

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



ĐỒ ÁN MÔN HỌC TRÍ TUỆ NHÂN TẠO
ĐỀ TÀI: XÂY DỰNG GAME CỜ VUA

NHÓM SINH VIÊN:

- | | |
|----------------------------|-----------------|
| 1. Đào Mạnh Khá | 20142275 |
| 2. Trần Gia Nghĩa | 20143180 |
| 3. Nguyễn Văn Thắng | 20144223 |
| 4. Nguyễn Ngọc Hải | 20141382 |

GIẢNG VIÊN HƯỚNG DẪN:

TS. Nguyễn Nhật Quang

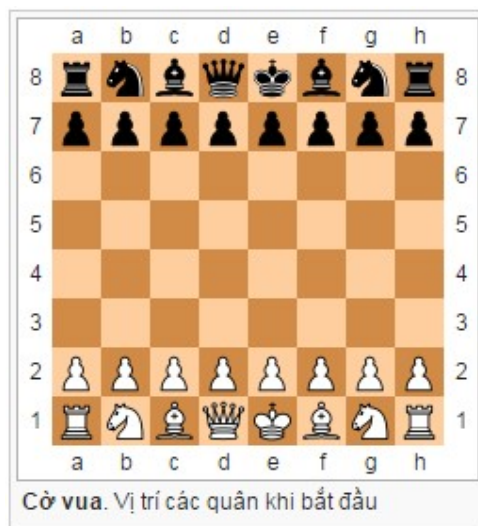
MỤC LỤC

CHƯƠNG 1: GIỚI THIỆU VÀ MÔ TẢ BÀI TOÁN.....	3
<i>I, Đề tài.....</i>	<i>3</i>
<i>II, Mô tả bài toán.....</i>	<i>4</i>
CHƯƠNG 2: PHƯƠNG PHÁP GIẢI QUYẾT BÀI TOÁN.....	5
<i>I, Game tree (cây biểu diễn trò chơi).....</i>	<i>5</i>
<i>II, Giải thuật Minimax.....</i>	<i>6</i>
<i>III, Giải thuật Alpha-beta Prunning.....</i>	<i>8</i>
CHƯƠNG 3: CHỨC NĂNG VÀ CÁCH SỬ DỤNG HỆ THỐNG....	10
CHƯƠNG 4: CÁC PHƯƠNG PHÁP VÀ CÔNG NGHỆ ĐƯỢC KHAİ THÁC VÀ SỬ DỤNG.....	13
<i>I, Hàm đánh giá thế cờ.....</i>	<i>13</i>
<i>II, Cách tính PieceValue cho mỗi quân cờ.....</i>	<i>14</i>
<i>III, Cách tính PieceSquare cho mỗi quân cờ.....</i>	<i>15</i>
<i>IV, Xác định AddPoint cho thế cờ.....</i>	<i>18</i>
CHƯƠNG 5: CÁC VẤN ĐỀ GẶP PHẢI VÀ HƯỚNG GIẢI QUYẾT.....	21
CHƯƠNG 6: MỘT SỐ ĐỀ XUẤT PHÁT TRIỂN VÀ CẢI TIẾN HỆ THỐNG TRONG TƯƠNG LAI.....	23
<i>1, Về các chức năng của chương trình.....</i>	<i>23</i>
<i>2, Về giao diện chương trình.....</i>	<i>23</i>
<i>3, Về thuật toán sử dụng.....</i>	<i>23</i>
TÀI LIỆU THAM KHẢO.....	24

CHƯƠNG 1: GIỚI THIỆU VÀ MÔ TẢ BÀI TOÁN



I, ĐỀ TÀI:

- ✓ Xây dựng chương trình game cờ vua.
- ✓ Về trò chơi cờ vua:
 - + Cờ vua là một trò chơi ở trên bàn và là một môn thể thao trí tuệ đối kháng giữa 2 người chơi.
 - + Bàn cờ vua gồm 8 hàng (đánh số từ 1 đến 8) và 8 cột (đánh các chữ cái từ a đến h), tạo thành 64 ô vuông với các màu đậm nhạt xen kẽ nhau.
 - + Mỗi bên thi đấu sẽ bắt đầu ván cờ bằng 16 quân cờ được sắp xếp sẵn vị trí ban đầu (như hình vẽ) và sẽ lần lượt đi các quân của mình sau khi đối phương đã đi xong một nước.



- + Danh sách các quân cờ mỗi bên bao gồm:

- ❖ 8 con Tốt (Pawn)  
- ❖ 2 con Mã (Knight)  
- ❖ 2 con Tượng (Bishop)  
- ❖ 2 con Xe (Rook)  
- ❖ 1 con Hậu (Queen)  

❖ 1 con Vua (King)  

+ Cờ vua là trò chơi có tổng bằng 0 (Zero-sum-game). Việc chiến thắng của một bên (+1) sẽ làm cho bên kia thua cuộc (-1). Không bao giờ có trường hợp cả 2 bên đều cùng thắng hoặc cùng thua. Hai bên được cho là hòa cờ nếu trên bàn cờ chỉ còn lại 2 con vua.

+ Game cờ vua với AI là một hình thức chơi game cờ vua với máy tính. Khi người chơi điều khiển một bên cờ, máy tính sẽ tự động điều khiển phe còn lại.

II, MÔ TẢ BÀI TOÁN:

1, Mục đích:

- ✓ Tạo ra một trò chơi đối kháng giữa hai đối thủ là con người với máy tính (AI) hoặc máy tính với chính nó.

2, Yêu cầu:

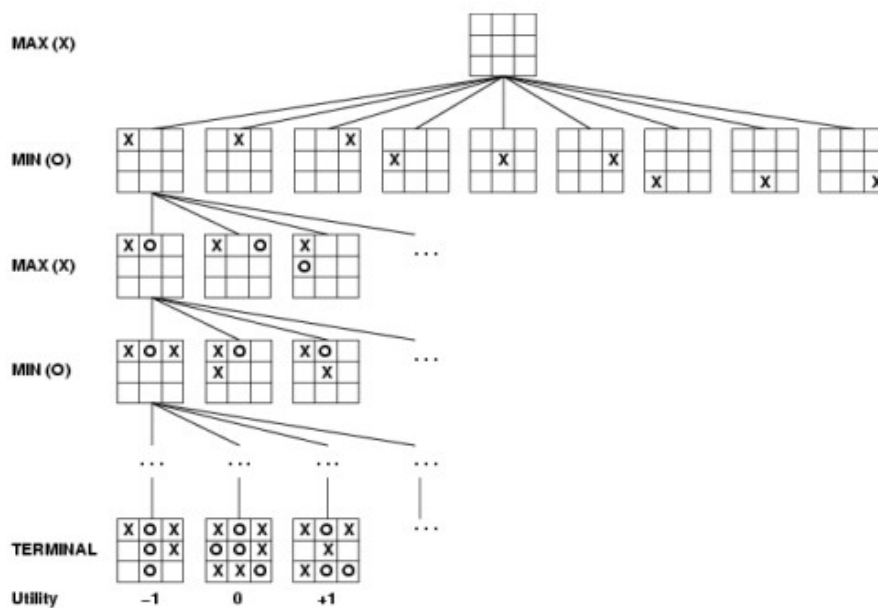
- ✓ Đưa ra các thuật toán và phương pháp giải quyết cho bài toán thực tế đặt ra, sao cho khả năng để máy tính thắng là lớn nhất.
- ✓ Hoàn thiện chương trình, đảm bảo có đồ họa rõ ràng và đầy đủ các chức năng của một trò chơi cờ vua căn bản.
- ✓ Đảm bảo chương trình không xảy ra lỗi trong quá trình chạy, AI thông minh, có thời gian suy nghĩ và ra nước đi hợp lý, theo đúng luật cờ vua.

CHƯƠNG 2: PHƯƠNG PHÁP GIẢI QUYẾT BÀI TOÁN

Cờ vua là một trò chơi đối kháng giữa hai tác tử, thuật toán hiệu quả nhất để giải quyết bài toán này là các thuật toán tìm kiếm thông minh Minimax. Với yêu cầu chung về chi phí của một PC game, thuật toán cần có thời gian thực hiện trong một khoảng chấp nhận được. Đối với trò chơi cờ vua có hệ số phân nhánh cao (~35), thì khi độ sâu của cây tìm kiếm càng lớn thì thời gian chạy của máy tính càng dài (có thể lên đến hàng phút, thậm chí hàng chục phút). Do đó, để đạt được kết quả hiệu quả hơn, cần cài đặt một thuật toán cải tiến là cắt tỉa alpha-beta (Alpha-beta Pruning) là một việc cần thiết.

I, GAME TREE (CÂY BIỂU DIỄN TRÒ CHƠI):

Xem xét vấn đề thực hiện một chương trình máy tính để chơi một trò chơi đối kháng giữa 2 người chơi và chỉ có một người thắng duy nhất. Khi người chơi này thắng thì người chơi còn lại là người thua và không có bất kỳ sự hợp tác nào giữa 2 người chơi này. Với các loại trò chơi có các bàn cờ cổ điển như cờ vua (Chess), cờ ca-rô (Tic Tac Toe), ... thì ta có thể mô hình hóa các trạng thái của chúng bằng cách sử dụng cây biểu diễn trò chơi (Game tree).



Hình 2.1. Cây biểu diễn trò chơi cờ ca-rô(Nguồn: Slide AI thầy Nguyễn Nhật Quang)

Trong ví dụ trên, mỗi nút đại diện cho một trạng thái của bàn cờ. Nút gốc biểu diễn cho trạng thái bắt đầu trò chơi và các nút con của mỗi nút đại diện cho các trạng thái có thể ứng với trạng thái tại nút cha khi thực hiện các nước đi tiếp theo. Ở đây, người chơi thứ nhất đánh dấu 'X', người chơi thứ hai đánh dấu 'O'. Cả 2 đấu thủ sẽ cố gắng đi những nước đi để đạt được điểm tuyệt đối lớn nhất. Đối với người chơi thứ nhất, anh ta sẽ tìm những nước đi khiến điểm của mình càng dương càng tốt hay điểm của đối thủ bớt âm đi, còn người chơi thứ hai thì ngược lại. Trạng thái cuối cùng (các nút lá) được đánh giá với ba điểm (-1, 0, +1). Trong đó, +1 thể hiện chiến thắng của 'X', -1 thể hiện chiến thắng của 'O' và 0 nếu 2 bên hòa cờ.

II, GIẢI THUẬT MINIMAX:

- ✓ Một chiến lược tối ưu là một chuỗi các nước đi giúp đưa đến trạng thái đích mong muốn (Vd: chiến thắng). Và một thuật toán để tính toán nước đi tốt nhất là thuật toán Minimax.
- ✓ Trong 2 người chơi thì một người là MAX, người còn lại là MIN. MAX đại diện cho đối thủ quyết giành thắng lợi hay cố gắng tối đa hóa lợi thế của mình. Ngược lại, MIN là đối thủ cố gắng tối thiểu hóa điểm số của MAX.
- ✓ MAX và MIN cùng sử dụng lượng thông tin như nhau và đều có khả năng hiểu biết và xử lý như nhau.
- ✓ Chiến lược của MAX bị ảnh hưởng vào các nước đi của MIN và ngược lại.

```

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 ←  $-\infty$ 
    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 ←  $\infty$ 
    for a, s in SUCCESSORS(state) do
        v ← MIN(v, MAX-VALUE(s))
    return v

```

Hình 2.2. Giải thuật Minimax

(Nguồn: Slide AI thầy Nguyễn Nhật Quang)

- ✓ Sử dụng giá trị của các nút lá để định trị các nút trên cây trò chơi:
 - + Nút thuộc lớp MAX thì gán cho nó giá trị lớn nhất trong các nút con của nó.
 - + Nút thuộc lớp MIN thì gán cho nó giá trị nhỏ nhất trong các nút con của nó.
- ✓ Các bước trong giải thuật Minimax:
 - + Nếu đạt đến giới hạn tìm kiếm (lần tới nút lá), tính giá trị của thế cờ hiện tại ứng với người chơi đó, ghi nhớ kết quả.
 - + Nếu mức đang xét là của MIN, áp dụng thuật toán cho các nút con của nó, ghi nhớ kết quả nhỏ nhất.
 - + Nếu mức đang xét là của MAX, áp dụng thuật toán cho các nút con của nó, ghi nhớ kết quả lớn nhất.
- ✓ Áp dụng đến độ sâu cố định: Tính trước n nước đi.
- ✓ Kết quả truyền về nút nguồn là heuristic cho giá trị tốt nhất, thông báo kết quả thắng-thua.
- ✓ Sau mỗi nước đi, thông tin trên bàn cờ sẽ thay đổi, do đó sẽ xác định một lớp cây MIN-MAX mới và các ước lượng n nước đi mới.

- ✓ Thuật toán Minimax thăm toàn bộ cây trò chơi bằng việc dùng chiến lược tìm kiếm theo chiều sâu nên độ phức tạp của thuật toán này tương ứng trực tiếp với kích thước không gian tìm kiếm. Do đó, để tiết kiệm thời gian và tăng tối đa độ sâu tìm kiếm trên cây trò chơi, ta sử dụng giải thuật cải tiến là Alpha-beta Prunning.

III, GIẢI THUẬT ALPHA-BETA PRUNNING:

- ❖ Đây là thuật toán cải tiến của thuật toán Minimax với tư tưởng: “Nếu biết điều đó thực sự tồi thì đừng mất thời gian tìm hiểu nó sẽ tồi tệ đến đâu.”
- ❖ Làm giảm số nút cần thiết để không lãng phí thời gian tìm kiếm những nút trong những nước đi gây bất lợi.
- ❖ Thêm vào 2 tham số α và β , cho biết các giá trị nằm ngoài $[\alpha; \beta]$ là những giá trị không cần xem xét đến nữa.
- ❖ Giá trị α liên quan đến nút MAX, có khuynh hướng không bao giờ giảm.
- ❖ Giá trị β liên quan đến nút MIN, có khuynh hướng không bao giờ tăng.
- ❖ Hai luật cắt tỉa:
 - + Quá trình tìm kiếm có thể kết thúc bên dưới một nút MIN nào có giá trị β nhỏ hơn hoặc bằng giá trị α của một nút cha MAX bất kỳ của nó.
 - + Quá trình tìm kiếm có thể kết thúc bên dưới một nút MAX nào có giá trị α lớn hơn hoặc bằng giá trị β của một nút cha MIN bất kỳ của nó.
- ❖ Nếu mức đang xét là đỉnh (gốc cây), đặt $\alpha = -\infty$ và $\beta = +\infty$.
- ❖ Nếu như đạt tới giới hạn tìm kiếm, tính giá trị tĩnh của thế cờ hiện tại ứng với người chơi đó. Ghi lại kết quả.
- ❖ Nếu như mức đang xét là của người chơi cực tiểu (MIN), thực hiện các công việc sau cho đến khi tất cả các con của nó đã được xét với thủ tục $\alpha - \beta$ hoặc cho đến khi $\alpha \geq \beta$:
 - + Áp dụng thủ tục Alpha-beta với giá trị α và β hiện tại cho một con. Ghi nhớ kết quả.

+ So sánh giá trị ghi nhớ này với giá trị β , nếu giá trị đó nhỏ hơn thì đặt β là giá trị mới này. Ghi nhớ lại β .

```

function ALPHA-BETA-SEARCH(state) returns an action
  inputs: state, current state in game
   $v \leftarrow \text{MAX-VALUE}(\text{state}, -\infty, +\infty)$ 
  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
            $\alpha$ , the value of the best alternative for MAX along the path to state
            $\beta$ , 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
     $v \leftarrow \text{MAX}(v, \text{MIN-VALUE}(s, \alpha, \beta))$ 
    if  $v \geq \beta$  then return  $v$ 
     $\alpha \leftarrow \text{MAX}(\alpha, v)$ 
  return  $v$ 



---


function MIN-VALUE(state,  $\alpha$ ,  $\beta$ ) returns a utility value
  inputs: state, current state in game
            $\alpha$ , the value of the best alternative for MAX along the path to state
            $\beta$ , 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
     $v \leftarrow \text{MIN}(v, \text{MAX-VALUE}(s, \alpha, \beta))$ 
    if  $v \leq \alpha$  then return  $v$ 
     $\beta \leftarrow \text{MIN}(\beta, v)$ 
  return  $v$ 

```

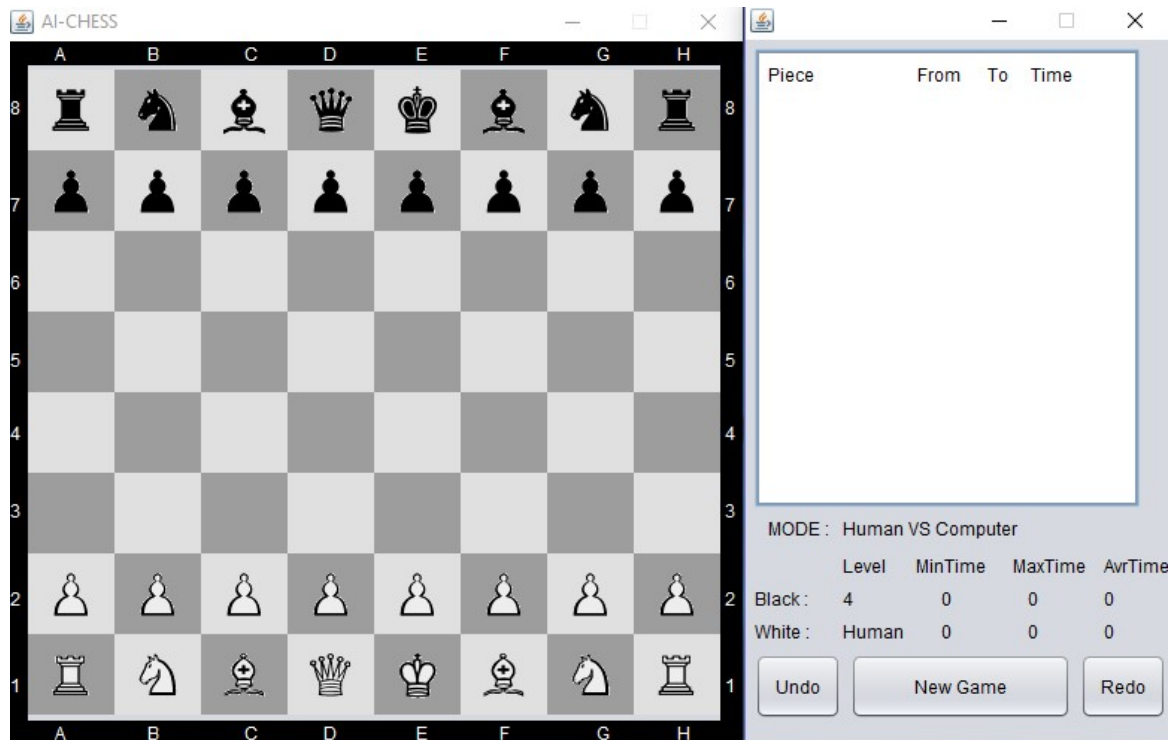
Hình 2.3. Giải thuật Alpha-beta Prunning
(Nguồn: Slide AI thầy Nguyễn Nhật Quang)

CHƯƠNG 3: CHỨC NĂNG VÀ CÁCH SỬ DỤNG HỆ THỐNG

1, Cài đặt và chạy chương trình:

- + Yêu cầu: Máy tính cài đặt sẵn J2SE 1.8 trở lên.
- + Trong thư mục 'Executable_File', thực hiện các công việc sau:
 - Chạy trực tiếp bằng file AIChessProject.jar

2, Giao diện trò chơi:



- + Bàn cờ gồm các ô vuông đen và trắng xen kẽ nhau.
- + Mỗi bên sẽ xuất phát với 16 quân cờ được sắp xếp sẵn trên bàn cờ như hình.
- + Giao diện bên trái là giao diện trò chơi chính thức. Bên phải là phần giao diện phụ dùng để ghi lại và hiển thị các nước đi của cả 2 bên cũng như thời gian cần thiết để thực hiện nước đi đó.
- + Góc dưới cùng bên phải hiển thị chế độ chơi (giữa người với máy hoặc giữa máy với chính nó), cấp độ chơi cũng như thời gian nhỏ nhất, thời gian lớn nhất và thời gian trung bình của tất cả các nước đã đi.

+ Chức năng ‘Undo’ dùng để bỏ qua nước đi vừa chọn và quay lại trạng thái trước liền kề.

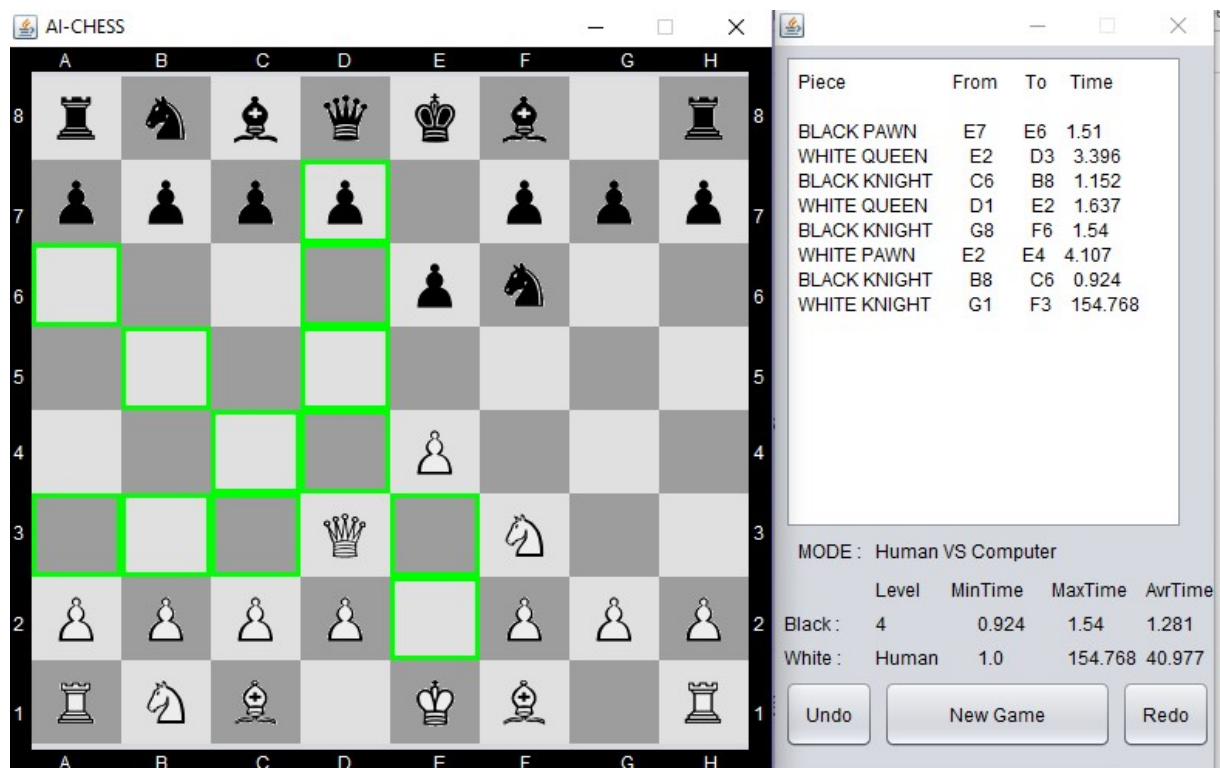
3, Khi bắt đầu chơi:

+ Tiến hành lựa chọn chế độ chơi (người với máy hoặc xem máy đấu với chính nó).

+ Tiếp theo lựa chọn level chơi và click nút ‘Play’. Trong chế độ người đấu với máy thì người được ưu tiên đi trước.

4, Cách chơi:

+ Đối với chế độ người đấu với máy, để di chuyển quân cờ nào, ta kích chuột vào quân cờ đó, khi đó trên bàn cờ sẽ hiển thị các ô cờ có viền màu xanh (ô được phép đi).



+ Nếu đối phương có quân cờ nằm trong ô đi được thì được phép ăn con cờ đó.

+ Khi con Tốt di chuyển tới hàng cuối cùng của đối phương thì sẽ được thăng cấp.

- + Nhập thành là trường hợp quân Xe (Rook) có thể nhảy qua quân Vua (King) của mình để đứng cạnh nó.
- + Mỗi bên sẽ lần lượt điều khiển các quân cờ của mình theo đúng luật cờ vua sau khi đối phương đã hoàn thành một nước đi.

5, *Game over:*

- + Lượt đi của người và máy xen kẽ nhau, cứ như vậy đi hết bàn cờ.
- + Nếu con Vua của bên nào không còn nước nào để đi thì bên đó thua cuộc. Cờ hòa khi 2 bên chỉ còn lại 2 con Vua.

CHƯƠNG 4: CÁC PHƯƠNG PHÁP VÀ CÔNG NGHỆ ĐƯỢC KHAI THÁC VÀ SỬ DỤNG

I, Hàm đánh giá thế cờ:

+ Lấy quân trắng làm người chơi MAX, hàm đánh giá thế cờ sẽ đưa ra con số mô tả mức độ có lợi của trạng thái bàn cờ đối với quân trắng. Quân đen làm người chơi MIN.

+ Hàm đánh giá thế cờ được tính thông qua các yếu tố sau :

PieceValue[i] : Giá trị cố định của mỗi quân cờ (Không phụ thuộc vào vị trí quân cờ cũng như màu sắc quân cờ là trắng hay đen mà chỉ phụ thuộc vào nó là quân cờ nào)

PieceSquare[i] : Điểm cộng thêm khi quân cờ ở một vị trí nào đó trên bàn cờ. Điểm này phụ thuộc vào vị trí mà quân cờ đó đang đứng trên bàn cờ và màu của nó(trắng hoặc đen)

AddPoint : Điểm cộng thêm hoặc trừ bớt cho thế cờ khi xác định được 3 yếu tố sau trên bàn cờ :

- 1 . Quân tốt xếp chồng (trừ điểm)
- 2 . Quân tốt bị cô lập (trừ điểm)
- 3 . Quân cờ có nước đi mở (cộng điểm)

Từ các yếu tố trên, ta lập công thức tính điểm cho mỗi thế cờ :

$$F(Position) = \sum(WhitePieceValue[i] + WhitePieceSquare[i] + WhiteAddPoint) - \sum(BlackPieceValue[i] + BlackPieceSquare[i] + BlackAddPoint)$$

Trong đó :

- + *WhitePieceSquare[i]*: là *PieceSquare[i]* của quân trắng;
- + *WhiteAddPoint*: điểm *AddPoint* của quân trắng;
- + *BlackPieceSquare[i]*: là *PieceSquare[i]* của quân đen;
- + *BlackAddPoint*: điểm *AddPoint* của quân đen.

II, Cách tính *PieceValue*[i] cho mỗi quân cờ:

+ Về cơ bản, ta đánh giá độ quan trọng và khả năng bao quát bàn cờ của các quân cờ như sau:

$$\text{Hậu} > \text{Xe} > \text{Tượng} > \text{Mã} > \text{Tốt}$$

- Vua không thể trao đổi với bất kỳ quân nào khác nên không xét đến và vua sẽ phải nhận giá trị lớn hơn rất nhiều so với các quân cờ khác.

- Trao đổi 1 quân Mã với 3 quân Tốt là ý tưởng tồi, do đó:

$$\text{Hậu} > \text{Xe} > \text{Tượng} > \text{Mã} > 3 * \text{Tốt}$$

- Trao đổi quân Xe lấy 6 quân Tốt làm bàn cờ có lợi thế nên:

$$\text{Hậu} > 6 * \text{Tốt} > \text{Xe} > \text{Tượng} > \text{Mã} > 3 * \text{Tốt} \quad (1)$$

+ Việc đổi 1 Xe lấy 1 Tượng và 1 Mã là có lợi nên: $\text{Tượng} + \text{Mã} > \text{Xe}$

- Kể cả khi phải mất thêm 1 con Tốt thì việc đổi này vẫn có lợi:

$$\text{Tượng} + \text{Mã} > \text{Xe} + \text{Tốt}$$

- Nhưng nếu phải mất 2 con Tốt và 1 con Xe trong một lần trao đổi thì khả năng phòng thủ sẽ giảm nhiều, vì vậy:

$$\text{Xe} + 2 * \text{Tốt} > \text{Tượng} + \text{Mã} > \text{Xe} + \text{Tốt}$$

- Từ 2 điều trên, ta có thể giả sử:

$$\text{Tượng} + \text{Mã} = \text{Xe} + 1.5 * \text{Tốt} \quad (2)$$

+ Đổi 1 Xe và 1 Tượng lấy 1 Hậu là có lợi:

$$\text{Xe} + \text{Tượng} < \text{Hậu}$$

$$\rightarrow 2 * \text{Hậu} > 2 * \text{Xe} + 2 * \text{Tượng} > 2 * \text{Xe} + \text{Tượng} + \text{Mã} > 2 * \text{Xe} + \text{Xe} + 1.5 * \text{Tốt}$$

$$\rightarrow 2 * \text{Hậu} + 4 * \text{Tốt} > 3 * \text{Xe} + 5.5 * \text{Tốt} = 4 * \text{Xe}$$

$$\rightarrow \text{Hậu} + 2 * \text{Tốt} > 2 * \text{Xe}$$

- Mặt khác, 2 con Xe luôn có lợi hơn 1 con Hậu, do đó:

$$\text{Hậu} + 2 * \text{Tốt} > 2 * \text{Xe} > \text{Hậu}$$

$$\text{Vậy: } \text{Hậu} + \text{Tốt} = 2 * \text{Xe} \quad (3)$$

+ Từ (1), (2) và (3) ta có hệ:

$$\text{Hậu} > 6 * \text{Tốt} > \text{Xe} > \text{Tượng} > \text{Mã} > 3 * \text{Tốt}$$

$$\text{Tượng} + \text{Mã} = \text{Xe} + 1.5 * \text{Tốt}$$

$$\text{Hậu} + \text{Tốt} = 2 * \text{Xe}$$

+ Xét $\text{Tốt} = 100$, ta có bộ giá trị thỏa mãn:

1, $\text{Tốt} = 100$

2, $\text{Mã} = 320$

3, $\text{Tượng} = 330$

4, $\text{Xe} = 500$

5, $\text{Hậu} = 900$

6, $\text{Vua} = 10000$

+ Thế cờ là một tập hợp các quân cờ với vị trí xác định trên bàn cờ.

+ Khi các quân cờ ở đúng vị trí thích hợp sẽ tạo nên những thế cờ có lợi lớn cho người chơi.

+ Mỗi quân cờ có cách di chuyển và khả năng bao quát bàn cờ khác nhau nên có điểm số sức mạnh được cộng thêm tương ứng với mỗi ô khác nhau.

+ Điểm số cộng thêm có thể âm khi vị trí của quân cờ làm che mắt quân cờ khác hoặc làm cho khả năng di chuyển của quân cờ bị hạn chế.

III, Cách tính *PieceSquare[i]* cho mỗi quân cờ:

Do bàn cờ và vị trí các quân cờ đều có tính đối xứng trục nên số điểm *PieceSquare[i]* của mỗi quân cờ cũng sẽ có tính đối xứng trục.

1, Mã:



+ Mã di chuyển theo đường chéo hình chữ nhật (2x3) khi không bị cản.

+ Mã càng mạnh khi càng ở gần trung tâm bàn cờ, do đó ta có:

-50	-40	-30	-30	-30	-30	-40	-50
-40	-20	0	5	5	0	-20	-40
-30	5	10	15	15	10	5	-30
-30	0	15	20	20	15	0	-30
-30	5	15	20	20	15	5	-30
-30	0	10	15	15	10	0	-30
-40	-20	0	0	0	0	-20	-40
-50	-40	-30	-30	-30	-30	-40	-50

Điểm *PieceSquare* cho quân mã trắng

-50	-40	-30	-30	-30	-30	-40	-50
-40	-20	0	5	5	0	-20	-40
-30	5	10	15	15	10	5	-30
-30	0	15	20	20	15	0	-30
-30	5	15	20	20	15	5	-30
-30	0	10	15	15	10	0	-30
-40	-20	0	0	0	0	-20	-40
-50	-40	-30	-30	-30	-30	-40	-50

Điểm *PieceSquare* cho quân mã đen

2, Tượng:



+ Tượng di chuyển theo đường chéo bàn cờ, bị cản nếu có quân cờ nào đó đứng trên đường di chuyển.

+ Tượng càng mạnh khi ở trung tâm bàn cờ, do đó:

-20	-10	-10	-10	-10	-10	-10	-20
-10	0	0	0	0	0	0	-10
-10	0	5	10	10	5	0	-10
-10	5	5	10	10	5	5	-10
-10	0	10	10	10	10	0	-10
-10	10	10	10	10	10	10	-10
-10	5	0	0	0	0	5	-10
-20	-10	-10	-10	-10	-10	-10	-20

Điểm *PieceSquare* cho quân tượng trắng

-20	-10	-10	-10	-10	-10	-10	-20
-10	5	0	0	0	0	5	-10
-10	10	10	10	10	10	10	-10
-10	0	10	10	10	10	0	-10
-10	5	5	10	10	5	5	-10
-10	0	5	10	10	5	0	-10
-10	0	0	0	0	0	0	-10
-20	-10	-10	-10	-10	-10	-10	-20

Điểm *PieceSquare* cho quân tượng đen

3, Xe:



+ Xe di chuyển theo đường ngang và dọc bàn cờ, bị cản nếu có quân nào đó đứng trên đường di chuyển.

+ Xe càng mạnh khi ở vị trí có thể tấn công Vua đối phương, do đó:

0	0	0	0	0	0	0	0
5	20	20	20	20	20	20	5
-5	0	0	0	0	0	0	-5
-5	0	0	0	0	0	0	-5
-5	0	0	0	0	0	0	-5
-5	0	0	0	0	0	0	-5
-5	0	0	0	0	0	0	-5
0	0	0	5	5	0	0	0

Điểm *PieceSquare* cho quân xe trắng

0	0	0	5	5	0	0	0
-5	0	0	0	0	0	0	-5
-5	0	0	0	0	0	0	-5
-5	0	0	0	0	0	0	-5
-5	0	0	0	0	0	0	-5
-5	0	0	0	0	0	0	-5
-5	0	0	0	0	0	0	-5
5	20	20	20	20	20	20	5
0	0	0	0	0	0	0	0

Điểm *PieceSquare* cho quân xe đen

4, Hậu:



- + Di chuyển của Hậu bằng di chuyển của Xe và Tượng cộng lại.
- + Hậu càng mạnh khi ở trung tâm bàn cờ, yếu khi ở các rìa và các góc của bàn cờ, do đó:

-20	-10	-10	-5	-5	-10	-10	-20
-10	0	0	0	0	0	0	-10
-10	0	5	5	5	5	0	-10
-5	0	5	5	5	5	0	-5
0	0	5	5	5	5	0	-5
-10	5	5	5	5	5	0	-10
-10	0	5	0	0	0	0	-10
-20	-10	-10	-5	-5	-10	-10	-20

Điểm *PieceSquare* cho quân hậu trắng

-20	-10	-10	-5	-5	-10	-10	-20
-10	0	5	0	0	0	0	-10
-10	5	5	5	5	5	0	-10
0	0	5	5	5	5	0	-5
0	0	5	5	5	5	0	-5
-10	0	5	5	5	5	0	-10
-10	0	0	0	0	0	0	-10
-20	-10	-10	-5	-5	-10	-10	-20

Điểm *PieceSquare* cho quân hậu đen

5, Vua:



- + Di chuyển của Vua giống di chuyển của Hậu nhưng chỉ có thể đi 1 ô trong 1 nước đi.
- + Vua cần ở những nơi khó có thể tấn công và tránh những vị trí dễ bị tấn công nhất là phía sân của địch, khuyến khích nhập thành, do đó:

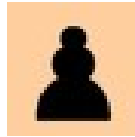
-30	-40	-40	-50	-50	-40	-40	-30
-30	-40	-40	-50	-50	-40	-40	-30
-30	-40	-40	-50	-50	-40	-40	-30
-30	-40	-40	-50	-50	-40	-40	-30
-20	-30	-30	-40	-40	-30	-30	-20
-10	-20	-20	-20	-20	-20	-20	-10
20	20	0	0	0	0	20	20
20	30	10	0	0	10	30	20

Điểm *PieceSquare* cho quân vua trắng

20	30	10	0	0	10	30	20
20	20	0	0	0	0	20	20
-10	-20	-20	-20	-20	-20	-20	-10
-20	-30	-30	-40	-40	-30	-30	-20
-30	-40	-40	-50	-50	-40	-40	-30
-30	-40	-40	-50	-50	-40	-40	-30
-30	-40	-40	-50	-50	-40	-40	-30
-30	-40	-40	-50	-50	-40	-40	-30

Điểm *PieceSquare* cho quân vua đen

6, Tốt:



+ Tốt đi thẳng, ăn chéo và càng mạnh khi càng tiến sâu sang phần sân đối phương.

+ Tốt là quân cờ có di chuyển chậm, đơn giản nhưng cách bố trí 8 quân tốt trên bàn cờ thích hợp luôn làm nên cục diện thế cờ, tốt không nên chắn ngay trước mặt vua và hậu. Tốt được đánh giá là linh hồn của trò chơi cờ vua.

0	0	0	0	0	0	0	0
50	50	50	50	50	50	50	50
10	10	20	30	30	20	10	10
5	5	10	25	25	10	5	5
0	0	0	20	20	0	0	0
5	-5	-10	0	0	-10	-5	5
5	10	10	-20	-20	10	10	5
0	0	0	0	0	0	0	0

Điểm *PieceSquare* cho quân tốt trắng

0	0	0	0	0	0	0	0
5	10	10	-20	-20	10	10	5
5	-5	-10	0	0	-10	-5	5
0	0	0	20	20	0	0	0
5	5	10	25	25	10	5	5
10	10	20	30	30	20	10	10
50	50	50	50	50	50	50	50
0	0	0	0	0	0	0	0

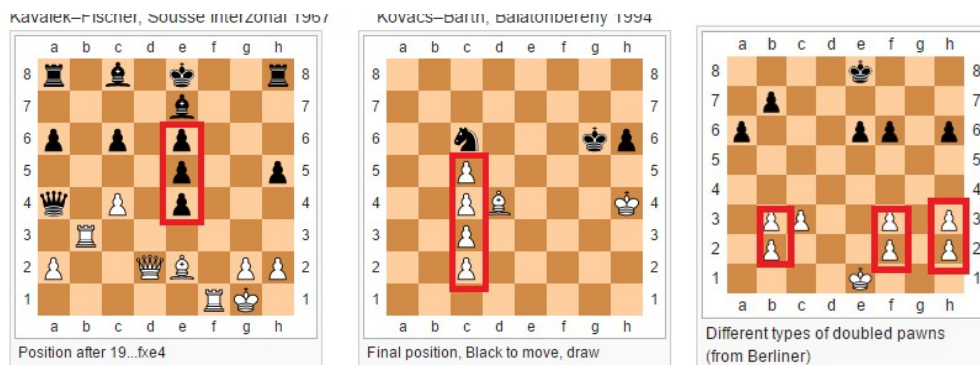
Điểm *PieceSquare* cho quân tốt đen

IV, Xác định *AddPoint* cho thế cờ:

1, Quân tốt xếp chồng:

Là trường hợp trong bàn cờ xuất hiện 2 quân tốt cùng màu đứng ở trên cùng một cột và chúng đứng ở hai ô liền nhau. Trường hợp này sẽ phải trừ điểm thế cờ bởi sự thiếu linh hoạt của quân tốt khi rơi vào trường hợp này. Có bao nhiêu cột chứa quân tốt xếp chồng thì trừ bấy nhiêu lần.

Ví dụ về tốt xếp chồng:

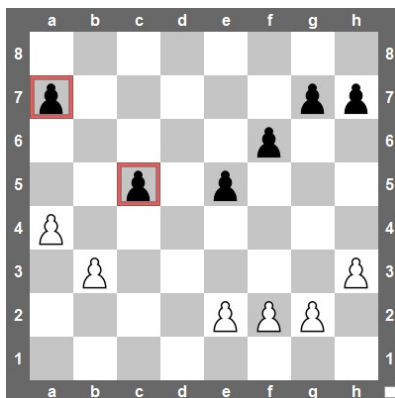


(Những vùng bị khoanh đỏ là trường hợp quân tốt bị xếp chồng)

2, Con tốt bị cô lập:

Là trường hợp quân tốt mà hai cột kề với quân tốt đó không có quân tốt nào cùng màu với nó. Trường hợp này sẽ phải trừ điểm thế cờ bởi quân tốt rơi vào tình huống này sẽ gây ra phòng thủ yếu cho thế cờ. Có bao nhiêu quân tốt bị cô lập thì trừ bấy nhiêu lần.

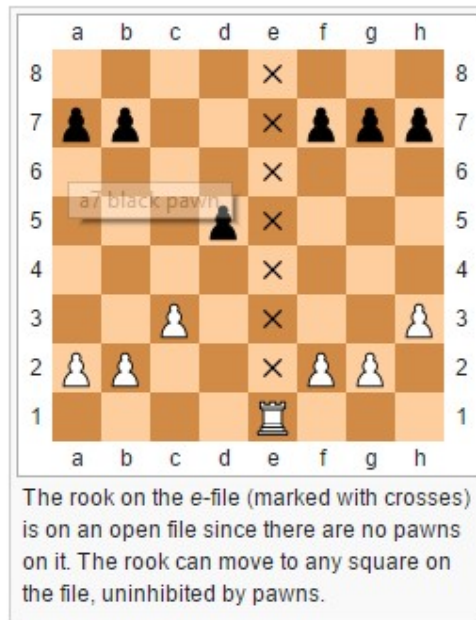
Ví dụ: Hai quân tốt bị tô đỏ là hai quân bị cô lập



3, Quân cờ có nước đi mở:

Là trường hợp một quân Xe hoặc Hậu ở vào một cột mà không có bất kỳ quân Tốt nào (không phân biệt cùng màu hay khác màu) trên cột đó. Trường hợp này sẽ cộng điểm cho thế cờ bởi tạo ra được con đường thuận lợi cho Xe hoặc Hậu di chuyển hoặc thâm nhập sâu vào trong quân địch.

Ví dụ :



Quân Xe trắng ở vào vị trí có nước đi mở

CHƯƠNG 5: CÁC VẤN ĐỀ GẶP PHẢI VÀ HƯỚNG GIẢI QUYẾT

Trong quá trình thực hiện đề tài này, nhóm em đã gặp phải một số vấn đề như sau:

1, Mô hình hóa bàn cờ một cách phù hợp để máy tính có thể đánh giá được các thế cờ.

- ✓ Hướng giải quyết: Nhóm em đã tiến hành tổ chức bàn cờ thành 64 ô đánh số từ 0 đến 63, sắp xếp thành bàn cờ 8x8. Những ô có quân cờ sẽ được lưu vào HashMap với key là chỉ số của ô, value là quân cờ trên ô đó. Ô nào không có quân cờ chiếm giữ thì không cần lưu.

2, Việc tính toán các ô có thể đi tới của một quân cờ nào đó khi quân cờ đó ở vị trí bất kỳ trên bàn cờ, đặc biệt là ở các vị trí biên giới của bàn cờ.

- ✓ Hướng giải quyết: Do các quân cờ có tất cả 6 loại (Tốt, Mã, Tượng, Xe, Hậu và Vua) nên với mỗi loại này, chúng em sẽ sử dụng một mảng để lưu lại các độ lệch tương đối nhỏ nhất so với vị trí hiện tại của quân cờ, rồi lấy vị trí hiện tại của quân cờ cộng với các độ lệch tương đối đó để ra được chỉ số của ô có thể đi tới, nếu chỉ số của ô phù hợp thì sẽ coi ô đó là vị trí hiện tại và tính các chỉ số tiếp theo.

3, Việc đánh giá vị trí của một quân cờ trên bàn cờ có thực sự tốt.

- ✓ Hướng giải quyết: Với mỗi loại quân cờ nêu trên, ứng với mỗi tọa độ trên bàn cờ thì sẽ có những điểm khác nhau.

4, Việc xây dựng hàm lượng giá. Do việc xây dựng một hàm đánh giá tốt phần lớn dựa vào kinh nghiệm chơi cờ của từng đấu thủ. Vì vậy lúc đầu nhóm em gặp một số vấn đề vì trong nhóm chưa có ai có kinh nghiệm chơi cờ vua và chơi thực sự tốt.

- ✓ Hướng giải quyết: Sau một thời gian tìm hiểu kỹ hơn về cờ vua, đồng thời tham khảo về cách xây dựng hàm lượng giá tại :

<https://chessprogramming.wikispaces.com>

Nhóm chúng em đã hiểu hơn về một số chiến thuật dùng trong cờ vua, từ đó đã xây dựng được một hàm đánh giá thế cờ cơ bản, đủ thông

minh để có thể chiến thắng người chơi có trình độ khá. Việc đánh giá một thế cờ (bao gồm các vị trí của tất cả các quân cờ trên bàn) có thực sự tốt đã được giải quyết bằng việc sử dụng điểm của thế cờ thông qua hàm lượng giá (bao gồm cả các điều kiện được thêm hoặc trừ điểm). Đồng thời, việc lựa chọn thế cờ nào là có lợi nhất để trở thành nước đi tiếp theo cũng được xử lý thông qua việc so sánh điểm của thế cờ kết hợp với 2 thuật toán là Minimax và cắt tỉa Alpha-beta để có thể chọn ra chính xác thế cờ có điểm số cao nhất hoặc thấp nhất.

5, Vấn đề giảm thiểu thời gian tính toán. Do giải thuật Minimax có chi phí quá lớn nếu duyệt tới độ sâu càng gần với trạng thái đích (các nút lá).

- ✓ Hướng giải quyết: Sử dụng giải thuật Alpha-beta Pruning. Tiến hành kiểm tra xem thế cờ tiếp theo có giống hệt với một thế cờ nào đó trước đó đã xét hay không, nếu trùng thì không đi sâu xuống các nhánh con của thế cờ đó nữa, tuy nhiên lại không thực sự cải thiện được nhiều do chi phí cho việc đối chiếu các bàn cờ với nhau là khá lớn.

6, Giao diện chương trình. Vì chưa có kinh nghiệm lập trình trò chơi, do đó ban đầu nhóm em chưa tìm ra được hướng giải quyết tốt nhất cho việc thiết kế giao diện và còn gặp khá nhiều lỗi.

- ✓ Hướng giải quyết: Sau khi tìm hiểu kỹ về ngôn ngữ lập trình Java, chúng em nhận thấy ngôn ngữ này có nhiều thư viện hỗ trợ tạo giao diện đồ họa với đầy đủ tiện ích để tạo được một giao diện tốt và cũng không quá phức tạp trong việc lập trình. Hơn nữa Java lại được hỗ trợ bởi một số trình biên dịch như Netbeans, Eclipse, ...có hỗ trợ kéo thả giao diện nên một số form cơ bản như form chọn chế độ chơi, form chọn level được thực hiện nhanh chóng hơn.

CHƯƠNG 6: MỘT SỐ ĐỀ XUẤT PHÁT TRIỂN VÀ CẢI TIẾN HỆ THỐNG TRONG TƯƠNG LAI

1, Về các chức năng của chương trình:

- Hoàn thiện chế độ máy tính (AI) đấu với máy tính, để máy tính có thể đi theo nhiều kịch bản khác nhau khi mà nước đi tiếp theo lại tạo ra thế cờ giống hệt thế cờ trước đó mặc dù vẫn còn nhiều nước khác có thể chọn lựa, nếu không sẽ tạo ra vòng lặp vô hạn.
→ Đề xuất giải pháp: Khi xảy ra việc trùng lặp thế cờ, cần loại bỏ nước đi của quân cờ gây ra sự lặp lại thế cờ, và chỉ tìm nước đi tốt nhất trên tập các nước đi còn lại.
- Thêm hình ảnh, bình luận cho AI, lồng âm thanh cho trò chơi thêm hấp dẫn.
- Viết thêm chức năng tính thời gian và tính điểm cho mỗi lượt suy nghĩ của người chơi. Nếu người chơi vượt quá số thời gian quy định mà vẫn chưa đưa ra được nước đi thì sẽ trừ điểm của người chơi.
- Thêm chế độ chơi Người – Người.

2, Về giao diện chương trình:

- Tạo sự chuyển động khi một quân cờ di chuyển từ ô này sang ô khác để người chơi tiện theo dõi.
- Khi chiếu tướng cần có dấu hiệu để người chơi biết được.
- Cải tiến lại giao diện, thiết kế lại hình ảnh các quân cờ cho đẹp hơn.

3, Về thuật toán sử dụng:

- Để có thể cắt được nhiều nhánh tìm kiếm hơn, cần phải bổ sung cách thức sắp xếp các nước đi tiếp theo có thể của một nước đi nào đó sao cho dễ dàng nhận được $\alpha \geq \beta$.
- Cải tiến hàm lượng giá để AI thông minh hơn, có nhiều kinh nghiệm hơn.

TÀI LIỆU THAM KHẢO

1, Slide Trí tuệ nhân tạo (thầy Nguyễn Nhật Quang)

2, <https://chessprogramming.wikispaces.com>