Mô tả bài toán di chuyển 8 ô số trên bàn cờ:

1**. Điều kiện của trạng thái đầu và đích:**

Có những trạng thái của bảng số không thể chuyển về trạng thái đích. Người ta

chứng minh được rằng, để có thể chuyển từ trạng thái đầu tới trạng thái đích, thì

trạng thái đầu này phải thỏa mãn điều kiện được xác định như sau:

Ta xét lần lượt từ trên xuống dười, từ trái sang phải, với mỗi ô số đang xét (giả sử

là ô thứ i), ta kiểm tra xem phía sau có bao nhiêu ô số có giá trị nhỏ hơn ô đó. Sau

đó ta tính tổng N = n1 + n2 + … + n8.

Ta có quy tắc chung sau cho bài toán n-puzzle:

Nếu số ô vuông lẻ:

N mod 2 = 0 (1)

Nếu số ô vuông chẵn:

N mod 2 = 0 và ô trống phải nằm ở hàng chẵn xét từ trên xuống. (2)

N mod 2 = 1 và ô trống phải nằm ở hàng lẻ xét từ trên xuống. (3)

Cụ thể, ta đang xét bài toán 8-puzzle – có 9 ô vuông nên trạng thái đầu phải thỏa

mãn điều kiện (1).

Ví dụ: Cho trạng thái đầu sau 2-0-6-8-7-5-4-3-1

Xét ô thứ nhất có giá trị 2: Phía sau có 1 ô nhỏ hơn (1) => n1 = 1

Xét ô thứ hai có giá trị 6: Phía sau có 4 ô nhỏ hơn (5,4,3,1) => n2 = 4

Xét ô thứ ba có giá trị 8: Phía sau có 5 ô nhỏ hơn (7,5,4,3,1) => n3 = 5

Xét ô thứ tư có giá trị 7: Phía sau có 4 ô nhỏ hơn (5,4,3,1) => n4 = 4

Xét ô thứ năm có giá trị 5: Phía sau có 3 ô nhỏ hơn (4,3,1) => n5 = 3

Xét ô thứ sáu có giá trị 4: Phía sau có 2 ô nhỏ hơn (3,1) => n6 = 2

Xét ô thứ bảy có giá trị 3: Phía sau có 1 ô nhỏ hơn (1) => n7 = 1

Xét ô thứ tám có giá trị 1: Phía sau không còn ô nào nhỏ hơn => n8 = 0

N = 1 + 4 + 5 + 4 + 3 + 2 + 1 + 0 = 20

Ta có 20 mod 2 = 0 => Thỏa mãn.

Những trạng thái của bảng số mà có thể chuyển về trạng thái đích gọi là cấu hình

hợp lệ, ngược lại gọi là cấu hình không hợp lệ.

Với bảng số kích thước mxm (m là cạnh) thì ta có không gian trạng thái là (mxm)!

Với bài toán 8-puzzle, các trạng thái có thể có của bảng số là (3x3)! = 362880. Nếu

m tăng lên 1, sẽ làm không gian trạng thái tăng lên rất lớn, do vậy các phiên bản

bài toán với m>3 ít khi được áp dụng.

**2. Hàm Heuristic được sử dụng:**

Ta sẽ cải tiến việc giải bài toán bằng cách sử dụng phương pháp tìm kiếm có thông tin A\*. Tìm kiếm A\* là một ví dụ của tìm kiếm theo lựa chọn tốt nhất (Best-first search) – một thuật toán tìm kiếm tối ưu hóa theo độ sâu bằng cách mở rộng nút “hứa hẹn” nhất có thể dẫn đến đích.

Triển vọng của một nút (mỗi nút tương ứng với một trạng thái của bài toán) được ước lượng bằng một hàm đánh giá heuristic. Hàm này có thể là số lượng ô đặt sai vị trí (h1) hoặc là tổng khoảng cách Manhattan (h2). Ví dụ: Có trạng thái bài toán sau:

Từ : 1-3-4-8-5- -7-6-2 đến đích : 1-2-3-4-5-6-7-8- -

Có 5 ô số nằm sai vị trí so với trạng thái đích

=> h1 = 5

• Tính khoảng cách Mahattan, xác định tọa độ (dòng, cột) của ô khi ở sai vị trí và khi ở đúng vị trí:

− Ô số 3 khi ở sai vị trí có tọa độ (0,1)

− Ô số 3 khi ở đúng vị trí có tọa độ (0,2)

=> Khoảng cách Mahattan của ô số 3 bằng |0 – 0| + |2 – 1| = 1

Tương tự, ô số 4 có khoảng cách Mahattan là 3, ô số 8 có khoảng cách Mahattan là 2, ô số 6 có khoảng cách Mahattan là 2, ô số 2 có khoảng cách Mahattan là 3. Những ô số 1, 5, 7 đã ở đúng vị trí nên có khoảng cách Mahattan là 0.

= > h2 = 0 + 1 + 3 + 2 + 0 + 0 + 2 + 3 = 11

Điểm khác biệt của A\* so với tìm kiếm theo lựa chọn tốt nhất đó là nó còn xét tới đoạn đường đã đi qua. Ta có:

f(n) = g(n) + h(n)

Trong đó, g(n) là chi phí đi từ trạng thái đầu đến trạng thái đang xét, h(n) là hàm heuristic ước lượng chi phí đi từ trạng thái hiện tại tới đích. Hàm f(n) có giá trị càng thấp thì trạng thái đó càng có độ ưu tiên và triển vọng cao.

**3. Giải thuật bài toán:**

Ban đầu ta có Open là tập chứa các trạng thái chưa được xét (sắp xếp theo f tăng dần), Close là tập các trạng thái đã được xét. Ban đầu Open chỉ chứa trạng thái ban đầu, tập Close rỗng.

Begin

Open:={Start};

Close:=∅;

While (Open <> ∅) do

Begin X=Retrieve(Open);

{Chọn X sao cho f(X) đạt là nhỏ nhất}

If (X=Goal) then return True

Else

Begin

Sinh ra các trạng thái con của X;

For mỗi nút con Y của X do

If (Y ∉ Open) và (Y ∉ Close)

Begin

Tính f(Y);

Open = Open ∪ {Y};

End;

If (Y ∈ Open)

If (g(Y) < g(Y’)) cập nhật lại giá trị f(Y’), đặt cha của Y’ là X;

If (Y ∈ Close)

If (g(Y) < g(Y’)) cập nhật lại giá trị f(Y’) , đặt cha của Y’ là X, cập nhật lại giá trị f và g của tất cả các con của Y đã có trong Open và Close.

End;

Close = Close ∪ {X};

End;

Return False;

End.