1 \mid \Theta &\sim \text{Bernoulli}(\Theta) \\ E_i \mid \Theta &\overset{iid}{\sim} \text{Bernoulli}(\Theta) \qquad \text{for } i = 1, \ldots \end{aligned}$$

where the density of a $\text{Beta}(\alpha, \beta)$ distribution is proportional to $\theta^{\alpha-1}(1-\theta)^{\beta-1}$, so $\text{Beta}(1,1)$ is the uniform distribution. The beta distribution is convenient as it is conjugate to the Bernoulli, so the posterior distribution over Θ conditioned on n samples is also beta. In particular, given n samples $E_1, \ldots, E_n$ of which n_1 have value 1 and n_0 have value 0, it is easy to see that

$$\Theta \mid E_1, \ldots, E_n \sim \text{Beta}(1 + n_1, 1 + n_0).$$

Since $U_1, E_{n+1}, \ldots$ are conditionally independent of $E_1, \ldots, E_n$ given Θ, it is sufficient to keep track of the parameters (α, β) of the posterior distribution over Θ. In this case we have:

$$\begin{aligned} E_j \mid E_1, \ldots, E_n &\sim \text{Bernoulli}(\alpha/(\alpha+\beta)) \\ E[U_1 \mid E_1, \ldots, E_n] &= \alpha/(\alpha+\beta) \end{aligned}$$

for $j > n$. Under this model the parameters (α, β) form a Markov chain (see Figure 2) that starts in state $(1,1)$ and, on sampling, transitions to $(\alpha+1, \beta)$ with probability $\alpha/(\alpha, \beta)$ and to $(\alpha, \beta+1)$ otherwise. The expected value of U_1 in state (α, β) is simply $\alpha/(\alpha, \beta)$.

The above describes the Bernoulli model for sampling a single action's value, and how to convert this into an Markov chain. But we need at least two actions to have a non-trivial metalevel decision problem: there is no point computing if we have nothing to decide between. There are two natural ways to extend this to a nontrivial metalevel decision problem:

1. Have $k \geq 2$ Bernoulli actions with independently distributed utilities:

$$\begin{aligned} \Theta_j &\overset{iid}{\sim} \text{Beta}(1,1) && \text{for } j = 1, \ldots, k \\ U_j \mid \Theta_j &\sim \text{Bernoulli}(\Theta_j) && \text{for } j = 1, \ldots, k \\ E_{ji} \mid \Theta_j &\overset{iid}{\sim} \text{Bernoulli}(\Theta_j) && \text{for } j = 1, \ldots, k,\ i = 1, \ldots \end{aligned}$$

   Since these actions are completely independent, the equivalent metalevel MDP has states of the form $s = (\alpha_1, \beta_1, \ldots, \alpha_k, \beta_k)$, sampling action j increments α_j with probability $\alpha_j/(\alpha_j + \beta_j)$ and increments β_j otherwise, and $u_j(s) = \alpha_j/(\alpha_j + \beta_j)$.

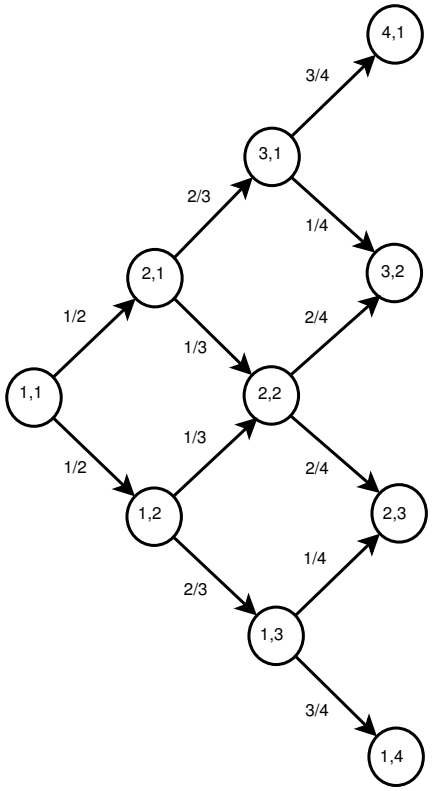

Figure 2: Fragment of the Bernoulli state space for a single action. States are labeled with the parameters (α, β) of the posterior Beta distribution over the action's utility in that state. Outgoing edges are labeled with the probability of following that transition upon sampling. The state space continues infinitely out to the right.

2. Add a single action of known value $U_2 = u$ (call this the **contextual** Bernoulli problem). This retains the same state space and transition model as a single action (Figure 2). It is equivalent to the previous problem where $k = 2$ and $(\alpha_2, \beta_2) = (nu, n(1-u))$ in the limit $n \to \infty$.

In the following we'll call both the Bernoulli sampling problem, the latter special case being the contextual version.

4.2 Myopic Bernoulli sampling

Section 3.3 derived the general form for the myopic sampling policy. Combining its results with the Bernoulli metalevel MDP from the previous section, we find the advantage of sampling action i and stopping, versus immediately stopping, is

$$a(i, u_i) = -c + \left(\mu_i \max\left(\mu_i^+, u_i\right) + (1-\mu_i)\max\left(\mu_i^-, u_i\right)\right) - \max\left(\mu_i, u_i\right) \quad (7)$$

where

$$\begin{aligned}
\mu_i &= \frac{\alpha_i}{n_i} \\
\mu_i^+ &= \frac{\alpha_i + 1}{n_i + 1} = \mu_i + \frac{1}{n_i + 1}(1 - \mu_i) \\
\mu_i^- &= \frac{\alpha_i}{n_i + 1} = \mu_i - \frac{1}{n_i + 1}\mu_i \\
u_i &= \max_{j \neq i} \mu_j
\end{aligned}$$

and where $n_i = \alpha_i + \beta_i$. Figure 3 illustrates $a(i, u_i)$ as a function of u_i.

The myopic policy samples the action that maximizes $a(i, u_i)$ so long as that maximum exceeds zero, stopping otherwise. Intuitively, one expects that that the expected improvement in decision quality from one additional sample will decrease as more samples are obtained, because the action utility estimates change less and less. This is in fact the case:

Theorem 4. *The myopic policy for the Bernoulli sampling problem is guaranteed to halt in any state such that*

$$\frac{1}{4c} - 1 \leq \min_i n_i \quad (8)$$

and will eventually reach such a state or halt earlier.

Proof. We can bound $a(i, u_i)$ below zero as follows:

$$\begin{aligned}
a(i, u_i) &\leq -c + a(i, \mu_i) \\
&= -c + \mu_i(1 - \mu_i)/(n_i + 1) \\
&\leq -c + 1/4(n_i + 1)
\end{aligned}$$

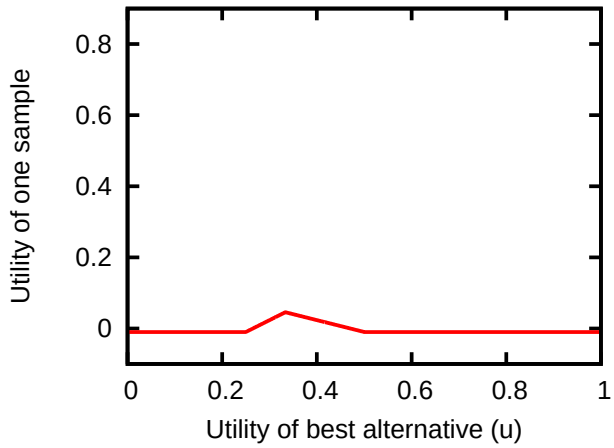

Figure 3: Equation 7 for parameters $(\alpha, \beta) = (1, 2)$ and cost $c = 0.01$. The derivative changes at the points $(\mu^-, \mu, \mu^+) = (1/4, 1/3, 2/4)$. Notice that the maximum value of $\mu(1-\mu)/(n+1) - c = 1/18 - c$ is attained at $u = \mu$.

where this inequality is tight when $\mu_i = u_i = 1/2$. The myopic policy halts if all options have nonpositive net value, in particular as soon as:

$$\max_i \frac{1}{4(n_i + 1)} \leq c,$$

or equivalently $\frac{1}{4c} - 1 \leq \min_i n_i$. That it will eventually reach such a state (if it does not halt earlier) follows from the fact that it can allocate at most $\frac{1}{4c} - 1$ samples to a given action before abandoning it, so after $\frac{k}{4c} - k$ samples it must have halted. □

4.3 Optimal Bernoulli sampling

There is no general solution for the optimal policy in a metalevel MDP. As the state space of the Bernoulli model is infinite (Figure 2), it is not clear whether this can even be numerically solved. Fortunately, we can extend the result of the preceding section from myopic to optimal policies:

Theorem 5. *The optimal policy is guaranteed to halt in any state such that*

$$\frac{1}{4c} - 1 \leq \min_i n_i \tag{9}$$

and will eventually reach such a state or halt earlier.

Proof. Theorem 4 established that the myopic policy will halt if a state satisfies

$$\frac{1}{4c} - 1 \leq \min_i n_i.$$

Moreover, all states reachable from such a state (in the sense of definition 3) must satisfy the same condition, since the counts n_i can only increase. Hence, by Theorem 2 we know that the *optimal* policy will halt if this same bound holds. □

Thus, we can compute the optimal policy exactly via dynamic programming, backtracking from states with $\frac{1}{4c} - 1$ samples per action.

4.4 Normal sampling

The **normal sampling model** is defined by

$$\begin{aligned} M &\sim \text{Normal}(0, \sigma_0^2) \\ U_1 \mid M &\sim \text{Normal}(M, 1) \\ E_i \mid M &\overset{iid}{\sim} \text{Normal}(M, 1) \qquad \text{for } i = 1, \ldots \end{aligned}$$

for some prior variance σ_0^2.

We briefly state the results analogous to those in sections 4.1 through 4.3 for the Bernoulli model.

A state is given by the mean and precision of the posterior normal over M given the samples observed so far, i.e., a pair $(\mu, \lambda) \in \mathbb{R}^2$ where $\lambda > 0$. The initial state is $(0, \sigma_0^{-2})$. Upon sampling, the state (μ, λ) transitions to

$$(\mu + sN,\ \lambda + 1)$$

where

$$s^2 = 1/\lambda - 1/(\lambda + 1) = \frac{1}{\lambda(\lambda + 1)}$$

and where $N \sim \text{Normal}(0, 1)$.

With k independent normal actions, the myopic advantage of sampling i is

$$a(i, u_i) = -c + s_i f(|\mu_i - u_i|/s_i)$$

where

$$\begin{aligned} s_i^2 &= 1/\lambda_i - 1/(\lambda_i + 1) \\ f(\delta) &= \phi(|\delta|) - |\delta|\Phi(|\delta|) \qquad \text{(See Figure 4.)} \\ u_i &= \max_{j \neq i} \mu_j \end{aligned}$$

and where ϕ and Φ are the unit normal PDF and CDF, respectively.

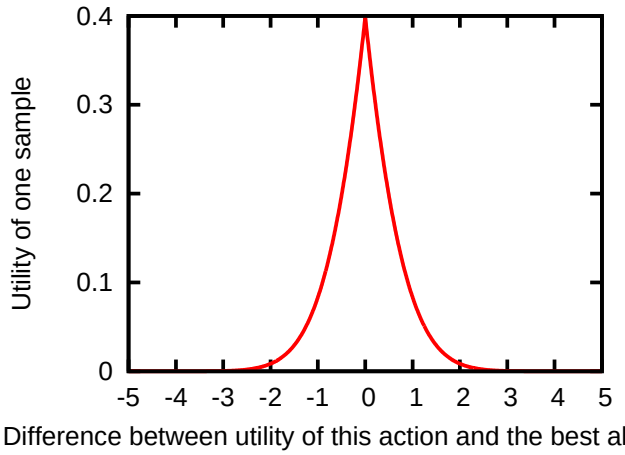

Figure 4: The value $f(\delta)$ of sampling a normally distributed action where δ is the difference between the utility of the sampled action and the next best one. The graph is scaled so that the distribution of the posterior mean given the sample is a standard normal distribution.

Finally, observe that

$$\begin{aligned} a(i, u_i) &\leq -c + a(i, \mu_i) \\ &= -c + (2\pi)^{-1/2}(\lambda_i(\lambda_i + 1))^{-1/2} \\ &\leq -c + (2\pi)^{-1/2}\lambda_i^{-1} \end{aligned}$$

so the normal myopic policy, and thus the normal optimal policy, will stop sampling when

$$\frac{1}{c\sqrt{2\pi}} \leq \max_i \lambda_i.$$

4.5 Regret bounds for sampling

UCB1 (Auer et al., 2002), the selection algorithm used within UCT, is distinguished by meeting the optimal regret bound of bandit proved by Lai and Robbins (1985). Is there an analogous algorithm with optimal regret for the ranking and selection problem? This may be a more appropriate choice for selection within UCT, particularly close to the root.

The regret R^π of a metalevel policy π is the gap between the expected utility of π and the expected utility of the best choice given the results of all possible

computations $\mathcal{E}$:

$$\begin{aligned} R_c^\pi &= \mathrm{E}[\max_i \mathrm{E}[U_i|\mathcal{E}]] - V^\pi \\ &= c\,\mathrm{E}_\pi[N^\pi] + \mathrm{E}\max_i \mathrm{E}[U_i|\mathcal{E}] - \mathrm{E}_\pi \max_i \mathrm{E}[U_i|S_{N^\pi}] \ . \end{aligned}$$

Define the *optimal regret* to be $R_c^* = \sup_\pi R^\pi$. The rest of this section derives an asymptotic lower bound on the optimal regret. Note that the bound due to Lai and Robbins assumes an exogenously determined number of samples n and examines the regret as $n \to \infty$. In the context of metareasoning problems, where the sampling policy also includes its own stopping decision, this limit does not make sense. Instead, we consider the limit as the cost of computation $c \to 0$.

Theorem 6. *For any metareasoning problem, the optimal regret tends to zero as the cost of computation does;*

$$\lim_{c\to 0} R_c^* = 0.$$

Proof. As $\mathcal{E}$ is countable it can be ordered, let $\mathcal{E}_n$ denote the first n variables of $\mathcal{E}$, and let π_n be the policy that observes exactly the n variables in $\mathcal{E}_n$. Then,

$$V^{\pi_n} = -cn + \mathrm{E}\max_i \mathrm{E}[U_i|\mathcal{E}_n].$$

We have

$$\lim_{n\to\infty} \mathrm{E}[U_i|\mathcal{E}_n] = \mathrm{E}[U_i|\mathcal{E}] \qquad \text{a.s.}$$

by Lévy's zero-one law, and so

$$\lim_{n\to\infty} \mathrm{E}\max_i \mathrm{E}[U_i|\mathcal{E}_n] = \mathrm{E}\max_i \mathrm{E}[U_i|\mathcal{E}] \qquad \text{a.s.}$$

But then for any desired $\epsilon > 0$ we can choose n and let $c = \epsilon/n$ such that

$$R_c^* \le R_c^{\pi_n} = cn + \mathrm{E}\max_i \mathrm{E}[U_i|\mathcal{E}] - \mathrm{E}\max_i \mathrm{E}[U_i|\mathcal{E}_n] \le 2\epsilon. \qquad \square$$

Now consider a metareasoning problem with k external actions to choose from. The policy robin_n samples each of the k actions n times. It is clear from the proof of Theorem 6 the the regret of the round robin policy tends to zero as c does for suitably increasing n. In fact, we can give a tighter bound:

Theorem 7. *Consider the normal sampling model, where k actions have independent unit normal prior and unit normal error distribution. Then the regret of the round robin policy is*

$$\begin{aligned} R_c^{\mathrm{robin}_n} &= cn + \sqrt{\frac{1}{n+1}} M(k) \\ \min_n R_c^{\mathrm{robin}_n} &\approx (2^{-2/3} + 2^{1/3}) M(k)^{2/3} c^{1/3} \end{aligned}$$

where $M(k)$ is the expected maximum of k independent unit normals (see Figure 5), and where the approximation converges in the limit $1/c \to \infty$.

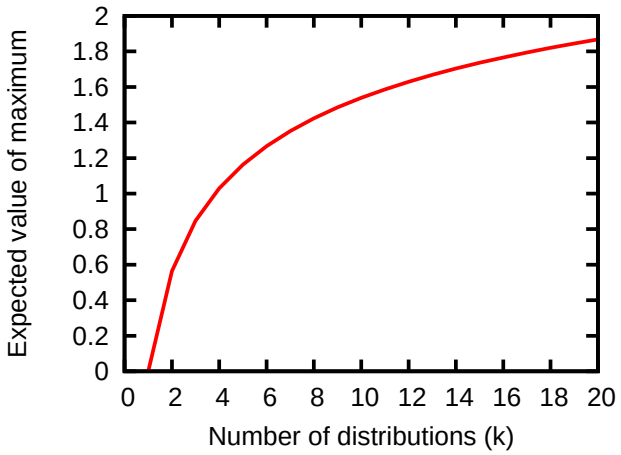

Figure 5: The expected value of $M(k)$, the maximum of k independent normal distributions.

Proof. For the first equation observe that after observing n samples of the ith action its mean is distributed according to $N(0, 1/(n+1))$, which equals a unit normal scaled by $\sqrt{\frac{1}{n+1}}$. The maximum of k of these is, therefore, $M(k)$ multiplied by $\sqrt{\frac{1}{n+1}}$.

For the second equation, we approximate $\sqrt{\frac{1}{n+1}}$ as $\sqrt{\frac{1}{n}}$ then maximize over n (treating it as a continuous variable) by finding the critical point and verifying it is a maximum. Both approximations result in overestimating the true value of $\min_n R_c^{\mathrm{robin}_n}$, but as $1/c \to \infty$ we have $n \to \infty$ and the error introduced tends to zero. □

Note that $(2^{-2/3} + 2^{1/3}) \approx 1.8899$.

By extending the round-robin policy to allow stopping to be dependent on the sampling results, rather than being fixed in advance, we get a lower bound on the regret of the optimal policy:

Theorem 8. *Consider the normal sampling model, where k actions have independent unit normal prior and unit normal error distribution. Then*

$$R_c^* \geq \sup_T R_{c/k}^{\mathrm{robin}_T}$$

where the supremum is over all policies for stopping dependent on past sampling results.

Proof. Observe that any policy π can be expanded to a round robin policy robin_T with a random number T of samples by sampling all actions whenever π

samples one, and choosing when to stop according to only the information that π would have, i.e., ignoring the extra samples. Then

$$\begin{aligned} R_{c/k}^{\text{robin}_T} &= (c/k)\mathrm{E}_{\text{robin}_T}[N^{\text{robin}_T}] + \mathrm{E}\max_i \mathrm{E}[U_i|\mathcal{E}] - \mathrm{E}_{\text{robin}_T}\max_i \mathrm{E}[U_i|S_{N^{\text{robin}_T}}] \\ &\leq c\,\mathrm{E}_\pi[N^\pi] + \mathrm{E}\max_i \mathrm{E}[U_i|\mathcal{E}] - \mathrm{E}_\pi \max_i \mathrm{E}[U_i|S_{N^\pi}] \\ &= R_c^\pi. \end{aligned}$$

thus $R_c^* \geq \sup_T R_{c/k}^{\text{robin}_T}$. □

It may be possible to extend Theorem 7 to this wider class of round-robin policy, perhaps with the same $O(c^{1/3})$ growth rate.

5 Prospects for MCTS

The preceding sections have studied metareasoning in general as well as its application to Monte Carlo sampling. How can these results be applied to the design of improved MCTS algorithms?

One obvious possibility is to find a drop-in replacement policy for UCB1 within the UCT algorithm. For example, we might explore the use of the myopic normal policy and variants thereof.

A more substantial alternative would be to apply reinforcement learning techniques to learn a metalevel policy (Russell and Wefald, 1991; Russell, 1997). This is particularly natural as Section 3 modeled the metalevel decision problem as a metalevel MDP. There are two primary difficulties to such an approach:

- *Credit assignment*: An agent might do millions or billions of computations for each external action, so external rewards are extremely sparse in the metalevel trace. This can be addressed by **reward shaping** (Dorigo and Colombetti, 1994; Ng et al., 1999), whereby intermediate rewards can be given that guide the learning process. One option is the shaping reward $\varphi(s)$ used in the proof of Theorem 2.
- *Function approximation*: The state space is exceedingly large—in fact, infinite—and intricately structured. A single metalevel state is an entire tree of millions of object-level states and their evaluations. Function approximation is necessary to represent the learned value functions. The above theory suggests certain general features, such as the myopic value of various actions, which may be useful to such an approximation. It may also be necessary to use recursive function approximators to represent the Q-functions over leaf expansions. Gradients for these functions can be computed with automatic differentiation (Griewank and Walther, 2008), allowing standard reinforcement learning algorithms to be applied.

A natural first step is to develop metalevel reinforcement learning for the simple Monte Carlo sampling problems of section 4 before proceeding to the more complex, sequential decision settings.

References

Stuart Russell andd Eric Wefald. Principles of metareasoning. *Artificial Intelligence*, 1991.

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 2002.

E.B. Baum and W.D. Smith. Propagating distributions up directed acyclic graphs. *Neural computation*, 11(1):215–227, 1999.

D.A. Berry and B. Fristedt. *Bandit problems: sequential allocation of experiments*. Chapman and Hall London, 1985.

T. Dean and Mark Boddy. An analysis of time-dependent planning. In *AAAI-88*, pages 49–54, 1988.

M. Dorigo and M. Colombetti. Robot shaping: Developing autonomous agents through learning. *Artificial Intelligence*, 71(2):321–370, 1994.

J. Doyle. Artificial intelligence and rational self-government. Technical Report CMU-CS-88-124, Computer Science Department, Carnegie-Mellon University, Pittsburgh, PA, 1988.

M.R. Fehling and J.S. Breese. A computational model for the decision-theoretic control of problem solving under uncertainty. In *Proceedings of the Fourth Workshop on Uncertainty in Artificial Intelligence, Minneapolis, MN: AAAI*, 1988.

Peter Frazier. *Knowledge-gradient methods for statistical learning*. PhD thesis, Princeton University, 2009.

Sylvain Gelly and David Silver. Monte-carlo tree search and rapid action value estimation in computer go. *Artificial Intelligence*, 2011.

I. J. Good. A five-year plan for automatic chess. In Ella Dale and Donald Michie, editors, *Machine Intelligence 2*, pages 89–118. Oliver and Boyd, 1968.

A. Griewank and A. Walther. *Evaluating derivatives: principles and techniques of algorithmic differentiation*. Society for Industrial and Applied Mathematics (SIAM), 2008.

Daishi Harada. Reinforcement learning with time. In *Proc. Fourteenth National Conference on Artificial Intelligence*, pages 577–582. AAAI Press, 1997.

E. J. Horvitz. Reasoning about beliefs and actions under computational resource constraints. In *UAI*. American Association for Artificial Intelligence, July 1987. Also in L. Kanal, T. Levitt, and J. Lemmer, ed., *Uncertainty in Artificial Intelligence 3,* Elsevier, 1988, pps. 301-324.

Ronald A Howard. Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, 1966.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. *ECML*, 2006.

T L Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics*, 1985.

O. Madani, D.J. Lizotte, and R. Greiner. Active model selection. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 357–365. AUAI Press, 2004.

James E Matheson. The economic value of analysis and computation. *Systems Science and Cybernetics*, 4:325–332, 1968.

Andrew Y. Ng, Daishi Harada, and Stuart J. Russell. Policy invariance under reward transformations: Theory and application to reward shaping. In *Proc. Sixteenth International Conference on Machine Learning*. Morgan Kaufmann, 1999.

Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58:527–535, 1952.

Stuart Russell. Rationality and intelligence. *Artificial intelligence*, 94(1-2):57–77, 1997.

Stuart Russell and Eric Wefald. *Do The Right Thing*. The MIT Press, 1991.

Stuart J. Russell and Eric H. Wefald. Multi-level decision-theoretic search. In *Proceedings of the AAAI Spring Symposium Series on Computer Game-Playing*, Stanford, California, 1988. AAAI.

Stuart J. Russell and Eric H. Wefald. On optimal game-tree search using rational meta-reasoning. In *Proc. Eleventh International Joint Conference on Artificial Intelligence*, pages 334–340, Detroit, 1989. Morgan Kaufmann.

Stuart J. Russell and Eric H. Wefald. Decision-theoretic control of search: General theory and an application to game-playing. Technical Report UCB/CSD 88/435, Computer Science Division, University of California, Berkeley, 2008.

R.S. Sutton and A.G. Barto. *Reinforcement learning: An introduction*, volume 116. Cambridge University Press, 1998.

James R. Swisher, Sheldon H. Jacobson, and Enver Yücesan. Discrete-event simulation optimization using ranking, selection, and multiple comparison procedures: A survey. *ACM Transactions on Modeling and Computer Simulation*, 2003.

Y. Yu. Structural properties of bayesian bandits with exponential family distributions. *Arxiv preprint arXiv:1103.3089*, 2011.