Aalto University
School of Electrical Engineering

ELEC-E8125 Reinforcement Learning
Reinforcement learning in discrete domains

Ville Kyrki

22.9.2020

Today

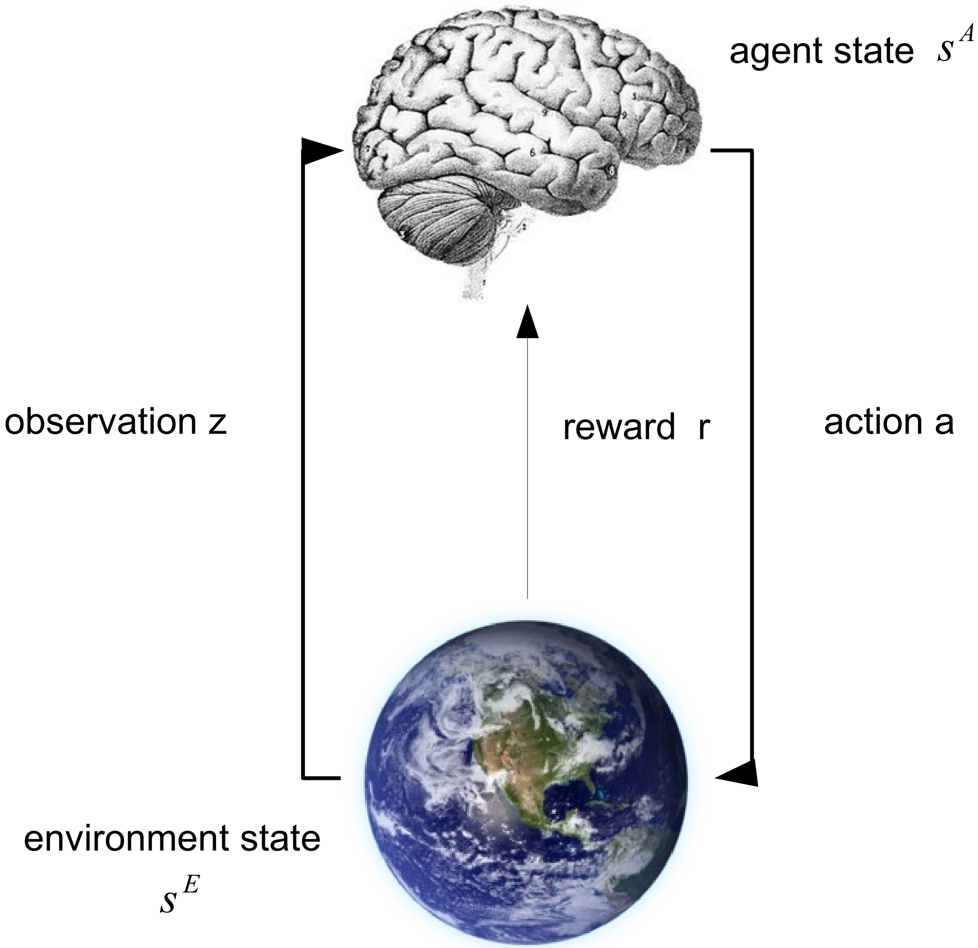
- Reinforcement learning
- Policy evaluation vs control problems
- Monte-Carlo and Temporal difference

Learning goals

- Understand basic concepts of RL.
- Understand Monte-Carlo and temporal difference approaches for policy evaluation and control.

- Be able to implement MC and TD.

Reinforcement learning



RL
MDP with **unknown**
Markovian dynamics
$P(s_{t+1}|s_t, a_t)$

Unknown reward function
$r_t = r(s_{t+1}, s_t)$

Solution similar, e.g.
$a^*_{1,\dots,T} = max_{a_1,\dots,a_T} \sum_{t=1}^{T} r_t$

Learning must **explore** policies

How does this then relate to the control case?

Reinforcement learning

- MDP with unknown dynamics (*T*) and reward function (*r*)

- Model based RL: Estimate MDP, apply MDP methods.
  - Estimate MDP transition and reward functions from data.

- Can we do without *T* and *r*?
  - Can we evaluate a policy (estimate value function) if we have multiple episodes (in episodic tasks) available?

Monte-Carlo policy evaluation

- Complete episodes give us samples of return *G*.
- Learn value of particular policy from episodes under that policy.

$$V_\pi(s) = E_\pi\left[G_t \middle| s_t = s\right] \qquad G_t = \sum_{k=0}^{H} \gamma^k r_{t+k+1}$$

- Estimate value as empirical mean return.
  - Each time state *s* visited in an episode,

$$N(s) = N(s) + 1 \quad S(s) = S(s) + G_t \quad V(s) = S(s)/N(s)$$

- When number of episodes approaches infinity,

  V(s) converges $V(s) \rightarrow V_\pi(s)$

Empirical mean approaches true mean.

Incremental and running mean

- *S(s)* does not need to be stored

$$V(s) = V(s) + \frac{1}{N(s)}\left(G_t - V(s)\right)$$

- We can also track a running mean

$$V(s) = (1-\alpha)V(s) + \alpha G_t = V(s) + \alpha\left(G_t - V(s)\right)$$

Adjust estimate
toward observation.

Can we do without episodes?

Temporal difference (TD) – learning without episodes

- For each state transition, update estimate towards another estimate:

$$V(s_t) = V(s_t) + \alpha\left(\underline{r_{t+1} + \gamma V(s_{t+1})} - V(s_t)\right)$$

- Approach called TD(0)

Estimated return.

- Compare to MC

$$V(s_t) = V(s_t) + \alpha\left(G_t - V(s_t)\right)$$

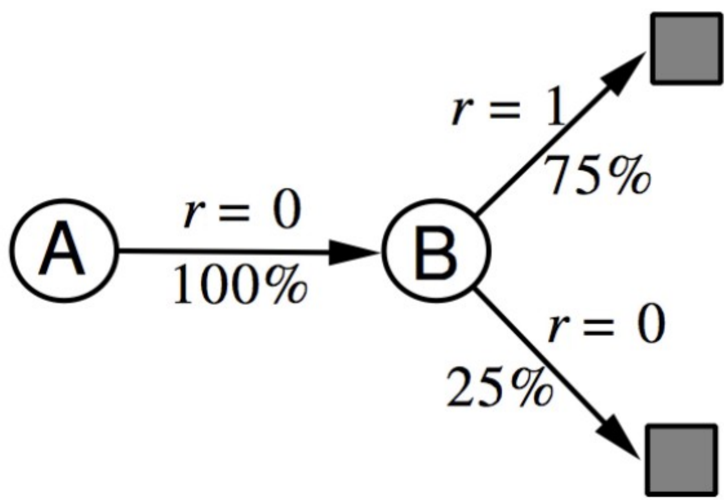
True return.

What if we have limited data?

Batch learning

- For limited number of trials available:
  - Sample episode *k.*
  - Apply MC or TD(0) to episode *k.*

A, 0, B, 0
B, 1
B, 1
B, 1
B, 1
B, 1
B, 1
B, 0



What is *V(A)*?

Beware small sample numbers.

MC vs TD

- MC
  - Needs full episodes. Only works in episodic environments.
  - High variance, zero bias → good but slow convergence.
  - Does not exploit Markov property → often better in non-Markov env.

- TD (esp. TD(0))
  - Can learn from incomplete episodes and on-line after each step.
  - Works in continuing environments.
  - Low variance, some bias → often more efficient than MC, discrete state TD(0) converges, more sensitive to initial value.
  - Exploits Markov property → often more efficient in Markov env.

Can we combine these / find intermediate solution?

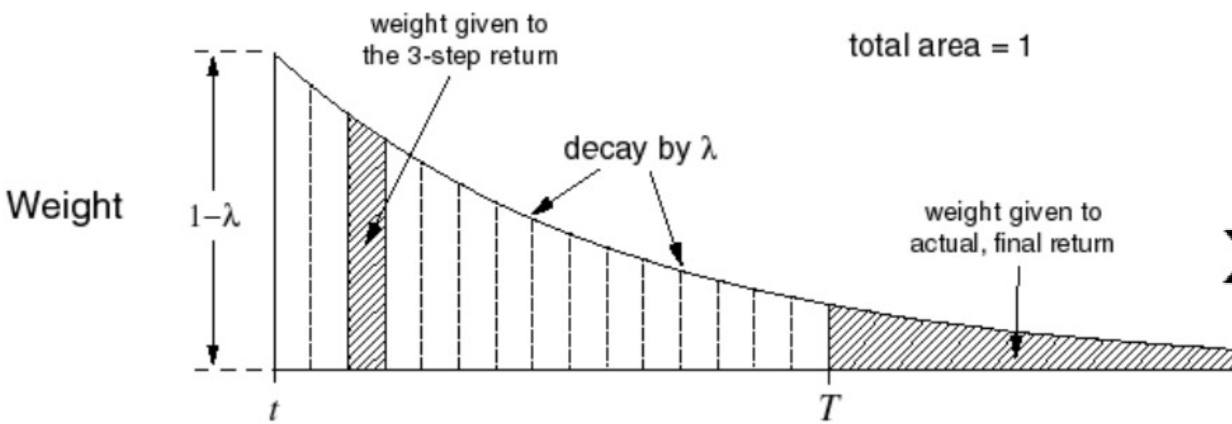
λ-return

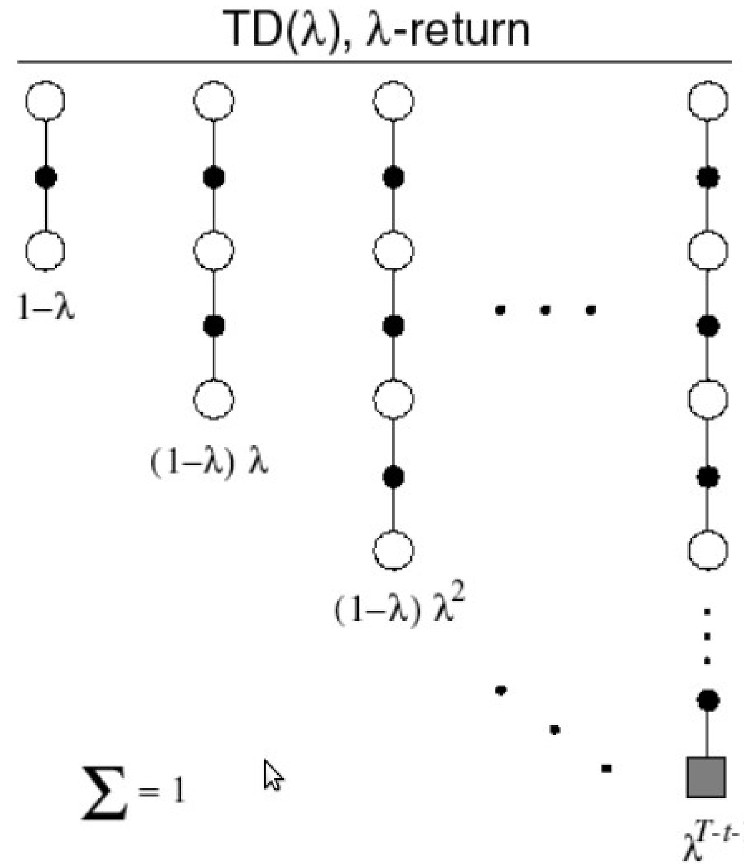
k-step return: $G_t^{(k)} = \sum_{i=1}^{k} \gamma^{i-1} r_{t+i} + \gamma^k V(s_{t+k})$

- Combine returns in different horizons.

$$G_t^{\lambda} = (1-\lambda) \sum_{k=1}^{\infty} \lambda^{k-1} G_t^{(k)}$$

$$V(s_t) = V(s_t) + \alpha \left(G_t^{\lambda} - V(s_t) \right)$$





Aalto University
School of Electrical
Engineering

Requires complete episodes! Can we survive without?
First: an alternative viewpoint!

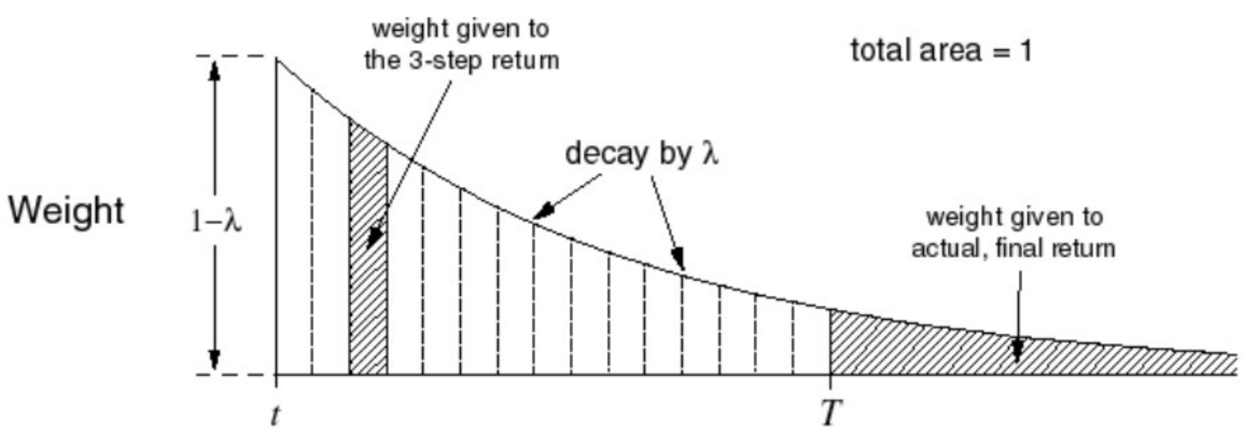
Causes and effects – eligibility traces

- Which state is the “cause” of a reward?
- Frequency heuristic: most frequent states likely.
- Recency heuristic: most recent states likely.
- *Eligibility trace* for a state combines these:

$$E_t(s) = \gamma \lambda E_{t-1}(s) + \mathbf{1}(s_t = s)$$

“How often a particular state was visited recently on average?”

Not exactly, also considers discount.



Backward-TD(λ)

- Extend TD time horizon with decay (λ).

Note: all states are updated after each step, not only the "current" state

- After episode, update

$$V(s) = V(s) + \alpha E_t(s)\left(r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\right)$$

- TD(1) equal to MC.

What if $\lambda = 0$

$$E_t(s) = \gamma \lambda E_{t-1}(s) + \mathbf{1}(s_t = s)$$

- Eligibility traces way to implement *backward* TD(λ), *forward* TD(λ) requires episodes.

Slightly different in on-line case.

Control / decision making?

- So far we only found out how to estimate value functions for a particular policy.

- Can we use this to optimize a policy?

Policy improvement and policy iteration

- Given a policy π, it can be improved by
  - Evaluating its value function
  - Forming a new policy by acting greedily with respect to the value function

- This always improves the policy.

- Iterating multiple times called *policy* iteration.
  - Converges to optimal policy.

Monte-Carlo Policy iteration

- Can we choose action using value function *V(s)*?

- Greedy policy improvement using action-value function *Q(s,a)* does not require model.

$$\pi'(s) = arg\,max_u\, Q(s,a)$$

- Estimate *Q(s,a)* using MC (empirical mean = "calculate average").

Note: calculate frequencies for all state-action pairs.

Exploration-exploitation trade-off:
How can we ensure that we try different actions?

Ensuring exploration

- Simple approach: ε-greedy exploration:
  - Explore: Choose action at random with probability ε.
  - Exploit: Be greedy with probability 1-ε.

$$\pi(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon & \text{if } a = \arg\max_{a'} Q(s, a') \\ \epsilon/m & \text{for any other action} \end{cases}$$

Number of different actions

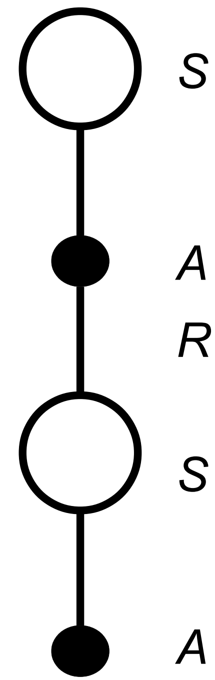
- How to converge to optimal policy?
  - Idea: reduce ε over time.
  - For example, for *k*:th episode $\epsilon = \frac{b}{b+k}$

"Greedy in Limit with Infinite Exploration" (GLIE)

constant

Can we use TD instead of MC for control?

SARSA



- Idea: Apply TD to *Q(S,A)*.
  - With ε-greedy policy improvement.
  - Update each time step.

$$Q(s,a)=Q(s,a)+\alpha\left(r+\gamma Q(s',a')-Q(s,a)\right)$$

Compare with

$$V(s_t)=V(s_t)+\alpha\left(r_{t+1}+\gamma V(s_{t+1})-V(s_t)\right)$$

- SARSA converges under
  - GLIE policy,
  - $\sum_{t=0}^{\infty}\alpha_t=\infty \quad \sum_{t=0}^{\infty}\alpha_t^2<\infty$

For example $\alpha_t=1/t$

SARSA(λ)

- Instead of TD(0) update in SARSA, use TD(λ) update.
- Backward SARSA(λ)

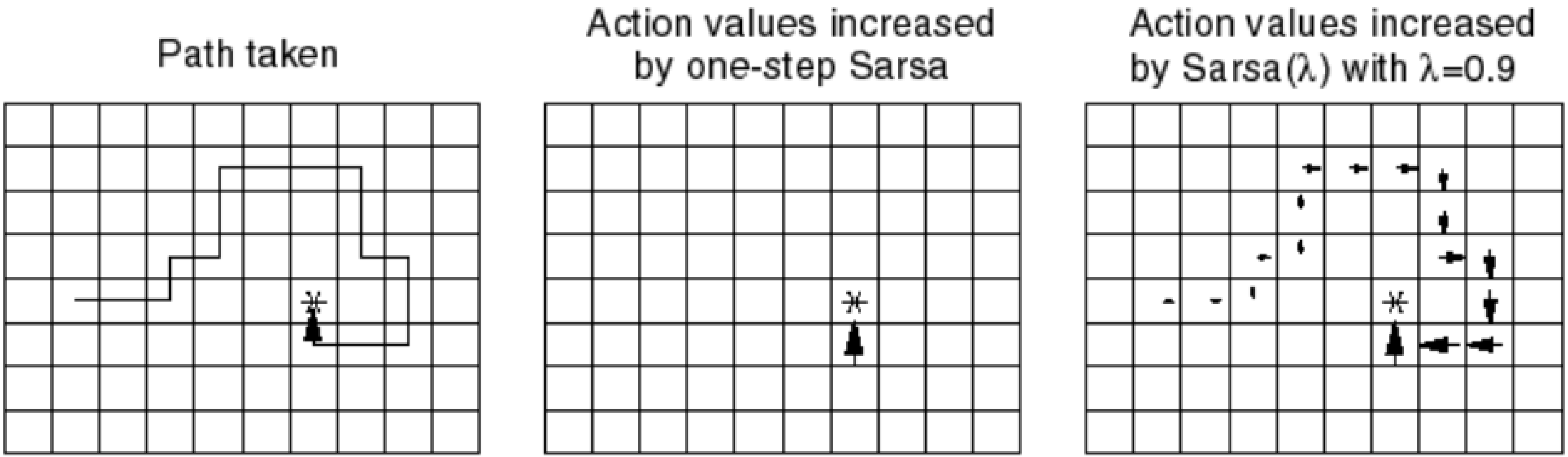
$$E_t(s,a) = \gamma \lambda E_{t-1}(s,a) + \mathbf{1}(s_t = s, a_t = a)$$

$$Q(s,a) = Q(s,a) + \alpha E_t(s,a)\left(r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right)$$

Compare to

$$E_t(s) = \gamma \lambda E_{t-1}(s) + \mathbf{1}(s_t = s)$$

$$V(s) = V(s) + \alpha E_t(s)\left(r_{t+1} + \gamma V(s_{t+1}) - V(s_t)\right)$$


Path taken
Action values increased by one-step Sarsa
Action values increased by Sarsa(λ) with λ=0.9

On-policy vs off-policy learning

- *On-policy learning* (methods so far)
  - Use a policy while learning how to optimize it.
  - “Learn on the job”.

- *Off-policy learning*
  - Use another policy while learning about optimal policy.
  - Can learn from observation of other agents.
  - Can learn about optimal policy when using exploratory policy.

Q-learning

- Use ε-greedy *behavior policy* to choose actions.
- *Target policy* is greedy with respect to *Q*.

$$\pi(s) = \arg\max_a Q(s, a)$$

- Update target policy greedily:

$$Q(s,a) = Q(s,a) + \alpha\left(r + \gamma \max_{a'} Q(s', a') - Q(s,a)\right)$$

- *Q* converges to *Q**.

Assume we take greedy action at next step.

Summary

- In reinforcement learning, dynamics and reward function of MDP are unknown.

- MC approaches sample returns from full episodes.
- TD approaches sample estimated returns (biased).

Next: Extending state spaces

- What to do if
  - discrete state space is too large?
  - state space is continuous?

- Readings
  - Sutton & Barto, ch. 9-9.3, 10-10.1