**TRÍ TUỆ NHÂN TẠO**

**TÌM ĐƯỜNG ĐI NGẮN NHẤT GIỮA HAI THÀNH PHỐ**

***GVHD: Huỳnh Xuân Phụng***

**Đỗ Thị Thanh Ngân; Đỗ Phạm Trúc Quỳnh; Bùi Mạc Tùng Lâm**

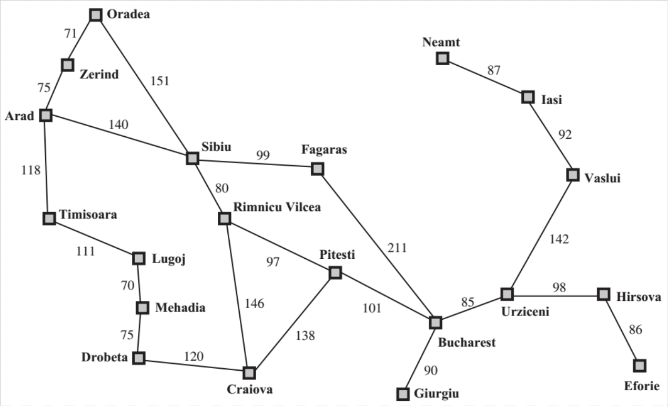
*1 Trường đại học Sư phạm Kỹ thuật TP.HCM*

**TÓM TẮT**

*Hiện nay, với sự phát triển của kinh tế nước ta, các tuyến đường giao thông ngày càng được mở rộng để phục vụ cho nhu cầu đi lại của con người. Tuy nhiên cũng gây khó khăn trong việc định hướng đường đi. Giả sử có 2 người, mỗi người ở 2 vị trí bất kỳ trên đồ thị. Mỗi bước di chuyển 2 người đồng thời đi đến một đỉnh bất kỳ liền kề với đỉnh hiện tại. Trong 1 bước di chuyển, 2 người cần phải đợi nhau để tiếp tục đi tiếp: người 1 đi đến 1 đỉnh thì phải đợi người 2 đi đến 1 đỉnh thì mới tiếp tục cho 2 người đi (có thể gọi điện để xác nhận), có rất nhiều tuyến đường có thể cho 2 người gặp nhau nhưng đâu là tuyến đường ngắn nhất và ít tốn chi phí nhất? Với nội dung trên,**nhóm em đã xây dựng một chương trình nhỏ để giúp cho chúng ta có thể tìm kiếm đường đi tối ưu nhất về chi phí và thời gian thông qua việc sử dụng các thuật toán, đặc biệt là thuật toán tìm kiếm Greedy Best First Search và A\*. Thời gian chạy thuật toán A\* nhanh hơn Greedy Best First Search.*

***Từ khóa***: ngắn nhất; Greedy Best First Search; A\*

1. **Đồ thị**



***Hình 1:*** *Đồ thị thành phố*

*Có tất cả 20 thành phố và chi phí đường đi giữa chúng.Chi phí từ thành phố Oradea đến thành phố Zerind.là 71. Chi phí từ thành phố Oradea đến thành phố Sibiu là 151. Chi phí từ thành phố Zerind đến thành phố Arad là 75. Chi phí từ thành phố Arad đến thành phố Sibiu là 140. Chi phí từ thành phố Arad đến thành phố Timisoara là 118. Chi phí từ thành phố Sibiuđến thành phố Fagaras là 99. Chi phí từ thành phố Sibiu đến thành phố RiminicuVilcea là 88. Chi phí từ thành phố Timisoara đến thành phố Lugoj là 111. Chi phí từ thành phố Lugoj đến thành phố Mehadia là 70. Chi phí từ thành phố Mehadia đến thành phố Drobeta là 75. Chi phí từ thành phố Drobeta đến thành phố Craivoa là 120. Chi phí từ thành phố Craivoa đến thành phố RiminicuVilcea là 146. Chi phí từ thành phố Craivoa đến thành phố Pitesti là 138. Chi phí từ thành phố RiminicuVilcea đến thành phố Pitesti là 97. Chí phí từ thành phố Pitesti đến thành phố Bucharest là 101. Chi phí từ thành phố Fagaras đến thành phố Bucharest là 211. Chi phí từ thành phố Bucharest đến thành phố Giurgiu là 90. Chi phí từ thành phố Bucharest đến thành phố Urziceni là 85. Chi phí từ thành phố Urziceni đến thành phố Hirsova là 98. Chi phí từ thành phố Urziceni đến thành phố Vaslui là 142. Chi phí từ thành phố Vaslui đến thành phố Iasi là 92. Chi phí từ thành phố Iasi đến thành phố Neamt là 87. Chi phí từ thành phố Hirsova đến thành phố Eforie là 86.*

1. **Các thuật toán được sử dụng**
   1. **Thuật toán Best First Search**

Sử dụng phương pháp đo lường cụ thể phần tốt nhất của một trạng thái để đạt đến đích nhanh nhất hoặc tìm thấy trạng thái đích mong đợi. Dùng hàng đợi ưu tiên.

1. **Mã giả**

**Best First Search Pseudo Code**

**-------------------------------------------**

Best First Search (Start, Goal\_test, f)

Insert(Start,Open);

Repeat forever {

If (empty(Open)) then return “fail”;

Remove “best” Node from Open using f;

Insert (Node, Close)

If (Goal\_test (Node)) then return Path Node to Initial State;

For each Child of Node do {

If (Child not in Close) then {

Child.previous := Node;

Insert (Child, Open) }}

}

1. **Độ phức tạp**: O()
   1. **Thuật toán A\***

A\* là giải thuật tìm kiếm trong đồ thị, tìm đường đi từ một từ một đỉnh bắt đầu đến đỉnh đích có sử dụng hàm ước lượng khoảng cách từ đỉnh hiện tại đến đỉnh đích để đánh giá khoảng cách từ trạng thái hiện tại đến trạng thái đích

1. **Mã giả**

**A\* Pseudo Code**

**--------------------------------------------------**

Let the openList equal empty list of nodes

Let the closedList equal empty list of nodes

Put the startNode on the openList (leave it’s f at zero)

While the openList is not empty

Let the currentNode equal the node with the least f value

Remove the currentNode from the openList

Add the currentNode to the closedList

If currentNode is the goal

You’ve found the end!

Let the children of the currentNode equal the adjacent nodes

For each child in the children

If child is in the closedList

Continue to beginning of for loop

child.g = currentNode.g + distance between child and current

child.h = distance from child to end

child.f = child.g + child.h

If child.position is in the openList’s nodes positions

If the child.g is higher than the openList node’s g

Continue to beginning of for loop

Add the child to the openList

1. **Độ phức tạp:** O()
2. **Ví dụ**

**\*Bảng biểu:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Người A** | **Người B** | **Thời gian đi người A** | **Thời gian đi người B** | **Thời gian chờ người A** | **Thời gian chờ người B** | **Sum** |
| Oradea | Arad | 0 | 0 | 0 | 0 | 0 |
| Zerimd | Sibiu | 71 | 140 | 69 | 0 | 280 |
| Zerind | Zerind | 71 | 75 | 4 | 0 | 150 |
| Zerind | Timisoara | 71 | 118 | 47 | 0 | 236 |

Giả sử người A bắt đầu từ thành phố Oredea và người B bắt đầu từ thành phố Arad. Người A và người B cùng xuất phát. Người A đi đến thành phố Zerind với thời gian là 71. Người B đi đến thành phố Zerind với thời gian là 75. Vậy thời gian người A chờ người B đi đến thành phố Zerind là 4, thời gian chờ của người B là 0.

1. **Kết luận**

Ta có thể tìm được đường đi ngắn nhất thông qua các thuật toán. Thuật toán A\* giải quyết vấn đề nhanh hơn thuật toán Greedy Best First Search vì có sử dụng thêm hàm đánh giá heuristic. Ứng dụng thực tiễn của thuật toán tìm đường đi ngắn nhất là Google map, Grab, giao hành nhanh,…Tuy nhiên nếu hai người ở hai thành phố kề nhau thì phải đi theo đường vòng ( Vì cùng xuất phát ).

**TÀI LIỆU THAM KHẢO**

1. Stuart J. Russell and Peter Norvig, Artificial Intelligence A Modern Approach, Pearson Education, Third Edition 2010.
2. <https://www.geeksforgeeks.org/a-search-algorithm/>
3. <https://www.geeksforgeeks.org/shortest-path-unweighted-graph/>
4. <https://www.geeksforgeeks.org/best-first-search-informed-search/>
5. <https://slideplayer.com/slide/11573804/>
6. <https://www.edureka.co/blog/a-search-algorithm/?fbclid=IwAR0LcDcJhtne5vihjyB_qHTLhiDL4-3887Sb9ciEO0vrLcfiQu8mUEBgAO8>