TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI

KHOA CÔNG NGHỆ THÔNG TIN



**BÁO CÁO MÔN**

**TRÍ TUỆ NHÂN TẠO**

**Sinh viên thực hiện : Đỗ Phú Cường , Đào Văn Cường**

**Lớp : Công nghệ thông tin 2 – K56**

**Giáo viên hướng dẫn : Nguyễn Quốc Tuấn**

*Hà Nội, ngày 20 tháng 10 năm 2018*

MỤC LỤC

**Nội dung** **Trang**

[I. GIỚI THIỆU TRÒ CHƠI 3](#_Toc468714336)

[*1. Khai cuộc* 3](#_Toc468714337)

[*2. Luật chơi* 3](#_Toc468714338)

[*3. Điều kiện thắng* 3](#_Toc468714338)

[II. XÂY DỰNG TRÒ CHƠI 3](#_Toc468714339)

[*1. Công cụ, ngôn ngữ* 3](#_Toc468714340)

[*2. Ý tưởng xây dựng* 4](#_Toc468714341)

[*. Khâu xúc bốc* 4](#_Toc468714342)

[*4. Khâu vận tải* 5](#_Toc468714343)

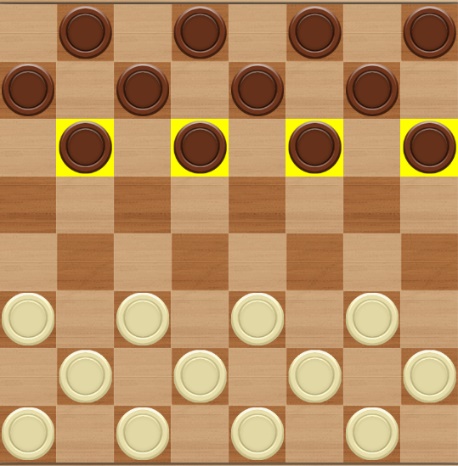
[*5. Khâu chế biến sản phẩm* 5](#_Toc468714344)

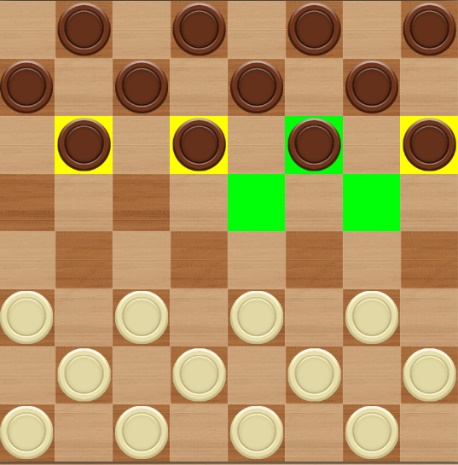
[*6. Khâu bán hàng* 6](#_Toc468714345)

[II. XÂY DỰNG TRÒ CHƠI 3](#_Toc468714339)

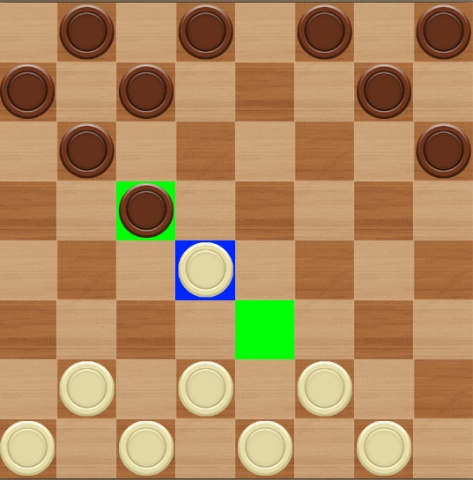
**I. Giới thiệu trò chơi**

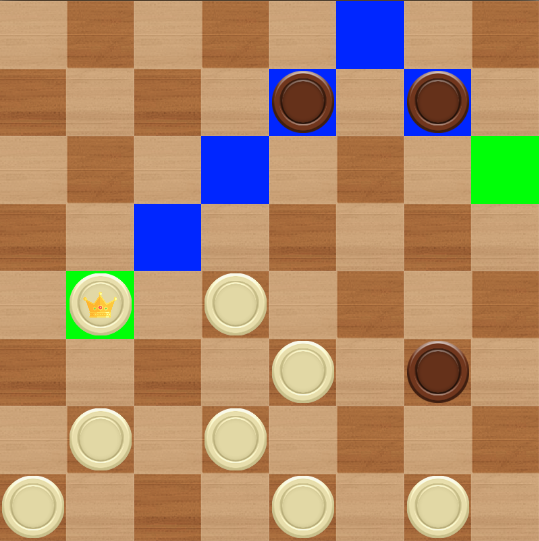
***1.Khai cuộc***

******- 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**
   1. Xây dựng bàn cờ, quân cờ

* Xây dựng bàn cờ 8x8 mỗi ô cờ là một panel được phân biệt bởi tọa độ (x,y).
* Các ô cờ được đặt màu đen, trắng caro
* Xây dựng các quân cờ: mỗi quân cờ là một button có icon phân biệt cho từng quân
  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)

* 1. Xây dựng sự kiện đi quân.

\* 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)

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

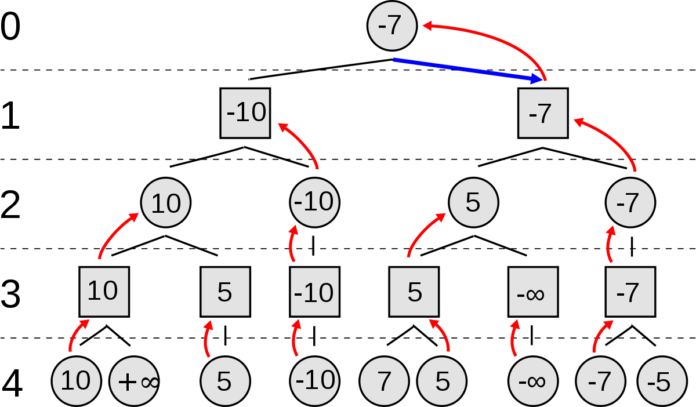
- If (getListMoves(Red) = 0) then (Đen thắng)

- 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.

* 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());

}

}

}