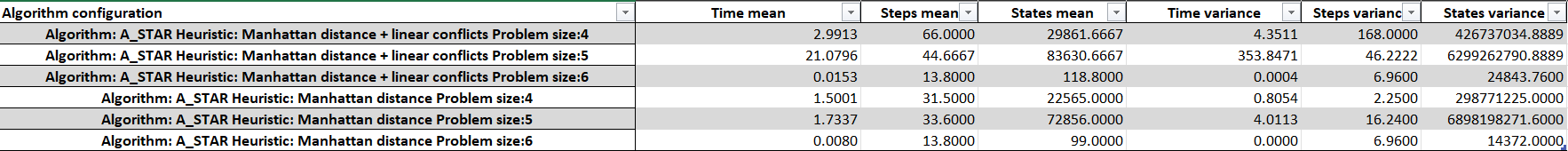
Tema 1 – IA

Student: Prioteasa Cristi Andrei - Grupa: 341 C4

**Nota**: *In toate datele generate am folosit termenul de conflicte liniare incorect, referindu-ma, de fapt la numarul de inversiuni ( pozitiile i,j din Npuzzle pt care i < j => tile[i] > tile[j] ).*

**Beam Search pentru jocul N-Puzzle**

**Algoritmul A star**



Graphical user interface

Description automatically generated with medium confidence

Fiind o cautare exhaustiva in spatiul starilor, ghidata dupa o euristica, limita de memorie impusa per problem size este depasita pentru o parte din stele 4-easy pentru euristica Manhattan distance + 2 \* no. inversiuni si Manhattan distance simplu, analog pentru problem size 5.

Pentru a pastra consistenta rezultatelor, in cazul in care A\* depaseste memoria alocata per problem size, acesta intoarce esec si nu o solutie partiala.

Numarul de stari pastrate in memorie este mare chiar si pentru problem size 4 (aprox 20 000 – 30 000).

In cazul testului pe problem size 5 cu suma de euristici, varianta timpului este foarte mare deoarece algoritmul gaseste o solutie doar pe 3 din cele 5 teste.

Varianta starilor tinute in memorie este foarte mare pentru fiecare tip de problema deoarece arborele creste exponential ( cu factorul de ramificare B ), deci si starile tinute in memorie vor creste exponential.

Manhattan distance + numarul de inversiuni nu reprezinta o euristica admisibila, dar obtine rezultate bune pe unele teste ( pt 4-easy reuseste sa gaseasca o solutie optima pentru 60% din teste vs 40% pt Manhattan distance)

**Algoritmul Beam Search**

Pentru algoritmul Beam search rezultatele pentru diferitele configuratii ale parametrilor ( beam, euristica, problem-size) pot fi observate in tabelul urmator. Liniile marcate cu rosu sunt instante ale algoritmului in care s-a atins limita de memorie impusa.

Se observa ca euristica Manhattan Distance obtine in medie timpi de rulare mai mici decat Manhattan + no. inversiuni pentru oricare valori ale beam-ului.

**Nota**: *Datele pot fi consultate mai usor in datasheetul atasat. Celulele in care se afla 0, reprezinta de fapt un numar foarte mic de forma 10^-6 pe care excel nu il reprezinta corect.*

A picture containing table

Description automatically generated

A picture containing table

Description automatically generated

Deoarece Beam Search incearca diferite cai, media timpului de rulare pana la gasirea unei solutii este mult mai mic decat la A\* (e.g: 2.99 s pe 4-easy versus 0.08 s). Si media numarului de stari tinute in memorie este mai mic, deorece in dictionarul de vizitat se adauga la un moment dat maxim B \* 4 vecini noi.

Timpul de rulare creste o data cu dimensiunea beam-ului si dimensiunea problemei, asemenea numarul de stari tinute in memorie. Varianta timpulu si varianta lungimii caii gasite la beam search tinde catre varianta lungimii caii gasite in cazul algoritmului A\* (respectiv varianta timpului ) cu cat beam-ul este mai mare. Acest lucru are sens pentru ca daca Beam -> infinit, atunci Beam search devine un A\*.

Algoritmul A\* gaseste solutia optima, dar ocupa multe stari in memorie si ia un timp mai indelungat. Algoritmul Beam Search cu un Beam mic ( 100 – 500 ) face un compromis intre optimalitatea solutiei si viteza de cautare / numarul de stari tinute in memorie.

De exemplu pentru problema 4 easy folosind euristica Manhattan distance + no inversiuni, A\* obtine o cale optime de 66 pasi, pe cand Beam search gaseste solutii de lungime 1384 cu beam 10, 332 cu Beam 50 si 193 cu Beam 100. Estimez ca pentru un Beam 1000 se va gasi o solutie cu un numar de pasi comparabil cu cel al A\*, dar folosind mai putine resurse si un timp mai scurt de rulare. In cele mai multe cazuri o solutie suboptima, dar apropiata de cea reala cu o marja de eroare epsilon este suficienta pentru majoritatea aplicatiilor practice, deci un Beam Search ar fi de preferat.

**Generalized Limited Discrepancy Search pentru jocul N-puzzle**

Am decis sa implementez cazul de baza, cand discrepancy = 0 in mod iterativ. Chiar si in acest mod, GLDS nu gaseste solutii in cel putin 50% din testele propuse cu oricare din cele doua euristici. Se observa totusi ca euristica Manhattan Distance + no. Inversions gaseste cu 13% mai multe solutii decat Manhattan distance si intr-un timp mai bun, desi pe testele problemei 4-normal, Manhattan Distance gaseste solutii intr-un timp de 100x mai bun ca Manhattan distance + no. Inversions.

**Chart, bar chart

Description automatically generated**

**Problem size: 4-easy**

**Manhattan Distance + no. inversions**

**Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated**

Evident timpul de rulare pentru GLDS este mult mai mare decat variantele cu Beam search, chiar si pt beam mare. Numarul de pasi pana la solutie este mai mare decat la Beam search. ( facand algoritmul iterativ, numarul de pasi a fost numarat intre discrepante incercate). Numarul de stari tinute in memorie >> numarul de stari tinute in memorie de catre orice varianta de Beam Search (GLDS iterativ).

**Manhattan Distance**

Box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Numarul de pasi pana la solutie, asemanator cu cel de la un Beam search cu b = 10, este totusi mai mare ca cel de la GLDS folosind euristica Manhattan Distance + no.inversiuni. Timpul de rulare este de peste 100x mai mare ca la Beam Search, iar numarul de stari tinute in memorie este extrem de mare.

**Nota**: *am ales sa impun limita 1 000 000 pe stari pentru orice problem size, altfel nu ar fi trecut nici macar un test cu euristica mea*

**Problem size: 6-easy**

**Manhattan Distance + no. inversions**

Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

**Manhattan Distance**

Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated

In cazul 6-easy fiind teste usoare, rezultatele sunt comparabile in privinta numarului de pasi pana la solutie, dar timpul de executie si numarul de stari este cu acelasi ordin de marime mai mare ca la testele precedente fata de Beam Search.

**Problem size: 4-normal**

**Manhattan Distance + no.inversions**

Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Din forma box-ului se observa ca timpul de rulare pentru majoritatea testelor 4-normal cu GLDS este in jur de 35 secunde, crescand apoi cu dificulteatea testelor. Timpul de rulare pentru toate instantele prezente de Beam Search este sub 5 secunde. Numarul de pasi pana la gasirea solutiei este doar mai mic ca un Beam Search cu b = 10, care probabil ia o cale gresita si o expandeaza pana ajunge la o solutie, fiind foarte lunga. Numarul de stari tinute in memorie (GLDS iterativ) este in medie 250 000, cu multe ordine de marime peste Beam Search.

**Manhattan Distance**

Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated

Chart, box and whisker chart

Description automatically generated

In afara de timpul de rulare mult mai mic ca GLDS folosind euristica Manhattan Distance + no. Inversiuni, numarul de pasi este mult mai mic iar in medie se tin 150 000 stari in memorie, deci cu 40% mai putin.

**Table

Description automatically generated with medium confidence**

**Beam Search cu Limited Discrepancy Backtracking (BLDS) pentru jocurile N-puzzle si Turnurile din Hanoi**

**Rezultate NPuzzle**

**Table

Description automatically generated**

**Table

Description automatically generated**

**Chart, bar chart

Description automatically generated**

**Nota**: *Restul graficelor pot fi generate cu ajutorul scheletului de cod pentru euristica si configuratia dorita. Am inclus aici doar graficele pt BLDS vs Beam Search cu Manhattan Distance si B = [50,100]*

**Problem size 4 easy with Manhattan Distance**

Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated

**Problem size 5 easy with Manhattan Distance**

Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated

**Problem size 6 easy with Manhattan Distance**

Chart, box and whisker chart

Description automatically generated Chart, box and whisker chart

Description automatically generated

**Chart, box and whisker chart

Description automatically generated**

**Problem size 4 normal with Manhattan Distance**

**Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated**Chart, box and whisker chart

Description automatically generated

**Problem size 5 normal with Manhattan Distance**

**Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated**

**Chart, box and whisker chart

Description automatically generated**

**Problem size 6 normal with Manhattan Distance**

**Chart, box and whisker chart

Description automatically generatedChart, box and whisker chart

Description automatically generated**Chart, box and whisker chart

Description automatically generated

Concluzie: *Pentru testele 4-easy, 5-easy,6-easy, 4-normal, 5-normal rezultatele obtinute te Beam Search si BLDS sunt comparabile. Pentru problema 6-normal se poate observa ca BLDS performeaza mult mai bine, cu un timp de cautare mult mai bun si mult mai putine stari in memorie decat Beam Search pentru acelasi beam.*

**Turnurile din Hanoi**

Implementarea jocului Turnurile din Hanoi se bazeaza pe formalismul explicat aici [1]. <https://aries.ektf.hu/~gkusper/ArtificialIntelligence_LectureNotes.v.1.0.4.pdf>

Euristica folosita reprezinta numarul de misplaced tiles.

**Nota**: *Liniile marcate cu rosu reprezinta instante ale problemei in care s-a atins limita maxima de memorie impusa.*

Table

Description automatically generatedTable

Description automatically generatedTable

Description automatically generatedTable

Description automatically generated

Se poate observa ca pentru valori mici ale beam-ului se gaseste o solutie suboptima doar pentru instantele problemei cu numarul de discuri <= 9, iar pentru B = 500,1000 se gaseste o solutie in memoria alocata si pentru N = 10.

Nu se gaseste solutia optima, cu numarul de pasi 2nr\_discuri -1 ( formula care sper ca se pastreaza si daca numarul de stive este egal cu 4) pentru nicio dimensiune a beam-ului pentru probleme cu un numar de discuri mai mare ca 6.