BÁO CÁO BÀI TẬP LỚN

Môn : Trí tuệ nhân tạo

Họ tên : Lê Đình Linh. *Mã sv: 12020219.*

Đề bài : Xây dựng chương trình cài đặt giải thuật tìm kiếm heuristic để giải bài toán N-puzzle.

I.Cấu trúc chương trình:

1.File zip gồm:

+ Project gồm có 3 file .java : Algorithm.java,Tile.java,State.java,1 file input : nPuzzle.inp,sinh ra một file output : nPuzzle.out và 1 file chạy n-Puzzle.jar.

2.Chi tiết các hàm trong file java:

a) Class Algorithm : + void print: in ra các state.

+ boolean Equals: check xem hai state có bằng nhau hay không,dùng để tránh khi duyệt sẽ đi vòng lại state cũ.

+ int misplacing : hàm đánh giá số vị trí khác nhau của mỗi state đối với state khác.

+int mahatan: hàm đánh giá theo khoảng cách mahatan giữa 2 state.

+int inventedHeuristic: hàm đánh giá mới tự phát minh.

+boolean Contain : check xem danh sách các trang thái đang chờ phát triển arrayState có chứa trạng thái mà tại đó trạng thái con temp.child.get(i) của trạng thái temp hiện thời được sinh ra có trùng với nó không->để đảm bảo không sinh ra các trạng thái lặp .

+ int NumOfMove: hàm tính giá trị của số bước di chuyển từ trạng thái đầu đến trạng thái đích bằng cách truy ngược lại trạng thái cha của trạng thái hiện tại cho đến khi truy xuất đến trạng thái ban đầu.

+ void A-star : cài đặt thuật toán A*

+ void readfile : đọc dữ liệu input cho chương trình.

+void writefile: xuất dữ liệu output ra file nPuzzle.out.

Các biến : numOfMove : số bước di chuyển từ trạng thái đầu đến trạng thái hiện tại.

numOfGenerate: số đỉnh đã sinh ra.

Class State:

Các hàm:

+State: định nghĩa mỗi state gồm có một bảng các tiles,khởi tạo từ mảng nguyên.

Biến Tile blank:biến trỏ tới ô trống trong bảng.

+void swapTiles: dùng để swap 2 tile một tiles là blank và tiles kế bên để tạo ra state mới.

+boolean checkBack: kiểm tra mỗi khi phát triển một trạng thái xem trạng thái con được sinh ra có trùng với các trạng thái cha,tổ,tổ tiên,cho đến trạng thái bắt đầu không bằng cách duyệt ngược theo thuộc tính father của mỗi state.

+void initChild: hàm phát triển trạng thái.tại mỗi trạng thái: căn cứ vào tọa độ của ô trống blank mà sinh ra các trạng thái con khác nhau.Khởi tạo state mới bằng cách sao chép từ state cũ và swapTile.

Sau đó gán thuộc tính father cho trạng thái vừa được sinh ra.

+initArray: do sau khi swap để khởi tạo state mới,giá trị của thuộc tính Array của state vừa sinh ra vẫn chưa được thay đổi nên cần phải initial lại.

II. Mô tả thuật toán:

1.Hàm miplacing: hàm đánh giá tính số Tiles không đúng vị trí bằng cách sử dụng vòng lặp.

2.Hàm mahatan: hàm đánh giá tính khoảng cách mahatan giữa các state.

* Hàm misplacing có hời gian chạy lâu hơn hàm mahatan.số state sinh ra nhiều hơn,và số bước di chuyển để tới trạng thái đích nhiều hơn.

3.Hàm inventedHeuristic: hàm đánh giá sử dụng tính chất sau:

Nếu board có size=n.

Khi có 2 state st1,st2. Nếu st1 có n tiles nằm ở sát bên lề của board,và n tile có thứ tự liên tiếp thì nếu dịch chuyển(không quan tâm có ô trống để dịch chuyển được không) cũng theo đường biên của board một bước cho mỗi tiles thì được vị trí đó tại st2,khi đó ta phạt hàm đánh giá bằng cách tăng thêm một lượng:

h(st1,st2)+=n; vì cần ít nhất n bước để move n Tiles đó tới vị trí giống như st2.

Ví dụ ta có st1 : 0 4 5 st2: 2 4 5

2 1 6 3 1 6

3 8 7 8 0 7

3 tile cần move là 2 ,3,8.

Ngoài ra để tăng giá trị của hàm đánh giá cho sát với chi phí thực tế ta sử dụng tính chất:

Nếu có 2 Tiles kề nhau và đảo ngược vị trí cho nhau giữa 2 state thì cần 4 bước để di chuyển 2 tiles đó của state này về vị trí giống state kia.(không quan tâm có di chuyển ngay được không).

Ví dụ ta có st1 : 0 4 5 st2 : 2 6 5

2 1 6 3 1 4

3 8 7 2 8 7

Giữa 2 tile 2 và 3 cần 4 move.

Khi đó ta tính h(st1,st2) += 4;

Hàm đánh giá inventedHeuristic là hàm đánh giá thấp,nghĩa là nếu một state có giá trị h(state) càng nhỏ thì khoảng cách phải di chuyển từ state đó tới goal và từ start tới đó thấp .Do đó nó là admissible,A* là thuật toán tối ưu.

.Trong thuật toán A*,mỗi khi lấy phần tử state temp từ danh sách arrState-danh sách các đỉnh chờ để phát triển,ta kiểm tra luôn giá trị của nó xem có phải bằng Goal state không ngay,và sau khi phát triển ta cũng kiểm tra ngay các state con có phải Goal state không,nếu có thoát luôn vòng lặp.

Để sắp xếp lại danh sách arrState theo hàm đánh giá f(temp).Trước khi add các state vừa được sinh ra vào arrState ta check lại xem trong danh sách arrState có tồn tại chưa để tránh trùng lặp. ta sử dụng lệnh sort sau khi đã add tất cả các state vừa sinh ra vào arrState.

Như vậy ta đã tránh được việc lặp lại các state.

III.Chạy chương trình:

Để run được chương trình ta cần để file nPuzzle.inp cùng folder với file n-Puzzle.jar.

Biên dịch file jar bằng lệnh cmd : java –jar n-Puzzle.jar.

Sau khi chạy: sẽ có lựa chọn hàm heuristic để chạy,chương trình sẽ in ra output là các state và số đỉnh visited và số đỉnh generate ,thời gian chạy.
