Planning and Optimization

G7. Monte-Carlo Tree Search: Algorithms Part II

Gabriele Röger and Thomas Keller

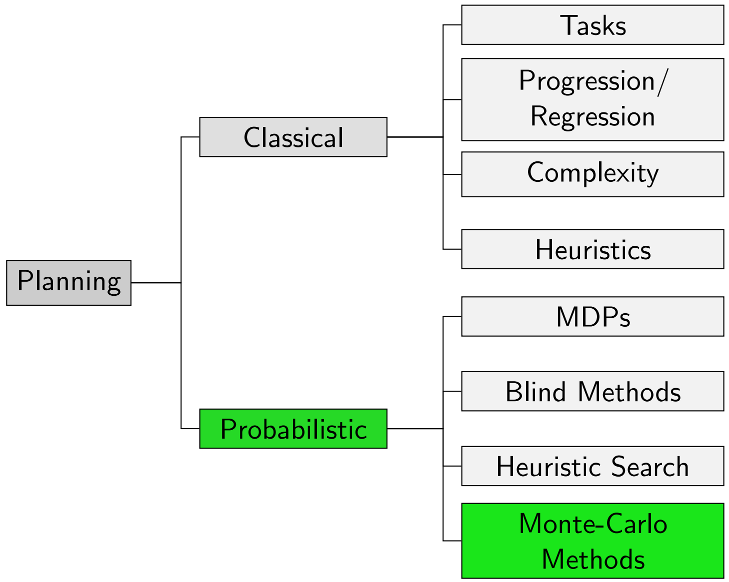
Universität Basel

December 12, 2018

Planning and Optimization

December 12, 2018 — G7. Monte-Carlo Tree Search: Algorithms Part II

Content of this Course



G7.1 Motivation

Motivation

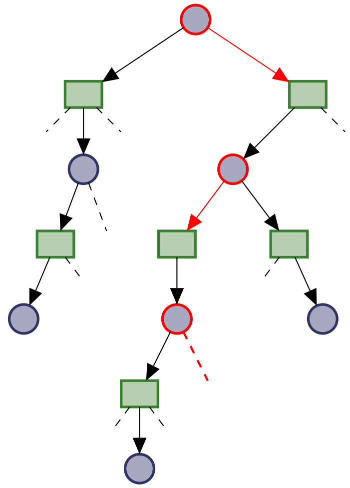
- Monte-Carlo Tree Search is a framework of algorithms
- Concrete MCTS algorithms are specified in terms of:
  - tree policy
  - default policy
- For most tasks, a well-suited MCTS configuration exists
- But: for each task, many MCTS configurations ill-suited
- And: every MCTS configuration that works well in one problem performs poorly in another problem

⇒ no dominating MCTS configuration

⇒ we present and analyze different tree and default policies

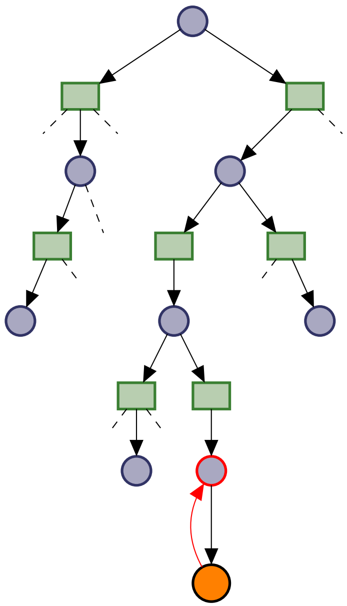
Tree Policy: Recap

- Tree policy used to traverse explicated tree, starting at root
- Assigns probability distribution over actions to each decision node
- May access information from current search tree
- Comparable to evaluation function in best-first search
- Tree policy more general: evaluation function determined upon node generation, while tree policy dynamic in each trial



Default Policy: Recap

- Default policy used to simulate run, starting at recently added decision node
- Assigns probability distribution over actions to each state
- Independent from current search tree
- Same role in MCTS as heuristic in heuristic search
- Heuristic more general: default policy is a specific kind of heuristic



G7.2 ε-greedy

ε-greedy: Idea

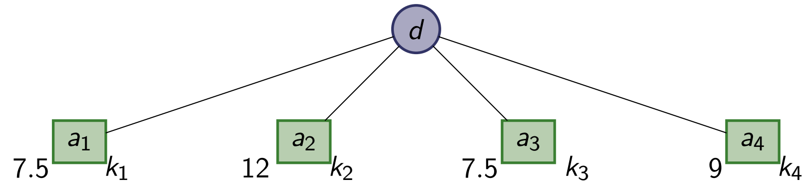
- Tree policy parametrized with constant parameter ε
- With probability $1 - \varepsilon$, pick one of the greedy actions uniformly at random
- Otherwise, pick non-greedy successor uniformly at random

ε-greedy Tree Policy

$$\pi(a \mid d) = \begin{cases} \frac{1-\epsilon}{|L_\star^k(d)|} & \text{if } a \in L_\star^k(d) \\ \frac{\epsilon}{|L(d(s)) \setminus L_\star^k(d)|} & \text{otherwise,} \end{cases}$$

with $L_\star^k(d) = \{a(c) \in L(s(d)) \mid c \in \arg\min_{c' \in \text{children}(d)} \hat{Q}^k(c')\}$.

ε-greedy: Example



Assuming $\varepsilon = 0.2$ and an SSP setting, we get:

- $\pi(a_1 \mid d) = 0.4$
- $\pi(a_2 \mid d) = 0.1$
- $\pi(a_3 \mid d) = 0.4$
- $\pi(a_4 \mid d) = 0.1$

ε-greedy: Asymptotic Optimality

Asymptotic Optimality of ε-greedy

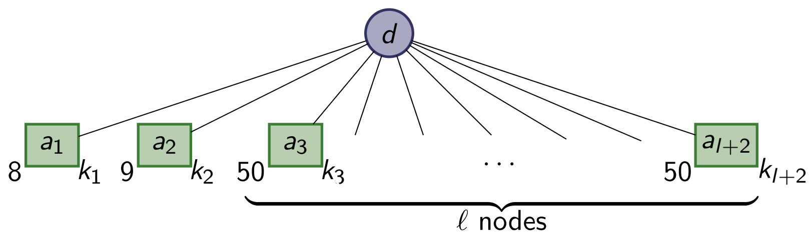
- explores forever
- not greedy in the limit

$\rightsquigarrow$ not asymptotically optimal

asymptotically optimal variant uses decaying ε, e.g. $\varepsilon = \frac{1}{k}$

ε-greedy: Weakness

Problem:
when ε-greedy explores, all non-greedy actions are treated equally



Assuming $\varepsilon = 0.2$, $\ell = 9$ and an SSP setting, we get:

- $\pi(a_1 \mid d) = 0.8$
- $\pi(a_2 \mid d) = \pi(a_3 \mid d) = \cdots = \pi(a_{11} \mid d) = 0.02$

G7.3 Softmax

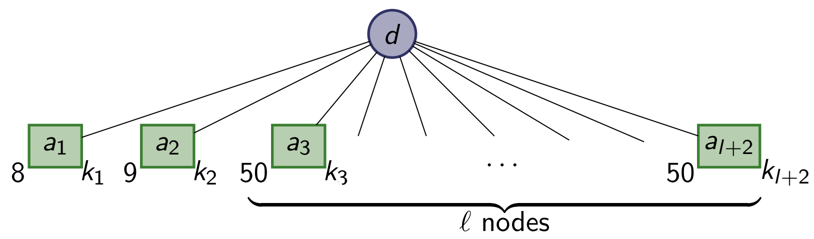
Softmax: Idea

- Tree policy with constant parameter τ
- Select actions proportionally to their action-value estimate
- Most popular softmax tree policy uses Boltzmann exploration
- $\Rightarrow$ selects actions proportionally to $e^{\frac{-\hat{Q}_k(c)}{\tau}}$

Tree Policy based on Boltzmann Exploration

$$\pi(a(c) \mid d) = \frac{e^{\frac{-\hat{Q}_k(c)}{\tau}}}{\sum_{c' \in \text{children}(d)} e^{\frac{-\hat{Q}_k(c')}{\tau}}}$$

Softmax: Example



Assuming $\varepsilon = 0.2$, $\ell = 9$, $\tau = 10$ and an SSP setting, we get:

- $\pi(a_1 \mid d) = 0.49$
- $\pi(a_2 \mid d) = 0.45$
- $\pi(a_3 \mid d) = \cdots = \pi(a_{11} \mid d) = 0.007$

Boltzmann Exploration: Asymptotic Optimality

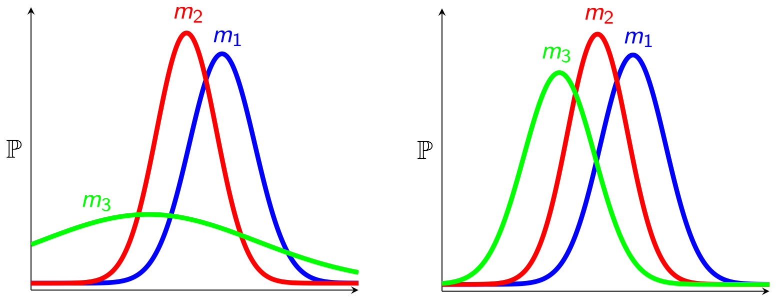
Asymptotic Optimality of Boltzmann Exploration

- explores forever
- not greedy in the limit:
  - state- and action-value estimates converge to finite value
  - therefore, probabilities also converge to positive, finite value

$\rightsquigarrow$ not asymptotically optimal

asymptotically optimal variant uses decaying τ, e.g. $\tau = \frac{1}{\log k}$
careful: τ must not decay faster than logarithmically
(i.e., must have $\tau \geq \frac{\text{const}}{\log k}$) to explore infinitely

Boltzmann Exploration: Weakness



scenario 1: high variance for m_3 scenario 2: low variance for m_3

- Boltzmann exploration (as well as ε-greedy) only considers mean of sampled action-values
- as we sample the same node many times, we can also gather information about variance (how reliable the information is)
- Boltzmann exploration ignores the variance, treating the two scenarios equally

G7.4 UCB1

Upper Confidence Bounds: Idea

Balance exploration and exploitation by preferring actions that

- have been successful in earlier iterations (exploit)
- have been selected rarely (explore)

Upper Confidence Bounds: Idea

- Select successor c of d that maximizes $\hat{Q}_k(c) + E(d) \cdot B(c)$
- based on action-value estimate $\hat{Q}(c)$,
- exploration factor $E(d)$ and
- bonus term $B(c)$.
- Select $B(c)$ such that $Q_\star(s(c), a(c)) \leq \hat{Q}^k(c) + E(d) \cdot B(c)$ with high probability
- Idea: $\hat{Q}^k(c) + E(d) \cdot B(c)$ is an upper confidence bound on $Q_\star(s(c), a(c))$ under the collected information

Careful: MDP setting considered here,
replace $\hat{Q}_k(c)$ with $-\hat{Q}_k(c)$ for SSPs

Bonus Term of UCB1

- Use $B(c) = \sqrt{\frac{2 \cdot \ln N_k(d)}{N_k(c)}}$ as bonus term
- Bonus term is derived from Chernoff-Hoeffding bound:
  - gives the probability that a sampled value (here: $\hat{Q}^k(c)$)
  - is far from its true expected value (here: $Q_\star(s(c), a(c))$)
  - in dependence of the number of samples (here: $N^k(c)$)
- Picks the optimal action exponentially more often
- Concrete MCTS algorithm that uses UCB1 is called UCT

Exploration Factor

- Exploration factor serves two roles
- UCB1 designed for MAB with reward in $[0, 1]$
  $\Rightarrow \hat{Q}_k(c) \in [0; 1]$ for all k and c
- Bonus term always ≥ 0 and most of the time ≤ 1
- First role: to make sure $\hat{Q}_k(c)$ and $B(c)$ are of comparable size, set $E(d) := \hat{V}_k(d)$ (dynamically for each decision)
- Second role: $E(d)$ allows to adjust balance between exploration and exploitation
- Search with $E(d) = \hat{V}_k(d)$ very greedy
- In practice, $E(d)$ is often multiplied with constant > 1
- UCB1 often requires hand-tailored $E(d)$ to work well

Asymptotic Optimality

Asymptotic Optimality of UCB1

- explores forever
- greedy in the limit

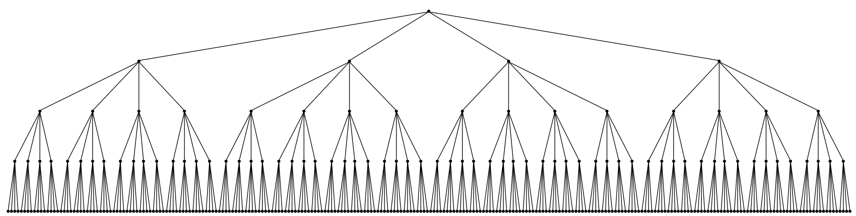
$\rightsquigarrow$ asymptotically optimal

However:

- No theoretical justification to use UCB1 in MDPs
  (MAB proof requires stationary rewards)
- Development of tree policies active research topic

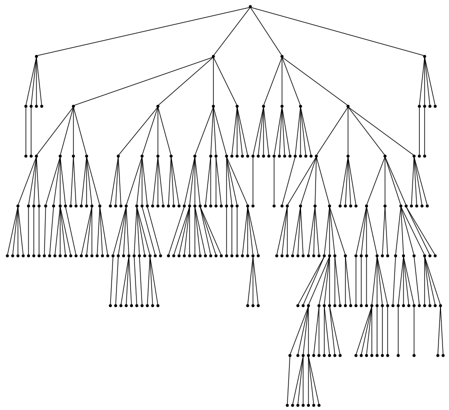
Symmetric Search Tree up to depth 4

full tree up to depth 4



Asymmetric Search Tree of UCB1

(equal number of search nodes)



G7.5 Summary

Summary

- ε-greedy, Boltzmann exploration and UCB1 balance exploration and exploitation with different methods
- ε-greedy selects greedy action with probability $1 - \varepsilon$ and another action uniformly at random otherwise
- ε-greedy selects non-greedy actions with same probability
- Boltzmann exploration selects each action proportional to its action-value estimate
- Boltzmann exploration does not take confidence of estimate into account
- UCB1 selects actions greedily w.r.t. upper confidence bound on action-value estimate