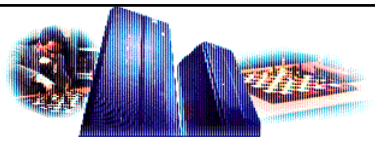
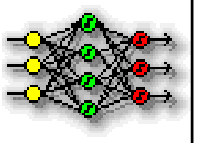




Sistemi Intelligenti
Reinforcement Learning: Q-learning

Alberto Borghese

Università degli Studi di Milano
Laboratorio di Sistemi Intelligenti Applicati (AIS-Lab)
Dipartimento di Scienze dell'Informazione
borghese@dsi.unimi.it

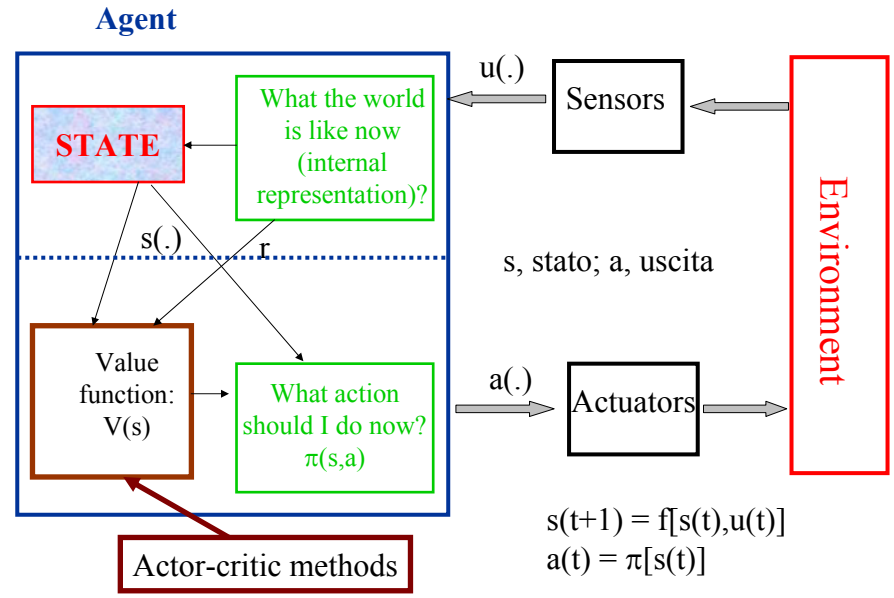
Sommario

Calcolo ricorsivo della Q function

Q-learning

Esempio

Schematic diagram of an agent



How About Learning the Value Function?

Facciamo imparare all'agente la value function, per una certa politica: V^{π}:

$$V^{\pi}(s) = \left[\sum_{a_j} \pi(a_j, s)\right] \sum_{s'} P_{s \to s'|a_j} \left[R_{s \to s'|a_j} + \gamma V^{\pi}(s')\right]$$

È una funzione dello stato.

Una volta imparata la value function, V^*, l'agente seleziona la policy ottima passo per passo, "one step lookahead":

$$\pi^*(s) = \arg\max_a \sum_{s'} P^a_{s->s'} \left[R^a_{s->s'} + \gamma V^{\pi}(s')\right]$$

Full backup, for all states

Value function iteration

Facciamo imparare all'agente la value function, per una certa politica: V^{π}, analizzando quello che succede in uno step temporale:

$$V^{\pi}{}_{k+1}(s) = \left[\sum_{a_j} \pi(a_j, s)\right] \sum_{s'} P_{s \to s'|a_j} \left[R_{s \to s'|a_j} + \gamma V^{\pi}{}_k(s')\right]$$

L'apprendimento della policy si può inglobare nella value iteration:

$$V_{k+1}(s) = \max_a \sum_{s'} P_{s \to s'|a} \left[R_{s \to s'|a} + \gamma V_k(s')\right]$$

Full backup, for all states

Asynchronous DP

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P_{s \to s'|a} \left[R_{s \to s'|a} + \gamma V_k(s')\right]$$

Full backup, single state, s, all future states s'

+ policy improvement

Temporal Differences

Viene rimossa ogni conoscenza a-priori sull'ambiente.

Value iteration and policy iteration are merged into one update equation:

$$V^{\pi}(s_t) = V^{\pi}(s_t) + \alpha\left[r_{t+1} + \gamma V^{\pi}(s_{t+1}) - V^{\pi}(s_t)\right]$$

Single backup, single state, s_t, single future state s_{t+1}

+ policy improvement

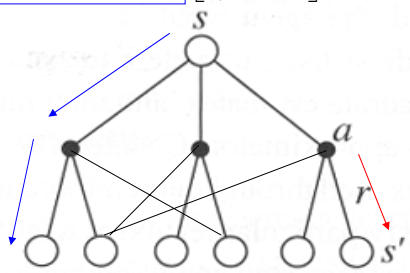
Serve davvero la Value Function?

La Value Function deriva dalla visione della Programmazione Dinamica.

Ma è proprio necessario conoscere la Value function? In fondo a noi interessa determinare la Policy.

Le value function

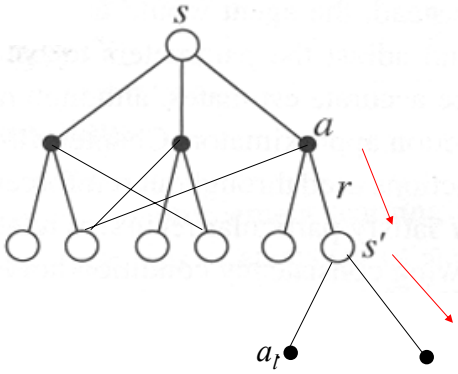
$$V^{\pi}(s) = E_{\pi}\{R_t \mid s_t = s\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \,\middle|\, s_t = s\right\} = \left[\sum_{a_j} \pi(a_j, s)\right] \sum_{s'} P_{s \to s' | a_j} \left[R_{s \to s' | a_j} + \gamma V^{\pi}(s')\right]$$



$$Q^{\pi}(s,a) = E_{\pi}\{R_t \mid s_t = s,\ a_t = a\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \,\middle|\, s_t = s, a_t = a\right\}$$

$$= \sum_{s'} P_{s \to s' | a_j} \left[R_{s \to s' | a_j} + \gamma V^{\pi}(s')\right]$$

Calcolo ricorsivo della value function Q



$$Q^{\pi}(s,a) = E_{\pi}\{R_t \mid s_t = s,\ a_t = a\} = E_{\pi}\left\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \,\middle|\, s_t = s, a_t = a\right\}$$

$$= \sum_{s'} P_{s \to s' | a} \left[R_{s \to s' | a} + \gamma \sum_{l} \pi(s', a_l) Q^{\pi}(s', a_l)\right]$$

Sommario

Calcolo ricorsivo della Q function

Q-learning

Esempio

Q Functions

$$\pi^*(s) = \arg\max_a \sum_{s'} P^a_{s->s'}\left[R^a_{s->s'} + \gamma V^\pi(s')\right] \qquad Q^\pi(s,a_j) = \sum_{s'} P_{s\to s'|a_j}\left[R_{s\to s'|a_j} + \gamma V^\pi(s')\right]$$

V = Cumulative reward of being in *s*.

Idea chiave:

- Unire il rinforzo che si ottiene passando da uno stato al successivo in un'unica funzione

$$Q(s,a) = \left[R^a_{s->s'} + \gamma V^\pi(s')\right]$$

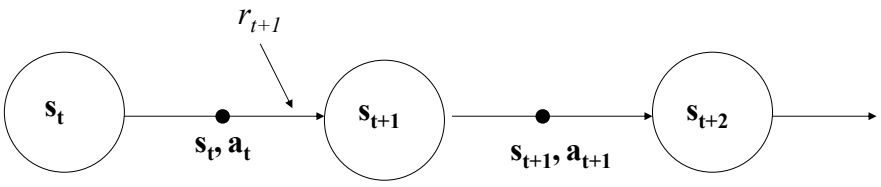
- Questa funzione valuta la bontà dell'azione e non più dello stato (a = π(s)).

- A questo punto posso massimizzare Q senza conoscere separatamente il reward istantaneo e la value function come:

$$\pi^*(s) = \text{argmax}_a\, Q(s,a)$$

Q = Cumulative reward of being in *s* and taking action *a*.

Rappresentazione grafica



$V(s_t)$ $V(s_{t+1})$

One step for Value Iteration

$Q(s_t, a_t)$ $Q(s_{t+1}, a_{t+1})$

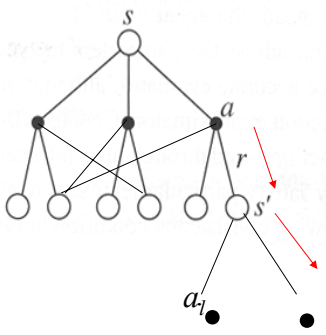
One step for **Q Iteration**

Come apprendere Q: SARSA

$$Q(s_t, a_t) = Q(s_t, a_t) + \alpha\left[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)\right]$$

1) Apprendiamo il valore di Q per una policy data (on-policy).

2) Dopo avere appreso la funzione Q, possiamo modificare la policy in modo da migliorarla.



SARSA Algorithm (progetto)

```
Q(s,a) = rand(); ∀s, ∀a
Repeat                          // for each episode
{       s = s0;
        Repeat                  // for each step of the single episode
        {
                a = Policy(s);          // eventualmente ε-greedy
                s_next = NextState(s,a);
                reward = Reward(s,s_next,a);
                a_next = Policy(s_next);        // ε-greedy?
                Q(s,a) = Q(s,a) + α [reward + γ Q(s_next, a_next) – Q(s,a)];
                s = s_next;
                        // a = a_next;
        }               // until last state
}                       // until the end of learning
```

1) Apprendiamo il valore di Q per una policy data (on-policy).
2) Dopo avere appreso la funzione Q, possiamo modificare la policy in modo da migliorarla.

Come integrare i due passi?

Off-policy Temporal Difference: Q-learning

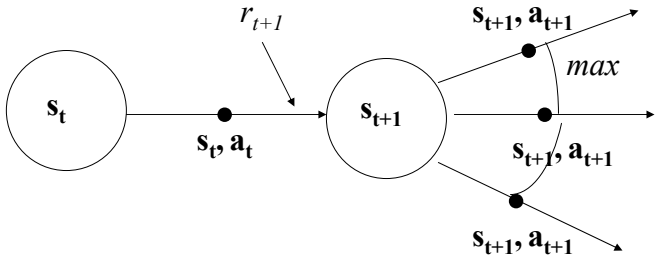
$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha\left[r_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)\right]$$

Non imparo semplicemente la funzione valore Q, ma la funzione valore Q ottima.

Q-learning algorithm (progetto)

```
Q(s,a) = rand(); ∀s, ∀a
Repeat                          // for each episode
{       s = s0;
        Repeat                  // for each step of the single episode
        {
                a = Policy(s);          // eventualmente ε-greedy
                s_next = NextState(s,a);
                reward = Reward(s,s_next,a);
                a_next = Policy(s_next);        // ε-greedy?
                Q(s,a) = Q(s,a) + α [reward + γ max Q(s_next, a_next) – Q(s,a)];
                s = s_next;
                UpdatePolicy(a_next, s_next);
        }               // until last state
}                       // until the end of learning
```


Rappresentazione grafica



$Q(s_t, a_t)$ $Q(s_{t+1}, a_{t+1})$

One step for **Q Iteration**

Sommario

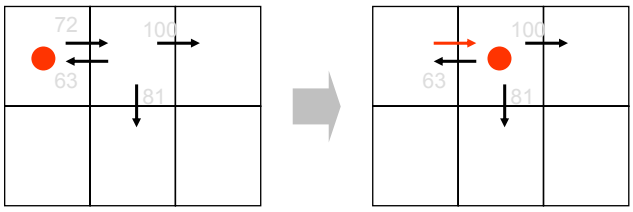
Dal modello dell'ambiente all'esperienza

Q-learning

Esempio

Example – Q Learning Update

$\gamma = 0.9$

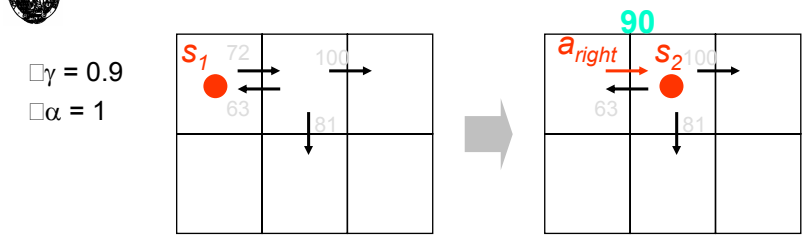


0 reward received

Esempio tratto dai lucidi del corso di Brian C. Williams su RL.
Modificati dalle slide di: Manuela Veloso, Reid Simmons, & Tom Mitchell, CMU

Apprendimento della funzione valore Q. Versione SARSA.

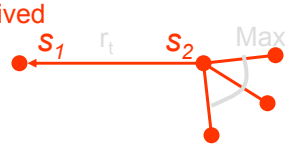
Example – Q Learning Update



$\gamma = 0.9$

$\alpha = 1$

0 reward received



$\underline{Q}(s_1,a_{right}) \leftarrow r(s_1,a_{right}) + \gamma \max_{a'} \underline{Q}(s_2,a')$

$\leftarrow 0 + 0.9 \max \{63, 81, 100\}$

$\leftarrow 90$

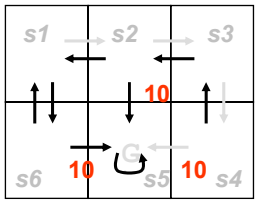
In grigio i valori di Q(s,a). Nessun reward istantaneo.
Correzione di $\underline{Q}(s_1,a_{right})$

Q-Learning Iterations: Episodic

- Start at upper left – move clockwise; table initially 0; $\gamma = 0.8$

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

Reward istanteo in rosso

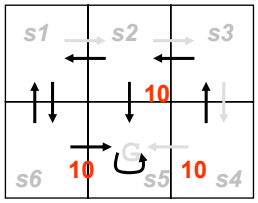


Q(S1,E)	Q(s2,E)	Q(s3,S)	Q(s4,W)
0			

Q-Learning Iterations: Episodic

- Start at upper left – move clockwise; table initially 0; $\gamma = 0.8$

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

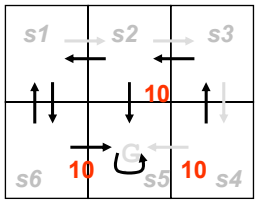


Q(S1,E)	Q(s2,E)	Q(s3,S)	Q(s4,W)
0	0	0	

Q-Learning Iterations

- Start at upper left – move clockwise; $\gamma = 0.8$

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

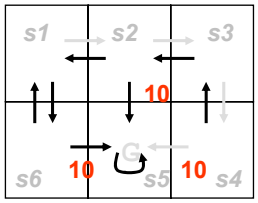


Q(S1,E)	Q(s2,E)	Q(s3,S)	Q(s4,W)
0	0	0	$r + \gamma \max_{a'} \{\underline{Q}(s5,loop)\} =$ $10 + 0.8 \times 0 = \mathbf{10}$

Q-Learning Iterations

- Start at upper left – move clockwise; $\gamma = 0.8$

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

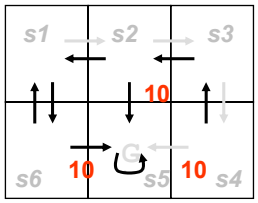


Q(S1,E)	Q(s2,E)	Q(s3,S)	Q(s4,W)
0	0	0	$r + \gamma \max_{a'} \{\underline{Q}(s5,loop)\} =$ 10 + 0.8 x 0 = 10
0	0	$r + \gamma \max_{a'} \{\underline{Q}(s4,W), \underline{Q}(s4,N)\} = 0 +$ 0.8 x max{10,0) = 8	

Q-Learning Iterations

- Start at upper left – move clockwise; $\gamma = 0.8$

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

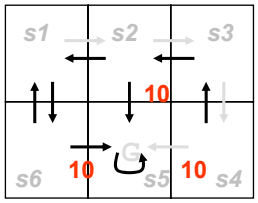


Q(S1,E)	Q(s2,E)	Q(s3,S)	Q(s4,W)
0	0	0	$r + \gamma \max_{a'} \{\underline{Q}(s5,loop)\} =$ 10 + 0.8 x 0 = 10
0	0	$r + \gamma \max_{a'} \{\underline{Q}(s4,W), \underline{Q}(s4,N)\} = 0 +$ 0.8 x max{10,0) = 8	10
0	$r + \gamma \max_{a'} \{\underline{Q}(s3,W), \underline{Q}(s3,S)\} = 0 +$ 0.8 x max{0,8) = 6.4		

Q-Learning Iterations

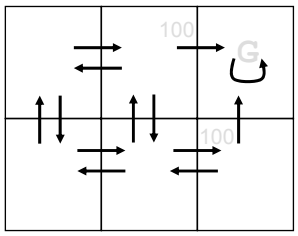
- Start at upper left – move clockwise; $\gamma = 0.8$

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

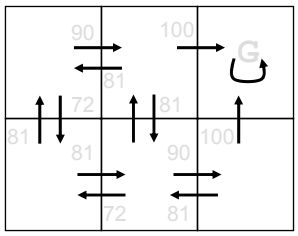


Q(S1,E)	Q(s2,E)	Q(s3,S)	Q(s4,W)
0	0	0	$r + \gamma \max_{a'} \{\underline{Q}(s5,loop)\} =$ 10 + 0.8 x 0 = 10
0	0	$r + \gamma \max_{a'} \{\underline{Q}(s4,W), \underline{Q}(s4,N)\} = 0 +$ 0.8 x max{10,0) = 8	10
0	$r + \gamma \max_{a'} \{\underline{Q}(s3,W), \underline{Q}(s3,S)\} = 0 +$ 0.8 x max{0,8) = 6.4	8	10

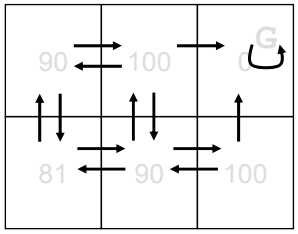
Example Summary: Value Iteration and Q-Learning



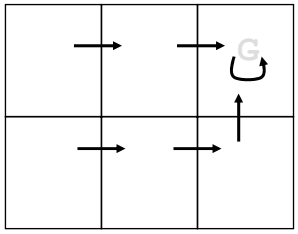
R(*s*, *a*) values



Q(*s*, *a*) values



V*(*s*) values



One Optimal Policy

Sommario

Dal modello dell'ambiente all'esperienza

Q-learning

Esempio