# BÁO CÁO BÀI TẬP LỚN
# ĐỀ TÀI: TRÒ CHƠI CỜ ĐAM

**Học phần: Trí tuệ nhân tạo**

**Giảng viên hướng dẫn: Nguyễn Nhật Quang**

**Nhóm: 33**

# MỤC LỤC

# CHƯƠNG I. GIỚI THIỆU VÀ MÔ TẢ BÀI TOÁN

## 1. Giới thiệu bài toán:

Cờ đam (hay draughts hoặc checkers) là một nhóm các trò chơi chiến lược trên bàn đối kháng dành cho hai người. Tên draughts của loại cờ này có nguồn gốc từ động từ mang nghĩa di chuyển.

Cờ đam là một trò chơi phổ biến ở nhiều nước phương Tây song mỗi khu vực lại có một hệ thống luật khác nhau. Các hình thức phổ biến nhất của cờ đam là cờ đam quốc tế, chơi trên một bàn cờ 10×10 và cờ đam Anh, còn được gọi là American checkers, chơi trên một bàn cờ 8×8. Tuy nhiên có rất nhiều biến thể khác chơi trên bảng 12×12.

Khái niệm cờ đam mà nhóm sử dụng xuyên suốt báo cáo nói riêng và đề tài nói chung là cờ đam phiên bản Anh (hay còn gọi là English draughts hoặc American checkers) với những đặc trưng nhất định (sẽ được trình bày ở các đề mục sau).

Trò chơi cờ đam là một bài toán tìm kiếm có đối thủ. Cụ thể hơn, đây là một trò chơi đối kháng hay trò chơi có tổng bằng không (zero-sum games). Bài toán được xem là tương đối phức tạp với khoảng 10^20 vị trí hợp lệ chỉ tính riêng trên bàn cờ 8x8.

Bài toán cờ đam là một bài toán được khá nhiều học giả và nhóm học giả quan tâm nghiên cứu. Những nỗ lực đầu tiên để xây dựng một chương trình máy tính chơi được loại cờ này bắt đầu từ những năm 50 của thế kỷ trước. Trong số đó, Chinook, một chương trình máy tính được phát triển bởi một nhóm tác giả thuộc trường Đại học Alberta.trong gần 18 năm (từ 1989 - 2007) là chương trình máy tính đầu tiên có được chiến thắng trước đại kiện tướng cờ đam quốc tế.

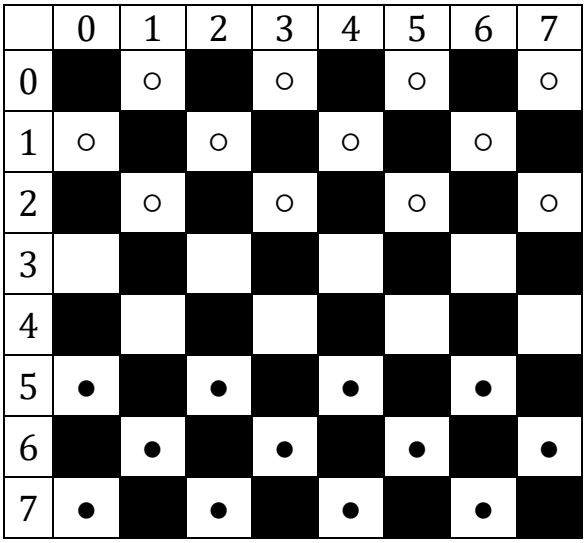
## 2. Mô tả bài toán:

### 2.1. Trạng thái bắt đầu (Initial state):

- Bàn cờ bao gồm 64 ô, đan xen sáng tối. Ô tối nằm ở góc trái phía dưới bàn cờ và các ô sáng tối lần lượt được xác định bởi nguyên tắc các ô chung cạnh thì khác màu nhau.

- Cờ được chơi bởi hai người chơi. Mỗi người chơi bắt đầu trò chơi với 12 quân khác màu. Thông thường, bên giữ quân đen được phép đi trước.

## 2.2. Một số luật cơ bản trong trò chơi cờ đam:



Hình 1. *Trạng thái ban đầu của trò chơi*

Có 4 kiểu nước đi:

- Nước đi thường của quân tốt:
  + Đi chéo, tiến về phía trước một ô bên trái hoặc bên phải.
  + Khi quân tốt chạm đến hàng xa nhất trên bàn cờ (được biết đến với tên king-row hay crown-head), nó trở thành vua. Trong báo cáo, trạng thái này được gọi là "phong".
- Nước đi thường của quân vua: Đi chéo, tiến hoặc lùi một ô về phía bên trái hoặc bên phải của ô hiện tại.
- Nước "ăn" của quân tốt:
  + Một quân có thể "ăn" được một quân đối thủ nếu như chúng ở cạnh nhau trên bàn cờ và phía sau quân bị "ăn" không chứa bất cứ quân cờ nào.
  + Các quân bị "ăn" bị xóa khỏi bàn cờ.
- Nước "ăn" của quân vua: Hoàn toàn tương tự nước "ăn" của quân tốt nhưng có thể thực hiện theo hai chiều tiến hoặc lùi.
- Tổng quan về nước ăn:
  + Hành động "ăn" còn được gọi là thực hiện một bước nhảy.

+ Nếu người chơi có thể thực hiện một bước nhảy thì họ bắt buộc phải thực hiện bước nhảy đó. Nếu như người chơi có cùng lúc nhiều bước nhảy khác nhau thì họ có thể chọn thực hiện một bước nhảy bất kỳ trong số đó.
+ Trò chơi cho phép thực hiện các bước nhảy liên tiếp nếu như sau khi nhảy, quân đang xét vẫn có thể thực hiện được các bước nhảy khác.

## 2.3. Kết quả của trò chơi:

- Trò chơi kết thúc với phần thắng thuộc về một bên nếu như bên còn lại không còn quân nào trên bàn cờ hoặc không còn nước đi nào hợp lệ.
- Trò chơi kết thúc với kết quả hòa nếu như hai bên đồng ý thỏa thuận. Trong trường hợp một bên không chấp nhận thỏa thuận này, trọng tài sẽ xác nhận kết quả hòa nếu:
  - Tại bất kỳ trạng thái nào của trò chơi, người chơi có thể chứng minh rằng nước đi kế tiếp của họ sẽ tạo ra vị trí tương tự lần thứ 3 trong suốt thời gian chơi.
  - Tại bất kỳ trạng thái nào của trò chơi, người chơi có thể chứng minh được cả hai trường hợp sau thỏa mãn đồng thời:
    + Không ai trong số cả hai bên có lợi thế có một quân tốt đang tiến về phía king-row trong 40 nước đi mà họ thực hiện trước đó.
    + Không có quân cờ nào bị xóa khỏi bàn cờ trong suốt 40 nước đi mà họ thực hiện trước đó.

## 2.4. Tổng quan về hàm đánh giá:

*a) Các thành phần của hàm đánh giá:*

Hàm đánh giá hay hàm ước lượng (evaluation function) được cấu tạo từ rất nhiều thành phần, một số thành phần cơ bản được đề cập đến ở nhiều tài liệu chuyên sâu và một số bài báo khoa học bao gồm:

- 8 thành phần đơn giản (simple features):
  (1) Số lượng quân tốt.
  (2) Số lượng quân vua.
  (3) Số lượng quân tốt "an toàn". (VD. Nằm kề các cạnh bàn cờ)
  (4) Số lượng quân vua "an toàn".

(5) Số lượng quân tốt có thể di chuyển.

(6) Số lượng quân vua có thể di chuyển. (Không tập trung vào ưu thế của bước nhảy so với nước đi thường)

(7) Tổng khoảng cách từ các quân tốt đến king-row.

(8) Số lượng ô chưa bị chiếm chỗ ở king-row.

- 11 thành phần bố cục (layout features):

  (9) Số lượng quân phòng thủ. (VD. Vua và tốt nằm ở hai hàng dưới cùng của bàn cờ)

  (10) Số lượng quân tốt tấn công. (VD. Những quân nằm ở 3 hàng trên cùng của bàn cờ)

  (11) Số lượng quân tốt nằm ở vị trí trung tâm. (VD. 8 ô ở giữa)

  (12) Số lượng quân vua nằm ở vị trí trung tâm.

  (13) Số lượng quân tốt nằm trên đường chéo chính.

  (14) Số lượng quân vua nằm trên đường chéo chính.

  (15) Số lượng quân tốt nằm ở trên 2 đường gần nhất song song với đường chéo chính.

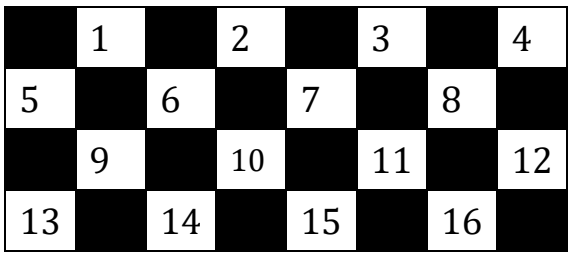
  (16) Số lượng quân vua nằm ở trên 2 đường gần nhất song song với đường chéo chính.

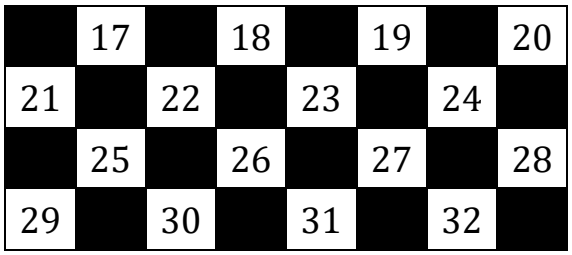
  (17) Số lượng quân tốt "cô đơn" (loner). Một quân cờ được xem là "cô đơn" nếu nó không ở cạnh bất kỳ quân cờ nào.

  (18) Số lượng quân vua "cô đơn".

  (19) Số lượng "hố" - ô trống ở cạnh 3 hoặc 4 quân cùng màu.

- 6 thành phần mẫu (pattern features): (Giả sử quân mình là quân đen)
  - Có chứa mẫu Triangle. (Quân đen nằm ở các ô 27, 31, 32)
  - Có chứa mẫu Oreo. (Quân đen nằm ở các ô 26, 30, 31)
  - Có chứa mẫu Bridge. (Quân đen nằm ở các ô 30, 32)
  - Có chứa mẫu Dog. (Quân đen ở ô 32, quân trắng nằm ở ô 28)
  - Có chứa quân tốt ở góc. (Quân tốt đen nằm ở ô 29)
  - Có chứa quân vua ở góc. (Quân vua đen nằm ở ô 4)





Hình 2. *Đánh dấu các ô trên bàn cờ*

*b) Sơ bộ về hàm đánh giá:*

Nhìn chung, có hai kiểu hàm đánh giá: Tuyến tính và không tuyến tính.

- Hàm đánh giá tuyến tính là sự kết hợp tuyến tính của một hoặc nhiều các tham số kể trên:

$$LinearCombinationOfParameters = a_1.param_1 + a_2.param_2 + a_3.param_3 + \cdots + a_j.param_j \quad (1)$$

- Hàm đánh giá không tuyến tính thường là sự kết hợp của một số ít các tham số kể trên dưới dạng:

$$IF[NOT]\ (min_1 \le par_1 \le max_1 \\ AND/OR\ min_2 \le par_2 \le max_2 \\ AND/OR\ \ldots \\ AND/OR\ min_j \le par_j \le max_j) \\ THEN\ Heuristic\_Result\ +=\ LinearCombinationOfParameters$$

- Các hàm đánh giá tuyến tính thông dụng bao gồm **8F** (bao gồm 8 thành phần từ (1) đến (8)), **15F** (bao gồm 15 thành phần từ (1) – (4), (9) – (12), (19) – (25)), **19F** (bao gồm 19 thành phần từ (1) – (19)) và **25F…**

- Các hàm đánh giá không tuyến tính có thể kể đến **3Ph** và **E3Ph.** Cả hai loại này đều có chung nguyên tắc là chia trò chơi thành 3 thời kỳ:
  + *Beginning*: Mỗi bên có nhiều hơn 3 quân tốt và không có quân vua nào.
  + *Kings:* Mỗi bên có nhiều hơn 3 quân và ít nhất một quân vua trên bàn cờ.
  + *Ending*: Một hoặc cả 2 người chơi đều có ít hơn 3 quân cờ.
  + Sự khác biệt giữa hai hàm này là trong mỗi thời kỳ, 3Ph chỉ xét các tham số từ (1) đến (8), E3Ph xét trên 10 tham số: (3), (10) – (12), (16), (20) – (23) và (25)

# CHƯƠNG II. CÁC CHI TIẾT CỦA PHƯƠNG PHÁP ĐƯỢC DÙNG ĐỂ GIẢI QUYẾT BÀI TOÁN

## 1. Thông tin chung về chương trình:

- Mỗi quân cờ có dạng "b" (quân tốt đen), "B" (quân vua đen), "w" (quân tốt trắng), "W" (quân vua trắng) và "" (ô trống).
- Mỗi trạng thái của bàn cờ tương ứng với một trạng thái của checkerBoard[][]
- Máy được xem như quân đen trên bàn cờ. Tùy vào lựa chọn của người dùng mà máy có thể chơi trước hoặc chơi sau.
  - Mỗi nước đi được biểu diễn bằng một xâu có dạng: $x_1y_1x_2y_2o_1o_2$

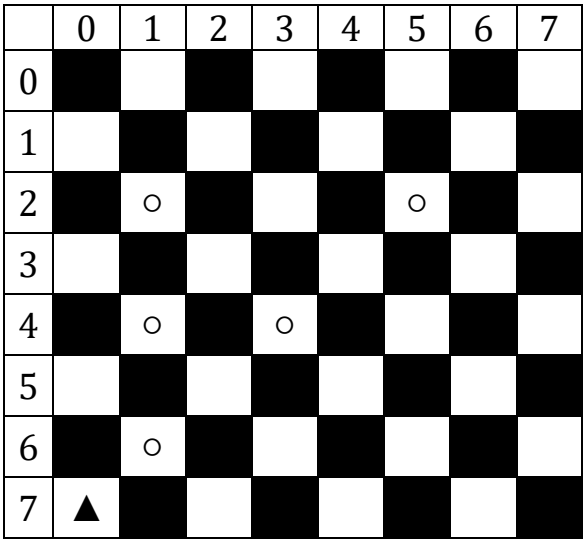
  Trong đó
  + $(x_1, y_1)$ là tọa độ của ô nguồn, $(x_2, y_2)$ là tọa độ của ô đích.
  + $o_1$ là kiểu nước đi. Nó nhận 1 trong 6 giá trị:
    1 nếu quân đang xét là quân tốt, nước đi thường.
    2 nếu quân đang xét là quân tốt, nước đi là một bước nhảy.
    3 nếu quân đang xét là quân tốt, nước đi thường, có phong.
    4 nếu quân đang xét là quân tốt, nước đi là một bước nhảy, có phong.
    5 nếu quân đang xét là quân vua, nước đi thường.
    6 nếu quân đang xét là quân vua, nước đi là mổ bước nhảy.
  + o2 là tham số của o1. Nếu o1 = 1, 3, 5 thì o2 = " ". Nếu o1 = 2, 4, 6 thì o2 nhận giá trị "w" hoặc "W" tùy vào việc bước nhảy này ăn được quân tốt hay quân vua của đối phương.

## 2. Các hàm chuyển trạng thái:

### 2.1. Xây dựng hàm thu thập các nước đi hợp lệ từ trạng thái hiện tại:

- Tập các nước đi là một mảng các xâu nước đi hợp lệ ứng với trạng thái hiện tại của bàn cờ.
- Vì trò chơi bắt buộc khi có nước đi thì người chơi buộc phải ăn nên một tập các nước đi hoặc là một tập các bước nhảy (biểu diễn bởi hàm attackMoves()), hoặc là một tập các nước đi thường (biểu diễn bởi hàm possibleMoves()) khi không có bước nhảy nào hợp lệ.
  + Hàm possibleMoves(): Duyệt qua tất cả các ô trên bàn cờ, ô nào có quân đen thì gọi đến một trong hai hàm possible_b() hoặc possible_B() rồi add vào list.

+ Hàm attackMoves(): Cơ chế hoàn toàn tương tự. Tuy nhiên, vì trò chơi có hỗ trợ multi-jump (tức nhảy liên tiếp) nên nhóm đã tiến hành xây dựng một hàm đệ quy multiJump(move) nhận tham số đầu vào là một nước đi và trả về những "chuỗi nước đi" liên tiếp có thể bắt đầu từ nước đi này.




Hình 3. *Ví dụ về các chuỗi nhảy liên tiếp*

Ví dụ, với đầu vào là nước đi 70526w thì hàm multiJump(move) sẽ trả về 2 chuỗi: 70526w52306w30126w và 70526w52346w34166w. (giả sử quân ▲ là quân vua đen)

## 2.2. Chuyển đến trạng thái mới

- Hàm alphaBeta() dựa vào thuật toán alpha-beta cắt tỉa để trả về một xâu bao gồm hai thành phần: Nước đi "tối ưu" nhất mà máy tính tìm ra được cộng với "giá trị" của nước đi này. Trong hàm này có một số lưu ý:
  + Như đã nói, vì trò chơi buộc phải ăn khi có bước nhảy hợp lệ nên khi chọn tập các nước đi, luôn phải chọn hoặc từ attackMoves(), hoặc từ possibleMoves() nếu attackMoves() rỗng.
  + Nhóm có xây dựng một hàm sortMoves() nhận đầu vào là một list các nước đi và trả về dãy nước đi đó sau khi sắp xếp *một vài* nước đi "tốt nhất" tại thời điểm đó ở đầu của list. Một nước đi gọi là tốt nhất tại thời điểm đó nếu trạng thái *ngay sau khi* thực hiện nước đi này có ước lượng tốt nhất. Tuy vậy không có gì đảm bảo là các nước đi này sẽ dẫn đến một trạng thái thực sự tốt nhất ở độ sâu sâu hơn.

Hơn nữa, vì chi phí thời gian của hàm ước lượng khá lớn nên hàm này chưa thực sự chứng tỏ được điều gì.

- Các hàm makeMove() và makeRealMove() nhận đầu vào là một nước đi move dùng để chuyển trạng thái của bàn cờ sau khi thực hiện move. makeMove là chuyển một nước đi không tính nước nhảy liên tiếp, makeRealMove() là tính cả nước đi liên tiếp.
- Bên cạnh đó, nhóm còn xây dựng hàm undoMove(), undoRealMove() để phục vụ khôi phục trạng thái ban đầu sau các khi thực hiện "thử" các nước đi để tính hàm đánh giá trong thuật toán alpha – beta cắt tỉa. Nhóm cũng xây dựng hàm flipBoard() để nhằm đổi chiều bàn cờ nhằm giảm thiểu việc phải viết thêm các hàm tương ứng cho quân trắng.

**3. Kiểm tra kết thúc trò chơi:**

- Hàm isItWin() dùng để kiểm tra xem trạng thái hiện tại có là một chiến thắng cho bên nào đó hay không. Hàm này gọi đến hai hàm boolean là noMoreMoves() và noMoreWhitePieces() ứng với hai điều kiện dẫn đến chiến thắng là không còn nước đi hợp lệ và không còn quân cờ nào của đối thủ trên bàn cờ. Chi tiết cài đặt của hai hàm này tương đối đơn giản.
- Hàm isItDraw() dùng để kiểm tra xem trạng thái hiện tại có phải là một trận hòa hay không. Nhóm chỉ xây dựng hàm để kiểm tra điều kiện thứ hai (tức bỏ qua việc thỏa thuận), cụ thể:
    + Hàm countValidMovesToDraw() dùng để tính số nước đi mà không làm gia tăng lợi thế nào cho mỗi bên (không có quân tốt hướng về phía king-row và không có quân nào bị mất). Rõ ràng những nước đi đảm bảo trạng thái này là những nước đi loại 5 (quân vua, nước thường, không nhảy). Vì yêu cầu 40 nước đi liên tiếp phải thỏa mãn trạng thái này nên nếu chỉ một nước đi vi phạm thì biến đếm sẽ đếm lại từ đầu.
    + Đối với điều kiện trạng thái lặp lại lần thứ 3 liên tiếp. Nhóm xây dựng hàm addDrawList() để lưu lại các trạng thái bàn cờ (đã được chuyển về dạng xâu) có thể lặp lại đến lần thứ 3. Các trạng thái, cũng hoàn toàn tương tự, là các trạng thái trước và sau khi thực hiện một nước đi loại 5. Nếu vi phạm điều này, xóa toàn bộ các trạng thái đã lưu.

+ Sở dĩ chỉ có nước 5 là nước thỏa mãn những điều kiện trên là do các nước 2, 4, 6 là các nước nhảy, nó làm mất quân trên bàn cờ. Các nước 1, 3 là các nước của quân tốt, chúng luôn hướng về phía trước nên luôn có lợi thế có thể phong vua. Rõ ràng sau khi thực hiện các nước đi này thì trạng thái bàn cờ không thể khôi phục lại (quân mất đi không thể vào lại và quân tốt thì không thể đi lùi). Suy cho cùng, điều kiện hòa là điều kiện không trực tiếp làm tăng lợi thế cho bên nào cả, do đó chỉ có nước 5 là nước được dùng để xét điều kiện hòa.

**4. Hàm đánh giá:**

Vì hạn chế trong trình độ cũng như không có chuyên môn sâu trong lĩnh vực cờ đam nên nhóm xây dựng hàm ước lượng dựa theo tiêu chuẩn 19F.

- Biểu hiện của các thành phần:

+ Các tiêu chuẩn (1), (2) được tính ở hàm evaluateMaterial(), trong đó
  $a_{(1)} = POINT\_NORMAL, a_{(2)} = POINT\_KING$
+ Các tiêu chuẩn (3), (4), (9) – (16) được thể hiện ở hai ma trận và hàm evaluationPositional():
  Đối với quân tốt:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | 22 | | 17 | | 17 | | 27 |
| 22 | | 20 | | 15 | | 25 | |
| | 20 | | 20 | | 25 | | 17 |
| 2 | | 15 | | 15 | | 0 | |
| | 0 | | 25 | | 15 | | 2 |
| 2 | | 10 | | 10 | | 5 | |
| | 20 | | 10 | | 15 | | 17 |
| 22 | | 12 | | 12 | | 17 | |

  Đối với quân vua:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | 27 | | 22 | | 22 | | 37 |
| 27 | | 25 | | 20 | | 35 | |
| | 25 | | 25 | | 35 | | 22 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 2 | | 30 | | 45 | | 0 | |
| | 0 | | 45 | | 30 | | 2 |
| 2 | | 15 | | 5 | | 5 | |
| | 25 | | 10 | | 15 | | 17 |
| 27 | | 12 | | 12 | | 17 | |

Hình 4 và 5. *Điểm vị trí của quân tốt và quân vua*

Trong đó:

$$a_{(3)} = a_{(4)} = 2,\ a_{(9)} = a_{(10)} = 10, a_{(11)} = 10, a_{(12)} = 25,$$

$$a_{(13)} = 10, a_{(14)} = 15, a_{(15)} = a_{(16)} = 5$$

+ Các tiêu chuẩn (5), (6) được tính ở hàm evaluateMoveability(), trong đó

$$a_{(5)} = a_{(6)} = POINT\_MOVE$$

+ Các tiêu chuẩn còn lại được tính ở hàm evaluateAnotherComponent(), trong đó:

$$a_{(7)} = POINT\_PROMOTION - 2 * row$$

$$a_{(8)} = POINT\_UNOCCUPIED$$

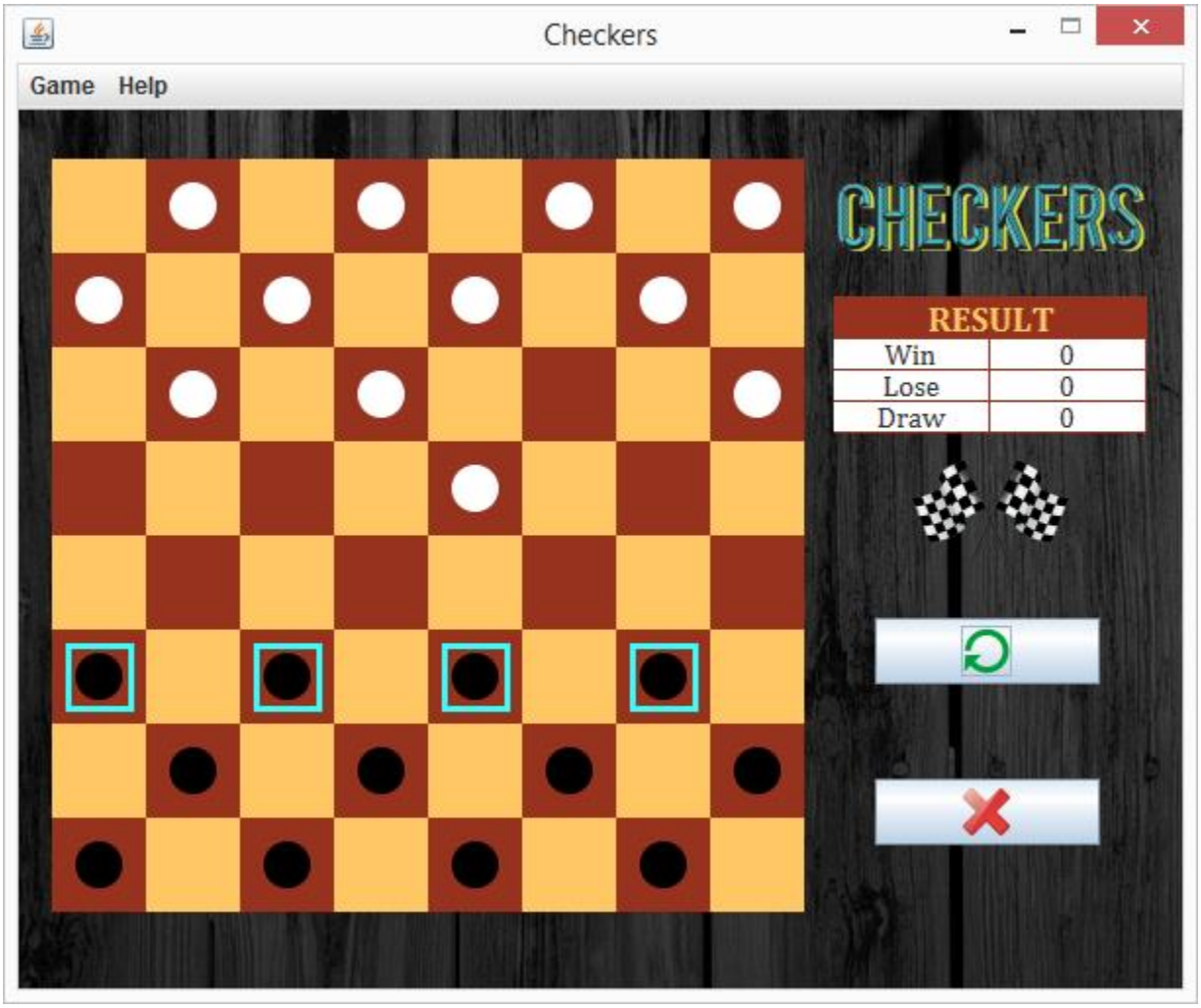
$$a_{(17)} = POINT\_LONERPOWN, a_{(18)} = POINT\_LONEKING$$

$$a_{(19)} = POINT\_HOLE$$

# CHƯƠNG III. MỘT SỐ VẤN ĐỀ XUNG QUANH CHƯƠNG TRÌNH

## 1. Chức năng chính và cách thức sử dụng

### 1.1. Chức năng chính




Hình 6. *Giao diện chương trình*

Trò chơi được thiết kế dành cho một người chơi với đối thủ là trí máy với trí tuệ nhân tạo. Do đó trò chơi cần có một số những yêu cầu cụ thể như sau:

i. Trò chơi phải thực hiện đầy đủ các quy tắc mà trò chơi thực tế đặt ra bao gồm luật chơi cũng như là điều kiện để kết thúc một ván cờ.

ii. Người chơi có thể quyết định được ai là người đi trước.

iii. Trong khi chơi thì người chơi có thể nhìn thấy được những nước đi có thể của mình.

iv. Người chơi có thể thực hiện các nước đi một cách dễ dàng.

v. Thời gian chờ đợi một nước đi của máy phải trong khoảng thời gian chờ đợi cho phép của người chơi.

vi. Giao diện trò chơi cần được thiết kế dễ nhìn và dễ sử dụng.

### 1.2. Cách thức sử dụng

Trước khi chơi, người dùng được phép lựa chọn người đi trước. Sau khi lựa chọn thì có thể bắt đầu trò chơi một cách bình thường.

Trong quá trình chơi, khi máy thực hiện các nước đi thì trên bàn cờ hiện lên các ô vuông viền màu xanh ngọc để thông báo cho người chơi biết các quân cờ nào có thể di chuyển được và các quân cờ nào không.

Rõ ràng, khi người chơi có nước nhảy, thì người chơi phải thực hiện các bước này như trong luật chơi thực tế.

Người chơi rê chuột từ ô có hiện vành màu xanh ngọc sang ô mà mình muốn đi. Nếu nước đi hợp lệ, chương trình sẽ ghi nhận và thực hiện nước đi này.

Sau khi kết thúc trận đấu, chương trình sẽ hiện ra thông báo cho người chơi đồng thời cập nhật thành tích trên góc phải màn hình.

Trong và sau khi chơi xong một ván. người chơi có thể chơi ván mới bằng cách nhấn vào nút RESTART hoặc thoát chương trình bằng cách nhấn vào nút EXIT bên phải bàn cờ.

## 2. Các phương pháp được tái sử dụng lại trong quá trình thực hiện đề tài:

Quá trình xây dựng và phát triển chương trình được dựa theo khóa học: Java Chess Engine Tutorial của tài khoản Youtube **Logic Crazy Chess.** Bên cạnh đó, một số luận điểm được trình bày trong tài liệu này được tham khảo từ một số nguồn tài nguyên khác nhau được lấy từ Internet.(sẽ được trình bày ở phần danh mục tài liệu tham khảo).

## 3. Những khó khăn gặp phải trong quá trình thực hiện công việc:

### 3.1. Khó khăn về tổ chức:

- Trong bước đầu thành lập, nhóm gặp khó khăn trong việc lựa chọn đề tài. Do hiểu biết chưa đầy đủ về những nội dung trong chương trình học mà nhóm lựa chọn đề tài "quá sức" và sau đấy bị nộp muộn đề xuất đề tài.
- 4 thành viên trong nhóm dự định theo học các chuyên ngành khác nhau nên trong quá trình phân chia công việc, trưởng nhóm buộc phải lựa chọn hoặc chia ngang công việc, tức mỗi người làm một ít, để ai cũng có thể hiểu một phần nội dung của chương trình, hoặc bổ dọc công việc, tức mỗi người làm một phần theo chuyên ngành mà mình định theo học, dù điều này có thể dẫn đến một số thành viên trong nhóm bị lệch về thuật toán, một số thành viên thì khác bị lệch về báo cáo/ giao diện. Để giải quyết điều này, nhóm đã chia quá trình thực hiện đồ án thành 3 thời kỳ, 2 tuần đầu chia ngang, mỗi người tự góp nguồn tài liệu tham khảo của mình, 3 đến 4 tuần tiếp theo chia dọc, mỗi người tự làm phần việc theo hiểu biết của mình và những tuần cuối là chia sẻ công việc cho những thành viên còn lại.

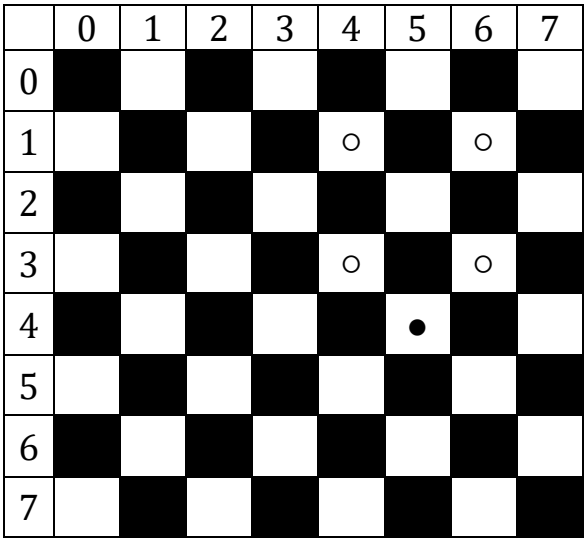
### 3.2. Những khó khăn trong quá trình thực hiện xây dựng chương trình:

Trong quá trình thực hiện chương trình theo khóa học Java Chess Engine Tutorial, nhóm xây dựng chương trình song song với chương trình trong trong khóa học, chuyển những tri thức của môn cờ vua sang tri thức phù hợp với bộ môn cờ đam, tuy nhiên, quá trình này gặp một số vấn đề khó khăn, có thể kể đến như:

+ Trong cờ đam, như đã nói ở các mục trước, vì luật buộc người chơi phải ăn khi có nước ăn nên khi xây dựng chương trình, khác với cờ vua, tập các nước đi của cờ đam phải chia rõ ràng thành hai tập. tập các nước đi và tập các nước ăn. Nếu tập các nước ăn rỗng thì mới chọn đến tập các nước đi. Luật này trong một số thời điểm có thể khiến cho chiến thuật của trò chơi bị hạn chế ở trường hợp tổng quát.
- Trong cờ vua, vì mỗi nước đi chỉ ăn đúng một lần nên mỗi nước đi luôn có độ dài cố định. Khóa học trên xây dựng tập các nước đi thành một xâu chứ không phải là một list các xâu như trong chương trình của nhóm. Trong cờ đam, người chơi được phép thực hiện các

nước đi liên tiếp. Điều này đặt nhóm trước hai lựa chọn: Hoặc là biểu diễn một nước đi thành một xâu không nhất thiết phải có độ dài bằng 6 (thay vào đó là 6*x), điều này buộc nhóm phải thay đổi gần như toàn bộ các thủ thuật liên quan đến xâu đã được thực hiện. Hoặc giữ nguyên mỗi nước đi là một xâu 6 ký tự, nhưng cây Max-Min không còn ở dạng nguyên thủy mà một số nhánh sẽ chứa các dãy liên tiếp max – max hoặc min – min, điều mà nhóm chưa có đủ hiểu biết để giải quyết.

- Trong quá trình xây dựng hàm multiJump(), nhóm cũng cân nhắc có nên xây dựng một hàm chống việc lặp các chuỗi liên tiếp, chẳng hạn:




Hình 7. *Chuỗi nước đi bị lặp lại*

Trạng thái này có thể bị multiJump() tạo ra hai chuỗi liên tiếp " lặp lại" nhau (khái niệm "lặp lại" ở đây để nói đến việc chúng cùng dẫn đến một trạng thái của bàn cờ).

Tuy nhiên vì số lượng quân cờ trên bàn cờ là 12 nên thực chất, cùng lắm thì việc lặp này cũng chỉ lặp lại 3 lần, so với các chi phí khác thì không đáng kể và không nhất thiết phải xây dựng thêm một phương thức để giải quyết vấn đề này.

- Vì chương trình của khóa học trên xây dựng tập các nước đi là một xâu nên khi sortMove(), các thao tác thực hiện trên xâu sẽ rất ít tốn thời gian. Do đó thời gian để tìm một nước đi trong thuật giải alpha-beta cắt tỉa giảm đi 5 lần. Tuy nhiên, tập các nước đi trong chương

trình của nhóm lại được biểu diễn bằng ArrayList, hàm ước lượng cũng cồng kềnh nên hiệu quả hàm này là không có.

- Vấn đề cuối cùng mà nhóm gặp phải chính là cơ sở khoa học của các thành phần trong hàm ước lượng. Cho đến hiện tại, các con số này vẫn chỉ là những con số mang tính áng chừng và chưa có luận chứng khoa học thích đáng.

## 4. <u>Những hướng đi tiếp theo</u>:

- Mục tiêu chính của nhóm là tiếp tục cải thiện sự thông minh của chương trình bằng cách xây dựng và hoàn thiện hàm đánh giá, thử nghiệm các hàm hàm đánh giá dạng tuyến tính và phi tuyến tính khác. Có thể tìm hiểu thêm những cấu trúc dữ liệu khác để kiểm nghiệm tính hiệu quả của hàm sortMoves().Tìm hiểu thêm về các khái niệm HG và SHG để tăng chất lượng hàm đánh giá.
- Một trong những hướng đi cần học hỏi từ phần mềm Chinook là bên cạnh xây dựng hiệu quả hàm đánh giá, cần tìm hiểu chuyên sâu và xây dựng một bộ cơ sở dữ liệu lưu trữ các nước đi khi trên bàn cờ chỉ còn lại một số quân nhất định nào đấy.
- Tăng khả năng học cho chương trình.
- Bên cạnh đó, có thể xây dựng lại mức độ phù hợp của trò chơi với từng nhóm người dùng ở các trình độ khác nhau.
- Cải thiện về giao diện và tính "phần mềm" của chương trình. Xây dựng và hoàn thiện tài liệu hướng dẫn sử dụng.

## DANH MỤC TÀI LIỆU THAM KHẢO

[1] Jacek Mańdziuk, Magdalena Kusiak and Karol Waldęzik, "*Evolutionary-based heuristic generators for checkers andgive-away checkers"*, Faculty of Mathematics and Information Science,Warsaw University of Technology, 2007.

<URL: https://www.researchgate.net/publication/227727266_Evolutionary-based_heuristic_generators_for_checkers_and_give-away_checkers > [Truy cập ngày 1/12/2016]

[2] WCDF, *"Rules of Checkers",* 2012.

<URL: http://www.wcdf.net/rules/rules_of_checkers_english.pdf > [Truy cập ngày 20/11.2016]