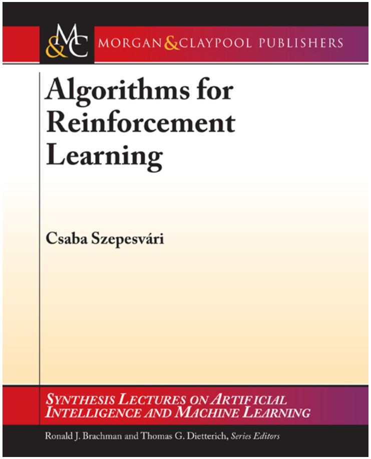
Reinforcement Learning, yet another introduction. Part 1/3: foundations

Emmanuel Rachelson (ISAE - SUPAERO)

Books



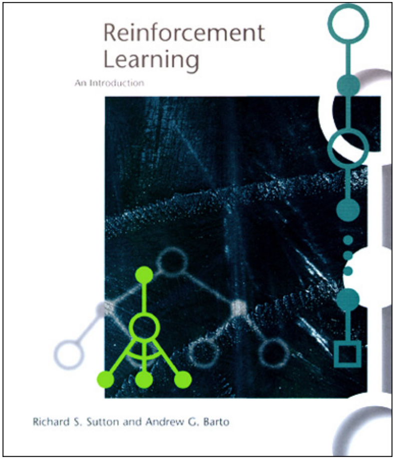
MORGAN & CLAYPOOL PUBLISHERS

Algorithms for Reinforcement Learning

Csaba Szepesvári

SYNTHESIS LECTURES ON ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

Ronald J. Brachman and Thomas G. Dietterich, *Series Editors*

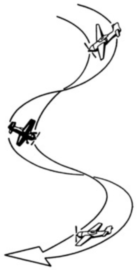


Reinforcement Learning

An Introduction

Richard S. Sutton and Andrew G. Barto

Example 1: exiting a spiral

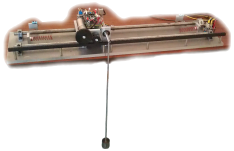


- Air speed, plane orientation and angular velocities, pressure measurements, commands' positions . . .
- Thrust, rudder, yoke.
- Vital maneuver.

Example 2: dynamic treatement regimes for HIV patients

- Anti-bodies concentration . . .
- Choice of drugs (or absence of treatment).
- Long-term, chronic diseases (HIV, depression, . . .).

Example 3: inverted pendulum



- x, $\dot{x}$, θ, $\dot{\theta}$.
- Push left or right (or don't push).
- Toy problem representative of many examples (Segway PT, juggling, plane control).

Example 4: queuing systems

- Line length, number of open counters, ...
- Open or close counters.
- Try to get all passengers on the plane in time (at minimal cost)!

Example 5: portfolio management

- Economic indicators, prices, . . .
- Dispatch investment over financial assets.
- Maximize long term revenue.

Example 6: hydroelectric production

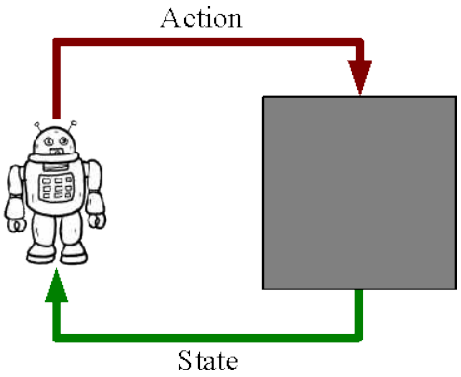
- Water level, electricity demand, weather forecast, other sources of energy . . .
- Evaluate “water value” and decide to use it or not.

Intuition

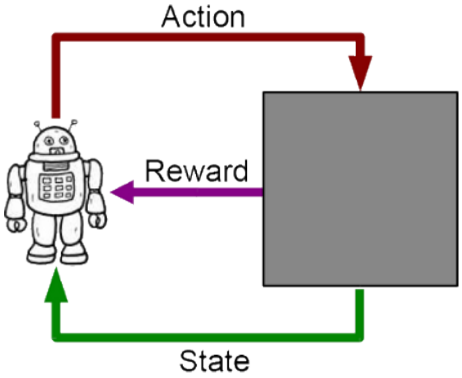
What do all these systems have in common?

- Prediction or decision over the future.
- Complex / high-dimension / non-linear / non-deterministic environments.
- What matters is not a single decision but the *sequence* of decisions.
- One can quantify the *value* of a sequence of decisions.

Sequential agent-environment interaction



Sequential agent-environment interaction



Questions

A general theory of sequential decision making?

What hypothesis on the “environment”?

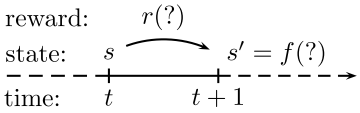
What is a “good” behaviour?

Is the knowledge of a model always necessary?

Balancing information acquisition and knowledge exploitation?

The ingredients

- Set T of *time steps* T.
- Set S of possible *states* s for the system.
- Set A of possible *actions* a of the agent.
- *Transition* dynamics of the system $s' \leftarrow f(?)$.
- *Rewards* (reinforcement signal) at each time step $r(?)$.



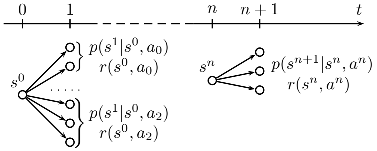
Markov Decision Processes

Sequential decision under probabilistic action uncertainty:

Markov Decision Process (MDP)

5-tuple $\langle S, A, p, r, T \rangle$
Markov transition model $p(s'|s,a)$
Reward model $r(s,a)$
Set T of decision epochs $\{0, 1, \ldots, H\}$

Infinite (or unbounded) horizon: $H \rightarrow \infty$



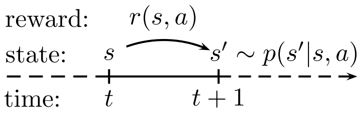
Markov Decision Processes

Sequential decision under probabilistic action uncertainty:

Markov Decision Process (MDP)

5-tuple $\langle S, A, p, r, T \rangle$
Markov transition model $p(s'|s,a)$
Reward model $r(s,a)$
Set T of decision epochs $\{0, 1, \ldots, H\}$

Infinite (or unbounded) horizon: $H \to \infty$



What is a behaviour?

Policy

A policy is a sequence of decision rules δ_t: $\pi = \{\delta_t\}_{t\in\mathbb{N}}$,

with $\delta_t : \left\{ \begin{array}{ccc} S^{t+1} \times A^t & \rightarrow & \mathscr{P}'(A) \\ h & \mapsto & \delta_t(a|h) \end{array} \right.$

$\delta_t(a|h)$ indicates

the distribution over action a

to undertake at time t, given

the history of states/actions h.

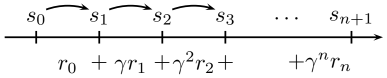
Evaluating a sequence of policy?

What can I expect on the long-term,

from this sequence of actions,

in my current state?

E.g.



Several criteria:

- Average reward $$V(s) = \mathbb{E}\left(\lim_{H\to\infty} \frac{1}{H}\sum_{\delta=0}^{H} r_\delta \,\middle|\, s_0 = s\right)$$
- Total reward $$V(s) = \mathbb{E}\left(\lim_{H\to\infty} \sum_{\delta=0}^{H} r_\delta \,\middle|\, s_0 = s\right)$$
- γ-discounted reward $$V(s) = \mathbb{E}\left(\lim_{H\to\infty} \sum_{\delta=0}^{H} \gamma^\delta r_\delta \,\middle|\, s_0 = s\right)$$

$\rightarrow$ value of a state under a certain behaviour.

Evaluating a policy

Value function of a policy under a γ-discounted criterion

$$V^\pi : \left\{ \begin{array}{lcl} S & \rightarrow & \mathbb{R} \\ s & \mapsto & V^\pi(s) = \mathbb{E}\left(\lim\limits_{H\to\infty} \sum\limits_{\delta=0}^{H} \gamma^\delta r_\delta \middle| s_0 = s, \pi \right) \end{array} \right.$$

Optimal policies

Optimal policy

π^* is said to be optimal iff $\pi^* \in \underset{\pi}{\operatorname{argmax}}\, V^{\pi}$.

A policy is optimal if it *dominates* over any other policy in every state:

$$\pi^* \text{ is optimal} \Leftrightarrow \forall s \in S,\ \forall \pi,\ V^{\pi^*}(s) \geq V^{\pi}(s)$$

First fundamental result

Fortunately...

Optimal policy

For $\left\{ \begin{array}{l} \gamma\text{-discounted criterion} \\ \text{infinite horizon} \end{array} \right.$, there always exists at least one optimal stationary, deterministic, Markovian policy.

- Markovian :
  $$\begin{array}{l} \forall (s_i, a_i) \in (S \times A)^{t-1} \\ \forall (s'_i, a'_i) \in (S \times A)^{t-1} \end{array}, \delta_t\left(a|s_0, a_0, \ldots, s_t\right) = \delta_t\left(a|s'_0, a'_0, \ldots, s_t\right).$$
  One writes $\delta_t(a|s)$.
- Stationary : $\forall (t, t') \in \mathbb{N}^2, \delta_t = \delta'_t$.
  One writes $\pi = \delta_0$.
- Deterministic : $\delta_t(a|h) = \left\{ \begin{array}{l} 1 \text{ for a single } a \\ 0 \text{ otherwise} \end{array} \right.$.

Let's play with actions

What's the value of "*a* then π"?

$$\begin{aligned}
Q^\pi(s,a) &= \mathbb{E}\left(\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\middle| s_0=s, a_0=a,\pi\right)\\
&= r(s,a)+\mathbb{E}\left(\sum_{t=1}^{\infty}\gamma^t r(s_t,a_t)\middle| s_0=s, a_0=a,\pi\right)\\
&= r(s,a)+\gamma\sum_{s'\in S} p\left(s'|s,a\right)\mathbb{E}\left(\sum_{t=1}^{\infty}\gamma^{t-1} r(s_t,a_t)\middle| s_1=s',\pi\right)\\
&= r(s,a)+\gamma\sum_{s'\in S} p\left(s'|s,a\right)V^\pi\left(s'\right)
\end{aligned}$$

The best one-step lookahead action can be selected by maximizing Q^π.

- To improve on a policy π, it is more useful to know Q^π than V^π and pick the *greedy* action.
- Also $V^\pi(s) = Q^\pi(s,\pi(s))$. Let's replace that above (next slide).

Computing a policy's value function

Evaluation equation

V^π is a solution to the linear system:

$$V^\pi(s) = r(s,\pi(s)) + \gamma \sum_{s'\in S} p(s'|s,\pi(s))\, V^\pi(s')$$
$$V^\pi = r^\pi + \gamma P^\pi V^\pi = T^\pi V^\pi$$

Similarly:

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s'\in S} p(s'|s,a)\, Q^\pi(s',\pi(s'))$$
$$Q^\pi = r + \gamma P Q^\pi = T^\pi Q^\pi$$

Recall also that $V^\pi(s) = \mathbb{E}\left(\sum_{\delta=0}^{\infty} \gamma^\delta r(s_\delta,\pi(s_\delta)) \middle| s_0 = s \right)$

Computing a policy's value function

Evaluation equation

V^π is a solution to the linear system:

$$V^\pi(s) = r(s,\pi(s)) + \gamma \sum_{s'\in S} p(s'|s,\pi(s))\, V^\pi(s')$$
$$V^\pi = r^\pi + \gamma P^\pi V^\pi = T^\pi V^\pi$$

Similarly:

$$Q^\pi(s,a) = r(s,a) + \gamma \sum_{s'\in S} p(s'|s,a)\, Q^\pi(s',\pi(s'))$$
$$Q^\pi = r + \gamma P Q^\pi = T^\pi Q^\pi$$

Notes:

- For continuous state and action spaces $\sum \rightarrow \int$
- For stochastic policies: $\forall s \in S, \quad V^\pi(s) = \sum_{a\in A} \pi(s,a) \left(r(s,a) + \gamma \sum_{s'\in S} p(s'|s,a) V^\pi(s') \right)$

Properties of T^π

$$T^\pi V^\pi(s) = r(s,\pi(s)) + \gamma \sum_{s'\in S} p\left(s'|s,\pi(s)\right) V^\pi(s')$$
$$T^\pi V^\pi = r^\pi + \gamma P^\pi V^\pi$$

Solving the evaluation equation

- T^π is linear.

$\Rightarrow$ Solving $V^\pi = T^\pi V^\pi$ and $Q^\pi = T^\pi Q^\pi$ by matrix inversion?
With $\gamma < 1$, $V^\pi = (I - \gamma P^\pi)^{-1} r^\pi$ and $Q^\pi = (I - \gamma P)^{-1} r^\pi$

- With $\gamma < 1$, T^π is a $\|\cdot\|_\infty$-contraction mapping over the $\mathscr{F}(S,\mathbb{R})$ (resp. $\mathscr{F}(S\times A,\mathbb{R})$) Banach space.

$\Rightarrow$ With $\gamma < 1$, V^π (resp. Q^π) is the unique solution to the (linear) fixed point equation $V = T^\pi V$ (resp. $Q = T^\pi Q$).

Characterizing an optimal policy

$$\text{Find } \pi^* \text{ such that } \pi^* \in \underset{\pi}{\operatorname{argmax}}\, V^{\pi}(s).$$
$$\text{Notation: } V^{\pi^*} = V^*,\ Q^{\pi^*} = Q^*$$

Let's play with our intuitions:

- One has $Q^*(s,a) = r(s,a) + \gamma \sum\limits_{s' \in S} p(s'|s,a)\, V^*(s')$.
- If π^* is an optimal policy, then $V^*(s) = Q^*(s, \pi^*(s))$.

Optimal greedy policy

Any policy π defined by $\pi(s) \in \underset{a \in A}{\operatorname{argmax}}\, Q^*(s,a)$ is an optimal policy.

Bellman optimality equation

The key theorem:

Bellman optimality equation

The optimal value function obeys:

$$V^*(s) = \max_{a \in A} \left\{ r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^*(s') \right\} = T^* V^*$$

or in terms of Q-functions:

$$Q^*(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) \max_{a' \in A} Q^*(s',a') = T^* Q^*$$

Properties of T^*

$$V^*(s) = \max_\pi V^\pi(s)$$

$$V^*(s) = \max_{a\in A} \left\{ r(s,a) + \gamma \sum_{s'\in S} p(s'|s,a) V^*(s) \right\} = T^* V^*$$

Solving the optimality equation

- T^* is non-linear.
- T^* is a $\|\cdot\|_\infty$-contraction mapping over the $\mathscr{F}(S,\mathbb{R})$ (resp. $\mathscr{F}(S\times A,\mathbb{R})$) Banach space.

$\Rightarrow$ V^* (resp. Q^*) is the unique solution to the fixed point equation $V = TV$ (resp. $Q = TQ$).

Let's summarize

Formalizing the control problem:

- Environment (discrete time, non-deterministic, non-linear) $\leftrightarrow$ MDP.
- Behaviour $\leftrightarrow$ control policy $\pi : s \mapsto a$.
- Policy evaluation criterion $\leftrightarrow$ γ-discounted criterion.
- Goal $\leftrightarrow$ Maximize value function $V^\pi(s)$, $Q^\pi(s,a)$.
- Evaluation eq. $\leftrightarrow$ $V^\pi = T^\pi V^\pi$, $Q^\pi = T^\pi Q^\pi$.
- Bellman optimality eq. $\leftrightarrow$ $V^* = T^* V^*$, $Q^* = T^* Q^*$.

Now what?

- p and r are known $\rightarrow$ Probab. Planning, Stochastic Optimal Control.
- p and r are unknown $\rightarrow$ Reinforcement Learning.

How does one find π^*?

Three “standard” approaches:

- Dynamic Programming in value function space
  $\rightarrow$Value Iteration
- Dynamic Programming in policy space
  $\rightarrow$Policy Iteration
- Linear Programming in value function space

We won’t see each algorithm in detail, nor explain all their variants. The goal of this section is to illustrate three fundamentally different ways of computing an optimal policy, based on Bellman’s optimality equation.

Value Iteration

Key idea:

$$V^*(s) = \max_{a\in A}\left\{ r(s,a) + \gamma \sum_{s'\in S} p(s'|s,a) V^*(s) \right\} = T^* V^*$$

Value iteration

- T^* is a contraction mapping,
- Value function space is a Banach space.

$\Rightarrow$ The sequence $V_{n+1} = T^* V_n$ converges to V^*.

π^* is the V^*-greedy policy.

Value Iteration

```
Init: $V' \leftarrow V_0$.
repeat
    $V = V'$
    for $s \in S$ do
        $V'(s) \leftarrow \max_{a \in A} \left\{ r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V(s') \right\}$
until $\|V' - V\| \leq \varepsilon$
return greedy policy w.r.t. $V'$
```

Value Iteration

```
Init: $Q' \leftarrow Q_0$.
repeat
    $Q = Q'$
    for $(s,a) \in S \times A$ do
        $Q'(s,a) \leftarrow r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) \max_{a' \in A} Q(s',a')$
until $\|Q' - Q\| \leq \varepsilon$
return greedy policy w.r.t. $Q'$
```

Illustration - an investment dilemma

A gambler's bet on a coin flip:

- Tails $\Rightarrow$ looses his stake.
- Heads $\Rightarrow$ wins as much as his stake.

Goal	reach 100 pesos!
States	$S = \{1, \dots, 99\}$.
Actions	$A = \{1, 2, \dots, \min(s, 100 - s)\}$
Rewards	$+1$ when the gambler reaches 100 pesos.
Transitions	Probability of heads-up $= p$.
Discount	$\gamma = 1$

$V^{\pi} \rightarrow$ probability of reaching the goal.

π^* maximizes V^{π}

Illustration - an investment dilemma, $p = 0.4$

Matlab demo.

Policy Iteration

Key idea:

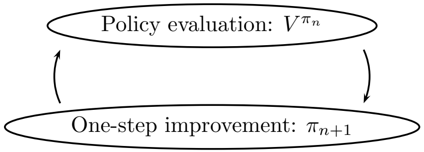
$$\pi^* = \arg\max_{\pi} V^{\pi}$$
$$V^{\pi} = r(s,\pi(s)) + \gamma \sum_{s' \in S} p(s'|s,\pi(s)) V^{\pi}(s') = T^{\pi} V^{\pi}$$

Policy iteration

A policy that is Q^{π}-greedy is not worse than π.
$\rightarrow$ iteratively improve and evaluate the policy.

Instead of a path $V_0, V_1, \ldots$ among value functions, let's search for a path $\pi_0, \pi_1, \ldots$ among policies.

Policy Iteration


Policy evaluation: V^{π_n}
One-step improvement: π_{n+1}

Policy Iteration

```
Init: $\pi' \leftarrow \pi_0$.
repeat
    $\pi \leftarrow \pi'$
    $V^{\pi} \leftarrow$ Solve $V = T^{\pi} V$
    for $s \in S$ do
        $\pi'(s) \leftarrow \arg\max_{a \in A} \left\{ r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V^{\pi}(s') \right\}$
until $\pi' = \pi$
return $\pi$
```

Policy Iteration

```
Init: $\pi' \leftarrow \pi_0$.
repeat
    $\pi \leftarrow \pi'$
    $Q^{\pi} \leftarrow$ Solve $Q = T^{\pi} Q$
    for $s \in S$ do
        $\pi'(s) \leftarrow \arg\max_{a \in A} Q^{\pi}(s, a)$
until $\pi' = \pi$
return $\pi$
```

Illustration - an investment dilemma, $p = 0.4$

Matlab demo.

Linear Programming

Key idea: formulate the optimality equation as a linear problem.
"V^* is the smallest value that dominates over all policy values"

$$\forall s \in S, V(s) = \max_{a \in A} \left\{ r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V(s') \right\}$$

Linear Programming

Key idea: formulate the optimality equation as a linear problem.
"V^* is the smallest value that dominates over all policy values"

$$\forall s \in S, V(s) = \max_{a \in A} \left\{ r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V(s') \right\}$$

$$\Leftrightarrow$$

$$\left\{ \begin{array}{c} \min \sum_{s \in S} V(s) \\ s.t.\ \forall \pi,\ V \geq T^{\pi} V \end{array} \right.$$

Linear Programming

Key idea: formulate the optimality equation as a linear problem.
"V^* is the smallest value that dominates over all policy values"

$$\forall s \in S, V(s) = \max_{a \in A} \left\{ r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V(s') \right\}$$

$$\Leftrightarrow$$

$$\left\{ \begin{array}{c} \min \sum_{s \in S} V(s) \\ s.t.\ \forall \pi,\ V \geq T^{\pi} V \end{array} \right.$$

$$\Leftrightarrow$$

$$\left\{ \begin{array}{c} \min \sum_{s \in S} V(s) \\ s.t.\ \forall (s,a) \in S \times A, \quad V(s) - \gamma \sum_{s' \in S} p(s'|s,a) V(s') \geq r(s,a) \end{array} \right.$$

In a nutshell

To solve Bellman's optimality equation:

- The sequence of $V_{n+1} = T^* V_n$ converges to V^*
  $\rightarrow$ Value Iteration.
- The sequence of $\pi_{n+1} \in \underset{a}{\operatorname{argmax}} Q^{\pi_n}$ converges to π^*
  $\rightarrow$ Policy Iteration.
- V^* is the smallest function s.t. $V(s) \geq r(s,a) + \gamma \sum\limits_{s' \in S} p(s'|s,a) V(s)$
  $\rightarrow$ Linear Programming resolution.

Wait a minute…

…so far, we've *characterized* and *searched for* optimal policies, using the supposed properties (p and r) of the environment.

We've been using p and r each time! We're cheating!

Where's the *learning* you promised?

We're coming to it. Let's put it all in perspective.

Let’s put it all in perspective: RL within ML

A taxonomy of Machine Learning

Supervised Learning

Unsupervised Learning

Reinforcement Learning

Let’s put it all in perspective: RL within ML

Different learning tasks

$\boxed{\text{Supervised Learning}}$

- learning from a teacher

$\boxed{\text{Unsupervised Learning}}$

- learning from similarity

$\boxed{\text{Reinforcement Learning}}$

- learning by interaction

Let's put it all in perspective: RL within ML

What kind of input?

$\boxed{\text{Supervised Learning}}$

- learning from a teacher
- information: correct examples

$\boxed{\text{Unsupervised Learning}}$

- learning from similarity
- information: unlabeled examples

$\boxed{\text{Reinforcement Learning}}$

- learning by interaction
- information: trial and error

Let’s put it all in perspective: RL within ML

For what goal?

Supervised Learning

- learning from a teacher
- information: correct examples
- generalize from examples

Unsupervised Learning

- learning from similarity
- information: unlabeled examples
- identify structure in the data

Reinforcement Learning

- learning by interaction
- information: trial and error
- reinforce good choices

Let’s put it all in perspective: RL within ML

Which outputs?

Supervised Learning

- learning from a teacher
- information: correct examples
- generalize from examples
- classifier, regressor

Unsupervised Learning

- learning from similarity
- information: unlabeled examples
- identify structure in the data
- clusters, self-organized data

Reinforcement Learning

- learning by interaction
- information: trial and error
- reinforce good choices
- value function, control policy

Let’s put it all in perspective: RL within ML

Examples of algorithms

Supervised Learning

- learning from a teacher
- information: correct examples
- generalize from examples
- classifier, regressor
- SVMs, neural networks, trees

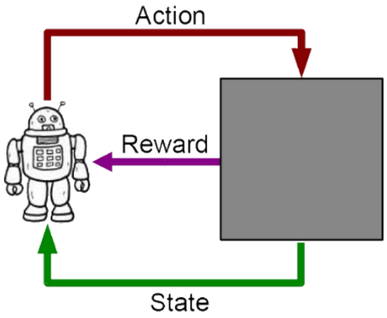
Unsupervised Learning

- learning from similarity
- information: unlabeled examples
- identify structure in the data
- clusters, self-organized data
- k-means, Kohonen maps, PCA

Reinforcement Learning

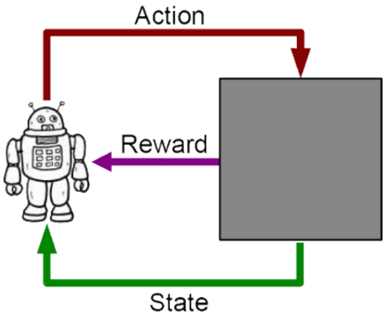
- learning by interaction
- information: trial and error
- reinforce good choices
- value function, control policy
- TD-learning, Q-learning

Reinforcement Learning



Evaluate and *improve* a policy based on *experience samples*.

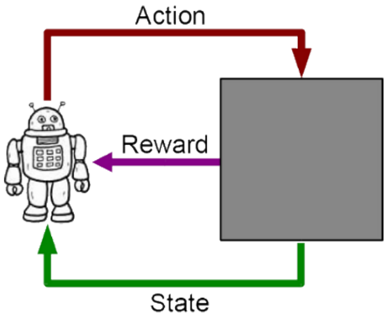
Reinforcement Learning



experience samples?

$$\rightarrow (s, a, r, s')$$

Reinforcement Learning



Two problems in RL:

- *Predict* a policy's value.
- *Control* the system.

A little vocabulary

- Curse of dimensionality

Number of states, actions or outcomes grows exponentially with number of dimensions.
E.g. continuous control problem in $S = [0;1]^{10}$, discretized with a step-size of $1/10 \rightarrow 10^{10}$ states!

A little vocabulary

- Curse of dimensionality
- Exploration/exploitation dilemma

Where are the good rewards?
Exploit whatever good policy has been found so far or explore unknown transitions hoping for more?
How to balance exploration and exploitation?

A little vocabulary

- Curse of dimensionality
- Exploration/exploitation dilemma
- Model-based vs model-free RL

Also called indirect vs. direct RL.
Indirect: $\{(s,a,r,s')\} \quad \rightarrow \quad (p,r) \quad \rightarrow \quad V^{\pi} \text{ or } \pi^*$
Direct: $\{(s,a,r,s')\} \quad \rightarrow \quad V^{\pi} \text{ or } \pi^*$

A little vocabulary

- Curse of dimensionality
- Exploration/exploitation dilemma
- Model-based vs model-free RL
- Interactive vs. non-interactive algorithms

Non-interactive: $\mathscr{D} = \{(s_i, a_i, r_i, s_i')\}_{i\in[1;N]}$
$\rightarrow$ no exploration/exploitation dilemma; batch learning.
Interactive episodic: trajectories $(s_0, a_0, r_0, s_1, \dots, s_N, a_N, r_N, s_{N+1})$
$\rightarrow$ Interactive "with reset"; Monte-Carlo-like methods; is s_0 known?
Interactive non-episodic: (s, a, r, s') at each time step
$\rightarrow$ the most general case!

A little vocabulary

- Curse of dimensionality
- Exploration/exploitation dilemma
- Model-based vs model-free RL
- Interactive vs. non-interactive algorithms
- On-policy vs. off-policy algorithms

Evaluate/improve policy π while applying π'?

Next classes

1. *Predict* a policy's value.
    1. Model-based prediction
    2. Monte-Carlo methods
    3. Temporal differences
    4. Unifying MC and TD: TD(λ)
2. *Control* the system.
    1. Actor-Critic architectures
    2. Online problems, the exploration vs. exploitation dilemma
    3. Offline problems, focussing on the critic alone
    4. An overview of control learning