Meno: Jakub Hlavačka
AIS ID: 96876

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Ilkovičova 2, 842 16 Bratislava 4

2.Zadanie - Hlavolam dokumentácia

Študijný program: Informatika
Ročník: 2
Cvičenie: Ut 14:00 -1.30b
Cvičiaci: Ing. Matúš Pikuliak
Predmet: Umelá inteligencia
Akademický rok: 2019/2020

Meno: Jakub Hlavačka
AIS ID: 96876

Definovanie problému:

Našou úlohou je nájsť riešenie 8-hlavolamu. Hlavolam je zložený z 8 očíslovaných políčok a jedného prázdneho miesta. Políčka je možné presúvať hore, dole, vľavo alebo vpravo, ale len ak je tým smerom medzera. Je vždy daná nejaká východisková a nejaká cieľová pozícia a je potrebné nájsť postupnosť krokov, ktoré vedú z jednej pozície do druhej.

Znenie úlohy:

Použite algoritmus lačného hľadania, porovnajte výsledky heuristík 1. a 2.

Použité heuristiky:

1.Heuristika - Počet políčok, ktoré nie sú na svojom mieste, vrátane prázdneho políčka

2.Heuristika - Súčet vzdialeností jednotlivých políčok od ich cieľovej pozície, vrátane prázdneho políčka

Stručný opis riešenia:

Program funguje na princípe lačného algoritmu, takže vyberá vždy lokálne najlepšie riešenie, zadaní problém dokáže riešiť dvomi rôznymi heuristikami. Celý algoritmus je riadený pomocou haldy, z ktorej vždy popujem uzol, ktorý je najmenej rozdielny od koncového. Na uzol, ktorý som vytiahol z haldy vyskúšam vykonať všetky štyri operácie DOLE, HORE, VPRAVO, VLAVO. V danom poradí a iba ak je to možné a daná operácie nebude zasahovať mimo mapu. Keď nad mapou vykonám operáciu, tak z novo vytvorenej mapy vytvorím string, ktorý zahashujem a skontrolujem, či sa nenachádza v hash sete(python fičúra, niečo ako hash tabulka). Ak sa tam ešte nenachádza, tak ho tam pridám a vytvorím uzol, ktorý vhodím do haldy. Vďaka tomuto nedochádza k opakovanému vkladaniu prvok do haldy, ktoré sa tam už nachádzajú.

Algoritmus pokračuje, pokým nie je uzol popnutý z haldy identický s koncovým alebo kým nie je halda prázdna, čo signalizuje, že zadaná úloha nemá riešenie. Program najprv vyrieši úlohu pre prvú heuristiku a následne druhú. Na konci vypíše operácie, ktorými sa dostal z počiatočného uzla do koncového, celkový počet operácií, počet spracovaných uzlov(tie, ktoré vytiahol z haldy) a počet vygenerovaných uzlov(tie, ktoré vložil do haldy).

Reprezentácia údajov:

Jednotlivé uzly, ktoré hádžem do haldy si reprezentujem classou StavMapy. V classe sa nachádzajú tieto atribúty:

- mapa – v Pythone dvojrozmerný list, ktorý mi reprezentuje ako vyzerá mapa pre daný uzol, ktorý bol vytiahnutý z haldy
- pocetRoznych – pre prvú heuristiku to je počet políčok, ktoré nie sú na svojom mieste a pre druhú heuristiku to je súčet vzdialeností jednotlivých políčok od svojho finálneho miesta

Meno: Jakub Hlavačka
AIS ID: 96876

- operácia – názov operácie, ktorú bolo treba vykonať na predošlú mapu, aby sme sa dostali do tohto stavu
- predosli – odkazuje na uzol, z ktorého bol vygenerovaný aktuálny

Konkrétny algoritmus:

Využil som algoritmus, ktorý sa pozerá na lokálne najlepšie hodnoty, preto sa môže často stať, že nenájde optimálne riešenie. Algoritmus, ktorý som použil je úplný, využíva prehľadávanie do šírky a jeho časová zložitosť je O(b^m), pričom **b** je vetviaci faktor a **m** je maximálna hĺbka listového uzla. Pamäťová zložitosť je tiež O(b^m).

Spôsob testovania:

Program som testoval na viacerých prípadoch vstupu, väčšina z nich boli veľkosti 3x3, ale tiež som testoval mapy o veľkosti 4x4, 3x4, 3x2, 2x3 a 2x2 aby som si potvrdil, že algoritmus funguje na rôzne veľkom vstupe.

Aby som si overil správnosť vykonávania operácií nad danou mapou som používal funkciu vypisMapa(mapa) a kontroloval výpisy operácií.

Nula na mape znázorňuje prázdne políčko.

1.vstup:

Počiatočný stav: 1 0

2 3

Koncový stav: 3 0

2 1

Pre takýto vstup neexistuje riešenie.

Výsledok z prvej heuristiky:

Pocet krokov: 0

Pocet spracovanych uzlov: 12

Pocet vygenerovanych uzlov: 11

Výsledok z druhej heuristiky:

Pocet krokov: 0

Pocet spracovanych uzlov: 12

Pocet vygenerovanych uzlov: 11

Meno: Jakub Hlavačka
AIS ID: 96876

2.vstup:

Počiatočný stav: 1 0

2 3

Koncový stav: 3 2

0 1

Pre takýto vstup použili obidve heuristiky rovnaký počet krokov a aj operácie, ktoré použili mali to isté poradie. Avšak druhá heuristika spracovala a vygenerovala o 2 uzly menej.

Výsledok z prvej heuristiky:

Pocet krokov: 6

Pocet spracovanych uzlov: 9

Pocet vygenerovanych uzlov: 9

Výsledok z druhej heuristiky:

Pocet krokov: 6

Pocet spracovanych uzlov: 7

Pocet vygenerovanych uzlov: 7

3.vstup:

Počiatočný stav: 1 0 4

2 3 5

Koncový stav: 3 2 5

0 1 4

Pre tento vstup neexistuje riešenie.

Výsledok z prvej heuristiky:

Pocet krokov: 0

Pocet spracovanych uzlov: 360

Pocet vygenerovanych uzlov: 359

Výsledok z druhej heuristiky:

Pocet krokov: 0

Pocet spracovanych uzlov: 360

Pocet vygenerovanych uzlov: 359

Meno: Jakub Hlavačka
AIS ID: 96876

4.vstup:

Počiatočný stav: 1 0 4

2 3 5

Koncový stav: 3 0 5

4 2 1

Pre tento vstup našla prvá heuristika rýchlejšie riešenie, ale vygenerovala a spracovala podstatne viac uzlov ako druhá pre tento rozmer mapy. Druhá heuristika začala operáciou HORE a prvá VLAVO.

Výsledok z prvej heuristiky:

Pocet krokov: 26

Pocet spracovanych uzlov: 139

Pocet vygenerovanych uzlov: 168

Výsledok z druhej heuristiky:

Pocet krokov: 30

Pocet spracovanych uzlov: 109

Pocet vygenerovanych uzlov: 143

5.vstup:

Počiatočný stav: 0 5

3 4

2 1

Koncový stav: 3 0

4 2

1 5

Pre tento vstup našla druhá heuristika omnoho lepšie riešenie a ešte aj vygenerovala a spracovala extrémne menej uzlov ako prvá. Obidve heuristiky začali operáciou HORE.

Výsledok z prvej heuristiky:

Pocet krokov: 49

Pocet spracovanych uzlov: 137

Pocet vygenerovanych uzlov: 168

Výsledok z druhej heuristiky:

Meno: Jakub Hlavačka
AIS ID: 96876

Pocet krokov: 11

Pocet spracovanych uzlov: 33

Pocet vygenerovanych uzlov: 44

6.vstup:

Počiatočný stav: 0 5

3 4

2 1

Koncový stav: 3 0

2 4

1 5

Pre tento vstup nevedia heuristiky nájsť riešenie, pretože neexistuje.

Výsledok z prvej heuristiky:

Pocet krokov: 0

Pocet spracovanych uzlov: 360

Pocet vygenerovanych uzlov: 359

Výsledok z druhej heuristiky:

Pocet krokov: 0

Pocet spracovanych uzlov: 360

Pocet vygenerovanych uzlov: 359

7.vstup:

Počiatočný stav: 1 3 6

4 8 0

7 5 2

Koncový stav: 1 2 3

4 5 6

7 8 0

Pre tento vstup našla prvá heuristika omnoho horšie riešenie, tiež začala najprv operáciou DOLE a druhá heuristika začala operáciou HORE.

Výsledok z prvej heuristiky:

Pocet krokov: 35

Meno: Jakub Hlavačka
AIS ID: 96876

Pocet spracovanych uzlov: 97

Pocet vygenerovanych uzlov: 162

Výsledok z druhej heuristiky:

Pocet krokov: 9

Pocet spracovanych uzlov: 12

Pocet vygenerovanych uzlov: 22

8.vstup:

Počiatočný stav: 1 2 3

0 4 5

6 7 8

Koncový stav: 1 2 3

4 5 6

7 8 0

Pre tento vstup dopadli obidve heuristiky rovnako. Avšak prvá spracovala a vygenerovala omnoho viac uzlov.

Výsledok z prvej heuristiky:

Pocet krokov: 27

Pocet spracovanych uzlov: 131

Pocet vygenerovanych uzlov: 220

Výsledok z druhej heuristiky:

Pocet krokov: 27

Pocet spracovanych uzlov: 54

Pocet vygenerovanych uzlov: 95

9.vstup:

Počiatočný stav: 1 2 5

3 4 6

7 8 0

Koncový stav: 1 2 3

4 5 6

7 8 0

Meno: Jakub Hlavačka
AIS ID: 96876

Pre tento vstup bola lepšia druhá heuristika a spracovala výrazne menej uzlov, tiež ich výrazne menej vygenerovala. Obidve heuristiky začali operáciou VPRAVO.

Výsledok z prvej heuristiky:

Pocet krokov: 76

Pocet spracovanych uzlov: 893

Pocet vygenerovanych uzlov: 1465

Výsledok z druhej heuristiky:

Pocet krokov: 62

Pocet spracovanych uzlov: 375

Pocet vygenerovanych uzlov: 620

10.vstup:

Počiatočný stav: 2 3 4

5 1 6

7 8 0

Koncový stav: 1 2 3

4 5 6

7 8 0

Prvá heuristika v tomto prípade vykonala skoro o 20 menej operácií, tiež vygenerovala a spracovala o niečo menej uzlov. Obidve heuristiky začali dvakrát operáciou DOLE.

Výsledok z prvej heuristiky:

Pocet krokov: 34

Pocet spracovanych uzlov: 271

Pocet vygenerovanych uzlov: 447

Výsledok z druhej heuristiky:

Pocet krokov: 52

Pocet spracovanych uzlov: 363

Pocet vygenerovanych uzlov: 593

Meno: Jakub Hlavačka
AIS ID: 96876

11.vstup:

Počiatočný stav: 1 2 7 6

0 4 5 3

9 8 10 11

Koncový stav: 7 0 1 4

2 3 11 6

10 9 8 5

Druhá heuristika bola extrémne lepšia ako prvá vo všetkých smeroch. Každá začala svoju postupnosť krokov inou operáciou.

Výsledok z prvej heuristiky:

Pocet krokov: 188

Pocet spracovanych uzlov: 1692

Pocet vygenerovanych uzlov: 3124

Výsledok z druhej heuristiky:

Pocet krokov: 82

Pocet spracovanych uzlov: 333

Pocet vygenerovanych uzlov: 647

12.vstup:

Počiatočný stav: 1 2 3 4

5 6 7 8

9 10 11 12

13 14 15 0

Koncový stav: 1 2 3 4

5 6 7 8

9 10 12 15

13 14 11 0

Keby sme chceli skúsiť, mapu 4x4, ktorá nemá riešenie, tak kým na to tento algoritmus príde to zoberie veľmi veľa času oproti 3x3, preto je toto riešenie jednoduché.

Pre tento vstup použili obidve heuristiky rovnaký počet operácií, aby sa dostali od počiatočného stavu ku koncovému, avšak prvá vygenerovala o niečo viac uzlov a o 1 viac spracovala.

Meno: Jakub Hlavačka
AIS ID: 96876

Výsledok z prvej heuristiky:

Pocet krokov: 4

Pocet spracovanych uzlov: 6

Pocet vygenerovanych uzlov: 11

Výsledok z druhej heuristiky:

Pocet krokov: 4

Pocet spracovanych uzlov: 5

Pocet vygenerovanych uzlov: 9

Výsledok testovania:

Druhá heuristika dopadla vo väčšine prípadov lepšie ako prvá, takže by mala poskytovať optimálnejšie riešenia. Avšak podľa môjho názoru je to diskutabilné, pretože to dosť závisí od rozloženia mapy.

Výhody a nevýhody implementácie:

Program je schopný riešiť rôzne veľkosti mapy, nie len 3x3, avšak mapu je treba zadať v zdrojovom kóde ako dvojrozmerný list. Nevýhodou je, že som na každú heuristiku použil inú haldu, teda program zaberá viac pamäte, ale som ušetril čas, pretože nemusím popovať zvyšné prvky z haldy, keď algoritmus skončí pre prvú heuristiku.

Implementácia závisí od programovacieho prostredia, pretože som použil knižnice ako **copy** a **heapq**, ktoré sú špecifické pre python. Zároveň som si jednotlivé uzly reprezentoval ako objekty.

Zhodnotenie riešenia a možné rozšírenie:

Pre daný algoritmus som sa snažil využiť čo najmenej pamäte, ale nie na úkor časovej zložitosti. Aby nedochádzalo v algoritme k opakovanému generovania uzlov, tak som využil set(python fičúra), ktorá tomu zabraňuje, tým pádom som ušetril veľa času a zároveň som predišiel možnému zacykleniu.

Program by sa dal rozšíriť o ďalšiu heuristiku, ktorá by zohľadňovala viacero prvkov a tým pádom by mohla byť optimálnejšie.

Ďalšie rozšírenia mi už nenapadajú vzhľadom na to, že som musel použiť lačné algoritmy, ktoré vždy berú lokálne najlepšie riešenie.

Použitie:

Program stačí jednoducho spustiť a všetko sa vykoná automaticky. Počiatočnú a koncovú mapu je potrebné meniť v zdrojovom kóde, avšak riešenie je optimalizované na rôzne veľkosti.
V textovom súbore vstupy.txt sú všetky mapy, ktoré som testoval.

Meno: Jakub Hlavačka
AIS ID: 96876