

## Apprendimento con Rinforzo

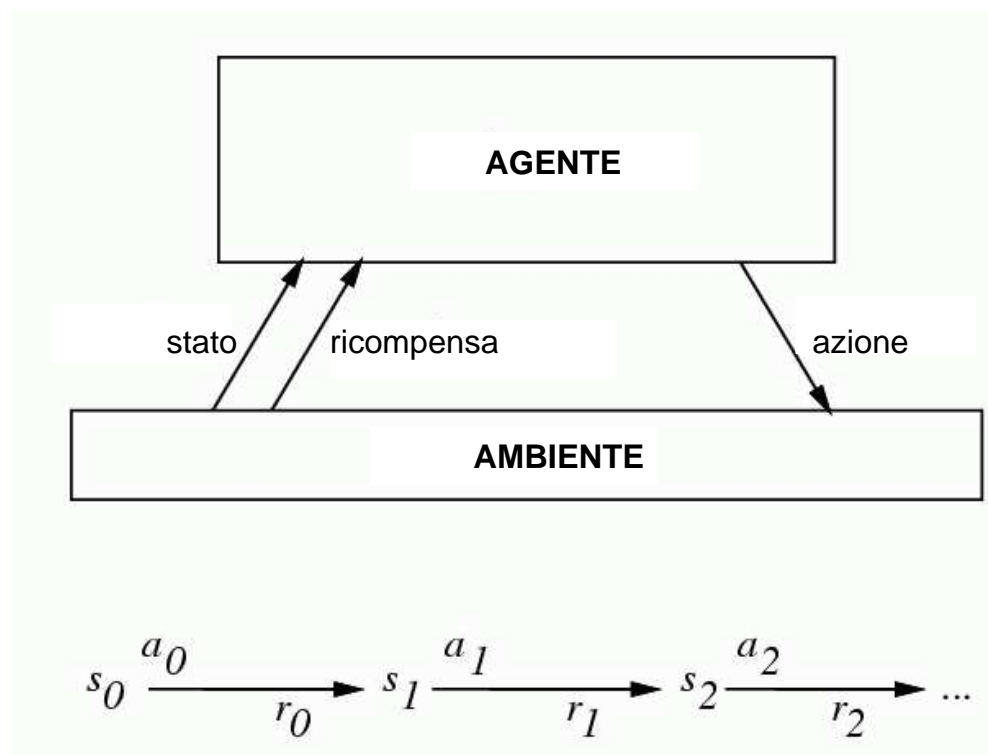
- Sono dati:
  - agente (intelligente ?), che può
    - \* trovarsi in uno stato  $s$ , ed
    - \* eseguire una azione  $a$  (all'interno delle azioni possibili nello stato corrente)
  - ed opera in un ambiente  $e$ , che applicando una azione  $a$  nello stato  $s$  restituisce
    - \* lo stato successivo, e
    - \* una ricompensa  $r$ , che può essere positiva (+), negativa (-), o neutra (0).
- Scopo dell'agente è quello di massimizzare una funzione delle ricompense  
(es. ricompensa scontata:  $\sum_{t=0}^{\infty} \gamma^t r_{t+1}$  dove  $0 \leq \gamma < 1$ )

Esempio di applicazione: sistema che imparare a giocare a dama

Esempio di applicazione: modulo di navigazione per un robot in un ambiente sconosciuto

Esempio di applicazione: navigare sul Web alla ricerca di informazione focalizzata

## Agente/Ambiente



Goal: apprendere le azioni che massimizzano

$$r_0 + \gamma r_1 + \gamma^2 r_2 + \dots, \text{ dove } 0 \leq \gamma < 1$$

## Processi di Decisione di Markov

Si assume

- insieme finito di stati  $S$
- insieme di azioni  $A$
- ad ogni istante di tempo  $t$  l'agente osserva lo stato  $s_t \in S$  e sceglie l'azione  $a_t \in A$
- poi riceve la ricompensa immediata  $r_t$
- e cambia lo stato in  $s_{t+1}$
- **assunzione di Markov**:  $s_{t+1} = \delta(s_t, a_t)$  e  $r_t = r(s_t, a_t)$ 
  - cioè,  $r_t$  e  $s_{t+1}$  dipendono **SOLO** dallo stato ed azione **correnti**
  - le funzioni  $\delta$  e  $r$  possono essere **DETERMINISTICHE** o **NONDETERMINISTICHE**
  - le funzioni  $\delta$  e  $r$  possono essere **CONOSCIUTE** o **IGNOTE**

## Compito di Apprendimento per l'Agente

Eseguire azioni nell'Ambiente, osservare i risultati e

- apprendere la politica (per le azioni)  $\pi : S \rightarrow A$  che massimizza (nel caso più generale)

$$E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots]$$

per ogni stato iniziale in  $S$

- $0 \leq \gamma < 1$  è il *fattore di sconto* per ricompense future

Novità:

- la funzione target è  $\pi : S \rightarrow A$
- ma non si hanno a disposizione esempi nella forma  $(s, a)$
- gli esempi di apprendimento sono nella forma  $((s, a), r)$

## La Funzione di Valutazione

Consideriamo il **CASO DETERMINISTICO**:

- per ogni possibile politica  $\pi$  che l'agente può adottare, possiamo definire una funzione di valutazione sugli stati

$$\begin{aligned} V^\pi(s) &\equiv r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots \\ &\equiv \sum_{i=0}^{\infty} \gamma^i r_{t+i} \end{aligned}$$

dove  $r_t, r_{t+1}, \dots$  sono generati seguendo la politica  $\pi$  a partire dallo stato  $s$

- riformulato, il compito consiste nell'apprendere la politica ottima  $\pi^*$

$$\pi^* \equiv \arg \max_{\pi} V^\pi(s), (\forall s)$$

## Cosa Apprendere ?

Possiamo tentare di far apprendere all'agente la funzione di valutazione  $V^{\pi^*}$  (riferita nel seguito semplicemente con  $V^*$ )

L'agente potrebbe effettuare una ricerca in avanti per scegliere la migliore azione a partire da ogni stato  $s$  perché

$$\pi^*(s) = \mathit{arg\,max}_a [r(s, a) + \gamma V^*(\delta(s, a))]$$

### Problema:

- Funziona solo se l'agente conosce  $\delta : S \times A \rightarrow S$ , e  $r : S \times A \rightarrow \mathfrak{R}$
- ... ma se l'agente non ha questa conoscenza (come succede in molte applicazioni del mondo reale) non si può procedere in questo modo...

## La Funzione $Q()$

Soluzione:

- definire una nuova funzione molto simile a  $V^*$

$$Q(s, a) \equiv r(s, a) + \gamma V^*(\delta(s, a))$$

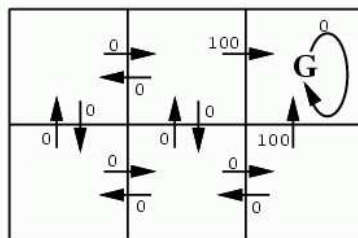
- se l'agente apprende  $Q$ , esso può scegliere l'azione ottima anche senza conoscere  $\delta$ !

$$\pi^*(s) = \arg \max_a [r(s, a) + \gamma V^*(\delta(s, a))]$$

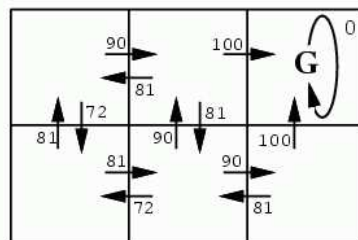
$$\pi^*(s) = \arg \max_a Q(s, a)$$

Quindi,  $Q$  è la funzione di valutazione che l'agente deve apprendere

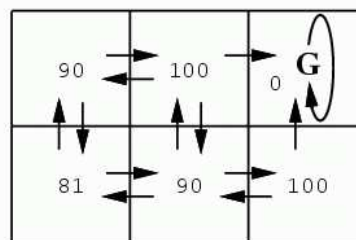
# Esempio



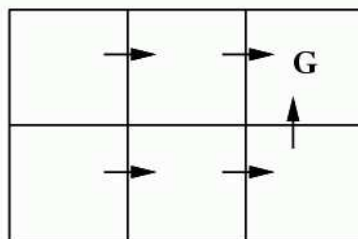
$r(s, a)$



$Q(s, a)$



$V^*(s)$



una politica ottima



## Come Apprendere $Q()$ ?

Notare che  $Q$  e  $V^*$  sono intimamente correlate:

$$V^*(s) = \max_{a'} Q(s, a')$$

Ciò permette di scrivere  $Q$  RICORSIVAMENTE come

$$\begin{aligned} Q(s_t, a_t) &= r(s_t, a_t) + \gamma V^*(\delta(s_t, a_t)) \\ &= r(s_t, a_t) + \gamma \max_{a'} Q(s_{t+1}, a') \end{aligned}$$

Bene! Poniamo  $\hat{Q}$  essere la funzione corrente appresa dall'agente che approssima  $Q$ .

Consideriamo la seguente regola di apprendimento:

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a')$$

dove  $s'$  è lo stato risultante dalla applicazione della azione  $a$  allo stato  $s$

## Algoritmo $Q$ – *Learning*

1. **per ogni**  $s, a$  inizializza la entry della tabella  $\hat{Q}(s, a) \leftarrow 0$
2. osserva lo stato corrente  $s$
3. **esegui per sempre**
  - (a) seleziona una azione  $a$  ed eseguila
  - (b) ricevi la ricompensa immediata  $r$
  - (c) osserva il nuovo stato  $s'$
  - (d) aggiorna la entry  $\hat{Q}(s, a)$  come segue:

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a')$$

- (e)  $s \leftarrow s'$

## Algoritmo $Q$ – *Learning*

1. per ogni  $s, a$  inizializza la entry della tabella  $\hat{Q}(s, a) \leftarrow 0$
2. osserva lo stato corrente  $s$
3. esegui per sempre
  - (a) seleziona una azione  $a$  ed eseguila come selezionarla ?
  - (b) ricevi la ricompensa immediata  $r$
  - (c) osserva il nuovo stato  $s'$
  - (d) aggiorna la entry  $\hat{Q}(s, a)$  come segue:

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a')$$

- (e)  $s \leftarrow s'$

## Algoritmo $Q$ – Learning

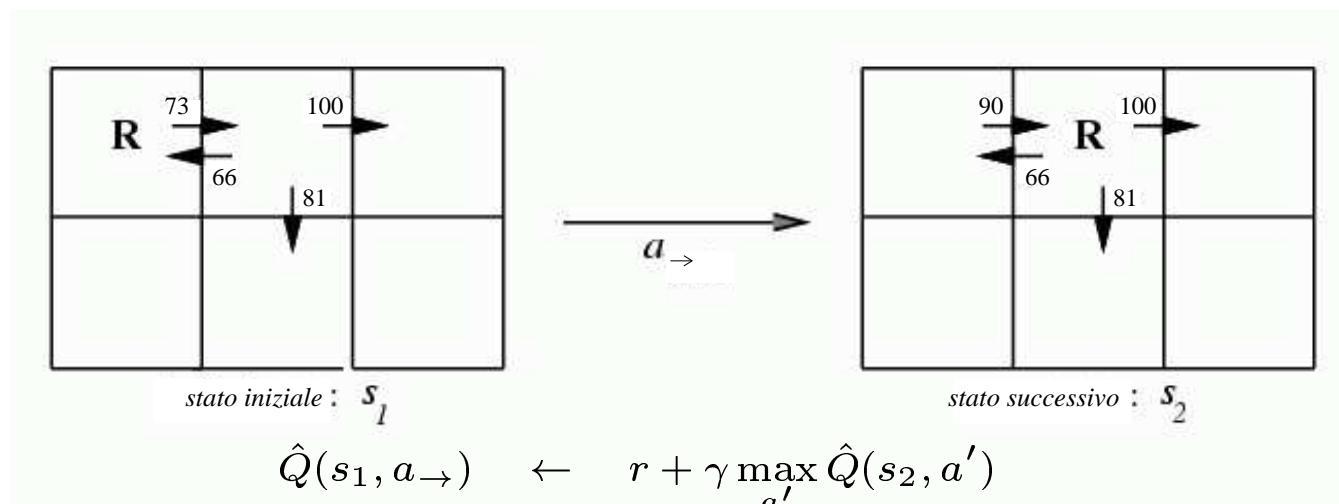
1. per ogni  $s, a$  inizializza la entry della tabella  $\hat{Q}(s, a) \leftarrow 0$
2. osserva lo stato corrente  $s$
3. esegui per sempre

- (a) seleziona una azione  $a \dots$  strategia
- random (esplorazione)
- $\arg \max_a \hat{Q}(s, a)$  (sfruttamento)
- (b) ricevi la ricompensa immediata  $r$
- (c) osserva il nuovo stato  $s'$
- (d) aggiorna la entry  $\hat{Q}(s, a)$  come segue:

$$\hat{Q}(s, a) \leftarrow r + \gamma \max_{a'} \hat{Q}(s', a')$$

- (e)  $s \leftarrow s'$

## Esempio di Applicazione



notare che se le ricompense sono non-negative, allora

$$(\forall s, a, n) \hat{Q}_{n+1}(s, a) \geq \hat{Q}_n(s, a)$$

e

$$(\forall s, a, n) 0 \leq \hat{Q}_n(s, a) \leq Q(s, a)$$

## Convergenza

$\hat{Q}$  converge a  $Q$

Consideriamo il caso deterministico dove ogni  $(s, a)$  è visitato un numero infinito di volte

*Prova:* Definiamo un intervallo pieno un intervallo durante il quale ogni  $(s, a)$  è visitato.

Durante ogni intervallo pieno l'errore più grande nella tabella  $\hat{Q}$  è ridotto di un fattore  $\gamma$

Infatti, sia  $\hat{Q}_n$  la tabella dopo  $n$  aggiornamenti, e  $\Delta_n$  l'errore massimo in  $\hat{Q}_n$ , cioè

$$\Delta_n = \max_{s,a} |\hat{Q}_n(s, a) - Q(s, a)|$$

Per ogni entry della tabella  $\hat{Q}_n(s, a)$  aggiornata alla iterazione  $n + 1$ , l'errore nella stima rivista  $\hat{Q}_{n+1}(s, a)$  è

$$\begin{aligned}
|\hat{Q}_{n+1}(s, a) - Q(s, a)| &= |(r + \gamma \max_{a'} \hat{Q}_n(s', a')) \\
&\quad - (r + \gamma \max_{a'} Q(s', a'))| \\
&= \gamma |\max_{a'} \hat{Q}_n(s', a') - \max_{a'} Q(s', a')| \\
&\leq \gamma \max_{a'} |\hat{Q}_n(s', a') - Q(s', a')| \\
&\leq \gamma \max_{s'', a'} |\hat{Q}_n(s'', a') - Q(s'', a')| \\
|\hat{Q}_{n+1}(s, a) - Q(s, a)| &\leq \gamma \Delta_n
\end{aligned}$$

Notare che abbiamo usato il risultato generale

$$|\max_a f_1(a) - \max_a f_2(a)| \leq \max_a |f_1(a) - f_2(a)|$$

## Caso NONDETERMINISTICO

Si ridefiniscono  $V, Q$  considerando i valori aspettati

$$\begin{aligned} V^\pi(s) &\equiv E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots] \\ &\equiv E\left[\sum_{i=0}^{\infty} \gamma^i r_{t+i}\right] \end{aligned}$$

$$Q(s, a) \equiv E[r(s, a) + \gamma V^*(\delta(s, a))]$$

Per l'apprendimento, sostituire la regola di aggiornamento con

$$\hat{Q}_n(s, a) \leftarrow (1 - \alpha_n) \hat{Q}_{n-1}(s, a) + \alpha_n [r + \max_{a'} \hat{Q}_{n-1}(s', a')] \quad \text{dove } \alpha_n = \frac{1}{1 + \text{visite}_n(s, a)}$$

Sotto determinate condizioni si dimostra la convergenza