Međuispit iz Uvoda u umjetnu inteligenciju (2023./2024.)

– NEKORIGIRANA VERZIJA –

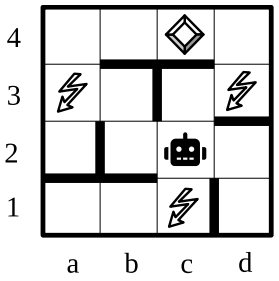
*Ispit se sastoji od **20 pitanja** i ukupno nosi **20 bodova**. Točan odgovor nosi 1 bod, a netočan -1/3 boda. Trajanje ispita je **150 minuta**. Primjerak ispita trebate predati zajedno sa svojim rješenjima.*

1. Pretraživanje prostora stanja i heurističko pretraživanje (7 pitanja)

1 (P) Neka je S skup stanja te neka su s_1, s_2 ,s_3 i s_4 stanja iz S. Vrijedi $succ(s_1) = \{(s_2, 4)\}$ i $succ(s_2) = \{(s_3, c)\}$, gdje je c neka cijena prijelaza. Razmatramo dvije heuristike, h_1 i h_2. Heuristika h_1 definirana je u odnosu na ciljno stanje s_3, a heuristika h_2 u odnosu na ciljno stanje s_4. Vrijedi $h_1(s_1) = 11$ i $h_1(s_2) = 9$ te $h_2(s_1) = 20$, $h_2(s_2) = 16$ i $h_2(s_3) = 8$. Obje su heuristike optimistične. Međutim, na temelju ovih informacija heuristiku h_2 možemo unaprijediti tako da bude obavještenija (bliža stvarnoj cijeni). **Koje od navedenih vrijednosti za h_2 sigurno znamo da daju obavještenију, ali još uvijek optimističnu heuristiku?**

A $h_2(s_1) = 20$, $h_2(s_2) = 17$
B $h_2(s_1) = 21$, $h_2(s_2) = 17$
C $h_2(s_1) = 20$, $h_2(s_2) = 18$
D $h_2(s_1) = 21$, $h_2(s_2) = 16$

2 (R) Na slici desno prikazana je mapa dvodimenzijskoga svijeta od 16 polja. Robot Robby početno se nalazi na polju $c2$ i treba doći na polje $c4$ na kojemu se nalazi dijamant. Robby se može kretati prema gore, dolje, lijevo i desno (npr., sa $c2$ se može kretati na $c3$, $c1$, $b2$ i $d2$), ali ne može prolaziti kroz zidove (npr., ne može proći sa $c3$ na $c4$). Robby želi pronaći put s najmanjim utroškom baterije. Svaki potez Robbyja košta jednu energetsku jedinicu iz baterije, osim kada zakorači na polja $a3$, $c1$ ili $d3$, gdje je zbog energetskog polja utrošak baterije jednak 3 energetske jedinice. Primijenite algoritam A^*. Stanje odgovara Robbyjevoj poziciji, a cijena puta odgovara broju utrošenih energetskih jedinica. Za heuristiku koristite L1-udaljenost (Manhattan-udaljenost). U slučaju izjednačenja cijene, proširuje se čvor sa stanjem koje alfanumerički dolazi prvo (npr., $a2$ prije $b2$ prije $b3$). **Koliko čvorova će algoritam A^* ukupno proširiti?**



A 9 B 8 C 10 D 7

3 (P) 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 $\ldots, b, c, a, b, c, \ldots$?**

A Pretraživanje u dubinu
B Iterativno pretraživanje u dubinu
C Pretraživanje u širinu
D Ograničeno pretraživanje u dubinu ($k = 2$)

4 (T) Jedno od poželjnih svojstava heurističke funkcije je *konzistentnost* (ili *monotonost*). Razmotrite algoritam A^*. **Koja je prednost algoritma A^* s konzistentnom heuristikom u odnosu na A^* s nekonzistentnom ali optimističnom heuristikom?**

A Algoritam je optimalan
B Algoritam je potpun
C Algoritam ispituje manji broj čvorova
D Algoritam ima manju prostornu složenost

5 (T) Jedna od komponenti problema pretraživanja prostora stanja je funkcija sljedbenika *succ*. Razmotrite algoritam pretraživanja s jednolikom cijenom (UCS). **Kod tog algoritma, u što funkcija *succ* preslikava svako stanje?**

A Skup svih podskupova stanja (moguće prazan)
B Skup parova sačinjenih od stanja i pozitivnog broja
C Kartezijev umnožak skupa stanja i skupa nenegativnih brojeva
D Skup parova sačinjenih od početnog i završnog stanja

6 (P) Rješavamo problem pretraživanja prostora stanja za koji vrijedi $m = 10$, $d = 7$ i $b = 3$. Koristimo pretraživanje u širinu (BFS) i pretraživanje u dubinu (DFS). Za oba algoritma razmotrite najgori slučaj vremenske složenosti. **Koliko će u takvom slučaju algoritam DFS generirati više čvorova od algoritma BFS?**

A 85293 B 78729 C 78696 D 85257

7 (R) Algoritmom slijepog pretraživanja rješavamo problem misionara i kanibala. Tri misionara i tri kanibala potrebno je jednim čamcem prevesti s jedne strane obale rijeke na drugu, pri čemu se ne smije dogoditi da se na jednoj strani nađu i kanibali i misionari, ali da je kanibala više nego misionara. Čamac može prevesti najviše dvije osobe i ne može ploviti prazan. Stanje prikazujemo kao trojku MKC, gdje su M i K broj misionara odnosno kanibala na lijevoj obali rijeke, a C je strana obale na kojoj se nalazi čamac. Npr., u stanju 32D na lijevoj obali su tri misionara i dva kanibala (jedan kanibal je onda na desnoj obali), a čamac je na desnoj obali. Početno stanje je 33L, a završno 00D. Funkcija sljedbenika generira sva legitimna stanja, uključivo i ona koja rezultiraju neuspjehom i nemaju sljedbenika, npr., stanje 23D. Koristimo algoritam iterativnog pretraživanja u dubinu (IDS) bez liste posjećenih stanja. Kod proširenja čvora, algoritam čvorove djecu generira prema numeričkom redoslijedu; npr., stanje 13D prethodi stanju 23D. **Koliko će čvorova algoritam IDS ispitati prije nego što proširi čvor sa stanjem** 31D?

A 11 B 18 C 16 D 14

2. Igranje igara (3 pitanja)

8 (P) Alfa-beta podrezivanje može značajno smanjiti broj generiranih čvorova kod algoritma minimax. Ušteda ovisi o redoslijedu kojim se generiraju čvorovi. Optimalan redoslijed je onaj koji djecu čvora MAX generira silazno po minimax vrijednostima, a djecu čvora MIN uzlazno po minimax vrijednostima. Razmotite igru s dva moguća poteza u svakome stanju. Stablo se pretražuje do dubine 3. Vrijednosti heuristike igrača MAX za stanja u listovima stabla igre, kada su čvorovi-djeca generirani nasumičnim redoslijedom, su sljedeća (slijeva nadesno): $0, 3, -3, 4, 4, -1, 0, 2$. Na potezu je igrač MAX. Odredite broj čvorova koji će biti uklonjeni podrezivanjem alfa-beta. Zatim ponovite izračun, ali s optimalnim redoslijedom generiranja čvorova. **Koliko više čvorova će biti podrezano s optimalnim redoslijedom u odnosu na slučajan redoslijed generiranja čvorova?**

A 4 B 2 C 1 D 3

9 (T) Igrači MAX i MIN primjenjuju algoritam minimax prije svakog svog poteza, pretražujući do neke unaprijed definirane dubine. Zamislite, međutim, da igrač MAX algoritam minimax primjenjuje samo jednom, i to na početku igre, a da zatim tako dobivene minimax vrijednosti koristi za sve svoje buduće poteze. **U kojem bi slučaju tako implementirani igrač MAX sigurno maksimizirao svoju minimalnu dobit?**

A Nema garancije da bi takav igrač maksimizirao svoju minimalnu dobit

B Ako je pri izračunu minimax-vrijednosti dosegao sva završna stanja igre

C Ako je heuristika igrača MAX obavještenija od heuristike igrača MIN

D Ako koristi identičnu heuristiku kao igrač MIN

10 (R) Razmatramo igru dvaju minimax-igrača. Stablo igre definirano je prijelazima $succ(A) = \{B, C, D\}$, $succ(B) = \{E, F\}$, $succ(C) = \{G, H\}$, $succ(D) = \{I, J\}$, $succ(F) = succ(G) = \{K, L\}$, $succ(E) = succ(H) = succ(I) = \{M, N\}$ te $succ(J) = \{O, P\}$. Igrač MAX koristi heuristiku h_1, a igrač MIN heuristiku h_2. Vrijednosti heuristika su sljedeće (obje heuristike definiraju dobit dotičnog igrača):

	E	F	G	H	I	J	K	L	M	N	O	P
h_1	1	-2	3	2	3	5	-1	0	-2	-1	0	2
h_2	0	5	2	4	-1	2	3	2	-3	-2	1	-4

Oba igrača pretražuju najviše dva poteza unaprijed (jedan svoj i jedan suparnički). Igru započinje igrač MAX u stanju A. **Kroz koja stanja će se razvijati igra?**

A $A, C, H, \ldots$ B $A, D, J, \ldots$ C $A, C, G, \ldots$ D $A, D, I, \ldots$

3. Prikazivanje znanja i automatsko zaključivanje (6 pitanja)

11 (T) Propozicijska logika (PL) i logika prvog reda (FOL) dva su formalizma za prikaz znanja. **Što je prednost, a što nedostatak PL u odnosu na FOL?** (Podsjetnik: Problem je *traktabilan* ako je rješiv u polinomijalnom vremenu.)

A Dokazivanje u PL je odlučivo, ali je PL manje ekspresivan od FOL

B Dokazivanje u PL je traktabilno, ali je FOL poluodlučiv

C PL je ekspresivniji od FOL, ali je dokazivanje netraktabilno

D Dokazivanje u PL je odlučivo, ali nije traktabilno kao u FOL

12 (T) Neka je R skup pravila zaključivanja te neka su F i G dobro oblikovane formule logike. **Kada za skup R kažemo da *nije* potpun skup pravila zaključivanja?**

A Ako je formula $F \to G$ zadovoljiva, ali ne postoji pravilo u R kojim iz $F \wedge \neg G$ možemo izvesti NIL

B Ako je formula $\neg F \vee G$ valjana, ali ne postoji lanac pravila u R kojim iz F možemo izvesti $\neg G$

C Ako je svaki model od F također model od G, ali primjenom pravila iz R na formule F ne možemo izvesti G

D Ako je svaki model od G također model od F, ali postoji pravilo u R kojim iz F ne možemo izvesti G

13 (P) Zadane su FOL klauzule $Q(f(x), b) \vee \neg P(y) \vee Q(x, y)$ i $P(z) \vee \neg Q(g(c), z) \vee P(a)$. **Koja od sljedećih klauzula je rezolventa ovih dviju klauzula?**

A $Q(f(x), b) \vee \neg Q(g(c), b) \vee P(a)$

B $Q(g(c), a) \vee Q(f(x), b)$

C $Q(f(x), b) \vee Q(x, a) \vee P(y)$

D $Q(f(g(c)), b) \vee \neg P(a) \vee P(a)$

14 (R) Zadane su premise: (1) *U svakom gradu postoji pošta*, (2) *Virovitica je grad*, (3) *Svi vole one koji su sretni*, (4) *Lucija je dijete u Virovitici*, (5) *Onaj koji nekoga/nešto voli je sretan*, (6) *Svako dijete voli svaku poštu.* Želimo dokazati da iz ovih premisa logički slijedi *U Virovitici postoji netko koga svi vole.* Pretvorite ove izjave u FOL formule koristeći predikate $U(x, y)$ za "x je u y", $V(x, y)$ za "x voli y", $P(x)$ za "x je pošta", $G(x)$ za "x je grad", $S(x)$ za "x je sretan" i $D(x)$ za "x je dijete". Zatim primijenite rezoluciju opovrgavanjem uz strategiju skupa potpore. **Koliko klauzula ulazi u rezolucijski postupak i koji je najmanji broj rezolucijskih koraka potrebnih za dokaz?** (1 rezolucijski korak = 1 primjena rezolucijskog pravila.)

A 9 klauzula, 7 koraka

B 8 klauzula, 6 koraka

C 9 klauzula, 5 koraka

D 8 klauzula, 8 koraka

15 (R) Zadane su premise: *"Bela laje (L) ako je gladna (G) ili ako je uštap (U) ili ako je preplašena (P). Ako nije uštap, Bela dobro spava (S). Kada previše jede (J), onda Bela ne spava dobro. Bela nije preplašena ili je gladna."* **Koja je od sljedećih tvrdnji logička posljedica ovih premisa?**

A *Bela ne spava dobro.*

B *Ako Bela ne spava dobro, onda laje.*

C *Ako Bela laje, onda previše jede.*

D *Ako Bela laje, onda ne spava dobro.*

16 (P) Pravila zaključivanja poželjno su i ispravna i potpuna. Razmotrite pravilo zaključivanja r definirano kao $(A \vee B) \to C, \neg B \vdash_r A \to C$. **Kakvo je pravilo r?**

A Potpuno, ali nije ispravno

B Ispravno, ali nije potpuno

C Ispravno i potpuno

D Niti ispravno niti potpuno

4. Logičko programiranje i ekspertni sustavi (4 pitanja)

17 (T) Izvođenje programa u Prologu odgovara zaključivanju u Hornovoj klauzalnoj logici (HKL). HKL je podskup logike prvoga reda (FOL). **Zašto Prolog koristi HKL, a ne FOL?**

A Klauzule u HKL su zatvorene pod rezolucijom

B Rezolucija opovrgavanjem u HKL je vremenski učinkovita

C Rezolucije opovrgavanjem u HKL je ispravna i potpuna

D Zaključivanje u HKL uz dodatak negacije je odlučivo

18 (T) Ekspertni sustavi koriste se za automatsko zaključivanje u specifičnim domenama ljudskog znanja. **Što je karakteristika ekspertnih sustava?**

A Domenski specifično znanje definirano je u obliku činjenica i ako-onda pravila pohranjenih u stroju za zaključivanje

B Mogu zaključivati ulančavanjem unazad, što je prikladno kada postoji mnogo mogućih zaključaka, a malo početnih činjenica

C Baza znanja odvojena je od mehanizma zaključivanja, kako bi se isti mehanizam mogao koristiti s različitim bazama znanja

D Međuciljevi koje treba dokazati pohranjuju se na stog, a u slučaju omogućenosti većeg broja pravila, prednost ima pravilo koje je navedeno posljednje

19 (R) Baza znanja u Prologu modelira tramvajsku mrežu u Zagrebu. Baza sadrži sljedeće činjenice i pravila:

```
pruga(frankopanska, zrinjevac).
pruga(frankopanska, vodnikova).
pruga(zrinjevac, branimirova).
pruga(vodnikova, branimirova).
zastoj(zrinjevac).
prohodno(X, Y) :- pruga(X, Y), not(zastoj(X)).
povezano(X, Y) :- prohodno(X,Y); prohodno(Y, X).
dosezljivo(X, Y) :- povezano(X, Y).
dosezljivo(X, Y) :- povezano(X, Z), dosezljivo(Z, Y).
```

Nad ovako definiranom bazom znanja izvodimo upit `dosezljivo(branimirova, frankopanska)`. Prolog će za ovaj upit vratiti `True`. Nacrtajte Prologovo stablo dokaza za ovaj upit sve do mjesta gdje se postupak pretrage po četvrti put vraća na točku odabira. Pritom za dokazivanje negiranog cilja koristite dva čvora (jedan za `not(P(X))` i drugi za `P(X)`)). **Koliko čvorova ima stablo dokaza do trenutka četvrtog povratka u postupku pretraživanja?**

A 9 B 15 C 11 D 13

20 (R) Baza znanja ekspertnog sustava sadrži sljedeća pravila:

(1) AKO $(A = a_2) \wedge (D = d_2)$ ONDA $C = c_1$
(2) AKO $(F = f_3) \vee (B = b_3)$ ONDA $C = c_2$
(3) AKO $(E = e_1) \vee (B = b_1)$ ONDA $(A = a_1) \wedge (D = d_2)$
(4) AKO $F = f_1$ ONDA $D = d_2$
(5) AKO $F = f_2$ ONDA $E = e_2$
(6) AKO $(B = b_3) \vee (D = d_1)$ ONDA $(E = e_1) \wedge (A = a_2)$

Sustav koristimo za izvođenje vrijednosti varijable C ulančavanjem unazad. Prednost imaju pravila s manjim rednim brojem. Pravila koja su jednom palila više ne mogu paliti. Jednom izvedeni međuciljevi brišu se sa stoga, neovisno o njihovoj poziciji na stogu. Na možebitne upite od strane sustava, korisnik odgovara sa $B = b_3$ i $F = f_1$. **Što radi ekspertni sustav pri izvođenju vrijednosti varijable C?**

A Pali tri pravila i izvodi $C = c_2$
B Odbacuje pravilo 2 te kasnije pali pravilo 5
C Izvodi $E = e_1$ te kasnije $E = e_2$
D Pali pet pravila i izvodi $C = c_1$

Međuispit iz Uvoda u umjetnu inteligenciju (2023./2024.)

– 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 i heurističko pretraživanje (7 pitanja)

1 (P) Neka je S skup stanja te neka su s_1, s_2 ,s_3 i s_4 stanja iz S. Vrijedi $succ(s_1) = \{(s_2, 4)\}$ i $succ(s_2) = \{(s_3, c)\}$, gdje je c neka cijena prijelaza. Razmatramo dvije heuristike, h_1 i h_2. Heuristika h_1 definirana je u odnosu na ciljno stanje s_3, a heuristika h_2 u odnosu na ciljno stanje s_4. Vrijedi $h_1(s_1) = 11$ i $h_1(s_2) = 9$ te $h_2(s_1) = 20$, $h_2(s_2) = 16$ i $h_2(s_3) = 8$. Obje su heuristike optimistične. Međutim, na temelju ovih informacija heuristiku h_2 možemo unaprijediti tako da bude obavještenija (bliža stvarnoj cijeni). **Koje od navedenih vrijednosti za h_2 sigurno znamo da daju obavještenju, ali još uvijek optimističnu heuristiku?**

A $h_2(s_1) = 21$, $h_2(s_2) = 16$
B $h_2(s_1) = 20$, $h_2(s_2) = 18$
C $h_2(s_1) = 20$, $h_2(s_2) = 17$
D $h_2(s_1) = 21$, $h_2(s_2) = 17$

2 (T) Jedno od poželjnih svojstava heurističke funkcije je *konzistentnost* (ili *monotonost*). Razmotrite algoritam A^*. **Koja je prednost algoritma A^* s konzistentnom heuristikom u odnosu na A^* s nekonzistentnom ali optimističnom heuristikom?**

A Algoritam je potpun
B Algoritam je optimalan
C Algoritam ima manju prostornu složenost
D Algoritam ispituje manji broj čvorova

3 (R) Algoritmom slijepog pretraživanja rješavamo problem misionara i kanibala. Tri misionara i tri kanibala potrebno je jednim čamcem prevesti s jedne strane obale rijeke na drugu, pri čemu se ne smije dogoditi da se na jednoj strani nađu i kanibali i misionari, ali da je kanibala više nego misionara. Čamac može prevesti najviše dvije osobe i ne može ploviti prazan. Stanje prikazujemo kao trojku MKC, gdje su M i K broj misionara odnosno kanibala na lijevoj obali rijeke, a C je strana obale na kojoj se nalazi čamac. Npr., u stanju 32D na lijevoj obali su tri misionara i dva kanibala (jedan kanibal je onda na desnoj obali), a čamac je na desnoj obali. Početno stanje je 33L, a završno 00D. Funkcija sljedbenika generira sva legitimna stanja, uključivo i ona koja rezultiraju neuspjehom i nemaju sljedbenika, npr., stanje 23D. Koristimo algoritam iterativnog pretraživanja u dubinu (IDS) bez liste posjećenih stanja. Kod proširenja čvora, algoritam čvorove djecu generira prema numeričkom redoslijedu; npr., stanje 13D prethodi stanju 23D. **Koliko će čvorova algoritam IDS ispitati prije nego što proširi čvor sa stanjem** 31D?

A 11 B 18 C 16 D 14

4 (P) Rješavamo problem pretraživanja prostora stanja za koji vrijedi $m = 10$, $d = 7$ i $b = 4$. Koristimo pretraživanje u širinu (BFS) i pretraživanje u dubinu (DFS). Za oba algoritma razmotrite najgori slučaj vremenske složenosti. **Koliko će u takvom slučaju algoritam DFS generirati više čvorova od algoritma BFS?**

A 1376176 B 1310716 C 1376256 D 1310640

5 (P) 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 $\ldots, b, c, d, a, b, \ldots$?**

A Ograničeno pretraživanje u dubinu ($k = 2$)
B Pretraživanje u dubinu
C Iterativno pretraživanje u dubinu
D Pretraživanje u širinu

6 (T) Jedna od komponenti problema pretraživanja prostora stanja je funkcija sljedbenika *succ*. Razmotrite algoritam pretraživanja s jednolikom cijenom (UCS). **Kod tog algoritma, u što funkcija *succ* preslikava svako stanje?**

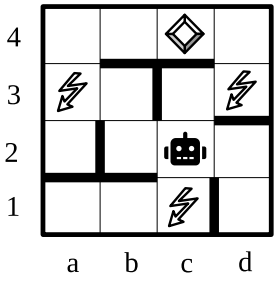
A Kartezijev umnožak skupa stanja i skupa nenegativnih brojeva

B Skup parova sačinjenih od početnog i završnog stanja

C Skup parova sačinjenih od stanja i pozitivnog broja

D Skup svih podskupova stanja (moguće prazan)

7 (R) Na slici desno prikazana je mapa dvodimenzijskoga svijeta od 16 polja. Robot Robby početno se nalazi na polju $c2$ i treba doći na polje $c4$ na kojemu se nalazi dijamant. Robby se može kretati prema gore, dolje, lijevo i desno (npr., sa $c2$ se može kretati na $c3$, $c1$, $b2$ i $d2$), ali ne može prolaziti kroz zidove (npr., ne može proći sa $c3$ na $c4$). Robby želi pronaći put s najmanjim utroškom baterije. Svaki potez Robbyja košta jednu energetsku jedinicu iz baterije, osim kada zakorači na polja $a3$, $c1$ ili $d3$, gdje je zbog energetskog polja utrošak baterije jednak 3 energetske jedinice. Primijenite algoritam A^*. Stanje odgovara Robbyjevoj poziciji, a cijena puta odgovara broju utrošenih energetskih jedinica. Za heuristiku koristite L1-udaljenost (Manhattan-udaljenost). U slučaju izjednačenja cijene, proširuje se čvor sa stanjem koje alfanumerički dolazi prvo (npr., $a2$ prije $b2$ prije $b3$). **Koliko čvorova će algoritam A^* ukupno proširiti?**



A 9 B 8 C 10 D 7

2. Igranje igara (3 pitanja)

8 (P) Alfa-beta podrezivanje može značajno smanjiti broj generiranih čvorova kod algoritma minimax. Ušteda ovisi o redoslijedu kojim se generiraju čvorovi. Optimalan redoslijed je onaj koji djecu čvora MAX generira silazno po minimax vrijednostima, a djecu čvora MIN uzlazno po minimax vrijednostima. Razmotite igru s dva moguća poteza u svakome stanju. Stablo se pretražuje do dubine 3. Vrijednosti heuristike igrača MAX za stanja u listovima stabla igre, kada su čvorovi-djeca generirani nasumičnim redoslijedom, su sljedeća (slijeva nadesno): $0, 3, 4, -3, 4, -1, 0, 2$. Na potezu je igrač MAX. Odredite broj čvorova koji će biti uklonjeni podrezivanjem alfa-beta. Zatim ponovite izračun, ali s optimalnim redoslijedom generiranja čvorova. **Koliko više čvorova će biti podrezano s optimalnim redoslijedom u odnosu na slučajan redoslijed generiranja čvorova?**

A 4 B 2 C 1 D 3

9 (R) Razmatramo igru dvaju minimax-igrača. Stablo igre definirano je prijelazima $succ(A) = \{B, C, D\}$, $succ(B) = \{E, F\}$, $succ(C) = \{G, H\}$, $succ(D) = \{I, J\}$, $succ(F) = succ(G) = \{K, L\}$, $succ(E) = succ(H) = succ(I) = \{M, N\}$ te $succ(J) = \{O, P\}$. Igrač MAX koristi heuristiku h_1, a igrač MIN heuristiku h_2. Vrijednosti heuristika su sljedeće (obje heuristike definiraju dobit dotičnog igrača):

	E	F	G	H	I	J	K	L	M	N	O	P
h_1	1	-2	3	2	3	5	-1	0	-2	-1	0	2
h_2	0	5	2	4	-1	2	3	2	-3	-2	1	-4

Oba igrača pretražuju najviše dva poteza unaprijed (jedan svoj i jedan suparnički). Igru započinje igrač MAX u stanju A. **Kroz koja stanja će se razvijati igra?**

A $A, D, J, \ldots$ B $A, D, I, \ldots$ C $A, C, H, \ldots$ D $A, C, G, \ldots$

10 (T) Igrači MAX i MIN primjenjuju algoritam minimax prije svakog svog poteza, pretražujući do neke unaprijed definirane dubine. Zamislite, međutim, da igrač MAX algoritam minimax primjenjuje samo jednom, i to na početku igre, a da zatim tako dobivene minimax vrijednosti koristi za sve svoje buduće poteze. **U kojem bi slučaju tako implementirani igrač MAX sigurno maksimizirao svoju minimalnu dobit?**

A Ako je pri izračunu minimax-vrijednosti dosegao sva završna stanja igre

B Ako koristi identičnu heuristiku kao igrač MIN

C Ako je heuristika igrača MAX obavještenija od heuristike igrača MIN

D Nema garancije da bi takav igrač maksimizirao svoju minimalnu dobit

3. Prikazivanje znanja i automatsko zaključivanje (6 pitanja)

11 (R) Zadane su premise: *"Bela laje (L) ako je gladna (G) ili ako je uštap (U) ili ako je preplašena (P). Ako nije uštap, Bela dobro spava (S). Kada previše jede (J), onda Bela ne spava dobro. Bela nije preplašena ili je gladna."* **Koja je od sljedećih tvrdnji logička posljedica ovih premisa?**

A *Ako Bela ne spava dobro, onda previše jede.*
B *Bela laje ako previše jede.*
C *Bela laje.*
D *Bela ne spava dobro.*

12 (P) Pravila zaključivanja poželjno su i ispravna i potpuna. Razmotrite pravilo zaključivanja r definirano kao $(A \vee B) \to C, \neg B \vdash_r A \to C$. **Kakvo je pravilo r?**

A Potpuno, ali nije ispravno
B Niti ispravno niti potpuno
C Ispravno, ali nije potpuno
D Ispravno i potpuno

13 (R) Zadane su premise: (1) *U svakom gradu postoji pošta*, (2) *Virovitica je grad*, (3) *Svi vole one koji su sretni*, (4) *Lucija je dijete u Virovitici*, (5) *Onaj koji nekoga/nešto voli je sretan*, (6) *Svako dijete voli svaku poštu.* Želimo dokazati da iz ovih premisa logički slijedi *U Virovitici postoji netko koga svi vole.* Pretvorite ove izjave u FOL formule koristeći predikate $U(x, y)$ za "x je u y", $V(x, y)$ za "x voli y", $P(x)$ za "x je pošta", $G(x)$ za "x je grad", $S(x)$ za "x je sretan" i $D(x)$ za "x je dijete". Zatim primijenite rezoluciju opovrgavanjem uz strategiju skupa potpore. **Koliko klauzula ulazi u rezolucijski postupak i koji je najmanji broj rezolucijskih koraka potrebnih za dokaz?** (1 rezolucijski korak = 1 primjena rezolucijskog pravila.)

A 8 klauzula, 6 koraka
B 8 klauzula, 8 koraka
C 9 klauzula, 5 koraka
D 9 klauzula, 7 koraka

14 (T) Propozicijska logika (PL) i logika prvog reda (FOL) dva su formalizma za prikaz znanja. **Što je prednost, a što nedostatak PL u odnosu na FOL?** (Podsjetnik: Problem je *traktabilan* ako je rješiv u polinomijalnom vremenu.)

A Dokazivanje u PL je odlučivo, ali nije traktabilno kao u FOL
B PL je ekspresivniji od FOL, ali je dokazivanje netraktabilno
C Dokazivanje u PL je odlučivo, ali je PL manje ekspresivan od FOL
D Dokazivanje u PL je traktabilno, ali je FOL poluodlučiv

15 (T) Neka je R skup pravila zaključivanja te neka su F i G dobro oblikovane formule logike. **Kada za skup R kažemo da *nije* potpun skup pravila zaključivanja?**

A Ako je formula $F \to G$ zadovoljiva, ali ne postoji pravilo u R kojim iz $F \wedge \neg G$ možemo izvesti NIL
B Ako je formula $\neg F \vee G$ valjana, ali ne postoji lanac pravila u R kojim iz F možemo izvesti $\neg G$
C Ako je svaki model od G također model od F, ali postoji pravilo u R kojim iz F ne možemo izvesti G
D Ako je svaki model od F također model od G, ali primjenom pravila iz R na formule F ne možemo izvesti G

16 (P) Zadane su FOL klauzule $Q(f(x), b) \vee \neg P(y) \vee Q(x, y)$ i $P(z) \vee \neg Q(g(c), z) \vee P(a)$. **Koja od sljedećih klauzula je rezolventa ovih dviju klauzula?**

A $Q(f(g(c)), b) \vee \neg P(a) \vee P(a)$
B $Q(f(x), b) \vee \neg Q(g(c), b) \vee P(a)$
C $Q(g(c), a) \vee Q(f(x), b)$
D $Q(f(x), b) \vee Q(x, a) \vee P(y)$

4. Logičko programiranje i ekspertni sustavi (4 pitanja)

17 (T) Izvođenje programa u Prologu odgovara zaključivanju u Hornovoj klauzalnoj logici (HKL). HKL je podskup logike prvoga reda (FOL). **Zašto Prolog koristi HKL, a ne FOL?**

A Rezolucije opovrgavanjem u HKL je ispravna i potpuna
B Klauzule u HKL su zatvorene pod rezolucijom
C Rezolucija opovrgavanjem u HKL je vremenski učinkovita
D Zaključivanje u HKL uz dodatak negacije je odlučivo

18 (R) Baza znanja u Prologu modelira tramvajsku mrežu u Zagrebu. Baza sadrži sljedeće činjenice i pravila:

```
pruga(frankopanska, zrinjevac).
pruga(frankopanska, vodnikova).
pruga(zrinjevac, branimirova).
pruga(vodnikova, branimirova).
zastoj(zrinjevac).
prohodno(X, Y) :- pruga(X, Y), not(zastoj(X)).
povezano(X, Y) :- prohodno(X,Y); prohodno(Y, X).
dosezljivo(X, Y) :- povezano(X, Y).
dosezljivo(X, Y) :- povezano(X, Z), dosezljivo(Z, Y).
```

Nad ovako definiranom bazom znanja izvodimo upit `dosezljivo(branimirova, frankopanska)`. Prolog će za ovaj upit vratiti `True`. Nacrtajte Prologovo stablo dokaza za ovaj upit sve do mjesta gdje se postupak pretrage po četvrti put vraća na točku odabira. Pritom za dokazivanje negiranog cilja koristite dva čvora (jedan za `not(P(X))` i drugi za `P(X)`). **Koliko čvorova ima stablo dokaza do trenutka četvrtog povratka u postupku pretraživanja?**

A 9 B 11 C 13 D 15

19 (T) Ekspertni sustavi koriste se za automatsko zaključivanje u specifičnim domenama ljudskog znanja. **Što je karakteristika ekspertnih sustava?**

A Baza znanja odvojena je od mehanizma zaključivanja, kako bi se isti mehanizam mogao koristiti s različitim bazama znanja

B Mogu zaključivati ulančavanjem unazad, što je prikladno kada postoji mnogo mogućih zaključaka, a malo početnih činjenica

C Domenski specifično znanje definirano je u obliku činjenica i ako-onda pravila pohranjenih u stroju za zaključivanje

D Međuciljevi koje treba dokazati pohranjuju se na stog, a u slučaju omogućenosti većeg broja pravila, prednost ima pravilo koje je navedeno posljednje

20 (R) Baza znanja ekspertnog sustava sadrži sljedeća pravila:

(1) AKO $(A = a_2) \wedge (D = d_2)$ ONDA $C = c_1$
(2) AKO $(F = f_3) \vee (B = b_3)$ ONDA $C = c_2$
(3) AKO $(E = e_1) \vee (B = b_1)$ ONDA $(A = a_1) \wedge (D = d_2)$
(4) AKO $F = f_1$ ONDA $D = d_2$
(5) AKO $F = f_2$ ONDA $E = e_2$
(6) AKO $(B = b_3) \vee (D = d_1)$ ONDA $(E = e_1) \wedge (A = a_2)$

Sustav koristimo za izvođenje vrijednosti varijable C ulančavanjem unazad. Prednost imaju pravila s manjim rednim brojem. Pravila koja su jednom palila više ne mogu paliti. Jednom izvedeni međuciljevi brišu se sa stoga, neovisno o njihovoj poziciji na stogu. Na možebitne upite od strane sustava, korisnik odgovara sa $B = b_3$ i $F = f_1$. **Što radi ekspertni sustav pri izvođenju vrijednosti varijable C?**

A Završava s četiri činjenice u radnoj memoriji
B Izvodi $E = e_1$ te kasnije $E = e_2$
C Pali četiri pravila i izvodi $C = c_1$
D Odbacuje pravilo 5 te kasnije pali pravilo 6

Međuispit iz Uvoda u umjetnu inteligenciju (2023./2024.)

– 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 i heurističko pretraživanje (7 pitanja)

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

A Pretraživanje u dubinu
B Ograničeno pretraživanje u dubinu ($k = 2$)
C Pretraživanje u širinu
D Iterativno pretraživanje u dubinu

2 (P) Rješavamo problem pretraživanja prostora stanja za koji vrijedi $m = 10$, $d = 7$ i $b = 3$. Koristimo pretraživanje u širinu (BFS) i pretraživanje u dubinu (DFS). Za oba algoritma razmotrite najgori slučaj vremenske složenosti. **Koliko će u takvom slučaju algoritam DFS generirati više čvorova od algoritma BFS?**

A 78729 B 78696 C 85293 D 85257

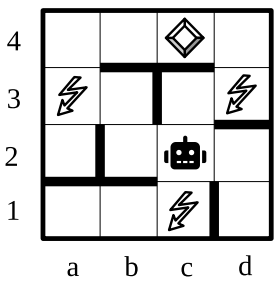
3 (T) Jedno od poželjnih svojstava heurističke funkcije je *konzistentnost* (ili *monotonost*). Razmotrite algoritam A^*. **Koja je prednost algoritma A^* s konzistentnom heuristikom u odnosu na A^* s nekonzistentnom ali optimističnom heuristikom?**

A Algoritam je potpun
B Algoritam ima manju prostornu složenost
C Algoritam je optimalan
D Algoritam ispituje manji broj čvorova

4 (P) Neka je S skup stanja te neka su s_1, s_2 ,s_3 i s_4 stanja iz S. Vrijedi $succ(s_1) = \{(s_2, 4)\}$ i $succ(s_2) = \{(s_3, c)\}$, gdje je c neka cijena prijelaza. Razmatramo dvije heuristike, h_1 i h_2. Heuristika h_1 definirana je u odnosu na ciljno stanje s_3, a heuristika h_2 u odnosu na ciljno stanje s_4. Vrijedi $h_1(s_1) = 11$ i $h_1(s_2) = 9$ te $h_2(s_1) = 20$, $h_2(s_2) = 16$ i $h_2(s_3) = 8$. Obje su heuristike optimistične. Međutim, na temelju ovih informacija heuristiku h_2 možemo unaprijediti tako da bude obavještenija (bliža stvarnoj cijeni). **Koje od navedenih vrijednosti za h_2 sigurno znamo da daju obavještenju, ali još uvijek optimističnu heuristiku?**

A $h_2(s_1) = 20$, $h_2(s_2) = 17$
B $h_2(s_1) = 21$, $h_2(s_2) = 17$
C $h_2(s_1) = 20$, $h_2(s_2) = 18$
D $h_2(s_1) = 21$, $h_2(s_2) = 16$

5 (R) Na slici desno prikazana je mapa dvodimenzijskoga svijeta od 16 polja. Robot Robby početno se nalazi na polju $c2$ i treba doći na polje $c4$ na kojemu se nalazi dijamant. Robby se može kretati prema gore, dolje, lijevo i desno (npr., sa $c2$ se može kretati na $c3$, $c1$, $b2$ i $d2$), ali ne može prolaziti kroz zidove (npr., ne može proći sa $c3$ na $c4$). Robby želi pronaći put s najmanjim utroškom baterije. Svaki potez Robbyja košta jednu energetsku jedinicu iz baterije, osim kada zakorači na polja $a3$, $c1$ ili $d3$, gdje je zbog energetskog polja utrošak baterije jednak 3 energetske jedinice. Primijenite algoritam A^*. Stanje odgovara Robbyjevoj poziciji, a cijena puta odgovara broju utrošenih energetskih jedinica. Za heuristiku koristite L1-udaljenost (Manhattan-udaljenost). U slučaju izjednačenja cijene, proširuje se čvor sa stanjem koje alfanumerički dolazi prvo (npr., $a2$ prije $b2$ prije $b3$). **Koliko čvorova će algoritam A^* ukupno proširiti?**



A 9 B 10 C 7 D 8

6 (R) Algoritmom slijepog pretraživanja rješavamo problem misionara i kanibala. Tri misionara i tri kanibala potrebno je jednim čamcem prevesti s jedne strane obale rijeke na drugu, pri čemu se ne smije dogoditi da se na jednoj strani nađu i kanibali i misionari, ali da je kanibala više nego misionara. Čamac može prevesti najviše dvije osobe i ne može ploviti prazan. Stanje prikazujemo kao trojku MKC, gdje su M i K broj misionara odnosno kanibala na lijevoj obali rijeke, a C je strana obale na kojoj se nalazi čamac. Npr., u stanju 32D na lijevoj obali su tri misionara i dva kanibala (jedan kanibal je onda na desnoj obali), a čamac je na desnoj obali. Početno stanje je 33L, a završno 00D. Funkcija sljedbenika generira sva legitimna stanja, uključivo i ona koja rezultiraju neuspjehom i nemaju sljedbenika, npr., stanje 23D. Koristimo algoritam iterativnog pretraživanja u dubinu (IDS) bez liste posjećenih stanja. Kod proširenja čvora, algoritam čvorove djecu generira prema numeričkom redoslijedu; npr., stanje 13D prethodi stanju 23D. **Koliko će čvorova algoritam IDS ispitati prije nego što proširi čvor sa stanjem** 31D?

A 11 B 18 C 16 D 14

7 (T) Jedna od komponenti problema pretraživanja prostora stanja je funkcija sljedbenika *succ*. Razmotrite algoritam pretraživanja s jednolikom cijenom (UCS). **Kod tog algoritma, u što funkcija** *succ* **preslikava svako stanje?**

A Skup parova sačinjenih od početnog i završnog stanja

B Kartezijev umnožak skupa stanja i skupa nenegativnih brojeva

C Skup parova sačinjenih od stanja i pozitivnog broja

D Skup svih podskupova stanja (moguće prazan)

2. Igranje igara (3 pitanja)

8 (R) Razmatramo igru dvaju minimax-igrača. Stablo igre definirano je prijelazima $succ(A) = \{B, C, D\}$, $succ(B) = \{E, F\}$, $succ(C) = \{G, H\}$, $succ(D) = \{I, J\}$, $succ(F) = succ(G) = \{K, L\}$, $succ(E) = succ(H) = succ(I) = \{M, N\}$ te $succ(J) = \{O, P\}$. Igrač MAX koristi heuristiku h_1, a igrač MIN heuristiku h_2. Vrijednosti heuristika su sljedeće (obje heuristike definiraju dobit dotičnog igrača):

	E	F	G	H	I	J	K	L	M	N	O	P
h_1	1	−2	3	2	3	5	−1	0	−2	−1	0	2
h_2	0	5	2	4	−1	2	3	2	1	−4	−3	−2

Oba igrača pretražuju najviše dva poteza unaprijed (jedan svoj i jedan suparnički). Igru započinje igrač MAX u stanju A. **Kroz koja stanja će se razvijati igra?**

A $A, D, J, \ldots$ B $A, D, I, \ldots$ C $A, C, H, \ldots$ D $A, C, G, \ldots$

9 (P) Alfa-beta podrezivanje može značajno smanjiti broj generiranih čvorova kod algoritma minimax. Ušteda ovisi o redoslijedu kojim se generiraju čvorovi. Optimalan redoslijed je onaj koji djecu čvora MAX generira silazno po minimax vrijednostima, a djecu čvora MIN uzlazno po minimax vrijednostima. Razmotite igru s dva moguća poteza u svakome stanju. Stablo se pretražuje do dubine 3. Vrijednosti heuristike igrača MAX za stanja u listovima stabla igre, kada su čvorovi-djeca generirani nasumičnim redoslijedom, su sljedeća (slijeva nadesno): $0, 3, -3, 4, 4, -1, 0, 2$. Na potezu je igrač MAX. Odredite broj čvorova koji će biti uklonjeni podrezivanjem alfa-beta. Zatim ponovite izračun, ali s optimalnim redoslijedom generiranja čvorova. **Koliko više čvorova će biti podrezano s optimalnim redoslijedom u odnosu na slučajan redoslijed generiranja čvorova?**

A 3 B 1 C 4 D 2

10 (T) Igrači MAX i MIN primjenjuju algoritam minimax prije svakog svog poteza, pretražujući do neke unaprijed definirane dubine. Zamislite, međutim, da igrač MAX algoritam minimax primjenjuje samo jednom, i to na početku igre, a da zatim tako dobivene minimax vrijednosti koristi za sve svoje buduće poteze. **U kojem bi slučaju tako implementirani igrač MAX sigurno maksimizirao svoju minimalnu dobit?**

A Nema garancije da bi takav igrač maksimizirao svoju minimalnu dobit

B Ako je pri izračunu minimax-vrijednosti dosegao sva završna stanja igre

C Ako je heuristika igrača MAX obavještenija od heuristike igrača MIN

D Ako koristi identičnu heuristiku kao igrač MIN

3. Prikazivanje znanja i automatsko zaključivanje (6 pitanja)

11 (T) Propozicijska logika (PL) i logika prvog reda (FOL) dva su formalizma za prikaz znanja. **Što je prednost, a što nedostatak PL u odnosu na FOL?** (Podsjetnik: Problem je *traktabilan* ako je rješiv u polinomijalnom vremenu.)

A Dokazivanje u PL je odlučivo, ali je PL manje ekspresivan od FOL

B PL je ekspresivniji od FOL, ali je dokazivanje netraktabilno

C Dokazivanje u PL je odlučivo, ali nije traktabilno kao u FOL

D Dokazivanje u PL je traktabilno, ali je FOL poluodlučiv

12 (P) Pravila zaključivanja poželjno su i ispravna i potpuna. Razmotrite pravilo zaključivanja r definirano kao $(A \vee B) \rightarrow C, \neg B \vdash_r \neg(C \rightarrow A)$. **Kakvo je pravilo r?**

A Potpuno, ali nije ispravno

B Ispravno i potpuno

C Ispravno, ali nije potpuno

D Niti ispravno niti potpuno

13 (P) Zadane su FOL klauzule $Q(f(x), b) \vee \neg P(y) \vee Q(x, y)$ i $P(z) \vee \neg Q(g(c), z) \vee P(a)$. **Koja od sljedećih klauzula je rezolventa ovih dviju klauzula?**

A $Q(f(x), b) \vee Q(x, a) \vee P(y)$

B $Q(g(c), a) \vee Q(f(x), b)$

C $Q(f(x), b) \vee \neg Q(g(c), b) \vee P(a)$

D $Q(f(g(c)), b) \vee \neg P(a) \vee P(a)$

14 (R) Zadane su premise: (1) *U svakom gradu postoji pošta*, (2) *Virovitica je grad*, (3) *Svi vole one koji su sretni*, (4) *Lucija je dijete u Virovitici*, (5) *Onaj koji nekoga/nešto voli je sretan*, (6) *Svako dijete voli svaku poštu.* Želimo dokazati da iz ovih premisa logički slijedi *U Virovitici postoji netko koga svi vole.* Pretvorite ove izjave u FOL formule koristeći predikate $U(x, y)$ za "x je u y", $V(x, y)$ za "x voli y", $P(x)$ za "x je pošta", $G(x)$ za "x je grad", $S(x)$ za "x je sretan" i $D(x)$ za "x je dijete". Zatim primijenite rezoluciju opovrgavanjem uz strategiju skupa potpore. **Koliko klauzula ulazi u rezolucijski postupak i koji je najmanji broj rezolucijskih koraka potrebnih za dokaz?** (1 rezolucijski korak = 1 primjena rezolucijskog pravila.)

A 8 klauzula, 8 koraka

B 8 klauzula, 6 koraka

C 9 klauzula, 7 koraka

D 9 klauzula, 5 koraka

15 (T) Neka je R skup pravila zaključivanja te neka su F i G dobro oblikovane formule logike. **Kada za skup R kažemo da *nije* potpun skup pravila zaključivanja?**

A Ako je formula $\neg F \vee G$ valjana, ali ne postoji lanac pravila u R kojim iz F možemo izvesti $\neg G$

B Ako je svaki model od G također model od F, ali postoji pravilo u R kojim iz F ne možemo izvesti G

C Ako je svaki model od F također model od G, ali primjenom pravila iz R na formule F ne možemo izvesti G

D Ako je formula $F \rightarrow G$ zadovoljiva, ali ne postoji pravilo u R kojim iz $F \wedge \neg G$ možemo izvesti NIL

16 (R) Zadane su premise: *"Bela laje (L) ako je gladna (G) ili ako je uštap (U) ili ako je preplašena (P). Ako nije uštap, Bela dobro spava (S). Kada previše jede (J), onda Bela ne spava dobro. Bela nije preplašena ili je gladna."* **Koja je od sljedećih tvrdnji logička posljedica ovih premisa?**

A *Bela laje ili ne jede previše.*

B *Ako Bela laje, onda previše jede.*

C *Ako Bela laje, onda ne spava dobro.*

D *Bela je gladna.*

4. Logičko programiranje i ekspertni sustavi (4 pitanja)

17 (R) Baza znanja ekspertnog sustava sadrži sljedeća pravila:

(1) AKO $(A = a_2) \wedge (D = d_2)$ ONDA $C = c_1$
(2) AKO $(F = f_3) \vee (B = b_3)$ ONDA $C = c_2$
(3) AKO $(E = e_1) \vee (B = b_1)$ ONDA $(A = a_1) \wedge (D = d_2)$
(4) AKO $F = f_1$ ONDA $D = d_2$
(5) AKO $F = f_2$ ONDA $E = e_2$
(6) AKO $(B = b_3) \vee (D = d_1)$ ONDA $(E = e_1) \wedge (A = a_2)$

Sustav koristimo za izvođenje vrijednosti varijable C ulančavanjem unazad. Prednost imaju pravila s manjim rednim brojem. Pravila koja su jednom palila više ne mogu paliti. Jednom izvedeni međuciljevi brišu se sa stoga,

neovisno o njihovoj poziciji na stogu. Na možebitne upite od strane sustava, korisnik odgovara sa $B = b_3$ i $F = f_1$. **Što radi ekspertni sustav pri izvođenju vrijednosti varijable C?**

A Izvodi $E = e_1$ te kasnije $E = e_2$

B Odbacuje pravilo 5 te kasnije pali pravilo 6

C Završava s četiri činjenice u radnoj memoriji

D Odbacuje pravilo 2 te kasnije pali pravilo 5

18 (T) Ekspertni sustavi koriste se za automatsko zaključivanje u specifičnim domenama ljudskog znanja. **Što je karakteristika ekspertnih sustava?**

A Baza znanja odvojena je od mehanizma zaključivanja, kako bi se isti mehanizam mogao koristiti s različitim bazama znanja

B Međuciljevi koje treba dokazati pohranjuju se na stog, a u slučaju omogućenosti većeg broja pravila, prednost ima pravilo koje je navedeno posljednje

C Mogu zaključivati ulančavanjem unazad, što je prikladno kada postoji mnogo mogućih zaključaka, a malo početnih činjenica

D Domenski specifično znanje definirano je u obliku činjenica i ako-onda pravila pohranjenih u stroju za zaključivanje

19 (T) Izvođenje programa u Prologu odgovara zaključivanju u Hornovoj klauzalnoj logici (HKL). HKL je podskup logike prvoga reda (FOL). **Zašto Prolog koristi HKL, a ne FOL?**

A Zaključivanje u HKL uz dodatak negacije je odlučivo

B Klauzule u HKL su zatvorene pod rezolucijom

C Rezolucija opovrgavanjem u HKL je vremenski učinkovita

D Rezolucije opovrgavanjem u HKL je ispravna i potpuna

20 (R) Baza znanja u Prologu modelira tramvajsku mrežu u Zagrebu. Baza sadrži sljedeće činjenice i pravila:

```
pruga(frankopanska, zrinjevac).
pruga(frankopanska, vodnikova).
pruga(zrinjevac, branimirova).
pruga(vodnikova, branimirova).
zastoj(zrinjevac).
prohodno(X, Y) :- pruga(X, Y), not(zastoj(X)).
povezano(X, Y) :- prohodno(X,Y); prohodno(Y, X).
dosezljivo(X, Y) :- povezano(X, Y).
dosezljivo(X, Y) :- povezano(X, Z), dosezljivo(Z, Y).
```

Nad ovako definiranom bazom znanja izvodimo upit `dosezljivo(branimirova, frankopanska)`. Prolog će za ovaj upit vratiti `True`. Nacrtajte Prologovo stablo dokaza za ovaj upit sve do mjesta gdje se postupak pretrage po četvrti put vraća na točku odabira. Pritom za dokazivanje negiranog cilja koristite dva čvora (jedan za `not(P(X))` i drugi za `P(X)`). **Koliko čvorova ima stablo dokaza do trenutka četvrtog povratka u postupku pretraživanja?**

A 15 B 9 C 11 D 13

Međuispit iz Uvoda u umjetnu inteligenciju (2023./2024.)

– 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 i heurističko pretraživanje (7 pitanja)

1 (P) 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 $\ldots, b, c, d, a, b, \ldots$?**

A Iterativno pretraživanje u dubinu
B Pretraživanje u dubinu
C Ograničeno pretraživanje u dubinu ($k = 4$)
D Pretraživanje u širinu

2 (P) Rješavamo problem pretraživanja prostora stanja za koji vrijedi $m = 10$, $d = 7$ i $b = 3$. Koristimo pretraživanje u širinu (BFS) i pretraživanje u dubinu (DFS). Za oba algoritma razmotrite najgori slučaj vremenske složenosti. **Koliko će u takvom slučaju algoritam DFS generirati više čvorova od algoritma BFS?**

A 85257 B 85293 C 78696 D 78729

3 (R) Algoritmom slijepog pretraživanja rješavamo problem misionara i kanibala. Tri misionara i tri kanibala potrebno je jednim čamcem prevesti s jedne strane obale rijeke na drugu, pri čemu se ne smije dogoditi da se na jednoj strani nađu i kanibali i misionari, ali da je kanibala više nego misionara. Čamac može prevesti najviše dvije osobe i ne može ploviti prazan. Stanje prikazujemo kao trojku MKC, gdje su M i K broj misionara odnosno kanibala na lijevoj obali rijeke, a C je strana obale na kojoj se nalazi čamac. Npr., u stanju 32D na lijevoj obali su tri misionara i dva kanibala (jedan kanibal je onda na desnoj obali), a čamac je na desnoj obali. Početno stanje je 33L, a završno 00D. Funkcija sljedbenika generira sva legitimna stanja, uključivo i ona koja rezultiraju neuspjehom i nemaju sljedbenika, npr., stanje 23D. Koristimo algoritam iterativnog pretraživanja u dubinu (IDS) bez liste posjećenih stanja. Kod proširenja čvora, algoritam čvorove djecu generira prema numeričkom redoslijedu; npr., stanje 13D prethodi stanju 23D. **Koliko će čvorova algoritam IDS ispitati prije nego što proširi čvor sa stanjem** 31D?

A 11 B 16 C 18 D 14

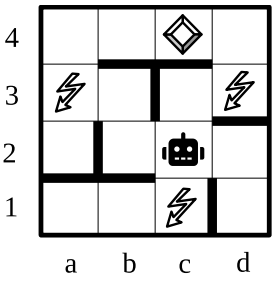
4 (T) Jedna od komponenti problema pretraživanja prostora stanja je funkcija sljedbenika *succ*. Razmotrite algoritam pretraživanja s jednolikom cijenom (UCS). **Kod tog algoritma, u što funkcija *succ* preslikava svako stanje?**

A Skup parova sačinjenih od stanja i pozitivnog broja
B Skup parova sačinjenih od početnog i završnog stanja
C Kartezijev umnožak skupa stanja i skupa nenegativnih brojeva
D Skup svih podskupova stanja (moguće prazan)

5 (T) Jedno od poželjnih svojstava heurističke funkcije je *konzistentnost* (ili *monotonost*). Razmotrite algoritam A^*. **Koja je prednost algoritma A^* s konzistentnom heuristikom u odnosu na A^* s nekonzistentnom ali optimističnom heuristikom?**

A Algoritam je optimalan
B Algoritam je potpun
C Algoritam ispituje manji broj čvorova
D Algoritam ima manju prostornu složenost

6 (R) Na slici desno prikazana je mapa dvodimenzijskoga svijeta od 16 polja. Robot Robby početno se nalazi na polju $c2$ i treba doći na polje $c4$ na kojemu se nalazi dijamant. Robby se može kretati prema gore, dolje, lijevo i desno (npr., sa $c2$ se može kretati na $c3$, $c1$, $b2$ i $d2$), ali ne može prolaziti kroz zidove (npr., ne može proći sa $c3$ na $c4$). Robby želi pronaći put s najmanjim utroškom baterije. Svaki potez Robbyja košta jednu energetsku jedinicu iz baterije, osim kada zakorači na polja $a3$, $c1$ ili $d3$, gdje je zbog energetskog polja utrošak baterije jednak 3 energetske jedinice. Primijenite algoritam A^*. Stanje odgovara Robbyjevoj poziciji, a cijena puta odgovara broju utrošenih energetskih jedinica. Za heuristiku koristite L1-udaljenost (Manhattan-udaljenost). U slučaju izjednačenja cijene, proširuje se čvor sa stanjem koje alfanumerički dolazi prvo (npr., $a2$ prije $b2$ prije $b3$). **Koliko čvorova će algoritam A^* ukupno proširiti?**



A 8 B 7 C 9 D 10

7 (P) Neka je S skup stanja te neka su s_1, s_2 ,s_3 i s_4 stanja iz S. Vrijedi $succ(s_1) = \{(s_2, 4)\}$ i $succ(s_2) = \{(s_3, c)\}$, gdje je c neka cijena prijelaza. Razmatramo dvije heuristike, h_1 i h_2. Heuristika h_1 definirana je u odnosu na ciljno stanje s_3, a heuristika h_2 u odnosu na ciljno stanje s_4. Vrijedi $h_1(s_1) = 11$ i $h_1(s_2) = 9$ te $h_2(s_1) = 20$, $h_2(s_2) = 16$ i $h_2(s_3) = 8$. Obje su heuristike optimistične. Međutim, na temelju ovih informacija heuristiku h_2 možemo unaprijediti tako da bude obavještenija (bliža stvarnoj cijeni). **Koje od navedenih vrijednosti za h_2 sigurno znamo da daju obavještenije, ali još uvijek optimističnu heuristiku?**

A $h_2(s_1) = 20$, $h_2(s_2) = 18$ C $h_2(s_1) = 21$, $h_2(s_2) = 17$
B $h_2(s_1) = 21$, $h_2(s_2) = 16$ D $h_2(s_1) = 20$, $h_2(s_2) = 17$

2. Igranje igara (3 pitanja)

8 (P) Alfa-beta podrezivanje može značajno smanjiti broj generiranih čvorova kod algoritma minimax. Ušteda ovisi o redoslijedu kojim se generiraju čvorovi. Optimalan redoslijed je onaj koji djecu čvora MAX generira silazno po minimax vrijednostima, a djecu čvora MIN uzlazno po minimax vrijednostima. Razmotite igru s dva moguća poteza u svakome stanju. Stablo se pretražuje do dubine 3. Vrijednosti heuristike igrača MAX za stanja u listovima stabla igre, kada su čvorovi-djeca generirani nasumičnim redoslijedom, su sljedeća (slijeva nadesno): $0, 3, 4, -3, 4, -1, 0, 2$. Na potezu je igrač MAX. Odredite broj čvorova koji će biti uklonjeni podrezivanjem alfa-beta. Zatim ponovite izračun, ali s optimalnim redoslijedom generiranja čvorova. **Koliko više čvorova će biti podrezano s optimalnim redoslijedom u odnosu na slučajan redoslijed generiranja čvorova?**

A 1 B 2 C 3 D 4

9 (R) Razmatramo igru dvaju minimax-igrača. Stablo igre definirano je prijelazima $succ(A) = \{B, C, D\}$, $succ(B) = \{E, F\}$, $succ(C) = \{G, H\}$, $succ(D) = \{I, J\}$, $succ(F) = succ(G) = \{K, L\}$, $succ(E) = succ(H) = succ(I) = \{M, N\}$ te $succ(J) = \{O, P\}$. Igrač MAX koristi heuristiku h_1, a igrač MIN heuristiku h_2. Vrijednosti heuristika su sljedeće (obje heuristike definiraju dobit dotičnog igrača):

	E	F	G	H	I	J	K	L	M	N	O	P
h_1	1	−2	3	2	3	5	−1	0	−2	−1	0	2
h_2	0	5	2	4	−1	2	3	2	−3	−2	1	−4

Oba igrača pretražuju najviše dva poteza unaprijed (jedan svoj i jedan suparnički). Igru započinje igrač MAX u stanju A. **Kroz koja stanja će se razvijati igra?**

A $A, D, I, \ldots$ B $A, C, G, \ldots$ C $A, D, J, \ldots$ D $A, C, H, \ldots$

10 (T) Igrači MAX i MIN primjenjuju algoritam minimax prije svakog svog poteza, pretražujući do neke unaprijed definirane dubine. Zamislite, međutim, da igrač MAX algoritam minimax primjenjuje samo jednom, i to na početku igre, a da zatim tako dobivene minimax vrijednosti koristi za sve svoje buduće poteze. **U kojem bi slučaju tako implementirani igrač MAX sigurno maksimizirao svoju minimalnu dobit?**

A Nema garancije da bi takav igrač maksimizirao svoju minimalnu dobit

B Ako koristi identičnu heuristiku kao igrač MIN

C Ako je pri izračunu minimax-vrijednosti dosegao sva završna stanja igre

D Ako je heuristika igrača MAX obavještenija od heuristike igrača MIN

3. Prikazivanje znanja i automatsko zaključivanje (6 pitanja)

11 (T) Propozicijska logika (PL) i logika prvog reda (FOL) dva su formalizma za prikaz znanja. **Što je prednost, a što nedostatak PL u odnosu na FOL?** (Podsjetnik: Problem je *traktabilan* ako je rješiv u polinomijalnom vremenu.)

A PL je ekspresivniji od FOL, ali je dokazivanje netraktabilno

B Dokazivanje u PL je odlučivo, ali nije traktabilno kao u FOL

C Dokazivanje u PL je odlučivo, ali je PL manje ekspresivan od FOL

D Dokazivanje u PL je traktabilno, ali je FOL poluodlučiv

12 (R) Zadane su premise: *"Bela laje (L) ako je gladna (G) ili ako je uštap (U) ili ako je preplašena (P). Ako nije uštap, Bela dobro spava (S). Kada previše jede (J), onda Bela ne spava dobro. Bela nije preplašena ili je gladna."* **Koja je od sljedećih tvrdnji logička posljedica ovih premisa?**

A *Bela laje ili ne jede previše.*

B *Ako Bela ne spava dobro, onda previše jede.*

C *Ako Bela laje, onda previše jede.*

D *Bela ne spava dobro.*

13 (P) Zadane su FOL klauzule $Q(f(x), b) \vee \neg P(y) \vee Q(x, y)$ i $P(z) \vee \neg Q(g(c), z) \vee P(a)$. **Koja od sljedećih klauzula je rezolventa ovih dviju klauzula?**

A $Q(g(c), a) \vee Q(f(x), b)$

B $Q(f(g(c)), b) \vee \neg P(a) \vee P(a)$

C $Q(f(x), b) \vee \neg Q(g(c), b) \vee P(a)$

D $Q(f(x), b) \vee Q(x, a) \vee P(y)$

14 (T) Neka je R skup pravila zaključivanja te neka su F i G dobro oblikovane formule logike. **Kada za skup R kažemo da *nije* potpun skup pravila zaključivanja?**

A Ako je formula $\neg F \vee G$ valjana, ali ne postoji lanac pravila u R kojim iz F možemo izvesti $\neg G$

B Ako je svaki model od F također model od G, ali primjenom pravila iz R na formule F ne možemo izvesti G

C Ako je formula $F \to G$ zadovoljiva, ali ne postoji pravilo u R kojim iz $F \wedge \neg G$ možemo izvesti NIL

D Ako je svaki model od G također model od F, ali postoji pravilo u R kojim iz F ne možemo izvesti G

15 (P) Pravila zaključivanja poželjno su i ispravna i potpuna. Razmotrite pravilo zaključivanja r definirano kao $(A \vee B) \to C, \neg B \vdash_r \neg(C \to A)$. **Kakvo je pravilo r?**

A Ispravno, ali nije potpuno

B Niti ispravno niti potpuno

C Potpuno, ali nije ispravno

D Ispravno i potpuno

16 (R) Zadane su premise: (1) *U svakom gradu postoji pošta*, (2) *Virovitica je grad*, (3) *Svi vole one koji su sretni*, (4) *Lucija je dijete u Virovitici*, (5) *Onaj koji nekoga/nešto voli je sretan*, (6) *Svako dijete voli svaku poštu.* Želimo dokazati da iz ovih premisa logički slijedi *U Virovitici postoji netko koga svi vole.* Pretvorite ove izjave u FOL formule koristeći predikate $U(x, y)$ za "x je u y", $V(x, y)$ za "x voli y", $P(x)$ za "x je pošta", $G(x)$ za "x je grad", $S(x)$ za "x je sretan" i $D(x)$ za "x je dijete". Zatim primijenite rezoluciju opovrgavanjem uz strategiju skupa potpore. **Koliko klauzula ulazi u rezolucijski postupak i koji je najmanji broj rezolucijskih koraka potrebnih za dokaz?** (1 rezolucijski korak = 1 primjena rezolucijskog pravila.)

A 9 klauzula, 5 koraka

B 8 klauzula, 6 koraka

C 9 klauzula, 7 koraka

D 8 klauzula, 8 koraka

4. Logičko programiranje i ekspertni sustavi (4 pitanja)

17 (R) Baza znanja ekspertnog sustava sadrži sljedeća pravila:

(1) AKO $(A = a_2) \wedge (D = d_2)$ ONDA $C = c_1$
(2) AKO $(F = f_3) \vee (B = b_3)$ ONDA $C = c_2$
(3) AKO $(E = e_1) \vee (B = b_1)$ ONDA $(A = a_1) \wedge (D = d_2)$
(4) AKO $F = f_1$ ONDA $D = d_2$
(5) AKO $F = f_2$ ONDA $E = e_2$
(6) AKO $(B = b_3) \vee (D = d_1)$ ONDA $(E = e_1) \wedge (A = a_2)$

Sustav koristimo za izvođenje vrijednosti varijable C ulančavanjem unazad. Prednost imaju pravila s manjim rednim brojem. Pravila koja su jednom palila više ne mogu paliti. Jednom izvedeni međuciljevi brišu se sa stoga,

neovisno o njihovoj poziciji na stogu. Na možebitne upite od strane sustava, korisnik odgovara sa $B = b_3$ i $F = f_1$. **Što radi ekspertni sustav pri izvođenju vrijednosti varijable C?**

A Izvodi $E = e_1$ te kasnije $E = e_2$
B Pali tri pravila i izvodi $C = c_2$
C Odbacuje pravilo 6 te kasnije pali pravilo 3
D Završava s četiri činjenice u radnoj memoriji

18 (T) Ekspertni sustavi koriste se za automatsko zaključivanje u specifičnim domenama ljudskog znanja. **Što je karakteristika ekspertnih sustava?**

A Domenski specifično znanje definirano je u obliku činjenica i ako-onda pravila pohranjenih u stroju za zaključivanje
B Međuciljevi koje treba dokazati pohranjuju se na stog, a u slučaju omogućenosti većeg broja pravila, prednost ima pravilo koje je navedeno posljednje
C Mogu zaključivati ulančavanjem unazad, što je prikladno kada postoji mnogo mogućih zaključaka, a malo početnih činjenica
D Baza znanja odvojena je od mehanizma zaključivanja, kako bi se isti mehanizam mogao koristiti s različitim bazama znanja

19 (T) Izvođenje programa u Prologu odgovara zaključivanju u Hornovoj klauzalnoj logici (HKL). HKL je podskup logike prvoga reda (FOL). **Zašto Prolog koristi HKL, a ne FOL?**

A Klauzule u HKL su zatvorene pod rezolucijom
B Rezolucija opovrgavanjem u HKL je vremenski učinkovita
C Zaključivanje u HKL uz dodatak negacije je odlučivo
D Rezolucije opovrgavanjem u HKL je ispravna i potpuna

20 (R) Baza znanja u Prologu modelira tramvajsku mrežu u Zagrebu. Baza sadrži sljedeće činjenice i pravila:

```
pruga(frankopanska, zrinjevac).
pruga(frankopanska, vodnikova).
pruga(zrinjevac, branimirova).
pruga(vodnikova, branimirova).
zastoj(zrinjevac).
prohodno(X, Y) :- pruga(X, Y), not(zastoj(X)).
povezano(X, Y) :- prohodno(X,Y); prohodno(Y, X).
dosezljivo(X, Y) :- povezano(X, Y).
dosezljivo(X, Y) :- povezano(X, Z), dosezljivo(Z, Y).
```

Nad ovako definiranom bazom znanja izvodimo upit `dosezljivo(branimirova, frankopanska)`. Prolog će za ovaj upit vratiti `True`. Nacrtajte Prologovo stablo dokaza za ovaj upit sve do mjesta gdje se postupak pretrage po četvrti put vraća na točku odabira. Pritom za dokazivanje negiranog cilja koristite dva čvora (jedan za `not(P(X))` i drugi za `P(X)`). **Koliko čvorova ima stablo dokaza do trenutka četvrtog povratka u postupku pretraživanja?**

A 13 B 15 C 9 D 11

Međuispit iz Uvoda u umjetnu inteligenciju (2023./2024.)

– 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 i heurističko pretraživanje (7 pitanja)

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

A Iterativno pretraživanje u dubinu
B Pretraživanje u širinu
C Pretraživanje u dubinu
D Ograničeno pretraživanje u dubinu ($k = 2$)

2 (T) Jedno od poželjnih svojstava heurističke funkcije je *konzistentnost* (ili *monotonost*). Razmotrite algoritam A^*. **Koja je prednost algoritma A^* s konzistentnom heuristikom u odnosu na A^* s nekonzistentnom ali optimističnom heuristikom?**

A Algoritam je potpun
B Algoritam ima manju prostornu složenost
C Algoritam ispituje manji broj čvorova
D Algoritam je optimalan

3 (P) Neka je S skup stanja te neka su s_1, s_2 ,s_3 i s_4 stanja iz S. Vrijedi $succ(s_1) = \{(s_2, 4)\}$ i $succ(s_2) = \{(s_3, c)\}$, gdje je c neka cijena prijelaza. Razmatramo dvije heuristike, h_1 i h_2. Heuristika h_1 definirana je u odnosu na ciljno stanje s_3, a heuristika h_2 u odnosu na ciljno stanje s_4. Vrijedi $h_1(s_1) = 11$ i $h_1(s_2) = 9$ te $h_2(s_1) = 20$, $h_2(s_2) = 16$ i $h_2(s_3) = 8$. Obje su heuristike optimistične. Međutim, na temelju ovih informacija heuristiku h_2 možemo unaprijediti tako da bude obavještenija (bliža stvarnoj cijeni). **Koje od navedenih vrijednosti za h_2 sigurno znamo da daju obavještenju, ali još uvijek optimističnu heuristiku?**

A $h_2(s_1) = 21$, $h_2(s_2) = 16$
B $h_2(s_1) = 20$, $h_2(s_2) = 17$
C $h_2(s_1) = 20$, $h_2(s_2) = 18$
D $h_2(s_1) = 21$, $h_2(s_2) = 17$

4 (R) Algoritmom slijepog pretraživanja rješavamo problem misionara i kanibala. Tri misionara i tri kanibala potrebno je jednim čamcem prevesti s jedne strane obale rijeke na drugu, pri čemu se ne smije dogoditi da se na jednoj strani nađu i kanibali i misionari, ali da je kanibala više nego misionara. Čamac može prevesti najviše dvije osobe i ne može ploviti prazan. Stanje prikazujemo kao trojku MKC, gdje su M i K broj misionara odnosno kanibala na lijevoj obali rijeke, a C je strana obale na kojoj se nalazi čamac. Npr., u stanju 32D na lijevoj obali su tri misionara i dva kanibala (jedan kanibal je onda na desnoj obali), a čamac je na desnoj obali. Početno stanje je 33L, a završno 00D. Funkcija sljedbenika generira sva legitimna stanja, uključivo i ona koja rezultiraju neuspjehom i nemaju sljedbenika, npr., stanje 23D. Koristimo algoritam iterativnog pretraživanja u dubinu (IDS) bez liste posjećenih stanja. Kod proširenja čvora, algoritam čvorove djecu generira prema numeričkom redoslijedu; npr., stanje 13D prethodi stanju 23D. **Koliko će čvorova algoritam IDS ispitati prije nego što proširi čvor sa stanjem** 31D?

A 14 B 16 C 18 D 11

5 (P) Rješavamo problem pretraživanja prostora stanja za koji vrijedi $m = 9$, $d = 6$ i $b = 3$. Koristimo pretraživanje u širinu (BFS) i pretraživanje u dubinu (DFS). Za oba algoritma razmotrite najgori slučaj vremenske složenosti. **Koliko će u takvom slučaju algoritam DFS generirati više čvorova od algoritma BFS?**

A 26208 B 26241 C 28431 D 28395

6 (T) Jedna od komponenti problema pretraživanja prostora stanja je funkcija sljedbenika *succ*. Razmotrite algoritam pretraživanja s jednolikom cijenom (UCS). **Kod tog algoritma, u što funkcija *succ* preslikava svako stanje?**

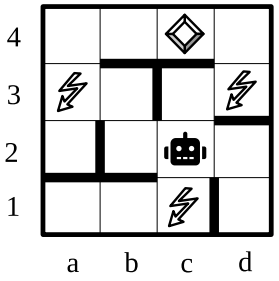
A Skup parova sačinjenih od početnog i završnog stanja

B Kartezijev umnožak skupa stanja i skupa nenegativnih brojeva

C Skup svih podskupova stanja (moguće prazan)

D Skup parova sačinjenih od stanja i pozitivnog broja

7 (R) Na slici desno prikazana je mapa dvodimenzijskoga svijeta od 16 polja. Robot Robby početno se nalazi na polju $c2$ i treba doći na polje $c4$ na kojemu se nalazi dijamant. Robby se može kretati prema gore, dolje, lijevo i desno (npr., sa $c2$ se može kretati na $c3$, $c1$, $b2$ i $d2$), ali ne može prolaziti kroz zidove (npr., ne može proći sa $c3$ na $c4$). Robby želi pronaći put s najmanjim utroškom baterije. Svaki potez Robbyja košta jednu energetsku jedinicu iz baterije, osim kada zakorači na polja $a3$, $c1$ ili $d3$, gdje je zbog energetskog polja utrošak baterije jednak 3 energetske jedinice. Primijenite algoritam A^*. Stanje odgovara Robbyjevoj poziciji, a cijena puta odgovara broju utrošenih energetskih jedinica. Za heuristiku koristite L1-udaljenost (Manhattan-udaljenost). U slučaju izjednačenja cijene, proširuje se čvor sa stanjem koje alfanumerički dolazi prvo (npr., $a2$ prije $b2$ prije $b3$). **Koliko čvorova će algoritam A^* ukupno proširiti?**



A 7 B 10 C 8 D 9

2. Igranje igara (3 pitanja)

8 (P) Alfa-beta podrezivanje može značajno smanjiti broj generiranih čvorova kod algoritma minimax. Ušteda ovisi o redoslijedu kojim se generiraju čvorovi. Optimalan redoslijed je onaj koji djecu čvora MAX generira silazno po minimax vrijednostima, a djecu čvora MIN uzlazno po minimax vrijednostima. Razmotite igru s dva moguća poteza u svakome stanju. Stablo se pretražuje do dubine 3. Vrijednosti heuristike igrača MAX za stanja u listovima stabla igre, kada su čvorovi-djeca generirani nasumičnim redoslijedom, su sljedeća (slijeva nadesno): $0, 3, 4, -3, 4, -1, 0, 2$. Na potezu je igrač MAX. Odredite broj čvorova koji će biti uklonjeni podrezivanjem alfa-beta. Zatim ponovite izračun, ali s optimalnim redoslijedom generiranja čvorova. **Koliko više čvorova će biti podrezano s optimalnim redoslijedom u odnosu na slučajan redoslijed generiranja čvorova?**

A 4 B 3 C 1 D 2

9 (R) Razmatramo igru dvaju minimax-igrača. Stablo igre definirano je prijelazima $succ(A) = \{B, C, D\}$, $succ(B) = \{E, F\}$, $succ(C) = \{G, H\}$, $succ(D) = \{I, J\}$, $succ(F) = succ(G) = \{K, L\}$, $succ(E) = succ(H) = succ(I) = \{M, N\}$ te $succ(J) = \{O, P\}$. Igrač MAX koristi heuristiku h_1, a igrač MIN heuristiku h_2. Vrijednosti heuristika su sljedeće (obje heuristike definiraju dobit dotičnog igrača):

	E	F	G	H	I	J	K	L	M	N	O	P
h_1	1	−2	3	2	3	5	−1	0	−2	−1	0	2
h_2	0	5	2	4	−1	2	3	2	−3	−2	1	−4

Oba igrača pretražuju najviše dva poteza unaprijed (jedan svoj i jedan suparnički). Igru započinje igrač MAX u stanju A. **Kroz koja stanja će se razvijati igra?**

A $A, D, J, \ldots$ B $A, C, G, \ldots$ C $A, C, H, \ldots$ D $A, D, I, \ldots$

10 (T) Igrači MAX i MIN primjenjuju algoritam minimax prije svakog svog poteza, pretražujući do neke unaprijed definirane dubine. Zamislite, međutim, da igrač MAX algoritam minimax primjenjuje samo jednom, i to na početku igre, a da zatim tako dobivene minimax vrijednosti koristi za sve svoje buduće poteze. **U kojem bi slučaju tako implementirani igrač MAX sigurno maksimizirao svoju minimalnu dobit?**

A Ako koristi identičnu heuristiku kao igrač MIN

B Nema garancije da bi takav igrač maksimizirao svoju minimalnu dobit

C Ako je pri izračunu minimax-vrijednosti dosegao sva završna stanja igre

D Ako je heuristika igrača MAX obavještenija od heuristike igrača MIN

3. Prikazivanje znanja i automatsko zaključivanje (6 pitanja)

11 (P) Pravila zaključivanja poželjno su i ispravna i potpuna. Razmotrite pravilo zaključivanja r definirano kao $(A \vee B) \rightarrow C, \neg B \vdash_r A \rightarrow C$. **Kakvo je pravilo r?**

A Ispravno, ali nije potpuno
B Ispravno i potpuno
C Niti ispravno niti potpuno
D Potpuno, ali nije ispravno

12 (T) Propozicijska logika (PL) i logika prvog reda (FOL) dva su formalizma za prikaz znanja. **Što je prednost, a što nedostatak PL u odnosu na FOL?** (Podsjetnik: Problem je *traktabilan* ako je rješiv u polinomijalnom vremenu.)

A Dokazivanje u PL je odlučivo, ali je PL manje ekspresivan od FOL
B Dokazivanje u PL je traktabilno, ali je FOL poluodlučiv
C Dokazivanje u PL je odlučivo, ali nije traktabilno kao u FOL
D PL je ekspresivniji od FOL, ali je dokazivanje netraktabilno

13 (T) Neka je R skup pravila zaključivanja te neka su F i G dobro oblikovane formule logike. **Kada za skup R kažemo da *nije* potpun skup pravila zaključivanja?**

A Ako je svaki model od G također model od F, ali postoji pravilo u R kojim iz F ne možemo izvesti G
B Ako je formula $\neg F \vee G$ valjana, ali ne postoji lanac pravila u R kojim iz F možemo izvesti $\neg G$
C Ako je formula $F \rightarrow G$ zadovoljiva, ali ne postoji pravilo u R kojim iz $F \wedge \neg G$ možemo izvesti NIL
D Ako je svaki model od F također model od G, ali primjenom pravila iz R na formule F ne možemo izvesti G

14 (P) Zadane su FOL klauzule $Q(f(x), b) \vee \neg P(y) \vee Q(x, y)$ i $P(z) \vee \neg Q(g(c), z) \vee P(a)$. **Koja od sljedećih klauzula je rezolventa ovih dviju klauzula?**

A $Q(g(c), a) \vee Q(f(x), b)$
B $Q(f(g(c)), b) \vee \neg P(a) \vee P(a)$
C $Q(f(x), b) \vee Q(x, a) \vee P(y)$
D $Q(f(x), b) \vee \neg Q(g(c), b) \vee P(a)$

15 (R) Zadane su premise: (1) *U svakom gradu postoji pošta*, (2) *Virovitica je grad*, (3) *Svi vole one koji su sretni*, (4) *Lucija je dijete u Virovitici*, (5) *Onaj koji nekoga/nešto voli je sretan*, (6) *Svako dijete voli svaku poštu.* Želimo dokazati da iz ovih premisa logički slijedi *U Virovitici postoji netko koga svi vole.* Pretvorite ove izjave u FOL formule koristeći predikate $U(x, y)$ za "x je u y", $V(x, y)$ za "x voli y", $P(x)$ za "x je pošta", $G(x)$ za "x je grad", $S(x)$ za "x je sretan" i $D(x)$ za "x je dijete". Zatim primijenite rezoluciju opovrgavanjem uz strategiju skupa potpore. **Koliko klauzula ulazi u rezolucijski postupak i koji je najmanji broj rezolucijskih koraka potrebnih za dokaz?** (1 rezolucijski korak = 1 primjena rezolucijskog pravila.)

A 9 klauzula, 5 koraka
B 8 klauzula, 8 koraka
C 8 klauzula, 6 koraka
D 9 klauzula, 7 koraka

16 (R) Zadane su premise: *"Bela laje (L) ako je gladna (G) ili ako je uštap (U) ili ako je preplašena (P). Ako nije uštap, Bela dobro spava (S). Kada previše jede (J), onda Bela ne spava dobro. Bela nije preplašena ili je gladna."* **Koja je od sljedećih tvrdnji logička posljedica ovih premisa?**

A *Ako Bela laje, onda ne spava dobro.*
B *Ako Bela laje, onda previše jede.*
C *Bela ne spava dobro.*
D *Bela laje ako previše jede.*

4. Logičko programiranje i ekspertni sustavi (4 pitanja)

17 (R) Baza znanja ekspertnog sustava sadrži sljedeća pravila:

(1) AKO $(A = a_2) \wedge (D = d_2)$ ONDA $C = c_1$
(2) AKO $(F = f_3) \vee (B = b_3)$ ONDA $C = c_2$
(3) AKO $(E = e_1) \vee (B = b_1)$ ONDA $(A = a_1) \wedge (D = d_2)$
(4) AKO $F = f_1$ ONDA $D = d_2$
(5) AKO $F = f_2$ ONDA $E = e_2$
(6) AKO $(B = b_3) \vee (D = d_1)$ ONDA $(E = e_1) \wedge (A = a_2)$

Sustav koristimo za izvođenje vrijednosti varijable C ulančavanjem unazad. Prednost imaju pravila s manjim rednim brojem. Pravila koja su jednom palila više ne mogu paliti. Jednom izvedeni međuciljevi brišu se sa stoga,

neovisno o njihovoj poziciji na stogu. Na možebitne upite od strane sustava, korisnik odgovara sa $B = b_3$ i $F = f_1$. **Što radi ekspertni sustav pri izvođenju vrijednosti varijable C?**

A Odbacuje pravilo 2 te kasnije pali pravilo 5

B Odbacuje pravilo 5 te kasnije pali pravilo 6

C Završava s tri činjenice u radnoj memoriji

D Odbacuje pravilo 6 te kasnije pali pravilo 3

18 (T) Ekspertni sustavi koriste se za automatsko zaključivanje u specifičnim domenama ljudskog znanja. **Što je karakteristika ekspertnih sustava?**

A Baza znanja odvojena je od mehanizma zaključivanja, kako bi se isti mehanizam mogao koristiti s različitim bazama znanja

B Međuciljevi koje treba dokazati pohranjuju se na stog, a u slučaju omogućenosti većeg broja pravila, prednost ima pravilo koje je navedeno posljednje

C Mogu zaključivati ulančavanjem unazad, što je prikladno kada postoji mnogo mogućih zaključaka, a malo početnih činjenica

D Domenski specifično znanje definirano je u obliku činjenica i ako-onda pravila pohranjenih u stroju za zaključivanje

19 (T) Izvođenje programa u Prologu odgovara zaključivanju u Hornovoj klauzalnoj logici (HKL). HKL je podskup logike prvoga reda (FOL). **Zašto Prolog koristi HKL, a ne FOL?**

A Rezolucija opovrgavanjem u HKL je vremenski učinkovita

B Rezolucije opovrgavanjem u HKL je ispravna i potpuna

C Zaključivanje u HKL uz dodatak negacije je odlučivo

D Klauzule u HKL su zatvorene pod rezolucijom

20 (R) Baza znanja u Prologu modelira tramvajsku mrežu u Zagrebu. Baza sadrži sljedeće činjenice i pravila:

```
pruga(frankopanska, zrinjevac).
pruga(frankopanska, vodnikova).
pruga(zrinjevac, branimirova).
pruga(vodnikova, branimirova).
zastoj(zrinjevac).
prohodno(X, Y) :- pruga(X, Y), not(zastoj(X)).
povezano(X, Y) :- prohodno(X,Y); prohodno(Y, X).
dosezljivo(X, Y) :- povezano(X, Y).
dosezljivo(X, Y) :- povezano(X, Z), dosezljivo(Z, Y).
```

Nad ovako definiranom bazom znanja izvodimo upit `dosezljivo(branimirova, frankopanska)`. Prolog će za ovaj upit vratiti `True`. Nacrtajte Prologovo stablo dokaza za ovaj upit sve do mjesta gdje se postupak pretrage po četvrti put vraća na točku odabira. Pritom za dokazivanje negiranog cilja koristite dva čvora (jedan za `not(P(X))` i drugi za `P(X)`). **Koliko čvorova ima stablo dokaza do trenutka četvrtog povratka u postupku pretraživanja?**

A 13 B 15 C 9 D 11

Međuispit iz Uvoda u umjetnu inteligenciju (2023./2024.)

– 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 i heurističko pretraživanje (7 pitanja)

1 (T) Jedno od poželjnih svojstava heurističke funkcije je *konzistentnost* (ili *monotonost*). Razmotrite algoritam A^*. **Koja je prednost algoritma A^* s konzistentnom heuristikom u odnosu na A^* s nekonzistentnom ali optimističnom heuristikom?**

A Algoritam je optimalan
B Algoritam ispituje manji broj čvorova
C Algoritam ima manju prostornu složenost
D Algoritam je potpun

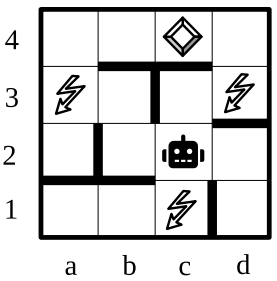
2 (T) Jedna od komponenti problema pretraživanja prostora stanja je funkcija sljedbenika *succ*. Razmotrite algoritam pretraživanja s jednolikom cijenom (UCS). **Kod tog algoritma, u što funkcija *succ* preslikava svako stanje?**

A Kartezijev umnožak skupa stanja i skupa nenegativnih brojeva
B Skup parova sačinjenih od početnog i završnog stanja
C Skup parova sačinjenih od stanja i pozitivnog broja
D Skup svih podskupova stanja (moguće prazan)

3 (P) Neka je S skup stanja te neka su s_1, s_2 ,s_3 i s_4 stanja iz S. Vrijedi $succ(s_1) = \{(s_2, 4)\}$ i $succ(s_2) = \{(s_3, c)\}$, gdje je c neka cijena prijelaza. Razmatramo dvije heuristike, h_1 i h_2. Heuristika h_1 definirana je u odnosu na ciljno stanje s_3, a heuristika h_2 u odnosu na ciljno stanje s_4. Vrijedi $h_1(s_1) = 11$ i $h_1(s_2) = 9$ te $h_2(s_1) = 20$, $h_2(s_2) = 16$ i $h_2(s_3) = 8$. Obje su heuristike optimistične. Međutim, na temelju ovih informacija heuristiku h_2 možemo unaprijediti tako da bude obavještenija (bliža stvarnoj cijeni). **Koje od navedenih vrijednosti za h_2 sigurno znamo da daju obavještenju, ali još uvijek optimističnu heuristiku?**

A $h_2(s_1) = 20$, $h_2(s_2) = 17$
B $h_2(s_1) = 21$, $h_2(s_2) = 16$
C $h_2(s_1) = 20$, $h_2(s_2) = 18$
D $h_2(s_1) = 21$, $h_2(s_2) = 17$

4 (R) Na slici desno prikazana je mapa dvodimenzijskoga svijeta od 16 polja. Robot Robby početno se nalazi na polju $c2$ i treba doći na polje $c4$ na kojemu se nalazi dijamant. Robby se može kretati prema gore, dolje, lijevo i desno (npr., sa $c2$ se može kretati na $c3$, $c1$, $b2$ i $d2$), ali ne može prolaziti kroz zidove (npr., ne može proći sa $c3$ na $c4$). Robby želi pronaći put s najmanjim utroškom baterije. Svaki potez Robbyja košta jednu energetsku jedinicu iz baterije, osim kada zakorači na polja $a3$, $c1$ ili $d3$, gdje je zbog energetskog polja utrošak baterije jednak 3 energetske jedinice. Primijenite algoritam A^*. Stanje odgovara Robbyjevoj poziciji, a cijena puta odgovara broju utrošenih energetskih jedinica. Za heuristiku koristite L1-udaljenost (Manhattan-udaljenost). U slučaju izjednačenja cijene, proširuje se čvor sa stanjem koje alfanumerički dolazi prvo (npr., $a2$ prije $b2$ prije $b3$). **Koliko čvorova će algoritam A^* ukupno proširiti?**



A 7 B 9 C 10 D 8

5 (P) Rješavamo problem pretraživanja prostora stanja za koji vrijedi $m = 10$, $d = 7$ i $b = 3$. Koristimo pretraživanje u širinu (BFS) i pretraživanje u dubinu (DFS). Za oba algoritma razmotrite najgori slučaj vremenske složenosti. **Koliko će u takvom slučaju algoritam DFS generirati više čvorova od algoritma BFS?**

A 85293 B 85257 C 78729 D 78696

6 (R) Algoritmom slijepog pretraživanja rješavamo problem misionara i kanibala. Tri misionara i tri kanibala potrebno je jednim čamcem prevesti s jedne strane obale rijeke na drugu, pri čemu se ne smije dogoditi da se na jednoj strani nađu i kanibali i misionari, ali da je kanibala više nego misionara. Čamac može prevesti najviše dvije osobe i ne može ploviti prazan. Stanje prikazujemo kao trojku MKC, gdje su M i K broj misionara odnosno kanibala na lijevoj obali rijeke, a C je strana obale na kojoj se nalazi čamac. Npr., u stanju 32D na lijevoj obali su tri misionara i dva kanibala (jedan kanibal je onda na desnoj obali), a čamac je na desnoj obali. Početno stanje je 33L, a završno 00D. Funkcija sljedbenika generira sva legitimna stanja, uključivo i ona koja rezultiraju neuspjehom i nemaju sljedbenika, npr., stanje 23D. Koristimo algoritam iterativnog pretraživanja u dubinu (IDS) bez liste posjećenih stanja. Kod proširenja čvora, algoritam čvorove djecu generira prema numeričkom redoslijedu; npr., stanje 13D prethodi stanju 23D. **Koliko će čvorova algoritam IDS ispitati prije nego što proširi čvor sa stanjem** 31D?

A 18 B 14 C 16 D 11

7 (P) 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** $\ldots, b, c, a, b, c, \ldots$ **?**

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

2. Igranje igara (3 pitanja)

8 (P) Alfa-beta podrezivanje može značajno smanjiti broj generiranih čvorova kod algoritma minimax. Ušteda ovisi o redoslijedu kojim se generiraju čvorovi. Optimalan redoslijed je onaj koji djecu čvora MAX generira silazno po minimax vrijednostima, a djecu čvora MIN uzlazno po minimax vrijednostima. Razmotite igru s dva moguća poteza u svakome stanju. Stablo se pretražuje do dubine 3. Vrijednosti heuristike igrača MAX za stanja u listovima stabla igre, kada su čvorovi-djeca generirani nasumičnim redoslijedom, su sljedeća (slijeva nadesno): $0, 3, -3, 4, 4, -1, 0, 2$. Na potezu je igrač MAX. Odredite broj čvorova koji će biti uklonjeni podrezivanjem alfa-beta. Zatim ponovite izračun, ali s optimalnim redoslijedom generiranja čvorova. **Koliko više čvorova će biti podrezano s optimalnim redoslijedom u odnosu na slučajan redoslijed generiranja čvorova?**

A 4 B 2 C 1 D 3

9 (R) Razmatramo igru dvaju minimax-igrača. Stablo igre definirano je prijelazima $succ(A) = \{B, C, D\}$, $succ(B) = \{E, F\}$, $succ(C) = \{G, H\}$, $succ(D) = \{I, J\}$, $succ(F) = succ(G) = \{K, L\}$, $succ(E) = succ(H) = succ(I) = \{M, N\}$ te $succ(J) = \{O, P\}$. Igrač MAX koristi heuristiku h_1, a igrač MIN heuristiku h_2. Vrijednosti heuristika su sljedeće (obje heuristike definiraju dobit dotičnog igrača):

	E	F	G	H	I	J	K	L	M	N	O	P
h_1	1	−2	3	2	3	5	−1	0	−2	−1	0	2
h_2	0	5	2	4	−1	2	3	2	1	−4	−3	−2

Oba igrača pretražuju najviše dva poteza unaprijed (jedan svoj i jedan suparnički). Igru započinje igrač MAX u stanju A. **Kroz koja stanja će se razvijati igra?**

A $A, D, J, \ldots$ B $A, C, G, \ldots$ C $A, C, H, \ldots$ D $A, D, I, \ldots$

10 (T) Igrači MAX i MIN primjenjuju algoritam minimax prije svakog svog poteza, pretražujući do neke unaprijed definirane dubine. Zamislite, međutim, da igrač MAX algoritam minimax primjenjuje samo jednom, i to na početku igre, a da zatim tako dobivene minimax vrijednosti koristi za sve svoje buduće poteze. **U kojem bi slučaju tako implementirani igrač MAX sigurno maksimizirao svoju minimalnu dobit?**

A Ako je heuristika igrača MAX obavještenija od heuristike igrača MIN
B Ako koristi identičnu heuristiku kao igrač MIN
C Ako je pri izračunu minimax-vrijednosti dosegao sva završna stanja igre
D Nema garancije da bi takav igrač maksimizirao svoju minimalnu dobit

3. Prikazivanje znanja i automatsko zaključivanje (6 pitanja)

11 (P) Pravila zaključivanja poželjno su i ispravna i potpuna. Razmotrite pravilo zaključivanja r definirano kao $(A \vee B) \rightarrow C, \neg B \vdash_r \neg(C \rightarrow A)$. **Kakvo je pravilo r?**

A Potpuno, ali nije ispravno
C Ispravno, ali nije potpuno
B Ispravno i potpuno
D Niti ispravno niti potpuno

12 (T) Propozicijska logika (PL) i logika prvog reda (FOL) dva su formalizma za prikaz znanja. **Što je prednost, a što nedostatak PL u odnosu na FOL?** (Podsjetnik: Problem je *traktabilan* ako je rješiv u polinomijalnom vremenu.)

A PL je ekspresivniji od FOL, ali je dokazivanje netraktabilno
B Dokazivanje u PL je odlučivo, ali nije traktabilno kao u FOL
C Dokazivanje u PL je odlučivo, ali je PL manje ekspresivan od FOL
D Dokazivanje u PL je traktabilno, ali je FOL poluodlučiv

13 (T) Neka je R skup pravila zaključivanja te neka su F i G dobro oblikovane formule logike. **Kada za skup R kažemo da *nije* potpun skup pravila zaključivanja?**

A Ako je svaki model od G također model od F, ali postoji pravilo u R kojim iz F ne možemo izvesti G
B Ako je formula $\neg F \vee G$ valjana, ali ne postoji lanac pravila u R kojim iz F možemo izvesti $\neg G$
C Ako je svaki model od F također model od G, ali primjenom pravila iz R na formule F ne možemo izvesti G
D Ako je formula $F \rightarrow G$ zadovoljiva, ali ne postoji pravilo u R kojim iz $F \wedge \neg G$ možemo izvesti NIL

14 (P) Zadane su FOL klauzule $Q(f(x), b) \vee \neg P(y) \vee Q(x, y)$ i $P(z) \vee \neg Q(g(c), z) \vee P(a)$. **Koja od sljedećih klauzula je rezolventa ovih dviju klauzula?**

A $Q(f(g(c)), b) \vee \neg P(a) \vee P(a)$
C $Q(f(x), b) \vee Q(x, a) \vee P(y)$
B $Q(f(x), b) \vee \neg Q(g(c), b) \vee P(a)$
D $Q(g(c), a) \vee Q(f(x), b)$

15 (R) Zadane su premise: (1) *U svakom gradu postoji pošta*, (2) *Virovitica je grad*, (3) *Svi vole one koji su sretni*, (4) *Lucija je dijete u Virovitici*, (5) *Onaj koji nekoga/nešto voli je sretan*, (6) *Svako dijete voli svaku poštu.* Želimo dokazati da iz ovih premisa logički slijedi *U Virovitici postoji netko koga svi vole.* Pretvorite ove izjave u FOL formule koristeći predikate $U(x, y)$ za "x je u y", $V(x, y)$ za "x voli y", $P(x)$ za "x je pošta", $G(x)$ za "x je grad", $S(x)$ za "x je sretan" i $D(x)$ za "x je dijete". Zatim primijenite rezoluciju opovrgavanjem uz strategiju skupa potpore. **Koliko klauzula ulazi u rezolucijski postupak i koji je najmanji broj rezolucijskih koraka potrebnih za dokaz?** (1 rezolucijski korak = 1 primjena rezolucijskog pravila.)

A 9 klauzula, 5 koraka
C 8 klauzula, 6 koraka
B 9 klauzula, 7 koraka
D 8 klauzula, 8 koraka

16 (R) Zadane su premise: *"Bela laje (L) ako je gladna (G) ili ako je uštap (U) ili ako je preplašena (P). Ako nije uštap, Bela dobro spava (S). Kada previše jede (J), onda Bela ne spava dobro. Bela nije preplašena ili je gladna."* **Koja je od sljedećih tvrdnji logička posljedica ovih premisa?**

A *Ako Bela laje, onda ne spava dobro.*
C *Bela laje ili ne jede previše.*
B *Ako Bela ne spava dobro, onda previše jede.*
D *Bela ne spava dobro.*

4. Logičko programiranje i ekspertni sustavi (4 pitanja)

17 (R) Baza znanja u Prologu modelira tramvajsku mrežu u Zagrebu. Baza sadrži sljedeće činjenice i pravila:

```
pruga(frankopanska, zrinjevac).
pruga(frankopanska, vodnikova).
pruga(zrinjevac, branimirova).
pruga(vodnikova, branimirova).
```

```
zastoj(zrinjevac).
prohodno(X, Y) :- pruga(X, Y), not(zastoj(X)).
povezano(X, Y) :- prohodno(X,Y); prohodno(Y, X).
dosezljivo(X, Y) :- povezano(X, Y).
dosezljivo(X, Y) :- povezano(X, Z), dosezljivo(Z, Y).
```

Nad ovako definiranom bazom znanja izvodimo upit `dosezljivo(branimirova, frankopanska)`. Prolog će za ovaj upit vratiti `True`. Nacrtajte Prologovo stablo dokaza za ovaj upit sve do mjesta gdje se postupak pretrage po četvrti put vraća na točku odabira. Pritom za dokazivanje negiranog cilja koristite dva čvora (jedan za `not(P(X))` i drugi za `P(X)`)). **Koliko čvorova ima stablo dokaza do trenutka četvrtog povratka u postupku pretraživanja?**

A 9 B 15 C 11 D 13

18 (T) Izvođenje programa u Prologu odgovara zaključivanju u Hornovoj klauzalnoj logici (HKL). HKL je podskup logike prvoga reda (FOL). **Zašto Prolog koristi HKL, a ne FOL?**

A Rezolucija opovrgavanjem u HKL je vremenski učinkovita

B Klauzule u HKL su zatvorene pod rezolucijom

C Zaključivanje u HKL uz dodatak negacije je odlučivo

D Rezolucije opovrgavanjem u HKL je ispravna i potpuna

19 (T) Ekspertni sustavi koriste se za automatsko zaključivanje u specifičnim domenama ljudskog znanja. **Što je karakteristika ekspertnih sustava?**

A Domenski specifično znanje definirano je u obliku činjenica i ako-onda pravila pohranjenih u stroju za zaključivanje

B Mogu zaključivati ulančavanjem unazad, što je prikladno kada postoji mnogo mogućih zaključaka, a malo početnih činjenica

C Međuciljevi koje treba dokazati pohranjuju se na stog, a u slučaju omogućenosti većeg broja pravila, prednost ima pravilo koje je navedeno posljednje

D Baza znanja odvojena je od mehanizma zaključivanja, kako bi se isti mehanizam mogao koristiti s različitim bazama znanja

20 (R) Baza znanja ekspertnog sustava sadrži sljedeća pravila:

(1) AKO $(A = a_2) \wedge (D = d_2)$ ONDA $C = c_1$
(2) AKO $(F = f_3) \vee (B = b_3)$ ONDA $C = c_2$
(3) AKO $(E = e_1) \vee (B = b_1)$ ONDA $(A = a_1) \wedge (D = d_2)$
(4) AKO $F = f_1$ ONDA $D = d_2$
(5) AKO $F = f_2$ ONDA $E = e_2$
(6) AKO $(B = b_3) \vee (D = d_1)$ ONDA $(E = e_1) \wedge (A = a_2)$

Sustav koristimo za izvođenje vrijednosti varijable C ulančavanjem unazad. Prednost imaju pravila s manjim rednim brojem. Pravila koja su jednom palila više ne mogu paliti. Jednom izvedeni međuciljevi brišu se sa stoga, neovisno o njihovoj poziciji na stogu. Na možebitne upite od strane sustava, korisnik odgovara sa $B = b_3$ i $F = f_1$. **Što radi ekspertni sustav pri izvođenju vrijednosti varijable C?**

A Odbacuje pravilo 6 te kasnije pali pravilo 3
B Pali četiri pravila i izvodi $C = c_1$
C Izvodi $E = e_1$ te kasnije $E = e_2$
D Izvodi $A = a_2$ te kasnije $A = a_1$

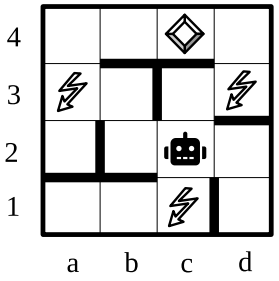
Međuispit iz Uvoda u umjetnu inteligenciju (2023./2024.)

– 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 i heurističko pretraživanje (7 pitanja)

1 (R) Na slici desno prikazana je mapa dvodimenzijskoga svijeta od 16 polja. Robot Robby početno se nalazi na polju $c2$ i treba doći na polje $c4$ na kojemu se nalazi dijamant. Robby se može kretati prema gore, dolje, lijevo i desno (npr., sa $c2$ se može kretati na $c3$, $c1$, $b2$ i $d2$), ali ne može prolaziti kroz zidove (npr., ne može proći sa $c3$ na $c4$). Robby želi pronaći put s najmanjim utroškom baterije. Svaki potez Robbyja košta jednu energetsku jedinicu iz baterije, osim kada zakorači na polja $a3$, $c1$ ili $d3$, gdje je zbog energetskog polja utrošak baterije jednak 3 energetske jedinice. Primijenite algoritam A^*. Stanje odgovara Robbyjevoj poziciji, a cijena puta odgovara broju utrošenih energetskih jedinica. Za heuristiku koristite L1-udaljenost (Manhattan-udaljenost). U slučaju izjednačenja cijene, proširuje se čvor sa stanjem koje alfanumerički dolazi prvo (npr., $a2$ prije $b2$ prije $b3$). **Koliko čvorova će algoritam A^* ukupno proširiti?**



A 10 B 8 C 9 D 7

2 (P) 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 $\ldots, b, c, d, a, b, \ldots$?**

A Pretraživanje u dubinu
B Ograničeno pretraživanje u dubinu ($k = 2$)
C Iterativno pretraživanje u dubinu
D Pretraživanje u širinu

3 (T) Jedno od poželjnih svojstava heurističke funkcije je *konzistentnost* (ili *monotonost*). Razmotrite algoritam A^*. **Koja je prednost algoritma A^* s konzistentnom heuristikom u odnosu na A^* s nekonzistentnom ali optimističnom heuristikom?**

A Algoritam je potpun
B Algoritam je optimalan
C Algoritam ispituje manji broj čvorova
D Algoritam ima manju prostornu složenost

4 (T) Jedna od komponenti problema pretraživanja prostora stanja je funkcija sljedbenika *succ*. Razmotrite algoritam pretraživanja s jednolikom cijenom (UCS). **Kod tog algoritma, u što funkcija *succ* preslikava svako stanje?**

A Skup svih podskupova stanja (moguće prazan)
B Skup parova sačinjenih od početnog i završnog stanja
C Kartezijev umnožak skupa stanja i skupa nenegativnih brojeva
D Skup parova sačinjenih od stanja i pozitivnog broja

5 (R) Algoritmom slijepog pretraživanja rješavamo problem misionara i kanibala. Tri misionara i tri kanibala potrebno je jednim čamcem prevesti s jedne strane obale rijeke na drugu, pri čemu se ne smije dogoditi da se na jednoj strani nađu i kanibali i misionari, ali da je kanibala više nego misionara. Čamac može prevesti najviše dvije osobe i ne može ploviti prazan. Stanje prikazujemo kao trojku MKC, gdje su M i K broj misionara odnosno kanibala na lijevoj obali rijeke, a C je strana obale na kojoj se nalazi čamac. Npr., u stanju 32D na lijevoj obali su tri misionara i dva kanibala (jedan kanibal je onda na desnoj obali), a čamac je na desnoj obali. Početno stanje je 33L, a završno 00D. Funkcija sljedbenika generira sva legitimna stanja, uključivo i ona koja rezultiraju

neuspjehom i nemaju sljedbenika, npr., stanje 23D. Koristimo algoritam iterativnog pretraživanja u dubinu (IDS) bez liste posjećenih stanja. Kod proširenja čvora, algoritam čvorove djecu generira prema numeričkom redoslijedu; npr., stanje 13D prethodi stanju 23D. **Koliko će čvorova algoritam IDS ispitati prije nego što proširi čvor sa stanjem** 31D?

A 16 B 14 C 18 D 11

6 (P) Rješavamo problem pretraživanja prostora stanja za koji vrijedi $m = 10$, $d = 7$ i $b = 4$. Koristimo pretraživanje u širinu (BFS) i pretraživanje u dubinu (DFS). Za oba algoritma razmotrite najgori slučaj vremenske složenosti. **Koliko će u takvom slučaju algoritam DFS generirati više čvorova od algoritma BFS?**

A 1310640 B 1376256 C 1376176 D 1310716

7 (P) Neka je S skup stanja te neka su s_1, s_2 ,s_3 i s_4 stanja iz S. Vrijedi $succ(s_1) = \{(s_2, 4)\}$ i $succ(s_2) = \{(s_3, c)\}$, gdje je c neka cijena prijelaza. Razmatramo dvije heuristike, h_1 i h_2. Heuristika h_1 definirana je u odnosu na ciljno stanje s_3, a heuristika h_2 u odnosu na ciljno stanje s_4. Vrijedi $h_1(s_1) = 11$ i $h_1(s_2) = 9$ te $h_2(s_1) = 20$, $h_2(s_2) = 16$ i $h_2(s_3) = 8$. Obje su heuristike optimistične. Međutim, na temelju ovih informacija heuristiku h_2 možemo unaprijediti tako da bude obavještenija (bliža stvarnoj cijeni). **Koje od navedenih vrijednosti za h_2 sigurno znamo da daju obavještenije, ali još uvijek optimističnu heuristiku?**

A $h_2(s_1) = 20$, $h_2(s_2) = 18$ C $h_2(s_1) = 20$, $h_2(s_2) = 17$
B $h_2(s_1) = 21$, $h_2(s_2) = 17$ D $h_2(s_1) = 21$, $h_2(s_2) = 16$

2. Igranje igara (3 pitanja)

8 (T) Igrači MAX i MIN primjenjuju algoritam minimax prije svakog svog poteza, pretražujući do neke unaprijed definirane dubine. Zamislite, međutim, da igrač MAX algoritam minimax primjenjuje samo jednom, i to na početku igre, a da zatim tako dobivene minimax vrijednosti koristi za sve svoje buduće poteze. **U kojem bi slučaju tako implementirani igrač MAX sigurno maksimizirao svoju minimalnu dobit?**

A Ako koristi identičnu heuristiku kao igrač MIN

B Ako je pri izračunu minimax-vrijednosti dosegao sva završna stanja igre

C Nema garancije da bi takav igrač maksimizirao svoju minimalnu dobit

D Ako je heuristika igrača MAX obavještenija od heuristike igrača MIN

9 (P) Alfa-beta podrezivanje može značajno smanjiti broj generiranih čvorova kod algoritma minimax. Ušteda ovisi o redoslijedu kojim se generiraju čvorovi. Optimalan redoslijed je onaj koji djecu čvora MAX generira silazno po minimax vrijednostima, a djecu čvora MIN uzlazno po minimax vrijednostima. Razmotite igru s dva moguća poteza u svakome stanju. Stablo se pretražuje do dubine 3. Vrijednosti heuristike igrača MAX za stanja u listovima stabla igre, kada su čvorovi-djeca generirani nasumičnim redoslijedom, su sljedeća (slijeva nadesno): $0, 3, -3, 4, 4, -1, 0, 2$. Na potezu je igrač MAX. Odredite broj čvorova koji će biti uklonjeni podrezivanjem alfa-beta. Zatim ponovite izračun, ali s optimalnim redoslijedom generiranja čvorova. **Koliko više čvorova će biti podrezano s optimalnim redoslijedom u odnosu na slučajan redoslijed generiranja čvorova?**

A 1 B 4 C 2 D 3

10 (R) Razmatramo igru dvaju minimax-igrača. Stablo igre definirano je prijelazima $succ(A) = \{B, C, D\}$, $succ(B) = \{E, F\}$, $succ(C) = \{G, H\}$, $succ(D) = \{I, J\}$, $succ(F) = succ(G) = \{K, L\}$, $succ(E) = succ(H) = succ(I) = \{M, N\}$ te $succ(J) = \{O, P\}$. Igrač MAX koristi heuristiku h_1, a igrač MIN heuristiku h_2. Vrijednosti heuristika su sljedeće (obje heuristike definiraju dobit dotičnog igrača):

	E	F	G	H	I	J	K	L	M	N	O	P
h_1	1	−2	3	2	3	5	−1	0	−2	−1	0	2
h_2	0	5	2	4	−1	2	3	2	−3	−2	1	−4

Oba igrača pretražuju najviše dva poteza unaprijed (jedan svoj i jedan suparnički). Igru započinje igrač MAX u stanju A. **Kroz koja stanja će se razvijati igra?**

A $A, C, G, \ldots$ B $A, D, J, \ldots$ C $A, C, H, \ldots$ D $A, D, I, \ldots$

3. Prikazivanje znanja i automatsko zaključivanje (6 pitanja)

11 (P) Zadane su FOL klauzule $Q(f(x), b) \vee \neg P(y) \vee Q(x, y)$ i $P(z) \vee \neg Q(g(c), z) \vee P(a)$. **Koja od sljedećih klauzula je rezolventa ovih dviju klauzula?**

A $Q(f(x), b) \vee Q(x, a) \vee P(y)$
B $Q(g(c), a) \vee Q(f(x), b)$
C $Q(f(x), b) \vee \neg Q(g(c), b) \vee P(a)$
D $Q(f(g(c)), b) \vee \neg P(a) \vee P(a)$

12 (R) Zadane su premise: *"Bela laje (L) ako je gladna (G) ili ako je uštap (U) ili ako je preplašena (P). Ako nije uštap, Bela dobro spava (S). Kada previše jede (J), onda Bela ne spava dobro. Bela nije preplašena ili je gladna."* **Koja je od sljedećih tvrdnji logička posljedica ovih premisa?**

A *Ako Bela ne spava dobro, onda laje.*
B *Bela nije gladna.*
C *Ako Bela ne spava dobro, onda previše jede.*
D *Bela je gladna.*

13 (P) Pravila zaključivanja poželjno su i ispravna i potpuna. Razmotrite pravilo zaključivanja r definirano kao $(A \vee B) \rightarrow C, \neg B \vdash_r \neg(C \rightarrow A)$. **Kakvo je pravilo r?**

A Ispravno, ali nije potpuno
B Niti ispravno niti potpuno
C Potpuno, ali nije ispravno
D Ispravno i potpuno

14 (T) Neka je R skup pravila zaključivanja te neka su F i G dobro oblikovane formule logike. **Kada za skup R kažemo da *nije* potpun skup pravila zaključivanja?**

A Ako je svaki model od G također model od F, ali postoji pravilo u R kojim iz F ne možemo izvesti G
B Ako je formula $F \rightarrow G$ zadovoljiva, ali ne postoji pravilo u R kojim iz $F \wedge \neg G$ možemo izvesti NIL
C Ako je svaki model od F također model od G, ali primjenom pravila iz R na formule F ne možemo izvesti G
D Ako je formula $\neg F \vee G$ valjana, ali ne postoji lanac pravila u R kojim iz F možemo izvesti $\neg G$

15 (R) Zadane su premise: (1) *U svakom gradu postoji pošta*, (2) *Virovitica je grad*, (3) *Svi vole one koji su sretni*, (4) *Lucija je dijete u Virovitici*, (5) *Onaj koji nekoga/nešto voli je sretan*, (6) *Svako dijete voli svaku poštu.* Želimo dokazati da iz ovih premisa logički slijedi *U Virovitici postoji netko koga svi vole.* Pretvorite ove izjave u FOL formule koristeći predikate $U(x, y)$ za "x je u y", $V(x, y)$ za "x voli y", $P(x)$ za "x je pošta", $G(x)$ za "x je grad", $S(x)$ za "x je sretan" i $D(x)$ za "x je dijete". Zatim primijenite rezoluciju opovrgavanjem uz strategiju skupa potpore. **Koliko klauzula ulazi u rezolucijski postupak i koji je najmanji broj rezolucijskih koraka potrebnih za dokaz?** (1 rezolucijski korak = 1 primjena rezolucijskog pravila.)

A 9 klauzula, 5 koraka
B 8 klauzula, 6 koraka
C 8 klauzula, 8 koraka
D 9 klauzula, 7 koraka

16 (T) Propozicijska logika (PL) i logika prvog reda (FOL) dva su formalizma za prikaz znanja. **Što je prednost, a što nedostatak PL u odnosu na FOL?** (Podsjetnik: Problem je *traktabilan* ako je rješiv u polinomijalnom vremenu.)

A PL je ekspresivniji od FOL, ali je dokazivanje netraktabilno
B Dokazivanje u PL je odlučivo, ali je PL manje ekspresivan od FOL
C Dokazivanje u PL je traktabilno, ali je FOL poluodlučiv
D Dokazivanje u PL je odlučivo, ali nije traktabilno kao u FOL

4. Logičko programiranje i ekspertni sustavi (4 pitanja)

17 (T) Ekspertni sustavi koriste se za automatsko zaključivanje u specifičnim domenama ljudskog znanja. **Što je karakteristika ekspertnih sustava?**

A Međuciljevi koje treba dokazati pohranjuju se na stog, a u slučaju omogućenosti većeg broja pravila, prednost ima pravilo koje je navedeno posljednje

B Mogu zaključivati ulančavanjem unazad, što je prikladno kada postoji mnogo mogućih zaključaka, a malo početnih činjenica

C Domenski specifično znanje definirano je u obliku činjenica i ako-onda pravila pohranjenih u stroju za zaključivanje

D Baza znanja odvojena je od mehanizma zaključivanja, kako bi se isti mehanizam mogao koristiti s različitim bazama znanja

18 (R) Baza znanja ekspertnog sustava sadrži sljedeća pravila:

(1) AKO $(A = a_2) \wedge (D = d_2)$ ONDA $C = c_1$
(2) AKO $(F = f_3) \vee (B = b_3)$ ONDA $C = c_2$
(3) AKO $(E = e_1) \vee (B = b_1)$ ONDA $(A = a_1) \wedge (D = d_2)$
(4) AKO $F = f_1$ ONDA $D = d_2$
(5) AKO $F = f_2$ ONDA $E = e_2$
(6) AKO $(B = b_3) \vee (D = d_1)$ ONDA $(E = e_1) \wedge (A = a_2)$

Sustav koristimo za izvođenje vrijednosti varijable C ulančavanjem unazad. Prednost imaju pravila s manjim rednim brojem. Pravila koja su jednom palila više ne mogu paliti. Jednom izvedeni međuciljevi brišu se sa stoga, neovisno o njihovoj poziciji na stogu. Na možebitne upite od strane sustava, korisnik odgovara sa $B = b_3$ i $F = f_1$. **Što radi ekspertni sustav pri izvođenju vrijednosti varijable C?**

A Odbacuje pravilo 5 te kasnije pali pravilo 6

B Odbacuje pravilo 6 te kasnije pali pravilo 3

C Izvodi $E = e_1$ te kasnije $E = e_2$

D Završava s četiri činjenice u radnoj memoriji

19 (R) Baza znanja u Prologu modelira tramvajsku mrežu u Zagrebu. Baza sadrži sljedeće činjenice i pravila:

```
pruga(frankopanska, zrinjevac).
pruga(frankopanska, vodnikova).
pruga(zrinjevac, branimirova).
pruga(vodnikova, branimirova).
zastoj(zrinjevac).
prohodno(X, Y) :- pruga(X, Y), not(zastoj(X)).
povezano(X, Y) :- prohodno(X,Y); prohodno(Y, X).
dosezljivo(X, Y) :- povezano(X, Y).
dosezljivo(X, Y) :- povezano(X, Z), dosezljivo(Z, Y).
```

Nad ovako definiranom bazom znanja izvodimo upit `dosezljivo(branimirova, frankopanska)`. Prolog će za ovaj upit vratiti `True`. Nacrtajte Prologovo stablo dokaza za ovaj upit sve do mjesta gdje se postupak pretrage po četvrti put vraća na točku odabira. Pritom za dokazivanje negiranog cilja koristite dva čvora (jedan za `not(P(X))` i drugi za `P(X)`). **Koliko čvorova ima stablo dokaza do trenutka četvrtog povratka u postupku pretraživanja?**

A 15 B 11 C 9 D 13

20 (T) Izvođenje programa u Prologu odgovara zaključivanju u Hornovoj klauzalnoj logici (HKL). HKL je podskup logike prvoga reda (FOL). **Zašto Prolog koristi HKL, a ne FOL?**

A Rezolucije opovrgavanjem u HKL je ispravna i potpuna

B Rezolucija opovrgavanjem u HKL je vremenski učinkovita

C Klauzule u HKL su zatvorene pod rezolucijom

D Zaključivanje u HKL uz dodatak negacije je odlučivo

Međuispit iz Uvoda u umjetnu inteligenciju (2023./2024.)

– 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 i heurističko pretraživanje (7 pitanja)

1 (R) Algoritmom slijepog pretraživanja rješavamo problem misionara i kanibala. Tri misionara i tri kanibala potrebno je jednim čamcem prevesti s jedne strane obale rijeke na drugu, pri čemu se ne smije dogoditi da se na jednoj strani nađu i kanibali i misionari, ali da je kanibala više nego misionara. Čamac može prevesti najviše dvije osobe i ne može ploviti prazan. Stanje prikazujemo kao trojku MKC, gdje su M i K broj misionara odnosno kanibala na lijevoj obali rijeke, a C je strana obale na kojoj se nalazi čamac. Npr., u stanju 32D na lijevoj obali su tri misionara i dva kanibala (jedan kanibal je onda na desnoj obali), a čamac je na desnoj obali. Početno stanje je 33L, a završno 00D. Funkcija sljedbenika generira sva legitimna stanja, uključivo i ona koja rezultiraju neuspjehom i nemaju sljedbenika, npr., stanje 23D. Koristimo algoritam iterativnog pretraživanja u dubinu (IDS) bez liste posjećenih stanja. Kod proširenja čvora, algoritam čvorove djecu generira prema numeričkom redoslijedu; npr., stanje 13D prethodi stanju 23D. **Koliko će čvorova algoritam IDS ispitati prije nego što proširi čvor sa stanjem** 31D?

A 14 B 16 C 11 D 18

2 (P) Rješavamo problem pretraživanja prostora stanja za koji vrijedi $m = 10$, $d = 7$ i $b = 4$. Koristimo pretraživanje u širinu (BFS) i pretraživanje u dubinu (DFS). Za oba algoritma razmotrite najgori slučaj vremenske složenosti. **Koliko će u takvom slučaju algoritam DFS generirati više čvorova od algoritma BFS?**

A 1376176 B 1310640 C 1376256 D 1310716

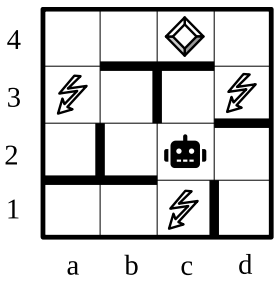
3 (T) Jedno od poželjnih svojstava heurističke funkcije je *konzistentnost* (ili *monotonost*). Razmotrite algoritam A^*. **Koja je prednost algoritma A^* s konzistentnom heuristikom u odnosu na A^* s nekonzistentnom ali optimističnom heuristikom?**

A Algoritam ima manju prostornu složenost
B Algoritam je optimalan
C Algoritam ispituje manji broj čvorova
D Algoritam je potpun

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

A Ograničeno pretraživanje u dubinu ($k = 4$)
B Iterativno pretraživanje u dubinu
C Pretraživanje u širinu
D Pretraživanje u dubinu

5 (R) Na slici desno prikazana je mapa dvodimenzijskoga svijeta od 16 polja. Robot Robby početno se nalazi na polju $c2$ i treba doći na polje $c4$ na kojemu se nalazi dijamant. Robby se može kretati prema gore, dolje, lijevo i desno (npr., sa $c2$ se može kretati na $c3$, $c1$, $b2$ i $d2$), ali ne može prolaziti kroz zidove (npr., ne može proći sa $c3$ na $c4$). Robby želi pronaći put s najmanjim utroškom baterije. Svaki potez Robbyja košta jednu energetsku jedinicu iz baterije, osim kada zakorači na polja $a3$, $c1$ ili $d3$, gdje je zbog energetskog polja utrošak baterije jednak 3 energetske jedinice. Primijenite algoritam A^*. Stanje odgovara Robbyjevoj poziciji, a cijena puta odgovara broju utrošenih energetskih jedinica. Za heuristiku koristite L1-udaljenost (Manhattan-udaljenost). U slučaju izjednačenja cijene, proširuje se čvor sa stanjem koje alfanumerički dolazi prvo (npr., $a2$ prije $b2$ prije $b3$). **Koliko čvorova će algoritam A^* ukupno proširiti?**



A 9 B 10 C 8 D 7

6 (T) Jedna od komponenti problema pretraživanja prostora stanja je funkcija sljedbenika *succ*. Razmotrite algoritam pretraživanja s jednolikom cijenom (UCS). **Kod tog algoritma, u što funkcija *succ* preslikava svako stanje?**

A Skup parova sačinjenih od početnog i završnog stanja

B Skup svih podskupova stanja (moguće prazan)

C Kartezijev umnožak skupa stanja i skupa nenegativnih brojeva

D Skup parova sačinjenih od stanja i pozitivnog broja

7 (P) Neka je S skup stanja te neka su s_1, s_2 ,s_3 i s_4 stanja iz S. Vrijedi $succ(s_1) = \{(s_2, 4)\}$ i $succ(s_2) = \{(s_3, c)\}$, gdje je c neka cijena prijelaza. Razmatramo dvije heuristike, h_1 i h_2. Heuristika h_1 definirana je u odnosu na ciljno stanje s_3, a heuristika h_2 u odnosu na ciljno stanje s_4. Vrijedi $h_1(s_1) = 11$ i $h_1(s_2) = 9$ te $h_2(s_1) = 20$, $h_2(s_2) = 16$ i $h_2(s_3) = 8$. Obje su heuristike optimistične. Međutim, na temelju ovih informacija heuristiku h_2 možemo unaprijediti tako da bude obavještenija (bliža stvarnoj cijeni). **Koje od navedenih vrijednosti za h_2 sigurno znamo da daju obavještenију, ali još uvijek optimističnu heuristiku?**

A $h_2(s_1) = 21$, $h_2(s_2) = 16$ C $h_2(s_1) = 20$, $h_2(s_2) = 17$

B $h_2(s_1) = 21$, $h_2(s_2) = 17$ D $h_2(s_1) = 20$, $h_2(s_2) = 18$

2. Igranje igara (3 pitanja)

8 (R) Razmatramo igru dvaju minimax-igrača. Stablo igre definirano je prijelazima $succ(A) = \{B, C, D\}$, $succ(B) = \{E, F\}$, $succ(C) = \{G, H\}$, $succ(D) = \{I, J\}$, $succ(F) = succ(G) = \{K, L\}$, $succ(E) = succ(H) = succ(I) = \{M, N\}$ te $succ(J) = \{O, P\}$. Igrač MAX koristi heuristiku h_1, a igrač MIN heuristiku h_2. Vrijednosti heuristika su sljedeće (obje heuristike definiraju dobit dotičnog igrača):

	E	F	G	H	I	J	K	L	M	N	O	P
h_1	1	−2	3	2	3	5	−1	0	−2	−1	0	2
h_2	0	5	2	4	−1	2	3	2	1	−4	−3	−2

Oba igrača pretražuju najviše dva poteza unaprijed (jedan svoj i jedan suparnički). Igru započinje igrač MAX u stanju A. **Kroz koja stanja će se razvijati igra?**

A $A, D, I, \ldots$ B $A, C, G, \ldots$ C $A, D, J, \ldots$ D $A, C, H, \ldots$

9 (T) Igrači MAX i MIN primjenjuju algoritam minimax prije svakog svog poteza, pretražujući do neke unaprijed definirane dubine. Zamislite, međutim, da igrač MAX algoritam minimax primjenjuje samo jednom, i to na početku igre, a da zatim tako dobivene minimax vrijednosti koristi za sve svoje buduće poteze. **U kojem bi slučaju tako implementirani igrač MAX sigurno maksimizirao svoju minimalnu dobit?**

A Ako koristi identičnu heuristiku kao igrač MIN

B Ako je pri izračunu minimax-vrijednosti dosegao sva završna stanja igre

C Ako je heuristika igrača MAX obavještenija od heuristike igrača MIN

D Nema garancije da bi takav igrač maksimizirao svoju minimalnu dobit

10 (P) Alfa-beta podrezivanje može značajno smanjiti broj generiranih čvorova kod algoritma minimax. Ušteda ovisi o redoslijedu kojim se generiraju čvorovi. Optimalan redoslijed je onaj koji djecu čvora MAX generira silazno po minimax vrijednostima, a djecu čvora MIN uzlazno po minimax vrijednostima. Razmotite igru s dva moguća poteza u svakome stanju. Stablo se pretražuje do dubine 3. Vrijednosti heuristike igrača MAX za stanja u listovima stabla igre, kada su čvorovi-djeca generirani nasumičnim redoslijedom, su sljedeća (slijeva nadesno): $0, 3, -3, 4, 4, -1, 0, 2$. Na potezu je igrač MAX. Odredite broj čvorova koji će biti uklonjeni podrezivanjem alfa-beta. Zatim ponovite izračun, ali s optimalnim redoslijedom generiranja čvorova. **Koliko više čvorova će biti podrezano s optimalnim redoslijedom u odnosu na slučajan redoslijed generiranja čvorova?**

A 1 B 2 C 4 D 3

3. Prikazivanje znanja i automatsko zaključivanje (6 pitanja)

11 (T) Propozicijska logika (PL) i logika prvog reda (FOL) dva su formalizma za prikaz znanja. **Što je prednost, a što nedostatak PL u odnosu na FOL?** (Podsjetnik: Problem je *traktabilan* ako je rješiv u polinomijalnom vremenu.)

A Dokazivanje u PL je odlučivo, ali je PL manje ekspresivan od FOL

B PL je ekspresivniji od FOL, ali je dokazivanje netraktabilno

C Dokazivanje u PL je odlučivo, ali nije traktabilno kao u FOL

D Dokazivanje u PL je traktabilno, ali je FOL poluodlučiv

12 (R) Zadane su premise: *"Bela laje (L) ako je gladna (G) ili ako je uštap (U) ili ako je preplašena (P). Ako nije uštap, Bela dobro spava (S). Kada previše jede (J), onda Bela ne spava dobro. Bela nije preplašena ili je gladna."* **Koja je od sljedećih tvrdnji logička posljedica ovih premisa?**

A *Bela nije gladna.*

B *Ako Bela ne spava dobro, onda previše jede.*

C *Bela laje.*

D *Ako Bela ne spava dobro, onda laje.*

13 (P) Pravila zaključivanja poželjno su i ispravna i potpuna. Razmotrite pravilo zaključivanja r definirano kao $(A \vee B) \rightarrow C, \neg B \vdash_r \neg(C \rightarrow A)$. **Kakvo je pravilo r?**

A Ispravno, ali nije potpuno

B Potpuno, ali nije ispravno

C Ispravno i potpuno

D Niti ispravno niti potpuno

14 (R) Zadane su premise: (1) *U svakom gradu postoji pošta*, (2) *Virovitica je grad*, (3) *Svi vole one koji su sretni*, (4) *Lucija je dijete u Virovitici*, (5) *Onaj koji nekoga/nešto voli je sretan*, (6) *Svako dijete voli svaku poštu.* Želimo dokazati da iz ovih premisa logički slijedi *U Virovitici postoji netko koga svi vole.* Pretvorite ove izjave u FOL formule koristeći predikate $U(x, y)$ za "x je u y", $V(x, y)$ za "x voli y", $P(x)$ za "x je pošta", $G(x)$ za "x je grad", $S(x)$ za "x je sretan" i $D(x)$ za "x je dijete". Zatim primijenite rezoluciju opovrgavanjem uz strategiju skupa potpore. **Koliko klauzula ulazi u rezolucijski postupak i koji je najmanji broj rezolucijskih koraka potrebnih za dokaz?** (1 rezolucijski korak = 1 primjena rezolucijskog pravila.)

A 9 klauzula, 7 koraka

B 9 klauzula, 5 koraka

C 8 klauzula, 6 koraka

D 8 klauzula, 8 koraka

15 (P) Zadane su FOL klauzule $Q(f(x), b) \vee \neg P(y) \vee Q(x, y)$ i $P(z) \vee \neg Q(g(c), z) \vee P(a)$. **Koja od sljedećih klauzula je rezolventa ovih dviju klauzula?**

A $Q(f(x), b) \vee Q(x, a) \vee P(y)$

B $Q(f(g(c)), b) \vee \neg P(a) \vee P(a)$

C $Q(g(c), a) \vee Q(f(x), b)$

D $Q(f(x), b) \vee \neg Q(g(c), b) \vee P(a)$

16 (T) Neka je R skup pravila zaključivanja te neka su F i G dobro oblikovane formule logike. **Kada za skup R kažemo da *nije* potpun skup pravila zaključivanja?**

A Ako je svaki model od G također model od F, ali postoji pravilo u R kojim iz F ne možemo izvesti G

B Ako je formula $F \rightarrow G$ zadovoljiva, ali ne postoji pravilo u R kojim iz $F \wedge \neg G$ možemo izvesti NIL

C Ako je svaki model od F također model od G, ali primjenom pravila iz R na formule F ne možemo izvesti G

D Ako je formula $\neg F \vee G$ valjana, ali ne postoji lanac pravila u R kojim iz F možemo izvesti $\neg G$

4. Logičko programiranje i ekspertni sustavi (4 pitanja)

17 (T) Ekspertni sustavi koriste se za automatsko zaključivanje u specifičnim domenama ljudskog znanja. **Što je karakteristika ekspertnih sustava?**

A Međuciljevi koje treba dokazati pohranjuju se na stog, a u slučaju omogućenosti većeg broja pravila, prednost ima pravilo koje je navedeno posljednje

B Baza znanja odvojena je od mehanizma zaključivanja, kako bi se isti mehanizam mogao koristiti s različitim bazama znanja

C Domenski specifično znanje definirano je u obliku činjenica i ako-onda pravila pohranjenih u stroju za zaključivanje

D Mogu zaključivati ulančavanjem unazad, što je prikladno kada postoji mnogo mogućih zaključaka, a malo početnih činjenica

18 (R) Baza znanja u Prologu modelira tramvajsku mrežu u Zagrebu. Baza sadrži sljedeće činjenice i pravila:

```
pruga(frankopanska, zrinjevac).
pruga(frankopanska, vodnikova).
pruga(zrinjevac, branimirova).
pruga(vodnikova, branimirova).
zastoj(zrinjevac).
prohodno(X, Y) :- pruga(X, Y), not(zastoj(X)).
povezano(X, Y) :- prohodno(X,Y); prohodno(Y, X).
dosezljivo(X, Y) :- povezano(X, Y).
dosezljivo(X, Y) :- povezano(X, Z), dosezljivo(Z, Y).
```

Nad ovako definiranom bazom znanja izvodimo upit `dosezljivo(branimirova, frankopanska)`. Prolog će za ovaj upit vratiti `True`. Nacrtajte Prologovo stablo dokaza za ovaj upit sve do mjesta gdje se postupak pretrage po četvrti put vraća na točku odabira. Pritom za dokazivanje negiranog cilja koristite dva čvora (jedan za `not(P(X))` i drugi za `P(X)`). **Koliko čvorova ima stablo dokaza do trenutka četvrtog povratka u postupku pretraživanja?**

A 11 B 9 C 13 D 15

19 (T) Izvođenje programa u Prologu odgovara zaključivanju u Hornovoj klauzalnoj logici (HKL). HKL je podskup logike prvoga reda (FOL). **Zašto Prolog koristi HKL, a ne FOL?**

A Rezolucije opovrgavanjem u HKL je ispravna i potpuna

B Zaključivanje u HKL uz dodatak negacije je odlučivo

C Klauzule u HKL su zatvorene pod rezolucijom

D Rezolucija opovrgavanjem u HKL je vremenski učinkovita

20 (R) Baza znanja ekspertnog sustava sadrži sljedeća pravila:

(1) AKO $(A = a_2) \wedge (D = d_2)$ ONDA $C = c_1$
(2) AKO $(F = f_3) \vee (B = b_3)$ ONDA $C = c_2$
(3) AKO $(E = e_1) \vee (B = b_1)$ ONDA $(A = a_1) \wedge (D = d_2)$
(4) AKO $F = f_1$ ONDA $D = d_2$
(5) AKO $F = f_2$ ONDA $E = e_2$
(6) AKO $(B = b_3) \vee (D = d_1)$ ONDA $(E = e_1) \wedge (A = a_2)$

Sustav koristimo za izvođenje vrijednosti varijable C ulančavanjem unazad. Prednost imaju pravila s manjim rednim brojem. Pravila koja su jednom palila više ne mogu paliti. Jednom izvedeni međuciljevi brišu se sa stoga, neovisno o njihovoj poziciji na stogu. Na možebitne upite od strane sustava, korisnik odgovara sa $B = b_3$ i $F = f_1$. **Što radi ekspertni sustav pri izvođenju vrijednosti varijable C?**

A Izvodi $E = e_1$ te kasnije $E = e_2$
B Pali tri pravila i izvodi $C = c_2$
C Odbacuje pravilo 2 te kasnije pali pravilo 5
D Pali pet pravila i izvodi $C = c_1$

Grupa	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
A	B	A	B	C	B	C	C	A	A	D	A	C	D	A	B	B	B	C	D	A
B	D	D	C	D	B	C	A	D	B	D	B	C	D	C	D	A	C	C	A	D
C	D	B	D	B	A	C	C	A	C	A	A	D	D	C	C	A	B	A	C	D
D	B	C	B	A	C	C	C	C	A	A	C	A	B	B	B	C	B	D	B	A
E	B	C	D	B	A	D	D	B	D	B	A	A	D	B	D	D	B	A	A	A
F	B	C	D	B	D	C	C	A	A	D	D	C	C	A	B	C	D	A	D	D
G	C	A	C	D	A	A	B	C	B	D	D	A	B	C	D	B	D	A	D	B
H	B	B	C	B	A	D	B	C	D	C	A	D	D	A	B	C	B	C	D	B