HEURISTIC SEARCH

Bài toán người bán hàng (Greedy Traveling Salesman)

- Cho *N* thành phố, trong đó hai thành phố bất kỳ đều có nối với nhau.
- Một người xuất phát tại một thành phố, đi qua tất cả các thành phố còn lại một lần duy nhất, và trở về thành phố xuất phát.
- Hãy xác định lộ trình cho người đó sao cho tổng chi phí là thấp nhất.

Thuật giải GTS1:

Ý tưởng: cố gắng đạt được lời giải tốt nhất ở mỗi bước thực hiện bằng cách chọn đường đi có chi phí thấp nhất tại đường đi hiện tại và tiếp tục đi.

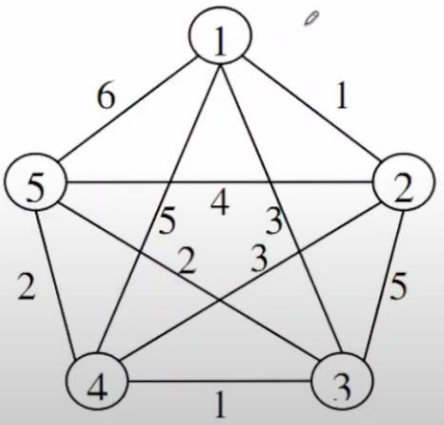
Xây dựng một lịch trình TOUR có chi phí là COST tối thiểu cho bài toán, trong trường hợp phải qua N – thành phố với ma trận chi phí là C và bắt đầu tại đỉnh U.

Không tối ưu, tối ưu tại thời điểm chọn điểm mới nhưng không tổng quát.

Không đảm bảo được lộ trình ngắn nhất.

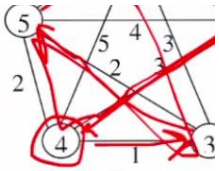
Ví dụ minh họa 1: GTS1

Hãy xác định lộ trình cho người du lịch đi qua 5 thành phố như đồ thị bên dưới bằng thuật toán GTS1.



$$c_{ij} = \begin{pmatrix} \infty & 1 & 3 & 5 & 6 \\ 1 & \infty & 5 & 3 & 4 \\ 3 & 5 & \infty & 1 & 2 \\ 5 & 3 & 1 & \infty & 2 \\ 6 & 4 & 2 & 2 & \infty \end{pmatrix}$$

Khởi đầu: Tour := Ø, Cost := 0, V := 1



Chọn W = 2: Tour = {(1,2)}, Cost = 1

Chọn W = 4: Tour = {(1,2), (2,4)}, Cost = 4

Chọn W = 3: Tour = {(1,2), (2,4), (4,3)}, Cost = 5

Chọn W = 5: Tour = {(1,2), (2,4), (4,3), (3,5)}, Cost = 7

Trở về: Tour = {(1,2), (2,4), (4,3), (3,5), (5,1)}, Cost = 13

Chu trình: 1 → 2 → 4 → 3 → 5 → 1, chi phí là 13

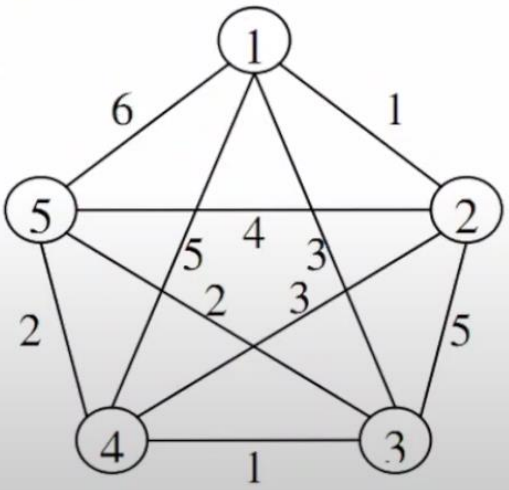
Chu trình: 1 → 2 → 5 → 4 → 3 → 1, chi phí là 11

Thuật giải GTS2:

GTS2 - tạo ra các lịch trình từ p thành phố xuất phát riêng biệt, cho bài toán: *Tìm lịch trình của người bán hàng qua N thành phố đã nói, trong đó 1<P<N. Khi đó, P lịch trình được tạo ra tuần tự và chỉ lộ trình tốt nhất đã tìm thấy được giữ lại mà thôi.*

GST2: Các giá trị Input: N, P, ma trân chi phí C và P thành phố khởi tạo: $\{V_1, V_2, \ldots, V_p\}$

Hãy xác định lộ trình cho người du lịch đi qua 5 thành phố như đồ thị bên dưới bằng thuật toán GTS2 với P = 3 (chọn đỉnh 1, 2, 4).



Chọn $P = 3$ đỉnh xuất phát khác nhau, kết quả thực hiện của thuật toán là:

- Chọn xuất phát từ thành phố 1:
  - Chu trình: $1 \to 2 \to 4 \to 3 \to 5 \to 1$, chi phí là 13. Cost(1)=13
  - Cost = Cost (1) = 13.
- Chọn xuất phát từ thành phố 2:
  - Chu trình: $2 \to 1 \to 3 \to 4 \to 5 \to 2$, chi phí là 11. Cost(2)=11
  - Cost(2)=11< 13=Cost(1) $\Rightarrow$ Cost = Cost(2)=11.
- Chọn xuất phát từ thành phố 4:
  - Chu trình: $4 \to 3 \to 5 \to 2 \to 1 \to 4$, chi phí là 13. Cost(4)=13
  - Cost=11<13=Cost(4) $\Rightarrow$ Cost = 11

Vậy lời giải tốt nhất là xuất phát từ thành phố 2 với chi phí là 11.

Thuật giải leo đồi (Hill-Climbing Search)

Trong tìm kiếm trên đồ thị, chiến lược của việc thử đạt tới đích bằng cách *lựa chọn những đỉnh mà được dự đoán trước là gần tới đích nhất* thì được gọi là Hill-Climbing.

Đặc biệt là thứ tự chọn lựa những đỉnh sau n. Tính $\hat{h}(n_i)$ cho những đỉnh con: (n_1, n_2,…, n_m) của *n* . *Sau đó chọn đỉnh có* $\hat{h}(n_i)$ *là nhỏ nhất làm đỉnh kế.*

n:= Start node.

Loop: If Goal(n) Then Exit (*Success*)

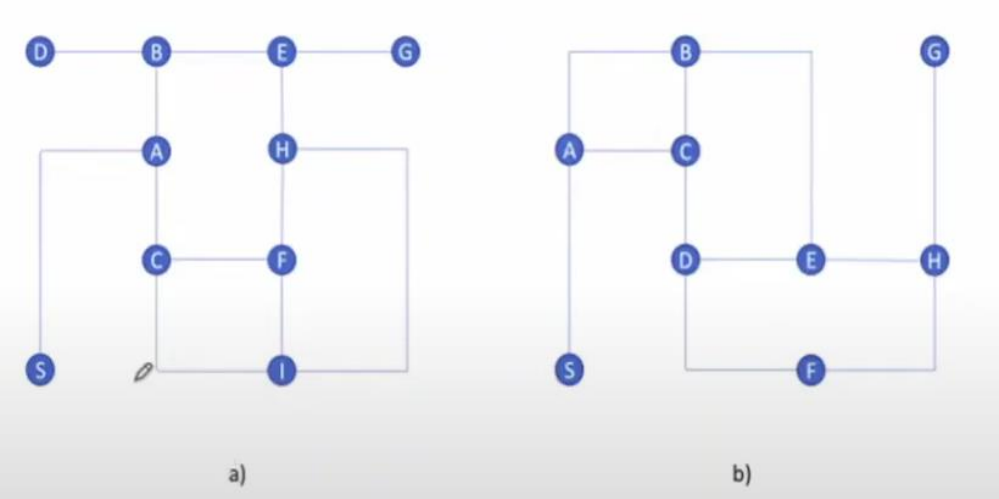
Expand n. Compute $\hat{h}(n_i)$ for all Child node n_i and take the child node which gives minimum value as *next n*.

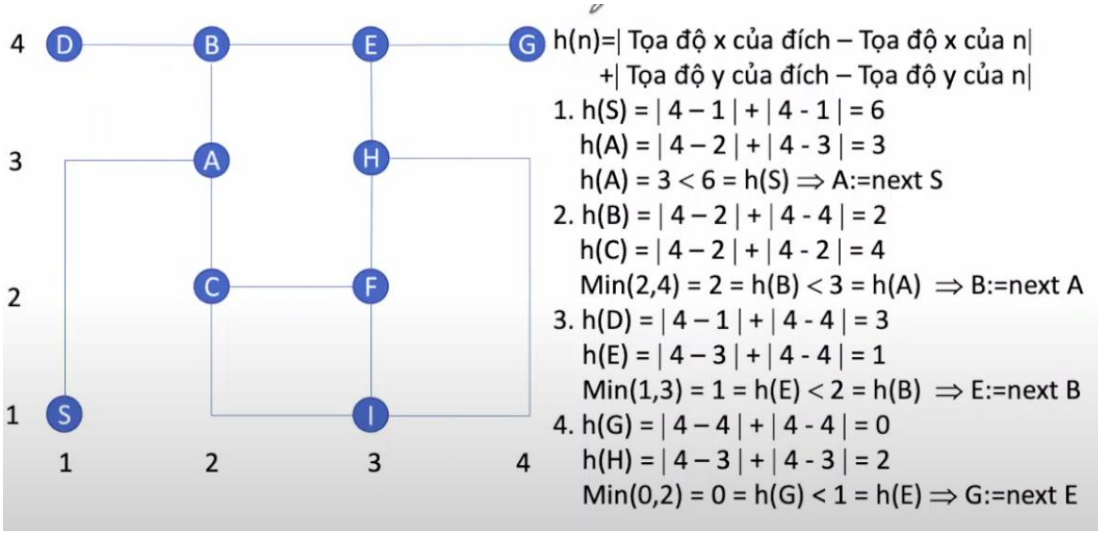
If $\hat{h}(n_i) < \hat{h}(next\ n)$ Then Exit (*Fail*)

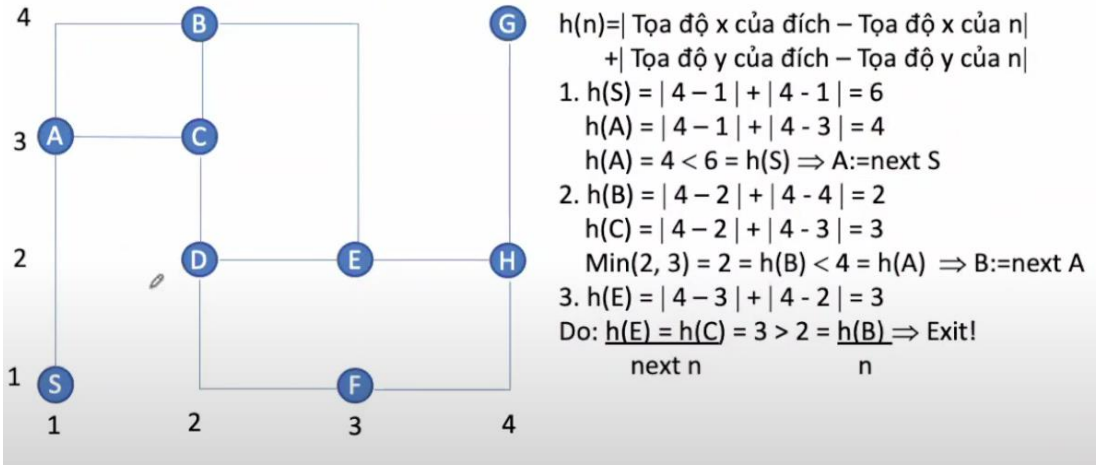
n:= $next\ n$.

Goto **Loop**.

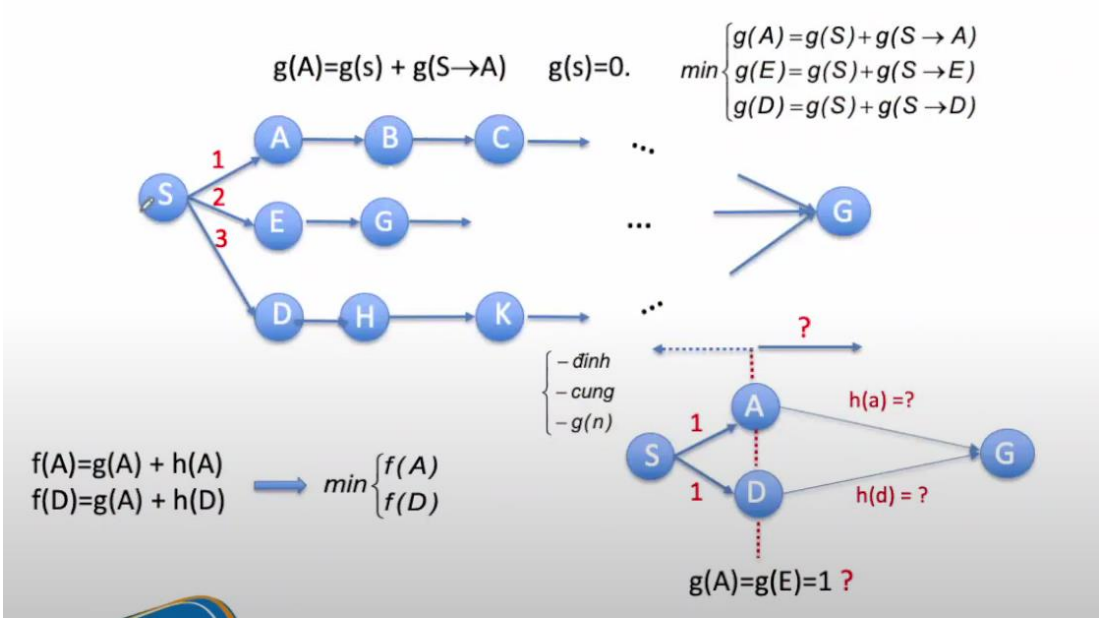
Ví dụ minh họa 1:





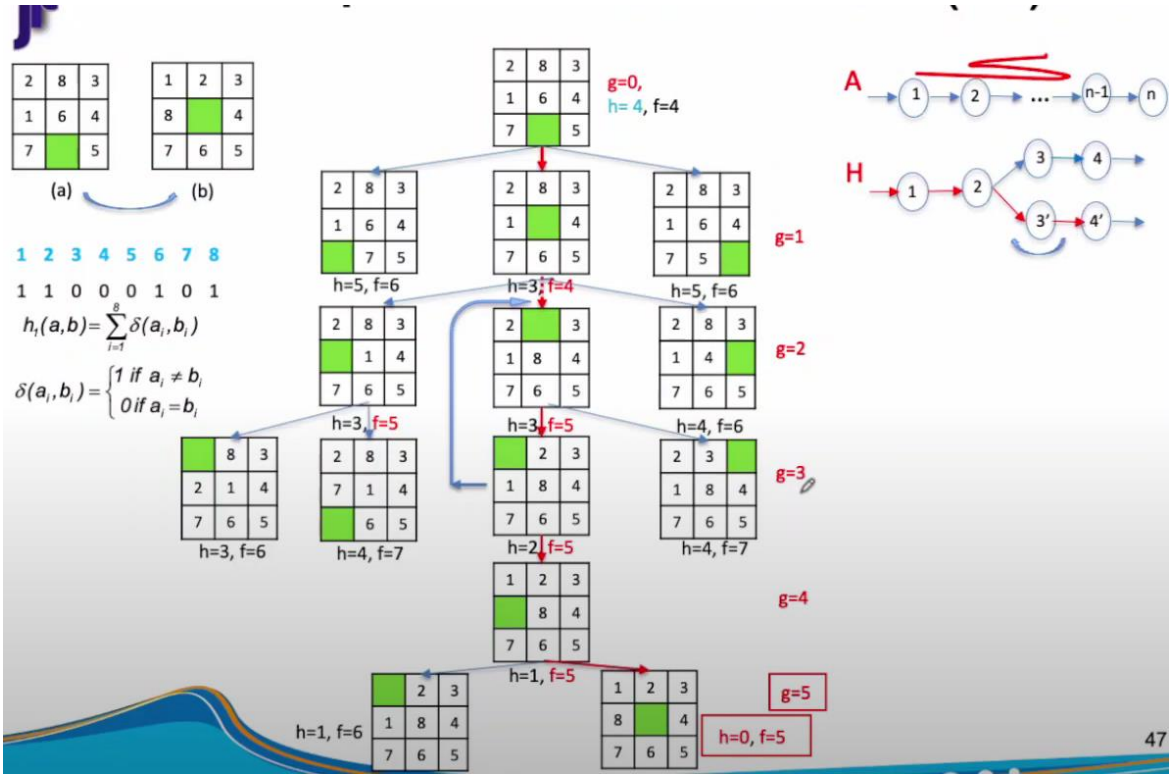


Hàm Heuristic

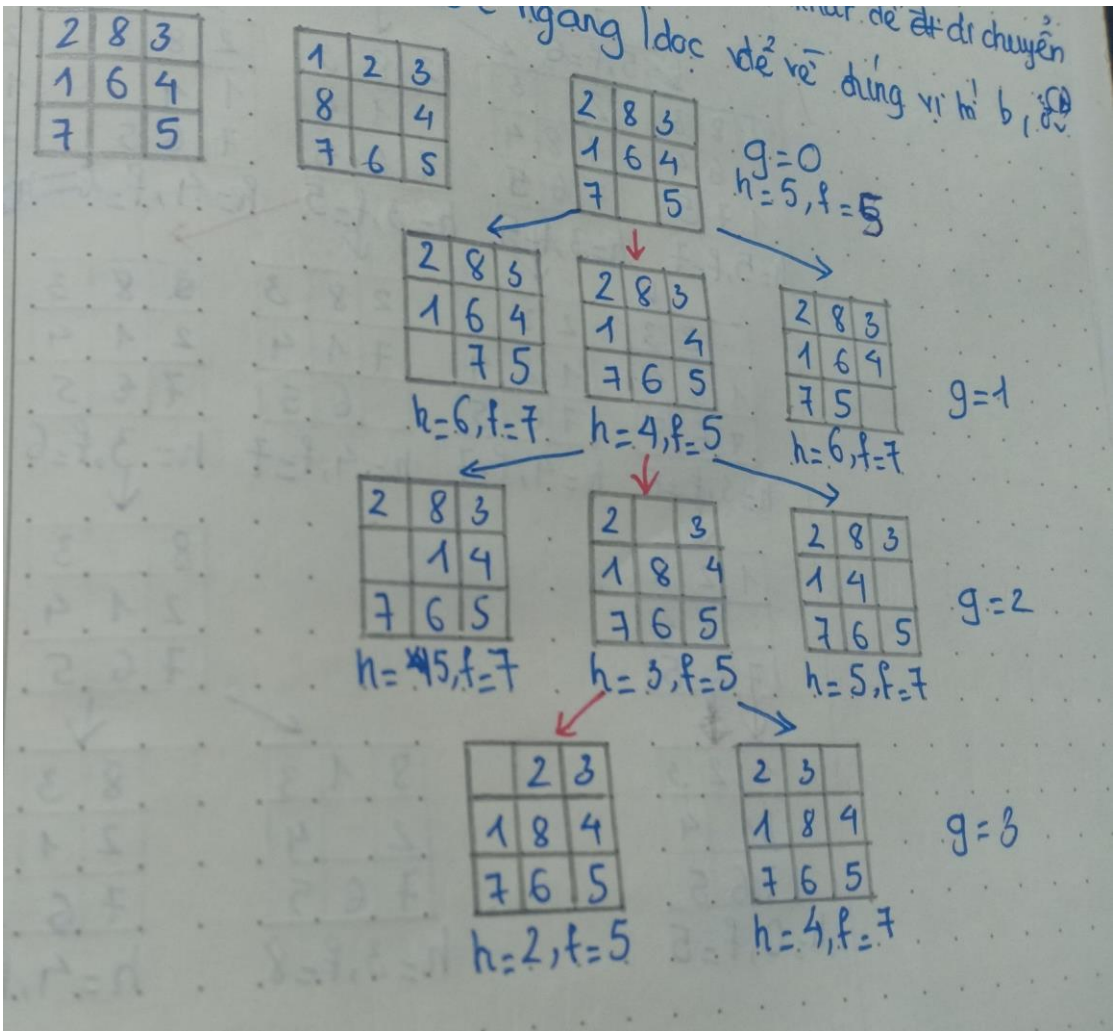


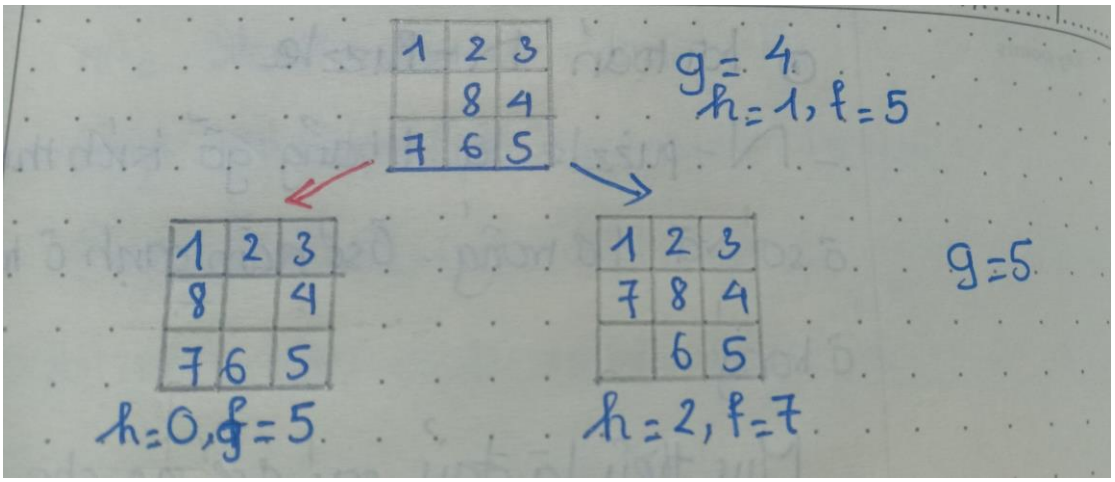
Bài toán N-Puzzle

<u>Ví dụ minh họa 1: Chuyển ô 1 lần (Hàm Heuristic là nếu vị trí ko đúng = 1, đúng = 0)</u>

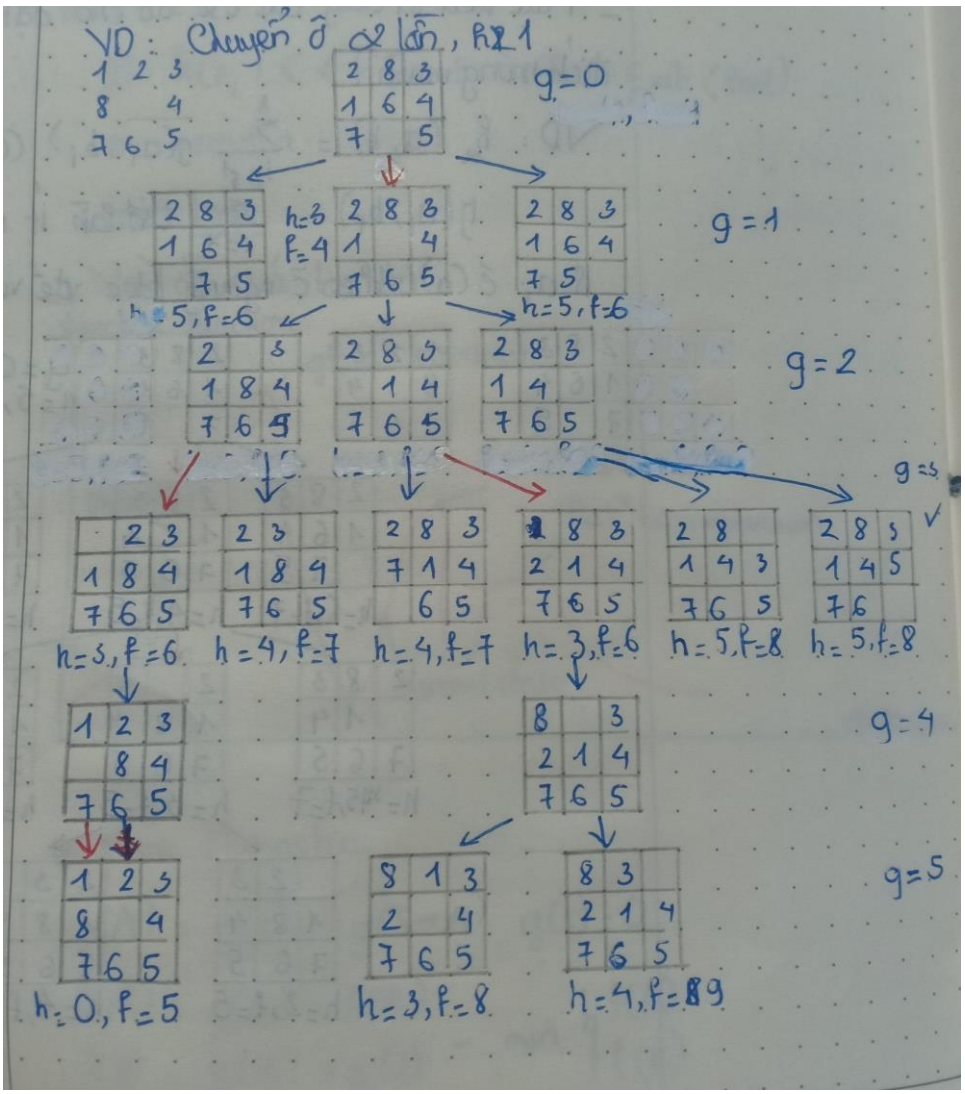


<u>Ví dụ minh họa 2: Chuyển ô 1 lần với hàm heuristic là số lần ít nhất di chuyển để ô về đúng</u>

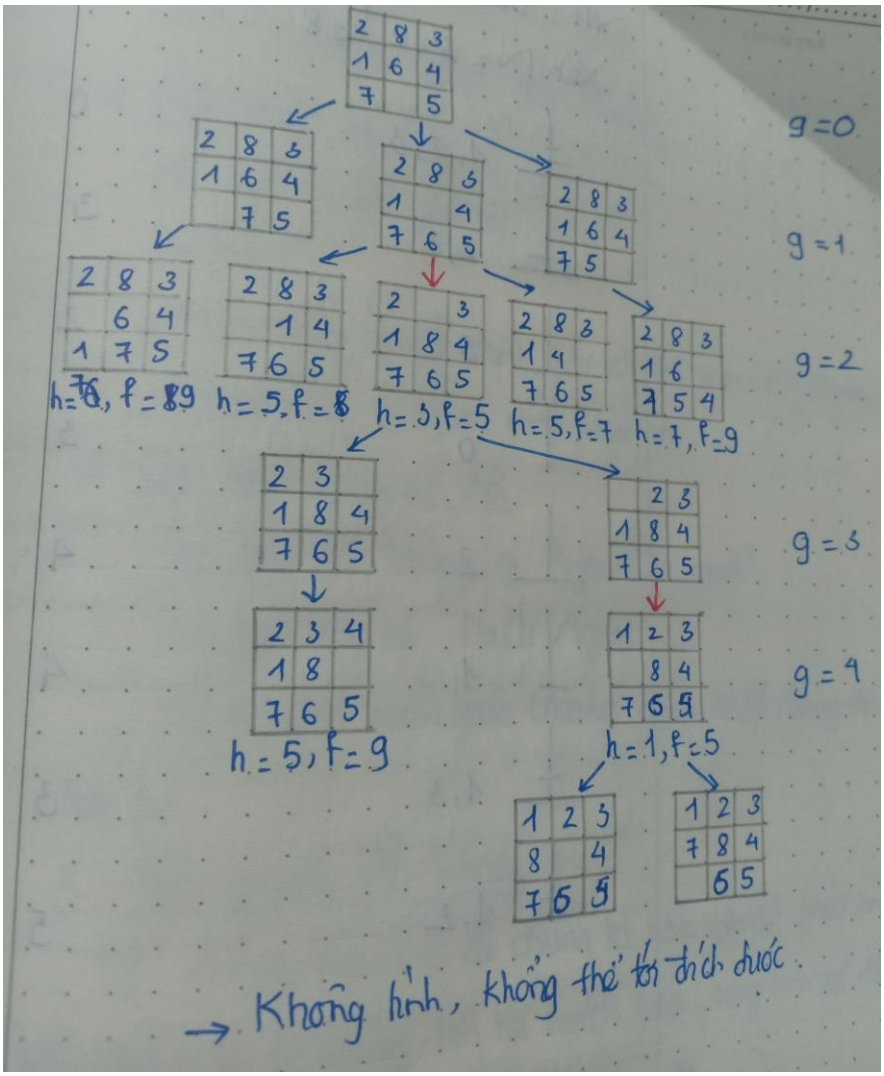




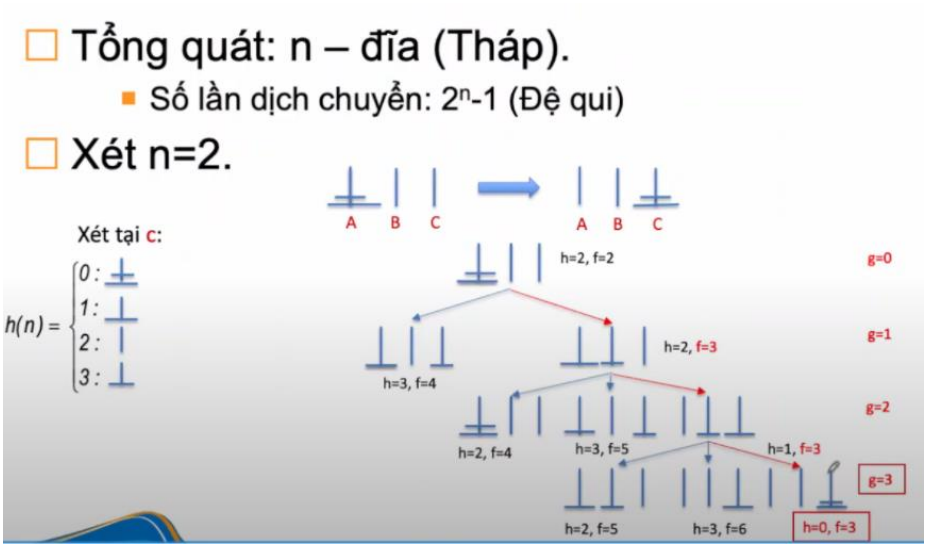
<u>Ví dụ minh họa 3: Chuyển ô 2 lần (Hàm Heuristic là nếu vị trí ko đúng = 1, đúng = 0)</u>



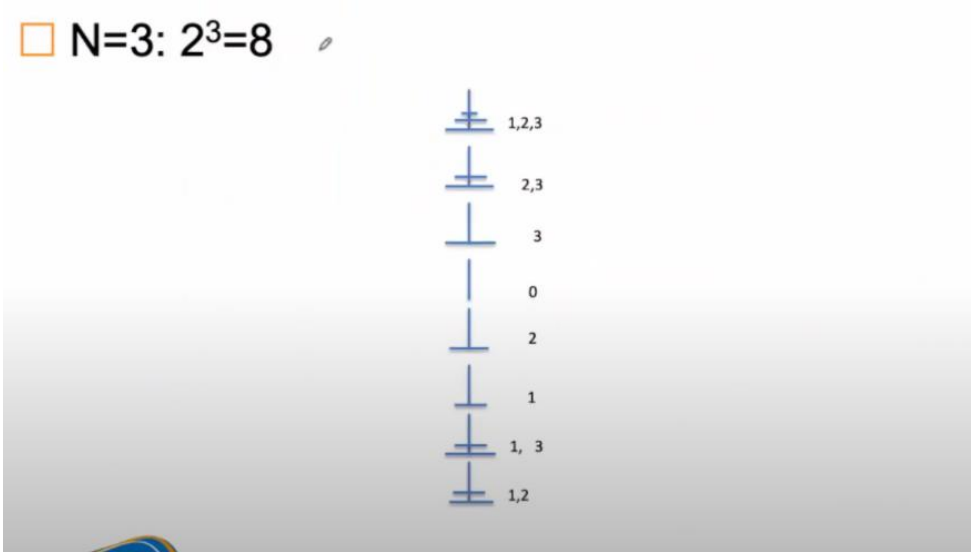
<u>Ví dụ minh họa 4: Chuyển ô 2 lần với hàm heuristic là số lần ít nhất di chuyển để ô về đúng</u>

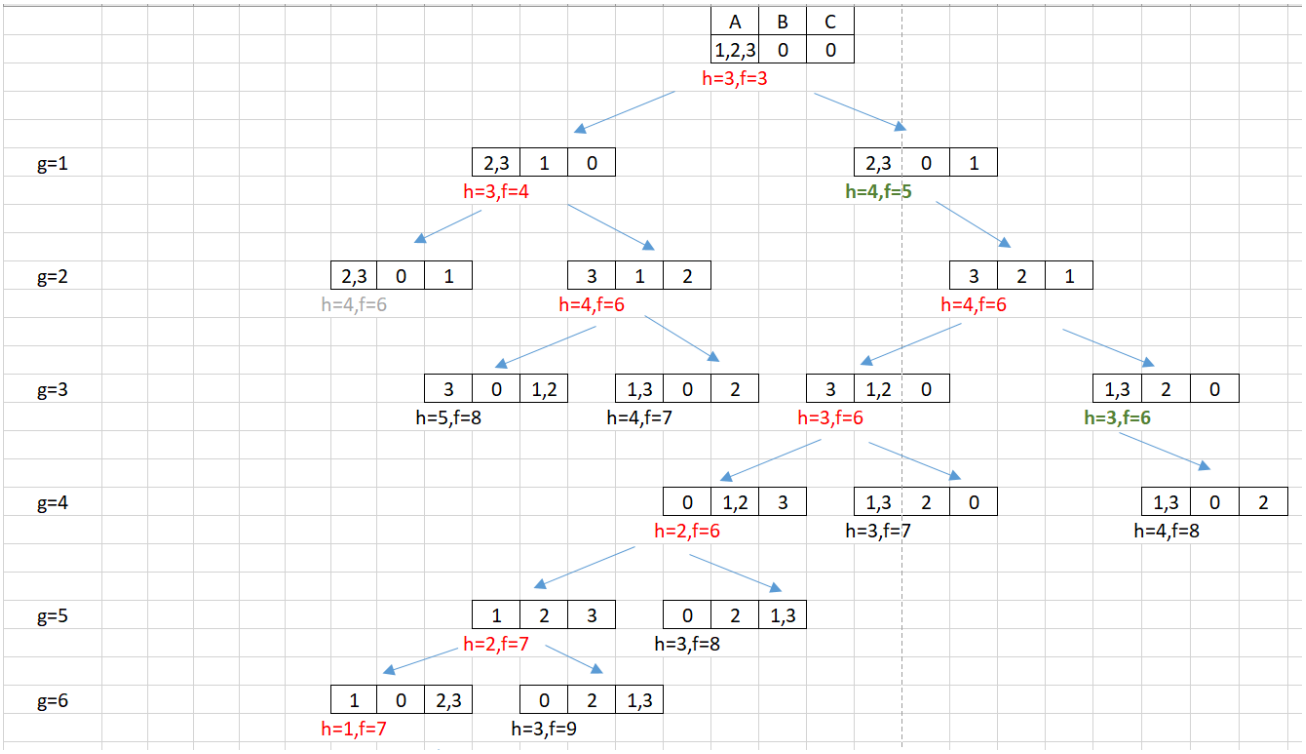


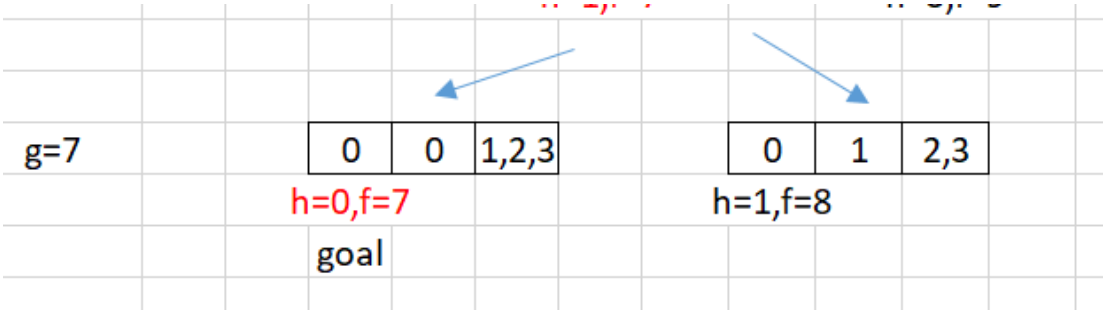
<u>Ví dụ minh họa 5: Tháp Hà Nội với n= 2</u>



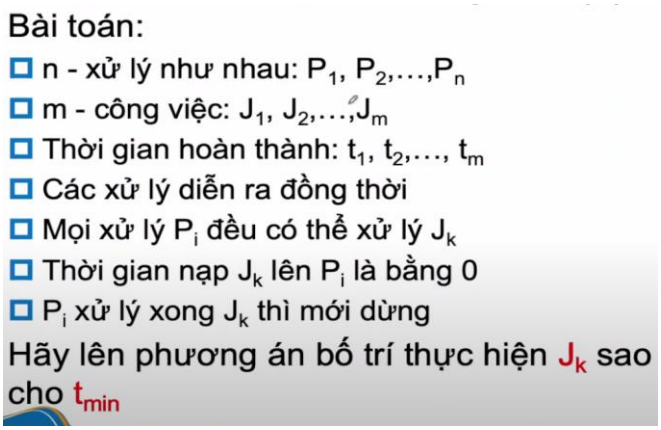
Ví dụ minh họa 6: Tháp Hà Nội với n= 3

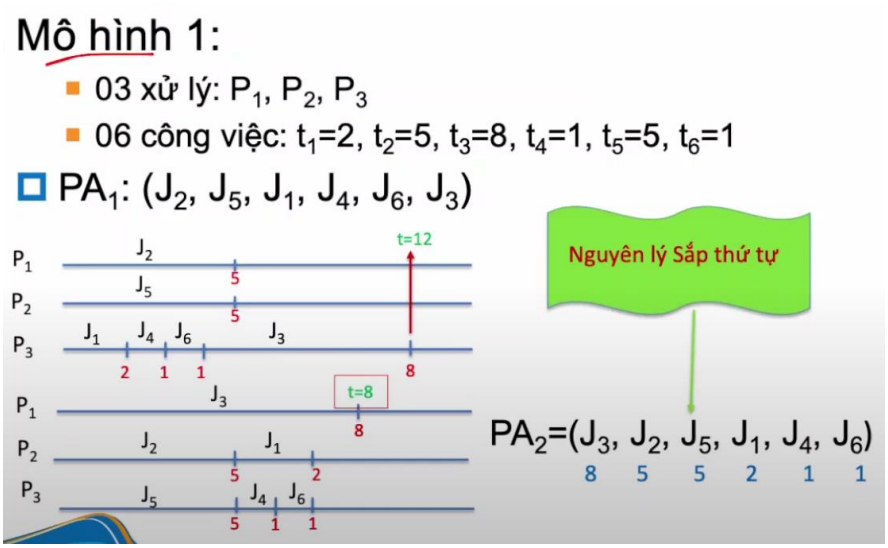


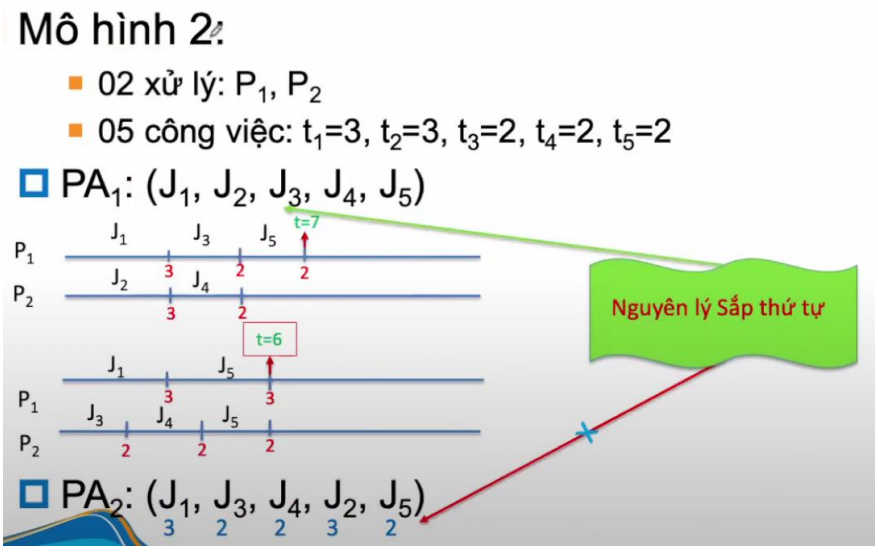




Ví dụ minh họa 7: Bài toán phân công việc







Ví dụ minh họa 8: Bài toán đóng gói (Packing Problem)

Chúng ta có các loại thùng như sau:

H_1: Thứ tự các đối tượng: tùy ý
Cách xếp: theo thứ tự tùy ý

H_2: Thứ tự các đối tượng: sắp theo thứ tự giảm dần
Cách xếp: theo thứ tự giảm dần

H_3: Thứ tự các đối tượng: tùy ý
Cách xếp: ưu tiên trùng khít

H_4: Thứ tự các đối tượng: sắp theo thứ tự giảm dần
Cách xếp: ưu tiên trùng khít