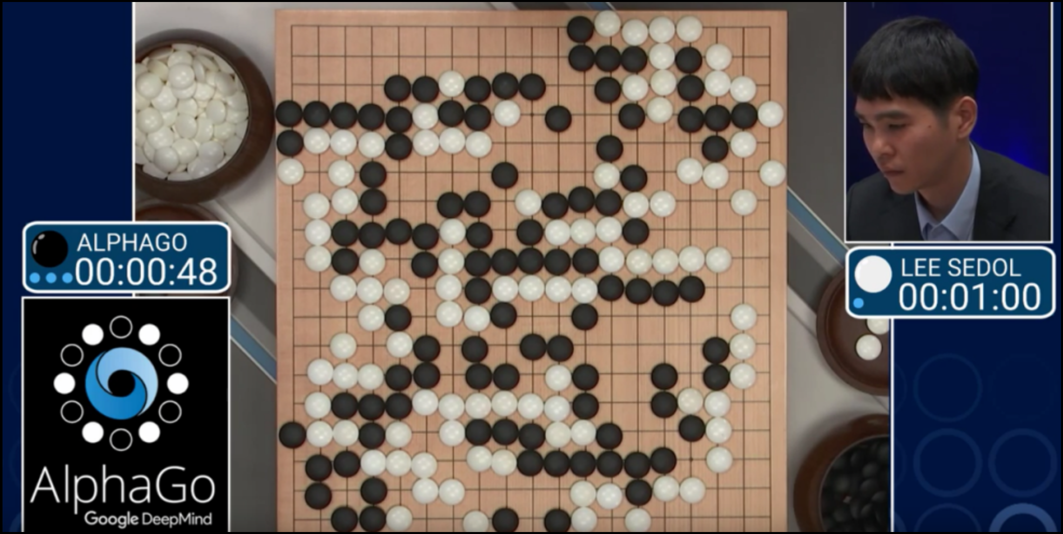
Introduction to Reinforcement Learning

Ather Gattami
Senior Research Scientist
RISE SICS
Stockholm, Sweden

February 21, 2018

Success Stories



Video

Outline

Reinforcement Learning in A Nutshell

Used in problems where actions (decisions) have to be made

Reinforcement Learning in A Nutshell

Used in problems where actions (decisions) have to be made

Each action (decision) affects future states of the system

Reinforcement Learning in A Nutshell

Used in problems where actions (decisions) have to be made

Each action (decision) affects future states of the system

Success is measured by a scalar reward signal

Reinforcement Learning in A Nutshell

Used in problems where actions (decisions) have to be made

Each action (decision) affects future states of the system

Success is measured by a scalar reward signal

Goal: Take actions (decisions) to maximize reward (or minimize cost) where **no system model is available**

Dynamical Systems

Let s_k, y_k, a_k be the state, observation, and action at time step k, respectively.

Deterministic model:

$$\begin{aligned} s_{k+1} &= f_k(s_k, a_k) \\ y_k &= g_k(s_k, a_k) \end{aligned}$$

Stochastic model (Markov Decision Process):

$$\begin{aligned} \mathbf{P}(s_{k+1} \mid s_k, a_k, s_{k-1}, a_{k-1}, \ldots) &= \mathbf{P}(s_{k+1} \mid s_k, a_k) \\ \mathbf{P}(y_k \mid s_k, a_k, s_{k-1}, a_{k-1}, \ldots) &= \mathbf{P}(y_k \mid s_k, a_k) \end{aligned}$$

We assume perfect state observation, that is $y_k = s_k$.

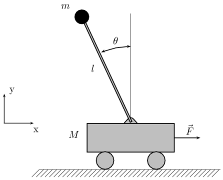
Dynamical Systems

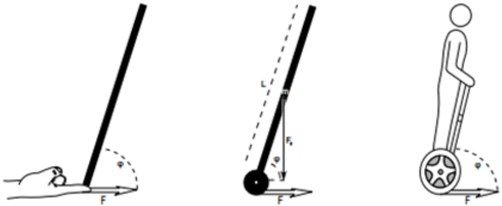
Given a dynamical system with states, observations, and actions given by s_k, y_k and a_k, respectively, and scalar valued rewards $r_k(s_k, a_k)$, find the actions a_k that maximize the average reward

$$R_T = \mathbb{E}\left(\sum_{k=1}^{T} \delta^k r_k(s_k, a_k)\right)$$

where $0 < \delta < 1$ is the discount factor.

Example





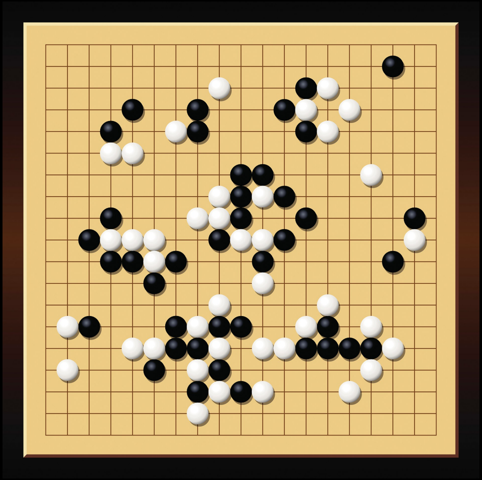
$$l\ddot{\theta} + \ddot{x}\cos\theta - g\sin\theta = 0$$

$$(M+m)\ddot{x} + ml\left(\ddot{\theta}\cos\theta - \dot{\theta}^2\sin\theta\right) = F(t)$$

Let $r_k(\theta_k, F_k) = -\theta_k^2$ where θ_k and F_k are time discretized values.

Example

What are the rewards in Go?



Bellman's Principle of Optimality

An optimal trajectory on the time interval $[T_1, T]$ must be optimal also on each of the subintervals $[T_1, T_1 + \epsilon]$ and $[T_1 + \epsilon, T]$.

Bellman's Equation

Definition

A policy $\pi(s_k)$ defines a probability distribution over actions given a state s_k,

$$\mathbf{P}(A_k = a_k \mid S_k = s_k)$$

For deterministic policies, the action is given by $a_k = \pi(s_k)$ with probability 1.

Bellman's Equation

Let $s_t = s$, $a_t = a$.

The total reward:

$$Q_0^\pi(s,a) = \mathbb{E}\left(\sum_{k=0}^{T} \delta^k r_k(s_k, a_k) \middle| s_0 = s, a_0 = a\right)$$

The reward to go:

$$Q_t^\pi(s,a) = \mathbb{E}\left(\sum_{k=t}^{T} \delta^{k-t} r_k(s_k, a_k) \middle| s_t = s, a_t = a\right)$$

Bellman's Equation

Let $s_t = s$, $a_t = a$.

The infinite reward (to go):

$$Q_t^\pi(s,a) = \mathbb{E}\left(\sum_{k=t}^{\infty} \delta^{k-t} r_k(s_k, a_k) \middle| s_t = s, a_t = a\right)$$

Bellman's Equation

$$
\begin{aligned}
&Q_i^\pi(s, a) \\
&= \mathbb{E}\left(r_i(s_i, a_i) + \sum_{k=i+1}^{T} \delta^{k-i} r_k(s_k, a_k) \middle| s_i = s, a_i = a \right) \\
&= \mathbb{E}\left(r_i(s, a) + \delta \cdot \sum_{k=i+1}^{T} \delta^{k-i-1} r_k(s_k, a_k) \right) \\
&= \mathbb{E}\left(r_i(s, a) + \delta \cdot \sum_{k=i+1}^{T} \delta^{k-(i+1)} r_k(s_k, a_k) \right) \\
&= \mathbb{E}\left(r_i(s, a) + \delta \cdot Q_{i+1}^\pi(s_+, a_+) \right)
\end{aligned}
$$

Bellman's Equation

Define

$$\pi^\star(s) = \arg\max_{\pi} Q_i^{\pi}(s, \pi(s))$$

and

$$Q_i^*(s, a) = Q_i^{\pi^*}(s, a)$$

The policy is optimal for all i:

$$\pi^*(s) = \arg\max_{a} Q_i^*(s, a)$$

Bellman's Equation

$$Q_i^*(s, a) = \mathbb{E}\left(r_i(s, a) + \delta \cdot Q_{i+1}^*(s_+, a_+)\right)$$

Dynamic Programming

$$Q_i^*(s,a) = \mathbb{E}\left(r_i(s,a) + \delta \cdot Q_{i+1}^*(s_+,a_+)\right)$$

The value function

$$V_i(s) = \max_a Q_i^*(s,a)$$

The **Bellman Equation** is given by

$$\begin{aligned} V_i(s) &= \max_a \mathbb{E}\left(r_i(s,a) + V_{i+1}(s_+)\right) \\ &= \max_a \sum_{s_+ \in S} \mathbf{P}(s_+ \mid s,a)\left(r_i(s,a) + V_{i+1}(s_+)\right) \end{aligned}$$

Model Free Optimization and Reinforcement Learning

What if we don't have the system model?

If the system is deterministic, the model is given by

$$s_{k+1} = f_k(s_k, a_k)$$

If the system is stochastic, the model is given by

$$\mathbf{P}(s_{k+1} \mid s_k, a_k)$$

Q-Learning

Let $s = s_k$ and $s_+ = s_{k+1}$.

Update rule with some $0 < \alpha_k(s_k, a_k) < 1$:

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(r(s,a) + \delta \max_{a_+} Q(s_+, a_+) - Q(s,a)\right)$$

The optimal policy is estimated from $Q(s,a)$:

$$\pi(s) = \arg\max_{a} Q(s,a)$$

Q-Learning

Theorem

Consider the Q-learning algorithm given by

$$Q(s,a) \leftarrow Q(s,a) + \alpha(s,a)(r(s,a) + \delta \max_{a_+} Q(s_+,a_+) - Q(s,a))$$

where

$$\sum_k \alpha_k(s,a) = \infty, \quad \sum_k \alpha_k^2(s,a) < \infty, \quad \forall\ (s,a)$$

The Q-learning algorithm converges to the optimal action-value function, $Q(s,a) \to Q^*(s,a)$.

Q-Learning

What if the state/action spaces are very large?

Deep Reinforcement Learning

Deep Reinforcement Learning:

Q function is approximated with a deep neural network.

Training:

Minimize the loss function with respect to the neural network weights $\mathbf{w}$

$$l(\mathbf{w}) = \left(r(s,a) + \delta \max_{a_+} Q(s_+, a_+, \mathbf{w}_-) - Q(s,a,\mathbf{w})\right)^2$$

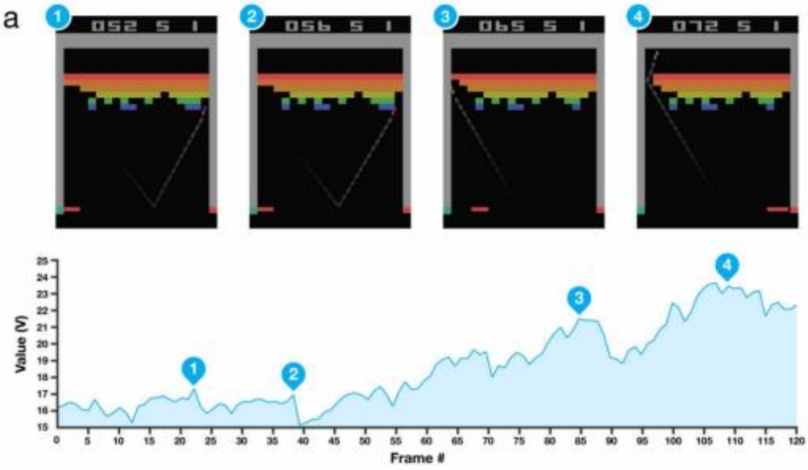
Deep Reinforcement Learning

Initialize $\mathbf{w}, \mathbf{w}_-$ arbitrarily
for (each episode): **do**
  Initialize s
  repeat
    Set $a = \arg\max_a Q(s, a, \mathbf{w})$
    Apply a, observe s_+
    Set $a_+ = \arg\max_{a_+} Q(s_+, a_+, \mathbf{w}_-)$
    $V(s) = r(s, a) + \delta Q(s_+, a_+, \mathbf{w})$
    $\mathbf{w}_- \leftarrow \mathbf{w}$
    Minimize $l(\mathbf{w}) = (V(s) - Q(s, a, \mathbf{w}))^2$
    $s \leftarrow s_+$
  until final state s
end for

Simulations

SIMULATIONS

Simulations



Research Questions

1. Extend Q-learning to continuous state/action spaces.

- $s(k+1) = As(k) + Ba(k)$, $y(k) = Cs(k)$. Only solved when C is left invertible and A is stable, simultaneously.

Research Questions

1. Extend Q-learning to continuous state/action spaces.
   - $s(k+1) = As(k) + Ba(k)$, $y(k) = Cs(k)$. Only solved when C is left invertible and A is stable, simultaneously.
   - Unsolved for unstable matrix A.
2. Explore structures for Q-learning to find $\arg\max_a Q(s,a)$ efficiently.

Research Questions

1. Extend Q-learning to continuous state/action spaces.
    - $s(k+1) = As(k) + Ba(k)$, $y(k) = Cs(k)$. Only solved when C is left invertible and A is stable, simultaneously.
    - Unsolved for unstable matrix A.
2. Explore structures for Q-learning to find $\arg\max_a Q(s,a)$ efficiently.
3. Analyze convergence of Deep Reinforcement Learning.

End of Presentation

QUESTIONS?