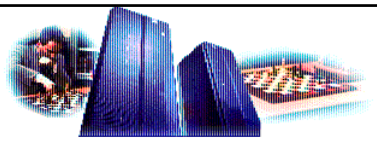
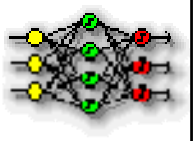




Sistemi Intelligenti
Reinforcement Learning:
Processi Markoviani e Value function

Alberto Borghese

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

Sommario

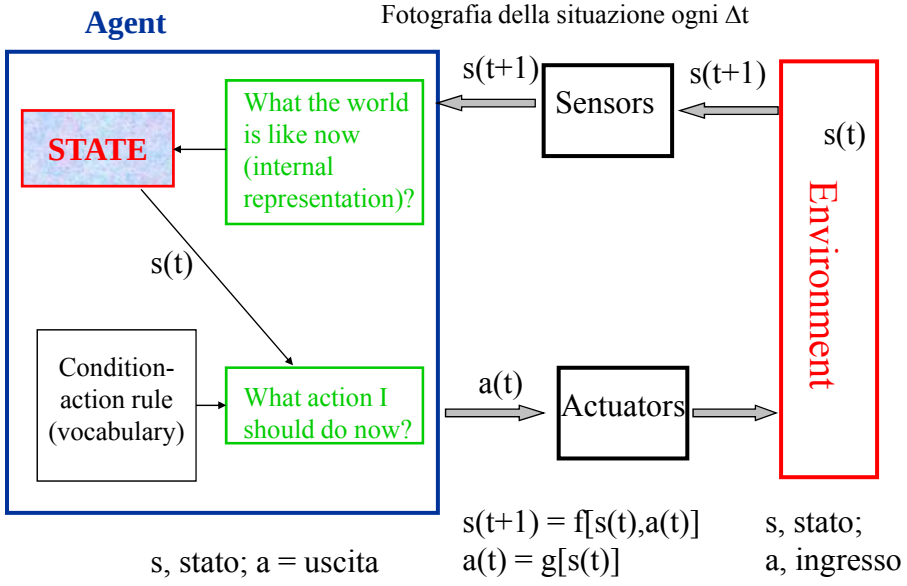
Il Reinforcement Learning in setting non associativi

Apprendimernto del gioco del Tris

Processi Markoviani.

La value function: ricompensa a lungo termine: formulazione ricorsiva.

Schematic diagram of an agent

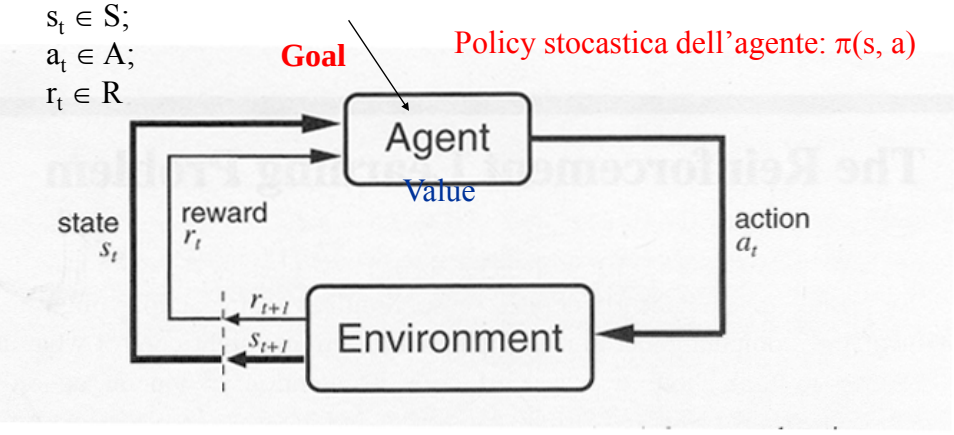


Interazione Agente-Ambiente

$s_t \in S$;
$a_t \in A$;
$r_t \in R$

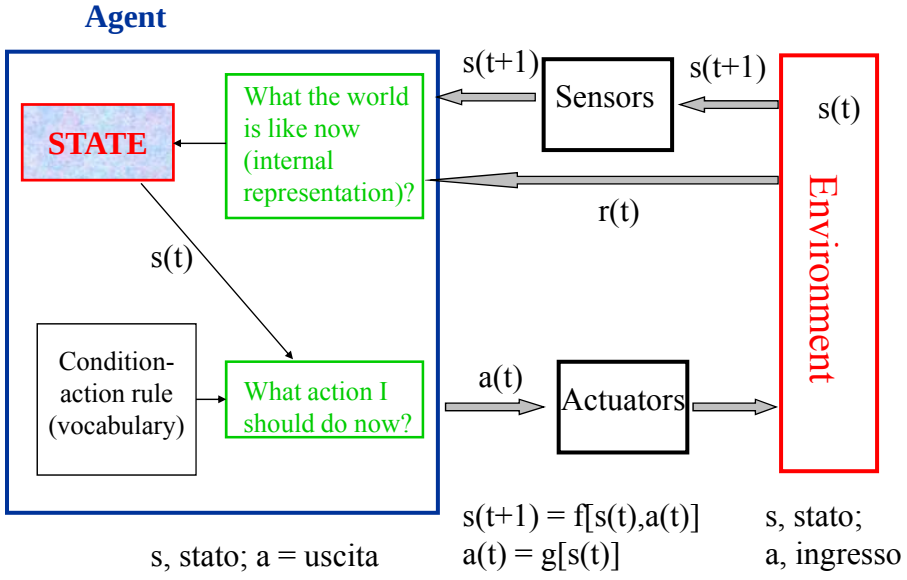
Goal

Policy stocastica dell'agente: $\pi(s, a)$



Il **reward** viene fornito dall'ambiente all'agente.
Il reward è un valore numerico, che viene fornito in certi istanti di tempo: $r_t = r_t(s_t, a_t, s_{t+1})$.

Schematic diagram of an agent



Gli elementi del RL

Goal. Obbiettivo che deve raggiungere l'agente. Si può aggiungere che l'agente deve raggiungere l'obbiettivo con una policy ottima.

Policy. Descrive l'azione scelta dall'agente: mapping tra stato (output dell'ambiente) e azioni dell'agente. Funzione di controllo. Le policy possono avere una componente stocastica. Viene utilizzata una modalità adeguata pe rappresentare il comportamento dell'agente (e.g. tabella, funzione continua parametrica…).

Reward function. Ricompensa **immediata**. Associata all'azione intrapresa in un certo stato. Può essere data al raggiungimento di un goal (esempio: successo / fallimento). E' uno scalare. Rinforzo primario, solitamente ***qualitativo***.

Value function. "Cost-to-go". Ricompensa a **lungo termine**. Somma dei reward: costi associati alle azioni scelte istante per istante + costo associato allo stato finale. ***Orizzonte temporale ampio.*** Rinforzo secondario.

Environment. Fornisce la reward function, fornisce l'input in base al quale l'agente aggiorna lo stato. Reagisce agli output dell'agente.

Meccanismo di apprendimento nel RL

Ciclo dell'agente (le tre fasi sono sequenziali):

1) Implemento una policy
2) Aggiorno la Value function
3) Aggiorno la policy.

Osservazioni

Formulazione generale che si adatta ad una grande quantità di problemi.

Agente = Controllore.
Tempo = tempo, ma anche stadio della decisione, del planning….
Azione = forza, voltaggio, decisioni…..
Stato = situazione = misura di grandezze fisiche, di grandezze interne, stato mentale,….
Pre-processing di misure fisiche. E' importante per un efficiente RL.
Ambiente = **tutto quanto non è modificabile direttamente dall'agente**. Può essere noto o meno.
Reward = viene generato all'esterno dell'agente.
Value = viene stimata all'interno dell'agente.

Caratteristiche dello stato

Policy di un agente: $\pi(s, a)$.

Caratteristiche dello stato, s:
• Contiene gli stimoli pre-elaborati a partire dagli stimoli semplici misurati sull'ambiente.
• Gli stimoli pre-elaborati analizzano una sequenza temporale di stimoli semplici.
• Lo stato deve potere essere misurato dall'agente.
• Lo stato deve essere efficiente per il raggiungimento del goal.

Come rappresento s? Memorizzo la sequenza temporale degli stimoli semplici di interesse?

Reward e Obbiettivi

Il reward è "esterno" all'agente.

Massimizzare la ricompensa a lungo termine, Value, cumulando le ricompense istantanee: $r_t(a(t)) \in R$.

Defininendo una ricompensa che viene massimizzata solamente quando il goal viene raggiunto, possiamo ottenere che l'agente impari il task (raggiunga il goal).

Collegamento tra reward e goal.

Il reward consente di comunicare COSA si vuole ottenere; nulla è detto sul COME.

Value function

Cosa si intende per ricompensa a lungo termine?
Questa è rappresentata dalla **Value Function**; cosa rappresenta?

Al tempo t, data una certa policy: $\pi(s, a)$, la ricompensa sarà una funzione dei reward negli istanti di tempo successivi a t, ad esempio:

$$R^{\pi}{}_t = r_{t+1} + r_{t+2} + r_{t+3} + \ldots\ldots r_T$$

Terminal State

Quando è adeguata?

- Problemi ad orizzonte finito (episodic tasks, a terminal state is defined).
- Problemi stazionari.

Infinite horizon problems (continuing tasks)

Il concetto fondamentale è il "**discount**".

Discounted reward o discounted return:

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+2} + \ldots\ldots = \sum_{k=0}^{+\infty} \gamma^k r_{t+(k+1)}$$

Dove $0 \leq \gamma \leq 1$ è il "discount rate".
Present value of future rewards.

$$R_t \to \frac{r}{1-\gamma} \quad if \quad r_t = r_{t+k} \quad \forall k$$

Relazione con il caso non-stazionario nel setting non-associativo?

Cosa succede se $\gamma \to 0$ *e* $\gamma \to 1$?

Sommario

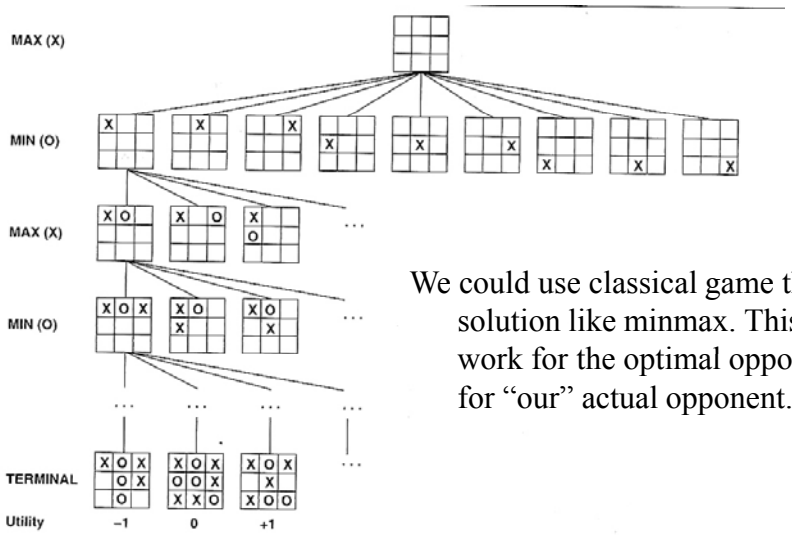
Il Reinforcement Learning in setting non associativi

Apprendimernto del gioco del Tris

Processi Markoviani.

La value function: ricompensa a lungo termine: formulazione ricorsiva.

Apprendimento della strategia per il gioco del tris



We could use classical game theory solution like minmax. This can work for the optimal opponent, not for "our" actual opponent.

Come impostare il problema mediante RL?

State – configuration of ‘X’ and ‘O’

Value (of the state) – probability of winning associated to that state (= configurazione).

Which is the probability of a state in which we have 3 ‘X’ in a row (or column or diagonal)?
Which is the probability of a state in which we have 3 ‘0’ in a row (or column or diagonal)?

We set all the other states to 0.5.

We set the initial policy at random and the initial value of the Value function to zero.

Come decidere la mossa?

Supponiamo di essere in una configurazione non terminale.

Per ogni mossa valida, possiamo valutare il valore della nuova configurazione che si verrebbe a trovare. Come?

Possiamo occasionalmente scegliere delle mosse esploratorie. Quando non ha senso scegliere delle mosse esploratorie?

Dobbiamo perciò capire qual’è il valore delle diverse configurazioni della scacchiera.

Come stimare il valore di ogni configurazione?

$V(s(t)) \leftarrow V(s(t)) + \alpha\,[V(s(t+1)) - V(s) + R]$

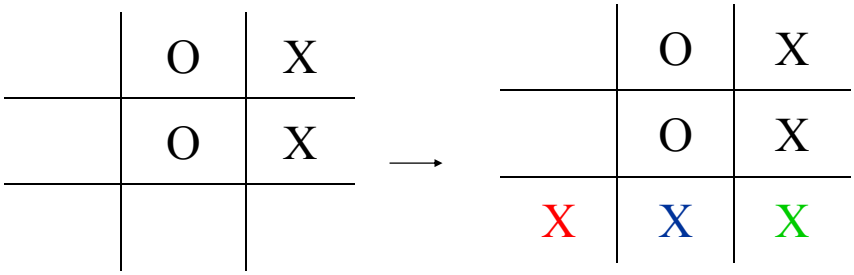
Tendo ad avvicinare il valore della mia configurazione al valore della configurazione successiva + reward locale.

Esempio di *temporal difference learning*.

Diminuendo α con il numero di partite, la policy converge alla policy ottima per un avversario fissato (cioè che utilizzi sempre la strategia, ovvero sia la stessa distribuzione statistica di mosse).

Diminuendo α con il numero di partite, ma tenendolo > 0, la policy converge alla policy ottima anche per un avversario che cambi molto lentamente la sua strategia.

Esempio



X – scelta perdente -> dopo la mossa dell'avversario ho uno stato con value function = 0.
X – scelta neutrale -> dopo la mossa dell'avversario ho uno stato con value function intermedia (pareggio).
X – scelta vincente -> vado in una configurazione con value function = 1.

Cambio la policy e rivaluto la Value function.

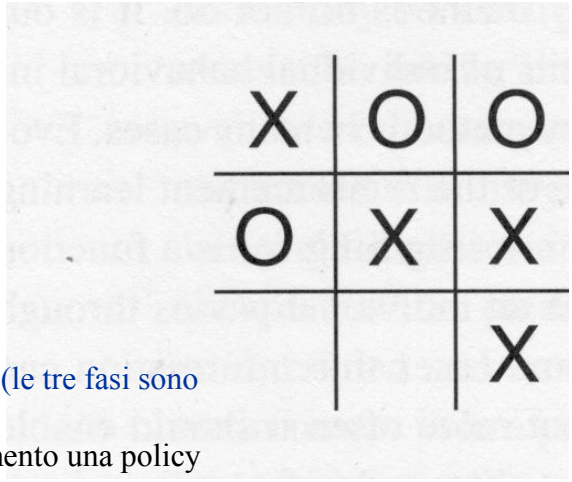
Come valutare le mosse

- Assegno un valore alle diverse configurazione sulla scacchiera

$$V(s) = V(s) + \alpha\,(R + \gamma V(s'))$$

α-> 0 con l'apprendimento: policy consolidata
α -> ε con l'apprendimento: possibilità di adattarsi ad altri giocatori.

Cosa fa l'agente?



Ciclo dell'agente (le tre fasi sono sequenziali):

1) Implemento una policy
2) Aggiorno la Value function
3) Aggiorno la policy.

Riflessioni su RL ed il gioco del tris

Supponete che l'agente dotato di RL giochi, invece che con un avveresario, contro sé stesso. Cosa pensate che succeda? Secondo voi imparerebbe una diversa strategia di gioco?

Molte posizioni del tris sembrano diverse ma sono in realtà la stessa per effetto delle simmetrie. Come si può modificare l'algoritmo di RL (definizione dello stato) per sfruttare le simmetrie? Come si può migliorare il meccanismo di apprendimento?

Supponiamo che l'avversario non sfrutti le simmetrie. In questo caso noi possiamo sfruttarle? E' vero che configurazioni della scacchiera equivalenti per simmetria devono avere la stessa funzione valore.

Potete pensare a modi per migliorare il gioco dell'agente? Potete pensare a metodi migliori (più veloci, più robusti….) perché un agente impari a giocare a tris?

Sommario

Il Reinforcement Learning.

Apprendimento del gioco del tris

Processi Markoviani.

La value function: ricompensa a lungo termine: formulazione ricorsiva.

Environment Markoviano

Una variabile di stato (non funzione del tempo), che riassume le informazioni sulla storia del task, utili all'agente per agire, è detta variabile Markoviana.

Formalizziamo. Supponiamo s ed r variabili discrete appartenenti ad un insieme finito di valori.

$\Pr\{s_{t+1} = s' \mid s_t, a_t ; s_{t-1}, a_{t-1}; \ldots\ldots ; s_0, a_0 \}$

Se lo stato è Markoviano:

$\Pr\{s_{t+1} = s' \mid s_t, a_t\}$

NB: Non sempre $Pr\{s_{t+1} = s' \mid s_t, a_t\}$ è nota!

Reinforcement Markoviano

Reward stocastico.

$\Pr\{r_{t+1} = r' \mid s_t, a_t , s_{t+1}; s_{t-1}, a_{t-1} , r_t; \ldots\ldots ; s_0, a_0 , r_1\}$

Se lo stato è Markoviano:

Reward stocastico: $\Pr\{r_{t+1} = r' \mid s_t, a_t, s_{t+1}\}$
Reward deterministico: $r_{t+1} = r(s_t, a_t, s_{t+1})$.

L'ambiente ha completamente proprietà Markoviane.

I modelli Markoviani sono modelli molto generali!

Markov decision process

(Finite) Markov Decision Process.

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

Probabilità di transizione

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

Descrizione della dinamica dell'ambiente
Azione che dall'esterno viene imposta all'ambiente

Approssimazione

L'ipotesi di ambiente e rinforzo Markoviani possono essere soddisfatte in modo approssimato.

Environment – cart + pole.

Stato – Posizione e velocità di cart e di pole, attuatore che trasforma la corrente per il motore in forza di traslazione.

Agente – Controllore del motore che pilata la traslazione (corrente).

Reward – Cade / non_cade - 0 /1.

Value Function ?

Postural control is a complex problem

Complex system: multi-input – multi-output (each leg has 56 major muscle groups).

It is a non-linear system. High coupling between body segments (e.g. biarticular muscles).

Muscles bandwidth is limited.

The control system introduces delays, increasing from the periphery to the CNS.

Classical control theory is “difficult”.

Nevertheless, we learn upright posture in the very first year of our life.

Sommario

Il Reinforcement Learning.

Processi Markoviani.

La value function: ricompensa a lungo termine: formulazione ricorsiva.

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}

Goal: 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.

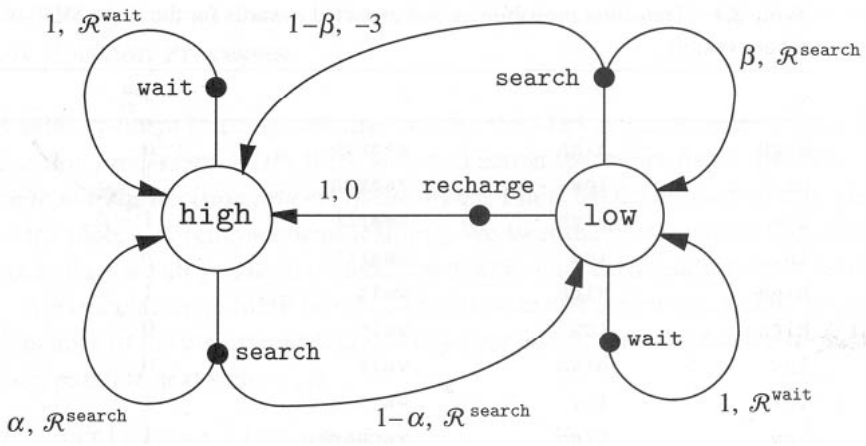
0 se il robot va autonomamente a ricaricarsi.

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

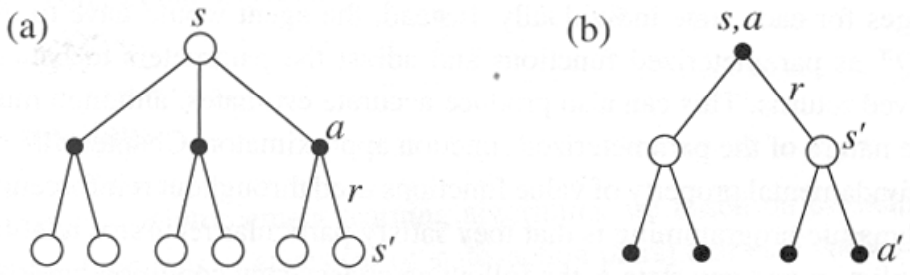
Forma tabellare

s	s'	a	$P_{s->s'|a}$	$R_{s->s'|a}$
alta	alta	ricerca	α	R^{search}
alta	bassa	ricerca	$1 - \alpha$	R^{search}
bassa	alta	ricerca	$1 - \beta$	-3
bassa	bassa	ricerca	β	R^{search}
alta	alta	attesa	1	R^{wait}
alta	bassa	attesa	0	R^{wait}
bassa	alta	attesa	0	R^{wait}
bassa	bassa	attesa	1	R^{wait}
bassa	alta	ricarica	1	0
bassa	bassa	ricarica	0	0
alta	Non esiste	ricarica	X	X

State Transition Graph



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.

Il nostro filo logico

La Value function ci serve per decidere l'azione migliore.
Per calcolare la Value function devo collezionare reward futuro.

Come se ne esce?

Determinazione algebrica della Value Function
Determinazione "esplorativa" della Value Function
Determinazione della Value Function e scelta della policy via via sempre più intrecciate tra loro.

Sommario

Il Reinforcement Learning.

Processi Markoviani.

La value function: ricompensa a lungo termine: formulazione ricorsiva.