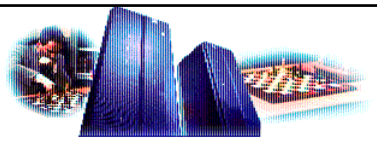
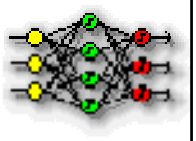




Sistemi Intelligenti
Reinforcement Learning: Temporal Difference

Alberto Borghese

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

Sommario

Asyncronous solutions

Temporal differences

Value Iteration

Policy iteration:

Iterative policy evaluation

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

$\forall s$ $\qquad \pi'(s_k) = \arg\max_a \sum_{s'} P^a_{s_k \to s'} \left[R^a_{s \to s'} + \gamma V^{\pi}(s')\right] = \arg\max_a Q(s, a)$

Value iteration:

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

Problemi

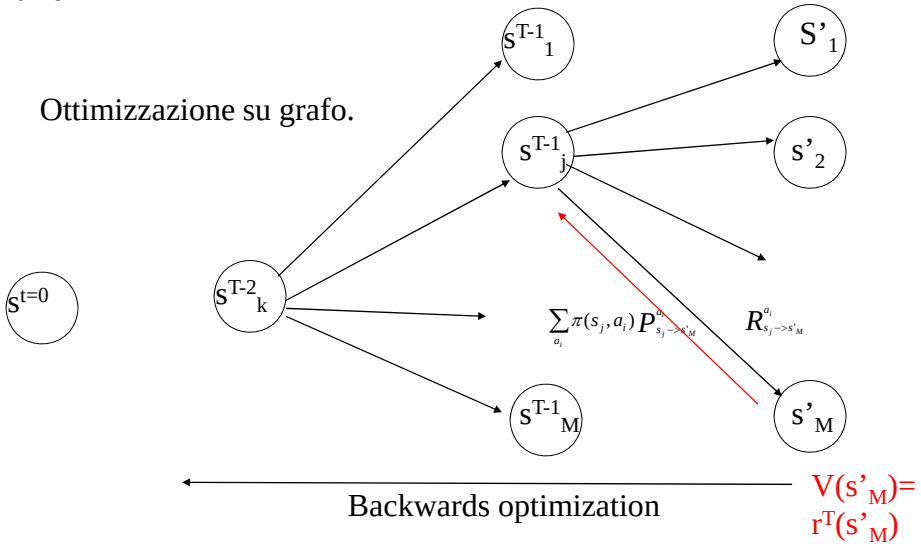
Framework: DP

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.

Problemi ad orizzonte finito: DP



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.

Evoluzione della metodologia

Policy iteration:

Iterative policy evaluation

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

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

∀s

$$\pi'(s_k) = \arg\max_a \sum_{s'} P^a_{s_k \to s'} \left[R^a_{s \to s'} + \gamma V^{\pi}(s')\right]$$

Value iteration:

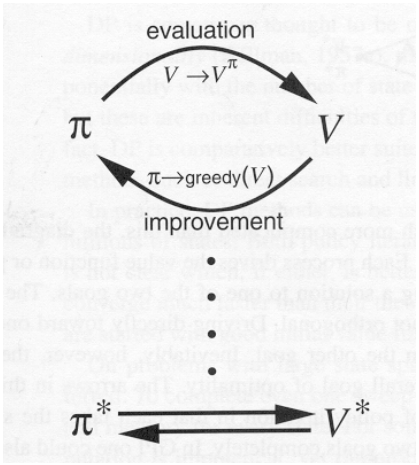
$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} P_{s \to s' | a} \left[R_{s \to s' | a} + \gamma V_k(s')\right]$$

∀s

Asynchonous solution:

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

1 stato



Riassunto

Generalized Policy iteration {
- Policy iteration
- Value iteration
- Asynchronous DP

Competizione e cooperazione -> V corretta e policy ottimale.

Sommario

Asyncronous solutions

Temporal differences

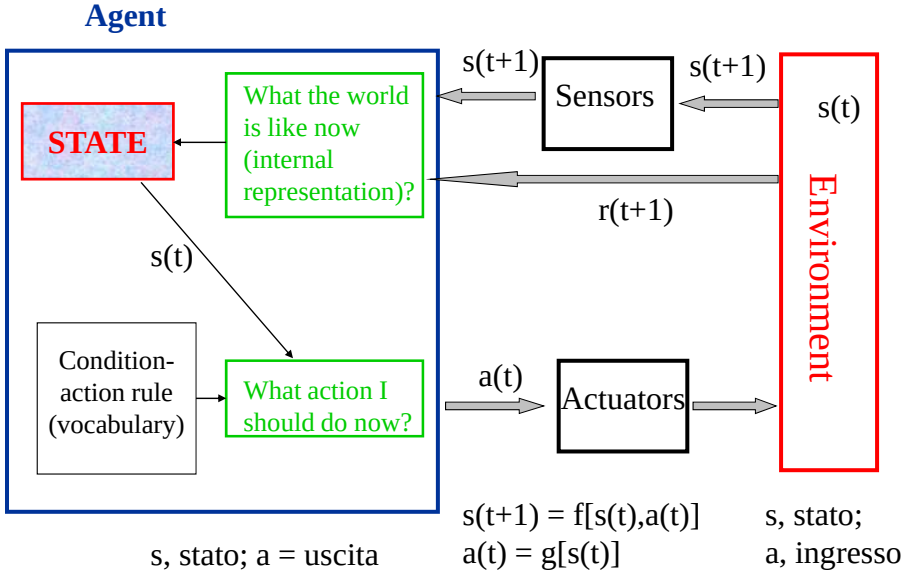
World RL competition

Started in NIP2006.

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

http://rl-competition.org

Schematic diagram of an agent



How About Learning the Value Function?

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

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

È una funzione dello stato.

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

$$\pi^*(s) = \arg\max_a \sum_{s'} P^a_{s->s'} \left[R^a_{s->s'} + \gamma V^{\pi}(s')\right]$$

Full backup, for all states

Value function iteration

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

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

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

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

Full backup, for all states

Asynchronous DP

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

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

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

•Miglioramento della policy

$$= \arg\max_a \sum_{s'} P^a_{s->s'} \left[R^a_{s->s'} + \gamma V^{\pi}(s') \right]$$

Non sono noti

Background su Temporal Difference (TD) Learning

Al tempo t abbiamo a disposizione:

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

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

Reward certo
Transizione certa
vengono misurati dall'ambiente

Come si possono utilizzare per apprendere?

Confronto con il setting associativo

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

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:

NewEstimate = OldEstimate + StepSize [Target – OldEstimate]
*NewEstimate = OldEstimate + StepSize * Error.*

StepSize = α = 1/k *a = cost*

Qual è la differenza introdotta dall'approccio DP?

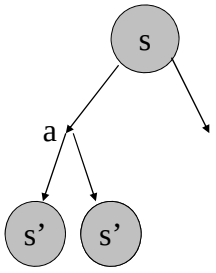
Un possibile aggiornamento

In iterative policy evaluation ottengo questo aggiornamento:

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

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)

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

$$V_{k+1}(s) = \left[r' + \gamma V_k(s')\right]$$ Quanto vale α?

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

Come calcolo $\Delta V_k(s)$?

TD(0) update

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

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

Da confrontare con la iterative policy evaluation:

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

Sample backup

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

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

Quanto vale α?

Confronto con il setting associativo

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

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:

NewEstimate = OldEstimate + StepSize [Target – OldEstimate]
*NewEstimate = OldEstimate + StepSize * Error.*

StepSize = α = 1/k *a = cost*

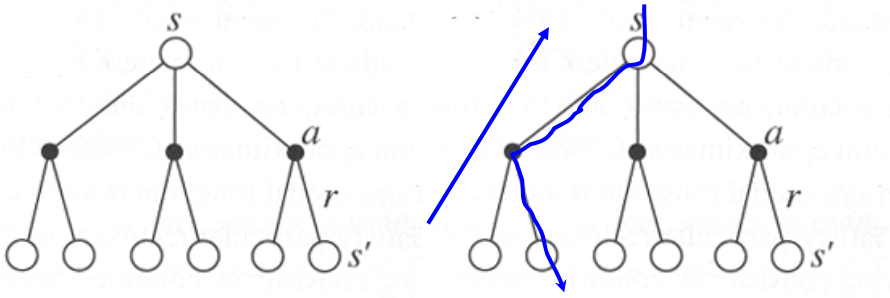
Qual è la differenza introdotta dall'approccio DP?

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

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

Inizializziamo V(s) = 0.
Inizializziamo la policy: π(s,a).da valutare
Repeat
{ s = s0;
Repeat // For each state until terminal state, analyze an episode
{ a = π(s);
s_next = NextState(s, a);
reward = Reward(s, s_next, a);
V(s) = V(s) + α [reward + γ V(s_next) – V(s)];
s = s_next;
} until TerminalState
} Until convergence of V(s) for policy π(s,a)

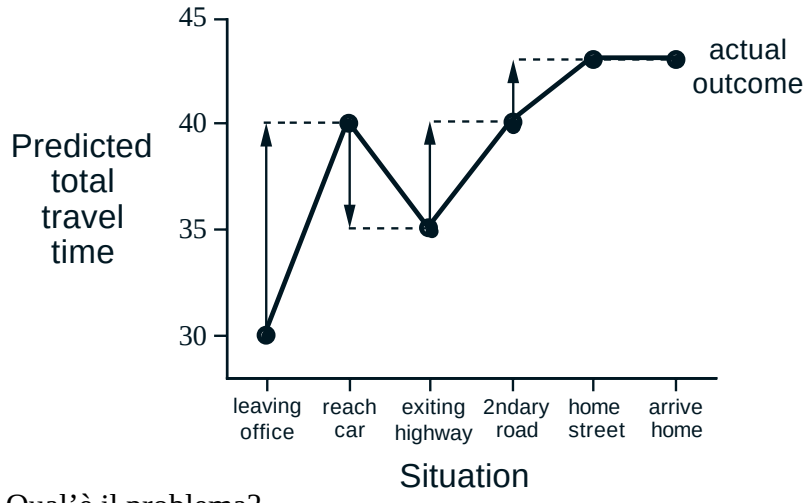
Esempio: valutazione della policy mediante TD

Obbiettivo: predirre la durata del percorso per tornare a casa.

Stato	Tempo trascorso	Tempo segmento (tempo medio)	Tempo previsto	Tempo totale
Esco dall'ufficio	0	0 (0)	30	30
Salgo in auto (nevica)	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*" α = cost.

Modifiche di V(s) run-time



Qual'è il problema?

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 $\alpha << 1$ 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 α = 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!!

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

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

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

Sommario

Asyncronous solutions

Temporal differences