

Model de subiect (Examen)

Se consideră notațiile și convențiile de la curs.

1. Se consideră problema blocurilor cu următoarele modificări.

- Există două tipuri de blocuri:
 - Blocuri numerice - Informația fiecărui bloc este un 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 restricții:
 - 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 inițială cu S stive, **scopul** este să avem pe toate stivele, la bază numere egale între ele (să nu existe două 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: $S \geq 3$, $NB > 4$, $SV < S$.

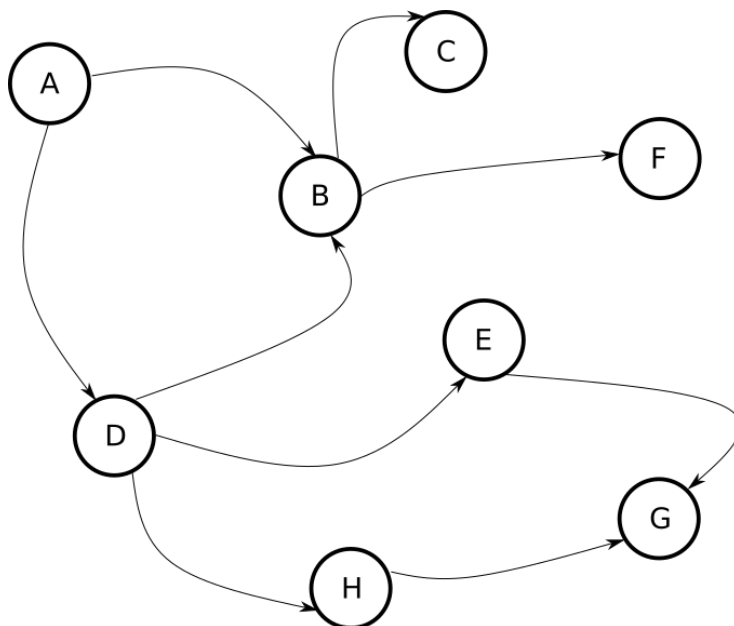
Care dintre următoarele afirmații sunt adevărate?

- a. În mulțimea de stări inițiale posibile (cu parametri 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 parametri 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 $2 \cdot (S-SV-2)$
- d. Numărul de succesori valizi ai unei stări oarecare este $(S-SV-2) \cdot (S-3)$
- e. **Edit: lipsa o restricție pe care am uitat să 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 $(S-SV-2) \cdot (S-1)$

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

- O estimație admisibilă e numărul de stive nevide care au la bază o valoare diferită față de valoarea primei stive.
- O estimație admisibilă e numărul de stive vide (SV) din cadrul stării.
- Consideram $\min_baza(stare) = \{\text{valoarea minimă dintre valorile aflate la baza stivelor din stare}\}$ și $\max_baza(stare) = \{\text{valoarea maximă dintre valorile aflate la baza stivelor din stare}\}$. O estimație admisibilă $h(stare)$ este $\max_baza(stare) - \min_baza(stare)$
- Consideram $\min_baza(stare) = \{\text{valoarea minimă dintre valorile aflate la baza stivelor din stare}\}$. O estimație admisibilă $h(stare)$ este $\min_baza(stare)$
- Consideram $\min_varf(stare) = \{\text{valoarea minimă dintre valorile numerice aflate în vârful stivelor din stare}\}$. O estimație admisibilă $h(stare)$ este $\min_varf(stare)$ pentru o stare nefinală și 0 pentru una finală.

3. Se dă următoarea topologie de rețea Bayesiană:



Care dintre următoarele afirmații sunt adevărate?

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

4. Se consideră următorul joc. Avem un grid cu 8 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 singură poziție 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?

- Numărul de simboluri ale lui MAX din care scădem numărul de simboluri ale lui MIN.
- Câte configurații de 3 simboluri are MIN din care scădem câte configurații de 3 simboluri are MAX.
- Numerotarea liniilor și coloanelor începe de la 0 din colțul stânga sus. Calculatorul joacă 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 L_m și C_m . O evaluare ar fi $NRLIN-LM-L_m$

- d. Numerotarea liniilor și coloanelor începe de la 0 din colțul stânga sus. Calculatorul joacă 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 L_m și C_m . O evaluare ar fi $LM+L_m$ (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

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 rădăcină.

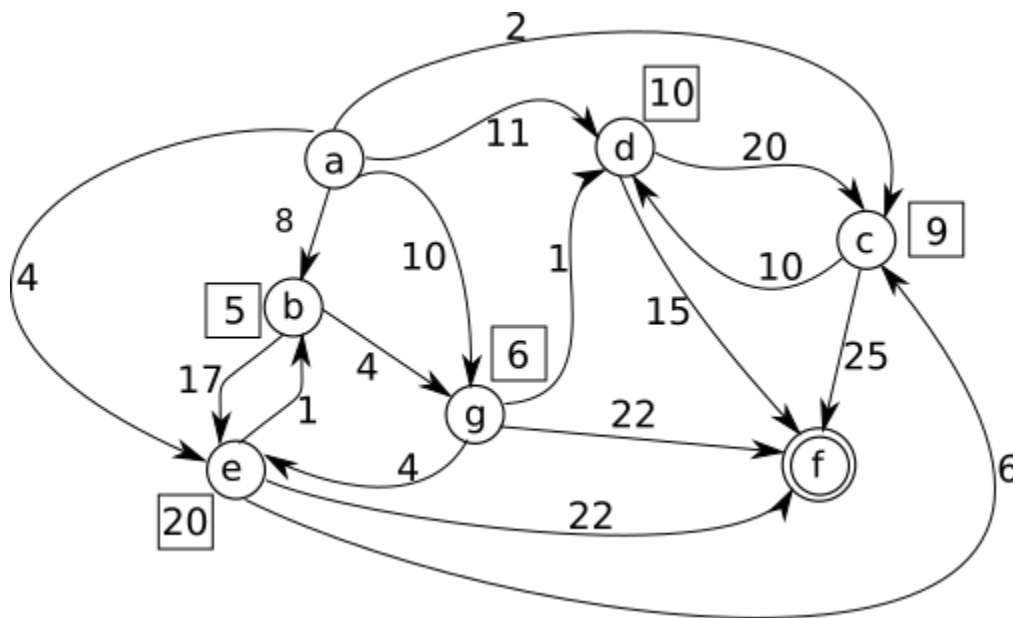
6. Considerăm următoarea problemă asemănătoare cu problema blocurilor. Avem un număr N de stive pe care avem așezate blocuri ce conțin **litere**. Atunci când mutăm un bloc b_1 peste un bloc b_2 , dacă sub blocul b_2 e un bloc b_3 care are aceeași literă ca și b_1 , atunci blocul b_2 dispăre și blocul b_1 ajunge direct plasat peste b_3 (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 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 .

- Căutăm care este litera (notată cu Lit) cu număr maxim de apariții relativ la toată configurația. Numărăm câte blocuri sunt în configurație cu litera diferită de Lit și considerăm estimația egală cu acest număr.
- Căutăm care este litera (notată cu Lit) cu număr minim de apariții relativ la toată configurația. Numărăm câte blocuri sunt în configurație cu litera diferită de Lit și considerăm estimația egală cu acest număr.
- Considerăm estimația ca fiind numărul de stive cu litere diferite pe ele.
- 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).

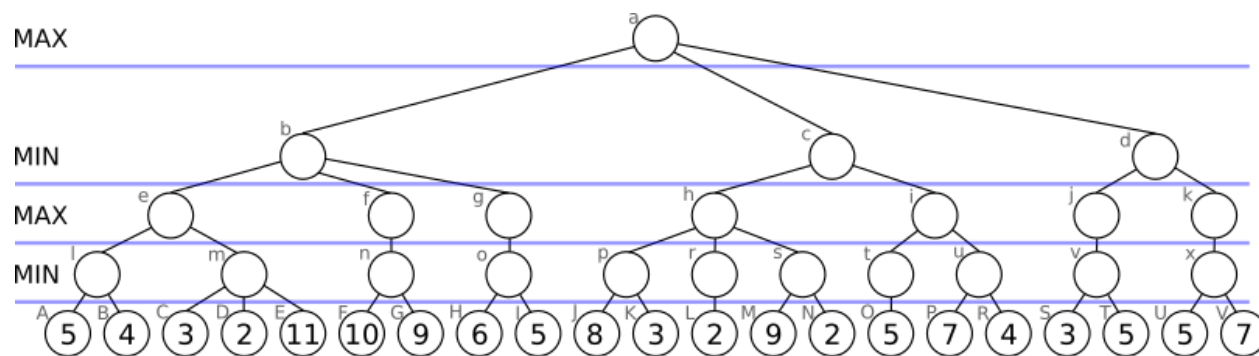


- a
- b
- c
- d
- e
- f
- g

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

- Nodurile scop sunt pe altă componentă conexă a grafului față de nodul de start.
- Se poate ajunge la un nod scop doar trecând printr-o muchie cu cost foarte mare în graf.
- Condiția scop este imposibilă pentru problema dată.
- Nodul scop este succesor direct al nodului de start.
- 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?

- Numărul de retezări efectuate de Alpha-Beta este 5 (edit: am corectat) (am considerat o rețezare ca fiind necalcularea unui întreg subarbore, nu se contorizează și nodurile individuale din subarboarele tăiat).
- Dacă am aplica alpha-beta pe acest joc atunci nodurile D și E nu ar mai fi evaluate
- Valoare minimax a nodului f este 10.
- Pentru a calcula valoarea nodului b este suficient să calculăm minimul dintre valorile minimax înscrise în nodurile de la A la I (*i mare*).
- 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?

- 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 $1!+2!+3!+4!+5!+6!+7!+8!+9!$ ($9 \text{ factorial} = 1*2*3*4*5*6*7*8*9$).

--	--	--

- 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 maximă 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. Dacă înlocuim 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 obținute 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, f aplicată fiecărui nod din OPEN.

- c. Un nod poate fi în același timp atât în 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 estimția $\hat{h}(a)$, dar pentru orice alt nod n , estimția îndeplinește condiția $\hat{h}(n) \leq h(n)$ atunci soluția returnată de A^* e în mod cert drumul de cost minim de la a la un nod scop.

12. Se consideră un grid $N \times N$ care fiind o hartă pătratică în care sunt **NROZ** obiecte roz, notate cu r și un obiect albastru, notat cu a . Numărul N este natural $5 \leq N \leq 10$. 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 poziția 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
- $NR \cdot 5$ unde NR e numărul de obiecte roz pe care le are asupra lui.

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

Notății:

- 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 $coloana(roz[i])$. Numărul total de obiecte roz este **NROZ**.
- $distMH(linie1, coloana1, linie2, coloana2) = |linie1 - linie2| + |coloana1 - coloana2|$ (distanța Manhattan)
- \sqrt{x} = rădăcina pătrată a numărului x .

Care dintre estimările 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: opțiunea a este corectă din aceleași motive ca și opțiunea b. Am uitat să marchez inițial răspunsul când 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. $h(stare) = \min(100, K \cdot 5)$ pentru o stare nefinală și $h(stare) = 0$ pentru o stare finală.

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