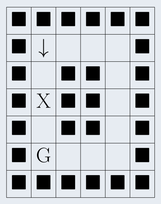
Reinforcement Learning of Risk-Constrained Policies in Markov Decision Processes

Tomáš Brázdil[1], Krishnendu Chatterjee[2], Petr Novotný[1], Jiří Vahala[1]

[1]Faculty of Informatics, Masaryk University, Brno, Czech Republic
[2] Institute of Science and Technology Austria, Klosterneuburg, Austria

Motivation



- step penalty: 5
- probability of destruction on trap(X): 0.2
- gold(G) reward: 100
- $\mathbb{E}$[long path] = 35
- $\mathbb{E}$[short path] = 62
- Do we know optimal solution for the agent?

- Expected cumulative reward might not be the ideal measure.
- Explicitly control the rate of destruction - use **risk parameter τ**.
- **Problem: Find a policy to maximize the cumulative reward given under the constraint that the probability of reaching a failure state is below the risk threshold τ.**
- **MDP: $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \delta, rew, s_0, \gamma)$** where $\mathcal{S}$ is a set of *states*, $\mathcal{A}$ is a set of *actions*, $\delta : \mathcal{S} \times \mathcal{A} \to \mathcal{D}(\mathcal{S})$ is a *probabilistic transition function*, $rew : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is a *reward function*, s_0 is the *initial state*, and $\gamma \in (0, 1]$ is the *discount factor*

Reinforcement learning

- Reinforcement learning(RL) is one of the basic machine learning paradigms.
- MDP decision loop:

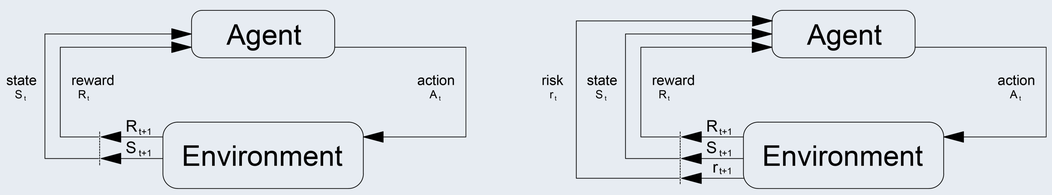
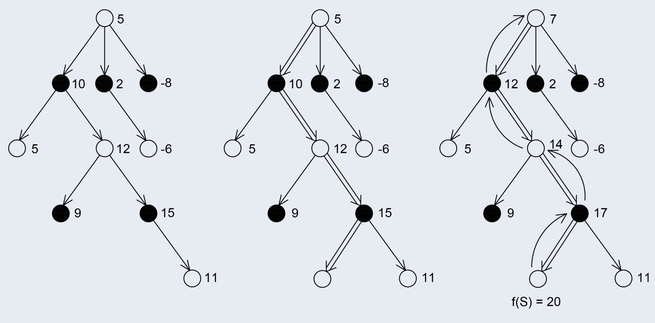


Figure: A typical agent-environment interaction scheme in MDP on left and modified version enriched by risk information on right

Tree search

- For every decision made by the agent, a lookahead tree $\mathcal{T}$ of near-future outcomes is constructed in an iterative way. Each iteration consists of three phases:



- **Select** the next leaf for expansion.
- **Expand** the search tree by adding all children of the selected node.
- **Evaluate** all freshly added nodes with an evaluator:

$$f : \mathcal{S} \to \mathbb{R} \times [0,1] \times [0,1]^{|\mathcal{A}|}$$

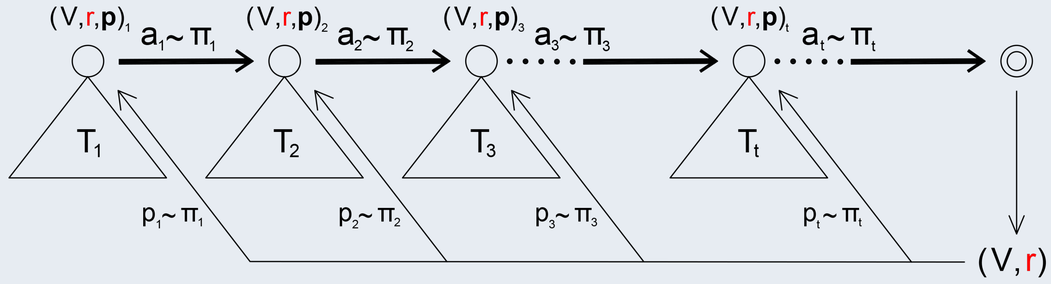
- Leaf selection proceeds by downward traversal using a variant of the UCT formula:

$$\texttt{UCT}(s,a) = \frac{s.V_a - s.V_{\min}}{s.V_{\max} - s.V_{\min}} + C \cdot s.p_a \cdot \sqrt{\frac{\ln(s.N)}{s.N_a + 1}}$$

- The tree $\mathcal{T}$ is built and used as a decision making process for every agent decision in the environment.

RAlph algorithm

- **Combines lookahead tree with linear optimization.**
- Uses predictor f predicting tuple $(V, r, \boldsymbol{p})$ corresponding to **expected discounted cumulative reward**, **risk** and **prior transition probability vector**.
- **The predictor f is episodically trained.**



- **In every step, linear optimization of probability flow is performed on top of the lookahead tree $\mathcal{T}$ to balance expected cumulative reward with given risk parameter τ_t.**

$$\begin{aligned}
&max \sum_{h \in leaf(\mathcal{T})} x_h \cdot (Payoff(h) + \gamma^{len(h)} \cdot h.v) \ subject\ to \\
&x_{root(\mathcal{T})} = 1 \\
&x_h = \sum_{a \in \mathcal{A}} x_{h,a} \text{for } h \in \mathcal{T} \setminus leaf(\mathcal{T}) \\
&x_{hbt} = x_{h,b} \cdot \delta(t \mid last(h), b) \quad \text{for } h, hbt \in \mathcal{T} \\
&0 \le x_h \le 1, \quad 0 \le x_{h,a} \le 1 \qquad \text{for } h \in \mathcal{T}, a \in \mathcal{A} \\
&\sum_{h \in leaf(\mathcal{T})} x_h \cdot h.r \le \Delta
\end{aligned}$$

Properties

- RAlph is **locally risk-driven** → it might purposely avoid close failure states.
- RAlph can **effectively traverse a state-space** and prune a significant amount of paths leading to states with higher risk for most of the training. Hence, the (sub)optimal path to the goal is found in smaller number of training episodes.

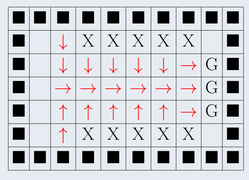
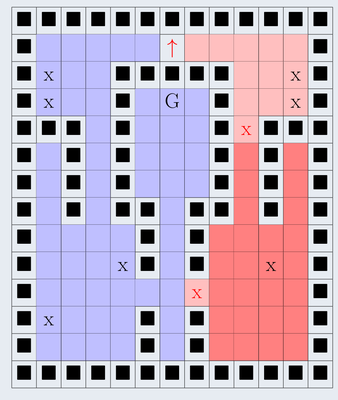


Table: Locally risk-driven agent actively returns back to the safe middle path to avoid possible random shifts into trap



Take-Home Message

- New reinforcement learning algorithm RAlph for finding risk-constrained policies maximizing expected cumulative reward in MDPs.
- Ralph builds a lookahead tree in every step and then selects next action based on linear optimization of probability flow inside the tree.
- The lookahead tree is built using combination of modified UCT with a predictor as a node evaluator.
- Locally risk-driven policies can avoid dangerous states and hence sample efficient episodes.
- Risk-averse policies traverse search-space more efficiently and can find optimal path in fewer number of episodes.

Contact: {xbrazdil, petr.novotny, xvahala1}@fi.muni.cz
Krishnendu.Chatterjee@ist.ac.at