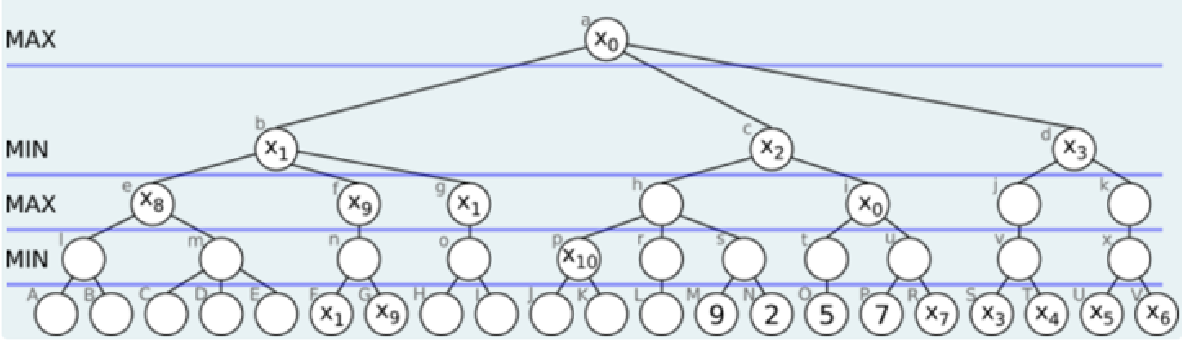
⏱ **20 minute**

Laborator 7
(seria 24) (150 puncte)

Se consideră cunoscute enunțurile pentru cele 4 probleme studiate la laborator

1

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



- [ ] x3 este sigur mai mic sau egal cu x4, x5, x6 ✓
- [ ] x0 aparține intervalului [5,7] ✓
- [ ] x2 este sigur egal cu x0
- [ ] x1 este sigur egal cu x9 ✓
- [ ] Dacă aplicăm algoritmul Alpha-Beta asupra acestui arbore și x8 este mai mare decât x9, atunci nodul g (g-literă mică) va fi retezat
- [ ] Dacă x10 e mai mic decât 9, nodul N (N - literă mare) va fi retezat.

2

Se consideră următoarea problemă de căutare:
Avem un grid de dimensiune N x N, cu N natural, N > 3. În grid sunt C cutii colorate. În total există K culori, numerotate cu numere de la 1 la K.
Putem muta doar câte o cutie pe rând. **Cutia se poate deplasa doar cu o poziție** în direcțiile sus, jos, stânga, dreapta și **doar pe o poziție liberă care nu are ca vecină imediată (pe linie sau pe coloană) o cutie de altă culoare față de cutia mutată.** Această regulă se aplică inclusiv stării inițiale în are nu putem avea cutii de culori diferite imediat vecine pe linie sau coloană.
Costul pe o mutare e egal cu 1.
Două cutii se consideră vecine dacă sunt fie pe aceeași linie dar pe coloane consecutive, fie pe aceeași coloană, dar pe linii consecutive.
Scopul este să grupăm cutiile de culori identice, astfel încât pentru orice culoare cl, dacă numărul de cutii de culoare cl este mai mare decât 1, orice cutie de culoare cl să fie vecină (pe linie/coloană) cu altă cutie de culoare cl.
Aveți exemple de stare inițială și finală la https://drive.google.com/drive/u/0/folders/1eLBRBusjat7ZrOETtGmKoaFJI_M096EX
Care dintre afirmațiile de mai jos sunt adevărate?
(15 puncte)

- [ ] Pentru K=2, dacă avem N*(N-1)/2 cutii de culoare 1, dar strict mai mult de (N-2)*(N-1) cutii de culoare 2, atunci problema sigur nu are soluție, indiferent de configurația principală. ✓
- [ ] Pentru N=5 numărul maxim de culori pentru ca problema să fie posibilă pentru măcar o configurație (să existe o stare inițială și o stare finală validă cu un număr de mutări natural nu neapărat nenul care duce de la starea inițială la cea finală) este 13. ✓
- [ ] Numărul de succesori (valizi) ai fiecărei stări e suma numărului de locuri vecine (pe linie sau coloană) libere din jurul fiecărei cutii.
- [ ] Notăm dist_culoare(cl)=min({mulţimea distanțelor Manhattan dintre toate plăcuțele de culoare cl, nu neaparat distincte }). O estimație admisibilă $\hat{h}$(nod)=(suma valorilor dist_culoare(cl) pentru toate culorile cl). ✓
- [ ] O estimație admisibilă $\hat{h}$(nod)=max({mulțimea distanțelor euclidiene dintre plăcuțele distincte de aceeași culoare}).

3

Se consideră problema blocurilor, cu următoare modificări:
Pot exista mai multe blocuri cu aceeași informație (de exemplu, mai multe blocuri cu litera "a").
Un bloc oarecare **se poate muta** în două moduri:
- fie se mută de pe un vârf de stivă pe altul
- fie, dacă blocul se află la nivelul n și stiva din dreapta ori din stânga are înălțime n-1, atunci blocul poate glisa în vârful stivei vecine,și toate blocurile care erau deasupra lui cad cu o poziție pe stiva respectivă.

Pentru ca o stare să fie considerată **scop** orice stivă din stare trebuie să conțină blocuri cu aceeași informație (de exemplu toate blocurile din stivă să aibă informația "a") și niciun bloc cu informația egală cu cele de pe stiva respectivă să nu se mai găsească pe alte stive (de exemplu, să nu fie două stive cu informația "a").
Costul pe o mutare este 1.
Notații:
- S = numărul de stive
- SV(stare) = numărul de stive vide dintr-o stare
- N numărul de blocuri
- NR_INF - numărul de informații distincte din configurație (de exemplu dacă litera "a" apare de 3 ori în configurație, pentru ea se adună doar 1 în NR_INF).

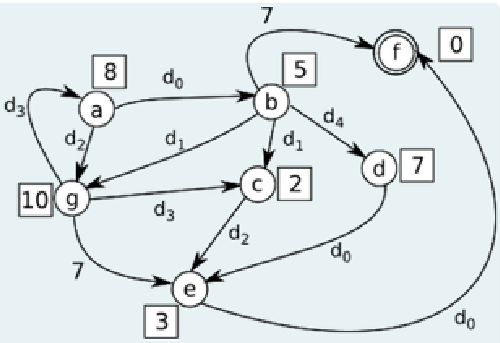
Care din afirmațiile de mai jos sunt adevărate? (15 puncte)

- [ ] Dacă NR_INF>S atunci problema nu are soluție ✓
- [ ] Pentru o stare în care orice stivă are proprietatea că stiva din dreapta are înălțimea strict mai mare decât a ei, numărul de succesori este (S-SV(stare)+1)*(S-1) ✓
- [ ] Notăm cu nr_st(info)numărul de stive distincte în care se găsesc blocuri cu informația info. O estimație admisibilă este ĥ(stare)=max(nr_st(info) pentru orice informație din configurație)-1. ✓
- [ ] Notăm cu max(info) numărul maxim de blocuri cu aceeași infornație din configurație. Notăm cu h_max înălțimea celei mai înalte stive. O estimație admisibilă este abs(h_max- max(info)), unde abs(x) e valoarea absolută a lui x. ✓
- [ ] Pentru o stare inițială în care problema are soluție, numărul total de stări scop posibile este NR_INF! (factorial). Două stări se consideră distincte dacă există o poziție i astfel încât informațiile de pe stiva de pe acea poziție, din prima stare diferă ca ordine și/sau conținut față de informațiile de pe stiva de pe poziția i din cealaltă stare.

4

Se dă graful orientat cu arce ponderate din imagine. Pentru unele arce ponderile nu sunt precizate, fiind înlocuite de identificatorii di. Euristica dată este una admisibilă (estimația pentru fiecare nod e trecută în pătrățelul de lângă nod).
Costurile arcelor sunt numere naturale nenule.
Drumul de cost minim returnat de A* pentru nodul de start a este a->b->c->e->f cu costul 10.

Care dintre afirmațiile de mai jos sunt adevărate:
(15 puncte)



- [ ] d0 are valoarea 3
- [ ] d1 este mai mare strict decât 1
- [ ] d2 este 1 ✓
- [ ] d0 are valoare mai mare sau egala cu 3 ✓
- [ ] d3 este 8
- [ ] pentru nodul de start a, nu mai există un drum în graf de același cost cu drumul de cost minim (reformulare, nu există două drumuri de cost minim pentru nodul de start a) ✓

5

Se consideră următoarea problemă de căutare:
Nodurile grafului conțin numere naturale mai mari sau egale cu 10.
Avem **arc de la un nod n1 la un nod n2** dacă:
- **n2 se obține prin n1 prin inversarea a două cifre vecine**, caz in care **costul e egal cu maximul dintre cele două cifre inversate**. De exemplu, dacă n1=12483, dacă alegem cifrele 1 și 2, atunci putem avea arcul 12483->21483 (de cost 2). Există o singură excepție în generarea acestui tip de succesor, și anume nu e voie prin inversare să punem prima cifră a numărului 0. De exemplu pentru 1023 nu putem inversa 1 cu 0.
- **n2 se obține prin extragerea din n1 a două cifre de pe poziții consecutive, însumarea lor, iar dacă suma este în continuare o cifră, adăugarea cifrei în aceeași poziție de unde au fost extrase cele două cifre (dacă suma lor nu e tot o cifră, nu se poate obține un succesor în acest mod). Costul unei astfel de transformări este chiar suma.** De exemplu, dacă n1=12483, dacă alegem cifrele 1 și 2, atunci putem avea arcul 12483->3483 (de cost 3), dar pentru același număr nu putem alege 4 și 8. Nu putem aplica acest mod de obținere a succesorilor pentru numere de 2 cifre deoarece am ieși din domeniul valorilor posibile pentru noduri.

Scopul este să ajungem la numere care încep și se termină cu aceeași cifră, de exemplu, 1321.
Exemplu de drum valid (nu neaparat de cost minim): 151772 -> 16772->7772->7727 (costul este (5+1)+(1+6)+7 =20)

Care dintre afirmațiile de mai jos sunt adevărate? (15 puncte)

- [ ] Există minim 90 de noduri inițiale pentru care nu avem soluție ✓
- [ ] Nu există noduri inițiale de 3 cifre pentru care să nu avem soluție.
- [ ] Pentru orice număr cu minim 9 cifre nenule avem sigur soluție. ✓
- [ ] O estimație admisibilă ĥ(nod) este, pentru cazul în care avem 2 cifre egale în număr, ĥ(nod)=min(d1+d2) pentru toate cifrele c care au o dublură în număr, unde d1 este distanța (în poziții) de la capătul din stânga la cea mai apropiată instanță a lui c de el, iar d2 este distanța de la capătul din dreapta la cea mai apropiată instanță a lui c de el.
- [ ] O estimație admisibilă ĥ(nod) este, pentru cazul în care avem 2 cifre egale în număr, este ĥ(nod)=0 dacă nodul este final și ĥ(nod)= max(prima cifră și ultima cifră) ✓
- [ ] Pentru nodul de start 18494 drumul de cost minim are costul 16
- [ ] Pentru numarul 10010 singura stare scop în care poate ajunge e 10001

6

Un joc se desfășoară pe un grid N x N. Jucătorii joacă cu simbolurile x și 0. Jucătorul cu simbolul x mută primul. Mutarea constă în plasarea unui simbol pe tablă.
La fiecare **diagonală de 3 simboluri** S de același fel, jucătorul cu simbolul S primește **scorul 2**, iar **la fiecare linie, sau coloană cu 3 simboluri** de același fel, primește **un scor de 1**. Simbolurile din configurația câștigătoare dispar. Dacă printr-o mutare se obțin mai multe configurații câștigătoare se adună toate scorurile pentru ele și dispar toate simbolurile implicate.

Care dintre variantele de mai jos oferă o funcție de evaluare care să arate în mod corect cât de favorabilă este o stare pentru calculator (MAX), cu alte cuvinte să aibă o valoare mai mare pentru stări mai favorabile și mai mică pentru stări mai nefavorabile pentru **orice stare intermediară (nefinală)** a jocului aflată la adâncimea maximă în arborele minimax? (15 puncte)

- [ ] Numărul de coloane cu simbol majoritar ale lui MAX - numărul de coloane cu simbol majoritar ale lui MIN
- [ ] scorul lui MAX de până acum din care scădem scorul lui MIN de până acum ✓
- [ ] numărul de linii deschise ale lui MAX - numărul de linii deschise ale lui MIN. O linie deschisă e un set de 3 căsuțe vecine pe rând, coloană sau diagonală care conțin doar simboluri ale jucătorului pentru care se calculează linia deschisă ✓
- [ ] numărul de simboluri izolate (fără vecini) ale lui MIN+numărul de simboluri izolate (fără vecini) ale lui MAX
- [ ] numărul de randuri și coloane deschise ale lui MAX /numărul de randuri și coloane deschise ale lui MIN+(numărul de diagonale deschise ale lui MAX) /(numărul de diagonale deschise ale lui MIN) . O linie deschisă e un set de 3 căsuțe vecine pe rând, coloană sau diagonală care conțin doar simboluri ale jucătorului pentru care se calculează linia deschisă

7

Se consideră problema blocurilor modificată astfel:
Fiecare bloc are asociată o literă. Literele se pot repeta (pot exista 2 blocuri cu aceeași informație).
Mutări posibile:

- Putem muta un bloc din vârful unei stive doar în vârful unei alte stive.
- Putem lua (șterge) un bloc din configurație.

Pe lângă configurația inițială se dă și o listă de cuvinte.
Vom considera **costul pe mutarea unui bloc** ca fiind numărul de ordine al stivei pe care este mutat.
Stivele se numerotează de la 1, pornind de la stânga spre dreapta.
Eliminarea unui bloc costă 1.

Scopul este ca prin mutările blocurilor să se ajungă ca fiecare dintre aceste cuvinte să reprezinte conținutul stivelor, citite de jos în sus, iar în rest să nu existe blocuri neîncadrate în cuvinte.

Se dă **o configurație inițială cu blocurile amestecate și o listă de șiruri LS**.
De exemplu pentru starea inițială din folderul: https://drive.google.com/drive/u/0/folders/1ynotW8u6WiLmFHUKOXUk0TmtA5zUP8oh
și LS conținând șirurile "mac", "bau" și "ba" putem avea ca stare finală posibilă cealaltă imagine din folder (atenție, nu e singurul caz posibil de stare finală pentru acest exemplu).

Vom nota cu **prefix(stiva,sir)**=numarul de caractere din prefixul comun al stivei și al cuvantului (de exemplu daca stiva este "mih" și cuvântul este "miau", atunci prefix("mih,"miau")=2.
Vom nota cu **lungime(sir)** lungimea șirului, respectiv lungime(stiva) înălțimea stivei
Vom nota cu **stive[i]** stiva de pe poziția i, unde cel mai mic i posibil este 1.

Care dintre următoarele afirmații sunt adevărate?
(15 puncte)

- [ ] Pentru orice stare inițială împreună cu orice set de șiruri există soluție
- [ ] O estimație admisibilă ĥ(nod) este numărul de blocuri din configurația din nodul curent din care scădem lungimile adunate ale șirurilor. ✓
- [ ] Consideram sirMax ca fiind șirul de lungime maximă. Fie np=prefix(stive[1],sirMax) atunci estimația ĥ(nod)=lungime(sirMax)-np este admisibilă
- [ ] Consideram sirMin ca fiind șirul de lungime minimă. Fie np=prefix(stive[1],sirMin) atunci estimația ĥ(nod)=lungime(sirMin)-np este admisibilă
- [ ] Notăm cu NS numărul de șiruri din LS, și cu NGATA numărul de șiruri (din LS) deja formate în stive (un șir e format într-o stivă dacă stiva conține acel șir). Starea este fără blocuri suplimentare). Notăm NR=NS-NGATA. O estimație admisibilă ĥ(nod) este NR*(NR+1)/2 ✓

8

Pentru problema X și 0, care dintre următoarele afirmații sunt adevărate? Considerăm # simbolul pt loc liber (15 puncte)

- [ ] Pentru starea inițială a tablei de joc, reprezentată prin [['#','#','#'],['#','#','#'],['#','#','#']], 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).
- [ ] O stare finală a jocului este ori una în care a câștigat MAX ori una în care a câștigat MIN.
- [ ] Arborele minimax pentru X și 0 întotdeauna va avea un număr de niveluri mai mic sau egal cu 10. ✓
- [ ] Pentru starea curentă [['x','#','#'],['0','x','0'],['x','#','0']] (rădăcină a arborelui curent minimax), presupunând că simbolul calculatorului este X, arborele minimax de adâncime maximă 3 (în care 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. ✓
- [ ] 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.

9

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. (15 puncte)

- [ ] $\hat{h}$1(nod)=$\hat{h}$(nod)/2
- [ ] $\hat{h}$1(nod)=$\hat{h}$(nod)*2
- [ ] $\hat{h}$1(nod)=$\hat{h}$(nod)+2, dacă nod nu e nod scop și $\hat{h}$1(nod)=0 dacă nod este scop
- [ ] $\hat{h}$1(nod)=$\hat{h}$(nod)^3 (cu sensul de ridicare la puterea a 3-a)
- [ ] niciuna dintre formulele de la celelalte subpuncte ✓

10

Care dintre afirmațiile de mai jos sunt adevărate? (15 puncte)

- [ ] 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. ✓
- [ ] Metoda înlănțuirii înapoi presupune o căutare a răspunsului pornind de la date în încercarea de a găsi scopul.
- [ ] Un sistem expert este întotdeauna proiectat pentru a răspunde la întrebări în general din absolut orice domeniu (atotștiutor).
- [ ] Un sistem expert poate funcționa și răspunde corect și fără o bază de cunoștințe.