TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

KHOA CÔNG NGHỆ THÔNG TIN

BÀI TẬP LỚN

ỨNG DỤNG GIẢI THUẬT MINIMAX VÀO THIẾT KẾ GAME TICTACTOE

Giáo viên hướng dẫn: Thầy **LÊ ĐÌNH THẬN**

Sinh viên thực hiện: **Hà Ngọc Mỹ - MSSV: 51800081**

Nguyễn Tân Hoàng Phúc - MSSV: 51800801

Huỳnh Hữu Thiên - MSSV: 51800811

Lớp: 18050201

Nhóm: 16 - *Tổ*: 1

Khóa: 22

THÀNH PHỐ HỒ CHÍ MINH, NĂM 2019

BÁO CÁO ĐƯỢC HOÀN THÀNH

TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG

Tôi xin cam đoan đây là báo cáo của riêng chúng tôi và được hướng dẫn của thầy Lê Đình Thận. Các nội dung nghiên cứu và kết quả trong đề tài này là trung thực. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra trong báo cáo còn sử dụng một số nhận xét, đánh giá, số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

TP. Hồ Chí Minh, ngày 21 tháng 04 năm 2019

Tác giả

(ký tên và ghi rõ họ tên)

Hà Ngọc Mỹ

Huỳnh Hữu Thiên

Nguyễn Tân Hoàng Phúc

PHẦN XÁC NHẬN VÀ ĐÁNH GIÁ CỦA GIẢNG VIÊN

Phần xác nhận của GV hướng dẫn

_____
_____
_____
_____
_____
_____
_____
_____

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

Phần đánh giá của GV chấm bài

_____
_____
_____
_____
_____
_____
_____
_____

Tp. Hồ Chí Minh, ngày tháng năm

(kí và ghi họ tên)

TÓM TẮT

Bài báo cáo gồm có hai chương. Chương I, giới thiệu tổng quan về giải thuật Minimax, bao quát chung về các phương thức, sơ lược về các dạng. Chương 2, tìm hiểu, phân tích cụ thể về giải thuật Minimax và cách ứng dụng giải thuật Minimax vào game Tic Tac Toe.

MỤC LỤC

PHẦN 1: TỔNG QUAN VỀ GIẢI THUẬT MINIMAX

I) Minimax là gì?

Minimax là giải thuật là một thuật toán đệ quy lựa chọn bước đi kế tiếp trong một trò chơi có hai người bằng cách định giá trị cho các Node trên cây trò chơi sau đó tìm Node có giá trị phù hợp để đi bước tiếp theo.

II) Tại sao phải cần dùng minimax?

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.

III) Các khái niệm

1.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.

2. 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)

PHẦN 2 - GIẢI THUẬT MINIMAX

I) Giải thuật Minimax

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.

II) Các bước thuật giải Minimax

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).

Tính giá trị 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 (nút MIN), 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 (nút MAX), áp dụng thủ tục Minimax này cho các con của nó. Ghi nhớ kết quả lớn nhất.

III) Ví dụ mô phỏng giải thuật Minimax cho trò chơi Tic-Tac-toe

- 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.

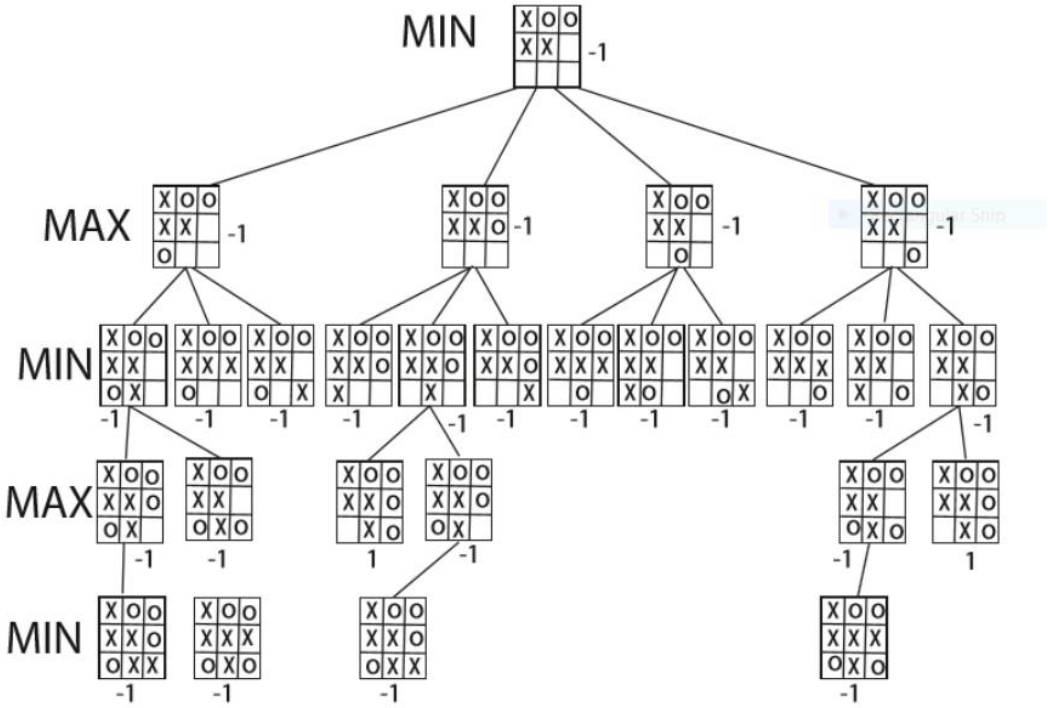
Sau đó tính ngược lại cây trò chơi theo quy tắc:

- 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.



Giải thuật Minimax Hai người chơi trong game được đại diện là MAX và MIN. MAX đại diện cho người chơi luôn muốn chiến thắng và cố gắng tối ưu hóa ưu thế của mình còn MIN đại diện cho người chơi cố gắng cho người MAX giành số điểm càng thấp càng tốt. Giải thuật Minimax thể hiện bằng cách định trị các Node trên cây trò chơi: Node thuộc lớp MAX thì gán cho nó giá trị lớn nhất của con Node đó. Node thuộc lớp MIN thì gán cho nó giá trị nhỏ nhất của con Node đó. Từ các giá trị này người chơi sẽ lựa chọn cho mình nước đi tiếp theo hợp lý nhất.

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

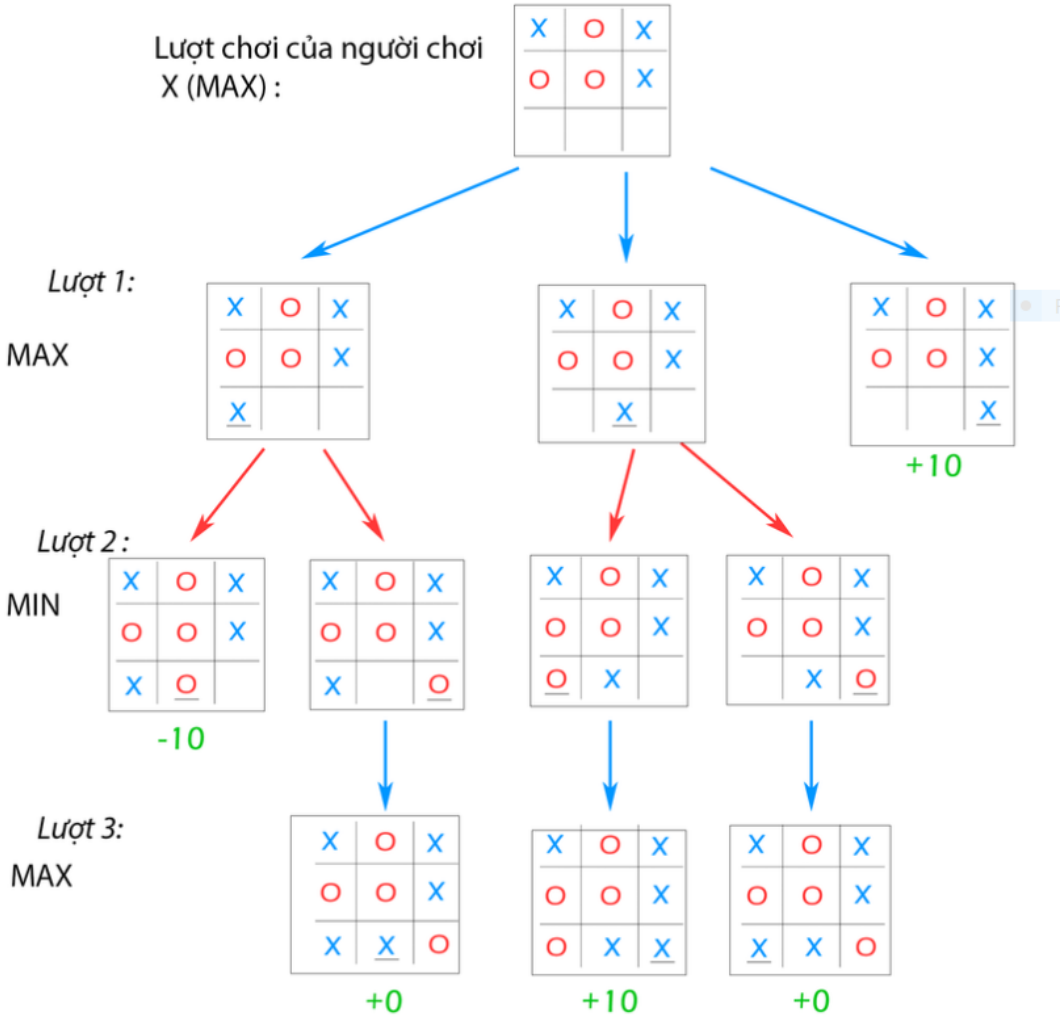
node lá (X win game) và có giá trị là +10. Còn 2 node 1,2 thì chưa biết giá trị của nó tại lượt 1 nên chúng ta dựa vào giá trị của các node con để định giá trị và bằng giá trị bé nhất của các node con ở lớp MIN tại lượt 2. Cứ tiếp tục tương tự như vậy đến lúc gặp node lá thì từ node lá đó ta suy ngược 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. 5. Áp dụng vào code

- Đầu tiên chúng ta cần 1 hàm để biết trạng thái game hiện tại đã thắng, thua hay hòa. `CheckStateGame(Move)`
- Tiếp theo là cần tìm nước tốt nhất cần đi.
- Và tiếp đến là hàm tính giá trị minimax của các nước đó.

Vậy là chúng ta implement được thuật toán minimax. 6. Thuật toán Minimax với độ sâu

Như ở hình này ta cần tìm giá trị lớn nhất của các node con. Mà ta tính được giá trị node 1,2,3 tương ứng là -10, +10, +10. Vậy giá trị ở node 2,3 là bằng nhau = +10. Nên ta đang phân vân giữa 2 node 2,3. Từ đó ta nâng cấp thuật toán minimax với độ sâu depth:

Áp dụng nâng cấp trên thì ta sẽ có giá trị mới của node 1,2,3 tương ứng là -9,+8,+10 => Max = +10 giá trị của node 3. Vậy node 3 là node cần tìm. 7. Tối ưu thuật toán minimax

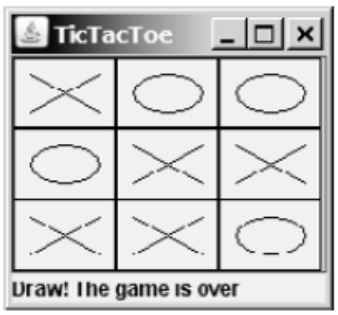

Lượt chơi của người chơi
X (MAX) :
Lượt 1:
MAX
+10
Lượt 2 :
MIN
-10
Lượt 3:
MAX
+0
+10
+0

PHẦN 3: GAME TIC TAC TOE

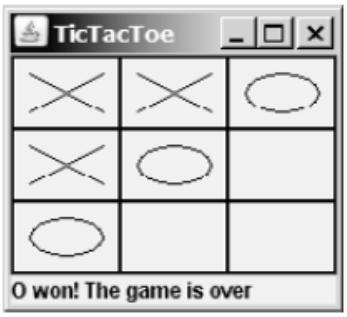
I. Giới thiệu

Tic Tac Toe là một trò chơi khá phổ biến dùng viết trên bàn cờ giấy có 9 ô. Hai người chơi, một người dùng ký hiệu **O**, người kia dùng ký hiệu **X**, lần lượt điền ký hiệu của mình vào các ô. Người thắng cuộc là người đầu tiên tạo được một dãy 3 ký hiệu của mình theo các chiều ngang, dọc hay chéo đều được. Nếu sau khi đã lấp đầy các ô trống mà vẫn không có ai đạt được một dãy 3 ô thẳng hàng thì sẽ là hòa. Hình sau mô tả 3 trường hợp ví dụ:

(a) Người chơi X thắng



(b) Cờ hòa



(c) Người chơi **O** thắng

II. Phân tích và thiết kế

Những ví dụ mà bạn vừa nhìn thấy ở trên chỉ thể hiện những hành vi đơn giản và có thể dễ dàng mô hình hóa bằng các lớp, nhưng ngoài ra thì hành vi của trò chơi Tic Tac Toe còn có một số điểm khác phức tạp hơn thế. Để có thể tạo ra được các lớp mô phỏng các hành vi này, chúng ta sẽ cần phải nghiên cứu và hiểu kỹ hơn về trò chơi này.

Giả sử rằng ban đầu thì tất cả các ô (cell) đều trống, và người chơi thứ nhất dùng ký hiệu **X**, người chơi thứ hai dùng ký hiệu **O**. Để thực hiện một bước đi thì người chơi sẽ trỏ chuột đến một ô và nhấn chuột vào đó, nếu ô đó còn trống thì ký hiệu **O** (hoặc **X**) sẽ được điền vào đó, nếu ô đó không còn trống thì nước đi này sẽ không được ghi nhận.

Với những mô tả như trên, ta thấy rằng mỗi ô cờ là một đối tượng GUI (Graphic User Interface – Giao diện Đồ họa Người dùng) mà có thể xử lý sự kiện nhấn chuột và hiển thị các ký hiệu. Đối tượng này có thể là một nút (button) hoặc là một bảng (panel). Việc vẽ trên một panel sẽ là linh hoạt hơn nhiều so với việc vẽ trên một button, bởi vì trên một panel thì ta có thể vẽ các ký hiện **X** và **O** với bất kỳ kích thước nào mà ta mong muốn, còn trên một button thì ta chỉ có thể hiển thị các ký hiệu đó như là một nhãn ký tự (text label). Vì lí do đó chúng ta sẽ sử dụng một panel để mô phỏng một ô của bàn cờ. Làm thế nào chúng ta có thể biết được trạng thái của mỗi ô (trống, **X**, hay **O**)? Trong lớp **Cell** chúng ta sẽ sử dụng một thuộc tính có tên *token* với kiểu dữ liệu là *char*. Lớp **Cell** có nhiệm vụ vẽ ký hiệu khi một ô trống được nhấn chuột.

Bàn cờ của trò chơi Tic Tac Toe bao gồm 9 ô, được tạo ra bằng câu lệnh **new Cell[3][3]**. Để xác định được lượt chơi của người chơi thì chúng ta sẽ sử dụng một biến có tên là *whoseTurn* với kiểu dữ liệu là *char*. Ban đầu thì *whoseTurn* sẽ có giá trị là '**X'**, sau đó chuyển thành '**O'** và cứ thay đổi lần lượt giữa hai giá trị đó mỗi khi các ô mới được đánh. Khi trò chơi kết thúc thì giá trị của *whoseTurn* sẽ chuyển thành ' '.
Làm sao để biết được trò chơi đã kết thúc hay chưa? Có ai thắng cuộc hay không? Và ai là người thắng, nếu có? Chúng ta có thể tạo một phương thức có tên là *isWon(char token)* để kiểm tra xem một người chơi với ký hiệu *token* đã thắng cuộc hay chưa, và một phương thức khác là *isFull()* để kiểm tra xem liệu tất cả các ô đều đã được đánh hay chưa.

Với những phân tích như vậy, chúng ta thấy lộ rõ ra 2 lớp. Lớp đầu tiên là **Cell**, với chức năng là điều khiển các thao tác cho một ô. Lớp thứ hai là **TicTacToe** có chức năng là thực hiện toàn bộ trò chơi và xử lý tất cả các ô.

III. Ý tưởng

+Đầu tiên ta tạo một mảng a chứa 9 phần tử tương ứng với 9 ô.
+Khi người chơi đánh X máy sẽ đánh O ở vị trí random.
+Giá trị mặc định của mỗi ô là -1.
+Khi người chơi đánh X thì giá trị đổi thành 1.
+Máy đánh O thì giá trị đổi thành 2
+Do có 9 ô nên sẽ có 8 khả năng X thắng (3 hàng dọc, 3 hàng ngang, 2 hàng chéo) và tương tự 8 trường hợp cho O =>16 trường hợp kiểm tra thắng thua . Nếu không rơi vào 16 trường hợp trên thì hòa cờ.

Mảng a chứa 9 phần tử mang giá trị -1
- Biến countNumber là để đếm số ô đã được đánh (dùng nó để dừng việc đánh khi đủ 9 ô)
- Biến check dùng để kiểm tra nếu X hoặc O thắng thì dừng không cho máy đánh nữa

- Action của button 1 : kiểm tra nếu ô đó là rổng (có giá trị -1) thì mới cho người dùng đánh
- Người chơi đánh X thì ta đổi giá trị -1 thành 1 (remove -1 và insert 1) và gán hình ảnh X vào
- Phát sinh random O (gọi hàm random)
- Gọi hàm kiểm tra thắng thua
- Làm tương tự cho 8 button còn lại

Điều kiện trong hàm while
+Phát sinh số ngẫu nhiên cho tới khi không trùng với ô đã đánh (flag=FALSE)
+Kiểm tra còn ô trống thì mới cho đánh(countNumber<9)
+Nếu X hoặc O thắng thì không đánh nữa(check == TRUE)
- Còn trong switch case thì tương tự như khi đánh X, khác là O thì thay -1 bằng 2