7.5 Sarsa(λ)

How can eligibility traces be used not just for prediction, as in TD(λ), but for control? As usual, the main idea of one popular approach is simply to learn action values, $Q_t(s,a)$, rather than state values, $V_t(s)$. In this section we show how eligibility traces can be combined with Sarsa in a straightforward way to produce an on-policy TD control method. The eligibility trace version of Sarsa we call *Sarsa*(λ), and the original version presented in the previous chapter we henceforth call *one-step Sarsa*.

The idea in Sarsa(λ) is to apply the TD(λ) prediction method to state-action pairs rather than to states. Obviously, then, we need a trace not just for each state, but for each state-action pair. Let $e_t(s,a)$ denote the trace for state-action pair s, a. Otherwise the method is just like TD(λ), substituting state-action variables for state variables--$Q(s,a)$ for $V(s)$ and $e_t(s,a)$ for $e_t(s)$:

$$Q_{t+1}(s,a) = Q_t(s,a) + \alpha \delta_t e_t(s,a), \qquad \text{for all } s, a$$

where

$$\delta_t = r_{t+1} + \gamma Q_t(s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)$$

and

$$e_t(s,a) = \begin{cases} \gamma\lambda e_{t-1}(s,a) + 1 & \text{if } s = s_t \text{ and } a = a_t; \\ \gamma\lambda e_{t-1}(s,a) & \text{otherwise.} \end{cases} \qquad \text{for all } s, a \tag{7.13}$$

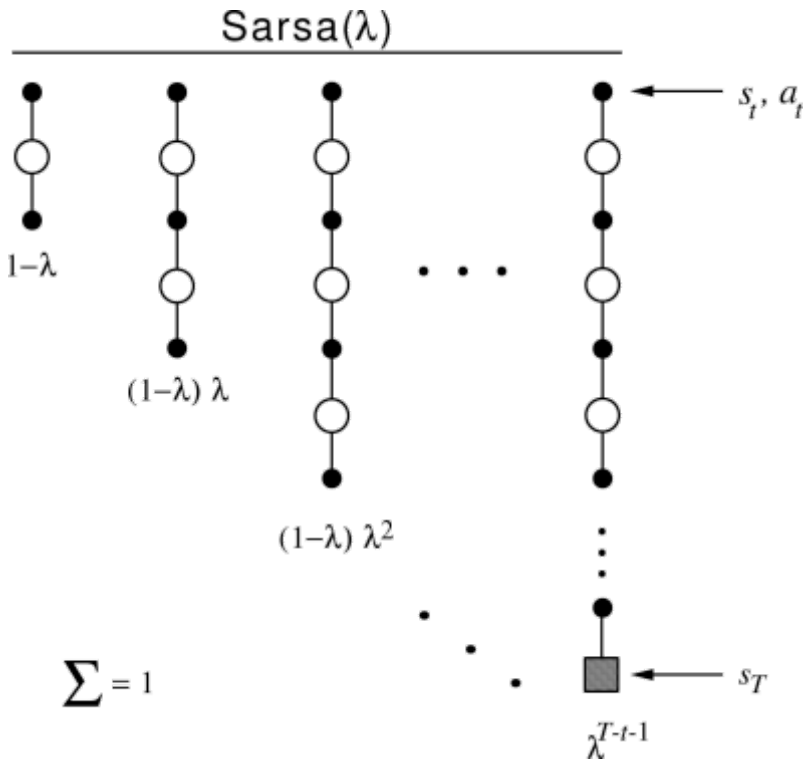

Figure 7.10: Sarsa(λ)'s backup diagram.

Figure 7.10 shows the backup diagram for Sarsa(λ). Notice the similarity to the diagram of the TD(λ) algorithm (Figure 7.3). The first backup looks ahead one full step, to the next state-action pair, the second looks ahead two steps, and so on. A final backup is based on the complete return. The weighting of each backup is just as in TD(λ) and the λ-return algorithm.

One-step Sarsa and Sarsa(λ) are on-policy algorithms, meaning that they approximate $Q^\pi(s,a)$, the action values for the current policy, π, then improve the policy gradually based on the approximate values for the current policy. The policy improvement can be done in many different ways, as we have seen throughout this book. For example, the simplest approach is to use the ε-greedy policy with respect to the current action-value estimates. Figure 7.11 shows the complete Sarsa(λ) algorithm for this case.

Initialize $Q(s,a)$ arbitrarily and $e(s,a) = 0$, for all s,a
Repeat (for each episode):
  Initialize s, a
  Repeat (for each step of episode):
    Take action a, observe r, s'
    Choose a' from s' using policy derived from Q (e.g., ε-greedy)
    $\delta \leftarrow r + \gamma Q(s',a') - Q(s,a)$
    $e(s,a) \leftarrow e(s,a) + 1$
    For all s,a:
      $Q(s,a) \leftarrow Q(s,a) + \alpha\delta e(s,a)$
      $e(s,a) \leftarrow \gamma\lambda e(s,a)$
    $s \leftarrow s'$; $a \leftarrow a'$
  until s is terminal

Figure 7.11: Tabular Sarsa(λ).

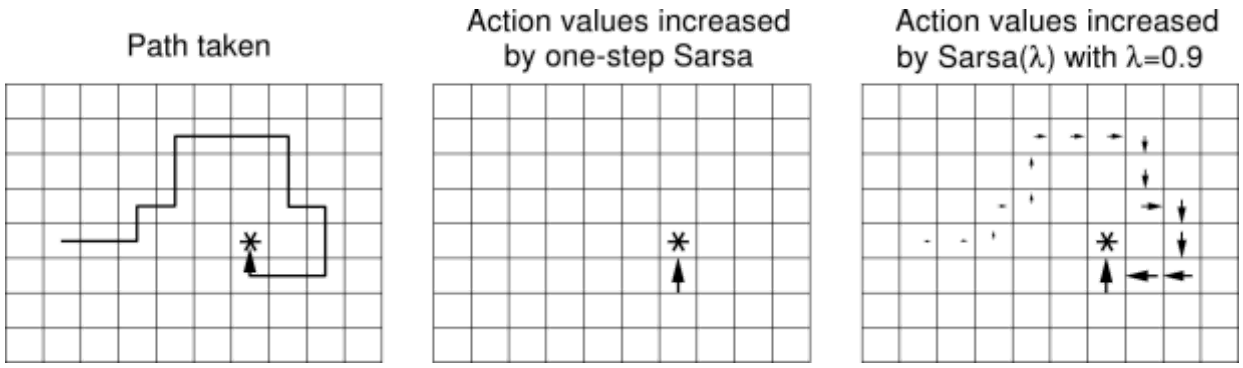

Figure 7.12: Gridworld example of the speedup of policy learning due to the use of eligibility traces.

Example 7.4: Traces in Gridworld The use of eligibility traces can substantially increase the efficiency of control algorithms. The reason for this is illustrated by the gridworld example in Figure 7.12. The first panel shows the path taken by an agent in a single episode, ending at a location of high reward, marked by the *. In this example the values were all initially 0, and all rewards were zero except for a positive reward at the * location. The arrows in the other two panels show which action values were strengthened as a result of this path by one-step Sarsa and Sarsa(λ) methods. The one-step method strengthens only the last action of the sequence of actions that led to the high reward, whereas the trace method strengthens many actions of the sequence. The degree of strengthening (indicated by the size of the arrows) falls off (according to $\gamma \lambda$) with steps from the reward. In this example, $\gamma = 1$ and $\lambda = 0.9$.