Međuispit iz Uvoda u umjetnu inteligenciju (2020./2021.)

– NEKORIGIRANA VERZIJA –

*Ispit se sastoji od **20 pitanja** i ukupno nosi **20 bodova**. Točan odgovor nosi 1 bod a netočan -1/3 boda. Trajanje ispita je **150 minuta**. Primjerak ispita trebate predati zajedno sa svojim rješenjima.*

1. Pretraživanje prostora stanja (7 pitanja)

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 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

B 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

C Kod algoritma minimax heuristika se ne primjenjuje na sve čvorove u stablu igre već samo na čvorove na određenoj dubini, dok se kod algoritama pretraživanja heuristika primjenjuje na sve generirane čvorove

D 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

2 (R) Funkcijom *succ* definirani su prijelazi između stanja $\{a, b, c, d, e, f\}$ na sljedeći način: $succ(a) = \{b, c\}$, $succ(b) = \{c, d\}$, $succ(c) = \{d, e\}$, $succ(d) = \{a, e\}$, $succ(e) = \{f\}$, $succ(f) = \{d\}$. Početno stanje je a, a ciljno f. Pretpostavite leksikografski poredak između čvorova. **Koji će algoritam čvorove ispitivati redoslijedom $\ldots, c, c, d, d, e, \ldots$?**

A Iterativno pretraživanje u dubinu
B Pretraživanje u dubinu
C Ograničeno pretraživanje u dubinu ($k = 3$)
D Pretraživanje u širinu

3 (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 Jelačića (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 dodatnih 4 minuta. Za nalaženje puta do Trga Sv. Marka koristimo algoritam A^* s nekom heuristikom h, koja je konzistentna. Pritom vrijedi $h(s_1) = 13$, $h(s_2) = 14$, $h(s_3) = 11$, $h(s_4) = 10$. 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 $h(s_1) = 15$
B $h(s_1) = 10$
C $h(s_1) = 13$
D $h(s_1) = 14$

4 (T) Prostorna složenost algoritma pretraživanja u dubinu je linearna, točnije $\mathcal{O}(bm)$, gdje je b faktor grananja, a m je maksimalna dubina stabla. S druge strane, vremenska složenost istog tog algoritma je eksponencijalna, točnije $\mathcal{O}(b^m)$. **Zašto vremenska složenost algoritma pretraživanja u dubinu također nije linearna?**

A Vremenska složenost samo je u najgorem slučaju eksponencijalna, ali očekivano je linearna, jer algoritam pri proširenju čvorova slijedi uvijek samo prvoproširei čvor

B Vremenska složenost ne može biti linearna jer je vremenska složenost uvijek veća od prostorne složenosti, budući da algoritam uvijek treba izvesti barem onoliko koraka koliko treba za izgradnju strukture podataka

C Vremenska složenost ne može biti linearna jer algoritam može zaglaviti u beskonačnoj petlji, ako prostor stanja ima usmjerene cikluse ili ako je skup stanja beskonačan

D Vremenska složenost je eksponencijalna jer će u najgorem slučaju algoritam morati ispitati sve čvorove stabla pretraživanja prije nego što ispita ciljni čvor, ako ga uopće dosegne

5 (T) Dio definicije problema pretraživanja prostora stanja jest funkcija sljedbenika, $succ : S \to \wp(S)$, koja definira prijelaze između stanja. Tu funkciju možemo definirati eksplicitno ili implicitno. **Koja je prednost implicitne definicije funkcije sljedbenika nad eksplicitnom definicijom?**

A Eksplicitna definicija u praksi nije moguća kada je broj stanja prevelik, i tada je lakše implicitno (algoritamski) definirati na koje se sve načine stanje može promijeniti kako bi se dobilo iduće stanje

B Implicitna definicija koristi skup operatora koji definiraju sljedbenička stanja, čime se osigurava potpunost algoritma pretraživanja, dok s eksplicitnom definicijom algoritam može zaglaviti u beskonačnoj petlji

C Implicitna definicija lako se može proširiti pokazivačem na roditeljski čvor, što je potrebno za rekonstrukciju rješenja, dok je kod eksplicitne definicije za to potrebno dodatno pohranjivati otvorene čvorove

D Eksplicitna 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

6 (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 od čvora sa stanjem s_0 do čvora sa stanjem s_x iznosi 16. Stanje s_x i ciljno stanje s_g su sljedeći:

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

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 $23, 23, 23$ D $22, 22, 20$

7 (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 modificirano tako da u svakoj iteraciji dubinsko ograničenje povećavamo za 3 umjesto za 1. Heuristika vraća dubinu pronađenog rješenja. Konačno, heuristiku h_3 izveli smo kao iterativno pretraživanje u dubinu, ali ograničeno na dubinu $k = 3$. I ta heuristika vraća dubinu na kojoj je pronađeno rješenje, ili k ako rješenje nije pronađeno. Želimo da algoritam A^* bude što obavještеniji, 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 najobavještеniji, ali još uvijek potpun i optimalan algoritam A^*?**

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

2. Igranje igara (3 pitanja)

8 (R) Stablo igre definirano je prijelazima $succ(A) = \{B, C\}, succ(B) = \{D, E\}, succ(D) = \{H, I\}, succ(E) = \{J, K, L\}$,$succ(C) = \{F, G\}$, $succ(F) = \{M, N, O\}$, $succ(G) = \{P, Q\}$. Heurističke vrijednosti listova su $h(H) = 10, h(I) = -h(O) = 3, h(J) = 13, h(K) = h(N) = -9, h(L) = -11, h(M) = 5, h(P) = 7, h(Q) = -16$. Optimalna strategija određuje se algoritmom minimax uz alfa-beta podrezivanje. **Koji će čvorovi pritom biti podrezani (preskočeni pri izračunu minimax vrijednosti)?**

A L, F, M, N, O B I, L C O, P, Q D K, L, G, P, Q

9 (P) U šahovskom turniru sudjeluju četiri igraća 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 prvome 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. Uzvrati 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 A_2 ili A_3 B A_2 ili A_4 C A_1 ili A_3 D A_3 ili A_4

10 (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., $succ(235) = \{335, 245, 236\}$. Međutim, stanja koja sadrže znamenku 9 su završna stanja i ona nemaju sljedbeničkih stanja, npr., $succ(932) = \emptyset$. U završnim stanjima, isplata za prvog igrača (MAX) jednaka je razlici prve i treće znamenke, npr., $utility(932) = 9 - 2 = 7$. 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 znamenci iz s, a heuristika h_2 algorithma A_2 (definirana iz perspektive tog algoritma) jednaka je trećoj znamenci 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 $175 \to 176 \to 186$ B $175 \to 275 \to 276$ C $175 \to 275 \to 375$ D $175 \to 176 \to 276$

3. Prikazivanje znanja, automatsko zaključivanje i logičko programiranje (10 pitanja)

11 (T) Postupak rezolucije prikladan 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?**

A 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

B 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

C 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

D 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

12 (R) Zadane su tri premise: *"Ljubavnik je svaki onaj i samo onaj koji nekoga voli"*, (2) *"Svi vole ljubavnike"* i (3) *"Ana voli Ivana"*. Formalizirajte ove premise 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 Dokaziv u 5 koraka B Nije dokaziv C Dokaziv u 4 koraka D Dokaziv u 3 koraka

13 (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 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

C 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 omguće samo uz potpunu strategiju

D 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 premisa

14 (T) Logičko pogramiranje 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 je dokazivanje ispravno, dok je nedostatak što su definitne klauzule ograničene ekspresivnosti

B 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

C Prednost je što se dokazivanje može učinkovito izvesti ulančavanjem unazad, dok je nedostatak što neke formule FOL ne možemo iskazati kao definitne klauzule

D Prednost je što je dokazivanje potpuno, dok je nedostatak što nije odlučivo, pa u nekim slučajevima postupak dokaza nikada ne završava

15 (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 $G(x) \land P(a) \land U(a, x)$

B $G(x'') \land P(f(x')) \land U(f(x), x)$

C $\neg G(x) \lor \neg P(f(x)) \lor U(f(x), x)$

D $\big(\neg G(x) \lor U(f(x), x)\big) \land \big(\neg G(x') \lor P(f(x'))\big)$

16 (P) Hornova logika koristi Hornove klauzule, koje su podskup formula FOL. **Koja se od sljedećih formula može zapisati u obliku definitne Hornove klauzule** (ili konjunkcije više takvih klauzula)?

A $P \to \neg Q$ B $\neg P \to \neg Q$ C $\neg(P_1 \lor P_2) \to Q$ D $\neg P \to Q$

17 (P) Dana je FOL interpretacija s domenom $D = \{a, b\}$ i ekstenzijom $P(a) \equiv \top$. **Ova interpretacija model je koje od sljedećih formula?**

A $\exists x P(x) \to \forall x P(x)$ B $\exists x \neg P(x)$ C $\forall x P(x) \lor \neg P(a)$ D $\forall x P(x)$

18 (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 Neispravnost D Nedeterminističnost

19 (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 10 B 6 C 12 D 8

20 (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 premisa?**

A *Ako je Ines sretna, onda je Arsen zove na telefon.*

B *Ako je Ines sretna, onda se ne javlja na telefon.*

C *Ines ima novaca.*

D *Ines je sretna ako je Arsen zove na telefon.*

Tocni odgovori:

	0 1	0 2	0 3	0 4	0 5	0 6	0 7	0 8	0 9	1 0	1 1	1 2	1 3	1 4	1 5	1 6	1 7	1 8	1 9	2 0
Grupa A	C	D	D	D	A	C	A	D	C	B	C	A	D	C	D	B	B	B	A	D
Grupa B	C	C	D	C	D	D	C	D	D	D	B	B	B	D	D	B	A	B	B	A
Grupa C	A	B	C	B	D	A	B	B	D	C	C	C	C	B	D	B	A	B	B	B
Grupa D	A	D	D	C	B	A	A	B	B	A	C	B	C	D	A	C	C	B	B	C
Grupa E	B	C	D	D	A	C	A	D	D	D	A	C	A	B	A	D	B	D	C	D
Grupa F	B	D	D	A	A	A	A	C	C	B	B	B	C	D	C	A	C	B	D	C

Permutacije zadataka (grupa A kao referentna grupa):

A.01: A.01
A.02: A.02
A.03: A.03
A.04: A.04
A.05: A.05
A.06: A.06
A.07: A.07
A.08: A.08
A.09: A.09
A.10: A.10
A.11: A.11
A.12: A.12
A.13: A.13
A.14: A.14
A.15: A.15
A.16: A.16
A.17: A.17
A.18: A.18
A.19: A.19
A.20: A.20

B.01: A.03
B.02: A.01
B.03: A.06
B.04: A.07
B.05: A.04
B.06: A.02
B.07: A.05
B.08: A.09
B.09: A.08
B.10: A.10
B.11: A.19
B.12: A.16
B.13: A.15
B.14: A.20
B.15: A.14
B.16: A.13
B.17: A.11
B.18: A.17
B.19: A.12
B.20: A.18

C.01: A.07
C.02: A.06
C.03: A.01
C.04: A.03
C.05: A.04
C.06: A.02
C.07: A.05
C.08: A.08
C.09: A.09
C.10: A.10
C.11: A.11
C.12: A.13
C.13: A.17
C.14: A.14
C.15: A.15
C.16: A.12

C.17: A.18
C.18: A.16
C.19: A.19
C.20: A.20

D.01: A.06
D.02: A.02
D.03: A.03
D.04: A.04
D.05: A.05
D.06: A.01
D.07: A.07
D.08: A.10
D.09: A.09
D.10: A.08
D.11: A.19
D.12: A.12
D.13: A.14
D.14: A.13
D.15: A.17
D.16: A.18
D.17: A.11
D.18: A.15
D.19: A.20
D.20: A.16

E.01: A.01
E.02: A.05
E.03: A.03
E.04: A.04
E.05: A.07
E.06: A.06
E.07: A.02
E.08: A.08
E.09: A.09
E.10: A.10
E.11: A.16
E.12: A.13
E.13: A.19
E.14: A.18
E.15: A.14
E.16: A.12
E.17: A.20
E.18: A.15
E.19: A.17
E.20: A.11

F.01: A.05
F.02: A.06
F.03: A.07
F.04: A.03
F.05: A.01
F.06: A.04
F.07: A.02
F.08: A.08
F.09: A.10
F.10: A.09
F.11: A.16
F.12: A.17
F.13: A.20
F.14: A.14
F.15: A.15
F.16: A.19
F.17: A.12
F.18: A.13
F.19: A.18
F.20: A.11