Reinforcement Learning, yet another introduction. Part 2/3: Prediction problems

Emmanuel Rachelson (ISAE - SUPAERO)

Quiz!

- What's an MDP?

Quiz!

- What's an MDP?

$$\{S, A, p, r, T\}$$

Quiz!

- What's an MDP?

$$\{S, A, p, r, T\}$$

- Deterministic, Markovian, Stationary policies?

Quiz!

- What's an MDP?

$$\{S, A, p, r, T\}$$

- Deterministic, Markovian, Stationary policies?

At least one is optimal

Quiz!

- What's an MDP?

$$\{S, A, p, r, T\}$$

- Deterministic, Markovian, Stationary policies?

At least one is optimal

- Evaluation equation?

Quiz!

- What's an MDP?

$$\{S, A, p, r, T\}$$

- Deterministic, Markovian, Stationary policies?

At least one is optimal

- Evaluation equation?

$$Q^{\pi}(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,\pi(s)) Q^{\pi}(s',\pi(s'))$$

Quiz!

- What's an MDP?

$$\{S, A, p, r, T\}$$

- Deterministic, Markovian, Stationary policies?

At least one is optimal

- Evaluation equation?

$$Q^{\pi}(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,\pi(s)) Q^{\pi}(s',\pi(s'))$$

- Value Iteration?

Quiz!

- What's an MDP?

$$\{S, A, p, r, T\}$$

- Deterministic, Markovian, Stationary policies?

At least one is optimal

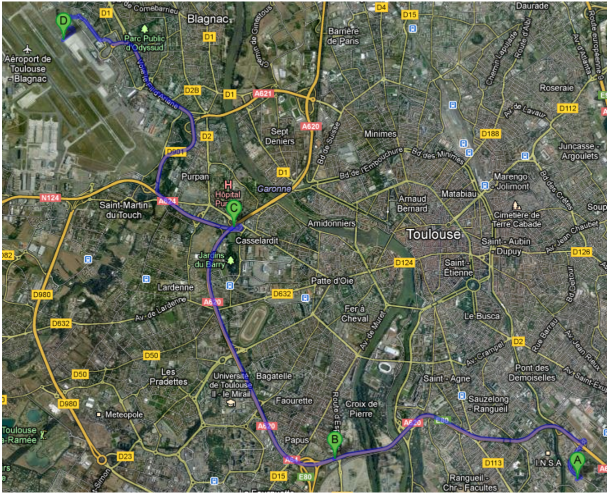
- Evaluation equation?

$$Q^{\pi}(s,a) = r(s,a) + \gamma \sum_{s' \in S} p(s'|s,\pi(s)) Q^{\pi}(s',\pi(s'))$$

- Value Iteration?

$$\text{Repeat } V_{n+1}(s) = \max_{a \in A} \left\{ r(s,a) + \gamma \sum_{s' \in S} p(s'|s,a) V_n^*(s) \right\}$$

Challenge!



Estimate the travel time to D, when in A, B, C ?

Model-based prediction

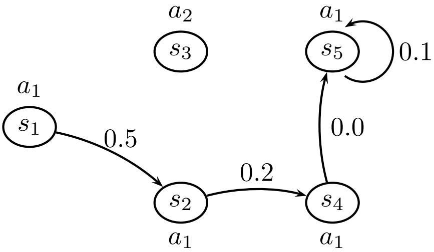
...or Adaptive Dynamic Programming, or Indirect RL.

$$\{(s,a,r,s')\}$$
$$\downarrow$$
$$\begin{cases} \text{Frequency count or parametric adaptation} & \rightarrow \quad p \\ \text{Average} & \rightarrow \quad r \end{cases}$$
$$\downarrow$$
$$\text{Solve } V = T^\pi V.$$

Properties

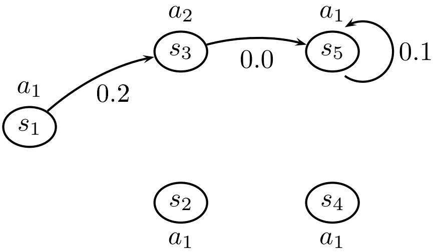
- Converges to p, r and V^π if i.i.d. samples.
- Works online and offline.

Example



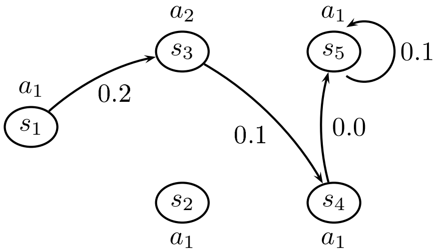
Happened once

Example



Happened 3 times

Example



Happened 6 times

Example

$$\Rightarrow \hat{P}(s'|s_1,\pi(s_1)) = \begin{cases} 0.1 & \text{if } s' = s_2 \\ 0.9 & \text{if } s' = s_3 \\ 0 & \text{otherwise} \end{cases}$$

and $r(s_1,\pi(s_1)) = 0.1 \cdot 0.5 + 0.9 \cdot 0.2 = 0.23$

...

$\hat{P}(s'|s,\pi(s)) \to P^\pi$ and $r(s,\pi(s)) \to r^\pi$

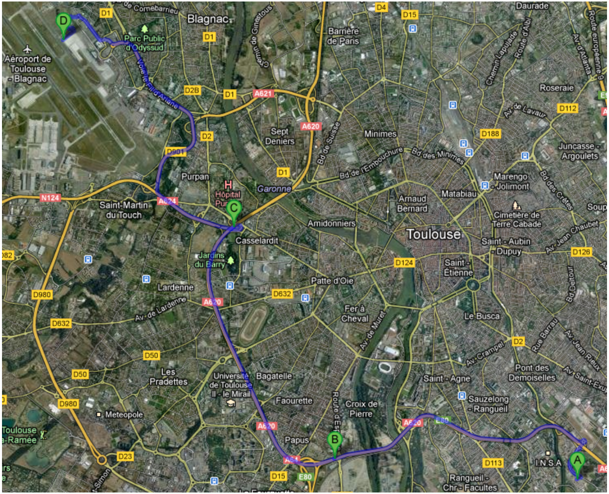
Solve $V^\pi = (I - \gamma P^\pi)^{-1} r^\pi$

Model-based prediction

- Incremental version: straightforward
- Does not require full episodes for model updates
- Requires maintaining a memory of the model
- Has to be adapted for continuous domains
- Requires many resolutions of $V^\pi = T^\pi V^\pi$

Question

And without a model ?



Offline Monte-Carlo

Episode-based method

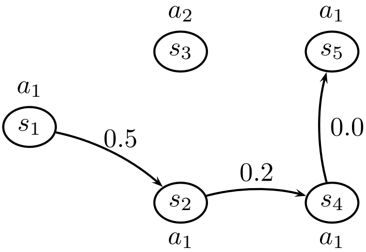
Data: a set of trajectories.

$$\mathscr{D} = \{h_i\}_{i\in[1,N]},\ h_i = (s_{i0}, a_{i0}, r_{i0}, s_{i1}, a_{i1}, r_{i1}, \ldots)$$

$$R_{ij} = \sum_{k\geq j} \gamma^{k-j} r_{ik}$$

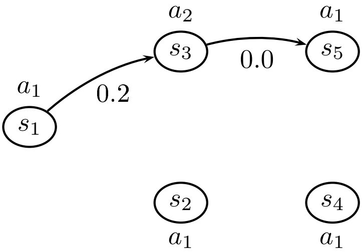
$$V^{\pi}(s) = \frac{\sum\limits_{ij} R_{ij} \mathbb{1}_s(s_{ij})}{\sum\limits_{ij} \mathbb{1}_s(s_{ij})}$$

Example



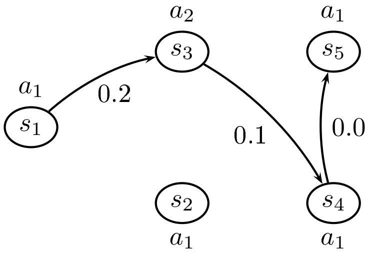
Happened once

Example



Happened 3 times

Example



Happened 6 times

Example

$$V^{\pi}(s_3) = \frac{3 \cdot 0.0 + 6 \cdot 0.1}{3 + 6} = \frac{1}{15}$$

Offline Monte-Carlo

- Requires finite-length episodes
- Requires to remember full episodes
- Online version ?

Online Monte-Carlo

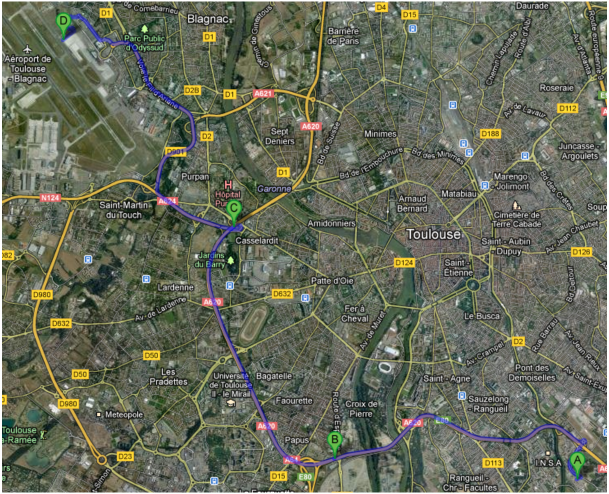
After each episode, update each encountered state's value.
Episode: $h = (s_0, a_0, r_0, \ldots)$

$$R_t = \sum_{i>t} \gamma^{i-t} r_t$$

$$V^\pi(s_t) \leftarrow V^\pi(s_t) + \alpha \left[R_t - V^\pi(s_t)\right]$$

Example

Driving home!



Online Monte-Carlo

- Requires finite-length episodes.
- Only requires to remember one episode at a time.
- Converges to V^π if (Robbins-Monroe conditions):

$$\sum_{t=0}^{\infty} \alpha_t = \infty \quad \text{and} \quad \sum_{t=0}^{\infty} \alpha_t^2 < \infty.$$

- One rare event along the episode affects the estimate of all previous states.

Wasn't it possible to update $A \rightarrow D$'s expected value as soon as we observe a new $A \rightarrow B$?

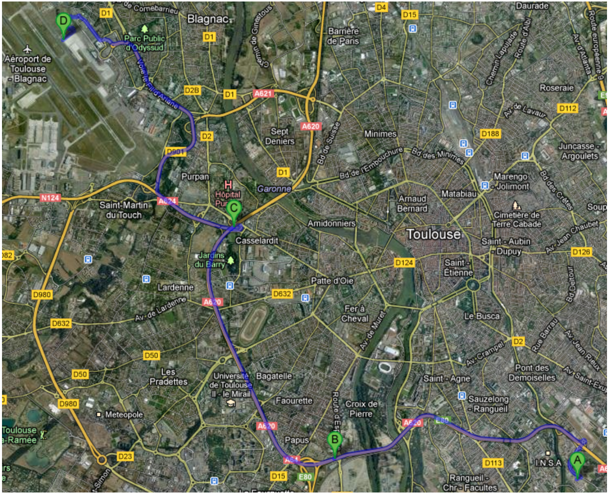
TD(0)

With each sample (s_t, a_t, r_t, s_{t+1}):

$$V^\pi(s_t) \leftarrow V^\pi(s_t) + \alpha\left[r_t + \gamma V^\pi(s_{t+1}) - V^\pi(s_t)\right]$$

Example

Driving home!



- $r_t + \gamma V^\pi(s_{t+1}) - V^\pi(s_t)$ = prediction *temporal difference*
- Using $V^\pi(s_{t+1})$ to update $V^\pi(s_t)$ is called *bootstrapping*
- Sample-by-sample update, no need to remember full episodes.
- Adapted to non-episodic problems.
- Converges to V^π if (Robbins-Monroe conditions):

$$\sum_{t=0}^{\infty} \alpha_t = \infty \quad \text{and} \quad \sum_{t=0}^{\infty} \alpha_t^2 < \infty.$$

- Usually, TD methods converge faster than MC, but not always!

TD(λ)

Can we have the advantages of both MC and TD methods?

What's inbetween TD and MC?

TD(0): 1-sample update with bootstrapping
MC: ∞-sample update no bootstrapping
inbetween: *n*-sample update with bootstrapping

n-step TD updates

Take a finite-length episode $(s_t, r_t, s_{t+1}, \ldots, s_T)$

$R_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots + \gamma^{T-t-1} r_{T-1}$	MC
$R_t^{(1)} = r_t + \gamma V_t(s_{t+1})$	1-step TD = TD(0)
$R_t^{(2)} = r_t + \gamma r_{t+1} + \gamma^2 V_t(s_{t+2})$	2-step TD
$R_t^{(n)} = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \ldots + \gamma^n V_t(s_{t+n})$	n-step TD

$R_t^{(n)}$ is the n-step *target* or n-step *return*.

MC method: ∞-step returns.

n-step temporal difference:

$$V(s_t) \leftarrow V(s_t) + \alpha \left[R_t^{(n)} - V(s_t) \right]$$

n-step TD updates

- Converge to the true V^π, just like TD(0) and MC methods.
- Needs to wait for n steps to perform updates.
- Not really used but useful for what follows.

Mixing n-step and m-step returns

Consider $R_t^{mix} = \frac{1}{3} R_t^{(2)} + \frac{2}{3} R_t^{(4)}$.

$$V(s_t) \leftarrow V(s_t) + \alpha \left[R_t^{mix} - V(s_t) \right]$$

Converges to V^π as long as the weights sum to 1!

λ-return (1/2)

Consider the λ-return $R_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} R_t^{(n)}$.

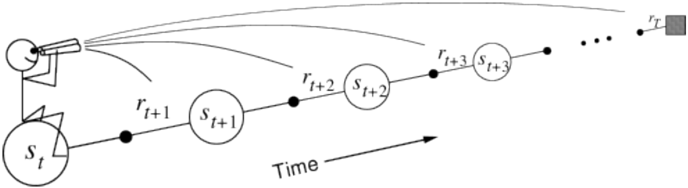
The λ-return is the mixing of *all* n-step returns, with weights $(1-\lambda)\lambda^n$.

λ-return (1/2)

Consider the λ-return $R_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} R_t^{(n)}$.

The λ-return is the mixing of *all* n-step returns, with weights $(1-\lambda)\lambda^n$.

$$V(s_t) \leftarrow V(s_t) + \alpha\left[R_t^\lambda - V(s_t)\right]$$



λ-return (1/2)

Consider the λ-return $R_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} R_t^{(n)}$.

The λ-return is the mixing of *all* n-step returns, with weights $(1-\lambda)\lambda^n$.

On a finite length episode of length T, $\forall k > 0,\ R_t^{(T-t+k)} = R_t$.

$$\begin{aligned}
R_t^\lambda &= (1-\lambda)\sum_{n=1}^{T-t-1} \lambda^{n-1} R_t^{(n)} + (1-\lambda)\sum_{n=T-t}^{\infty} \lambda^{n-1} R_t^{(n)} \\
&= (1-\lambda)\sum_{n=1}^{T-t-1} \lambda^{n-1} R_t^{(n)} + (1-\lambda)\lambda^{T-t-1}\sum_{n=T-t}^{\infty} \lambda^{n-T+t} R_t^{(n)} \\
&= (1-\lambda)\sum_{n=1}^{T-t-1} \lambda^{n-1} R_t^{(n)} + (1-\lambda)\lambda^{T-t-1}\sum_{k=0}^{\infty} \lambda^{k} R_t^{(T-t+k)} \\
&= (1-\lambda)\sum_{n=1}^{T-t-1} \lambda^{n-1} R_t^{(n)} + \lambda^{T-t-1} R_t
\end{aligned}$$

λ-return (2/2)

With $R_t^\lambda = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} R_t^{(n)}$, on finite episodes:

$$R_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1} \lambda^{n-1} R_t^{(n)} + \lambda^{T-t-1} R_t$$

When $\lambda = 0$, TD(0)!
When $\lambda = 1$, MC!

$$V(s_t) \leftarrow V(s_t) + \alpha \left[R_t^\lambda - V(s_t) \right]$$

λ-return algorithm.

But how do we compute R_t^λ without running infinite episodes?

Eligibility traces

Eligibity trace of state s: $e_t(s)$.

$$e_t(s) = \begin{cases} \gamma\lambda\, e_{t-1}(s) & \text{if } s \neq s_t \\ \gamma\lambda\, e_{t-1}(s) + 1 & \text{if } s = s_t \end{cases}$$

If no visit to a state, exponential decay.

$\rightarrow$ $e_t(s)$ measures how old the last visit to s is.

TD(λ)

Given a new sample (s_t, a_t, r_t, s_t').

1. Temporal difference $\delta = r_t + \gamma V(s_t') - V(s_t)$.
2. Update eligibility traces for all states
$$e(s) \leftarrow \begin{cases} \gamma\lambda e(s) & \text{if } s \neq s_t \\ \gamma\lambda e_{t-1}(s) + 1 & \text{if } s = s_t \end{cases}$$
3. Update all state's values $V(s) \leftarrow V(s) + \alpha e(s)\delta$

Initially, $e(s) = 0$.

TD(λ)

Given a new sample (s_t, a_t, r_t, s'_t).

1. Temporal difference $\delta = r_t + \gamma V(s'_t) - V(s_t)$.
2. Update eligibility traces for all states
$$e(s) \leftarrow \begin{cases} \gamma\lambda e(s) & \text{if } s \neq s_t \\ \gamma\lambda e_{t-1}(s) + 1 & \text{if } s = s_t \end{cases}$$
3. Update all state's values $V(s) \leftarrow V(s) + \alpha e(s)\delta$

Initially, $e(s) = 0$.

- If $\lambda = 0$, $e(s) = 0$ except in $s_t \Rightarrow$ standard TD(0)
- For $0 < \lambda < 1$, $e(s)$ indicates a distance $s \leftrightarrow s_t$ is in the episode.
- If $\lambda = 1$, $e(s) = \gamma^\tau$ where $\tau =$ duration since last visit to $s_t \Rightarrow$ MC method

Earlier states are given $e(s)$ *credit* for the TD error δ

TD(1)

- TD(1) implements Monte Carlo estimation on non-episodic problems!
- TD(1) learns incrementally for the same result as MC

Equivalence

TD(λ) is equivalent to the λ-return algorithm.

Prediction problems — summary

- Prediction = evaluation of a given behaviour
- Model-based prediction
- Monte Carlo (offline and online)
- Temporal Differences, TD(0)
- Unifying MC and TD: TD(λ)

Going further

- Best value of λ?
- Other variants?
- Very large state spaces?
- Continuous state spaces?
- Value function approximation?

Next class

Control

1. Online problems
   1. Q-learning
   2. SARSA
2. Offline learning
   1. (fitted) Q-iteration
   2. (least squares) Policy Iteration