UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

Intelligenza Artificiale

Model Free Prediction

a.a. 2022/2023

Outline

- Introduzione
- Approcci di Monte-Carlo
- Temporal-Difference (TD) learning
- TD(λ)

Model-Free Reinforcement Learning

- Finora ci siamo concentrati sulla risoluzione di un MDP noto (composto da stati, transizioni, ricompense, azioni)
- Model free
  - Nessun modello dell'ambiente
  - Nessuna conoscenza delle transizioni/ricompense del MDP
- Model-free prediction – Stimare la value function di un MDP non noto
- Model-free control – Ottimizzare la value function di un MDP non noto

Monte-Carlo (MC) Reinforcement Learning

- I metodi MC apprendono direttamente dagli episodi di esperienza
- MC è model-free: nessuna conoscenza delle transizioni/ricompense del MDP
- MC apprende da episodi completi: nessun bootstrapping
- Model-free control – Ottimizzare la value function di un MDP non noto
- MC utilizza l'idea più semplice possibile: valore = guadagno medio degli episodi
- Warning: può applicare MC solo a MDP episodiche
- Tutti gli episodi devono terminare

Monte-Carlo Policy Evaluation

- Obiettivo: apprendere v_π da episodi di esperienza in base alla policy π

$$S_1, A_1, R_2, \dots, R_k \sim \pi$$

- Ricordiamo che il guadagno è la ricompensa totale scontata

$$G_t = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_T$$

- Ricordiamo che la value function è il guadagno atteso

$$v_\pi(s) = \mathbb{E}\left[G_t | S_t = s\right]$$

- La Monte-Carlo policy evaluation utilizza il guadagno medio empirico al posto del guadagno atteso

First-Visit Monte-Carlo Policy Evaluation

- Per valutare lo stato *s*
- Il primo time-step *t* in cui lo stato *s* viene visitato in un episodio
  - Incremento del contatore N(*s*) <- N(*s*) + 1
  - Incremento del guadagno totale S(*s*) <- S(*s*) + G(*t*)
  - Stima del valore dal guadagno medio V(*s*) = S(*s*)/N(*s*)
- Per la legge dei grandi numeri

$$V(s) \rightarrow v_\pi(s) \text{ as } N(s) \rightarrow \infty$$

First-Visit Monte-Carlo Policy Evaluation

First-visit MC prediction, for estimating $V \approx v_\pi$

Input: a policy π to be evaluated

Initialize:

$V(s) \in \mathbb{R}$, arbitrarily, for all $s \in \mathcal{S}$

$Returns(s) \leftarrow$ an empty list, for all $s \in \mathcal{S}$

Loop forever (for each episode):

Generate an episode following π: $S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$

$G \leftarrow 0$

Loop for each step of episode, $t = T-1, T-2, \ldots, 0$:

$G \leftarrow \gamma G + R_{t+1}$

Unless S_t appears in $S_0, S_1, \ldots, S_{t-1}$:

Append G to $Returns(S_t)$

$V(S_t) \leftarrow$ average$(Returns(S_t))$

Every-Visit Monte-Carlo Policy Evaluation

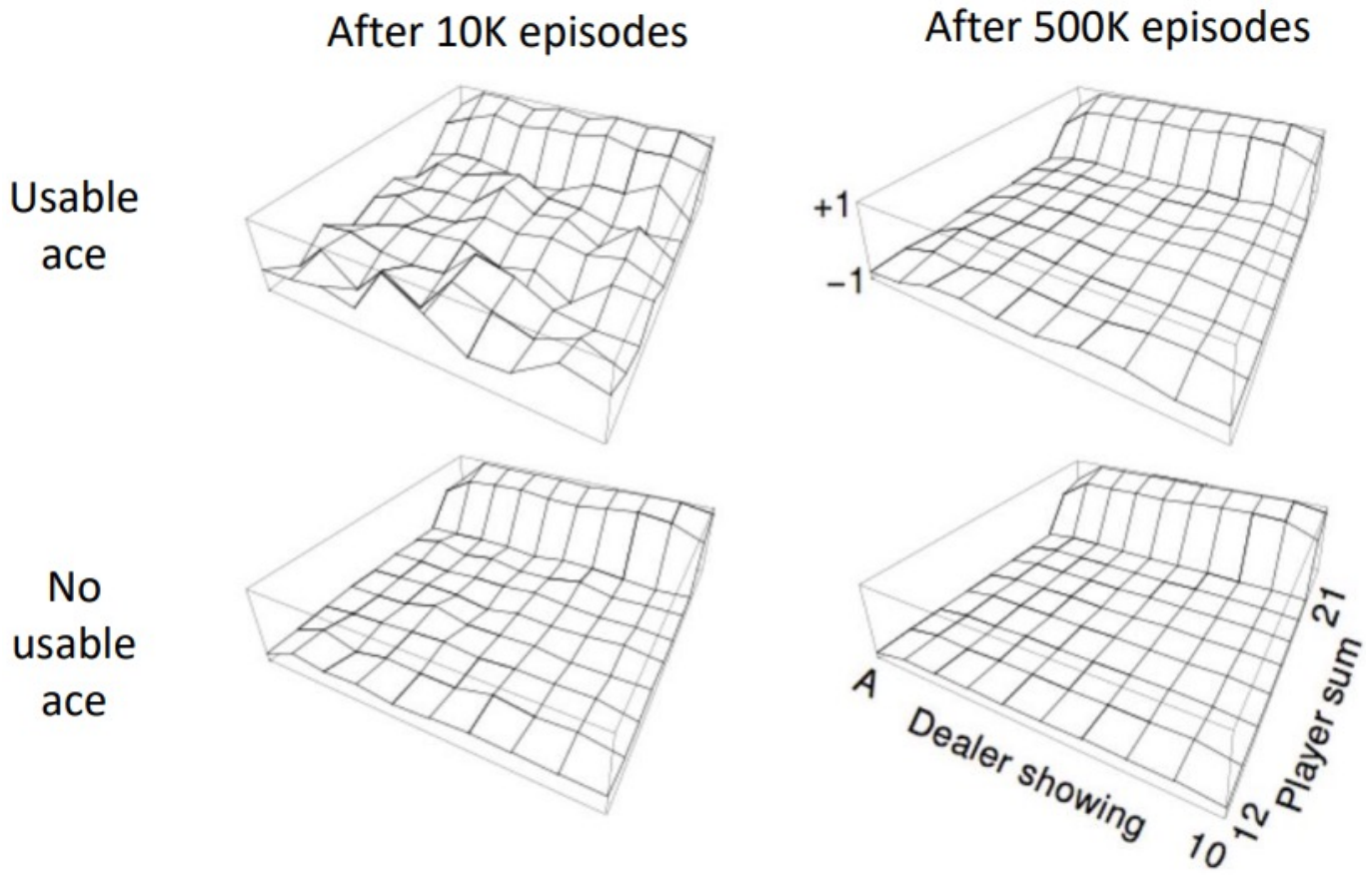
- Per valutare lo stato *s*
- Ogni time-step *t* in cui lo stato *s* viene visitato in un episodio
  - Incremento del contatore N(*s*) <- N(*s*) + 1
  - Incremento del guadagno totale S(*s*) <- S(*s*) + G(*t*)
  - Stima del valore dal guadagno medio V(*s*) = S(*s*)/N(*s*)
- Di nuovo, per la legge dei grandi numeri

$$V(s) \rightarrow v_\pi(s) \text{ as } N(s) \rightarrow \infty$$

Esempio Blackjack

- Stati (200)
  - Somma attuale (12-21)
  - Carta scoperta dal banco (asso-10)
  - Ho uno *usable ace*? (si-no)
- Ricompensa per l'azione stick (smettere di ricevere carte (e terminare)):
  - +1 se la somma delle carte > somma delle carte del banco
  - 0 se la somma delle carte = somma delle carte del banco
  - -1 se la somma delle carte < somma delle carte del banco
- Ricompensa per l'azione twist (prendere un'altra carta (senza sostituirla)):
  - -1 se la somma delle carte > 21 (e termina)
  - 0 altrimenti
  - Transizioni: automaticamente twist se la somma delle carte è < 12

Value Function del Blackjack dopo il MC Learning



Policy: stick se la somma delle carte è ≥ 20, altrimenti twist

Media incrementale

- La media $\mu_1, \mu_2, \ldots$ di una sequenza $x_1, x_2, \ldots$ può essere calcolata in modo incrementale

$$\mu_k = \frac{1}{k}\sum_{j=1}^{k} x_j = \frac{1}{k}\left(x_k + \sum_{j=1}^{k-1} x_j\right)$$

$$\mu_k = \frac{1}{k}(x_k + (k-1)\mu_{k-1}) = \mu_{k-1} + \frac{1}{k}(x_k - \mu_{k-1})$$

Aggiornamento Media incrementale del MC

- Aggiorna V(s) in modo incrementale dopo l'episodio

$$S_1, A_1, R_2, \dots, R_T$$

- Per ogni stato S_t con guadagno G_t
  - Incremento del contatore N(s) <- N(s) + 1
  - Aggiornamento della value function (con media incrementale)

$$\mathrm{V}(S_t) \leftarrow \mathrm{V}(S_t) + \frac{1}{N(S_t)}(G_t - V(S_t))$$

- In problemi non stazionari si traccia una media mobile (dimenticando i vecchi episodi)

$$\mathrm{V}(S_t) \leftarrow \mathrm{V}(S_t) + \alpha(G_t - V(S_t))$$

$$NewEstimate \leftarrow OldEstimate + StepSize\left[Target - OldEstimate\right]$$

Temporal-Difference Learning

Temporal-Difference (TD) Learning

- I metodi TD apprendono direttamente da episodi di esperienza
- TD è model-free: nessuna conoscenza delle transizioni/ricompense del MDP
- TD apprende da episodi incompleti, tramite bootstrapping
- TD aggiorna un'ipotesi verso un'altra ipotesi

MC vs TD Learning

- Obiettivo: apprendere v_π da episodi di esperienza in base alla policy π
- MC incrementale ad ogni visita
  - Aggiornamento del valore $V(S_t)$ verso il guadagno effettivo G_t

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$$

- Algoritmo più semplice del temporal-difference learning (TD(0))
  - Aggiornamento del valore $V(S_t)$ verso il guadagno stimato $R_{t+1} + \gamma V(S_{t+1})$

$$V(S_t) \leftarrow V(S_t) + \alpha\Big[\underbrace{R_{t+1} + \gamma V(S_{t+1})}_{\text{TD target}} - V(S_t)\Big]$$

$$\underbrace{\qquad\qquad\qquad\qquad}_{\text{TD error } \delta_t}$$

TD(0) Policy Evaluation

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

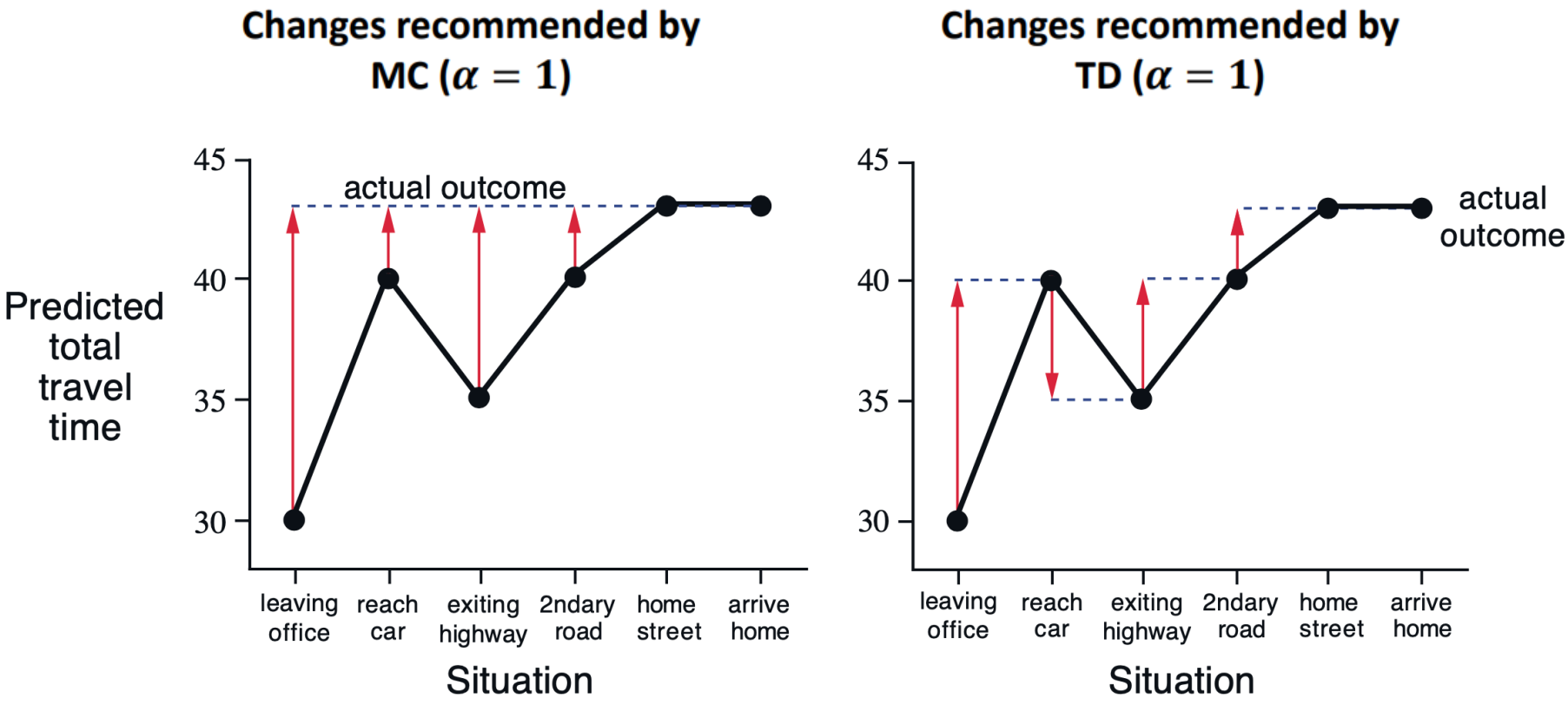
Esempio Temporal Difference

https://cs.stanford.edu/people/karpathy/reinforcejs/gridworld_td.html

Esempio Guida Verso Casa

State	Elapsed Time (minutes)	Predicted Time to Go	Predicted Total Time
leaving office	0	30	30
reach car, raining	5	35	40
exit highway	20	15	35
behind truck	30	10	40
home street	40	3	43
arrive home	43	0	43

Esempio Guida Verso Casa – MC vs TD



Vantaggi e svantaggi di MC vs. TD (I)

- TD può apprendere prima di conoscere il risultato finale
  - TD può apprendere online dopo ogni step
  - MC deve aspettare la fine dell'episodio prima di conoscere il guadagno
- TD può apprendere senza il risultato finale
  - TD può apprendere da sequenze incomplete
  - MC può apprendere solo da sequenze complete
  - TD funziona in ambienti continui (non terminanti)
  - MC funziona solo in ambienti episodici (terminanti)

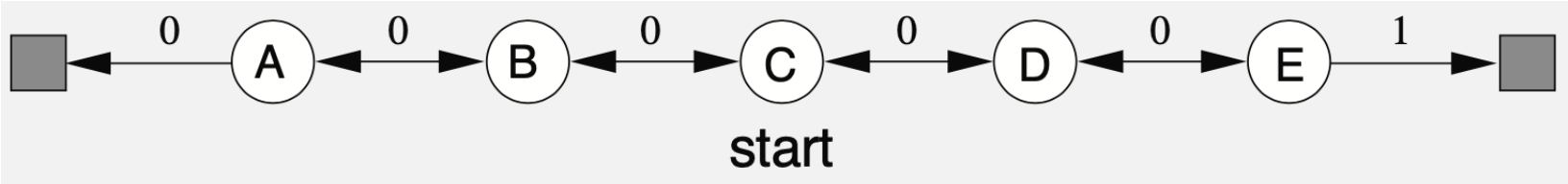
Bias-Variance Tradeoff

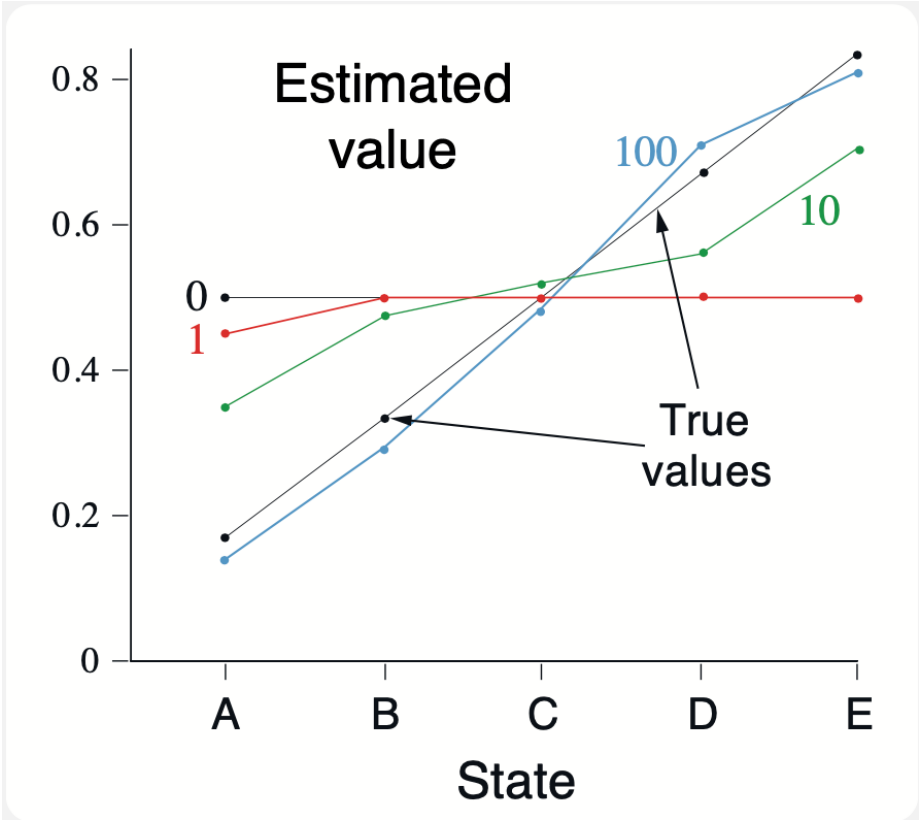
- Il guadagno $G_t = R_{t+1} + \gamma R_{t+2} + \cdots, \gamma^{T-1} R_T$ è una stima imparziale (unbiased) di $v_\pi(S_t)$
- Il vero target di TD $R_{t+1} + \gamma v_\pi(S_{t+1})$ è una stima imparziale (unbiased) di $v_\pi(S_t)$
- Il target di TD $R_{t+1} + \gamma V(S_{t+1})$ è una stima distorta (biased) di $v_\pi(S_t)$
- Il target di TD ha una varianza molto più bassa rispetto al guadagno:
  - Il guadagno G_t dipende da molte azioni, transizioni e ricompense casuali
  - Il target di TD dipende da una sola azione, transizione e ricompensa casuale

Vantaggi e svantaggi di MC vs. TD (II)

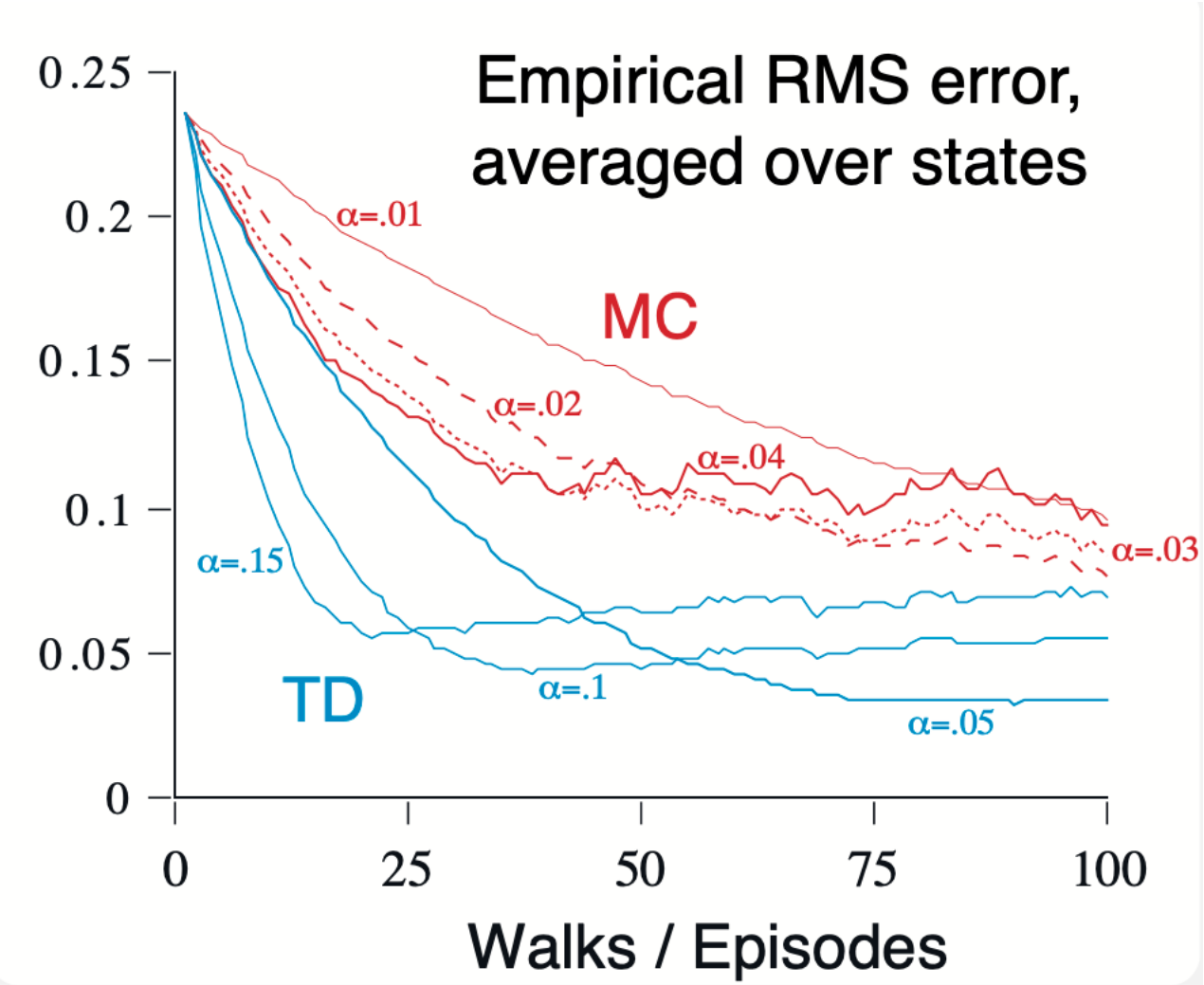
- MC ha un'alta varianza e zero bias
  - Buone proprietà di convergenza (anche con approssimazione della funzione)
  - Non è molto sensibile al valore iniziale
  - Molto semplice da comprendere e utilizzare
- TD ha una bassa varianza ma alcuni bias
  - Solitamente più efficiente di MC
  - TD(0) converge a $v_\pi(s)$ (ma non sempre con l'approssimazione della funzione)
  - Più sensibile al valore iniziale

Esempio Random Walk





Esempio Random Walk – MC vs TD



Batch MC e TD

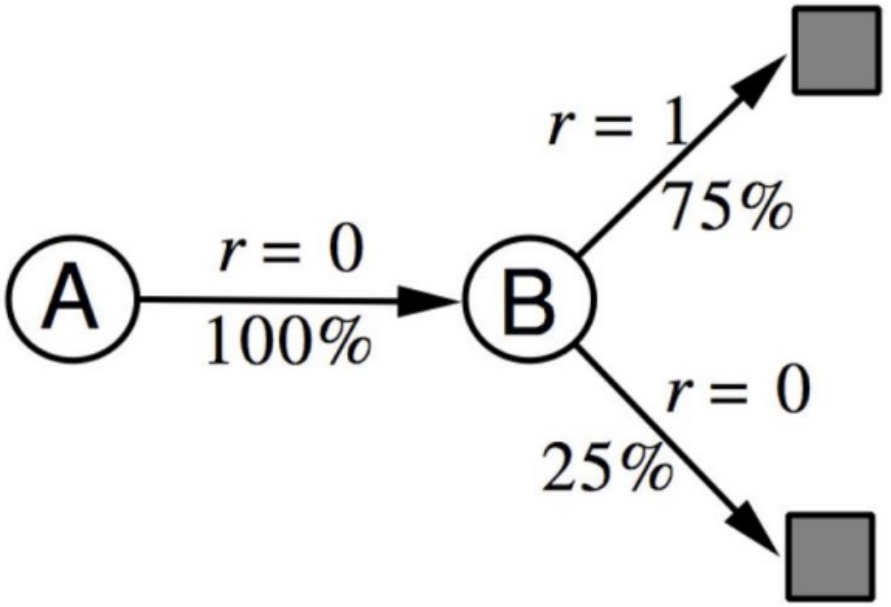
- MC e TD convergono: V(s) → $v_\pi(s)$ con l'esperienza → ∞
- Ma che dire della soluzione batch per l'esperienza finita?

$$s_1^1, a_1^1, r_2^1, \dots, s_{T_1}^1$$
$$\vdots$$
$$s_1^K, a_1^K, r_2^K, \dots, s_{T_K}^K$$

- ad es. campionare ripetutamente l'episodio $k \in [1,K]$
- Applicare MC o TD(0) all'episodio k

Un semplice esempio

- Due stati A e B; nessuna scontistica; 8 episodi di esperienza
  1. A, 0, B, 0
  2. B, 1
  3. B, 1
  4. B, 1
  5. B, 1
  6. B, 1
  7. B, 1
  8. B, 0
- Cos'è V(A); V(B)?



Certainty Equivariance

- MC converge alla soluzione con il il minimo errore quadratico medio
  - Miglior adattamento ai guadagni osservati

$$\sum_{k=1}^{K}\sum_{t=1}^{T_k}\left(G_t^k - V(s_t^k)\right)^2$$

- TD(0) converge alla soluzione del modello di Markov a massima verosimiglianza
  - Soluzione del MDP <$\mathcal{S}$, $\mathcal{A}$, $\boldsymbol{P}$, $\mathcal{R}$, γ> che meglio si adatta ai dati

$$\hat{P}_{ss'}^a = \frac{1}{N(s,a)}\sum_{k=1}^{K}\sum_{t=1}^{T_k}\mathbf{1}(s_t^k, a_t^k, s_{t+1}^k; s, a, s')$$

$$\hat{\mathcal{R}}_s^a = \frac{1}{N(s,a)}\sum_{k=1}^{K}\sum_{t=1}^{T_k}\mathbf{1}(s_t^k, a_t^k; s, a) r_t^k$$

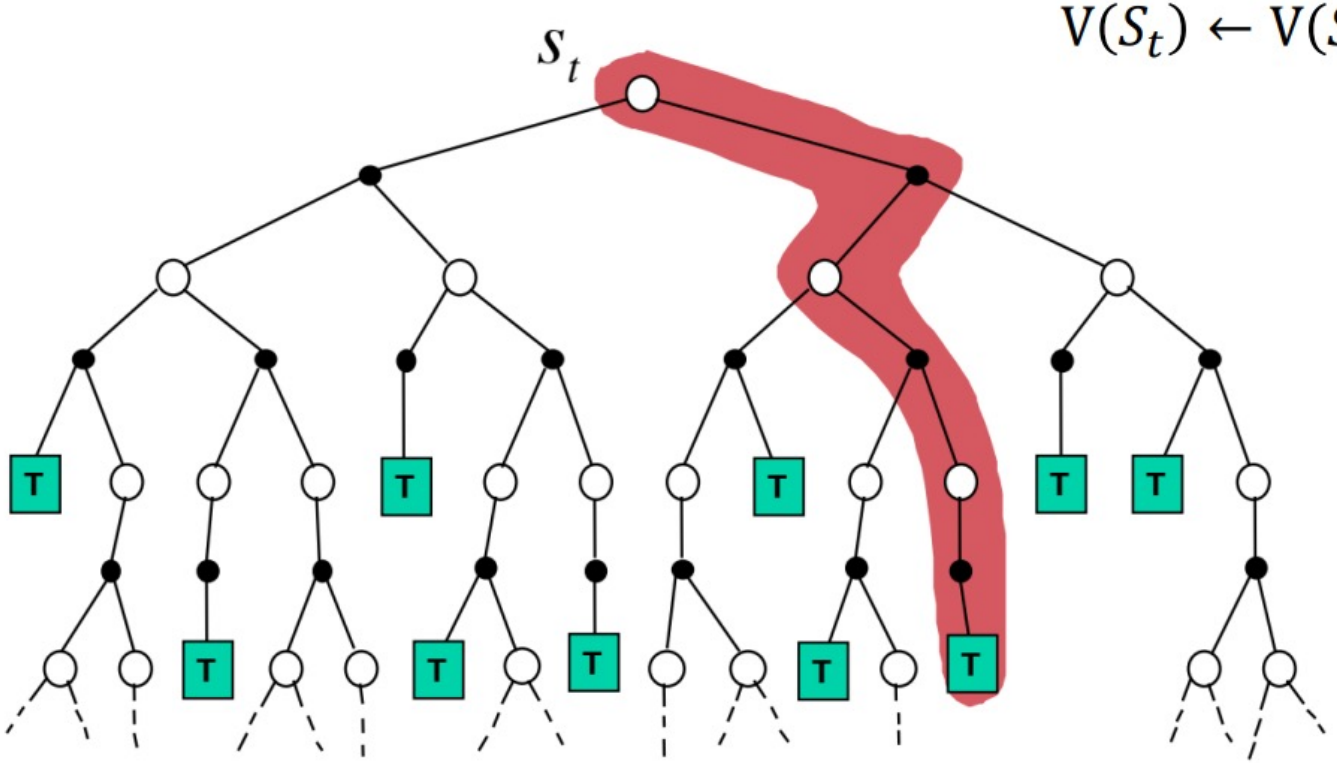
Vantaggi e svantaggi di MC vs. TD (III)

- TD sfrutta la proprietà di Markov
  - Solitamente più efficiente in ambienti di Markov
- MC non sfrutta la proprietà di Markov
  - Solitamente più efficace in ambienti non di Markov

Visione Unificata

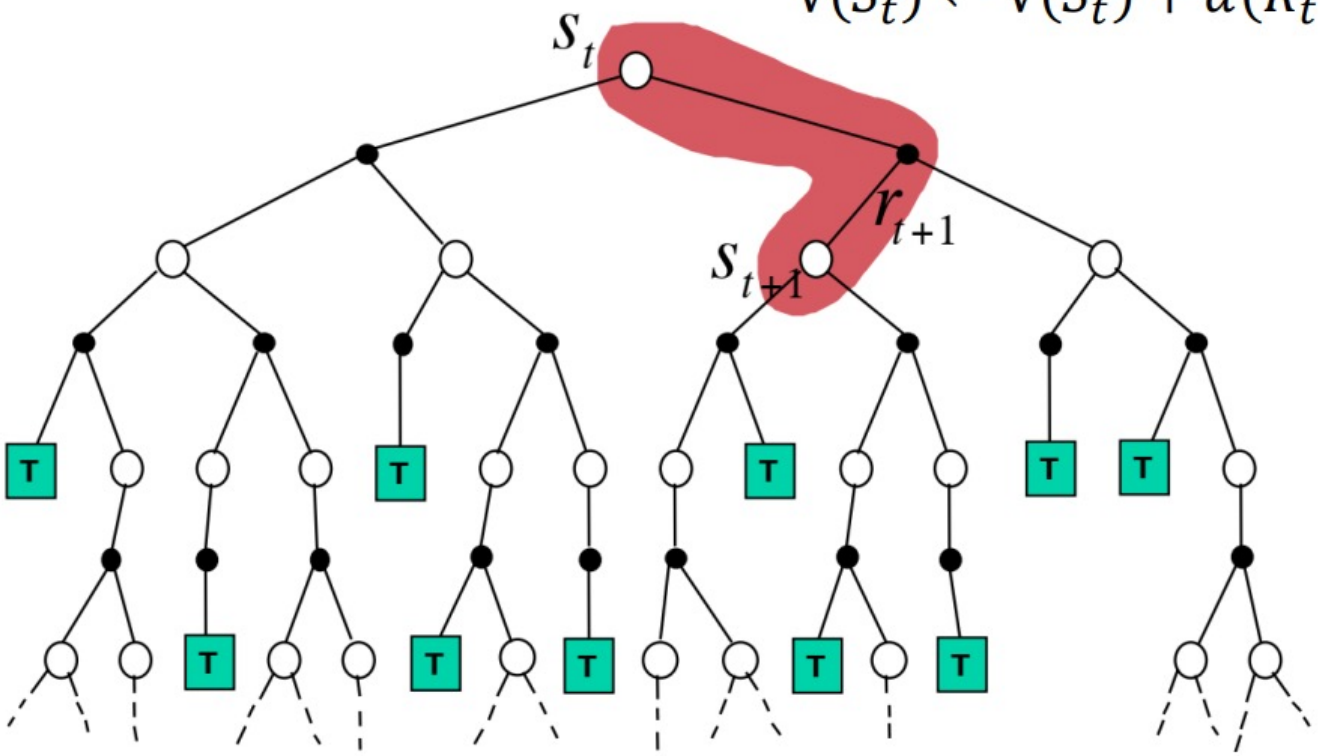
MC Update

$$V(S_t) \leftarrow V(S_t) + \alpha(G_t - V(S_t))$$



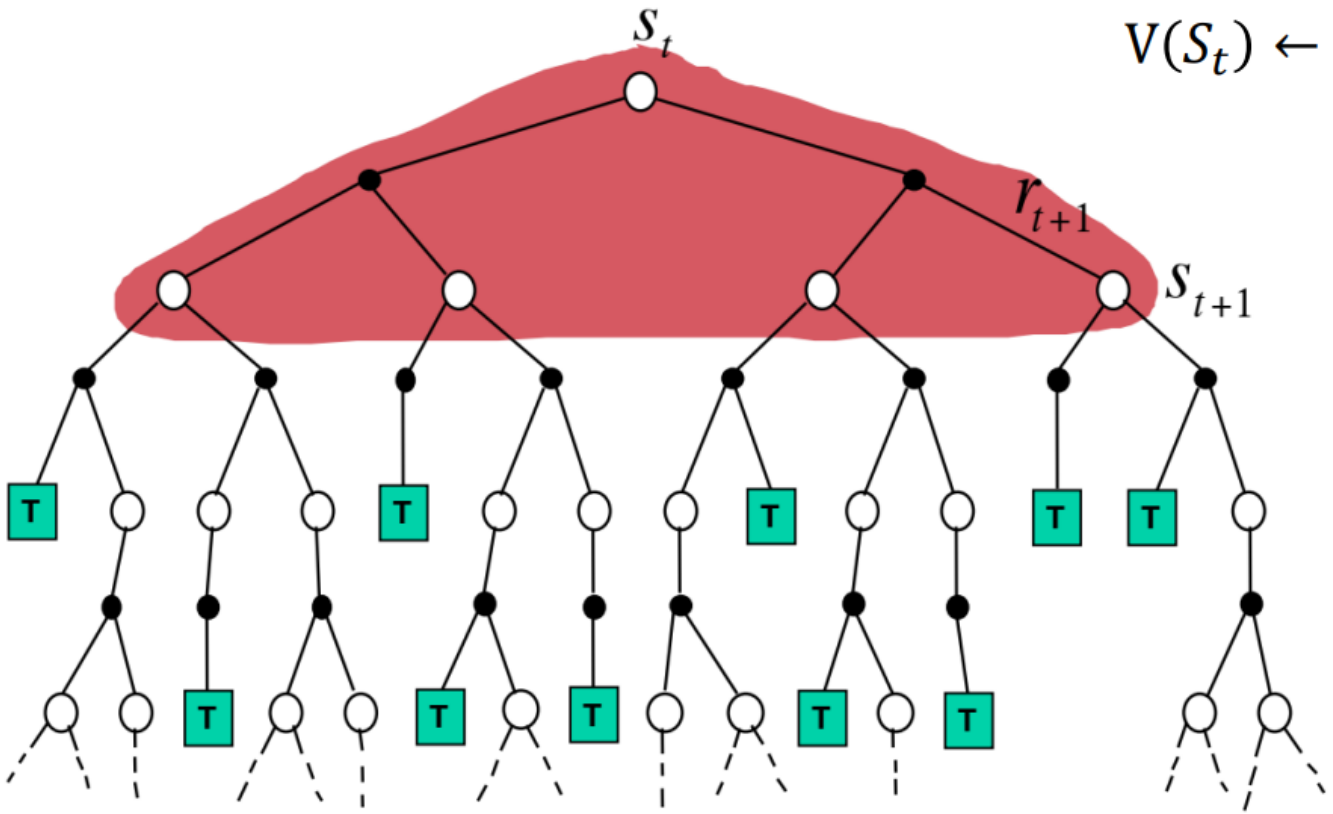
TD Update

$$V(S_t) \leftarrow V(S_t) + \alpha(R_t + \gamma V(S_{t+1}) - V(S_t))$$



Programmazione dinamica

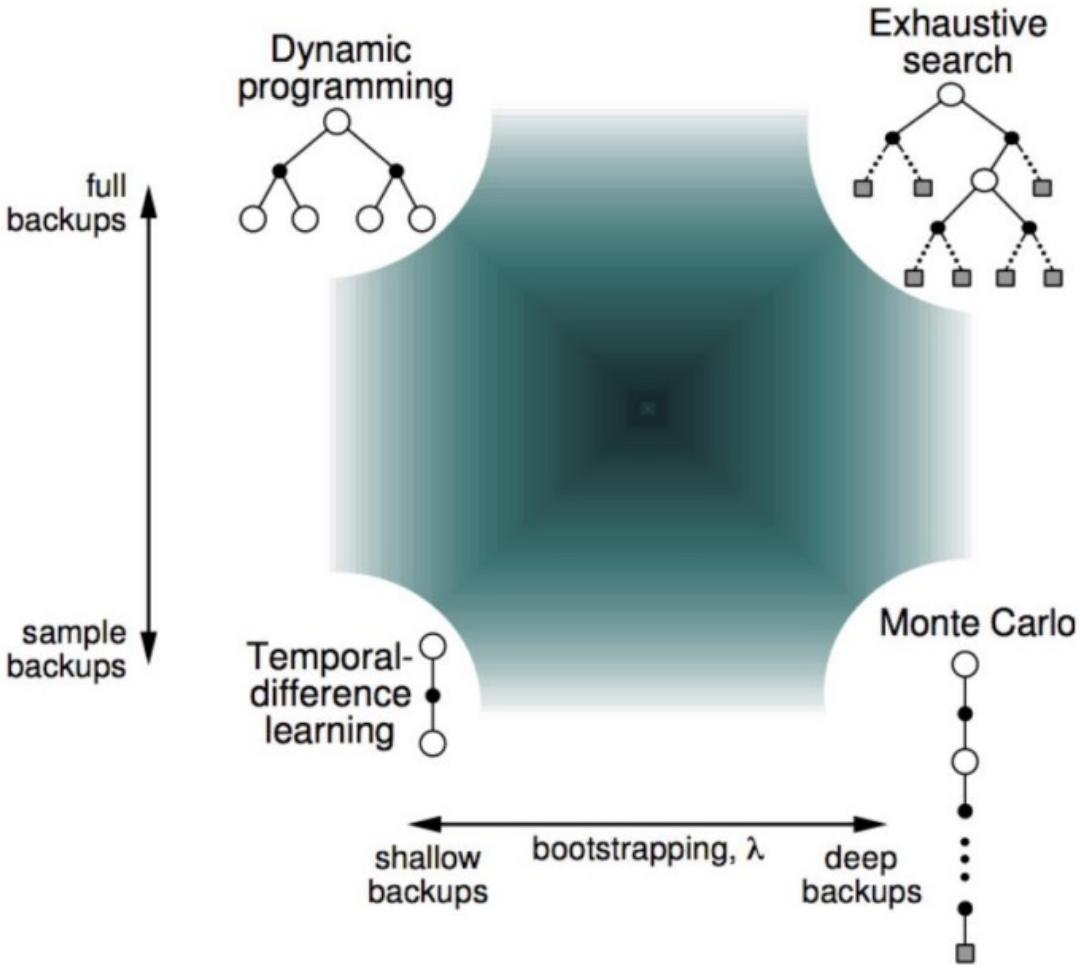
$$V(S_t) \leftarrow \mathbb{E}[R_{t+1} + \gamma V(S_{t+1})]$$



Bootstrapping and Sampling

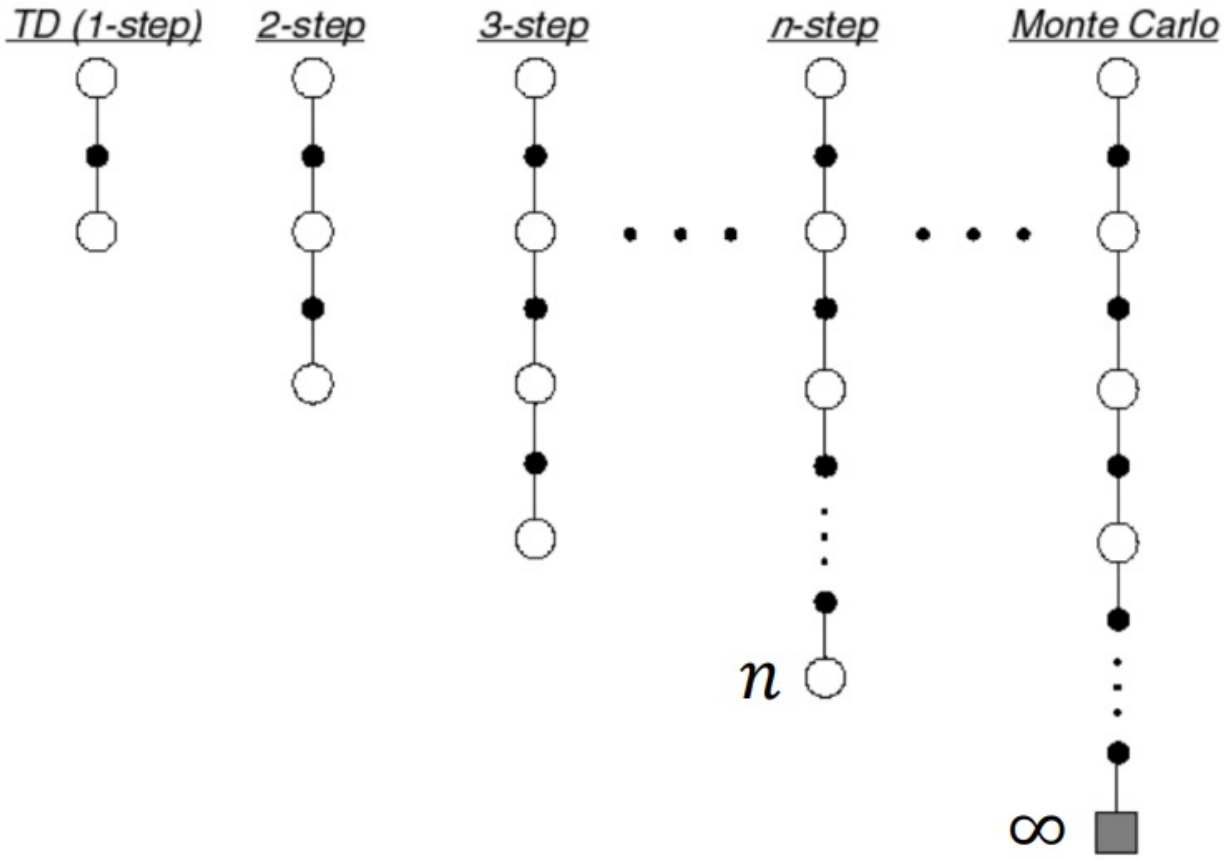
- Bootstrapping - L'aggiornamento prevede una stima
  - MC non esegue il bootstrap
  - DP esegue il bootstrap
  - TD esegue il bootstrap
- Sampling - L'aggiornamento campiona una previsione
  - MC esegue il sampling
  - DP non esegue il sampling
  - TD esegue il sampling

Unified View del RL



Generalizzazione TD

Generalizzazione del TD: n-step Prediction



Generalizzazione del TD: n-step Return

- Consideriamo i seguenti n-step return per $n = 1, 2, \ldots, \infty$

$$n = 1 \quad (\text{TD}) \quad G_t^{(1)} = R_{t+1} + \gamma V(S_{t+1})$$
$$n = 2 \qquad G_t^{(2)} = R_{t+1} + \gamma R_{t+2} + \gamma^2 V(S_{t+2})$$
$$\ldots$$
$$n = \infty \quad (\text{MC}) \quad G_t^{(\infty)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{T-1} R_T$$

- Definiamo l'n-step return

$$G_t^{(n)} = R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})$$

- Apprendimento basato sull'n-step difference

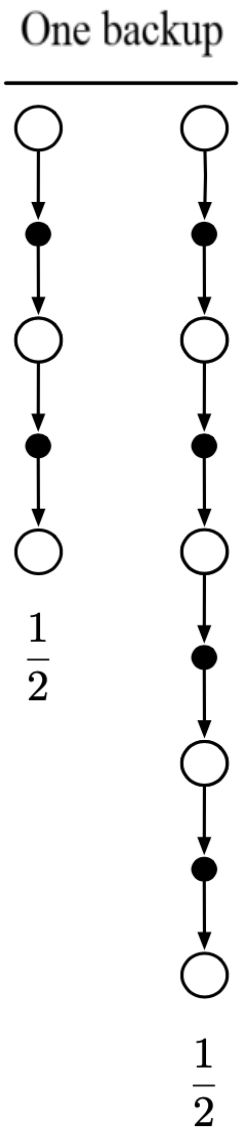
$$\mathrm{V}(S_t) \leftarrow \mathrm{V}(S_t) + \alpha \left(G_t^{(n)} - V(S_t) \right)$$

Media degli n-step Return

- Possiamo eseguire una media degli n-step return su differenti n
  - Ad esempio, la media dei 2-step e 4-step return

$$\frac{1}{2}G^{(2)} + \frac{1}{4}G^{(4)}$$

- Combina informazioni provenienti da due fasi temporali diverse
- Possiamo combinare in modo efficiente le informazioni provenienti da tutti i time-step?



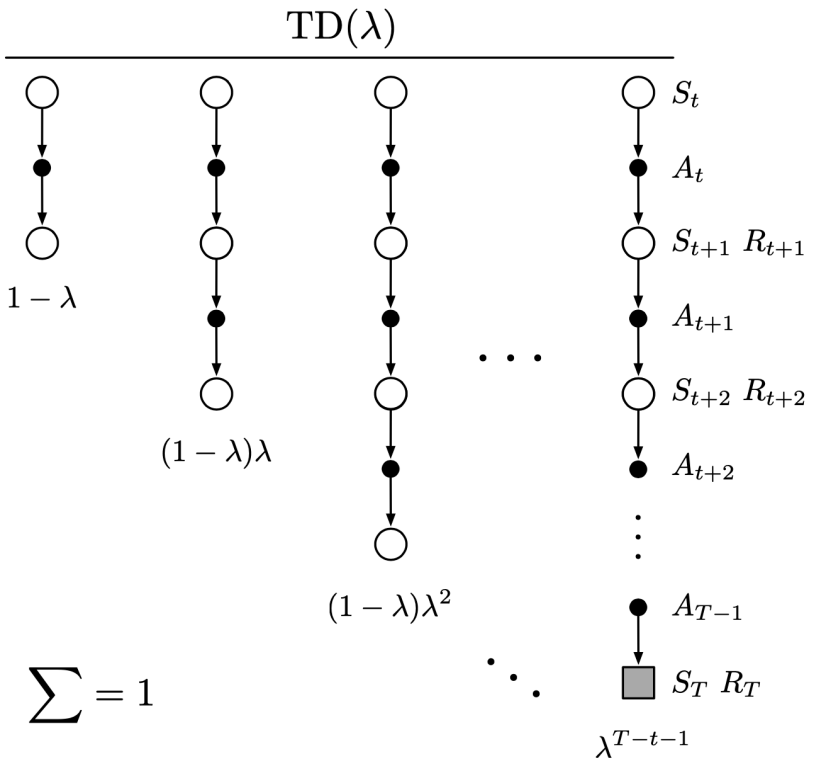
λ-return

- L'λ-return G_t^{λ} combina tutti gli gli n-step return $G_t^{(n)}$
- Utilizza i pesi (1-λ)λ^{n-1}

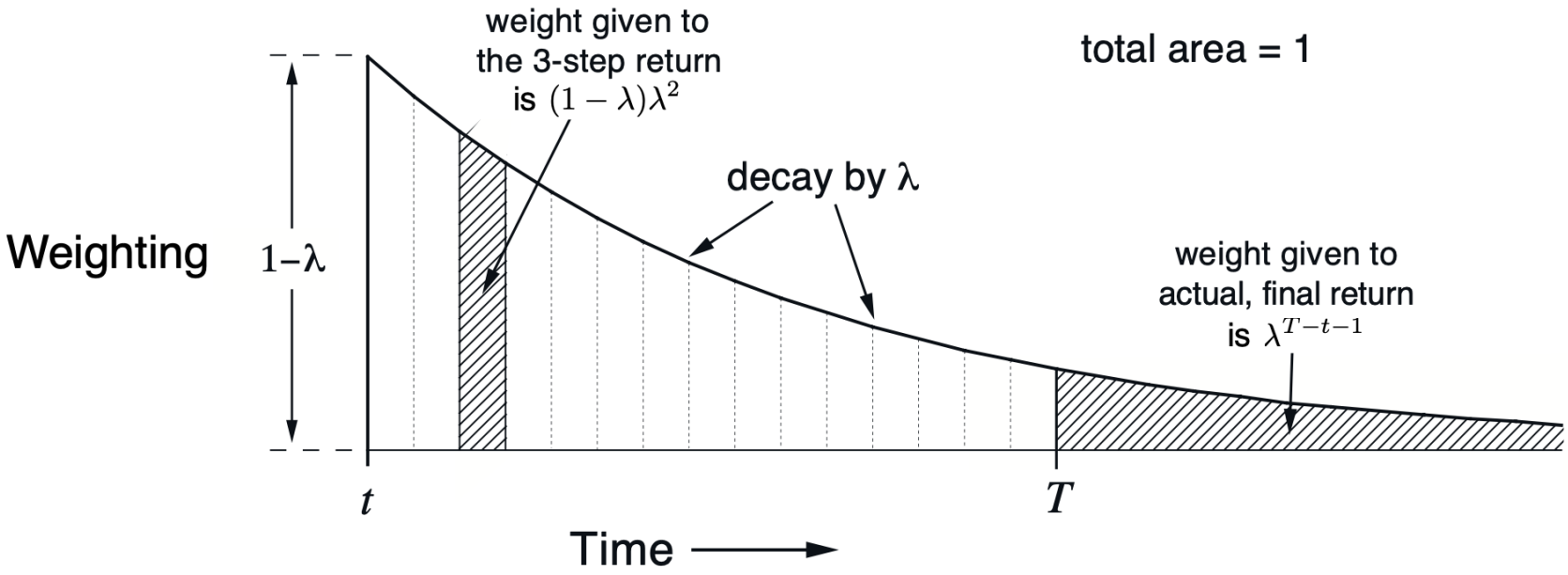
$$G_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{n-1} G_t^{(n)}$$

- Aggiorna in modo appropriato (TD(λ))

$$V(S_t) \leftarrow V(S_t) + \alpha\left(G_t^{\lambda} - V(S_t)\right)$$



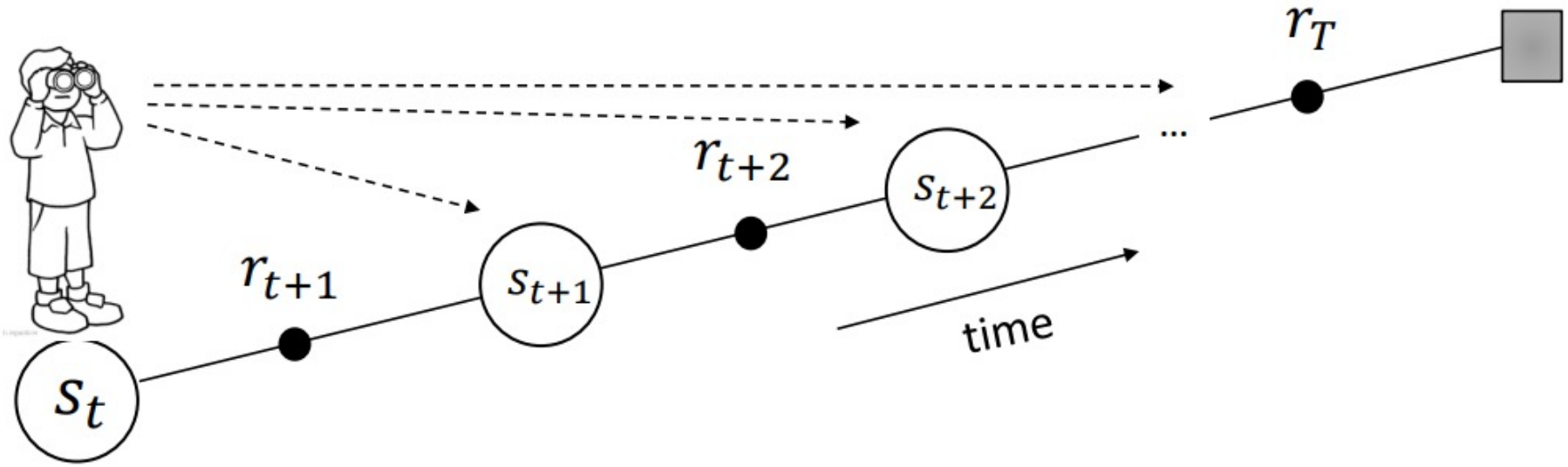
TD(λ) Weight Function



$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1}\lambda^{n-1}G_{t:t+n} + \lambda^{T-t-1}G_t$$

- $\lambda = 1$ Monte Carlo
- $\lambda = 0$ one-step TD

Forward View TD(λ)



- Aggiorna la value function verso il λ-return
- La forward view guarda al futuro per calcolare G_t^{λ}
- Come il MC, può essere calcolato solo da episodi completi

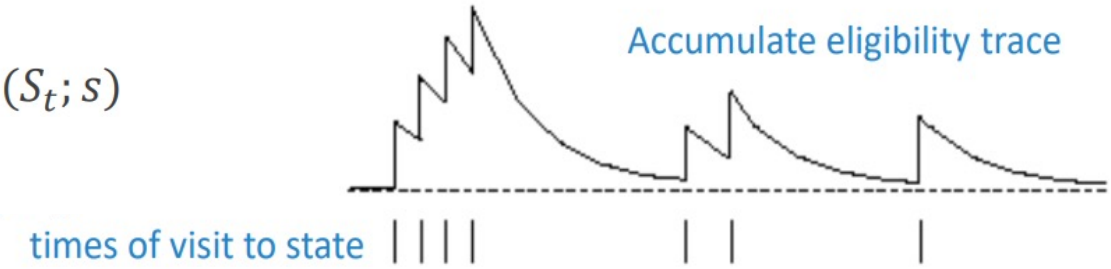
Backward View TD(λ)

- La forward view fornisce la teoria
- La backward view fornisce il meccanismo
- Aggiornamento online, ad ogni step, da sequenze incomplete

Eligibility Traces

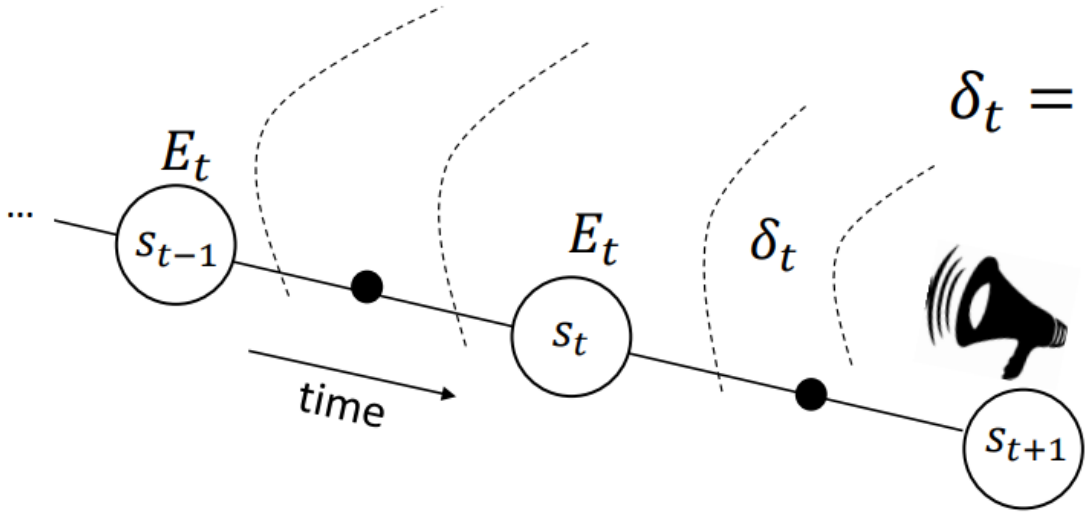
- Credit assignment problem: il problema di determinare le azioni che portano a un determinato risultato. Cosa ha causato lo shock?
  - Frequency heuristic: assegna credito agli stati più frequenti
  - Recency heuristic: assegna credito agli stati più recenti
- Le eligibility traces combinano entrambe le euristiche

$$E_0(s) = 0$$
$$E_t(s) = \gamma\lambda E_{t-1}(s) + \mathbf{1}(S_t; s)$$



Backward View TD(λ)

- Si tiene traccia di una eligibility trace per ogni stato s
- Aggiorna il valore V(s) per ogni stato s
- In proporzione al TD-error δ_t e alla eligibility trace $E_t(s)$



$$\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t)$$

$$V(s) = V(s) + \alpha \delta_t E_t(s)$$

TD(λ) e TD(0)

- Quando $\lambda = 0$, viene aggiornato solo lo stato attuale

$$E_t(s) = \mathbf{1}(S_t; s)$$

$$V(s) \leftarrow V(s) + \alpha\delta_t E_t(s)$$

- Equivalente all'aggiornamento TD(0)

$$V(S_t) \leftarrow V(S_t) + \alpha\delta_t$$

TD(λ) e MC

- Quando $\lambda = 1$, il credito è differito fino alla fine dell'episodio
- Si considerano gli ambienti episodici con aggiornamenti offline
- Durante un episodio, l'aggiornamento totale per TD(1) è uguale all'aggiornamento totale per MC
- Teorema
  - La somma degli aggiornamenti offline è identica per la forward e backward view TD(λ)

$$\sum_{t=1}^{T} \alpha \delta_t E_t(s) = \sum_{t=1}^{T} \alpha \left(G_t^{\lambda} - V(S_t) \right) \mathbf{1}(S_t; s)$$

Telescoping in TD(1)

- Quando $\lambda = 1$, la somma degli errori di TD si trasforma nell'errore di MC
- TD(1) è approssimativamente equivalente a un every-visit Monte-Carlo
- Gli errori vengono accumulati online, step-by-step
- Se la value function viene aggiornata offline solo alla fine dell'episodio, allora l'aggiornamento totale è uguale a quello di MC

Take home messages

- La Model-free prediction è la stima della value function di un MDP non noto
  - È basato sui sample-update
- Metodi di Monte-Carlo
  - Stima della value function attraverso la media dei sample return
  - Solo per compiti episodici (che terminano indipendentemente dalle azioni intraprese)
- TD learning
  - Apprendere dalle stime esistenti (distorte) del guadagno futuro (bootstrapping)
  - Esplorare il futuro fino all'*n*-esimo step