Extreme Value Monte Carlo Tree Search for Classical Planning

Masataro Asai[*1], **Stephen Wissow**[*2]

[1]MIT-IBM Watson AI Lab
[2]University of New Hampshire
masataro.asai@ibm.com, sjw@cs.unh.edu

The python code for MCTS is attached at the end of this supplement.

A1 Domain-Independent Heuristics in Classical Planning

A domain-independent heuristic function h in classical planning is a function of a state s and the problem $\langle P, A, I, G\rangle$, but the notation $h(s)$ usually omits the latter. In addition to what we discussed in the main article, this section also uses a notation $h(s, G)$. It returns an estimate of the cumulative cost from s to one of the goal states (states that satisfy G), typically through a symbolic, non-statistical means including problem relaxation and abstraction. Notable state-of-the-art functions that appear in this paper includes $h^{\mathrm{FF}}, h^{\max}, h^{\mathrm{add}}, h^{\mathrm{GC}}$ (Hoffmann and Nebel 2001; Bonet and Geffner 2001; Fikes, Hart, and Nilsson 1972).

A significant class of heuristics is called delete relaxation heuristics, which solve a relaxed problem which does not contain delete effects, and then returns the cost of the solution of the relaxed problem as an output. The cost of the optimal solution of a delete relaxed planning problem from a state s is denoted by $h^+(s)$, but this is too expensive to compute in practice (NP-complete) (Bylander 1996). Therefore, practical heuristics typically try to obtain its further relaxations that can be computed in polynomial time.

One such admissible heuristic based on delete-relaxation is called $h^{\max}$ (Bonet and Geffner 2001) that is recursively defined as follows:

$$h^{\max}(s, G) = \max_{p \in G} \begin{cases} 0 \text{ if } p \in s. \text{ Otherwise,} \\ \min_{\{a \in A | p \in \mathrm{ADD}(a)\}} \\ \quad \left[\mathrm{COST}(a) + h^{\mathrm{add}}(s, \mathrm{PRE}(a))\right]. \end{cases} \tag{1}$$

Its inadmissible variant is called additive heuristics h^{add} (Bonet and Geffner 2001) that is recursively defined as follows:

$$h^{\mathrm{add}}(s, G) = \sum_{p \in G} \begin{cases} 0 \text{ if } p \in s. \text{ Otherwise,} \\ \min_{\{a \in A | p \in \mathrm{ADD}(a)\}} \\ \quad \left[\mathrm{COST}(a) + h^{\mathrm{add}}(s, \mathrm{PRE}(a))\right]. \end{cases} \tag{2}$$

Another inadmissible delete-relaxation heuristics called h^{FF} (Hoffmann and Nebel 2001) is defined based on another heuristics h, such as $h = h^{\mathrm{add}}$, as a subprocedure. For each unachieved subgoal $p \in G \setminus s$, the action a that adds p with the minimal $[\mathrm{COST}(a) + h(s, \mathrm{PRE}(a))]$ is conceptually "the cheapest action that achieves a subgoal p for the first time under delete relaxation", called the *cheapest achiever* / *best supporter* $\mathrm{bs}(p, s, h)$ of p. h^{FF} is defined as the sum of actions in a relaxed plan Π^+ constructed as follows:

$$h^{\mathrm{FF}}(s, G, h) = \sum_{a \in \Pi^+(s,G,h)} \mathrm{COST}(a) \tag{3}$$

$$\Pi^+(s, G, h) = \bigcup_{p \in G} \begin{cases} \emptyset \text{ if } p \in s. \text{ Otherwise,} \\ \{a\} \cup \Pi^+(s, \mathrm{PRE}(a)) \\ \quad \text{where } a = \mathrm{bs}(p, s, h). \end{cases} \tag{4}$$

$$\mathrm{bs}(p, s, h) = \operatorname*{arg\,min}_{\{a \in A | p \in \mathrm{ADD}(a)\}} \left[\mathrm{COST}(a) + h(s, \mathrm{PRE}(a))\right]. \tag{5}$$

Goal Count heuristics h^{GC} is a simple heuristic proposed in (Fikes, Hart, and Nilsson 1972) that counts the number of propositions that are not satisfied yet. $\langle \text{condition} \rangle$ is a cronecker's delta / indicator function that returns 1 when the condition is satisfied.

$$h^{\mathrm{GC}}(s, G) = \sum_{p \in G} [\![p \notin s]\!]. \tag{6}$$

A2 Deferred Evaluation (DE)

DE is a technique that reduces the runtime of search algorithm while sacrificing the memory efficiency. Standard forward-search algorithms evaluate the value of a heuristic function $h(n)$ of each node n, which is called *Eager Evaluation* in comparison to DE. In DE (also sometimes called *Lazy Evaluation*), the search algorithm instead computes and caches the heuristic value of a node's parent p and uses it in order to sort the nodes in the OPEN list. Since the computation

[*]These authors contributed equally.

of heuristic values is the largest bottleneck in the entire search algorithm compared to other operations (e.g., inserting a node into and retrieving the current best node from the OPEN list / priority queue), this saves a large amount of computation. However, this operation also makes the heuristic function less informative because the cost-to-go estimate by the heuristic is not calculated based on the nodes' own state information. Therefore, the search algorithm tends to visit more irrelevant nodes, which results in an increased number of nodes inserted into the OPEN list, thus more memory usage. DE affects the optimality guarantee of the solution, therefore can only be used in a satisficing/agile search scenario.

In the Pyperplan-based experiment performed in this paper, DE should perform worse than eager evaluations because DE trades the number of calls to heuristics with the number of nodes inserted to the tree, which is limited to 10,000. When CPU time is the limiting resource, DE can sometimes solve more instances (Richter and Helmert 2009), assuming the implementation is optimized for speed (e.g., using C++). However, our Python implementation (typically 100–1,000 times slower than C++) is not able to measure this effect because this low-level bottleneck could hide the effect of speed improvements.

A3 Preferred Operators (PO)

In addition to the heuristic value of a state, some heuristic functions are able to return a list of actions called "helpful actions" (Hoffmann and Nebel 2001) or "preferred operators" (Richter and Helmert 2009) and are used by a planner in a variety of ways (e.g., initial incomplete search of FF planner (Hoffmann and Nebel 2001) and alternating open list in LAMA planner (Richter, Westphal, and Helmert 2011)). We reimplemented Schulte and Keller (2014)'s strategy which limits the action selection to the preferred operators and falling back to the normal behavior if there are none. In MCTS terminology (Sec. 2.2), this is a way to modify the tree policy by re-weighting it with a mask.

A4 Detailed Explanation for the Base MCTS for Graph Search

Alg. 1 shows the pseudocode of MCTS adjusted for graph search (Schulte and Keller 2014). Aside from what was described from the main section, it has a node-locking mechanism that avoids duplicate search effort.

Following THTS, our MCTS has a hash table that implements a *CLOSE* list and a *Transposition Table* (TT). A CLOSE list stores the generated states and avoids instantiating nodes with duplicate states. A TT stores various information about the states such as the parent information and the action used at the parent. The close list is implemented by a lock mechanism.

Since an efficient graph search algorithm must avoid visiting the same state multiple times, MCTS for graph search marks certain nodes as *locked*, and excludes them from the selection candidates. A node is locked either (1) when a node is a dead-end that will never reach a goal (detected by having no applicable actions, by a heuristic function, or other facilities), (2) when there is a node with the same state in the search tree with a smaller g-value, (3) when all of its children are locked, or (4) when a node is a goal (relevant in an anytime iterated search setting (Richter, Thayer, and Ruml 2010; Richter, Westphal, and Helmert 2011), but not in this paper). Thus, in the expansion step, when a generated node n has the same state as a node n' already in the search tree, MCTS discards n if $g(n) > g(n')$, else moves the subtree of n' to n and marks n' as locked. It also implicitly detects a cycle, as this is identical to the duplicate detection in Dijkstra/A*/GBFS.

The queueing step backpropagates necessary information from the leaf to the root. Efficient backpropagation uses a priority queue ordered by descending g-value. The queue is initialized with the expanded node p; each newly generated node n that is not discarded is inserted into the queue, and if a node n' for the same state was already present in the tree it is also inserted into the queue. In each backpropagation iteration, (1) the enqueued node with the highest g-value is popped, (2) its information is updated by aggregating its children's information (including the lock status), (3) and its parent is queued.

Algorithm 1 High-level general MCTS. **Input**: Root node r, successor function S, NEC f, heuristic function h, priority queue Q sorted by g. Initialize $\forall n; g(n) \leftarrow \infty$.

```
while True do
  Parent p ← r
  while not leaf p do                                  # Selection
    p ← arg min_{n∈S(p)} f(n)
  Q ← {p}
  for n ∈ S(p) do                                      # Expansion
    return n if n is goal.                  # Early goal detection
    if ∃n' already in tree with same state s_{n'} = s_n then
      if g(n) > g(n') then
        continue
      Lock n', S(n) ← S(n'), Q ← Q ∪ {n, n'}
    else
      Compute h(s_n)                                   # Evaluation
      Q ← Q ∪ {n}
  while n ← Q.POPMAX() do                  # Backpropagation
    Update n's statistics and lock status
    Q ← Q ∪ {n's parent}
```

A5 Proof of MLE

We show the maximum likelihood estimators of Gaussian and Uniform distribution (Thm. 1-5). First, let us restate the definition of MLE:

Definition 1. *Given i.i.d. samples $x_1, \ldots, x_N \sim p(\mathrm{x}|\theta)$, the Maximum Likelihood Estimate (MLE) of the parameter θ is:*

$$\begin{aligned}\hat{\theta} &= \arg\max_\theta p(x_1, \ldots, x_N|\theta) \\ &= \arg\max_\theta \log p(x_1, \ldots, x_N|\theta) \quad (\log \textit{ is monotinic.}) \\ &= \arg\max_\theta \sum_i \log p(x_i|\theta). \quad \because \textit{i.i.d.}\end{aligned}$$

Sometimes the MLE is solved by the following equation for the target parameter θ.

$$\begin{aligned}0 &= \frac{\partial}{\partial\theta} p(x_1, \ldots, x_N|\theta) \\ &= \frac{\partial}{\partial\theta} \log p(x_1, \ldots, x_N|\theta) \\ &= \frac{\partial}{\partial\theta} \log \prod_i p(x_i|\theta) = \frac{\partial}{\partial\theta} \sum_i \log p(x_i|\theta).\end{aligned}$$

An estimator that is obtained by maximizing some quantity is called an *M-estimator*. MLE is an M-estimator. Some M-estimators can be solved by setting the derivatives to zero, in which case they are called *Z-estimator*. Not all M-estimators are solved in this way, as seen below (l and u).

Theorem 1 (MLE of Gaussian)**.** *Given i.i.d. $x_1, \ldots, x_N \sim \mathcal{N}(\mathrm{x}|\mu, \sigma)$, the Maximum Likelihood Estimate of μ is the sample average $\hat{\mu} = \frac{1}{N}\sum_i x_i$. Similarly, the Maximum Likelihood Estimate of σ^2 is the sample variance $\frac{1}{N-1}\sum_i (x_i - \hat{\mu})$.*

Proof. For μ,

$$\begin{aligned}0 &= \frac{\partial}{\partial\mu} \sum_i \log \mathcal{N}(x_i|\mu, \sigma) \\ &= \frac{\partial}{\partial\mu} \sum_i \log \left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp -\frac{(x_i-\mu)^2}{2\sigma^2} \right) \\ &= \frac{\partial}{\partial\mu} \sum_i \left(-\frac{(x_i-\mu)^2}{2\sigma^2} - \log\sqrt{2\pi\sigma^2} \right) \\ &= \frac{\partial}{\partial\mu} \sum_i \left(\frac{-x_i^2 + 2\mu x_i - \mu^2}{2\sigma^2} - \log\sqrt{2\pi\sigma^2} \right) \\ &= \sum_i \frac{2x_i - 2\mu}{2\sigma^2}. \quad \therefore \hat{\mu} = \frac{\sum_i x_i}{N}.\end{aligned}$$

For σ^2,

$$\begin{aligned}0 &= \frac{\partial}{\partial\sigma^2} \sum_i \log \mathcal{N}(x_i|\mu, \sigma) \\ &= \frac{\partial}{\partial\sigma^2} \sum_i \log \left(\frac{1}{\sqrt{2\pi\sigma^2}} \exp -\frac{(x_i-\mu)^2}{2\sigma^2} \right) \\ &= \frac{\partial}{\partial\sigma^2} \sum_i \left(-\frac{(x_i-\mu)^2}{2\sigma^2} - \frac{\log 2\pi + \log\sigma^2}{2} \right) \\ &= \sum_i \left(\frac{(x_i-\mu)^2}{2(\sigma^2)^2} - \frac{1}{2\sigma^2} \right). \\ \therefore \hat{\sigma}^2 &= \frac{\sum_i (x_i-\mu)^2}{N} = \frac{\sum_i (x_i-\hat{\mu})^2}{N-1}.\end{aligned}$$

□

Theorem 2 (MLE of Uniform)**.** *Given i.i.d. $x_1, \ldots, x_N \sim U(\mathrm{x}|l, u)$, $\hat{u} = \max_i x_i$ and $\hat{l} = \min_i x_i$.*

Proof. For u, we cannot use the differentiation-based proof. Instead, we maximize the log likelihood directly.

$$\begin{aligned}&\sum_i \log U(x_i|l, u) \\ &= \sum_i \log \frac{1}{u-l} \\ &= \sum_i -\log(u-l).\end{aligned}$$

The value of u that maximizes this formula is the infimum of possible values u can take. Since $\forall i; x_i \le u$, $u = \max_i x_i$.

Similarly, the value of l that maximizes this formula is the supremum of possible values l can take. Since $\forall i; l \le x_i$, $l = \min_i x_i$. □

A6 Proof of Bandit Algorithms (Overview)

This section serves as a tutorial for understanding our main proof in the later sections. To help understand the proof of various confidence bounds, we first describe the general procedure for proving the regret of bandit algorithms, demonstrate the proof of UCB1 using this scheme, then finally show the proof of other bandits.

The ingredients for proving an upper/lower confidence bound are as follows:

- **Ingredient 1: The main term and the exploration term.** For example, in the standard UCB1 (Auer, Cesa-Bianchi, and Fischer 2002), the *main term* is the empirical mean $\hat{\mu}$ while the *exploration term* is $c\sqrt{\frac{2\log T}{n_i}}$. Their forms are heavily affected by the proof of the upper bound on the regret, and an arbitrary exploration term is not guaranteed to have a provable bound.
- **Ingredient 2: A specification of reward distributions.** For example, in the standard UCB1 (Auer, Cesa-Bianchi, and Fischer 2002), one assumes a reward distribution bounded in $[0, c]$. Different algorithms assume different reward distributions, and in general, more information about the distribution gives a tighter bound (and faster convergence). For example, one can assume an unbounded distribution with known variance, etc.
- **Ingredient 3: A concentration inequality.** It is also called a tail probability bound. For example, in the standard UCB1, one uses Hoeffding's inequality. Different algorithms use different inequalities to prove the bound, based on what reward distribution it assumes and what main term it uses. Examples include the Chernoff bound, Chebishev's inequality, Bernstein's inequality, Bennett's inequality, etc. Note that the inequality may be two-sided or one-sided.

The general procedure for proving the bound is as follows.

1. Let the main term be a random variable X_n and the exploration term be δ. Then write down the concentration inequality for X_n as follows.
   - $P(|\mathrm{X}_n - \mathbb{E}[\mathrm{X}_n]| \geq \delta) \leq F(\delta)$. (two-sided)

   F is an inequality-specific formula. If necessary, simplify the inequality based on the assumptions made in the reward distribution, e.g., bounds, mean, variance.
2. Expand $|\mathrm{X}_n - \mathbb{E}[\mathrm{X}_n]| \geq \delta$ into $\delta \geq \mathrm{X}_n - \mathbb{E}[\mathrm{X}_n] \geq -\delta$.
3. Change the notations to model the bandit problem because each concentration inequality is a general statement about RVs. Before this step, the notation was:
   - n (number of samples)
   - X_n is a function of i.i.d. random variables $(\mathrm{x}_1, \ldots, \mathrm{x}_n)$
   - $\mathbb{E}[\mathrm{X}_n] = \mathbb{E}[\mathrm{x}_1] = \ldots = \mathbb{E}[\mathrm{x}_n]$ is assumed.

   For example,
   - $\mu_n = \frac{1}{n}\sum_{i=1}^{n} \mathrm{x}_i$
   - $\mathbb{E}[\mu_n] = \mathbb{E}[\mathrm{x}_1] = \ldots = \mathbb{E}[\mathrm{x}_n]$

   After the change, they correspond to:
   - n_i (number of pulls of arm i).
   - $\hat{X}_i$ (empirical value of arm i from n_i pulls),
   - X_i (true value of arm i).

   For example,
   - $\hat{\mu}_i$ (sample mean of arm i from n_i pulls),
   - μ_i (true mean of arm i),
4. Let i be a suboptimal arm, $*$ be an optimal arm, $\mathrm{UCB}_i = \hat{\mu}_i + \delta$, and $\mathrm{LCB}_i = \hat{\mu}_i - \delta$. Derive the relationship between δ and the gap $\Delta_i = \mu_i - \mu_*$ so that the following conditions for the best arm holds:
   - $\mathrm{UCB}_i \leq \mathrm{UCB}_*$ (for maximization)
   - $\mathrm{LCB}_i \geq \mathrm{LCB}_*$ (for minimization)

   This results in $2\delta \leq \Delta_i$.
5. Replace the δ with the exploration term. For example, in UCB1, $\delta = \sqrt{\frac{2\log T}{n_i}}$.
6. Derive the lower bound L for n_i from $2\delta \leq \Delta_i$.
7. Find the upper-bound of the probability of selecting a suboptimal arm i. This is typically done by a union-bound argument.
8. Derive the upper bound of the expected number of pulls $\mathbb{E}[n_i]$ of a suboptimal arm i using a triple loop summation. This is typically the heaviest part that needs mathematical tricks. The tricks do not seem generally transferable between approaches.
9. Finally, derive an upper bound of the regret $TX_* - \sum_{i=1}^{K} X_i\mathbb{E}[n_i]$ by

$$TX_* - \sum_{i=1}^{K} X_i\mathbb{E}[n_i] = \sum_{i=1}^{K}(X_* - X_i)\mathbb{E}[n_i] = \sum_{i=1}^{K}\Delta_i\mathbb{E}[n_i].$$

A6.1 The Proof of UCB1 (Tutorial Example)

We prove the logarithmic upper bound of the cumulative regret of the UCB1 $\hat{\mu}_i - c\sqrt{\frac{2\log T}{n_i}}$ where $\hat{\mu}_i$ is the empirical mean of samples from arm i, n_i is the number of pulls from arm i, and $T = \sum_{i=1}^{K} n_i$ is the total pulls from all K arms.

1. UCB1 assumes a reward distribution with a known bound. For such a distribution, we can use Hoeffding's inequality. Given RVs $\mathrm{x}_1 \dots \mathrm{x}_n$, where $\mathrm{x}_i \in [l_i, u_i]$, and their sum $\mathrm{S}_n = \sum_{i=1}^{n} \mathrm{x}_i$,

   $$P(|\mathrm{S}_n - \mathbb{E}[\mathrm{S}_n]| \geq \epsilon) \leq 2\exp -\frac{2\epsilon^2}{\sum_{i=1}^{n}(u_i - l_i)^2}.$$

   Using $\delta = \frac{\epsilon}{n}$ and $\mu_n = \frac{\mathrm{S}_n}{n}$,

   $$P(|\mu_n - \mathbb{E}[\mu_n] \geq \delta|) \leq 2\exp -\frac{2n^2\delta^2}{\sum_{i=1}^{n}(u_i - l_i)^2}.$$

   UCB1 assumes x_i are i.i.d. copies, thus $\forall i; u_i - l_i = c$.

   $$P(|\mu_n - \mathbb{E}[\mu_n] \geq \delta|) \leq 2\exp -\frac{2n^2\delta^2}{nc^2} = 2\exp -\frac{2n\delta^2}{c^2}.$$

2. Expanding the two-sided error:

   $$\delta \geq \mu_n - \mathbb{E}[\mu_n] \geq -\delta.$$

3. Changing the notation:

   $$\delta \geq \hat{\mu}_i - \mu_i \geq -\delta.$$

4. Adding $\mu_i - \delta$ to both sides,

   $$\mu_i \geq \hat{\mu}_i - \delta = \mathrm{LCB}_i(T, n_i) \geq \mu_i - 2\delta.$$

   Substituting $i = *$ (optimal arm), the first inequality is

   $$\mu_* \geq \hat{\mu}_* - \delta = \mathrm{LCB}_*(T, n_*).$$

   Assuming $2\delta \leq \Delta_i = \mu_i - \mu_*$, the second inequality is

   $$\mathrm{LCB}_i(T, n_i) \geq \mu_i - 2\delta \geq \mu_i - \Delta_i = \mu_*.$$

   Therefore

   $$\mathrm{LCB}_i(T, n_i) \geq \mu_* \geq \mathrm{LCB}_*(T, n_*).$$

5. Let $\delta = c\sqrt{\frac{2\log T}{n_i}}$. Then

   $$P(\mu_{n_i} - \mathbb{E}[\mu_{n_i}] \geq \delta) \leq \exp -\frac{2n_i c^2 \frac{2\log T}{n_i}}{c^2} = T^{-4}.$$

6. From $2\delta \leq \Delta_i$, considering n_i is an integer,

   $$2c\sqrt{\frac{2\log T}{n_i}} \leq \Delta_i \Leftrightarrow 4c^2\frac{2\log T}{n_i} \leq \Delta_i^2$$
   $$\Leftrightarrow \frac{8c^2\log T}{\Delta_i^2} \leq \left\lceil \frac{8c^2\log T}{\Delta_i^2} \right\rceil = L \leq n_i.$$

7. $\mathrm{LCB}_i(T, n_i) \geq \mu_* \geq \mathrm{LCB}_*(T, n_*)$ does not hold when either inequality does not hold. $\mathrm{LCB}_i(T, n_i) \geq \mu_*$ does not hold with probability less than T^{-4}. $\mu_* \geq \mathrm{LCB}_i(T, n_*)$ does not hold with probability less than T^{-4}. Thus, by union-bound (probability of disjunctions),

   $$P(\mathrm{LCB}_i(T, n_i) \leq \mathrm{LCB}_*(T, n_*)) \leq 2T^{-4}.$$

8. Assume we followed the UCB1 strategy, i.e., we pulled the arm that minimizes the LCB. The expected number of pulls $\mathbb{E}[n_i]$ from a suboptimal arm i is as follows. Note that for K arms, every arm is at least pulled once.

   $$\begin{aligned}
   \mathbb{E}[n_i] &= 1 + \sum_{t=K+1}^{T} P(i \text{ is pulled at time } t) \\
   &\leq L + \sum_{t=K+1}^{T} P(i \text{ is pulled at time } t \wedge n_i > L) \\
   &= L + \sum_{t=K+1}^{T} P(\forall j; \mathrm{LCB}_j(t, n_j) \geq \mathrm{LCB}_i(t, n_i)) \\
   &\leq L + \sum_{t=K+1}^{T} P(\mathrm{LCB}_*(t, n_*) \geq \mathrm{LCB}_i(t, n_i)) \\
   &\leq L + \sum_{t=K+1}^{T} P(\exists u, v; \mathrm{LCB}_*(t, u) \geq \mathrm{LCB}_i(t, v)) \\
   &\leq L + \sum_{t=K+1}^{T} \sum_{u=1}^{t-1} \sum_{v=L}^{t-1} P(\mathrm{LCB}_*(t, u) \geq \mathrm{LCB}_i(t, v)) \\
   &\leq L + \sum_{t=K+1}^{T} \sum_{u=1}^{t-1} \sum_{v=L}^{t-1} 2t^{-4} \\
   &\leq L + \sum_{t=1}^{\infty} \sum_{u=1}^{t} \sum_{v=1}^{t} 2t^{-4} = L + \sum_{t=1}^{\infty} t^2 \cdot 2t^{-4} \\
   &= L + 2\sum_{t=1}^{\infty} t^{-2} = L + 2 \cdot \frac{\pi}{6} = L + \frac{\pi}{3} \\
   &\leq c^2\frac{8\log T}{\Delta_i^2} + 1 + \frac{\pi}{3} \quad \because \lceil x \rceil \leq x + 1
   \end{aligned}$$

9. The regret is

   $$\begin{aligned}
   T\mu_* - \sum_{i=1}^{K} \mu_i \mathbb{E}[n_i] &= \sum_{i=1}^{K} (\mu_* - \mu_i)\mathbb{E}[n_i] = \sum_{i=1}^{K} \Delta_i \mathbb{E}[n_i] \\
   &\leq \sum_{i=1}^{K} \Delta_i \left(c^2\frac{8\log T}{\Delta_i^2} + 1 + \frac{\pi}{3} \right) \\
   &\leq \sum_{i=1}^{K} \left(c^2\frac{8\log T}{\Delta_i} + \left(1 + \frac{\pi}{3}\right)\Delta_i \right).
   \end{aligned}$$

A7 The Proof of UCB1-Uniform's Regret Bound

Recall the definitions of various LCBs.

$$\begin{array}{lcccl}
\text{LCB1}_i & = & \hat{\mu} & - & c\sqrt{(2\log T)/n_i}. \\
\text{LCB1-Normal}_i & = & \hat{\mu} & - & \hat{\sigma}\sqrt{(16\log T)/n_i}. \\
\text{LCB1-Normal2}_i & = & \hat{\mu} & - & \hat{\sigma}\sqrt{2\log T}. \\
\text{LCB1-Uniform}_i & = & \frac{\hat{u}+\hat{l}}{2} & - & (\hat{u}-\hat{l})\sqrt{6n_i\log T}.
\end{array}$$

In the definition of UCB1/-Normal/-Normal2, the main estimate $\hat{\mu}$ is defined by the sum of i.i.d. random variables for individual rewards, which made it possible to use Hoeffding's inequality in its proof for regret bounds. In contrast, the main estimate $\frac{\hat{u}+\hat{l}}{2}$ of UCB1-Uniform is not defined from the sum of data points, i.e., $\hat{l} = \min_i x_i$ uses a minimum, and $\hat{u} = \max_i x_i$ uses a maximum. To handle these cases, we need concentration inequalities for more general functions other than sums. Among many concentration inequalities for general functions (e.g., Efron-Stein inequality, Han's inequality, logarithmic Sobolev inequality), we found that *bounded differences inequality* (Boucheron, Lugosi, and Massart 2013) can be used as the basis of the proof.

A7.1 Preliminary: Bounded Difference Inequality

To use the inequality, we must assume a function with bounded differences, which we define first.

Definition 2 (Functions with Bounded Differences)**.** *A n-ary function $g : X^n \to \mathbb{R}$ for a set X has a* bounded difference *when, for all $1 \le i \le n$, there exists a constant $c_i \in \mathbb{R}$ such that*

$$\max_{y \in X} \left| \begin{array}{c} g(\ldots, x_{i-1}, x_i, x_{i+1}, \ldots) \\ -g(\ldots, x_{i-1}, y, x_{i+1}, \ldots) \end{array} \right| \le c_i.$$

Note that an MLE of a parameter obtained from n sample points $(x_1, \ldots, x_n)$ are n-ary function. Moreover, quantities defined from the parameters obtained by MLE are also one of them. For example, the MLE is u in $U(l, u)$ is $\hat{u} = \max_i x_i$, which is a function of $(x_1, \ldots, x_n)$, and the mean $\frac{\hat{l}+\hat{u}}{2}$ of the fitted distribution $U(\hat{l}, \hat{u})$ is also a function of $(x_1, \ldots, x_n)$.

When the distribution function of a random variable has bounded differences, then we can bound its tail probability as follows. This inequality generalizes Hoeffding's inequality by $g(\mathrm{x}_1, \ldots, \mathrm{x}_n) = \sum_{i=1}^n \mathrm{x}_i$.

Theorem 3 (Bounded Differences Inequality)**.** *When a n-ary function $g : X^n \to \mathbb{R}$ for a set X has a bounded difference $c_1, \ldots c_n$ for each argument, let $C = \sum_{i=1}^n c_i^2$. Then a random variable $\mathrm{z}_n = g(\mathrm{x}_1, \ldots, \mathrm{x}_n)$ satisfies*

$$P(|\mathrm{z}_n - \mathbb{E}[\mathrm{z}_n]| \ge \delta) \le \exp -2\delta^2/C.$$

Theorem 4 (Hoeffding's Inequality)**.** *Given random variables $\mathrm{x}_i \in [a, b]$, let their sum be $\mathrm{S}_n = \sum_{i=1}^n \mathrm{x}_i$ and $C = \sum_{i=1}^n |b_i - a_i|^2$. Then S_n satisfies*

$$P(|\mathrm{S}_n - \mathbb{E}[\mathrm{S}_n]| \ge \delta) \le \exp -2\delta^2/C.$$

Due to the similarity of the form, once we show that the quantities used in LCB-Uniform have bounded difference, it is straightforward to prove its regret.

A7.2 Preliminary: Order Statistics

The proof of UCB1-Uniform relies on several theorems in order statistics (Hryniv 2017).

Definition 3 (Order Variable)**.** *Assume n random variables $\mathrm{x}_1, \ldots \mathrm{x}_n$. Let $\mathrm{x}_{(k)}$ denote a k-th* order variable *defined as*

$$\mathrm{x}_{(k)} = \textit{the } k\textit{-th largest of } \mathrm{x}_1, \ldots \mathrm{x}_n.$$

In other words, $\mathrm{x}_{(1)} \le \mathrm{x}_{(2)} \le \ldots \le \mathrm{x}_{(n)}$, where $\mathrm{x}_{(1)}$ is the smallest and $\mathrm{x}_{(k)}$ is the largest.

Definition 4 (Range Variable)**.** *Given order variables $\mathrm{x}_{(1)}, \ldots \mathrm{x}_{(n)}$, a* range variable *is $\mathrm{R}_n = \mathrm{x}_{(n)} - \mathrm{x}_{(1)}$.*

Lemma 1 (CDF of a Range Variable)**.** *Assume n i.i.d. random variables $\mathrm{x}_1, \ldots \mathrm{x}_n$ with a CDF F and a PDF f. The cumulative distribution function of R_n is given by*

$$P(\mathrm{R}_n < r) = n \int (F(x + r) - F(x))^{n-1} f(x) dx.$$

A7.3 The mean of Uniform has a bounded difference

To prove the regret bound for UCB1-Uniform, we first prove two lemmas to show that the mean $\frac{u+l}{2}$ of a Uniform distribution $U(l, u)$ has a bounded difference.

Lemma 2 (CDF of a Range Variable for $U(0, 1)$)**.** *Assume n i.i.d. random variables $\mathrm{x}_1, \ldots \mathrm{x}_n \sim U(0, 1)$. The cumulative distribution function of R_n is $P(\mathrm{R}_n < r) = nr^{n-1}$.*

Proof. $P(\mathrm{R}_n < r)$

$$= n \int_0^1 ((x + r) - x)^{n-1} \cdot 1 \cdot dx$$

$$= n \int_0^1 r^{n-1} dx = n \left\langle r^{n-1} x \right\rangle_0^1 = nr^{n-1}(1 - 0) = nr^{n-1}.$$

□

Lemma 3. *Assume i.i.d. RVs $\mathrm{x}_1, \ldots \mathrm{x}_n \sim U(l, u)$, $\mathrm{x}_{(1)} = \hat{l}$ and $\mathrm{x}_{(n)} = \hat{u}$. Then $\mathrm{z}_n = g(\mathrm{x}_1, \mathrm{x}_2, \ldots, \mathrm{x}_n) = \frac{\hat{u}+\hat{l}}{2} = \frac{\mathrm{x}_{(n)}+\mathrm{x}_{(1)}}{2}$ has a bounded difference $u - l$.*

Proof. First, $g(\mathrm{x}_1, \ldots \mathrm{x}_i, \ldots \mathrm{x}_n) \in [l, u]$ because

$$l = \frac{l + l}{2} \le \frac{\mathrm{x}_{(n)} + \mathrm{x}_{(1)}}{2} \le \frac{u + u}{2} = u.$$

The same trivially holds for $g(\mathrm{x}_1, \ldots y, \ldots \mathrm{x}_n) \in [l, u]$ when $y \in [l, u]$. Their difference is largest when either one is u and another one is l. Thus

$$| \ g(\ldots, \mathrm{x}_i, \ldots) - g(\ldots, y, \ldots) \ | \le u - l = c_i.$$

□

This gives us a following concentration inequality based on $C = \sum_i c_i^2 = n(u - l)^2$:

$$P(A : |\mathrm{z}_n - \mathbb{E}\mathrm{z}_n| \ge \delta) \le \exp -\frac{2\delta^2}{n(u - l)^2}.$$

A7.4 The Proof of UCB1-Uniform

Now we prove the upper bound of the cumulative regret of the LCB1-Uniform parameterized by b, LCB1-Uniform$_i(b) = \frac{\hat{u}_i+\hat{l}_i}{2} - (\hat{u}_i - \hat{l}_i)\sqrt{bn_i \log T}$, where $\hat{u}_i, \hat{l}_i$ are the empirical max/min of samples from arm i, n_i is the number of pulls from arm i, and $T = \sum_{i=1}^{K} n_i$ is the total pulls from all K arms. Later, the best b is determined to be $b = 6$.

1. As shown above.
2. Same as UCB1.
3. Same as UCB1.
4. Same as UCB1.
5. Let $\delta = (\hat{u}_i - \hat{l}_i)\sqrt{bn_i \log T}$. and $r_i = \frac{\hat{u}_i - \hat{l}_i}{u_i - l_i} \in [0, 1]$. Then

$$P(A) \leq \exp -\frac{2(\hat{u}_i - \hat{l}_i)^2 \cdot bn_i \log T}{n_i(u_i - l_i)^2} = \exp -2br_i^2 \log T = T^{-2br_i^2}.$$

   The trick starts here. The value of $r_i = \frac{\hat{u}_i - \hat{l}_i}{u_i - l_i}$ is unavailable because we do not know the true bounds u_i and l_i. However, if event $B : r_i \geq X$ holds for some constant X, we would obtain a more convenient form T^{-2bX^2}:

$$P(A) \leq T^{-2br_i^2} \leq T^{-2bX^2}.$$

   Assume $P(B) = \alpha$. Note that $r_i = \frac{\hat{u}_i - \hat{l}_i}{u_i - l_i}$ is a range variable for n_i i.i.d. RVs from $U(0, 1)$. Therefore, from its CDF (Lemma. 2), for some $X \in [0, 1]$,

$$P(r_i < X) = nX^{n-1}.$$

   Solving X for $P(r_i < X) = P(\neg B) = 1 - \alpha$,

$$X = X_{n,1-\alpha} = \left(\frac{1-\alpha}{n}\right)^{\frac{1}{n-1}}.$$

   A and B could be correlated. Using union-bound

$$\begin{aligned} P(\neg(A \wedge B)) = P(\neg A \vee \neg B) &\leq P(\neg A) + P(\neg B) \\ 1 - P(A \wedge B) &\leq 1 - P(A) + P(\neg B) \\ \therefore P(A) &\leq P(A \wedge B) + P(\neg B) \\ &= T^{-2gX_{n,1-\alpha}^2} + 1 - \alpha. \end{aligned}$$

6. From $2\delta \leq \Delta_i$ and using the fact that n_i is an integer,

$$\begin{aligned} \Delta_i &\geq 2(\hat{u}_i - \hat{l}_i)\sqrt{bn_i \log T} \\ \Leftrightarrow \Delta_i^2 &\geq 4b(\hat{u}_i - \hat{l}_i)^2 n_i \log T \\ &= 4b(u_i - l_i)^2 r_i^2 n_i \log T \\ &\geq 4b(u_i - l_i)^2 X_{n_i,1-\alpha}^2 n_i \log T \ (\because B) \\ &= 4b(u_i - l_i)^2 \left(\frac{1-\alpha}{n_i}\right)^{\frac{2}{n_i-1}} n_i \log T \\ &\geq 4b(u_i - l_i)^2 \left(\frac{1-\alpha}{n_i}\right)^{2} n_i \log T \\ &\quad (\because 0 < \frac{1-\alpha}{n_i} < 1,\ \frac{2}{n_i - 1} < 2) \\ \Leftrightarrow n_i &\geq \frac{4b(u_i - l_i)^2(1-\alpha)^2 \log T}{\Delta_i^2} \\ \therefore n_i &\geq L = \left\lceil \frac{4b(u_i - l_i)^2(1-\alpha)^2 \log T}{\Delta_i^2} \right\rceil \end{aligned}$$

7. Using the same union-bound argument as UCB1, $P(\mathrm{LCB}_i(T, n_i) \leq \mathrm{LCB}_*(T, n_*)) \leq 2T^{-2bX_{n_i,1-\alpha}} + 2 - 2\alpha$.
8. For a reason explained later, we require each arm to be pulled at least M times. For a fixed α, the solution $X_{n_i,1-\alpha}$ of X for $P(r_i < X) = 1 - \alpha$ increases monotonically as n_i increases. Therefore, $X_{n_i,1-\alpha} \geq X_{M,1-\alpha}$ under $n_i \geq M$. Then, using the same argument as UCB1,

$$\begin{aligned} \mathbb{E}[n_i] &\leq L + \sum_{t=1}^{T} t^2(2t^{-2bX_{M,1-\alpha}} + 2(1-\alpha)) \\ &\leq L + 2\sum_{t=1}^{\infty} t^{2-2bX_{M,1-\alpha}} + 2(1-\alpha)\sum_{t=1}^{T} t^2 \\ &\leq L + 2C + \frac{(1-\alpha)T(T+1)(2T+1)}{3} \\ &\leq \frac{4b(u_i - l_i)^2(1-\alpha)^2 \log T}{\Delta_i^2} + 1 \\ &\quad + 2C + \frac{(1-\alpha)T(T+1)(2T+1)}{3} \\ &\quad (\because \lceil x \rceil \leq x + 1) \end{aligned}$$

   We introduced M because C is convergent only when the exponentiator of t is below -1. We consider the condition in which an integer M exists for a given α. Consider:

$$\begin{aligned} 2 - 2bX_{x,1-\alpha} = 2 - 2b\left(\frac{1-\alpha}{x}\right)^{\frac{2}{x-1}} &< -1 \\ \Leftrightarrow f(x) = x^2\left(\frac{3}{2b}\right)^{x-1} &< (1-\alpha)^2 \end{aligned}$$

   Since $1 - \alpha \in [0, 1]$, this is satisfiable only when $0 \leq f(x) \leq 1$. Positivity is trivial. To satisfy $\forall x \geq M; f(x) \leq 1$, it is sufficient if $f(M) \leq 1$ and $\forall x \geq M; f'(x) < 0$. From the derivative below,

$$f'(x) = x\left(\frac{3}{2b}\right)^{x-1}\left(2 + x \log \frac{3}{2b}\right),$$

   this condition is achieved when $3/2b < 1$. If $b = 2$, then $f(22) > 1 > f(23)$, thus it requires too many initialization pulls: $M = 23$. For MCTS where we evaluate each node at least once, we wish to achieve $M = 2$. $b = 6$ achieves $f(2) = 1$, thus is the ideal parameter for MCTS.
9. The regret is polynomial.

$$\begin{aligned} Tz_* - \sum_{i=1}^{K} z_i \mathbb{E}[n_i] &= \sum_{i=1}^{K}(z_* - z_i)\mathbb{E}[n_i] = \sum_{i=1}^{K} \Delta_i \mathbb{E}[n_i] \\ &\leq \sum_{i=1}^{K} \Delta_i \left(\begin{array}{l} \frac{4b(u_i - l_i)^2(1-\alpha)^2 \log T}{\Delta_i^2} + 1 + 2C \\ + \frac{(1-\alpha)T(T+1)(2T+1)}{3} \end{array} \right). \end{aligned}$$

A8 Max-k Bandits vs. UCB1-Uniform

Our methods can be easily confused with a similarly-named bandit framework called the *Max k-Armed Bandit* (Cicirello and Smith 2004, 2005; Streeter and Smith 2006b,a; Achab et al. 2017), later rediscovered as *Extreme Bandit* (Carpentier and Valko 2014). While both UCB1-Uniform and Max-k bandits are based on Extreme Value Theory, **UCB1-Uniform is not a Max-k bandit algorithm**. Also, the Max-k bandit fails to conceptually align with the classical planning application. Moreover, existing Max-k bandits target long-tail distributions while we target short-tail distributions.

Our UCB1-Uniform maximizes the cumulative reward by estimating the reward distribution using the maximum over the reward samples collected in the subtree. This is formalized as optimizing the cumulative regret, which is the difference between the (unknown) *expected* rewards of a suboptimal arm and the best arm. In contrast, the Max-k bandit aims at *maximizing the single maximum reward sampled/obtained*, which is a significantly different objective. It is formalized as optimizing the *extreme regret*, which is the difference between the (unknown, as yet unseen) *maximum* rewards of a suboptimal arm and the best arm, *We merely use the extreme (max/min) information for the cumulative regret minimization, and do not minimize the extreme regret.*

In addition, *Max-k bandit does not make sense in classical planning*. If we use a Max-k bandit in classical planning, we would use the minimization version that estimates the (yet unseen / unknown) minimum heuristic value in a subtree. **However, we know as a fact that the minimum possible heuristic value is 0** for **any** state from which the goal is reachable, because goal-aware heuristics are guaranteed to be 0 at goal states, and otherwise ∞ because they are dead-ends. In other words, Max-k bandit in classical planning estimates whether the goal is reachable or not. Such an estimate (0 or ∞) does not help the search algorithm *approach* the goal. Indeed, we are not interested in whether the goal is reachable (as we by default assume it is). Approaching the goal requires knowing how far the leaves in a subtree tend to be from the goal on average (cumulative regret).

Moreover, existing Max-k bandits target *long-tail distributions*. For example, ExtremeHunter is designed for heavy-tailed reward distributons, and estimates the parameters of a Pareto distribution, which is a subclass of Generalized Pareto distribution with $\xi > 0$. On the contrary, our approach (UCB1-Uniform) targets short-tailed distribution with a finite bound, which is a subclass of Generalized Pareto distribution with $\xi < 0$.

A9 Additional Results

A9.1 Cumulative Histograms for All Heuristics and All Search Statistics

Fig. A1-A3 show the cumulative histogram of the number of instances solved by our Pyperplan implementation within a particular evaluation/expansion/runtime limit.

References

Achab, M.; Clémençon, S.; Garivier, A.; Sabourin, A.; and Vernade, C. 2017. Max K-Armed Bandit: On the ExtremeHunter Algorithm and Beyond. In *Proc. of ECMLKDD*, 389–404. Springer.

Auer, P.; Cesa-Bianchi, N.; and Fischer, P. 2002. Finite-Time Analysis of the Multiarmed Bandit Problem. *Machine Learning*, 47(2-3): 235–256.

Bonet, B.; and Geffner, H. 2001. Planning as Heuristic Search. *Artificial Intelligence*, 129(1): 5–33.

Boucheron, S.; Lugosi, G.; and Massart, P. 2013. *Concentration Inequalities: A Nonasymptotic Theory of Independence*. Oxford University Press.

Bylander, T. 1996. A Probabilistic Analysis of Prepositional STRIPS Planning. *Artificial Intelligence*, 81(1): 241–271.

Carpentier, A.; and Valko, M. 2014. Extreme Bandits. *Advances in Neural Information Processing Systems*, 27.

Cicirello, V. A.; and Smith, S. F. 2004. Heuristic Selection for Stochastic Search Optimization: Modeling Solution Quality by Extreme Value Theory. In *Proc. of CP*, 197–211. Springer.

Cicirello, V. A.; and Smith, S. F. 2005. The MAX K-Armed Bandit: A New Model of Exploration Applied to Search Heuristic Selection. In *Proc. of AAAI*, volume 3, 1355–1361.

Fikes, R. E.; Hart, P. E.; and Nilsson, N. J. 1972. Learning and Executing Generalized Robot Plans. *Artificial Intelligence*, 3(1-3): 251–288.

Hoffmann, J.; and Nebel, B. 2001. The FF Planning System: Fast Plan Generation through Heuristic Search. *J. Artif. Intell. Res.(JAIR)*, 14: 253–302.

Hryniv, O. 2017. Durham University Lecture Note. Probability III-IV – MATH3211/4131.

Richter, S.; and Helmert, M. 2009. Preferred Operators and Deferred Evaluation in Satisficing Planning. In *Proc. of ICAPS*, 273–280.

Richter, S.; Thayer, J. T.; and Ruml, W. 2010. The Joy of Forgetting: Faster Anytime Search via Restarting. In *Proc. of ICAPS*.

Richter, S.; Westphal, M.; and Helmert, M. 2011. LAMA 2008 and 2011. In *Proc. of IPC*, 117–124.

Schulte, T.; and Keller, T. 2014. Balancing Exploration and Exploitation in Classical Planning. In *Proc. of SOCS*.

Streeter, M. J.; and Smith, S. F. 2006a. A Simple Distribution-Free Approach to the Max K-Armed Bandit Problem. In *Proc. of CP*, 560–574. Springer.

Streeter, M. J.; and Smith, S. F. 2006b. An Asymptotically Optimal Algorithm for the Max K-Armed Bandit Problem. In *Proc. of AAAI*, 135–142.

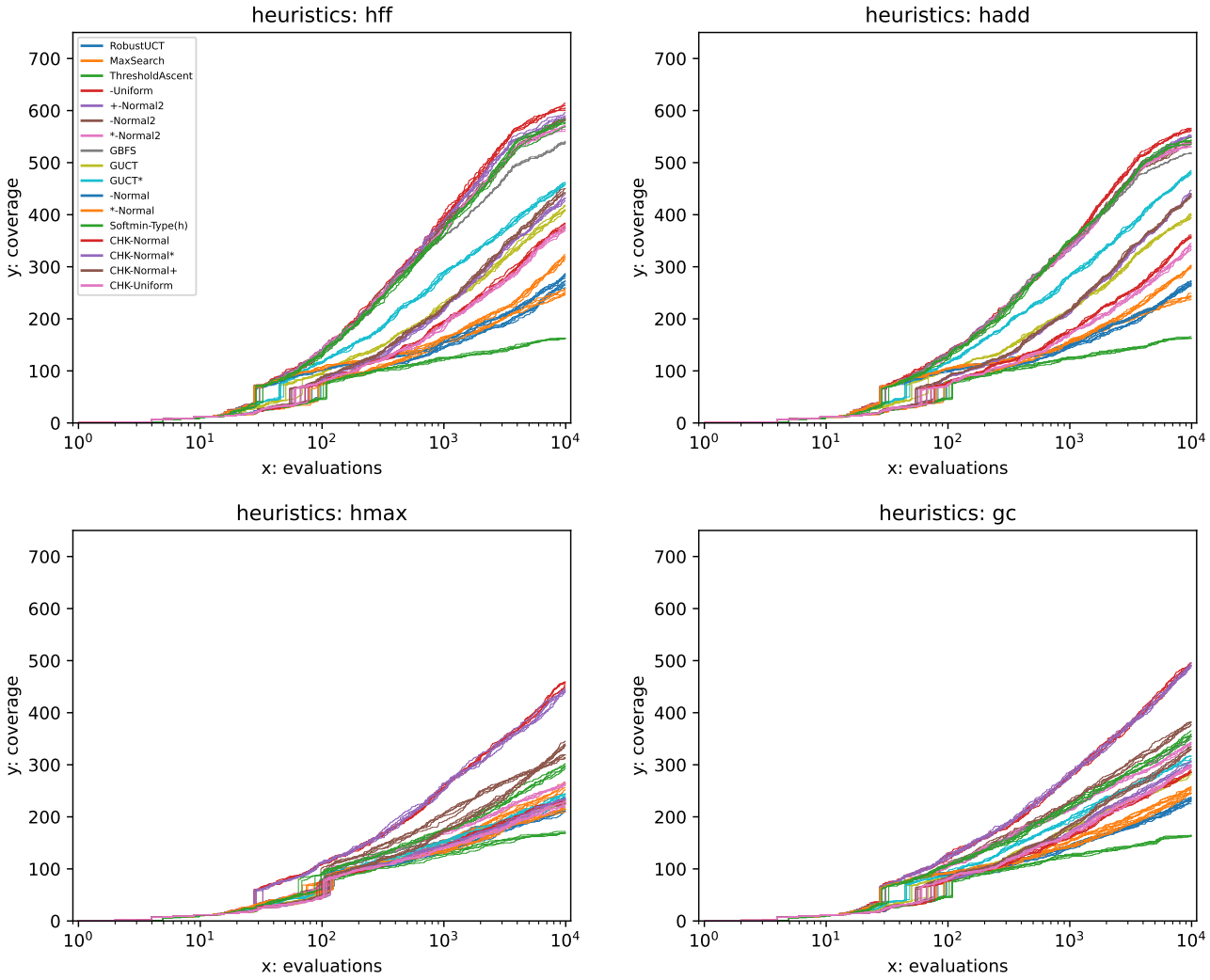

Figure A1: The cumulative histogram of the number of problem instances solved (y-axis) below a certain number of node evaluations (x-axis, 10,000 nodes maximum). Each line represents a random seed. The total numbers at the limit differ from those in other plots (this result does not limit the expansions or the runtime).

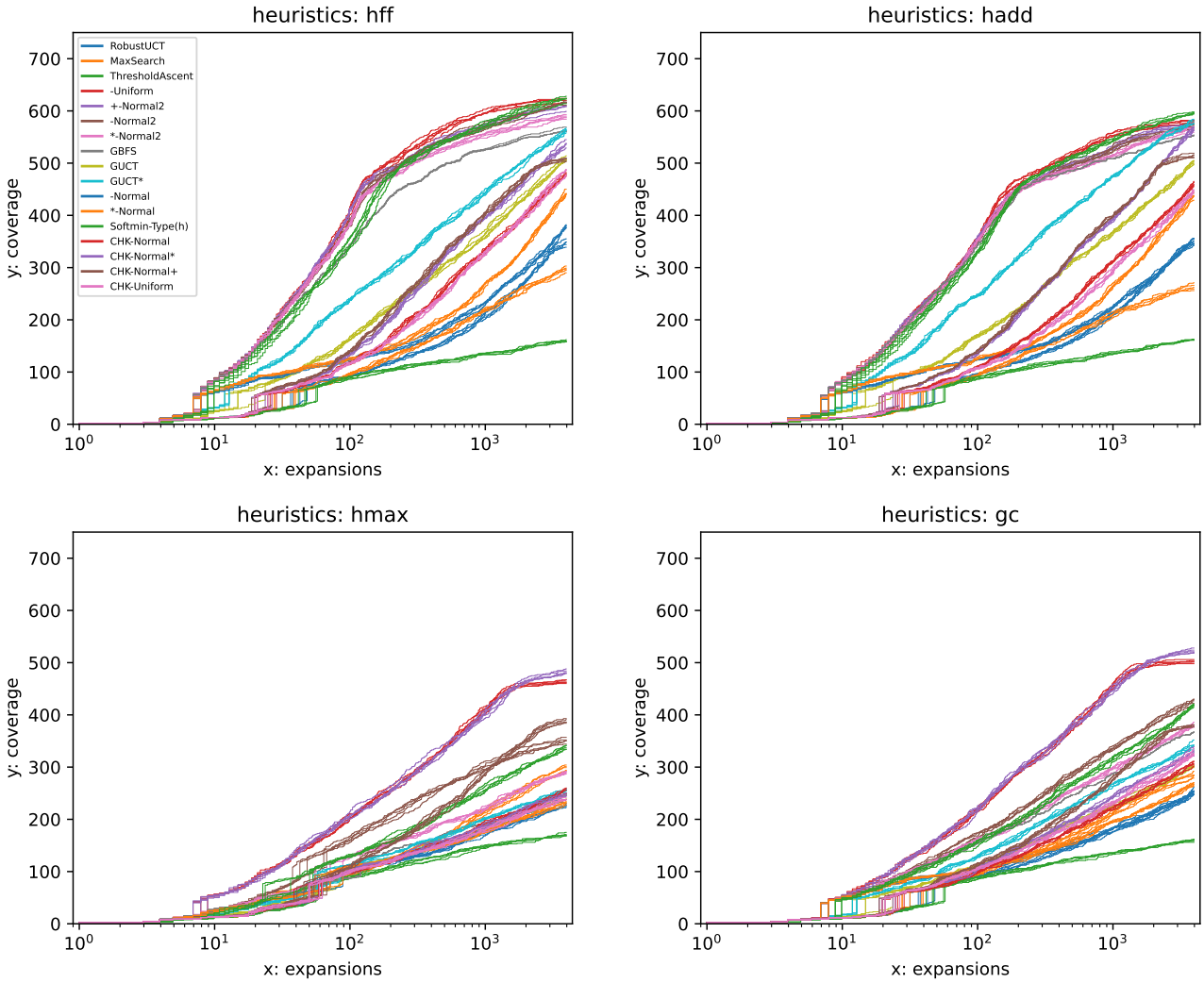

Figure A2: The cumulative histogram of the number of problem instances solved (y-axis) below a certain number of node expansions (x-axis, 4,000 nodes maximum). Each line represents a random seed. The total numbers at the limit differ from those in other plots (this result does not limit the evaluations or the runtime).

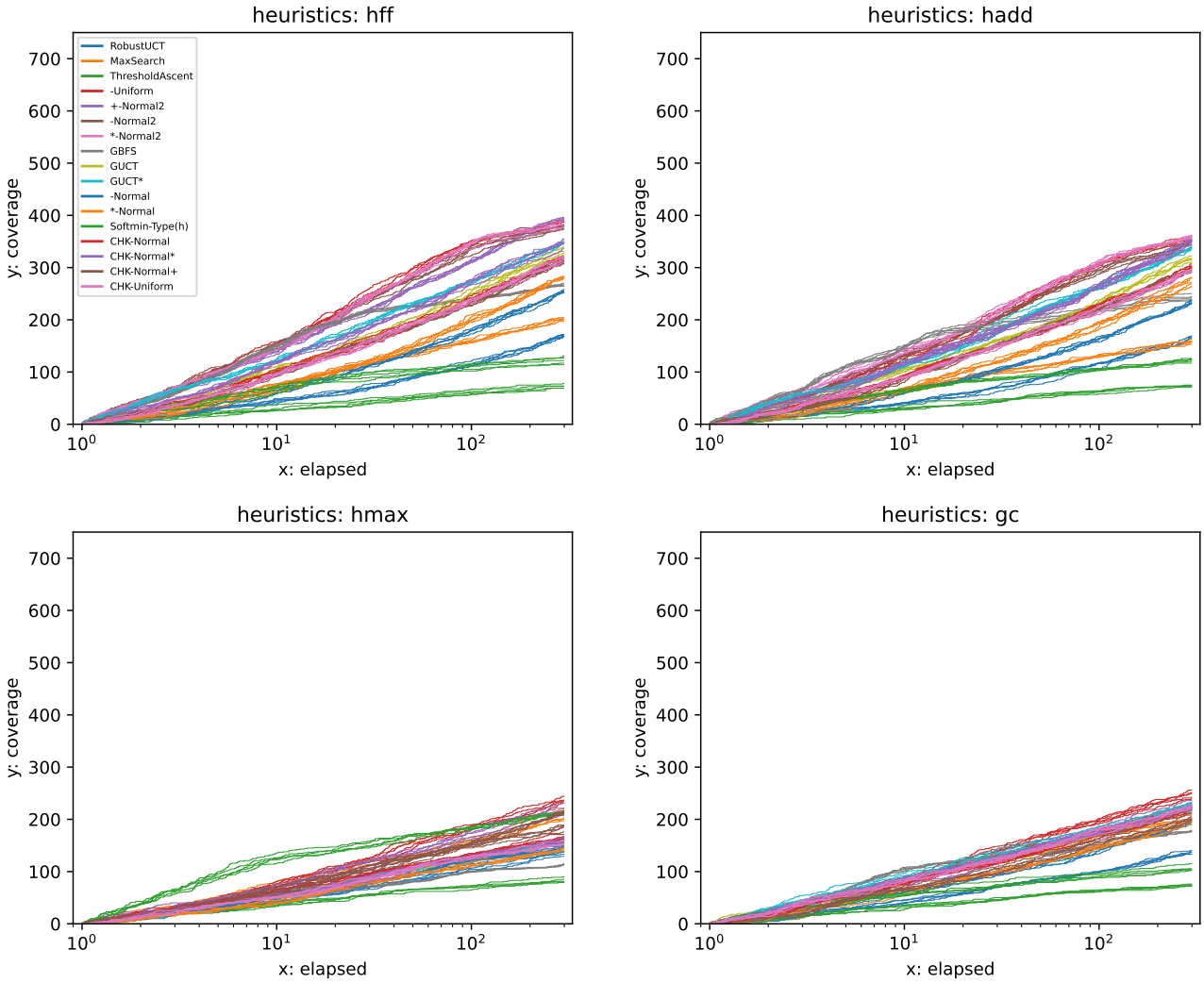

Figure A3: The cumulative histogram of the number of problem instances solved (y-axis) below a certain runtime (x-axis, 300 seconds maximum). Each line represents a random seed. The total numbers at the limit differ from those in other plots (this result does not limit the evaluations or the expansion).