Peter Smreček, AIS ID: 103130
Utorok 16:00

Umelá inteligencia

Zadanie 2 – Problém 3 g)

Zadanie – Riešený problém

Úlohou je prejsť šachovnicu legálnymi ťahmi šachového koňa tak, aby každé políčko šachovnice bolo prejdené (navštívené) práve raz. Riešenie treba navrhnúť tak, aby bolo možné problém riešiť pre štvorcové šachovnice rôznych veľkostí (minimálne od veľkosti 5 x 5 do 20 x 20) a aby cestu po šachovnici bolo možné začať na ľubovoľnom východziom políčku.

V riešení je potrebné implementovať heuristiku (Warnsdorff's rule) do algoritmu prehľadávania stromu do hĺbky a pre šachovnicu 8x8 nájsť pre 10 rôznych východzích bodov jedno (prvé) správne riešenie (pre každý východzí bod). Algoritmus s heuristikou treba navrhnúť a implementovať tak, aby bol spustiteľný aj pre šachovnice iných rozmerov než 8x8.

Je odporúčané otestovať implementovaný algoritmus aj na šachovnici rozmerov 7x7, 9x9, prípadne 20x20 (máme úspešne odskúšaný aj rozmer 255x255) a prípadné zistené rozdiely v úspešnosti heuristiky analyzovať a diskutovať.

Opis riešenia

Pre riešenie som si zvolil implementáciu heuristiky do rekurzívneho prehľadávania stromu do hĺbky. Keďže ide o graf reprezentovaný šachovnicou, používal som implicitne reprezentovaný graf.

Kód riešenia aj s pomocnými funkciami sa nachádza v súbore `Smrecek_Zadanie2_Kod.py`.

Funkcia `pocetMoznostiPohybu` vráti počet možností pohybu z aktuálneho políčka na ešte nenavštívené políčka.

Funkcia `najdiSuradniceMoznostiPohybu` vráti pole súradníc dostupných a nenavštívených z aktuálneho políčka.

Funkcia `heuristika` vráti pole súradníc s minimálnym počtom nasledujúcich ťahov. Zo zadaných súradníc nájde možné legálne ťahy koňa na doposiaľ nenavštívené políčka a z nich vyberie tie, ktoré majú minimálny počet nasledujúcich políčok.

Funkcia `DFSrekurzia` je samotná funkcia hľadania do hĺbky. Hľadanie prebieha dovtedy, dokým nie je nájdená cesta, nie sú prehľadané všetky možnosti, alebo nie je prekročený maximálny počet krokov hľadania.

Peter Smreček, AIS ID: 103130
Utorok 16:00

Funkcia `riadicHladania` je riadiaca funkcia hľadania cesty koňa. Nastavuje globálne premenné na počiatočné hodnoty pred každým hľadaním, spúšťa rekurzívne hľadanie a vracia nájdenú cestu.

Kód rekurzívnej funkcie hľadania do hĺbky s heuristikou:

```
def DFSrekurzia(n, pociatocneX, pociatocneY, navstivene, cesta, maxKrokov):
    '''
    Rekurzivna funkcia hladania do hlbky. Hladanie prebieha dovtedy, dokym
nie je najdena cesta, nie su prehladane
    vsetky moznosti, alebo nie je prekroceny maximalny pocet krokov
hladania.

    :param n: rozmer sachovnice
    :param pociatocneX: sucasna suradnica x
    :param pociatocneY: sucasna suradnica y
    :param navstivene: pole booleanov obsahujuce informacie o navstivenych
policKach
    :param cesta: aktualne najdena cesta
    :param maxKrokov: maximalny pocet krokov hladania ktory sa nesmie
prekrocit
    :return: True ak je cesta najdena alebo je prekroceny pocet krokov,
False inak
    '''

    global pocitadlo
    pocitadlo += 1
    if pocitadlo == maxKrokov:
        # Ak je prektoceny maximalny pocet krokov, funkcia sa rekurzivne
vracia
        return True

    navstivene[pociatocneX][pociatocneY] = True
    # Sucasne policko kde sa nachadza kon je oznacene za navstivene

    cesta.append((pociatocneX, pociatocneY))
    # Sucasne policko sa pridava do pola reprezentujuceho sucasnu cestu
kona

    susedne = heuristika(pociatocneX, pociatocneY, n, navstivene)
    # Vytvorenie pola moznosti skoku podla heuristiky

    if len(cesta) == n ** 2:
        # Ak bola cesta najdena cela, funkcia sa rekurzivne vracia
        # Globalna premenna s najdenou cestou je nastavena na sucasnu cestu
        global navrat
        navrat = cesta
        return True

    for policko in susedne:
        # Cyklus prehladavania heuristicky susednych policok (policok,
ktore maju rovnaky pocet nasledujucich tahov)
        if not navstivene[policko[0]][policko[1]]:
            # Kontrola, ci policko este nebolo navstivene, pretoze hoci
pole susednosti zahrnalo len nenavstivene
            # policka v case jeho vytvarania, stav ich navstivenia sa mohol
v rekurzii zmenit
            if DFSrekurzia(n, policko[0], policko[1], navstivene, cesta,
maxKrokov):
```

```
                # Rekurzivne volanie funkcie prehladavania do hlbky. Ak sa
vratila hodnota True, funkcia sa rekurzivne vracia
                return True

    cesta.pop()
    navstivene[pociatocneX][pociatocneY] = False
    # Sucasne policko bolo prehladane a nevyhovuje. Odobera sa z cesty a
nastavuje sa ako neprehladane. Funkcia vracia False
    return False
```

Myšlienka riešenia:

Na začiatku každého rekurzívneho volania funkcie načítam a inkrementujem hodnotu globálnej premennej určujúcej počet krokov algoritmu. Ak je počet krokov prekročený, funkcia vráti True, čím spustí rekurzívny návrat. Ak počet krokov prekročený nie je, súčasné políčko označím za navštívené, pridám ho do stacku reprezentujúceho cestu koňa po šachovnici a vytvorím zoznam susednosti súčasného políčka na základe heuristiky. Heuristika z doposiaľ nenavštívených políčok vyberie tie, z ktorých existuje najmenší počet možností pohybu. Takáto možnosť existovať nemusí, môže existovať iba jedna alebo môže existovať aj viac políčok, z ktorých sa dá dostať na rovnaké množstvo políčok. Heuristika vráti prázdne pole ak neexistuje nasledujúci ťah, pole o dĺžke 1 ak existuje práve jeden nasledujúci ťah alebo pole o dĺžke 2-8 ak existuje viacero pohybov koňa z ktorých existuje rovnaké množstvo nasledujúcich ťahov.

Následne sa skontroluje dĺžka stacku. Ak je stack plný, teda sa v ňom nachádzajú všetky políčka šachovnice v poradí ich navštívenia, funkcia skopíruje do globálneho poľa súčasnú cestu, vráti True a rekurzívne sa vracia. Ak cesta ešte nie je kompletná tak sa v cykle pre každé políčko zo zoznamu susednosti skontroluje, či je naozaj nenavštívené a ak je nenavštívené, tak sa rekurzívne zavolá funkcia prehľadávania s týmto políčkom. Ak niektoré z nasledujúcich volaní vráti True, tak aj teraz sa vráti True a funkcia sa rekurzívne vracia, pretože buď bola cesta nájdená, alebo cesta neexistuje, alebo sa prekročil maximálny počet krokov. Ak ale rekurzívne volaná funkcia vráti False, pokračuje sa v cykle s prehľadávaním ďalšieho susedného políčka. Ak sa prehľadajú všetky susedné políčka súčasného políčka, cyklus končí neúspechom a preto sa súčasné políčko vyberie zo stacku reprezentujúceho cestu šachovnicou a nastaví na neprehľadané. Funkcia vráti False, čo reprezentuje, že cesta nebola nájdená a buď sa prehľadáva šachovnica iným smerom, alebo ak sú všetky možnosti prehľadané, hľadanie definitívne končí.

Túto funkcia je prvýkrát volaná funkciou `riadicHladania`, ktorá nastaví hodnoty globálnych premenných, stacku aj poľa reprezentujúceho navštívené políčka na úvodné hodnoty a po dokončení rekurzívneho hľadania vráti cestu uloženú v globálnej premennej.

Reprezentácia údajov problému

Keďže ide o pohyb koňa po šachovnici a z každého políčka existuje 2 – 8 možností pohybu, rozhodol som sa, že zvolím implicitnú reprezentáciu grafu v algoritme. Keďže je implicitne reprezentovaný graf, implicitne reprezentovaný je aj uzol grafu. Pre konkrétne súradnice sa generuje pole dostupných nenavštívených políčok šachovnice, ktoré sa v cykle

postupne prehľadávajú. Keďže ide o rekurzívnu funkciu, poradie prehľadaných uzlov v stacku zabezpečuje možnosť vrátiť sa späť.

To, či je vrchol navštívený, alebo nie, ukladám do 2D poľa booleanov, kde sa na pozícií podľa súradníc nachádza True, ak je vrchol navštívený, False, ak ešte navštívený nebol. Toto pole reprezentuje Stav.

Operátory sú reprezentované funkciami v súbore `Smrecek_Zadanie2_Pohyby.py`. Prvých 8 funkcií kontroluje možnosť navštíviť dané políčko, tj. či naň kôň môže skočiť. Políčko musí byť na šachovnici a nenavštívené. Ďalších 8 funkcií v tomto súbore vracajú súradnice skoku, ak je skok možné zrealizovať.

Heuristická funkcia bola zostrojená na základe Warnsdorff's rule. Heuristika z doposiaľ nenavštívených políčok vyberie tie, z ktorých existuje najmenší počet možností pohybu.

Cestu koňa šachovnicou ukladám do poľa, ktoré reprezentuje stack. Práve prezerané políčko pridám do stacku ako tuple súradníc X a Y. Ak z tohto políčka neexistuje správne riešenie tak ho zo stacku vyberiem a pokračujem ďalším políčkom. Ak stack obsahuje všetky políčka šachovnice, skopírujem ho do globálnej premennej reprezentujúcej cestu koňa po šachovnici.

Testovanie a výsledky experimentov

Pre potreby testovania som vytvoril niekoľko funkcií. Všetky sa nachádzajú v súbore `Smrecek_Zadanie2_Testovanie.py`.

Funkcia `vytvorSachovnicu` vygeneruje graf so zoznamom susednosti používaný na kontrolu správnosti riešenia.

Funkcia `vizualizuj` vykreslí šachovnicu zo zadaného poľa navštívených súradníc.

Funkcia `skontroluj` skontroluje, či je šachovnica vyplnená iba legálnymi heuristickými ťahmi koňa a vypíše informáciu, či je šachovnica kompletná.

Funkcia `existuje` oznámi, či očakávame, že heuristika z týchto súradníc na šachovnici nájde cestu.

Funkcia `najdiAvypis` je funkcia výpisu, ktorá nad jednou cestou zavolá funkcie na jej kontrolu a vizualizáciu.

Funkcia `generujVstupy` generuje požadovaný počet unikátnych súradníc na mape pre potreby kontroly.

Funkcia `otestujVsetky` je testovacia funkcia, ktorá otestuje, či existuje heuristická cesta pre koňa zo všetkých súradníc mapy.

Funkcia `main` obsahuje používateľské rozhranie a menu programu pre potreby jeho ovládania.

Peter Smreček, AIS ID: 103130
Utorok 16:00

Šachovnica 8 x 8 - analýza:

Podľa zadania mám nájsť prvé riešenie problému pre 10 súradníc. Zvolil som si preto náhodné pole súradníc [(5, 7), (1, 6), (0, 2), (4, 5), (5, 6), (7, 2), (7, 3), (6, 5), (2, 7), (0, 7)].

Pre párne mapy riešenie existuje z každého políčka. Pre mapu tejto veľkosti heuristika nájde mnohokrát riešenie na prvý pokus. Ako maximálne množstvo krokov pre jedno hľadanie cesty som nastavil 20000. Keďže šachovnica 8 x 8 je explicitne požadovaná v zadaní na otestovanie, uvádzam aj výstup vzorového vstupu.

Šachovnica 8 x 8 - výstup programu:

```
Zvolte 0 pre ukoncenie programu

Zvolte 1 pre testovanie nahodnych vstupov

Zvolte 2 pre testovanie konkretneho vstupu

Zvolte 3 pre testovanie vsetkych vstupov pre interval rozmerov

Zvolte 4 pre vzorovy test

4

Spustil sa vzorovy test pre 10 vstupov pre mapu velkosti 8 x 8

Boli zvolene suradnice [(5, 7), (1, 6), (0, 2), (4, 5), (5, 6), (7,
2), (7, 3), (6, 5), (2, 7), (0, 7)]

------------------------------------------------------------------

Vypis 1

Sachovnica 8x8 s pociatocnymi bodmi 5, 7:

Pre parne sachovnice riesenie existuje

### | 000 001 002 003 004 005 006 007

-------------------------------------

000 | 045 008 027 024 041 010 029 014

001 | 026 023 044 009 028 013 042 011

002 | 007 046 025 040 043 064 015 030

003 | 022 039 058 055 048 031 012 063

004 | 053 006 047 032 059 056 049 016

005 | 038 021 054 057 050 033 062 001

006 | 005 052 019 036 003 060 017 034

007 | 020 037 004 051 018 035 002 061

Sachovnica je heuristicky vyplnena
```

```
Sachovnica je vyplnena legalnymi tahmi
Sachovnica je kompletna
----------------------------------------------------------------------
Vypis 2
Sachovnica 8x8 s pociatocnymi bodmi 1, 6:
Pre parne sachovnice riesenie existuje
### | 000 001 002 003 004 005 006 007
-----------------------------------
000 | 010 017 048 031 012 015 050 029
001 | 047 020 011 016 049 030 001 014
002 | 018 009 054 063 032 013 028 051
003 | 021 046 019 042 055 052 033 002
004 | 008 039 064 053 062 041 056 027
005 | 045 022 043 040 057 060 003 034
006 | 038 007 024 061 036 005 026 059
007 | 023 044 037 006 025 058 035 004
Sachovnica je heuristicky vyplnena
Sachovnica je vyplnena legalnymi tahmi
Sachovnica je kompletna
----------------------------------------------------------------------
Vypis 3
Sachovnica 8x8 s pociatocnymi bodmi 0, 2:
Pre parne sachovnice riesenie existuje
### | 000 001 002 003 004 005 006 007
-----------------------------------
000 | 041 032 001 016 049 030 011 014
001 | 002 017 042 031 012 015 050 029
002 | 043 040 033 048 045 052 013 010
003 | 018 003 044 053 034 047 028 051
004 | 039 056 035 046 061 054 009 024
```

```
005 | 004 019 060 055 036 025 062 027
006 | 057 038 021 006 059 064 023 008
007 | 020 005 058 037 022 007 026 063
Sachovnica je heuristicky vyplnena
Sachovnica je vyplnena legalnymi tahmi
Sachovnica je kompletna
------------------------------------------------------------------------
Vypis 4
Sachovnica 8x8 s pociatocnymi bodmi 4, 5:
Pre parne sachovnice riesenie existuje
### | 000 001 002 003 004 005 006 007
-----------------------------------
000 | 060 009 064 029 058 011 050 027
001 | 041 030 059 010 063 028 015 012
002 | 008 061 040 057 014 049 026 051
003 | 039 042 031 062 053 056 013 016
004 | 032 007 054 043 048 001 052 025
005 | 035 038 033 022 055 044 017 002
006 | 006 021 036 047 004 019 024 045
007 | 037 034 005 020 023 046 003 018
Sachovnica je heuristicky vyplnena
Sachovnica je vyplnena legalnymi tahmi
Sachovnica je kompletna
------------------------------------------------------------------------
Vypis 5
Sachovnica 8x8 s pociatocnymi bodmi 5, 6:
Pre parne sachovnice riesenie existuje
### | 000 001 002 003 004 005 006 007
-----------------------------------
000 | 008 011 026 055 038 013 028 057
```

```
001 | 025 048 009 012 027 056 037 014
002 | 010 007 060 039 054 035 058 029
003 | 047 024 049 064 059 040 015 036
004 | 006 061 046 053 034 063 030 041
005 | 023 050 021 062 045 042 001 016
006 | 020 005 052 033 018 003 044 031
007 | 051 022 019 004 043 032 017 002
Sachovnica je heuristicky vyplnena
Sachovnica je vyplnena legalnymi tahmi
Sachovnica je kompletna
------------------------------------------------------------------
Vypis 6
Sachovnica 8x8 s pociatocnymi bodmi 7, 2:
Pre parne sachovnice riesenie existuje
### | 000 001 002 003 004 005 006 007
-----------------------------------
000 | 058 005 052 041 064 007 050 039
001 | 045 026 059 006 051 040 063 008
002 | 004 057 046 053 042 061 038 049
003 | 025 044 027 060 047 054 009 062
004 | 018 003 056 043 028 035 048 037
005 | 021 024 019 034 055 032 013 010
006 | 002 017 022 029 012 015 036 031
007 | 023 020 001 016 033 030 011 014
Sachovnica je heuristicky vyplnena
Sachovnica je vyplnena legalnymi tahmi
Sachovnica je kompletna
------------------------------------------------------------------
Vypis 7
Sachovnica 8x8 s pociatocnymi bodmi 7, 3:
```

Peter Smreček, AIS ID: 103130
Utorok 16:00

```
Pre parne sachovnice riesenie existuje
### | 000 001 002 003 004 005 006 007
-------------------------------------
000 | 005 008 023 038 057 010 025 028
001 | 022 037 006 009 024 027 058 011
002 | 007 004 039 056 045 060 029 026
003 | 036 021 044 041 052 055 012 059
004 | 003 040 051 046 061 042 053 030
005 | 020 035 018 043 054 047 062 013
006 | 017 002 033 050 015 064 031 048
007 | 034 019 016 001 032 049 014 063
Sachovnica je heuristicky vyplnena
Sachovnica je vyplnena legalnymi tahmi
Sachovnica je kompletna
----------------------------------------------------------------------
Vypis 8
Sachovnica 8x8 s pociatocnymi bodmi 6, 5:
Pre parne sachovnice riesenie existuje
### | 000 001 002 003 004 005 006 007
-------------------------------------
000 | 056 035 010 031 064 027 008 029
001 | 011 032 057 054 009 030 063 026
002 | 036 055 034 049 058 061 028 007
003 | 033 012 053 060 047 042 025 062
004 | 052 037 048 041 050 059 006 021
005 | 013 016 051 046 043 022 003 024
006 | 038 045 018 015 040 001 020 005
007 | 017 014 039 044 019 004 023 002
Sachovnica je heuristicky vyplnena
Sachovnica je vyplnena legalnymi tahmi
```

```
Sachovnica je kompletna
----------------------------------------------------------------------
Vypis 9
Sachovnica 8x8 s pociatocnymi bodmi 2, 7:
Pre parne sachovnice riesenie existuje
### | 000 001 002 003 004 005 006 007
-----------------------------------
000 | 020 049 004 053 018 059 002 063
001 | 005 052 019 060 003 064 017 058
002 | 050 021 048 043 054 057 062 001
003 | 041 006 051 036 061 044 055 016
004 | 022 035 042 047 056 031 012 045
005 | 007 040 025 034 037 046 015 030
006 | 026 023 038 009 028 013 032 011
007 | 039 008 027 024 033 010 029 014
Sachovnica je heuristicky vyplnena
Sachovnica je vyplnena legalnymi tahmi
Sachovnica je kompletna
----------------------------------------------------------------------
Vypis 10
Sachovnica 8x8 s pociatocnymi bodmi 0, 7:
Pre parne sachovnice riesenie existuje
### | 000 001 002 003 004 005 006 007
-----------------------------------
000 | 022 063 006 047 020 049 004 001
001 | 007 046 021 064 005 002 019 034
002 | 062 023 056 037 048 035 050 003
003 | 045 008 061 052 055 038 033 018
004 | 024 053 044 057 036 051 014 039
005 | 009 060 027 054 043 040 017 032
```

```
006 | 028 025 058 011 030 015 042 013

007 | 059 010 029 026 041 012 031 016

Sachovnica je heuristicky vyplnena

Sachovnica je vyplnena legalnymi tahmi

Sachovnica je kompletna

----------------------------------------------------------------
```

Vzorové riešenie našlo pre všetkých 10 bodov na šachovnici cestu koňa pomocou heuristiky. Všetky nájdené cesty boli heuristické, legálne vyplnené a plne vyplnené.

Šachovnica 7 x 7 – analýza:

Táto šachovnica s nepárnym rozmerom je heuristicky vyplniteľná len z niektorých políčok. Testom som zistil, že cesta existuje iba pre políčka jednej farby, teda ak je [0, 0] čierne políčko, tak riešenie existuje pre čierne políčka. Teda, cesta existuje vtedy, ak je súčet súradníc X a Y párny.

Pre šachovnicu 7 x 7 uvádzam len skrátený výstup popisujúci, či z daného políčka existuje cesta alebo nie. Ani pri 10 miliónoch možných krokov algoritmu nebolo nájdené riešenie pre políčka, ktorých súčet súradníc je nepárny.

Šachovnica 7 x 7 – výstup programu:

```
Zvolte 0 pre ukoncenie programu

Zvolte 1 pre testovanie nahodnych vstupov

Zvolte 2 pre testovanie konkretneho vstupu

Zvolte 3 pre testovanie vsetkych vstupov pre interval rozmerov

Zvolte 4 pre vzorovy test

3

Pre otestovanie existencie cesty pre vsetky suradnice daneho rozmeru
zadajte "pociatocneN koncoveN maximalnyPocetKrokovVtisicoch":

7 7 10000

Testujem rozmer 7 x 7

Zaciatok 00,00 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 00, 01 neexistuje alebo nebolo najdene
v zadanom pocte krokov

Zaciatok 00,02 - Sachovnica bola najdena a spravne vygenerovana
```

Pre n = 7 riesenie zacinajuce na 00, 03 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 00,04 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 00, 05 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 00,06 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 01, 00 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 01,01 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 01, 02 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 01,03 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 01, 04 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 01,05 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 01, 06 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 02,00 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 02, 01 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 02,02 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 02, 03 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 02,04 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 02, 05 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 02,06 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 03, 00 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 03,01 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 03, 02 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 03,03 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 03, 04 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 03,05 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 03, 06 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 04,00 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 04, 01 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 04,02 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 04, 03 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 04,04 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 04, 05 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 04,06 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 05, 00 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 05,01 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 05, 02 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 05,03 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 05, 04 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 05,05 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 05, 06 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 06,00 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 06, 01 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 06,02 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 7 riesenie zacinajuce na 06, 03 neexistuje alebo nebolo najdene v zadanom pocte krokov

Zaciatok 06,04 - Sachovnica bola najdena a spravne vygenerovana

```
Pre n = 7 riesenie zacinajuce na 06, 05 neexistuje alebo nebolo najdene
v zadanom pocte krokov

Zaciatok 06,06 - Sachovnica bola najdena a spravne vygenerovana


Pocet zlych je 0

Pocet neexistujucich rieseni je  24

Pocet existujucich rieseni je  25
```

Šachovnica 9 x 9 – analýza:

Ďalší nepárny rozmer šachovnice. Analýza je podobná ako pri rozmere 7 x 7.

Uvádzam časť skráteného výstupu pre tento rozmer.

Šachovnica 9 x 9 – časť výstupu programu:

```
Zvolte 0 pre ukoncenie programu

Zvolte 1 pre testovanie nahodnych vstupov

Zvolte 2 pre testovanie konkretneho vstupu

Zvolte 3 pre testovanie vsetkych vstupov pre interval rozmerov

Zvolte 4 pre vzorovy test

3

Pre otestovanie existencie cesty pre vsetky suradnice daneho rozmeru
zadajte "pociatocneN koncoveN maximalnyPocetKrokovVtisicoch":

9 9 20

Testujem rozmer 9 x 9

Zaciatok 00,00 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 9 riesenie zacinajuce na 00, 01 neexistuje alebo nebolo najdene
v zadanom pocte krokov

Zaciatok 00,02 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 9 riesenie zacinajuce na 00, 03 neexistuje alebo nebolo najdene
v zadanom pocte krokov

...

...

...
```

Peter Smreček, AIS ID: 103130
Utorok 16:00

```
Pre n = 9 riesenie zacinajuce na 08, 05 neexistuje alebo nebolo najdene
v zadanom pocte krokov

Zaciatok 08,06 - Sachovnica bola najdena a spravne vygenerovana

Pre n = 9 riesenie zacinajuce na 08, 07 neexistuje alebo nebolo najdene
v zadanom pocte krokov

Zaciatok 08,08 - Sachovnica bola najdena a spravne vygenerovana


Pocet zlych je 0

Pocet neexistujucich rieseni je  40

Pocet existujucich rieseni je  41
```

Zhodnotenie riešenia a dosiahnutých výsledkov

Riešenie je implementované rekurzívne a to mu uberá na efektivite. Riešenie ale počas testov ukázalo, že pre šachovnice od rozmeru 5 x 5 do rozmeru 17 x 17 vrátane a 19 x 19 potrebuje iba maximálne 20 000 krokov na nájdenie všetkých existujúcich riešení. Výhodou riešenia, že je jednoduché. Nevýhodou riešenia je, že oproti iteratívnej forme riešenia má rekurzívna forma oveľa väčšie pamäťové nároky.

Pre rozmery 18 x 18 a 20 x 20 vrátane potrebuje riešenie niekedy až 1 300 000 krokov na nájdenie všetkých riešení. Pri 18 x 18 hranicu 20 000 krokov prekračuje 5 možností začiatku: [(7, 5), (9, 6), (12, 5), (12, 16), (17, 1)], pri rozmere 20 x 20 hranicu 20 000 krokov prekračuje 1 možnosť začiatku a to (12, 16). Dá sa to vyriešiť zvýšením hranice na 1 300 000 a nájde riešenie pre všetky začiatky na šachovnici 18 x 18 a 20 x 20. V prípade takto veľkých šachovníc prudko narástol potrebný počet krokov na ich riešenie. Samotná heuristika už na optimalizáciu nestačí, bolo by nutné ju rozšíriť o pravidlá výberu nasledujúcich políčok, keď má viacero políčok zhodné hodnotenie, napríklad tak, že z políčok so zhodným hodnotením, by sa vybralo to, ktoré je najbližšie k okraju šachovnice.

Pri dostatočnom počte krokov pre párne rozmery šachovníc je riešenie vždy z každej súradnice nájduteľné. Pre nepárne rozmery šachovníc je riešenie nájduteľné len pre políčka, ktorých súčet súradníc je párny, tj. políčka jednej farby, ktorá sa na nepárnej šachovnici nachádza častejšie.

Pre párne šachovnice a dostatočný počet krokov je počet súradníc začiatku pre ktorý môj program nájde riešenie rovný n^2 a počet súradníc pre ktoré nenájde riešenie rovný 0.

Pre nepárne šachovnice a dostatočný počet krokov je počet súradníc začiatku pre ktorý môj program nájde riešenie rovný $\left\lceil \frac{n^2}{2} \right\rceil$ a počet súradníc pre ktoré nenájde riešenie rovný $\left\lfloor \frac{n^2}{2} \right\rfloor$.

Heuristika výrazne urýchľuje hľadanie. Bez heuristiky, teda len DFS algoritmus, má problém v časovom limite nájsť aj len jedno riešenie pre šachovnicu 8 x 8 s jedným počiatočným bodom. Pri použití DFS s heuristikou pre rozmery od 5 x 5 do 19 x 19 funguje

riešenie dostatočne rýchlo, kedy ani jedno hľadanie neprekročí maximum 10 sekúnd. Rozmer 20 x 20 je ale príliš veľký a konkrétne pre moje riešenie potrebuje hľadanie z počiatočného bodu [12, 16] až 1,3 milióna krokov. Ďalšie väčšie šachovnice, aj párne aj nepárne, už potrebujú na nájdenie riešení začínajúcich na niektorých konkrétnych políčkach veľmi veľa krokov a čas hľadania sa výrazne predlžuje. Preto sa dá tvrdiť, že čím menšia je mapa, tým je menej potrebných krokov na nájdenie riešenia. Čím väčšia je mapa, tým väčší je počet krokov na nájdenie riešenia. Pre mapy od 5 x 5 do 8 x 8 je možné v prijateľnom čase nájsť riešenie aj bez použitia heuristiky. Pre mapy do veľkosti 20 x 20 DFS a heuristika funguje relatívne dobre a efektívne. Pre väčšie mapy je potrebné vylepšiť heuristiku, aby fungovala optimálne aj na týchto mapách.

Používateľská príručka

Riešenie je implementované v programovacom jazyku Python verzie 3.8.3. Program sa skladá z 3 súborov `Smrecek_Zadanie2_Pohyby.py`, `Smrecek_Zadanie2_Kod.py` a `Smrecek_Zadanie2_Testovanie.py`. Program obsahuje main funkciu, tá sa spúšťa z `Smrecek_Zadanie2_Testovanie.py`.

Importujem iba `random` na generovanie náhodných čísel, inak je celý kód vlastný.

Po spustení programu sa objaví menu, v ktorom si používateľ vyberie možnosť, ktorou chce pokračovať:

Voľba 0: Ukončenie programu.

Voľba 1: Testovanie náhodných vstupov. Po tejto voľbe musí používateľ zadať 3 čísla oddelené 2 medzerami, reprezentujúce rozmer šachovnice, požadovaný počet náhodných vstupov, ktoré sa pre šachovnicu majú vygenerovať a maximálny počet krokov algoritmu v tisícoch. Vstup `5 10 20` znamená, že pre mapu veľkosti 5 x 5 sa vygeneruje 10 náhodných vstupov z ktorých sa každý otestuje maximálne do 20 000 spustení rekurzívneho algoritmu hľadania. Vypíše sa poradové číslo hľadania, vykreslia sa nájdené mapy a vypíšu sa hlásenia o správnosti/ nesprávnosti vygenerovanej mapy.

Voľba 2: Testovanie konkrétneho vstupu. Po tejto voľbe musí používateľ zadať 4 čísla oddelené 3 medzerami reprezentujúce rozmer šachovnice, súradnicu riadku, súradnicu stĺpca a maximálny počet krokov algoritmu v tisícoch. Vstup `7 2 4 20` znamená, že sa hľadá cesta koňa pre mapu veľkosti 7 x 7 a počiatočné súradnice 2, 4 s maximom 20 000 krokov algoritmu. Vypíše sa vizualizovaná cesta na šachovnici a vypíšu sa hlásenia o správnosti/ nesprávnosti vygenerovanej mapy.

Voľba 3: Testovanie všetkých vstupov pre interval rozmerov máp. Po tejto voľbe musí používateľ zadať 3 čísla oddelené 2 medzerami reprezentujúce rozmer prvej šachovnice, rozmer poslednej prehľadávanej šachovnice a maximálny počet krokov algoritmu v tisícoch. Vstup `5 10 20` znamená, že sa skontroluje existencia ciest zo všetkých políčok každej mapy od veľkosti 5 x 5 po veľkosť 10 x 10 vrátane, s maximálnym počtom 20 000 rekurzívnych spustení vyhľadávacieho algoritmu pre jednotlivé súradnice. Vypíšu sa hlásenia, o každom políčku všetkých šachovníc z intervalu, či sa z neho našla cesta, alebo nie. Vždy, po skontrolovaní

celého rozmeru a všetkých políčok jednej šachovnice sa vypíše, koľko bolo nájdených políčok z ktorých riešenie existuje a koľko bolo nájdených takých, z ktorých riešenie neexistuje alebo nebolo v danom počte krokov nájdené. Ak by program niekde zlyhal a vrátil nesprávnu mapu, táto mapa by sa vykreslila. Vzory výstupov tejto voľby sú uvedené pri testovaní rozmerov 7 x 7 a 9 x 9 v časti Testovanie a výsledky experimentov.

Voľba 4: Vzorový test. Táto voľba spustí a vypíše výpis voľby 1 pre konkrétnych 10 súradníc šachovnice rozmeru 8 x 8, na ktorých som testoval toto riešenie. Výpis z tohto bodu je uvedený v časti Testovanie a výsledky experimentov. Vybrané súradnice šachovnice 8 x 8: [(5, 7), (1, 6), (0, 2), (4, 5), (5, 6), (7, 2), (7, 3), (6, 5), (2, 7), (0, 7)].

Po zvolení voľby sa vykoná príslušná akcia a opäť sa vypíše menu s možnosťami voľby. Toto sa opakuje, až kým používateľ neukončí program zadaním voľby 0.