Croitoru Tudor-Alin, Grupa 233

Inteligență articială — Tema 1

Căutări

Punctul de intrare în program

Fișierul principal este `message.py`. Lansarea în execuție se face din linia comandă invocând executabilul pentru python3.8 urmat `message.py`. Lista de argumente posibile este următoarea:

- `--i – Calea directorului în care vor fi cautate fișierele de input. Toate fișierele din directorul dat vor fi parsate. Implicit: <directorul în care se află message.py>/inputs/`
- `--o – Calea directorului în care vor fi scrise fișierele de output. Fișierele cu același nume vor fi suprascrise. Implicit: <directorul în care se află message.py>/outputs/`
- `--t – Timpul de timeout în secunde. Implicit: 1`
- `--s – Numărul de soluții de calculat pentru fiecare fișier, pentru fiecare algortim. Implicit: 4`
- `--h – Afișează mesajul de ajutor.`

Exemplu de apelare:

```
$ python3 message.py ––i=.. --t=2 ––o=/home/user/seaches_outputs ––s=3
```

Această apelare va rula programul cautând inputul în directorul părinte apelării, cu timeout de 2 secunde, calculând câte 3 soluții și scriind output-urile în folderul `/home/user/searches_outputs`.
Toate argumentele sunt opționale.

Reprezentarea datelor

Funcția `parse(file)` din folderul `input_parser` se ocupă de parcurgerea fișierelor de intrare și de validarea lor. Reprezentarea posibiltății copiilor de a-și transmite unul altuia biletul este o matrice de adiacență în care `adiacenta[i][j]` reprezintă posibilitatea copilului de la indexul i

să ii transmită biletul copilului de la indexul j. Indecșii copiilor sunt pozițiile lor în vectorul de copii. Acesta este pur și simplu concatenarea tuturor rândurilor, menținand inclusiv pozițiile locurilor libere. Așa putem obține poziția în clasă din index și indexul din poziția în clasă în timp constant.

De asemenea, funcția returnează și starea inițială și cea finală în variabile diferite care conțin numele copiilor de început și, respectiv, sfârșit.

Euristici admisibile

1. **Distanța Euclidiană**
   Având în vedere că în clasă copiii sunt așezați într-o matrice, putem considera matricea de locuri drept un spațiu metric bidimensional. Atunci distanța Euclidiană va fi mereu mai mică sau egală cu drumul mesajului prin punctele în care se află copiii.
2. **Distanța Manhattan**
   Distanța Manhattan, într-o matrice în care toate mișcările de o poziție au distanța 1, este cel mai scurt drum, fiind distanța drumul calculat de BFS. Astfel, niciun drum prin elementele matricei nu poate fi mai scurt decât distanța Manhattan.
   În problema de aici, matricea e reprezentată de pozițiile copiilor în clasă. Deci orice drum ar fi folosit pentru a transmite biletul de la un copil la altul nu poate fi mai scurt decât distanța Manhattan dintre cei doi copii.

Euristica neadmisibilă

Formula este următoarea:
`h_neadmisibil(index_copil) = index_copil ** 2`

Demonstrație:
Euristica crește în funcție de poziția copilului în clasă și e complet independentă de orice distanță față de nodul scop. Astfel, există cazul în care un copil mai apropiat de nodul scop are euristica mai mare decât un copil mai apropiat.

De exemplu:
Fie 3 elevi în colțul din dreapta-jos al clasei, cu copilul scop fiind cel din colț (row · 6 + 5).
((row-1) · 6 + 4)
(row · 6 + 4) (row · 6 + 5)

$h\_neadmisibil((row-1) \cdot 6 + 4) = 36 \cdot row^2 - 24 \cdot row + 4$
$h\_neadmisibil(row \cdot 6 + 4) = 36 \cdot row^2 + 48 \cdot row + 16$

$h\_neadmisibil((row-1) \cdot 6 + 4) < h\_neadmisibil(row \cdot 6 + 4)$, deși copilul în diagonală față de copilul scop are cel puțin 2 mutări de făcut până la scop, iar cel din stânga are o singura mutare de făcut.

Validări și optimizări

1. **MalformedInputException**
   Această excepție este ridicată atunci când fișierul e prost formatat sau gol. Verificarea constă în faptul că există copii în clasă și există o stare inițială și una finală. Dacă fișierul nu este formatat corect, cel puțin una dintre aceste validări va eșua.
   Dacă fișierul de input nu respectă formatul, atunci algoritmii nu vor rula, iar programul va trece la următorul fișier.
2. **EarlyNoSolution**
   Excepția este ridicată atunci când nodul de copilul de la care pleacă biletul sau cel la care trebuie să ajungă nu pot cominca cu niciun dintre vecinii lor. Indiferent de cauză – locurile din jurul lor sunt libere, sau sunt supărați – în matricea de adiacență ei vor apărea fără muchii, deci nu ar putea transmite sau primi biletul.
3. **Pentru A* și IDA*, optimizarea pentru găsirea succesorilor**
   Funcția `conditie(indexCopil2)` verifică ca potențialul succesor sa aibă cel puțin o muchie în plus față de cea de întoarcere la copilul de la care a primit biletul. Astfel se evită copiii „izolați”.

Rezultate

Parametrii de rulare:
`--t=1`
Restul sunt valorile implicite (4 soluții, cale de input/output default)

	UCS	A*			
		Banală	Euclidiană	Manhattan	Neadmisibilă
input0.txt Soluție scurtă care nu blochează	Lungimi: 19, 21, 21, 21 Costuri: 18, 20, 20, 20 Timpi: 0.49, 0.49, 0.59, 0.59 (ms) Max noduri în mem: 167 Noduri procesate: 246	Lungimi: 19, 21, 21, 21 Costuri:18, 20, 20, 20 Timpi: 0.09, 019, 0.19, 0.19 (ms) Max noduri în mem: 141 Noduri procesate: 200	Lungimi: 19, 21, 21, 21 Costuri:18, 20, 20, 20 Timpi: 0.29, 0.49, 0.49, 0.49 (ms) Max noduri în mem: 132 Noduri procesate: 180	Lungimi: 19, 21, 21, 21 Costuri:18, 20, 20, 20 Timpi: 0.19, 0.29, 0.39, 0.39 (ms) Max noduri în mem: 141 Noduri procesate: 189	Lungimi: 19, 21, 23, 25 Costuri:18, 20, 22, 24 Timpi: 0.2, 0.3, 0.3, 0.3 (ms) Max noduri în mem: 136 Noduri procesate: 205
input1.txt Fără soluție	Nu există cale timp solutie: 0.0 (ms) noduri in memorie: 4 noduri procesate: 4	Nu exista cale timp solutie: 0.0(ms) noduri in memorie: 2 noduri procesate: 2	Nu exista cale timp solutie: 0.0 noduri in memorie: 2 noduri procesate: 2	Nu exista cale timp solutie: 0.0 noduri in memorie: 2 noduri procesate: 2	Nu exista cale timp solutie: 0.0 noduri in memorie: 2 noduri procesate: 2
input2.txt Starea finală este aceeași cu cea inițială	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 (ms) noduri in memorie: 7 noduri procesate: 7	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 (ms) noduri in memorie: 7 noduri procesate: 7	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 noduri in memorie: 7 noduri procesate: 7	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 noduri in memorie: 6 noduri procesate: 7	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 noduri in memorie: 6 noduri procesate: 7
input3.txt Blochează, dar măcar un algoritm returnează	UCS timed out	A* timed out	Lungimi: 20, 20, 20, 20 Costuri: 19, 19, 19, 19 Timpi: 7.1, 7.2, 7.3, 7.4 (ms) Max noduri în mem: 1957 Noduri procesate: 1957	Lungimi: 20, 20, 20, 20 Costuri: 19, 19, 19, 19 Timpi: 0.0, 0.09, 0.09, 0.09 (ms) Max noduri în mem: 114 Noduri procesate: 8227	A* timed out

	A* Optimizat			
	Banală	Euclidiană	Manhattan	Neadmisibilă
input0.txt Soluție scurtă care nu blochează	lungime: 19 cost: 18 timp solutie: 0.0 noduri in memorie: 29 noduri procesate: 37	lungime: 19 cost: 18 timp solutie: 0.09 noduri in memorie: 29 noduri procesate: 37	lungime: 19 cost: 18 timp solutie: 0.09 noduri in memorie: 29 noduri procesate: 37	lungime: 19 cost: 18 timp solutie: 0.09 noduri in memorie: 36 noduri procesate: 47
input1.txt Fără soluție	Nu exista cale timp solutie: 0.0(ms) noduri in memorie: 2 noduri procesate: 2	Nu exista cale timp solutie: 0.0 noduri in memorie: 2 noduri procesate: 2	Nu exista cale timp solutie: 0.0 noduri in memorie: 2 noduri procesate: 2	Nu exista cale timp solutie: 0.0 noduri in memorie: 2 noduri procesate: 2
input2.txt Starea finală este aceeași cu cea inițială	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 (ms) noduri in memorie: 1 noduri procesate: 1	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 noduri in memorie: 1 noduri procesate: 1	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 noduri in memorie: 1 noduri procesate: 1	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 noduri in memorie: 1 noduri procesate: 1
input3.txt Blochează, dar măcar un algoritm returnează	lungime: 20 cost: 19 timp solutie: 0.20 noduri in memorie: 88 noduri procesate: 49092	lungime: 20 cost: 19 timp solutie: 0.20 noduri in memorie: 88 noduri procesate: 143	lungime: 20 cost: 19 timp solutie: 0.09 noduri in memorie: 35 noduri procesate: 36	lungime: 24 cost: 23 timp solutie: 0.19 noduri in memorie: 88 noduri procesate: 140

	IDA*			
	Banală	Euclidiană	Manhattan	Neadmisibilă
input0.txt Soluție scurtă care nu blochează	Lungimi: 19, 21, 21, 21 Costuri: 18, 20, 20, 20 Timpi: 13.0, 15.99, 16.99, 16.99 (ms) Max noduri în mem: 30 Noduri procesate: 1349	Lungimi: 19, 21, 21, 21 Costuri: 18, 20, 20, 20 Timpi: 24.99, 43.99, 44.99, 46.00 (ms) Max noduri în mem: 30 Noduri procesate: 2665	Lungimi: 19, 21, 21, 21 Costuri: 18, 20, 20, 20 Timpi: 6.0, 6.99, 9.0, 9.0 (ms) Max noduri în mem: 30 Noduri procesate: 512	Nu exista drum timp solutie: 82.00 noduri in memorie: 41 noduri procesate: 4745
input1.txt Fără soluție	Nu exista cale timp solutie: 0.0(ms) noduri in memorie: 2 noduri procesate: 2	Nu exista cale timp solutie: 0.0 noduri in memorie: 2 noduri procesate: 3	Nu exista cale timp solutie: 0.0 noduri in memorie: 2 noduri procesate: 3	Nu exista cale timp solutie: 0.0 noduri in memorie: 2 noduri procesate: 3
input2.txt Starea finală este aceeași cu cea inițială	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 (ms) noduri in memorie: 5 noduri procesate: 7	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 noduri in memorie: 5 noduri procesate: 7	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 noduri in memorie: 5 noduri procesate: 7	copil-0-2 lungime: 1 cost: 0 timp solutie: 0.0 noduri in memorie: 5 noduri procesate: 16
input3.txt Blochează, dar măcar un algoritm returnează	IDA* timed out	IDA* timed out	IDA* timed out	IDA* timed out