Spodbujevalno učenje pri igranju namiznih iger

(angl. *Reinforcement learning in board games*)

Tim Kalan

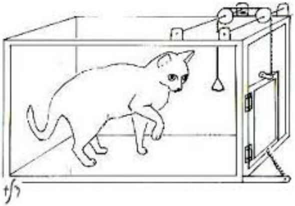
Mentor: prof. dr. Marjetka Knez

Fakulteta za matematiko in fiziko

14. december 2021

Motivacija: Instrumentalno pogojevanje

- Psihološko motivirana podlaga.
- **Nagrade in kazni**.

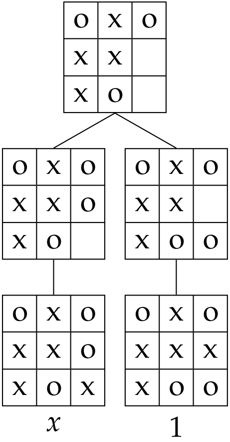



Okvir

- Agent »pade« v okolje.
- S poskušanjem se nauči pravilnih akcij.
- Svoje znanje izkoristi za maksimizacijo nagrade.

Primer: križci in krožci

- **Situacija/Stanje**: stanje na plošči,
- **Nagrada**: 1 za zmago, -1 za poraz, x za izenačenje/potezo,
- **Okolje**: nasprotnik, plošča, sodnik, nagrajevalec,
- **Akcija**: postavitev X oz. O na ploščo.



Agent

Naj bo $\mathcal{S}$ množica vseh stanj, $\mathcal{A}$ množica vseh akcij, S_t, R_t in A_t pa (slučajno) stanje, nagrada in akcija ob času t.

Agent

Naj bo $\mathcal{S}$ množica vseh stanj, $\mathcal{A}$ množica vseh akcij, S_t, R_t in A_t pa (slučajno) stanje, nagrada in akcija ob času t.

Agent ima tri glavne komponente:

- Strategija (angl. *Policy*)
- Vrednostna funkcija (angl. *Value function*)
- Model

Agent: strategija

Definicija 1.

- ***Deterministična strategija*** *je preslikava* $\pi : \mathcal{S} \to \mathcal{A}$,

$$\pi(s) = a.$$

- ***Stohastična strategija*** *je preslikava* $\pi : \mathcal{A} \times \mathcal{S} \to [0, 1]$,

$$\pi(a|s) = P(A_t = a \mid S_t = s).$$

Agent 3: vrednostna funkcija

Definicija 2 (Povračilo).

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Agent 3: vrednostna funkcija

Definicija 2 (Povračilo).

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Definicija 3 (Vrednostna funkcija).

- ***Vrednostna funkcija stanja*** *je pogojno matematično upanje povračila, če začnemo v stanju s in se nato vedemo skladno s strategijo* π

$$v_\pi(s) = \mathrm{E}[G_t \mid S_t = s].$$

Agent 3: vrednostna funkcija

Definicija 2 (Povračilo).

$$G_t = R_{t+1} + \gamma R_{t+2} + \ldots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Definicija 3 (Vrednostna funkcija).

- ***Vrednostna funkcija stanja*** *je pogojno matematično upanje povračila, če začnemo v stanju s in se nato vedemo skladno s strategijo* π

$$v_\pi(s) = \mathrm{E}[G_t \mid S_t = s].$$

- ***Vrednostna funkcija akcije*** *je podobna prejšnji, le da sprosti prvo akcijo*

$$q_\pi(s, a) = \mathrm{E}[G_t \mid S_t = s, A_t = a].$$

Formalizacija: Markovski proces odločanja 1

Definicija 4 (Markovska veriga).
Slučajni proces $(S_t)_{t=0}^{T}$ *na končnem verjetnostnem prostoru* $(\Omega, \mathcal{F}, P)$ *je* ***Markovska veriga****, če velja Markovska lastnost*

$$P(S_{t+1} = s_{t+1} \mid S_t = s_t, ..., S_0 = s_0) = P(S_{t+1} = s_{t+1} \mid S_t = s_t)$$

Formalizacija: Markovski proces odločanja 1

Definicija 4 (Markovska veriga).
Slučajni proces $(S_t)_{t=0}^{T}$ *na končnem verjetnostnem prostoru* $(\Omega, \mathcal{F}, P)$ *je* ***Markovska veriga****, če velja Markovska lastnost*

$$P(S_{t+1} = s_{t+1} \mid S_t = s_t, ..., S_0 = s_0) = P(S_{t+1} = s_{t+1} \mid S_t = s_t)$$

- Prihodnost je neodvisna od preteklosti, če poznamo sedanjost

Formalizacija: Markovski proces odločanja 1

Definicija 4 (Markovska veriga).
Slučajni proces $(S_t)_{t=0}^{T}$ *na končnem verjetnostnem prostoru* $(\Omega, \mathcal{F}, P)$ *je* ***Markovska veriga****, če velja Markovska lastnost*

$$P(S_{t+1} = s_{t+1} \mid S_t = s_t, ..., S_0 = s_0) = P(S_{t+1} = s_{t+1} \mid S_t = s_t)$$

- Prihodnost je neodvisna od preteklosti, če poznamo sedanjost
- $p_{ss'} := P(S_{t+1} = s' \mid S_t = s) \rightarrow \mathcal{P} := [p_{ss'}]_{s,s' \in \mathcal{S}}$,

Formalizacija: Markovski proces odločanja 1

Definicija 4 (Markovska veriga).
Slučajni proces $(S_t)_{t=0}^{T}$ *na končnem verjetnostnem prostoru* $(\Omega, \mathcal{F}, P)$ *je* ***Markovska veriga****, če velja Markovska lastnost*

$$P(S_{t+1} = s_{t+1} \mid S_t = s_t, ..., S_0 = s_0) = P(S_{t+1} = s_{t+1} \mid S_t = s_t)$$

- Prihodnost je neodvisna od preteklosti, če poznamo sedanjost
- $p_{ss'} := P(S_{t+1} = s' \mid S_t = s) \to \mathcal{P} := [p_{ss'}]_{s,s' \in \mathcal{S}}$,
- Markovska veriga je torej dvojica $(\mathcal{S}, \mathcal{P})$

Formalizacija: Markovski proces odločanja 2

Definicija 5 (Markovski proces nagrajevanja).
Markovski proces nagrajevanja *je nabor* $(\mathcal{S}, \mathcal{P}, \mathcal{R}, \gamma)$*, kjer je*

- $\mathcal{S}$ *(končna) množica stanj,*
- $\mathcal{P}$ *prehodna matrika, kjer* $\mathcal{P}_{ss'} = P(S_{t+1} = s' \mid S_t = s)$*,*
- $\mathcal{R}$ *nagradna funkcija* $\mathcal{R}_s = E[R_{t+1} \mid S_t = s]$*,*
- $\gamma \in [0, 1]$ *je diskontni faktor.*

Formalizacija: Markovski proces odločanja 3

Definicija 6 (Markovski proces odločanja).

Markovski proces odločanja (MDP) *je nabor* $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$*, kjer je*

- $\mathcal{S}$ *(končna) množica stanj,*
- $\mathcal{A}$ *(končna) množica akcij oz. dejanj,*
- $\mathcal{P}$ *prehodna matrika, kjer* $\mathcal{P}^a_{ss'} = P(S_{t+1} = s' \mid S_t = s, \mathbf{A_t} = \mathbf{a})$*,*
- $\mathcal{R}$ *nagradna funkcija* $\mathcal{R}^a_s = E[R_{t+1} \mid S_t = s, \mathbf{A_t} = \mathbf{a}]$*,*
- $\gamma \in [0, 1]$ *diskontni faktor.*

Algoritmi

- Učenje prek strategije ali **vrednostne funkcije**.
- Celoten problem je **načrtovanje**:
  - Napovedovanje - ugotvaljanje vrednosti.
  - Upravljanje - iskanje optimalne strategije.

Algoritmi: dinamično programiranje

Bellmanova enačba pričakovanja:

$$v_\pi(s) = \mathrm{E}[G_t \mid S_t = s] = \cdots = \mathrm{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s],$$
$$v_\pi = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi v_\pi,$$
$$v_{k+1} = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi v_k.$$

Izrek 7.
Za vsak končni Markovski proces odločanja s končnimi nagradami velja:

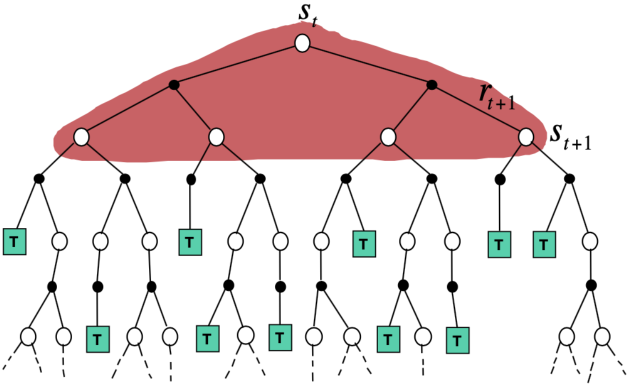
1. *Obstaja optimalna strategija π_*, ki je boljša ali enaka kot vse ostale strategije; $\pi_* \geq \pi$ za vsak $\pi \in \Pi$.*
2. *Vedno obstaja optimalna strategija, ki je deterministična.*
3. *Vse optimalne strategije določajo enako optimalno vrednostno funkcijo stanja in optimalno vrednostno funkcijo akcije:*

$$v_{\pi*}(s) = v_*(s),$$
$$q_{\pi*}(s,a) = q_*(s,a).$$

Algoritmi: dinamično programiranje

Bellmanova enačba pričakovanja:

$$v_\pi(s) = \mathrm{E}[G_t \mid S_t = s] = \cdots = \mathrm{E}[R_{t+1} + \gamma v_\pi(S_{t+1}) \mid S_t = s],$$
$$v_\pi = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi v_\pi,$$
$$v_{k+1} = \mathcal{R}^\pi + \gamma \mathcal{P}^\pi v_k.$$



Algoritmi: Monte Carlo 1

- Nepoznan epizodični MDP,
- problem napovedovanja,
- empirično povračilo,
- štejemo obiske stanj.

Algoritmi: Monte Carlo 2

- Ob **vsakem** obisku stanja s:

$$N(s) \leftarrow N(s) + 1$$
$$S(s) \leftarrow S(s) + G_t$$

- Po koncu učenja:

$$V(s) \leftarrow S(s)/N(s)$$

Algoritmi: Monte Carlo 2

- Ob **vsakem** obisku stanja s:

$$N(s) \leftarrow N(s) + 1$$
$$S(s) \leftarrow S(s) + G_t$$

- Po koncu učenja:

$$V(s) \leftarrow S(s)/N(s)$$

- Pomni: Računanje povprečja zaporedja $(X_i)_{i\in\mathbb{N}}$

$$\mu_k = \frac{1}{k}\sum_{j=1}^{k} X_j = \mu_{k-1} + \frac{1}{k}(X_k - \mu_{k-1})$$

Algoritmi: Monte Carlo 2

- Ob **vsakem** obisku stanja s:

$$N(s) \leftarrow N(s) + 1$$
$$S(s) \leftarrow S(s) + G_t$$

- Po koncu učenja:

$$V(s) \leftarrow S(s)/N(s)$$

- Pomni: Računanje povprečja zaporedja $(X_i)_{i\in\mathbb{N}}$

$$\mu_k = \frac{1}{k}\sum_{j=1}^{k} X_j = \mu_{k-1} + \frac{1}{k}(X_k - \mu_{k-1})$$

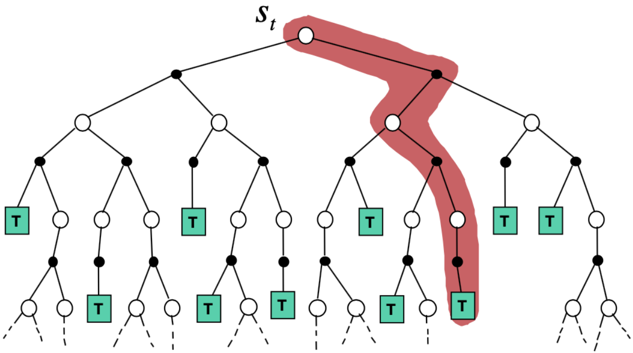
- Inkrementalni Monte Carlo:

$$V(S_t) \leftarrow V(S_t) + \frac{1}{N(S_t)}(G_t - V(S_t))$$

Algoritmi: Monte Carlo 3

- Inkrementalni Monte Carlo:

$$V(S_t) \leftarrow V(S_t) + \alpha \left(G_t - V(S_t) \right)$$



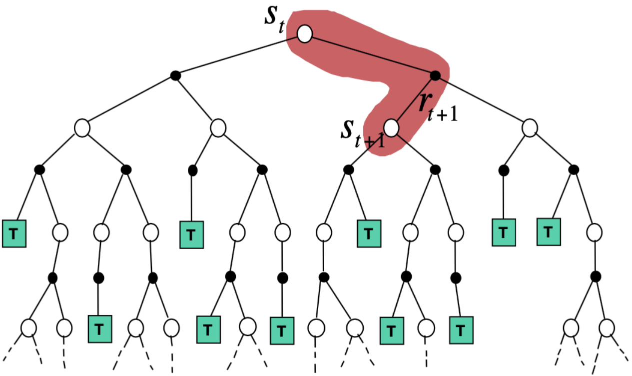
- Splošni obrazec:

$$\textit{nova ocena} \leftarrow \textit{stara ocena} + \textit{korak} \left(\textit{tarča} - \textit{stara ocena} \right).$$

Algoritmi: TD(0)

- Učenje s časovno razliko.
- *Bootstrapping*.
- Ne potrebujejo povračila.
- $G_t \approx R_{t+1} + \gamma V(S_{t+1})$.

$$V(S_t) \leftarrow V(S_t) + \alpha \left(R_{t+1} + \gamma V(S_{t+1}) - V(S_t)\right)$$



Algoritmi: TD(λ) 1

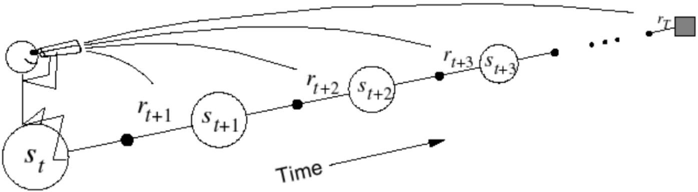
- Povezava med MC in TD(0).
- $G_t^{(n)} = R_{t+1} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})$.
- Povprečenje različnih $G_t^{(n)}$: $G_t^\lambda = (1 - \lambda) \sum_{n=1}^\infty \lambda^{(n-1)} G_t^{(n)}$.

Algoritmi: TD(λ) 1

- Povezava med MC in TD(0).
- $G_t^{(n)} = R_{t+1} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n V(S_{t+n})$.
- Povprečenje različnih $G_t^{(n)}$: $G_t^{\lambda} = (1-\lambda)\sum_{n=1}^{\infty} \lambda^{(n-1)} G_t^{(n)}$.

TD(λ) s **pogledom naprej**:

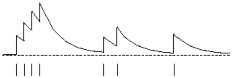
$$V(S_t) \leftarrow V(S_t) + \alpha(G_t^{\lambda} - V(S_t)).$$



Algoritmi: TD(λ) 2

- **Sledi upravičenosti** (angl. *eligibility traces*):

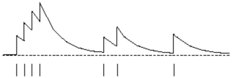
$$E_0(s) = 0,$$
$$E_t(s) = \gamma\lambda E_{t-1}(s) + \mathbb{1}(S_t = s),$$



Algoritmi: TD(λ) 2

- **Sledi upravičenosti** (angl. *eligibility traces*):

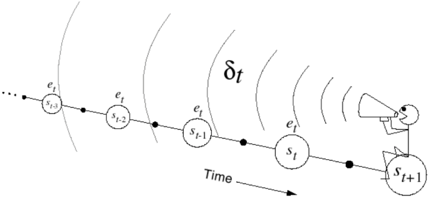
$$E_0(s) = 0,$$
$$E_t(s) = \gamma\lambda E_{t-1}(s) + \mathbb{1}(S_t = s),$$



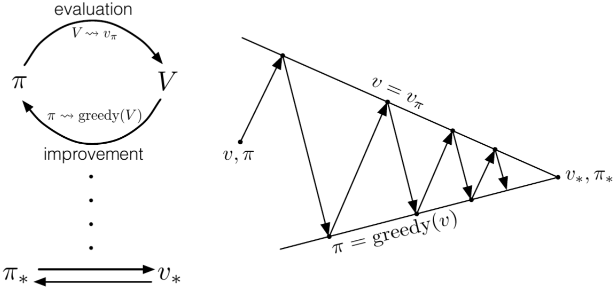
- $\delta_t = R_{t+1} + \gamma V(S_{t+1}) - V(S_t).$

TD(λ) s **pogledom nazaj**:

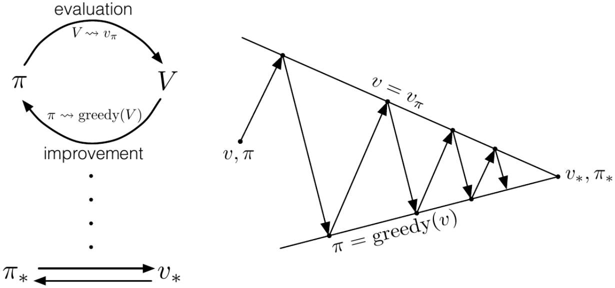
$$V(s) \leftarrow V(s) + \alpha\delta_t E_t(s).$$



Spreminjanje strategije - upravljanje



Spreminjanje strategije - upravljanje



- Potrebujemo vrednostno funkcijo akcij.
- Raziskovanje in izkoriščanje.
- ϵ-požrešna izbira akcij:

$$\pi'(a|s) = \begin{cases} \epsilon/m + 1 - \epsilon & \text{če } a = \arg\max_{a \in \mathcal{A}} q_\pi(s,a), \\ \epsilon/m & \text{sicer,} \end{cases}$$

Primer algoritma

Algorithm 1 SARSA

Podatki: parameter hitrosti učenja α, število epizod *stEpizod*, diskontni faktor γ, parameter požrešnosti ϵ
Poljubno nastavimo vrednosti $Q(s, a)$ za vsak $s \in \mathcal{S}$ in $a \in \mathcal{A}$
for $k = 1, 2, \ldots, stEpizod$ **do**
  Generiraj epizodo prek funkcije Q ϵ-požrešno
  $stanja \leftarrow$ seznam vseh opaženih stanj
  $nagrade \leftarrow$ seznam vseh opaženih nagrad
  $akcije \leftarrow$ seznam vseh opaženih akcij
  for $t = 1, 2, \ldots, length(stanja)$ **do**
    $s = stanja[t]$
    $s' = stanja[t+1]$
    $a = akcije[t]$
    $a' = akcije[t+1]$
    $Q(s,a) \leftarrow Q(s,a) + \alpha(nagrade[s'] + \gamma Q(s',a') - Q(s,a))$
  end for
end for

Konvergenca

PLNR:

- Vsi pari stanj in akcij so obiskani oz. uporabljeni neskončnokrat.
- Zaporedje strategij konvergira proti požrešni, torej

$$\lim_{k \to \infty} \pi_k(a|s) = \mathbb{1}(a = \arg\max_{a \in \mathcal{A}} q(s, a)).$$

Izrek 7.

Algoritem SARSA konvergira proti optimalni vrednostni funkciji akcij q_*, če veljata naslednja pogoja:

- *Zaporedje strategij je PLNR (npr. ϵ-požrešno z $\epsilon_k = 1/k$).*
- *Zaporedje parametrov hitrosti učenja α_k zadošča pogojema*

$$\sum_{k=1}^{\infty} \alpha_k = \infty \ \ \textit{in} \ \ \sum_{k=1}^{\infty} \alpha_k^2 < \infty.$$

Aproksimacija

- Veliki MDP-ji
  - križci in krožci: 3^9 / 4578 / 765 stanj,
  - štiri v vrsto: 4.531.985.219.092 stanj,
  - šah: približno 10^{46} stanj,
  - go: 10^{170} stanj,
- Vsi zgornji algoritmi so tabelarični.
- $\hat{v}(s, w) \approx v_\pi(s)$ oz. $\hat{q}(s, a, w) \approx q_\pi(s, a)$
- Linearna aproksimacija ali nevronske mreže
- Konvergenca?

Gradientni spust

- Premikamo se proti minimumu funkcije J,
$$\Delta w = -\alpha \nabla_w J(w).$$
- V našem primeru srednja kvadratična napaka
$$\begin{aligned} J(w) &= E_\pi[(v_\pi(s) - \hat{v}(s,w))^2], \\ \Delta w &= \alpha E_\pi[(v_\pi(s) - \hat{v}(s,w)) \nabla_w \hat{v}(s,w)]. \end{aligned}$$
- Ker ne poznamo okolja, vzorčimo
$$\Delta w = \alpha (v_\pi(s) - \hat{v}(s,w)) \nabla_w \hat{v}(s,w).$$
- Algoritem TD(λ) je potem
$$\Delta w = \alpha (G_t^\lambda - \hat{v}(s,w)) \nabla_w \hat{v}(s,w).$$

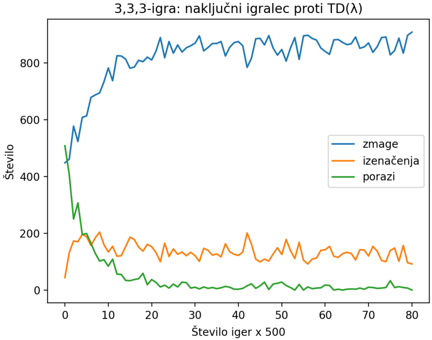
Namizne igre

- Kombinatorne igre: dva igralca, popolna informacija, izmenične poteze,
- Enostavno nagrajevanje,
- »po-stanja«,
- Pridobivanje iger:
  - Nasprotnik iz podatkovne baze,
  - naključni nasprotnik,
  - fiksiran nasprotnik,
  - samoigra.
- Gledamo skupno strategijo $\pi = (\pi_1, \pi_2)$ in minimax vrednostno funkcijo

$$v_*(s) = \max_{\pi_1} \min_{\pi_2} v_\pi(s).$$

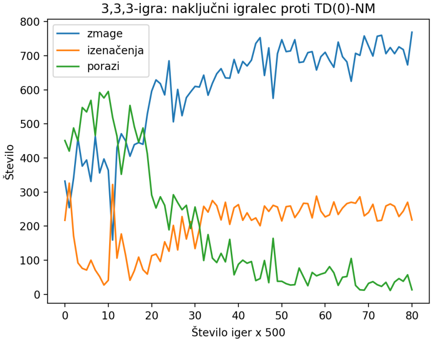
Rezultati

- 3,3,3-igra in tabelarični agent



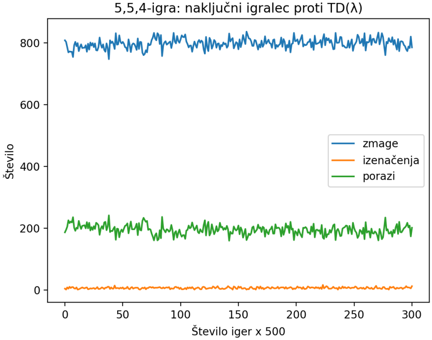
Rezultati

- 3,3,3-igra in tabelarični agent
- 3,3,3-igra in agent z nevronsko mrežo



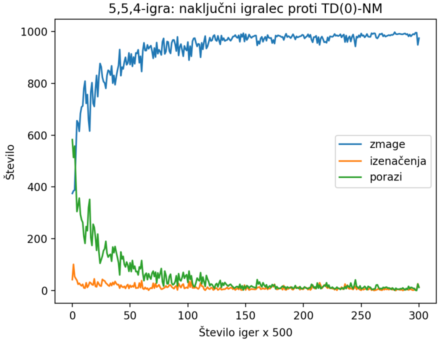
Rezultati

- 3,3,3-igra in tabelarični agent
- 3,3,3-igra in agent z nevronsko mrežo
- 5,5,4-igra in tabelarični agent



Rezultati

- 3,3,3-igra in tabelarični agent
- 3,3,3-igra in agent z nevronsko mrežo
- 5,5,4-igra in tabelarični agent
- 5,5,4-igra in agent z nevronsko mrežo



Literatura I

R. E. Bellman.
Dynamic Programming.
Princeton University Press, Princeton, 1957.

R. E. Bellman.
A markov decision process.
Journal of Mathematical Mechanics, 6, 1957.

I. Ghory.
Reinforcement learning in board games,
2004.

A. W. Hales in R. I. Jewett.
Regularity and positional games.
Transactions of the American Mathematical Society, 106, 1963.

K. Hornik, M. Stinchcombe in H. White.
Multilayer feedforward networks are universal approximators.
Neural networks, 2, 1989.

D. Silver.
Introduction to reinforcement learning.
https://deepmind.com/learning-resources/-introduction-reinforcement-learning-david-silver, 2015.

D. Silver idr.
Mastering the game of go with deep neural networks and tree search.
Nature, 529, 2016.

Literatura II

S. Singh, T. Jaakkola, M. L. Littmanm in C. Szepesvari.
Convergence results for single-step on-policy reinforcement-learning algorithms.
Machine Learning, 39, 2000.

S. Singh in R. S. Sutton.
Reinforcement learning with replacing eligibility traces.
Machine Learning, 22, 1996.

R. S. Sutton.
Learning to predict by the methods of temporal differences.
Machine Learning, 3, 1988.

R. S. Sutton in A. G. Barto.
Reinforcement Learning: An introduction.
The MIT Press, Cambridge, Massachusetts, 2 edition, 2015.

C. Szepesvari.
Algorithms for Reinforcement Learning.
Morgan & Claypool Publishers, Alberta, Canada, 2009.

The Agent-Environment Interaction figure from Reinforcement Learning: An Introduction by Richard S. Sutton and Andrew G. Barto reproduced in Tikz, v: GitHub, [ogled 5. 8. 2021], dostopno na https://gist.github.com/pierrelux/6501790.

Creating Tic-Tac-Toe boards with LaTeX/TikZ, v: TeX - LaTeX Stack Exchange, [ogled 12. 9. 2021], dostopno na https://tex.stackexchange.com/questions/139782/creating-tic-tac-toe-boards-with-latex-tikz.