KHOA CÔNG NGHỆ THÔNG TIN

**BỘ MÔN KHOA HỌC MÁY TÍNH**



**BÀI TẬP LỚN**

**HỌC PHẦN: TRÍ TUỆ NHÂN TẠO**

**ĐỀ TÀI: XÂY DỰNG PHẦN MỀM TRÒ CHƠI PLUZZ (9 ô, 8 số)**

**SỬ DỤNG 2 GIẢI THUẬT TÌM KIẾM (BFS và A\*)**

**DEMO VÀ SO SÁNH KẾT QUẢ**

***Nhóm sinh viên thực hiên:***

|  |  |  |
| --- | --- | --- |
| **STT** | **Họ tên** | **Mã SV** |
| **1** | **Lưu Văn Hưng** | **621093** |
| **2** | **Trần Văn Thắng** | **621079** |
| **3** | **Trần Thanh Bình** | **621057** |

**Giáo viên hướng dẫn: TS.Phan Thị Thu Hồng**

**MỤC LỤC**

[**LỜI MỞ ĐẦU** 3](#_Toc26614208)

[**GIỚI THIỆU BÀI TOÁN** 4](#_Toc26614209)

[**I. GIẢI THẬT BFS** 5](#_Toc26614210)

[**a**.Thuật toán 6](#_Toc26614211)

[**II. GIẢI THUẬT A\*** 8](#_Toc26614212)

[**a**.Thuật toán 11](#_Toc26614213)

[**DEMO** 13](#_Toc26614214)

[a. Giao diện chương trình 13](#_Toc26614215)

[b. Giải thuật BFS 14](#_Toc26614216)

[**c**. Giải thuật A\* 15](#_Toc26614217)

[**ĐÁNH GIÁ** 16](#_Toc26614218)

# **LỜI MỞ ĐẦU**

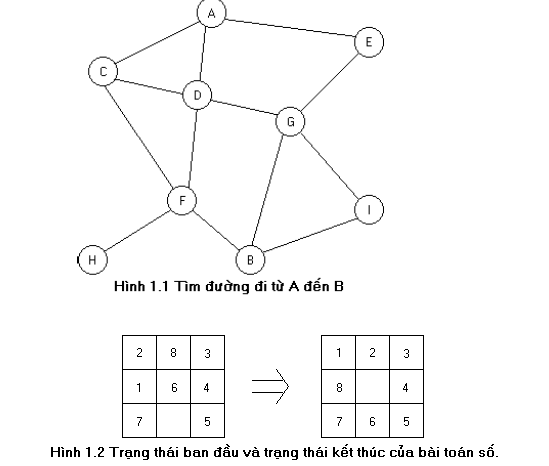
Trí tuệ nhân tạo là một ngành khoa học máy tính\_nghiên cứu xử lý thông tinbằng máy tính, do đó trí tuệ nhân tạo đặt ra mục tiêu nghiên cứu: làm thế nào thể hiện được các hành vi thông minh bằng thuật toán, rồi nghiên cứu các phương pháp cài đặt chương trình có thể thực hiện được các hành vi thông minh bằng thuật toán, tiếp theo chúng ta chỉ cần chỉ ra tính hiệu quả, tính khả thi của thuật toán thực hiện một nhiệm vụ và đưa ra các phương pháp cài đặt.

Trong bài tạp lớn này chúng em tìm hiểu giải thuật BFS và A\* áp dụng cho trò chơi pluzz(9 ô 8 số).

Trong quá trình tìm hiểu nhóm em còn nhiểu vấn đề thiếu sót, mong cô và các bạn góp ý để nhóm em hoàn thiện hơn.

# **GIỚI THIỆU BÀI TOÁN**

Trong bảng ô vuông 3 hàng, 3 cột , mỗi ô chứa một số nằm trong phạm vi từ 1 đến 8 sao cho không có 2 ô có cùng giá trị, có một ô trong bảng bị trống (không chứa giá trị nào cả). Xuất phát từ một sắp xếp nào đó các số trong bảng, hãy dịch chuyển ô trống sang phải, sang trái, lên trên hoặc xuống dưới (nếu có thể được) để đưa về bảng ban đầu về bảng qui ước trước.



Bài toán đặt ra là tìm phương án tối ưu cho số lần di chuyển là ít nhất.

Đầu tiên cần phải quan tâm để giải bài toán là xác định trạng thái đích . Trạng thái đích được xác định theo trạng thái ban đầu.

## **I. GIẢI THẬT BFS**

Tại mỗi bước cảu thuật toán BFS ta chọn theo trạng thái có khả năng cao nhất trong số các trạng thái đã được xét cho đến thời điểm đó, như vậy với phương pháp này ta ưu tiên đi vào những nhánh tìm kiếm có khả năng cao nhất.

Tư tưởng chủ đạo của Beardth First Search giúp ta sẽ không bị đi những vòng luẩn quẩn bởi vì nếu đi càng sâu vào một hướng mà ta phát hiện hướng đi càng ngày càng tệ, thậm chí còn tệ hơn cả những hướng mà ta chưa đi thì sẽ không tiếp tục trên hướng đi hiện tại nữa mà sẽ lựa chọn theo một trong những hướng đi tốt nhất trong tập hợp chưa đi.

Cài đặt giải thuật BFS.

Heap: Tập hợp biến động các đối tượng theo một thứ tự ưu tiên nhất định mỗi đối tượng được đặc trưng bởi 1 khóa thể hiện mức độ ưu tiên của nó, thông thường các đối tượng được thao tác để chọn ra đối tượng có độ ưu tiên cao nhất đê xử lý tiếp tục.

Heap là cấu trúc dữ liệu trừu tượng, với 4 thao tác cơ bản là đưa một phần tử mới vào trong Heap (Insert), lấy phần tử có độ ưu tiên lớn nhất (Remove), đưa nút có khóa K về đúng vị trí theo chiều hướng đến gốc cây (UnHeap), và đưa nút có khóa K về đúng vị trí theo chiều hướng đến ngọn (các nút lá) (DownHeap).

* Cài đặt giải thuật BFS.

Để cài đặt giải thuật BFS, ta thường dùng 2 tập hợp:

* **OPEN:** Tập hợp những trạng thái đã được sinh ra nhưng chưa được xét đến, OPEN là một hàng đợi ưu tiên mà trong đó phần tử có độ ưu tiên cao nhất là phần tử tốt nhất. Hàng đợi ưu tiên này thường được cài đặt bằng Heap.
* **CLOSE:** Tập chứa các trạng thái đã được xét đến, ta cần lưu trữ những trạng thái này trong bộ nhớ để phòng trường hợp khi một trạng thái mới sinh ra trong quá trình duyệt lại trùng hợp với trạng thái đã xét trước đó. Khi không gian tìm kiếm có dạng cây thì ta không cần sử dụng với tập này
* **Cài đặt cách khác.**

+ Đặt OPEN chứa trạng thái khởi đầu.

Cho đến khi tìm được trạng thái đích hoặc không còn nút nào trong tập OPEN vòng lặp thực hiện duyệt các trạng thái tiếp theo, mỗi bước:

+ Chọn trạng thái tốt nhất Tbest trong OPEN ( và xóa trạng thái Tbest khỏi OPEN)

+ Nếu Tbest là trạng thái kết thúc thì thoát.

+ Nếu Tbest không phải là trạng thái tốt nhất, tạo các nút trạng thái Tn con của Tbest. Đối với mỗi trạng thái kế tiếp, thực hiện tính hàm chi phí f(Tn) và thêm Tn vào trong OPEN

### **a**.Thuật toán

procedure bfs;

begin

open:=[start]; closed:=[];

while open<>[] do

begin

loại tt ngoài cùng bên trái của open, gọi nó là u

if (u là một đích) then thông báo kết quả, thoát

else begin

Đưa u vào closed

Phát sinh các con v của u

Loại các con đã có trong open+closed

Đưa các con còn lại vào bên phải open (1)

end

end

Thông báo thất bại

End

## **II. GIẢI THUẬT A\***

Trò chơi 8 số ở mức độ khó vừa phải nên là một trò chơi rất thú vị. một giải pháp điển hình gồm khoảng 20 bước, mặc dù con số này biến đổi phụ thuộc vào trạng thái đầu. Hệ số rẽ nhánh khoảng bằng 3 (khi ô trống ở giữa, có bốn khả năng di chuyển; khi nó ở góc có hai khả năng di chuyển; và khi nó ở trên các cạnh, có ba khả năng đi). Để giải bài toán này ta cần tìm một hàm Heuristic tốt. Ta có hai hàm ước lượng:

H1 = số lượng các số sai vị trí

H2 = tổng số khoảng cách của các số so với vị trí mục tiêu, là tổng khoảng cách theo chiều ngang và theo chiều dọc.

Bài toán được giải bằng thuật toán A\* sẽ thực hiện theo các bước sau:

* Từ trạng thái ban đầu ta xác định được trạng thái đích.
* Gọi G là số bước đã di chuyển ô trống
* H là hàm heuristic, ước tính số hao tổn để tới trạng thái đích, tính bằng tổng các quãng đường của các ô ở vị trí sai để về vị trí đúng.
* F = G +H.
* Có hai danh sách Open và Close, Open chứa các trạng thái chưa xét, danh sách Close chứa các trạng thái đã xét.
* Ban đầu ta them trạng thái khởi đầu vào Open, sau đó chọn trạng thái có f = g + h nhỏ nhất, lúc này danh sách Open chứa duy nhất trạng thái khởi đầu nên ta lấy trạng thái khởi đầu khỏi Open, và đưa vào danh sách Close các trạng thái đã xét. Từ trạng thái đang xét ta xác định được trạng thái tiếp theo, dựa vào các hướng di chuyển của ô trống. Đưa tất cả các trạng thái mới mà chưa có trong Close và Open vào danh sách Open. Ta tiếp tục chọn ra trạng thái có f = g + h nhỏ nhất khỏi Open như bước đầu tiên cho đến tìm ra trạng thái đích thì dừng lại.

Từ trạng thái đích vừa tìm được ngược lại danh sách ta sẽ tìm được đường đi từ trạng thái khởi đầu đến trạng thái đích.

Các bước giải bài toán trên:

Đầu tiên ta xác định trạng thái tiếp theo của bài toán trên:

Có 3 trường hợp xảy ra:

|  |  |  |
| --- | --- | --- |
| 2 | 8 | 3 |
| 1 |  | 4 |
| 7 | 6 | 5 |

|  |  |  |
| --- | --- | --- |
| 2 | 8 | 3 |
| 1 | 6 | 4 |
|  | 7 | 5 |

|  |  |  |
| --- | --- | --- |
| 2 | 8 | 3 |
| 1 | 6 | 4 |
| 7 | 5 |  |

1 2 3

Đối với trường hợp 1 có g=1, h=3, f=h+g=4

Đối với trường hợp 2 có g=1, h=4, f=h+g=5

Đối với trường hợp 3 có g=1, h=5, f=h+g=6

So sánh các f với nhau ta thấy f của TH1 nhỏ nhất nên trạng thái tiếp theo là trạng thái 1.

|  |  |  |
| --- | --- | --- |
| 2 | 8 | 3 |
|  | 1 | 4 |
| 7 | 6 | 5 |

Từ 1 ta có ba trạng thái:

|  |  |  |
| --- | --- | --- |
| 2 |  | 3 |
| 1 | 8 | 4 |
| 7 | 6 | 5 |

|  |  |  |
| --- | --- | --- |
| 2 | 8 | 3 |
| 1 | 4 |  |
| 7 | 6 | 5 |

1.1 1.2 1.3

Đối với trường hợp 1.1 có g=2, h=3, f=h+g=5

Đối với trường hợp 1.2 có g=2, h=4, f=h+g=6

Đối với trường hợp 1.3 có g=2, h=4, f=h+g=6

So sánh các f với nhau ta thấy f của trường hợp 1 nhỏ nhất lên trạng thái tiếp theo là trạng thái 1.1

Từ 1.1 có hai trạng thái:

|  |  |  |
| --- | --- | --- |
|  | 2 | 3 |
| 1 | 8 | 4 |
| 7 | 6 | 5 |

|  |  |  |
| --- | --- | --- |
| 2 | 3 |  |
| 1 | 8 | 4 |
| 7 | 6 | 5 |

1.1.1 1.1.2

Đối với trường hợp 1.1.1 có g=3, h=2, f=h+g=5

Đối với trường hợp 1.1.2 có g=3, h=4, f=h+g=7

So sánh các f với nhau ta thấy f của trường hợp 1 nhỏ nhất nên trạng thái tiếp theo là trạng thái 1.1.1

Từ trạng thái 1.1.1 ta có một trạng thái:

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
|  | 8 | 4 |
| 7 | 6 | 5 |

Trạng thái này có: g=4, h=1 , f=5.

Ta có trạng thái đích:

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| 8 |  | 4 |
| 7 | 6 | 5 |

### **a**.Thuật toán

procedure A\*(uo);

begin

g(uo)=0; f(uo)=h(uo);

open:=[uo]; closed:=[];

while open<>[] do

begin

loại u ngoài cùng bên trái của open

Đưa u vào closed;

if (u là một đích) then thông báo thành công, thoát else

begin

Sinh các con v của u;

For v thuộc con(u) do begin

g(v):=g(u)+c[u,v];

If v không thuộc open hay closed

begin

f(v):=g(v)+h(v);

cha(v):=u;

Bỏ v vào open;

end

If v thuộc open (tồn tại v’ thuộc open, sao cho v=v’)

If g(v)<g(v’) then

Begin

g(v’):=g(v);

f(v’):=g(v’)+h(v’);

Cha(v’):=u;

End;

If v thuộc closed (tồn tại v’ thuộc closed, sao cho v=v’)

If g(v)<g(v’) then

Begin

f(v):=g(v)+h(v); cha(v):=u;

Đưa v vào open;

Loại v’ khỏi closed;

End;

End;{for}

Sắp xếp open để t.thái tốt nhất nằm bên trái;

End{else}

End;{while}

Thông báo thất bại

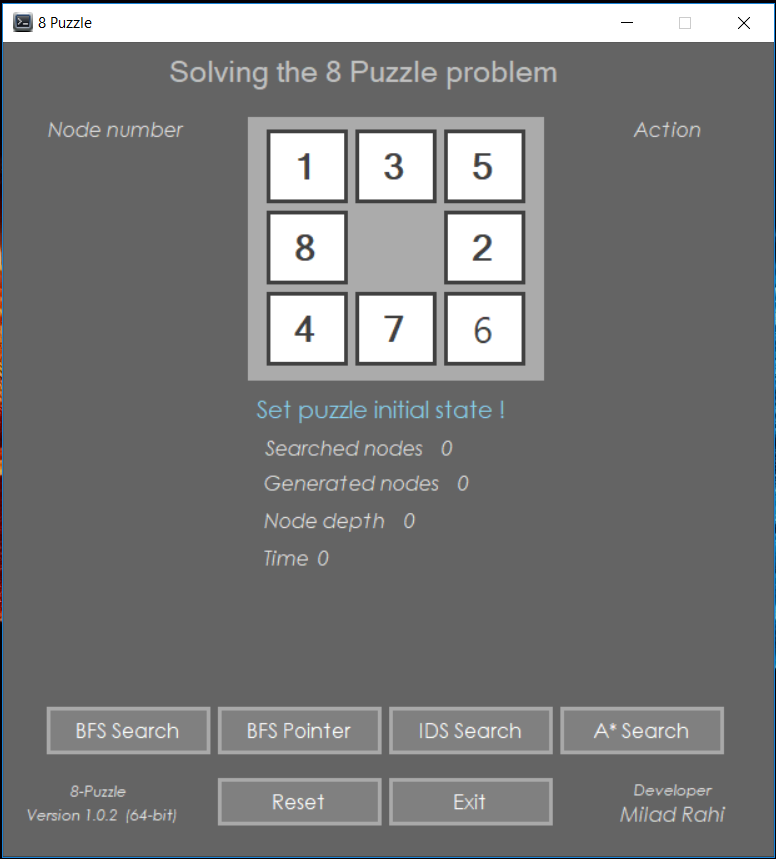
End;{procedure}

# **DEMO**

## a. Giao diện chương trình

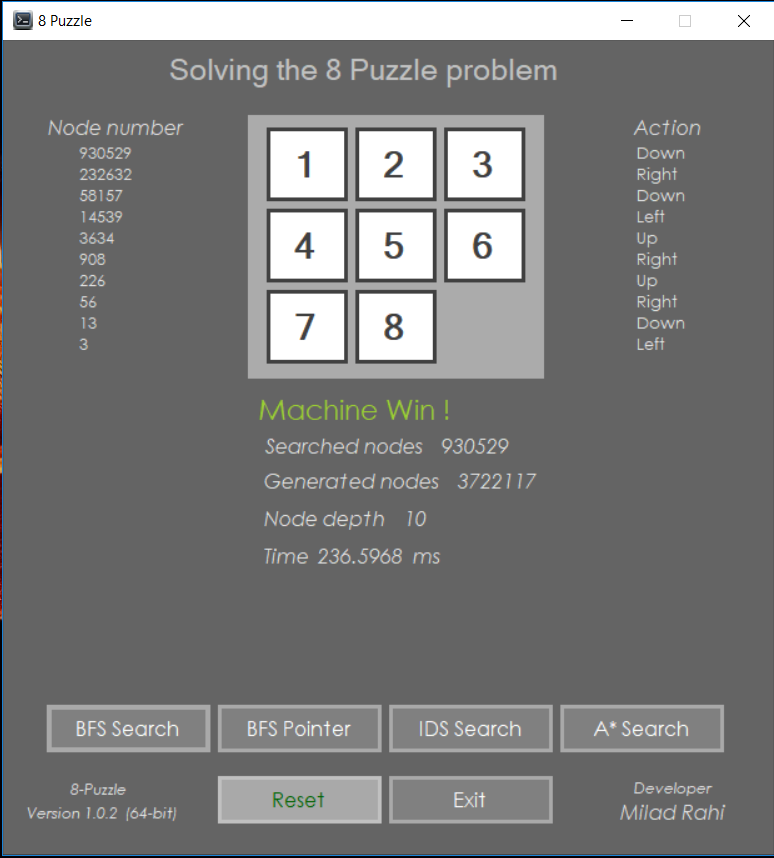
Giao diện khi chạy chương trình cho trò chơi pluzz 9 ô mặc định là trạng thái kết thúc.

Chúng ta có thể thay đổi các giá trị trong hình để không giống trạng thái kết thúc, gọi là trạng thái ban đầu.



* Action: hướng đi của mỗi ô khi d chuyển qua ô trống.
* Reset: Trở về trạng thái ban đầu
* Exit: Thoát chương trình

## b. Giải thuật BFS

* Sau khi chạy giải thuật BFS
* Ta thu được kết quả ở màn hình sau.
* Thời gian thực hiện: 236.5968ms

## **c**. Giải thuật A\*

* Cũng từ trạng thái ban đầu ta thực hiện giải A\* thu được kết quả ở màn hình sau.
* Thời gian thực hiện: 6.7851ms

# **ĐÁNH GIÁ**

* Ta thấy số bước thực hiện của 2 giải thuật trên giống nhau nhưng khác về bước di chuyển của các ô.
* Thời gian thực hiện của giải thuật A\* ngắn hơn giải thuật BFS.
* Thuật toán BFS khảo sát tất cả các trường hợp nên chắc chắn tìm được nghiệm.
* Thuật toán A\* lại chọn hàm có f=h+g nhỏ nhất nên không phải khảo sát tất cả các trường hợp và vì thế không mất nhiều thời gian thực hiện hơn.