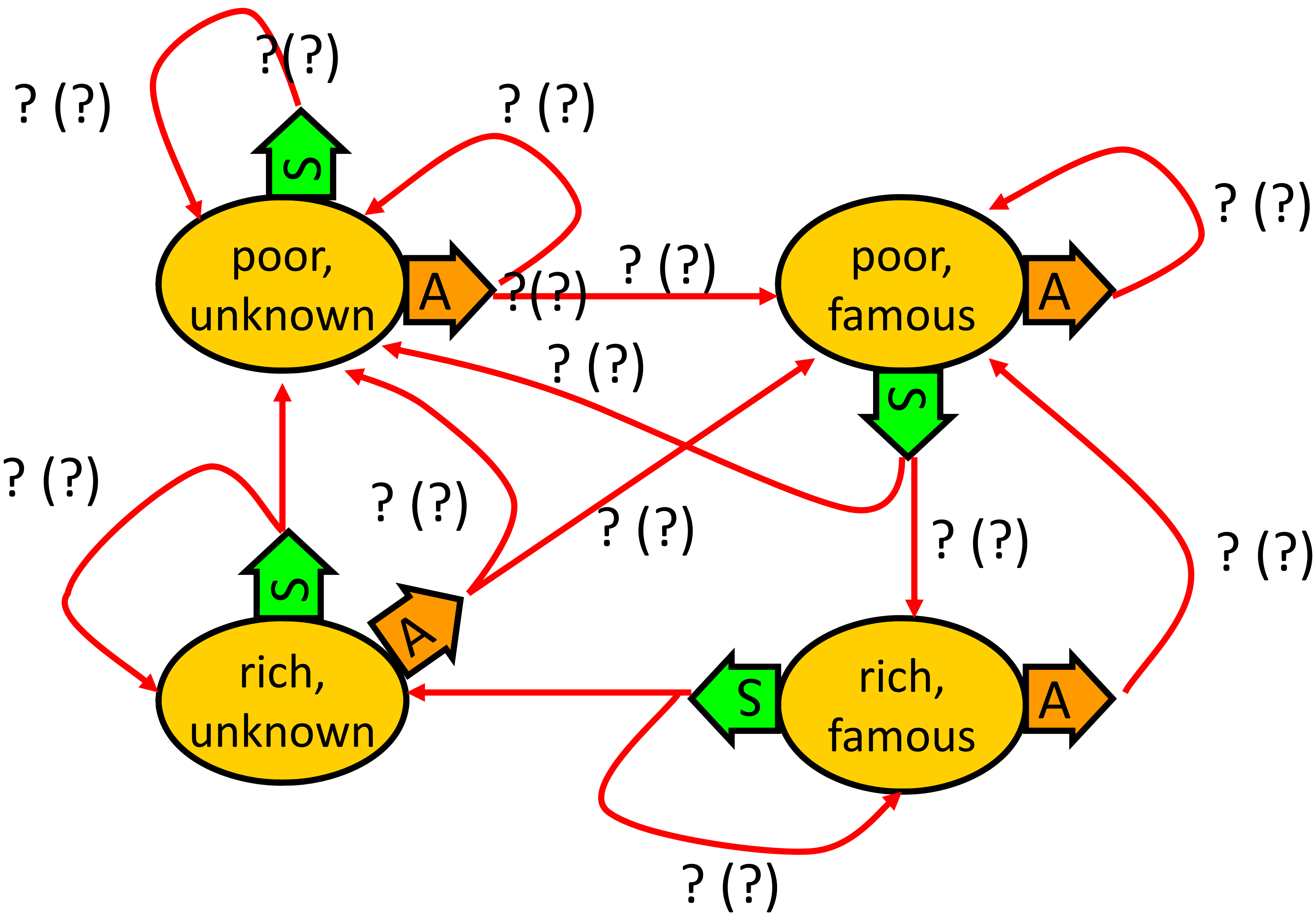
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Probabilistic Foundations of Artificial Intelligence

Reinforcement Learning

Prof. Andreas Krause
Learning and Adaptive Systems (las.ethz.ch)

RL = Planning in unknown MDPs



Reinforcement Learning

- RL is different from supervised learning
  - The data we get is not i.i.d.
  - In reinforcement learning, the data we get *depends on our actions*!
  - Some actions have higher rewards than others!

- *Exploration—Exploitation Dilemma*: Should we
  - Explore**:** gather more data to avoid missing out on a potentially large reward?
  - Exploit**:** stick with our current knowledge and build an optimal policy for the data we' ve seen?

Possible approaches

- Always pick a random action?
  - Will eventually correctly estimate all probabilities and rewards ☺
  - May do extremely poorly! ☹

- Always pick the best action according to current knowledge?
  - Quickly get some reward
  - Can get stuck in suboptimal action! ☹

- Balance exploration and exploitation (more later)

Two basic approaches

1) Model-based RL ("Approximate dynamic programming")

- Learn the MDP

  Estimate transition probabilities $P(x' \mid x, a)$

  Estimate reward function $r(x, a)$

- Optimize policy based on estimated MDP

2) Model-free RL

- Estimate the value function directly;
- Actor-critic methods;
- Policy gradient methods

Learning the MDP

- Need to estimate
  - transition probabilities $P(X_{t+1} \mid X_t, A)$
  - Reward function $r(X, A)$
- Can use techniques from learning Bayes Nets: (regularized) maximum likelihood estimation
- Data set: "Experiences" $x_1, a_1, r_1, x_2, a_2, r_2, x_3, \ldots$

  $D = \{(x_1, a_1, r_1, x_2), (x_2, a_2, r_2, x_3), \ldots (x_{t-1}, a_{t-1}, r_{t-1}, x_t)\}$
- Estimate transitions:

$$P(X_{t+1} \mid X_t, A) \approx \frac{Count(X_{t+1}, X_t, A)}{Count(X_t, A)}$$

- Estimate rewards:

$$r(x,a) \approx \frac{1}{N_{x,a}} \sum_{t: X_t = x, A_t = a} R_t$$

Encouraging Exploration

- ε_t greedy
  - With probability ε_t: Pick random action
  - With probability $(1-\varepsilon_t)$: Pick best action
- If sequence ε_t satisfies Robbins Monro (RM) conditions then will converge to optimal policy with probability 1

$\varepsilon_t \to 0$, in particular: $\sum_{t=1}^{\infty} \varepsilon_t \to \infty$

E.g. $\varepsilon_t = \frac{1}{t}$ $\qquad \sum_{t=1}^{\infty} \varepsilon_t^2$ finite

$\varepsilon_t = \min\left(1, \frac{c}{t}\right)$

- Often performs quite well
- Doesn't quickly eliminate clearly suboptimal actions

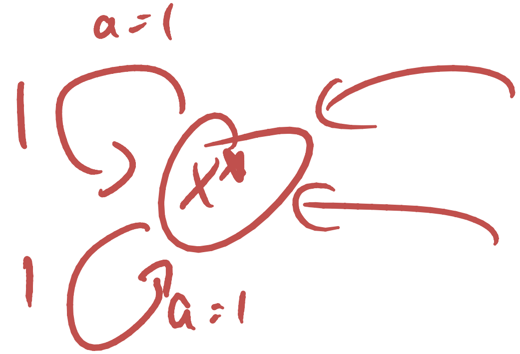
The R_{max} Algorithm [Brafman & Tennenholz '02]

Optimism in the face of uncertainty!

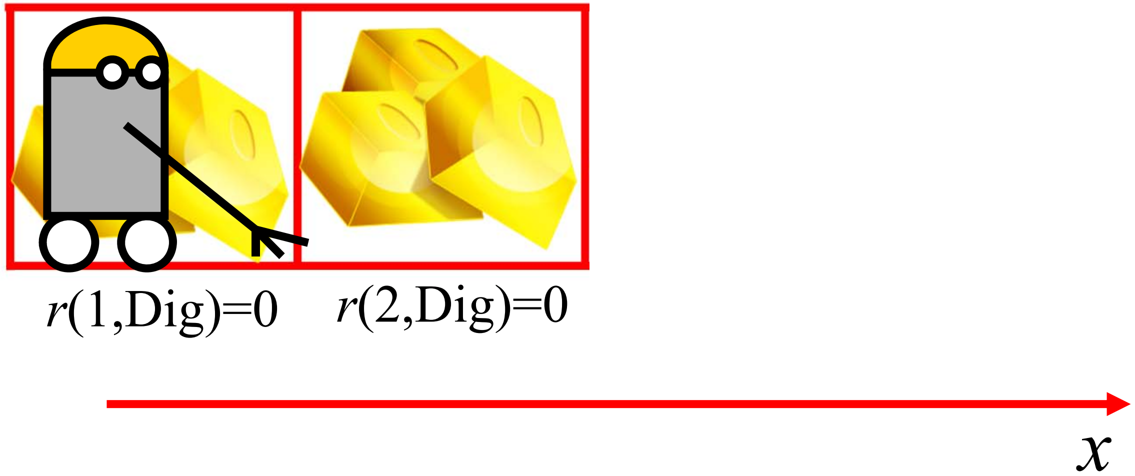
- If you don't know $r(x, a)$:
  - Set it to $R_{\max}$!
- If you don't know $P(x' \mid x, a)$:
  - Set $P(x^* \mid x, a) = 1$ where x^* is a "**fairy tale**" state:

$P(x^* \mid x^*, a) = 1 \quad \forall a$

$r(x^*, a) = R_{max}$



Implicit Exploration Exploitation in R_{max}



Three actions:

- Left
- Right
- Dig

$$r(i, \text{Left}) = 0$$
$$r(i, \text{Right}) = 0$$

Never need to explicitly choose whether we're exploring or exploiting!

Can rule out clearly suboptimal actions very quickly

Exploration—Exploitation Lemma

Theorem: Every T timesteps, w.h.p., $R_{\max}$ either

- Obtains near-optimal reward, or
- Visits at least one unknown state-action pair

- T is related to the mixing time of the Markov chain of the MDP induced by the optimal policy

The R_{max} algorithm

Input: Starting state x_0, discount factor γ

Initially:

- Add fairy tale state x^* to MDP
- Set $r(x, a) = R_{max}$ for all states x and actions a
- Set $P(x^* \mid x, a) = 1$ for all states x and actions a
- Choose optimal policy for r and P

Repeat:

- Execute policy π
- For each visited state action pair x, a, update $r(x, a)$
- Estimate transition probabilities $P(x' \mid x, a)$
- If observed "enough" transitions / rewards, recompute policy π according to current model P and r

How much is "enough"?

How many samples do we need to accurately estimate $P(x' \mid x, a)$ or $r(x,a)$?

Hoeffding-Chernoff bound:

- $X_1, \ldots, X_n$ i.i.d. samples from Bernoulli distribution w. mean μ

$$P\left(\left|\mu - \frac{1}{n}\sum_i X_i\right| \geq \varepsilon\right) \leq 2\exp(-2n\varepsilon^2)$$

Performance of R_{max} [Brafman & Tennenholz]

Theorem:

With probability $1\text{-}\delta$, $R_{\max}$ will reach an ε-optimal policy in a number of steps that is polynomial in $|X|$, $|A|$, T, $1/\varepsilon$ and $\log(1/\delta)$

Problems of model-based RL?

- Memory required:

  "Dense" MDPs: $O(|X|^2 \cdot |A|)$ to store frequencies / rewards
  (in general $O(\#$ edges in MDP$)$)

- Computation time:

  need to replan "often"
  each replanning step is poly$(|X|, |A|)$

Two basic approaches

1) Model-based RL ("Approximate dynamic programming")

- Learn the MDP

  Estimate transition probabilities $P(x' \mid x, a)$

  Estimate reward function $r(x, a)$

- Optimize policy based on estimated MDP

2) Model-free RL

- Estimate the value function directly;
- Actor-critic methods;
- Policy gradient methods

Model free RL

- Recall:
  1. Optimal value function $V^*(x)$ ➔ opt. policy π^*
  2. For optimal value function it holds:

$$\boxed{\begin{aligned} &V^*(x) = \max_a Q^*(x,a) \\ &\text{where } Q^*(x,a) = r(x,a) + \gamma \sum_{x'} P(x' \mid x,a) V^*(x') \end{aligned}}$$

Key idea: Estimate $Q^*(x, a)$ directly from samples!

Q-learning

- Estimate $$Q^*(x,a) = r(x,a) + \underbrace{\gamma \sum_{x'} P(x' \mid x,a) V^*(x')}_{\text{Expected value of next state}}$$

$$V^*(x) = \max_a Q^*(x,a)$$

↳ approx. by
- old est. of V^*
- single sample $x' \sim P(x' \mid x,a)$

- Suppose we
  - Have initial estimate of $Q(x, a)$
  - observe transition x, a, x' with reward r

$$Q(x,a) \leftarrow (1-\alpha_t)\underbrace{\overset{\text{old}}{Q}(x,a)}_{\text{old estimate for } Q} + \alpha_t \underbrace{\left(r + \gamma \underbrace{\max_{a'} \overset{\text{old}}{Q}(x',a')}_{\text{old estimate for } V(x')}\right)}_{\text{"new estimate for } Q \text{ given single exp. only}}$$

Q-learning

$$Q(x,a) \leftarrow (1-\alpha_t)Q(x,a) + \alpha_t\Big(r + \gamma \max_{a'} Q(x',a')\Big)$$

Theorem: If learning rate α_t satisfies

$\sum_t \alpha_t = \infty$

$\sum_t \alpha_t^2 < \infty$

and actions are chosen at random*, then Q learning converges to optimal Q^* with probability 1

How can we trade off exploration and exploitation?

Convergence of Optimistic Q-learning

[Even-dar & Mansour '02]

Similar to $R_{\max}$:

Initialize $Q(x,a) = \frac{R_{\max}}{1-\gamma} \prod_{t=1}^{T_{\text{init}}} (1-\alpha_t)^{-1}$

Theorem: With prob. 1-δ, optimistic Q-learning obtains an ε-optimal policy after a number of time steps that is polynomial in $|X|$, $|A|$, $1/\varepsilon$ and $\log(1/\delta)$

Properties of Q-learning

- Memory required: $O(|X| \cdot |A|)$ to store Q fn
- Computation time: $O(|A|)$

Acknowledgments

- Slides based on material accompanying the textbook "AI: A Modern Approach" (3rd edition) by S. Russell and P. Norvig, the textbook "Reinforcement Learning: An Introduction" by R. S. Sutton and A. G. Barto