

Control prin învățare

Master ICAF, An 1 Sem 2

Lucian Bușoniu



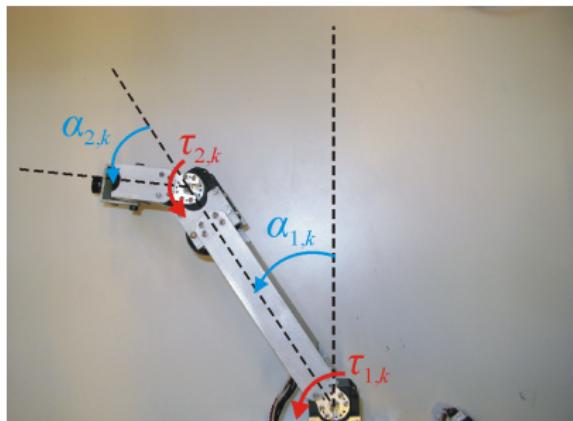
Partea V

Învățarea prin recompensă online cu
aproximare



Recap: Nevoia de aproximare

- În aplicații reale de control, x, u **continue!**



- Reprezentarea prin tabel **imposibilă**
- Aproximarea** funcțiilor de interes
 $Q(x, u)$, $V(x)$, $h(x)$ necesară

Recap: Partea 4 – Algoritmi offline

pornind de la:

- model f, ρ
- sau date $(x_s, u_s, r_s, x'_s), s = 1, \dots, n_s$

- ➊ găsește soluție aproximată $\hat{Q}(x, u), \hat{h}(x)$, etc.
- ➋ controlează sistemul folosind soluția găsită

Algoritmi exemplificați:

- iterarea fuzzy Q
- iterarea Q bazată pe date



Partea V În plan

- Problema de învățare prin recompensă
- Soluția optimală
- Programarea dinamică exactă
- Învățarea prin recompensă exactă
- Tehnici de aproximare
- Programarea dinamică cu aproximare (var. continue)
- **Învățarea prin recompensă cu aproximare (var. continue)**

Există mulți algoritmi, doar 2 sunt selectați pentru discuție



Conținut partea 5

- 1 Învățarea Q și SARSA cu aproximare
- 2 Accelerarea RL cu aproximare
- 3 Perspective

1 Învățarea Q și SARSA cu aproximare

- Învățarea Q approximată
- SARSA approximată

2 Accelerarea RL cu aproximare

3 Perspective

Reamintim: Învățarea Q

Învățarea Q cu ε -greedy

for fiecare traiectorie **do**

 initializează x_0

repeat la fiecare pas k

$$u_k = \begin{cases} \arg \max_u Q(x_k, u) & \text{cu prob. } (1 - \varepsilon_k) \\ \text{aleatoare} & \text{cu prob. } \varepsilon_k \end{cases}$$

 aplică u_k , măsoară x_{k+1} , primește r_{k+1}

$$Q(x_k, u_k) \leftarrow Q(x_k, u_k) + \alpha_k \cdot$$

$$[r_{k+1} + \gamma \max_{u'} Q(x_{k+1}, u') - Q(x_k, u_k)]$$

until traiectoria terminată

end for

Diferența temporală: $[r_{k+1} + \gamma \max_{u'} Q(x_{k+1}, u') - Q(x_k, u_k)]$



Învățarea Q aproximată

- Învățarea Q **scade diferența temporală**:

$$Q(x_k, u_k) \leftarrow Q(x_k, u_k) + \alpha_k [r_{k+1} + \gamma \max_{u'} Q(x_{k+1}, u') - Q(x_k, u_k)]$$

- $r_{k+1} + \gamma \max_{u'} Q(x_{k+1}, u')$ înlocuiește **idealul** $Q^*(x_k, u_k)$
[Vezi și Bellman: $Q^*(x, u) = \rho(x, u) + \gamma \max_{u'} Q^*(x', u')$]
- ⇒ Ideal, scade eroarea $[Q^*(x_k, u_k) - Q(x_k, u_k)]$

Învățarea Q aproximată (continuare)

Aproximare: folosim $\hat{Q}(x, u; \theta)$, actualizăm **parametri**

- **Gradient** pe eroarea $[Q^*(x_k, u_k) - \hat{Q}(x_k, u_k; \theta)]$:

$$\begin{aligned}\theta_{k+1} &= \theta_k - \frac{1}{2} \alpha_k \frac{\partial}{\partial \theta} \left[Q^*(x_k, u_k) - \hat{Q}(x_k, u_k; \theta_k) \right]^2 \\ &= \theta_k + \alpha_k \frac{\partial}{\partial \theta} \hat{Q}(x_k, u_k; \theta_k) \cdot \left[Q^*(x_k, u_k) - \hat{Q}(x_k, u_k; \theta_k) \right]\end{aligned}$$

- Folosește **estimare** pentru $Q^*(x_k, u_k)$:

$$\begin{aligned}\theta_{k+1} &= \theta_k + \alpha_k \frac{\partial}{\partial \theta} \hat{Q}(x_k, u_k; \theta_k) \cdot \\ &\quad \left[r_{k+1} + \gamma \max_{u'} \hat{Q}(x_{k+1}, u'; \theta_k) - \hat{Q}(x_k, u_k; \theta_k) \right]\end{aligned}$$

(diferență temporală aproximată)

Învățarea Q aproximată: algoritm

Învățarea Q aproximată cu explorare ε -greedy

for fiecare traiectorie **do**

 inițializează x_0

repeat la fiecare pas k

$$u_k = \begin{cases} \arg \max_u \hat{Q}(x_k, u; \theta_k) & \text{cu prob. } (1 - \varepsilon_k) \\ \text{aleatoare} & \text{cu prob. } \varepsilon_k \end{cases}$$

 aplică u_k , măsoară x_{k+1} , primește r_{k+1}

$$\theta_{k+1} = \theta_k + \alpha_k \frac{\partial}{\partial \theta} \hat{Q}(x_k, u_k; \theta_k) \cdot$$

$$\left[r_{k+1} + \gamma \max_{u'} \hat{Q}(x_{k+1}, u'; \theta_k) - \hat{Q}(x_k, u_k; \theta_k) \right]$$

until traiectoria terminată

end for

Desigur, **explorarea** necesară și în cazul approximat



Reamintim: Maximizare

Soluția 1:

- Legea de control nu este reprezentată explicit
- Acțiuni greedy calculate la cerere din \hat{Q}

Soluția 2:

- Legea de control aproximată explicit

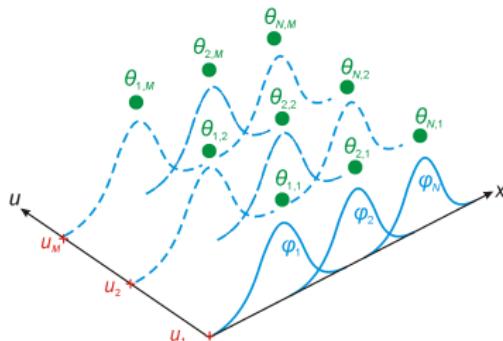
Maximizare în Învățarea Q aproximată

- Acțiuni greedy calculate la cerere din \hat{Q} :

$$\dots \max_u \hat{Q}(x, u; \theta) \dots$$

⇒ Soluția 1: Legea de control reprezentată implicit

- Aproximatorul funcției Q trebuie să garanteze
soluție eficientă pentru max
- Ex. acțiuni discrete & funcții de bază în x



Învățarea Q aprox.: demo mers robotic (E. Schuitema)

Aproximator: **tile coding**



1 Învățarea Q și SARSA cu aproximare

- Învățarea Q aproximată
- SARSA aproximată

2 Accelerarea RL cu aproximare

3 Perspective

SARSA aproximată

Reamintim SARSA clasică:

$$Q(x_k, u_k) \leftarrow Q(x_k, u_k) + \alpha_k [r_{k+1} + \gamma Q(x_{k+1}, u_{k+1}) - Q(x_k, u_k)]$$

Aproximare: similar cu învățarea Q

- actualizăm parametri
- bazat pe **gradientul** funcției Q
- și diferența temporală aproximată

$$\theta_{k+1} = \theta_k + \alpha_k \frac{\partial}{\partial \theta} \hat{Q}(x_k, u_k; \theta_k) \cdot$$

$$\left[r_{k+1} + \gamma \hat{Q}(x_{k+1}, u_{k+1}; \theta_k) - \hat{Q}(x_k, u_k; \theta_k) \right]$$



SARSA

SARSA aproximată

for fiecare traекторie **do**

 initializează x_0

 alege u_0 (ex. ε -greedy din $Q(x_0, \cdot; \theta_0)$)

repeat la fiecare pas k

 aplică u_k , măsoară x_{k+1} , primește r_{k+1}

 alege u_{k+1} (ex. ε -greedy din $Q(x_{k+1}, \cdot; \theta_k)$)

$$\theta_{k+1} = \theta_k + \alpha_k \frac{\partial}{\partial \theta} \hat{Q}(x_k, u_k; \theta_k) \cdot$$

$$\left[r_{k+1} + \gamma \hat{Q}(x_{k+1}, u_{k+1}; \theta_k) - \hat{Q}(x_k, u_k; \theta_k) \right]$$

until traectoria terminată

end for

Discuție învățarea Q, SARSA

Convergență

- Convergență garantată pentru **variante modificate**
- Complexitate scăzută
- Explorarea **crucială** pentru toate metodele
- Rata de învățare α delicată



1 Învățarea Q și SARSA cu aproximare

2 Accelerarea RL cu aproximare

- Urme de eligibilitate
- Reluarea experienței

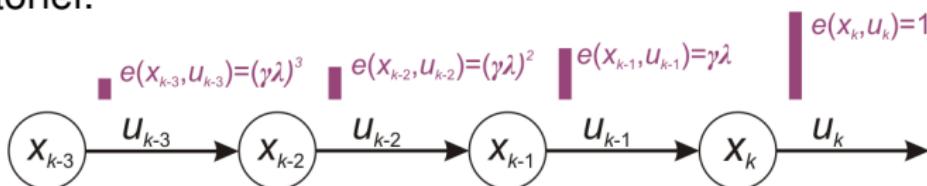
3 Perspective

Motivare

- Dezavantaj metode TD approximate:
ca și în cazul discret, învață încet
 - ⇒ Timp, uzură crescute, profituri scăzute
- **Accelerarea învățării** este necesară

Urme de eligibilitate

- Reamintim cazul discret – urmă $e(x, u)$ de-a lungul trajectoriei:



$$Q(x, u) \leftarrow Q(x, u) + \alpha_k \cdot e(x, u) \cdot [r_{k+1} + \gamma \max_{u'} Q(x_{k+1}, u') - Q(x_k, u_k)] \quad \forall x, u$$

- Când x, u continue, $e(x, u)$ nu se poate reprezenta direct

Urme de eligibilitate în cazul aproximat

- Idee: în actualizarea cu gradient, de ex. învățarea Q:

$$\theta_{k+1} = \theta_k + \alpha_k \frac{\partial}{\partial \theta} \hat{Q}(x_k, u_k; \theta_k) \cdot [r_{k+1} + \gamma \max_{u'} \hat{Q}(x_{k+1}, u'; \theta_k) - \hat{Q}(x_k, u_k; \theta_k)]$$

- ...trăm gradientul $\frac{\partial}{\partial \theta_i} \hat{Q}(x_k, u_k; \theta_k)$ ca pe o **contribuție** a parametrului i la actualizarea curentă
- Luăm în considerare **contribuția cumulativă** (scăzând cu $\gamma \lambda$) până la pasul curent:

$$\theta_{k+1} = \theta_k + \alpha_k e_{k+1} \cdot [r_{k+1} + \gamma \max_{u'} \hat{Q}(x_{k+1}, u'; \theta_k) - \hat{Q}(x_k, u_k; \theta_k)]$$

$$e_{k+1} = \sum_{\ell=0}^k (\gamma \lambda)^{k-\ell} \frac{\partial}{\partial \theta} \hat{Q}(x_\ell, u_\ell; \theta_\ell)$$

Urme de eligibilitate în învățarea Q aproximată

Implementare iterativă în învățarea Q:

Învățarea Q(λ) aproximată

for fiecare traiectorie **do**

 initializează x_0 , $e_0 = [0, \dots, 0]^\top$

repeat la fiecare pas k

$u_k = \begin{cases} \arg \max_u \hat{Q}(x_k, u; \theta_k) & \text{cu prob. } (1 - \varepsilon_k) \\ \text{aleatoare} & \text{cu prob. } \varepsilon_k \end{cases}$

 aplică u_k , măsoară x_{k+1} , primește r_{k+1}

 actualizează $e_{k+1} = (\gamma \lambda) e_k + \frac{\partial}{\partial \theta} \hat{Q}(x_k, u_k; \theta_k)$

$\theta_{k+1} = \theta_k + \alpha_k e_{k+1}$.

$\left[r_{k+1} + \gamma \max_{u'} \hat{Q}(x_{k+1}, u'; \theta_k) - \hat{Q}(x_k, u_k; \theta_k) \right]$

until traiectoria terminată

end for



Urme de eligibilitate în alți algoritmi

- Urmele de eligibilitate se pot adapta simplu la SARSA și Învățarea criticului din actor-critic
- Ideea se extinde și la LSPI, dar mai complicat

1 Învățarea Q și SARSA cu aproximare

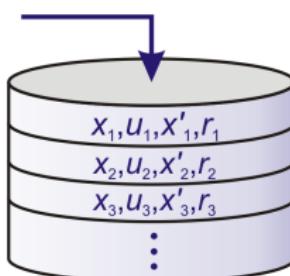
2 Accelerarea RL cu aproximare

- Urme de eligibilitate
- Reluarea experienței

3 Perspective

Reluarea experienței

- Stochează tranzitiiile $(x_k, u_k, x_{k+1}, r_{k+1})$ într-o bază de date



- La fiecare pas, reia n tranzitii din baza de date pe lângă actualizările normale

SARSA aproximată cu reluarea experienței

SARSA aproximată cu reluarea experienței

for fiecare traiectorie **do**

 inițializează x_0

 alege u_0

repeat la fiecare pas k

 aplică u_k , măsoară x_{k+1} , primește r_{k+1}

 alege u_{k+1}

$$\theta_{k+1} = \theta_k + \alpha_k \frac{\partial}{\partial \theta} \hat{Q}(x_k, u_k; \theta_k) \cdot$$

$$\left[r_{k+1} + \gamma \hat{Q}(x_{k+1}, u_{k+1}; \theta_k) - \hat{Q}(x_k, u_k; \theta_k) \right]$$

 adaugă $(x_k, u_k, x_{k+1}, r_{k+1})$ la baza de date

 ReiaExperiența

until traiectoria terminată

end for



Procedura ReiaExperiența

ReiaExperiența

loop de N ori

preia o tranziție (x, u, x', r) din baza de date

$$\theta_{k+1} = \theta_k + \alpha_k \frac{\partial}{\partial \theta} \hat{Q}(x_k, u_k; \theta_k) \cdot$$

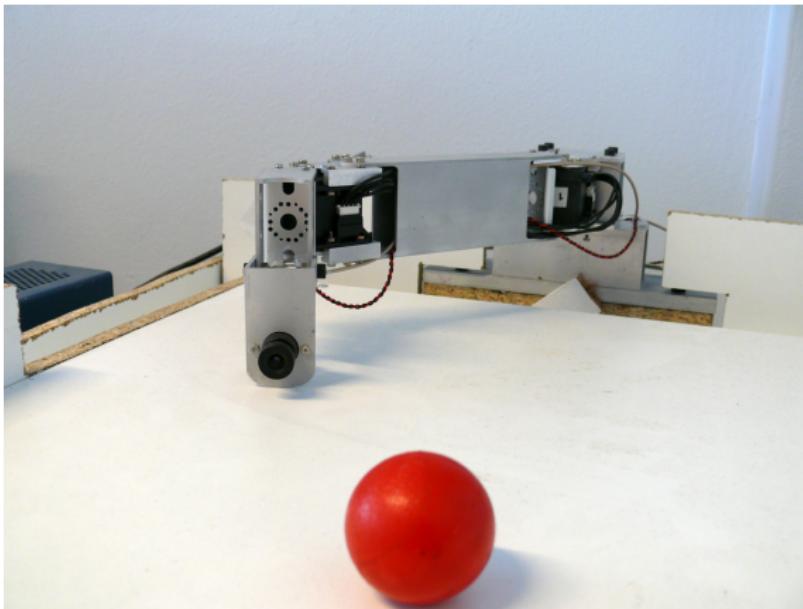
$$\left[r_{k+1} + \gamma \hat{Q}(x_{k+1}, u_{k+1}; \theta_k) - \hat{Q}(x_k, u_k; \theta_k) \right]$$

end loop

Pendul: RL cu reluarea exp., demo (Sander Adam)



Robot portar: RL cu reluarea exp., demo (S. Adam)



- 1 Învățarea Q și SARSA cu aproximare
- 2 Accelerarea RL cu aproximare
- 3 Perspective

Conexiuni: Control optimal

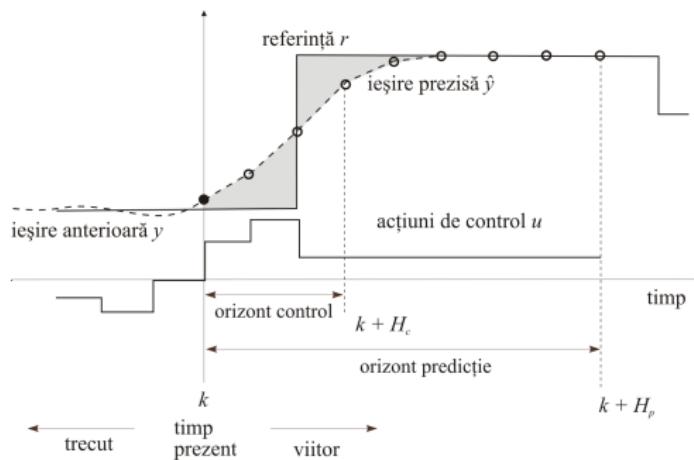
- Controlează un sistem minimizând costul J
- Bazat pe model
- Posibil în timp continuu, orizont finit

RL, DP **sunt control optimal!**

- Controlează un sistem maximizând returnul $R^h(x)$
- În timp discret, orizont infinit
- RL **fără model**, bazat pe date/interacțiune

Conexiuni: Control predictiv

- Bazat pe model, clasic liniar
- Principiu de bază: **receding horizon**



Controlul predictiv este și el o variantă de control optimal, bazată pe model

Probleme deschise

RL & DP **în curs de dezvoltare**

Probleme deschise:

- Garanții de siguranță și stabilitate
- Stări și acțiuni cu dimensionalitate mare
- Stări care nu pot fi măsurate
- Strategii de explorare
- Sisteme multiagent



Învățarea prin recompensă cu rețelele adânci – Deep RL

O modalitate de a trata stările cu dimensionalitate mare *când acestea sunt imagini* (sau pot fi asimilate unor imagini)

Deep Q-network, DQN

- Funcția Q reprezentată via o rețea neuronală $Q(x_{k+1}, \cdot; \theta_k)$
- Rețea neuronală “deep” cu multe nivele, cu structuri și funcții de activare specifice
- Rețeaua antrenată pentru a minimiza diferența temporală, similar cu algoritmul standard
- Antrenare pe mini-batch-uri de tranzitii, similar cu iterația Q bazată pe date \Rightarrow algoritmul combină RL online și offline

(DeepMind, [Human-level control through deep reinforcement learning](#), Nature 2015)

