

# Sistemi Intelligenti Reinforcement Learning

Alberto Borghese

Università degli Studi di Milano  
Laboratorio di Sistemi Intelligenti Applicati (AIS-Lab)

Dipartimento di Informatica

[alberto.borghese@unimi.it](mailto:alberto.borghese@unimi.it)

*Chapter 2 – Barto Sutton*



## Riassunto



- **Gli agenti ed il Reinforcement Learning**
- Apprendimento nel condizionamento classico
- Implementazione ricorsiva
- Caso non stazionario



## L'agente intelligente

- Inizialmente l'attenzione era concentrata sulla progettazione dei sistemi di "controllo". Valutazione, sintesi...
- L'intelligenza artificiale e la "computational intelligence" hanno consentito di spostare l'attenzione sull'apprendimento delle **strategie di controllo** e più in generale di comportamento (**planning, control, search, choice, prediction ...**). Sono tutti modelli utilizzati in modo forward.
- **Macchine dotate di meccanismi (algoritmi, SW), per apprendere queste strategie (identificazione dei modelli).**
- **RPA – Robot Process Automation**



## Come acquisire l'intelligenza

L'agente **interagisce con l'ambiente** che fornisce **stimoli**.

Esperimenti di Ivan Pavlov (Nobel medicina nel 1904).



Quando un cane vede la ciotola del cibo inizia a salivare.



Si fa partire un metronomo poco prima della presentazione del cibo.



Dopo poco tempo il cane inizia a salivare appena vede/sente il metronomo.

Alla base del dog training. Perché non applicarlo anche alle macchine?





## Osservazioni



Triggering of behaviour (or choice) that is most adequate to what?

Key is the **reward**. I cani sono molto golosi...

Lo stimolo condizionato funziona, perchè per gli esseri viventi (e la “lentezza” della risposta del sistema nervosa) è fondamentale **anticipare**.



## Framework dell'apprendimento

- Un agente interagisce con l'ambiente.
- Può scegliere un'azione sull'ambiente tra un insieme continuo o discreto (qui il cane può mangiare, oppure può continuare a correre, ...).
- L'agente monitora l'ambiente (**input**). In questo caso il cane vede (e annusa) l'ambiente intorno, ma sente anche il metronomo.
- La scelta dell'azione è non banale e richiede un certo grado di “intelligenza”. Il cane «capisce» che mangiare dà più reward di continuare a correre.
- L'agente è in grado di modificare il proprio comportamento (dopo un po' di tentativi, il cane inizia a salivare (a prepararsi al pranzo) al suono del metronomo).

**Un agente sceglie la propria azione in funzione del reward che ottiene dall'ambiente.**

*Il reward deve essere sufficientemente vicino nel tempo (associabile all'azione)*

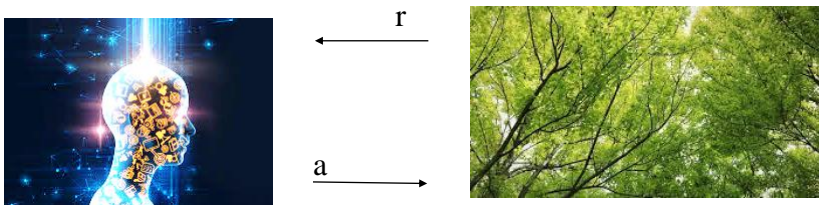




## Apprendimento mediante RL



- L'agente deve scegliere un'azione,  $a$  (tra diverse azioni possibili discrete o continue).
- L'agente misura le ricompensa,  $r$  (reward) associate alla sua azione.



- L'agente deve scoprire autonomamente quale azione dà un reward massimo.
- Apprendimento all'interno di un singolo trial.



## Reinforcement learning



Spesso si ha a disposizione solamente un'informazione qualitativa di ricompensa, detta **reward**, (a volte binaria, giusto/sbagliato successo/fallimento), puntuale fornita dall'ambiente.

Apprendimento con rinforzo è un apprendimento attraverso reward.

*L'informazione disponibile di ricompensa si chiama **segnale di rinforzo**. Non dà alcuna informazione su come aggiornare il comportamento dell'agente (e.g. i parametri), non è **istruttiva**. Non è possibile definire una funzione costo o un gradiente.*

*Il reward è un'informazione qualitative che dà una **valutazione**.*

*Obiettivo: creare degli agenti "intelligenti" che abbiano una "machinery" per apprendere dalla loro esperienza, dalla valutazione ricavare delle istruzioni su come migliorare le azioni.*

Ambiente viene considerato stocastico (non si conosce il suo comportamento e il suo comportamento non è stereotipato).



## Reward



Il reward è il cibo!

Il cibo non dice al cane esplicitamente che è meglio mangiare di continuare a correre.

Il cibo dà una soddisfazione maggiore del correre.

Di conseguenza il cane, decide di mangiare invece di andare a correre in futuro.

Il reward non istruisce il cane su cosa fare (molto diverso dall'apprendimento supervisionato che vedremo: classificazione e regressione)

Il reward dà un'informazione **qualitativa** sull'azione, non **istruttiva**.

E' il cane che deve capire come utilizzare questa informazione qualitativa per migliorare le proprie azioni.



## Esempio applicativo

Dove posizionare gli avvertimenti pubblicitari? Quanto valgono?



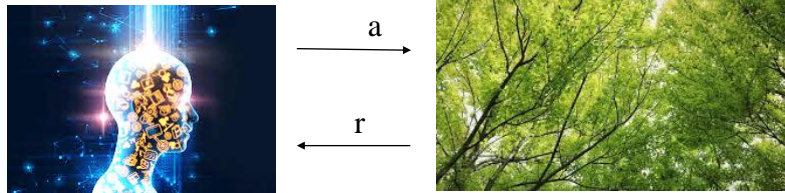
Ogni click sull'avvertimento è un rinforzo positivo.

L'agente prova diverse posizioni. Cerca di massimizzare il reward, cioè il numero di click.



## Come migliorare l'azione

- L'agente deve scegliere un'azione,  $a$  (tra diverse azioni possibili discrete o continue).
- L'agente misura la ricompensa,  $r$  (reward) associata alla sua azione.



Evaluate the implemented solution.

- Success or fail? Adequate or not adequate?
- How much adequate? How to measure the success or failure of the performance?
- Optimization of the performance to create better agents.



## Riassunto

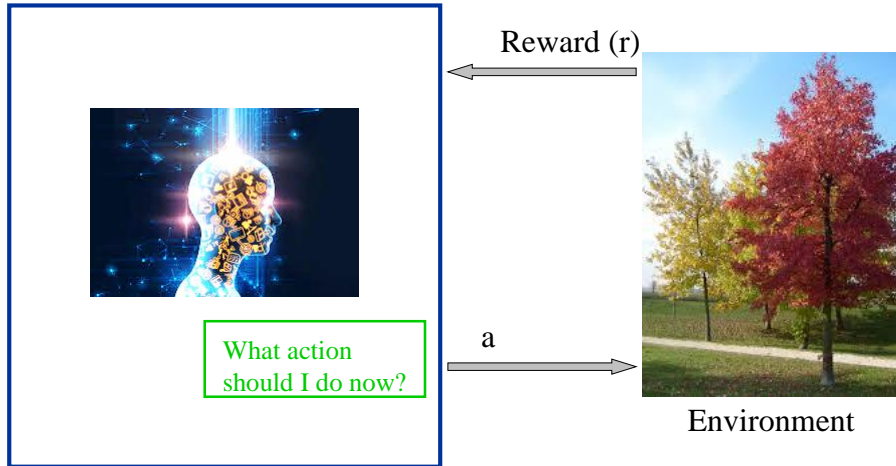
- Gli agenti ed il Reinforcement Learning
- **Apprendimento nel condizionamento classico**
- Implementazione ricorsiva
- Caso non stazionario



## Schematic diagram of an agent



### Agent



## Il problema del "n-Armed bandit"



Situazione iniziale sempre uguale.

Scelta tra  $n$  azioni.

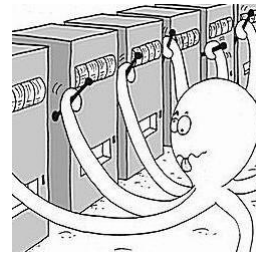
Il ciclo: azione + reward esaurisce l'episodio - trial)

La richiesta di scegliere viene ripetuta più volte nel tempo. Ogni volta si riparte dalla stessa situazione (**problema stazionario**).

La ricompensa è **stocastica** (e.g. slot machine).

Obiettivo: viene massimizzata la ricompensa a lungo termine.

Come selezionare l'azione che fornisce la massima ricompensa a lungo termine se non si conosce la statistica delle macchine?





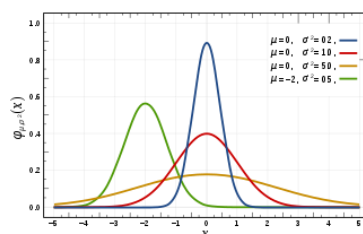
## Slot machine stocastica

Il reward della slot machine è completamente definito dalla densità di probabilità associata alla macchina.

Si suppone la densità di probabilità del reward costante nel tempo e **NON NOTA**.

Per semplicità si suppone che la densità di probabilità sia descrivibile da una funzione analitica, ad esempio una Gaussiana. In questo caso la densità di probabilità è definita dai parametri della Gaussiana: media e standard deviation.

*Potrei estrarre un reward +0.1 dalla Gaussiana verde (ben dentro la coda destra) e -1 dalla Gaussiana blu (dentro la coda sinistra) con probabilità diversa da zero.*



## Come massimizzare la ricompensa

**Consento all'agente di avere memoria.**

Memorizzo il valore associato alle diverse azioni.

Posso ad un certo punto scegliere **SEMPRE** l'azione che mi ha dato la **Ricompensa TOTALE maggiore**.

Scelgo cioè l'azione **GREEDY (Greedy = Goloso)**.

Sfruttiamo cioè la conoscenza raggiunta fino a quel momento (**Exploitation**).

Perché dovremmo scegliere un'azione che non appare la migliore (NON GREEDY)?

Perché dovremmo esplorare altre azioni che non hanno dato la ricompensa maggiore?





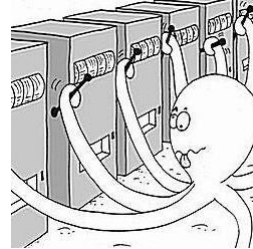
# Exploration



## Perchè esploriamo soluzioni diverse?

La ricompensa **non è deterministica**. Potremmo ottenere di più con altre azioni.

Quello che conta non è la ricompensa istantanea ma la somma delle ricompense ottenute nel tempo. Voglio massimizzare il mio guadagno a lungo termine, ad esempio dopo avere premuto 1000 volte un braccio.



Occorre quindi mantenere un istinto ad esplorare azioni diverse.

Il bilanciamento di “exploration” e di “exploitation” è un compito complesso.



# Exploration vs Exploitation



Esplorazione (**exploration**) dello spazio delle azioni per scoprire le azioni migliori. Un agente che esplora solamente raramente troverà una buona soluzione.

Le azioni migliori vengono scelte ripetutamente (**exploitation**) perchè garantiscono ricompensa (**reward**). Se un agente non esplora nuove soluzioni potrebbe venire surclassato da nuovi agenti più dinamici.

Occorre non interrompere l’esplorazione.

Occorre un approccio statistico per valutare le bontà delle azioni.

**Exploration ed exploitation vanno bilanciate.** Come?

Qual è il comportamento ottimale?



## La Value Function e la scelta delle azioni



Razionale: associare un **valore** alle diverse azioni.

Il reward istantaneo non può rappresentare il valore perchè è 1 singolo campione estratto da una distribuzione statistica.

Devo cercare qualcosa di più significativo. Ad esempio il **reward medio**.

Posso scegliere come valore, il risultato della media campionaria: **funzione valore** (value function).

La **funzione valore** viene costruita dall'**agente** a partire dai reward istantanei forniti dall'**ambiente**. E' l'elemento chiave nell'apprendimento con rinforzo.

La funzione valore caratterizza l'apprendimento con rinforzo.



## La Value Function e la scelta delle azioni



Posso selezionare n-azioni:  $a = a_1 \dots a_n$ . **Policy**.

In funzione dell'azione, la macchina mi darà un **reward**,  $r(a_k)$ , non noto a-priori.

Posso valutare le diverse azioni al tempo t attraverso la media campionaria dei reward associati a ciascuna azione: **funzione valore** (value function):

$$Q_{t(a_k)} = \frac{r_{1(a_k)} + r_{2(a_k)} + r_{3(a_k)} + \dots + r_{N(a_k)}}{N(a_k)}$$

Per il teorema del limite centrale il valore medio campionario dei reward tende al valore medio della distribuzione dei reward per  $N \rightarrow \infty$ .

$$\lim_{N(a_k) \rightarrow \infty} Q_{t(a_k)} = Q^*(a_k) = \mu_k$$

Voglio scegliere  $a_k$  che massimizza il valore di  $a_k$ :  $Q(a_k)$ .

$Q(a_k)$  è un'approssimazione di  $\mu_k$ .



## Caratteristiche della Value Function



Value function calcolata come media campionaria:

$Q_t(a_k) \rightarrow Q^*(a_k)$  per  $t \rightarrow \infty$

$Q_t(a_k) = 0$   $t = 0$ . Nessuna stima disponibile.

**Come calcolo**  $Q(a)$ ?

Selezionare l'azione,  $a_k$ , che dà all'istante  $t$ , la massima Value Function al tempo  $t$ :

$$a^* : Q_t(a^*) = \max_{a_k} \{Q_t(a_k)\}$$

Così viene **EXPLOITED** la conoscenza accumulata, è una politica GREEDY.

Non vengono esplorate soluzioni alternative, calcolo solo  $Q(a_k)$

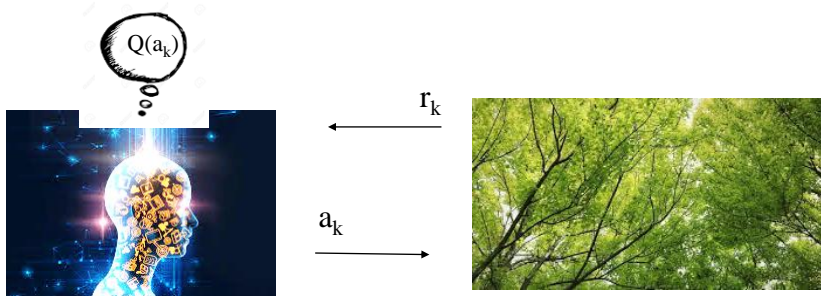
Come si può formalizzare un'alternativa?



## Interazione rivista



- L'agente deve scegliere un'azione,  $a$  (tra diverse azioni possibili discrete o continue).
- L'agente misura la ricompensa,  $r$  (reward) associate alla sua azione.
- Apprendimento all'interno di un singolo trial.
  
- L'agente costruisce una stima del valore di ciascuna azione,  $a_k$ ,  $Q(a_k)$ .
- Prende una decisione in base a  $Q(a_k)$ , reward secondario, a lungo termine.





## Politiche $\epsilon$ -Greedy



Scelgo l'azione che dà al  $Q_t$  massima (greedy) con probabilità  $P = 1 - \epsilon$

Con probabilità piccola,  $\epsilon$ , viene scelta un'azione diversa.

$$a^* : Q_t(a^*) = \max_{a_k} \{Q_t(a_k)\} \quad P = 1 - \epsilon$$

$$a \neq a^* \quad P = \epsilon$$

L'azione non greedy viene scelta con probabilità uniforme tra le  $n$  possibili azioni a disposizione.

$$Q_t(a_k) \rightarrow Q^*(a_k) \quad t \rightarrow \infty$$

Near-greedy policy. *Mantengo una certa capacità di esplorazione*



## Beyond $\epsilon$ -greedy: pursuit methods



Dopo ogni episodio, la probabilità di scegliere un'azione viene aggiornata:

- L'azione associata alla value function migliore **aumenta la probabilità** di essere prescelta.
- La probabilità di scegliere le altre azioni viene **decrementata**.
- Chiamo  $\pi(a)$  la probabilità di scegliere l'azione  $a$ .

$$\pi_{t+1}(a^*_{t+1}) = \pi_t(a^*_{t+1}) + \beta [1 - \pi_t(a^*_{t+1})]$$

$$\pi_{t+1}(a) = \pi_t(a) + \beta [0 - \pi_t(a)] \quad \text{for } a \neq a^*_{t+1}$$

The preference for an action is always “**pursuing**” (inseguendo) the action that is greedy according to the current action-value estimate.

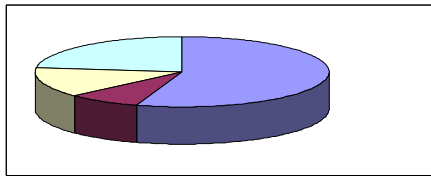
La probabilità di scegliere l'azione “migliore” aumenta con  $t$  (diminuisce l'exploration, aumenta l'exploitation).



## Approccio generale della roulette

Le azioni vengono selezionate proporzionalmente ad una misura di performance nel momento attuale. Migliore essa è e più alta è la probabilità di selezione.

1. Si immagini una roulette di area unitaria, dove siano sistemate tutte le possibili azioni,.
2. La dimensione del settore della roulette associata a ogni azione è proporzionale al valore di misura di performance di ciascuna azione.
3. La pallina viene lanciata all'interno della roulette e la possibilità che cada in una qualsiasi posizione è uniforme.
4. L'azione in corrispondenza della quale si ferma è quella selezionata



In questo esempio ci sono 4 possibili azioni.

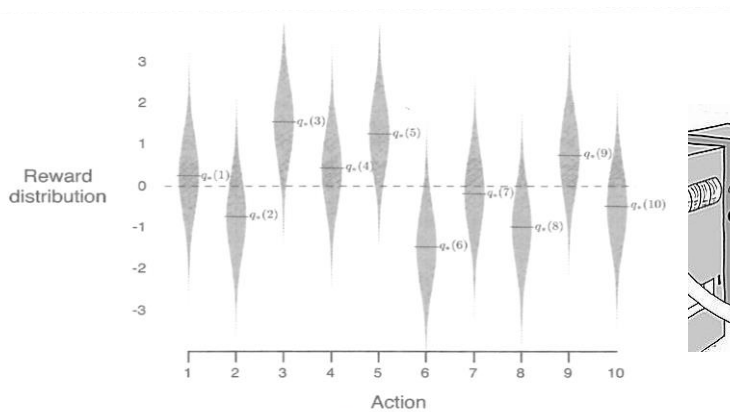


## n-armed bandit problem

n-armed bandit problem:  $n = 10$ :  $a = a_1, a_2, \dots, a_k, \dots, a_{10}$ .

### Reward:

- Valor medio di ogni azione viene estratto da una Gaussiana con media zero e varianza = 1.
- Valore del reward estratto da una Gaussiana con media definite sopra e varianza = 1.



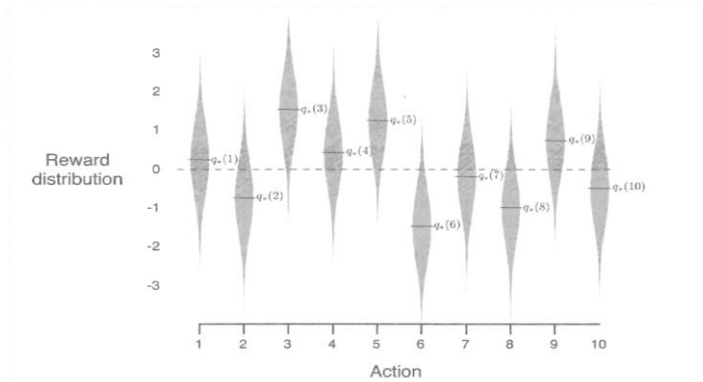


## Analisi del problema

Iniziamo prendendo l'azione 4:  $a_{t=0} = a_4$ . Supponiamo di ricevere un reward pari a 0.1.

$$Q(a_1) = Q(a_2) = Q(a_3) = Q(a_5) = Q(a_6) = Q(a_7) = Q(a_8) = Q(a_9) = Q(a_{10}) = 0 \quad Q(a_4) = 0.1$$

$$a^* : Q_t(a^*) = \max_{a_k} \{Q_t(a_k)\} \longrightarrow \text{We may be stuck on } a_4.$$



A.A. 2023-2024

[borghese.di.unimi.it](http://borghese.di.unimi.it)



## Esempio: risultati su 10-armed bandit

n-armed bandit problem:  $n = 10$ :  $a = a_1, a_2, \dots, a_k, \dots, a_{10}$ .

Eseguo 2000 task (esperimenti).

Per ogni task (esperimento), eseguo 1000 volte la scelta di un'azione:

$$t = t_1, t_2, \dots, t_{1000}$$

$$a = a(t_1), a(t_2), \dots, a(t_{1000})$$

$$r = r(a(t_1)), r(a(t_2)), \dots, r(a(t_{1000}))$$

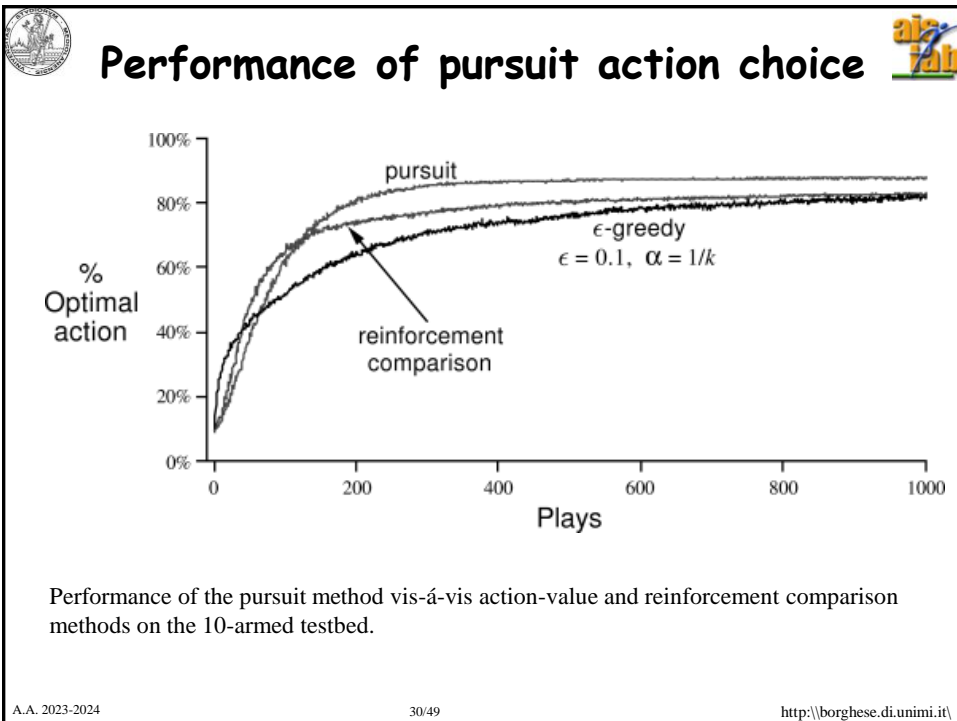
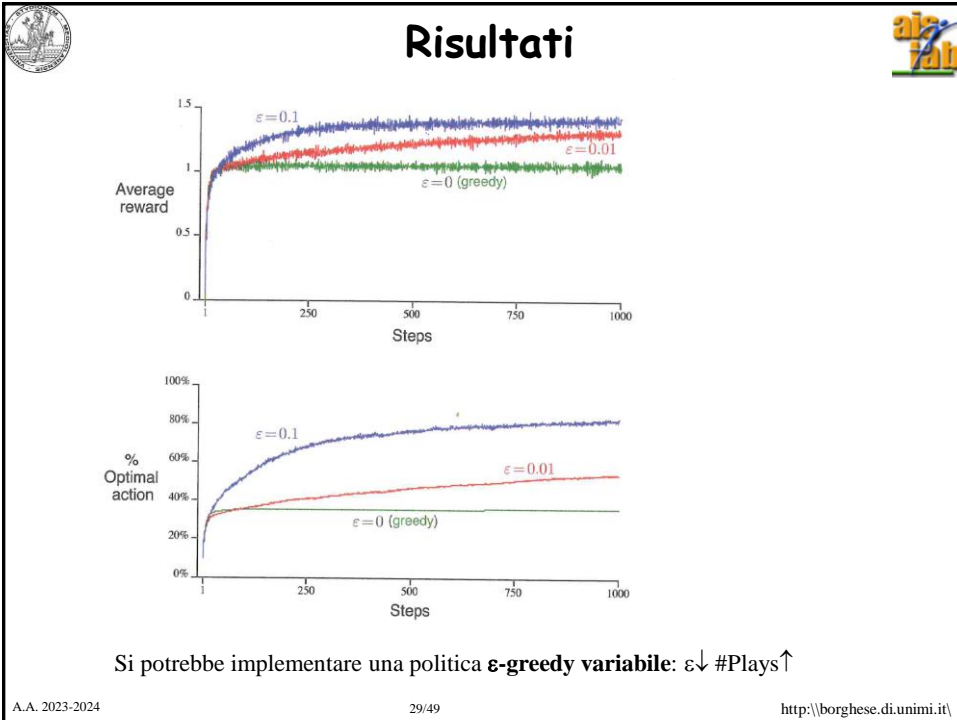
Valuto la performance dopo le 1000 giocate di ogni task (trial).

Viene fatta la media sulla performance (total reward) calcolata alla fine dei 2000 task.

A.A. 2023-2024

28/49

<http://borghese.di.unimi.it>





## Domande



Supponiamo che la distribuzione da cui si sceglie il valore medio del reward abbia varianza nulla. Quale metodo funziona meglio: Greedy o  $\epsilon$ -Greedy?

Supponiamo che la distribuzione da cui si sceglie il valore medio del reward abbia varianza maggiore (e.g. = 10). Cosa succede? Quale metodo si comporterebbe meglio?

In quali altre condizioni sarebbe utile avere esplorazione?



## Riassunto



- Gli agenti ed il Reinforcement Learning
- Apprendimento nel condizionamento classico
- Implementazione ricorsiva
- Caso non stazionario





## Calcolo ricorsivo di $Q(.)$

$$Q_i(a_k) = \frac{r_1 + r_2 + \dots + r_{N(a_k)}}{N(a_k)}$$

Occorre scegliere un algoritmo che calcoli  $Q_i(.)$  con un piccolo carico computazionale e di memoria.

Supponiamo di fare Exploitation dell'azione  $a_k$ . Calcoliamo la media dei reward dopo  $N$  reward e la chiamiamo  $Q_N(a_k)$ .  $Q_N(a_k)$  coinciderà con la media delle prime  $N(a_k)$  ricompense associate all'azione  $a_k$ :

$$Q_{N(a_k)} = \frac{r_1(a_k) + r_2(a_k) + r_3(a_k) + \dots + r_{N(a_k)}}{N(a_k)}$$

Scegliendo ancora  $a_k$ , otteniamo il  $r_{N+1}(a_k)$  e il seguente valore di  $Q$  dopo  $N+1$  reward:

$$Q_{N+1(a_k)} = \frac{r_1(a_k) + r_2(a_k) + r_3(a_k) + \dots + r_N(a_k) + r_{N+1}(a_k)}{(N+1)(a_k)}$$



## Determinazione ricorsiva di $Q_N$

$$Q_N(a_k) = \frac{r_1 + r_2 + \dots + r_N}{N}$$

$$Q_{N+1}(a_k) = \frac{r_1 + r_2 + \dots + r_N + r_{N+1}}{N+1}$$

$$Q_{N+1} = \frac{r_1 + r_2 + \dots + r_N}{N+1} + \frac{r_{N+1}}{N+1} = \frac{Q_N N}{N+1} + \frac{r_{N+1}}{N+1} =$$

$$\frac{Q_N(N+1-1)}{N+1} + \frac{r_{N+1}}{N+1} = \frac{Q_N(N+1) - Q_N}{N+1} + \frac{r_{N+1}}{N+1} \Rightarrow$$

$$Q_{N+1} = Q_N - \frac{Q_N}{N+1} + \frac{r_{N+1}}{N+1} \leftarrow \text{Dipende da } N+1$$

Non dipende da  $N+1$

$$Q_{N+1} = Q_N + \frac{1}{N+1} (r_{N+1} - Q_N)$$



## Osservazioni su $Q_N$



$$Q_{N+1} = Q_N + \frac{1}{N+1} (r_{N+1} - Q_N)$$

L'agente ragiona sul trial attuale.

Dipende solo da quantità:

- all'istante  $N$  abbiamo  $Q_N$
- all'istante  $N+1$  abbiamo  $N+1$ ,  $r_{N+1}$

NB  $N$  è il numero di volte in cui è stata scelta  $a_k$ , non è necessariamente coincidente con il tempo  $t$ !

Occupazione limitata della memoria.

Considero solo quello che avviene all'interno di un trial.

Il passato che mi interessa è riassunto da  $Q_N$

**$Q_{N+1}$  rappresenta la media dei reward  $\{r(a_k)\}$ , tutti pesati  $1/N+1$**



## Formulazione generale di $Q_N$



$$Q_{N+1} = Q_N - \frac{Q_N}{N+1} + \frac{r_{N+1}}{N+1} = Q_N + \alpha [r_{N+1} - Q_N] \quad \alpha = 1/(N+1)$$

$$Q_{N+1} = Q_N - \frac{Q_N}{N+1} + \frac{r_{N+1}}{N+1} = Q_N + \alpha [r_{N+1} - Q_N]$$

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

$$\text{NewEstimate} = \text{OldEstimate} + \text{StepSize} [\text{Target} - \text{OldEstimate}]$$

$$\text{NewEstimate} = \text{OldEstimate} + \text{StepSize} * \text{Error}$$

$$\text{StepSize} = \alpha = 1/(N+1)$$

$\alpha$  pesa il bilanciamento tra "innovazione" e "tradizione"

### Osservazione:

Se  $\alpha = 0$ , conta solo la funzione valore (considera solo il passato).

Se  $\alpha = 1$ ,  $Q_{N+1}$  assume il valore di  $Q_N$  (dimentica tutto il passato).



## Esempio

$$Q_{N+1} = Q_N - \frac{Q_N}{N+1} + \frac{r_{N+1}}{N+1} = Q_N + \frac{1}{N+1}(r_{N+1} - Q_N)$$

$$\begin{array}{cccc} r_1 = 2 & r_2 = 3 & r_3 = 7 & r_4 = 2 \\ Q_1 = 2 & Q_2 = 2,5 & Q_3 = 4 & Q_4 = 3,5 \end{array}$$

$$Q_3 = Q_2 + \frac{1}{3}(r_3 - Q_2)$$

$$Q_3 = 2,5 + 1/3 (7 - 2,5) = 2,5 + 1,5 = 4,0$$

$$Q_4 = 4 + 1/4 (2 - 4) = 4 - 0,5 = 3,5$$



## Caso stazionario

$$Q_{N+1} = \frac{r_1 + r_2 + \dots + r_N + r_{N+1}}{N+1}$$

Il peso di ciascun campione è pari a  $1/(N+1)$

$$Q_{k+1} = \sum_{i=1}^{k+1} \frac{r_i}{N_{k+1}}$$

$$Q_{N+1} = Q_N - \frac{Q_N}{N+1} + \frac{r_{N+1}}{N+1}$$

Ogni nuovo campione viene pesato con  $1/(N+1)$

$$Q_{N+1} = Q_N + \frac{1}{N+1}(r_{N+1} - Q_N)$$

Peso decrescente (con  $1/(N+1)$ ) dei nuovi campioni

$$Q_{N+1} = Q_N + \alpha [r_{N+1} - Q_N] \quad \alpha = 1/(N+1)$$



# Riassunto



- Gli agenti ed il Reinforcement Learning
- Apprendimento nel condizionamento classico
- Implementazione ricorsiva
- **Caso non stazionario**

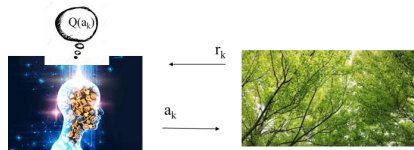


# Caso stazionario



$$Q_t(a_k) = \frac{r_1(a_k) + r_2(a_k) + r_3 + \dots + r_N(a_k) + r_{N+1}(a_k)}{(N+1)(a_k)}$$

$$= \sum_{i=1}^{N+1} \frac{r_i(a_k)}{N+1} = \sum_{i=1}^{N+1} w_i r_i(a_k)$$

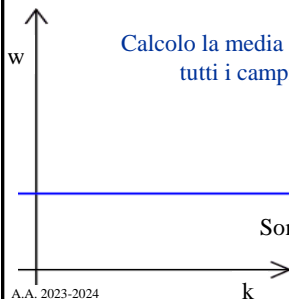


$$Q_{N+1} = Q_N + \frac{1}{N+1} (r_{N+1} - Q_N)$$

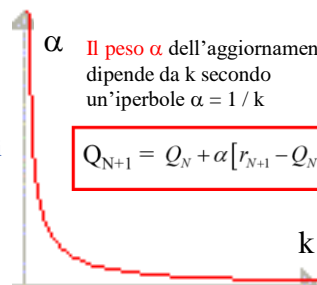
Calcolo la media in modo ricorsivo, in cui i pesi di tutti i campioni sono uguali a  $1/(N+1)$ .

$\alpha$  Il peso  $\alpha$  dell'aggiornamento dipende da k secondo un'iperbole  $\alpha = 1/k$

$$Q_{N+1} = Q_N + \alpha [r_{N+1} - Q_N]$$



Somma: peso tutti i campioni allo stesso modo:  $w_i = 1/(N+1)$





## Caso non stazionario



$$Q_{N+1} = Q_N + \alpha [r_{N+1} - Q_N]$$

Al tempo  $N+1$ , ottengo  $r_{N+1}$   
 $Q_0$  è il valore a cui è inizializzata  $Q$

$$Q_{N+1} = \alpha r_{N+1} + (1 - \alpha) Q_N$$

Suppongo  $\alpha = \text{cost} \rightarrow \alpha_{N+1} = \alpha \forall k \quad 0 \leq \alpha \leq 1$   
In precedenza la costante era  $\alpha_{N+1} = 1/(N+1)$

$$\begin{aligned} Q_N &= Q_{N-1} + \alpha [r_N - Q_{N-1}] = \\ &= \alpha r_N + (1 - \alpha) Q_{N-1} = \\ &= \alpha r_N + (1 - \alpha) [\alpha r_{N-1} + (1 - \alpha) Q_{N-2}] = \\ &= \alpha r_N + (1 - \alpha) \alpha r_{N-1} + (1 - \alpha)^2 Q_{N-2} = \end{aligned}$$

$$(1 - \alpha)^0 \alpha r_N + (1 - \alpha)^1 \alpha r_{N-1} + (1 - \alpha)^2 \alpha r_{N-2} + \dots + (1 - \alpha)^{N-1} \alpha r_1 + (1 - \alpha)^N Q_0$$



## Caso non stazionario



$$Q_N = \alpha r_N + (1 - \alpha) Q_{N-1}$$

Al passo  $N$ , ottengo  $r_N$   
 $Q_0$  è il valore a cui è inizializzata  $Q$

Suppongo  $\alpha = \text{cost} \rightarrow \alpha_k = \alpha \forall k \quad 0 \leq \alpha \leq 1$

$$\begin{aligned} \alpha r_N + (1 - \alpha) \alpha r_{N-1} + (1 - \alpha)^2 \alpha r_{N-2} + \dots + (1 - \alpha)^{N-1} \alpha r_1 + (1 - \alpha)^N Q_0 = \\ (1 - \alpha)^0 \alpha r_N + (1 - \alpha)^1 \alpha r_{N-1} + (1 - \alpha)^2 \alpha r_{N-2} + \dots + (1 - \alpha)^{N-1} \alpha r_1 + (1 - \alpha)^N Q_0 \end{aligned}$$

Per  $r_i \quad 1 \leq i \leq N$   
Per  $(1 - \alpha)^{N-i} \quad 1 \leq i \leq N$

$r_1 \rightarrow r_N$   
 $(1 - \alpha)^{N-1} \rightarrow (1 - \alpha)^0$

$$Q_{N+1} = \sum_{i=1}^N \alpha (1 - \alpha)^{N-i} r_i + (1 - \alpha)^N Q_0$$



## Osservazioni

$$Q_{N+1} = (1-\alpha)^N Q_0 + \sum_{i=1}^N \alpha(1-\alpha)^{N-i} r_i = (1-\alpha)^N Q_0 + \sum_{i=1}^N w_i r_i$$

$$w_i = \alpha(1-\alpha)^{N-i} \quad \alpha < 1$$

I reward **non** sono pesati tutti allo stesso modo: weighted average.

Il peso di ciascun campione decresce esponenzialmente a partire da  $i = N$  (tempo presente,  $w_N = \alpha$ ) fino a  $i = 1$  (tempo iniziale  $w_1 = \alpha(1-\alpha)^{(N-1)}$ ), secondo:

$$\alpha(1-\alpha)^{N-i} \quad \alpha < 1$$

Voglio calcolare il reward totale, *pesando i vecchi campioni sempre meno?*

**Media pesata, con peso decrescente nel tempo.**

E.g. weight decrease

$$\alpha = 0.8 * (1-0.8)^0 \text{ for } i=N,$$

$$\alpha = 0.8 * (1-0.8)^{N-1} \text{ for } i=1 - \text{for } N = 10 \quad \alpha = 0,0000004096$$

$$\alpha = 10 * (1-0.8)^N \text{ for } i=0 - \text{for } N = 10 \quad \alpha = 0,0000001024$$

A **Recency-weighted average.**

<http://borghese.di.unimi.it/>



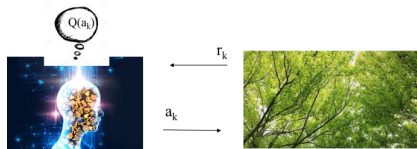
## Osservazioni

$$Q_{N+1} = Q_N + \alpha_{N+1} (r_{N+1} - Q_N)$$

L'agente ragiona sul trial attuale, k.

Dipende da solo da  $Q_k$  e  $r_k$

$Q_N$  rappresenta la media pesata dei primi N reward



$$Q_{N+1} = \sum_{i=1}^N \alpha(1-\alpha)^{N-i} r_i + (1-\alpha)^N Q_0$$

Peso: esponenziale decrescente

$$\alpha_{N+1} = \text{costante} = \alpha \quad \forall N$$

$$Q_{N+1} = \sum_{i=1}^{N+1} \frac{1}{N+1} r_i$$

Peso: costante

$$\alpha_{N+1} = 1/(N+1)$$



## Somma dei pesi dei reward è unitaria



$$Q_{N+1} = (1-\alpha)^N Q_0 + \sum_{i=1}^N \alpha(1-\alpha)^{N-i} r_i \quad \begin{array}{l} i=1 \rightarrow (1-\alpha)^{N-1} \\ i=N \rightarrow (1-\alpha)^0 \end{array}$$

Riscrivo considerando solamente i coefficienti.

Pongo  $i = N - i$ , la somma diventa da 0 a  $N-1$ :

$$\alpha \sum_{i=0}^{N-1} (1-\alpha)^i + (1-\alpha)^N =$$

Somma di una successione geometrica:

$$\sum_{i=0}^{N-1} r^i = \frac{1-r^N}{1-r}$$

$$\alpha \frac{1-(1-\alpha)^N}{1-(1-\alpha)} + (1-\alpha)^N = \frac{\alpha - \alpha(1-\alpha)^N + (1-\alpha)^N \square (1-\alpha)^N + \alpha(1-\alpha)^N}{1-1+\alpha} = 1 \quad \text{cvd}$$



## Condizioni iniziali



$$Q_{N+1} = (1-\alpha)^N Q_0 + \sum_{i=1}^N \alpha(1-\alpha)^{N-i} r_i$$

Metodi ad  $\alpha = 1/N_{k+1}$ ,  $Q_0$  non viene utilizzato se non al primo passo, viene poi sostituito da  $Q_1$ .

Metodi ad  $\alpha$  costante,  $Q_0$  conta sempre meno, ma la polarizzazione è permanente ( $Q_0 = 0$ ).

$Q_0$  può essere utilizzato per fornire della **conoscenza a-priori** o per favorire l'esplorazione (e.g. transfer learning).

Come posso gestire una situazione in cui la slot machine cambia improvvisamente la sua densità di probabilità di reward?



# Pseudo-codice per il calcolo di $Q_k$ .



```
##### 1) Definizione delle variabili:
N_scelte = m; eps_greedy = 0.1; // epsilon dipende dal grado di greedy che voglio dare all'agente
## Variabili dell'agente
A = {1, 2, ..., m}; // Azioni possibili
Q = {Q1, Q2, ..., Qm} = 0; // Value function per ogni azione
N_azioni = {1, 2, ..., m}; // Numero di volte in cui è scelta l'azione j (e collezionato il reward associato).
## Variabili dell'ambiente. Date nella simulazione, misurate nell'ambiente nella realtà
// Inizializzo i parametri della distribuzione (stazionaria) dei reward per ogni azione
meanReward = [mean_1, mean_2, ..., mean_m]; stdReward = [std_1, std_2, ..., std_m];

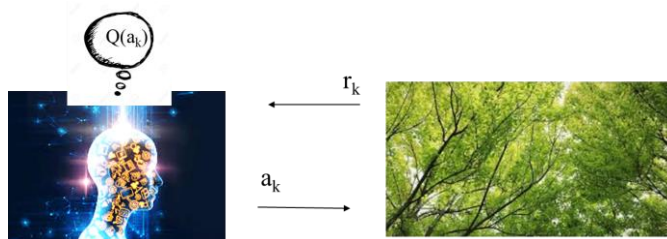
##### 2) Ciclo di funzionamento
while (true)
{
  eps = rand_unif([0 1]); // Per politica epsilon-greedy
  // Exploitation
  [a_attuale Q_attuale] = SearchMax(Q); // Cerca l'azione ottima secondo Q
  // Exploration: se eps < eps_greedy, allora exploration
  if (eps < eps_greedy)
  // Devo trovare un'azione diversa da a_attuale -> a_ref
  {
    trovato = false; a_ref = a_attuale;
    while (trovato == false)
    {
      a_attuale = rand_unif(A);
      if (a_attuale != a_ref)
      {
        trovato = true; Q_attuale = Q(a_attuale);
      }
    }
  }
  // Eseguo l'azione a_attuale e misuro il reward ottenuto dalla slot machine
  r_attuale = rand_Gauss([meanReward(a_attuale), stdReward(a_attuale)]);
  // Update i dati per l'azione a_attuale: il numero di azioni e la value function Q
  N_azioni(a_attuale)++;
  Q(a_attuale) = Q(a_attuale) + 1/[N_azioni(a_attuale)] * (r_attuale - Q(a_attuale));
}
}
```



# RL su trial



- L'agente deve scegliere un'azione,  $a$  (tra diverse azioni possibili discrete o continue).
- L'agente misura la ricompensa,  $r$  (reward) associate alla sua azione.
- Apprendimento all'interno di un singolo trial.
- L'agente costruisce una stima del valore di ciascuna azione,  $a_k$ ,  $Q(a_k)$ .
- Prende una decisione in base a  $Q(a_k)$ , reward secondario, a lungo termine.







# Riassunto



- Gli agenti ed il Reinforcement Learning
- Apprendimento nel condizionamento classico
- Implementazione ricorsiva
- Caso non stazionario