Uvod u umjetnu inteligenciju – pismeni ispit (2021./2022.)

– 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 igranje igara (4 pitanja)

1 (R) Neka su definirani skup stanja $S = \{a, b, c, d, e, f\}$ i funkcija sljedbenika $succ(a) = \{(b, 1), (c, 2)\}$, $succ(b) = succ(f) = \emptyset$, $succ(c) = \{(d, 3), (e, 5)\}$, $succ(d) = \{(b, 8), (f, 5)\}$, te $succ(e) = \{(d, 1), (f, 2)\}$. Heurističke vrijednosti čvorova neka su $h(a) = 7$, $h(b) = 4$, $h(c) = 6$, $h(d) = h(e) = 2$, $h(f) = 0$. Početno stanje neka je a, a ciljno f. Izvršite pretraživanje algoritmom A^*, bilježeći pritom sadržaje liste otvorenih čvorova O i zatvorenih čvorova C u svakom koraku algoritma (u nultom koraku $O = [(a, 0)]$ i $C = \emptyset$). **Koji su sadržaji listi O i C nakon petog koraka izvođenja algoritma A^*?**

A Algoritam ne dostiže peti korak

B $O = [(e, 7), (f, 10)]$, $C = \{(a, 0), (c, 2), (d, 5)\}$

C $O = [(f, 9)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5), (e, 7)\}$

D $O = [(f, 10)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5), (e, 7)\}$

2 (T) Algoritam minimax za čvorove stabla igre na dubini d poziva heurističku funkciju, koja procjenjuje minimax-vrijednost za dotični čvor. **Zašto algoritam minimax procjenjuje minimax-vrijednost, umjesto da izračuna stvarnu minimax-vrijednost?**

A Protivnički igrač ne mora igrati optimalno, pa je potrebno nanovo izračunati minimax-vrijednost svakog čvora

B Broj čvorova stabla igre raste eksponencijalno, pa stablo nije uvijek moguće pohraniti u memoriju

C Minimax-vrijednost može se izračunati samo za završne čvorove, dok je za unutarnje čvorove treba procijeniti

D Izračun minimax-vijednosti nekog čvora iziskuje dosezanje završnih čvorova, a to je vremenski netraktabilno

3 (R) Stablo igre definirano je prijelazima $succ(A) = \{B, C\}$, $succ(B) = \{D, E\}$, $succ(D) = \{H, I\}$, $succ(E) = \{J, K, L\}$, $succ(C) = \{F, G\}$, $succ(F) = \{M, N, O\}$ te $succ(G) = \{P, Q\}$. Heurističke vrijednosti listova su $h(H) = 3$, $h(I) = -14$, $h(J) = -3$, $h(K) = h(Q) = -5$, $h(L) = 7$, $h(M) = -1$, $h(N) = -10$, $h(O) = -1$, $h(P) = 17$. Optimalna strategija određuje algoritmom minimax uz alfa-beta podrezivanje. **Koji će čvorovi koji će pritom biti podrezani (preskočeni pri izračunu minimax vrijednosti)?**

A N, O, P, Q B G, P, Q C Q D L, N, O, Q

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

A $\max(h_1, h_2, h_3)$

B $\max(\min(h_1, h_2), h_3)$

C $\min(\max(h_1, h_2), \min(h_1, h_3))$

D $\min(\min(h_1, h_2), \max(h_2, h_3))$

2. Prikazivanje znanja i automatsko zaključivanje (4 pitanja)

5 (R) Zadane su premise: *"Ivo voli sve vrste hrane. Jabuka i piletina su hrana. Branko jede lješnjake i nije mu zlo od njih. Vesna jede sve što i Branko. Ako netko nešto jede i nije mu od toga zlo, onda je to hrana."* Formalizirajte ove premise u FOL i pretvorite ih u klauzule. Pritom koristite $V(x, y)$ za "x voli y", $J(x, y)$ za "x jede y", "$Z(x, y)$" za "x-u je zlo od y", i "$H(x)$" "x je hrana". Zatim rezolucijom opovrgavanjem uz strategiju skupa potpore dokažite cilj *Ivo voli lješnjake.* **Koliko je minimalno koraka (primjena rezolucijskog pravila) potrebno za dokaz?**

A 4 B 5 C 3 D 6

6 (T) Rezolucija opovrgavanjem uz faktorizaciju ispravno je i potpuno pravilo zaključivanja propozicijske logike (PL). **Što se događa ako radimo rezoluciju opovrgavanjem, ali ne provodimo faktorizaciju?**

A Za neke premise $F_1, \dots, F_n$ i cilj G postupak će se zaustaviti prije nego što izvede NIL, premda $F_1, \dots, F_n \models G$

B Za neke premise $F_1, \dots, F_n$ i neku ciljnu formulu G postupak će izvesti NIL, premda $F_1, \dots, F_n \not\models G$

C Ako $\models F_1 \wedge \cdots \wedge F_n \rightarrow G$, postupak će iz premisa $F_1, \dots, F_n, \neg G$ izvesti NIL, inače se može ne zaustaviti

D Ako $\models \neg(F_1 \wedge \cdots \wedge F_n \wedge \neg G)$, postupak primijenjen na premise $F_1, \dots, F_n, \neg G$ nikada neće izvesti NIL

7 (T) Svaki formalizam za prikaz znanja ima svoje ontološke i epistemološke pretpostavke. **Što je epistemološka pretpostavka logike prvoga reda (FOL)?**

A Svaka formula je ili istinita ili lažna

B Predikati definiraju svojstva i relacije

C Postoje objekti i relacije između njih

D Skup mogućih interpretacija je konačan

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

A 60 B 104 C 119 D 74

3. Logičko programiranje i ekspertni sustavi (2 pitanja)

9 (T) Široj primjeni ekspertnih sustava značajno je doprinio razvoj ljusaka ekspertnih sustava (engl. *expert system shell*), kao što je CLIPS. **Što je ljuska ekspertnog sustava?**

A Korisničko sučelje koje podržava objašnjavanje zaključaka, bez mehanizma za zaključivanje i baze znanja

B Mehanizam za zaključivanje na temelju pravila i za uređivanje baze znanje, bez same baze znanja

C Cjelokupni ekspertni sustav, uključivo baza znanja, mehanizam za zaključivanje i korisničko sučelje

D Skup pravila i činjenica koje opisuju stručno znanje u nekoj zatvorenoj domeni

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

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

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

A 10 B 6 C 12 D 8

4. Modeliranje neizvjesnosti (2 pitanja)

11 (R) U sustavu neizrazitog zaključivanja modeliramo vezu između dijabetesa i sklonosti depresiji. Skup G je prosječna razina glukoze u krvi u mmol/L, $G = \{1, 2, \ldots, 11\}$, a skup D je stupanj depresije prema Beckovom upitniku depresije, $D = \{0, 1, \ldots, 63\}$. Nad ovim univerzalnim skupovima definirali smo neizrazite skupove "predijabetes" kao $G_p = \{0.3/5, 1/6, 0.1/7\}$ te "umjerena depresija" kao $D_u = \{0.5/15, 0.75/16, 1/17, 0.5/18\}$. U sustavu imamo pravilo "ako manje-više(G_p), onda D_u", modelirano kao neizrazita relacija. Zanima nas kolika je sklonost depresiji osoba čija razina glukoze u krvi odgovara neizrazitom skupu $G' = \{0.7/5, 1/6, 0.3/7, 0.1/8\}$. **Kako glasi neizraziti skup D' koji dobivamo generaliziranim modusom ponensom?**

A $\{0.3/15, 0.75/16, 0.75/17, \sqrt{0.1}/18\}$
C $\{0.25/15, 0.75/16, 1/17, 0.75/18\}$
B $\{0.5/15, 0.75/16, 1/17, 0.5/18\}$
D $\{\sqrt{0.1}/15, 0.3/16, 0.75/17, 0.3/18\}$

12 (T) Bayesovo pravilo može se koristiti za modeliranje zaključivanja s više dokaza. U tom slučaju tipično uvodimo pretpostavku da su dokazi međusobno uvjetno nezavisni. **Zašto uvodimo tu pretpostavku?**

A Ako su dokazi uvjetno nezavisni, broj vjerojatnosti koje trebamo procijeniti mnogo je manji nego kada dokazi nisu uvjetno nezavisni

B Ako postoje više od dvije hipoteze, iz uvjeta međusobne isključivosti hipoteza slijedi da dokazi moraju biti uvjetno nezavisni za danu hipotezu

C Dokazi su u stvarnosti uglavnom uvjetno nezavisni, pa uvođenjem te pretpostavke procjena aposteriorne vjerojatnosti postaje točnija

D Bez pretpostavke o uvjetnoj nezavisnosti dokaza nazivnik Bayesovog pravila nije definiran

5. Strojno učenje, umjetne neuronske mreže i podržano učenje (5 pitanja)

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

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

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

A 1.3137 B 1.2627 C 1.3521 D 1.4767

14 (P) Treniramo model stabla odluke na skupu podataka u kojemu, nažalost, ima i nešto šuma. Svjesni smo da prisustvo šuma može dovesti do prenaučenosti modela, pa smo odlučili primijeniti unakrsnu provjeru da bismo podrezali stablo odluke. Skup označenih primjera podijelili smo na skup za učenje D_u i skup za provjeru D_p, tako da $D_u \cap D_p = \varnothing$. Sada isprobavamo nekoliko različitih dubina stabla, od $d = 1$ do $d = 42$. Ispitivanjem pogreške tih stabala različite dubine, zaključili smo da je optimalna dubina stabla $d = 17$. **Što to konkretno znači?**

A Da je pogreška na skupu D_p za stablo sa $d = 17$ veća od pogreške za isto to stablo na skupu D_u, ali veća od pogreške za bilo koju drugu dubinu stabla d na skupu D_p

B Da su pogreške za stabla sa $d = 16$ i $d = 18$ veće i na skupu D_u i na skupu D_p, s time da su na skupu D_p očekivano veće nego na skupu D_u

C Da je pogreška na skupu D_p za stablo sa $d = 17$ manja od pogreške na D_p za $d < 17$ i $d > 17$, ali očekivano ta je pogreška na D_p veća od pogreške na skupu D_u

D Da stablo za $d = 17$ ostvaruje najmanju pogrešku na skupu D_u, dok na skupu D_p stablo ostvaruje uvijek veću pogrešku, s maksimumom pogreške za $d = 1$

15 (R) Mali je Ivica svakog svakog ljeta u zadnjih sedam godina naučio jedan novi programski jezik. Svoja vrijedna iskustva sažeo je u listu *"Programski jezik koji mi se sviđa"*, gdje je svaki jezik opisao četirima značajkama, te je naznačio je li mu se dotični jezik svidio ($y = 1$) ili nije ($y = 0$). Ta lista izgleda ovako:

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

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

A 0.856 B 0.799 C 0.694 D 0.431

16 (T) Umjetne neuronske mreže sastoje se od više neurona s nelinearnim prijenosnim funkcijama. Takve mreže tipično učimo algoritmom propagacije pogreške unazad (algoritam BP). **Što bi se dogodilo da, umjesto više neurona, koristimo jedan neuron, i to sa sigmoidnom prijenosnom funkcijom?**

A Mogli bismo koristiti algoritam BP, ali bismo mogli riješiti samo linearno odvojive probleme

B Granica između klasa bila bi nelinearna, ali manje nelinearna nego da koristimo više neurona

C Granica između klasa bila ista kao i s više neurona, ali bi mreža lošije generalizirala na neviđene primjere

D Mogli bismo odvojiti linearno neodvojive klase, ali ne bismo mogli koristiti algoritam BP

17 (P) Rešetkasti svijet sastoji se od četiri ćelije, označene brojevima 1 do 4 i poredane vodoravno slijeva nadesno. Ćelija 4 je terminalna i agentu donosi 1 bod; nema drugih bodova. Svijet je okružen zidovima. Agent može poduzeti akcije *lijevo* i *desno*. Ako bi ga akcija dovela na zid, ostaje u ćeliji u kojoj je bio. Agent uvijek kreće iz ćelije 1. Provodimo postupak Q-učenja. Početni iznosi Q-funkcije su 0. Parametri su $\alpha = 0.5$ i $\gamma = 1$. Agent najprije igra tri epizode u kojima uvijek bira akciju *desno* i uči. Nakon toga igra još 10 epizoda u kojima u svakoj ćeliji pohlepno odabire potez koji mu se čini optimalan i također uči. **Nakon provedenih svih 13 epizoda, koja je očekivana vrijednost** $q(3, desno)$**?**

A Između 0.75 i 0.9995 B Manja od 0.75 C Veća od 0.9995 D Nije moguće odgovoriti

6. Prirodom inspirirani optimizacijski algoritmi (3 pitanja)

18 (P) Genetskim algoritmom tražimo maksimum funkcije $f(x, y, z)$, koristeći binarnu reprezentaciju rješenja. Vrijednost svake od varijabli pretražuje se u intervalu $[-20, 10]$, pri čemu je potrebno osigurati da se to pretraživanje provodi barem s preciznošću 0.05. **Od koliko se *minimalno* bitova treba sastojati kromosom?**

A 9 B 28 C 27 D 30

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

A 53 B 44 C 2 D 21

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

A Imamo garanciju da traženo rješenje postoji (no problem je pronaći ga)

B Rješenje omogućava rekonstrukciju puta do početnog stanja

C Bitno nam je samo konačno stanje

D Ispitni predikat definiramo tako da uspoređuje stanje sa zadanim predloškom (kao kod slagalice)

Uvod u umjetnu inteligenciju – pismeni ispit (2021./2022.)

– 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 igranje igara (4 pitanja)

1 (T) Algoritam minimax za čvorove stabla igre na dubini d poziva heurističku funkciju, koja procjenjuje minimax-vrijednost za dotični čvor. **Zašto algoritam minimax procjenjuje minimax-vrijednost, umjesto da izračuna stvarnu minimax-vrijednost?**

A Minimax-vrijednost može se izračunati samo za završne čvorove, dok je za unutarnje čvorove treba procijeniti

B Protivnički igrač ne mora igrati optimalno, pa je potrebno nanovo izračunati minimax-vrijednost svakog čvora

C Broj čvorova stabla igre raste eksponencijalno, pa stablo nije uvijek moguće pohraniti u memoriju

D Izračun minimax-vijednosti nekog čvora iziskuje dosezanje završnih čvorova, a to je vremenski netraktabilno

2 (R) Stablo igre definirano je prijelazima $succ(A) = \{B, C\}$, $succ(B) = \{D, E\}$, $succ(D) = \{H, I\}$, $succ(E) = \{J, K, L\}$, $succ(C) = \{F, G\}$, $succ(F) = \{M, N, O\}$ te $succ(G) = \{P, Q\}$. Heurističke vrijednosti listova su $h(H) = h(J) = 8$, $h(I) = -15$, $h(K) = -h(Q) = 17$, $h(L) = h(O) = 1$, $h(M) = -12$, $h(N) = -19$, $h(P) = -14$. Optimalna strategija određuje algoritmom minimax uz alfa-beta podrezivanje. **Koji će čvorovi koji će pritom biti podrezani (preskočeni pri izračunu minimax vrijednosti)?**

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

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

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

4 (R) Neka su definirani skup stanja $S = \{a, b, c, d, e, f\}$ i funkcija sljedbenika $succ(a) = \{(b, 1), (c, 2)\}$, $succ(b) = succ(f) = \emptyset$, $succ(c) = \{(d, 3), (e, 5)\}$, $succ(d) = \{(b, 8), (f, 5)\}$, te $succ(e) = \{(d, 1), (f, 2)\}$. Heurističke vrijednosti čvorova neka su $h(a) = 7$, $h(b) = 4$, $h(c) = 6$, $h(d) = h(e) = 2$, $h(f) = 0$. Početno stanje neka je a, a ciljno f. Izvršite pretraživanje algoritmom A^*, bilježeći pritom sadržaje liste otvorenih čvorova O i zatvorenih čvorova C u svakom koraku algoritma (u nultom koraku $O = [(a, 0)]$ i $C = \emptyset$). **Koji su sadržaji listi O i C nakon petog koraka izvođenja algoritma A^*?**

A $O = [(e, 7), (f, 10), (b, 13)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5)\}$

B Algoritam ne dostiže peti korak

C $O = [(f, 9)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5), (e, 7)\}$

D $O = [(f, 0), (e, 2), (b, 4)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5)\}$

2. Prikazivanje znanja i automatsko zaključivanje (4 pitanja)

5 (T) Svaki formalizam za prikaz znanja ima svoje ontološke i epistemološke pretpostavke. **Što je epistemološka pretpostavka logike prvoga reda (FOL)?**

A Skup mogućih interpretacija je konačan

B Predikati definiraju svojstva i relacije

C Svaka formula je ili istinita ili lažna

D Postoje objekti i relacije između njih

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

A 104 B 119 C 60 D 74

7 (R) Zadane su premise: *"Ivo voli sve vrste hrane. Jabuka i piletina su hrana. Branko jede lješnjake i nije mu zlo od njih. Vesna jede sve što i Branko. Ako netko nešto jede i nije mu od toga zlo, onda je to hrana."* Formalizirajte ove premise u FOL i pretvorite ih u klauzule. Pritom koristite $V(x, y)$ za "x voli y", $J(x, y)$ za "x jede y", "$Z(x, y)$" za "x-u je zlo od y", i "$H(x)$" "x je hrana". Zatim rezolucijom opovrgavanjem uz strategiju skupa potpore dokažite cilj *Ivo voli lješnjake.* **Koliko je minimalno koraka (primjena rezolucijskog pravila) potrebno za dokaz?**

A 5 B 3 C 4 D 6

8 (T) Rezolucija opovrgavanjem uz faktorizaciju ispravno je i potpuno pravilo zaključivanja propozicijske logike (PL). **Što se događa ako radimo rezoluciju opovrgavanjem, ali ne provodimo faktorizaciju?**

A Ako $\models \neg(F_1 \wedge \cdots \wedge F_n \wedge \neg G)$, postupak primijenjen na premise $F_1, \ldots, F_n, \neg G$ nikada neće izvesti NIL

B Za neke premise $F_1, \ldots, F_n$ i neku ciljnu formulu G postupak će izvesti NIL, premda $F_1, \ldots, F_n \not\models G$

C Ako $\models F_1 \wedge \cdots \wedge F_n \rightarrow G$, postupak će iz premisa $F_1, \ldots, F_n, \neg G$ izvesti NIL, inače se može ne zaustaviti

D Za neke premise $F_1, \ldots, F_n$ i cilj G postupak će se zaustaviti prije nego što izvede NIL, premda $F_1, \ldots, F_n \models G$

3. Logičko programiranje i ekspertni sustavi (2 pitanja)

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

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

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

A 10 B 6 C 8 D 12

10 (T) Široj primjeni ekspertnih sustava značajno je doprinio razvoj ljusaka ekspertnih sustava (engl. *expert system shell*), kao što je CLIPS. **Što je ljuska ekspertnog sustava?**

A Korisničko sučelje koje podržava objašnjavanje zaključaka, bez mehanizma za zaključivanje i baze znanja

B Skup pravila i činjenica koje opisuju stručno znanje u nekoj zatvorenoj domeni

C Cjelokupni ekspertni sustav, uključivo baza znanja, mehanizam za zaključivanje i korisničko sučelje

D Mehanizam za zaključivanje na temelju pravila i za uređivanje baze znanje, bez same baze znanja

4. Modeliranje neizvjesnosti (2 pitanja)

11 (T) Bayesovo pravilo može se koristiti za modeliranje zaključivanja s više dokaza. U tom slučaju tipično uvodimo pretpostavku da su dokazi međusobno uvjetno nezavisni. **Zašto uvodimo tu pretpostavku?**

- A Dokazi su u stvarnosti uglavnom uvjetno nezavisni, pa uvođenjem te pretpostavke procjena aposteriorne vjerojatnosti postaje točnija
- B Ako postoje više od dvije hipoteze, iz uvjeta međusobne isključivosti hipoteza slijedi da dokazi moraju biti uvjetno nezavisni za danu hipotezu
- C Ako su dokazi uvjetno nezavisni, broj vjerojatnosti koje trebamo procijeniti mnogo je manji nego kada dokazi nisu uvjetno nezavisni
- D Bez pretpostavke o uvjetnoj nezavisnosti dokaza nazivnik Bayesovog pravila nije definiran

12 (R) U sustavu neizrazitog zaključivanja modeliramo vezu između dijabetesa i sklonosti depresiji. Skup G je prosječna razina glukoze u krvi u mmol/L, $G = \{1, 2, \dots, 11\}$, a skup D je stupanj depresije prema Beckovom upitniku depresije, $D = \{0, 1, \dots, 63\}$. Nad ovim univerzalnim skupovima definirali smo neizrazite skupove "predijabetes" kao $G_p = \{0.3/5, 1/6, 0.1/7\}$ te "umjerena depresija" kao $D_u = \{0.25/15, 0.75/16, 1/17, 0.75/18\}$. U sustavu imamo pravilo "ako manje-više(G_p), onda D_u", modelirano kao neizrazita relacija. Zanima nas kolika je sklonost depresiji osoba čija razina glukoze u krvi odgovara neizrazitom skupu $G' = \{0.7/5, 1/6, 0.3/7, 0.1/8\}$. **Kako glasi neizraziti skup D' koji dobivamo generaliziranim modusom ponensom?**

- A $\{0.3/15, 0.75/16, 0.75/17, \sqrt{0.1}/18\}$
- C $\{0.5/15, 0.75/16, 1/17, 0.5/18\}$
- B $\{\sqrt{0.1}/15, 0.3/16, 0.75/17, 0.3/18\}$
- D $\{0.25/15, 0.75/16, 1/17, 0.75/18\}$

5. Strojno učenje, umjetne neuronske mreže i podržano učenje (5 pitanja)

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

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

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

A 1.3417 B 1.2627 C 1.3137 D 1.5116

14 (T) Umjetne neuronske mreže sastoje se od više neurona s nelinearnim prijenosnim funkcijama. Takve mreže tipično učimo algoritmom propagacije pogreške unazad (algoritam BP). **Što bi se dogodilo da, umjesto više neurona, koristimo jedan neuron, i to sa sigmoidnom prijenosnom funkcijom?**

- A Mogli bismo koristiti algoritam BP, ali bismo mogli riješiti samo linearno odvojive probleme
- B Mogli bismo odvojiti linearno neodvojive klase, ali ne bismo mogli koristiti algoritam BP
- C Granica između klasa bila bi nelinearna, ali manje nelinearna nego da koristimo više neurona
- D Granica između klasa bila ista kao i s više neurona, ali bi mreža lošije generalizirala na neviđene primjere

15 (R) Mali je Ivica svakog svakog ljeta u zadnjih sedam godina naučio jedan novi programski jezik. Svoja vrijedna iskustva sažeo je u listu *"Programski jezik koji mi se sviđa"*, gdje je svaki jezik opisao četirima značajkama, te je naznačio je li mu se dotični jezik svidio ($y = 1$) ili nije ($y = 0$). Ta lista izgleda ovako:

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

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

A 0.431 B 0.856 C 0.799 D 0.694

16 (P) Treniramo model stabla odluke na skupu podataka u kojemu, nažalost, ima i nešto šuma. Svjesni smo da prisustvo šuma može dovesti do prenaučenosti modela, pa smo odlučili primijeniti unakrsnu provjeru da bismo podrezali stablo odluke. Skup označenih primjera podijelili smo na skup za učenje D_u i skup za provjeru D_p, tako da $D_u \cap D_p = \varnothing$. Sada isprobavamo nekoliko različitih dubina stabla, od $d = 1$ do $d = 42$. Ispitivanjem pogreške tih stabala različite dubine, zaključili smo da je optimalna dubina stabla $d = 17$. **Što to konkretno znači?**

A Da je pogreška na skupu D_p za stablo sa $d = 17$ veća od pogreške za isto to stablo na skupu D_u, ali veća od pogreške za bilo koju drugu dubinu stabla d na skupu D_p

B Da su pogreške za stabla sa $d = 16$ i $d = 18$ veće i na skupu D_u i na skupu D_p, s time da su na skupu D_p očekivano veće nego na skupu D_u

C Da stablo za $d = 17$ ostvaruje najmanju pogrešku na skupu D_u, dok na skupu D_p stablo ostvaruje uvijek veću pogrešku, s maksimumom pogreške za $d = 1$

D Da je pogreška na skupu D_p za stablo sa $d = 17$ manja od pogreške na D_p za $d < 17$ i $d > 17$, ali očekivano ta je pogreška na D_p veća od pogreške na skupu D_u

17 (P) Rešetkasti svijet sastoji se od četiri ćelije, označene brojevima 1 do 4 i poredane vodoravno slijeva nadesno. Ćelija 4 je terminalna i agentu donosi 1 bod; nema drugih bodova. Svijet je okružen zidovima. Agent može poduzeti akcije *lijevo* i *desno*. Ako bi ga akcija dovela na zid, ostaje u ćeliji u kojoj je bio. Agent uvijek kreće iz ćelije 1. Provodimo postupak Q-učenja. Početni iznosi Q-funkcije su 0. Parametri su $\alpha = 0.5$ i $\gamma = 1$. Agent najprije igra tri epizode u kojima uvijek bira akciju *desno* i uči. Nakon toga igra još 10 epizoda u kojima u svakoj ćeliji pohlepno odabire potez koji mu se čini optimalan i također uči. **Nakon provedenih svih 13 epizoda, koja je očekivana vrijednost** $q(3, \mathit{desno})$**?**

A Manja od 0.75 B Veća od 0.9995 C Između 0.75 i 0.9995 D Nije moguće odgovoriti

6. Prirodom inspirirani optimizacijski algoritmi (3 pitanja)

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

A 34 B 33 C 36 D 11

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

A Mogu se rješavati algoritmom A*, ali samo uz optimističnu heuristiku i skup posjećenih stanja

B Imamo garanciju da traženo rješenje postoji (no problem je pronaći ga)

C Bitno nam je samo konačno stanje

D Rješenje omogućava rekonstrukciju puta do početnog stanja

20 (P) Algoritmom kolonije mrava rješavamo problem trgovačkog putnika nad odabranim gradovima u Republici Hrvatskoj. Za algoritam su poznati sljedeći parametri: $\tau_0 = 20$, $\alpha = 2$, $\beta = 3$, $\rho = 0.2$, kolonija se sastoji od 40 mrava te ažuriranje u svakoj epohi obavlja samo mrav koji je u toj epohi pronašao najbolje rješenje. Razmotrimo brid koji povezuje gradove Zagreb i Dubrovnik. Pretpostavimo da niti u jednoj epohi ta dva grada nisu sljedbenici u pronađenom putu. **Nakon koliko će epoha algoritma količina feromonskih tragova na spomenutom bridu postati manja od 0.1?**

A 52 B 31 C 24 D 4

Uvod u umjetnu inteligenciju – pismeni ispit (2021./2022.)

– 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 igranje igara (4 pitanja)

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

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

2 (T) Algoritam minimax za čvorove stabla igre na dubini d poziva heurističku funkciju, koja procjenjuje minimax-vrijednost za dotični čvor. **Zašto algoritam minimax procjenjuje minimax-vrijednost, umjesto da izračuna stvarnu minimax-vrijednost?**

A Izračun minimax-vijednosti nekog čvora iziskuje dosezanje završnih čvorova, a to je vremenski netraktabilno

B Broj čvorova stabla igre raste eksponencijalno, pa stablo nije uvijek moguće pohraniti u memoriju

C Minimax-vrijednost može se izračunati samo za završne čvorove, dok je za unutarnje čvorove treba procijeniti

D Protivnički igrač ne mora igrati optimalno, pa je potrebno nanovo izračunati minimax-vrijednost svakog čvora

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

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

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

4 (R) Neka su definirani skup stanja $S = \{a, b, c, d, e, f\}$ i funkcija sljedbenika $succ(a) = \{(b, 1), (c, 2)\}$, $succ(b) = succ(f) = \emptyset$, $succ(c) = \{(d, 3), (e, 5)\}$, $succ(d) = \{(b, 8), (f, 5)\}$, te $succ(e) = \{(d, 1), (f, 2)\}$. Heurističke vrijednosti čvorova neka su $h(a) = 7$, $h(b) = 4$, $h(c) = 6$, $h(d) = h(e) = 2$, $h(f) = 0$. Početno stanje neka je a, a ciljno f. Izvršite pretraživanje algoritmom A^*, bilježeći pritom sadržaje liste otvorenih čvorova O i zatvorenih čvorova C u svakom koraku algoritma (u nultom koraku $O = [(a, 0)]$ i $C = \emptyset$). **Koji su sadržaji listi O i C nakon petog koraka izvođenja algoritma A^*?**

A $O = [(e, 7), (f, 10), (b, 13)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5)\}$

B $O = [(f, 0), (e, 2), (b, 4)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5)\}$

C Algoritam ne dostiže peti korak

D $O = [(f, 9)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5), (e, 7)\}$

2. Prikazivanje znanja i automatsko zaključivanje (4 pitanja)

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

A 60 B 119 C 74 D 104

6 (T) Svaki formalizam za prikaz znanja ima svoje ontološke i epistemološke pretpostavke. **Što je epistemološka pretpostavka logike prvoga reda (FOL)?**

A Postoje objekti i relacije između njih
B Skup mogućih interpretacija je konačan
C Predikati definiraju svojstva i relacije
D Svaka formula je ili istinita ili lažna

7 (R) Zadane su premise: *"Ivo voli sve vrste hrane. Jabuka i piletina su hrana. Branko jede lješnjake i nije mu zlo od njih. Vesna jede sve što i Branko. Ako netko nešto jede i nije mu od toga zlo, onda je to hrana."* Formalizirajte ove premise u FOL i pretvorite ih u klauzule. Pritom koristite $V(x, y)$ za "x voli y", $J(x, y)$ za "x jede y", "$Z(x, y)$" za "x-u je zlo od y", i "$H(x)$" "x je hrana". Zatim rezolucijom opovrgavanjem uz strategiju skupa potpore dokažite cilj *Ivo voli lješnjake.* **Koliko je minimalno koraka (primjena rezolucijskog pravila) potrebno za dokaz?**

A 5 B 6 C 3 D 4

8 (T) Rezolucija opovrgavanjem uz faktorizaciju ispravno je i potpuno pravilo zaključivanja propozicijske logike (PL). **Što se događa ako radimo rezoluciju opovrgavanjem, ali ne provodimo faktorizaciju?**

A Za neke premise $F_1, \ldots, F_n$ i cilj G postupak će se zaustaviti prije nego što izvede NIL, premda $F_1, \ldots, F_n \vDash G$
B Ako $\vDash \neg(F_1 \wedge \cdots \wedge F_n \wedge \neg G)$, postupak primijenjen na premise $F_1, \ldots, F_n, \neg G$ nikada neće izvesti NIL
C Za neke premise $F_1, \ldots, F_n$ i neku ciljnu formulu G postupak će izvesti NIL, premda $F_1, \ldots, F_n \nvDash G$
D Ako $\vDash F_1 \wedge \cdots \wedge F_n \rightarrow G$, postupak će iz premisa $F_1, \ldots, F_n, \neg G$ izvesti NIL, inače se može ne zaustaviti

3. Logičko programiranje i ekspertni sustavi (2 pitanja)

9 (T) Široj primjeni ekspertnih sustava značajno je doprinio razvoj ljusaka ekspertnih sustava (engl. *expert system shell*), kao što je CLIPS. **Što je ljuska ekspertnog sustava?**

A Cjelokupni ekspertni sustav, uključivo baza znanja, mehanizam za zaključivanje i korisničko sučelje
B Mehanizam za zaključivanje na temelju pravila i za uređivanje baze znanje, bez same baze znanja
C Skup pravila i činjenica koje opisuju stručno znanje u nekoj zatvorenoj domeni
D Korisničko sučelje koje podržava objašnjavanje zaključaka, bez mehanizma za zaključivanje i baze znanja

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

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

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

A 10 B 8 C 12 D 6

4. Modeliranje neizvjesnosti (2 pitanja)

11 (T) Bayesovo pravilo može se koristiti za modeliranje zaključivanja s više dokaza. U tom slučaju tipično uvodimo pretpostavku da su dokazi međusobno uvjetno nezavisni. **Zašto uvodimo tu pretpostavku?**

- A Dokazi su u stvarnosti uglavnom uvjetno nezavisni, pa uvođenjem te pretpostavke procjena aposteriorne vjerojatnosti postaje točnija
- B Ako postoje više od dvije hipoteze, iz uvjeta međusobne isključivosti hipoteza slijedi da dokazi moraju biti uvjetno nezavisni za danu hipotezu
- C Ako su dokazi uvjetno nezavisni, broj vjerojatnosti koje trebamo procijeniti mnogo je manji nego kada dokazi nisu uvjetno nezavisni
- D Bez pretpostavke o uvjetnoj nezavisnosti dokaza nazivnik Bayesovog pravila nije definiran

12 (R) U sustavu neizrazitog zaključivanja modeliramo vezu između dijabetesa i sklonosti depresiji. Skup G je prosječna razina glukoze u krvi u mmol/L, $G = \{1, 2, \ldots, 11\}$, a skup D je stupanj depresije prema Beckovom upitniku depresije, $D = \{0, 1, \ldots, 63\}$. Nad ovim univerzalnim skupovima definirali smo neizrazite skupove "predijabetes" kao $G_p = \{0.3/5, 1/6, 0.1/7\}$ te "umjerena depresija" kao $D_u = \{0.5/15, 0.75/16, 1/17, 0.5/18\}$. U sustavu imamo pravilo "ako manje-više(G_p), onda D_u", modelirano kao neizrazita relacija. Zanima nas kolika je sklonost depresiji osoba čija razina glukoze u krvi odgovara neizrazitom skupu $G' = \{0.7/5, 1/6, 0.3/7, 0.1/8\}$. **Kako glasi neizraziti skup D' koji dobivamo generaliziranim modusom ponensom?**

- A $\{\sqrt{0.1}/15, 0.3/16, 0.75/17, 0.3/18\}$
- C $\{0.25/15, 0.75/16, 1/17, 0.75/18\}$
- B $\{0.3/15, 0.75/16, 0.75/17, \sqrt{0.1}/18\}$
- D $\{0.5/15, 0.75/16, 1/17, 0.5/18\}$

5. Strojno učenje, umjetne neuronske mreže i podržano učenje (5 pitanja)

13 (T) Umjetne neuronske mreže sastoje se od više neurona s nelinearnim prijenosnim funkcijama. Takve mreže tipično učimo algoritmom propagacije pogreške unazad (algoritam BP). **Što bi se dogodilo da, umjesto više neurona, koristimo jedan neuron, i to sa sigmoidnom prijenosnom funkcijom?**

- A Granica između klasa bila bi nelinearna, ali manje nelinearna nego da koristimo više neurona
- B Mogli bismo koristiti algoritam BP, ali bismo mogli riješiti samo linearno odvojive probleme
- C Granica između klasa bila ista kao i s više neurona, ali bi mreža lošije generalizirala na neviđene primjere
- D Mogli bismo odvojiti linearno neodvojive klase, ali ne bismo mogli koristiti algoritam BP

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

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

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

A 1.2627 B 1.3521 C 1.4752 D 1.3137

15 (R) Mali je Ivica svakog svakog ljeta u zadnjih sedam godina naučio jedan novi programski jezik. Svoja vrijedna iskustva sažeo je u listu *"Programski jezik koji mi se sviđa"*, gdje je svaki jezik opisao četirima značajkama, te je naznačio je li mu se dotični jezik svidio ($y = 1$) ili nije ($y = 0$). Ta lista izgleda ovako:

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

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

A 0.431 B 0.694 C 0.856 D 0.799

16 (P) Treniramo model stabla odluke na skupu podataka u kojemu, nažalost, ima i nešto šuma. Svjesni smo da prisustvo šuma može dovesti do prenaučenosti modela, pa smo odlučili primijeniti unakrsnu provjeru da bismo podrezali stablo odluke. Skup označenih primjera podijelili smo na skup za učenje D_u i skup za provjeru D_p, tako da $D_u \cap D_p = \varnothing$. Sada isprobavamo nekoliko različitih dubina stabla, od $d = 1$ do $d = 42$. Ispitivanjem pogreške tih stabala različite dubine, zaključili smo da je optimalna dubina stabla $d = 17$. **Što to konkretno znači?**

A Da je pogreška na skupu D_p za stablo sa $d = 17$ veća od pogreške za isto to stablo na skupu D_u, ali veća od pogreške za bilo koju drugu dubinu stabla d na skupu D_p

B Da je pogreška na skupu D_p za stablo sa $d = 17$ manja od pogreške na D_p za $d < 17$ i $d > 17$, ali očekivano ta je pogreška na D_p veća od pogreške na skupu D_u

C Da stablo za $d = 17$ ostvaruje najmanju pogrešku na skupu D_u, dok na skupu D_p stablo ostvaruje uvijek veću pogrešku, s maksimumom pogreške za $d = 1$

D Da su pogreške za stabla sa $d = 16$ i $d = 18$ veće i na skupu D_u i na skupu D_p, s time da su na skupu D_p očekivano veće nego na skupu D_u

17 (P) Rešetkasti svijet sastoji se od četiri ćelije, označene brojevima 1 do 4 i poredane vodoravno slijeva nadesno. Ćelija 4 je terminalna i agentu donosi 1 bod; nema drugih bodova. Svijet je okružen zidovima. Agent može poduzeti akcije *lijevo* i *desno*. Ako bi ga akcija dovela na zid, ostaje u ćeliji u kojoj je bio. Agent uvijek kreće iz ćelije 1. Provodimo postupak Q-učenja. Početni iznosi Q-funkcije su 0. Parametri su $\alpha = 0.5$ i $\gamma = 1$. Agent najprije igra tri epizode u kojima uvijek bira akciju *desno* i uči. Nakon toga igra još 10 epizoda u kojima u svakoj ćeliji pohlepno odabire potez koji mu se čini optimalan i također uči. **Nakon provedenih svih 13 epizoda, koja je očekivana vrijednost $q(3, desno)$?**

A Između 0.75 i 0.9995 B Veća od 0.9995 C Nije moguće odgovoriti D Manja od 0.75

6. Prirodom inspirirani optimizacijski algoritmi (3 pitanja)

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

A Ispitni predikat definiramo tako da uspoređuje stanje sa zadanim predloškom (kao kod slagalice)

B Rješenje omogućava rekonstrukciju puta do početnog stanja

C Bitno nam je samo konačno stanje

D Mogu se rješavati algoritmom A*, ali samo uz optimističnu heuristiku i skup posjećenih stanja

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

A 53 B 21 C 2 D 44

20 (P) Genetskim algoritmom tražimo maksimum funkcije $f(x, y, z)$, koristeći binarnu reprezentaciju rješenja. Vrijednost svake od varijabli pretražuje se u intervalu $[-100, 100]$, pri čemu je potrebno osigurati da se to pretraživanje provodi barem s preciznošću 0.001. **Od koliko se *minimalno* bitova treba sastojati kromosom?**

A 54 B 53 C 51 D 17

Uvod u umjetnu inteligenciju – pismeni ispit (2021./2022.)

– 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 igranje igara (4 pitanja)

1 (R) Neka su definirani skup stanja $S = \{a, b, c, d, e, f\}$ i funkcija sljedbenika $succ(a) = \{(b, 1), (c, 2)\}$, $succ(b) = succ(f) = \emptyset$, $succ(c) = \{(d, 3), (e, 5)\}$, $succ(d) = \{(b, 8), (f, 5)\}$, te $succ(e) = \{(d, 1), (f, 2)\}$. Heurističke vrijednosti čvorova neka su $h(a) = 7$, $h(b) = 4$, $h(c) = 6$, $h(d) = h(e) = 2$, $h(f) = 0$. Početno stanje neka je a, a ciljno f. Izvršite pretraživanje algoritmom A^*, bilježeći pritom sadržaje liste otvorenih čvorova O i zatvorenih čvorova C u svakom koraku algoritma (u nultom koraku $O = [(a, 0)]$ i $C = \emptyset$). **Koji su sadržaji listi O i C nakon petog koraka izvođenja algoritma A^*?**

- A $O = [(f, 9)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5), (e, 7)\}$
- B Algoritam ne dostiže peti korak
- C $O = [(e, 7), (f, 10)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5)\}$
- D $O = [(f, 10)]$, $C = \{(a, 0), (b, 1), (c, 2), (d, 5), (e, 7)\}$

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

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

3 (R) Stablo igre definirano je prijelazima $succ(A) = \{B, C\}$, $succ(B) = \{D, E\}$, $succ(D) = \{H, I\}$, $succ(E) = \{J, K, L\}$, $succ(C) = \{F, G\}$, $succ(F) = \{M, N, O\}$ te $succ(G) = \{P, Q\}$. Heurističke vrijednosti listova su $h(H) = -12$, $h(I) = -4$, $h(J) = -14$, $h(K) = 17$, $h(L) = 18$, $h(M) = -10$, $h(N) = -15$, $h(O) = -13$, $h(P) = 0$, $h(Q) = 11$. Optimalna strategija određuje algoritmom minimax uz alfa-beta podrezivanje. **Koji će čvorovi koji će pritom biti podrezani (preskočeni pri izračunu minimax vrijednosti)?**

- A L, G, P, Q
- B O, P, Q
- C L, Q
- D L, O, Q

4 (T) Algoritam minimax za čvorove stabla igre na dubini d poziva heurističku funkciju, koja procjenjuje minimax-vrijednost za dotični čvor. **Zašto algoritam minimax procjenjuje minimax-vrijednost, umjesto da izračuna stvarnu minimax-vrijednost?**

- A Izračun minimax-vijednosti nekog čvora iziskuje dosezanje završnih čvorova, a to je vremenski netraktabilno
- B Broj čvorova stabla igre raste eksponencijalno, pa stablo nije uvijek moguće pohraniti u memoriju
- C Minimax-vrijednost može se izračunati samo za završne čvorove, dok je za unutarnje čvorove treba procijeniti
- D Protivnički igrač ne mora igrati optimalno, pa je potrebno nanovo izračunati minimax-vrijednost svakog čvora

2. Prikazivanje znanja i automatsko zaključivanje (4 pitanja)

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

A 60 B 74 C 119 D 104

6 (T) Svaki formalizam za prikaz znanja ima svoje ontološke i epistemološke pretpostavke. **Što je epistemološka pretpostavka logike prvoga reda (FOL)?**

A Postoje objekti i relacije između njih

B Predikati definiraju svojstva i relacije

C Svaka formula je ili istinita ili lažna

D Skup mogućih interpretacija je konačan

7 (R) Zadane su premise: *"Ivo voli sve vrste hrane. Jabuka i piletina su hrana. Branko jede lješnjake i nije mu zlo od njih. Vesna jede sve što i Branko. Ako netko nešto jede i nije mu od toga zlo, onda je to hrana."* Formalizirajte ove premise u FOL i pretvorite ih u klauzule. Pritom koristite $V(x, y)$ za "x voli y", $J(x, y)$ za "x jede y", "$Z(x, y)$" za "x-u je zlo od y", i "$H(x)$" "x je hrana". Zatim rezolucijom opovrgavanjem uz strategiju skupa potpore dokažite cilj *Ivo voli lješnjake.* **Koliko je minimalno koraka (primjena rezolucijskog pravila) potrebno za dokaz?**

A 4 B 6 C 5 D 3

8 (T) Rezolucija opovrgavanjem uz faktorizaciju ispravno je i potpuno pravilo zaključivanja propozicijske logike (PL). **Što se događa ako radimo rezoluciju opovrgavanjem, ali ne provodimo faktorizaciju?**

A Za neke premise $F_1, \ldots, F_n$ i cilj G postupak će se zaustaviti prije nego što izvede NIL, premda $F_1, \ldots, F_n \vDash G$

B Ako $\vDash \neg(F_1 \land \cdots \land F_n \land \neg G)$, postupak primijenjen na premise $F_1, \ldots, F_n, \neg G$ nikada neće izvesti NIL

C Ako $\vDash F_1 \land \cdots \land F_n \to G$, postupak će iz premisa $F_1, \ldots, F_n, \neg G$ izvesti NIL, inače se može ne zaustaviti

D Za neke premise $F_1, \ldots, F_n$ i neku ciljnu formulu G postupak će izvesti NIL, premda $F_1, \ldots, F_n \nvDash G$

3. Logičko programiranje i ekspertni sustavi (2 pitanja)

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

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

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

A 6 B 8 C 12 D 10

10 (T) Širој primjeni ekspertnih sustava značajno je doprinio razvoj ljusaka ekspertnih sustava (engl. *expert system shell*), kao što je CLIPS. **Što je ljuska ekspertnog sustava?**

A Mehanizam za zaključivanje na temelju pravila i za uređivanje baze znanje, bez same baze znanja

B Skup pravila i činjenica koje opisuju stručno znanje u nekoj zatvorenoj domeni

C Cjelokupni ekspertni sustav, uključivo baza znanja, mehanizam za zaključivanje i korisničko sučelje

D Korisničko sučelje koje podržava objašnjavanje zaključaka, bez mehanizma za zaključivanje i baze znanja

4. Modeliranje neizvjesnosti (2 pitanja)

11 (R) U sustavu neizrazitog zaključivanja modeliramo vezu između dijabetesa i sklonosti depresiji. Skup G je prosječna razina glukoze u krvi u mmol/L, $G = \{1, 2, \ldots, 11\}$, a skup D je stupanj depresije prema Beckovom upitniku depresije, $D = \{0, 1, \ldots, 63\}$. Nad ovim univerzalnim skupovima definirali smo neizrazite skupove "predijabetes" kao $G_p = \{0.3/5, 1/6, 0.1/7\}$ te "umjerena depresija" kao $D_u = \{0.5/15, 0.75/16, 1/17, 0.5/18\}$. U sustavu imamo pravilo "ako manje-više(G_p), onda D_u", modelirano kao neizrazita relacija. Zanima nas kolika je sklonost depresiji osoba čija razina glukoze u krvi odgovara neizrazitom skupu $G' = \{0.7/5, 1/6, 0.3/7, 0.1/8\}$. **Kako glasi neizraziti skup D' koji dobivamo generaliziranim modusom ponensom?**

A $\{\sqrt{0.1}/15, 0.3/16, 0.75/17, 0.3/18\}$ C $\{0.3/15, 0.75/16, 0.75/17, \sqrt{0.1}/18\}$

B $\{0.5/15, 0.75/16, 1/17, 0.5/18\}$ D $\{0.25/15, 0.75/16, 1/17, 0.75/18\}$

12 (T) Bayesovo pravilo može se koristiti za modeliranje zaključivanja s više dokaza. U tom slučaju tipično uvodimo pretpostavku da su dokazi međusobno uvjetno nezavisni. **Zašto uvodimo tu pretpostavku?**

A Bez pretpostavke o uvjetnoj nezavisnosti dokaza nazivnik Bayesovog pravila nije definiran

B Ako postoje više od dvije hipoteze, iz uvjeta međusobne isključivosti hipoteza slijedi da dokazi moraju biti uvjetno nezavisni za danu hipotezu

C Dokazi su u stvarnosti uglavnom uvjetno nezavisni, pa uvođenjem te pretpostavke procjena aposteriorne vjerojatnosti postaje točnija

D Ako su dokazi uvjetno nezavisni, broj vjerojatnosti koje trebamo procijeniti mnogo je manji nego kada dokazi nisu uvjetno nezavisni

5. Strojno učenje, umjetne neuronske mreže i podržano učenje (5 pitanja)

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

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

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

A 1.2627 B 1.4767 C 1.3137 D 1.3521

14 (T) Umjetne neuronske mreže sastoje se od više neurona s nelinearnim prijenosnim funkcijama. Takve mreže tipično učimo algoritmom propagacije pogreške unazad (algoritam BP). **Što bi se dogodilo da, umjesto više neurona, koristimo jedan neuron, i to sa sigmoidnom prijenosnom funkcijom?**

A Mogli bismo koristiti algoritam BP, ali bismo mogli riješiti samo linearno odvojive probleme

B Mogli bismo odvojiti linearno neodvojive klase, ali ne bismo mogli koristiti algoritam BP

C Granica između klasa bila ista kao i s više neurona, ali bi mreža lošije generalizirala na neviđene primjere

D Granica između klasa bila bi nelinearna, ali manje nelinearna nego da koristimo više neurona

15 (R) Mali je Ivica svakog svakog ljeta u zadnjih sedam godina naučio jedan novi programski jezik. Svoja vrijedna iskustva sažeo je u listu *"Programski jezik koji mi se sviđa"*, gdje je svaki jezik opisao četirima značajkama, te je naznačio je li mu se dotični jezik svidio ($y = 1$) ili nije ($y = 0$). Ta lista izgleda ovako:

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

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

A 0.856 B 0.431 C 0.799 D 0.694

16 (P) Treniramo model stabla odluke na skupu podataka u kojemu, nažalost, ima i nešto šuma. Svjesni smo da prisustvo šuma može dovesti do prenaučenosti modela, pa smo odlučili primijeniti unakrsnu provjeru da bismo podrezali stablo odluke. Skup označenih primjera podijelili smo na skup za učenje D_u i skup za provjeru D_p, tako da $D_u \cap D_p = \varnothing$. Sada isprobavamo nekoliko različitih dubina stabla, od $d = 1$ do $d = 42$. Ispitivanjem pogreške tih stabala različite dubine, zaključili smo da je optimalna dubina stabla $d = 17$. **Što to konkretno znači?**

A Da je pogreška na skupu D_p za stablo sa $d = 17$ veća od pogreške za isto to stablo na skupu D_u, ali veća od pogreške za bilo koju drugu dubinu stabla d na skupu D_p

B Da stablo za $d = 17$ ostvaruje najmanju pogrešku na skupu D_u, dok na skupu D_p stablo ostvaruje uvijek veću pogrešku, s maksimumom pogreške za $d = 1$

C Da je pogreška na skupu D_p za stablo sa $d = 17$ manja od pogreške na D_p za $d < 17$ i $d > 17$, ali očekivano ta je pogreška na D_p veća od pogreške na skupu D_u

D Da su pogreške za stabla sa $d = 16$ i $d = 18$ veće i na skupu D_u i na skupu D_p, s time da su na skupu D_p očekivano veće nego na skupu D_u

17 (P) Rešetkasti svijet sastoji se od četiri ćelije, označene brojevima 1 do 4 i poredane vodoravno slijeva nadesno. Ćelija 4 je terminalna i agentu donosi 1 bod; nema drugih bodova. Svijet je okružen zidovima. Agent može poduzeti akcije *lijevo* i *desno*. Ako bi ga akcija dovela na zid, ostaje u ćeliji u kojoj je bio. Agent uvijek kreće iz ćelije 1. Provodimo postupak Q-učenja. Početni iznosi Q-funkcije su 0. Parametri su $\alpha = 0.5$ i $\gamma = 1$. Agent najprije igra tri epizode u kojima uvijek bira akciju *desno* i uči. Nakon toga igra još 10 epizoda u kojima u svakoj ćeliji pohlepno odabire potez koji mu se čini optimalan i također uči. **Nakon provedenih svih 13 epizoda, koja je očekivana vrijednost** $q(3, \textit{desno})$**?**

A Veća od 0.9995 B Nije moguće odgovoriti C Manja od 0.75 D Između 0.75 i 0.9995

6. Prirodom inspirirani optimizacijski algoritmi (3 pitanja)

18 (P) Algoritmom kolonije mrava rješavamo problem trgovačkog putnika nad odabranim gradovima u Republici Hrvatskoj. Za algoritam su poznati sljedeći parametri: $\tau_0 = 40$, $\alpha = 2$, $\beta = 3$, $\rho = 0.15$, kolonija se sastoji od 55 mrava te ažuriranje u svakoj epohi obavlja samo mrav koji je u toj epohi pronašao najbolje rješenje. Razmotrimo brid koji povezuje gradove Zagreb i Dubrovnik. Pretpostavimo da niti u jednoj epohi ta dva grada nisu sljedbenici u pronađenom putu. **Nakon koliko će epoha algoritma količina feromonskih tragova na spomenutom bridu postati manja od 0.1?**

A 37 B 43 C 52 D 4

19 (P) Genetskim algoritmom tražimo maksimum funkcije $f(x, y, z)$, koristeći binarnu reprezentaciju rješenja. Vrijednost svake od varijabli pretražuje se u intervalu $[-100, 100]$, pri čemu je potrebno osigurati da se to pretraživanje provodi barem s preciznošću 0.001. **Od koliko se *minimalno* bitova treba sastojati kromosom?**

A 54 B 51 C 53 D 17

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

A Ispitni predikat definiramo tako da uspoređuje stanje sa zadanim predloškom (kao kod slagalice)

B Rješenje omogućava rekonstrukciju puta do početnog stanja

C Mogu se rješavati algoritmom A*, ali samo uz optimističnu heuristiku i skup posjećenih stanja

D Bitno nam je samo konačno stanje

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