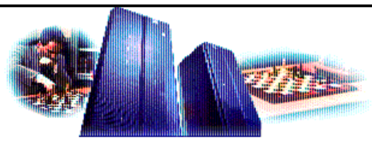
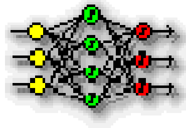




# Sistemi Intelligenti Reinforcement Learning: Equazioni di Bellman e miglioramento policy

Alberto Borghese

Università degli Studi di Milano
Laboratorio di Sistemi Intelligenti Applicati (AIS-La
artimento di Scienze dell'Informazione
borghese@di.unimi.it



## Sommario

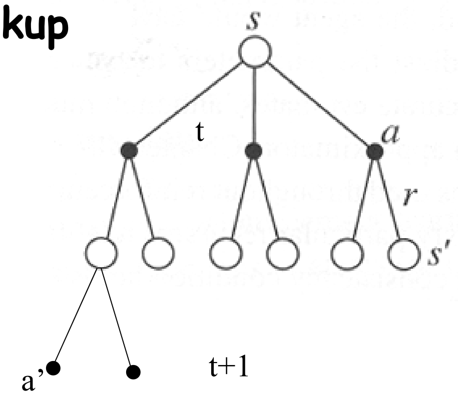
Come migliorare la policy (Value iteration)

Esempi

# Tecnica full-backup

Back-up

$\pi(s,a)$ fissata




Conosciamo $Q_k(s_t,a_t)$ $\forall s_t$, anche per $s'_{t+1}$ quindi:
Analizziamo la transizione da $s_t,a_t$ -> $(s'_{t+1}, a'_{t+1})$
Calcoliamo un nuovo valore di Q per s,a: $Q_{k+1}(s_t,a_t)$ congruente con:
$Q_k(s_{t+1},a_{t+1})$ ed $r_{t+1}$
*Full backup* se esaminiamo tutti gli s',a' (cf. DP).
Da s' mi guardo indietro ed aggiorno Q(s, a).

$\boxed{\pi \text{ fissata}}$



# Calcolo iterativo della Value Function

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

Equazione di Bellman per "iterative policy evaluation":

$$Q_{k+1}^{\pi}(s,a)=\left\{\sum_{s_l'}\left\{P_{s\to s_l'|a}\left[R_{s\to s_l'|a}+\gamma\sum_{a'_j}\pi\left(a'_j,s_l'\right)Q_k^{\pi}(s_l',a_j')\right]\right\}\right\}$$

Mi fido di $Q_{k+1}(s',a')$ (Backup)

$$\lim_{k\to\infty}\{Q_k(s,a)\}=Q^{\pi}(s,a)$$

## Relazione soddisfatta da Q*(s,a)

$$Q^*(s,a) = \underset{a_{t+1}}{Max}\left[E_\pi\{R_t \mid s_t = s, a_t = a\}\right] =$$

$$\underset{a_{t+1}}{Max}\left[E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+1} \middle| s_t = s, a_t = a\right\}\right] =$$

$$\underset{a_{t+1}}{Max}\left[r_{t+1} + \gamma E_\pi\left\{\sum_{k=0}^{\infty}\gamma^k r_{t+k+2} \middle| s_t = s, a_t = a\right\}\right] =$$

$$\underset{a_{t+1}}{Max}\left[r_{t+1} + \gamma Q^*(s_{t+1}, a_{t+1}) \mid s_t = s, a_t = a\right] =>$$

$$\boxed{Q^*(s,a) = \underset{a'}{Max}\left\{P_{s->s'|a}\left[R_{s->s'|a} + \gamma Q^*(s',a')\right]\right\}}$$

Bellman's Equation For optimal policy



## Miglioramento della policy

Tutti gli stati sono valutati in funzione di una policy data.

Condizioni di funzionamento dell'agente:
•Policy **deterministica**: a = π(s).
•Ambiente **stocastico**.

Cosa succede se cambiamo la policy per un certo stato $s_m$? $a_{new} \neq \pi(s_m)$.
Cosa viene influenzato?

Scelgo $a_{new}$ in $s_m$, visiterò una certa sequenza di stati, per questi stati seguirò la policy precedente per $s \neq s_m$. Cosa viene influenzato?

Come faccio a valutare se miglioro la policy o no?

## Effetto del cambiamento della policy

Cambia, a, cambiano i possibili stati successivi ad $s_m$, $\{s_{t+k}\}$, ed il reward a lungo termine:

$$Q^\pi(s_m, a_{new}) = E_\pi\left\{r_{t+1} + \gamma V^\pi(s_{t+1}) \mid s_t = s_m, a_t = a_{new} \neq \pi(s_m)\right\} =$$

$$\sum_{s'} P^{a_{new}}_{s_m -> s'}\left[R^{a_{new}}_{s_m -> s'} + \gamma V^\pi(s')\right]$$

V(s) = value function sullo stato

?

$$Q^\pi(s_m, a_{new}) >=< Q^\pi(s_m, a = \pi(s_m)) \quad \forall s, a\,?$$

Se il reward fosse migliore con $a_{new}$, sceglierò sempre $a_{new}$ in $s_m$.

Il reward a lungo termine può essere maggiore (minore) solamente se aumenta (diminuisce) il reward totale "visto" ad un passo (reward del passo + reward successivo).



## Enunciato del teorema del miglioramento della policy

$$Q^\pi(s, a) = \sum_k P_{s \to s_k | a}\left[R_{s \to s_k | a} + \gamma V^\pi(s_k)\right]$$

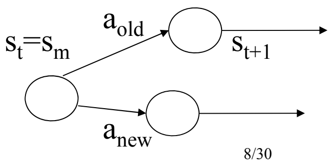
**Ipotesi:** $\pi$ and $\pi$' deterministi policies

$Q^\pi(s_m, \pi'(s_m)) \geq V^\pi(s_m)$

$$Q^\pi(s, a_{new} = \pi'(s_m)) = \sum_k P_{s_m \to s_k | a_{new}}\left[R_{s_m \to s_k | a_{new}} + \gamma V^\pi(s_k)\right]$$

**Tesi:** $\pi$' è meglio di $\pi$. Cioè: $V^{\pi'}(s) >= V^\pi(s)$ $\forall$**s.**

$Q^{\pi'}(s, a_{new}) >= Q^\pi(s, a_{old})$



## Dimostrazione del teorema del miglioramento della policy

**Analizziamo la seguente condizione:**

$\pi' = \pi$ $\forall$s tranne che per $s_m$ per il quale si applica l'azione:
$a_{new} = \pi'(s_m)$

Risulta che il reward a lungo termine è maggiore per $a_{new} = \pi'(s)$.

$V^{\pi'}(s) = Q^{\pi'}(s, a_{new} = \pi'(s)) >= Q^{\pi}(s, a = \pi(s)) = V^{\pi}(s)$

**Tesi:** $\pi'$ è meglio di $\pi$. Cioè: $V^{\pi'}(s) >= V^{\pi}(s)$ $\forall$s (ed in particolare per gli altri stati s)



## Dimostrazione del teorema del miglioramento della policy

Hp: $Q^{\pi}(s,\pi'(s)) >= V^{\pi}(s)$ $\forall$s $\pi'(s,a)$ è migliore per almeno uno stato

$$
\begin{aligned}
V^{\pi}(s) \le \; & Q^{\pi}(s,\pi'(s)) \\
&= E_{\pi'}\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t = s\} \\
&<= E_{\pi'}\{r_{t+1} + \gamma Q^{\pi}(s_{t+1}, \pi'(s_{t+1})) \mid s_t = s\} \\
&<= E_{\pi'}\{r_{t+1} + \gamma E_{\pi'}(r_{t+2} + \gamma V^{\pi}(s_{t+2})) \mid s_t = s\} \\
&= E_{\pi'}\{r_{t+1} + \gamma r_{t+2} + \gamma^2 V^{\pi}(s_{t+2})) \mid s_t = s\}
\end{aligned}
$$

Sostituisco ancora $Q^{\pi*}(.)$

$$<= E_{\pi'}\{r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots\ldots \mid s_t = s\}$$

Th: $V^{\pi}(s) <= V^{\pi'}(s)$

# Osservazioni

$s = s_m \qquad Q^{\pi}(s_m, \pi'(s)) \geq Q^{\pi}(s_m, \pi(s))$

$$s \neq s_m \qquad Q^{\pi}(s,a) = E_{\pi'}\{r_{t+1} + \gamma V^{\pi}(s_{t+1}) \mid s_t = s\}$$
$$= E_{\pi'}\{r_{t+1} + \gamma Q^{\pi}(s_{t+1}, \pi(s_{t+1})) \mid s_t = s\}$$

Se $s_{t+k}$ = sm miglioro la Q(s,a).

Se nessun $s_{t+k} = s_m$. Non varia la Q(s,a).



# Ottimizzazione policy

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

Posso ordinare la Value function Q(s,a) in ordine decrescente, in funzione delle azioni scelte in s (policy).

Si definisce una policy, $\pi_1$, migliore di un'altra, $\pi_2$, se e solo se:
$Q^{\pi 1}(s,a(s)) > = Q^{\pi 2}(s,a(s))\ \forall s$.

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

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

## Calcolo ricorsivo della Value function ottima::confronti

$$Q_{k+1}^{\pi}(s,a)=\left\{\sum_{s_l'}\left\{P_{s\to s_l'|a}\left[R_{s\to s_l'|a}+\gamma\sum_{a'_j}\pi\left(a'_j,s_l'\right)Q_k^{\pi}(s_l',a_j')\right]\right\}\right\}$$

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

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

Politica greedy: scelgo l'azione ottimale.
Ha senso per il robot raccogli-lattine?

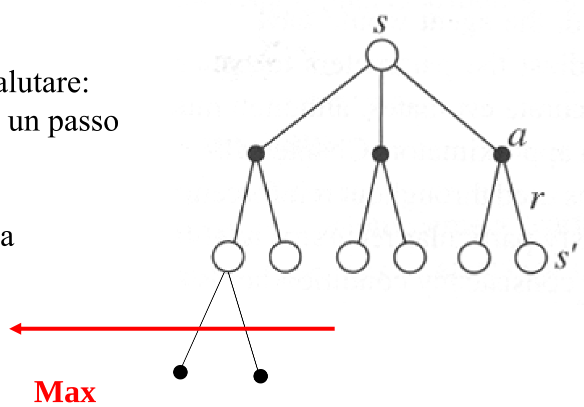


## Q*(s,a) – Osservazioni

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

Per ogni stato devo valutare:
•L'azione migliore ad un passo

Come valuto?
• analizzando reward a lungo termine




*t+1* **Max**

## Policy iteration

Iterazione tra:

- Calcolo iterativo della Value function (iterative policy evaluation)
- Miglioramento della policy (policy improvement)

$\pi_0 \rightarrow V^{\pi 0} \rightarrow \pi_1 \rightarrow V^{\pi 1} \rightarrow \pi_2 \rightarrow V^{\pi 2} \rightarrow \dots\dots$
$\rightarrow \pi^* \rightarrow V^*$

Converge velocemente ad una buona politica
(cf. Software Sommaruga)



## Algoritmo (progetto per esame) - I

**Inizialization**

Q(s,a) = 0;
π(s,a) = random (e.g. equiprobabile);

Repeat
    point 2.
    point 3.
until policy_stable

# Algoritmo (progetto per esame) - II

**Policy evaluation – versione per trial**

Repeat
  Th = 0; // small value;
  for s = 1:N
    for a = 1:M
      Q_temp = $\sum_{s'} \Pr_{s->s'|a}[R_{s->s'|a} + \gamma \sum_{a'} \Pr_{a'|s'} Q(s',a')]$

      $\Delta$Q = |Q(s,a) – Q_temp)|
      Q(s,a) = Q_temp;
      th = max(th, $\Delta$Q)
    end:
  end;
until th < th_max;



# Algoritmo (progetto per esame) - III

**Policy improvement**

policy_stable = true;
for s = 1:N  // in alternativa, scelgo uno stato
  a_old = $\pi$(s);
  a_new = $\arg\max_{a'} \left( \sum_{s'} \Pr_{s->s'|a} \left[ R_{s->s'|a} + \gamma Q(s',a') \right] \right)$;

  if (a_new $\neq$ a_old)
    policy_stable = false;
end;

# Algoritmo (progetto per esame) - II

**Policy evaluation – versione per epoch**

Repeat
  Th = 0; // small value;
  for s = 1:N
    for a = 1:M
      Q_temp(s,a) = $\sum_{s'} \Pr_{s->s'|a}[R_{s->s'|a} + \gamma \sum_{a'} \Pr_{a'|s'} Q(s',a')]$

      ΔQ = |Q(s,a) – Q_temp(s,a))|
      th = max(th, ΔQ)
    end:
  end;
  for s = 1:N, for a=1:m
    Q(s,a) = Q_temp(s,a);
  end; end;
until th < th_max;



# Max or soft max

**Policy improvement**

policy_stable = true;
for s = 1:N   // in alternativa, scelgo uno stato
  a_old = π(s);

  a_new = $\boxed{\arg\max_{a'} \left( \sum_{s'} \Pr_{s->s'|a} \left[ R_{s->s'|a} + \gamma Q(s',a') \right] \right)}$;

  if (a_new ≠ a_old)
    policy_stable = false;
end;

Max con policy ε-greedy, soft-max, ...

# Iterative policy evaluation - problema

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

Converge al limite a $V^{\pi}(s)$. Come facciamo a troncare?



# Value iteration

$$Q_{k+1}(s,a) = \sum_{s'} P_{s \to s'|a} \left[ R_{s \to s'|a} + \gamma \left( \sum_{a'_j} \pi(a'_j, s') Q_k(s', a') \right) \right]$$

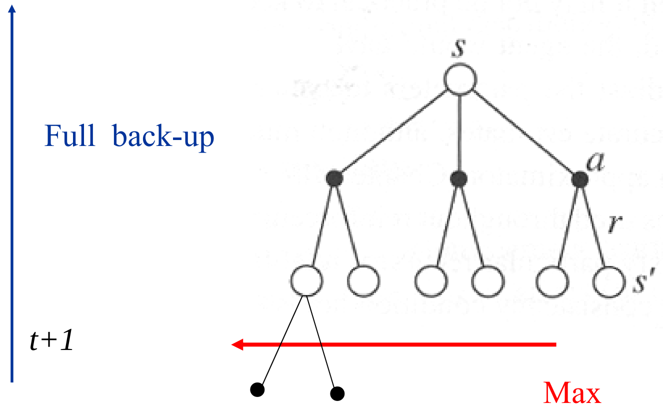
Invece di considerare una policy stocastica, consideriamo l'azione migliore:

$$Q_{k+1}(s,a) = \max_{a'} \sum_{s'} P_{s \to s'|a} \left[ R_{s \to s'|a} + \gamma Q_k(s', a') \right]$$

$\forall$**s,a**

# Visualizzazione grafica

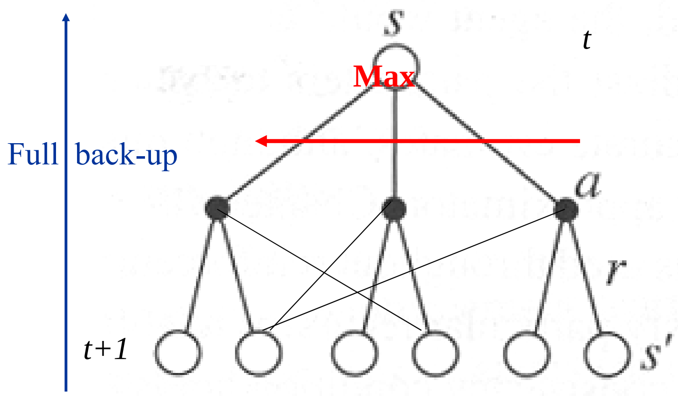
$$V_{k+1}(s) = \max_a \sum_{s'} P_{s \to s'|a} \left[ R_{s \to s'|a} + \gamma V_k(s') \right]$$






# Visualizzazione grafica

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



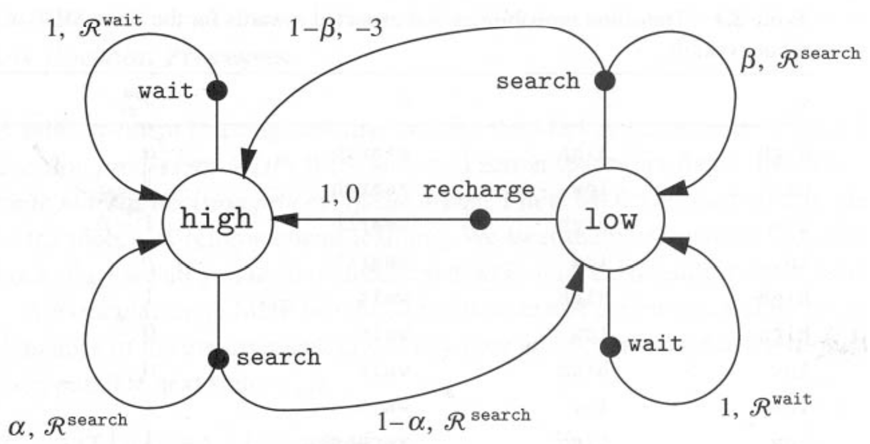
## Sommario

Come migliorare la policy (Value iteration)

Esempi



## Robot cerca-lattine



## Esempio: robot – Policy deterministica

Q(h,search) = Pr(h->l, search) x [R(h->h, search) + γ x Q(h,search)]
Pr(h->h, search) x [R(h->l, search) + γ Q(l,wait)]
Q(h,search) = 0.4x[3+0.8xQ(h,search)]+0.6x[3+0.8Q(l,wait)]

Q(l,wait) = Pr(l->l, wait) x [R(l->l, wait)+0.8 x Q(l,wait)]
Q(l,wait) = 1 x [1+0.8 Q(l,wait)]

**Policy iniziale deterministica:**

**STATO: Q(h,search) ➔**
Q(h,s) ≅ 4,4 + 0.7 Q(l,w) ≅ 7.95
**STATO: Q(l, wait) ➔**
Q(l,wait) = 5

Posso migliorare la policy?



## Esempio: robot – miglioramento policy

**Miglioro la policy, modificando l'azione associata a s = low:**

**STATO: high**
a = search ➔ Q(h,search) ≅ 4,4 + 0.7 Q(l,recharge) ≠7.95
**STATO: low**
a = recharge ➔ Q(l,recharge) = 0 + 0.8 Q(h,search) = ???

Ho stimato correttamente Q(h,search)? No
Applico iterative policy evaluation

**STATO: Vl**
a = recharge → $Q_1$(l,r) = 0.8 $Q_1$(h,s) = 0.8 x 7.95 = 6.36
**STATO: high**
a = search ➔ $Q_2$(h,s) ≅ 4.4 + 0.7 $Q_1$(l,r) ≅ 4.4 + 0.7 x 6.36 = 8.85
Ho stimato correttamente Q(s,a)? No. Devo iterare la policy evaluation.

# Esempio: robot - IV

**Asintoticamente calcolo il valore vero delle coppie stato-azione:**

**STATO: high**
a = search → $Q(h,s) \cong Q_2(h,s) \cong 4.4+0.7\ Q_1(l,r) = 4.4 + 0.7x6.36 = 8.85$
**STATO: low**
a = recharge → $Q(l,r) = 0.8\ Q(h,s)$ → 7.1

Potrei ottenere gli stessi valori ottenuti asintoticamente, risolvendo il sistema lineare:
$Q(h,s) = 4.4 + 0.7\ Q(l,r) =$
$Q(l,r) = 0.8\ Q(h,s) =$

Ho terminato?



# Sommario

Come migliorare la policy (Value iteration)

Esempi