МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Штучний інтелект

Лабораторна робота №1. Неінформативний пошук

Варіант №4

*Виконав студент гр. ІС-72*

*Гороховський І.О.*

Київ

НТУУ «КПІ»

2020

### 1.Мета

Ознайомитися з алгоритмами неінформативного пошуку та написати алгоритм для розв’язання логічної задачі, використовуючи один з алгоритмів неінформативного пошуку

### 2.Завдання

Розробити алгоритм вирішення старовинної логічної задачі: "Як за допомогою 5-ти літрового і 9-ти літрового відра набрати з річки 3 літри води?"

## Методи пошуку рішення - Пошук з обмеженням глибини

### Код

Знаходиться на платформі Github, містить 4 файла:

[bucket\_problem.py](https://github.com/doctorblinch/AI_Lab/blob/master/Uninformed search/bucket_problem.py" \o "bucket_problem.py) - містить основну логіку програми.

[main.py](https://github.com/doctorblinch/AI_Lab/blob/master/Uninformed search/main.py" \o "main.py) - запускає програму, виводить результати роботи в файл та у консоль.

[tests.py](https://github.com/doctorblinch/AI_Lab/blob/master/Uninformed search/tests.py" \o "tests.py) - містить тести на ключові функції програми.

[answer.txt](https://github.com/doctorblinch/AI_Lab/blob/master/Uninformed search/answer.txt" \o "answer.txt) - текстовий файл з відповіддю.

[https://github.com/doctorblinch/AI\_Lab/tree/master/Uninformed%20search](https://github.com/doctorblinch/AI_Lab/tree/master/Uninformed search)

### Результати роботи

|  |
| --- |
| Program found 21 solutions. |

|  |  |
| --- | --- |
|  | #################################################################################################### |
|  |  |
|  | 9 [Node(0, 0), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 10 [Node(0, 0), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 11 [Node(0, 0), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3), Node(3, 0)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 11 [Node(0, 0), Node(5, 0), Node(0, 5), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 11 [Node(0, 0), Node(5, 0), Node(5, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 12 [Node(0, 0), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3), Node(3, 0), Node(3, 9)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 12 [Node(0, 0), Node(5, 0), Node(0, 5), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 12 [Node(0, 0), Node(5, 0), Node(5, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 13 [Node(0, 0), Node(5, 0), Node(0, 5), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3), Node(3, 0)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 13 [Node(0, 0), Node(5, 0), Node(0, 5), Node(5, 5), Node(1, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 13 [Node(0, 0), Node(5, 0), Node(0, 5), Node(5, 5), Node(5, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 13 [Node(0, 0), Node(5, 0), Node(5, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3), Node(3, 0)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 14 [Node(0, 0), Node(5, 0), Node(0, 5), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3), Node(3, 0), Node(3, 9)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 14 [Node(0, 0), Node(5, 0), Node(0, 5), Node(5, 5), Node(1, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 14 [Node(0, 0), Node(5, 0), Node(0, 5), Node(5, 5), Node(1, 9), Node(5, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 14 [Node(0, 0), Node(5, 0), Node(0, 5), Node(5, 5), Node(5, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 14 [Node(0, 0), Node(5, 0), Node(5, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3), Node(3, 0), Node(3, 9)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 15 [Node(0, 0), Node(5, 0), Node(0, 5), Node(5, 5), Node(1, 9), Node(1, 0), Node(0, 1), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 15 [Node(0, 0), Node(5, 0), Node(0, 5), Node(5, 5), Node(1, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3), Node(3, 0)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 15 [Node(0, 0), Node(5, 0), Node(0, 5), Node(5, 5), Node(1, 9), Node(5, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3)] |
|  |  |
|  |  |
|  | #################################################################################################### |
|  |  |
|  | 15 [Node(0, 0), Node(5, 0), Node(0, 5), Node(5, 5), Node(5, 9), Node(0, 9), Node(5, 4), Node(0, 4), Node(4, 0), Node(4, 9), Node(5, 8), Node(0, 8), Node(5, 3), Node(0, 3), Node(3, 0)] |

### 5.Висновки

В результаті виконання лабораторної роботи було розроблено алгоритм для вирішення задачі, що включає генерацію графу та алгоритм пошуку з обмеженням глибини з урахуванням умов задачі. Програма видає шлях при успішному пошуку.