

Uvod u umjetnu inteligenciju

UNIZG FER, ak. god. 2021/2022

Vježbe, v1

2 Pretraživanje prostora stanja

- 1 (T) Dio definicije problema pretraživanja prostora stanja jest funkcija sljedbenika, $\text{succ} : S \rightarrow \wp(S)$, koja definira prijelaze između stanja. Tu funkciju možemo definirati eksplisitno ili implicitno. **Koja je prednost implicitne definicije sljedbenika nad eksplisitnom definicijom?**
- [A] Eksplisitna definicija nije moguća ako je skup stanja beskonačan ili ako u prostoru stanja postoje ciklusi, dok to nije problem s implicitnom definicijom, pod uvjetom da u skupu stanja postoji barem jedno ciljno stanje
 - [B] Implicitna definicija koristi skup operatora koji definiraju sljedbenička stanja, čime se osigurava potpunost algoritma pretraživanja, dok s eksplisitnom definicijom algoritam može zaglaviti u beskonačnoj petlji
 - [C] Eksplisitna definicija u praksi nije moguća kada je broj stanja previelik, i tada je lakše implicitno (algoritamski) definirati na koje se sve načine stanje može promijeniti kako bi se dobilo iduće stanje
 - [D] Implicitna definicija lako se može proširiti pokazivačem na roditeljski čvor, što je potrebno za rekonstrukciju rješenja, dok je kod eksplisitne definicije za to potrebno dodatno pohranjivati otvorene čvorove
- 2 (R) Funkcijom succ definirani su prijelazi između stanja $\{a, b, c, d, e, f\}$ na sljedeći način: $\text{succ}(a) = \{b, c\}$, $\text{succ}(b) = \{c, d\}$, $\text{succ}(c) = \{d, e\}$, $\text{succ}(d) = \{a, e\}$, $\text{succ}(e) = \{f\}$, $\text{succ}(f) = \{d\}$. Početno stanje je a , a ciljno f . Pretpostavite leksikografski poredak između čvorova. **Koji će algoritam čvorove ispitivati redoslijedom $\dots, b, c, a, b, c, \dots$?**
- [A] Pretraživanje u širinu
 - [B] Pretraživanje u dubinu
 - [C] Iterativno pretraživanje u dubinu
 - [D] Ograničeno pretraživanje u dubinu sa $k = 3$

3 Heurističko pretraživanje

- 3 (T) Algoritam pretraživanja “najbolji prvi” nije optimalan i nazivamo ga “pohlepnim algoritmom”. Međutim, algoritam pretraživanja u dubinu također nije optimalan, ali ga ne nazivamo pohlepnim. **Zašto algoritam “najbolji prvi” nazivamo pohlepnim, a pretraživanje u dubinu ne**

nazivamo tako?

- A Zato što pretraživanje u dubinu za listu otvorenih čvorova koristi stog i zatvara jednom proširene čvorove, dok "najbolji prvi" koristi prioritetni red i nikada ne zatvara proširene čvorove, pa ima veću prostornu složenost od pretraživanja u dubinu
- B Zato što "najbolji prvi" uvijek odabire čvor koji se trenutačno čini najboljim s obzirom na heurističku vrijednost, neovisno o tome što je ukupno najbolji put, dok pretraživanje u dubinu ne odabire čvor prema heurističkoj vrijednosti nego po dubini
- C Oba algoritma su neoptimalna jer ne pretražuju razinu po razinu, međutim "najbolji prvi" je pohlepan jer ne uzima u obzir heurističku vrijednost čvora već samo ukupnu cijenu puta, pa će u pravilu davati bolja (kraća) rješenja od pretraživanja u dubinu
- D Zato što "najbolji prvi" koristi heurističku funkciju koja, ako nije optimalna, ne mora rezultirati optimalnim putem, dok algoritam u dubinu ne koristi heurističku vrijednost već čvorove odabire po dubini, pa može zaglaviti u beskonačnoj petlji

- 4 (R) Neka je definiran skup stanja $S = \{a, b, c, d, e, f\}$ te funkcija sljedbenika $\text{succ}(a) = \{(b, 6), (d, 2)\}$, $\text{succ}(b) = \{(c, 3), (d, 5)\}$, $\text{succ}(c) = \{(d, 4), (f, 1)\}$, $\text{succ}(d) = \{(e, 2)\}$, $\text{succ}(e) = \{(b, 1), (f, 6)\}$, $\text{succ}(f) = \emptyset$. Početno stanje je a , a ciljno f . Heurističke vrijednosti stanja neka su $h(a) = 9$, $h(b) = 2$, $h(c) = 1$, $h(d) = 7$, $h(e) = 2$ te $h(f) = 0$. Provedite pretraživanje algoritmom A^* , zapisujući u svakoj iteraciji sadržaje liste otvorenih čvorova (O) i liste zatvorenih čvorova (C). U nultoj iteraciji, vrijedi $O = [(a, 0)]$ i $C = \emptyset$. **Kako izgledaju liste O i C nakon pете iteracije algoritma A^* ?**

- A $O = [(c, 8), (f, 10)]$, $C = \{(a, 0), (b, 5), (d, 2), (e, 4)\}$
- B $O = [(e, 4), (c, 9)]$, $C = \{(a, 0), (b, 6), (d, 2)\}$
- C $O = [(f, 10)]$, $C = \{(a, 0), (b, 6), (d, 2), (c, 9)\}$
- D Algoritam završava prije pете iteracije

- 5 (R) Algoritmom A^* rješavamo problem slagalice 3×3 . Cijene svih prijelaza jednake su 1, a za heuristiku koristimo broj pločica koje nisu na svome mjestu (pritom se prazna pozicija ne računa, pa je najveća moguća vrijednost heuristike jednaka 8). Tražeći put od početnog stanja s_0 do ciljnog stanja s_g , algoritam A^* u jednom trenutku generira čvor sa stanjem s_x . Cijena puta $g(n)$ od čvora sa stanjem s_0 do čvora sa stanjem s_x iznosi 15. Stanje s_x i ciljno stanje s_g su sljedeći:

$$s_x = \begin{array}{|c|c|c|} \hline 4 & 1 & 3 \\ \hline 7 & 2 & 5 \\ \hline 8 & \square & 6 \\ \hline \end{array} \quad s_g = \begin{array}{|c|c|c|} \hline 1 & 2 & 3 \\ \hline 4 & 5 & 6 \\ \hline 7 & 8 & \square \\ \hline \end{array}$$

Neposredno prije proširenja čvora sa stanjem s_x , svi ostali čvorovi u listi otvorenih čvorova imaju cijenu $f(n)$ koja je za barem 5 veća od cijene $f(n)$ čvora sa stanjem s_x . Izvedite sljedeće tri iteracije algoritma A^* . **Koliko iznose cijene $f(n)$ iduća tri čvora koje će algoritam ispitati i proširiti?**

- A 22, 22, 22
- B 22, 23, 21
- C 23, 22, 20
- D 23, 23, 23

- 6 (P) Rješavamo problem slagalice 4×4 i implementirali smo tri heurističke funkcije: h_1 , h_2 i h_3 . Heuristiku h_1 implementirali smo tako da interno izvodi pretraživanje u dubinu ograničeno na dubinu $k = 5$, a kao procjenu vraća dubinu na kojoj je pronađeno rješenje, ili k ako rješenje nije pronađeno. Heuristiku h_2 implementirali smo kao iterativno pretraživanje u dubinu, ali ograničeno na dubinu $k = 3$. Heuristika vraća dubinu na kojoj je pronađeno rješenje, ili k ako rješenje nije pronađeno. Konačno, heuristiku h_3 izveli smo kao iterativno pretraživanje u dubinu, ali modificirano tako da u svakoj iteraciji dubinsko ograničenje povećavamo za 3 umjesto za 1. I ta heuristika vraća dubinu pronađenog rješenja. Želimo da algoritam A^* bude što obavješteniji, ali da i dalje bude i potpun i optimalan. To možemo ostvariti kombinacijom heuristika h_1 , h_2 i h_3 . **Koja od navedenih**

kombinacija ovih triju heuristika daje najobavješteniji, ali još uvijek potpun i optimalan algoritam A^* ?

- A $\min(\min(h_1, h_2), \max(h_2, h_3))$
- B $\min(\max(h_1, h_2), \min(h_1, h_3))$
- C $\max(h_1, h_2, h_3)$
- D $\min(\max(h_1, h_2), h_3)$

- 7 (P) Rješavamo problem nalaženja najkraćeg puta iz bilo koje točke u gradu Zagrebu do Trga Sv. Marka na Gornjem gradu. Cijena puta je očekivano vrijeme pješačenja u minutama za prosječno zainteresiranog turista. Razmotrimo jedan detalj na našoj karti između Trga bana Josipa Jelacića (s_1), Dolca (s_2), početka Zakmardijevih stuba (s_3) i Krvavog mosta (s_4). Od s_1 do s_2 dolazi se po Splavnici za 2 minute, a od s_2 do s_4 Tkalčićevom za 5 minuta. Od s_1 do s_3 dolazi se Radićevom ulicom za 3 minute, a od s_3 do s_4 nastavkom Radićeve ulice za 4 minute. Za nalaženje puta do Trga Sv. Marka koristimo algoritam A^* s nekom heuristikom h , koja je konzistentna. Pritom vrijedi $h(s_1) = 13$, $h(s_2) = 14$, $h(s_3) = 12$, $h(s_4) = 10$. Željeli bismo ubrzati pretraživanje (smanjiti broj iteracija algoritma A^*), a da smo pritom sigurni da je nađeni put stvarno vremenski najkraći put. **Koja od navedenih promjena u vrijednosti heurističke funkcije h će najviše ubrzati pretragu, ali i dalje davati najkraći put do cilja?**

- A $h(s_1) = 16$
- B $h(s_1) = 12$
- C $h(s_1) = 14$
- D $h(s_1) = 15$

Uvod u umjetnu inteligenciju

UNIZG FER, ak. god. 2021/2022

Vježbe, v1

4 Igranje igara

- 1 (T) Heuristička funkcija koristi se kod algoritama usmjerenog pretraživanja, ali i kod algoritma minimax. U oba slučaja heuristička funkcija služi za numeričku procjenu kvalitete trenutačnog stanja. Međutim, postoji razlika u načinu na koji se heuristička funkcija koristi u ta dva slučaja. **Koja je razlika u načinu korištenja heurističke funkcije kod algoritama usmjerenog pretraživanja i kod algoritma minimax?**
- [A] Algoritam minimax koristi pretraživanje u dubinu, pa heurističku funkciju primjenjuje uvijek na najdublji čvor, dok se kod algoritma pretraživanja heuristika primjenjuje na čvor koji je prvi u listi otvorenih čvorova
- [B] Kod algoritma minimax heuristika se ne primjenjuje na sve čvorove u stablu igre već samo na čvorove na određenoj dubini, dok se kod algoritma pretraživanja heuristika primjenjuje na sve generirane čvorove
- [C] Kod algoritma pretraživanja uvijek se proširuje čvor koji ima najmanju vrijednost heuristike, jer to znači da je najbliži cilju, dok se kod algoritma minimax obabire čvor koji ima najveću vrijednost heurističke funkcije
- [D] Algoritmi pretraživanja listu otvorenih čvorova sortiraju na temelju vrijednosti heuristike, dok algoritam minimax uvijek proširuje čvor s najvećom heurističkom vrijednošću, i tako ostvaruje pretraživanje u dubinu
- 2 (R) Stablo igre definirano je prijelazima $\text{succ}(A) = \{B, C\}$, $\text{succ}(B) = \{D, E\}$, $\text{succ}(D) = \{H, I\}$, $\text{succ}(E) = \{J, K, L\}$, $\text{succ}(C) = \{F, G\}$, $\text{succ}(F) = \{M, N, O\}$, $\text{succ}(G) = \{P, Q\}$. Heurističke vrijednosti listova su $h(H) = h(J) = 8$, $h(I) = -15$, $h(K) = -h(Q) = 17$, $h(L) = h(O) = 1$, $h(M) = -12$, $h(N) = -19$, $h(P) = -14$. Optimalna strategija određuje se algoritmom minimax uz alfa-beta podrezivanje. Prvi na potezu je igrač MAX. **Koji će čvorovi pritom biti podrezani (preskočeni pri izračunu minimax vrijednosti)?**
- [A] L, Q [B] O, P, Q [C] L, F, M, N, O [D] K, L, G, P, Q
- 3 (R) Razmatramo igru za dva igrača sa sumom nula. Svako stanje $s \in S$ te igre može se sažeto opisati troznamenkastim prirodnim brojem između 111 i 999. Sljedbenička stanja od s jesu sva ona stanja koja se dobivaju iz s tako da se jedna znamenka poveća za jedan, npr., $\text{succ}(235) = \{335, 245, 236\}$. Međutim, stanja koja sadrže znamenku 9 su završna stanja i ona nemaju sljedbeničkih stanja, npr., $\text{succ}(932) = \emptyset$. U završnim stanjima, isplata za prvog igrača (MAX) jednaka je razlici prve i treće znamenke, npr., $\text{utility}(932) = 9 - 2 = 7$. Isplata za drugog igrača (MIN) je negativna vrijednost isplate za prvog igrača. U igri se natječu dva algoritma minimax, A_1 (igrač MAX) i A_2 (igrač MIN). Oba algoritma pretražuju do dva poteza unaprijed, tj. dubinsko ograničenje algoritma minimax jednako je 2. Međutim, algoritmi koriste različite heuristike. Heuristika h_1 algoritma A_1 jednaka je prvoj znamenici iz s , a heuristika h_2 algoritma A_2 (definirana iz perspektive tog algoritma) jednaka je trećoj znamenici iz s . Npr., $h_1(236) = 2$ i $h_2(236) = 6$ (kod izračuna minimax vrijednosti u stablu sa MAX korijenom vrijednost heuristike h_2 trebate negirati). Početno stanje igre neka je $s_0 = 175$. Prvi potez vuče algoritam A_1 (igrač MAX). **Koji je slijed stanja igre, ako oba igrača vuku svoje minimax-optimalne poteze?**
- [A] $175 \rightarrow 176 \rightarrow 276$ [B] $175 \rightarrow 176 \rightarrow 186$ [C] $175 \rightarrow 275 \rightarrow 375$ [D] $175 \rightarrow 275 \rightarrow 276$

4 (P) Razmatramo dva igrača algoritma: jedan s heurstikom h_1 i dubinom pretraživanja d_1 te drugi s heurstikom h_2 i dubinom pretraživanja d_2 . Oba algoritma koriste strategiju minimax. **Kada će prvi algoritam češće pobjedivati drugi algoritam?**

- Ako $h_1 > h_2$ Ako $d_1 > d_2$ Ako $h_1 = h_2$ i $d_1 > d_2$ Ako $d_1 = d_2$ i $h_1 > h_2$

5 (P) U šahovskom turniru sudjeluju četiri igrača algoritma, A_1 , A_2 , A_3 i A_4 . Sva četiri algoritma implementirana su kao minimax s podrezivanjem alfa-beta. Međutim, algoritmi se razlikuju po heuristici i dubinskom ograničenju. Neka je h_i heuristika a d_i dubinsko ograničenje algoritma A_i . Neposredno prije turnira u javnost su procurile sljedeće važne informacije: (1) h_3 je najbolja od četiriju heuristika, (2) h_2 i h_4 su identične heuristike, (3) kod algoritma A_3 greškom je isključeno alfa-beta podrezivanje te (4) dubinska ograničenja su takva da vrijedi $d_1 > d_2 = d_3 > d_4$. Turnir se odvija u dva kruga. U prvoj krugu igraju A_1 protiv A_2 (prvi par) te A_3 protiv A_4 (drugi par), dok se u drugome krugu sučeljavaju pobjednici iz prvog kruga. Uzvrat se ne igraju. Na turniru postoji vremensko ograničenje za svaki potez, ali su dubinska ograničenja algoritama takva da niti jedan algoritam nikada ne doseže vremensko ograničenje, pa je ono zapravo nebitno. **Na temelju dostupnih nam informacija, za koji algoritam očekujemo da će pobijediti na ovom šahovskom turniru?**

- A_1 ili A_3 A_2 ili A_4 A_3 ili A_4 A_2 ili A_3

Uvod u umjetnu inteligenciju

UNIZG FER, ak. god. 2021./2022.

Vježbe, v3

5 Prikazivanje znanja formalnom logikom

- 1 (T) Kod odluke koji sustav formalne logike upotrijebiti za prikaz znanja, valja uzeti u obzir kompromis između ekspresivnosti i odlučivosti. Taj kompromis postoji već kod izbora između propozicijske logike (PL) i logike prvog reda (FOL). **Kako se taj kompromis manifestira kod izbora između PL i FOL?**
- [A] FOL je ekspresivnija od PL jer sadrži varijable i kvantifikatore, međutim, za razliku od PL, FOL nije odlučiva, što znači da u FOL općenito ne postoji postupak za dokazivanje deduktivne posljedice koji bi bio istovremeno i potpun i ispravan
 - [B] PL je manje ekspresivna od FOL, ali je PL odlučiva, pa postoji postupak dokazivanja deduktivne posljedice koji je istovremeno i potpun i ispravan, što znači da možemo dokazati svaku logičku posljedicu, dok je kod FOL to moguće samo uz potpunu strategiju dokazivanja
 - [C] FOL je ekspresivnija od PL jer može prikazati objekte i odnose između njih, međutim, za razliku od PL, FOL nije odlučiva, što znači da u FOL ne možemo uvijek odrediti da neka formula nije logička posljedica skupa premissa
 - [D] PL je ekspresivnija od FOL jer se svaka varijabla iz PL može prikazati kao unarni predikat iz FOL, no PL nije odlučiva budući da je broj interpretacija eksponencijalan u odnosu na broj varijabli, dok je kod FOL broj interpretacija konačan

- 2 (P) Dana je FOL interpretacija s domenom $D = \{a, b, c\}$, preslikavanjem $f(a) = c, f(b) = a, f(c) = a$ te ekstenzijama predikata $\text{ext}(P) = \{(a, a), (a, c), (b, b), (c, a)\}$ i $\text{ext}(Q) = \{b, c\}$. Ova je interpretacija **model** nekih formula. **Koja od sljedećih formula ima ovu interpretaciju kao svoj model?**

- [A] $\forall x Q(f(x))$
- [B] $\forall x (P(a, x) \rightarrow Q(x))$
- [C] $\exists x \forall y P(x, y)$
- [D] $\exists x (Q(x) \rightarrow \forall y P(y, y))$

- 3 (P) Neka $F(x)$ označava "x je Francuz", $S(x)$ označava "x je sir" te $V(x, y)$ označava "x voli y". **Koja od sljedećih formula predikatne logike predstavlja ispravnu formalizaciju (stereotipne) rečenice "Svi Francuzi vole sir"?**

- [A] $\forall x \forall y (V(x, y) \rightarrow (F(x) \wedge S(y)))$
- [B] $\exists x \exists y ((F(x) \wedge S(y)) \rightarrow V(x, y))$
- [C] $\forall x \forall y ((F(x) \wedge S(y)) \wedge V(x, y))$
- [D] $\forall x (F(x) \rightarrow \forall y (S(y) \rightarrow V(x, y)))$

6 Automatsko zaključivanje

- 4 (T) Semantička posljedica fundamentalan je logički koncept, ali je njezino dokazivanje u praksi problematično. **Što je točno problem s dokazivanjem semantičke posljedice?**

- [A] Nepotpunost
- [B] Netraktabilnost
- [C] Nedeterminističnost
- [D] Neispravnost

5 (T) Postupak zaključivanja idealno je ispravan i potpun. **Koji je nužan i dovoljan uvjet za potpunost postupka zaključivanja?**

- A Potpun i ispravan skup pravila
- B Potpun skup pravila i potpuna strategija
- C Konačan broj interpretacija
- D Optimalna strategija

6 (T) Neka pravila zaključivanja nisu ispravna. Jedno takvo pravilo je abdukcija. **Što bi trebalo vrijediti, a da bi abdukcija bila ispravna?**

- A $A \rightarrow B, \neg B \vDash \neg A$
- B $A \rightarrow B, A \vDash B$
- C $A \rightarrow B, B \rightarrow C \vDash A \rightarrow C$
- D $A \rightarrow B, B \vDash A$

7 (P) Ispravnost je važno svojstvo pravila zaključivanja. **Koje je od sljedećih pravila zaključivanja ispravno?**

- A $A \rightarrow (B \rightarrow C) \vdash (A \vee B) \rightarrow C$
- B $A \rightarrow (B \vee C), \neg C \vdash A \rightarrow B$
- C $A \rightarrow B, B \rightarrow C \vdash C \rightarrow A$
- D $B, A \rightarrow B \vdash A$

8 (R) Zadane su premise: "Ines je sretna (S) ako ima novaca (N) ili ako je na godišnjem odmoru (O) ili ako je u Zagrebu (Z). Ako nije na godišnjem odmoru, Ines se javlja na telefon (J). Kada na telefon zove Arsen (A), onda se Ines ne javlja na telefon. Ines nije u Zagrebu ili ima novaca." **Koja je od sljedećih tvrdnji logička posljedica ovih premissa?**

- A Ako je Ines sretna, onda je Arsen zove na telefon.
- B Ines je sretna ako je Arsen zove na telefon.
- C Ako je Ines sretna, onda se ne javlja na telefon.
- D Ines se ne javlja na telefon.

9 (P) Za automatsko zaključivanje u PL koristimo rezoluciju opovrgavanjem uz strategiju skupa potpore (SOS). Na takav postupak zaključivanja možemo gledati kao na problem pretraživanje prostora stanja, gdje stanja odgovaraju skupu klauzula (onih zadanih i onih izvedenih), a prijelazi između stanja odgovaraju primjeni rezolucijskog pravila na jedan par klauzula. Takav problem pretraživanje ima i svoj faktor grananja, koji ovisi o dubini stabla, tj. o koraku zaključivanja. Neka skup premissa sadrži 10 klauzula, a negirani cilj 5 klauzula. **Koliko iznosi gornja ograda na faktor grananja u drugom koraku zaključivanja?** (Napomena: Jednom razriješeni par klauzula više se ne razrješava.)

- A 104
- B 74
- C 119
- D 60

Uvod u umjetnu inteligenciju

UNIZG FER, ak. god. 2021./2022.

Vježbe, v3

6 Automatsko zaključivanje

1 (T) Rezolucija opovrgavanjem je ispravno i potpuno pravilo zaključivanja u FOL. **Što to znači?**

- Ako formula nije logička posljedica, onda to ne možemo dokazati rezolucijom opovrgavanjem
- Postupak izvodi klauzulu NIL ako i samo ako je cilj logička posljedica premisa
- Kad god je skup premisa konzistentan, rezolucijsko pravilo izvodi NIL klauzulu
- Postupak završava u konačnom broju koraka s odlukom je li formula logička posljedica premisa ili nije

2 (T) Postupak rezolucije prikidan je za automatizaciju budući da se oslanja na samo jedno pravilo zaključivanja. Međutim, od svakog postupka zaključivanja, pa tako i od rezolucije, očekujemo da je ispravan i po mogućnosti potpun. Postupak rezolucije je ispravan i potpun, kako u PL tako i u FOL, ali pod određenim uvjetima. **Pod kojim je uvjetima postupak rezolucije ispravan i potpun postupak zaključivanja u FOL?**

- Postupak je ispravan uz skolemizaciju egzistencijalnih varijabli i korištenje strategije skupa potpore, a potpun ako se provodi kao rezolucija opovrgavanjem nad standardiziranim i faktoriziranim klauzulama
- Postupak je potpun uz rezoluciju opovrgavanjem, ako se ista varijabla ne javlja u različitim klauzulama i ako se varijabla ne supstituira izrazom koji ju sadrži, a ispravan ako se u svakom koraku provodi faktorizacija
- Postupak je bezuvjetno ispravan, a potpun uz rezoluciju opovrgavanjem nad standardiziranim klauzulama, ako se razrješavaju sve kombinacije izvornih klauzula i faktor-klauzula, i ako se koristi potpuna strategija dokazivanja
- Postupak je bezuvjetno potpun, a ispravan uz rezoluciju opovrgavanjem nad klauzulama koje ne dijele iste varijable te uz razrješavanje svih kombinacija izvornih klauzula i faktor-klauzula

3 (P) Razmotrite sljedeće dvije premise:

$$\forall x \text{KNOWS}(x, \text{Elizabeth})$$

$$\forall x (\text{KNOWS}(\text{John}, x) \rightarrow \text{HATES}(\text{John}, x))$$

Rezolucijom želimo izvesti zaključak da, budući da svi poznaju Elizabetu, a John mrzi sve koje poznaje, to John također mrzi i Elizabetu. **Što ovaj konkretan primjer ilustrira?**

- Da izravna rezolucija bez standardizacije ne može dokazati sve što i izravna rezolucija sa standardizacijom
- Da rezolucija opovrgavanjem gubi potpunost ako ne provodimo standardizaciju
- Da rezolucija opovrgavanjem gubi ispravnost ako dozvolimo da se varijabla supstituira izrazom koji ju koristi
- Da je izravna rezolucija nepotpuna, ali da je rezolucija opovrgavanjem potpuna

4 (P) Algoritam MGU primjenjujemo na par atoma $P(f(a, x), x, g(y), f(z, a))$ i $P(y, g(z), w, f(b, a))$. **Koju supstituciju vraća algoritam MGU?**

- A $\{g(b)/y, f(a, g(a))/x, b/z, x/w\}$
- B $\{g(f(a, b))/x, f(a, g(a))/x, b/z, g(g(b))/w\}$
- C $\{g(b)/x, f(a, g(b))/y, b/z, g(f(a, g(b)))/w\}$
- D Algoritam vraća pogrešku

5 (P) Dane su klauzule $\{P(g(y), x), \neg Q(x, b)\}$ i $\{Q(f(x), y)\}$. **Što je njihova rezolventa?**

- A $P(g(y), b)$
- B $P(g(y), x)$
- C $P(g(y), y)$
- D $P(g(y), f(z))$

6 (P) Neka $G(x)$ označava “ x je grad”, neka $U(x, y)$ označava “ x je u y ” te neka $P(x)$ označava “ x je pošta”. Napišite rečenicu “ U svakom gradu postoji pošta” kao FOL formulu. **Kako glasi klauzalni oblik te formule?**

- A $G(x'') \wedge P(f(x')) \wedge U(f(x), x)$
- B $\neg G(x) \vee \neg P(f(x)) \vee U(f(x), x)$
- C $(\neg G(x) \vee P(y)) \wedge (\neg G(x') \vee U(y', x'))$
- D $(\neg G(x) \vee U(f(x), x)) \wedge (\neg G(x') \vee P(f(x')))$

7 (R) Zadane su sljedeće dvije premise:

$$\exists x \left(\exists y R(y) \vee \forall z \neg Q(x, z) \right) \rightarrow P(a), \exists x \forall y \left(Q(x, y) \rightarrow R(y) \right)$$

Koju je od sljedećih formula moguće dokazati iz ovih premissa rezolucijom opovrgavanjem?

- A $\forall x Q(a, x)$
- B $\exists x P(x)$
- C $\exists x \forall y P(y)$
- D $\exists x P(x) \wedge P(b)$

8 (R) Zadane su tri premsise: (1) “*Ljubavnik je svaki onaj i samo onaj koji nekoga voli*”, (2) “*Svi vole ljubavnike*” i (3) “*Ana voli Ivana*”. Formalizirajte ove premsise sa $L(x)$ za “ x je ljubavnik” i $V(x, y)$ za “ x voli y ”. Zatim upotrijebite rezoluciju opovrgavanjem uz strategiju potpore kako biste dokazali cilj “*Svatko voli svakoga*”. **Je li cilj dokaziv i, ako jest, koji je najmanji broj rezolucijskih koraka potrebnih za dokaz?**

- A Nije dokaziv
- B Dokaziv u 5 koraka
- C Dokaziv u 4 koraka
- D Dokaziv u 3 koraka

Uvod u umjetnu inteligenciju

UNIZG FER, ak. god. 2021./2022.

Vježbe, v2

7 Logičko programiranje u Prologu

1 (T) Logičko programiranje u Prologu svodi se na definiciju programa kao niza definitnih klauzula. **Što je prednost, a što nedostatak toga što je program u Prologu sačinjen od definitnih klauzula, a ne od FOL formula?**

- A Prednost je što se dokazivanje može učinkovito izvesti ulančavanjem unazad, dok je nedostatak što neke formule FOL ne možemo iskazati kao definitne klauzule
- B Prednost je što je dokazivanje ispravno, dok je nedostatak što su definitne klauzule ograničene ekspresivnosti
- C Prednost je što je dokazivanje linearne vremenske složenosti, dok je nedostatak što postoji odstupanje između deklarativnog značenja definitnih klauzula i značenja kakvo bi te klauzule imale u FOL
- D Prednost je što je dokazivanje potpuno, dok je nedostatak što nije odlučivo, pa u nekim slučajevima postupak dokaza nikada ne završava

2 (T) Programi u Prologu sastoje se od definitnih klauzula, koje odgovaraju pravilima i činjenicama. **Što je razlika između pravila i činjenica u Prologu?**

- A Pravila imaju barem jedan negativan literal i točno jedan pozitivan literal, dok su činjenice jedinične klauzule sa samo jednim pozitivnim literalom
- B Pravila imaju najviše jedan negativan literal i barem jedan pozitivan literal, dok su činjenice jedinične klauzule sa samo jednim pozitivnim literalom
- C Pravila imaju antecedent s najviše jednim pozitivnim literalom, dok činjenice odgovaraju pravilima s uvijek lažnim antecedentom
- D Pravila imaju barem jedan negativan literal i točno jedan pozitivan literal, dok činjenice imaju jedan ili više pozitivnih literala

3 (P) Hornova logika koristi Hornove klauzule, koje su podskup formula FOL. Gradivni blok programa u Prologu je definitna klauzula, koja je vrsta Hornove klauzule. **Koja se od sljedećih formula može zapisati u obliku definitne klauzule ili konjunkcije više takvih klauzula?**

- A $\neg(P_1 \wedge P_2) \rightarrow Q$
- B $\neg P \rightarrow Q$
- C $P \rightarrow (Q_1 \vee Q_2)$
- D $(P_1 \vee P_2) \rightarrow Q$

4 (R) Baza znanja u Prologu modelira odnose između bioloških vrsta. Baza sadrži sljedeće činjenice i pravila:

```
podvrsta(sisavac, endoterm).  
podvrsta(šišmis, sisavac).  
podvrsta(ptica, endoterm).  
potomak(X, Y) :- podvrsta(X, Y).  
potomak(X, Y) :- podvrsta(X, Z), potomak(Z, Y).  
leti(X) :- potomak(X, ptica).
```

Nad ovako definiranom bazom znanja izvodimo upit leti(šišmis). Prolog će za ovaj upit nažalost vratiti False. **Koliko čvorova ima Prologovo stablo dokaza za ovaj upit?**

- A 8
- B 6
- C 12
- D 10

8 Ekspertni sustavi

5 (T) Prolog i CLIPS koriste pravila za izvođenje novog znanja. Međutim, ovi se sustavi međusobno razlikuju. **Koji je jedan od aspekata po kojem se Prolog i CLIPS razlikuju?**

- A CLIPS koristi ulančavanje unaprijed, dok Prolog koristi ulančavanje unazad
- B Prolog koristi varijable, dok ih CLIPS ne koristi
- C CLIPS omogućava da se definiraju činjenice, dok Prolog to ne omogućava
- D I Prolog i CLIPS koriste ulančavanje unazad, no CLIPS ne koristi rezolucijsko pravilo

6 (P) CLIPS je ljska ekspertnog sustava u kojoj se mogu izvoditi programi temeljeni na pravilima. Razmotrite sljedeći program u CLIPS-u, sastavljen od dvije činjenice i dva pravila. **Nakon što pokrenemo ovaj program, što će biti ispisano na ekran?**

```
(assert (a 2))
(assert (b 2))
(defrule F (declare (salience 20))
    ?x1 <-(a ?v1) ?x2 <-(b ?v2) (test (< ?v1 200))
    => (retract ?x1) (retract ?x2) (assert (a ?v2)) (assert (b (+ ?v1 ?v2))))
(defrule output (a ?v) => (printout t ?v))
```

- A 178
- B 288
- C 202
- D 198

7 (R) Baza znanja ekspertnog sustava sadržava sljedeća pravila:

- | | |
|--|--|
| (1) AKO $A = a_1 \wedge B = b_2$ ONDA $C = c_2$ | (5) AKO $F = f_3$ ONDA $E = e_2$ |
| (2) AKO $F = f_3 \wedge B = b_2$ ONDA $C = c_1$ | (6) AKO $D = d_3 \vee G = g_1$ ONDA $A = a_2 \wedge B = b_2$ |
| (3) AKO $E = e_1 \vee (D = d_1 \wedge E = e_2)$ ONDA $A = a_1$ | (7) AKO $A = a_1$ ONDA $G = g_1 \wedge E = e_1$ |
| (4) AKO $F = f_3 \vee Q = q_2$ ONDA $D = d_1$ | |

Sustav koristimo za izvođenje vrijednosti varijable C ulančavanjem unazad. U slučaju konflikta između pravila, sustav izabire pravilo s najmanjim rednim brojem. Na možebitne upite od strane sustava, korisnik će odgovoriti sa $D = d_2$ i $F = f_3$. **Što radi ekspertni sustav pri izvođenju vrijednosti varijable C ?**

- A Izvodi $E = e_1$ te kasnije izvodi $E = e_2$
- B Odbacuje pravilo (2) te pali pravilo (1)
- C Završava sa 6 činjenica u radnoj memoriji
- D Pali 6 pravila i izvodi $C = c_1$

Uvod u umjetnu inteligenciju

UNIZG FER, ak. god. 2021./2022.

Vježbe, v2

9 Modeliranje neizvjesnosti

- 1 (T) Bayesovo pravilo omogućava nam da na temelju opaženog dokaza E zaključimo o vjerojatnosti hipoteze H . Iz perspektive logike, takvo zaključivanje odgovara pravilu abdukcije. **Kojem dijelu pravila abdukcije odgovara uvjetna vjerojatnost $P(E|H)$ iz Bayesovog pravila?**
- [A] Implikaciji $E \rightarrow H$ [B] Činjenici E [C] Implikaciji $H \rightarrow E$ [D] Činjenici H
- 2 (R) Čudesni stroj profesora Baltazara može raditi u jednome od tri režima rada: "štedno" (S), "umjereni" (U) te "efikasno" (E). Stroj ima dvije lampice: zelenu (Z) i crvenu (C). Vjerojatnost da stroj radi u štednom režimu je $P(S) = 0.2$ a vjerojatnost da radi u umjernom režimu $P(U) = 0.7$. Detaljnim višednevnim promatranjem čudesnoga stroja došlo se i do sljedećih vjerojatnosti da su žaruljice upaljene: $P(Z|S) = 1$, $P(Z|U) = 0.3$, $P(Z|E) = 0.2$, $P(C|S) = 0.1$, $P(C|U) = 0.4$, $P(C|E) = 0.8$. Zanimaju nas dvije stvari: (1) vjerojatnost da je stroj uz upaljenu crvenu žaruljicu u umjerenom režimu rada te (2) vjerojatnost da je stroj uz obje upaljene žaruljice u efikasnom režimu rada. **Koliko iznose ove dvije vjerojatnosti?**
- [A] 0.052, 0.7 [B] 0.047, 0.7 [C] 0.211, 0.167 [D] 0.737, 0.133
- 3 (P) Za zaključivanje u domeni pedijatrije koristimo Bayesovo pravilo. Znamo da je vjerojatnost nastupanja osipa ako dijete ima šarlah deset puta veća nego vjerojatnost nastupanja osipa ako dijete nema šarlah. Također znamo da je vjerojatnost šarlaha ako dijete ima osip barem dva puta veća nego vjerojatnost šarlaha prije opažanja osipa. **Kolika je vjerojatnost P šarlaha prije opažanja osipa?**
- [A] $1/2 \leq P \leq 2/3$ [B] $P \leq 4/9$ [C] $P \geq 1/6$ [D] $P = 4/5$
- 4 (T) Neizrazita logika temelji se na teoriji neizrazitih skupova. **Koja je točno veza između neizrazite logike i neizrazitih skupova?**
- [A] Disjunkcija neizrazitih skupova P i Q jednaka je neizrazitom skupu sa $\mu(x) = \max(\mu_P(x), \mu_Q(x))$
[B] Stupanj istinitosti atoma $P(x)$ jednak je $\mu_P(x)$, tj. stupnju pripadnosti elementa x neizrazitom skupu P
[C] Atom $P(x)$ istinit je za one i samo one elemente za koje $\mu(x) \geq 0.5$
[D] Vrijednost $\mu(x)$ je donja ograda vjerojatnosti da je atom $P(x)$ istinit
- 5 (P) Primjenom standardnih (Zadehovih) neizrazitih operacija i jezičnih modifikatora konstruirali smo neizraziti skup *vrlo jak klokan*. Klokan Roger, najjači australski klokan, koji je od starosti preminuo 2018. godine, tom skupu pripada sa $\mu = 0.9$. Primjenom istih operatora i modifikatora konstruirali smo neizraziti skup *ne jak klokan*. **Koja je pripadnost klokana Rogera tom neizrazitom skupu?**
- [A] $1 - \sqrt{0.9}$ [B] $1 - 0.9^2$ [C] $(1 - 0.9)^2$ [D] $\sqrt{1 - 0.9^2}$
- 6 (R) Nad univerzalnim skupom $\{a, b, c, d\}$ zadani su neizraziti skupovi $X = \{0.1/a + 0/b + 0.3/c + 1/d\}$, $Y = \{0.5/a + 0.4/b + 0/c + 0.2/d\}$ i $Z = \{1/a + 0/b + 0/c + 0.5/d\}$. Koristeći Zadehove operatore

izvodimo neizraziti skup X tako da on odgovara jezičnome izrazu “*Ne X ili (Y i Z)*”. **Kako glasi neizraziti skup X ?**

- A $\{0.5/a + 0/b + 0.3/c + 1/d\}$
- B $\{0/a + 0.4/b + 0.3/c + 0.5/d\}$
- C $\{0.9/a + 1.0/b + 0.7/c + 0.2/d\}$
- D $\{0.5/a + 0/b + 0/c + 0/d\}$

- 7 (R) Razvijamo sustav neizrazitog zaključivanja za modeliranje dinamike fluida. Razmatramo odnos tlaka, temperature i volumena fluida. Definirali smo univerzalne skupove $P = \{100, 200, 300, 400, 500\}$ za tlak (u Pascalima), $T = \{-100, -50, 0, 50, 100\}$ za temperaturu (u Celzijima) i $V = \{0, 5, 10, 15, 20, 25\}$ za volumen (kubni metar po molu). Nad tim smo skupovima definirali neizrazite skupove *visok tlak* kao $P_v = \{0.1/200, 0.3/300, 0.6/400, 1/500\}$, skup *visoka temperatura* kao $T_v = \{0.2/0, 0.5/50, 0.8/100\}$, i skup *malen volumen* kao $V_m = \{1/5, 0.5/10, 0.1/15\}$. Definirali smo i dva pravila (implikacije), $P_v \rightarrow V_m$ i $T_v \rightarrow P_v$, modelirana kao neizrazite relacije. Zanima nas koliko je malen volumen fluida ako je temperatura fluida vrlo visoka, tj. želimo izvesti neizraziti skup V'_m ako je premla neizraziti skup $T'_v = \text{vrlo}(T_v)$. To možemo izračunati primjenom Zadehovog modifikatora intenzifikacije te uzastopnom primjenom generaliziranog modusa ponensa. **Kako glasi neizraziti skup V'_m koji dobivamo takvim neizrazitim zaključivanjem?**

- A $\{0.6/5, 0.5/10, 0.3/15\}$
- B $\{0.64/5, 0.5/10, 0.1/15\}$
- C $\{0.6/5, 0.3/10, 0.1/15\}$
- D $\{0.64/5, 0.5/10, 0.3/15\}$

Uvod u umjetnu inteligenciju

UNIZG FER, ak. god. 2021./2022.

Vježbe, v2

10 Strojno učenje

- 1 (T) Razvijamo model strojnog učenja za predviđanje broja gledatelja na kinoprojekcijama. U obzir smo uzeli tri značajke: dan u tjednu, žanr filma i cijena produkcije filma. **Koji bi algoritam strojnog učenja bilo prikladno upotrijebiti za ovaj problem, i zašto?**
- A Naivan Bayesov klasifikator, jer predviđamo diskretne vrijednosti (cijeli brojevi)
 - B Neuronsku mrežu, jer predviđamo brojčanu vrijednost
 - C Stablo odluke, jer imamo tri značajke s podjednakom informacijskom dobiti
 - D Neuronsku mrežu, jer je cijena produkcije filma brojčana značajka
- 2 (T) Naivan Bayesov klasifikator nazivamo "najnivim" jer model pretpostavlja uvjetnu nezavisnost značajki x_j unutar klase y . Uz tu pretpostavku, izglednost klase $P(x_1, \dots, x_n | y)$ možemo zamjeniti umnoškom $\prod_{j=1}^n P(x_j | y)$. **Koja je motivacija za uvođenje pretpostavke uvjetne nezavisnosti?**
- A Veća točnost modela na skupu za učenje
 - B Mogućnost korištenja značajki koje nisu binarne
 - C Mogućnost generalizacije na neviđene primjerke
 - D Sprječavanje podljeva pri množenju vjerojatnosti
- 3 (R) Mali je Ivica svakog svakog ljeta u zadnjih sedam godina naučio jedan novi programski jezik. Svoja vrijedna iskustva sažeo je u listu "*Programski jezik koji mi se sviđa*", gdje je svaki jezik opisao četirima značajkama, te je naznačio je li mu se dotični jezik svidio ($y = 1$) ili nije ($y = 0$). Ta lista izgleda ovako:

i	Evaluacija	Izvođenje	Paradigma	Provjera tipova	y
1	lijena	kompajler	imperativna	statička	0
2	striktna	interpreter	deklarativna	dinamička	0
3	lijena	kompajler	imperativna	dinamička	0
4	lijena	interpreter	hibridna	statička	0
5	striktna	interpreter	imperativna	statička	1
6	lijena	kompajler	hibridna	dinamička	1
7	striktna	kompajler	hibridna	dinamička	1

Ovog ljeta Mali Ivica želi puno jesti i spavati te opet naučiti novi programski jezik. U užem je izboru jezik \mathbf{x} sa sljedećim karakteristikama: $\mathbf{x} = (\text{striktna}, \text{interpreter}, \text{hibridna}, \text{dinamička})$. Međutim, ovog puta Mali bi Ivica volio unaprijed znati hoće li mu se dotični programski jezik svidjeti, tako da ne gubi cijelo ljeto bezveze. Pomozite Malom Ivici te na gornji skup primjera primjenite naivan Bayesov klasifikator s Laplaceovim zaglađivanjem "dodaj jedan". **Koliko iznosi vjerojatnost da bi se Malom Ivici programski jezik \mathbf{x} svidio?**

- A 0.856
- B 0.431
- C 0.799
- D 0.694

4 (P) Gradimo naivan Bayesov klasifikator za klasifikaciju poruka Twittera (tvitova) prema sentimentu. Svaki tvit želimo klasificirati u jednu od tri klase: pozitivan, negativan ili neutralan. Svaki tvit sastoji se od najviše 280 riječi i prikazujemo ga kao jedan binaran vektor značajki. Vektor se sastoji od 5000 značajki, i svaka značajka odgovara jednoj od 5000 riječi iz našeg vokabulara. Ako je značajka postavljena na 1, to znači da se dotična riječ pojavila u tvitu, inače je značajka postavljena na 0. Npr., ako je $x_{42} = 1$, onda to znači da se 42. riječ iz našeg vokabulara pojavila u tvitu. Učenje ovog modela svodi se na procjenu apriornih vjerojatnosti i izglednosti klase na temelju označenog skupa podataka. **Koliko ćemo ukupno vjerojatnosti trebati procijeniti za ovaj model?**

- A 1683 B 30003 C 569 D 31683

5 (T) Stabla odluke i Bayesov klasifikator dva su algoritma nadziranog učenja. Međutim, ti se algoritmi vrlo razlikuju. **Što je prednost, a što nedostatak stabla odluke u odnosu na Bayesov klasifikator?**

- A Prednost je što možemo bolje objasniti zašto je primjer klasificiran u neku klasu, a nedostatak što nemamo vjerojatnost klasifikacijske odluke
- B Prednost je što primjere možemo klasificirati u više od jedne klase, a nedostatak što se stablo odluke može lako prenaučiti
- C Prednost je što stablo odluke možemo podrezati kako bismo spriječili prenaučenost, a nedostatak što može doći do podljeva pri računanju vjerojatnosti
- D Prednost je što su stabla odluke otporna na male promjene u ulaznom skupu podataka, a nedostatak što pretpostavljamo uvjetnu nezavisnost značajki

6 (R) Raspolažemo skupom primjera za "Nezaboravno jadransko ljeto 2025., odmah nakon pandemije koronavirusa". Skup se sastoji od sljedećih primjera, svaki sa 4 značajke (Mjesto, Otok, Smještaj, Prijevoz) i ciljnom oznakom y :

i	Mjesto	Otok	Smještaj	Prijevoz	y
1	Istra	ne	privatni	auto	da
2	Istra	ne	privatni	avion	da
3	Dalmacija	da	hotel	auto	da
4	Dalmacija	da	hotel	bus	da
5	Kvarner	ne	kamp	bus	ne
6	Dalmacija	da	privatni	avion	ne
7	Istra	ne	kamp	auto	ne

Primijenite na ovaj skup primjera algoritam ID3. U slučaju da je u nekom koraku više značajki ima jednaku vrijednost informacijske dobiti, izaberite onu koja je u tablici navedena prva (ona lijevija). **Kako izgleda dobiveno stablo odluke?**

- A Korijenski čvor stabla je Mjesto, a njegovo dijete je čvor Smještaj s informacijskom dobiti 0.918
- B Korijenski čvor stabla je Smještaj, a njegovo dijete je čvor Mjesto s informacijskom dobiti 0.918
- C Korijenski čvor stabla je Mjesto, a njegovo dijete je čvor Smještaj s informacijskom dobiti 0.251
- D Korijenski čvor stabla je Smještaj, a njegovo dijete je čvor Mjesto s informacijskom dobiti 0.251

7 (P) Treniramo model stabla odluke na skupu podataka u kojem, nažalost, ima i nešto šuma. Svjesni smo da prisustvo šuma može dovesti do prenaučenosti modela, pa smo odlučili primjeniti unakrsnu provjeru da bismo podrezali stablo odluke. Skup označenih primjera podijelili smo na skup za učenje D_u i skup za provjeru D_p , tako da $D_u \cap D_p = \emptyset$. Sada isprobavamo nekoliko različitih dubina stabla, od $d = 1$ do $d = 42$. Ispitivanjem pogreške tih stabala različite dubine,

zaključili smo da je optimalna dubina stabla $d = 17$. **Što to konkretno znači?**

- A Da je pogreška na skupu D_p za stablo sa $d = 17$ veća od pogreške za isto to stablo na skupu D_u , ali veća od pogreške za bilo koju drugu dubinu stabla d na skupu D_p
- B Da je pogreška na skupu D_p za stablo sa $d = 17$ manja od pogreške na D_p za $d < 17$ i $d > 17$, ali očekivano ta je pogreška na D_p veća od pogreške na skupu D_u
- C Da stablo za $d = 17$ ostvaruje najmanju pogrešku na skupu D_u , dok na skupu D_p stablo ostvaruje uvijek veću pogrešku, s maksimumom pogreške za $d = 1$
- D Da su pogreške za stabla sa $d = 16$ i $d = 18$ veće i na skupu D_u i na skupu D_p , s time da su na skupu D_p očekivano veće nego na skupu D_u

Uvod u umjetnu inteligenciju

UNIZG FER, ak. god. 2021./2022.

Vježbe, v2

11 Umjetne neuronske mreže

1 (T) McCulloch-Pittsov model umjetnog neurona nastao je davne 1943. godine. **Što od sljedećega nije točno za McCulloch-Pittsov model?**

- A Tijelo neurona s ulaznim dendritima modelirano je težinskom sumom
- B Brzina širenja električnih impulsa modelirana je težinama
- C Uz fiksirane težine, neuron će za isti ulaz uvijek dati isti izlaz
- D Funkcioniranje aksonskog vlakna modelirano je prijenosnom funkcijom

2 (T) Razmatramo Rosenblattovo pravilo učenja perceptronu. Neka je t ciljna vrijednost neurona, a o izlazna vrijednost neurona). **Kako glasi izlaz za korekciju težina?**

- A $w(i+1) = w(i) + \eta(t+o)x(i)$
- B $w(i+1) = w(i) - \eta(t-o)x(i)$
- C $w(i+1) = w(i) - \eta(t+o)x(i)$
- D $w(i+1) = w(i) + \eta(t-o)x(i)$

3 (P) Na računalu implementiramo unaprijednu potpuno povezanu slojevitu umjetnu neuronsku mrežu arhitekture $3 \times 20 \times 10 \times 5 \times 2$. Neuroni kao prijenosne funkcije koriste zglobnicu. U memoriji težine mreže čuvaju se kao tip `double` koji zauzima 8 okteta. **Koliki je ukupni utrošak memorije za parametre ove mreže?**

- A 5096
- B 2856
- C 2560
- D 4218

4 (R) Skup primjera za učenje $\{(x_2, x_1, y)\}$ je sljedeći:

$$\{(1, 1, -1), (2, 4, 1), (1, 2, -1), (3, 3, 1), (2, 1, -1), (4, 2, 1)\}$$

Učenje se provodi uporabom perceptronu TLU s izlaznim vrijednostima -1 i 1 te stopom učenja $\eta = 1$. Početne vrijednosti težinskih faktora su $(w_2, w_1, w_0) = (1.3, 1.2, -3.2)$. Provedite postupak učenja Rosenblattovim algoritmom. **Koliko se puta tijekom učenja provode korekcije težina te koje su njihove konačne vrijednosti?**

- A 2 puta, $(3.3, 2.8, -10)$
- B 3 puta, $(1.3, 1.2, -5.2)$
- C 4 puta, $(1.3, -2.5, 12)$
- D Postupak ne konvergira

5 (P) Raspolažemo dvama skupovima primjeraka za učenje (iste su dimenzionalnosti ulaza) za binarni klasifikacijski problem. Kao klasifikator koristimo TLU-perceptron. Poznato je da nad oba skupa Rosenblattov algoritam uspješno dolazi do rješenja. Neka mu nad prvim skupom za to treba n_1 koraka, a nad drugim skupom n_2 koraka. Razmotrite sada treći skup primjeraka za učenje koji

predstavlja uniju prethodnih dvaju. **Koliko će koraka trebati Rosenblattovu algoritmu da nauči ispravno klasificirati taj skup?**

- A Nemamo garantije da će učenje biti uspješno
- B Minimalno n_1 koraka
- C Između n_1 i n_2 koraka
- D $\max(n_1, n_2)$ koraka

6 (P) Razmatramo četiri Booleove funkcije od dviye varijable: $f_1(A, B) = \bar{A} \cdot \bar{B} + \bar{A} \cdot B$, $f_2(A, B) = \bar{A} \cdot \bar{B} + A \cdot B$, $f_3(A, B) = \bar{A} \cdot B + A \cdot \bar{B} + A \cdot B$, $f_4(A, B) = (\bar{A} + \bar{B}) \cdot (A + B)$. Logičke vrijednosti varijabli A i B te vrijednost Booleove funkcije kodiramo brojevima 0 i 1 (ovisno o logičkoj vrijednosti). Zatim navedene funkcije pokušavamo naučiti prikladnim TLU-perceptronom. **Koje će funkcije TLU-perceptron uspjeti naučiti?**

- A $f_1(A, B)$ i $f_4(A, B)$
- B $f_1(A, B)$ i $f_3(A, B)$
- C $f_2(A, B)$ i $f_4(A, B)$
- D Sve ih može naučiti

7 (R) Unaprijednu potpuno povezanu slojevitu neuronsku mrežu arhitekture $3 \times 2 \times 2$ sa sigmoidnim prijenosnim funkcijama učimo preslikavanje $R^3 \rightarrow R^2$, odnosno skup primjeraka za učenje sadrži zapise oblika $(x_1, x_2, x_3) \mapsto (y_1, y_2)$. Trenutačne vrijednosti težina su:

$$w_{0,1}^{(1)} = -1, w_{1,1}^{(1)} = 0.1, w_{2,1}^{(1)} = 1, w_{3,1}^{(1)} = 1, w_{0,2}^{(1)} = 0.5, w_{1,2}^{(1)} = 0.4, w_{2,2}^{(1)} = -2, w_{3,2}^{(1)} = 0.8,$$
$$w_{0,1}^{(2)} = -0.4, w_{1,1}^{(2)} = -2, w_{2,1}^{(2)} = 1, w_{0,2}^{(2)} = 0.4, w_{1,2}^{(2)} = 1, w_{2,2}^{(2)} = 0.3.$$

Primjerak koji trenutačno razmatramo je $(0.2, -0.1, 0.2) \mapsto (1, 0)$. Učenje mreže provodi se postupkom propagacije pogreške unazad na temelju pojedinačnih primjeraka. Neka je iznos stope učenja jednak 10. Provedite postupak učenja za dani primjerak. **Koliko iznosi zbroj $w_{1,2}^{(1)} + w_{3,1}^{(1)}$ nakon provedenih korekcija?** (Odgovori su zaokruženi na četiri decimale.)

- A 1.3521
- B 1.2627
- C 1.4752
- D 1.3137

Uvod u umjetnu inteligenciju

UNIZG FER, ak. god. 2021./2022.

Vježbe, v1

12 Prirodnom inspirirani optimizacijski algoritmi

1 (T) Problemi zadovoljavanja ograničenja (engl. *constraint satisfaction problem*, CSP) podskupina su problema pretraživanja prostora stanja. **Što je karakteristično za ovu vrstu problema?**

- A Mogu se rješavati algoritmom A*, ali samo uz optimističnu heuristiku i skup posjećenih stanja
- B Imamo garanciju da traženo rješenje postoji (no problem je pronaći ga)
- C Ispitni predikat definiramo tako da uspoređuje stanje sa zadanim predloškom (kao kod slagalice)
- D Bitno nam je samo konačno stanje

2 (T) Za genetski algoritam koristimo proporcionalnu selekciju (engl. *roulette wheel selection*). **Što od sljedećeg nužno mora biti zadovoljeno?**

- A Iznosi funkcije dobrote moraju biti ograničeni na interval $[0, 1]$
- B Ne smije biti jedinki s negativnim iznosima funkcije dobrote
- C Suma iznosa funkcija dobota svih jedinki mora iznositi 1
- D Mora se koristiti binarna reprezentacija rješenja

3 (P) Genetskim algoritmom tražimo maksimum funkcije $f(x, y, z)$, koristeći binarnu reprezentaciju rješenja. Vrijednost svake od varijabli pretražuje se u intervalu $[-10, 15]$, pri čemu je potrebno osigurati da se to pretraživanje provodi barem s preciznošću 0.01. **Od koliko se minimalno bitova treba sastojati kromosom?**

- A 34
- B 36
- C 11
- D 33

4 (R) Minimum funkcije

$$f(x, y, z) = (x - 5)^2 + (y + 2)^2 + (z + 3)^2 + xy$$

pronalazi se genetskim algoritmom ostvarenim u obliku trotournirske selekcije. Svaka varijabla je pri tome ostvarena s 2 bita te se pretražuje cjelobrojno područje: x iz intervala $[0, 3]$, a y i z iz intervala $[-1, 2]$. U kromosomu najprije dolaze bitovi od x , pa od y , pa od z . U jednom koraku odabrana su tri kromosoma:

$$K1 = 000101, K2 = 001011, K3 = 100001$$

Kao funkcija dobrote koristi se negirana funkcija f , dakle dobrota od (x, y, z) je jednaka $-f(x, y, z)$. Koristi se križanje s jednom točkom prijeloma na polovici kromosoma. **Izračunajte dobrotu djeteta koje će biti vraćeno u populaciju.** Pretpostavite da prilikom križanja mutacija uvijek promjeni zadnji bit kromosoma (onaj najdesniji). Ako operator križanja generira više djece, u populaciju će se vratiti najbolje od generirane djece.

- A -30
- B -17
- C -25
- D -19

5 (R) Uporabom mravlje kolonije traži se ciklus kroz graf. Poznati su sljedeći podatci: $\tau_{1,2} = \frac{1}{\sqrt{3}}$, $\tau_{1,4} = 2$, $\tau_{1,6} = 0.5$, $\tau_{2,4} = \frac{1}{2}$, $\tau_{2,5} = \frac{1}{3}$, $\tau_{2,7} = 2$, $\tau_{3,5} = 1$, $\tau_{3,6} = 3$, $\tau_{3,7} = 10$, $\tau_{4,6} = 0.5$, $\tau_{5,7} = \sqrt{3}$, $\tau_{6,7} = 10\sqrt{2}$, $\eta_{1,2} = 3$, $\eta_{1,4} = \frac{1}{\sqrt{2}}$, $\eta_{1,6} = 2$, $\eta_{2,4} = 2\sqrt{2}$, $\eta_{2,5} = 3\sqrt{3}$, $\eta_{2,7} = 0.5$, $\eta_{3,5} = \sqrt{2}$, $\eta_{3,6} = 3\sqrt{3}$, $\eta_{3,7} = \frac{1}{2\sqrt{5}}$, $\eta_{4,6} = 0.5$, $\eta_{5,7} = 1$, $\eta_{6,7} = 0.1$. Također, sve dane vrijednosti su simetrične, tj. $\tau_{i,j} = \tau_{j,i}$ i $\eta_{i,j} = \eta_{j,i}$. Dodatno, $\alpha = 2$ i $\beta = 2$. Prvi mrav kreće iz čvora 1. Kada mrav treba donijeti vjerojatnosnu odluku, pretpostavite da će ishod slučajnog odabira odgovarati najvjerojatnijem. Ako iz nekog čvora mrav ne može dalje, konstrukcija ciklusa se prekida. Uz te pretpostavke odredite ciklus koji će taj mrav konstruirati. **Slijed od koja tri čvora je dio tog ciklusa?**

- A 5, 3, 6 B 6, 4, 1 C 2, 7, 5 D Mrav neće uspjeti konstruirati ciklus

6 (P) Ako kod algoritma Ant System postavimo τ_0 na vrijednost koja je puno manja od količine feromonskih tragova koju deponira jedan mrav, **što će biti posljedica?**

- A Algoritam će dugo vremena istraživati nasumične puteve, prije no što se mravi na neke puteve
 B Algoritam će rapidno konvergirati prema globalnom optimumu
 C Algoritam će izgubiti mogućnost istraživanja kvalitetnih rješenja
 D Isparavanje feromonskih tragova će biti prevveliko

7 (P) Algoritmom kolonije mrava rješavamo problem trgovačkog putnika nad odabranim gradovima u Republici Hrvatskoj. Za algoritam su poznati sljedeći parametri: $\tau_0 = 100$, $\alpha = 2$, $\beta = 3$, $\rho = 0.1$, kolonija se sastoji od 100 mrava te ažuriranje u svakoj epohi obavlja samo mrav koji je u toj epohi pronašao najbolje rješenje. Razmotrimo brid koji povezuje gradove Zagreb i Dubrovnik. Pretpostavimo da niti u jednoj epohi ta dva grada nisu sljedbenici u pronađenom putu. **Nakon koliko će epoha algoritma količina feromonskih tragova na spomenutom bridu postati manja od 1?**

- A 2 B 21 C 44 D 53