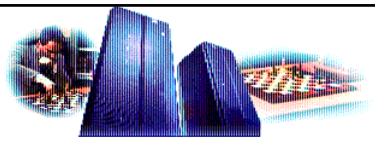
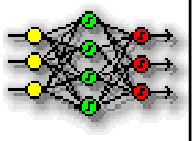




Sistemi Intelligenti
Reinforcement Learning: Equazioni di Bellman

Alberto Borghese

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

Sommario

Le equazioni di Bellman

Value function ottima

Il modello markoviano

Il comportamento dell'ambiente è definito dallo stato: S = $\{s_j\}$
Per ogni stato l'agente sceglie un'azione: a = a(s) A = $\{a_k\}$
Policy di un agente: π(s, a) è quanto dobbiamo definire.

L'ambiente ha una evoluzione stocastica rappresentata da un MDP:

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

Inoltre, ad ogni istante fornisce un reward immediato associato alla transizione, stimato all'istante t come:

$$R_{s_t=s->s_{t+1}=s'|a_t=a} = E\{r_{t+1}=r' | s_t=s, a_t=a, s_{t+1}=s'\}$$

$\forall s \in S$; $\forall a \in A$

La value function

Nulla è detto sulla policy: dato uno stato, quale azione scegliere? In quale stato mi sposto?

Vogliamo costruire agenti lungimiranti.

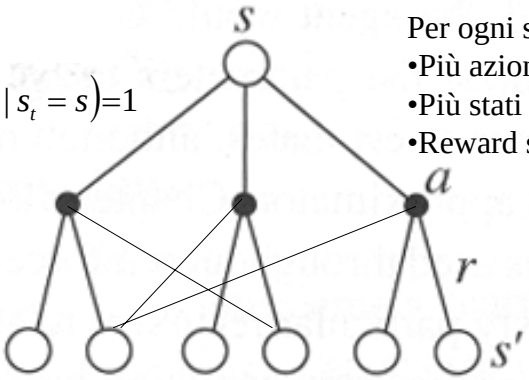
State-Value function:

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

Massimizzo la ricompensa a lungo termine, V(.). Dipende dalla policy:

Value function e modelli markoviani

$$\sum_{j=1}^{N\_azioni} \Pr(a_j \mid s_t = s) = 1$$



Per ogni stato devo valutare:
- Più azioni.
- Più stati prossimi
- Reward stocastici.

$$\sum_{k=1}^{N\_stati} \Pr(s_{t+1} = s_k \mid s_t = s; a_t = a_j) = 1$$

Esempio: AIBO search

Azioni:
1) Rimanere fermo e aspettare che qualcuno getti nel cestino una lattina vuota.
2) Muoversi attivamente in cerca di lattine.
3) Tornare alla sua base (recharge station) e ricaricarsi.

Stato:
1) Alto livello di energia.
2) Basso livello di energia.

Policy:
A(s = high) = {Search, Wait}
A(s = low) = {Search, Wait, Recharge}

Task: collezionare il maggior numero di lattine.

Funzionamento del Robot

Funzione Stato prossimo: $P_{s->s'|a} = \Pr\{s_{t+1} = s' | s_t = s, a_t = a\}$

Se il livello di energia è alto (s_t = alto):

se scelgo Wait – s_{t+1} = alto.

se scelgo Search, s_{t+1} avrà una certa probabilità di diventare low.

$$P_{high->low|Search} = \Pr\{s_{t+1} = low \,|\, s_t = high, a_t = Search\} = \alpha$$

Se il livello di energia è basso (s_t = basso):

se scelgo Wait – s_{t+1} = basso.

se scelgo Recharge - s_{t+1} = alto.

se scelgo Search, s_{t+1} avrà una certa probabilità di fermarsi.

$$P_{low->low|Search} = \Pr\{s_{t+1} = low \,|\, s_t = low, a_t = Search\} = \beta$$

Reward del Robot

Funzione Reward: $R_{s->s'|a} = E\{r_{t+1} = r' | s_t = s, a_t = a, s_{t+1} = s'\}$

R^{search} reward se il robot sta cercando.

R^{wait} reward se il robot sta cercando.

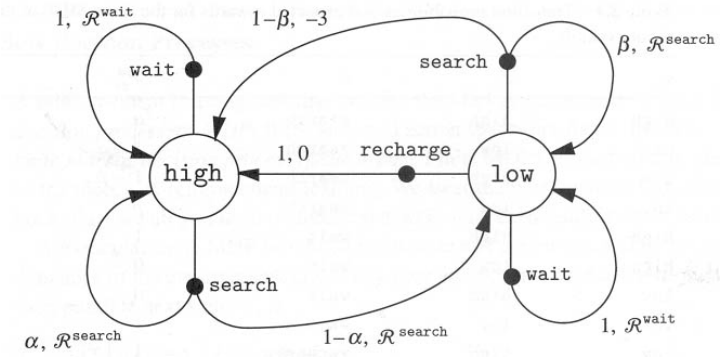
-3 se occcorre portarlo a ricaricarsi.

0 se il robot va autonomamente a ricaricarsi.

$R^{search} > R^{wait}$.

Esempio di calcolo della funzione valore

V(high) = ? a(high) = ?
V(low) = ? A(low) = ?



Calcolo ricorsivo della Value function

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

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

Relazione?

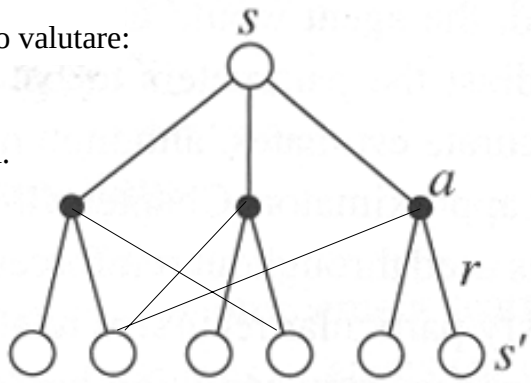
$$V^{\pi}(s) = E_{\pi}\left\{r_{t+1} + \sum_{k=1}^{\infty}\gamma^k r_{t+k+1} \,\middle|\, s_t = s\right\} =$$

$$V^{\pi}(s) = E_{\pi}\left\{r_{t+1} + \gamma\sum_{k=0}^{\infty}\gamma^k r_{t+k+2} \,\middle|\, s_t = s\right\}$$

V(s) : primo termine

$$E_\pi\{r_{t+1} \mid s_t = s\} = \left[\sum_{a_j} \pi(a_j, s)\right] \sum_{s'} P_{s \to s' | a_j} \left[R_{s \to s' | a_j}\right]$$

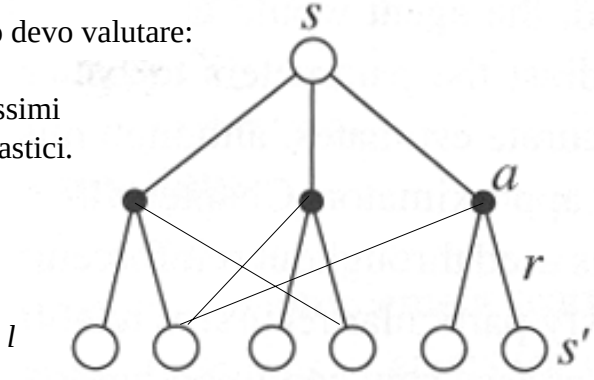
Per ogni stato devo valutare:
- Più azioni.
- Più stati prossimi
- Reward stocastici.



V(s) : secondo termine

$$E_\pi\left\{\gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+2} \,\middle|\, s_t = s\right\} = \gamma \sum_{l} E_\pi\{R_{t+1} \mid s_{t+1} = s'_l\} \Pr(s_{t+1} = s_k \mid s_t = s)$$

Per ogni stato devo valutare:
- Più azioni.
- Più stati prossimi
- Reward stocastici.



Calcolo ricorsivo della Value function

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

Legame?

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

Policy Next-state

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

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

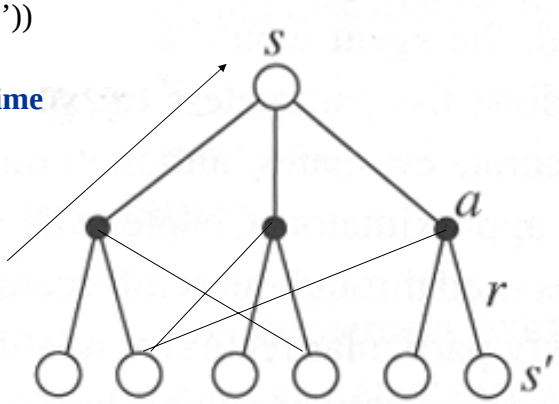
$$V^{\pi}(s) = \left\{\sum_{a_j} \pi(a_j, s) \sum_{s_l'} \left\{P_{s \to s_l'|a_j} \left[R_{s \to s_l'|a_j} + \gamma V^{\pi}(s_l')\right]\right\}\right\}$$

Bellman's equation

Osservazioni

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

Backwards in time



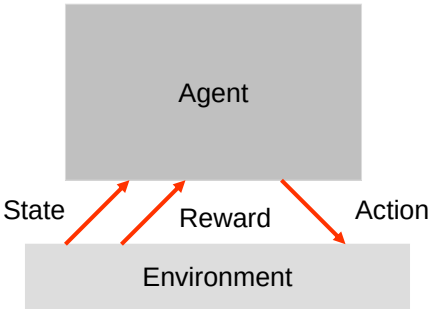
$$V^{\pi}(s) = \left\{\sum_{a_j} \pi(a_j, s) \sum_{s_l'} \left\{P_{s \to s_l'|a_j} \left[R_{s \to s_l'|a_j} + \gamma V^{\pi}(s_l')\right]\right\}\right\}$$

Reinforcement Learning Problem

Given: Repeatedly…
- Executed action
- Observed state
- Observed reward

Learn action policy $\pi: S \rightarrow A$
- Maximizes life reward
  $r_0 + \gamma r_1 + \gamma^2 r_2 \ldots$
  from any start state.
- Discount: $0 < \gamma < 1$

Note:
- Unsupervised learning
- Delayed reward



$s_0 \xrightarrow[r_0]{a_0} s_1 \xrightarrow[r_1]{a_1} s_2 \xrightarrow[r_2]{a_2} s_3$

Goal: Learn to choose actions that maximize life reward

$$r_0 + \gamma r_1 + \gamma^2 r_2 \ldots$$

Sommario

Le equazioni di Bellman

Value function ottima

Ottimizzazione Value function e 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, π_1, migliore di un'altra, π_2, se e solo se:
$V^{\pi 1}(s) >= V^{\pi 2}(s)\ \forall s$.

In particolare si definisce una politica ottima, π^*, se e solo se:
$V^*(s) >= V^{\pi}(s)\ \forall s$

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

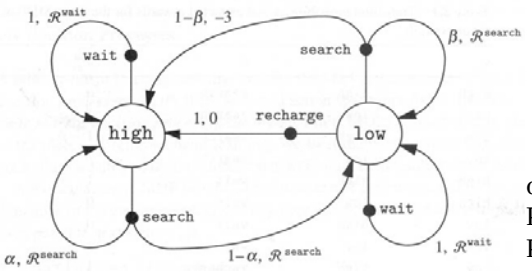
Calcolo ricorsivo della Value function ottima

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

V*(s) di uno stato, quando viene scelta la policy ottima, deve essere uguale al valore atteso del reward per l'azione migliore per lo stato s.

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

Esempio di calcolo della funzione valore



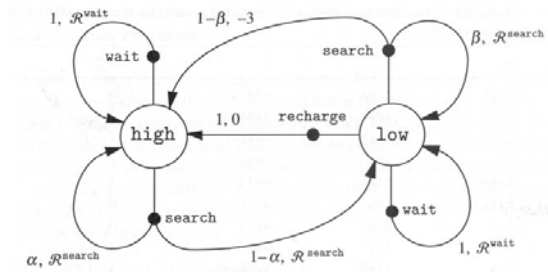
α=0.6, β=0.1, γ=0.8, R^{search}=3, R^{wait}=1
Pr(a, state=high) = [0.4, 0.6, 0]
Pr(a, state = low) = [0.4, 0.5, 0.1]

Vh = 0.4x1x[1+0.8Vh] + 0.6x0.4x[3+0.8xVh]+0.6x0.6x[3+0.8Vl]
Wait *Search -> high* *Search -> low*

Vl = 0.4x1x[1+0.8Vl] + 0.5x0.1x[3+0.8xVl]+0.6x0.9x[-3+0.8Vh]+
Wait 0.1x1x[0+0.8Vh] *Search -> High*
Search -> Low *Recharge*

Sistema lineare di 2 equazioni nelle 2 incognite: Vh e Vl
Come calcolo Vh* e Vl*?

Esempio di calcolo della policy ottima



V(high):

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

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

V(low):

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

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

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

Utilizzo di V(s)

Sistema di N equazioni non-lineari in N incognite (le entry della tabella della policy).

Politica greedy rispetto alla Value function.

Questa politica greedy (ad un passo) produce una politica ottima globalmente.

Vengono valutate le conseguente a breve termine delle azioni (1-step) ma non è una politca miope perché consente di ottenere una politica globalmente ottima.

Problematiche legate al calcolo di V(s)

Soluzione vicina alla ricerca esaustiva. Devo valutare per ogni stato tutte le possibili azioni (devo trovare il massimo).

Per tutte le possibili azioni devo calcolare la probabilità di transizione allo stato successivo e di ottenere una certa reward.

3 assunzioni:

1) Conoscenza della dinamica dell'ambiente: P(s->s'| a_j)
2) Potenza di calcolo sufficiente
3) Proprietà Markoviane dell'ambiente (definizione di uno stato).

Soluzioni approssimate.

Esempio (possibile progetto)

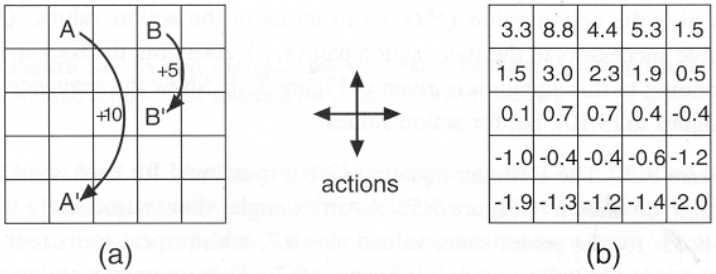


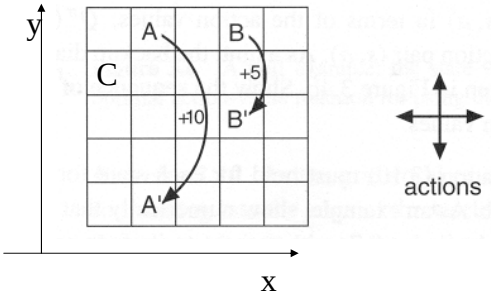
Figure 3.5 Grid example: (a) exceptional reward dynamics; (b) state-value function for the equiprobable random policy.

a = π(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)

Esempio di valore di uno stato

t: $s = s_t = C$

t + 1: $s = s' = s_{t+1}$
C + Dy p = 0.25 r = 0
C – Dy p = 0.25 r = 0
C + Dx p = 0.25 r = 0
C p = 0.25 r = -1
r_{t+1} = -0.25



t + 2, a = +Dy
→ C + Dy p = 0.25 r = -1
→ C p = 0.25 r = 0
→ A p = 0.25 r = 0
→ C + Dy p = 0.25 r = -1
r_{t+1} = -0.50

a = -Dy
→ C r = 0
→ C -2Dy r = 0
→ C – Dy + Dx r = 0
→ C – Dy r = -1
r_{t+1} = -0.25

Calcolo della funzione valore di uno stato

t: $s = s_t = C$

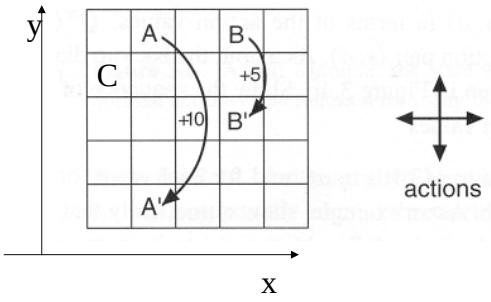
t + 1: $s = s' = s_{t+1}$

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':

$\pi(s,a)$ -- $P_{s \to s' \mid a}$ -- $R_{s \to s' \mid a}$ -- $V^{\pi}(s')$ ← Value of s'.

policy

Risposta ambiente

Reward

Sommario

Le equazioni di Bellman

Value function ottima