Reinforcement Learning (INF11010)

Lecture 9: TD Control

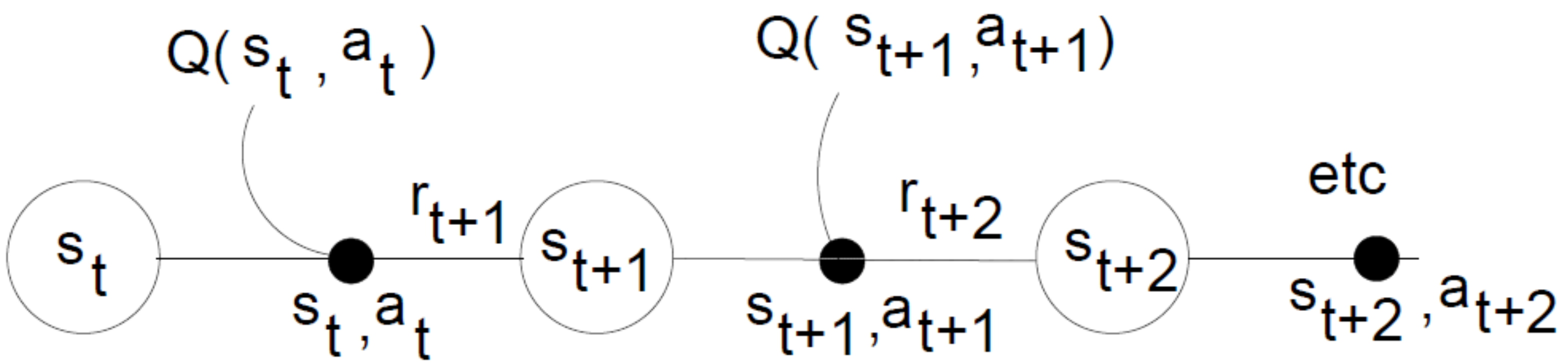
Pavlos Andreadis, February 16th 2018
with slides by Subramanian Ramamoorthy, 2017

Today's Content

- Temporal Difference Learning – Control
  - Sarsa
  - Q-Learning

TD for *Control*: Learning Q-Values

Learn action values $Q^{\pi}(s, a)$ for the policy π



SARSA update rule:

$$\Delta Q_t(s_t, a_t) = \alpha[r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)]$$

TD for Control: Learning Q-Values

- Choose a behaviour policy π and estimate the Q-values (Q^π) using the SARSA update rule. Change π towards greediness wrt Q^π.
- Use ϵ-greedy or ϵ-soft policies.
- Converges with probability 1 to optimal policy and Q-values if visit all state-action pairs infinitely many times and policy converges to greedy policy, e.g. by arranging for ϵ to tend towards 0.

Remember: learning optimal Q-values is useful since it tells us immediately which is(are) the optimal action(s) – they have the highest Q-value

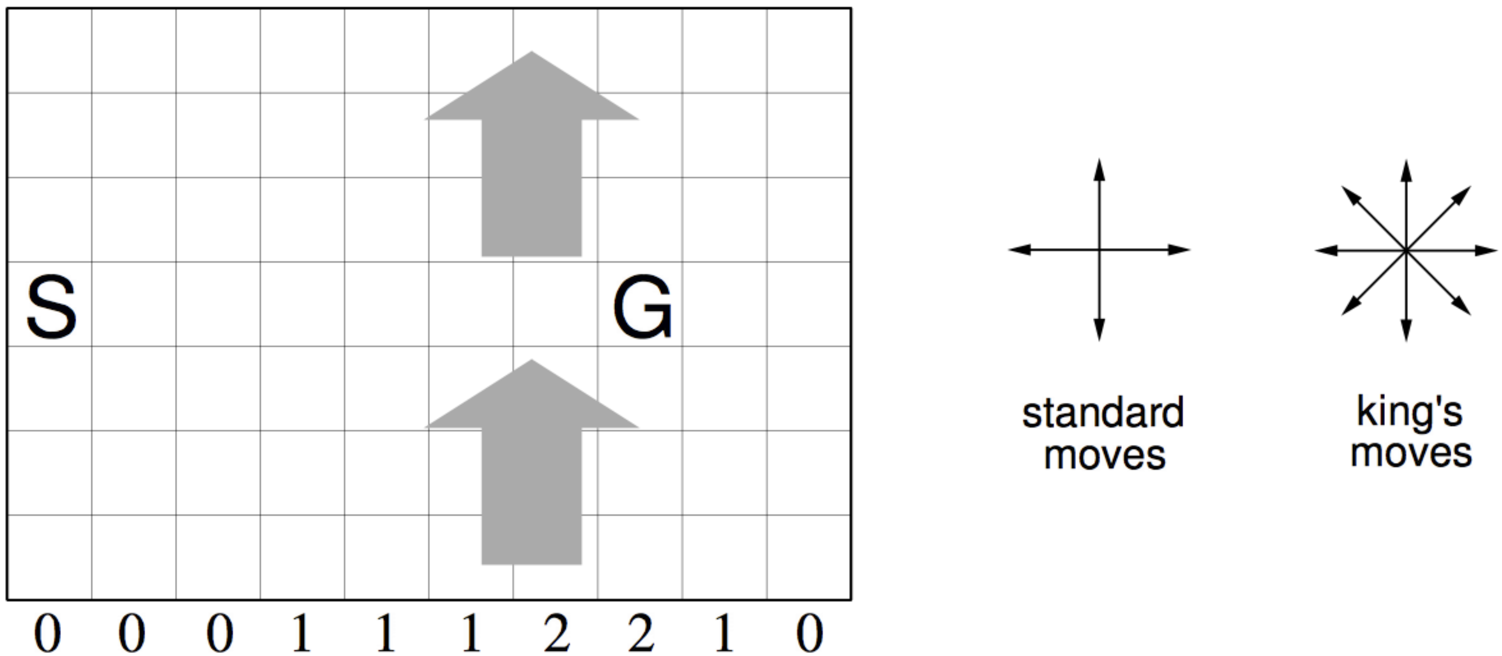
Algorithm: SARSA

- Initialise $Q(s, a)$
- Repeat `many times`
  - Pick s, a
  - Repeat `each step to goal`
    - Do a, observe r, s'
    - Choose a' based on $Q(s', a')$ ϵ`-greedy`
    - $Q(s, a) = Q(s, a) + \alpha[r + \gamma Q(s', a') - Q(s, a)]$
    - $s = s'$, $a = a'$
  - Until s terminal (where $Q(s', a') = 0$)

Use with policy iteration, i.e. change policy each time to be greedy wrt current estimate of Q

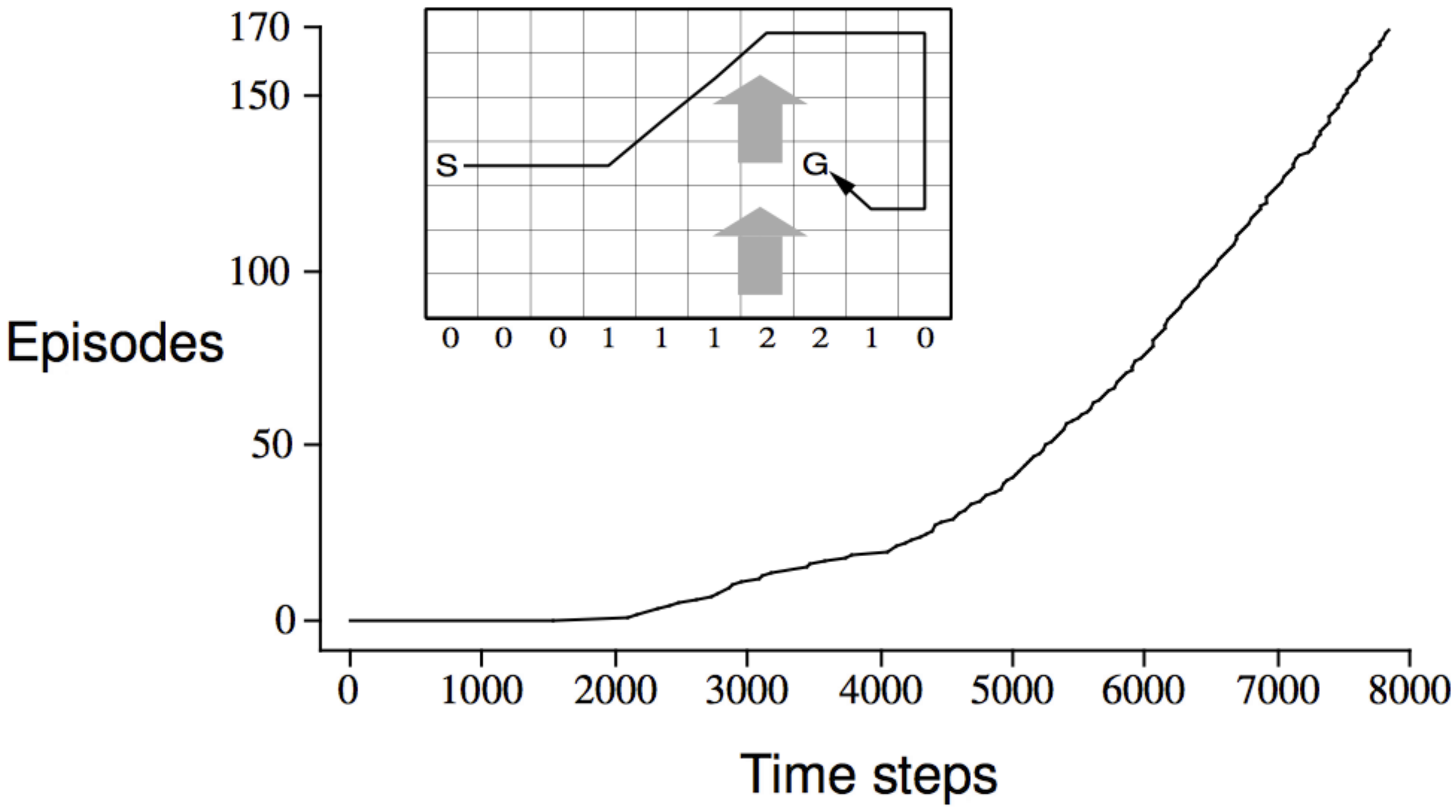
Example: windy gridworld, S+B sect. 6.4

Windy Gridworld



undiscounted, episodic, reward = –1 until goal

Results of Sarsa on the Windy Gridworld



Q-Learning

SARSA is an example of **on-policy** learning. Why?

Q-LEARNING is an example of **off-policy** learning

Update rule:

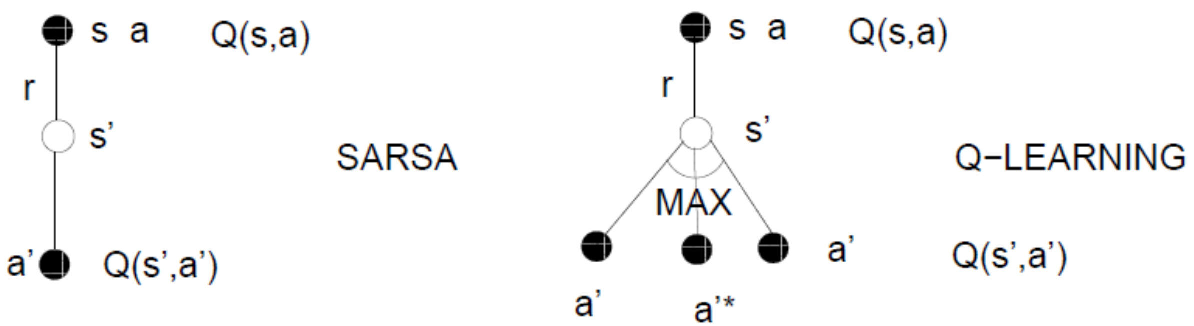
$$\Delta Q_t(s_t, a_t) \quad = \quad \alpha[r_{t+1} + \boxed{\gamma \max_a Q_t(s_{t+1}, a)} - Q_t(s_t, a_t)]$$

Always update using *maximum* Q value available from next state: then $Q \Rightarrow Q*$, optimal action-value function

Algorithm: Q-Learning

- Initialise $Q(s, a)$
- Repeat `many times`
  - Pick s `start state`
  - Repeat `each step to goal`
    - Choose a based on $Q(s, a)$ ϵ`-greedy`
    - Do a, observe r, s'
    - $Q(s, a) = Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)]$
    - $s = s'$
  - Until s terminal

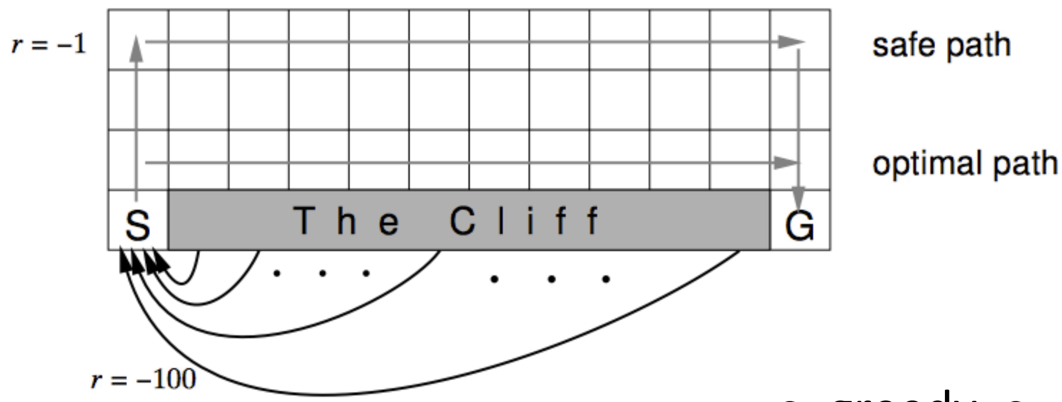
Backup Diagrams: SARSA and Q-Learning



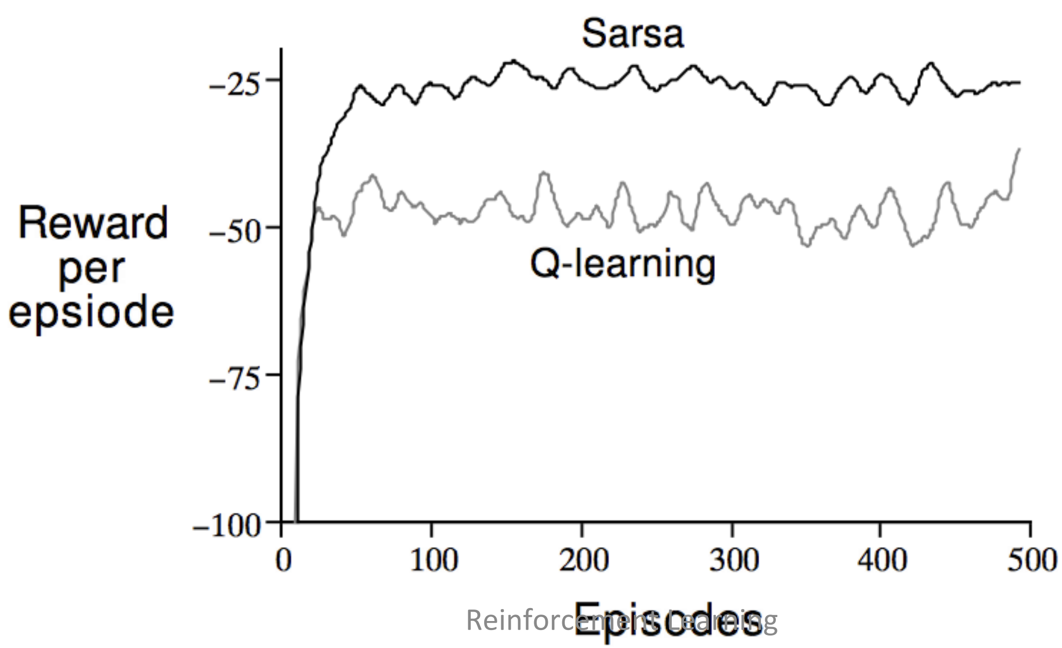
SARSA backs up using the action a' actually chosen by the behaviour policy.

Q-LEARNING backs up using the Q-value of the action a'^* that is the *best* next action, i.e. the one with the highest Q value, $Q(s', a'^*)$. The action actually chosen by the behaviour policy *and followed* is not necessarily a'^*

Cliffwalking



ε−greedy, $\varepsilon = 0.1$



Q-Learning vs. SARSA

QL: $Q(s,a) = Q(s,a) + \alpha[r + \gamma \max_{a'} Q(s',a') - Q(s,a)]$ off-policy

SARSA: $Q(s,a) = Q(s,a) + \alpha[r + \gamma Q(s',a') - Q(s,a)]$ on-policy

In the cliff-walking task:

QL: learns optimal policy along edge

SARSA: learns a safe non-optimal policy away from edge

ϵ-greedy algorithm

For $\epsilon \neq 0$ **SARSA** performs better online. Why?

For $\epsilon \to 0$ gradually, both converge to optimal.

Summary

- Idea of Temporal Difference Prediction
- 1-step tabular model-free TD method
- Can extend to the GPI approach:
  - On-policy: SARSA
  - Off-policy: Q-learning
- TD methods bootstrap and sample, combining benefits of DP and MC methods

Reading +

- Chapter 6 (6.4 to 6.5) of Sutton and Barto (1st Edition) http://incompleteideas.net/book/ebook/the-book.html