Model de subiect
(Examen)

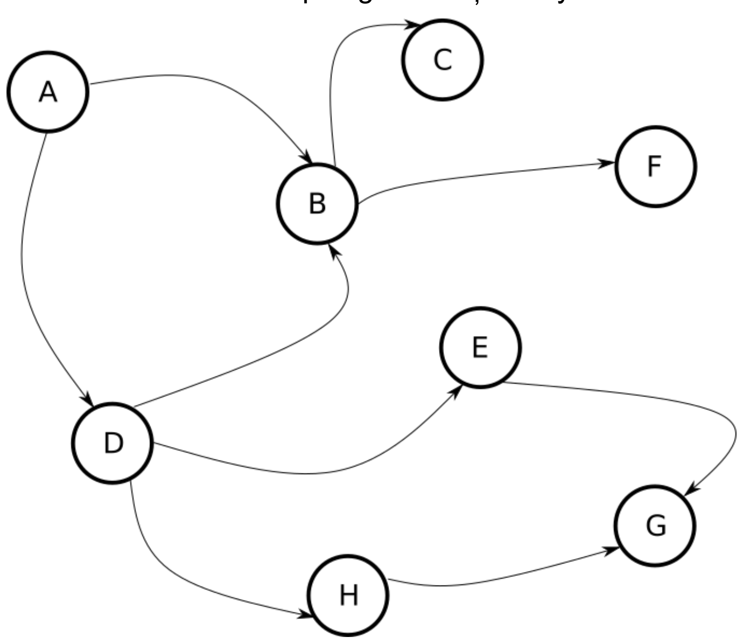
1. Pentru un arbore minimax de adâncime maximă A generat pentru un joc oarecare, care dintre următoarele afirmații sunt adevărate?

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.

2. Care dintre următoarele afirmații sunt adevărate?

a. Algoritmul A* parcurge întotdeauna întreg spațiul stărilor în căutarea unei soluții.
b. Algoritmul A* întoarce întotdeauna soluția de cost minim indiferent de euristică.
c. Algoritmul A* este o tehnică de căutare neinformată.
d. Algoritmul A* admite mai multe stări scop.
e. Algoritmul A* poate fi implementat folosind o coadă de priorități pentru lista OPEN (lista nodurilor de expandat).

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}

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).

Tabla inițială de joc arată așa pentru M=8

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 MAX din care scădem câte configurații de 3 simboluri are MIN.
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
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

5. Se consideră regulile de mai jos:
Dacă **miaună**, **e acoperit de blană** și are **culoare_închisa** atunci **animalul e pisică** cu certitudine 80.
Dacă **miaună** și **e acoperit de pene** atunci **animalul e păun** cu certitudine 50.
Dacă **latră** atunci **animalul e cățel** cu certitudine 90.
Dacă **are urechi lungi** e iepure cu certitudine 80.
Dacă **are culoare neagră** atunci **are culoare închisă** cu certitudine 100.
Dacă **are culoare maro** atunci **are culoare închisă** cu certitudine 100.
Dacă **are culoare albă** atunci **nu are culoare închisă** cu certitudine 100.

Presupunem că un utilizator observă un animal și dorește să consulte sistemul expert cu referire la acel animal. Care dintre următoarele afirmații este adevărată?
a. Presupunem că sistemul expert dorește să verifice că animalul este pisică și pune întrebări utilizatorului (precum "miauna?", "are blană?", respectiv "ce culoare are?" pentru a determina dacă are sau nu culoare închisă). In acest caz, sistemul expert realizează o căutare de tip înlănțuire înapoi.
b. Dacă animalul observat miaună dar nu are culoare inchisă, sistemul expert nu poate oferi o soluție.
c. Dacă sistemul expert cere întâi introducerea tuturor informațiilor despre animalul observat și pe baza lor trage concluzii intermediare, precum faptul că are sau nu o culoare închisă, iar prin deducții înlănțuite ajunge la concluzia că animalul este pisică, atunci sistemul expert realizează o căutare de tip înlănțuire înapoi.

d. **Dacă utilizatorul răspunde "da" la întrebarea "latră", atunci o soluție a sistemului expert este cățel, însă nu e obligatoriu să fie singura.**

6. Considerăm următoarea problemă asemănătoare cu problema blocurilor. Avem un număr N de stive pe care avem așezate blocuri ce conțin **litere**. Atunci cănd 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 $\hat{h}$ 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.**

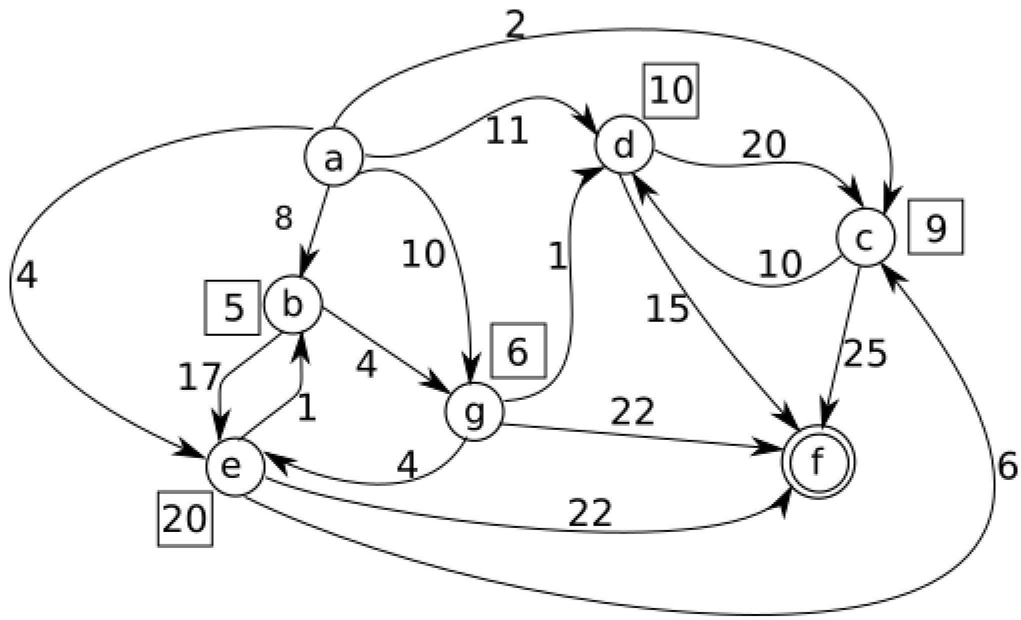
b. **Căutăm care este litera (notată cu Lit) cu număr <u>maxim</u> de apariții relativ la toată configurația. Numărăm câte blocuri sunt în configurație cu litera diferită de Lit și considerăm estimația egală cu acest număr.**

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.**

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).

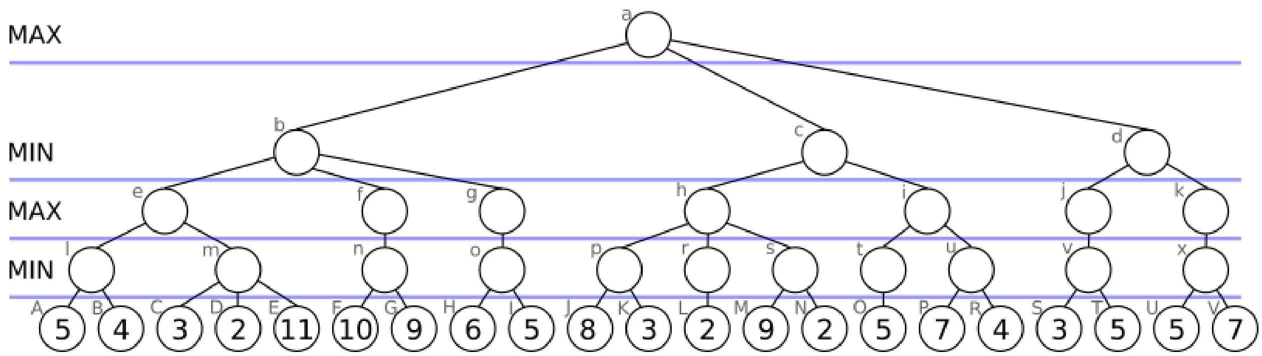


a. a
b. b
c. c
d. d
e. e
f. f
g. g

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.

9. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:

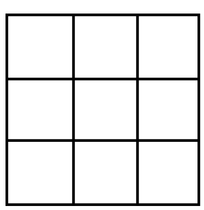


Care dintre următoarele afirmații sunt adevărate?

a. Dacă am aplica alpha-beta pe acest joc atunci nodul *m* nu ar mai fi evaluat.
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.

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

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

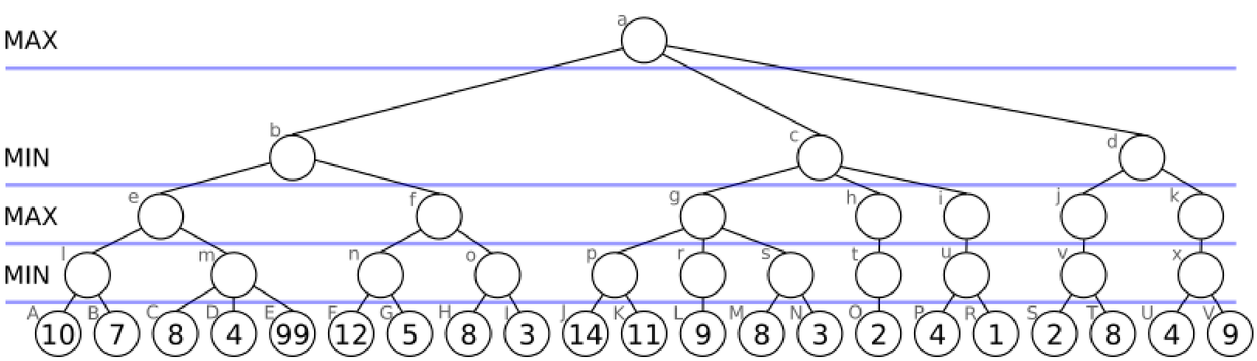
NUME, GRUPĂ:____
DATĂ:______________________

Examen la disciplina Inteligență Artificială,
partea de “Căutarea și Reprezentarea Cunoștințelor”
- Varianta 1 -

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)

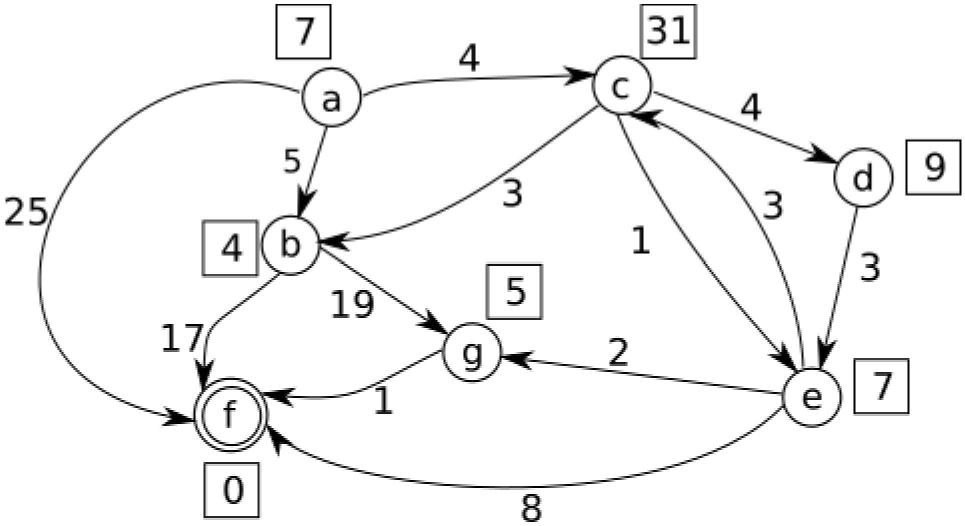
2. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:



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.

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

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.

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*, $\hat{h}(n) = \min(\{ct \mid ct$ cost al arcului a, $a \in A\})$ (altfel spus, ct este costul minim pe un arc din graf)
d. $\hat{h}(n) = \min(\{ct \mid ct$ 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. $\hat{h}(n) = \min(\{ct \mid ct$ 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

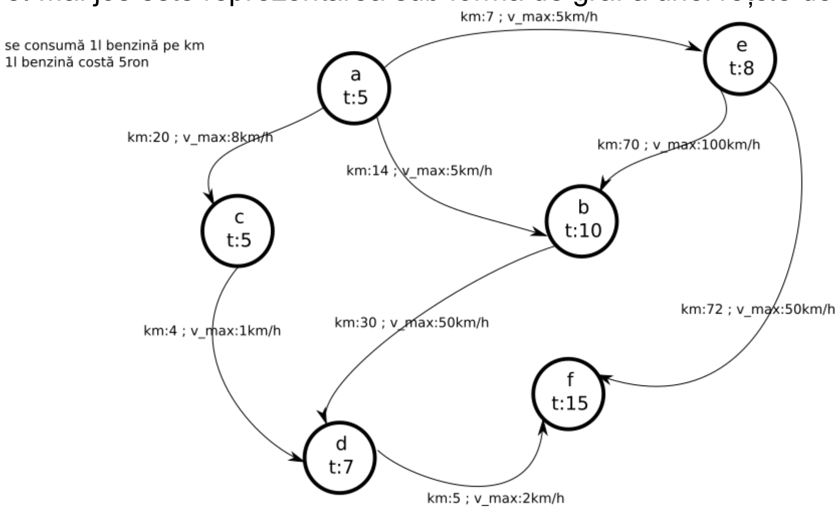
6. Pentru o problemă oarecare (deci afirmația trebuie să fie adevărată pentru orice problemă de căutare cu costuri strict pozitive pe tranziții), care dintre formulele următoare ar obține din estimația admisibilă $\hat{h}$(nod), oricare ar fi aceasta (adică formula trebuie să fie adevărată **pentru orice $\hat{h}$** 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. $\hat{h}_1(\text{nod}) = \hat{h}(\text{nod}) / 3$
b. $\hat{h}_1(\text{nod}) = \hat{h}(\text{nod}) * 2$
c. $\hat{h}_1(\text{nod}) = \hat{h}(\text{nod}) + 4$
d. $\hat{h}_1(\text{nod}) = \hat{h}(\text{nod})^3$ (cu sensul de ridicare la puterea a 3-a)
e. niciuna dintre formulele de la celelalte subpuncte

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.

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.

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**?

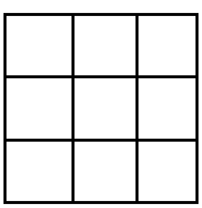
- **a. Numărul de perechi de plăcuțe cu același număr care nu sunt alăturate.**
- b. Numărul de plăcuțe care nu au niciun vecin cu același număr pe linie sau coloană.
- c. S-1, unde S este suma distanțelor d(i), unde d(i) este distanța Euclidiană dintre două plăcute cu numărul i, considerând drept coordonate 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.

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.**

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

NUME, GRUPĂ:____
DATĂ:___________________

Examen la disciplina Inteligență Artificială,
partea de "Căutarea și Reprezentarea Cunoștințelor"
- Varianta 2 -

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 ĥ(nod), oricare ar fi aceasta (adică formula trebuie să fie adevărată **pentru orice ĥ** nu pentru cazuri particulare de ĥ), 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)=ĥ(nod)/2
   b. $\hat{h}_1$(nod)=ĥ(nod)*2
   c. $\hat{h}_1$(nod)=ĥ(nod)+2, dacă *nod* nu e nod scop și $\hat{h}_1$(nod)=0 dacă *nod* este scop
   d. $\hat{h}_1$(nod)=ĥ(nod)^3 (cu sensul de ridicare la puterea a 3-a)
   e. niciuna dintre formulele de la celelalte subpuncte

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 ĥ sunt admisibile **pentru orice astfel de graf**?

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

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.

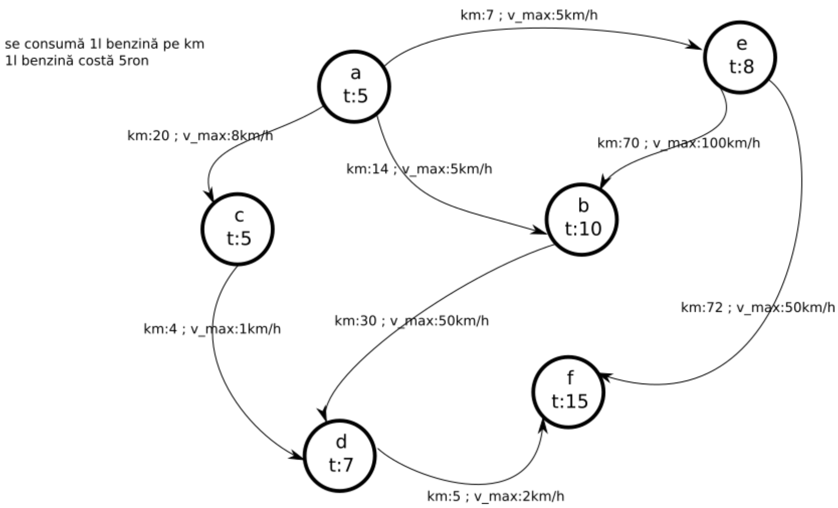
4. Pentru problema blocurilor (vezi anexa), considerând o stare cu N stive și în care niciuna dintre stive nu e vidă, care este numărul de succesori direcți ai acesteia (stări în care se poate ajunge din starea curentă printr-o singură mutare)?

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

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.**

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.

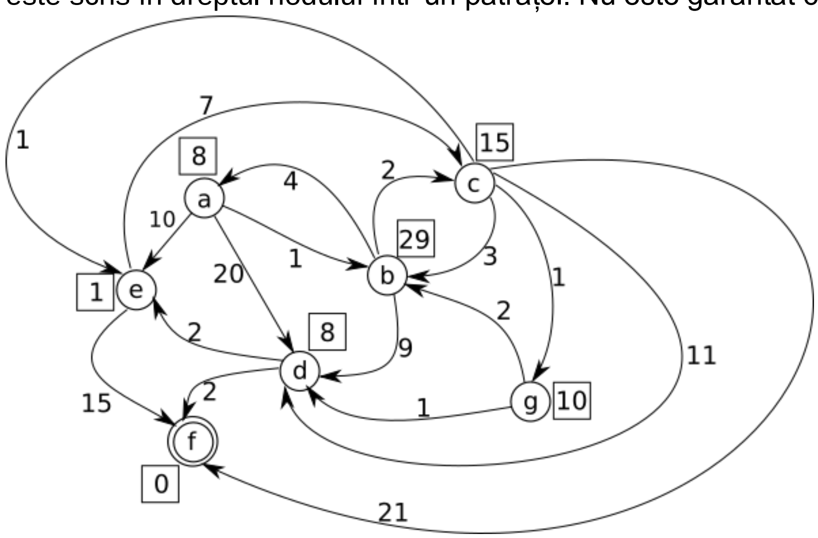
7. Considerăm problema 8-puzzle cu următoarea modificare: cele 8 plăcuțe sunt numerotate cu numere de la 1 la 4 și există exact câte două plăcuțe numerotate cu același număr. Un exemplu de stare inițială este:

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.

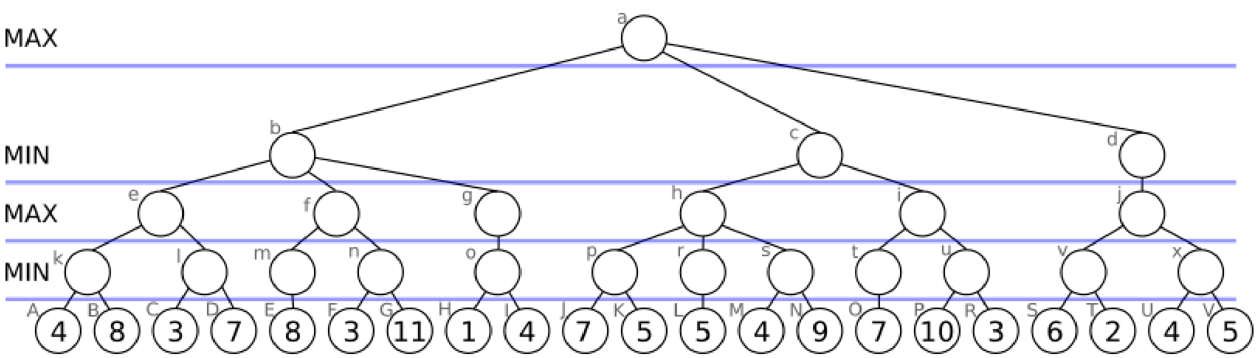
8. Se consideră următorul graf cu muchii ponderate (costul unei muchii este scris în dreptul ei). Nodul de start este nodul *a* și nodul scop este *f*. Factorul euristic al fiecărui nod (estimația ĥ) este scris în dreptul nodului într-un pătrățel. Nu este garantat că estimația este admisibilă.



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

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.

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.

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:

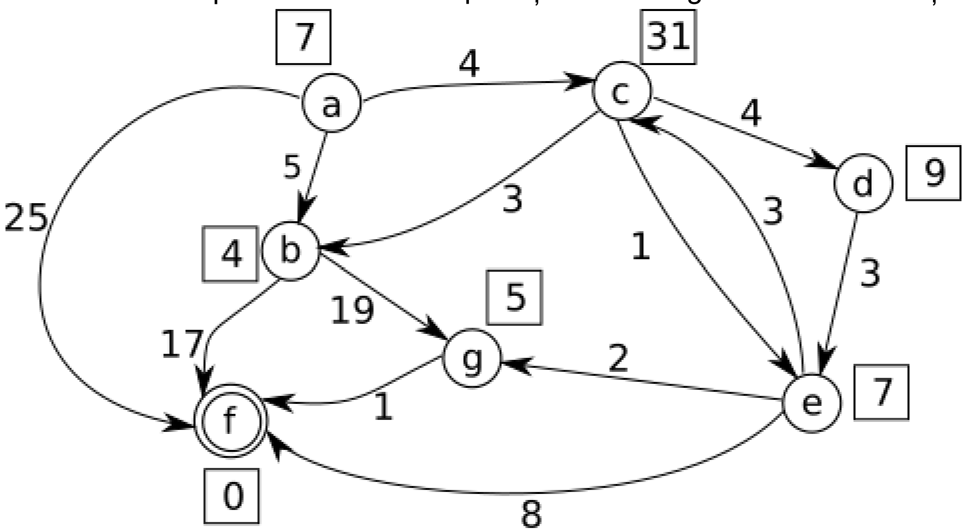
	X	
X	0	

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

NUME, GRUPĂ:____
DATĂ:______________________

Examen la disciplina Inteligență Artificială,
partea de "Căutarea și Reprezentarea Cunoștințelor"
- Varianta 3 -

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

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.**

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 $\in$ 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

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

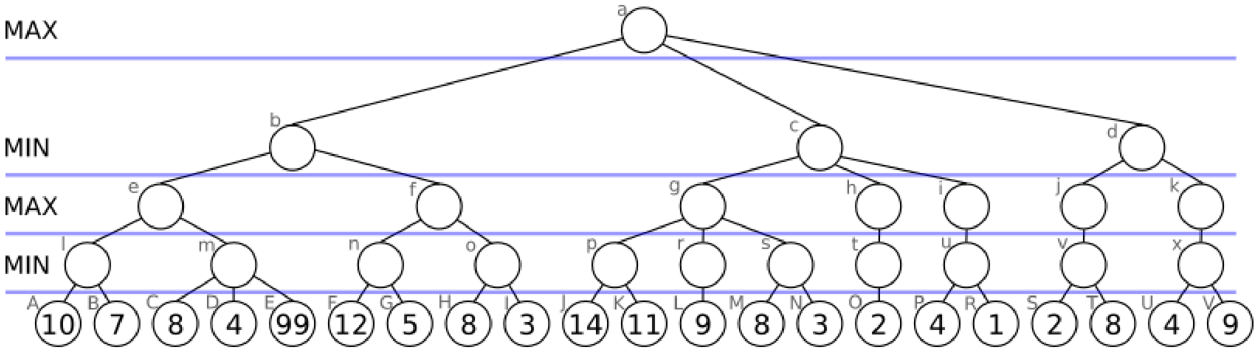
b. $\hat{h}_1$(nod)=$\hat{h}$(nod)*5

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)

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.

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).

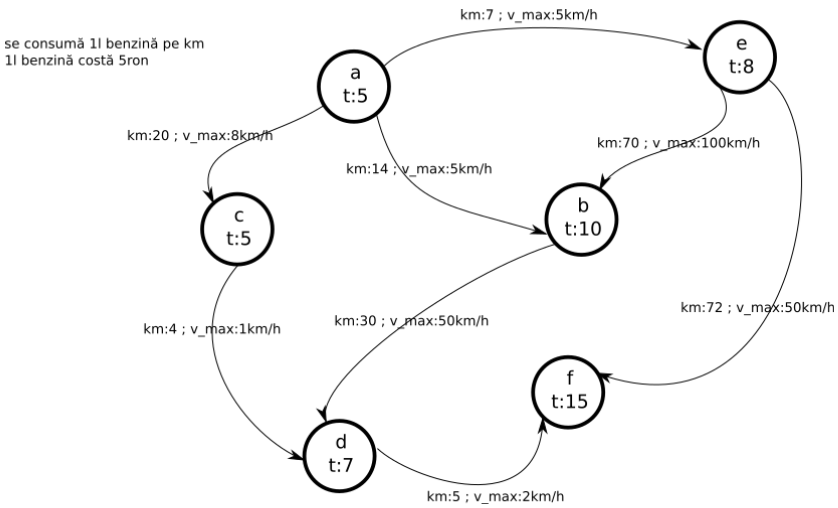
c. Întotdeauna gradul interior (numărul de arce care intră în nod) al unui nod este mai mic sau egal cu 1.

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.

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.**

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.

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)

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.**

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

NUME, GRUPĂ:____

DATĂ:___________________

Examen la disciplina Inteligență Artificială,
partea de "Căutarea și Reprezentarea Cunoștințelor"
- Varianta 4 -

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.

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

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).

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)

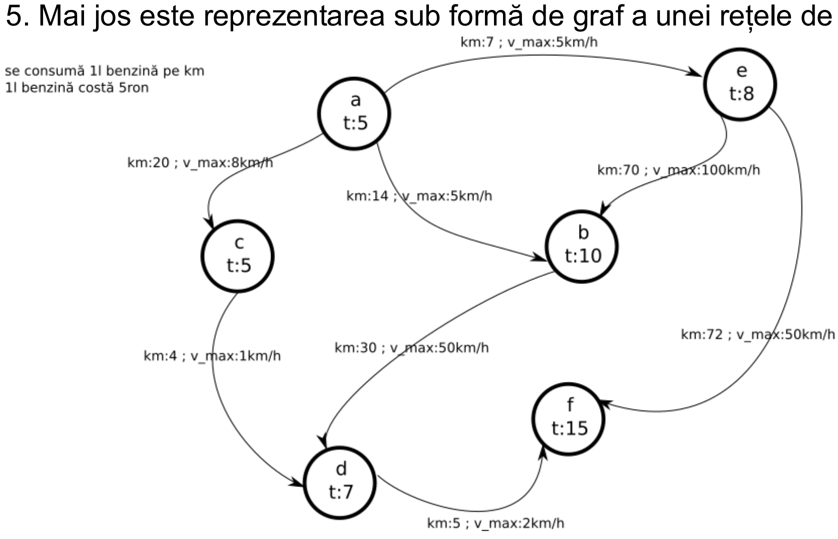
c. (N - NV - 1) * N

d. (N - 1) * N

e. (N - 1) * (N - NV)

f. (N - NV) * (N - NV)

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.**

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:

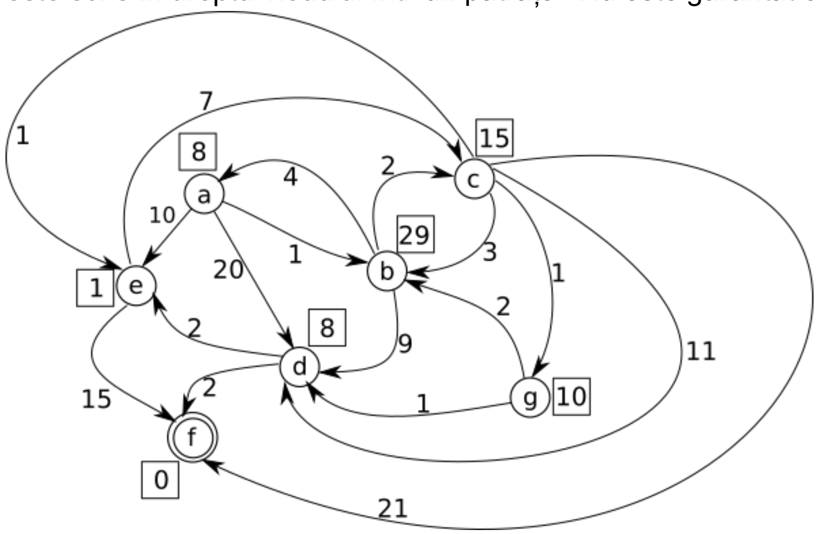
1	2	1
4	3	4
2	3	

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.

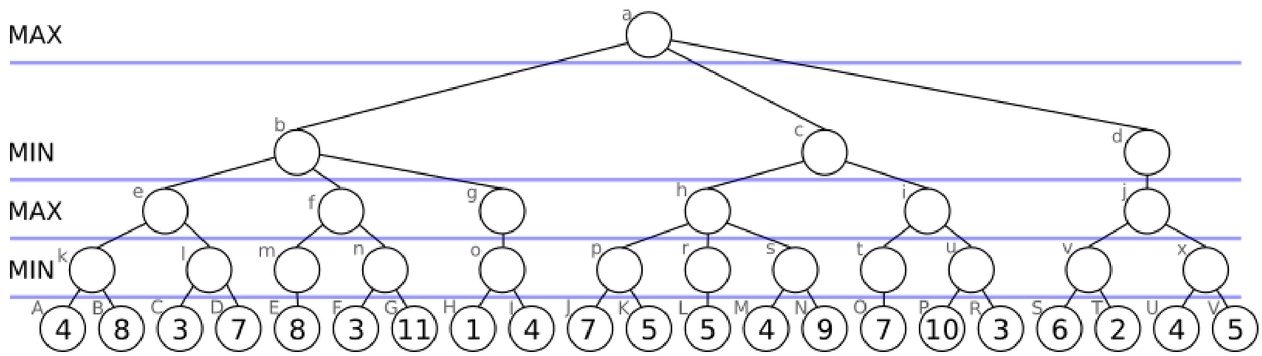
7. Se consideră următorul graf cu muchii ponderate (costul unei muchii este scris în dreptul ei). Nodul de start este nodul *a* și nodul scop este *f*. Factorul euristic al fiecărui nod (estimația ĥ) este scris în dreptul nodului într-un pătrățel. Nu este garantat că estimația este admisibilă.



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

8. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:



Care dintre următoarele afirmații sunt adevărate?

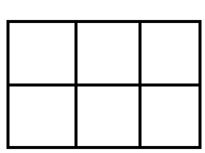
a. **În nodul *b* vom avea valoarea minimax 1.**
b. În nodul ***a*** vom avea valoarea minimax 11.
c. Dacă am aplica algoritmul alpha-beta pe acest arbore, nodul C nu ar mai fi evaluat
d. Aplicând algoritmul minimax, nodul *g* va avea în mod sigur valoarea 4.
e. **Oricare două noduri-frați am inversa,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

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 ĥ sunt admisibile **pentru orice astfel de graf**?

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

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).

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.**

NUME, GRUPĂ:____
DATĂ:______________________

Examen la disciplina Inteligență Artificială,
partea de "Căutarea și Reprezentarea Cunoștințelor"
- Varianta 5 -

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 ĥ sunt admisibile **pentru orice astfel de graf**?

- ĥ(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 ĥ(n) = 0 dacă *n* este nod scop
- ĥ(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 ĥ(n) = 0 dacă *n* este nod scop
- ĥ(n) = 1 dacă *n* nu este nod scop, ĥ(n) = 0 dacă n este nod scop
- Pentru orice nod *n*, ĥ(n) = min({ct| ct cost al arcului a, a ∈ A}) (altfel spus, ct este costul minim pe un arc din graf)
- ĥ(n) = 0 oricare ar fi nodul *n* din graf

2. Care dintre afirmațiile de mai jos sunt adevărate?

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

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 ĥ(nod), oricare ar fi aceasta (adică formula trebuie să fie adevărată **pentru orice ĥ** <u>nu pentru cazuri particulare de ĥ</u>), o nouă estimație $\hat{h}_1$(nod) **<u>în mod cert</u> 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.**

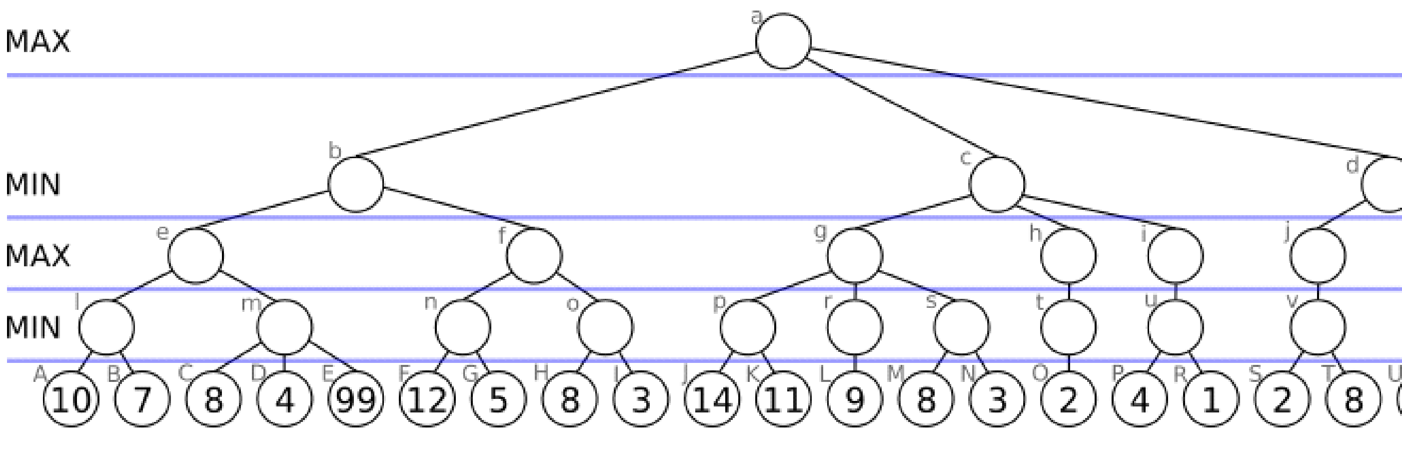
- $\hat{h}_1$(nod)=ĥ(nod)/3+5
- $\hat{h}_1$(nod)=ĥ(nod)+0.5
- $\hat{h}_1$(nod)=ĥ(nod)*1000

- $\hat{h}_1(nod)=2*\hat{h}(nod)^3$ (cu sensul de ridicare la puterea a 3-a)
- niciuna dintre formulele de la celelalte subpuncte

4. Care dintre următoarele afirmații sunt adevărate pentru rețele Bayesiene?

- 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).
- Întotdeauna gradul interior (numărul de arce care intră în nod) al unui nod este mai mic sau egal cu 1.
- Orice graf orientat cu arce având asociate ponderi poate reprezenta topologia unei rețele Bayesiene.
- Topologia unei rețele Bayesiene este un graf orientat (direcționat).
- 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.

5. Considerăm arborele Minimax de mai jos pentru care cunoaștem valorile din frunze:



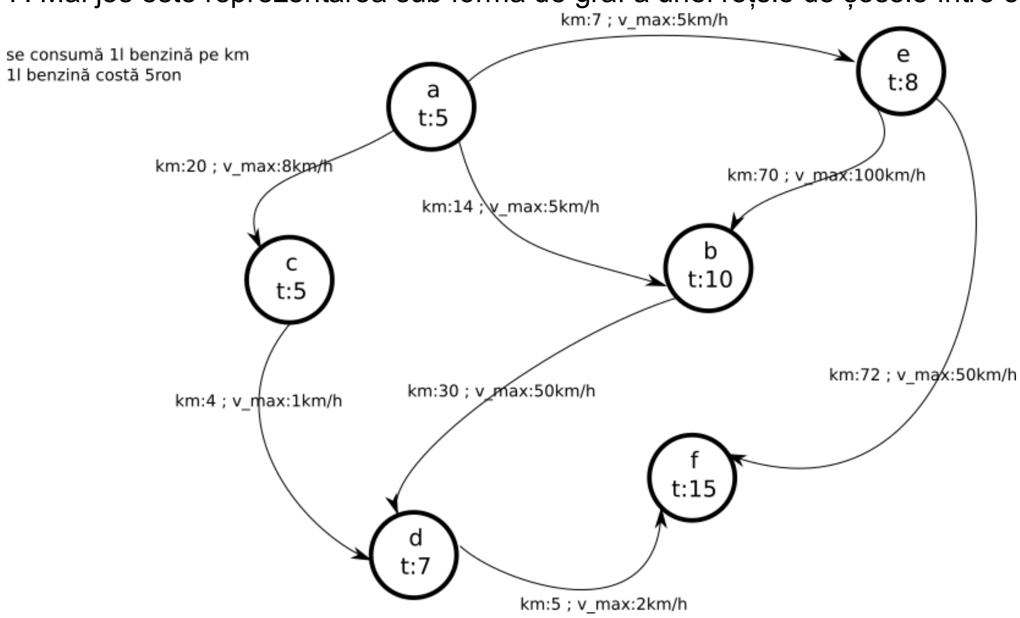
Care dintre următoarele afirmații sunt adevărate?

- În nodul ***a*** vom avea valoarea minimax 99.
- Aplicând algoritmul minimax, nodul *h* va avea în mod sigur valoarea 2.
- 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.
- Dacă am aplica algoritmul alpha-beta pe acest arbore, nodurile D și E nu ar mai fi evaluate.
- 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.

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)?

- (N - 1) * B
- N
- N * NB
- (N - 1) * (NB - 1)
- (N - 1) * NB
- N * (NB - 1)

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

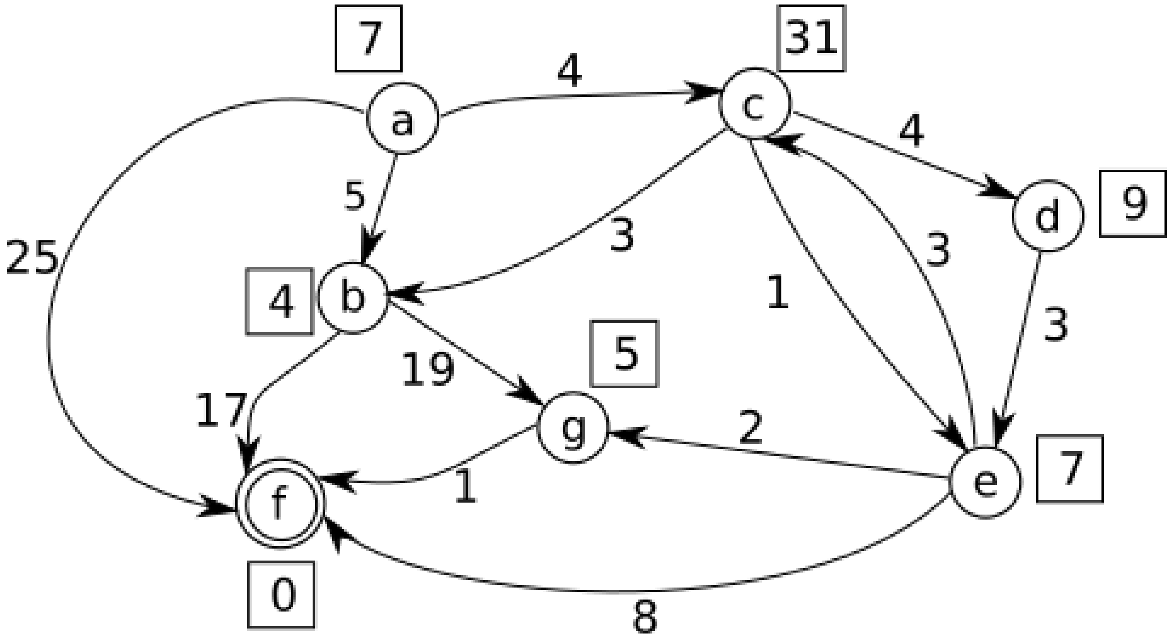


Orașele sunt reprezentate ca noduri, iar muchiile reprezintă șoselele. Pe fiecare muchie este trecută lungimea sa în kilometri, dar și viteza maximă admisă. La intrarea într-un oraș se plătește o taxă (specificată în fiecare nod prin t:taxă în RON). Dorim să folosim în mod corect A*, cu un cost relevant 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?

- 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.
- 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.
- 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.
- 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.

- 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.
- 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.

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

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**?

- M-1, unde M este minimul dintre distanțele d(i), unde d(i) este distanța Manhattan dintre două plăcuțe inscripționate cu numărul i.
- N/3, unde N este numărul de plăcuțe care nu au niciun vecin cu același număr pe linie sau coloană.
- Numărul de perechi de plăcuțe cu același număr care nu sunt alăturate.
- Numărul de plăcuțe care nu au niciun vecin cu același număr pe linie sau coloană.
- 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.
- S-1, unde S este suma distanțelor Manhattan dintre fiecare două plăcuțe inscripționate cu același număr.

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?

- 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.
- 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.

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

11. Pentru problema X și 0, care dintre următoarele afirmații sunt adevărate?

- 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.
- 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.
- 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

- 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

- 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).