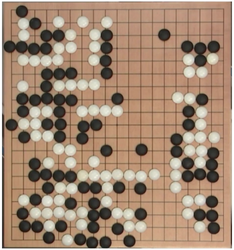
Markov Decision Process and Reinforcement Learning

Zeqian (Chris) Li

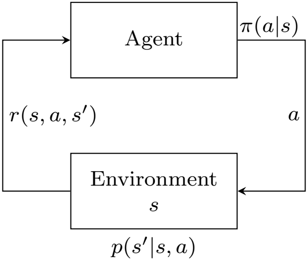
Feb 28, 2019



Outline

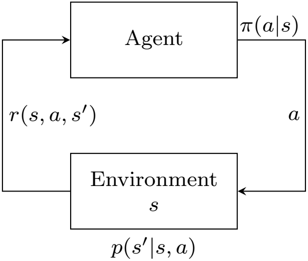
Introduction

- Hungry rat experiment, Yale, 1948
- Modeling reinforcement: **agend-based model**



- s: state; a: action; r: reward
- $p(s'|s, a)$: transitional probability; $r(s, a, s')$: reward model; $\pi(a|s)$: policy
- This is a dynamical process: $s_t, a_t, r_t; s_{t+1}, a_{t+1}, r_{t+1}; ...$

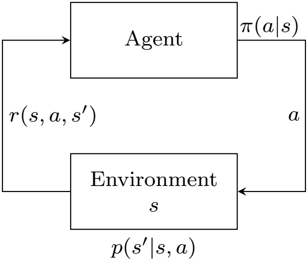
Examples: Atari games

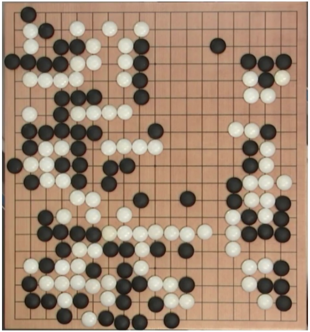


Atari games

- State: brick positions, board positions, ball coordinate and velocity
- Action: controller/keyboard inputs
- Reward: game score

Examples: Go

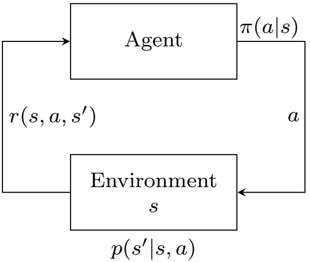




Go

- State: positions of stones
- Action: next move
- Reward: advantage evaluation

Examples: robots



(Boston Dynamics)

Robots

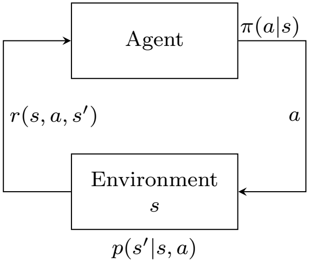
- State: positions, mass distribution, ...
- Action: adjusting forces on feet
- Reward: chance of falling

Other examples

- Example in physics?

Objective of reinforcement learning

- s_t, a_t
- $p(s'|s,a)$: transitional probability
  $r(s,a,s')$: reward model
  $\pi(a|s)$: policy



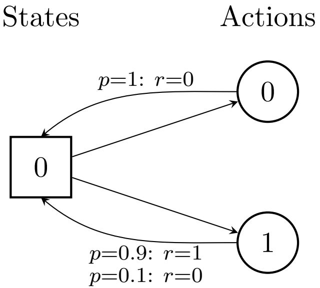
Objective of reinforcement learning

Find optimal policy $\pi^*(a|y)$ to maximize expected reward:

$$\pi^*(a|s) = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}[V] = \underset{\pi}{\operatorname{argmax}}\, \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(t)\right]$$

(γ: $0 \le \gamma < 1$, discount factor)

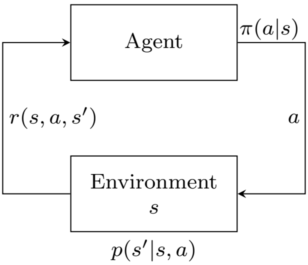
Simplest example: one-armed bandits



- Optimal policy:

$$\pi^*(0|0) = 0,\ \pi^*(1|0) = 1$$

Markov decision process



- Suppose that I have full knowledge of $p(s'|a, s), r(s, a, s')$.
- This is called **Markov Decision Process**.
- Objective of MDP: compute

$$\pi^*(a|s) = \operatorname*{argmax}_{\pi} \mathbb{E}[V] = \operatorname*{argmax}_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(t)\right]$$

- This is a **computing** problem. No learning.

Quality function $Q(s, a)$

- $\pi^*(a|s) = \text{argmax}_\pi \, \mathbb{E}[V] = \text{argmax}_\pi \, \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(t)\right]$
- Define
$$Q(s, a) = \mathbb{E}_{\pi^*}\left[\sum_{t=0}^{\infty} \gamma^t r(t) \middle| s_0 = s, a_0 = a\right]$$
Given the initial state s and the initial action a, Q is the maximum expected future reward.
- Recursive relationship:
$$\begin{aligned} Q(sa) &= \sum_{s'} p(s'|as)\left[r(sas') + \gamma \max_{a'} Q(s'a')\right] \\ &= \mathbb{E}_{s'}\left[r(sas') + \gamma \max_{a'} Q(s'a') \middle| sa\right] \end{aligned}$$

Bellman equation

$$Q(sa) = \mathbb{E}_{s'}\left[r(sas') + \gamma \max_{a'} Q(s'a') \middle| sa \right]$$

- Solve $Q(sa)$ (or $\psi(s)$) by Bellman equation, and the optimal policy is given by (when $\epsilon \to 0$):

$$\pi^*(a|s) = \begin{cases} 1 & , \ a^*(s) = \text{argmax}_a Q(a, s) \\ 0 & , \ \text{otherwise.} \end{cases}$$

- "Curse of dimensionality"

- Solve Bellman equation: iterative method

$$\begin{aligned} Q_{i+1}(sa) &= \mathbb{E}_{s'}\left[r(sas') + \gamma \max_{a'} Q_i(s'a') \middle| sa \right] \\ &= B[Q_i] \end{aligned}$$

- Start with Q_0, and update by $Q_{i+1} = B[Q_i]$.
- Can prove the convergence by calculating the Jacobian of B near the fixed point.

Problem: only update one entry (one (s, a) pair) at each iteration; converges too slow.

Statistical mechanics of MDP

- s_t, a_t; $p(s'|s,a), r(s,a,s'), \pi(a|s)$
- Find $\pi^*(a|s) = \text{argmax}_\pi \mathbb{E}[V] = \text{argmax}_\pi \mathbb{E}\big[\sum_{t=0}^{\infty} \gamma^t r(t)\big]$

- Define $\rho_t(s)$: probability in state s at time t
- Chapman–Kolmogorov equation:

$$\rho_{t+1}(s') = \sum_{s,a} p(s'|sa)\pi(a|s)\rho_t(s)$$

$$V_\pi = \mathbb{E}_{\pi,\rho}[R] = \sum_{t=0}^{\infty} \gamma^t \sum_{sas'} \rho_t(s)\pi(a|s)p(s'|sa)r(sas')$$

$$\text{(Let } \eta(s) \equiv \sum_{t=0}^{\infty} \gamma^t \rho_t(s) \text{, average residence time in } s \text{ before death)}$$

$$= \sum_{s'as} \eta(s)\pi(a|s)p(s'|sa)r(s'as)$$

- **Constraints**:
  - $\eta(s)$ depends on π:

$$\eta(s') = \rho_0(s') + \gamma \sum_{sa} p(s'|sa)\pi(a|s)\eta(s)$$

  - $\sum_a \pi(a|s) = 1$
  - introduce Lagrange multipliers

$$
\begin{aligned}
F_{\pi,\eta} =& V_{\pi,\eta} - \sum_{s'} \phi(s') \left[\eta(s') - \rho_0(s') - \gamma \sum_{sa} p(s'|sa)\pi(a|s)\eta(s) \right] \\
& - \sum_{s} \lambda(s) \left[\sum_{a} \pi(a|s) - 1 \right]
\end{aligned}
$$

- Optimization: $\frac{\delta F}{\delta \pi(a|s)} = 0, \frac{\delta F}{\delta \eta(s)} = 0.$
- **Problem**: linear function $\rightarrow$ derivative is constant $\rightarrow$ extreme value on the boundary $\rightarrow$ Optimal policy is deterministic (0 or 1)

- Introduce non-linearity: entropy

$$
H_s[\pi] = -\sum_a \pi(a|s) \log \pi(a|s)
$$

(Similar to regularization.)

$$
\begin{aligned}
F_{\pi,\eta} = &\sum_{s'as} \eta(s)\pi(a|s)p(s'|sa)r(s'as) \qquad (V_{\pi,\eta}) \\
&- \sum_{s'} \phi(s') \left[\eta(s') - \rho_0(s') - \gamma \sum_{sa} p(s'|sa)\pi(a|s)\eta(s) \right] \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad \text{(dynamical constraint)} \\
&- \sum_{s} \lambda(s) \left[\sum_{a} \pi(a|s) - 1 \right] \qquad \text{(normalization)} \\
&+ \epsilon \sum_{s} \eta(s) H_s[\pi] \qquad \text{(entropy)}
\end{aligned}
$$

- $\frac{\delta F}{\delta \pi(a|s)} = 0$, $\frac{\delta F}{\delta \eta(s)} = 0$.

Results

- $\pi^*(a|s) = \frac{\exp(Q(s,a)/\epsilon)}{\sum_b \exp(Q(s,b)/\epsilon)}$ - Boltzmann distribution!
- ϵ: temperature!
- Q: quality function - (minus) energy!

$$\begin{aligned} Q(sa) &= \sum_{s'} p(s'|sa)\left[r(sas') + \gamma\epsilon \log\left(\sum_{a'} \exp \frac{Q(s'a')}{\epsilon}\right)\right] \\ &= \mathbb{E}_{s'}\left[r(sas') + \gamma \operatorname*{softmax}_{a';\epsilon} Q(s'a')\right] \\ (\epsilon \to 0) \qquad &= \mathbb{E}_{s'}\left[r(sas') + \gamma \max_{a'} Q(s'a')\right] \end{aligned}$$

- Can show that

$$Q(sa) = \mathbb{E}_{\pi^*}\left[\sum_t \gamma^t r(t) \middle| s_0 = s, a_0 = a\right]$$

- $\phi(x)$: value function - (minus) free energy!

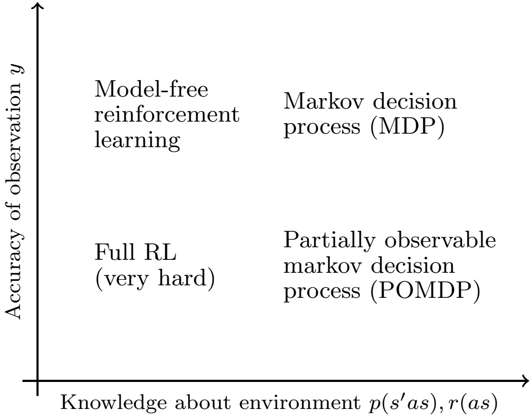
$$\phi(s) = \epsilon \log \left[\sum_a \exp\left(\frac{1}{\epsilon} Q(as) \right) \right]$$
$$= \operatorname*{softmax}_{a;\epsilon} Q(as)$$
$$(\epsilon \to 0) \quad = \max_a Q(as)$$

- Iterative equation:

$$\phi(s) = \operatorname*{softmax}_{a;\epsilon} \left\{ \mathbb{E}_{s'} \left[r(sas') + \gamma \phi(s') \right] \right\}$$
$$(\epsilon \to 0) \quad = \max_a \left\{ \mathbb{E}_{s'} \left[r(sas') + \gamma \phi(s') \right] \right\}$$

Physical meaning of $\phi(s)$: maximum expected future reward, given initial state s.

Spectrum of reinforcement learning problems



MDP Bellman equation ($\epsilon > 0$)

$$Q(s,a) = \mathbb{E}_{s'}\left[r(sas') + \gamma \operatorname*{softmax}_{a';\epsilon} Q(s'a') \middle| sa\right]$$

Reinforcement learning: don't know $r(s,a,s'), p(s'|s,a)$, only have samples of $(s_0, a_0, s_1; r_0), (s_1, a_1, s_2; r_2), ..., (s_t, a_t, s_{t+1}; r_t), ...$

Rewrite Bellman equation:

$$\mathbb{E}_{\text{samples of } (\cdot|sa)}\left(r(s,a,\cdot) + \gamma \operatorname*{softmax}_{a';\epsilon} Q(\cdot, a') - Q(s,a)\right) = 0$$

RL algorithm: soft Q-learning

- $\hat{Q}_{t+1}(s,a) =$
  $\hat{Q}_t(s,a) - \alpha_t\Big(r_{t+1} + \gamma\,\mathrm{softmax}_{a';\epsilon}\,\hat{Q}_t(s_{t+1},a') - \hat{Q}_t(s_t,a_t)\Big)\delta_{s,s_t}\delta_{a,a_t}$
  (Update if $s = s_t, a = a_t$; otherwise, $\hat{Q}_{t+1}(s,a) = \hat{Q}_t(s,a)$)
- $\hat{\pi}_{t+1}(a|s) = \frac{\exp\big(\hat{Q}_{t+1}(s,a)/\epsilon\big)}{\sum_b \exp\big(\hat{Q}_{t+1}(s,b)/\epsilon\big)}$

Problem: only update one entry (one (s,a) pair) at each iteration; converges too slow.

- Solution: parameterize $Q(s, a)$ by $Q(s, a; w)$, and update w in each iteration.
- Parameterize function with a small number of parameters: **neural network**.

- Deep reinforcement learning:
- RMnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529.

Mathematical foundation: stochastic root finding problem

Given $f(x)$ and $f'(x) > 0$, find ξ s.t. $f(\xi) = 0$
But, one doesn't have access to f: for each x, one can sample from a random variable $\Phi(x)$, and $\mathbb{E}[\Phi(x)] = f(x)$.
(Robbins, Monro, 1951)

- Bad idea: for each x, sample 1000 times $\rightarrow$ calculate $f(x)$ almost exactly $\rightarrow$ find root.
- Good idea: sample less at far places, sample more near root.
- Algorithm:

$$x_0 : \text{starting point; obtain a sample } \phi_0(x_0)$$
$$x_{n+1} = x_n - \alpha_n \phi_n(x_n) \quad (\phi_n(x_n)\text{: obtained sample})$$

- Can prove the convergence $x_n \rightarrow \xi$, if $\sum_{j=1}^{\infty} \alpha_j = \infty, \sum_{j=1}^{\infty} \alpha_j^2 < \infty$ (and some conditions on f and ϕ).

Discussion

- Neural implementation?
- Physics application?

Spring College on the Physics of Complex Systems

ICTP

- 2018 Spring College on the Physics of Complex Systems (ICTP, Trieste Italy)
- Reinforcement Learning course by Antonio Celani

- Lectures and notes available at ICTP YouTube channel and Spring College website.