TRƯỜNG ĐẠI HỌC ĐIỆN LỰC
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO CHUYÊN ĐỀ HỌC PHẦN
NHẬP MÔN TRÍ TUỆ NHÂN TẠO

ĐỀ TÀI:PHƯƠNG PHÁP TÌM KIẾM LỜI GIẢI THỎA MÃN CÁC RÀNG BUỘC VÀ BÀI TOÁN SUDOKU

Sinh viên thực hiện	**: NGUYỄN ĐỨC HOÀNG**
	: NGUYỄN HUY HOÀNG
Giảng viên hướng dẫn	**: PHẠM ĐỨC HỒNG**
Ngành	**: CÔNG NGHỆ THÔNG TIN**
Chuyên ngành	**: CÔNG NGHỆ PHẦN MỀM**
Lớp	**: D14CNPM5**
Khóa	**: 2019-2024**

Hà Nội, tháng 12 năm 2021

PHIẾU CHẤM ĐIỂM

Sinh viên thực hiện:

STT	Họ và tên	Chữ ký	Nhiệm vụ
1	Nguyễn Đức Hoàng Mã SV: 19810310299	(đã kí)	+ Sử dụng code C# và thuật toán quay lui (backtracking) để giải bài toán sudoku. + Viết Báo Cáo
2	Nguyễn Huy Hoàng Mã SV: 19810310305	(đã kí)	+ Code website(HTML,CSS, JS) giải thuật quay lui (backtracking) để giải bài toán + Viết Báo Cáo

Giảng viên chấm:

Họ và tên	Chữ ký	Ghi chú
Giảng viên chấm 1:		
Giảng viên chấm 2:		

MỤC LỤC

DANH MỤC HÌNH ẢNH

LỜI MỞ ĐẦU

Trò chơi Sudoku là một trò chơi chắc hẳn không còn quá xa lạ với mọi người, đặc biệt là thế hệ 9x. Nó nổi tiếng đến mức nhiều hãng điện thoại được cài đặt sẵn Sudoku trên máy. Để nói đến Sudoku, Sudoku là một trò chơi giải đố có bắt nguồn từ Nhật Bản có cách chơi rất dễ hiểu và đôi khi là rất dễ với chúng ta. Chúng ta cần mất tầm 10 phút để giải một được một câu đố Sudoku với mức trung bình, nhưng với máy tính, nó có thể giải trong chưa đến một giây. Trò chơi Sudoku là một ví dụ điển hình của thuật toán Quay Lui vét cạn, nó cho ta thấy rõ được Thuật toán Quay Lui trong bài toán tìm kiếm nhiều nghiệm.

Chính vì vậy chúng em đã đi đến việc lựa chọn đề tài "**Phương pháp tìm kiếm lời giải thỏa mãn các ràng buộc và Bài toán sudoku**" cho bài tập lớn Nhập Môn Trí Tuệ Nhân Tạo. Chúng em vô cùng biết ơn thầy Phạm Đức Hồng, người trực tiếp giảng dạy, hướng dẫn nhiệt tình cho chúng em trong quá trình nghiên cứu và thực hiện đề tài.

đề tài vẫn còn nhiều thiếu xót thiếu sót, vì vậy chúng em mong muốn nhận được các ý kiến đóng góp của các thầy cô để có thể hoàn thiện hơn nữa.

Chúng em xin chân thành cảm ơn!

CHƯƠNG 1: GIỚI THIỆU VỀ ARTIFICIAL INTELLIGENCE(AI) VÀ THUẬT TOÁN QUAY LUI VÉT CẠN

1.1. GIỚI THIỆU VỀ TRÍ TUỆ NHÂN TẠO(AI)

1.1.1. Trí tuệ nhân tạo là gì ?

Trí tuệ nhân tạo hay trí thông minh nhân tạo (Artificial intelligence – viết tắt là AI) là một ngành thuộc lĩnh vực khoa học máy tính (Computer science). Là trí tuệ do con người lập trình tạo nên với mục tiêu giúp máy tính có thể tự động hóa các hành vi thông minh như con người.

Trí tuệ nhân tạo khác với việc lập trình logic trong các ngôn ngữ lập trình là ở việc ứng dụng các hệ thống học máy (machine learning) để mô phỏng trí tuệ của con người trong các xử lý mà con người làm tốt hơn máy tính.

Cụ thể, trí tuệ nhân tạo giúp máy tính có được những trí tuệ của con người như: biết suy nghĩ và lập luận để giải quyết vấn đề, biết giao tiếp do hiểu ngôn ngữ, tiếng nói, biết học và tự thích nghi,…

Tuy rằng trí thông minh nhân tạo có nghĩa rộng như là trí thông minh trong các tác phẩm khoa học viễn tưởng, nó là một trong những ngành trọng yếu của tin học. Trí thông minh nhân tạo liên quan đến cách cư xử, sự học hỏi và khả năng thích ứng thông minh của máy móc.

1.1.2. Lịch sử

Tư tưởng có khả năng sinh vật nhân tạo xuất hiện như các thiết bị kể chuyện thời cổ đại, và đã được phổ biến trong tiểu thuyết, như
trong *Frankenstein* của Những nhân vật này và số phận của họ nêu ra nhiều vấn đề tương tự hiện đang được thảo luận trong đạo đức của trí tuệ nhân tạo.

Lĩnh vực nghiên cứu AI được ra đời tại một hội thảo tại Đại học Dartmouth năm 1956. Những người tham dự Allen Newell , Herbert Simon , John McCarthy, Marvin Minsky và Arthur Samuel đã trở thành những người sáng lập và lãnh đạo nghiên cứu AI.Họ và các sinh viên của họ đã tạo ra các chương trình mà báo chí mô tả là "đáng kinh ngạc":máy tính đang học chiến lược kiểm tra (c. 1954)(và đến năm 1959 được cho là chơi tốt hơn người bình thường), giải từ các vấn đề về đại số, chứng minh các định lý logic (Lý thuyết logic, lần chạy đầu tiên vào năm 1956) và nói tiếng Anh. Đến giữa thập niên 1960, nghiên cứu ở Mỹ được Bộ Quốc phòng tài trợ rất nhiều và các phòng thí nghiệm đã được thành lập

trên khắp thế giới. Những người sáng lập AI rất lạc quan về tương lai: Herbert Simon dự đoán, "máy móc sẽ có khả năng, trong vòng hai mươi năm nữa, làm bất kỳ công việc nào mà một người có thể làm.

Họ đã không nhận ra độ khó của một số nhiệm vụ còn lại. Tiến độ chậm lại và vào năm 1974, áp lực liên tục từ Quốc hội Hoa Kỳ để tài trợ cho các dự án năng suất cao hơn, cả chính phủ Hoa Kỳ và Anh đều dừng nghiên cứu khám phá về AI. Vài năm sau đó sẽ được gọi là " mùa đông AI ",giai đoạn mà việc kiếm được tài trợ cho các dự án AI là khó khăn.

Đầu những năm 1980, nghiên cứu AI đã được hồi sinh nhờ thành công thương mại của các hệ chuyên gia - một dạng chương trình AI mô phỏng kiến thức và kỹ năng phân tích của các chuyên gia về con người. Đến năm 1985, thị trường cho AI đã đạt hơn một tỷ đô la. Tuy nhiên, bắt đầu với sự sụp đổ của thị trường Máy Lisp vào năm 1987, AI một lần nữa rơi vào tình trạng khó khăn, và một sự gián đoạn thứ hai, kéo dài hơn đã bắt đầu.

Vào cuối những năm 1990 và đầu thế kỷ 21, AI bắt đầu được sử dụng cho hậu cần, khai thác dữ liệu, chẩn đoán y tế và các lĩnh vực khác. Thành công là nhờ sức mạnh tính toán học) và cam kết của các nhà nghiên cứu về phương pháp toán học và tiêu chuẩn ngày càng tăng (xem định luật Moore), nhấn mạnh hơn vào việc giải quyết các vấn đề cụ thể, mối quan hệ mới giữa AI và các lĩnh vực khác (như thống kê, kinh tế và toán khoa học.nổi tiếng nhất phải kể đến chương trình, Deep Blue đã trở thành hệ thống chơi cờ trên máy tính đầu tiên đánh bại một nhà đương kim vô địch cờ vua thế giới, Garry Kasparov, vào ngày 11 tháng 5 năm 1997.

Lĩnh vực của AI

- Lập luận, suy diễn tự động: Khái niệm lập luận (reasoning), và suy diễn (reference) được sử dụng rất phổ biến trong lĩnh vực AI. Lập luận là suy diễn logic, dùng để chỉ một tiến trình rút ra kết luận (tri thức mới) từ những giả thiết đã cho (được biểu diễn dưới dạng cơ sở tri thức). Như vậy, để thực hiện lập luận người ta cần có các phương pháp lưu trữ cơ sở tri thức và các thủ tục lập luận trên cơ sở tri thức đó.
- Biểu diễn tri thức: Muốn máy tính có thể lưu trữ và xử lý tri thức thì cần có các phương pháp biểu diễn tri thức. Các phương pháp biểu diễn tri thức ở

đây bao gồm các ngôn ngữ biểu diễn và các kỹ thuật xử lý tri thức. Một ngôn ngữ biểu diễn tri thức được đánh giá là "tốt" nếu nó có tính biểu đạt cao và các tính hiệu quả của thuật toán lập luận trên ngôn ngữ đó. Tính biểu đạt của ngôn ngữ thể hiện khả năng biểu diễn một phạm vi rộng lớn các thông tin trong một miền ứng dụng. Tính hiệu quả của các thuật toán lập luận thể hiện chi phí về thời gian và không gian dành cho việc lập luận. Tuy nhiên, hai yếu tố này dường như đối nghịch nhau, tức là nếu ngôn ngữ có tínhbiểu đạt cao thì thuật toán lập luận trên đó sẽ có độ phức tạp lớn (tính hiệu quả thấp)và ngược lại (ngôn ngữ đơn giản, có tính biểu đạt thấp thì thuật toán lập luận trên đó sẽ có hiệu quả cao). Do đó, một thách thức lớn trong lĩnh vực AI là xây dựng các ngôn ngữ biểu diễn tri thức mà có thể cân bằng hai yếu tố này, tức là ngôn ngữ có tínhbiểu đạt đủ tốt (tùy theo từng ứng dụng) và có thể lập luận hiệu quả.

- Lập kế hoạch: khả năng suy ra các mục đích cần đạt được đối với các nhiệm vụ đưa ra, và xác định dãy các hành động cần thực hiện để đạt được mục đích đó.

- Học máy: là một lĩnh vực nghiên cứu của AI đang được phát triển mạnh mẽ và có nhiều ứng dụng trong các lĩnh vực khác nhau như khai phá dữ liệu, khám phá tri thức,…

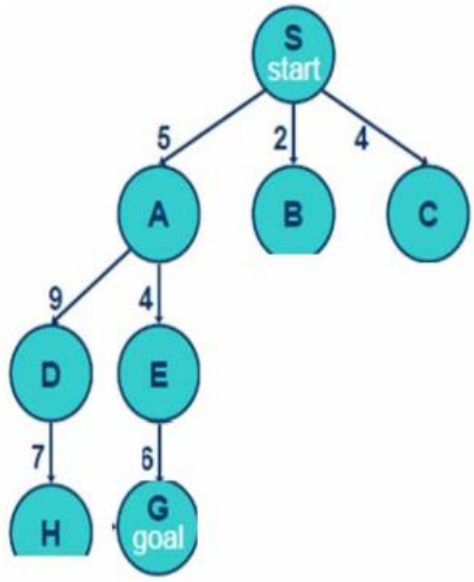
- Xử lý ngôn ngữ tự nhiên: là một nhánh của AI, tập trung vào các ứng dụng trên ngôn ngữ của con người. Các ứng dụng trong nhận dạng tiếng nói, nhận dạng chữ viết, dịch tự động, tìm kiếm thông tin,…

- Hệ chuyên gia: cung cấp các hệ thống có khả năng suy luận để đưa ra những kết luận. Các hệ chuyên gia có khả năng xử lý lượng thông tin lớn và cung cấp các kết luận dựa trên những thông tin đó. Có rất nhiều hệ chuyên gia nổi tiếng như các hệ chuyên gia y học MYCIN, đoán nhận cấu trúc phân tử từ công thức hóa học DENDRAL, …

1.2. GIỚI THIỆU VỀ KỸ THUẬT QUAY LUI

node	Stack	father
	S	
S	A, B, C	Father[A,B,C]=S
A	D, E, B, C	Father[D,E]=A
D	H, E, B, C	Father[H]=D
H	E, B, C	
E	G, B, C	Father[G]=E
G		

Giá trị các biến trong giải thuật theo chiều sâu



Cây tìm kiếm của giải thuật theo chiều

***Hình 1**: Thuật toán tìm kiếm theo chiều sâu*

Thuật toán quay lui vét cạn (Backtracking) là một kĩ thuật thiết kế giải thuật dựa trên đệ quy. Ý tưởng của quay lui là tìm lời giải từng bước, mỗi bước chọn một trong số các lựa chọn khả dĩ và đệ quy. Người đầu tiên đề ra thuật ngữ này (backtrack) là nhà toán học người Mỹ D. H. Lehmer vào những năm 1950.

1.2.1. Tư tưởng

Nét đặc trưng của phương pháp quay lui là các bước hướng tới lời giải cuối cùng của bài toán đều được làm thử. Tại mỗi bước, nếu có một lựa chọn được chấp nhận thì ghi lại lựa chọn này và tiến hành các bước thử tiếp theo. Còn ngược lại không có lựa chọn nào thích hợp thì làm lại bước trước, xóa bỏ sự ghi nhận và quay về chu trình thử các lựa chọn còn lại.

Hành động trên được gọi là quay lui, thuật toán thể hiện phương pháp này gọi là quay lui.

Điểm quan trọng của thuật toán là phải ghi nhớ tại mỗi bước đi qua để tránh trùng lặp khi quay lui. Các thông tin này được lưu trữ trong một ngăn xếp, nên thuật toán thể hiện ý thiết kế một cách đệ quy.

1.2.2. Heuristic

Người ta thường sử dụng một số phương pháp heuristic để tăng tốc cho quá trình quay lui. Do các biến có thể được xử lý theo thứ tự bất kỳ, việc thử các biến bị ràng buộc chặt nhất (nghĩa là các biến có ít lựa chọn về giá trị nhất) thường có hiệu quả do nó tỉa cây tìm kiếm từ sớm (cực đại hóa ảnh hưởng của lựa chọn sớm hiện hành).

Các cài đặt quay lui phức tạp thường sử dụng một hàm biên, hàm này kiểm tra xem từ lời giải chưa đầy đủ hiện tại có thể thu được một lời giải hay không, nghĩa là nếu đi tiếp theo hướng hiện tại thì liệu có ích hay không. Nhờ đó, một kiểm tra biên phát hiện ra các lời giải dở dang chắc chắn thất bại có thể nâng cao hiệu quả của tìm kiếm. Do hàm này hay được chạy, có thể tại mỗi bước, nên chi phí tính toán các biên cần tối thiểu, nếu không, hiệu quả toàn cục của thuật toán sẽ không được cải tiến. Các hàm kiểm tra biên được tạo theo kiểu gần như các hàm heuristic khác: nới lỏng một số điều kiện của bài toán.

1.2.3. Phương pháp

Dùng để giải bài toán liệt kê các cấu hình. Mỗi cấu hình được xây dựng bằng từng phần tử. Mỗi phần tử lại được chọn bằng cách thử tất cả các khả năng.

Các bước trong việc liệt kê cấu hình dạng X[1...n]:

- Xét tất cả các giá trị X[1] có thể nhận, thử X[1] nhận các giá trị đó. Với mỗi giá trị của X[1] ta sẽ:
- Xét tất cả giá trị X[2] có thể nhận, lại thử X[2] cho các giá trị đó. Với mỗi giá trị X[2] lại xét khả năng giá trị của X[3]...tiếp tục như vậy cho tới bước:
- ...
- Xét tất cả giá trị X[n] có thể nhận, thử cho X[n] nhận lần lượt giá trị đó.
- Thông báo cấu hình tìm được.

Bản chất của quay lui là một quá trình tìm kiếm theo chiều sâu(Depth-First Search).

1.2.4. Mô hình bài toán

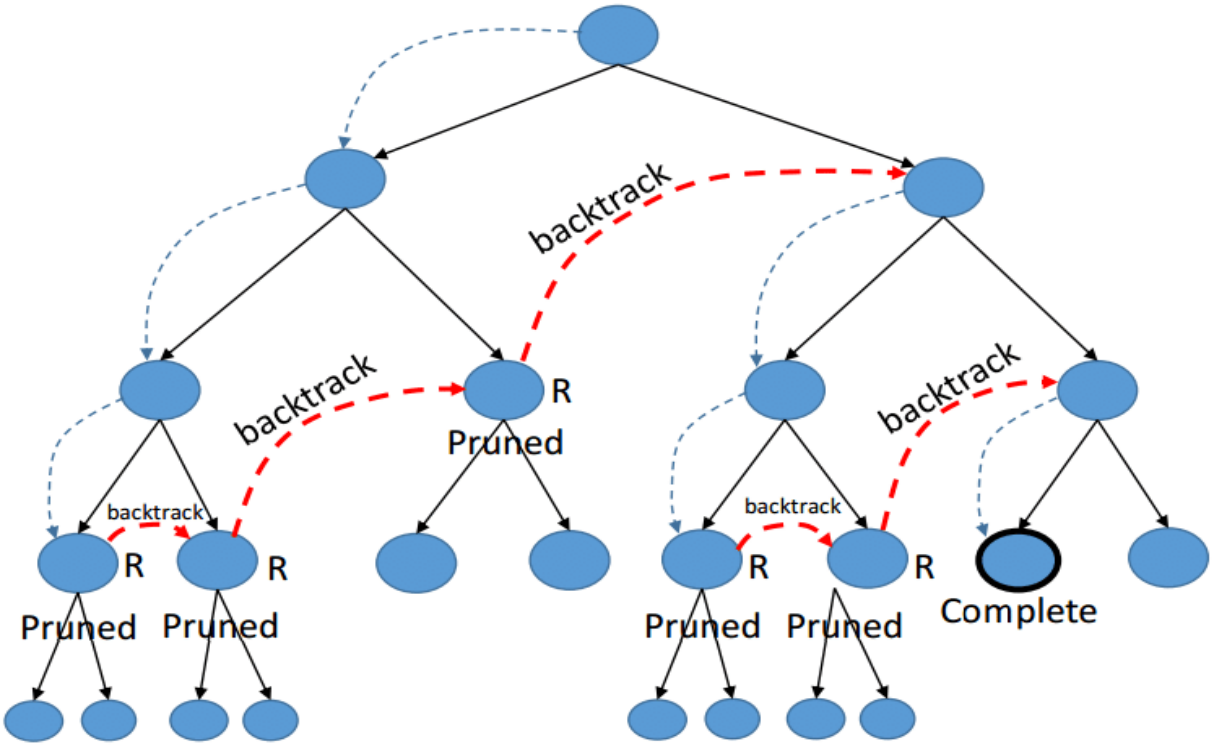
Thuật toán quay lui có thể được mô tả bằng đoạn mã giả (pseudocode) sau:

-Backtracking(k) {

for([Mỗi phương án chọn i(thuộc tập D)]) {

```
        if ([Chấp nhận i]) {
            [Chọn i cho X[k]];
            if ([Thành công]) {
                [Đưa ra kết quả];
            } else {
                Backtracking(k+1);
                [Bỏ chọn i cho X[k]];
            }
        }
    }
}
```

Ta có thể trình bày quá trình tìm kiếm lời giải quả thuật toán quay lui bằng cây sau:



***Hình 2**: Cây mô tả thuật toán quay lui*

Vấn đề khó nhất khi thiết kế thuật toán dạng quay lui đó là tìm ra tập các lựa chọn có thể trong mỗi bước. Tập lựa chọn này sẽ ảnh hưởng đến tính chính xác cũng như độ phức tạp (thời gian cũng như bộ nhớ) của thuật toán quay lui.

1.3. KẾT LUẬN

- Ưu điểm: Ưu điểm lớn nhất là sẽ luôn tìm ra lời giải cho một bài toán đúng.

- Nhược điểm:

+ Thời gian thực hiện một thuật toán phức tạp có thể kéo dài.

+ Vì sử dụng đệ quy nên không thể tránh khỏi việc tiêu tốn tài nguyên mỗi khi đánh giá hàng trăm nghiệm khả rĩ.

CHƯƠNG 2: TRÒ CHƠI SUDOKU

2.1 BÀI TOÁN

SUDOKU là một bài toán điển hình của thuật toán Quay Lui vét cạn. Giống như bài toán tìm nhiều nghiệm thỏa mãn các điều kiện. Chúng ta hãy cùng đi vào tìm hiểu.

2.1.1. Cách Chơi

Cách chơi khá đơn giản. Chúng ta chỉ cần điền những số còn trống sao cho luôn luôn tuân theo 3 nguyên tắc :

- Phải có đủ các chữ số từ 1 tới 9 ở hàng ngang mà không trùng số nhưng không cần đúng thứ tự.
- Phải có đủ các chữ số từ 1 tới 9 ở hàng dọc mà không trùng số nhưng không cần đúng thứ tự.
- Phải có đủ các chữ số từ 1 tới 9 ở từng vùng 3x3 mà không trùng số nào ở trong cùng một vùng 3x3.

Đây là trò chơi ngốn rất nhiều thời gian, tâm trí và đòi hỏi người chơi phải có sự kiên nhẫn, tính toán.

Thông thường chơi sẽ bắt đầu từ hàng đầu tiên từ trái qua phải và thử những giá trị trong miền giá trị có thể điền vào, thay và thử, nếu sai thay rồi thử. Quá trình đó được lặp đi lặp lại cho đến khi ô cuối cùng được điền vào mà không một ô nào vi phạm 3 nguyên tắc trên.

Theo thuật toán mà em cài đặt cũng vậy ! máy tính sẽ bắt đầu đi theo hàng dọc và tìm kiếm phần từ trống để bắt đầu tìm kiếm những giá trị phù hợp cho nó. Với hàng dọc và ô vuông 3x3 nhỏ cũng vậy!

2	8		3	6	1	4	9	
	6					3		
		1	7					
	2	7						
8				1				3
	1		6			2	7	5
	3		8	9		7	1	4
	7	8	4		2	6	3	
	4	9		7		8	5	

Hình 3: *Hình ảnh của một bài toán Sudoku*

2.1.2. Xuất xứ bài toán

Sudoku xuất hiện lần đầu tiên ở Mỹ với tên gọi "Đặt vị trí số". Theo Giám đốc Puzzler Media - một NXB tạp chí và sách câu đố tầm cỡ nhất của nước Anh Tim Preston cho rằng, đây thực chất là phát minh của Leonhard Euler - nhà toán học thế kỷ 18. Ông là người gốc Thuỵ Sĩ nhưng lại gắn với

hoàng gia Nga phần lớn cuộc đời của mình. Ở thời điểm đó, trò chơi có tên gọi Hình vuông Latin. Suốt nhiều năm, trò chơi này chỉ được biết đến bởi giới toán học. Cho đến thập kỷ 1970, khi nó được đưa vào tập sách đố của nhà xuất bản Dell ở Mỹ thì mọi người mới dần biết tới nó. Sau đó, trò chơi này được du nhập vào Nhật Bản và được nhà xuất bản Nikolo thay tên đổi họ thành Sudoku - có nghĩa là duy nhất. Cái tên này không chỉ bao hàm được cách chơi, mục đích chơi mà còn khẳng định vị trí duy nhất, độc đáo của trò chơi này. Ở Nhật, tất cả các câu đố Sudoku đều được viết tay. Với tên gọi Number Place khi còn ở Mỹ, Sudoku khi tới Nhật thì trò chơi này vẫn chưa tạo ra một sức hút đặc biệt đối với người chơi. Cho tới khi nó được Wayne Gould - vị thẩm phán về hưu đã từng làm việc ở Hong Kong trước đó mang về Anh quốc và đăng trên tờ Times thì nó thật sự trở thành một cơn sốt. Trong vài tháng ở xứ sở sương mù, trò chơi này đã khiến mọi người bị thu hút đặc biệt với rất nhiều người.

2.2 GIẢI QUYẾT BÀI TOÁN

2.2.1. Ý tưởng

Ý tưởng tưởng của bài toán khá đơn giản. Từ nỗi ô trống cần điền chúng ta sẽ tìm những giá trị khả dĩ để điền vào ô trống đó, sẽ lưu giá trị tại một biến nào đó, từ mỗi giá trị khả dĩ đó thì chúng ta sẽ đi điền vào những ô tiếp theo bằng cách sử dụng đệ quy. Nếu giá trị không đúng thì sẽ quay trở lại thay đổi giá trị khả dĩ rồi làm tương tự. Điều đó khiến cho thuật toán có tên là Quay lui.

2.2.2. Thiết kế thuật toán

Ý Tưởng bài toán được thiết kế như sau :

```
private void loadValues()
        {
            // Xóa giá trị từng ô
            foreach (var cell in cells)
            {
                cell.Value = 0;
                cell.Clear();
            }

            // Phương thức này sẽ được gọi đệ quy cho đến khi nó tìm thấy các giá trị phù
hợp cho từng ô
            findValueForNextCell(0, -1);
        }
        Random random = new Random();

    private bool findValueForNextCell(int i, int j)
```

```
        {
            // Tăng giá trị cho i và j để chuyển sang ô tiếp theo và nếu đến cột cuối
chuyển sang hàng tiếp theo

            if (++j > 8)
            {
                j = 0;

                // Thoát khi hết dòng
                if (++i > 8)
                    return true;
            }

            var value = 0;
            var numsLeft = new List<int> { 1, 2, 3, 4, 5, 6, 7, 8, 9 };

            //Tìm một số ngẫu nhiên và hợp lệ cho ô và chuyển đến ô tiếp theo và kiểm tra
xem nó có thể được phân bổ bằng một số ngẫu nhiên và hợp lệ khác không
            do
            {
                // Nếu không còn số nào trong danh sách để thử tiếp theo, hãy quay lại ô
trước đó và phân bổ nó bằng một số khác
                if (numsLeft.Count < 1)
                {
                    cells[i, j].Value = 0;
                    return false;
                }

                // Lấy một số ngẫu nhiên từ các số còn lại trong danh sách
                value = numsLeft[random.Next(0, numsLeft.Count)];
                cells[i, j].Value = value;

                // Xóa giá trị được phân bổ khỏi danh sách
                numsLeft.Remove(value);
            }
            while (!isValidNumber(value, i, j) || !findValueForNextCell(i, j));

            // Testing line
          //cells[i, j].Text = value.ToString();

            return true;
        }

        private bool isValidNumber(int value, int x, int y)
        {
            for (int i = 0; i < 9; i++)
            {

                // Kiểm tra tất cả các ô theo hướng dọc
                if (i != y && cells[x, i].Value == value)
                    return false;

                // Kiểm tra tất cả các ô theo hướng ngang
                if (i != x && cells[i, y].Value == value)
                    return false;
            }
```

```
// Kiểm tra tất cả các ô trong khối cụ thể
for (int i = x - (x % 3); i < x - (x % 3) + 3; i++)
{
    for (int j = y - (y % 3); j < y - (y % 3) + 3; j++)
    {
        if (i != x && j != y && cells[i, j].Value == value)
            return false;
    }
}

return true;
```

-Hàm findValueForNextCell : tìm giá trị tương ứng cho từng ô bằng cách Tăng giá trị cho i và j để chuyển sang ô tiếp theo và nếu đến cột cuối chuyển sang hàng tiếp theo

- biến numLeft: khai báo một danh sách từ 1 đến 9

- Vòng lặp do/while tìm một số ngẫu nhiên và hợp lệ cho ô và chuyển đến ô tiếp theo và kiểm tra xem nó có thể được phân bổ bằng một số ngẫu nhiên và hợp lệ khác không

- Hàm isValidNumber: là để kiểm tra các số thỏa mãn theo điều kiện

Thuật toán được mô tả như sau :

+ Đầu tiên chúng ta sẽ phải đi tìm ô Trống gần nhất trong mảng 2 chiều để điền.

+ Chúng ta sẽ thử các giá trị đó thỏa mãn 3 điều kiện của bài toán thì sẽ lưu vào ô trống hiện hành .

+ Tiếp đến sẽ gọi để quy để làm lại những bước tương tự để kiểm tra xem với giá trị hiện hành thì sudoku có khả dĩ hay không. Nếu không thì quay lui thay đổi giá trị hiện hành và tiếp tục đệ quy

CHƯƠNG 3: CÀI ĐẶT, ĐÁNH GIÁ THỬ NGIỆM

3.1 Cài đặt với C#

3.1.1 Code

Hàm 1 : Tìm kiếm ô trống tiếp theo

```
// Tăng giá trị cho i và j để chuyển sang ô tiếp theo và nếu đến cột cuối chuyển sang hàng tiếp theo

if (++j > 8)
{
    j = 0;
```

```
                // Thoát khi hết dòng
                if (++i > 8)
                    return true;
            }
```

Hàm 2 : Kiểm Tra Giá Trị Có Thỏa mãn 3 nguyên tắc hay không

```
private bool isValidNumber(int value, int x, int y)
        {
            for (int i = 0; i < 9; i++)
            {

                // Kiểm tra tất cả các ô theo hướng dọc
                if (i != y && cells[x, i].Value == value)
                    return false;

                // Kiểm tra tất cả các ô theo hướng ngang
                if (i != x && cells[i, y].Value == value)
                    return false;
            }


            // Kiểm tra tất cả các ô trong khối cụ thể
            for (int i = x - (x % 3); i < x - (x % 3) + 3; i++)
            {
                for (int j = y - (y % 3); j < y - (y % 3) + 3; j++)
                {
                    if (i != x && j != y && cells[i, j].Value == value)
                        return false;
                }
            }

            return true;
        }
```

Hàm 3 : Hàm Chính

```
private void showRandomValuesHints(int hintsCount)
        {
            // Show  random giá trị ô
            // Số ô hiển thị phụ thuộc vào cấp độ của phần chơi
            for (int i = 0; i < hintsCount; i++)
            {
                var rX = random.Next(9);
                var rY = random.Next(9);

                // Style của hint phải khác vs giá trị nhập
                //  khóa giá trị ô dc hiển thị người chơi ko dc sửa
                cells[rX, rY].Text = cells[rX, rY].Value.ToString();
                cells[rX, rY].ForeColor = Color.Black;
                cells[rX, rY].IsLocked = true;
            }
        }

  private void loadValues()
```

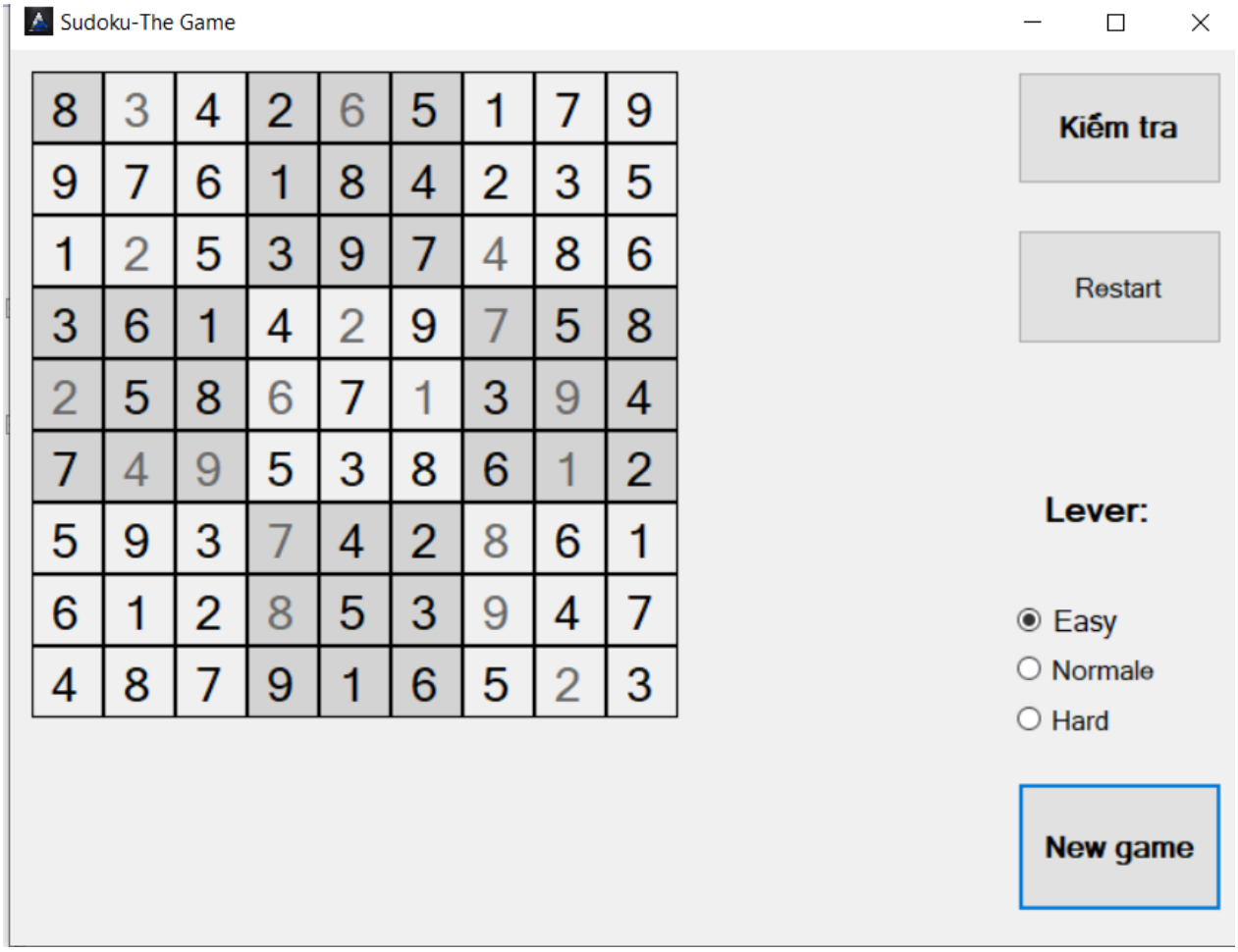
```
        {
            // Xóa giá trị từng ô
            foreach (var cell in cells)
            {
                cell.Value = 0;
                cell.Clear();
            }

            // Phương thức này sẽ được gọi đệ quy cho đến khi nó tìm thấy các giá trị phù
hợp cho từng ô
            findValueForNextCell(0, -1);
        }
        Random random = new Random();
```

ở đây chúng ta sẽ sử dụng một đề bài sudoku làm ví dụ chạy chương trình được gán vào một biến dạng ma trận 9x9. Ô trống sẽ đại điện cho giá trị ô trống chúng ta cần điền.

				7	2	8		
		5	9					6
	7				3		1	2
		7		5				
		4		8	9	6		
		2		4	7	5	3	
			6	9	4			
7		9	8					
6	8							

3.1.2. Kết quả đạt được



Hình 4: *Kết quả đạt được với C#*

3.1.3 Kết luận

- Ưu điểm: có giao diện đơn giản và đẹp, dễ thiết kế

- Nhược điểm : tương đối dễ lỗi trong quá trình cài đặt

3.2 Cặt đặt với javascipt

3.2.1 Cài đặt thuật toán

Hàm 1 : tìm kiếm ô trống tiếp theo

```
const findUnassignedPos = (grid, pos) => {
  for (let row = 0; row < CONSTANT.GRID_SIZE; row++) {
    for (let col = 0; col < CONSTANT.GRID_SIZE; col++) {
      if (grid[row][col] === CONSTANT.UNASSIGNED) {
        pos.row = row;
        pos.col = col;
        return true;
      }
    }
  }
  return false;
};
```

Hàm 2 : Kiểm tra giá trị có thỏa mãn 3 nguyên tắc của bài toán

```
/ kiểm tra 2 số trùng nhau trên cùng cột
const isColSafe = (grid, col, value) => {
  for (let row = 0; row < CONSTANT.GRID_SIZE; row++) {
    if (grid[row][col] === value) return false;
  }
  return true;
};

// kiểm tra 2 số trùng nhau trên cùng hàng
const isRowSafe = (grid, row, value) => {
  for (let col = 0; col < CONSTANT.GRID_SIZE; col++) {
    if (grid[row][col] === value) return false;
  }
  return true;
};

// kiểm tra 2 số trùng nhau trong khu vựng 3X3
const isBoxSafe = (grid, box_row, box_col, value) => {
  for (let row = 0; row < CONSTANT.BOX_SIZE; row++) {
    for (let col = 0; col < CONSTANT.BOX_SIZE; col++) {
      if (grid[row + box_row][col + box_col] === value) return false;
    }
  }
  return true;
};
```

```
// kiểm tra hàng và cột trong khuu 3x3
const isSafe = (grid, row, col, value) => {
  return (
    isColSafe(grid, col, value) &&
    isRowSafe(grid, row, value) &&
    isBoxSafe(grid, row - (row % 3), col - (col % 3), value) &&
    value !== CONSTANT.UNASSIGNED
  );
};
}
```

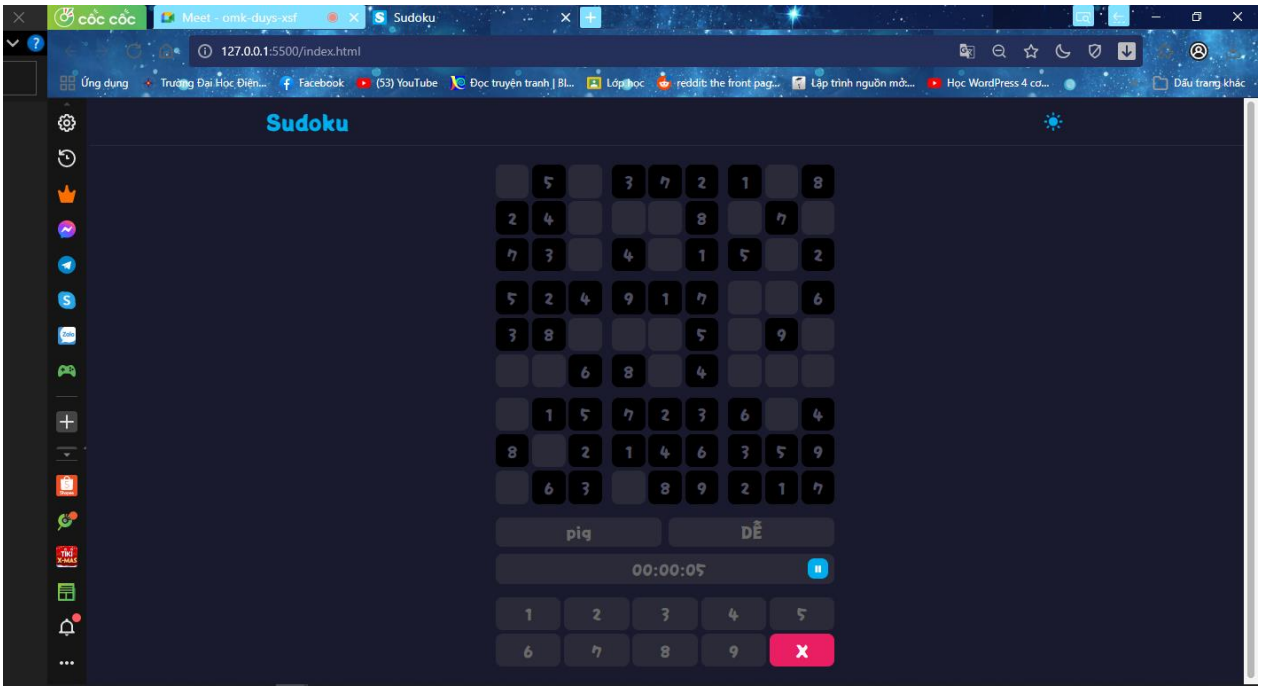
Hàm 3 : Hàm Chính

```
// tìm lỗi theo ô
const findUnassignedPos = (grid, pos) => {
  for (let row = 0; row < CONSTANT.GRID_SIZE; row++) {
    for (let col = 0; col < CONSTANT.GRID_SIZE; col++) {
      if (grid[row][col] === CONSTANT.UNASSIGNED) {
        pos.row = row;
        pos.col = col;
        return true;
      }
    }
  }
  return false;
};
    return false;
```

Thuật toán và logic code vẫn tương tự. Chỉ thêm một số các hàm để tùy chỉnh giao diện trên môi trường website

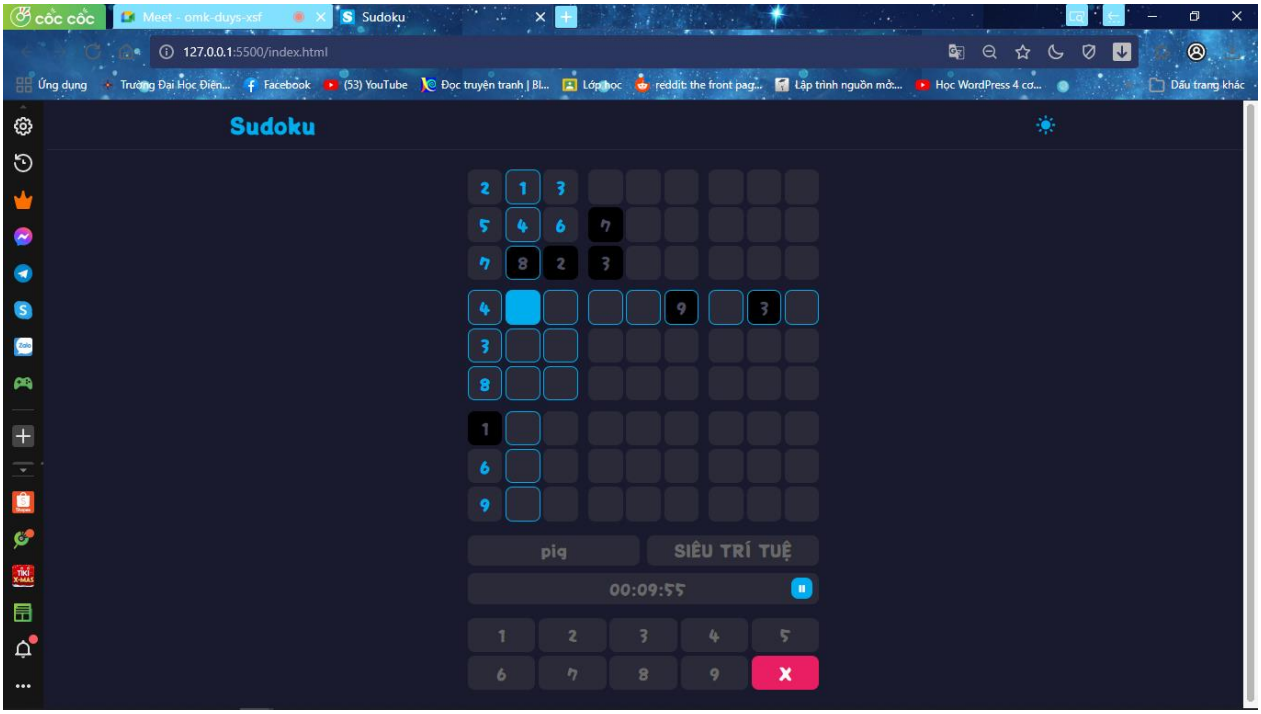
3.2.2 Giao diện chương trình

3.2.2.1 Giao diện khi chạy chương trình



***Hình 5**: Giao diện khi chạy chương trình*

3.2.2.4 Chọn mức độ chơi



***Hình 6**: Chế độ chơi dễ của trò chơi sudoku*

3.2.2.5 Bắt đầu Giải Sudoku

Hình 7: *Trò chơi đã được giải*

3.2.3 Kết luận

- Ưu Điểm: Có giao diện rất dễ sử dụng, gần gũi, đẹp và thân thiện với người dùng.

- Nhược Điểm: Code dài , dễ thiết kế code compiler nhưng khó để hiểu và khó hình dung.

KẾT LUẬN

Với kiến thức hiện có của mình, chúng em đã hoàn thành các yêu cầu ở trên khi tiến hành thực hiện đề tài "**Phương pháp tìm kiếm lời giải thỏa mãn các ràng buộc và Bài toán sudoku**". Tuy nhiên, trong quá trình làm sẽ không tránh khỏi những thiếu sót, hoặc cũng sẽ có những chỗ còn vướng mắc, chính vì vậy, chúng em mong được sự góp ý giúp đỡ của thầy giáo, để bài này được hoàn thiện hơn !

Chúng em xin gửi lời cảm ơn chân thành tới giảng viên Phạm Đức Hồng là giảng viên giảng dạy môn Nhập môn Trí tuệ nhân tạo của lớp D14CNPM7 đã tận tình hướng dẫn chúng em hoàn thành đề tài này!

Chúng em xin chân thành cảm ơn!

TÀI LIỆU THAM KHẢO

1. Phạm Đức Hồng, Giáo trình Nhập môn Trí tuệ Nhân tạo, Đại học Điện Lực
2. Youtube.com
3. Một số trang web giải thuật khác