TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

KHOA CÔNG NGHỆ THÔNG TIN



**BÁO CÁO MÔNTRÍ TUỆ NHÂN TẠO**

Đào Văn Cường – Đỗ Phú Cường



Hà NỘi, 20/10/2018

# **Lời mở đầu**

Ngày nay khoa học công nghệ đang ngày càng phát triển, ứng dụng trong thực tế đời sống hàng ngày rất nhiều. Chúng ta thường hay được nghe trên báo đài thời sự rằng những robot đang dần thay thế con người trong công việc. Những máy móc hiện đại đang dần được áp dụng hầu hết trong lĩnh vực sản xuất cũng như đời sống. Và chúng ta thường nghe đến trí tuệ nhận tạo (AI) được áp dụng vào trong máy móc. Vậy trí tuệ nhân tạo là gì? Chúng là những dòng code được do con người viết ra gắn vào các vi xử lý của máy móc nhằm học hỏi từ con người, làm cho các máy móc xử lý chính xác hơn, nhanh hơn.

Sau đây là những dòng code được do nhóm chúng em viết ra để xử lý với các quân cờ. Với mục đích quân cờ sẽ tự động đi, hoặc có thể đánh thắng con người.

Do kiến thức còn hạn chế, nên còn nhiều thiếu sót trong code cũng như các phần liên quan. Mong thầy cùng các bạn cùng đóng góp ý kiến để sản phẩm hoàn thiện hơn.

Nội dung

[**Lời mở đầu** 1](#_Toc528174419)

[**I. Giới thiệu trò chơi** 3](#_Toc528174420)

[***1.Khai cuộc*** 3](#_Toc528174421)

[***2.Luật chơi*** 3](#_Toc528174422)

[**3.Kết thúc** 4](#_Toc528174423)

[**4.Một vài chiến thuật** 4](#_Toc528174424)

[**II. Xây dựng trò chơi** 5](#_Toc528174425)

[**1.** **Công cụ, ngôn ngữ lập trình** 5](#_Toc528174426)

[**2.** **Ý tưởng xây dựng** 5](#_Toc528174427)

[2.1. Xây dựng bàn cờ, quân cờ 5](#_Toc528174428)

[2.2. Xây dựng lớp nước đi 6](#_Toc528174429)

[2.3. Xây dựng sự kiện đi quân. 6](#_Toc528174430)

[2.4. Kiểm tra điều kiện thắng (CheckWin) 7](#_Toc528174431)

[**II. Xây dựng giải thuật cho AI** 8](#_Toc528174432)

[**1.** **Giải thuật MiniMax** 8](#_Toc528174433)

[1.1. Ý tưởng: 8](#_Toc528174434)

[**2.** **Ý tưởng xây dựng giải thuật** 8](#_Toc528174435)

[2.1. Xây dựng phương thức đánh giá trạng thái bàn cờ (getPoint) 8](#_Toc528174436)

[2.2. Xây dựng giải thuật MiniMax 10](#_Toc528174437)

[**3.** **Áp dụng giải thuật xây dựng nước đi cho AI** 11](#_Toc528174438)

[3.1. Xây dựng phương thức lấy nước đi tốt nhất findBestMove() 11](#_Toc528174439)

[3.2. Xây dựng phương thức MiniMax tương ứng với mỗi nước đi 12](#_Toc528174440)

[3.3. Tạo nước đi cho AI 12](#_Toc528174441)

# **I. Giới thiệu trò chơi**

## ***1.Khai cuộc***

- Bàn cờ khai cuộc: Các quân cờ của mỗi bên được xếp kín vào các ô màu và để trống 2 hàng giữa của bàn cờ. Người chơi luân phiên đi nước khai cuộc trong mỗi ván cờ liên tiếp.

## ***2.Luật chơi***

\* Đi quân: Các quân cờ đi chéo một nước, chỉ được đi tiến. Người chơi thay phiên nhau lần lượt đi quân.

\* Ăn quân: Quân đối phương ở vị trí tiến và chéo 1 nước với quân mình, đằng sau quân đối phương là ô trống. Khi đi nước đi ăn quân, quân đối phương bị nhấc ra khỏi bàn cờ. Quân mình được đặt vào vị trí ô trống sau ô của quân đối phương vừa bị ăn.

\* Phong vương: Khi quân của người chơi tiến được đến hàng cuối cùng của đối phương. Quân cờ sẽ được phong vương và biến đổi thành hình dạng khác (giống 2 quân cờ xếp chồng). Lúc này, quân phong vương sẽ đi chéo ăn chéo, lùi hoặc tiến..

## **3.Kết thúc**

\*Một bên bị ăn hết quân hoặc khi đến lượt đánh, người chơi không còn nước đi, sẽ kết thúc ván và người chơi bị xử thua cuộc. \*Cờ Checkers áp dụng luật tính điểm Ello để đánh giá trình độ người chơi và thăng cấp.

## **4.Một vài chiến thuật**

\* Điểm biên: Một quân cờ ở điểm biên sẽ không bị ăn. \*Phòng bị: Chiến thuật khi tiến quân luôn đi kèm với một quân đứng chéo ngay phía sau để tránh bị ăn. Làm tiền đề tiến công tiêu giệt quân đối phương và phong vương.

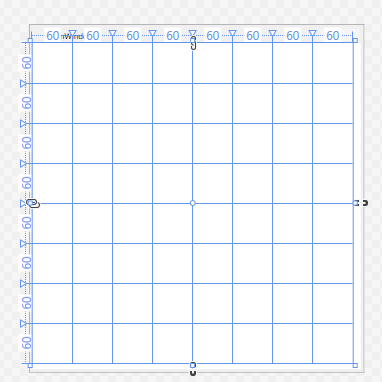
# **II. Xây dựng trò chơi**

1. **Công cụ, ngôn ngữ lập trình**

* Sử dụng ngôn ngữ Csharp
* Công cụ Visual Studio 2017

1. **Ý tưởng xây dựng**

Chúng ta thấy mỗi quân cờ hay mỗi ô cờ là một item. Vậy thì ý tưởng chúng ta sẽ xây dựng một ma trận 8x8 chứa các quân cờ và ô cờ đó. Ở đây, sử dụng một Grid là lưới để chứa các item, chiều dài chiều rộng của nó đều là 60.

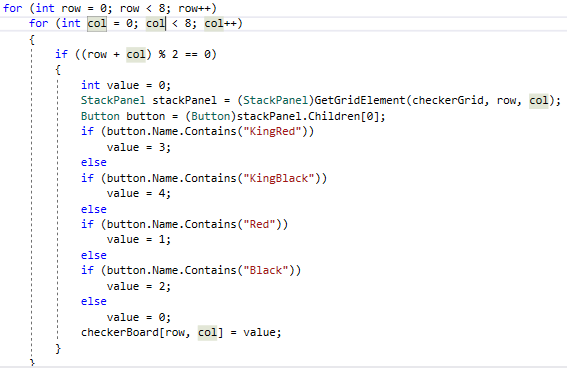


* 1. Xây dựng bàn cờ, quân cờ
* Xây dựng bàn cờ 8x8 mỗi ô cờ là một row được phân biệt bởi tọa độ (x,y).
* Trước tiên chúng ta sẽ set button cho mỗi row này. Mỗi button này sẽ được phân biệt với nhau với tên của nó để biết được đâu là ô cờ, đâu là quân cờ.
* Các ô cờ sẽ được set ảnh khác nhau để phân biệt ô đen và ô trắng.
* Xây dựng các quân cờ: mỗi quân cờ là một button có icon phân biệt cho từng quân.
* Từ vị trí của button trong Grid chúng ta có thể biết được tọa độ của nó.

và chúng ta sẽ set lấy ra tọa độ của quân cờ cho vào mảng 2 chiều.

Các giá trị của nó:

* - 1 ô không thể đi.
* 0 ô trống
* 1 quân đỏ
* 2 quân đen
* 3 vua đỏ
* 4 vua đen



* 1. Xây dựng lớp nước đi
* Tạo class Move:
* Thuộc tính:

+ piece1 : Lưu trữ tọa độ quân được chọn để đi.

+ piece2: Lưu trữ tọa độ ô mà quân muốn đi tới.

* Phương thức:

+ isAdjacent : Kiểm tra 2 vị trí có phải liền kề nhau hay không (đi quân)

+checkJump : Kiểm tra 2 vị trí có thể nhảy tới hay không (ăn quân)

Như đã nói ở trên chúng ta sẽ có các số trong mảng để biết được đâu là quân đen, đâu là quân trắng, đâu là nước đi được và đâu là nước không đi được.

Chúng ta sẽ check tọa độ của những nước kế tiếp mà người chơi muốn đi:

* Nếu những ô là -1 thì không thể đi được.
* Trước quân của mình lại có một quân của mình thì không thể đi được
* Nếu trước quân mình có quân đối thủ thì chúng ta sẽ check nếu sau đó là một ô trống thì có thể đi được, nếu sau đó là quân đối thủ thì không thể đi được
* Nếu bấm ra ngoài bàn cờ thì cũng sẽ không đi được
* Chúng ta cũng check đối với các quân vua. Vua thì có thể đi lùi….
  1. Xây dựng sự kiện đi quân.

Tiếp theo sẽ xử kiện đi quân của người chơi. Mỗi lần người chơi ấn vào quân cờ và ấn vào vị trí đi tiếp theo thì sẽ check xem có thể đi hay không cũng như có thể phong vua hay không …

\* Thuộc tính Move currentMove thể hiện nước đi:

Sự kiện chọn quân (click button):

- If (currentMove = null) then (khởi tạo currentMove)

- If (thuộc tính vị trí 1 của currentMove = null) then (vị trí 1 := tọa độ ô được chọn)

- If (thuộc tính vị trí 2 của currentMove = null) then (vị trí 2 := tọa độ ô được chọn)

\* Kiểm tra có phải nước đi hợp lệ hay không (CheckMove)

Sử dụng 2 phương thức của lớp Move:

- If (isAdjacent() = true ) then (Cho phép đi quân)

- If (checkJump() = false)then (Cho phép ăn quân)

\* Thuộc tính Turn thể hiện lượt đi

- Mỗi lần thực hiện một nước đi hợp lệ thì đặt lại lượt đi.

\* Kiểm tra phong vương

- If (thuộc tính vị trí 2 của currentMove = 0) then (Phong vua đen)

- If (thuộc tính vị trí 2 của currentMove = 7) then (Phong vua đỏ)

### 2.4. Kiểm tra điều kiện thắng (CheckWin)

Cuối cùng sẽ xử lý sự kiện thắng. Nếu đối thủ không có nước đi nữa, hoặc hết quân thì sẽ thắng.

Chúng ta sẽ lấy ra tất cả quân cờ và check xem các quân cờ đó có thể đi tiếp hay không. Nếu không có quân cờ nào có thể đi tiếp hoặc hết quân thì sẽ thắng.

\* Phương thức getListMoves() liệt kê các nước có thể đi

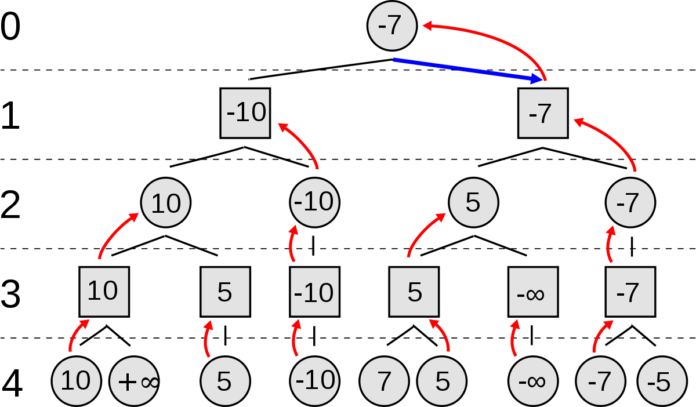
- If (getListMoves(Red) = 0) then (Đen thắng)

- If (getListMoves(Black) = 0) then (Đỏ thắng)

# **II. Xây dựng giải thuật cho AI**

1. **Giải thuật MiniMax**
   1. Ý tưởng:

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 còn 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



1. **Ý tưởng xây dựng giải thuật**
   1. Xây dựng phương thức đánh giá trạng thái bàn cờ (getPoint)

* Các tiêu chí đánh giá:
* Vị trí quân trên bàn cờ:

+ Quân đỏ ở hàng (row) càng cao càng có lợi thế và ngược lại.

+ Cùng một hang vị trí tại biên sẽ có ưu thế hơn các vị trí khác.

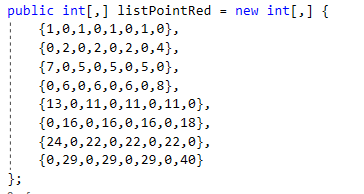
* Khả năng có thể phong vương:

+ Quân có khả năng phong vương sẽ được ưu tiên hơn so với các nước đi quân khác.

* Khả năng ăn quân đối thủ:

+ Quân có khả năng ăn quân của đối phương thì được ưu tiên.

Như vậy nhóm đã có mảng trọng số như sau:



Trọng số sẽ tăng từ đầu bàn cờ tới cuối bàn cờ của máy.

Ở những cạnh trong số sẽ cao hơn trong cùng một hàng.

Trong số sẽ cao nhất ở hàng cuối cùng.

* 1. Xây dựng giải thuật MiniMax

*function MiniMax(checkBoard): integer;*

*value, best: integer;*

*begin*

*if ( checkBoard là nút lá ) then return getPoint(checkBoard)*

*else*

*begin*

*{Khởi tạo giá trị tạm cho best}*

*if checkBoard là nút MAX then*

*best:= -INFINITY*

*else best:= INFINITY;*

*{hàm getListMoves() sinh ra mọi nước đi từ thế cờ checkBoard}*

*getListMoves(checkBoard);*

*{Xét tất cả các con của checkBoard, mỗi lần xác định được giá trị của một nút con, ta phải đặt lại giá trị tạm value. Khi đã xét hết tất cả các con thì value là giá trị của n}*

*while (còn lấy được một nước đi move) do*

*begin*

*checkBoard:= Tính thế cờ mới nhờ đi move;*

*value = Minimax (checkBoard);*

*if   checkBoard là nút MAX then*

*if  (value > best) then best := value;*

*if  checkBoard là nút MIN then*

*if (value < best) then best:= value;*

*end;*

*Minimax :=best;*

*end;*

*end;*

1. **Áp dụng giải thuật xây dựng nước đi cho AI**
   1. Xây dựng phương thức lấy nước đi tốt nhất findBestMove()

public Move findbestMove()

{

Move best = new Move();

int max = -10000;

foreach (Move move in baseCheckerBoard.getListMoves())

{

CheckerBoard checkerBoard1 = new CheckerBoard(baseCheckerBoard);

MakeMove(move, checkerBoard1);

if (max <= MiniMax(checkerBoard1, 1))

{

max = MiniMax(checkerBoard1, 1);

best = move;

}

}

return best;

}

* 1. Xây dựng phương thức MiniMax tương ứng với mỗi nước đi

public int MiniMax(CheckerBoard checkerBoard, int depth)

{

int value = 0, best;

if (depth >= 1 || checkerBoard.getListMoves() == null)

return checkerBoard.getPoint();

else

{

best = 0;

if (checkerBoard.getTeam() == "Red")

best = -10000;

else

best = 10000;

foreach (Move move in checkerBoard.getListMoves())

{

CheckerBoard checkerBoard1 = new CheckerBoard(checkerBoard);

MakeMove(move, checkerBoard1);

int depth2 = depth + 1;

value = MiniMax(checkerBoard1, depth2);

if (checkerBoard.getTeam() == "Red")

if (value >= best)

best = value;

if (checkerBoard.getTeam() == "Black")

if (value <= best)

best = value;

}

return best;

}

}

* 1. Tạo nước đi cho AI
* Sau mỗi lượt đi của người chơi thì tạo nước đi cho AI

private void MakeAI()

{

if (turn == "Red")

{

Move move = new Move();

CheckerBoard checkerBoard = new

CheckerBoard(CheckersGrid, currentMove, turn);

CheckerAI checkerAI = new CheckerAI(checkerBoard);

move = checkerAI.findbestMove();

if (move != null)

{

currentMove = move;

MakeMove(CheckMove());

}

}

}

**Lời kết**

Như vậy, nhìn vào game chúng ta có đã biết AI là gì và nó hoạt động như thế nào. Muốn AI chạy tốt thì chúng ta cần có các đánh giá trọng số chính xác hơn.

Phân tích kĩ càng với nhiều chiến thuật khác nhau.

Do kiến thức có hạn, nên game chạy chưa được mượt và chính xác. Quân cờ còn đi chưa chính xác như mong muốn ban đầu.

Cảm ơn thầy Nguyễn Quốc Tuấn đã đóng góp những ý kiến cũng như đã dạy chi tiết thuật toán này nên nhóm mới có sản phầm này.

Tài Liệu Tham Khảo