Mục Lục

[Chương 1: Trò chơi đối kháng và các phương pháp tìm kiếm 3](#_Toc86833283)

[1.1. Dạng trò chơi 3](#_Toc86833284)

[1.2. Cây trò chơi 4](#_Toc86833285)

[1.3. Vét cạn 5](#_Toc86833286)

[1.4. Chiến lược tìm kiếm trong trò chơi 6](#_Toc86833287)

[Chương 2: Thuật toán Minimax 7](#_Toc86833288)

[2.1. Định nghĩa thuật toán Minimax 7](#_Toc86833289)

[2.2. Thủ tục thuật toán Minimax 7](#_Toc86833290)

[2.3. Định lý Minimax và các bước đi tiếp theo 11](#_Toc86833291)

[2.4. Đánh giá thuật toán Minimax 12](#_Toc86833292)

[**2.4.1.** **Ưu điểm** 12](#_Toc86833293)

[**2.4.2.** **Khuyết điểm** 12](#_Toc86833294)

[Chương 3: Trò chơi Tic Tac Toe (carô 3X3) 13](#_Toc86833295)

[3.1. Giới thiệu trò chơi 13](#_Toc86833296)

[3.2. Mục tiêu trò chơi 13](#_Toc86833297)

[3.3. Hướng giải quyết 14](#_Toc86833298)

[3.4. Áp dụng thuật toán Minimax 17](#_Toc86833299)

[**3.4.1.** **Mô tả thuật toán Minimax áp dụng trong trò chơi** 17](#_Toc86833300)

[**3.4.2.** **Code demo** 19](#_Toc86833301)

[3.5. Giới thiệu phần mềm 24](#_Toc86833302)

[**3.5.1.** **Giao diện và một số chức năng chính của phần mềm** 24](#_Toc86833303)

[**3.5.2.** **Các trạng thái kết thúc** 25](#_Toc86833304)

[Chương 4: Tài liệu tham khảo 25](#_Toc86833305)

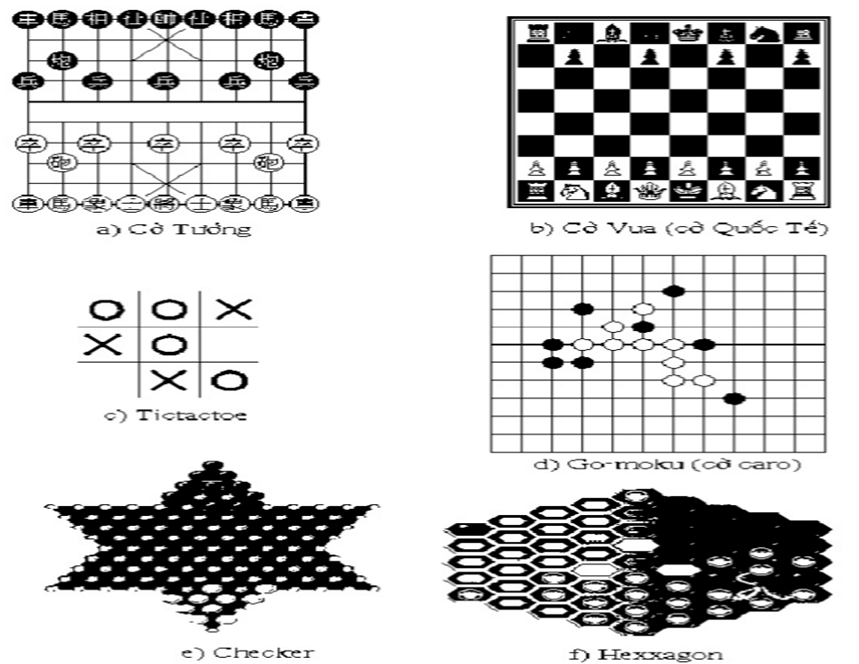
# Chương 1: Trò chơi đối kháng và các phương pháp tìm kiếm

* 1. **Dạng trò chơi**

Trong phần này, ta sẽ xem cách một chương trình máy tính có thể chơi được các trò chơi đấu trí như các trò chơi cờ Vua, cờ Tướng, cờ vây, cờ caro, go, checker... như thế nào. Các trò này còn gọi là các trò chơi đối kháng, diễn ra giữa hai đấu thủ. Nói chung, các trò chơi đó đều có thể chuyển về một dạng bài toán tìm kiếm đặc biệt: tìm đường đi đến các điểm cao nhất giữa hai đấu thủ. Đặc điểm của các trò chơi trên như sau:

* Có hai đấu thủ, mỗi người chỉ đi một nước khi tới lượt.
* Các đấu thủ đều biết mọi thông tin về tình trạng trận đấu.
* Trận đấu không kéo dài vô tận, phải diễn ra hòa, hoặc một bên thắng và bên kia thua.

Thông thường ta hay gọi các trò chơi này là các loại cờ. Đôi khi ta gọi đây là các trò chơi Minimax (dựa trên tên của thuật toán tìm kiếm cơ bản áp dụng cho chúng). Hình 1.1 là ví dụ về một số trò chơi nói trên. Các trò chơi như chơi bài, dò mìn, xúc sắc... không thuộc lớp trò chơi này.

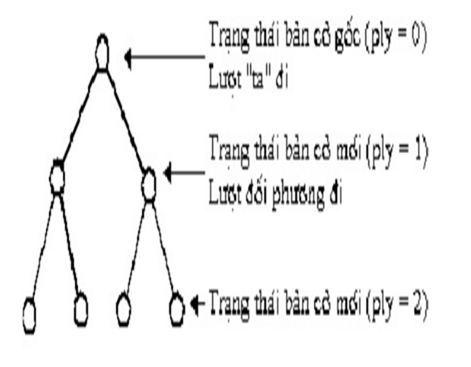
****

**Hình 1.1.1: Một số trò chơi đối kháng**

* 1. **Cây trò chơi**

Các trạng thái bàn cờ khác nhau (hay còn gọi là một thế cờ, tình huống cờ) trong quá trình chơi có thể biểu diễn thành một cây tìm kiếm (được gọi là cây trò chơi - hình 1.2) và ta sẽ tiến hành tìm kiếm trên cây để tìm được nước đi tốt nhất. Cây trò chơi có các nút của cây là các tình huống khác nhau của bàn cờ, các nhánh nối giữa các nút sẽ cho biết từ một tình huống bàn cờ chuyển sang tình huống khác thông qua chỉ một nước đi đơn nào đó.

Dĩ nhiên, các nước đi này diễn ra theo cặp do hai đấu thủ lần lượt tiến hành. Độ sâu của cây trò chơi ply là số tầng của cây (chính là độ sâu d của cây). Thuật ngữ “nước đi” trong sách được thống nhất chỉ bao gồm một lần đi của một đấu thủ hoặc một lần đi phản ứng lại của đối thủ bên kia. Chú ý nó khác với thói quen dùng trong thực tế một nước đi bao gồm lần đi của ta và một lần đi của đối thủ. Nói cách khác, nước đi ở đây thực chất chỉ là "nửa nước" theo cách hiểu của làng cờ.

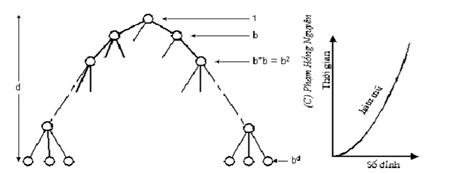


**Hình 1.2.1: Cây trò chơi**

* 1. **Vét cạn**

Dùng một thuật toán vét cạn để tìm kiếm trên cây trò chơi dường như là một ý tưởng đơn giản. Ta chỉ cần chọn nhánh cây sẽ dẫn tới nước thắng để đi quân là đảm bảo thắng lợi. Nếu đúng vậy, các loại cờ sẽ trở thành các trò chơi buồn tẻ, sẽ chẳng còn đâu những bí quyết huyền ảo thần kì và bàn cờ sẽ chẳng khác gì bàn... tính. Rất tiếc (hoặc rất may) rằng, cách làm này lại không thể thực hiện nổi do cái gọi là bùng nổ tổ hợp. Ví dụ, nếu từ một thế cờ, trung bình có khả năng đi được 16 nước đi khác nhau (ta gọi đó là hệ số nhánh con tại mỗi nút là b = 16). Như vậy, sau một tầng ta sẽ có 16 nút con, mỗi nút này lại có thể có 16 con nữa. Tổng số nút con ở độ sâu thứ hai là 16x16 = b2 . Cứ như vậy ở độ sâu d sẽ có b^d nút.

Nếu giả sử độ sâu của cây là 100 (hệ số nhánh 16 và độ sâu 100 đều là những con số còn nhỏ hơn con số thường gặp trong trò chơi cờ), thì số nhánh phải duyệt lên đến 16100 hay xấp xỉ 10120 - một con số lớn khủng khiếp. Để hình dung số đó lớn thế nào, ta giả sử tất cả các nguyên tử trong vũ trụ đều trở thành máy tính để tính nước đi với tốc độ một giây tính được cỡ 1010 (10 tỷ) nước đi, và nếu chúng hoạt động cật lực từ thời vụ nổ lớn đến nay (theo một số lý thuyết, thì thế giới này hình thành sau một vụ nổ gọi là vụ nổ lớn bigbang, trước đây cỡ 15 tỷ năm) thì đến bây giờ mới có thể đi được nước đi đầu tiên. Vì số các khả năng tăng quá nhanh, chỉ có một số ít những vấn đề đơn giản là thích hợp với kiểu tìm kiếm vét hết mọi khả năng này (kiểu tìm kiếm vét cạn đòi hỏi phải kiểm tra tất cả các đỉnh). Do đó, các phương pháp tìm kiếm khác đã ra đời và phát triển. Ngược lại, nếu có một phương pháp luôn luôn chính xác nhằm đánh giá một thế cờ này là tốt hay kém so với thế kia, thì trò chơi trở thành đơn giản bằng cách chọn nước đi dẫn tới thế cờ tốt nhất. Do đó sẽ không cần phải tìm kiếm gì nữa. Rất tiếc, các thủ tục như vậy không hề có. Ta cần có chiến lược tìm kiếm trong trò chơi.



**Hình 1.3.1: Cây tìm kiếm và sự bùng nổ tổ hợp**

* 1. **Chiến lược tìm kiếm trong trò chơi**

Một chiến lược thường được cả người lẫn máy dùng là phân tích thế cờ chỉ sau một số nước đi nào đó của cả hai bên. Sau khi "nhìn xa" xem bàn cờ có những khả năng biến đổi như thế nào sau một số nước, ta sẽ đánh giá độ xấu tốt của các thế cờ nhận được. Tiếp theo, ta sẽ chọn nước đi sẽ dẫn tới một thế cờ tốt nhất trong số đó có cân nhắc đến cách đi của cả hai bên. Với máy, thế cờ này được đánh giá là tốt hơn thế cờ kia nhờ so sánh điểm của thế đó do bộ lượng giá trả lại. Chúng ta chỉ có khả năng xét trước một số hữu hạn cácnước (ví dụ đại kiện tướng chơi cờ vua có thể xét trước 8-10 nước đi, người thường chỉ 2- 4 nước đi). Rõ ràng là nếu xét càng sâu thì chơi càng giỏi. Nhưng không thể thực hiện điều này với độ sâu quá lớn được do số nút ở độ sâu đó có thể trở nên lớn khủng khiếp và không đủ thời gian để phân tích. Nếu dừng ở một độ sâu hợp lý thì bộ phân tích có thể hoàn thành việc tính toán trong một thời gian hạn định.

# Chương 2: Thuật toán Minimax

* 1. **Định nghĩa thuật toán Minimax**

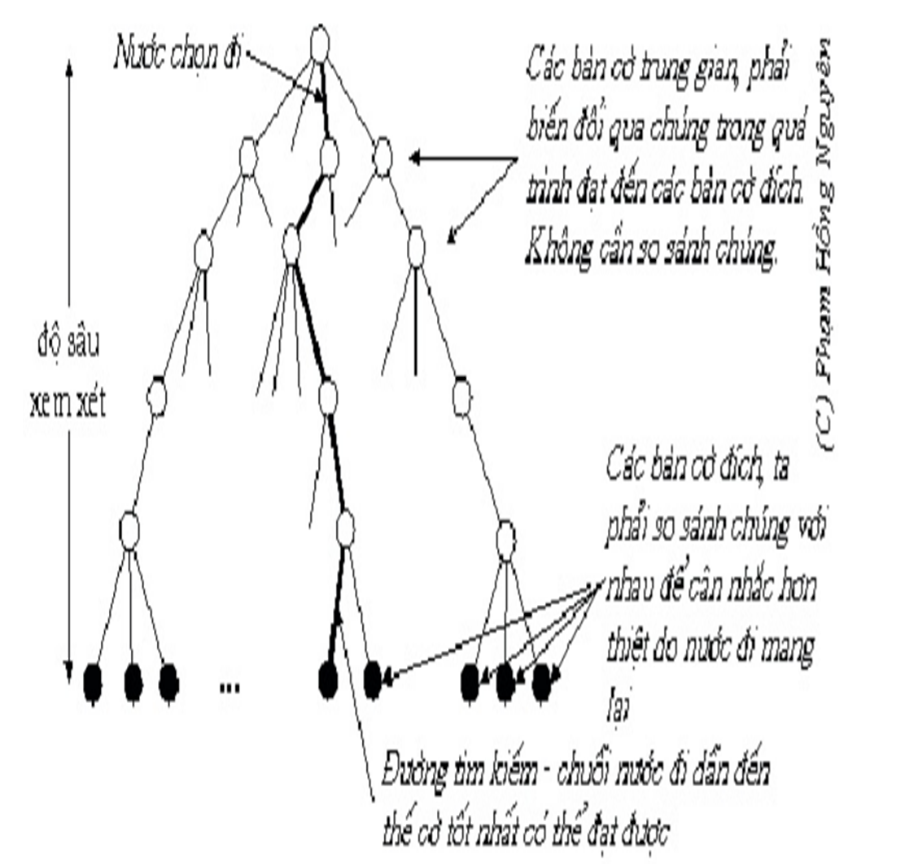
Minimax (còn gọi là minmax) là một phương pháp trong lý thuyết quyết định, có mục đích là tối thiểu hóa (minimize) tổn thất vốn được dự tính có thể là "tối đa" (maximize). Có thể hiểu ngược lại là, nó nhằm tối đa hóa lợi ích vốn được dự tính là tối thiểu (maximin). Nó bắt nguồn từ trò chơi có tổng bằng không. Nó cũng được mở rộng cho nhiều trò chơi phức tạp hơn và giúp đưa ra các quyết định chung khi có sự hiện diện của sự không chắc chắn.

Ta sẽ nắm rõ, thế nào là một nước đi "tốt nhất". Giải thuật Minimax giúp tìm ra nước đi tốt nhất, bằng cách đi ngược từ cuối trò chơi trở về đầu. Tại mỗi bước, nó sẽ ước định rằng người A đang cố gắng tối đa hóa cơ hội thắng của A khi đến phiên anh ta, còn ở nước đi kế tiếp thì người chơi B cố gắng để tối thiểu hóa cơ hội thắng của người A (nghĩa là tối đa hóa cơ hội thắng của B).

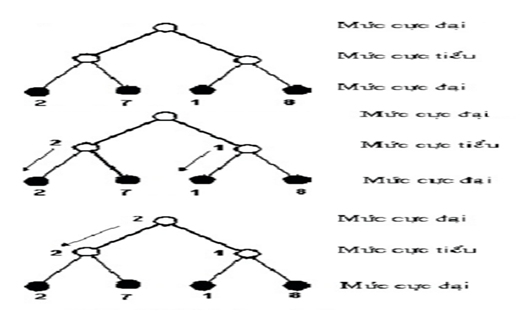
Một thuật toán minimax là một thuật toán đệ quy cho việc lựa chọn bước đi kế tiếp trong một trò chơi có hai người chơi. Một giá trị được gán cho mỗi vị trí hay một trạng thái của trò chơi. Giá trị này được tính toán bằng một hàm tính giá trị vị trí và nó cho biết độ tốt nếu như một người chơi đạt được đến đó. Người chơi sau đó đi một bước làm tối đa giá trị tối thiểu của vị trí là kết quả từ tập hợp những bước đi có thể của đối thủ. Nếu đó là phiên A sẽ đi, A sẽ cho một giá trị cho mỗi bước đi hợp pháp của anh ta.

* 1. **Thủ tục thuật toán Minimax**

Giả sử chúng ta có một bộ phân tích thế cờ có thể áp dụng tất cả các luật, các phương pháp đánh cờ khác nhau vào từng thế cờ và chuyển đổi chúng thành một con số đại diện (cho điểm thế cờ). Mặt khác, ta giả sử con số đó là dương khi áp dụng cho thế cờ của một đấu thủ (được gọi là người chơi cực đại - maximizer), và là âm khi áp dụng cho đấu thủ bên kia (được gọi là người chơi cực tiểu - minimizer). Quá trình tính toán cho điểm thế cờ được gọi là lượng giá tĩnh (static evaluation). Hàm thực hiện việc tính toán được gọi là một bộ lượng giá tĩnh, và giá trị nhận được gọi là điểm lượng giá tĩnh. Cả hai đấu thủ cờ được gọi là lượng giá tĩnh (static evaluation). Hàm thực hiện việc tính toán được gọi là một bộ lượng giá tĩnh, và giá trị nhận được gọi là điểm lượng giá tĩnh. Cả hai đấu thủ đều cố gắng đi như thế nào đó để đạt được điểm tuyệt đối lớn nhất. Người chơi cực đại sẽ tìm những nước đi dẫn đến điểm của mình trở nên lớn hơn (hay cao nhất có thể được) hay điểm của đối thủ bớt âm hơn (nhỏ hơn về giá trị tuyệt đối). Còn đấu thủ của anh ta, người chơi cực tiểu, lại ra sức phản kháng lại, để dẫn tới điểm âm của anh ta âm hơn hay điểm dương của đối thủ nhỏ đi

****

**Hình 2.2.1: Thủ tục thuật toán Minimax**

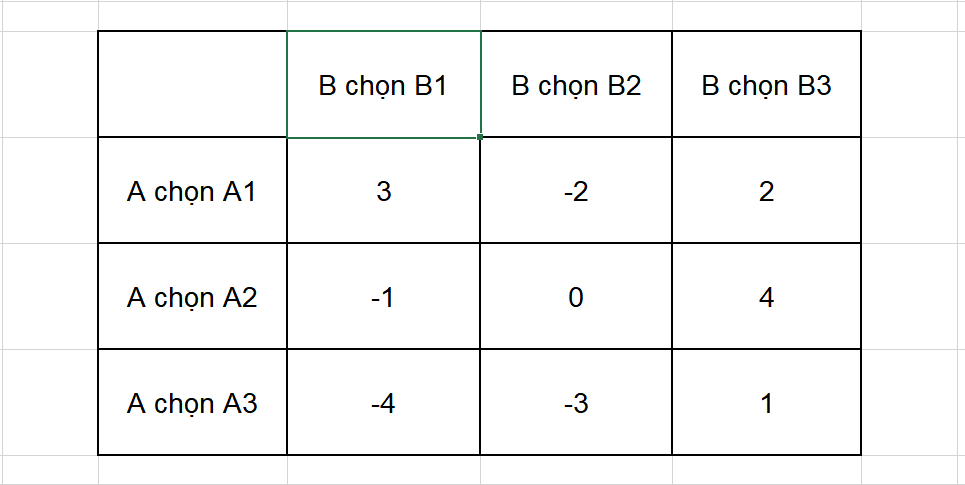


**Hình 2.2.2: Ví dụ một phần cây trò chơi áp dụng thuật toán Minimax**

Người chơi cực đại hi vọng chọn nước đi bên phải để đạt được điểm 8. Thế nhưng nếu đi như vậy thì khi đến lượt đi của người chơi cực tiểu, anh ta sẽ cố gắng không cho người chơi cực đại đạt được điểm này bằng cách chọn nước đi nhánh bên trái và như vậy, người chơi cực đại chỉ được có 1 điểm thay vì 8. Ngược lại, nếu người chơi cực đại chọn nước đi bên trái, thì trong tình huống xấu nhất anh ta vẫn còn được 2 điểm, lớn hơn là chọn nước đi bên phải. Nói chung, người chơi cực đại sẽ phải tìm cách nhận ra các nước đi của đối phương tiếp theo làm cho điểm giảm xuống. Và tương tự như vậy, người chơi cực tiểu phải nhận biết được nước đi của người chơi cực đại cố gắng làm tăng điểm lên. Thủ tục tìm nước đi tốt nhất trên cây trò chơi như trên được gọi là thủ tục Minimax do điểm ở mỗi nút có thể là điểm cực đại hoặc có thể là điểm cực tiểu và có thuật toán như sau:

* Nếu như đạt đến giới hạn tìm kiếm (đến tầng dưới cùng của cây tìm kiếm), tính giá trị tĩnh của thế cờ hiện tại ứng với người chơi ở đó. Ghi nhớ kết quả.
* Nếu như mức đang xét là của người chơi cực tiểu, áp dụng thủ tục Minimax này cho các con của nó. Ghi nhớ kết quả nhỏ nhất.
* Nếu như mức đang xét là của người chơi cực đại, áp dụng thủ tục Minimax này cho các con của nó. Ghi nhớ kết quả lớn nhất.
  1. **Định lý Minimax và các bước đi tiếp theo**

Trong ví dụ sau đây của một trò chơi tổng bằng 0, khi A và B đi các bước cùng một lúc, minh họa thuật toán minimax. Nếu như mỗi người chơi có 3 chọn lựa và ma trận lợi cho A là:



và B có ma trận có lợi như nhau nhưng ngược dấu (i.e. nếu các lựa chọn là A1 và B1 thì B trả 3 cho A) sau đó lựa chọn minimax đơn giản cho A là A2 bởi vì kết quả xấu nhất là sau khi phải trả 1, trong khi lựa chọn minimax đơn giản cho B là B2 bởi vì kết quả xấu nhất là sau đó không phải trả gì cả. Tuy vậy, lời giải này là không ổn định, bởi vì nếu B tin rằng A sẽ chọn A2 thì B sẽ chọn B1 để thắng 1; sau đó nếu A tin rằng B sẽ chọn B1 thì A sẽ chọn A1 để thắng 3; và sau đó B sẽ chọn B2; và cuối cùng cả hai người chơi sẽ nhận ra sự khó khăn của việc chọn lựa. Do đó một chiến lược ổn định hơn là cần thiết.

Một số chọn lựa bị thống trị bởi những người khác và có thể bị loại bỏ: A sẽ không chọn A3 bởi vì hoặc A1 hay A2 sẽ sinh ra một kết quả tốt hơn, bất kể là B chọn gì; B sẽ không chọn B3 bởi vì B2 sẽ sinh ra kết quả tốt hơn, bất kể là A chọn cái gì.

A có thể tránh việc phải trả số lượng dự định (expected payment) hơn 1/3 bằng cách chọn A1 với xác suất 1/6 và A2 với xác suất 5/6, bất kể là B đã chọn gì. B có thể tính chắc phần lợi dự định (expected gain) ít nhất 1/3 bằng cách sử dụng một chiến thuật ngẫu nhiên của việc chọn B1 với xác suất 1/3 và B2 với xác suất 2/3, bất kể là A chọn gì. Những chiến lược minimax hỗn hợp bây giờ là ổn định và không thể nào cải tiến nữa.

* 1. **Đánh giá thuật toán Minimax**
     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.

* + 1. **Khuyết điểm**

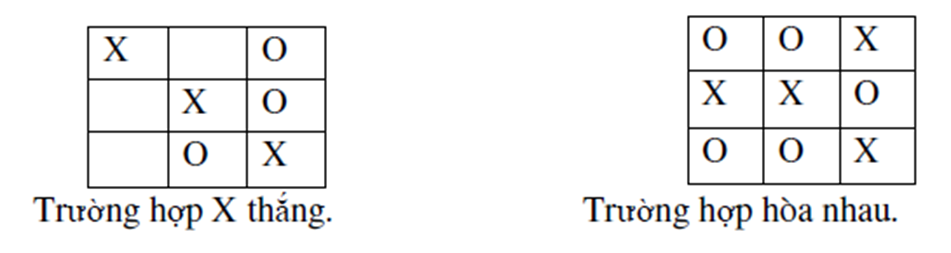
Đối với các trò chơi có không gian trạng thái lớn như cờ vua, cờ tướng… việc “chỉ” áp dụng giải thuật Minimax có lẽ không còn hiệu quả nữa do sự bùng nổ tổ hợp quá lớn.

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**.**

# Chương 3: Trò chơi Tic Tac Toe (carô 3X3)

* 1. **Giới thiệu trò chơi**

Trò chơi này được chơi bởi 2 đối thủ với 9 ô. Một trong 2 đối thủ sẽ đi trước đánh (O) hoặc (X) vào 1 ô bất kỳ trên bàn cờ , đối thủ còn lại chọn 1 trong 8 ô còn lại để đi .Hai đối thủ thay nhau đánh vào các ô trống cho đến khi có 1 đối thủ có 3 ô nằm trên 1 đường thẳng trước thì thắng. Nếu hết 9 ô cờ mà không có đối thủ nào có 3 ô nằm trên 1 đường thẳng thì ván cờ kết thúc với tỷ số hòa.



**Hình 3.1.1: Giới thiệu thuật toán Minimax**

* 1. **Mục tiêu trò chơi**

Đến lượt chơi mỗi người cố gắng để tạo ra 3 quân cờ nằm trên 1 đường thẳng để là người chiến thắng hoặc cố ngăn cản người kia tạo được 3 quân cờ trên một đường thẳng để đưa về kết quả Hòa tối thiểu có lợi cho mình.

* 1. **Hướng giải quyết**

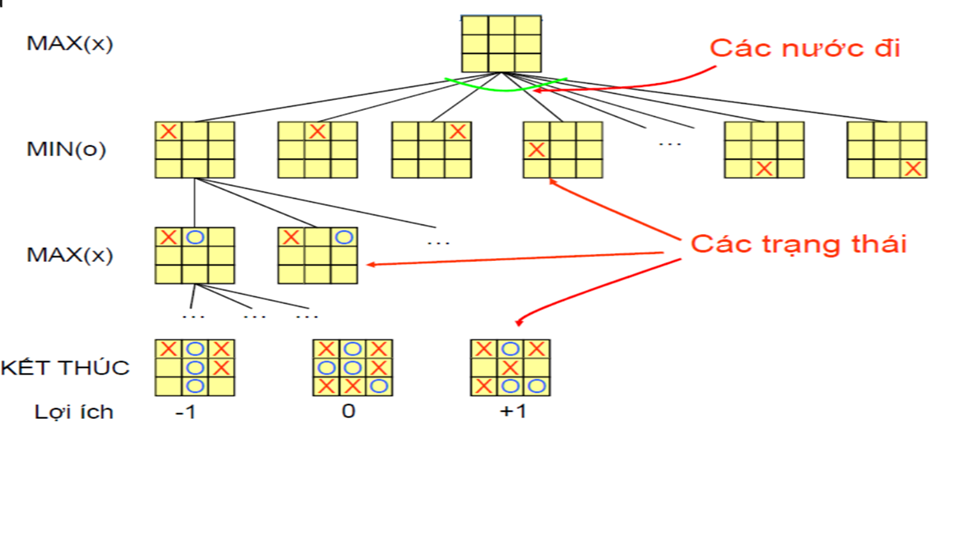
Sử dụng thuật toán Minnimax để tìm nước đi tốt nhất có lợi cho từng người chơi .

Hai đối thủ trong 1 trò chơi được gọi là MIN và MAX. MAX là đại diện cho đối thủ quyết dành thắng lợi hay cố gắng tối ưu hóa ưu thế của mình. Ngược lại MIN là đối thủ cố gắng tối thiểu hóa điểm số của MAX . Ta giả thiết MIN cũng dùng thông tin như MAX.

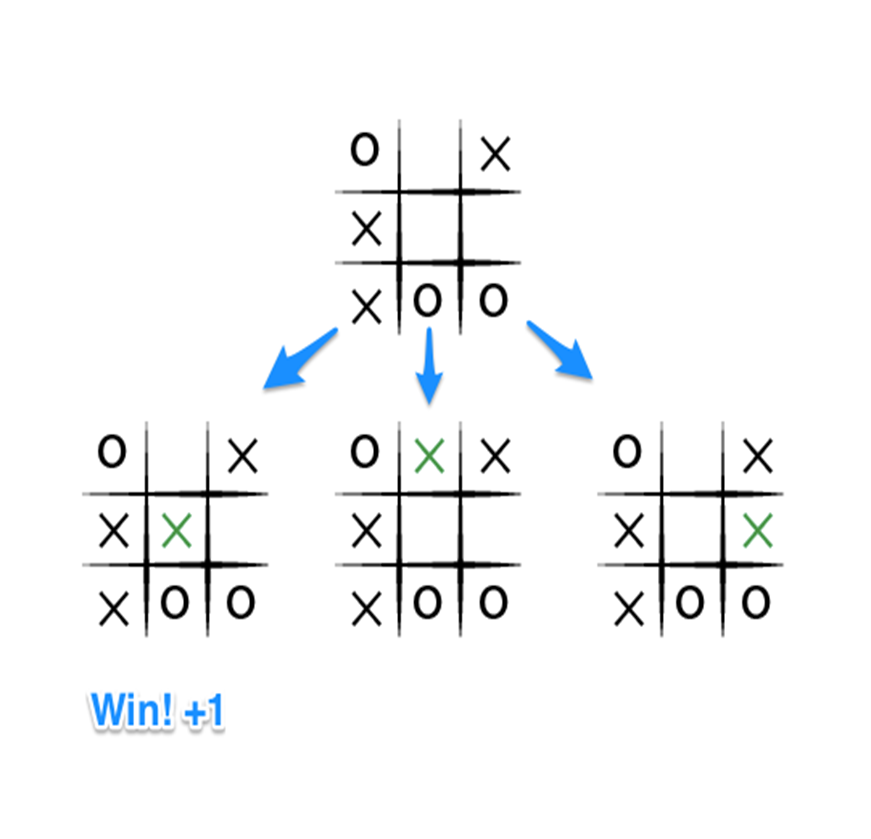
Khi áp dụng thủ tục Minnimax, ta đánh dấu luân phiên từng mức trong không gian tìm kiếm phù hợp với đối thủ có bước đi ở mức đó, từng nút lá được gán giá trị -1 hay 1 tùy theo đó là thắng cuộc với MIN hay MAX. Minimax sẽ truyền các giá trị này lên cao dần trên đồ thị qua các nút cha mẹ kế tiếp nhau theo luật sau:

* Nếu trạng thái cha mẹ là nút MAX , gán cho nó giá trị tối đa của các con cháu của nó.
* Nếu trạng thái cha mẹ là nút MIN , gán cho nó giá trị tối thiểu của con cháu của nó.

Giá trị được gán cho từng trạng thái bằng cách đó sẽ chỉ rõ trạng thái tốt nhất mà đối thủ này có thể đạt được. Các giá trị này sẽ được dùng để lựa chọn các bước đi có thể có. Vì tất cả các nước đi đầu tiên có thể xảy ra cho MIN sẽ dần đến giá trị 1 nên MAX luôn có thể chọn nước đi có lợi cho mình bất kể nước đi đầu tiên của MIN như nào.

**Hình 3.3.1:**

Ví dụ:

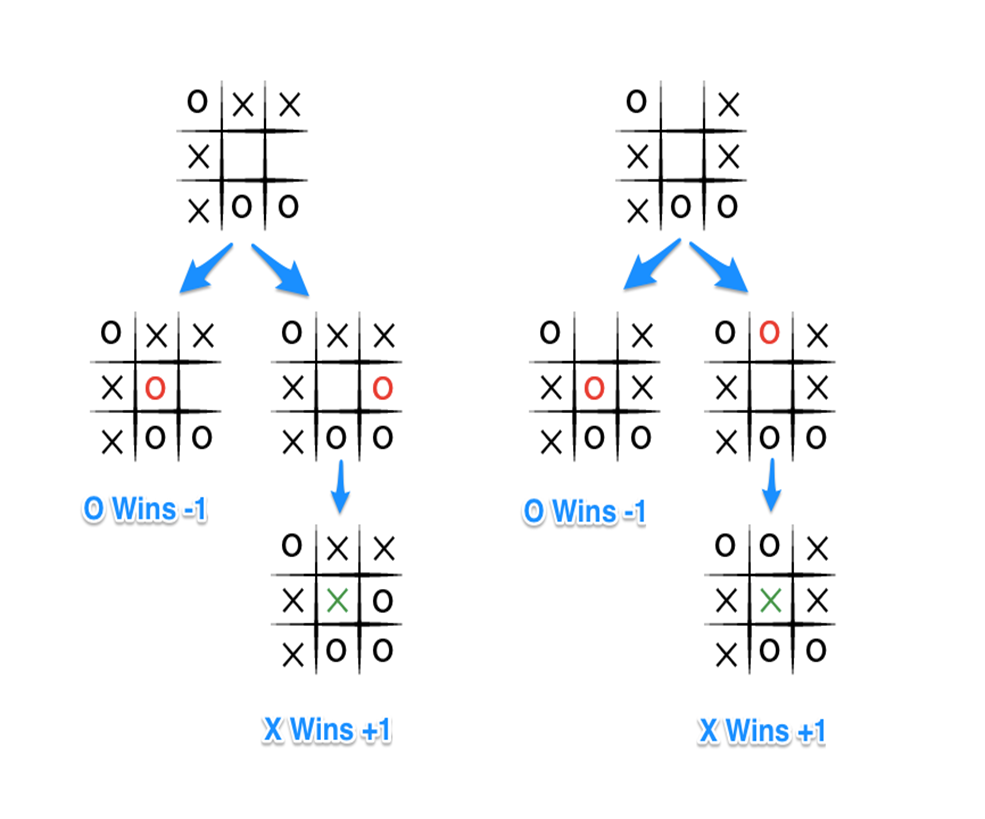


**Hình 3.3.2:**

Để hiểu rõ hơn, hãy lấy một ví dụ từ một ván Tic Tac Toe gần tới hồi kết, đồng thời là lượt đi của mình (mình là X). Mục tiêu của mình rõ ràng là tối ưu hóa điểm số, đồng nghĩa với việc chiến thắng và ra về với 1 điểm.

Hình ở trên miêu tả ba nước đi mình có thể đi khi rơi vào một nước như vậy, và rõ ràng một trong ba nước sẽ làm mình thắng trò chơi và có 1 điểm. Và nếu mình không đi nước đó, thì O sẽ cướp lấy cơ hội và chiến thắng trò chơi này. Cơ mà mình không muốn O thắng, nên mục tiêu của mình, với cơ hội là người đi trước trước khi O lội ngược dòng, là lựa nước đi dẫn tới kết quả thắng.

Còn O thì sao?



**Hình 3.3.3:**

Giả định O muốn chiến thắng trò chơi cũng như chúng ta, O đương nhiên sẽ chọn nước đi bất lợi cho chúng ta nhất (có lợi cho O), đồng nghĩa với việc lựa nước đi tối thiểu hoá điểm số của chúng ta. Hãy nhìn trò chơi dưới góc nhìn của O ở hai trường hợp ở trên, khi mà chúng ta không thể thắng ngay lập tức như trường hợp ở trên.

Rõ ràng O sẽ chọn nước đi để chúng ta bị trừ 1 điểm.

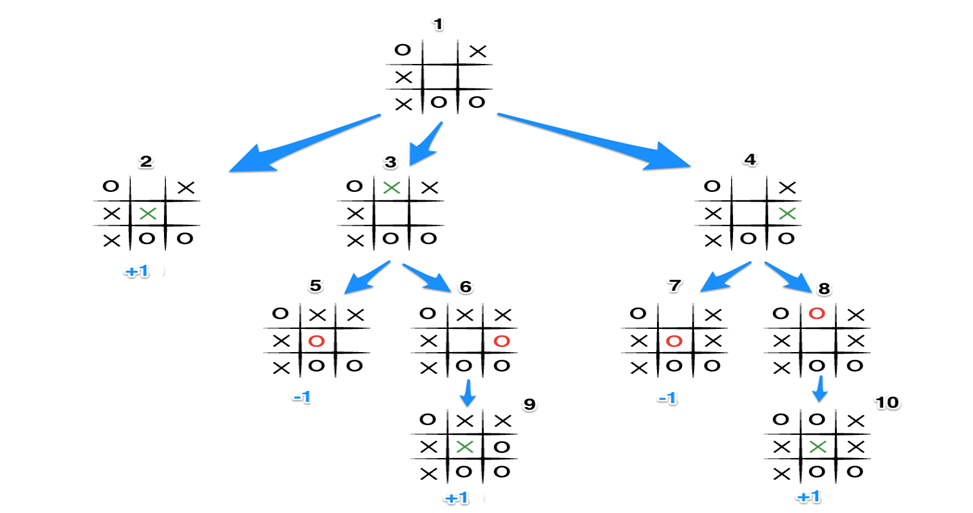
* 1. **Áp dụng thuật toán Minimax**
     1. **Mô tả thuật toán Minimax áp dụng trong trò chơi**

Chìa khóa của thuật toán Minimax chính là lượt chơi qua lại giữa hai người chơi, khi mà cả hai đều mong muốn nước đi có điểm cao nhất. Điểm số của từng nước đi của ta được quyết định bởi đối thủ, người quyết định nước nào sẽ mang lại điểm số nhỏ nhất cho chúng ta. Ngược lại, điểm số của đối thủ lại được quyết định bởi chúng ta, người tìm cách tối ưu hóa điểm số. Nói ngắn gọn thì thuật toán Minimax trong trò chơi Tic Tac Toe chính là thử hết các nước đi có thể cho đến khi trò chơi kết thúc.

Giả dụ đang tới lượt X, thì trò chơi sẽ có quy trình như sau:

* Nếu trò chơi kết thúc, trả về điểm số dưới góc nhìn của X.
* Không thì thu thập các nước đi có thể ở lượt mới.
* Tạo một mảng (một cấu trúc dữ liệu bao gồm một nhóm các giá trị phần tử hoặc biến) chứa các điểm số tương ứng với các nước nêu trên.
* Đối với từng tình huống, cập nhật điểm số tính bởi thuật Minimax vào mảng nêu trên.
* Nếu lượt đi là của X, trả về điểm số lớn nhất trong mảng.
* Nếu lượt đi là của O, trả về điểm số nhỏ nhất trong mảng.
* Bạn sẽ để ý rằng thuật toán Minimax là một thuật toán đệ quy – sự qua lại giữa lượt đi của hai người chơi cho đến khi số điểm cuối cùng được tìm thấy.

Thuật toán Minimax là một thuật toán đệ quy – sự qua lại giữa lượt đi của hai người chơi cho đến khi số điểm cuối cùng được tìm thấy. Hãy cũng xem qua nguyên lý của thuật toán đối với một cây nước đi đầy đủ và chỉ ra làm sao mà nước đi dẫn tới chiến thắng được lựa chọn:



**Hình 3.3.1.1:**

* Tình huống 1 là lượt đi của X: X tạo ra 3 tình huống 2, 3, và 4 và áp dụng thuật toán Minimax trên từng tình huống.
* Tình huống 2 trả về điểm +1 cho mảng điểm của tình huống 1, bởi vì trò chơi đã đến hồi kết.
* Tình huống 3 và 4 chưa tới hồi kết nên lần lượt tạo ra các tình huống 5, 6 và 7, 8 đồng thời áp dụng thuật toán Minimax.
* Tình huống 5 trả về điểm -1 cho mảng điểm của tình huống 3, tương tự với tình huống 7 trả về điểm -1 cho mảng 4.
* Tình huống 6 và tình huống 8 chỉ tạo ra một tình huống, đồng thời là tình huống cuối cùng dẫn tới trò chơi kết thúc với X là người thắng, nên cả hai sẽ lần lượt trả về điểm +1 cho mảng 3 và 4.
* Bởi vì sau tình huống thứ 3 và thứ 4 là lượt đi của O, nên O sẽ đi nước đi dẫn tới điểm số thấp nhất. Bởi vì cả hai tình huống đều có mảng chứa +1 và -1, nên tình huống 3 và 4 sẽ trả về -1 cho mảng điểm của tình huống 1 (điểm thấp nhất).
* Trong mảng điểm của tình huống 1 sẽ chứa các điểm lần lượt là +1, -1, -1. Vì tình huống 1 là lượt đi của X nên X sẽ lựa nước đi có điểm cao nhất, nghĩa là chọn nước đi dẫn tới số điểm +1, nước đi tương tự tình huống thứ 2.

Tuy đơn giản, nhưng quá trình trên cần rất nhiều công sức. Bởi vậy nên chúng ta mới cần máy tính để chạy chương trình thay cho chúng ta.

* + 1. **Code demo**

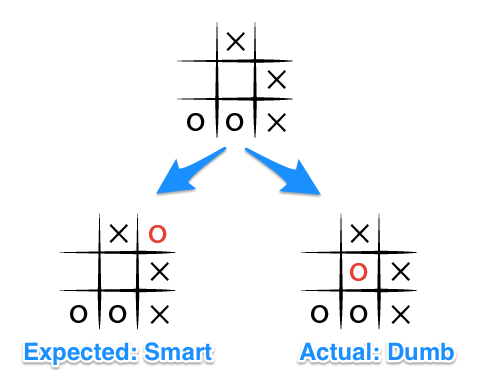


**Hình 3.4.2.1: Demo code**

Đoạn code trên khởi tạo một đối tượng là trạng thái hiện tại của bảng chơi.

Với mỗi trạng thái có một trạng thái sinh ra nó gọi là trạng thái bố mẹ, có một danh sách trạng thái con được sinh ra từ trạng thái hiện tại và một trạng thái con tốt nhất.

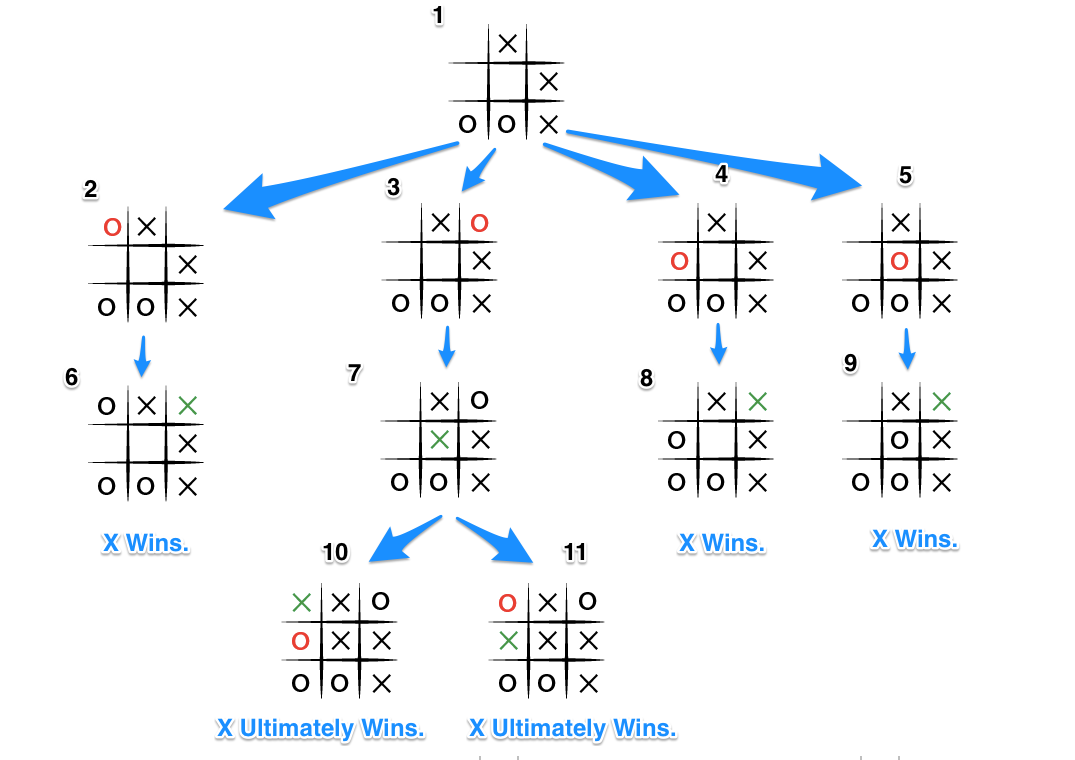
Giờ hãy thử xét một tình huống :



**Hình 3.4.2.2:**

Ở tình huống trên, X còn 1 chỗ đánh nữa là thắng, thay vì O đánh chặn X ] (gây bất lợi cho đối phương) ở ô [0][2] thì trên thực tế O lựa chọn đánh ở ô [1][1] (vị trí có lợi cho mình).

Hãy xem những gì xảy ra tiếp theo (đã lượt bớt một số tình huống) :



**Hình 3.4.2.3:**

* Giả định tình huống 1 khi cả hai cùng chơi một cách hoàn hảo, và O là máy tính.
* Chọn bước đi như tình huống 5 và thua ngay lập tức khi X chọn như tình huống 9.
* Nhưng nếu O chặn X như tình huống 3, X sẽ chặn O như tình huống 7.
* Chiến thắng chắc chắn cho X như tình huống 10 và 11. Như vậy, bất kể O đi nước nào trong tình huống 7, X vẫn thắng.

Bởi lẽ chúng ta đang chạy vòng lặp cho từng ô trống, từ trái sang phải, từ trên xuống dưới, tất cả nước đi đều tương đương nhau và dẫn tới phần thua cho O, nước đi trong tình huống 5 được chọn bởi vì nó là trạng thái cuối cùng trong tất cả các trạng thái con ở tình huống 1.

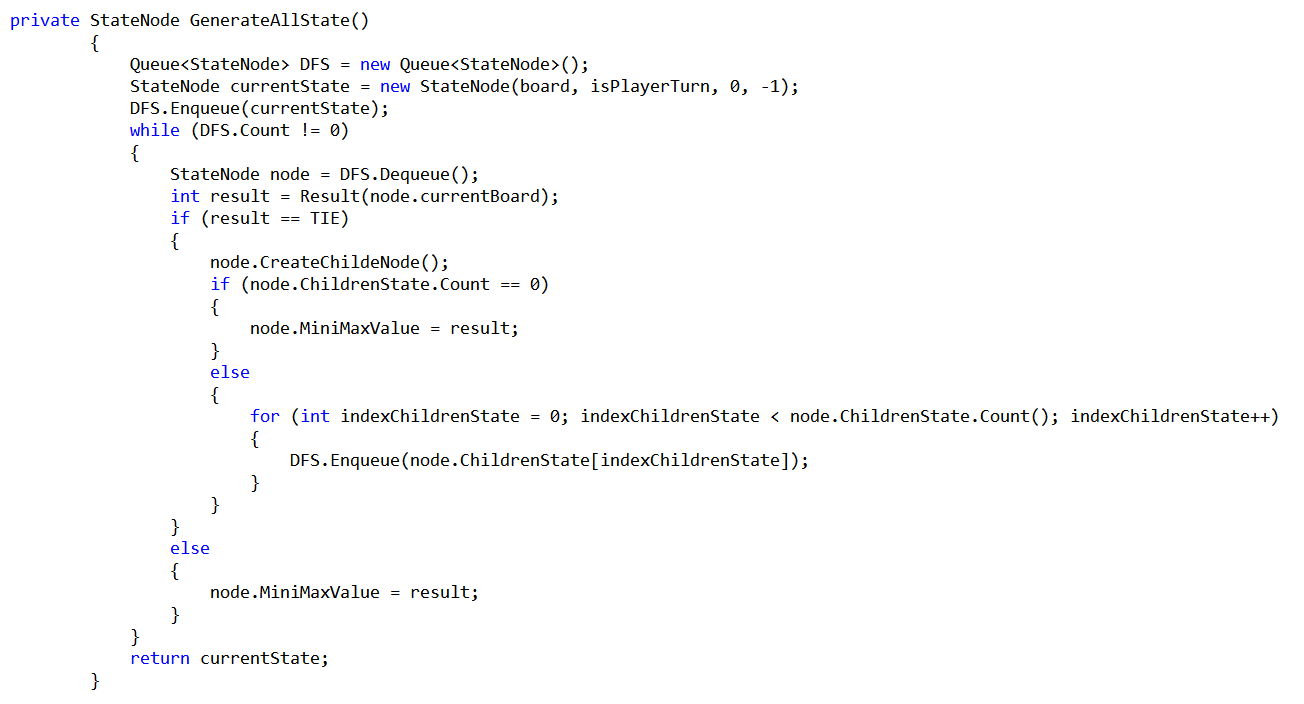
Và vì điều này, một chương trình Tic Tac Toe muốn vươn tới sự hoàn hảo buộc phải cải tiến. Chơi kiên cường (Duyệt chiều sâu) sẽ hỗ trợ cho điều này.

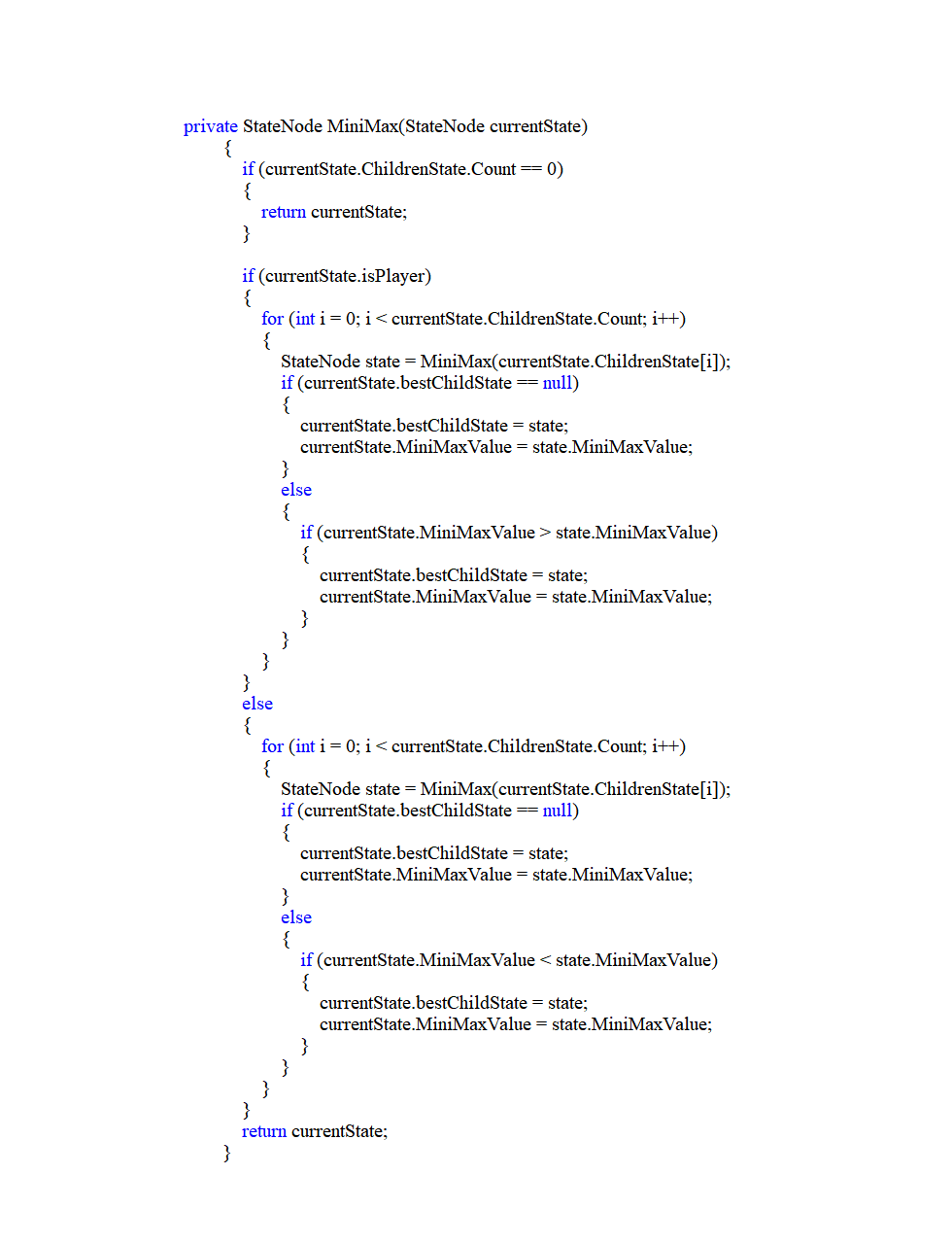
Chìa khóa để cải thiện thuật toán này, khiến cho máy tính kháng cự cho tới khi trò chơi thật sự ngã ngũ, chính là cân nhắc chiều sâu, hay nói cách khác là số lượt, trong quá trình tính toán. Về cơ bản thì người chơi hoàn hảo phải chơi một cách hoàn hảo, song phải kéo dài trò chơi lâu nhất có thể.

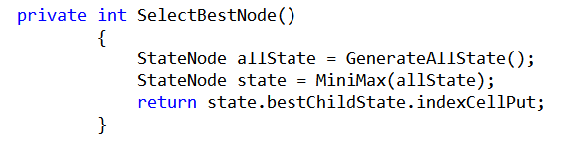
Và thuộc tính LEVEL (trong class StateNode) của các trạng thái con ( mỗi khi được sinh ra) sẽ bằng LEVEL(parent)+1. Khi đó, nước đi dẫn đến trạng thái có LEVEL cao nhất sẽ được ưu tiên.



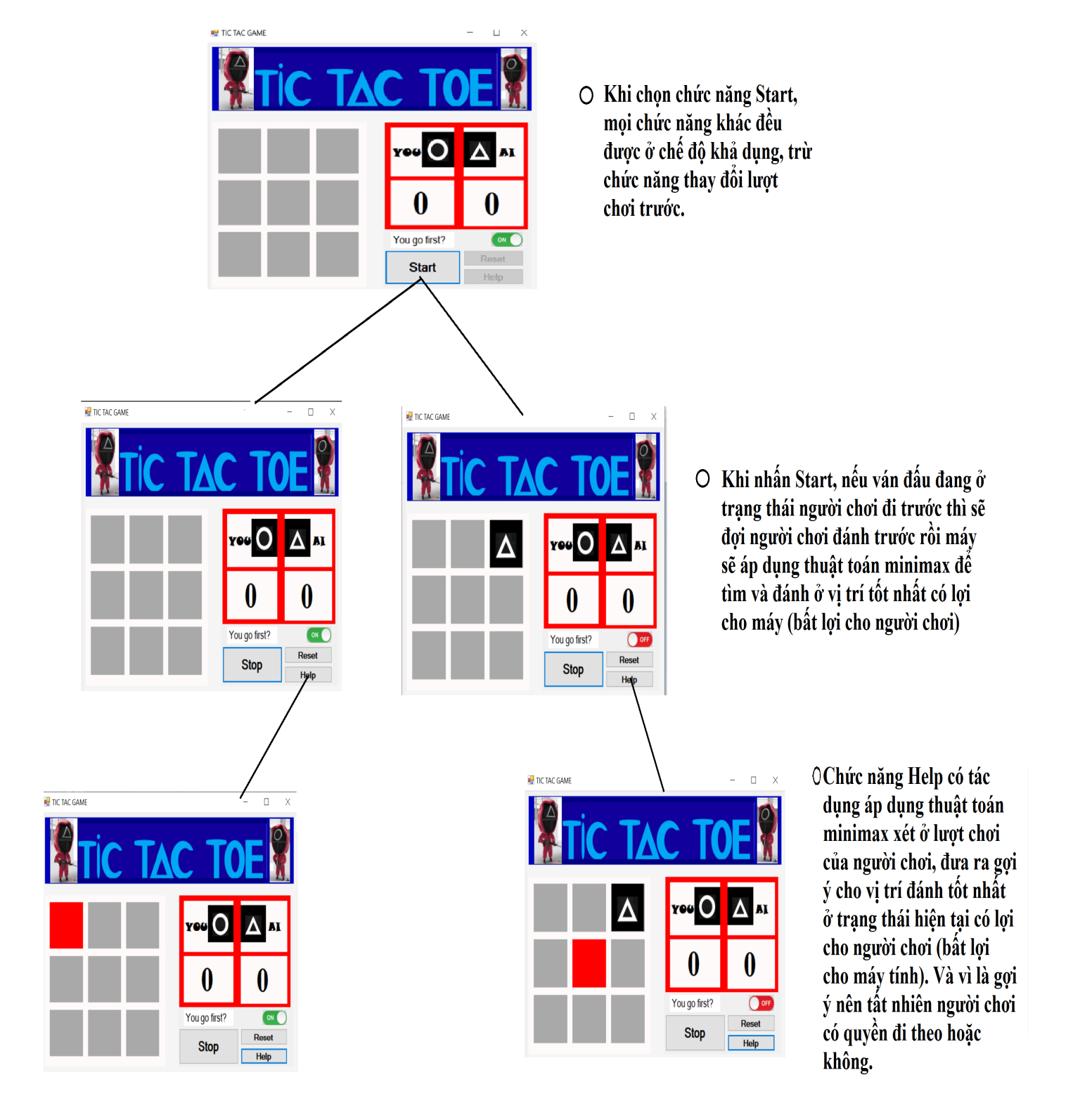
Sau khi hoàn thành việc khởi tạo các cấu trúc hoàn chỉnh của một trạng thái, các đoạn code sau có nhiệm vụ vén cạn không gian trạng thái và áp dụng thuật toán minimax để đưa ra nước đi vừa có lợi cho mình nhất, vừa bất lợi cho đối phương nhất.



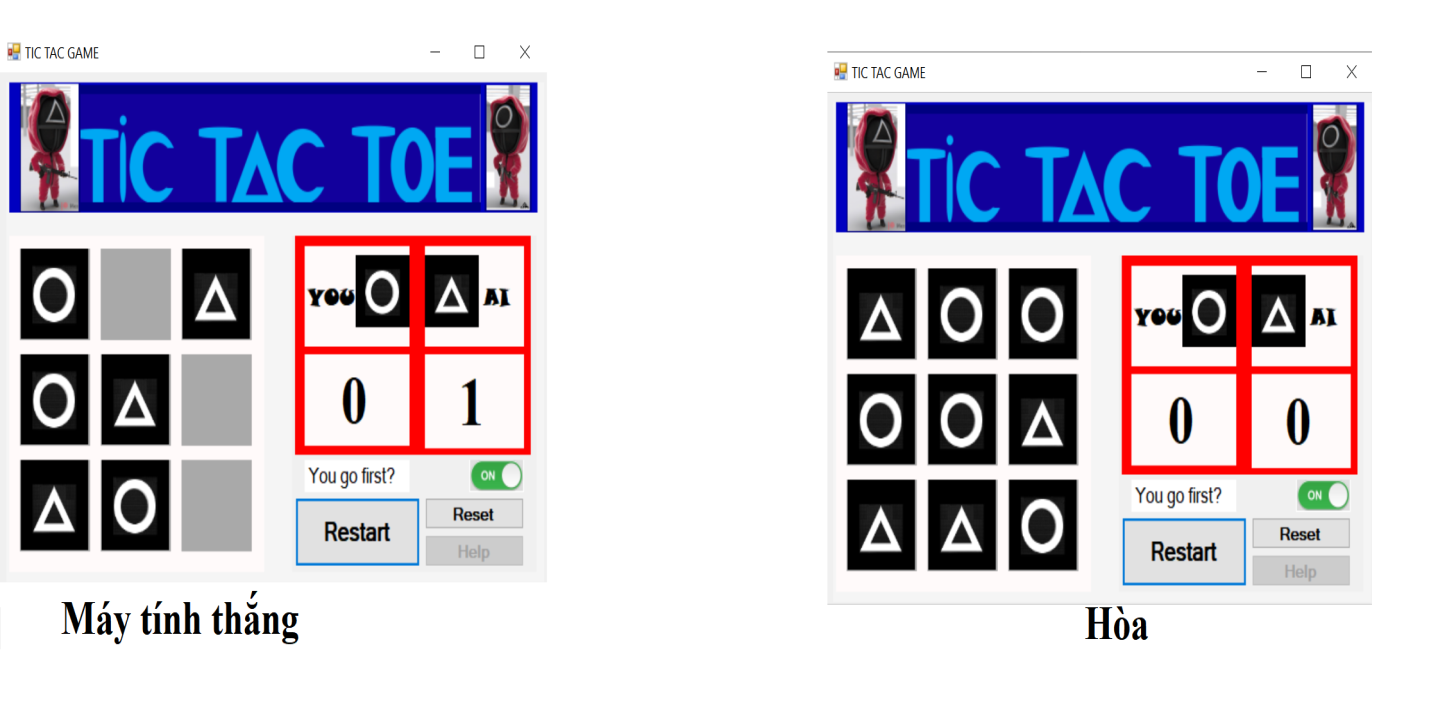


****

* 1. **Giới thiệu phần mềm**
     1. **Giao diện và một số chức năng chính của phần mềm**



* + 1. **Các trạng thái kết thúc**



Một chương trình trò chơi Tic Tac Toe “bất khả chiến bại” được định nghĩa rằng: Nếu mình chơi một cách hoàn hảo, thì trong tất cả các ván Tic Tac Toe mình sẽ Thắng hoặc Hòa, và nếu mình chơi với một người chơi hoàn hảo khác, thì cả hai sẽ luôn luôn Hòa. Vậy nên kết quả của chương trình sẽ luôn trả về một trong hai kết quả: hoặc là Máy THẮNG hoặc là cả hai cùng HÒA.

# Chương 4: Tài liệu tham khảo

<https://epixtemo.wordpress.com/2018/06/26/tic-tac-toe>

<https://vi.wikipedia.org/wiki/Minimax>