Sommario

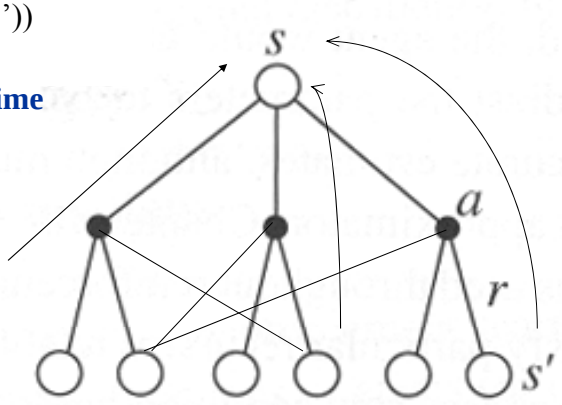
Determinazione ricorsiva della Value function

Determinazione ricorsiva della policy ottima.

Osservazioni

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

Backwards in time
(full backup)



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

Algoritmo per "iterative policy evaluation", versione batch

Partiamo da una politica $\pi(s,a)$ data.
Definiamo una soglia di convergenza τ
Inizializziamo $V(s) = 0\ \forall s$, compreso gli stati finali.
Repeat

{ $\Delta = 0$;

for s = 1 : N // $\forall s, \neq TS$ Forward pass

{

$W(s) = \sum_{a_j} \pi(s,a_j) \sum_{s'} P^{a_j}_{s->s'} \left[R^{a_j}_{s->s'} + \gamma V(s') \right]$ // W(s) è $V_{k+1}(s)$

$\Delta = \max(\Delta, | V(s) - W(s) |)$

}

for s=1:N

$V(s) = W(s)$;

} Until ($\Delta < \tau$);

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

3 assunzioni:

1) Conoscenza della dinamica dell'ambiente: P(s->s'| a_j)
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 $\pi(.)$, se questa si basa sul reward che riceverò negli istanti futuri?

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

Esempio

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

•L'agente può spostarsi: {dx, sx, alto, basso} se supera il bordo rimane nello stato di partenza.

• $\pi(s,a)$ è equiprobabile.

• R = -1 per ogni transizione, tranne quando s' = 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 – $V(4)_{k=1}$ per trial

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	??	0	0
0	0	0	0
0	0	-0	0

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

Per epoch:

- $V(s=5)_{k=1} = 0.25\ [1\ (-1 + 1\ V(9)_{k=0}] + 0.25\ [1\ (-1 + 1\ V(6)_{k=0}] +$
  $0.25\ [1\ (-1 + 1\ V(4)_{k=0}] + 0.25\ [1\ (-1 + 1\ V(1)_{k=0}] = -1$

Per trial:

- $V(s=5)_{k=1} = 0.25\ [1\ (-1 + 1\ V(9)\ ] + 0.25\ [1\ (-1 + 1\ V(6)] +$
  $0.25\ [1\ (0 + 1\ V(4)] + 0.25\ [1\ (-1 + 1\ V(1)] = -1.375$

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

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

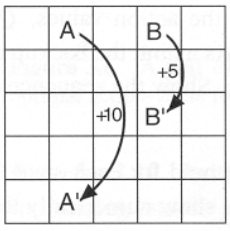
0	-1.4375	-1.9375	-2
-1.4375	-1.875	-2	-1.9375
-1.9375	-2	-1.875	-1.4375
-2	-1.9375	-1.4375	0

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

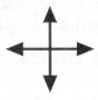
Esempi:

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

Esempio di 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.

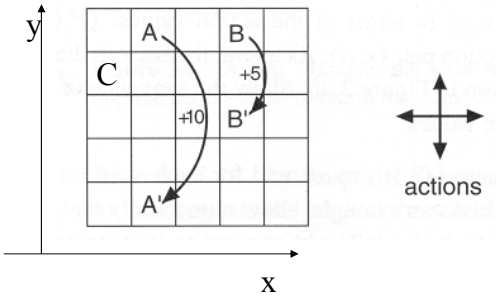
$a = \pi(s)$ equiprobabile $= 0.25\ \forall a$
R(s,a) per stati sui bordi e azioni che portano fuori dal bordo.
R(s) per tutti gli altri stati (= 0, +5, +10)

Calcolo della funzione valore per 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':

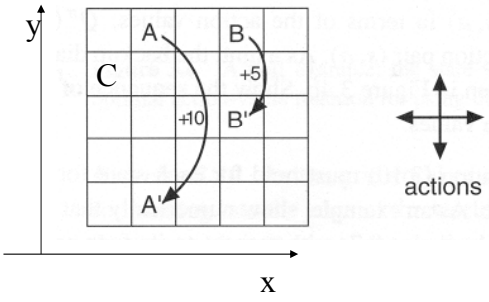
$\pi(s,a)$ -- $P_{s->s'|a}$ -- $R_{s->s'|a}$ -- $V^{\pi}(s')$ ← Value of s'.

policy — Risposta ambiente — Reward

Esempio di valore di uno stato

t: $s = s_t = C$

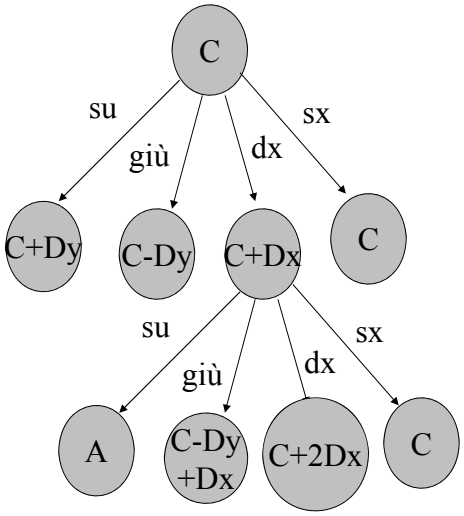
t + 1: $s = s' = s_{t+1}$
C + Dy p = 0.25 r = 0
C – Dy p = 0.25 r = 0
C + Dx p = 0.25 r = 0
C p = 0.25 r = -1
$r_{t+1} = -0.25$



t + 2, s = C+Dx
→ A p = 0.25 r = 0
→ C +Dx+Dy p = 0.25 r = 0
→ A +2Dx p = 0.25 r = 0
→ C p = 0.25 r = 0
$r_{t+2} = 0.0$

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

Sviluppo parziale a 2 passi

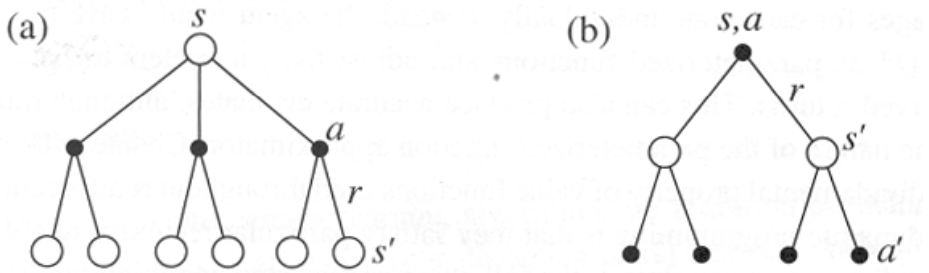


Sommario

Determinazione ricorsiva della Value function

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, 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:
$V^*(s) >= V^{\pi}(s)\ \forall s$

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

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.

Esempio: robot

$$Vh = Pr(W)x1x[1+0.8Vh] + Pr(S)x0.4x[3+0.8xVh]+Pr(S)x0.6x[3+0.8Vl]$$

$$Vl = Pr(W)x1x[1+0.8Vl] + Pr(S)x0.1x[3+0.8xVl]+Pr(S)x0.9x[-3+0.8Vh]+ Pr(R)x1x[0+0.8Vh]$$

$\alpha = 0.8$, $\beta = 0.4$, $R_w = 1$, $R_s = 3$, $R_r = 0$

Sono possibili un numero limitato di scelte deterministiche di azioni

STATO: Vh

a = search: $Vh \cong 6 + 0.65Vl$

a = wait: $Vh = 5$ (per ogni Vl)

STATO: Vl

a = search: $Vl \cong -2.2 + 6.6Vh$

a = wait: $Vl = 5$

a = recharge: $Vl = 0.8Vh$

Esempio: robot – policy I

$$Vh = Pr(W)x1x[1+0.8Vh] + Pr(S)x0.4x[3+0.8xVh]+Pr(S)x0.6x[3+0.8Vl]$$

$$Vl = Pr(W)x1x[1+0.8Vl] + Pr(S)x0.1x[3+0.8xVl]+Pr(S)x0.9x[-3+0.8Vh]+ Pr(R)x1x[0+0.8Vh]$$

Policy iniziale equiprobabile

Value function inizialmente: Vh = Vl = 0

$Vh \cong 0.5 \times (6 + 0.65Vl) + 0.5 \times 5 = 5.5$

$Vl \cong 0.33 \times (-2.2 + 6.6Vh) + 0.33 \times 5 + 0.33 \times 0.8\ Vh = -0.726 + 1.65 = 0.924$

Devo migliorare la stima di V(s)

Esempio: robot – policy I

$$Vh = Pr(W)x1x[1+0.8Vh] + Pr(S)x0.4x[3+0.8xVh]+Pr(S)x0.6x[3+0.8Vl]$$

$$Vl = Pr(W)x1x[1+0.8Vl] + Pr(S)x0.1x[3+0.8xVl]+Pr(S)x0.9x[-3+0.8Vh]+ Pr(R)x1x[0+0.8Vh]$$

Policy iniziale equiprobabile

Value function equal to: Vh = 5.5
Vl = 0.924

$Vh \cong 0.5 \times (6 + 0.65Vl) + 0.5 \times 5 = 5.8$

$Vl \cong 0.33 \times (-2.2 + 6.6Vh) + 0.33 \times 5 + 0.33 \times 0.8\ Vh = 12.870$

Esempio: robot – policy I correct value

Vh = Pr(W)x1x[1+0.8Vh] + Pr(S)x0.4x[3+0.8xVh]+Pr(S)x0.6x[3+0.8Vl]

Vl = Pr(W)x1x[1+0.8Vl] + Pr(S)x0.1x[3+0.8xVl]+Pr(S)x0.9x[-3+0.8Vh]+ Pr(R)x1x[0+0.8Vh]

Policy iniziale equiprobabile

Value function equal to: Vh = 5.5
Vl = 0.924

Vh ≅ 0.5 x (6 + 0.65Vl) + 0.5 x 5 = 5.8

Vl ≅ 0.33 x (-2.2 + 6.6Vh) + 0.33 x 5
+ 0.33 x 0.8 Vh = 12.870

Esempio: robot - II

Vh = Pr(W)x1x[1+0.8Vh] + Pr(S)x0.4x[3+0.8xVh]+Pr(S)x0.6x[3+0.8Vl]

Vl = Pr(W)x1x[1+0.8Vl] + Pr(S)x0.1x[3+0.8xVl]+Pr(S)x0.9x[-3+0.8Vh]+ Pr(R)x1x[0+0.8Vh]

Policy iniziale deterministica:

STATO: Vh
a = search → Vh ≅ 6 + 0.65Vl ≅ 9.25
STATO: Vl
a = wait → Vl = 5

Posso migliorare la policy?

Esempio: robot - III

Miglioro la policy, modificando l'azione associata a Vl:

STATO: Vh
a = search ➔ Vh ≅ 6 + 0.65Vl ≅ 9.25
STATO: Vl
a = recharge → Vl = 0.8Vh = 7,4

Ho stimato correttamente V(s)? No

STATO: Vh
a = search ➔ Vh ≅ 6 + 0.65Vl ≅ 11.06
STATO: Vl
a = recharge → Vl = 0.8Vh = 7,4

Ho stimato correttamente V(s)? No. Devo iterare la policy evaluation.

Esempio: robot - IV

<u>Asintoticamente</u> calcolo il valore degli stati:

STATO: Vh
a = search ➔ Vh ≅ 6 + 0.65Vl → 12.5
STATO: Vl
a = recharge → Vl = 0.8Vh → 10.0

Potrei ottenere gli stessi valori risolvendo il sistema lineare:
Vh = 6 + 0.65Vl = 12.5
Vl = 0.8Vh = 10.0

Ho terminato?

Algoritmo (progetto per esame)

Repeat until policy-stable

1. Initialization
   $V(s) \in \Re$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Repeat
   $\Delta \leftarrow 0$
   For each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s'} \mathcal{P}_{ss'}^{\pi(s)} \left[\mathcal{R}_{ss'}^{\pi(s)} + \gamma V(s') \right]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number)

3. Policy Improvement
   policy-stable $\leftarrow$ true
   For each $s \in \mathcal{S}$:
   $b \leftarrow \pi(s)$
   $\pi(s) \leftarrow \arg\max_a \sum_{s'} \mathcal{P}_{ss'}^{a} \left[\mathcal{R}_{ss'}^{a} + \gamma V(s') \right]$
   If $b \neq \pi(s)$, then *policy-stable* $\leftarrow$ false
   If *policy-stable*, then stop; else go to 2

Esempio

Consideriamo uno scommettitore che scommette su testa e croce.

Se esce testa, vince tanti Euro quanti ne ha scommesso. Se esce croce, perde gli Euro che ha scommesso.

Il gioco termina quando lo scommettitore arriva ad accumulare 100 Euro di vincita o perde tutto il suo capitale.

Il suo capitale di partenza è variabile tra 1 e 99 Euro. Ad ogni lancio della moneta, lo scommettitore può decidere quanto scommettere sul fatto che esca testa.

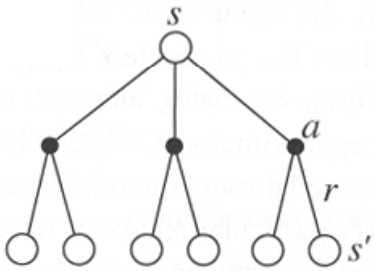
Undiscounted, episodico, MDP.

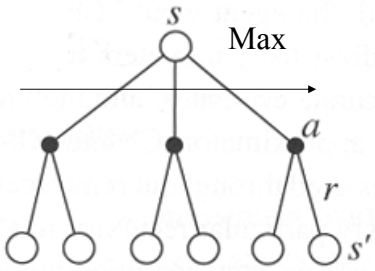
Lo stato è il capitale accumulato dal giocatore: s = {1, 2, …… , 99}.
L'azione è quanto viene scommesso ad ogni lancio: a = {1, 2, … min (s, 100 – s)}.
Conosciamo la probabilità p con cui esce testa (la moneta può essere truccata e quindi le due condizioni testa/croce possono essere non equiproababili).
La dinamica dell'ambiente non dipende da a ed è rappresentata dalla transizione di uno stato all'altro che a sua volta è funzione del fatto che esca testa o meno.
Il reward istantaneo = 0 tranne quando raggiunge 100 Euro, nel qual caso reward = +1.

Vogliamo massimizzare la probabilità di vincere. Si può fare tramite Value iteration.

1° passo di Value iteration

$P_{s->s'}$ = {testa, croce} non dipende da a. Sono indipendenti.



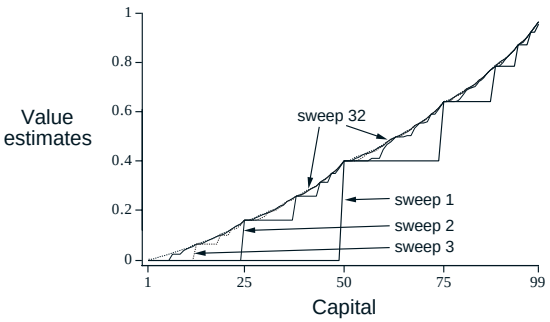


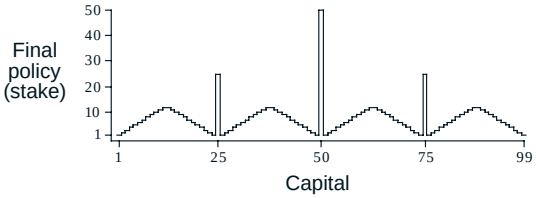
$s < 50$ | $s \geq 50$

$V_{k+1}(s) = 0.4\ (0 + \gamma\ V(s')) = 0$ | $V_{k+1}(s) = 0.4\ (1 + \gamma\ V(s')) = 0.4$

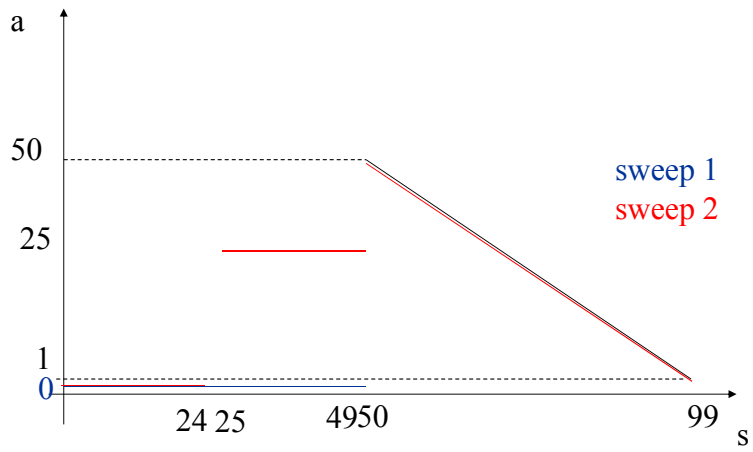
$\forall a$ | Per a scelto in modo da arrivare a 100€ in caso di vincita.

Stima della Value function





Policy selezionata



Ci sono diversi stati per cui la Q(s,.) assume lo stesso valore per diverse azioni.

Esercizio: autonoleggio

2 locazioni di autonoleggio

In ogni locazione, se quando arriva un cliente l'auto è disponibile, si guadagnano 10 euro.

Se l'auto non è disponibile si perde la gestione della locazione.

Le auto diventano disponibili il giorno dopo che sono state restituite dopo il noleggio.

Si possono portare le auto da un autonoleggio all'altro di notte al costo di 2 Euro per ogni auto.

Supponiamo che il numero di auto richieste e restituite in ognuno dei 2 autonoleggi sia rappresentato da una distribuzione di Poisson (probabilità che vengano richieste n auto: $(\lambda^n / n!)e^{-\lambda}$, dove λ è il valore atteso (media) di auto richieste o restituite.

Supponiamo che non ci possano essere più di 20 auto in uno dei 2 atuonoleggi (ciascuna auto aggiuntiva viene inviata al centro di raccolta nazionale della compagnia).

Supponiamo anche che un massimo di 5 auto possa essere spostato in una singola notte.

Questo problema is può formulare con un MDP dove lo stato è rappresentato dal numero di auto presenti in ciascuno dei 2 autonoleggi al termine di una giornata e le azioni il numero di auto che vengono spostate durante la notte.

Consideriamo $\gamma = 0.9$. Il reward sarà il reward accumulato durante il giorno + notte.

Partiamo dalla policy $\pi(s,a) = 0$: nessuna auto viene mossa. Siamo in grado di migliorarla? Come?

Sommario

Determinazione ricorsiva della Value function

Determinazione della policy ottima.