

Sistemi Intelligenti Reinforcement Learning: Temporal Difference

Alberto Borghese

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



A.A. 2009-2010

1/30

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



Sommario



Asynchronous solutions

Temporal differences

A.A. 2009-2010

2/30

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



Value Iteration

Policy iteration:

Iterative policy evaluation

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

$$\forall s \quad \pi^*(s_k) = \arg \max_a \sum_{s'} P_{s \rightarrow s' | a} [R_{s \rightarrow s' | a} + \gamma V^\pi(s')] = \arg \max_a Q(s, a)$$

Value iteration:

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P_{s \rightarrow s' | a} [R_{s \rightarrow s' | a} + \gamma V_k(s')] \quad \forall s$$



Problemi

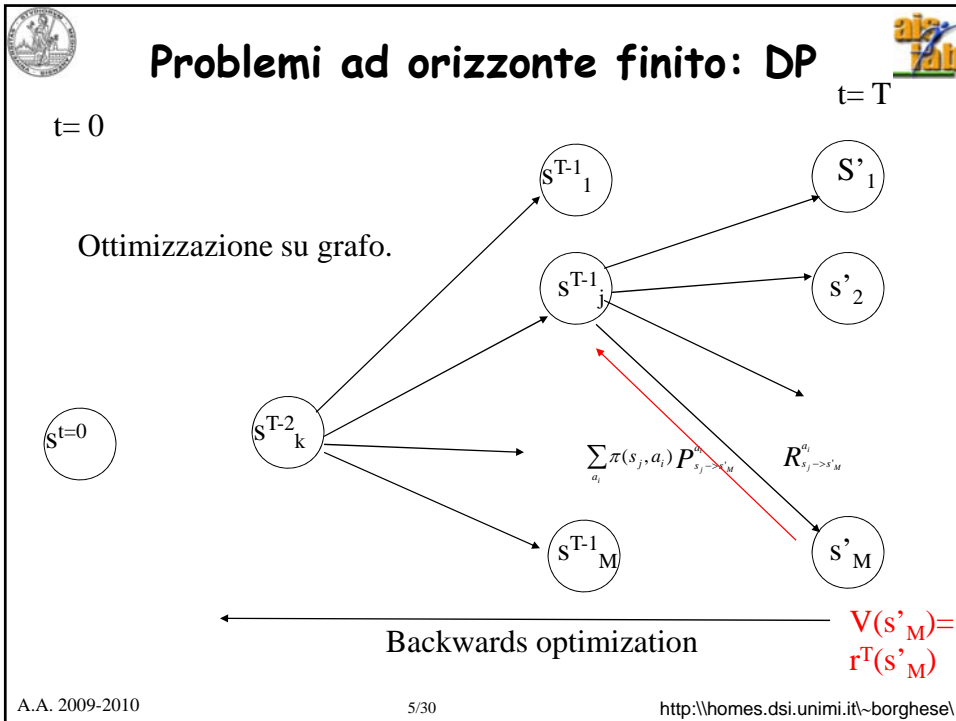
Framework: DP

L'analisi di tutti gli stati può essere computazionalmente molto pesante: *curse of dimensionality* (of the state space).

Nel frattempo la policy non evolve.

La risposta dell'ambiente è nota (ne è noto il modello statistico), è noto cioè il reward e lo stato prossimo.

Per problemi ad orizzonte finito -> ottimizzazione su grafo.



Asynchronous solution

Asynchronous DP.

Si può applicare ad un agente in azione. Vengono aggiornati solamente gli stati visitati dall'agente durante la sua azione.

Possiamo scegliere l'ordine in cui gli stati sono visitati. Non è più richiesto un ciclo.

A.A. 2009-2010 6/30 http://homes.dsi.unimi.it/~borghese/



Evoluzione della metodologia

Policy iteration:

Iterative policy evaluation

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

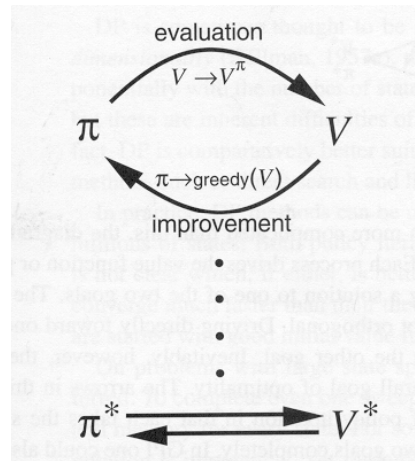
$$\forall s \quad \pi'(s_k) = \arg \max_a \sum_{s'} P_{s \rightarrow s' | a} [R_{s \rightarrow s' | a} + \gamma V^\pi(s')]$$

Value iteration:

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P_{s \rightarrow s' | a} [R_{s \rightarrow s' | a} + \gamma V_k(s')] \quad \forall s$$

Asynchronous solution:

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P_{s \rightarrow s' | a} [R_{s \rightarrow s' | a} + \gamma V_k(s')] \quad \mathbf{1 \text{ stato}}$$



Generalized Policy iteration

- Policy iteration
- Value iteration
- Asynchronous DP

Riassunto

Competizione e cooperazione -> V corretta e policy ottimale.



Sommario



Asynchronous solutions

Temporal differences



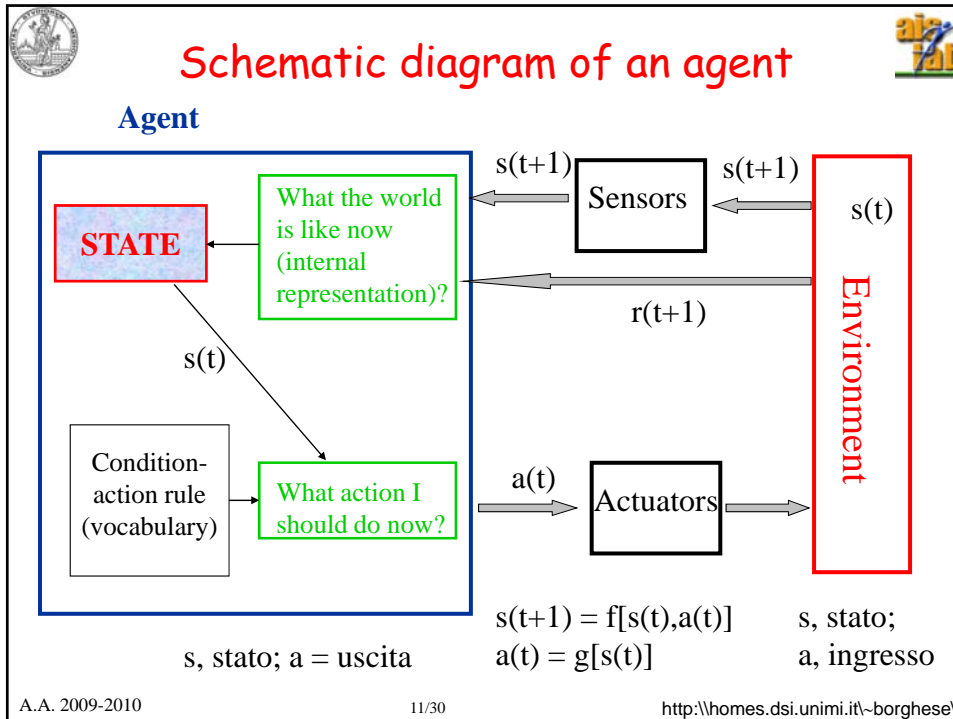
World RL competition



Started in NIP2006.

It became very popular and started a workshop on its own. Visit:

<http://rl-competition.org>



How About Learning the Value Function?

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

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

È una funzione dello stato.

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

$$\pi^*(s) = \arg \max_a \sum_{s'} P_{s \rightarrow s' | a} [R_{s \rightarrow s' | a} + \gamma V^\pi(s')]$$

Full backup, for all states

A.A. 2009-2010 12/30 <http://homes.dsi.unimi.it/~borghese/>



Value function iteration



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

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

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

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

Full backup, for all states



Asynchronous DP



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

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

Fino a questo punto, è noto un modello dell'ambiente:

- R(.)
- P(.)

Environment modeling -> Value function computation ->
Policy optimization.



Alcuni richiami: DP update



Iterazione tra:

- Calcolo della Value function

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

- Miglioramento della policy

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

Non sono noti



Background su Temporal Difference (TD) Learning



Al tempo t abbiamo a disposizione:

$$r_{t+1} = r' \quad R_{s \rightarrow s' | a_j}$$

$$s_{t+1} = s' \quad P_{s \rightarrow s' | a_j}$$

Reward certo

Transizione certa

vengono misurati dall'ambiente

Come si possono utilizzare per apprendere?



Confronto con il setting associativo



$$Q_{k+1} = Q_k - \frac{Q_k}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}} = Q_k + \alpha[r_{k+1} - Q_k]$$

Occupazione di memoria minima: Solo Q_k e k .
NB k è il numero di volte in cui è stata scelta a_j .

Questa forma è la base del RL. La sua forma generale è:

$$\begin{aligned} \text{NewEstimate} &= \text{OldEstimate} + \text{StepSize} [\text{Target} - \text{OldEstimate}] \\ \text{NewEstimate} &= \text{OldEstimate} + \text{StepSize} * \text{Error} \end{aligned}$$

$$\begin{aligned} \text{StepSize} &= \alpha = 1/k & a &= \text{cost} \\ \text{Rewards weight } w &= 1 & \text{Weight of } i\text{-th reward at time } k &: w = (1-\alpha)^{k-i} \end{aligned}$$

Qual è la differenza introdotta dall'approccio DP?



Un possibile aggiornamento

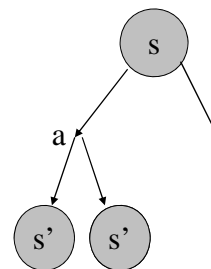


In iterative policy evaluation ottengo questo aggiornamento:

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

Ad ogni istante di tempo di ogni trial aggiorno
la Value function:

$$V_{k+1}(s) = [r' + \gamma V_k(s')]$$



Qual'è il problema?



Un possibile aggiornamento di $V(s)$



$$Q_{k+1} = Q_k - \frac{Q_k}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}} = Q_k + \alpha[r_{k+1} - Q_k] = Q_k + \Delta Q_k$$

$$V_{k+1}(s) = [r' + \gamma V_k(s')] \quad \text{Quanto vale } \alpha?$$

$$V_k(s) = V_k(s) + \Delta V_k(s)$$

Come calcolo $\Delta V_k(s)$?



TD(0) update



Ad ogni istante di tempo di ogni trial aggiorno la Value function:

$$V_{k+1}(s_t) = V_k(s_t) + \alpha[r_{t+1} + \gamma V_k(s_{t+1}) - V_k(s_t)]$$

Da confrontare con la iterative policy evaluation:

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

E con il valore di uno stato sotto la policy $\pi(s,a)$:

$$V^\pi(s) = E_\pi \{R_t | s_t = s\} = E_\pi \{r_{t+1} + \gamma V^\pi(s') | s_t = s\}$$

Quanto vale α ?

**Sample
backup**



Confronto con il setting associativo



$$Q_{k+1} = Q_k - \frac{Q_k}{N_{k+1}} + \frac{r_{k+1}}{N_{k+1}} = Q_k + \alpha[r_{k+1} - Q_k]$$

Occupazione di memoria minima: Solo Q_k e k .
NB k è il numero di volte in cui è stata scelta a_j .

Questa forma è la base del RL. La sua forma generale è:

$$\begin{aligned} \text{NewEstimate} &= \text{OldEstimate} + \text{StepSize} [\text{Target} - \text{OldEstimate}] \\ \text{NewEstimate} &= \text{OldEstimate} + \text{StepSize} * \text{Error}. \end{aligned}$$

$$\text{StepSize} = \alpha = 1/k \quad a = \text{cost}$$

Qual è la differenza introdotta dall'approccio DP?




Setting α value




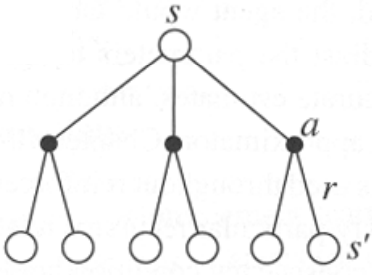
$\alpha(s_t, a_t, s_{t+1}) = 1/k(s_t, a_t, s_{t+1})$, where k represents the number of occurrences of s_t, a_t, s_{t+1} . With this setting the estimated Q tends to the expected value of $Q(s,a)$.

Per semplicità si assume solitamente $\alpha < 1$ costante. In questo caso, $Q(s,a)$ assume il valore di una media pesata dei reward a lungo termine collezionati da (s,a) , con peso: $(1-\alpha)^k$: *exponential recency-weighted average*.

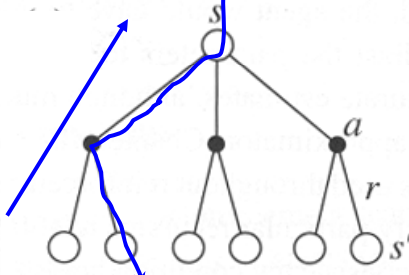


Sample backup





Full backup




Single sample is evaluated


A.A. 2009-2010

23/30

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



Algoritmo per TD(0) - Progetto per esame (da completare con scelta della policy)



Inizializziamo $V(s) = 0$.
 Inizializziamo la policy: $\pi(s,a)$ da valutare

Repeat

```

{
  s = s0;
  Repeat // For each state until terminal state, analyze an episode
  {
    a =  $\pi(s)$ ;
    s_next = NextState(s, a);
    reward = Reward(s, s_next, a);
     $V(s) = V(s) + \alpha [\text{reward} + \gamma V(s\_next) - V(s)]$ ;
    s = s_next;
  } until TerminalState
} Until convergence of  $V(s)$  for policy  $\pi(s,a)$ 
  
```

A.A. 2009-2010

24/30

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



Esempio: valutazione della policy mediante TD



Obiettivo: predire la durata del percorso per tornare a casa.

Stato	Tempo trascorso (reward totale)	Tempo segmento	Tempo previsto $V(s)$	Tempo totale previsto $V(s_0)$
Esco dall'ufficio	0	0 (0)	30	30
Salgo in auto (neve)	5	5 (5)	35	40
Esco dall'autostrada	20	15 (10)	15	35
Strada secondaria con camion davanti	30	10 (7)	10	40
Strada di casa	40	10 (5)	3	43
Entro in casa	43	3 (3)	0	43

$V(s)$ è l'expected "Time-to-go" $\alpha = 1$

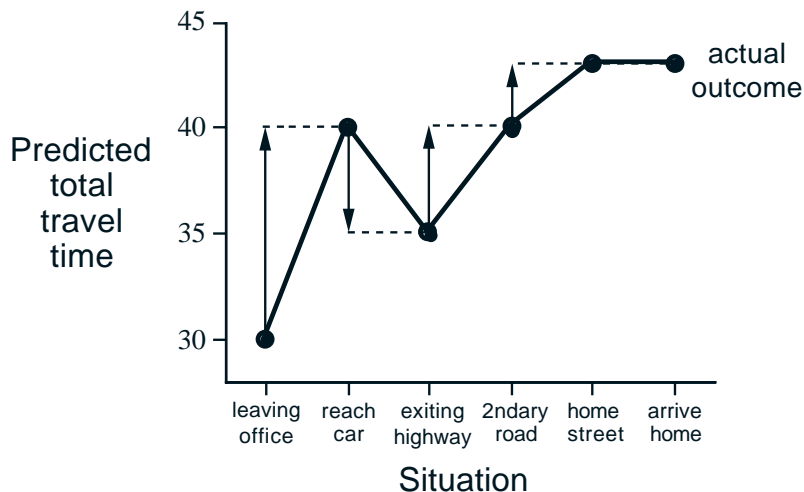
A.A. 2009-2010

25/30

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



Modifiche di $V(s)$ run-time



Qual'è il problema?

A.A. 2009-2010

26/30

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



Alcuni passi di iterazione per TD(0)



$$V(0) = V(0) + \alpha (r_1 + \gamma V(1) - V(0)) = 30 + \alpha (5 + 35 - 30) = 30 + \alpha * \Delta$$

Stima iniziale del tempo di percorrenza totale: 30m

Tempo di percorrenza fino all'auto: 5m

Stima del tempo di percorrenza dal parcheggio: 35m

$$V(1) = V(1) + \alpha (r_1 + \gamma V(2) - V(1)) = 35 + \alpha (20 + 15 - 35) = 35 + \alpha * \Delta$$

Stima iniziale del tempo di percorrenza dal parcheggio: 35m

Tempo di percorrenza fino ad uscita autostrada: 20m

Stima del tempo di percorrenza dall'uscita autostrada: 15m



Ruolo di α



$$V(1) = V(1) + \alpha (r_1 + \gamma V(2) - V(1)) = 35 + \alpha (20 + 15 - 35) = 35 + \alpha * \Delta$$

Stima iniziale del tempo di percorrenza dal parcheggio: 35m

Tempo di percorrenza fino ad uscita autostrada: 20m

Stima del tempo di percorrenza dall'uscita autostrada: 15m

$\alpha < 1$.

If $\alpha \ll 1$ aggiorno molto lentamente la value function.

If $\alpha = 1/k$ aggiorno la value function in modo da tendere al valore atteso. Devo memorizzare le occorrenze dello stato s .

If $\alpha = \text{cost}$. Aggiorno la value function, pesando maggiormente i risultati collezionati dalle visite dello stato più recenti.



Proprietà del metodo TD



Non richiede conoscenze a priori dell'ambiente.
L'agente stima dalle sue stesse stime precedenti (bootstrap).
Si dimostra che il metodo converge asintoticamente.

Batch vs trial learning.

Converge!!

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

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

Rimpiazza iterative Policy evaluation.
Rimane il passo di Policy iteration (improvement).



Sommario



Asynchronous solutions

Temporal differences