**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**



**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA CÔNG NGHỆ PHẦN MỀM**

**--------------------**

**ĐỒ ÁN MÔN MỘT SỐ THUẬT TOÁN THÔNG MINH**

**Đề tài: XÂY DỰNG GAME CARO**

**Giảng viên hướng dẫn:**

**Ths. NGUYỄN VĨNH KHA**

**Sinh viên thực hiện:**

**HUỲNH NGỌC – 13520567**

***Năm học 2007 - 2008***

***TPHCM, 31/05/2017***

**LỜI MỞ ĐẦU**

*Lý thuyết trò chơi* là một nhánh của toán học, nó sử dụng các mô hình để nghiên cứu các tình huống chiến thuật, trong đó các đối thủ cố gắng làm tối đa kết quả thu được của mình. Trong thời đại Công nghệ thông tin phát triển mạnh như hiện nay thì *Lý thuyết trò chơi* thu hút được rất nhiều sự chú ý từ các nhà khoa học máy tính do ứng dụng của nó trong Trí tuệ nhân tạo và Điều khiển học…

Trong báo cáo này, em sẽ trình bày một trong những ứng dụng của Lý thuyết trò chơi, đó là thuật giải tìm kiếm Min-Max, Alpha-Beta và ứng dụng trong việc xây dựng một chương trình trò chơi đối kháng, cụ thể là trò chơi cờ Caro.

Trong quá trình thực hiện đồ án, em xin cảm ơn đến thầy Nguyễn Vĩnh Kha đã nhiệt tình hướng dẫn, cũng như cung cấp tài liệu để em có thể hoàn thành được đồ án. Trong quá trình thực hiện không tránh khỏi những sai sót, mong nhận được sự góp ý của thầy. Em xin chân thành cảm ơn.

**Mục lục**

[CHƯƠNG 1: TỔNG QUAN 4](#_Toc482828156)

[CHƯƠNG 2: GIỚI THIỆU THUẬT TOÁN MIN-MAX VÀ ALPHA-BETA 4](#_Toc482828157)

[2.1 Thuật toán Min-Max 4](#_Toc482828158)

[2.1.1 Thuật toán Min-Max là gì? 4](#_Toc482828159)

[2.1.2 Tại sao phải cần dùng thuật toán Min-Max 4](#_Toc482828160)

[2.1.3 Các khái niệm 5](#_Toc482828161)

[2.1.4 Giải thuật Min-Max 5](#_Toc482828162)

[2.1.5 Các bước của thuật giải Max-Min 5](#_Toc482828163)

[2.1.6 Ví dụ mô phỏng thuật toán Min-Max 5](#_Toc482828164)

[2.1.7 Ưu điểm và khuyết điểm của thuật toán Min-Max 8](#_Toc482828165)

[2.1.8 Tối ưu thuật toán Min-Max 8](#_Toc482828166)

[CHƯƠNG 3: ÁP DỤNG THUẬT TOÁN MAX-MIN VÀO GAME CỜ CARO 8](#_Toc482828167)

[3.1 Giới thiệu trò chơi đối kháng 8](#_Toc482828168)

[3.2 Giới thiệu về trò chơi cờ caro 9](#_Toc482828169)

[3.2.1 Giới thiệu 9](#_Toc482828170)

[3.2.2 Luật chơi 9](#_Toc482828171)

[3.3 Phân tích và thiết kế 10](#_Toc482828172)

[3.3.1 Phân tích bài toán 10](#_Toc482828173)

[3.3.2 Các lớp sử dụng trong chương trình 11](#_Toc482828174)

[3.3.3 Thuật toán Min-Max trong chương trình 13](#_Toc482828175)

[3.3.4 Giao diện chương trình 17](#_Toc482828176)

[NHẬN XÉT VÀ KẾT LUẬN 20](#_Toc482828177)

[HƯỚNG PHÁT TRIỂN 20](#_Toc482828178)

[TÀI LIỆU THAM KHẢO 20](#_Toc482828179)

# **CHƯƠNG 1: GIỚI THIỆU CHUNG**

Sau nhiều năm phát triển thì hiện nay *Lý thuyết trò chơi* đã được sử dụng rộng rãi trong nhiều ngành khác nhau như: Kinh tế và kinh doanh, sinh học, chính trị học, triết học, khoa học máy tính và logic, viễn thông, một số show game trên truyền hình…

Trong thời đại Công nghệ thông tin phát triển như hiện nay thì *Lý thuyết trò chơi* đóng vai trò hết sức quan trọng, đặc biệt trong logic và khoa học máy tính. Một số lý thuyết logic có cơ sở trong ngữ nghĩa trò chơi. Thêm vào đó những khoa học gia máy tính đã sử dụng trò chơi để mô phỏng những tính toán tương tác với nhau.

Một số thuật toán trong *Lý thuyết trò chơi* giúp xây dựng, phát triển những trò chơi hay, như: thiết kế trò chơi Nim; thiết kế kiểu trò chơi có nhân, có tính đối xứng, thuật toán liên quan đến chiến lược tìm kiếm ,… Bài báo cáo đề cập đến thuật toán tìm kiếm Min-Max và thuật toán cắt tỉa Alpha-Beta trong việc xây dựng chương trình trò chơi cờ caro.

# **CHƯƠNG 2: GIỚI THIỆU THUẬT TOÁN MIN-MAX**

## 2.1 Thuật toán Min-Max

### 2.1.1 Thuật toán Min-Max là gì?

Giải thuật Minimax (hay Min-Max) là một thuật toán đệ quy lựa chọn bước đi kế tiếp trong một trò chơi có hai người. Xét một trò chơi đối kháng trong đó hai người thay phiên đi nước đi của mình như cờ vua, cờ tướng, cờ caro, cờ vây… Khi chơi bạn có thể khai triển hết không gian trạng thái nhưng khó khăn chủ yếu là bạn phải tính toán được phản ứng và nước đi của đối thủ mình như thế nào? Cách xử lý đơn giản là bạn giả sử đối thủ của bạn cũng sử dụng kiến thức về không gian trạng thái giống bạn. Giải thuật Minimax áp dụng giả thuyết này để tìm kiếm không gian trạng thái của trò chơi.

Giải thuật Minimax giúp chúng ta tìm được bước đi kế tiếp tốt nhất trong các không gian trạng thái tiếp theo bằng cách phát triển hết không gian trạng thái cây trò chơi định giá trị cho các Node tìm Node có trạng thái tốt nhất ở bước tiếp theo phát triển tiếp..

### 2.1.2 Tại sao phải cần dùng thuật toán Min-Max

Như các bạn đã biết thì có rất nhiều thuật toán tìm kiếm để làm AI trong game như A, Heuristic... Mỗi thuật toán thì sẽ phù hợp với từng loại game cho nó. Những game đối kháng trong đối người chơi luân phiên đánh như cờ vua, cờ tường, caro...

Khi chơi bạn có thể khai triển hết không gian trạng thái nhưng khó khăn chủ yếu là bạn phải tính toán được phản ứng và nước đi của đối thủ mình như thế nào? Cách xử lý đơn giản là bạn giả sử đối thủ của bạn cũng sử dụng kiến thức về không gian trạng thái giống bạn. Giải thuật Minimax áp dụng giả thuyết này để tìm kiếm không gian trạng thái của trò chơi. Trường hợp này thuật toán minimax sẽ đáp ứng những gì mình cần.

### 2.1.3 Các khái niệm

* Cây trò chơi (Game tree) - Đại khái là một sơ đồ hình cây thể hiện từng trạng thái, từng trường hợp của trò chơi theo từng nước đi.
* Mỗi node biểu diễn 1 trạng thái của trò chơi hiện tại trên cây trò chơi.
* Node được gọi nút lá là tại đó trò chơi kết thúc (trạng thái trò chơi lúc đó có thể thắng, thua hoặc hòa).

### 2.1.4 Giải thuật Min-Max

Hai đối thủ trong trò chơi được gọi là MIN và MAX luân phiên thay thế nhau đi. MAX đại diện cho người quyết dành thắng lợi và cố gắng tối đa hóa ưu thế của mình, ngược lại người chơi đại diện cho MIN lại cố gắng giảm điểm số của MAX và cố gắng làm cho điểm số của mình càng âm càng tốt. Giả thiết đưa ra MIN và MAX có kiến thức như nhau về không gian trạng thái trò chơi và cả hai đối thủ đều cố gắng như nhau.

Mỗi Node biểu diễn cho một trạng thái trên cây trò chơi. Node lá là Node chứa trạng thái kết thúc của trò chơi.

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.

### 2.1.5 Các bước của thuật giải Max-Min

* **Bước 1**: Nếu như đạt đến giới hạn tìm kiếm (đến tầng dưới cùng của cây tìm kiếm tức là trạng thái kết thúc của trò chơi).
* **Bước 2**: Tính giá trị của thế cờ hiện tại ứng với người chơi ở đó. Ghi nhớ kết quả.
* **Bước 3**: Nếu như mức đang xét là của người chơi cực tiểu (nút MIN), áp dụng thủ tục Minimax này cho các con của nó. Ghi nhớ kết quả nhỏ nhất.
* **Bước 4**: Nếu như mức đang xét là của người chơi cực đại (nút MAX), áp dụng thủ tục Minimax này cho các con của nó. Ghi nhớ kết quả lớn nhất.

### 2.1.6 Ví dụ mô phỏng thuật toán Min-Max

* MAX đại diện quân đi O.
* MIN đại diện quân đi X.

Trạng thái kết thúc là trạng thái có 3 ô liên tiếp ngang, dọc, chéo có cùng một quân cờ X hoặc O, nếu là X tức MIN thắng còn O tức MAX thắng còn nếu tất cả các ô cờ đều được đi và trạng thái chưa kết thúc thì bàn cờ hòa. Điểm thắng của X là -1, của O là 1, và bàn cờ hòa là 0.

* *Áp dụng giải thuật Minimax:*

Từ trạng thái bàn cờ hiện tại ta dự đoán nước đi của trạng thái tiếp theo nếu trạng thái tiếp theo ta tiến hành lượng giá cây trò chơi bằng cách ta tiến hành quét cạn tất cả các trạng thái tiếp theo cho đến lúc gặp trạng thái chiến thắng (Node lá) tính điểm cho Node lá bằng cách:

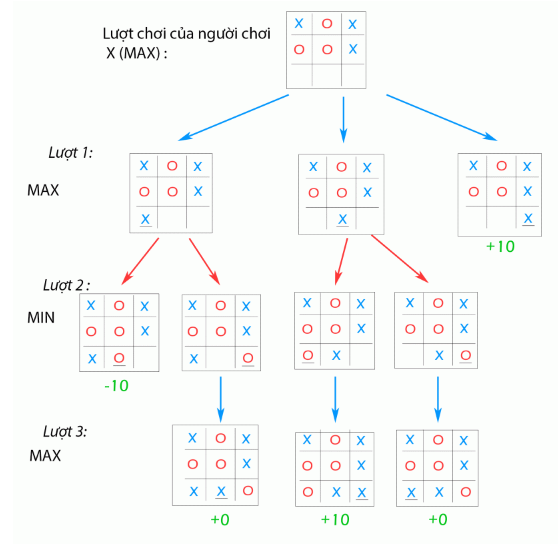
* Nếu ở trạng thái mà ta gặp chiến thắng nếu đó là lượt đi của quân X thì đánh giá điểm trạng thái đó là -1.
* Nếu ở trạng thái ta gặp chiến thắng nếu đó là lượt đi của quân O thì đánh giá điểm trạng thái đó là 1.
* Nếu là hòa thì điểm trạng thái đó là 0.

Sau đó tính ngược lại cây trò chơi theo quy tắc:

* Nút thuộc lớp MAX thì gán cho nó giá trị lớn nhất của các Node con của Node đó.
* Nút thuộc lớp MAX thì gán cho nó giá trị nhỏ nhất của các Node con của Node đó.

Sau khi lượng giá hết cây trò chơi ta tiến hành chọn bước đi tiếp theo nguyên tắc:

* Nếu lớp tiếp theo là MAX ta chọn Node con có giá trị lớn nhất.
* Nếu lớp tiếp theo là MIN ta chọn Node con có giá trị nhỏ nhất.



Figure‑1 Hình minh họa thuật toán Min-Max

Như hình trên ta thấy là trạng thái hiện tại của game đang đến lượt đánh của người chơi X đại diện cho MAX. Ta tạm quy định giá trị MAX lúc game thắng cho X = +10 và MIN lúc game thua cho X = -10 và lúc game hòa = 0. Lúc này ở lượt 1, MAX có thể đi được 1 trong 3 nước như hình.

Vậy làm sao để chọn 1 trong 3 nước đó nước nào là tốt nhất để đi. Chúng ta dựa vào giá trị của từng nước để chọn nước tốt nhất, như ở đây 3 node đó thuộc lớp MAX nên chọn giá trị lớn nhất. Chúng ta bắt đầu tìm giá trị của từng node đó.

Ở lớp MAX trong lượt 1, thì ta có node 1,2,3 được đánh số từ trái sáng phải như hình. Node 3 chúng ta đã là node lá (X win game ) và có giá trị là +10. Còn 2 node 1,2 thì chưa biết giá trị của nó tại lượt 1 nên chúng ta dựa vào giá trị của các node con để định giá trị và bằng giá trị bé nhất của các node con ở lớp MIN tại lượt 2.

Cứ tiếp tục tương tự như vậy đến lúc gặp node lá thì từ node lá đó ta suy ngược lại và ta tính được node 1 có giá trị là -10 và node 2 là 0. Vậy nước đi tốt nhất ở đây là như node 3 có giá trị lớn nhất là +10

### 2.1.7 Ưu điểm và khuyết điểm của thuật toán Min-Max

#### 2.1.7.1 Ưu điểm

Tìm kiếm được mọi nước đi tiếp theo sau đó lựa chọn nước đi tốt nhất, vì giải thuật có tính chất vét cạn nên không bỏ soát trạng thái

#### 2.1.7.2 Khuyết điểm

- Đối với các trò chơi có không gian trạng thái lớn như caro, cờ tướng… việc chỉ áp dụng giải thuật Minimax có lẽ không còn hiệu quả nữa do sự bùng nổ tổ hợp quá lớn.

- Giải thuật áp dụng nguyên lý vét cạn không tận dụng được thông tin của trạng thái hiện tại để lựa chọn nước đi, vì duyệt hết các trạng thái nên tốn thời gian.

### 2.1.8 Tối ưu thuật toán Min-Max

Nên sinh ra thuật toán để tối ưu thuật toán minimax là cắt tỉa Alpha Beta

# **CHƯƠNG 3: ÁP DỤNG THUẬT TOÁN MAX-MIN VÀO GAME CỜ CARO**

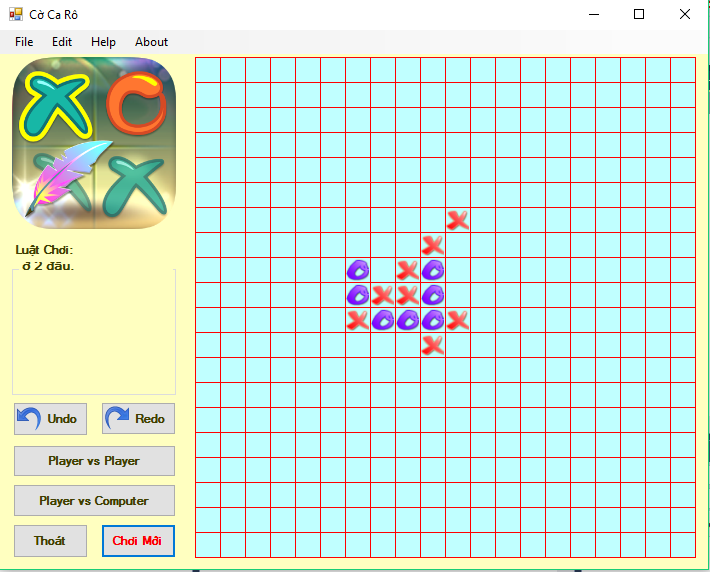
## 3.1 Giới thiệu trò chơi đối kháng

Trò chơi đối khàng diễn ra ở 2 đối thủ: Nhìn chung thì trò chơi thương có đặc điểm như sau:

* Mỗi trò chơi đều có 1 luật chơi mà các đấu thủ đều phải cố gắng để giành phần thắng về phần mình. Trận đấu phải có kết thúc hòa hoặc phân định thắng thua chứ không kéo dài vô tận.
* Mỗi đối thủ được đi 1 nước khi tới lượt mình.
* Các đối thủ đều được biết thông tin về trình trạng của trận đấu.

Một số trò chơi đối kháng như: Tictactoe, Cờ caro, cờ vua, cờ tướng….

## 3.2 Giới thiệu về trò chơi cờ caro



### 3.2.1 Giới thiệu

### 3.2.2 Luật chơi

– Ván cờ được chơi trên bàn cờ với 20x20 ô (bàn cờ trung bình). Người đi trước sẽ đi vào tâm của bàn cờ tức là giao của cột 10, hàng 10.

– Hai bên thay nhau đi những nước cờ được ký hiệu bằng chữ X-O

– Bên nào có đường 5 quân liền nhau trên một hàng, một cột hoặc một đường chéo là người thắng cuộc

– Nếu đi được 5 quân liền nhau nhưng bị đối phương chặn trước 2 đầu thì sẽ không được coi là thắng cuộc

– Các nước cờ phải đi vào giao điểm của hai đường thẳng và không được thay đổi khi đã xuất nước cờ.

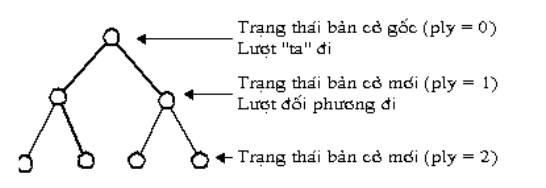
– Nước cờ sau không được trùng với nước cờ đã được đi trước.

## 3.3 Phân tích và thiết kế

### 3.3.1 Phân tích bài toán

#### 3.3.1.1 Biểu diễn bài toán dưới dạng cây trò chơi

Trò chơi có thể được biểu diễn như một cây gồm gốc, những nút , những lá và những nhánh.



* Gốc là trạng thái ban đầu của trò chơi. Với mỗi trò chơi cụ thể thì trạng thái (ở mỗi thời điểm) lại được đặc trưng bỏi những thông số riêng.
* Các nút (Node) của cây thể hiện trình trạng hiện tại của trò chơi, gồm nút cha (Parent Node) và nút con (Children Node)
* Các nhánh giữa các nút thể hiện nước đi, tức là cho biết một tình huống của trò chơi chuyển sang tình huống khác thông qua một nước đi nào đó
* Các lá (leave) hay còn gọi là nút lá (leave node), thể hiện thời điểm kết thúc khi mà kết quả của trò chơi đã rõ rang.
* Ngoài ra còn một thông số của cây nữa là độ sâu (Fly) hay còn gọi là mức của cây, số tầng của cây.

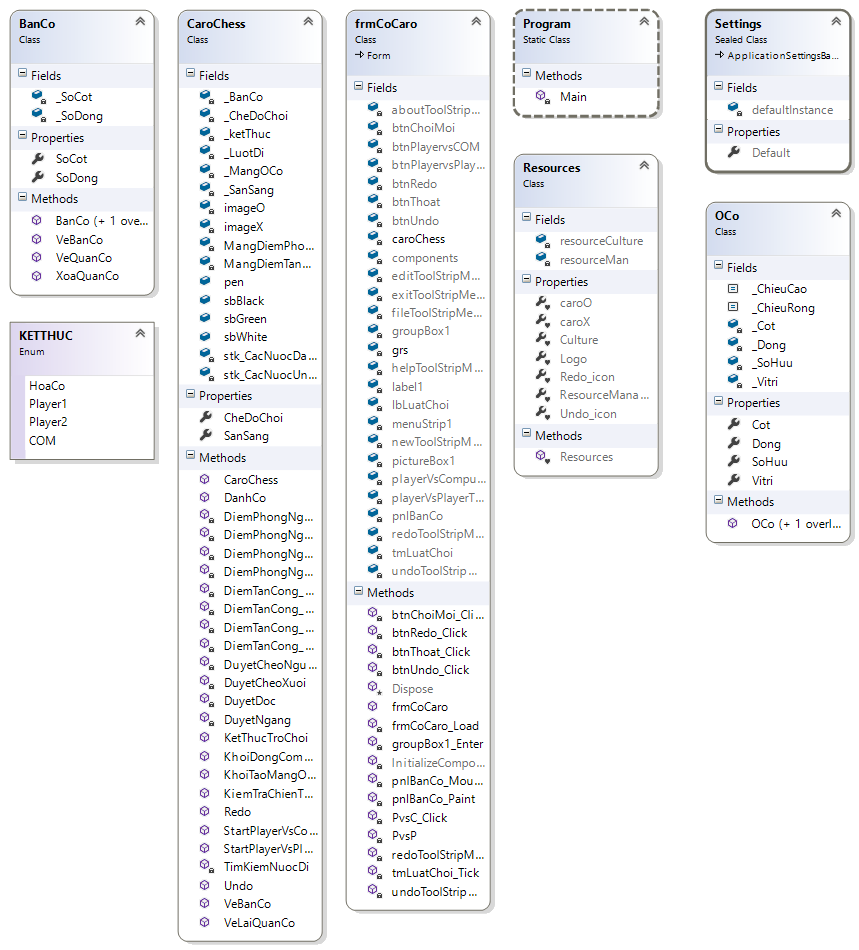
Thường thì mỗi vị trí kết thúc của trò chơi (nút lá) sẽ gán một trọng số, chẳng hạn gán 1 cho chiến tháng, 0 là hòa và -1 là thua. Tại mỗi nút cũng có một trọng số tương ứng được xác định rằng bằng một cách nào đó. Dựa vào cây trò chơi này, người ta có thể tìm nước đi “tốt” để giành chiến thắng cho mình bằng cách tìm kiếm trên cây để tìm ra nước đi tốt nhất.

#### 3.3.1.2 Không gian tìm kiếm nước đi và chiến lược tìm kiếm

Việc tìm kiếm nước đi tiếp theo là việc tìm kiếm trong không gian các ô trống còn lại, sau mỗi lượt đi thì không gian tìm kiếm sẽ giảm dần.

Chiến lược thường được cả người lẫn máy dùng là phân tích thế cờ chỉ sau một nước đi nào đó của 2 bên. Tức là trên cây trò chơi, việc tìm kiếm nước đi là chọn 1 nút trên cây sao cho nước đi đó là “tốt”. Và để đánh giá nút đó thì thường phải “nhìn xa”, liên quan đến độ sâu của cây (tương ứng với việc người chơi phải “nhìn xa” xem bàn cờ có những khả năng biến đổi nào sau một số nước, từ đó đánh giá được độ xấu của thế cờ hiện tại). Với máy tính thì thế cờ này được đánh giá tốt hơn thế cờ kia nhờ so sánh điểm của thế cờ đó do bộ lượng giá trả lại. Vì không gian tìm kiếm là quá lớn nên chúng ta giới hạn cho máy tính chỉ tìm kiếm ở một độ sâu nhất định, tất nhiên độ sâu càng lớn thì chương trình càng thông minh nhưng phải trả giá về mặt thời gian.

### 3.3.2 Các lớp sử dụng trong chương trình



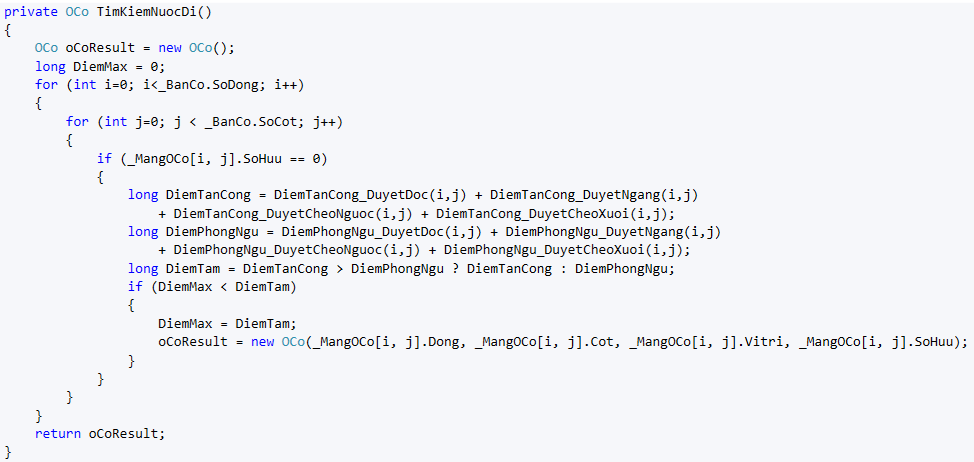
|  |  |  |
| --- | --- | --- |
| **STT** | **Tên Class** | **Diễn giải** |
| 1 | BanCo | Class cho phép vẽ bàn cờ, vẽ quân cờ lên bàn cờ, xóa một quân cờ khi Undo |
| 2 | CaroChess | Class cho phép:   * KhoiTaoMangOCo(): Khởi tạo một mảng các ô cờ * DanhCo(): Thực hiện phương thức đánh cờ * VeLaiQuanCo(): Vẽ lại quân cờ * StartPlayerVsPlayer(): Tạo một ván cờ mới giữa 2 người chơi. * StartPlayerVsCom(): Tạo một ván cờ mới giữa người chơi và máy. * Undo(): Chức năng Undo * Redo(): Chức năng Redo * KetThucTroChoi(): Thông báo Xác định thắng thua trong ván cờ. * Kiểm traChienThang(): Kiểm tra chiến thắng giữa 2 đối thủ. * Chức năng duyệt các ô cùng sở hữu của các đối thủ để kiểm tra chiến thắng (DuyetNgang(), DuyetDoc(), DuyetCheoXuoi(),DuyetCheoNguoc()) * KhoiDongComputer(): Xác định vị trí đi đầu tiên của Computer là ở vị trí hàng 10, cột 10. * TimKiemNuocDi(): Sử dụng thuật toán min-max để tìm kiếm nước đi thích hợp cho Computer dựa vào điểm tấn công và điểm phòng thủ * Tinh điểm tấn công của Computer (DiemTanCong\_DuyenNgang, DiemTanCong\_DuyetDoc, DiemTanCong\_DuyetCheoXuoi, DiemTanCong\_DhuyetCheoDoc) * Điểm phòng ngự của Computer (DiemTanCong\_DuyenNgang, DiemTanCong\_DuyetDoc, DiemTanCong\_DuyetCheoXuoi, DiemTanCong\_DhuyetCheoDoc) |
| 3 | frmCaro | Xử lí các button trên UI |
| 4 | OCo | Geter và Seter của một ô cờ |

### 3.3.3 Thuật toán Min-Max trong chương trình

Trong 2 người chơi thì một người gọi là người chơi cực đại(Max) và đối thủ của họ là người chơi cực tiểu. Cả 2 đấu thủ đều cố gắng đi những nước nào để điểm tuyệt đối của mình lớn hơn hay cao nhất có thể. Tức là người chơi Max sẽ tìm cách làm điểm của mình cao hơn và làm điểm của đối thủ bớt âm hơn (giảm về trị số). Trong khi người chơi Min thì ngược lại, sẽ cố gắng làm cho điểm của mình âm hơn và làm cho điểm của đối thủ giảm.

Giải thuật tìm kiếm Min-Max được sử dụng để xác định tất cả những “diễn biến” tiếp theo của trò chơi cho đến tầng được yêu cầu. Điểm số ban đầu được gán cho lá, sau đó bằng cách lượng giá các nước đi, điểm số được gán cho các tầng ở trên qua giải thuật Min Max, giải thuật thực hiện 1 lá cắt cho trước và tính điểm trên đó.

#### 3.3.3.1 Phương thức tìm kiếm nước đi cho Computer



Đoạn code trên cho chúng ta thấy:

* Ban đầu, Gán cho điểm Max=0
* Lần lượt duyệt các ô chưa được đánh (chưa được sở hữu) trên bàn cờ để xác định điểm tấn công (DiemTanCong) và điểm phòng ngự(DiemPhongNgu) trên mỗi ô cờ. Điểm tấn công và điểm phòng ngự được tính bằng phương thức DiemTanCong\_[…] và DiemPhongNgu\_[…] bên dưới.
* Tạo ra một biến tạm (DiemTam) để lấy giá trị lớn nhất giữa điểm tấn công và điểm phòng thủ.
* So sánh điểm Max và điểm tạm để lấy giá trị lớn nhất và lưu vị trí ô cờ trên bàn cờ tại nơi có điểm lớn nhất.

Cứ như thế cho đến khi duyệt hết bàn cờ ta lấy được vị trí “tốt” nhất mà computer có thể đi.

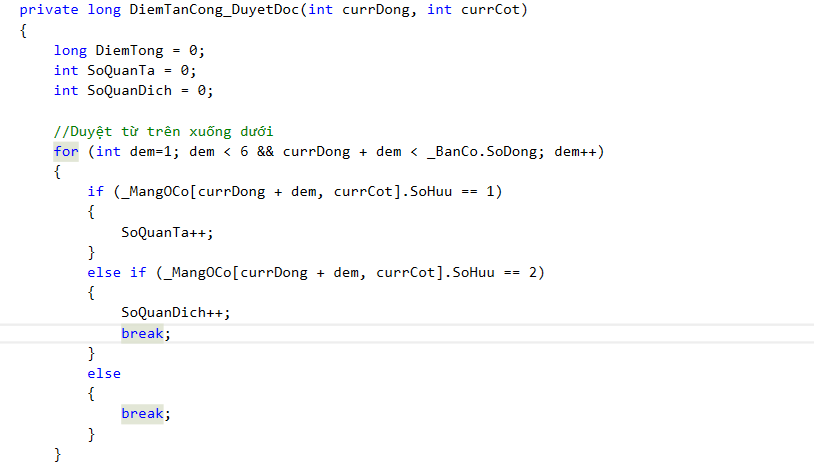
#### 3.3.3.2 Mảng điểm tấn công và mảng điểm phòng ngự

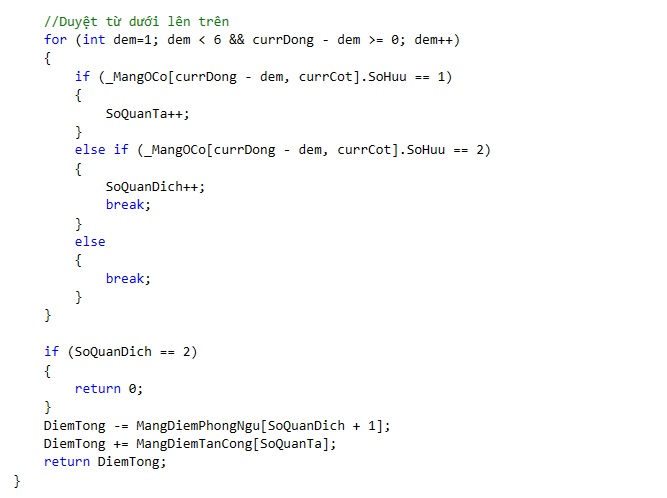


* Mảng điểm tấn công (MangDiemTanCong) dùng để tính điểm tấn công tại mỗi ô cờ cho computer.
* Mảng điểm phòng ngự (MangDiemPhongNgu) dùng để tính điểm phòng ngự tại mỗi ô cờ cho computer.

Các chỉ số trong mảng tùy thuộc vào người lập trình để tạo ra độ khó cũng như phương thức cho computer chơi tấn công hay phòng thủ với người chơi.

#### 3.3.3.3 Phương thức tính điểm tấn công



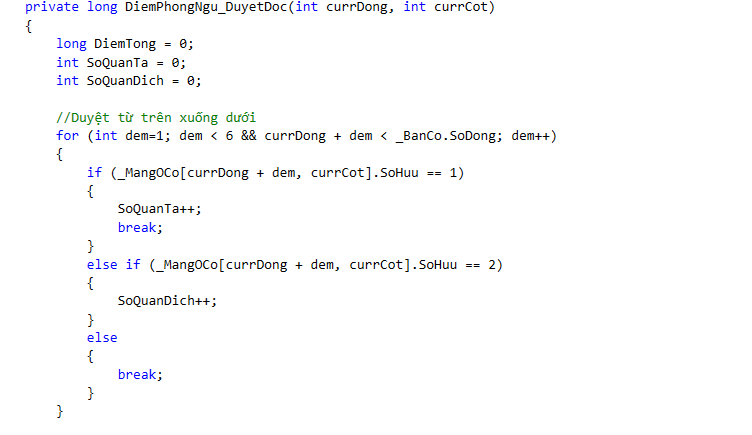


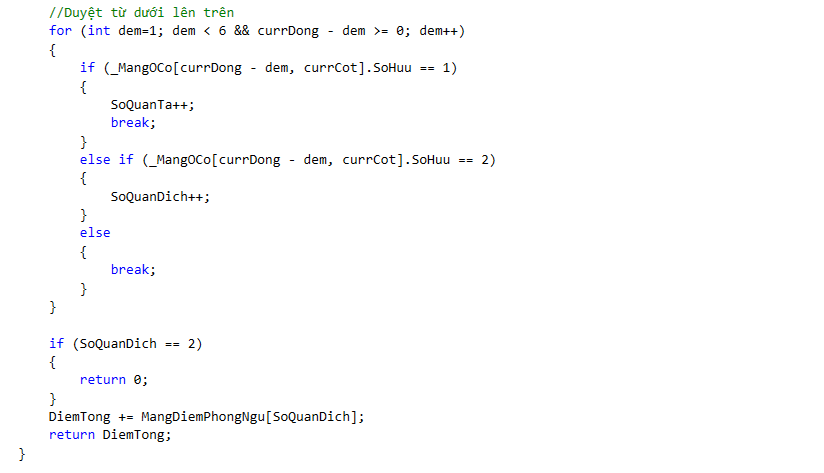
Đoạn code tính điểm tấn công được duyệt dọc được hiểu như sau:

* Ban đầu, ta gán các điểm tổng (DiemTong), Số quân ta (SoQuanTa) và quân địch (SoQuanDich) đều bằng 0.
* Lần lượt duyệt các ô cờ, Nếu tại vị trí cần tính điểm mà là quân của computer thì tăng số quân ta lên 1. Ngược lại, nếu sở hữu là quân của người chơi thì tăng số quân địch lên 1. Cứ thế, lần lượt duyệt hết các ô cần tính.
* Tính điểm tổng bằng cách dựa vào số quân ta và quân địch. Dựa vào mảng điểm tấn công. Ta lấy tổng các điểm tấn công trừ cho tổng các điểm phòng ngự. Ta được điểm tấn công.

Tương tự như cách trên, Ta tính được điểm tấn công khi xét theo duyệt ngang, Duyệt Chéo Xuôi, Duyệt Chéo Ngược.

#### 3.3.3.4 Phương thức tính điểm phòng ngự





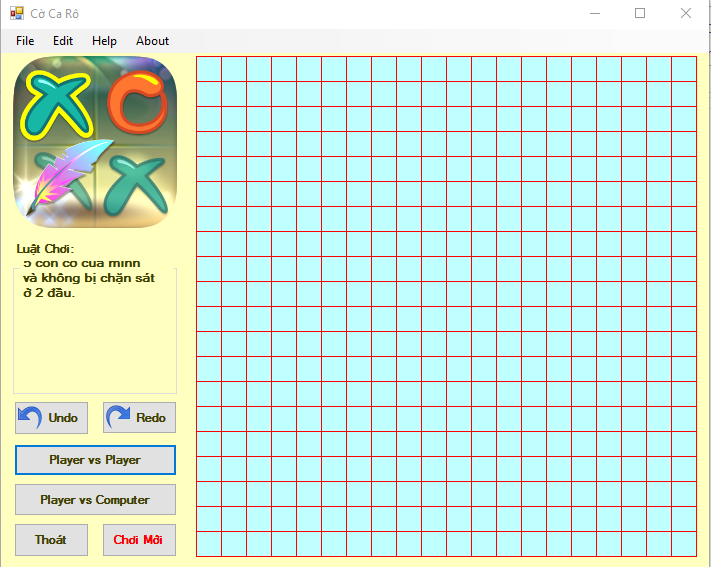
Đoạn code tính điểm tấn công được duyệt dọc được hiểu như sau:

* Ban đầu, ta gán các điểm tổng (DiemTong), Số quân ta (SoQuanTa) và quân địch (SoQuanDich) đều bằng 0.
* Lần lượt duyệt các ô cờ, Nếu tại vị trí cần tính điểm mà là quân của computer thì tăng số quân ta lên 1. Ngược lại, nếu sở hữu là quân của người chơi thì tăng số quân địch lên 1. Cứ thế, lần lượt duyệt hết các ô cần tính.
* Tính điểm tổng bằng cách dựa vào số quân địch. Dựa vào mảng điểm tấn công. Ta lấy tổng các điểm phòng ngự. Ta được điểm phòng ngự.
* Nếu số quân địch chỉ bằng 2 thì ta không tính điểm phòng ngự

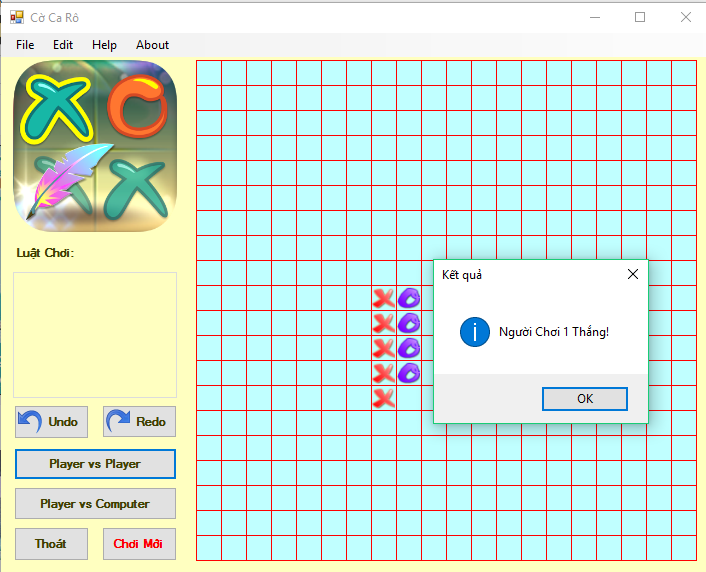
Tương tự như cách trên, Ta tính được phòng ngự khi xét theo duyệt ngang, Duyệt Chéo Xuôi, Duyệt Chéo Ngược.

### 3.3.4 Giao diện chương trình

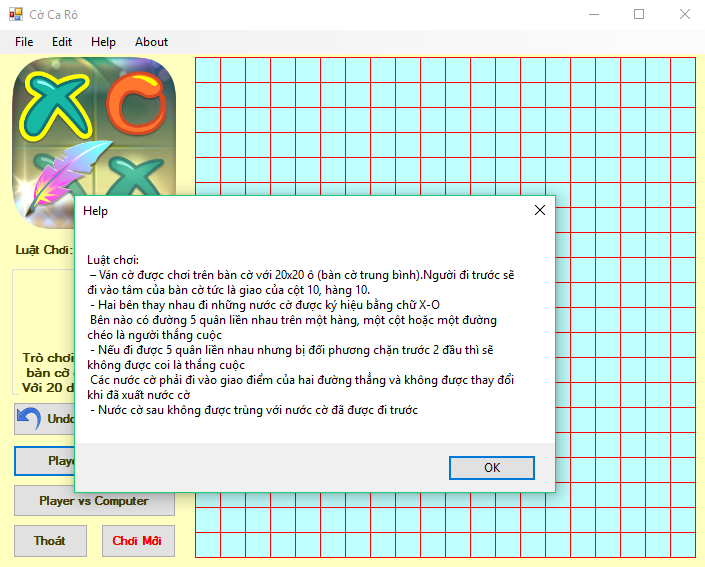
#### 3.3.4.1 Màn hình chính



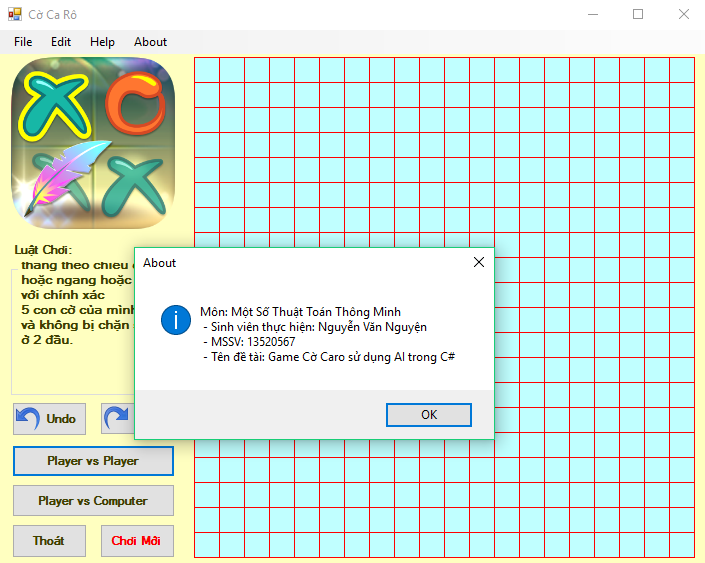
#### 3.3.4.2 Màn hình xác định thắng thua



#### 3.3.4.3 Màn hình thông tin về luật chơi



#### 3.3.4.4 Màn hình thông tin chương trình



# NHẬN XÉT VÀ KẾT LUẬN

# **HƯỚNG PHÁT TRIỂN**

* Xây dựng game chia theo các cấp độ của Computer
* Xây dựng giao diện thân thiện hơn
* Phát triển ứng dụng trên mobile
* Fix các lỗi tiềm ẩn trong chương trình

# **TÀI LIỆU THAM KHẢO**

[1] https://<vi.wikipedia.org/wiki/Lý_thuyết_trò_chơi>

[2] Youtube

[3] Khóa học C# nâng cao của Myclass.vn

[4] Giáo trình Một số thuật toán thông minh của PSG. TS Vũ Thanh Nguyên, Trường Đại Học Công Nghệ Thông Tin

[5] Một số nguồn tài liệu khác trên mạng