Examen 2023

1. Se consideră problema blocurilor (cu S stive și NB blocuri; S și NB naturale, S>=3, NB>=4, NB par) cu următoarele modificări:

- NB-1 blocuri conțin o informație numerică de forma 0 sau 1.
- Există un bloc special, numit blocul "not", peste care nu poate exista niciodată un alt element și care nu se poate găsi niciodată pe stivele din capetele configurației
- Ca la problema blocurilor clasică, putem **muta** un bloc dintr-un vârf de stivă pe alt vârf de stivă, cu următoarele restricții:
  - Blocul "not" se poate muta pe orice stivă mai puțin cele din capete (prima și ultima).
  - Când blocul "not" este mutat pe o stivă nevidă, valoarea tuturor blocurilor de pe acea stivă se modifică din 0 în 1 și din 1 în 0.
  - Peste blocul "not" nu putem muta niciun bloc numeric.
- **Costul** mutării unui *bloc numeric* este egal cu numărul de pe bloc + 1 (blocurile 0 au costul 1, blocurile 1 au costul 2). Costul mutării blocului "not" este 1 + nr_schimbari, unde nr_schimbari e numărul de blocuri numerice afectate de mutarea sa.
- **Scopul** este să ajungem la o configurație în care există exact o stivă doar cu blocuri 0, exact o stivă doar cu blocuri 1, iar ambele stive conțin exact NB/2 blocuri (NB precizat la început).

Notăm cu SV numărul de stive vide.

Care dintre următoarele afirmații sunt adevărate?

a. **Pentru S>=4, nu există nicio stare inițială pentru care să nu existe soluție.**
b. **Pentru S=3, există măcar o stare inițială în mulțimea de stări inițiale posibile (cu parametrii conform condițiilor din enunț) pentru care să nu putem găsi soluție folosind A*.**
c. Numărul de succesori valizi ai unei stări oarecare este 2*(S-SV-2)
d. **Presupunând că blocul "not" se află singur pe o stivă (sub el nu mai e alt bloc). Numărul de succesori valizi ai unei stări oarecare este (S-SV-1)*(S-2)+S-3**
e. Presupunând că blocul "not" se află singur pe o stivă (sub el nu mai e alt bloc). Numărul de succesori valizi ai unei stări oarecare este (S-SV-2)*(S-1)+S-3
f. Presupunând că blocul "not" se află singur pe o stivă (sub el nu mai e alt bloc). Numărul de succesori valizi ai unei stări oarecare este (S-1)+(S-SV)*(S-1)
g. **Presupunând că blocul "not" se află singur pe o stivă (sub el nu mai e alt bloc). Numărul de succesori valizi ai unei stări oarecare este (S-SV)*(S-2)-1**
h. Presupunând că blocul "not" se află singur pe o stivă (sub el nu mai e alt bloc). Numărul de succesori valizi ai unei stări oarecare este (S-SV)*(S-2)+1
i. Pentru S=3 nu putem avea o stare inițială conținând în configurație o stivă vidă.
j. niciuna dintre opțiunile de mai sus nu e corectă.

Using Custom Setup

2. Considerăm enunțul problemei anterioare. Care este o estimație admisibilă pentru o stare?

a. **S-SV-3, unde SV e numărul de stive vide din starea respectivă.**
b. SV, unde SV e numărul de stive vide din starea respectivă.
c. |Nr(0)-Nr(1)|, unde Nr(x) e numărul de blocuri cu valoarea x.
d. **|Nr(0)-Nr(1)|/2, unde Nr(x) e numărul de blocuri cu valoarea x.**
e. **0 indiferent de stare.**
f. **1 dacă starea nu e scop și 0 dacă e stare scop.**
g. 1 indiferent de stare.
h. niciuna dintre opțiunile de mai sus nu e corectă.

3. Considerăm un arbore minimax de adâncime maximă A și N noduri (inclusiv rădăcina) generat pentru un joc oarecare. Care dintre următoarele afirmații sunt adevărate?

a. **Un nod MIN nu poate fi frate cu un nod MAX**
b. Dacă aplicăm alpha-beta pe aceeași stare rădăcină putem să evităm calcularea a maxim N/2 noduri
c. **Dacă aplicăm alpha-beta pe aceeași stare rădăcină putem să evităm calcularea a maxim N-1 noduri**
d. Dacă aplicăm alpha-beta pe aceeași stare rădăcină putem întotdeauna să evităm calcularea a minim unui nod
e. **Alpha-beta nu va elimina (reteza) niciodată primul fiu al unui nod.**
f. Frunzele sunt întotdeauna noduri MIN.
g. Frunzele sunt întotdeauna noduri MAX.
h. niciuna dintre opțiunile de mai sus nu e corectă.

5. Pentru *N* număr natural (5 ≤ *N* ≤ 10) considerăm o tablă de joc de dimensiune NxN în care sunt *NL* lăzi notate pe hartă cu câte un număr *G* reprezentând greutatea fiecărei lăzi. *NL* și *G* sunt numere naturale nenule. Pe hartă avem și un omuleț notat cu *X*, așa cum se vede în desen (desenul arată doar un exemplu de stare inițială; exercițiul nu se referă doar la acest caz specific de stare inițială).

	3			10
				8
	5			
X	14			

Scopul omulețului este să împingă un număr K de lăzi (K natural nenul, K <= NL) în colțul din stânga sus al matricei (poziția 0,0). Omulețul poate împinge o ladă doar cu câte o poziție și doar într-un spațiu liber. În plus, el trebuie să fie în spațiul vecin lăzii pe aceeași direcție dar în sens opus față de sensul mutării (de exemplu, pentru a muta o ladă în dreapta, omulețul trebuie să fie în stânga ei; cum e pe desen în cazul lăzii 14). În starea inițială pe poziția (0,0) nu poate exista nicio ladă și aceasta nu poate reprezenta nici poziția inițială a omulețului. O ladă ajunsă (împinsă) în locația (0,0) dispare. Omulețul se poate deplasa și poate muta lăzi doar cu câte o poziție în direcțiile: sus, jos, stânga și dreapta (nu și pe diagonală). Fiecare deplasare **costă**:

- 1 dacă omulețul nu împinge o ladă
- greutatea lăzii dacă omulețul împinge o ladă

Notații:

- Coordonatele omulețului: linie(om), coloana(om).
- Coordonatele unei lăzi: linie(lada[i]) și coloana(lada[i]).
- Distanța Manhattan: distMH(linie1,coloana1,line2,coloana2)=|linie1-linie2|+|coloana1-coloana2|
- Greutatea lăzii x: greutate(x)

Care dintre estimațiile de mai jos sunt admisibile pentru această problemă?

a. **Dacă omulețul a scos deja K1 lăzi din hartă în starea curentă, o estimație admisibilă este K - K1**
b. O estimație admisibila este distMH(linie(om), coloana(om), 0, 0) indiferent de câte lăzi au fost scoase de pe hartă
c. **Notând cu min_lazi={greutatea minimă dintre greutățile lăzilor aflate pe hartă), și cu dist_min={cea mai mică distanță Manhattan la care se afla o ladă față de colțul stânga-sus}, o estimație admisibilă este dist_min*min_lazi dacă omulețul mai are de scos lăzi din configurație și 0 dacă nu.**
d. Notând cu min_lazi={greutatea minimă dintre greutățile lăzilor aflate pe hartă), și cu dist_min={cea mai mica distanta Manhattan la care se afla o lada de colțul stânga-sus}, o estimație admisibilă este dist_min*min_lazi*K
e. **Pentru NL>=4 există o configurație inițială (în spațiu stărilor posibile) fără soluție.**
f. Pentru NL=1 problema întotdeauna are soluție.
g. O estimație admisibila pentru o stare în care omul mai are de scos un număr nenul de lăzi, este distMH(linie(om), coloana(om), linie(lada_min), coloana(lada_min))*greutate(lada_min) , unde lada_min e lada cu cea mai mică greutate din configurație
h. niciuna dintre opțiunile de mai sus nu e corectă.

Using Custom Setup

5. Care dintre următoarele afirmații sunt adevărate pentru problema X și 0? (enunț în anexă)

a. Pentru starea reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 6=3!

x	0	
	x	
x	0	

b. Pentru starea reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 24=4!

x	0	
	x	
x	0	

c. Pentru starea reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, adâncimea arborelui este 1

x	0	
	x	
x	0	

d. Pentru starea reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, adâncimea arborelui este 1

0	**x**	**x**
	0	**x**
0	**x**	

e. Pentru starea reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 28

x	**0**	**x**
	x	
	0	

f. O stare finală a jocului are ca proprietate că fie a câștigat X, fie a câștigat 0.

g. Arborele minimax pentru X și 0 întotdeauna va avea cel mult 10 niveluri.

h. Pentru starea reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 5

x	0	x
0	x	
	0	x

i. Un nod frunză din arborele Minimax, pentru jocul x și 0 conține întotdeauna o tablă fără locuri libere (tabla e în întregime completată).

j. niciuna dintre opțiunile de mai sus nu e corectă.

Using Custom Setup

6. Se dă o problemă de căutare cu costuri strict pozitive pe tranziții. Se dau o estimație *admisibilă* ĥ1(nod) și o estimație *neadmisibilă* ĥ2(nod), oarecare. Care dintre formulele următoare ar obține din ĥ1 și ĥ2 o nouă estimație ĥ(nod) cu siguranță **neadmisibilă**?

Observație: Formula trebuie să fie adevărată pentru orice ĥ1, orice ĥ2 și orice nod din graf, nu doar pentru cazuri particulare. Știm că ĥ1(nod) ≥ 0 și ĥ2(nod) ≥ 0 pentru orice nod din graf.

- **a. ĥ(nod) = ĥ2(nod) + ĥ1(nod) / 3**
- b. ĥ(nod) = ĥ1(nod) + ĥ2(nod) / 3
- c. ĥ(nod) = ĥ2(nod) - ĥ1(nod)

- **d. ĥ(nod) = 2*ĥ2(nod) - ĥ1(nod)**
- **e. ĥ(nod) = max(ĥ2(nod), ĥ1(nod))**
- f. ĥ(nod) = min(ĥ2(nod), ĥ1(nod))
- g. ĥ(nod) = (ĥ1(nod) + ĥ2(nod)) / 2
- h. ĥ(nod) = ĥ1(nod) * ĥ2(nod)
- i. niciuna dintre formulele de mai sus nu e corectă

7. Considerăm următorul joc:

Pentru N număr natural (5 ≤ N ≤ 10) considerăm o tablă de joc de dimensiune NxN și 2 jucători (X și 0). Jucătorului X îi corespunde prima linie, iar lui 0 ultima. Jucătorul X mută primul.

O mutare constă în una dintre următoarele acțiuni:

1. Plasarea unui simbol nou pe linia proprie.
2. Mutarea unui simbol cu o singură poziție pe orizontală (în stânga sau dreapta) sau pe verticală, înspre linia adversarului (în jos pentru x și în sus pentru 0).

Restricții de mutare:

1. Nu pot exista mai mult de 2 simboluri ale aceluiași jucător pe vreo linie
2. Plasarea unui simbol nou se poate realiza doar dacă pe coloana respectivă nu mai sunt simboluri proprii.

Jucătorul *j* poate captura o piesă a adversarului, *ja*, dacă piesa adversarului se află între două piese ale jucătorului *j* aflate pe aceeași linie cu aceasta, iar cele două piese ale jucătorului *j* au între ele cel mult N/3 spații.

Un jucător **câștigă** fie când a capturat un număr K_MAX > 3 de simboluri, fie când a ajuns cu un simbol pe linia adversarului.

Care dintre variantele de mai jos oferă o funcție de evaluare care să arate în mod corect cât de favorabilă este starea jocului pentru calculator (MAX) (să aibă o valoare mai mare pentru stări mai favorabile și mai mică pentru stări mai nefavorabile) ?

- a. Numărul de simboluri ale lui MAX de pe linii pare din care scădem numărul de simboluri ale lui MIN de pe coloane impare.
- b. Câte configurații de 3 simboluri pe poziții consecutive (pe linie/coloană/diagonală) are MAX din care scădem câte configurații de 3 simboluri are MIN
- c. **Numărul de simboluri capturate ale lui MAX din care scădem numărul de simboluri capturate ale lui MIN.**
- **d. Considerăm LMIN linia celui mai depărtat simbol al lui MIN de linia sa de start și START_MIN, linia de start a lui MIN. Considerăm LMAX linia celui mai depărtat simbol al lui MAX de linia sa de start și START_MAX, linia de start a lui MAX. Atunci funcția de evaluare ar fi | LMAX-START_MAX | - | LMIN-START_MIN |**
- e. Numărul de simboluri ale lui MAX la care adunăm numărul de simboluri ale lui MIN.
- f. Numărul de simboluri capturate ale lui MAX înmulțite numărul de simboluri capturate ale lui MIN.
- g. Numărul de locuri libere de pe tablă.

8. Pentru graful din imagine avem stările cozii OPEN **ordonate** pentru algoritmul A* de mai jos. În fiecare stare sunt precizate doar informațiile nodurilor din coadă. Considerăm nodul **start a**, iar nodul **scop f**. Prima stare e cea inițială a cozii OPEN (cu nodul start inclus). Toate celelalte stări sunt afișate în urma repetării următoarelor acțiuni:

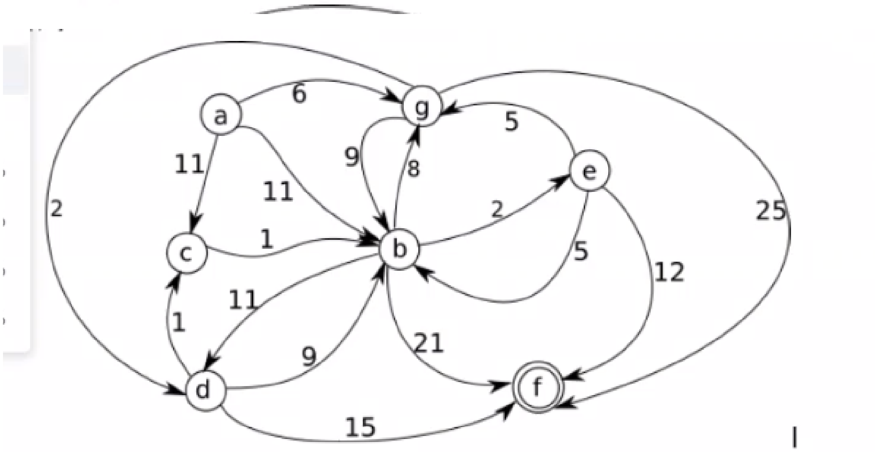
1) Extrage nodul cu f̂ minim (nu se afișează coada pentru această stare intermediară)

2) Adaugă succesorii în coadă astfel încât coada să fie mereu ordonată crescător după f̂ (după această acțiune se afișează coada).

Observatie: Pentru valori f̂ egale, elementele sunt în ordinea descrescătoare a valorii g. În cazul în care și f̂ și g sunt egale, nodurile se ordonează după informație. În cazul în care un succesor are deja informația în CLOSED, dar găsim un cost mai bun pentru aceeași informație, îl mutăm înapoi în OPEN pentru expandare.

Stările cozii OPEN (numărul din stânga e numărul de ordine al stării):

0: [a]
1: [c, b, g]
2: [b, g]
3: [g, e, f, d]
4: [d, e, f]
5: [c, f, e]
6: [b, f, e]
7: [f, e]



Care dintre următoarele afirmații sunt adevărate pentru **graful dat și stările cozii OPEN de mai sus**?

a. Estimația (cu valorile scrise între paranteze drepte pentru fiecare nod) a[10] b[1] c[3] d[5] e[4] f[0] g[6]) este admisibilă și în concordanță cu coada OPEN.

b. Estimația (cu valorile scrise între paranteze drepte pentru fiecare nod) a[10] b[10] c[3] d[10] e[11] f[0] g[17]) este admisibilă și în concondanță cu coada OPEN.

c. Estimația (cu valorile scrise între paranteze drepte pentru fiecare nod) a[0] b[0] c[0] d[0] e[0] f[0] g[0]) este admisibilă și în concordanță cu coada OPEN.

d. Estimația (cu valorile scrise între paranteze drepte pentru fiecare nod) a[1] b[1] c[1] d[1] e[1] f[1] g[1] este admisibilă și în concordanță cu coada OPEN.

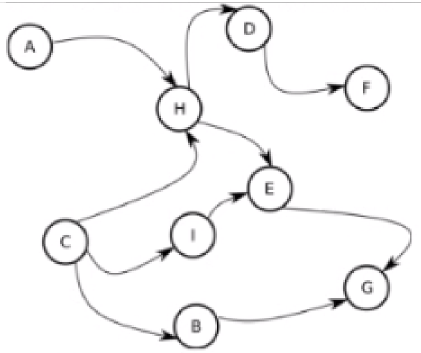
e. Presupunând că orice nod extras din OPEN e imediat adăugat în CLOSED, în starea 3 avem în lista CLOSED (nu neapărat în ordinea asta) nodurile a,b,c

f. În trecerea de la starea 2 la starea 3 a cozii OPEN se expandează nodul b

g. În trecerea de la starea 6 la starea 7 a cozii OPEN se expandează nodul f

h. Niciuna dintre opțiunile de mai sus nu e corectă.

9. Se da urmatoarea topoligie de retea Bayesiana:

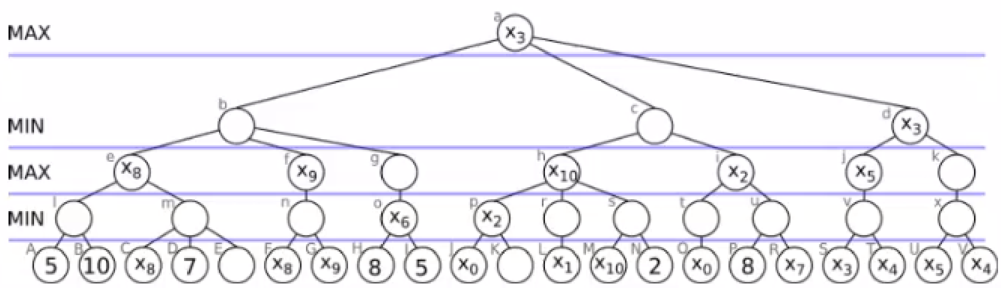


Care dintre următoarele afirmații sunt adevărate?

a. Mulțimea {I,B,E} d-separă mulțimea {C} de mulțimea {E,G}.
b. Dacă am adăuga arcul G->C graful nu ar mai fi o topologie corectă pentru o rețea Bayesiana.Mulțimea {I,B,E} d-separă mulțimea {C} de mulțimea {E,G}.
c. Graful dat nu este o topologie corectă pentru o rețea Bayesiană.
d. **Dacă am adăuga arcul C->A graful ar fi în continuare o topologie corectă pentru o rețea Bayesiana.Mulțimea {D} d-separă mulțimea {A} de mulțimea {F}**
e. **Dacă am adăuga arcul G->C graful nu ar mai fi o topologie corectă pentru o rețea Bayesiana.**
f. **Drumul de la nodul A la nodul F e blocat conditionat de mulțimea {D}**
g. Dacă am șterge nodul H, graful nu ar mai fi o topologie corectă pentru o rețea Bayesiana.Mulțimea {I,B,E} d-separă mulțimea {C} de mulțimea {E,G}.
h. **Mulțimea {D} d-separă mulțimea {A} de mulțimea {F}**
i. niciuna dintre opțiunile de mai sus nu e corectă.

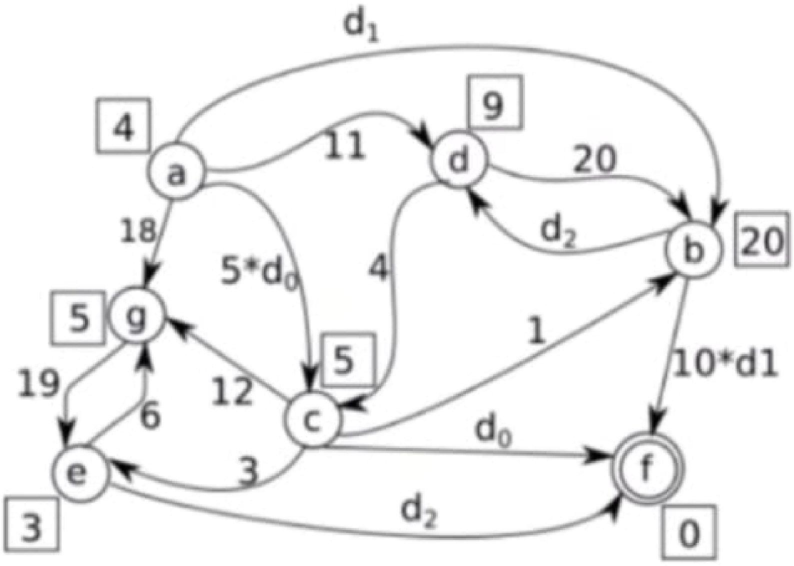
10. Se consideră arborele minimax din imagine, cu adâncime maximă 4 (rădăcina fiind considerată la adâncime 0). Se presupune că arborele a fost deja generat prin minimax, iar unele valori minimax au fost, apoi, fie șterse din imagine (nodurile fără conținut), fie înlocuite cu identificatori xi (două noduri cu același identificator xi au valorile minimax egale, însă putem avea xi==xj pentu i diferit de j). Care dintre afirmațiile de mai jos sunt sigur adevărate având în vedere informația dată despre arbore?

Observație: frunzele sunt notate cu litere mari iar nodurile interne cu litere mici.



a. **x2 este sigur egal cu x0**
b. x2 este sigur mai mare sau egal cu 8
c. **x10 e sigur mai mare sau egal cu x1**
d. **nodul c (literă mică) va avea sigur valoarea x2**
e. nodul c (literă mică) va avea sigur valoarea x10
f. **nodul k (literă mică) va avea sigur valoarea x5**
g. **valoarea x5 este întotdeauna egală cu valoarea x3**
h. valoarea x5 este întotdeauna egală cu valoarea x4
i. **x6 sigur are valoarea 5**
j. x7 este sigur mai mic strict decât x2
k. **x8 aparține sigur intervalului [5,7]**
l. **nodul b are sigur valoarea x9**
m. x3 este cea mai mare valoare dintre toate valorile nodurilor arborelui
n. niciuna dintre opțiunile de mai sus nu e corectă.

11. Se dă graful orientat cu arce ponderate din imagine. Pentru unele arce ponderile nu sunt precizate, fiind înlocuite de identificatorii di. Euristica dată este una admisibilă (estimația pentru fiecare nod e trecută în pătrățelul de lângă nod).
Costurile arcelor sunt numere naturale nenule.
Nodul start este a și nodul f este nod scop.
Drumul de cost minim returnat de A* pentru nodul de start a este a->b->d->c->f cu costul 14 și este unicul drum de acest cost.



Care dintre afirmațiile de mai jos sunt adevărate:

a. d0 are valoarea 5
b. d0 poate avea orice valoare mai mare sau egală cu 6.
c. d2 este 3
d. d2 este întotdeauna mai mic strict decât 2
e. d1+d2=5 întotdeauna
f. d1 este 4 și d2 este 1
g. d1 este 1
h. d1 este 2
i. există în graf un drum cu costul 15
j. niciuna dintre variantele de mai sus nu este corectă

12. Care dintre următoarele afirmații sunt adevărate pentru rețele Bayesiene (și topologiile lor)?

a. **O proprietate a topologiei unei rețele Bayesiene este faptul că este un graf orientat (direcționat).**
b. **Într-o rețea Bayesiană nu pot exista două noduri n_1 și n_2 cu proprietatea că există un drum de la n_1 la n_2 și în același timp există un drum de la n_2 la n_1.**
c. Întotdeauna gradul interior (numărul de arce care intră în nod) al unui nod este mai mic sau egal cu 1.
d. Întotdeauna gradul exterior al oricărui nod (numărul de arce care ies din nod) este mai mare sau egal cu gradul său interior (numărul de arce care intră în nod).
e. Orice graf orientat cu arce având asociate ponderi numere reale poate reprezenta topologia unei rețele Bayesiene.
f. **O proprietate a topologiei unei rețele Bayesiene este faptul că este un graf fără circuite (aciclic).**
g. O proprietate a topologiei unei rețele Bayesiene este faptul că este un arbore.
h. **Nodurile unei rețele Bayesiene sunt variabile aleatoare.**
i. Fiecărei conexiuni dintre nodurile unei rețele Bayesiene îi corespunde un tabel de probabilități condiționate care cuantifică efectele pe care părinții le au asupra nodului respectiv

j. **Fiecărui nod al unei rețele Bayesiene îi corespunde un tabel de probabilități condiționate care cuantifică efectele pe care părinții le au asupra nodului respectiv**
k. niciuna dintre opțiunile de mai sus nu e corectă.

(V1.2022) 1. Pentru problema blocurilor (vezi anexa), considerând o stare cu N stive, B blocuri în total și NV stive vide, care este numărul de succesori direcți ai acesteia (stări în care se poate ajunge din starea curentă printr-o singură mutare)?

a. N * (N - 1)
b. (N - 1) * NV
c. (N - NV - 1) * N
d. (N - NV) * (N - 1)
e. (N - NV) * (N - NV)

(V2.2022) 4. Pentru problema blocurilor (vezi anexa), considerând o stare cu N stive și în care niciuna dintre stive nu e vidă, care este numărul de succesori direcți ai acesteia (stări în care se poate ajunge din starea curentă printr-o singură mutare)?

a. N * (N - 1)
b. N - 1
c. N * N
d. (N - 1) * (N - 1)
e. (N - 1) * (N + 1)

(V3.2022) 9. Pentru problema blocurilor (vezi anexa), considerăm o stare cu N stive, dintre care NB sunt stive nevide, și în configurație există un total de B blocuri. Care este numărul de succesori direcți ai acesteia (stări în care se poate ajunge din starea curentă printr-o singură mutare)?

a. N * B
b. N
c. N * NB
d. (N - 1) * NB
e. N * (NB - 1)

(V4.2022) 4. Pentru problema blocurilor (vezi anexa), considerând o stare cu N stive, B blocuri în total și NV stive vide, care este numărul de succesori direcți ai acesteia (stări în care se poate ajunge din starea curentă printr-o singură mutare)?

a. NV * (N - 1)
b. (N - 1) * (NV - 1)
c. (N - NV - 1) * N
d. (N - 1) * N
e. (N - 1) * (N - NV)
f. (N - NV) * (N - NV)

(V5.2022) 6. Pentru problema blocurilor (vezi anexa), considerăm o stare cu N stive, dintre care NB sunt stive nevide, și în configurație există un total de B blocuri. Care este numărul de succesori direcți ai acesteia (stări în care se poate ajunge din starea curentă printr-o singură mutare)?

a. (N - 1) * B
b. N
c. N * NB
d. (N - 1) * (NB - 1)
e. (N - 1) * NB
f. N * (NB - 1)

(V1.2022) 4. Care dintre afirmațiile de mai jos sunt adevărate?

a. Un sistem expert, pe baza informațiilor primite, oferă întotdeauna răspunsuri în totalitate certe.
b. Un sistem expert trebuie să poată genera explicația (demonstrația) pentru un răspuns dat.
c. Metoda înlănțuirii înapoi presupune o căutare a răspunsului pornind de la date în încercarea de a găsi scopul.
d. Un sistem expert este întotdeauna proiectat pentru a răspunde la întrebări din orice domeniu.
e. Un sistem expert poate funcționa și fără o bază de cunoștințe.

(V2.2022) 3. Care dintre afirmațiile de mai jos sunt adevărate?

a. Metoda înlănțuirii înapoi presupune o căutare a răspunsului pornind de la date în încercarea de a găsi scopul.
b. Un sistem expert, pe baza informațiilor primite, oferă întotdeauna răspunsuri în totalitate certe.
c. Un sistem expert este întotdeauna proiectat pentru a răspunde la întrebări din orice domeniu.
d. **Un sistem expert trebuie să poată genera explicația (demonstrația) pentru un răspuns dat.**
e. Un sistem expert poate funcționa și fără o bază de cunoștințe.

(V3.2022) 2. Care dintre afirmațiile de mai jos sunt adevărate?

a. Metoda înlănțuirii înapoi presupune o căutare a răspunsului pornind de la date în încercarea de a găsi scopul.
b. Un sistem expert poate funcționa și fără o bază de cunoștințe.
c. Un sistem expert, pe baza informațiilor primite, oferă întotdeauna răspunsuri în totalitate certe.
d. Un sistem expert este întotdeauna proiectat pentru a răspunde la întrebări din orice domeniu.
e. **Un sistem expert trebuie să poată genera explicația (demonstrația) pentru un răspuns dat.**

(V4.2022) 1. Care dintre afirmațiile de mai jos sunt adevărate?

a. Metoda înlănțuirii înapoi presupune o căutare a răspunsului pornind de la date în încercarea de a găsi scopul.
b. **Un sistem expert trebuie să poată genera explicația (demonstrația) pentru un răspuns dat.**
c. Un sistem expert poate funcționa și fără o bază de cunoștințe.
d. Un sistem expert este întotdeauna proiectat pentru a răspunde la întrebări din orice domeniu.
e. Un sistem expert, pe baza informațiilor primite, oferă întotdeauna răspunsuri în totalitate certe.

(V5.2022) 2. Care dintre afirmațiile de mai jos sunt adevărate?

a. Metoda înlănțuirii înapoi presupune o căutare a răspunsului pornind de la date în încercarea de a găsi scopul.
b. Un sistem expert, pe baza informațiilor primite, oferă întotdeauna răspunsuri în totalitate certe.
c. **Un sistem expert trebuie să poată genera explicația (demonstrația) pentru un răspuns dat.**
d. Un sistem expert poate funcționa și fără o bază de cunoștințe.
e. Un sistem expert este întotdeauna proiectat pentru a răspunde la întrebări din orice domeniu.

(V1.2022) 5. Să presupunem că avem o problemă de căutare P pentru care vrem să aplicăm algoritmul A*. Graful asociat problemei este un graf orientat oarecare G = (N, A) cu arce ponderate (unde N este mulțimea nodurilor și A este mulțimea arcelor). Ponderile sunt **numere raționale pozitive nenule**. Care dintre următoarele estimații $\hat{h}$ sunt admisibile **pentru orice astfel de graf**?

a. $\hat{h}$(n) = 1 dacă *n* nu este nod scop, $\hat{h}$(n) = 0 dacă n este nod scop
b. $\hat{h}$(n) = 0 oricare ar fi nodul *n* din graf
c. Pentru orice nod *n*, $\hat{h}$(n) = min({ct| ct cost al arcului a, a ∈ A}) (altfel spus, ct este costul minim pe un arc din graf)
d. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care iese din nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop
e. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care intră în nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop

(V2.2022) 2. Să presupunem că avem o problemă de căutare P pentru care vrem să aplicăm algoritmul A*. Graful asociat problemei este un graf orientat oarecare G = (N, A) cu arce ponderate (unde N este mulțimea nodurilor și A este mulțimea arcelor). Ponderile sunt **numere raționale pozitive nenule**. Care dintre următoarele estimații $\hat{h}$ sunt admisibile **pentru orice astfel de graf**?

a. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care intră în nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop
b. $\hat{h}$(n) = 1 dacă *n* nu este nod scop, $\hat{h}$(n) = 0 dacă n este nod scop
c. $\hat{h}$(n) = 0 oricare ar fi nodul *n* din graf
d. Pentru orice nod *n*, $\hat{h}$(n) = min({ct| ct cost al arcului a, a ∈ A}) (altfel spus, ct este costul minim pe un arc din graf)
e. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care iese din nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop

(V3.2022) 3. Să presupunem că avem o problemă de căutare P pentru care vrem să aplicăm algoritmul A*. Graful asociat problemei este un graf orientat oarecare G = (N, A) cu arce ponderate (unde N este mulțimea nodurilor și A este mulțimea arcelor). Ponderile sunt **numere raționale pozitive nenule**. Care dintre următoarele estimații $\hat{h}$ sunt admisibile **pentru orice astfel de graf**?

a. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care intră în nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop
b. $\hat{h}$(n) = 1 dacă *n* nu este nod scop, $\hat{h}$(n) = 0 dacă n este nod scop
c. Pentru orice nod *n*, $\hat{h}$(n) = min({ct| ct cost al arcului a, a ∈ A}) (altfel spus, ct este costul minim pe un arc din graf)
d. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care iese din nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop
e. $\hat{h}$(n) = 0 oricare ar fi nodul *n* din graf

(V4.2022) 9. Să presupunem că avem o problemă de căutare P pentru care vrem să aplicăm algoritmul A*. Graful asociat problemei este un graf orientat oarecare G = (N, A) cu arce ponderate (unde N este mulțimea nodurilor și A este mulțimea arcelor). Ponderile sunt **numere raționale pozitive nenule**. Care dintre următoarele estimații $\hat{h}$ sunt admisibile **pentru orice astfel de graf**?

a. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care intră în nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop
b. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care iese din nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop
c. $\hat{h}$(n) = 0 oricare ar fi nodul *n* din graf
d. $\hat{h}$(n) = 1 dacă *n* nu este nod scop, $\hat{h}$(n) = 0 dacă n este nod scop
e. Pentru orice nod *n*, $\hat{h}$(n) = min({ct| ct cost al arcului a, a ∈ A}) (altfel spus, ct este costul minim pe un arc din graf)

(V5.2022) 1. Să presupunem că avem o problemă de căutare P pentru care vrem să aplicăm algoritmul A*. Graful asociat problemei este un graf orientat oarecare G = (N, A) cu arce ponderate (unde N este mulțimea nodurilor și A este mulțimea arcelor). Ponderile sunt **numere raționale pozitive nenule**. Care dintre următoarele estimații $\hat{h}$ sunt admisibile **pentru orice astfel de graf**?

a. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care intră în nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop
b. $\hat{h}$(n) = min({ct| ct cost al arcului a, unde a este un arc care iese din nodul n}) dacă *n* nu este nod scop și $\hat{h}$(n) = 0 dacă *n* este nod scop
c. $\hat{h}$(n) = 1 dacă *n* nu este nod scop, $\hat{h}$(n) = 0 dacă n este nod scop
d. Pentru orice nod *n*, $\hat{h}$(n) = min({ct| ct cost al arcului a, a ∈ A}) (altfel spus, ct este costul minim pe un arc din graf)
e. $\hat{h}$(n) = 0 oricare ar fi nodul *n* din graf

(V1.2022) 6. Pentru o problemă oarecare (deci afirmația trebuie să fie adevărată pentru orice problemă de căutare cu costuri strict pozitive pe tranziții), care dintre formulele următoare ar obține din estimația admisibilă $\hat{h}$(nod), oricare ar fi aceasta (adică formula trebuie să fie adevărată **pentru orice $\hat{h}$** nu pentru cazuri particulare de $\hat{h}$), o nouă estimație $\hat{h}_1$(nod) **în mod cert neadmisibilă? Observație: în caz că nu se specifică nimic despre nodul dat ca parametru, înseamnă că se aplică formula pentru orice nod din graf, de orice tip.**

a. $\hat{h}_1$(nod) = $\hat{h}$(nod) / 3
b. $\hat{h}_1$(nod) = $\hat{h}$(nod) * 2
c. $\hat{h}_1$(nod) = $\hat{h}$(nod) + 4
d. $\hat{h}_1$(nod) = $\hat{h}(nod)^3$ (cu sensul de ridicare la puterea a 3-a)
e. niciuna dintre formulele de la celelalte subpuncte

(V2.2022) 1. Pentru o problemă oarecare (deci afirmația trebuie să fie adevărată pentru orice problemă de căutare cu costuri strict pozitive pe tranziții), care dintre formulele următoare ar obține din estimația admisibilă $\hat{h}$(nod), oricare ar fi aceasta (adică formula trebuie să fie adevărată **pentru orice $\hat{h}$** nu pentru cazuri particulare de $\hat{h}$), o nouă estimație $\hat{h}_1$(nod) **în mod cert neadmisibilă? Observație: în caz că nu se specifică nimic despre nodul dat ca parametru, înseamnă că se aplică formula pentru orice nod din graf, de orice tip.**

a. $\hat{h}_1$(nod)=$\hat{h}$(nod)/2
b. $\hat{h}_1$(nod)=$\hat{h}$(nod)*2
c. $\hat{h}_1$(nod)=$\hat{h}$(nod)+2, dacă *nod* nu e nod scop și $\hat{h}_1$(nod)=0 dacă *nod* este scop
d. $\hat{h}_1$(nod)=$\hat{h}$(nod)^3 (cu sensul de ridicare la puterea a 3-a)
e. niciuna dintre formulele de la celelalte subpuncte

(V3.2022) 4. Pentru o problemă oarecare (deci afirmația trebuie să fie adevărată pentru orice problemă de căutare cu costuri strict pozitive pe tranziții), care dintre formulele următoare ar obține din estimația admisibilă $\hat{h}$(nod), oricare ar fi aceasta (adică formula trebuie să fie adevărată **pentru orice $\hat{h}$** nu pentru cazuri particulare de $\hat{h}$), o nouă estimație $\hat{h}_1$(nod) **în mod cert neadmisibilă? Observație: în caz că nu se specifică nimic despre nodul dat ca parametru, înseamnă că se aplică formula pentru orice nod din graf, de orice tip.**

a. $\hat{h}_1$(nod)=$\hat{h}$(nod)/3
b. $\hat{h}_1$(nod)=$\hat{h}$(nod)*5
c. **niciuna dintre formulele de la celelalte subpuncte**
d. $\hat{h}_1$(nod)=$\hat{h}$(nod)+7, dacă *nod* nu e nod scop și $\hat{h}_1$(nod)=0 dacă *nod* este scop
e. $\hat{h}_1$(nod)=$\hat{h}$(nod)^2 (cu sensul de ridicare la puterea a 2-a)

(V4.2022) 2. Pentru o problemă oarecare (deci afirmația trebuie să fie adevărată pentru orice problemă de căutare cu costuri strict pozitive pe tranziții), care dintre formulele următoare ar obține din estimația admisibilă $\hat{h}$(nod), oricare ar fi aceasta (adică formula trebuie să fie adevărată **pentru orice $\hat{h}$** nu pentru cazuri particulare de $\hat{h}$), o nouă estimație $\hat{h}_1$(nod) **în mod cert neadmisibilă? Observație: în caz că nu se specifică nimic despre nodul dat ca parametru, înseamnă că se aplică formula pentru orice nod din graf, de orice tip.**

a. $\hat{h}_1$(nod)=$\hat{h}$(nod)/3
b. $\hat{h}_1$(nod)=$\hat{h}$(nod)*3
c. $\hat{h}_1$(nod)=$\hat{h}$(nod)+0.5
d. $\hat{h}_1$(nod)=2*$\hat{h}$(nod)^3 (cu sensul de ridicare la puterea a 3-a)
e. niciuna dintre formulele de la celelalte subpuncte

(V5.2022) 3. Pentru o problemă oarecare (deci afirmația trebuie să fie adevărată pentru orice problemă de căutare cu costuri strict pozitive pe tranziții), care dintre formulele următoare ar obține din estimația admisibilă $\hat{h}$(nod), oricare ar fi aceasta (adică formula trebuie să fie adevărată **pentru orice $\hat{h}$** nu pentru cazuri particulare de $\hat{h}$), o nouă estimație $\hat{h}_1$(nod) **în mod cert neadmisibilă? Observație: în caz că nu se specifică nimic despre nodul dat ca parametru, înseamnă că se aplică formula pentru orice nod din graf, de orice tip.**

a. $\hat{h}_1$(nod)=$\hat{h}$(nod)/3+5
b. $\hat{h}_1$(nod)=$\hat{h}$(nod)+0.5
c. $\hat{h}_1$(nod)=$\hat{h}$(nod)*1000
d. $\hat{h}_1$(nod)=2*$\hat{h}$(nod)^3 (cu sensul de ridicare la puterea a 3-a)
e. niciuna dintre formulele de la celelalte subpuncte

(V1.2022) 7. Care dintre următoarele afirmații sunt adevărate pentru rețele Bayesiene?

a. **Topologia unei rețele Bayesiene este un graf orientat (direcționat).**

b. **Într-o rețea Bayesiană nu pot exista două noduri n_1 și n_2 cu proprietatea că există un drum de la n_1 la n_2 și în același timp există un drum de la n_2 la n_1.**

c. Întotdeauna gradul interior (numărul de arce care intră în nod) al unui nod este mai mic sau egal cu 1.

d. Întotdeauna gradul exterior al oricărui nod (numărul de arce care ies din nod) este mai mare sau egal cu gradul său interior (numărul de arce care intră în nod).

e. Orice graf orientat cu arce având asociate ponderi poate reprezenta topologia unei rețele Bayesiene.

(V2.2022) 5. Care dintre următoarele afirmații sunt adevărate pentru rețele Bayesiene?

a. **Topologia unei rețele Bayesiene este un graf orientat (direcționat).**

b. Întotdeauna gradul exterior al oricărui nod (numărul de arce care ies din nod) este mai mare sau egal cu gradul său interior (numărul de arce care intră în nod).

c. Întotdeauna gradul interior (numărul de arce care intră în nod) al unui nod este mai mic sau egal cu 1.

d. Orice graf orientat cu arce având asociate ponderi poate reprezenta topologia unei rețele Bayesiene.

e. **Într-o rețea Bayesiană nu pot exista două noduri n_1 și n_2 cu proprietatea că există un drum de la n_1 la n_2 și în același timp există un drum de la n_2 la n_1.**

(V3.2022) 6. Care dintre următoarele afirmații sunt adevărate pentru rețele Bayesiene?

a. Topologia unei rețele Bayesiene poate fi un graf neorientat.

b. Întotdeauna gradul exterior al oricărui nod (numărul de arce care ies din nod) este mai mare sau egal cu gradul său interior (numărul de arce care intră în nod).

c. Întotdeauna gradul interior (numărul de arce care intră în nod) al unui nod este mai mic sau egal cu 1.

d. **Într-o topologie de rețea Bayesiană pot exista atât noduri cu grad interior 0 (numărul de arce care intră în nod) cât și noduri cu grad exterior 0 (numărul de arce care ies din nod).**

e. Orice graf orientat cu arce având asociate ponderi poate reprezenta topologia unei rețele Bayesiene.

f. **Într-o rețea Bayesiană nu pot exista două noduri n_1 și n_2 cu proprietatea că există un drum de la n_1 la n_2 și în același timp există un drum de la n_2 la n_1.**

(V4.2022) 3. Care dintre următoarele afirmații sunt adevărate pentru rețele Bayesiene?

a. Topologia unei rețele Bayesiene poate fi un graf neorientat.

b. Orice graf orientat cu arce având asociate ponderi poate reprezenta topologia unei rețele Bayesiene.

c. Întotdeauna gradul interior (numărul de arce care intră în nod) al unui nod este mai mic sau egal cu 1.

d. **Într-o topologie de rețea Bayesiană pot exista atât noduri cu grad interior 0 (numărul de arce care intră în nod) cât și noduri cu grad exterior 0 (numărul de arce care ies din nod).**

e. **Într-o rețea Bayesiană nu pot exista două noduri n_1 și n_2 cu proprietatea că există un drum de la n_1 la n_2 și în același timp există un drum de la n_2 la n_1.**

f. Întotdeauna gradul exterior al oricărui nod (numărul de arce care ies din nod) este mai mare sau egal cu gradul său interior (numărul de arce care intră în nod).

(V5.2022) 4. Care dintre următoarele afirmații sunt adevărate pentru rețele Bayesiene?

a. Întotdeauna gradul exterior al oricărui nod (numărul de arce care ies din nod) este mai mare sau egal cu gradul său interior (numărul de arce care intră în nod).

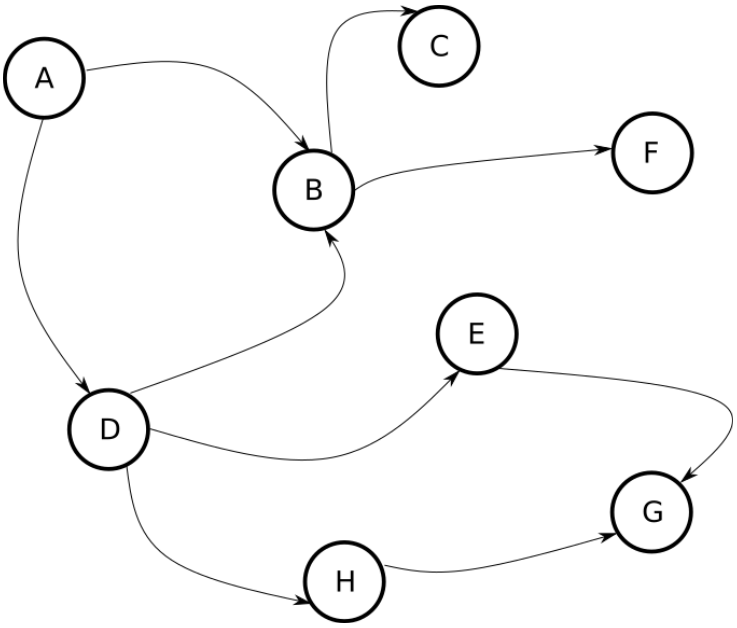
b. Întotdeauna gradul interior (numărul de arce care intră în nod) al unui nod este mai mic sau egal cu 1.

c. Orice graf orientat cu arce având asociate ponderi poate reprezenta topologia unei rețele Bayesiene.

d. **Topologia unei rețele Bayesiene este un graf orientat (direcționat).**

e. **Într-o rețea Bayesiană nu pot exista două noduri n_1 și n_2 cu proprietatea că există un drum de la n_1 la n_2 și în același timp există un drum de la n_2 la n_1.**

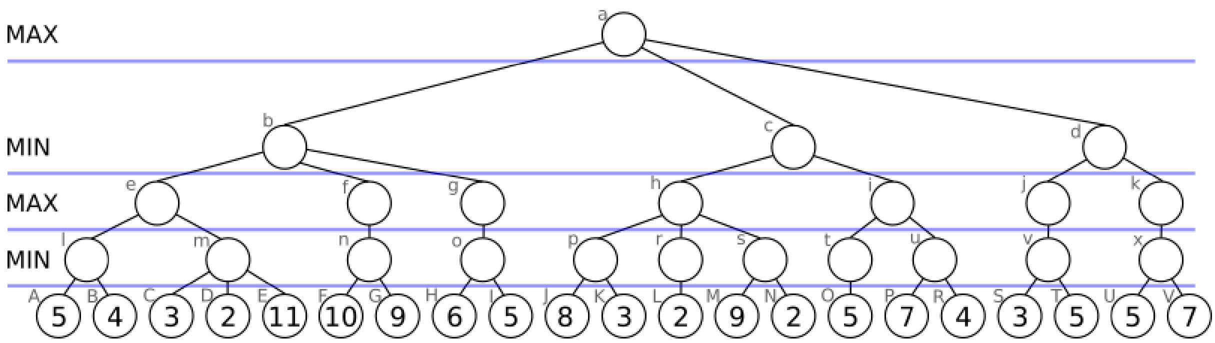
(Model - 2023) 3. Se dă următoarea topologie de rețea Bayesiană:



Care dintre următoarele afirmații sunt adevărate?

- **a. Mulțimea {E, H} d-separă mulțimea {A, B} de mulțimea {G}.**
- b. Graful dat nu este o topologie corectă pentru o rețea Bayesiană.
- **c. Dacă am adăuga arcul F->A graful nu ar mai fi o topologie corectă pentru o rețea Bayesiana.**
- d. Drumul de la nodul A la nodul F e blocat conditionat de mulțimea {C,F}
- e. Mulțimea {E,G,H} d-separă mulțimea {A} de mulțimea {D}

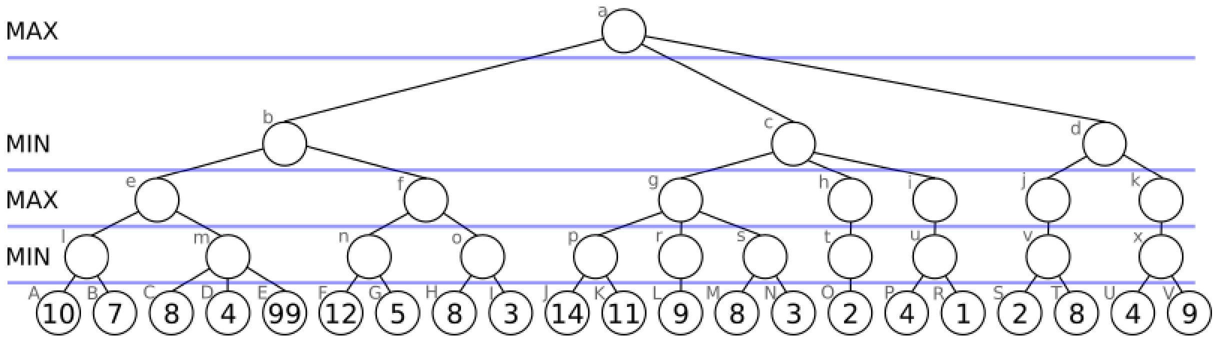
(Model - 2023) 9. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:



Care dintre următoarele afirmații sunt adevărate?

- **a.** **Numărul de retezări efectuate de Alpha-Beta este 5 (edit: am corectat) (am considerat o retezare ca fiind necalcularea unui întreg subarbore, nu se contorizează și nodurile individuale din subarborele tăiat).**
- b. **Dacă am aplica alpha-beta pe acest joc atunci nodurile D și E nu ar mai fi evaluate**
- c. Valoare minimax a nodului f este 10.
- d. Pentru a calcula valoarea nodului b este suficient să calculăm minimul dintre valorile minimax înscrise în nodurile de la A la I *(i mare)*.
- **e.** **Nu există nicio valoare posibilă cu care am putea înlocui valoarea minimax a lui C astfel încât să forțăm variația principală să treacă prin C.**

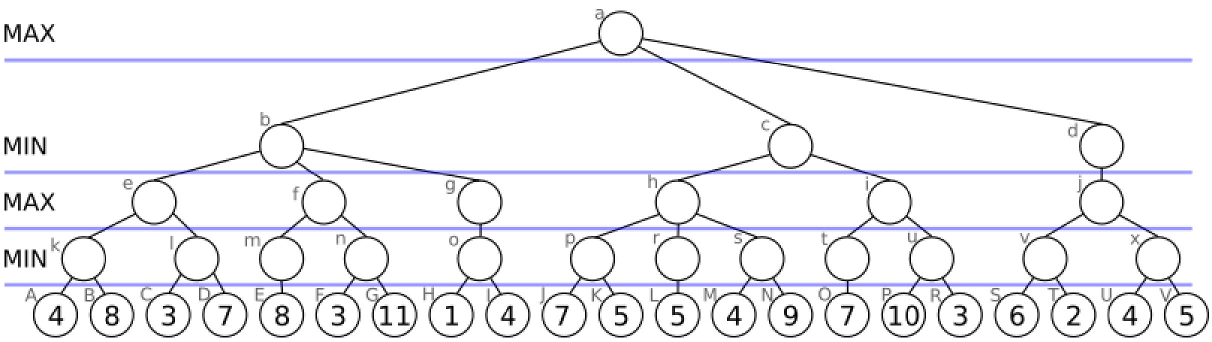
(V1.2022) 2. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:



Care dintre următoarele afirmații sunt adevărate?

- a. **Dacă am aplica algoritmul alpha-beta pe acest arbore și valoarea nodului *G* ar fi 55 în loc de 5, atunci nodul *o* nu ar mai fi evaluat.**
- b. În nodul ***a*** vom avea valoarea minimax 99.
- c. Dacă am aplica algoritmul alpha-beta pe acest arbore, nodurile D și E nu ar mai fi evaluate.
- **d.** **Aplicând algoritmul minimax, nodul *h* va avea în mod sigur valoarea 2.**
- **e.** **Dacă valorile frunzelor ar fi toate de 10 ori mai mari (pentru orice nod frunză, în loc de valoarea *v* am avea valoarea 10**v*) atunci setul de noduri din variația principală nu s-ar schimba.**

(V2.2022) 9. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:



Care dintre următoarele afirmații sunt adevărate?

a. Dacă ordinea fiilor lui ***b*** ar fi ***g,e,f*** în loc de ***e,f,g*** atunci, dacă am aplica algoritmul alpha-beta pe acest arbore, algoritmul ar decidecă nodurile e și f nu ar mai trebui evaluate

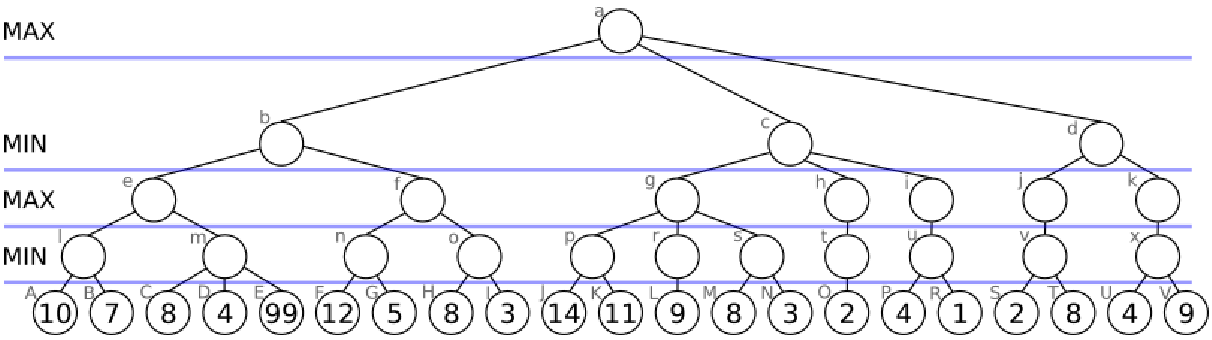
b. În nodul ***a*** vom avea valoarea minimax 11.

c. Dacă am aplica algoritmul alpha-beta pe acest arbore, nodul N nu ar mai fi evaluat

d. Aplicând algoritmul minimax, nodul *g* va avea în mod sigur valoarea 4.

e. Dacă valorile frunzelor ar fi toate cu 5 mai mari (pentru orice nod frunză, în loc de valoarea *v* am avea valoarea 5+*v*) atunci setul de noduri din variația principală nu s-ar schimba.

(V3.2022) 5. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:



Care dintre următoarele afirmații sunt adevărate?

a. În nodul ***a*** vom avea valoarea minimax 99.

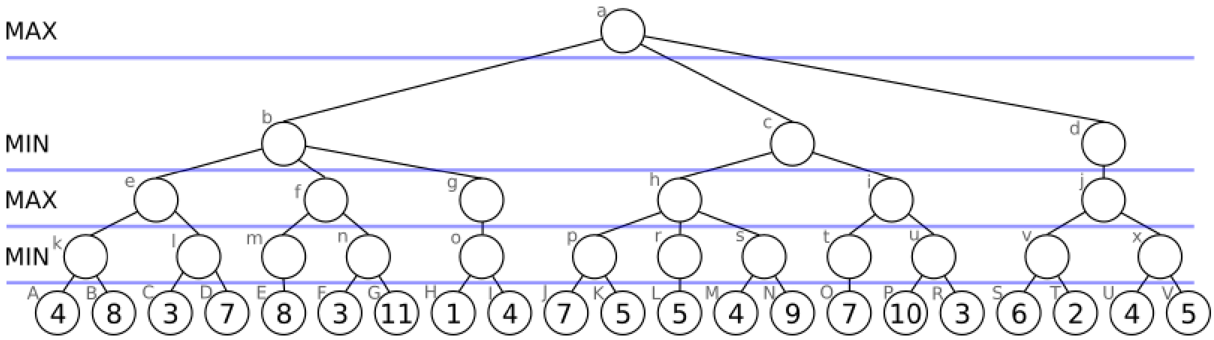
b. Dacă am aplica algoritmul alpha-beta pe acest arbore, nodurile D și E nu ar mai fi evaluate.

c. Aplicând algoritmul minimax, nodul *h* va avea în mod sigur valoarea 2.

d. Dacă valorile frunzelor ar fi toate de 10 ori mai mari (pentru orice nod frunză, în loc de valoarea *v* am avea valoarea 10*v) atunci setul de noduri din variația principală nu s-ar schimba.

e. Dacă am aplica algoritmul alpha-beta pe acest arbore și valoarea nodului *G* ar fi 55 în loc de 5, atunci nodul *o* nu ar mai fi evaluat.

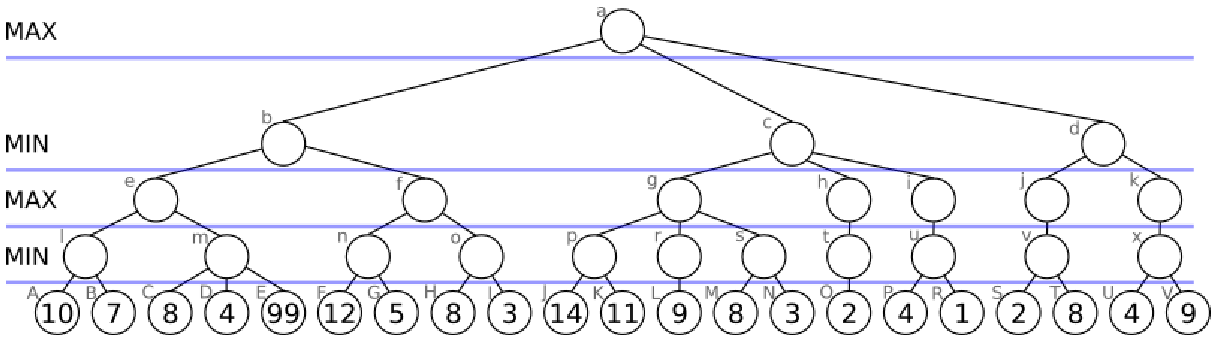
(V4.2022) 8. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:



Care dintre următoarele afirmații sunt adevărate?

a. **În nodul *b* vom avea valoarea minimax 1.**
b. În nodul ***a*** vom avea valoarea minimax 11.
c. Dacă am aplica algoritmul alpha-beta pe acest arbore, nodul C nu ar mai fi evaluat
d. Aplicând algoritmul minimax, nodul *g* va avea în mod sigur valoarea 4.
e. Oricare două noduri-frați am inversa, setul de noduri din variația principală nu se schimbă.

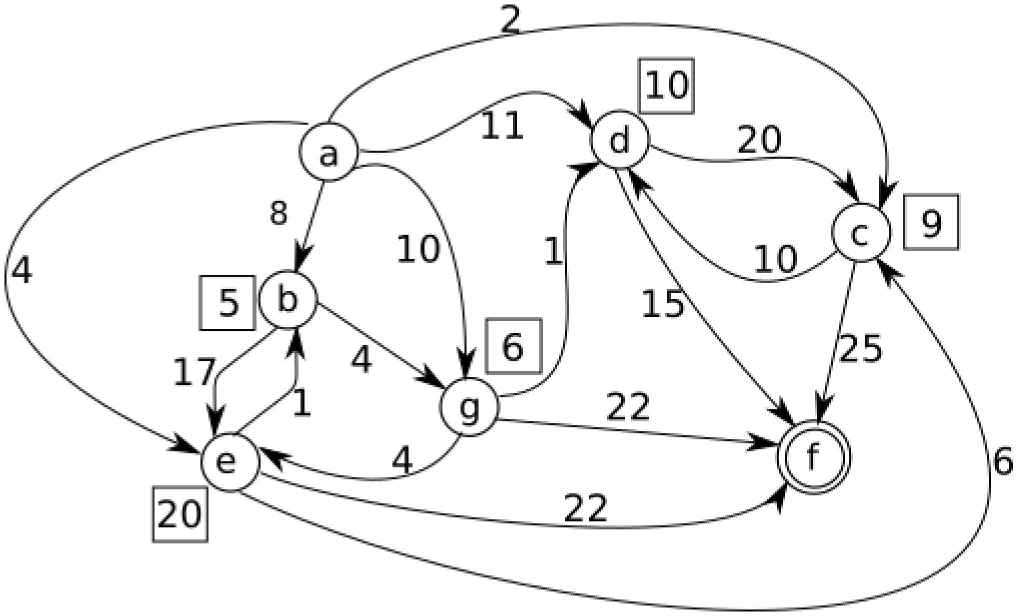
(V5.2022) 5. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:



Care dintre următoarele afirmații sunt adevărate?

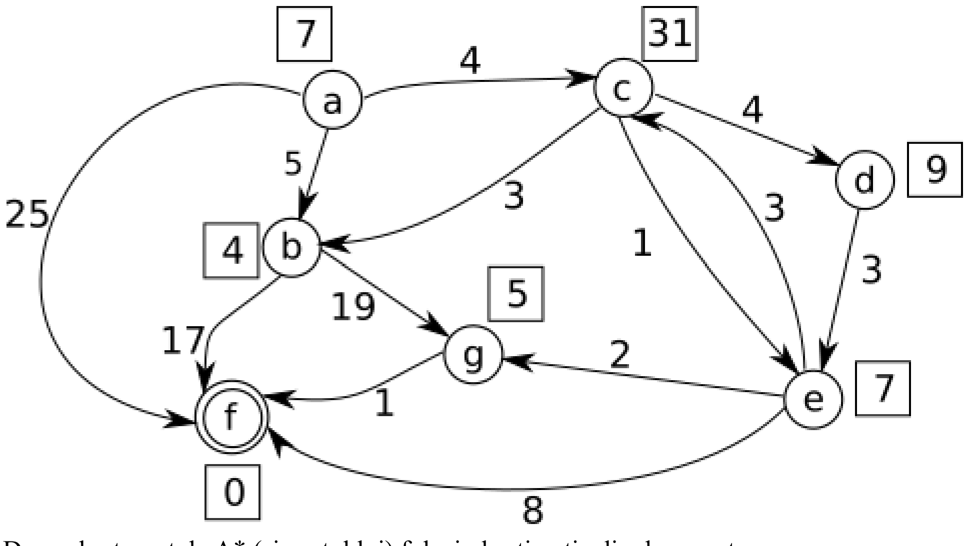
a. În nodul ***a*** vom avea valoarea minimax 99.
b. Aplicând algoritmul minimax, nodul *h* va avea în mod sigur valoarea 2.
c. Dacă valorile frunzelor ar fi toate de 10 ori mai mari (pentru orice nod frunză, în loc de valoarea *v* am avea valoarea 10*v) atunci setul de noduri din variația principală nu s-ar schimba.
d. Dacă am aplica algoritmul alpha-beta pe acest arbore, nodurile D și E nu ar mai fi evaluate.
e. Dacă am aplica algoritmul alpha-beta pe acest arbore și valoarea nodului *G* ar fi 55 în loc de 5, atunci nodul *o* nu ar mai fi evaluat.

(Model - 2023) **7.** Pentru graful de mai jos (cu nodul de start ***a*** și nodul final ***f***), care este al treilea nod care e extins de A*? (prima extindere, numerotată cu 1, se consideră a fi cea a rădăcinii).



a. a
b. b
c. c
d. d
e. e
f. f
g. g

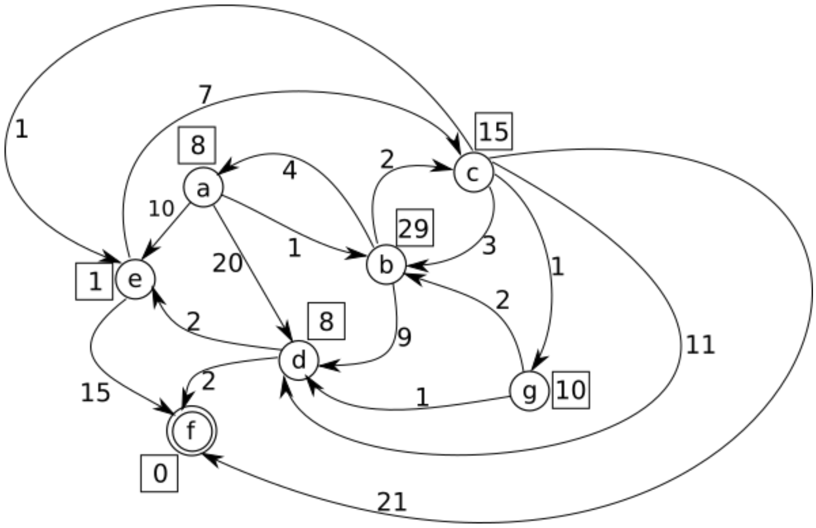
(V1.2022) 3. Se consideră următorul graf cu muchii ponderate (costul unei muchii este scris în dreptul ei). Nodul de start este nodul *a* și nodul scop este *f*. Factorul euristic al fiecărui nod (estimația ĥ) este scris în dreptul nodului într-un pătrățel. Nu este garantat că estimația este admisibilă.



Drumul returnat de A* (și costul lui) folosind estimația din desen este:

a. a -> b -> g -> f cu costul 25
b. a -> b -> f cu costul 22
c. a -> f cu costul 25
d. a -> c -> e -> g -> f cu costul 8
e. a -> c -> e -> f cu costul 13

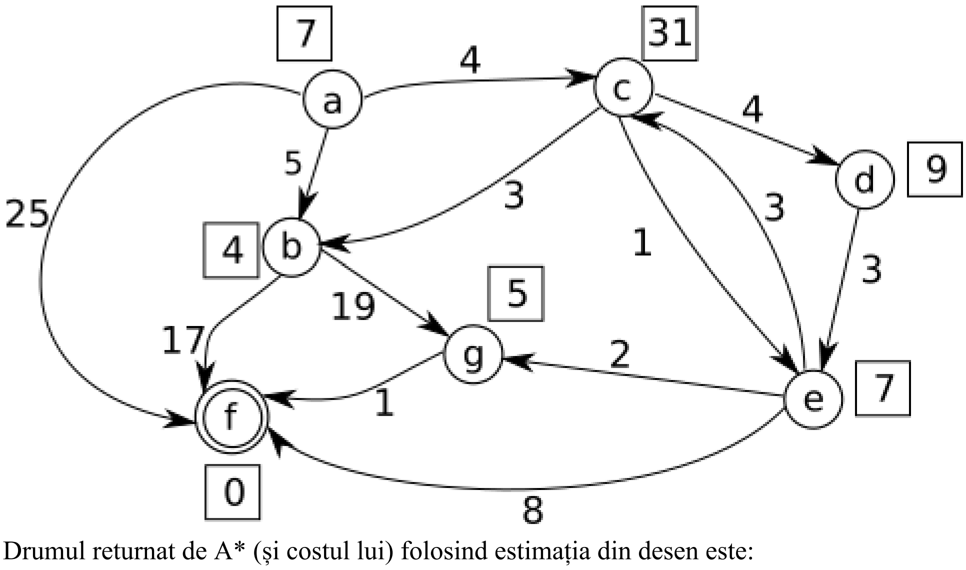
(V2.2022) 8. Se consideră următorul graf cu muchii ponderate (costul unei muchii este scris în dreptul ei). Nodul de start este nodul *a* și nodul scop este *f*. Factorul euristic al fiecărui nod (estimația ĥ) este scris în dreptul nodului într-un pătrățel. Nu este garantat că estimația este admisibilă.



Drumul returnat de A* (și costul lui) folosind estimația din desen este:

a. a ->b->c ->g->d ->f cu costul 15
b. a ->b->c ->g->d ->f cu costul 62
c. a->e->f cu costul 25
d. a ->b->c ->g->d ->f cu costul 7
e. a->b->g->d->f cu costul 6

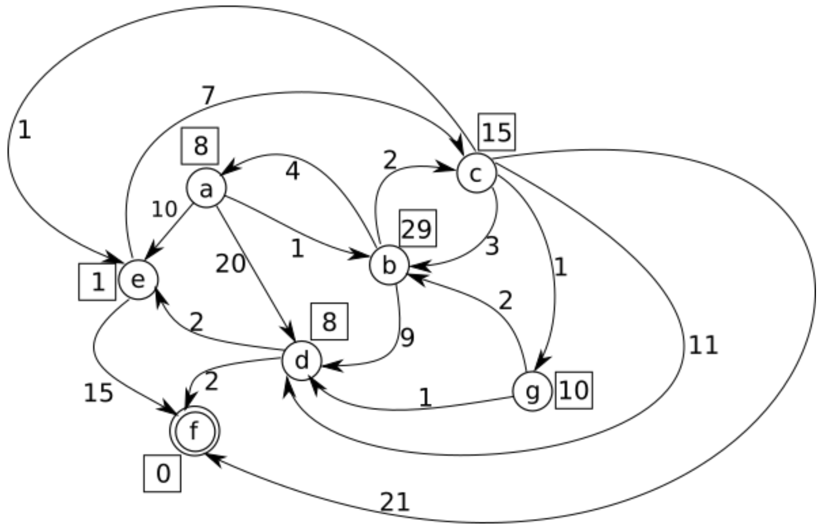
(V3.2022) 1. Se consideră următorul graf cu muchii ponderate (costul unei muchii este scris în dreptul ei). Nodul de start este nodul *a* și nodul scop este *f*. Factorul euristic al fiecărui nod (estimația ĥ) este scris în dreptul nodului într-un pătrățel. Nu este garantat că estimația este admisibilă.



Drumul returnat de A* (și costul lui) folosind estimația din desen este:

a. a -> c -> e -> f cu costul 13
b. a -> c -> e -> f cu costul 20
c. a -> b -> g -> f cu costul 25
d. a -> b -> f cu costul 22
e. a -> f cu costul 25
f. a -> f cu costul 32
g. a -> c -> e -> g -> f cu costul 8

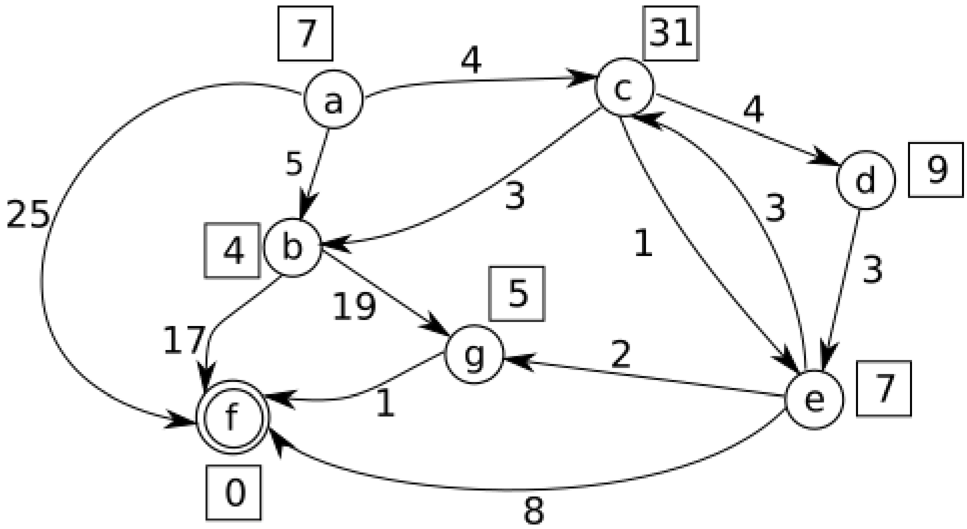
(V4.2022) 7. Se consideră următorul graf cu muchii ponderate (costul unei muchii este scris în dreptul ei). Nodul de start este nodul *a* și nodul scop este *f*. Factorul euristic al fiecărui nod (estimația ĥ) este scris în dreptul nodului într-un pătrățel. Nu este garantat că estimația este admisibilă.



Drumul returnat de A* (și costul lui) folosind estimația din desen este:

a. a ->b->c ->g->d ->f cu costul 15
b. a ->b->c ->g->d ->f cu costul 7
c. a->e->f cu costul 25
d. a->b->g->d->f cu costul 6
e. a ->b->c ->g->d -e->f cu costul 28
f. a->d->f cu costul 22

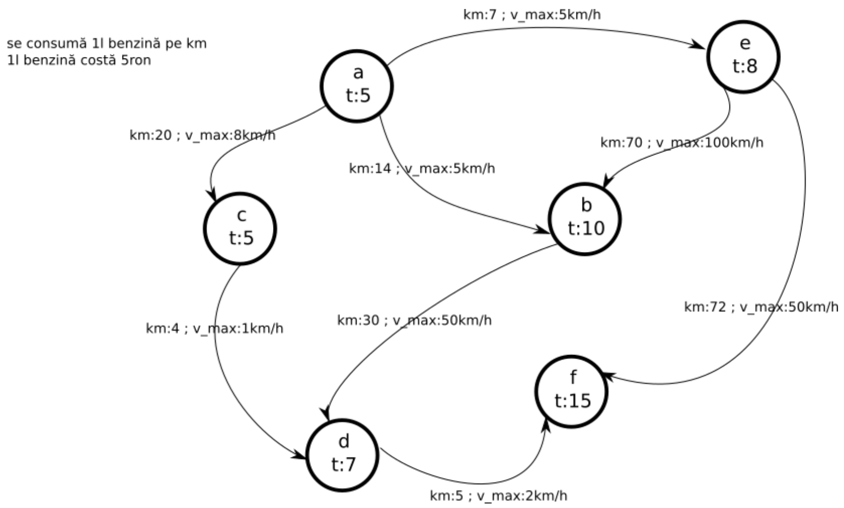
(V5.2022) 8. Se consideră următorul graf cu muchii ponderate (costul unei muchii este scris în dreptul ei). Nodul de start este nodul *a* și nodul scop este *f*. Factorul euristic al fiecărui nod (estimația ĥ) este scris în dreptul nodului într-un pătrățel. Nu este garantat că estimația este admisibilă.



Drumul returnat de A* (și costul lui) folosind estimația din desen este:

a. a -> c -> e -> f cu costul 13
b. a -> c -> e -> f cu costul 20
c. a -> b -> g -> f cu costul 25
d. a -> f cu costul 25
e. a -> f cu costul 32
f. a -> c -> e -> g -> f cu costul 8
g. a -> b -> f cu costul 22

(V1.2022) 8. Mai jos este reprezentarea sub formă de graf a unei rețele de șosele între orașe.



Orașele sunt reprezentate ca noduri, iar muchiile reprezintă șoselele. Pe fiecare muchie este trecută lungimea sa în kilometri, dar și viteza maximă admisă. La intrarea într-un oraș se plătește o taxă (specificată în fiecare nod prin t:taxă în RON). Dorim să folosim în mod corect A*, cu un cost relevant și o estimație admisibilă, pentru a găsi cel mai bun drum conform unui criteriu specificat mai jos. Nodul ***a*** este nodul de start, și ***f*** este nodul scop. Dorim să găsim cea mai potrivită funcție de cost care asociază unei tranziții n_i->n_j (de la nodul n_i la nodul n_j) un anumit cost. Care dintre următoare afirmații sunt corecte?

a. Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.

b. Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*cost_benzină +taxa(n_j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, cost_benzină este costul unui litru de benzină, iar taxa(n_j) este taxa de intrare în orașul n_j.

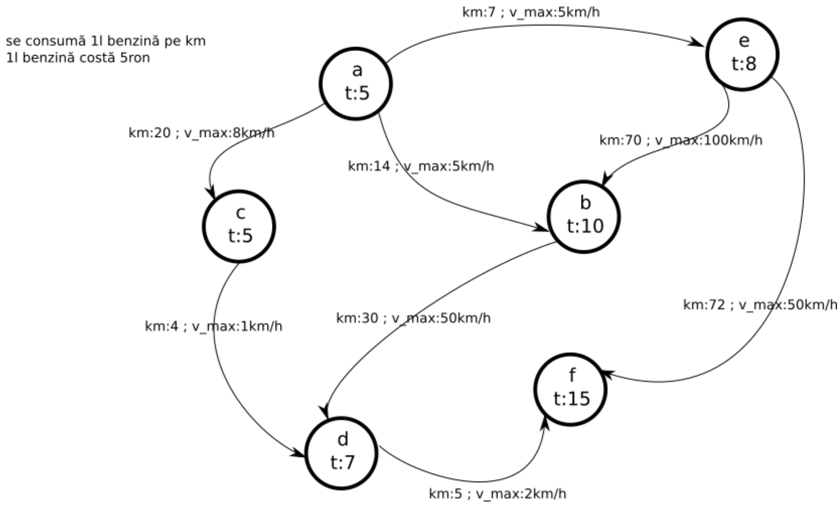
c. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.

d. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*viteza(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.

e. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)/viteza(i,j), unde nr_km(i,j) e numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.

f. Dacă dorim să găsim drumul cu număr minim de orașe prin care se trece, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul nr_km(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j.

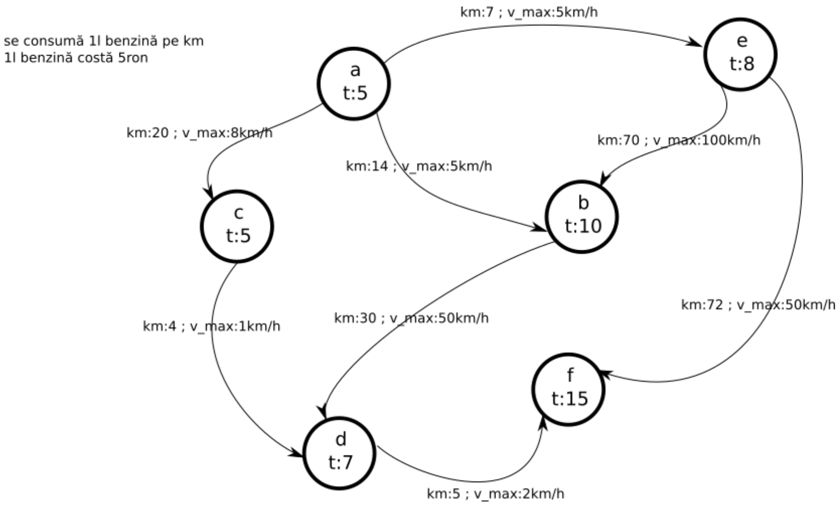
(V2.2022) 6. Mai jos este reprezentarea sub formă de graf a unei rețele de șosele între orașe.



Orașele sunt reprezentate ca noduri, iar muchiile reprezintă șoselele. Pe fiecare muchie este trecută lungimea sa în kilometri, dar și viteza maximă admisă. La intrarea într-un oraș se plătește o taxă (specificată în fiecare nod prin t:taxă în RON). Dorim să folosim în mod corect A*, cu un cost relevant și o estimație admisibilă, pentru a găsi cel mai bun drum conform unui criteriu specificat mai jos. Nodul ***a*** este nodul de start, și ***f*** este nodul scop. Dorim să găsim cea mai potrivită funcție de cost care asociază unei tranziții n_i->n_j (de la nodul n_i la nodul n_j) un anumit cost. Care dintre următoare afirmații sunt corecte?

a. **Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*cost_benzină +taxa(n_j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, cost_benzină este costul unui litru de benzină, iar taxa(n_j) este taxa de intrare în orașul n_j.**
b. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.
c. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*viteza(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.
d. Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.
e. **Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)/viteza(i,j), unde nr_km(i,j) e numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.**
f. Dacă dorim să găsim drumul cu număr minim de orașe prin care se trece, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul nr_km(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j.

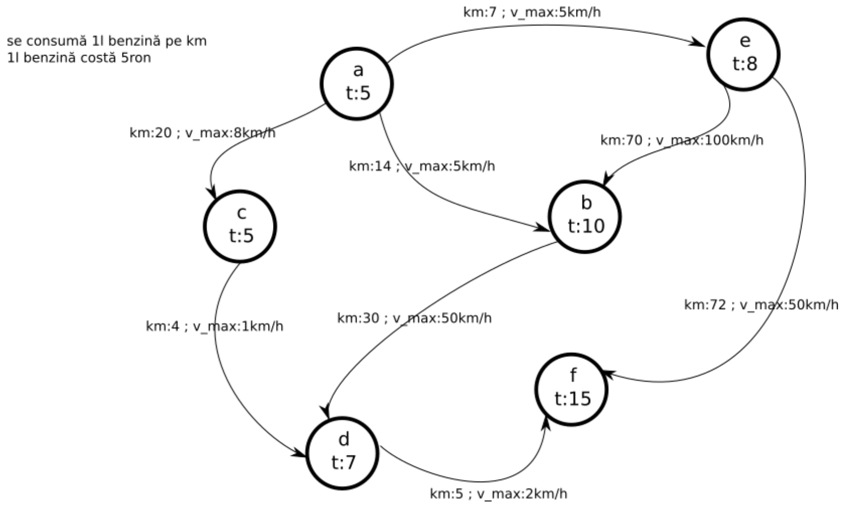
(V3.2022) 7. Mai jos este reprezentarea sub formă de graf a unei rețele de șosele între orașe.



Orașele sunt reprezentate ca noduri, iar muchiile reprezintă șoselele. Pe fiecare muchie este trecută lungimea sa în kilometri, dar și viteza maximă admisă. La intrarea într-un oraș se plătește o taxă (specificată în fiecare nod prin t:taxă în RON). Dorim să folosim în mod corect A*, cu un cost relevant și o estimație admisibilă, pentru a găsi cel mai bun drum conform unui criteriu specificat mai jos. Nodul ***a*** este nodul de start, și ***f*** este nodul scop. Dorim să găsim cea mai potrivită funcție de cost care asociază unei tranziții n_i->n_j (de la nodul n_i la nodul n_j) un anumit cost. Care dintre următoare afirmații sunt corecte?

a. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.
b. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*viteza(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.
c. Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.
d. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)/viteza(i,j), unde nr_km(i,j) e numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.
e. Dacă dorim să găsim drumul cu număr minim de orașe prin care se trece, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul nr_km(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j.
f. Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*cost_benzină +taxa(n_j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, cost_benzină este costul unui litru de benzină, iar taxa(n_j) este taxa de intrare în orașul n_j.

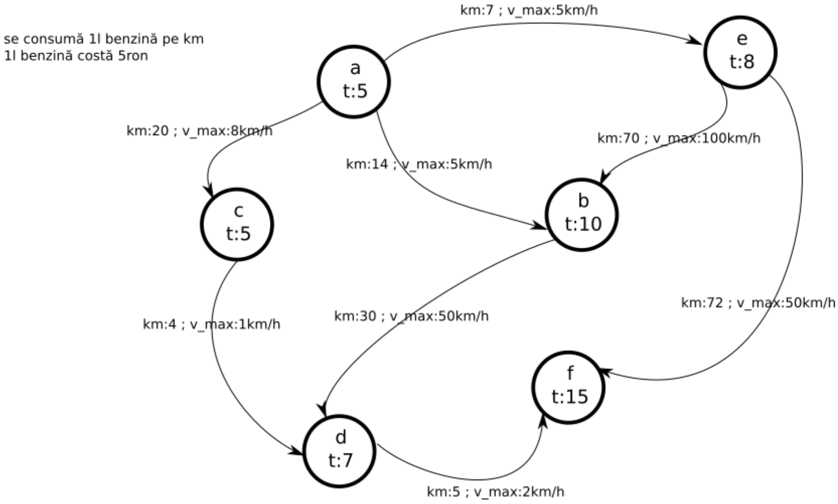
(V4.2022) 5. Mai jos este reprezentarea sub formă de graf a unei rețele de șosele între orașe.



Orașele sunt reprezentate ca noduri, iar muchiile reprezintă șoselele. Pe fiecare muchie este trecută lungimea sa în kilometri, dar și viteza maximă admisă. La intrarea într-un oraș se plătește o taxă (specificată în fiecare nod prin t:taxă în RON). Dorim să folosim în mod corect A*, cu un cost relevant și o estimație admisibilă, pentru a găsi cel mai bun drum conform unui criteriu specificat mai jos. Nodul ***a*** este nodul de start, și ***f*** este nodul scop. Dorim să găsim cea mai potrivită funcție de cost care asociază unei tranziții n_i->n_j (de la nodul n_i la nodul n_j) un anumit cost. Care dintre următoare afirmații sunt corecte?

a. Dacă dorim să găsim drumul cu număr minim de orașe prin care se trece, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul nr_km(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j.
b. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.
c. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*viteza(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.
d. Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.
e. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)/viteza(i,j), unde nr_km(i,j) e numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.
f. **Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*cost_benzină +taxa(n_j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, cost_benzină este costul unui litru de benzină, iar taxa(n_j) este taxa de intrare în orașul n_j.**

(V5.2022) 7. Mai jos este reprezentarea sub formă de graf a unei rețele de șosele între orașe.



Orașele sunt reprezentate ca noduri, iar muchiile reprezintă șoselele. Pe fiecare muchie este trecută lungimea sa în kilometri, dar și viteza maximă admisă. La intrarea într-un oraș se plătește o taxă (specificată în fiecare nod prin t:taxă în RON). Dorim să folosim în mod corect A*, cu un cost relevant și o estimație admisibilă, pentru a găsi cel mai bun drum conform unui criteriu specificat mai jos. Nodul ***a*** este nodul de start, și ***f*** este nodul scop. Dorim să găsim cea mai potrivită funcție de cost care asociază unei tranziții n_i->n_j (de la nodul n_i la nodul n_j) un anumit cost. Care dintre următoare afirmații sunt corecte?

a. Dacă dorim să găsim drumul cu număr minim de orașe prin care se trece, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul nr_km(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j.

b. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.

c. Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j numărul de kilometri dintre n_i și n_j.

d. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)/viteza(i,j), unde nr_km(i,j) e numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.

e. **Dacă dorim să găsim drumul de cost minim în bani, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*cost_benzină +taxa(n_j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, cost_benzină este costul unui litru de benzină, iar taxa(n_j) este taxa de intrare în orașul n_j.**

f. Dacă dorim să găsim drumul de timp minim, atunci o funcție potrivită de cost asociază tranziției n_i->n_j valoarea nr_km(i,j)*viteza(i,j), unde nr_km(i,j) este numărul de kilometri dintre n_i și n_j, iar *viteza(i,j)* este viteza maximă admisă pe acel drum.

(V1.2022) 9. Considerăm problema 8-puzzle cu următoarea modificare: cele 8 plăcuțe sunt numerotate cu numere de la 1 la 4 și există exact câte două plăcuțe numerotate cu același număr. Un exemplu de stare inițială este:

1	2	1
4	3	4
2	3	

Costul unei mutări de plăcuță este 1.
Scopul este să ajungem la o configurație în care cele două plăcuțe din fiecare pereche (având același număr) sunt vecine pe linie sau coloană.
Care dintre următoarele estimații sunt **admisibile**?

- **a. Numărul de perechi de plăcuțe cu același număr care nu sunt alăturate.**
- b. Numărul de plăcuțe care nu au niciun vecin cu același număr pe linie sau coloană.
- c. S-1, unde S este suma distanțelor d(i), unde d(i) este distanța Euclidiană dintre două plăcute cu numărul i, considerând drept coordonate pentru distanță linia și coloana fiecărei plăcuțe.
- d. S-1, unde S este suma distanțelor Manhattan dintre fiecare două plăcuțe inscripționate cu același număr.
- **e. M-1, unde M este minimul dintre distanțele d(i), unde d(i) este distanța Manhattan dintre două plăcuțe inscripționate cu numărul i.**
- f. S, unde S este suma distanțelor Manhattan dintre fiecare două plăcuțe inscripționate cu același număr.

(V2.2022) 7. Considerăm problema 8-puzzle cu următoarea modificare: cele 8 plăcuțe sunt numerotate cu numere de la 1 la 4 și există exact câte două plăcuțe numerotate cu același număr. Un exemplu de stare inițială este:

1	2	1
4	3	4
2	3	

Costul unei mutări de plăcuță este 1.
Scopul este să ajungem la o configurație în care cele două plăcuțe din fiecare pereche (având același număr) sunt vecine pe linie sau coloană.
Care dintre următoarele estimații sunt **admisibile**?

- a. M, unde M este minimul dintre distanțele d(i), unde d(i) este distanța Manhattan dintre două plăcuțe inscripționate cu numărul i.
- **b. Numărul de perechi de plăcuțe cu același număr care nu sunt alăturate.**
- c. Numărul de plăcuțe care nu au niciun vecin cu același număr pe linie sau coloană.
- d. S-1, unde S este suma distanțelor d(i), unde d(i) este distanța Euclidiană dintre două plăcute cu numărul i, considerând drept coordonate pentru distanță linia și coloana fiecărei plăcuțe.
- e. S-1, unde S este suma distanțelor Manhattan dintre fiecare două plăcuțe inscripționate cu același număr.
- **f. M-1, unde M este minimul dintre distanțele d(i), unde d(i) este distanța Manhattan dintre două plăcuțe inscripționate cu numărul i.**

(V3.2022) 8. Considerăm problema 8-puzzle cu următoarea modificare: cele 8 plăcuțe sunt numerotate cu numere de la 1 la 4 și există exact câte două plăcuțe numerotate cu același număr. Un exemplu de stare inițială este:

1	2	1
4	3	4
2	3	

Costul unei mutări de plăcuță este 1.
Scopul este să ajungem la o configurație în care cele două plăcuțe din fiecare pereche (având același număr) sunt vecine pe linie sau coloană.
Care dintre următoarele estimații sunt **admisibile**?

a. S, unde S este suma distanțelor d(i), unde d(i) este distanța Euclidiană dintre două plăcute cu numărul i, considerând drept coordonate pentru distanță linia și coloana fiecărei plăcuțe.
b. Numărul de perechi de plăcuțe cu același număr care nu sunt alăturate.
c. Numărul de plăcuțe care nu au niciun vecin cu același număr pe linie sau coloană.
d. S-1, unde S este suma distanțelor d(i), unde d(i) este distanța Euclidiană dintre două plăcute cu numărul i, considerând drept coordonate pentru distanță linia și coloana fiecărei plăcuțe.
e. S-1, unde S este suma distanțelor Manhattan dintre fiecare două plăcuțe inscripționate cu același număr.
f. M-1, unde M este minimul dintre distanțele d(i), unde d(i) este distanța Manhattan dintre două plăcuțe inscripționate cu numărul i.

(V4.2022) 6. Considerăm problema 8-puzzle cu următoarea modificare: cele 8 plăcuțe sunt numerotate cu numere de la 1 la 4 și există exact câte două plăcuțe numerotate cu același număr. Un exemplu de stare inițială este:

1	2	1
4	3	4
2	3	

Costul unei mutări de plăcuță este 1.
Scopul este să ajungem la o configurație în care cele două plăcuțe din fiecare pereche (având același număr) sunt vecine pe linie sau coloană.
Care dintre următoarele estimații sunt **admisibile**?

a. M-1, unde M este minimul dintre distanțele d(i), unde d(i) este distanța Manhattan dintre două plăcuțe inscripționate cu numărul i.
b. S, unde S este suma distanțelor Manhattan dintre fiecare două plăcuțe inscripționate cu același număr.
c. Numărul de perechi de plăcuțe cu același număr care nu sunt alăturate.
d. Numărul de plăcuțe care nu au niciun vecin cu același număr pe linie sau coloană.
e. S-1, unde S este suma distanțelor d(i), unde d(i) este distanța Euclidiană dintre două plăcute cu numărul i, considerând drept coordonate pentru distanță linia și coloana fiecărei plăcuțe.
f. S-1, unde S este suma distanțelor Manhattan dintre fiecare două plăcuțe inscripționate cu același număr.

(V5.2022) 9. Considerăm problema 8-puzzle cu următoarea modificare: cele 8 plăcuțe sunt numerotate cu numere de la 1 la 4 și există exact câte două plăcuțe numerotate cu același număr. Un exemplu de stare inițială este:

1	2	1
4	3	4
2	3	

Costul unei mutări de plăcuță este 1.

Scopul este să ajungem la o configurație în care cele două plăcuțe din fiecare pereche (având același număr) sunt vecine pe linie sau coloană.

Care dintre următoarele estimații sunt **admisibile**?

a. **M-1, unde M este minimul dintre distanțele d(i), unde d(i) este distanța Manhattan dintre două plăcuțe inscripționate cu numărul i.**

b. **N/3, unde N este numărul de plăcuțe care nu au niciun vecin cu același număr pe linie sau coloană.**

c. **Numărul de perechi de plăcuțe cu același număr care nu sunt alăturate.**

d. Numărul de plăcuțe care nu au niciun vecin cu același număr pe linie sau coloană.

e. S-1, unde S este suma distanțelor d(i), unde d(i) este distanța Euclidiană dintre două plăcute cu numărul i, considerând drept coordonate pentru distanță linia și coloana fiecărei plăcuțe.

f. S-1, unde S este suma distanțelor Manhattan dintre fiecare două plăcuțe inscripționate cu același număr.

(Model - 2023) 1. Se consideră problema blocurilor cu următoarele modificări.

- Există două tipuri de blocuri: - Blocuri numerice - Informația fiecărui bloc este unu număr natural nenul.
    - Blocuri semn. Sunt exact 2. Acestea au ca informație semnul +, respectiv -.
- Ca și la problema blocurilor clasică, putem **muta** un bloc dintr-un vârf de stivă pe alt vârf de stivă, cu următoarele resctricții: - Peste blocurile semn nu putem plasa niciun bloc.
    - Blocurile semn pot fi plasate doar pe stive nevide. Blocul semn **+** mutat pe o stivă *s* va incrementa cu 1 numerele tuturor blocurilor de pe stivă. Blocul semn **-** mutat pe o stivă *s* va decrementa cu 1 numerele tuturor blocurilor de pe stivă. O stare succesoare care conține numere negative e considerată invalidă și nu se va genera.
- **Costul** mutării unui *bloc numeric* este egal cu numărul înscris pe acel bloc. Mutarea unui *bloc semn* costă 1.
- Pentru o configurație initială cu S stive, **scopul** este sa avem pe toate stivele, la bază numere egale între ele (să nu existe doua stive s_i și s_j care să aibă la bază numere diferite) și să nu existe stive vide.

În opțiunile de răspuns vom folosi următoarele notații pentru parametrii problemei, care descriu stările: S - numărul de stive, SV - numărul de stive vide în cadrul stării curente, NB - numărul total de blocuri (numerice+semn). Cu restricțiile: S >= 3, NB > 4, SV < S.
Care dintre următoarele afirmații sunt adevărate?

a. **În mulțimea de stări inițiale posibile (cu parametrii conform condițiilor din enunț) există măcar o stare inițială pentru care să nu putem găsi o soluție folosind A*.**

b. În mulțimea de stări inițiale posibile (cu parametrii conform condițiilor din enunț, și, în plus, SV=0, există măcar o stare inițială pentru care să nu putem găsi soluție folosind A*.

c. Numărul de succesori valizi ai unei stări oarecare este 2*(S-SV-2)

d. Numărul de succesori valizi ai unei stări oarecare este (S-SV-2)*(S-3)

e. **Edit: lipsea o restrictie pe care am uitat sa o copiez din fisierul initial si anume blocurile numerice din stare au valoare strict mai mare de 1. Numărul de succesori valizi ai unei stări oarecare este (S-SV-2)*(S-1)**

(Model - 2023) 2. Pentru enunțul problemei de mai sus (asemănătoare cu problema blocurilor). Care dintre următoarele afirmații sunt adevărate?

a. O estimație admisibilă e numărul de stive nevide care au la bază o valoare diferită față de valoarea primei stive.

b. O estimație admisibilă e numărul de stive vide (SV) din cadrul stării.

c. Consideram min_baza(stare)= {valoarea minimă dintre valorile aflate la baza stivelor din stare} și max_baza(stare)={valoarea maximă dintre valorile aflate la baza stivelor din stare}. O estimație admisibilă h(stare) este max_baza(stare)-min_baza(stare)

d. Consideram min_baza(stare)= {valoarea minimă dintre valorile aflate la baza stivelor din stare}. O estimație admisibilă h(stare) este min_baza(stare)

e. Consideram min_varf(stare)= {valoarea minimă dintre valorile numerice aflate în vârful stivelor din stare}. O estimație admisibilă h(stare) este min_varf(stare) pentru o stare nefinală și 0 pentru una finală.

(Model - 2023) 6. Considerăm următoarea problemă asemănătoare cu problema blocurilor. Avem un număr N de stive pe care avem așezate blocuri ce conțin **litere**. Atunci când mutăm un bloc b1 peste un bloc b2, dacă sub blocul b2 e un bloc b3 care are aceeași literă ca și b1, atunci blocul b2 dispare și blocul b1 ajunge direct plasat peste b3 (atenție, dispariția se întâmplă doar când există exact un bloc între două blocuri cu aceeași literă). Se consideră o stare finală (scop) o stare în care în care au rămas doar blocuri cu aceeași literă (nu există în configurație două blocuri cu litere diferite). Costul mutării unui bloc este egal cu numărul de blocuri care au aceeași literă precum blocul de mutat, din configurație (de exemplu dacă în configurație sunt 5 blocuri cu litera "a" costul mutării oricărui bloc cu litera "a" este 5). Care dintre următoarele moduri de calculare a estimației ĥ duce la o estimație admisibilă?

a. Inițializăm h=0. Luăm pe rând fiecare stivă i și vedem care este litera cu număr maxim de apariții pe acea stivă (de exemplu, pentru stiva: a,a,b,c,a, b, litera ar fi "a"). Numărăm câte blocuri sunt pe stiva i cu literă diferită față de acea literă și adunăm numărul lor la h.

b. Căutăm care este litera (notată cu Lit) cu număr <u>maxim</u> de apariții relativ la toată configurația. Numărăm câte blocuri sunt în configurație cu litera diferită de Lit și considerăm estimația egală cu acest număr.

c. **Căutăm care este litera (notată cu Lit) cu număr <u>minim</u> de apariții relativ la toată configurația. Numărăm câte blocuri sunt în configurație cu litera diferită de Lit și considerăm estimația egală cu acest număr.**

d. Considerăm estimația ca fiind numărul de stive cu litere diferite pe ele.

e. Pentru o stare scop, considerăm estimația 0, iar pentru orice altă stare care nu e scop considerăm estimația ca fiind 2.

(V1.2022) 10. Considerăm următoarea problemă asemănătoare cu problema blocurilor. Avem un număr N de stive pe care avem așezate blocuri ce conțin **numere naturale nenule**. Se consideră o stare finală (scop) orice stare în care suma numerelor de pe fiecare stivă nevidă este **mai mică sau egală cu un număr *k*** dat în fișierul de intrare. Costul mutării unui bloc este egal cu numărul scris pe el. Vom presupune că niciun bloc nu are un număr mai mare decât *k* înscris pe el.

Care dintre urmatoarele moduri de a calcula estimația ĥ pentru o stare dată conduc la o estimație ĥ admisibilă?

a. **Pentru o stivă i, pornind de la vârful stivei în jos calculăm suma sv(i) a numerelor înscrise pe blocuri până când suma sv(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Pentru fiecare stivă i pentru care suma sv(i) nu cuprinde toate blocurile, adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sv(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**
b. **Pentru o stivă i, pornind de la baza stivei în sus calculăm suma sb(i) a numerelor înscrise în blocuri până când suma sb(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Dacă suma sb(i) nu cuprinde toate blocurile, atunci pentru fiecare stivă i adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sb(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**
c. Pentru fiecare stivă i calculăm media numerelor înscrise pe blocurile de pe stiva i. Dacă suma de pe stiva i depășește k, atunci pentru toate blocurile cu număr mai mare decât media, adunăm numărul blocului la valoarea estimației (inițializată cu 0).
d. **Pentru fiecare stivă a cărei sumă depășește k, adunăm 1 la valoarea estimației stării (inițializată cu 0).**
e. **Pentru fiecare stivă a cărei sumă depășește k, adunăm la valoarea estimației stării (inițializată cu 0) numărul cel mai mic dintre numerele de pe acea stivă.**

(V2.2022) 10. Considerăm următoarea problemă asemănătoare cu problema blocurilor. Avem un număr N de stive pe care avem așezate blocuri ce conțin **numere naturale nenule**. Se consideră o stare finală (scop) orice stare în care suma numerelor de pe fiecare stivă nevidă este **mai mică sau egală cu un număr *k*** dat în fișierul de intrare. Costul mutării unui bloc este egal cu numărul scris pe el. Vom presupune că niciun bloc nu are un număr mai mare decât *k* înscris pe el.

Care dintre urmatoarele moduri de a calcula estimația ĥ pentru o stare data conduc la o estimație ĥ admisibila?

a. **Pentru o stivă i, pornind de la vârful stivei în jos calculăm suma sv(i) a numerelor înscrise pe blocuri până când suma sv(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Pentru fiecare stivă i pentru care suma sv(i) nu cuprinde toate blocurile, adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sv(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**
b. Pentru fiecare stivă i calculăm media numerelor înscrise pe blocurile de pe stiva i. Dacă suma de pe stiva i depășește k, atunci pentru toate blocurile cu număr mai mare decât media, adunăm numărul blocului la valoarea estimației (inițializată cu 0).
c. **Pentru o stivă i, pornind de la baza stivei în sus calculăm suma sb(i) a numerelor înscrise în blocuri până când suma sb(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Dacă suma sb(i) nu cuprinde toate blocurile, atunci pentru fiecare stiva i adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sb(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**
d. **Pentru fiecare stivă a cărei sumă depășește k, adunăm 1 la valoarea estimației stării (inițializată cu 0).**
e. Pentru fiecare stivă a cărei sumă depășește k, adunăm la valoarea estimației stării (inițializată cu 0) numărul cel mai mare dintre numerele de pe acea stivă.

(V3.2022) 10. Considerăm următoarea problemă asemănătoare cu problema blocurilor. Avem un număr N de stive pe care avem așezate blocuri ce conțin **numere naturale nenule**. Se consideră o stare finală (scop) orice stare în care suma numerelor de pe fiecare stivă nevidă este **mai mică sau egală cu un număr *k*** dat în fișierul de intrare. Costul mutării unui bloc este egal cu numărul scris pe el. Vom presupune că niciun bloc nu are un număr mai mare decât *k* înscris pe el.

Care dintre urmatoarele moduri de a calcula estimația ĥ pentru o stare data conduc la o estimație ĥ admisibila?

a. **Pentru o stivă i, pornind de la baza stivei în sus calculăm suma sb(i) a numerelor înscrise în blocuri până când suma sb(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Dacă suma sb(i) nu cuprinde toate blocurile, atunci pentru fiecare stiva i adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sb(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**

b. Pentru fiecare stivă i calculăm media numerelor înscrise pe blocurile de pe stiva i. Dacă suma de pe stiva i depășește k, atunci pentru toate blocurile cu număr mai mare decât media, adunăm numărul blocului la valoarea estimației (inițializată cu 0).

c. **Pentru fiecare stivă a cărei sumă depășește k, adunăm la valoarea estimației stării (inițializată cu 0) numărul înscris pe blocul din vârful stivei.**

d. Pentru fiecare stivă a cărei sumă depășește k, adunăm la valoarea estimației stării (inițializată cu 0) numărul cel mai mare dintre numerele de pe acea stivă.

e. **Pentru o stivă i, pornind de la vârful stivei în jos calculăm suma sv(i) a numerelor înscrise pe blocuri până când suma sv(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Pentru fiecare stivă i pentru care suma sv(i) nu cuprinde toate blocurile, adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sv(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**

(V4.2022) 11. Considerăm următoarea problemă asemănătoare cu problema blocurilor. Avem un număr N de stive pe care avem așezate blocuri ce conțin **numere naturale nenule**. Se consideră o stare finală (scop) orice stare în care suma numerelor de pe fiecare stivă nevidă este **mai mică sau egală cu un număr *k*** dat în fișierul de intrare. Costul mutării unui bloc este egal cu numărul scris pe el. Vom presupune că niciun bloc nu are un număr mai mare decât *k* înscris pe el.

Care dintre urmatoarele moduri de a calcula estimația ĥ pentru o stare data conduc la o estimație ĥ admisibila?

a. **Pentru o stivă i, pornind de la baza stivei în sus calculăm suma sb(i) a numerelor înscrise în blocuri până când suma sb(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Dacă suma sb(i) nu cuprinde toate blocurile, atunci pentru fiecare stiva i adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sb(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**

b. **Pentru fiecare stivă a cărei sumă depășește k, adunăm 1 la valoarea estimației stării (inițializată cu 0).**

c. **Pentru o stivă i, pornind de la vârful stivei în jos calculăm suma sv(i) a numerelor înscrise pe blocuri până când suma sv(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Pentru fiecare stivă i pentru care suma sv(i) nu cuprinde toate blocurile, adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sv(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**

d. Pentru fiecare stivă i calculăm media numerelor înscrise pe blocurile de pe stiva i. Dacă suma de pe stiva i depășește k, atunci pentru toate blocurile cu număr mai mare decât media, adunăm numărul blocului la valoarea estimației (inițializată cu 0).

e. **Pentru fiecare stivă a cărei sumă depășește k, adunăm la valoarea estimației stării (inițializată cu 0) numărul înscris pe blocul din vârful stivei.**

(V5.2022) 10. Considerăm următoarea problemă asemănătoare cu problema blocurilor. Avem un număr N de stive pe care avem așezate blocuri ce conțin **numere naturale nenule**. Se consideră o stare finală (scop) orice stare în care suma numerelor de pe fiecare stivă nevidă este **mai mică sau egală cu un număr *k*** dat în fișierul de intrare. Costul mutării unui bloc este egal cu numărul scris pe el. Vom presupune că niciun bloc nu are un număr mai mare decât *k* înscris pe el.

Care dintre urmatoarele moduri de a calcula estimația ĥ pentru o stare data conduc la o estimație ĥ admisibila?

a. **Pentru o stivă i, pornind de la baza stivei în sus calculăm suma sb(i) a numerelor înscrise în blocuri până când suma sb(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Dacă suma sb(i) nu cuprinde toate blocurile, atunci pentru fiecare stiva i adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sb(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**

b. **Pentru o stivă i, pornind de la vârful stivei în jos calculăm suma sv(i) a numerelor înscrise pe blocuri până când suma sv(i) ajunge ori să cuprindă toate blocurile de pe stivă ori să depășească numărul k. Pentru fiecare stivă i pentru care suma sv(i) nu cuprinde toate blocurile, adunăm la valoarea estimației (inițializată cu 0) suma_totala(i)-sv(i), unde suma_totala(i) reprezintă suma numerelor scrise pe totalitatea blocurilor de pe stiva i.**

c. Pentru fiecare stivă i calculăm media numerelor înscrise pe blocurile de pe stiva i. Dacă suma de pe stiva i depășește k, atunci pentru toate blocurile cu număr mai mare decât media, adunăm numărul blocului la valoarea estimației (inițializată cu 0).

d. **Pentru fiecare stivă a cărei sumă depășește k, adunăm la valoarea estimației stării (inițializată cu 0) numărul înscris pe blocul din vârful stivei.**

e. Pentru fiecare stivă a cărei sumă depășește k, adunăm la valoarea estimației stării (inițializată cu 0) suma numerelor de pe toate blocurile aflate deasupra blocului cu cel mai mare număr de pe acea stivă

f. **Pentru fiecare stivă a cărei sumă depășește k, adunăm la valoarea estimației stării (inițializată cu 0) diferența S(i)-k, unde S(i) este suma tuturor numerelor înscrise pe blocurile de pe acea stivă**

(Model - 2023) **10.** Pentru problema X și 0, care dintre următoarele afirmații sunt adevărate?

a. Pentru starea inițială a tablei de joc, reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 1!+2!+3!+4!+5!+6!+7!+8!+9! (9 factorial = 1*2*3*4*5*6*7*8*9).

b. O stare finală a jocului este ori una în care a câștigat MAX ori una în care a câștigat MIN.

c. Arborele minimax pentru X și 0 întotdeauna va avea un număr de niveluri mai mic sau egal cu 10.

d. Pentru starea curentă de mai jos (rădăcină a arborelui curent minimax), presupunând că simbolul calculatorului este X, arborele minimax de adâncime maxima 3 (adică numărul de muchii de pe un lanț de la rădăcină la un nod frunză nu poate depăși 3) are exact 10 noduri, incluzând și rădăcina.

X		
0	X	0
X		0

e. Considerând simbolul calculatorului ca fiind 0 (deci la începutul jocului utilizatorul mută primul) și numerotarea nivelurilor începând de la 0 (0 fiind nivelul rădăcinii), atunci pentru arborele minimax de adâncime maximă 4 având ca rădăcină tabla:

0	X	
X		

putem găsi o stare finală a jocului (câștig, pierdere sau remiză) pe nivelul 1 al arborelui

(V1.2022) 11. Pentru problema X și 0, care dintre următoarele afirmații sunt adevărate?

a. Pentru starea inițială a tablei de joc, reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 9! (9 factorial = 1*2*3*4*5*6*7*8*9).

b. O funcție de evaluare minimax relevantă pentru jocul X și 0, pentru noduri nefinale, este NLD(MAX)-NLD(MIN) unde NLD=numarul de linii deschise pentru jucătorul respectiv. O linie deschisă pentru un jucător este o linie care nu conține simbolul jucătorului opus. Prin linie înțelegem oricare dintre rând, coloană sau diagonală în tabla de joc.

c. O funcție de evaluare minimax relevantă pentru jocul X și 0 este NLD(MAX)-NLD(MIN) când e rândul pentru MAX să mute, respectiv NLD(MIN)-NLD(MAX) când e rândul lui MIN să mute. Am notat cu NLD numărul de linii deschise pentru jucătorul respectiv. O linie deschisă pentru un jucător este o linie care nu conține simbolul jucătorului opus. Prin linie înțelegem oricare dintre rând, coloană sau diagonală în tabla de joc.

d. Numărul de succesori (fii direcți) ai unei stări nefinale în arborele minimax este egal cu n, unde n este numărul de locuri libere de pe tabla de joc.

e. Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiză e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi +99.

X		
0	X	0
X		0

f. Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiza e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi -99.

X		
0	X	0
X		0

(V2.2022) 11. Pentru problema X și 0, care dintre următoarele afirmații sunt adevărate?

a. Pentru starea inițială a tablei de joc, reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 1+(9+8*9+7*8*9+6*7*8*9+....+1*2*3*4*5*6*7*8*9)

x	0	
x	0	
x		

b. Orice stare finală a jocului are proprietatea că tabla este plină (nu mai are spații libere).

c. Numărul de succesori (fii direcți) ai unei stări nefinale în arborele minimax este egal cu n, unde n este numărul de locuri libere de pe tabla de joc.

d. Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiză e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi +99.

X		
0	X	0
X		0

e. Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiza e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi -99.

X		
0	X	0
X		0

f. Presupunem că MAX (calculatorul) foloseste simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiza e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi 0.

		X
0	0	
X	X	0

g. Considerând simbolul calculatorului ca fiind 0 (deci la începutul jocului utilizatorul mută primul) și numerotarea nivelurilor începând de la 0 (0 fiind nivelul rădăcinii), atunci pentru arborele minimax de adâncime maximă 4 având ca rădăcină tabla:

	X	
X	**0**	

putem găsi o stare finală a jocului (câștig, pierdere sau remiză) pe nivelul 2 al arborelui (se presupune că se aplică algoritmul minimax).

(V3.2022) 11. Pentru problema X și 0, care dintre următoarele afirmații sunt adevărate?

a. **O funcție de evaluare minimax relevantă pentru jocul x și 0, pentru noduri nefinale, este NLD(MAX)-NLD(MIN) unde NLD=numarul de linii deschise pentru jucătorul respectiv. O linie deschisă pentru un jucător este o linie care nu conține simbolul jucătorului opus. Prin linie înțelegem oricare dintre rând, coloană sau diagonală în tabla de joc.**

b. O funcție de evaluare minimax relevantă pentru jocul x și 0 este NLD(MAX)-NLD(MIN) cand e rândul pentru MAX să mute, respectiv NLD(MIN)-NLD(MAX) când e rândul lui MIN să mute. Am notat cu NLD numărul de linii deschise pentru jucătorul respectiv. O linie deschisă pentru un jucător este o linie care nu conține simbolul jucătorului opus. Prin linie înțelegem oricare dintre rând, coloană sau diagonală în tabla de joc.

c. Pentru starea inițială a tablei de joc, reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 9! (9 factorial = 1*2*3*4*5*6*7*8*9).

d. **Numărul de succesori (fii direcți) ai unei stări nefinale în arborele minimax este egal cu n, unde n este numărul de locuri libere de pe tabla de joc.**

e. **Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiză e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi +99.**

X		
0	X	0
X		0

f. **Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiza e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi -99.**

		X
0	0	
X	X	0

g. Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiza e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi +99.

		X
0	0	
X	X	0

(V4.2022) 10. Pentru problema X și 0, care dintre următoarele afirmații sunt adevărate?

a. Orice stare finală a jocului are proprietatea că tabla este plină (nu mai are spații libere).

b. Pentru starea inițială a tablei de joc, reprezentată mai jos, calculând arborele minimax în totalitate, fără a impune o adâncime maximă, numărul de noduri din arbore ar fi egal cu 1+(9+8*9+7*8*9+6*7*8*9+....+1*2*3*4*5*6*7*8*9)

c. O funcție de evaluare minimax relevantă pentru jocul x și 0 este NLD(MAX)-NLD(MIN) cand e rândul pentru MAX să mute, respectiv NLD(MIN)-NLD(MAX) când e rândul lui MIN să mute. Am notat cu NLD numărul de linii deschise pentru jucătorul respectiv. O linie deschisă pentru un jucător este o linie care nu conține simbolul jucătorului opus. Prin linie înțelegem oricare dintre rând, coloană sau diagonală în tabla de joc.

d. Numărul de succesori (fii direcți) ai unei stări nefinale în arborele minimax este egal cu n, unde n este numărul de locuri libere de pe tabla de joc.

e. Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiză e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi +99.

X		
0	X	0
X		0

f. Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiza e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi -99.

		X
0	0	
X	X	0

g. Considerând simbolul calculatorului ca fiind 0 (deci la începutul jocului utilizatorul mută primul) și numerotarea nivelurilor începând de la 0 (0 fiind nivelul rădăcinii), atunci pentru arborele minimax de adâncime maximă 4 având ca rădăcină tabla:

	X	
X	**0**	

putem găsi o stare finală a jocului (câștig, pierdere sau remiză) pe nivelul 2 al arborelui (se presupune că se aplică algoritmul minimax).

(V5.2022) 11. Pentru problema X și 0, care dintre următoarele afirmații sunt adevărate?

a. **O funcție de evaluare minimax relevantă pentru jocul x și 0, pentru noduri nefinale, este NLD(MAX)-NLD(MIN) unde NLD=numarul de linii deschise pentru jucătorul respectiv. O linie deschisă pentru un jucător este o linie care nu conține simbolul jucătorului opus. Prin linie înțelegem oricare dintre rând, coloană sau diagonală în tabla de joc.**

b. **Numărul de succesori (fii direcți) ai unei stări nefinale în arborele minimax este egal cu n, unde n este numărul de locuri libere de pe tabla de joc.**

c. **Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiză e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi +99.**

X		
0	X	0
X		0

d. **Presupunem că MAX (calculatorul) folosește simbolul X și utilizatorul simbolul 0. Vom considera următorul mod de calcul al valorii minimax: pentru o stare în care a câștigat MAX, valoarea este +99, pentru o stare în care a câștigat MIN, valoarea e -99, iar pentru remiza e 0. Pentru starea curentă de mai jos, și fără adâncime maximă setată, valoarea minimax va fi -99.**

		X
0	0	
X	X	0

e. **Considerând simbolul calculatorului ca fiind 0 (deci la începutul jocului utilizatorul mută primul) și numerotarea nivelurilor începând de la 0 (0 fiind nivelul rădăcinii), atunci pentru arborele minimax de adâncime maximă 4 având ca rădăcină tabla:**

	X	
X	**0**	

putem găsi o stare finală a jocului (câștig, pierdere sau remiză) pe nivelul 2 al arborelui (se presupune că se aplică algoritmul minimax).

(Model - 2023) 4. Se consideră următorul joc. Avem un grid cu NRLIN linii și 4 coloane în care în starea inițială primele două rânduri conțin simboluri x și ultimele două rânduri au simboluri 0 ca în configurația din imaginea de mai jos. Regulile de mutare sunt următoarele:

- jucătorul cu simbolul x mută primul
- jucătorii își pot muta simbolurile proprii cu o singura pozitie doar în direcția opusă configurației inițiale (x poate muta doar în jos pe coloană sau diagonală iar 0 poate muta doar în sus pe coloană sau diagonală). Jucătorii sunt obligați să facă o mutare. Dacă un jucător nu poate muta când îi vine rândul, atunci e remiză.
- în momentul în care un jucător face 3 simboluri pe linie, coloană sau diagonală poate muta un simbol al celuilalt jucător cu exact o poziție în spate, pe coloană sau diagonală
- scopul fiecărui jucător e să ajungă cu un simbol pe linia din capătul opus poziției sale de start (x să ajungă cu un simbol pe ultima linie și 0 cu un simbol pe prima linie).

Pentru M=8, tabla inițială de joc este ilustrată mai jos:

X	X	X	X
	X	X	
	0	0	
0	0	0	0

Care dintre variantele de mai jos oferă o funcție de evaluare care să arate în mod corect cât de favorabilă este starea jocului pentru calculator (MAX), cu alte cuvinte să aibă o valoare mai mare pentru stări mai favorabile și mai mică pentru stări mai nefavorabile?

a. Numărul de simboluri ale lui MAX din care scădem numărul de simboluri ale lui MIN.
b. Câte configurații de 3 simboluri are MIN din care scădem câte configurații de 3 simboluri are MAX.
c. Numerotarea liniilor și coloanelor începe de la 0 din colțul stânga sus. Calculatorul joaca cu X. Presupunem piesa P cu coordonatele(linie și coloană) LM și CM ca fiind cea mai apropiată piesă a jucătorului MAX de capătul în care trebuie să ajungă pentru a câștiga. Idem, avem piesa p a jucătorului MIN (cea mai apropiată de capatul câștigător al lui MIN) cu coordonatele Lm și Cm. O evaluare ar fi NRLIN-LM-Lm
d. **Numerotarea liniilor și coloanelor începe de la 0 din colțul stânga sus. Calculatorul joaca cu X. Presupunem piesa P cu coordonatele(linie și coloană) LM și CM ca fiind cea mai apropiată piesă a jucătorului MAX de capătul în care trebuie să ajungă pentru a câștiga. Idem, avem piesa p a jucătorului MIN (cea mai apropiată de capătul câștigător al lui MIN) cu coordonatele Lm și Cm. O evaluare ar fi LM+Lm (edit: am corectat)**
e. **O evaluare e reprezentată de câte configurații incomplete (adică de 2 simboluri vecine) are MAX din care scădem numărul de configurații incomplete ale lui MIN**

(Model - 2023) 12.Se consideră un grid NxN că fiind o hartă pătratică în care sunt NROZ obiecte roz, notate cu r și un obiect albastru, notat cu a. Numărul N este natural 5<=N<=10. De asemenea, pe hartă avem și un omuleț notat cu X, așa cum se vede în desen. Scopul omulețului este să adune fie K obiecte roz fie un singur obiect albastru și să se ducă apoi în colțul din stânga sus al matricii (poziția 0,0). În pozitia 0,0 nu poate exista niciun obiect și nici nu poate fi poziția inițială a omulețului. Omulețul se poate deplasa doar cu câte o poziție în direcțiile: sus, jos, stânga și dreapta. Fiecare deplasare costă:

- 1 dacă omulețul nu are obiecte asupra lui
- 100 dacă are obiectul albastru
- NR*5 unde NR e numărul de obiecte roz pe care le are asupra lui.

În momentul în care omulețul intră într-o căsuță(celulă) care contine un obiect automat ia obiectul asupra lui (nu poate refuza să îl ducă). Nu pot fi mai multe obiecte într-o celulă.

Notații:

- Coordonatele omulețului sunt notate cu linie(om), coloana(om).
- Coordonatele obiectului albastru linie(albastru), coloana(albastru) și pentru fiecare obiect roz i avem linie(roz[i]) și coloana(roz[i]). Numarul total de obiecte roz este NROZ.
- distMH(linie1,coloana1,line2,coloana2)=|linie1-linie2|+|coloana1-coloana2| (distanța Manhattan)
- sqrt(x) = radăcina pătrata a numărului x.

Care dintre estimațiile de mai jos sunt admisibile pentru această problemă?

a. **Pentru cazul în care NROZ<K, h(stare)=distMH(linie(om), coloana(om), 0,0)+distMH(linie(om), coloana(om), linie(albastru), coloana(albastru))** Edit: optiunea a este corecta din aceleasi motive ca si optinuea b. Am uitat sa marchez initial raspunsul cand am luat din fisier. Acum e corectat.

b. **Pentru cazul în care NROZ<K, h(stare)=distMH(linie(om), coloana(om),linie(albastru), coloana(albastru))+distMH(linie(albastru), coloana(albastru), 0,0)**

c. **h(stare)=min(100, K*5) pentru o stare nefinală și h(stare)=0 pentru o stare finală.**

d. Fie d1=distMH(linie(om), coloana(om), 0,0)+distMH(linie(om), coloana(om), linie(albastru), coloana(albastru)) și d2=suma(distMH(linie(om), coloana(om), linie(roz[i]), coloana(roz[i]) pentru primele k obiecte roz cele mai apropiate de omuleț în starea curentă). h(stare)=min(100*d1,5*d2).

e. Fie d1=distMH(linie(om), coloana(om),linie(albastru), coloana(albastru)) + 100*distMH(linie(albastru), coloana(albastru), 0,0) și d2=min(distMH(linie(om), coloana(om), linie(roz[i]), coloana(roz[i]) pentru primele k obiecte roz cele mai apropiate de omuleț în starea curentă). *h(stare)={*

   - *min(d1,5*d2) dacă omulețul are mai puțin de K obiecte roz asupra lui,*
   - *5 dacă are K obiecte roz asupra lui dar nu e în stare finală*
   - *0 pentru stare finală*

   }.

(Model - 2023) 5. Pentru un arbore minimax de adâncime maximă A generat pentru un joc oarecare, care dintre următoarele afirmații sunt adevărate?

a. Un nod MIN poate fi rădăcină a arborelui minimax.
b. Dacă aplicăm alpha-beta, nu putem să tăiem mai mult de jumătate dintre nodurile arborelui.
c. Alpha-beta niciodată nu va elimina (reteza) un nod care este fiu direct al rădăcinii.
d. Frunzele sunt întotdeauna noduri MAX.
e. Frunzele sunt întotdeauna la distanță de exact A muchii față de rădacină.

(Model - 2023) 8. În care dintre următoarele situații coada OPEN a algoritmului A* ajunge vidă înainte de returnarea unei soluții?

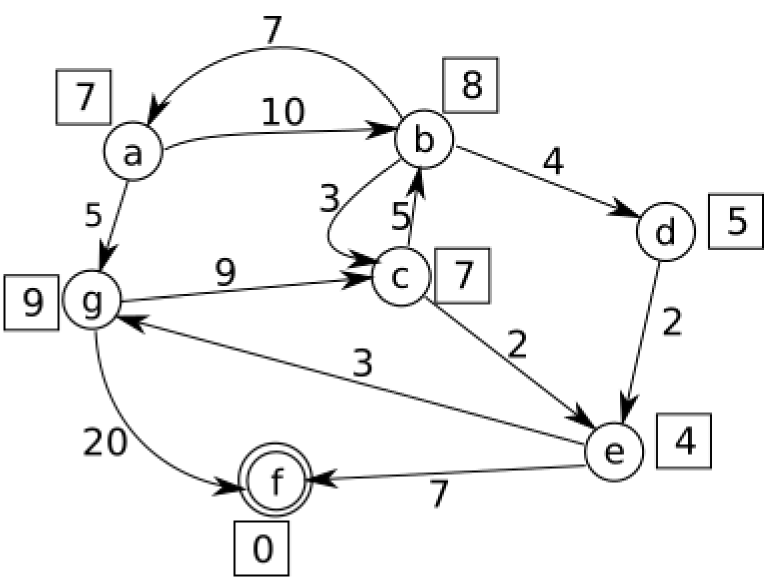
a. Nodurile scop sunt pe altă componentă conexă a grafului față de nodul de start.
b. Se poate ajunge la un nod scop doar trecând printr-o muchie cu cost foarte mare în graf.
c. Condiția scop este imposibilă pentru problema dată.
d. Nodul scop este succesor direct al nodului de start.
e. Coada OPEN nu ajunge niciodată vidă înainte de returnarea unei soluții.

(Model - 2023) 11. Această întrebare este referitoare la algoritmul A* (discutat la laborator sub numele de A* optimizat, cel implementat cu coada OPEN și lista CLOSED). Care dintre următoarele afirmații sunt adevărate?

a. Daca inlocuim costurile pe muchii cu negatul lor (costul c e înlocuit cu -c) putem folosi A* pentru a obține cel mai lung drum-soluție din graf, negând costul soluției oținute cu modificările anterioare.
b. A* returnează un drum soluție numai când un nod scop se află pe prima poziție în coada OPEN ordonată crescător după valoarea funcției euristice de evaluare, $\hat{f}$ aplicată fiecărui nod din OPEN.
c. Un nod poate fi in acelasi timp atat in lista open cât și în lista closed.
d. Ordinea nodurilor din lista open este data de ordinea descoperirii lor de către algoritm

Dacă aplicăm A* pe un anumit graf cu un anumit nod de start *a*, orice valoare (chiar și mai mare decât costul oricârui drum de la a la nodul scop am seta pentru estimația $\hat{h}$(a), dar pentru orice alt nod n, estimația îndeplinește condiția $\hat{h}(n) \leq h(n)$ atunci soluția returnată de A* e in mod cert drumul de cost minim de la a la un nod scop.

REZOLVARE A* SI ALPHA BETA (VARIANTA VECHE DE EXAMEN) - **Exemplu de enunț cu rezolvare**



Enunțuri:

I) Se dă graful de mai jos, cu următoarele caracteristici:
- Nodul *a* este nodul de start
- Nodul cu cerc dublu este nodul scop
- Numărul înscris lângă fiecare arc este costul acelui arc
- Numărul înscris în pătratul de lângă fiecare nod este h estimat (euristica).

Cerințe (**este bine să faceți subpunctele 1 si 2 împreună pentru a putea verifica dacă un nod nu există deja în drumul la care tocmai vreți să îl adăugați; practic cu ajutorul arborelui vedeți drumurile curent extinse. Atât la examen cât și în temă este bine să faceți arborele pe o foaie separată ca să îl desenați în timp ce scrieți listele open și closed de la fiecare pas**):

1) Aplicați algoritmul A* pe acest graf precizând următoarele:
- Cum se inițializează listele **open** și **closed**.
- Descrierea fiecărei iterații (cum se modifică listele open și closed).

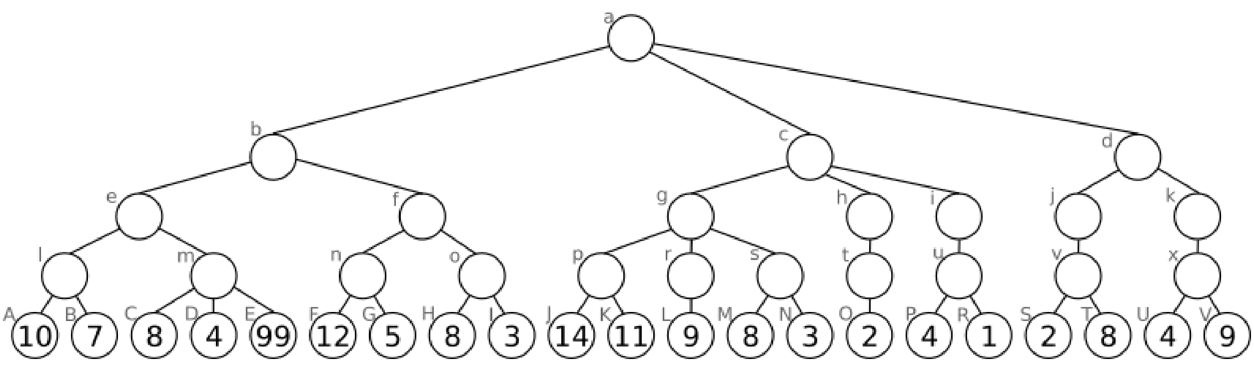
Pentru fiecare nod din listele open și closed se vor scrie următoarele informații în formatul următor (litera nodului, g-ul, f-ul estimat, părintele în arbore). Dacă o informație de nod este readusă din closed în open, trebuie specificat clar acest lucru și explicat de ce se întâmplă asta.
- Scrierea concluziei: care este drumul de cost minim și să se precizeze costul acestuia.

2) Desenați arborele asociat parcurgerii. Pentru fiecare nod scrieți g-ul și f-ul (sub forma unei etichete scrise lângă nodul corespunzător). În arbore se vor reprezenta toate nodurile parcurse și se vor tăia cele care cu o informație pentru care s-a găsit o rută mai bună (au fost înlocuite în coadă de un nod cu aceeași informație dar cu cost mai mic)..

3) Cum ați putea modifica minimal costurile muchiilor (modificați costul a cât mai puține muchii) astfel încât euristica să fie neadmisibilă.

4) Desenați primii 3 arbori generați de IDA*, precizând limita de cost pentru fiecare și arătând modificările prin care trece stiva, evidențiând și întoarcerile.

II) Se dă arborele Minimax (generat de calculator pentru determinarea următoarei mutări) din figură pentru care cunoaștem valorile frunzelor:



1) Etichetați nivelele cu MIN și MAX (atenție la rădăcină!). Completați valorile nodurilor conform algoritmului Minimax. Indicați valoarea jocului și variația principală.

2) Aplicați acestui arbore Algoritmul Alpha-Beta. Desenați arborele rezultat (pe arbore să se vadă cum s-au actualizat la fiecare pas informațiile nodurilor) și explicați operațiile de alpha-beta retezare care au fost efectuate.

3) Reordonați succesorii unui nod din arbore, astfel încât numărul de retezări efectuate de Alpha-Beta să fie mai mare

4) Schimbați valoarea unui singur nod frunză astfel încât să se schimbe valoarea jocului și variația principală să includă alte noduri. Desenați arborele rezultat.

Rezolvări:

I)

1)

Pas 1.

Inițializări:

Open: [(info:a, g:0 , f:7 , parinte:None)]

Closed:[]

Pas 2.

S-a extins a.

Open: [(info:g, g:5, f:14, parinte:a), (info:b, g:10, f:18, parinte:a)]

Closed:[(info:a, g:0 , f:7 , parinte:None)]

Pas 3.

S-a extins g.

Open: [(info:b, g:10, f:18, parinte:a), (info:c, g:14, f:21, parinte:g), (info:f, g:25:, f:25, parinte:g)]

Closed:[(info:a, g:0 , f:7 , parinte:None),(info:g, g:5, f:14, parinte:a)]

Pas 4.

S-a extins b.

Open: [(info:d, g:14, f:19, parinte:b), (info:c, g:13, f:20, parinte:b), (info:f, g:25:, f:25, parinte:g)] //nodul c vechi, din coada open, a fost înlocuit de unul cu cost mai mic

Closed:[(info:a, g:0 , f:7 , parinte:None),(info:g, g:5, f:14, parinte:a), (info:b, g:10, f:18, parinte:a)]

Pas 5.

S-a extins d.

Open: [(info: e, g:16, f:20, parinte:d), (info:c, g:13, f:20, parinte:b), (info:f, g:25:, f:25, parinte:g)] //cand avem f-uri egale ordonăm descrescător după g (în coada open)

Closed:[(info:a, g:0 , f:7 , parinte:None),(info:g, g:5, f:14, parinte:a), (info:b, g:10, f:18, parinte:a), (info:d, g:14, f:19, parinte:b)]

Pas 6.

S-a extins e.

Open: [(info:c, g:13, f:20, parinte:b), (info:f, g:23:, f:23, parinte:g)] //nodul f vechi a fost inlocuit de unul cu cost mai mic

Closed:[(info:a, g:0 , f:7 , parinte:None),(info:g, g:5, f:14, parinte:a), (info:b, g:10, f:18, parinte:a), (info:d, g:14, f:19, parinte:b), (info: e, g:16, f:20, parinte:d)]

Pas 7.

S-a extins c.

Open: [(info: e, g:15, f:19, parinte:c), (info:f, g:23:, f:23, parinte:g)]

//c-ul se extinde în b si e, dar b-ul a mai fost în drumul curent, conform arborelui, deci nu se adaugă, iar e-ul obtinut e cu cost mai mic decât cel din lista closed; prin urmare, îl ștergem din closed pe e și îl adăugăm în open

Closed:[(info:a, g:0 , f:7 , parinte:None),(info:g, g:5, f:14, parinte:a), (info:b, g:10, f:18, parinte:a), (info:d, g:14, f:19, parinte:b), (info:c, g:13, f:20, parinte:b)]

Pas 8.

S-a extins e.

Open: [(info:f, g:22:, f:22, parinte:e)]

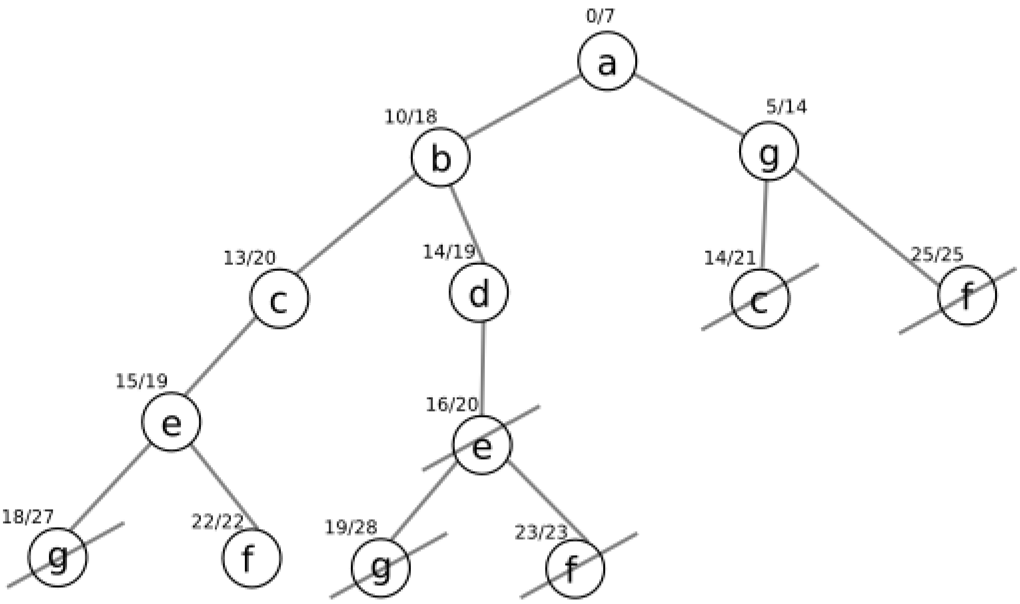
//e-ul se extinde în g si f, dar g-ul obținut e cu cost mai mare decât cel din lista closed, deci nu se adaugă; iar f-ul obținut e cu cost mai mic,deci îl înlocuiește pe cel din lista open.

Closed:[(info:a, g:0 , f:7 , parinte:None),(info:g, g:5, f:14, parinte:a), (info:b, g:10, f:18, parinte:a), (info:d, g:14, f:19, parinte:b), (info:c, g:13, f:20, parinte:b), (info: e, g:15, f:19, parinte:c)]

Primul nod din coada open este un nod scop, rezultă că am obținut drumul de cost minim:

a->b->c->e->f cu costul 22.

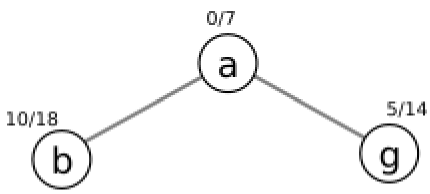
2) Arborele este:



3) Costul arcului e->f să fie modificat în 1; h-ul estimat pentru e, e egal cu 4 și există chiar drumul e->f de cost 1, până la nodul scop. Cum 4>1 reiesel că euristica nu mai e admisibilă.

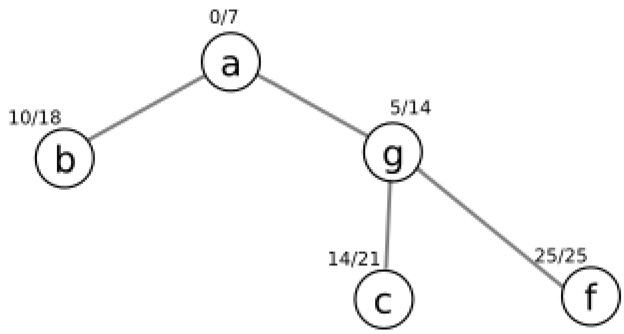
4)
Limita: 7

Stiva:
[a]
[a,b]
<- intoarcere
[a,g]



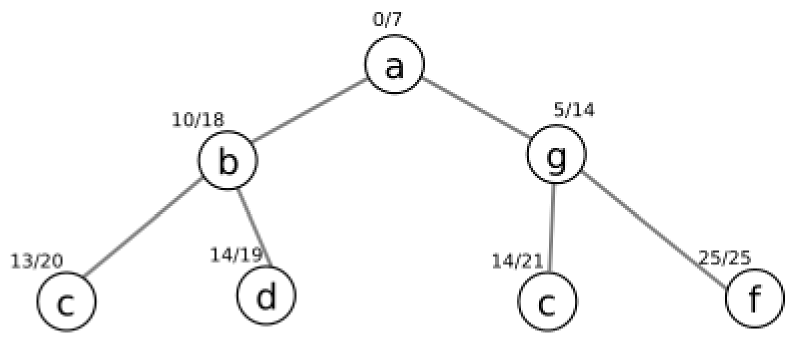
Limita 14:

Stiva:
[a]
[a,b]
<- intoarcere
[a,g]
[a,g,c]
<- intoarcere
[a,g,f]

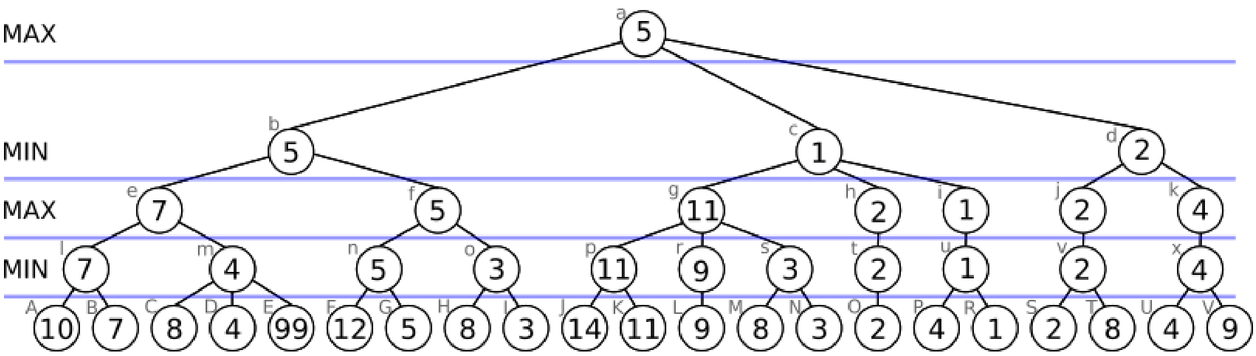


Limita 18:

Stiva:
[a]
[a,b]
[a,b,c]
<- intoarcere
[a,b,d]
<- intoarcere
<- intoarcere
[a,g]
[a,g,c]
<- intoarcere
[a,g,f]

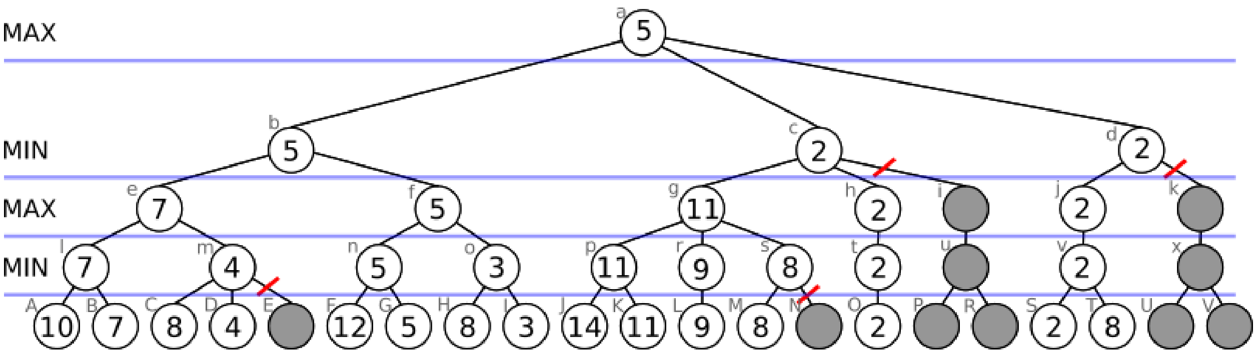


II)



1)

Variatia principală este a-b-f-n-G cu valoarea jocului 5.



2)

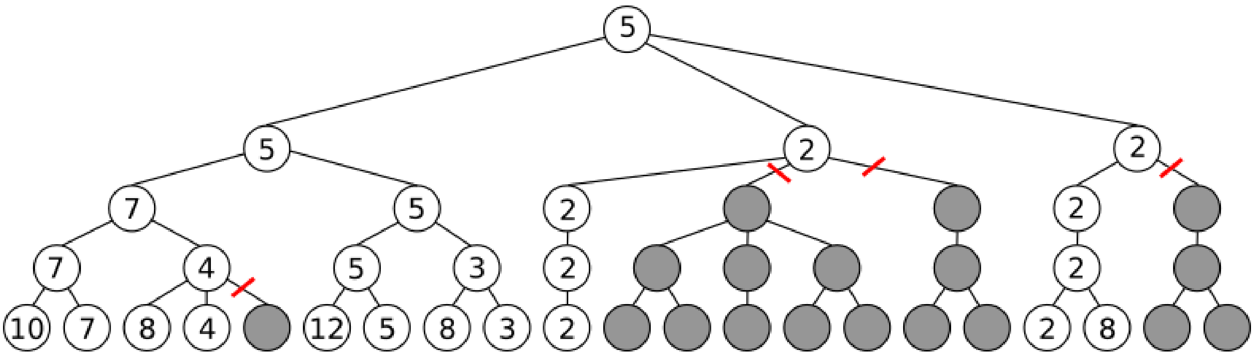
După generarea nodului D cu valoarea 4, MAX are de ales între 7 și o valoare mai mică sau egală cu 4, pentru nodul e, prin urmare frații lui D nu mai trebuie calculați

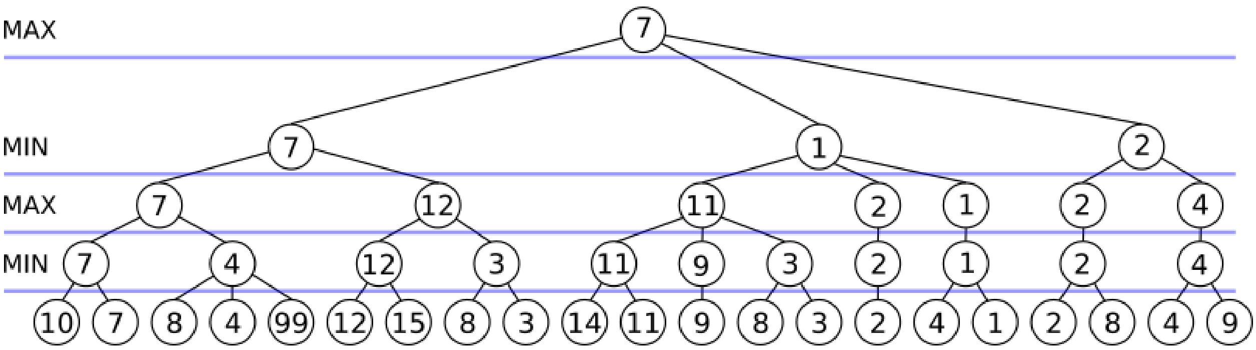
După generarea nodului M cu valoarea 8, MAX are de ales între 11 și o valoare mai mică sau egală cu 8, pentru nodul s, prin urmare frații lui M nu mai trebuie calculați

După evaluarea nodului h la valoarea 2, MAX are de ales între 5 și o valoare mai mică sau egală cu 2, pentru nodul a, prin urmare frații lui c nu mai trebuie calculați

După evaluarea nodului j la valoarea 2, MAX are de ales între 5 și o valoare mai mică sau egală cu 2, pentru nodul a, prin urmare frații lui j nu mai trebuie calculați

3) Reordonare: Observatie: aici initial uitasem sa marchez nodurile care se taie după reordonare; am corectat:





4)

1. (0.25p) Care este numărul de parametri al unui strat convoluțional cu 30 de filtre de dimensiune 5x5 aplicate cu un stride de 2 pe un tensor de dimensiune 64x64x10?

A. 7530 B. 7510 C. 780 D. 750

2. (0.25p) Dacă un nod dintr-un graf computațional reprezintă operația $z = 2xy$, intrările sunt $x = -0.5$ și $y = 2$, iar gradientul $\partial L/\partial z = -10$, atunci gradienții în raport cu intrările $\partial L/\partial x$ și $\partial L/\partial y$ sunt:

A. $\partial L/\partial x = -20$ și $\partial L/\partial y = 5$ B. $\partial L/\partial x = -40$ și $\partial L/\partial y = 10$
C. $\partial L/\partial x = 20$ și $\partial L/\partial y = -5$ D. $\partial L/\partial x = 40$ și $\partial L/\partial y = -10$

3. (0.25p) Cum se descompune eroarea unui model de învățare automată?

A. Eroare de modelare, estimare și optimizare B. Eroare de modelare, învățare și testare
C. Eroare de modelare, optimizare și generalizare D. Eroare de modelare, optimizare și testare

4. (0.25p) Ce determină capacitatea de modelare neliniară a rețelelor neuronale?

A. Matricile de ponderi B. Funcția de pierdere C. Funcțiile de transfer D. Ponderile de tip bias

5. (0.25p) Ce formulă ne dă legătura dintre ponderile primale și cele duale?

A. $w = X' \cdot \alpha$ B. $\alpha = X \cdot w$ C. $w = X \cdot X' \cdot \alpha$ D. $w = K \cdot \alpha$

6. (0.25p) Ce combinație de ponderi produce un hiperplan care separă spațiul $\mathbb{R}^2$ după axa Ox, a.î. partea de deasupra axei să aibă eticheta pozitivă?

A. $w = [-1,1],\ b = 1$ B. $w = [0,-2],\ b = 0$
C. $w = [0,0.5],\ b = 0$ D. $w = [0,1],\ b = 1$

7. (0.25p) Dacă avem o rețea cu un strat cu 12 neuroni, urmat de un strat cu 8 neuroni, atunci dispersia în cazul inițializării Xavier este?

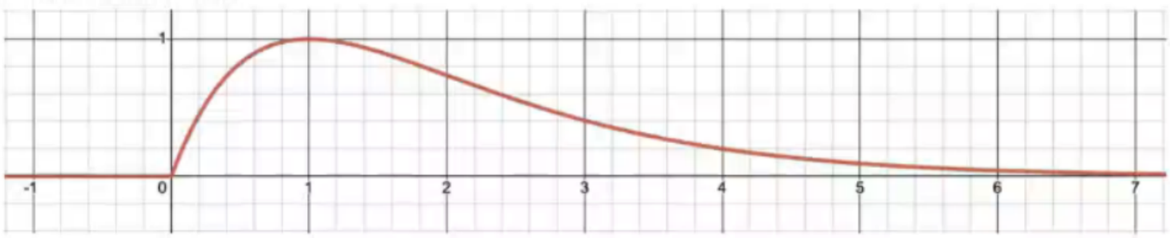
A. 20 B. 10 C. 0.1 D. 0.2

8. (0.25p) Ce garantează posibilitatea de a ajunge la acuratețe 100% pe antrenare cu un model SVM?

A. Eliminarea regularizării B. Parametrul C
C. Kernelul RBF D. Numărul de epoci de antrenare

9. (1p) Considerând funcția kernel $k(x,y) = 2(P - Q)$, unde $P = \left|\left\{(i,j): 1 \le i < j \le n, (x_i - x_j)\cdot(y_i - y_j) \ge 0\right\}\right|$ și Q= $\left|\left\{(i,j): 1 \le i < j \le n, (x_i - x_j)\cdot(y_i - y_j) < 0\right\}\right|$, definiți o funcție de scufundare ϕ, care aplicată lui x și y, produce: $k(x,y) = 2PQ = \langle\phi(x), \phi(y)\rangle$. Exemplificați aplicarea funcției de scufundare pe $x = (0,4,1,2)$ și $y = (1,3,5,1)$, demonstrând egalitatea $2(P - Q) = \langle\phi(x), \phi(y)\rangle$ pe acest exemplu.

10. (0.5p) Fiind dată funcția de activare $y = f(x) = max(0,x) \cdot exp(-x + 1)$ cu graficul de mai jos, precizați dacă un singur neuron artificial bazat pe această funcție de activare ar putea rezolva problema XOR. Justificați răspunsul printr-un exemplu.



10. (1p) Fiind dată rețeaua neuronală definită mai jos, să se rezolve următoarele subpuncte:

$$f(x) = hardlim\left([-1,-1,-1] \cdot sign\left(\begin{bmatrix} -1 & -1 \\ 0 & 1 \\ 1 & 1 \end{bmatrix} \cdot x + \begin{bmatrix} 0.5 \\ -0.5 \\ -1.5 \end{bmatrix}\right)\right), \text{ unde } hardlim(x) = \begin{cases} 1, x \ge 0 \\ 0, x < 0 \end{cases}$$

a) (0.5p) Desenați arhitectura rețelei definită mai sus;

b) (0.25p) Calculați ieșirea rețelei pe intrarea $x = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$;

c) (0.5p) Propuneți două modificări minimale ale arhitecturii a.î. aceasta să rezolve problema XOR;

d) (0.25p) Demonstrați că noua rețea (cea de la punctul c) produce rezultatul dorit.

Examen IA

ML - Varianta 2

1. Eroarea de generalizare este aproximată în practică prin:

A. Eroare de estimare B. Eroare optimală C. Eroare de optimizare

D. Eroare empirică

2. Funcția $f(x) = \begin{cases} x, & x \geq 0 \\ \lambda(e^x - 1), & x < 0 \end{cases}$ corespunde cărei funcții de activare?

A. Sigmoidă B. Tangentă hiperbolică C. PReLU D. ELU

3. În cazul în care folosim $\beta = 0$ în cadrul măsurii F_β, care din următoarele egalități este adevărată?

A. $F_\beta = R$ B. $F_\beta = P + R$ C. $F_\beta = P$ D. $F_\beta = P \cdot R$

4. Aplicând algoritmul coborârii pe gradient cu moment $m = 0.0$ și rata de învățare $r = 0.1$, optimizând funcția $f(x) = x^3 - 2x + 3$, se ajunge după mai mulți pași la $x = 3$. Care este următoarea valoare pentru x?

A. 4.6 B. 1.6 C. 1.4 D. Nu se poate calcula

5. Cum poate fi corectată varianța unei rețele neuronale cu 2 straturi

A. Eliminând un strat B. Adăugând un strat C. Folosind Xavier

D. Crescând rata de învățare

6. Care din următoarele modele nu își actualizează ponderile pe bază de gradienți?

A. Modelul Widrow-Hoff B. Perceptronul Rosenblatt

C. Rețeaua neuronală cu 1 strat D. Rețeaua neuronală cu cel puțin 2 straturi

7. Care din următoarele formule exprimă rata de fals pozitiv

A. $FP / (TP + FP)$ B. $FP / (FP + TN)$ C. $TP / (TP + FP)$

D. $FP / (FP + FN)$

8. Care din următoarele operații sunt echivalente cu Dropout?

A. Blurarea cu filtru Gaussian B. Blurarea cu filtru de medie

C. Augmentarea cu zgomot de tip piper D. Augmentarea cu zgomot Gaussian

9. Se dă mulțimea de exemple de antrenare $S = \{([-2, 1], 1), ([-3, 1], 1), ([0, 2], -1), ([-3, 2], 1)\}$ și exemplul de testare $[-1, 2]$. Rezolvați următoarele subpuncte

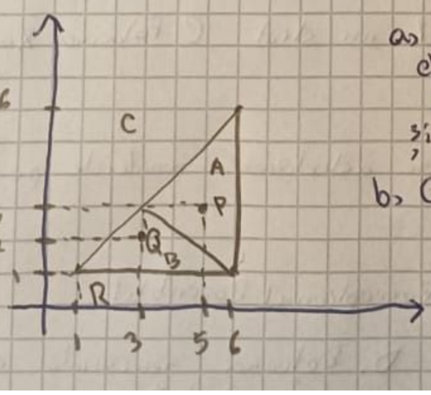
a) Calculați funcția de scufundare asociată funcției nucleu polinomiale cu $d = 3$ și $c = 3$

b) Aplicați funcția de scufundare de la pct-ul a) pe ex. de test

c) Știind că ponderile duale ale unui model SVM bazat pe funcție nucleu polinomială cu $d = 3$ și $c = 3$ sunt $\alpha_1 = 0,5$, $\alpha_2 = 0,5$, $\alpha_3 = -2$, $\alpha_4 = 1$, estimați ponderile primale aferente acestui model

d) Determinați eticheta exemplului de test folosind clasificatorul SVM primal cu w calculat la punctul b) și $b = -3$. Pentru a obține eticheta se aplică fct semn.

10. Fiind date mulțimile A, B și C din fig de mai jos:



a) Construiți o rețea neuronală care să clasifice corespunzător punctele din mulț. A, B și C. Desenați arhitectura rețelei și care ponderile asociate fiecărui strat

b) Care ieșirea rețelei pt pct P, Q, R din fig

11. În urma antrenării un SVM hard-margin pe mulțimea de 4 pct repr colțurile pătratului unitate cu etichetele date de funcția logică OR, se obțin ponderile $w = [1, 1]$ și $b = -0,5$. Cum se modifică poziția hiperplanului de separare prin adăugarea exemplului $[-1, -1]$ cu eticheta 0? Justificați

IA - Învățare automată **8 iunie 2022**

Examen – Varianta 1

Oficiu (0.5p)

1. (0.25p) Dacă un nod dintr-un graf computațional reprezintă operația $z = 2xy$, intrările sunt $x = -0.5$ și $y = 2$, iar gradientul $\partial L/\partial z = -5$, atunci gradienții în raport cu intrările $\partial L/\partial x$ și $\partial L/\partial y$ sunt:
A. $\partial L/\partial x = -20$ și $\partial L/\partial y = 5$ B. $\partial L/\partial x = -40$ și $\partial L/\partial y = 10$
C. $\partial L/\partial x = 20$ și $\partial L/\partial y = -5$ D. $\partial L/\partial x = 40$ și $\partial L/\partial y = -10$

2. (0.25p) Care este scufundarea asociată funcției nucleu RBF?
A. $f(x) = x$ B. $f(x) = \sqrt{x}$
C. $k(x,y) = e^{-\frac{\Delta(x,y)}{2\sigma}}$, unde $\Delta(x,y)$ este distanța Euclidiană D. Nu există

3. (0.25p) Cum se descompune eroarea unui model de învățare automată?
A. Eroare de modelare, estimare și optimizare B. Eroare de modelare, învățare și testare
C. Eroare de modelare, optimizare și generalizare D. Eroare de modelare, optimizare și testare

4. (0.25p) La ce se referă teorema „no free lunch"?
A. Lansarea procesului de antrenare este costisitoare B. Rețelele neuronale aproximează orice funcție
C. Nu există un singur model bun pe toate task-urile D. Rețele neuronale sunt în general soluția optimă

5. (0.25p) Când ar putea să apară fenomenul Hughes?
A. Când avem prea multe exemple B. Când avem prea multe trăsături
C. Când algoritmul de optimizare nu converge D. Când avem un model cu capacitate de modelare prea mică

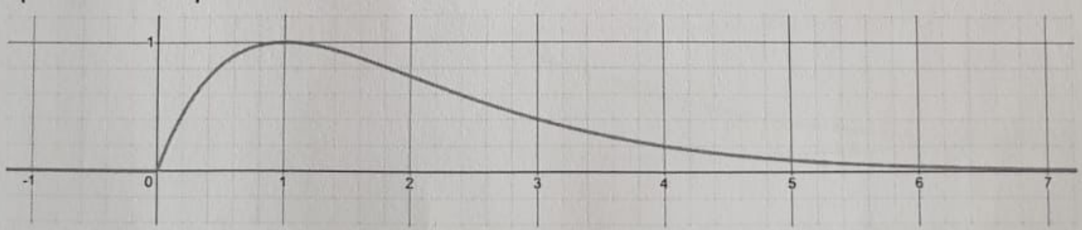
6. (0.25p) Care este numărul de parametri al unui strat convoluțional cu 30 de filtre de dimensiune 5x5 aplicate cu un stride de 2 pe un tensor de dimensiune 64x64x10?
A. 7510 B. 7530 C. 750 D. 780

7. (0.25p) Ce determină capacitatea de modelare neliniară a rețelelor neuronale?
A. Matricile de ponderi B. Funcția de pierdere C. Funcțiile de transfer D. Ponderile de tip bias

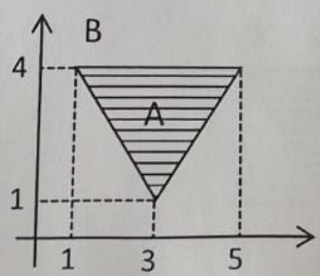
8. (0.25p) Care este dimensiunea tensorului de ieșire al unui strat de pooling cu filtre de 2x2 aplicate la stride de 4 pe un tensor de dimensiune 66x66x40?
A. 16x16x40 B. 31x31x40 C. 17x17x40 D. 32x32x40

9. (1p) Considerând funcția kernel $k(x,y) = 2PQ$, unde $P = |\{(i,j): 1 \leq i < j \leq n, (x_i - x_j) \cdot (y_i - y_j) \geq 0\}|$ și Q= $|\{(i,j): 1 \leq i < j \leq n, (x_i - x_j) \cdot (y_i - y_j) < 0\}|$, definiți o funcție de scufundare ϕ, care aplicată lui x și y, produce: $k(x,y) = 2PQ = \langle \phi(x), \phi(y) \rangle$. Exemplificați aplicarea funcției de scufundare pe $x = (0,4,1,2)$ și $y = (1,3,5,1)$, demonstrând egalitatea $2PQ = \langle \phi(x), \phi(y) \rangle$ pe acest exemplu.

10. (0.5p) Fiind dată funcția de activare $y = f(x) = max(0,x) \cdot exp(-x+1)$ cu graficul de mai jos, precizați dacă un singur neuron artificial bazat pe această funcție de activare ar putea rezolva problema XOR. Justificați răspunsul printr-un exemplu.



11. (1p) Configurați o rețea neuronală (specificând arhitectura și valorile parametrilor) care să discrimineze între mulțimile de puncte A (eticheta 1) și B (eticheta -1) reprezentate în figura de mai jos.



v1 a d a c b b c c

IA - Învățare automată

Examen – Varianta 2

8 iunie 2022

Oficiu (0.5p)

1. (0.25p) Eroarea de modelare provine din alegerea:
A. unui spațiu de ipoteze prea larg
B. unui model cu capacitate slabă de modelare
C. unui algoritm de optimizare nepotrivit
D. unui spațiu de ipoteze circular

2. (0.25p) La ce se referă teorema de aproximare universală?
A. Orice model de învățare poate aproxima orice funcție
B. Parametrii unei rețele se pot aproxima din date
C. Rețelele neuronale pot aproxima orice funcție
D. Hiperparametrii se pot aproxima pe validare

3. (0.25p) Dacă un nod dintr-un graf computațional reprezintă operația $z = 2xy^2$, intrările sunt $x = -0.5$ și $y = 2$, iar gradientul $\partial L/\partial z = -5$, atunci gradienții în raport cu intrările $\partial L/\partial x$ și $\partial L/\partial y$ sunt:
A. $\partial L/\partial x = 20$ și $\partial L/\partial y = -5$
B. $\partial L/\partial x = -40$ și $\partial L/\partial y = 10$
C. $\partial L/\partial x = -20$ și $\partial L/\partial y = 5$
D. $\partial L/\partial x = 40$ și $\partial L/\partial y = -10$

4. (0.25p) Care este numărul de parametri al unui strat convoluțional cu 10 de filtre de dimensiune 5x5 aplicate cu un stride de 2 pe un tensor de dimensiune 64x64x30?
A. 750
B. 7530
C. 7510
D. 760

5. (0.25p) Ce determină capacitatea de modelare neliniară a rețelelor neuronale?
A. Funcția de pierdere
B. Ponderile de tip bias
C. Matricile de ponderi
D. Funcțiile de activare

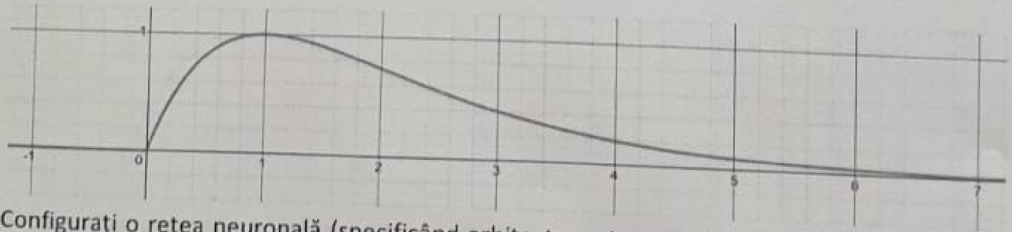
6. (0.25p) Care este dimensiunea tensorului de ieșire al unui strat de pooling cu filtre de 4x4 aplicate la stride de 2 pe un tensor de dimensiune 66x66x40?
A. 32x32x40
B. 31x31x40
C. 16x16x40
D. 17x17x40

7. (0.25p) Care este scufundarea asociată funcției nucleu Hellinger?
A. $f(x) = \sqrt{x}$
B. Nu există
C. $k(x, y) = \langle\sqrt{x}, \sqrt{y}\rangle$, unde $\langle\blacksquare, \blacksquare\rangle$ este produsul scalar
D. $f(x) = x$

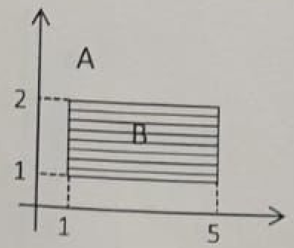
8. (0.25p) Când ar putea să apară fenomenul Hughes?
A. Când avem prea puține exemple
B. Când algoritmul de optimizare nu converge
C. Când avem prea puține trăsături
D. Când avem un model cu capacitate de modelare prea mică

9. (1p) Considerând funcția kernel $k(x, y) = \sum_i \min\{x_i, y_i\}$, unde x și y sunt histograme cu frecvența de apariție a unor cuvinte, având frecvența maximă egală cu 5, definiți o funcție de scufundare ϕ, care aplicată lui x și y, produce: $k(x, y) = \sum_i \min\{x_i, y_i\} = \langle\phi(x), \phi(y)\rangle$. Exemplificați aplicarea funcției de scufundare pe $x = (0,4,1,2)$ și $y = (3,2,5,3)$, demonstrând egalitatea $\sum_i \min\{x_i, y_i\} = \langle\phi(x), \phi(y)\rangle$ pe acest exemplu.

10. (0.5p) Fiind dată funcția de activare $y = f(x) = max(0, x) \cdot exp(-x + 1)$ cu graficul de mai jos, precizați dacă un singur neuron artificial bazat pe această funcție de activare ar putea rezolva problema XOR. Justificați răspunsul printr-un exemplu.



11. (1p) Configurați o rețea neuronală (specificând arhitectura și valorile parametrilor) care să discrimineze între mulțimile de puncte A (eticheta 1) și B (eticheta -1) reprezentate în figura de mai jos.



v2 b c b c d a a a

IA - Învățare automată

8 iunie 2022

Examen – Varianta 3

Oficiu (0.5p)

1. (0.25p) Cum se descompune eroarea unui model de învățare automată?
A. Eroare de modelare, optimizare și testare B. Eroare de modelare, estimare și optimizare
C. Eroare de modelare, învățare și testare D. Eroare de modelare, optimizare și generalizare

2. (0.25p) Când ar putea să apară fenomenul Hughes?
A. Când avem prea multe exemple B. Când algoritmul de optimizare nu converge
C. Când avem prea multe trăsături D. Când avem un model cu capacitate de modelare prea mică

3. (0.25p) Care este numărul de parametri al unui strat convoluțional cu 20 de filtre de dimensiune 3x3 aplicate cu un stride de 3 pe un tensor de dimensiune 128x128x40?
A. 200 B. 180 C. 7240 D. 7220

4. (0.25p) Care este scufundarea asociată funcției nucleu RBF?
A. Nu există B. $f(x) = x$
C. $k(x,y) = e^{-\frac{\Delta(x,y)}{2\sigma}}$, unde $\Delta(x,y)$ este distanța Euclidiană D. $f(x) = \sqrt{x}$

5. (0.25p) Ce determină capacitatea de modelare neliniară a rețelelor neuronale?
A. Matricile de ponderi B. Funcțiile de transfer C. Funcția de pierdere D. Ponderile de tip bias

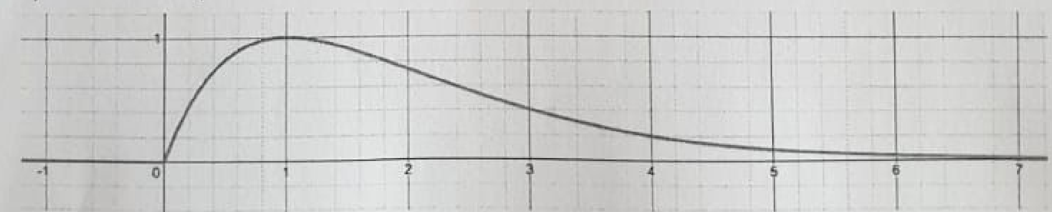
6. (0.25p) Dacă un nod dintr-un graf computațional reprezintă operația $z = 2xy$, intrările sunt $x = -0.5$ și $y = 2$, iar gradientul $\partial L/\partial z = 5$, atunci gradienții în raport cu intrările $\partial L/\partial x$ și $\partial L/\partial y$ sunt:
A. $\partial L/\partial x = -40$ și $\partial L/\partial y = 10$ B. $\partial L/\partial x = 40$ și $\partial L/\partial y = -10$
C. $\partial L/\partial x = 20$ și $\partial L/\partial y = -5$ D. $\partial L/\partial x = -20$ și $\partial L/\partial y = 5$

7. (0.25p) Care este dimensiunea tensorului de ieșire al unui strat convoluțional cu 40 filtre de 3x3 aplicate la stride de 2 pe un tensor de dimensiune 65x65x20?
A. 16x16x20 B. 31x31x20 C. 32x32x40 D. 31x31x40

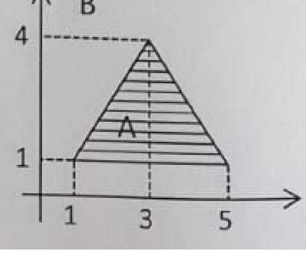
8. (0.25p) La ce se referă teorema „no free lunch"?
A. Rețele neuronale sunt în general soluția optimă B. Lansarea procesului de antrenare este costisitoare
C. Rețelele neuronale aproximează orice funcție D. Nu există un singur model bun pe toate task-urile

9. (1p) Considerând funcția kernel $k(x,y) = 2PQ$, unde $P = \left|\{(i,j): 1 \le i < j \le n, (x_i - x_j)\cdot(y_i - y_j) \ge 0\}\right|$ și $Q = \left|\{(i,j): 1 \le i < j \le n, (x_i - x_j)\cdot(y_i - y_j) < 0\}\right|$, definiți o funcție de scufundare ϕ, care aplicată lui x și y, produce: $k(x,y) = 2PQ = \langle\phi(x), \phi(y)\rangle$. Exemplificați aplicarea funcției de scufundare pe $x = (5,2,1,0)$ și $y = (1,3,5,1)$, demonstrând egalitatea $2PQ = \langle\phi(x), \phi(y)\rangle$ pe acest exemplu.

10. (0.5p) Fiind dată funcția de activare $y = f(x) = max(0,x)\cdot exp(-x+1)$ cu graficul de mai jos, precizați dacă un singur neuron artificial bazat pe această funcție de activare ar putea rezolva problema XOR. Justificați răspunsul printr-un exemplu.



11. (1p) Configurați o rețea neuronală (specificând arhitectura și valorile parametrilor) care să discrimineze între mulțimile de puncte A (eticheta 1) și B (eticheta -1) reprezentate în figura de mai jos.



v3 b c d a b c c d

IA - Învățare automată 8 iunie 2022

Examen – Varianta 4

Oficiu (0.5p)

1. (0.25p) Care este scufundarea asociată funcției nucleu Hellinger?
A. $f(x) = x$ B. $f(x) = \sqrt{x}$
C. $k(x, y) = \langle\sqrt{x}, \sqrt{y}\rangle$, unde $\langle\blacksquare,\blacksquare\rangle$ este produsul scalar D. Nu există

2. (0.25p) Ce determină capacitatea de modelare neliniară a rețelelor neuronale?
A. Funcțiile de activare B. Ponderile de tip bias C. Matricile de ponderi D. Funcția de pierdere

3. (0.25p) Care este dimensiunea tensorului de ieșire al unui strat convoluțional cu 20 filtre de 3x3 aplicate la stride de 2 pe un tensor de dimensiune 65x65x40?
A. 32x32x20 B. 31x31x20 C. 16x16x40 D. 31x31x40

4. (0.25p) La ce se referă teorema de aproximare universală?
A. Orice model de învățare poate aproxima orice funcție B. Parametrii unei rețele se pot aproxima din date
C. Hiperparametrii se pot aproxima pe validare D. Rețelele neuronale pot aproxima orice funcție

5. (0.25p) Eroarea de modelare provine din alegerea:
A. unui spațiu de ipoteze prea restrâns B. unui model cu capacitate prea mare de modelare
C. unui algoritm de optimizare nepotrivit D. unui spațiu de ipoteze circular

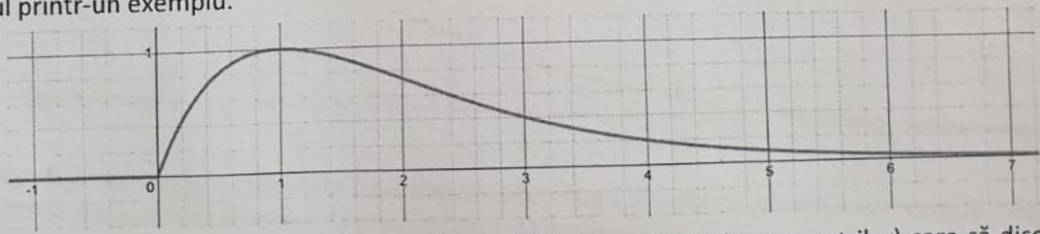
6. (0.25p) Care este numărul de parametri al unui strat convoluțional cu 40 de filtre de dimensiune 3x3 aplicate cu un stride de 3 pe un tensor de dimensiune 128x128x20?
A. 400 B. 360 C. 7240 D. 7220

7. (0.25p) Când ar putea să apară fenomenul Hughes?
A. Când algoritmul de optimizare nu converge B. Când avem prea puține exemple
C. Când avem un model cu capacitate de modelare prea mică D. Când avem prea puține trăsături

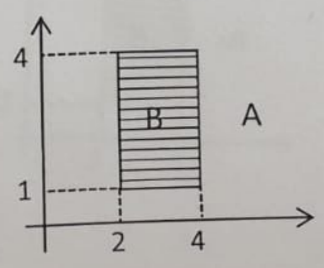
8. (0.25p) Dacă un nod dintr-un graf computațional reprezintă operația $z = 2xy^2$, intrările sunt $x = -0.5$ și $y = 2$, iar gradientul $\partial L/\partial z = -5$, atunci gradienții în raport cu intrările $\partial L/\partial x$ și $\partial L/\partial y$ sunt:
A. $\partial L/\partial x = 20$ și $\partial L/\partial y = -5$ B. $\partial L/\partial x = -20$ și $\partial L/\partial y = 5$
C. $\partial L/\partial x = -40$ și $\partial L/\partial y = 10$ D. $\partial L/\partial x = 40$ și $\partial L/\partial y = -10$

9. (1p) Considerând funcția kernel $k(x,y) = \sum_i \min\{x_i, y_i\}$, unde x și y sunt histograme cu frecvența de apariție a unor cuvinte, având frecvența maximă egală cu 5, definiți o funcție de scufundare ϕ, care aplicată lui x și y, produce: $k(x,y) = \sum_i \min\{x_i, y_i\} = \langle\phi(x), \phi(y)\rangle$. Exemplificați aplicarea funcției de scufundare pe $x = (4,0,3,2)$ și $y = (3,2,5,4)$, demonstrând egalitatea $\sum_i \min\{x_i, y_i\} = \langle\phi(x), \phi(y)\rangle$ pe acest exemplu.

10. (0.5p) Fiind dată funcția de activare $y = f(x) = max(0, x) \cdot exp(-x + 1)$ cu graficul de mai jos, precizați dacă un singur neuron artificial bazat pe această funcție de activare ar putea rezolva problema XOR. Justificați răspunsul printr-un exemplu.



11. (1p) Configurați o rețea neuronală (specificând arhitectura și valorile parametrilor) care să discrimineze între mulțimile de puncte A (eticheta 1) și B (eticheta -1) reprezentate în figura de mai jos.



v4 b a a d a c b c