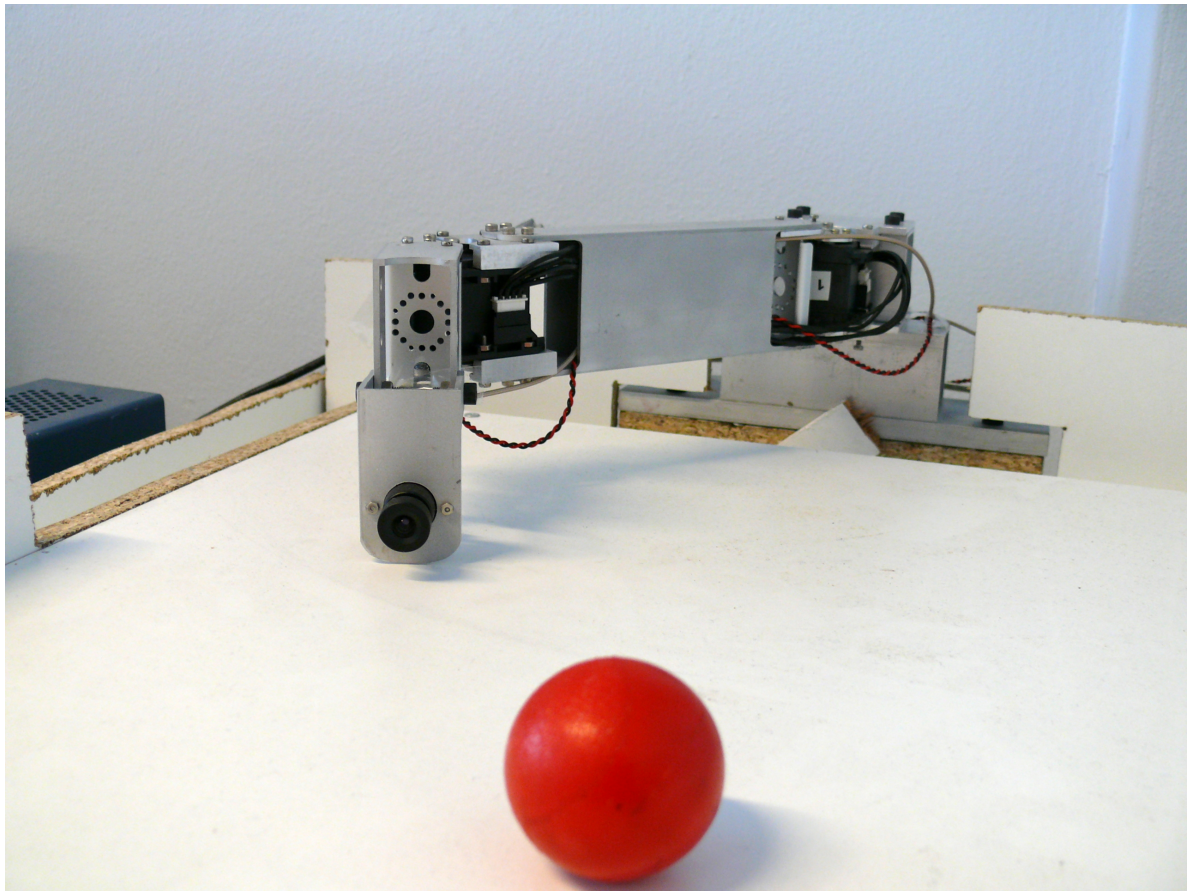
# Control prin învăţare

## Master ICAF, An 1 Sem 2

Lucian Buşoniu

# RL pentru un robot-portar (TUDelft)

Învaţă să prindă mingea folosind camera video
Control la nivel fizic



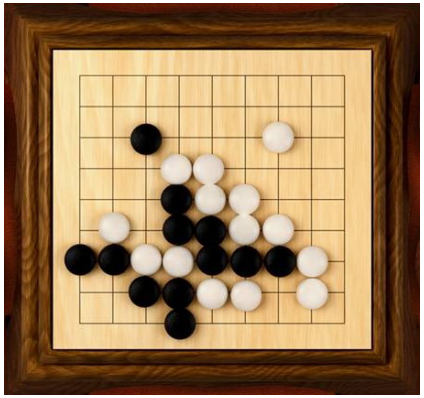
# Planificarea pentru un robot domestic (UTCluj)

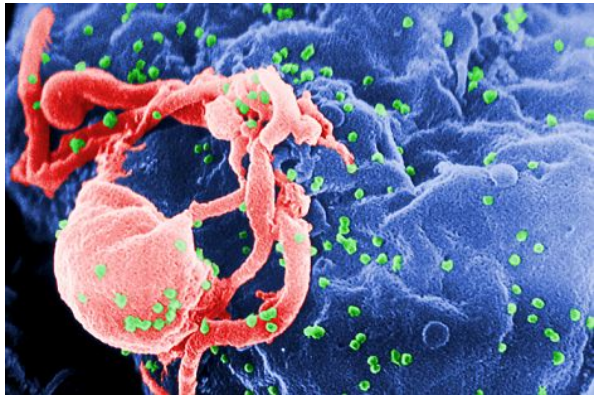
Robotul domestic se asigură că întrerupătoarele sunt oprite
Control la nivel înalt (acţiuni realizate de alte controlere la nivelul fizic)

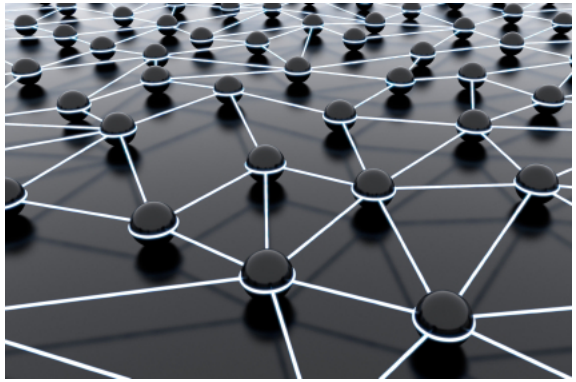
# Atari

# Alte aplicaţii

Inteligenţa artificială, medicină, sisteme multiagent, economie etc.







# Conţinut

- **Curs 1: Problema de învăţare prin recompensă**
- Soluţia optimală
- Programarea dinamică exactă
- Învăţarea prin recompensă exactă
- Tehnici de aproximare
- Programarea dinamică cu aproximare
- Învăţarea prin recompensă cu aproximare

# Partea I

## Problema de învăţare prin recompensă

# Conţinut curs 1

# De ce învăţare?

**Învăţarea** identifică soluţii care:

1. nu pot fi proiectate în avans
   - problema prea complexă
     (ex. controlul sistemelor puternic neliniare)
   - problema incomplet cunoscută
     (ex. explorarea robotizată a spaţiului cosmic)
2. se îmbunătăţesc permanent
3. se adaptează unui mediu variabil în timp

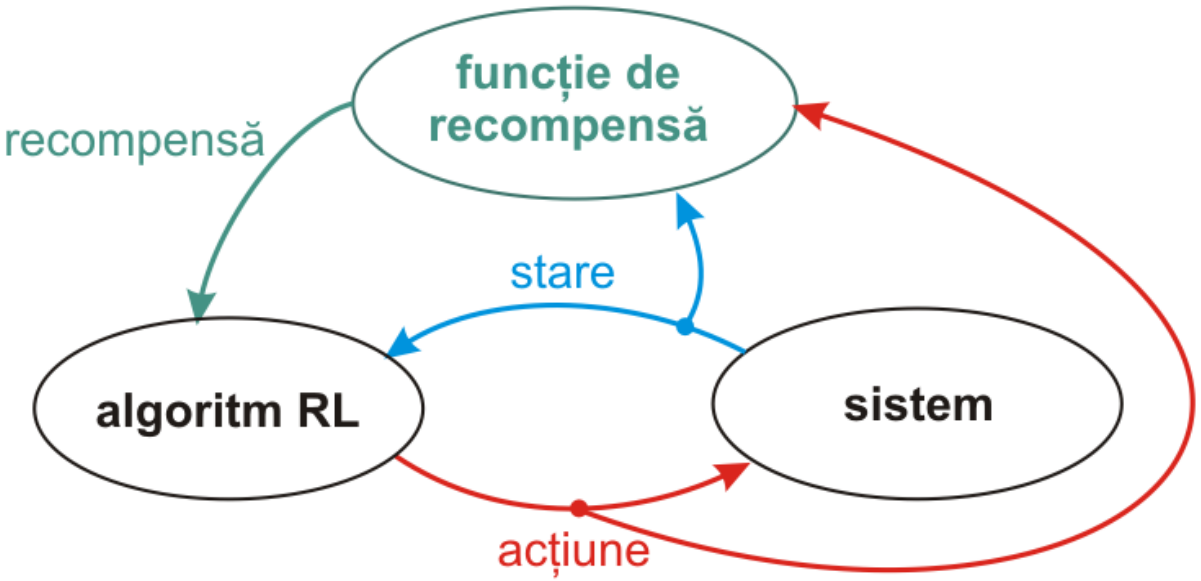
Esenţial pentru orice sistem **inteligent**

# Metode bazate pe model

Cursul va pune accent şi pe **metodele bazate pe model**:

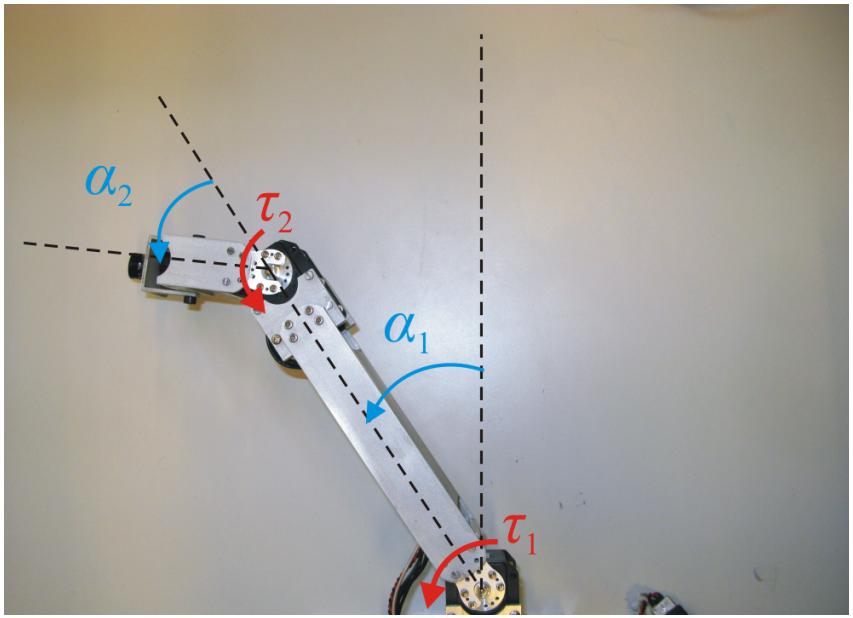
- Stau la baza învăţării prin recompensă
  (ex. programarea dinamică)
- Sunt inspirate din învăţarea prin recompensă
  (ex. planificarea)
- Sunt utile separat de învăţare, când avem modelul,
  fiindcă tratează probleme complexe (ex. neliniare)

# Principiul RL



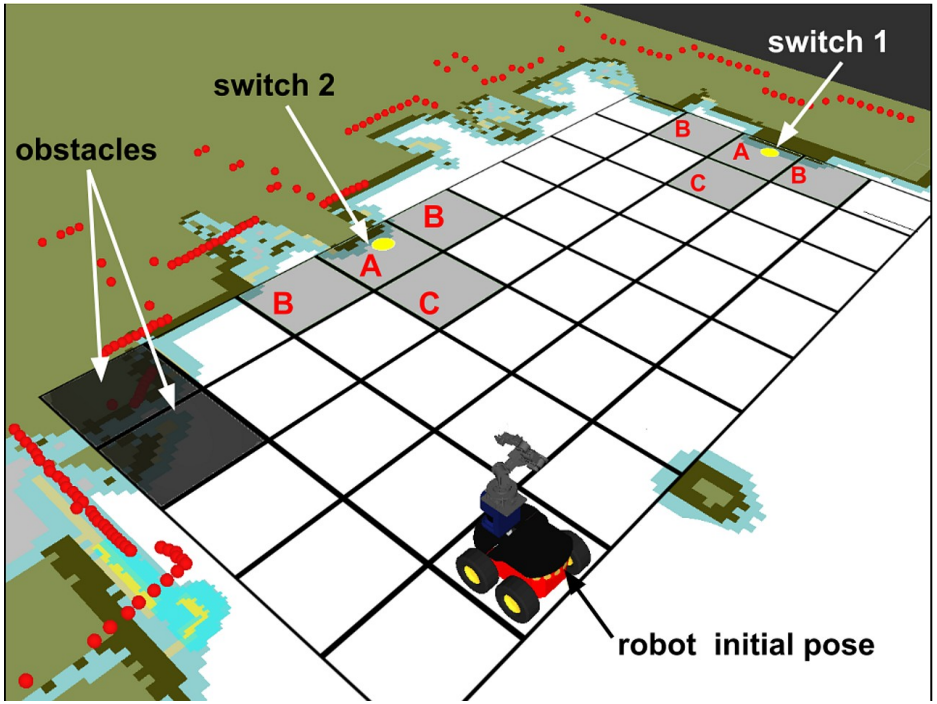

- Interacţiune cu un sistem prin stări şi acţiuni
- Feedback despre performanţă în forma recompensei
- Inspirată din învăţarea umană şi animală

# Exemplu: braţ robotic




- Stări: unghiuri, viteze unghiulare
- Acţiuni: voltaj (sau cuplu) motoare
- Recompense: ex., pentru atingerea unei configuraţii, recompensele cresc cu scăderea distanţei până la configuraţie

# Exemplu: robot domestic




- Stări: coordonate pe grid, stările întrerupătoarelor
- Acţiuni: mişcări NSEV, comutare întrerupător
- Recompense: când un întrerupător pornit este oprit (şi penalizare când unul oprit este pornit!)

Exemplu de **abstractizare**: problema rezolvată la nivel înalt, acţiunile efectuate de către controlere la nivelul fizic

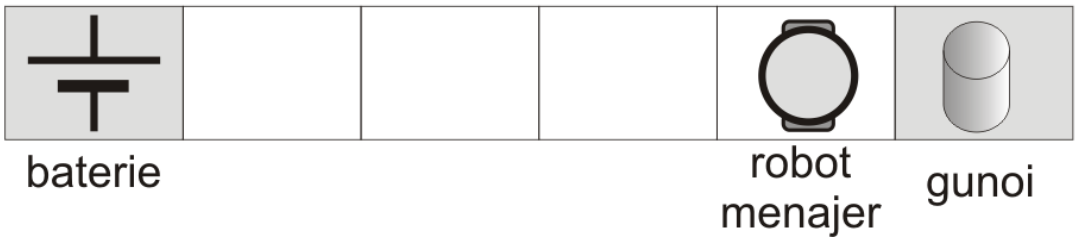
# Exact vs. aproximat; Determinist vs. stohastic

- **Cursurile 1–4: metode exacte** – stări şi acţiuni discrete cu un număr mic de valori
  - pas intermediar, necesar pentru a înţelege problema mai dificilă cu aproximare
  - util şi separat, dacă problema poate fi abstractizată într-una discretă la nivel înalt
- Cursurile 5–8: metode cu aproximare – stări şi acţiuni discrete cu multe valori, sau continue

Sistemul se poate comporta:

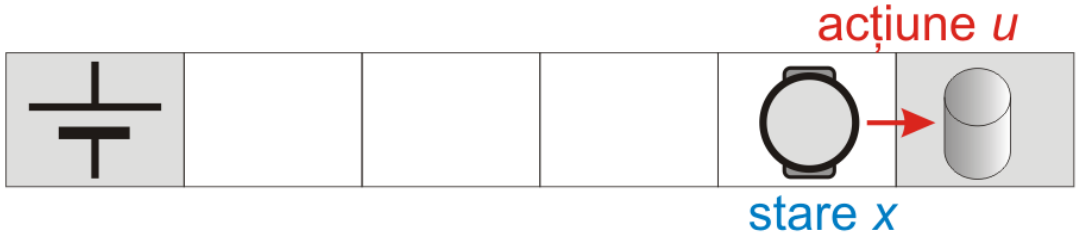
- Determinist – răspunde la aceeaşi acţiune în acelaşi fel
- Stohastic

# Un exemplu simplu: robot menajer

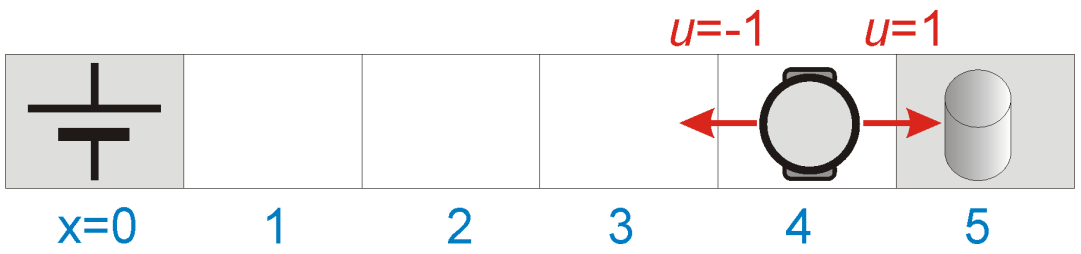



- Robot menajer într-o lume 1-D
- Colectează gunoi (recompensă $+5$) sau baterie (recompensă $+1$)
- După ce un obiect a fost colectat, episodul se termină

# Robot menajer: Stare & acţiune

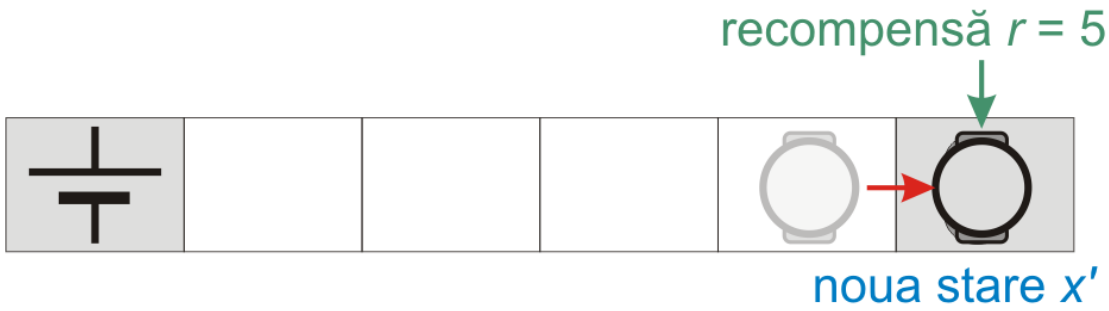



- Robotul se află într-o stare $x$ (căsuţă)
- şi aplică o acţiune $u$ (ex. înaintează la dreapta)



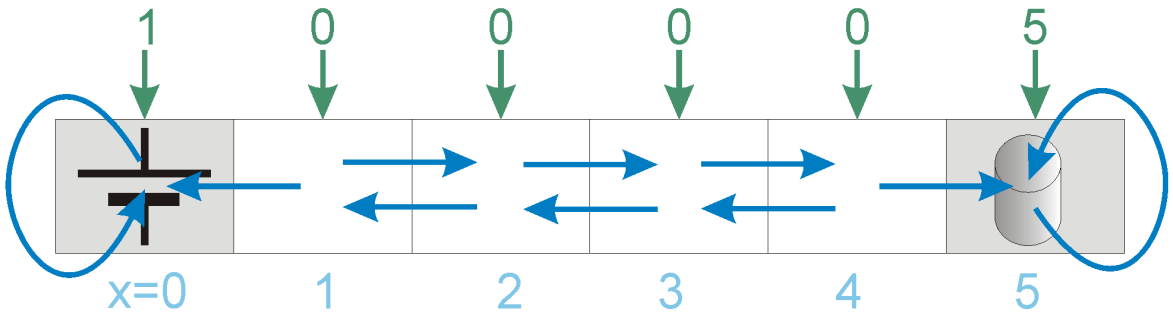

- Spaţiul stărilor $X = \{0, 1, 2, 3, 4, 5\}$
- Spaţiul acţiunilor $U = \{-1, 1\} = \{\text{stânga, dreapta}\}$

# Robot menajer: Tranziţie şi recompensă




- Robotul atinge o nouă stare $x'$
- şi primeşte o recompensă $r$ = calitatea tranziţiei (aici, $+5$ pentru colectarea gunoiului)

# Robot menajer: Funcţii de tranziţie & recompensă




- Funcţia de tranziţie (comportamentul sistemului):

$$x' = f(x,u) = \begin{cases} x & \text{dacă } x \text{ terminal (0 sau 5)} \\ x+u & \text{altfel} \end{cases}$$

- Funcţia de recompensă (performanţa imediată):

$$r = \rho(x,u) = \begin{cases} 1 & \text{dacă } x = 1 \text{ şi } u = -1 \text{ (baterie)} \\ 5 & \text{dacă } x = 4 \text{ şi } u = 1 \text{ (gunoi)} \\ 0 & \text{altfel} \end{cases}$$

- **De notat:** Stările terminale nu pot fi părăsite şi nu sunt recompensate!

# O notă asupra recompensei

- De fapt, recompensa depinde de **tranziţie** $r = \tilde{\rho}(x, u, x')$
- Dar $x'$ determinat de $(x, u)$ şi poate fi înlocuit în formulă:

$$\tilde{\rho}(x, u, x') = \tilde{\rho}(x, u, f(x, u)) = \rho(x, u)$$

$$r = \rho(x, u) = \begin{cases} 1 & \text{dacă } x = 1 \text{ şi } u = -1 \text{ (baterie)} \\ 5 & \text{dacă } x = 4 \text{ şi } u = 1 \text{ (gunoi)} \\ 0 & \text{altfel} \end{cases}$$

# Proces de decizie Markov, cazul determinist

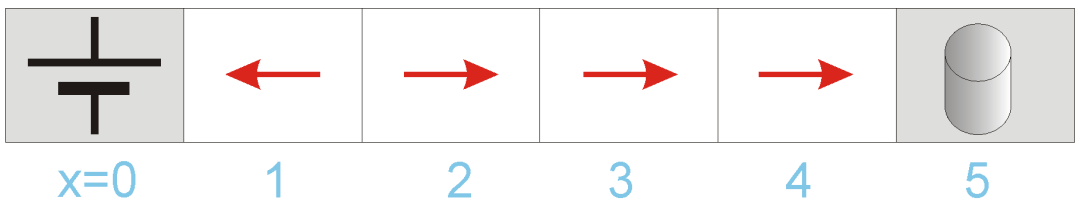
## Proces de decizie Markov

Format din:

1. Spaţiul stărilor $X$
2. Spaţiul acţiunilor $U$
3. Funcţia de tranziţie $x' = f(x, u), \quad f : X \times U \to X$
4. Funcţia de recompensă $r = \rho(x, u), \quad \rho : X \times U \to \mathbb{R}$

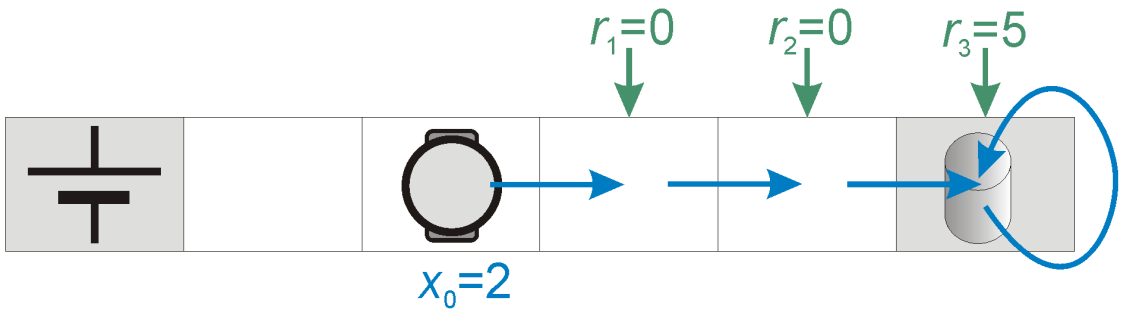
# Lege de control

- Legea de control $h$: funcţie din $x$ în $u$ (reacţie de la stare)




Exemplu: $h(0) = *$ (stare terminală, acţiunea este irelevantă), $h(1) = -1$, $h(2) = 1$, $h(3) = 1$, $h(4) = 1$, $h(5) = *$

# Robot menajer: Return




Luăm $h$ care merge întotdeauna la dreapta

$$R^h(2) = \gamma^0 r_1 + \gamma^1 r_2 + \gamma^2 r_3 + \gamma^3 0 + \gamma^4 0 + \dots$$
$$= \gamma^2 \cdot 5$$

Fiindcă $x_3$ terminală, toate recompensele ulterioare sunt 0

# Obiectiv

Găseşte $h$ care maximizează returnul:

$$R^h(x_0) = \sum_{k=0}^{\infty} \gamma^k r_{k+1} = \sum_{k=0}^{\infty} \gamma^k \rho(x_k, h(x_k))$$

din orice $x_0$

Factor de discount $\gamma \in [0, 1)$:

- induce un “pseudo-orizont” pentru optimizare
- mărgineşte suma infinită
- reprezintă incertitudinea crescândă despre viitor
- ajută convergenţa algoritmilor

De notat: Există şi alte tipuri de return!

# Alegerea factorului de discount

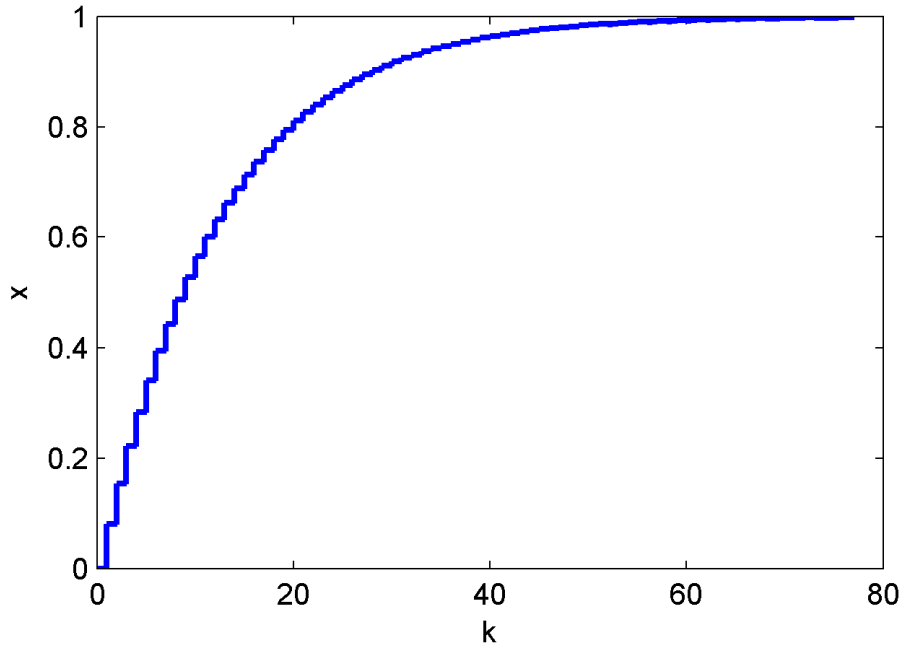
Pentru a alege $\gamma$, **compromis** între:

1. Calitatea pe termen lung a soluţiei ($\gamma$ mare)
2. "Simplitatea" problemei ($\gamma$ mic)

În practică, $\gamma$ suficient de mare pentru a nu ignora recompense importante de-a lungul traiectoriilor sistemului

# Exemplu: Alegerea $\gamma$ pentru un sistem simplu

Răspunsul unui sistem liniar de ordinul 1:




Valoarea $\gamma$ pentru ca recompensele la intrarea în regim staţionar să fie vizibile din starea iniţială?

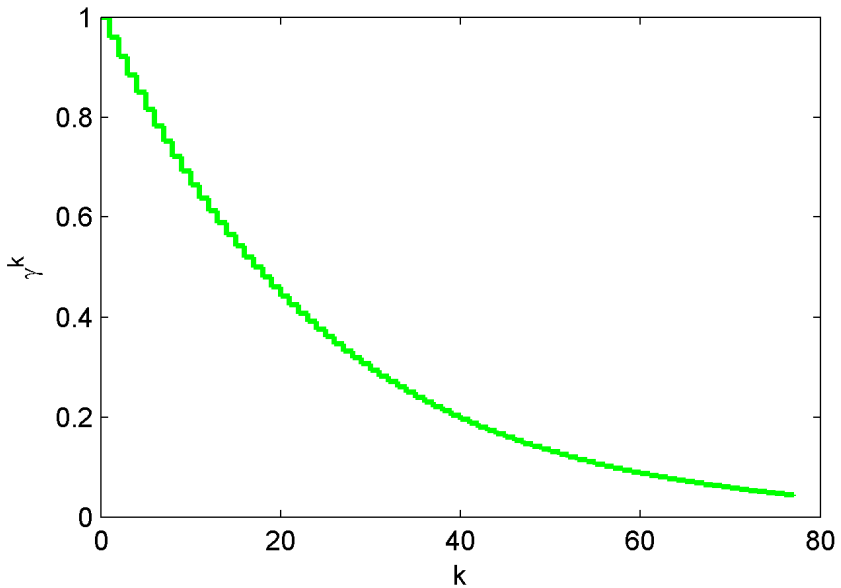
# Soluţie: Alegerea $\gamma$ pentru un sistem simplu

Pentru $k \approx 60$, $\gamma^k$ să nu fie prea mic, de ex.

$$\gamma^{60} \geq 0.05$$

$$\gamma \geq 0.05^{1/60} \approx 0.9513$$

$\gamma^k$ pentru $\gamma = 0.96$:



# Variabile aleatoare discrete

- O variabilă discretă $x$ poate lua $n$ valori, în setul $X = \{x_1, x_2, \ldots, x_n\}$.
- Fiecare valoare este asociată cu o probabilitate $p(x_1), p(x_2), \ldots, p(x_n)$, unde $p(x_i) \in [0, 1]$, $\sum_i p(x_i) = 1$. Funcţia $p : X \to [0, 1]$ se numeşte **funcţia de masă de probabilitate** (probability mass function, PMF).

Exemplu: Valoarea unui zar este o variabilă aleatoare discretă, cu $n = 6$ valori posibile, $x_1 = 1, \ldots, x_6 = 6$. Pentru un zar corect, $p(x_i) = \frac{1}{6}, \forall i = 1, \ldots, 6$

De notat: $n$ poate să crească la infinit; descrierea matematică rămâne validă

# Valoarea aşteptată (expectanţa)

- Media valorilor, ponderată de probabilităţi; valoarea "aşteptată" a priori, dată fiind distribuţia de probabilitate:

$$\mathrm{E}\{x\} = \sum_{x \in X} p(x)x$$

Exemplu: Pentru un zar corect, expectanţa este

$$\mathrm{E}\{x\} = \frac{1}{6}1 + \frac{1}{6}2 + \ldots + \frac{1}{6}6 = 7/2$$

- O **funcţie** cu o variabilă aleatoare ca argument, $g : X \to \mathbb{R}$ este la rândul ei o variabilă aleatoare, cu expectanţa:

$$\mathrm{E}\{g(x)\} = \sum_{x \in X} p(x)g(x)$$

Exemplu: Dacă feţele 1-4 câştigă 1\$, iar feţele 5-6, 10\$,

$$\mathrm{E}\{x\} = \frac{1}{6}1 + \frac{1}{6}1 + \frac{1}{6}1 + \frac{1}{6}1 + \frac{1}{6}10 + \frac{1}{6}10 = 4\$$$

# Independenţă

Variabilele aleatoare $x$, $y$ sunt independente dacă probabilitatea vectorului $z = (x, y)$ este $p_z(z) = p_x(x) \cdot p_y(y)$, unde $p_z, p_x, p_y$ sunt PMFurile celor trei variabile (conceptul se extinde la oricâte variabile)

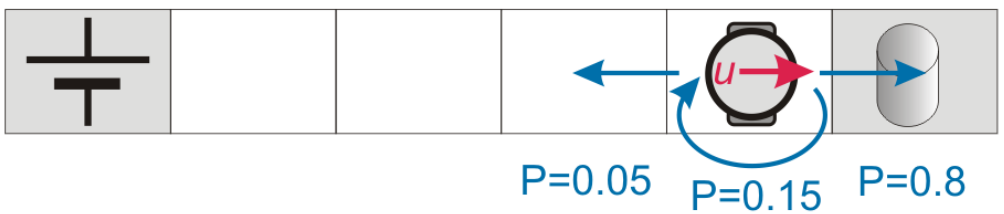
Exemple:

- Valorile unui zar aruncat la momente diferite de timp sunt independente. Printre altele, probabilitatea de a obţine 6 este independentă de câte valori 6 au fost obţinute la paşii anteriori
- Valorile temperaturii în două zile consecutive nu sunt independente! Sistemul este dinamic (are inerţie), valorile curente depind de cele anterioare

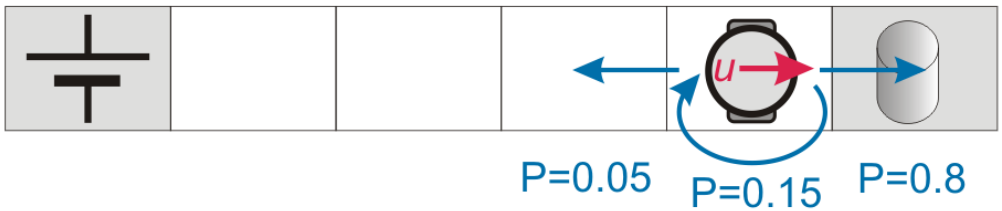
# Cazul stohastic

- Starea nu mai evoluează deterministic, ci **stohastic**

- Ex. robotul menajer “alunecă” şi:
  - se deplasează în direcţia intenţionată cu proba. 0.8
  - rămâne pe loc cu proba. 0.15
  - se deplasează în direcţia opusă cu proba. 0.05



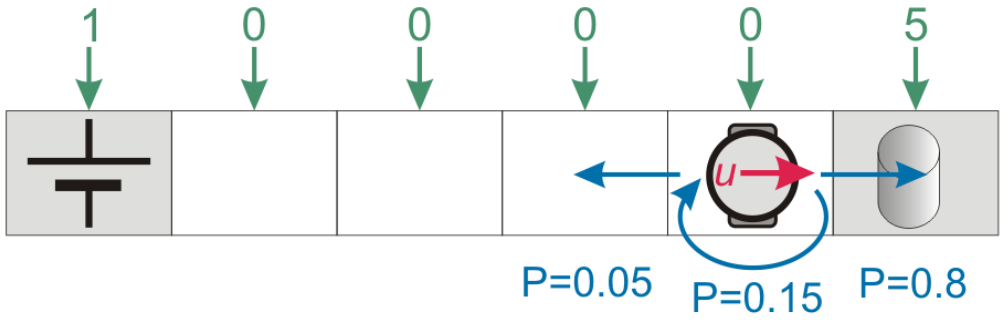
# Robot menajer stohastic: Funcţia de tranziţie




$\tilde{f}(x, u, x') =$ **probabilitatea** de a ajunge în $x'$
după ce $u$ a fost aplicată în $x$

$$\tilde{f}(x, u, x') = \begin{cases} 1 & \text{dacă } x \text{ terminal, } x' = x \\ 0.8 & \text{dacă } x \text{ neterminal, } x' = x + u \\ 0.15 & \text{dacă } x \text{ neterminal, } x' = x \\ 0.05 & \text{dacă } x \text{ neterminal, } x' = x - u \\ 0 & \text{altfel} \end{cases}$$

# Robot menajer stohastic: Funcţia de recompensă




- Tranziţia nu mai este complet determinată de $(x, u)$ $\Rightarrow$ starea următoare $x'$ trebuie inclusă explicit
- $\tilde{\rho}(x, u, x') =$ recompensa asociată atingerii $x'$ ca urmare a acţiunii $u$ în $x$
- Pentru robotul menajer:

$$\tilde{\rho}(x, u, x') = \begin{cases} 5 & \text{dacă } x \neq 5 \text{ şi } x' = 5 \\ 1 & \text{dacă } x \neq 0 \text{ şi } x' = 0 \\ 0 & \text{altfel} \end{cases}$$

# Proces de decizie Markov: cazul stohastic

## Proces de decizie Markov

1. Spaţiul stărilor $X$
2. Spaţiul acţiunilor $U$
3. Funcţia de tranziţie $\tilde{f}(x, u, x')$, $\quad \tilde{f} : X \times U \times X \to [0, 1]$
4. Funcţia de recompensă $\tilde{\rho}(x, u, x')$, $\quad \tilde{\rho} : X \times U \times X \to \mathbb{R}$

# Obiectiv în cazul stohastic

Găseşte $h$ care maximizează returnul aşteptat:

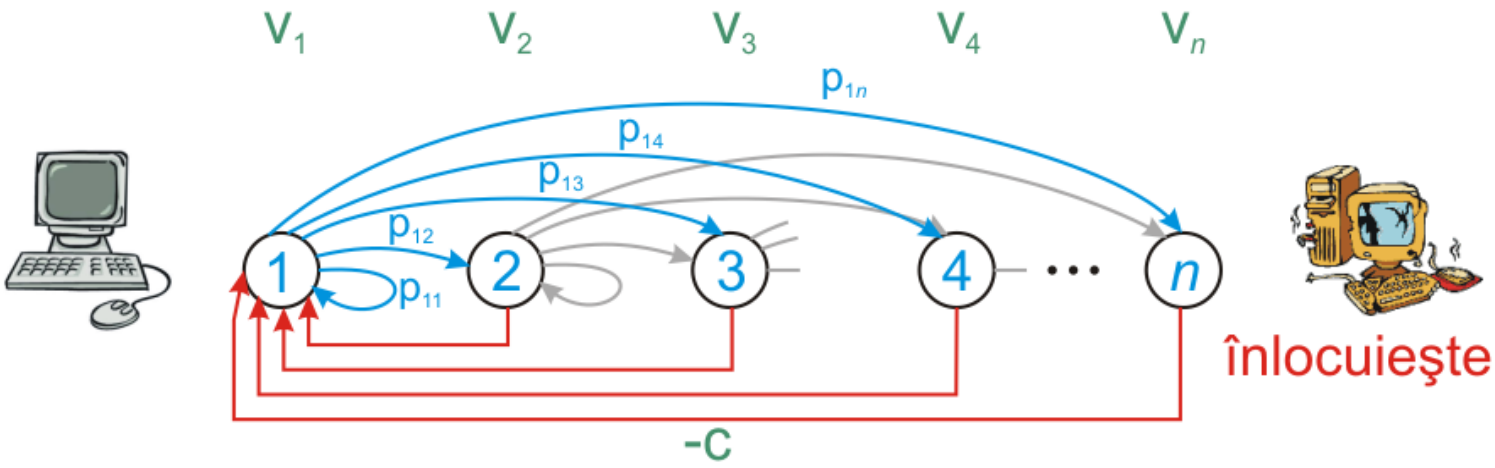
$$R^h(x_0) = \mathrm{E}_{x_1,x_2,\ldots}\left\{\sum_{k=0}^{\infty}\gamma^k\tilde{\rho}(x_k, h(x_k), x_{k+1})\right\}$$

din orice $x_0$

De notat:

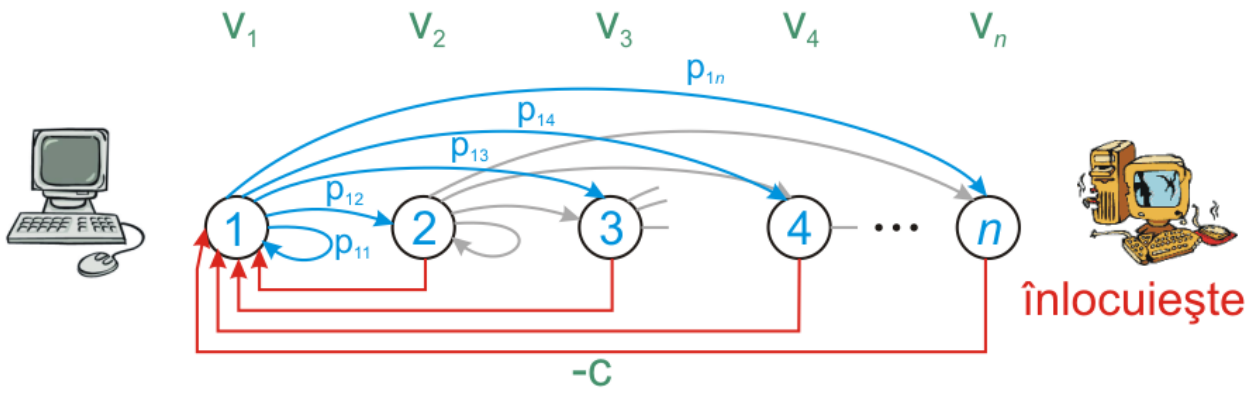
- legea de control $h(x)$ are aceeaşi structură
- factorul de discount $\gamma$ are aceeaşi semnificaţie

# Exemplu: Înlocuirea unei maşini



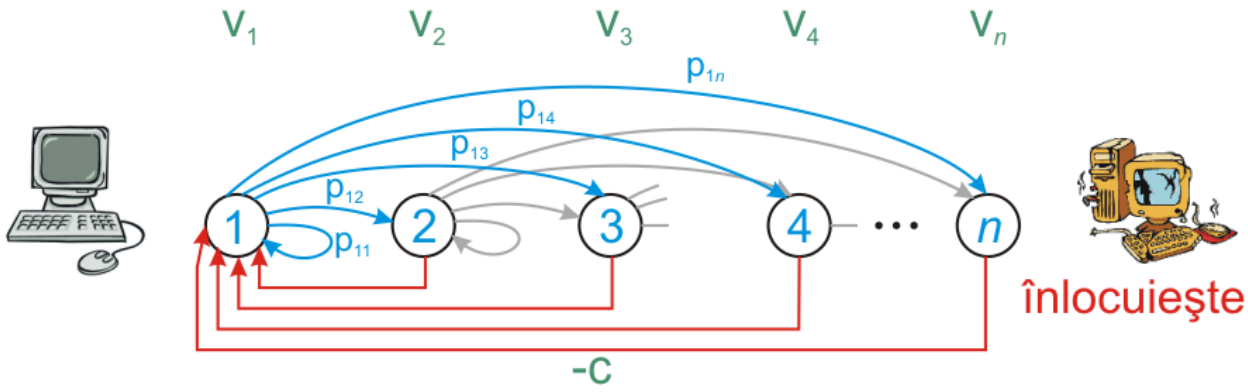

- Maşină de producţie cu $n$ stări diferite = grad de uzură
  1=stare perfectă, $n$=complet degradată
- Produce valoarea $v_i$ operând în starea $i$
- Uzură stohastică: starea $i$ trece în $j > i$ cu proba. $p_{ij}$, rămâne în $i$ cu $p_{ii} = 1 - p_{i,i+1} - \ldots - p_{i,n}$
- Maşina poate fi oricând înlocuită (presupunem instantaneu), plătind costul $c$

# Exemplu: Procesul de decizie Markov




- Spaţiul stărilor $X = \{1, 2, \ldots, n\}$
- Spaţiul acţiunilor $U = \left\{\text{Aşteaptă}, \text{Înlocuieşte}\right\}$
  (en. Wait, Replace)

# Exemplu: Procesul de decizie Markov (continuare)




- Funcţia de tranziţie:

$$\tilde{f}(x = i, u, x' = j) = \begin{cases} p_{ij} & \text{dacă } u = \text{A şi } i \leq j \\ 1 & \text{dacă } u = \text{I şi } j = 1 \\ 0 & \text{în orice altă situaţie} \end{cases}$$

- Funcţia de recompensă:

$$\tilde{\rho}(x = i, u, x' = j) = \begin{cases} v_i & \text{dacă } u = \text{A} \\ -c + v_1 & \text{dacă } u = \text{I} \end{cases}$$

# Înlocuirea unei maşini: motivare

Cadrul RL oferă o
lege de decizie optimală care
maximizează valoarea pe termen lung a maşinii

$$R^h(x_0) = \mathrm{E}_{x_1,x_2,\ldots}\left\{\sum_{k=0}^{\infty}\gamma^k\tilde{\rho}(x_k, h(x_k), x_{k+1})\right\}$$

# Terminologie engleză

| | |
|---|---|
| învăţarea prin recompensă | = reinforcement learning, RL |
| stare | = state |
| acţiune | = action |
| recompensă | = reward |
| funcţie de tranziţie | = transition function |
| funcţie de recompensă | = reward function |
| proces de decizie Markov | = Markov decision process |
| lege de control | = policy |
| return | = return |
| factor de discount | = discount factor |
| variabilă aleatoare | = random variable |
| funcţie de masă de probabilitate | = probability mass function |
| valoarea aşteptată | = expected value |

# Bibliografie

Material obligatoriu: slide-urile pentru cursuri


- D. Bertsekas, Dynamic Programming and Optimal Control, vol. 2, Athena Scientific, 2012.
- R. Sutton, A. Barto, Reinforcement Learning: An Introduction, ed. 2, 2018.
- L. Buşoniu, Reinforcement learning and dynamic programming for control, 2012 (lecture notes).

# Logistică

Notare:

- 50% examen
- 10% teste curs
- 30% laboratoare Matlab
- 20% laboratoare deep RL in Python

Regulament laboratoare:

- lab Matlab **obligatorii pentru participarea la examen**
- soluţie = raport PDF + cod: max 10p predată la timp, max 5p dacă întârzie
- soluţiile trebuie validate prin discuţii la colocviu
- orice laborator copiat $\Rightarrow$ recontractare

# Website, contact

`http://busoniu.net/teaching/ci2023`

Email: `lucian@busoniu.net`

## Info

- Material de curs (prezentări)
- Laboratoare
- Program
- etc.

# Test

Test