AI Planning
Lecture 11

Rostislav Horčík

Czech Technical University in Prague
Faculty of Electrical Engineering
xhorcik@fel.cvut.cz

Sampling methods

Why sampling?

- Limited computational resources
  - admissible heuristic defines a greedy policy that can be improved if there is time/resources.
  - Real-Time Dynamic Programming (RTDP)

- Too large branching factor
  - if action applications can lead to large number of successor states, Bellman's update is infeasible.
  - Monte Carlo methods (UCT)

Idea of RTDP

- RTDP maintains a greedy policy π^V of value function V initialized by an admissible heuristic h.
- If it has time, it executes the current policy π^V by sampling its execution, i.e., a path from the initial state to a goal state.
- Next, it performs Bellman's update for the states on this path.

Recall $Q(s,a) = \text{cost}(a) + \sum_{t \in T(s,a)} P(t \mid s,a)V(t)$

Require: SSP $\boldsymbol{\Sigma} = \langle S, A, T, s_0, G \rangle$, admissible heuristic h, time limit T_{max}

$V \leftarrow h$, $t \leftarrow 0$

while $t < T_{\text{max}}$ **do**

  $t \leftarrow t + 1$

  $s \leftarrow s_0$

  while $s \notin G$ and max. number of iterations not reached **do**

    $a \leftarrow \text{argmin}_{a' \in A(s)} Q(s, a')$

    $V(s) \leftarrow Q(s, a)$

    $s \leftarrow$ sample a successor of applying a in s

return π^V

Sysadmin domain

Consider n independent servers

$V = \{v_1, \ldots, v_n\}$, $\mathrm{dom}(v_i) = \{DOWN, UP\}$, $|S| = 2^n$

NOOP action, each server i

$$0.5 : v_i = UP \rightarrow v_i = UP$$
$$0.5 : v_i = UP \rightarrow v_i = DOWN$$
$$0.9 : v_i = DOWN \rightarrow v_i = DOWN$$
$$0.1 : v_i = DOWN \rightarrow v_i = UP$$

To represent NOOP in PFDR requires 2^n actions with 2^n probabilistic effects.

RDDL represents NOOP as a single action with $P(t \mid s, a) = \prod_{v \in V} P_v(t(v) \mid s, a)$.

Monte Carlo Methods

Idea of Monte Carlo methods

To make a simulated walk (rollout) from a given state s, it suffices if we can efficiently sample succesor states (can be done with RDDL).

Q-values $Q(s,a)$ can be approximated as average cost from particular rollouts.

$$r_j\colon s \xrightarrow{a} s_1 \xrightarrow{a_1} s_2 \xrightarrow{a_2} \cdots \xrightarrow{a_{d_{\max}-1}} s_{d_{\max}}$$

$$\hat{Q}_j(s,a) = \mathrm{cost}(a) + h(s_{d_{\max}}) + \sum_{i=1}^{d_{\max}-1} \mathrm{cost}(a_i)$$

$$\hat{Q}(s,a) = \frac{1}{N}\sum_{j=1}^{N} \hat{Q}_j(s,a)$$

Sampling strategy

Which actions to choose when simulating the rollouts?

1. We can follow the current greedy strategy (Exploitation)
2. We can try to explore actions that were rarely applied in previous rollouts (Exploration)

UCT applies a sampling strategy to balance exploration/exploitation tradeoff

$$\underset{a \in A(s)}{\operatorname{argmin}}\, Q(s, a) - C \cdot \sqrt{\frac{\ln(n(s))}{n(s, a)}}$$

- $n(s)$ of how often state s was visited
- $n(s, a)$ how often action a was applied in s
- $C \geq 0$ constant

Require: Simulator of SSP Σ, admiss. heur. h, max. depth d_{max}

$V \leftarrow h$, $Q(s', a) \leftarrow \mathrm{cost}(a)$ for $s' \in S \setminus G$, $a \in A(s')$
$n(s) \leftarrow 0$, $n(s, a) \leftarrow 0$
$\mathrm{UCT}(s_0, d_{\mathsf{max}})$
procedure $\mathrm{UCT}(s, d)$
  if $s \in G$ **then return** 0
  if $d = 0$ **then return** $V(s)$
  $\mathit{Untried} \leftarrow \{a \in A(s) \mid n(s, a) = 0\}$
  if $\mathit{Untried} \neq \emptyset$ **then**
    $a \leftarrow \mathrm{sample}(\mathit{Untried})$
  else
    $a \leftarrow \mathbf{argmin}_{a \in A(s)} Q(s, a) - C \cdot \sqrt{\frac{\ln(n(s))}{n(s,a)}}$
  $s' \leftarrow \mathrm{sample}(s, a)$
  $\mathit{cost} \leftarrow \mathrm{cost}(a) + \mathrm{UCT}(s', d - 1)$
  $Q(s, a) \leftarrow \frac{n(s,a)Q(s,a)+\mathit{cost}}{n(s,a)+1}$
  $n(s) \leftarrow n(s) + 1$, $n(s, a) \leftarrow n(s, a) + 1$
  return cost