TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

BÁO CÁO MÔN HỌC TRÍ TUỆ NHÂN TẠO

Đề tài: Trò chơi cờ CARO

Sinh viên thực hiện: Công Việt Hùng
Nguyễn Thị Diệu Mơ
Thoeun Rathana

NỘI DUNG

1. Tổng quan về trò chơi cờ caro
2. Phương pháp xây dựng

1. Tổng quan về trò chơi cờ caro

❖ Có nguồn gốc từ Nhật Bản

❖ Ở mỗi nước có tên gọi khác nhau như Nhật Bản: gomoku, Anh: Connect5, Trung Quốc: Guziqi,…

❖ Luật chơi:

- Có 2 người chơi đối kháng trực tiếp
- Người chơi đánh lần lượt hai quân X, O trên bàn cờ
- Người thắng là người đánh được 5 quân cờ cùng kiểu trên một hang ngang/dọc/chéo mà không bị chặn 2 đầu

2. Phương pháp xây dựng

2.1. Xây dựng hàm lượng giá tính điểm cho các nước đi

2.2. Ý tưởng xây dựng hàm lượng giá

2.3. Giải thuật Minimax

2.4. Giải thuật cắt tỉa Alpha – Beta để tìm nước đi tối ưu

2.5. Bảng so sánh số node của giải thuật MINIMAX và ALPHA-BETA

2.1. Xây dựng hàm lượng giá

❖ Mỗi nước đi sẽ có 2 loại điểm: điểm tấn công và điểm phòng ngự

- Điểm tấn công đại diện cho khả năng tạo các quân cờ liền nhau theo hàng ngang/dọc/chéo. Số quân cờ liền nhau tạo được càng nhiều thì điểm sẽ càng cao.

- Điểm phòng ngự đại diện cho khả năng ngăn chặn đối phương tạo các quân cờ liền nhau. Số quân cờ của đối phương bị chặn càng nhiều thì điểm sẽ càng cao.

2.1. Xây dựng hàm lượng giá

- Mỗi loại điểm (tấn công/phòng ngự) sẽ bằng tổng điểm của 4 hàm lượng giá tương ứng với 4 hướng đi (ngang/dọc/chéo xuôi/chéo ngược) của loại điểm đo.

- Điểm số sau cùng đánh giá nước đi đó được tính bằng điểm lớn hơn trong 2 loại điểm là tấn công và phòng ngự.

2.2. Ý tưởng xây dựng

- Đối với mảng điểm tấn công, xét theo 4 hướng là ngang, dọc, chéo xuôi, chéo ngược, mỗi hướng xét theo 2 phía.

- Với mỗi hướng, duyệt lần lượt từng ô xuất phát từ ô đang xét theo 4 hướng trên.

- Mỗi lần gặp 1 quân ta thì tăng chỉ số tấn công thêm 1 đơn vị (lúc đầu là 0)

2.2. Ý tưởng xây dựng

- Nếu gặp quân địch hoặc nước chưa có ai đánh thì dừng, tăng chỉ số phòng ngự thêm 1 đơn vị và chuyển sang xét phía còn lại.
- Nếu gặp quân ta thì tiếp tục tăng chỉ số tấn công.
- Nếu gặp quân địch thêm 1 lần nữa (chặn 2 đầu) thì chỉ số tấn công quay về 0.
- Nếu gặp ô trống thì thoát vòng lặp.

2.3. Giải thuật Minimax

- Hai người chơi trong game được đại diện là MAX và MIN
  - MAX đại diện cho người chơi luôn muốn chiến thắng và cố gắng tối ưu hóa ưu thế của mình
  - MIN đại diện cho người chơi cố gắng cho người MAX giành số điểm càng thấp càng tốt
- Giải thuật Minimax thể hiện bằng cách định trị các Node trên cây trò chơi
  - Node thuộc lớp MAX thì gán cho nó giá trị lớn nhất của con Node đó
  - Node thuộc lớp MIN thì gán cho nó giá trị nhỏ nhất của con Node đó
  - Từ các giá trị này người chơi sẽ lựa chọn cho mình nước đi tiếp theo hợp lý nhất

2.3. Giải thuật Minimax

```
function MINIMAX-DECISION(state) returns an action
    v ← MAX-VALUE(state)
    return the action in SUCCESSORS(state) with value v

function MAX-VALUE(state) returns a utility value
    if TERMINAL-TEST(state) then return UTILITY(state)
    v ← −∞
    for a, s in SUCCESSORS(state) do
        v ← MAX(v, MIN-VALUE(s))
    return v

function MIN-VALUE(state) returns a utility value
    if TERMINAL-TEST(state) then return UTILITY(state)
    v ← ∞
    for a, s in SUCCESSORS(state) do
        v ← MIN(v, MAX-VALUE(s))
    return v
```

2.4. Giải thuật cắt tỉa Alpha-Beta

---

function ALPHA-BETA-SEARCH(*state*) **returns** *an action*
  inputs: *state*, current state in game

  $v \leftarrow$ MAX-VALUE($state, -\infty, +\infty$)
  return the *action* in SUCCESSORS(*state*) with value v

---

function MAX-VALUE($state, \alpha, \beta$) **returns** *a utility value*
  inputs: *state*, current state in game
    α, the value of the best alternative for MAX along the path to *state*
    β, the value of the best alternative for MIN along the path to *state*

  if TERMINAL-TEST(*state*) **then return** UTILITY(*state*)
  $v \leftarrow -\infty$
  for a, s in SUCCESSORS(*state*) **do**
    $v \leftarrow$ MAX(v, MIN-VALUE(s, α, β))
    if $v \geq \beta$ **then return** v
    $\alpha \leftarrow$ MAX(α, v)
  return v

2.4. Giải thuật cắt tỉa Alpha-Beta

function MIN-VALUE($state, \alpha, \beta$) **returns** *a utility value*
  inputs: *state*, current state in game
    α, the value of the best alternative for MAX along the path to *state*
    β, the value of the best alternative for MIN along the path to *state*
  if TERMINAL-TEST(*state*) **then return** UTILITY(*state*)
  $v \leftarrow +\infty$
  for a, s in SUCCESSORS(*state*) **do**
    $v \leftarrow$ MIN(v, MAX-VALUE(s, α, β))
    if $v \leq \alpha$ **then return** v
    $\beta \leftarrow$ MIN(β, v)
  return v

2.4. Áp dụng vào tìm kiếm nước đi cho máy

- State: gồm một mảng 2 chiều đại diện cho bàn cờ, phần tử có giá trị bằng 0 khi ô trống, bằng 1 nếu đó là nước đi của máy tính, bằng 2 nếu đó là nước đi của người chơi, và một biến depth đại diện cho độ sâu muốn xét.

```
1 reference
private OCo maxValue(OCo[,] _MangOCo, long depth, long alpha, long beta)...
2 references
private OCo minValue(OCo[,] _MangOCo, long depth, long alpha, long beta)...
```

2.4. Áp dụng vào tìm kiếm nước đi cho máy

- Terminal-Test: hàm Max-Value và Min-Value sẽ trả về giá trị luôn nếu depth = 0 nghĩa là đã xét tới độ sâu mong muốn. Giá trị trả về là một đối tượng ô cờ, cụ thể là ô cờ có điểm cao nhất với state nhập vào.

```
1 reference
private OCo maxValue(OCo[,] _MangOCo, long depth, long alpha, long beta)
{
    if (depth == 0)
    {
        OCo oCoMax = new OCo();
        oCoMax = TimKiemNuocDi();
        return oCoMax;
    }
```

2.4. Áp dụng vào tìm kiếm nước đi cho máy

- Successors(state): với state nhập vào, successors(state) là mảng 2 chiều đại diện cho bàn cờ và có thêm 1 nước đi đang xét ngoài những nước đã có trong state nhập vào.

```
for (int i = 0; i < _BanCo.SoDong; i++)
{
    for (int j = 0; j < _BanCo.SoCot; j++)
    {
        if (_MangOCo[i, j].SoHuu == 0)
        {
            long currScore = TinhDiem(i, j);
            _MangOCo[i, j].SoHuu = 1;
```

2.4. Áp dụng vào tìm kiếm nước đi cho máy

❖ Điểm của mỗi state:

❖ Đối với MAX:

- Điểm(state) = Điểm(nước đi) – Điểm(nước đi tốt nhất của MIN sẽ đi nếu MAX đi nước đó)

❖ Đối với MIN:

- Điểm(state) = - Điểm(nước đi) + Điểm(nước đi tốt nhất của MAX sẽ đi nếu MIN đi nước đó)

2.5. Bảng so sánh số node cần duyệt

Độ sâu	Alpha – Beta Pruning	Minimax
2	795	157608
3	15761	62097552
4	312041	Vô cùng lớn

THANK YOU!