

# Sistemi Intelligenti Reinforcement Learning: Iterative policy evaluation

Alberto Borghese

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



A.A. 2008-2009

1/32

<http://homes.dsi.unimi.it/~borghese/>



## Sommario



### Determinazione ricorsiva della Value function

Dalle equazioni di Bellman alla iterative policy evaluation.

Determinazione della policy ottima.

A.A. 2008-2009

2/32

<http://homes.dsi.unimi.it/~borghese/>



## Esempio di calcolo della Value function



Value function

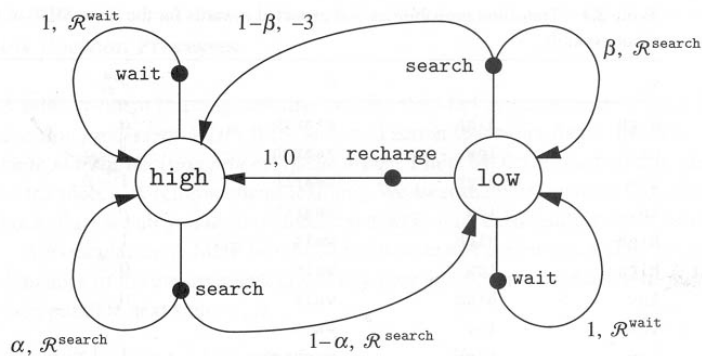
$V(\text{high}) = ?$

$V(\text{low}) = ?$

Policy

$a(s=\text{high}) = ?$

$a(s=\text{low}) = ?$



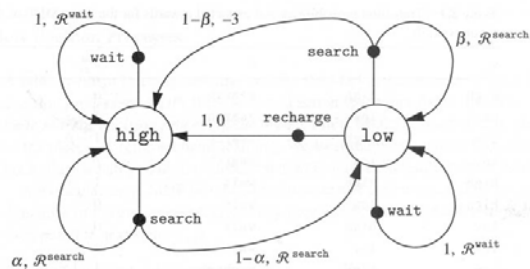
A.A. 2008-2009

3/32

<http://homes.dsi.unimi.it/~borghese/>



## Esempio di calcolo della funzione valore



$\alpha=0.4, \beta=0.1, \gamma=0.8, \mathcal{R}^{\text{search}}=3, \mathcal{R}^{\text{wait}}=1$

$$V_h = \underbrace{\text{Pr}(W)}_{\text{Wait}} \times 1 \times [1 + 0.8V_h] + \underbrace{\text{Pr}(S)}_{\text{Search} \rightarrow \text{high}} \times 0.4 \times [3 + 0.8V_h] + \underbrace{\text{Pr}(S)}_{\text{Search} \rightarrow \text{low}} \times 0.6 \times [3 + 0.8V_l]$$

$$V_l = \underbrace{\text{Pr}(W)}_{\text{Wait}} \times 1 \times [1 + 0.8V_l] + \underbrace{\text{Pr}(S)}_{\text{Search} \rightarrow \text{Low}} \times 0.1 \times [3 + 0.8V_l] + \underbrace{\text{Pr}(S)}_{\text{Search} \rightarrow \text{High}} \times 0.9 \times [-3 + 0.8V_h] + \underbrace{\text{Pr}(R)}_{\text{Recharge}} \times 1 \times [0 + 0.8V_h]$$

**Sistema lineare di 2 equazioni nelle 2 incognite:  $V_h$  e  $V_l$**   
**Come calcolo  $V_h$  e  $V_l$ ?**

A.A. 2008-2009

4/32

<http://homes.dsi.unimi.it/~borghese/>



# Calcolo ricorsivo della Value function



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

$$V^\pi(s') = E_\pi \{R_{t+1} | s_{t+1} = s'\}$$

Legame?

Policy Next-state

$$P_{s \rightarrow s'|a} = \Pr\{s_{t+1} = s' | s_t = s, a_t = a\}$$

$$V^\pi(s) = \sum_{a_j} \pi(a_j, s) \sum_{s'} P_{s \rightarrow s'|a_j} R_{s \rightarrow s'|a_j} + E_\pi \left\{ \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \mid s_t = s \right\}$$

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

Bellman's equation

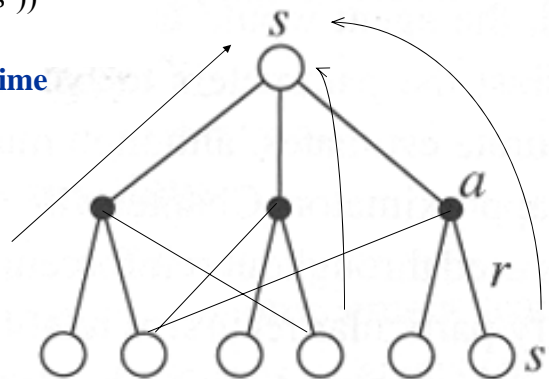


# Osservazioni



$$V^\pi(s) = \text{funz}(V^\pi(s'))$$

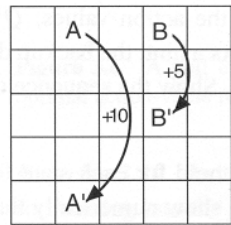
Backwards in time



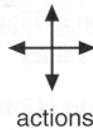
$$V^\pi(s) = \left\{ \sum_{a_j} \pi(a_j, s) \sum_{s'} \left\{ P_{s \rightarrow s'|a_j} \left[ R_{s \rightarrow s'|a_j} + \gamma V^\pi(s') \right] \right\} \right\}$$



## Esempio ed calcolo della funzione valore (possibile progetto)



(a)



3.3	8.8	4.4	5.3	1.5
1.5	3.0	2.3	1.9	0.5
0.1	0.7	0.7	0.4	-0.4
-1.0	-0.4	-0.4	-0.6	-1.2
-1.9	-1.3	-1.2	-1.4	-2.0

(b)

**Figure 3.5** Grid example: (a) exceptional reward dynamics; (b) state-value function for the equiprobable random policy.

$a = \pi(s)$  equiprobabile =  $0.25 \forall a$

$R(s,a)$  per stati sui bordi e azioni che portano fuori dal bordo.

$R(s)$  per tutti gli altri stati (= 0, +5, +10)



## Calcolo della funzione valore di uno stato



$t: s = s_t = C$

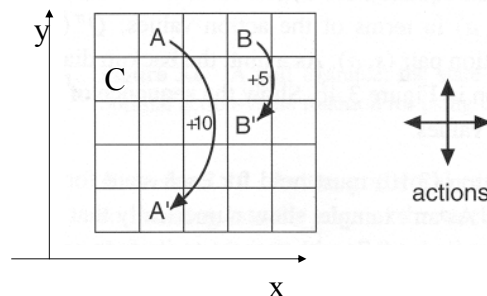
$t + 1: s = s' = s_{t+1}$

I  $C + Dy$   $p = 0.25$   $r = 0$

II  $C - Dy$   $p = 0.25$   $r = 0$

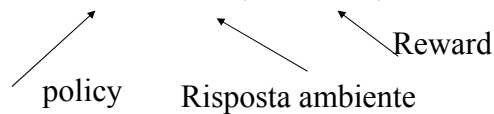
III  $C + Dx$   $p = 0.25$   $r = 0$

IV  $C$   $p = 0.25$   $r = -1$



Per calcolare il valore di  $s: V(s)$ , devo analizzare il valore di ogni  $s'$ :

$\pi(s,a) \rightarrow P_{s \rightarrow s' | a} \rightarrow R_{s \rightarrow s' | a} \rightarrow V^\pi(s')$  ← Value of  $s'$ .





## Esempio di valore di uno stato



$t: s = s_t = C$

$t + 1: s = s' = s_{t+1}$

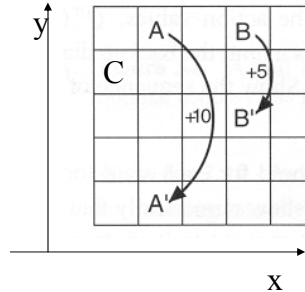
$C + Dy \quad p = 0.25 \quad r = 0$

$C - Dy \quad p = 0.25 \quad r = 0$

$C + Dx \quad p = 0.25 \quad r = 0$

$C \quad p = 0.25 \quad r = -1$

$r_{t+1} = -0.25$



$t + 2, a = +Dy$

$C + Dy \rightarrow C + Dy \quad p = 0.25 \quad r = -1$

$C - Dy \rightarrow C \quad p = 0.25 \quad r = 0$

$C + Dx \rightarrow A \quad p = 0.25 \quad r = 0$

$C \rightarrow C + Dy \quad p = 0.25 \quad r = -1$

$r_{t+1} = -0.50$

$a = -Dy$

$\rightarrow C \quad r = 0$

$\rightarrow C - 2Dy \quad r = 0$

$\rightarrow C - Dy + Dx \quad r = 0$

$\rightarrow C - Dy \quad r = -1$

$r_{t+1} = -0.25$

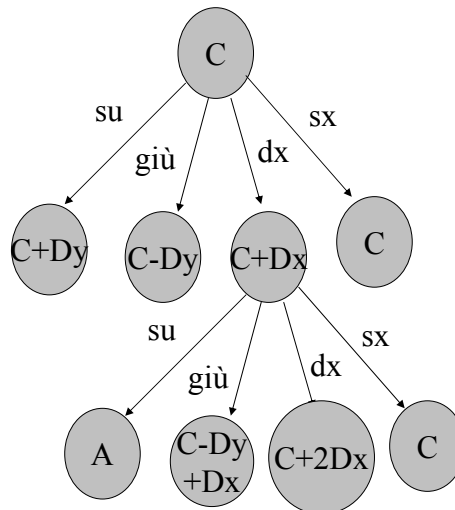
A.A. 2008-2009

9/32

<http://homes.dsi.unimi.it/~borghese/>



## Sviluppo parziale a 2 passi



A.A. 2008-2009

10/32

<http://homes.dsi.unimi.it/~borghese/>



## Problematiche legate al calcolo di $V(s)$ : problema di policy evaluation



3 assunzioni:

- 1) Conoscenza della dinamica dell'ambiente:  $P(s \rightarrow s' | a_t)$
- 2) Conoscenza della policy (eventualmente stocastica),  $\pi(s, a)$
- 3) Potenza di calcolo sufficiente
- 4) Proprietà Markoviane dell'ambiente (definizione di uno stato).

Le equazioni contengono dei termini statistici (valori attesi).

Soluzione di un sistema lineare in  $N$  incognite (numero di stati).

Come mai posso determinare la Value function per la policy  $p()$ , se questa si basa sul reward che riceverò negli istanti futuri?

C'è poca interazione con l'ambiente e molta simulazione (cf. metodi Montecarlo).



## La Value function sulle azioni



La value function può riguardare le azioni.

Action-Value function

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

Massimizzo la ricompensa a lungo termine,  $Q(\cdot)$ . Dipende dalla policy.

Dove abbiamo già trovato una Value function associata alle azioni?



## Sommario



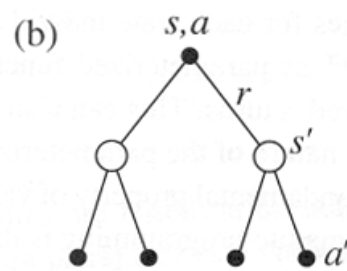
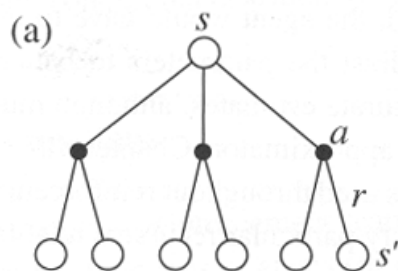
Determinazione ricorsiva della Value function

Dalle equazioni di Bellman alla iterative policy evaluation.

Determinazione della policy ottima.



## Policy



La policy deve essere ancora determinata. Come fa l'agente a determinare la policy ottimale?

Archi multipli fuoriuscenti da un'azione sono associati alla probabilità di scegliere quel cammino (ambiente stocastico).

Archi multipli fuoriuscenti da uno stato, sono associati alla policy.



## Ottimizzazione policy



Per ogni stato scelgo le azioni secondo la policy:  $\pi(s,a)$ .

Posso ordinare la Value function  $V(s)$  in funzione delle azioni scelte in  $s$  (policy).

Si definisce una policy,  $\pi_1$ , migliore di un'altra,  $\pi_2$ , se e solo se:

$$V^{\pi_1}(s) \geq V^{\pi_2}(s) \quad \forall s.$$

In particolare si definisce una politica ottima,  $\pi^*$ , se e solo se:

$$V^*(s) \geq V^{\pi}(s) \quad \forall s$$

$$Q^*(s,a) \geq Q^{\pi}(s,a) \quad \forall [s,a]$$



## Riassunto



Posso determinare la Value function in modo ricorsivo. Per ogni stato, sarà funzione dell'output dell'ambiente in quell'istante (attraverso la funzione stato prossimo ed il reward istantaneo) e della policy scelta in quell'istante e dei reward a lungo termine attesi negli stati in cui l'ambiente mi porta.

Per scegliere la policy devo esaminare il reward a lungo termine che mi si prospetta nello stato in cui mi trovo e scegliere l'azione che lo massimizza.





## Iterative policy evaluation



Evoluzione del sistema da  $s(t=0)$  a  $\{s'(t = T)\}$  utilizzando la policy  $\pi(s,a)$ , prefissata.

Quanto valgono gli stati?

Parto da  $V(s(t=0))_{k=0}$  arbitraria, otterrò una value function per ogni stato che sarà funzione di  $V(s(t=0))$ .

Devo migliorare, come?

Utilizziamo l'informazione sul **passato**.



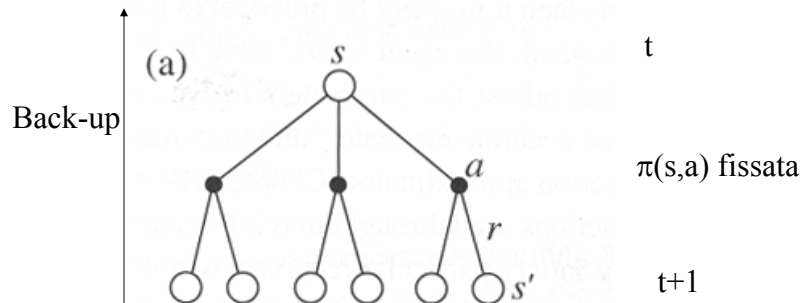
## Fondamenti del metodo



- Supponiamo di essere all'istante  $t$ . In questo istante  $t$ , si può passare ad un certo insieme di stati:  $\{s'_{t+1}\}$ .
- Analizziamo un solo passo: cosa succede nella transizione da  $t$  a  $t+1$ .
- Migliorare la stima della nostra Value Function ad ogni iterazione.



## Tecnica full-backup



Conosciamo  $V_k(s(t)) \forall s$ , anche per  $s'$  quindi  
 Analizziamo la transizione da  $s(t) \rightarrow \{s'(t+1)\}$   
 Calcoliamo un nuovo valore di  $s$ :  $V_{k+1}(s(t))$  congruente con  $V_k(s(t))$  ed  $r_{t+1}$   
 Full backup se esaminiamo tutti gli  $s'$  (cf. DP).

Da  $s'$  mi guardo indietro ed aggiorno  $V(s)$ .



## Calcolo iterativo della Value Function



Per ogni stato  $s$ , estratto a caso, analizziamo una singola transizione.

Equazione di Bellman per “iterative policy evaluation”:

$$V_{k+1}(s) = \left[ \sum_{a_j} \pi(a_j, s) \right] \sum_{s'} P_{s \rightarrow s' | a_j} \left[ R_{s \rightarrow s' | a_j} + \gamma V_k(s') \right]$$

Mi fido di  $V_k(s')$  (Backup)

$$\lim_{k \rightarrow \infty} \{V_k(s)\} = V^\pi(s)$$



## Algoritmo per "iterative policy evaluation"



Partiamo da una politica  $\pi(s,a)$  data.

Inizializziamo  $V(s) = 0 \forall s$ , compreso gli stati finali.

Repeat

```
{
  Δ = 0;
  for s = 1 : N // ∀s, ≠ TS
  {
    Value = V(s);
```

$$V_{k+1}(s) = \sum_{a_j} \pi(s, a_j) \sum_{s'} P_{s \rightarrow s'}^{a_j} [R_{s \rightarrow s'}^{a_j} + \gamma V(s')]$$

$$\Delta = \max(\Delta, | \text{Value} - V_{k+1}(s) |)$$

```
}
} Until (Δ < threshold);
```



## Esempio



- L'agente evolve esplorando gli stati 1-14. TS sono gli stati terminali.

- L'agente può spostarsi: {dx, sx, alto, basso}

- $\pi(s,a)$  è equiprobabile.

- $R = -1$  per ogni transizione, tranne quando  $s' = \text{TS}$  ( $R = 0$ ).

- $\gamma = 1$ .

TS	1	2	3
4	5	6	7
8	9	10	11
12	13	14	TS



## Esempio - $V(s)_{k=0}$



0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0



## Esempio - $V(s)_{k=1}$



0	0	0	0
0	0	0	0
0	0	0	0
0	0	0	0

$V(s)_{k=0}$

0	-0.75	-1	-1
-0.75	-1	-1	-1
-1	-1	-1	-0.75
-1	-1	-0.75	0

$V(s)_{k=1}$

Esempi:

- $V(s=10)_{k=1} = 0.25 [1 (-1 + 1 V(14))_{k=0}] + 0.25 [1 (-1 + 1 V(6))_{k=0}] + 0.25 [1 (-1 + 1 V(9))_{k=0}] + 0.25 [1 (-1 + 1 V(11))_{k=0}] = -1$
- $V(s=1)_{k=1} = 0.25 [1 (-1 + 1 V(1))_{k=0}] + 0.25 [1 (-1 + 1 V(5))_{k=0}] + 0.25 [1 (0 + 1 V(TS))_{k=0}] + 0.25 [1 (-1 + 1 V(2))_{k=0}] = -0.75$



## Esempio - $V(s)_{k=2}$



0	-0.75	-1	-1
-0.75	-1	-1	-1
-1	-1	-1	-0.75
-1	-1	-0.75	0

$V(s)_{k=1}$

0	-1.4375	-1.9375	-2
-1.4375	-1.875	-2	-1.9375
-1.9375	-2	-1.875	-1.4375
-2	-1.9375	-1.4375	0

$V(s)_{k=2}$

Esempi:

- $V(s=10)_{k=2} = 0.25 [1 (-1 + 1 V(14))_{k=1}] + 0.25 [1 (-1 + 1 V(6))_{k=1}] + 0.25 [1 (-1 + 1 V(9))_{k=1}] + 0.25 [1 (-1 + 1 V(11))_{k=0}] = -1.875$
- $V(s=1)_{k=2} = 0.25 [1 (-1 + 1 V(1))_{k=1}] + 0.25 [1 (-1 + 1 V(5))_{k=1}] + 0.25 [1 (0 + 1 V(TS))_{k=1}] + 0.25 [1 (-1 + 1 V(2))_{k=1}] = -1.4375$



## Interpretazione dell'update (batch o trial)



$$V_{\text{new}}(s) = \sum_{a_j} \pi(s, a_j) \sum_{s'} P_{s \rightarrow s'}^{a_j} [R_{s \rightarrow s'}^{a_j} + \gamma V(s')]$$

Al termine dell'aggiornamento dei  $V(s)$  per tutti gli stati,  $V(s) = V_{\text{new}}(s)$ . **Aggiornamento batch.**

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

Utilizzerò in parte già il nuovo valore di  $V(s)$  all'interno dell'equazione di aggiornamento. **Aggiornamento per trial.**

Entrambe le modalità di aggiornamento convergono.



# Sommario



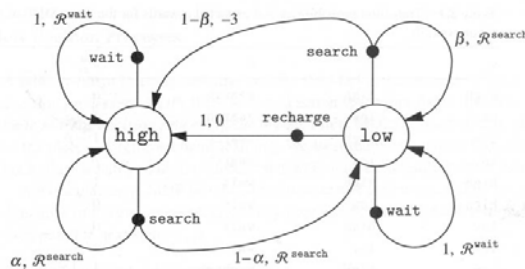
Determinazione ricorsiva della Value function

Dalle equazioni di Bellman alla iterative policy evaluation.

Determinazione della policy ottima.



# Esempio di calcolo di $V^*(s)$ - I



$\alpha = 0.4, \beta = 0.1, \gamma = 0.8, R_{wait} = 1, R_{search} = 3$

$V(\text{high}):$

state = high, action = wait  $\rightarrow Q(\text{high}, \text{wait}) = 1[R_{wait} + \gamma V^*(\text{high})]$

state = high, action = search  $\rightarrow Q(\text{high}, \text{search}) = \alpha[R_{search} + \gamma V^*(\text{high})] + (1-\alpha)[R_{search} + \gamma V^*(\text{low})]$

$V(\text{low}):$

state = low, action = wait  $\rightarrow Q(\text{low}, \text{wait}) = [R_{wait} + \gamma V^*(\text{low})]$

state = low, action = search  $\rightarrow Q(\text{low}, \text{search}) = \beta[R_{search} + \gamma V^*(\text{low})] + (1-\beta)[-3 + \gamma V^*(\text{high})]$

state = low, action = rechar  $\rightarrow Q(\text{low}, \text{rechar}) = \gamma V^*(h)$



## Esempio di calcolo di $V^*(s)$ - II



$s = \text{HighEnergy}$        $a = \text{Search}; V_h \approx 6 + 0.65V^*l$   
                                   $a = \text{Wait}; V_h = 5$  (per ogni  $V^*l$ )

$s = \text{Low}$                  $a = \text{Search}; V_l \approx -2.2 + 6.6 V^*h$   
                                   $a = \text{Wait}; V_l = 5$   
                                   $a = \text{Recharge}; V_l = 0.8 V^*h$

Come calcolo  $V^*h$  e  $V^*l$ ?



## Esempio di calcolo di $V^*(s)$ - III



$s = \text{HighEnergy}$        $a = \text{Search}; V_h \approx 6 + 0.65V^*l$   
                                   $a = \text{Wait}; V_h = 5$  (per ogni  $V^*l$ )

$s = \text{Low}$                  $a = \text{Search}; V_l \approx -2.2 + 6.6 V^*h$   
                                   $a = \text{Wait}; V_l = 5$   
                                   $a = \text{Recharge}; V_l = 0.8 V^*h$

Scelgo  $\underset{a}{Max}$  sia per lo stato High Energy che Low Energy

Sistema non-lineare..



## Esempio di calcolo di $V^*(s)$ - IV



$s = \text{HighEnergy}$	$a = \text{Search};$	$V_h \approx 6 + 0.65V_l \approx 9.25$
$s = \text{Low}$	$a = \text{Wait};$	$V_l = 5$
$s = \text{HighEnergy}$	$a = \text{Search};$	$V_h \approx 6 + 0.65V_l \approx 12$
$s = \text{Low}$	$a = \text{Recharge};$	$V_l = 0.8 * V_h = 9.6$
$s = \text{HighEnergy}$	$a = \text{Wait};$	$V_h \approx 5$
$s = \text{Low}$	$a = \text{Search};$	$V_l = -2.2 + 6.6V_h \approx 33$

Serve comunque una valutazione globale di  $V(s)$  per i vari stati.

La soluzione dipende dal valore dei parametri.



## Sommario



Determinazione ricorsiva della Value function

Dalle equazioni di Bellman alla iterative policy evaluation.

Determinazione della policy ottima.