Reinforcement Learning

Temporal-Difference Learning (RLbook 6)

Christopher Simpkins

Kennesaw State University

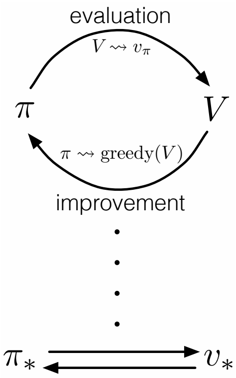
Temporal-Difference Learning

- DP Review
  - Dynamic programming
  - Generalized policy iteration (GPI)
- Model-free control
  - Monte Carlo control
  - Temporal-difference learning

Dynamic Programming

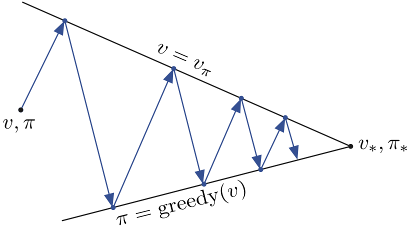
Dynamic programming algorithms, as well as RL algorithms in general, contain two phases (combined in value iteration):

- Prediction: estimate the value function
- Control: computing or approximating optimal policies



Generalized Policy Iteration

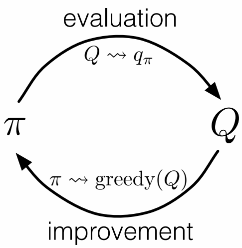
In policy iteration we sweep entire state space in each step.



In generalized policy iteration (GPI) we interleave prediction and control at arbitrary granularity.

- As long as we visit every state, still assured of convergence.

Monte Carlo Control



Temporal-Difference Learning

Combination of dynamic programming and Monte Carlo ideas.

- Like Monte Carlo, learn directly from experience without a model of the environment.
- Like dynamic programming, update estimates based in part on other learned estimates, without waiting for a final outcome (bootstrap).

TD Prediction

Use experience following a policy to update estimate of V, namely v_{pi}.

Monte Carlo methods wait until the return following the visit is known, then use that return as a target for $V(S_t)$:

$$V(S_t) \leftarrow V(S_t) + \alpha \left[G_t - V(S_t) \right] \tag{6.1}$$

TD methods only until the next time step. At time $t+1$ they immediately form a target and make a useful update using the observed reward R_{t+1} and the estimate $V(S_{t+1})$

A simple TD method makes the update:

$$V(S_t) \leftarrow V(S_t) + \alpha \left[R_{t+1} + V(S_{t+1}) - \gamma V(S_t) \right]$$

immediate on transition to S_{t+1} and receiving R_{t+1}.

- Target for Monte Carlo update is G_t.
- Target for TD update is $R_{t+1} + \gamma V(S_{t+1})$

Tabular TD(0) Algorithm

One-step TD is called $TD(0)$, which is a special case of $TD(\lambda)$ and n-step methods.

Tabular TD(0) for estimating v_π

Input: the policy π to be evaluated
Algorithm parameter: step size $\alpha \in (0, 1]$
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(\mathit{terminal}) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    $A \leftarrow$ action given by π for S
    Take action A, observe R, S'
    $V(S) \leftarrow V(S) + \alpha\big[R + \gamma V(S') - V(S)\big]$
    $S \leftarrow S'$
  until S is terminal

TD Error

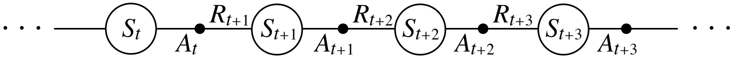
Recall TD update:

$$V(S_t) \leftarrow V(S_t) + \alpha \left[R_{t+1} + V(S_{t+1}) - \gamma V(S_t)\right]$$

The quantity in brackets is called *TD error* – the difference in the estimate value of S at time t and $t+1$:

$$\delta \doteq R_{t+1} + V(S_{t+1}) - \gamma V(S_t)$$

Sarsa: On-policy TD Control



Sarsa update:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(S_t, A_t)\right]$$

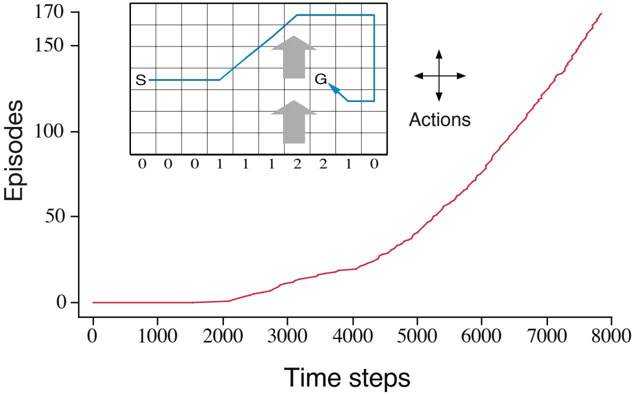
Sarsa Algorithm

Sarsa (on-policy TD control) for estimating $Q \approx q_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(\mathit{terminal}, \cdot) = 0$

Loop for each episode:
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  Loop for each step of episode:
    Take action A, observe R, S'
    Choose A' from S' using policy derived from Q (e.g., ε-greedy)
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma Q(S', A') - Q(S, A)\big]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

Example: Windy Grid World



Q-Learning: Off-policy TD Control

Sarsa update:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(S_t, A_t)\right]$$

Q-learning update:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[R_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(S_t, A_t)\right]$$

Q-learning is off-policy because the value update is made using $\max_a$ rather than the a recommended by the policy being followed.

Q-Learning Algorithm

Q-learning (off-policy TD control) for estimating $\pi \approx \pi_*$

Algorithm parameters: step size $\alpha \in (0, 1]$, small $\varepsilon > 0$
Initialize $Q(s, a)$, for all $s \in \mathcal{S}^+, a \in \mathcal{A}(s)$, arbitrarily except that $Q(\mathit{terminal}, \cdot) = 0$

Loop for each episode:
  Initialize S
  Loop for each step of episode:
    Choose A from S using policy derived from Q (e.g., ε-greedy)
    Take action A, observe R, S'
    $Q(S, A) \leftarrow Q(S, A) + \alpha\big[R + \gamma \max_a Q(S', a) - Q(S, A)\big]$
    $S \leftarrow S'$
  until S is terminal