Reinforcement Learning

Temporal-Difference Learning

Stefano Albrecht, Pavlos Andreadis

31 January 2020

THE UNIVERSITY of EDINBURGH informatics

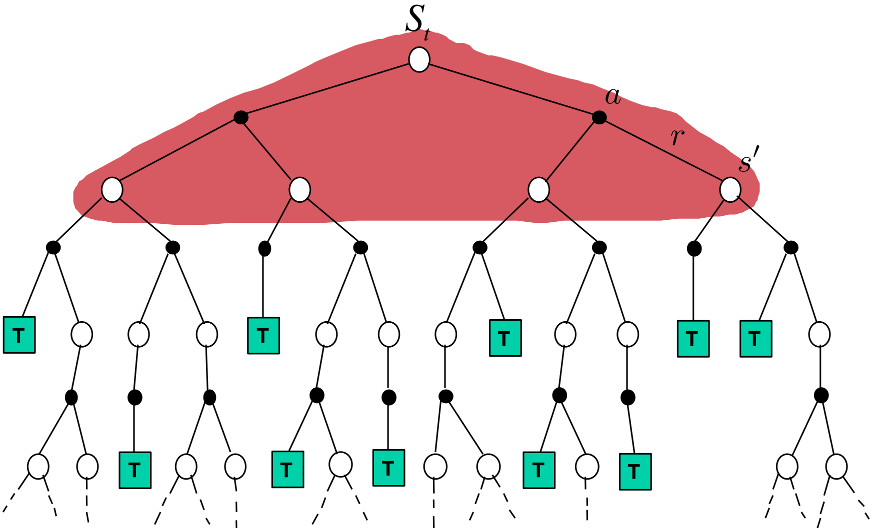
Lecture Outline

- Temporal-difference (TD) policy evaluation
- TD control:
  - Sarsa
  - Q-learning
  - Expected Sarsa
- n-step TD methods

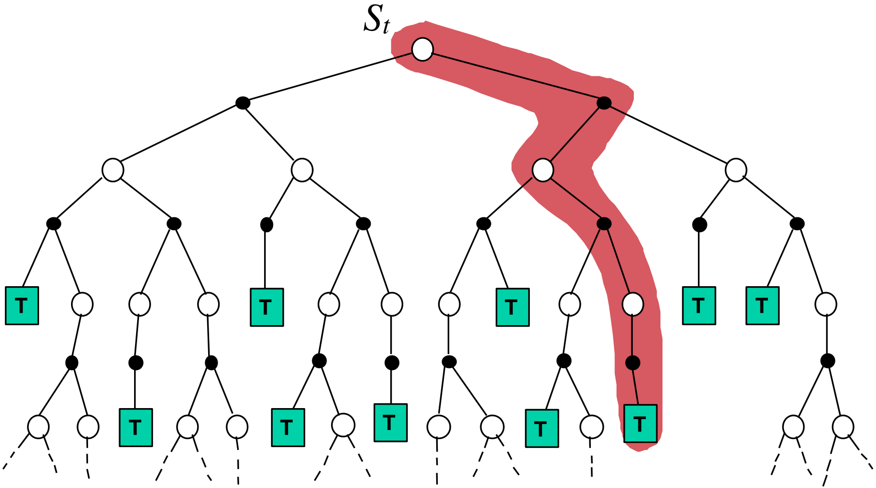
Method Comparison

Method	Model-free?	Bootstrap?
Dynamic Programming	No	Yes
Monte Carlo	Yes	No
Temporal-Difference	Yes	Yes

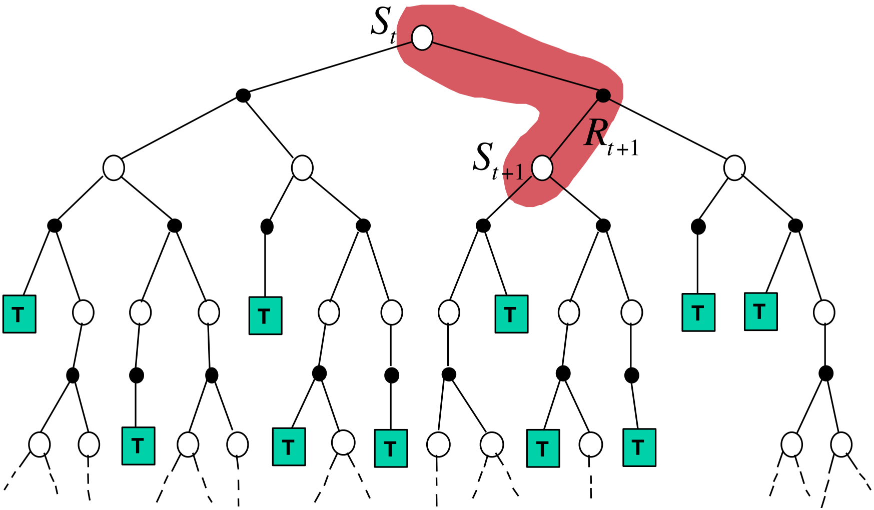
Recap: Dynamic Programming



Recap: Monte Carlo Methods



Now: Temporal-Difference Learning



Temporal-Difference Policy Evaluation

General iterative update rule:

$$\text{NewEstimate} \leftarrow \text{OldEstimate} + \text{StepSize}\,[\,\text{Target} - \text{OldEstimate}\,]$$

MC update:

$$V(S_t) \leftarrow V(S_t) + \alpha\,[\,G_t - V(S_t)\,]$$

Note:

$$\begin{aligned} v_\pi(s) &\doteq \mathbb{E}_\pi[G_t | S_t = s] \\ &= \mathbb{E}_\pi[R_{t+1} + \gamma G_{t+1} | S_t = s] \\ &= \mathbb{E}_\pi[\underbrace{R_{t+1} + \gamma v_\pi(S_{t+1})}_{\text{Use as target}} | S_t = s] \end{aligned}$$

Temporal-Difference Policy Evaluation

General iterative update rule:

$$\text{NewEstimate} \leftarrow \text{OldEstimate} + \text{StepSize}\,[\,\text{Target} - \text{OldEstimate}\,]$$

MC update:

$$V(S_t) \leftarrow V(S_t) + \alpha\,[\,G_t - V(S_t)\,]$$

TD(0) update:

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

TD(0) for Policy Evaluation

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 $V(\mathit{terminal}) = 0$

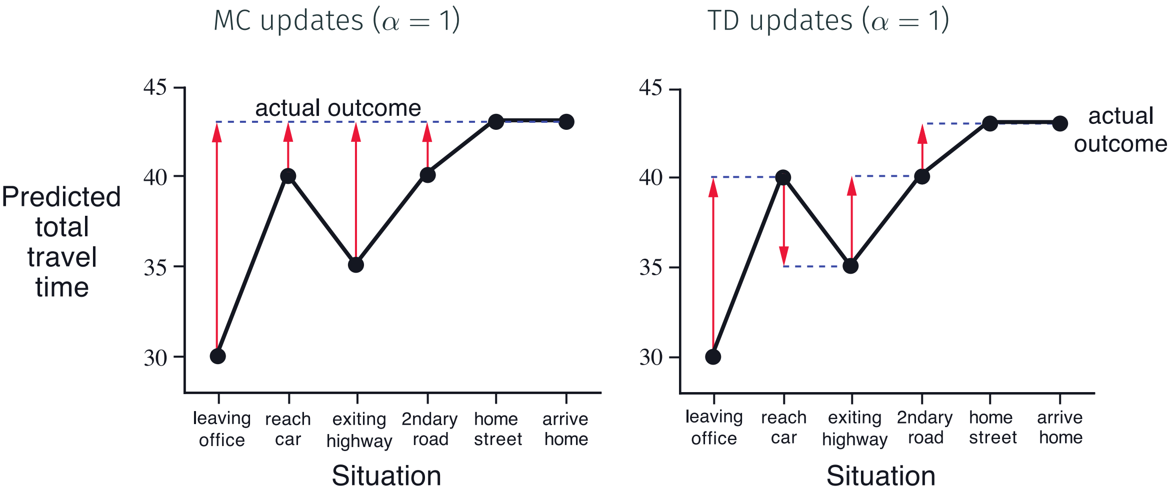
Loop for each episode:
    Initialize S
    Loop for each step of episode:
        $A \leftarrow$ action given by π for S
        Take action A, observe R, S'
        $V(S) \leftarrow V(S) + \alpha\big[R + \gamma V(S') - V(S)\big]$
        $S \leftarrow S'$
    until S is terminal

Example: Driving Home

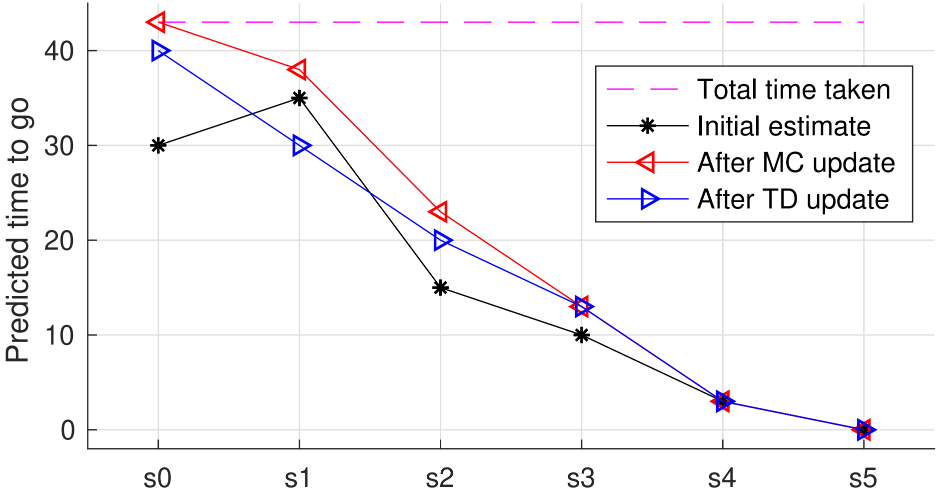
$\gamma = 1$

		R_t	$v(S_t)$	$v(S_0)$
	State	*Elapsed Time (minutes)*	*Predicted Time to Go*	*Predicted Total Time*
S_0	leaving office, friday at 6	0	30	30
S_1	reach car, raining	5	35	40
S_2	exiting highway	20	15	35
S_3	2ndary road, behind truck	30	10	40
S_4	entering home street	40	3	43
S_5	arrive home	43	0	43

Example: Driving Home



Example: Driving Home (Extra)



Convergence of TD(0)

TD(0) converges to v_π with prob. 1 if:

- all states visited infinitely often

  and

- standard stochastic approximation conditions (α-reduction)

$$\forall s : \quad \sum_{t:S_t=s} \alpha_t \to \infty \quad \text{and} \quad \sum_{t:S_t=s} \alpha_t^2 < \infty$$

Convergence of TD(0)

Intuition: what is TD(0) update on *expectation*?

$$
\begin{aligned}
V(S_t) &\leftarrow \mathbb{E}_\pi[(1-\alpha)V(S_t) + \alpha\,[R_{t+1} + \gamma V(S_{t+1})]] && \text{(rewrite)} \\
&= (1-\alpha)V(S_t) + \alpha \mathbb{E}_\pi[R_{t+1} + \gamma V(S_{t+1})] \\
&= (1-\alpha)V(S_t) + \alpha \sum_a \pi(a|S_t) \sum_{s',r} p(s', r|S_t, a)\,\left[r + \gamma V(s')\right] \\
&= (1-\alpha)V(S_t) + \alpha\, v_\pi(S_t)
\end{aligned}
$$

Bellman operator $v_\pi(S_t)$ is contraction mapping with fixed point v_π!

- Expected TD update moves $V(S_t)$ toward $v_\pi(S_t)$ by α
- α used to control averaging in sampling updates

Advantages of TD Learning

- Like MC: TD does not require full model $p(s', r|s, a)$, only *experience*

- Unlike MC: TD can be fully *incremental*

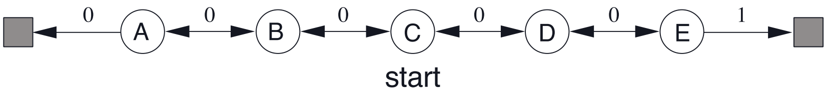
  $\Rightarrow$ Learn *before* final return is known

  $\Rightarrow$ Less memory and computation

- Both MC and TD converge to v_π/q_π under certain assumptions

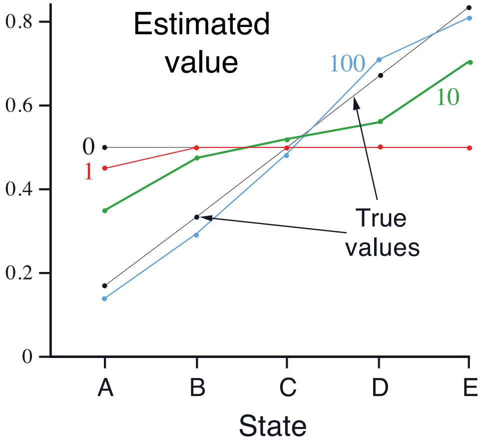
  $\Rightarrow$ But TD often faster in practice

Example: Random Walk

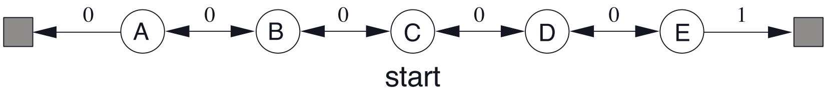


$\pi(left/right|s) = 0.5$

Values learned by TD(0) after 0/1/10/100 episodes ($\alpha = 0.1$)



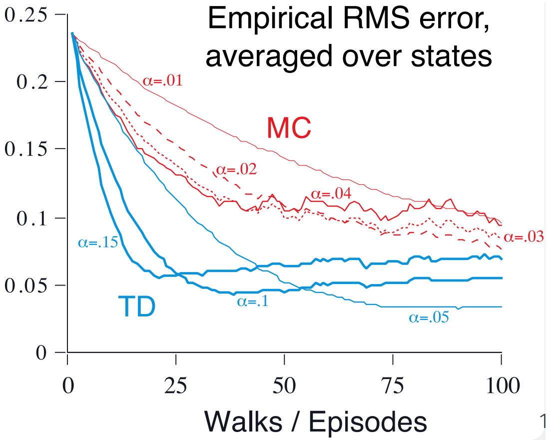
Example: Random Walk



$\pi(left/right|s) = 0.5$

Root mean-squared error averaged over all states and 100 episodes

TD methods usually learn faster than MC



On-Policy TD Control: Sarsa

On-policy: learn q_π and improve π while following π

Sarsa:

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

- If S_{t+1} terminal state, define $Q(S_{t+1}, A_{t+1}) = 0$
- Ensure exploration by using ϵ-soft policy π

On-Policy TD Control: Sarsa

On-policy: learn q_π and improve π while following π

Sarsa:

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

- If S_{t+1} terminal state, define $Q(S_{t+1}, A_{t+1}) = 0$
- Ensure exploration by using ϵ-soft policy π

Converges to π_* with prob 1. if all (s, a) infinitely visited and standard α-reduction

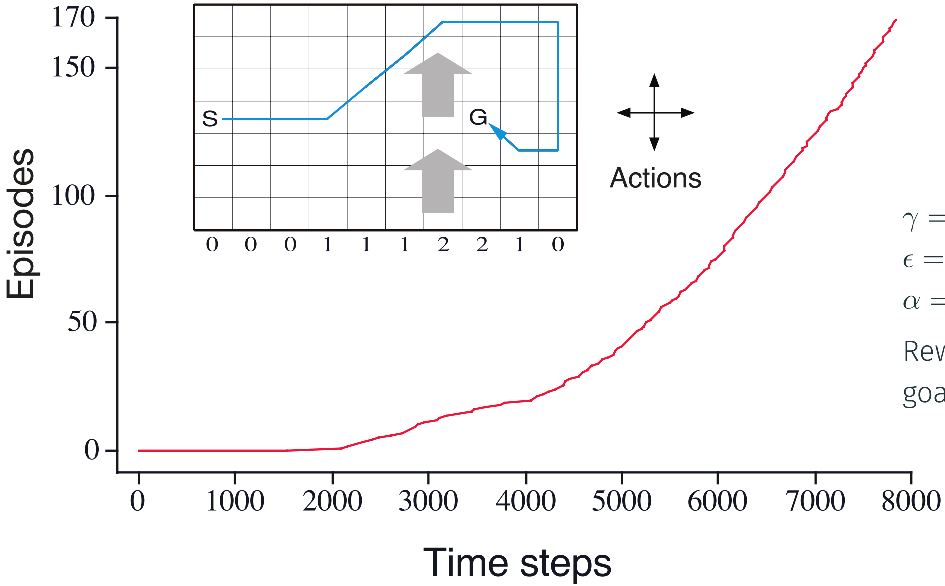
$$\forall s, a : \quad \sum_{t:S_t=s,A_t=a} \alpha_t \to \infty, \quad \sum_{t:S_t=s,A_t=a} \alpha_t^2 < \infty$$

and ϵ gradually goes to 0

On-Policy TD Control: Sarsa

Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
  Initialize S
  Choose A from S using policy derived from Q (e.g., ε-greedy)
  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)
    $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma Q(S',A') - Q(S,A)]$
    $S \leftarrow S'$; $A \leftarrow A'$;
  until S is terminal

Example: Windy Gridworld with Sarsa



$\gamma = 1.0$

$\epsilon = 0.1$

$\alpha = 0.5$

Reward -1 until goal reached

Off-Policy TD Control: Q-Learning

Off-policy: Learn q_π and improve π while following μ

Q-learning:

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

Why is there no importance sampling ratio?

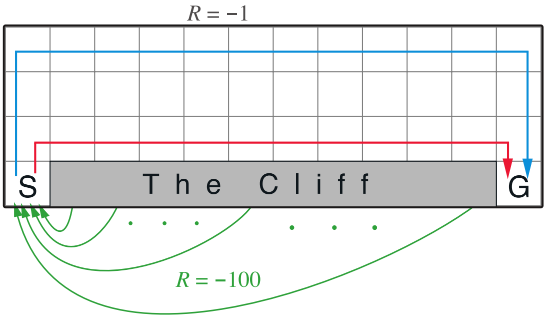
- Recall: for q_π, ratio defined as $\prod_{k=t+1}^{T-1} \pi(A_k|S_k)/\mu(A_k|S_k)$
- Because a in $q_\pi(s, a)$ is no random variable

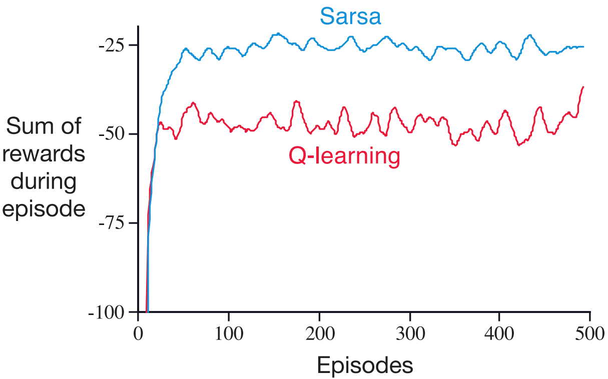
Converges to π_* with prob. 1 if all (s, a) infinitely visited and standard α-reduction

Off-Policy TD Control: Q-Learning

Initialize $Q(s,a), \forall s \in \mathcal{S}, a \in \mathcal{A}(s)$, arbitrarily, and $Q(\textit{terminal-state}, \cdot) = 0$
Repeat (for each episode):
    Initialize S
    Repeat (for each step of episode):
        Choose A from S using policy derived from Q (e.g., ε-greedy)
        Take action A, observe R, S'
        $Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$
        $S \leftarrow S'$;
    until S is terminal

Example: Cliff Walking with Sarsa and Q-Learning





ϵ-greedy exploration ($\epsilon = 0.1$)

Expected Sarsa

Can we speed-up learning by reducing variance of updates?

Expected Sarsa:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \mathbb{E}_\pi[Q(S_{t+1}, A_{t+1}) \mid S_{t+1}] - Q(S_t, A_t)\right]$$

$$= Q(S_t, A_t) + \alpha \left[R_{t+1} + \gamma \sum_a \pi(a|S_{t+1})\, Q(S_{t+1}, a) - Q(S_t, A_t)\right]$$

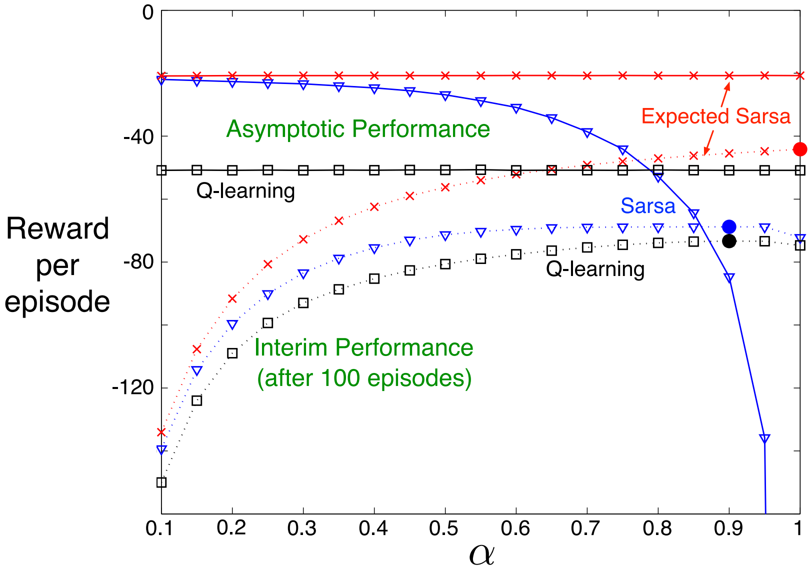
- Moves *deterministically* in same direction as Sarsa *on expectation*
- Can use as on-policy or off-policy

  $\Rightarrow$ Q-learning is special case where π is greedy and μ explores (e.g. ϵ-greedy)

Expected Sarsa in Cliff Walking

All algorithms used ϵ-greedy with $\epsilon = 0.1$

Solid circles mark best interim perfformance



n-step TD Methods

TD(0) uses 1-step return:

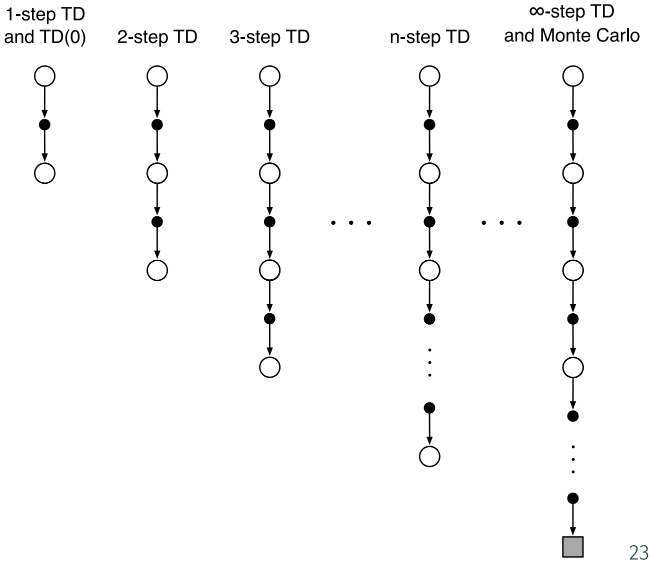
$$G_{t:t+1} \doteq R_{t+1} + \gamma V_t(S_{t+1})$$

MC uses full return:

$$G_{t:\infty} \doteq \sum_{k=1}^{\infty} \gamma^{k-1} R_{t+k}$$

n-step return bridges TD(0) and MC:

$$G_{t:t+n} = \sum_{k=1}^{n} \gamma^{k-1} R_{t+k} + \gamma^n V_{t+n-1}(S_{t+n})$$



n-step TD Methods

n-step return:

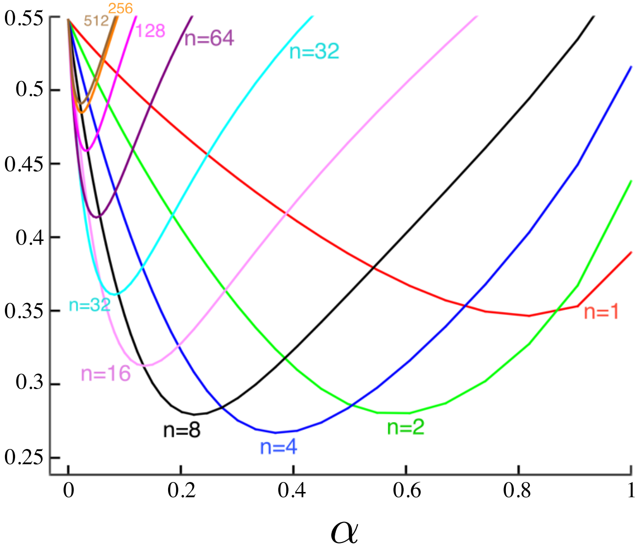
$$G_{t:t+n} = \sum_{k=1}^{n} \gamma^{k-1} R_{t+k} + \gamma^n V_{t+n-1}(S_{t+n})$$

n-step TD uses n-step return as target:

$$V_{t+n}(S_t) \doteq V_{t+n-1}(S_t) + \alpha \left[G_{t:t+n} - V_{t+n-1}(S_t) \right]$$

n-step TD Methods in Random Walk Example

Average RMS error over 19 states and first 10 episodes



On/Off-Policy Learning with n-Step Returns

Can similarly define n-step TD policy learning:

$$G_{t:t+n} = \sum_{k=1}^{n} \gamma^{k-1} R_{t+k} \; + \; \gamma^{n} Q_{t+n-1}(S_{t+n}, A_{t+n})$$

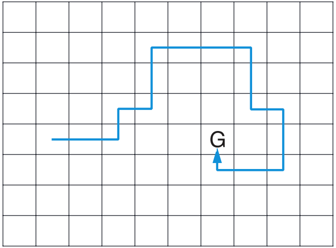
$$Q_{t+n}(S_t, A_t) \doteq Q_{t+n-1}(S_t, A_t) + \alpha \rho_{t+1:t+n} \left[G_{t:t+n} - Q_{t+n-1}(S_t, A_t) \right]$$

with importance ratio

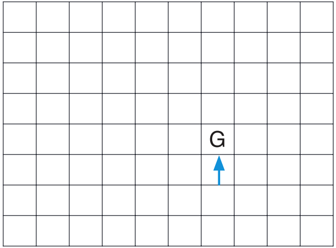
$$\rho_{t:h} \doteq \prod_{k=t}^{\min(h, T-1)} \frac{\pi(A_k|S_k)}{\mu(A_k|S_k)}$$

n-step TD Control in a Gridworld

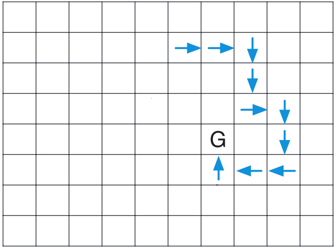
Path taken



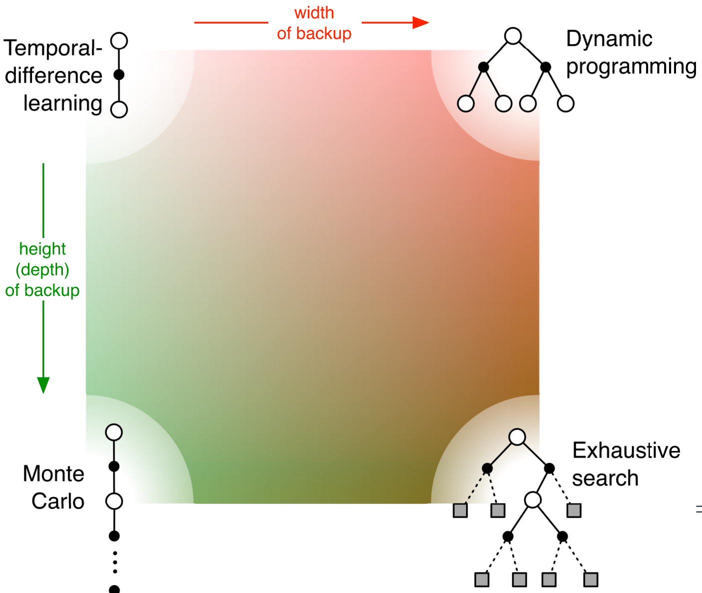
Action values increased
by one-step Sarsa



Action values increased
by 10-step Sarsa



Unified View



$\Rightarrow$ Planning

Reading

Required:

- RL book, chapter 6 (6.1–6.6) and chapter 7 (7.1–7.3)

Optional (convergence proofs):

- **For TD(0):** Dayan, P. (1992). The convergence of TD(λ) for general λ. Machine Learning, 8(3):341–362
- **For Sarsa:** Singh, S., Jaakkola, T., Littman, M., Szepesvri, C. (2000). Convergence results for single-step on-policy reinforcement-learning algorithms. Machine Learning, 38(3):287–308
- **For Q-learning:** Watkins, C., Dayan, P. (1992). Q-learning. Machine Learning, 8(3-4):279–292