ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

ĐỀ TÀI: AI CHƠI CỜ CARO

Môn: Nhập môn trí tuệ nhân tạo

GVHD: TS. Trần Thế Hùng

Nhóm thực hiện: Nhóm 4

Nguyễn Văn Minh	20215092
Hoàng Mạnh Kiên	20215068
Nguyễn Trung Đạt	20215029
Nguyễn Hồ Tấn Tài	20215134
Hoàng Trường Nam	20215096

Hà Nội, tháng 6 năm 2024

Mục lục

I. Phân công nhiệm vụ 3
II. Giới thiệu đề tài 3
1. Tổng quan đề tài 3
2. Game cờ Caro 3
3. Mô tả thuật toán Minimax 4
III. Mô tả chương trình 4
1. Thiết kế giải thuật Minimax 4
2. Thiết kế giải thuật cắt tỉa alpha - beta 6
3. Interface 6

I. Phân công nhiệm vụ

Họ và tên	Phân công nhiệm vụ	Phần trăm công việc
Nguyễn Văn Minh	Lập trình thuật toán minimax	30%
Hoàng Trường Nam	Điều chỉnh thuật toán cho luật chơi 4 nước. Làm báo cáo	20%
Nguyễn Trung Đạt	Thiết kế, lập trình giao diện bằng Turtle. Làm slide thuyết trình	20%
Hoàng Mạnh Kiên	Lập trình kiểm tra trạng thái thắng, thua. Lập trình cho bàn cờ 3x3	15%
Nguyễn Hồ Tấn Tài	Tìm hiểu và thiết kế thuật toán cắt tỉa anpla-beta	15%

II. Giới thiệu đề tài

1. Tổng quan đề tài

Chủ đề báo cáo của chúng em là trò chơi Cờ Caro được viết bằng Python, Cờ Caro là một trò chơi chiến thuật giữa hai người chơi một bàn cờ hình vuông. Chương trình AI cờ caro này không chỉ mang lại sự giải trí mà còn là một công cụ học tập hữu ích cho những ai quan tâm đến thuật toán và trí tuệ nhân tạo.

2. Game cờ Caro

+ Bàn cờ: bàn cờ caro thường là một hình vuông với kích thước tuỳ ý

+ Quân cờ: hai người chơi sử dụng hai loại quân cờ khác nhau, thường là X, O hoặc Trắng, Đen

+ Luật chơi: hai người chơi lần lượt đặt quân cờ của mình vào các ô trống trên bàn cờ. Người chơi đầu tiên thường là người đi quân X. Mục tiêu là tạo thành một đường thẳng (ngang, dọc, hoặc chéo) gồm 5 quân cờ liên tiếp của mình trước đối phương.

+ Kết thúc ván cờ: người chơi đầu tiên tạo được đường 5 quân cờ liên tiếp sẽ thắng. Nếu toàn bộ bàn cờ được lấp đầy mà không có người chơi nào tạo được đường 5 quân

cờ liên tiếp, ván cờ sẽ kết thúc với kết quả hòa.

+ Các biến thể và quy tắc bổ sung:

Cờ caro 4: Trong một số biến thể, người chơi chỉ cần tạo thành một đường 4 quân cờ liên tiếp để thắng.

Cờ caro 3x3 (Tic tac toe): Người chơi cần tạo thành 3 nước cờ liên tiếp trên bàn cờ 3x3 để thắng

3. Mô tả thuật toán Minimax

+ Thuật toán Minimax là một thuật toán tìm kiếm và ra quyết định phổ biến trong các trò chơi hai người chơi, đặc biệt là các trò chơi có tổng bằng không như cờ vua, cờ vây, tic-tac-toe (cờ caro), v.v. Thuật toán này giúp xác định nước đi tối ưu bằng cách giả định rằng đối thủ sẽ luôn chơi nước đi tốt nhất có thể để tối ưu hóa cơ hội chiến thắng của mình. Dưới đây là một mô tả chi tiết về thuật toán Minimax:

+ Cơ bản về thuật toán Minimax:

- Mục tiêu: Tìm nước đi tối ưu bằng cách tối đa hóa giá trị nhỏ nhất có thể đạt được.

- Nguyên tắc hoạt động: Thuật toán xem xét tất cả các nước đi có thể của người chơi và đối thủ, xây dựng một cây tìm kiếm trong đó các nút lá đại diện cho các trạng thái cuối cùng của trò chơi (thắng, thua, hòa), và các nút trung gian đại diện cho các trạng thái trung gian của trò chơi.

III. Mô tả chương trình

1. Thiết kế giải thuật Minimax

- Các bước thực hiện chọn nước đi tối ưu:

1. Tính toán giới hạn của của những ô đã đánh, mở rộng về các phía 4 nước. Sau đó chỉ kiểm tra những ôn chưa đánh trong phạm vi đó để đưa ra nước đánh của máy
2. Chọn 1 ô đã lọc ra trong bước 1 để kiểm tra. Lấy dữ liệu vào 4 mảng tương ứng với 4 đường (đường dọc, đường ngang, đường chéo thuận, đường chéo nghịch) chứa điểm đang xét

3. Duyệt 5 nước liên tiếp trong mỗi đường, nếu trong 5 nước đó có nước đánh của người chơi thì trả về -1 (với ý nghĩa là không thể tồn tại nước thắng của AI ở vị trí đó). Nếu không thì trả về số nước mà AI đã đi trong 5 nước đó
4. Tính các trường hợp có thể thắng luôn và các trường hợp bắt buộc phải chặn
5. Dựa vào kết quả đã duyệt. Lập công thức tính điểm cho nước đi đó
6. So sánh điểm số của các nước đi và lựa chọn nước đi có điểm số cao nhất

- **Các hàm sử dụng**

+ **forward(x, y, dx, dy, len**)

Mô tả: hàm tịnh tiến một điểm A(x, y) qua một vector độ dài len, hướng (dx, dy)

Giá trị trả về là một tuple (x2,y2) = (x,y) + length*(dx,dy). Nếu (x2, y2) nằm ngoài đồ thị thì tịnh tiến ngược lại theo hướng (-dx, -dy) để đưa trở lại đồ thị.

+ **get_score_sum(score_dict)**

Mô tả: đầu vào là một dictionary chứa các danh sách các nước đi 2 liên tiếp, 3 liên tiếp, 4 liên tiếp trên tất cả các hàng. Sau đó tính tổng lại bên X/ O có bao nhiêu nước 2 liên tiếp, 3 liên tiếp, 4 liên tiếp

+ **score_section(begin_point, dx, dy, end_point, bw)**

Trả về một list với mỗi phần tử đại diện cho số điểm của 5 ô liên tiếp từ begin_point đến end_point

+ **score_point(x, y, bw)**

Trả về điểm số của 4 đoạn thẳng xoay quan điểm (x, y), đoạn thẳng theo hàng dọc, hàng ngang, hàng chéo thuận, hàng chéo nghịch

+ **win_situation(score_sum)**

Tìm kiếm các nước đi buộc phải chặn như người chơi có nước thắng luôn, có 4 nước liên tiếp trống 2 đầu. Hay có 2 nước 3 hoặc 1 nước 3, 1 nước 4 trống 2 đầu

+ **AI_calc_score(x, y)**

Giả sử AI tấn công ở vị trí (x, y), sau đó tính số điểm lợi thế bên AI đạt được. Sau đó lại giả sử người chơi tấn công ở vị trí đó, nếu nước đó giúp người chơi có nhiều điểm

thì phải chặn

+ **get_expect_move()**

  Từ những ô đã đánh, lấy toạ độ x_min -> x_max, y_min -> y_max của hình chữ nhật bao quanh các ô đã đánh. Sau đó mở rộng phạm vi hình chữ nhật rộng hơn 4 ô mỗi chiều. Từ hình chữ nhật thu được lấy ra m

+ **best_move()**

  Trả lại ô có điểm số cao nhất trong những ô được đề xuất từ hàm get_expect_move()

2. Thiết kế giải thuật cắt tỉa alpha - beta

+ Hàm alpha_beta_pruning

Nếu độ sâu tìm kiếm là 0 hoặc đã có người thắng, hàm sẽ trả về giá trị đánh giá của bảng hiện tại thông qua hàm evaluate_board. Đồng thời duyệt tất cả các nước có thể đi được trên bảng

+ Hàm evaluate_board

Đánh giá bảng hiện tại bằng cách tính điểm cho cả người chơi và AI sử dụng hàm score_all. Tính tổng điểm của người chơi và AI từ điểm số riêng lẻ. Tính tổng điểm cuối cùng của người chơi và AI bằng cách nhân số lượng các chuỗi quân liên tiếp (1, 2, 3, 4, 5) với các trọng số tương ứng và trả về sự chênh lệch giữa tổng điểm của AI và người chơi

+ Hàm best_move_alpha_beta

best_score được khởi tạo là âm vô cực để tìm giá trị lớn nhất; move được khởi tạo là None. Sau đó duyệt qua tất cả các nước đi tiềm năng, tính giá trị của nước đi bằng hàm alpha_beta_pruning. Cập nhật nước đi tốt nhất nếu tìm được giá trị lớn hơn best_score.

3. Interface

Giao diện người chơi gồm có bảng(có thể thay đổi kích cỡ), hai ký hiệu cho 2 người chơi (đen và trắng)

- Giải thích các hàm

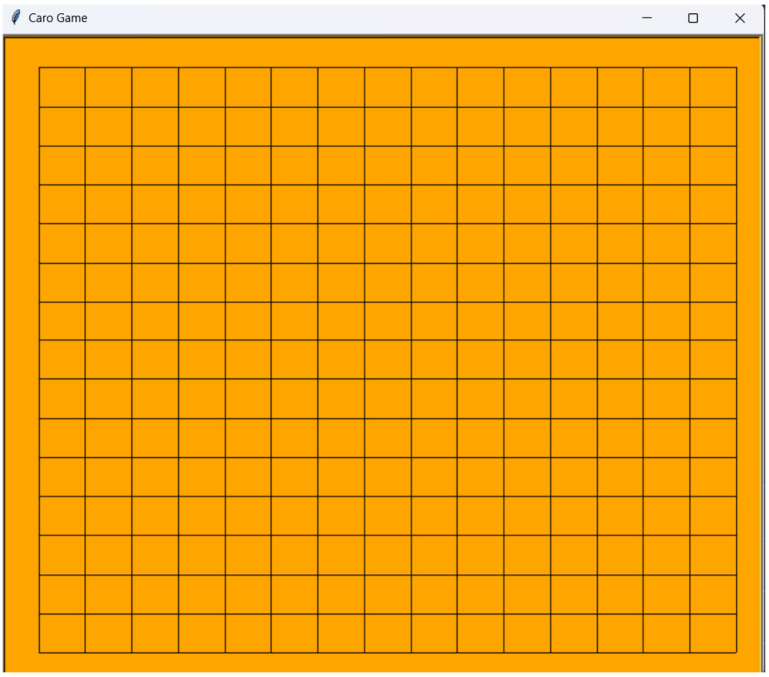
+ **display_winner(winner_text):** Hiển thị tên người chiến thắng lên màn hình

+ **click(x, y):** hàm xử lí click chuột:

+ **draw_stone(x, y, bw):** hàm vẽ quân cờ của người chơi lên bàn cờ

+ **init():** Hàm khởi tạo trò chơi

+ Giao diện khi bắt đầu trò chơi:



+ Hiển thị người chiến thắng (phiên bản caro 4):