

Recursion, Backtracking, Branch and Bound

Truong Ngoc Tuan

Nội dung

- ☐ Recursion
- ☐ Backtracking
- ☐ Generating Method
- ☐ Branch and Bound

Nội dung

□ Recursion

- Đệ quy là gì?
- Đệ quy có nhớ
- Bài toán ứng dụng

Nội dung

□ Backtracking

- Lược đồ chung
- Bài toán 8 hậu
- Bài toán mã đi tuần
- Liệt kê các hoán vị
- Liệt kê dãy nhị phân độ dài N
- Duyệt đồ thị

Nội dung

□ Generating Method

- Lược đồ chung
- Bài toán chuỗi 3 ký tự
- Liệt kê tập con của tập N phần tử
- Bài toán tập con K phần tử
- Hóa vị tập N phần tử

Nội dung

□ Branch and Bound

- Lược đồ chung
- Bài toán người đi du lịch
- Bài toán cái túi

Recursion

- ❑ Đệ quy: một hàm tự gọi lại chính nó
- ❑ Ví dụ:

```
int fact (int n) {  
    if(n <= 1) return 1;  
    return n * fact(n-1);  
}
```

```
int C(int k, int n) {  
    if(k == n || k == 0) return 1;  
    return C(k-1, n-1) + C(k, n-1);  
}
```

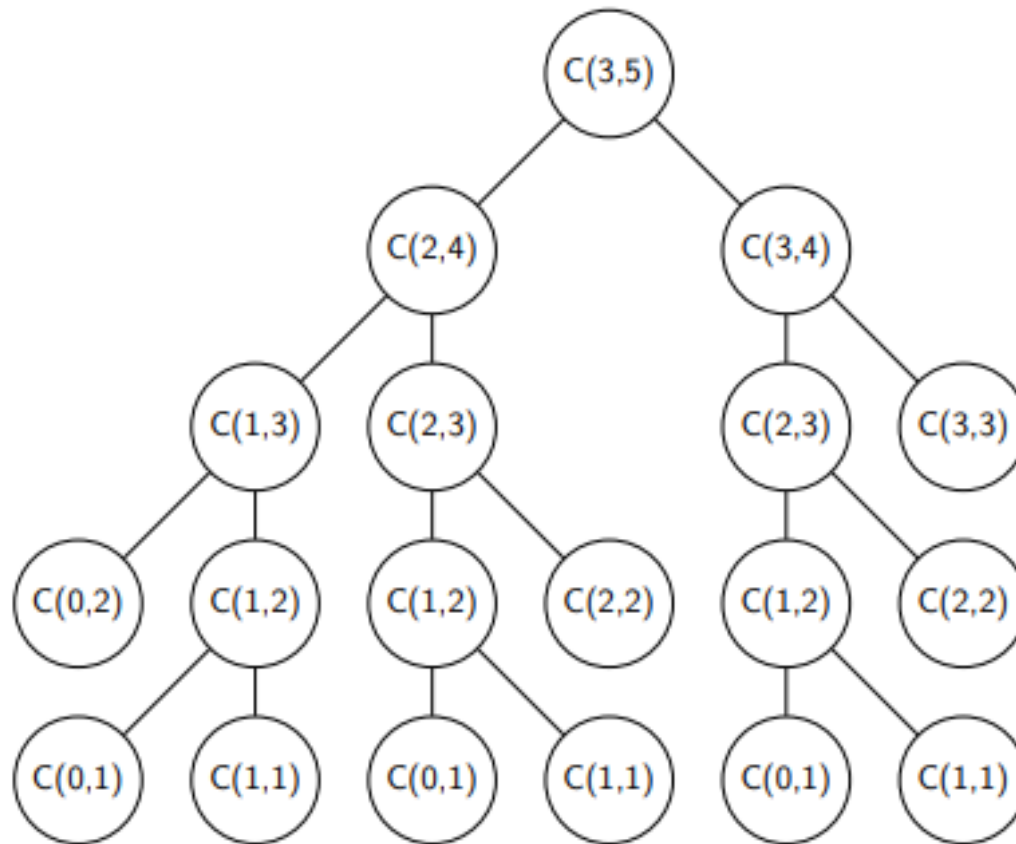
Recursion

□ Đệ quy có nhớ

- Các hàm có cùng tham số có thể được gọi nhiều lần
- Một hàm với bộ tham số nhất định được thực thi lần đầu và kết quả sẽ được lưu vào bộ nhớ
- Nếu hàm được gọi với cùng một bộ tham số, hàm sẽ không thực thi. Thay vào đó, kết quả của hàm đó có sẵn trong bộ nhớ sẽ được trả về trực tiếp.

Recursion

□ Đệ quy có nhớ



Recursion

```
public class Ckn {
    private int [][] M;
    public int C(int k, int n) {
        if (k == 0 || k == n) M[k][n] = 1;
        else if (M[k][n] < 0) {
            M[k][n] = C(k-1, n-1) + C(k, n-1);
        }
        return M[k][n];
    }
    public void test() {
        M = new int[100][100];
        for(int i = 0; i < 100; i++)
            for(int j = 0; j < 100; j++)
                M[i][j] = -1;

        System.out.println(C(15, 30));
    }
}
```

Nội dung

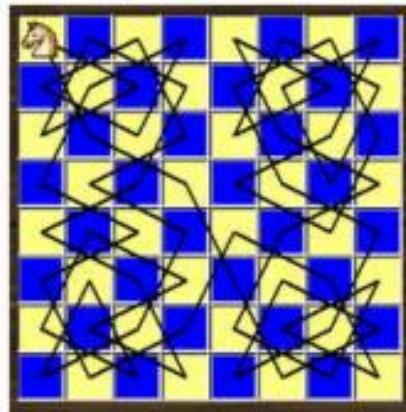
□ Backtracking

- **Lược đồ chung**
- Bài toán 8 hậu
- Bài toán mã đi tuần
- Liệt kê các hoán vị
- Liệt kê dãy nhị phân độ dài N
- Duyệt đồ thị

Backtracking

□ Bài toán mở đầu

- Phương pháp quay lui dùng để giải các bài toán mà lời giải của nó (X) là một tập các phần tử x_1, x_2, \dots, x_n .
- Ví dụ: bài toán 8 hậu, mã đi tuần



Backtracking

□ Ý tưởng

- Ý tưởng chính của phương pháp quay lui là các bước hướng tới lời giải cuối cùng của bài toán dựa trên việc “Thử và sai”

Backtracking

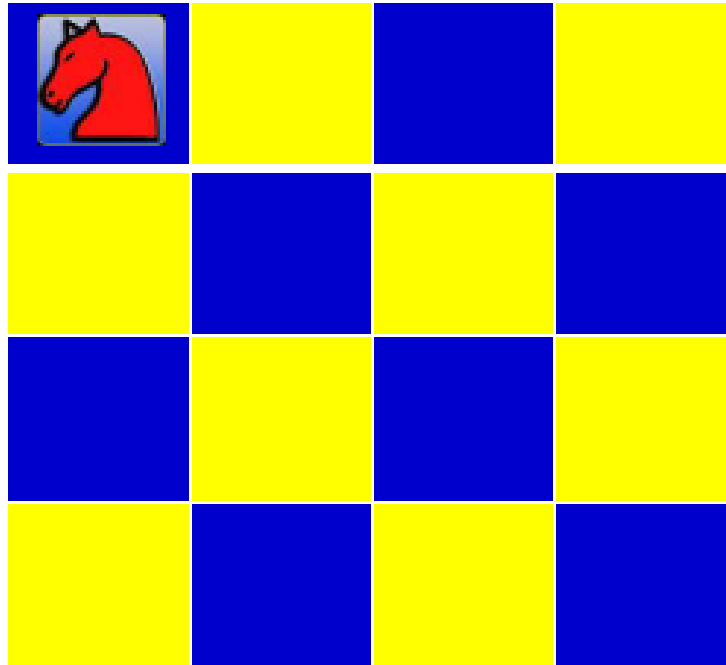
□ Ý tưởng

- Tại mỗi bước:
 - Nếu có 1 lựa chọn được chấp nhận thì ghi nhận lại lựa chọn này và tiến hành các bước thử tiếp theo
 - Nếu tất cả các lựa chọn không được chấp nhận thì trở lại bước trước, xóa bỏ sự ghi nhận ứng viên và chọn lựa ứng viên tiếp theo

Backtracking

□ Ví dụ

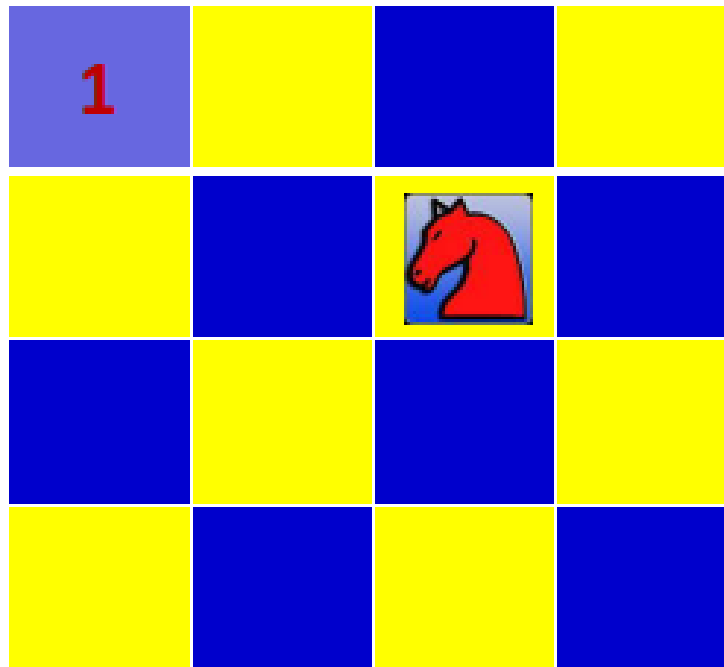
- Mã đi tuần trên bàn cờ 4x4 – bắt đầu từ ô (1, 1)



Backtracking

□ Ví dụ

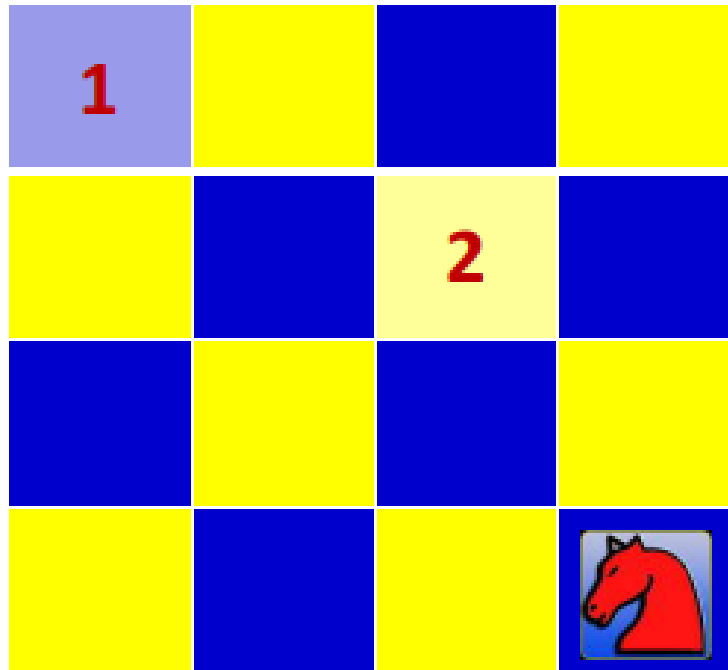
- Mã đi tuần trên bàn cờ 4x4 – bắt đầu từ ô (1, 1)



Backtracking

□ Ví dụ


- Mã đi tuần trên bàn cờ 4x4 – bắt đầu từ ô (1, 1)



Backtracking

□ Ví dụ


- Mã đi tuần trên bàn cờ 4x4 – bắt đầu từ ô (1, 1)

1	14	5	
8	11	2	
13	4	9	6
10	7	12	3

Backtracking

□ Ví dụ


- Mã đi tuần trên bàn cờ 4x4 – bắt đầu từ ô (1, 1)

1		5	
8	11	2	
13	4	9	6
10	7	12	3

Backtracking

□ Ví dụ


- Mã đi tuần trên bàn cờ 4x4 – bắt đầu từ ô (1, 1)

1		5	
8	11	2	
	4	9	6
10	7	12	3

Backtracking

□ Ví dụ

- Mã đi tuần trên bàn cờ 4x4 – bắt đầu từ ô (1, 1)

1		5	
8	11	2	
	4	9	6
10	7		3

Backtracking

□ Ví dụ

- Mã đi tuần trên bàn cờ 4x4 – bắt đầu từ ô (1, 1)

1		5	
8	11	2	
	4	9	6
10	7	12	3

Backtracking

□ Nhận xét

- Khi thực hiện Backtrack, điểm quan trọng của thuật toán là phải ghi nhớ mỗi bước đi để tránh trùng lặp khi “quay lui”
- Dễ thấy, cấu trúc ngăn xếp khá phù hợp để lưu trữ các thông tin cần ghi nhớ như đề cập trên
- Đệ quy là kỹ thuật thường được sử dụng trong phương pháp Backtrack.

Backtracking

□ Lược đồ chung

- Lời giải bài toán có thể mô tả dạng 1 vector n chiều $X = \{x_1, x_2, \dots, x_n\}$ thỏa mãn 1 điều kiện nào đó.

Backtracking

□ Lược đồ chung

- Giả sử đã xây dựng được $i-1$ thành phần $\{x_1, x_2, \dots, x_{i-1}\}$, cần xác định thành phần thứ i :
 - Nếu khả năng k nào đó phù hợp $\rightarrow x_i=k$, ghi nhận trạng thái đã dung của k . Nếu $i=n \rightarrow$ có được 1 lời giải
 - Nếu không có khả năng nào cho x_i thì quay lui và chọn lại x_{i-1}

Backtracking

□ Lược đồ chung

```
Try(i) {  
    for(j = 1 -> k) {  
        if ( $x_i$  chấp nhận được khả năng j) {  
            Xác định  $x_i$  theo khả năng j;  
            Ghi nhận trạng thái mới;  
            if (i < n)  
                Try (i + 1);  
            else  
                Ghi nhận nghiệm;  
            Trả lại trạng thái cũ cho bài toán  
        }  
    }  
}
```

Nội dung

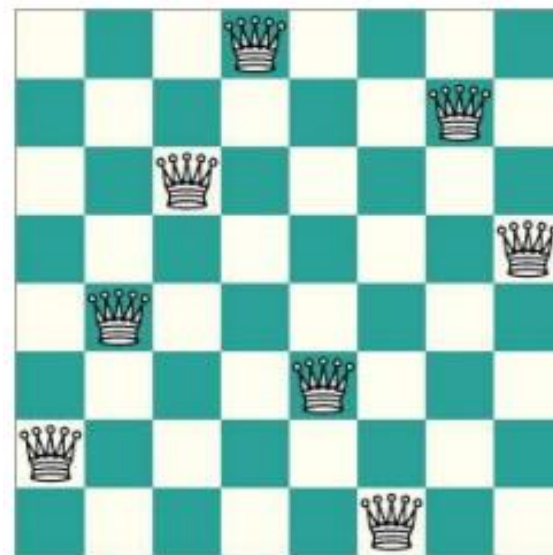
□ Backtracking

- Lược đồ chung
- Bài toán 8 hậu
- Bài toán mã đi tuần
- Liệt kê các hoán vị
- Liệt kê dãy nhị phân độ dài N
- Duyệt đồ thị

Backtracking

□ Bài toán 8 hậu

- Hãy tìm cách xếp 8 con hậu trên bàn cờ vua sao cho không con nào ăn con được nhau
- Ví dụ: đây là một phương án



Backtracking

□ Ý tưởng thuật toán (Thử và sai)

1. Lần lượt xếp các con hậu vào bàn cờ
2. Giả sử đã xếp được i con hậu (từ 1 đến i)
3. Xếp hậu thứ $i+1$
 - a. Nếu tìm được 1 ô hợp lệ (không bị các con hậu trước đó ăn) \rightarrow xếp hậu thứ $i+1$ vào vị trí vừa tìm thấy. Lặp lại bước 3
 - b. Nếu không tìm được ô hợp lệ \rightarrow tìm vị trí phù hợp khác để đặt lại hậu thứ i .

Backtracking

□ Phương án nghiệm

- Nhận xét: mỗi con hậu phải nằm trên 1 hàng
- Dùng mảng $x[1..8]$ để thể hiện một phương án của bài toán:
 - Chỉ số mảng i : dòng chứa con hậu thứ i (chỉ số dòng là cố định)
 - Giá trị $x[i]$ ($i=1..8$): là cột đặt con hậu thứ i
- Bài toán xếp hậu trở thành: lần lượt xác định giá trị các thành phần của $x[i]$, $x=1..8$

Backtracking

□ Ví dụ

- Phương án nghiệm

$$x[1] = 4$$

$$x[2] = 7$$

$$x[3] = 3$$

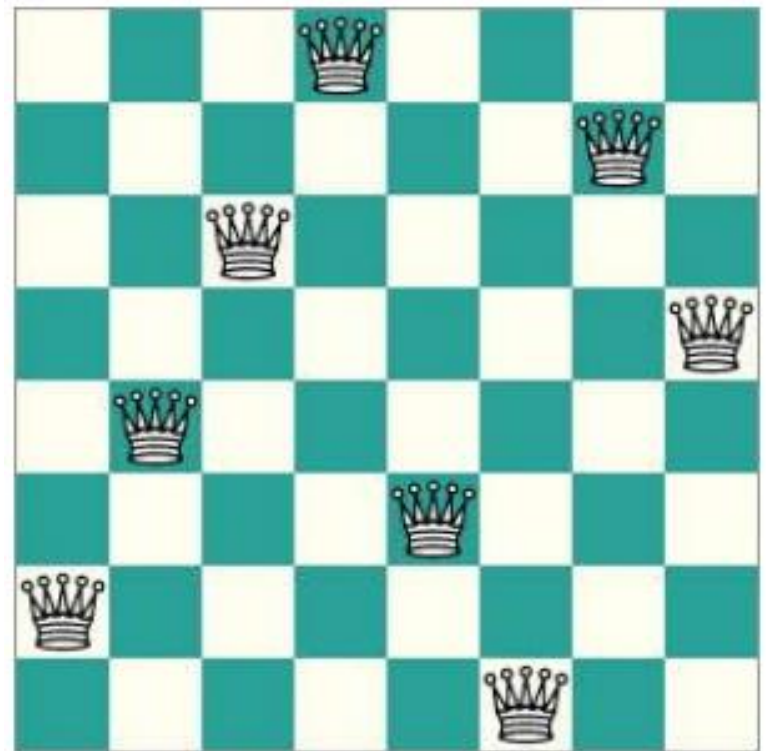
$$x[4] = 8$$

$$x[5] = 2$$

$$x[6] = 5$$

$$x[7] = 1$$

$$x[8] = 6$$



Backtracking

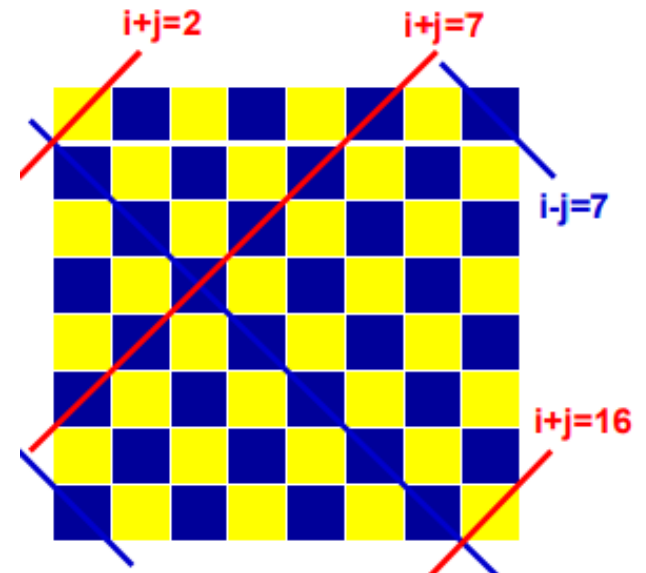
□ Ứng viên

- Tại bước i :
 - Cần xác định giá trị k , là chỉ số cột cho $x[i]$, $k = \{1, \dots, 8\}$.
 - Nếu ứng viên được chọn là j , nghĩa là $x[i]=j$, khi đó cần “đánh giá” là cột j đã được chọn để bước sau không chọn lại.
- Tổ chức mảng $a[j]$, $j=1..8$, để ghi nhận cột j đã được chọn hay chưa, $a[j]=1$ là cột j chưa được chọn và $a[j]=0$ là cột j đã được chọn

Backtracking

□ Tính hợp lệ

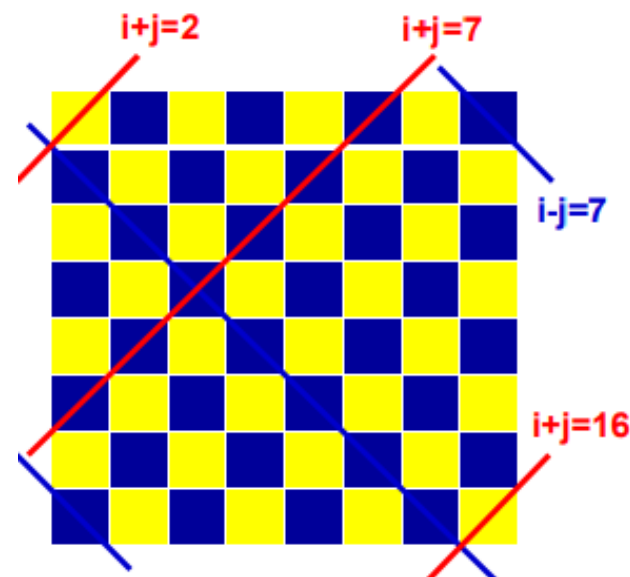
- Hậu ở dòng i , chỉ được đặt vào cột j nếu $i-1$ hậu đã được đặt trước đó không “ăn” được hậu ở vị trí $[i,j]$ (dòng i , cột j)
- Trên đường chéo đỏ:
 - Giá trị $i+j$ là hằng số
 - Có giá trị từ 2 đến 16
- Trên đường chéo xanh
 - Giá trị $i-j$ là hằng số
 - Có giá trị từ -7 đến 7



Backtracking

□ Tính hợp lệ

- Mảng $b[k]$, $k=2..16$, nếu $b[k]=1$ được đặt ở đường chéo thuận k .
- Mảng $c[k]$, $k=-7..7$, nếu $c[k]=1$ được đặt ở đường chéo nghịch k



Backtracking

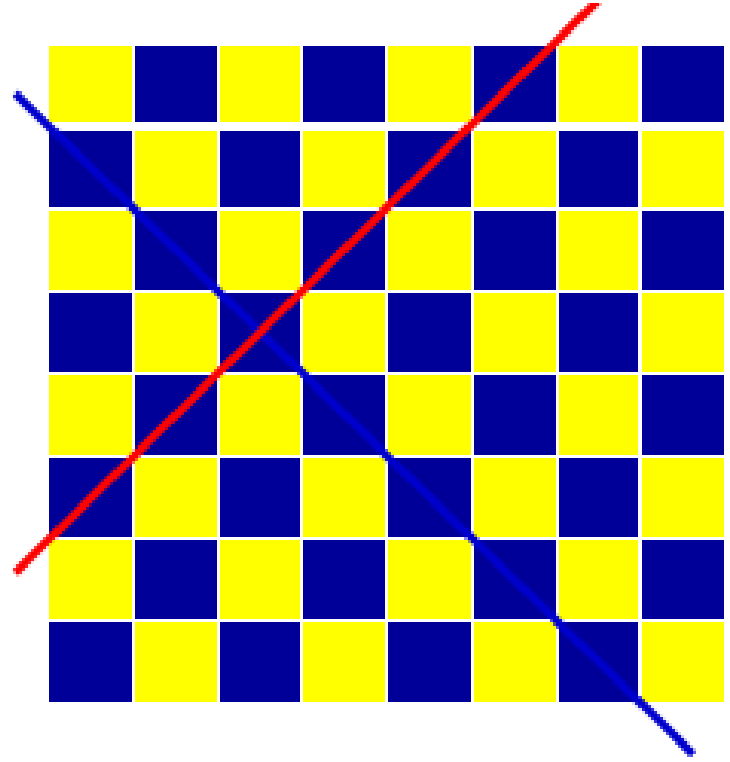
□ Tính hợp lệ

- Như vậy hậu i (dòng i) được đặt vào cột j nếu:

$$b[i+j] = 1$$

và

$$c[i-j] = 1$$



Backtracking

Khởi tạo

a[j] = 1
b[i+j] = 1
c[i-j] = 1

```
Try(i) {  
    for(j=1; j<=8; j++) {  
        if(a[j] && b[i+j] && c[i-j]) {  
            x[i] = j; a[j] = 0;  
            b[i+j] = 0; c[i-j] = 0;  
            if (i < 8)  
                Try(i + 1);  
            else  
                Xuất(x);  
            /* Sau khi in 1 lời giải xong, trả lại tình trạng  
               ban đầu còn trống cho hàng a[j], đường  
               chéo i+j và đường chéo i-j, để tìm lời giải  
               khác */  
            a[j] = 1; b[i+j] = 1; c[i-j] = 1;  
        }  
    }  
}
```

Backtracking

□ Minh họa

- Một lời giải của bài toán với $N=8$

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
1	5	8	6	3	7	2	4

H							
				H			
							H
					H		
		H					
						H	
	H						
			H				

Backtracking

□ Kết quả

- Độ phức tạp thuật toán: $T(n) = ?$
- Viết hàm $Xuat(x)$: in phương án lựa chọn
- Code, chạy thử và submit trên SPOJ

Nội dung

□ Backtracking

- Lược đồ chung
- Bài toán 8 hậu
- Bài toán mã đi tuần
- Liệt kê các hoán vị
- Liệt kê dãy nhị phân độ dài N
- Duyệt đồ thị

Backtracking

❑ Ý tưởng thuật toán (Thử và sai)

1. Đặt mã tại vị trí (x_0, y_0) di chuyển mã theo luật cờ vua
2. Giả sử đã xếp được $i-1$ bước
3. Xét nước đi thứ i
 - a. Nếu tìm được 1 ô hợp lệ (và mã chưa qua lần nào) \rightarrow xếp nước đi thứ i của mã vào vị trí vừa tìm thấy. Lặp lại bước 3
 - b. Nếu không tìm được ô hợp lệ \rightarrow tìm vị trí phù hợp khác để đặt lại bước đi thứ $i-1$ của mã.

Backtracking

□ Phương án nghiệm

- Dùng mảng 2 chiều $h[x, y]$ ($x=1..N, y=1..N$) với quy ước:
 - $h[x, y] = 0$ là ô (x, y) chưa có ngựa đi qua
 - $h[x, y] = k$ là mã đã qua ô (x, y) ở nước đi thứ k
- Bài toán trở thành: Xác định giá trị mảng h là nước đi của mã trong hành trình đi qua tất cả các ô bắt đầu từ (x_0, y_0) . Khi $N \times N$ ô được đi qua ta có 1 nghiệm thể hiện cách đi của mã.

Backtracking

□ Ví dụ

- Một phương án để mã đi tuần trên bàn cờ 5x5 bắt đầu từ ô (1, 1):

1	18	13	8	3
12	7	2	19	14
17	24	21	4	9
22	11	6	15	20
25	16	23	10	5

Backtracking

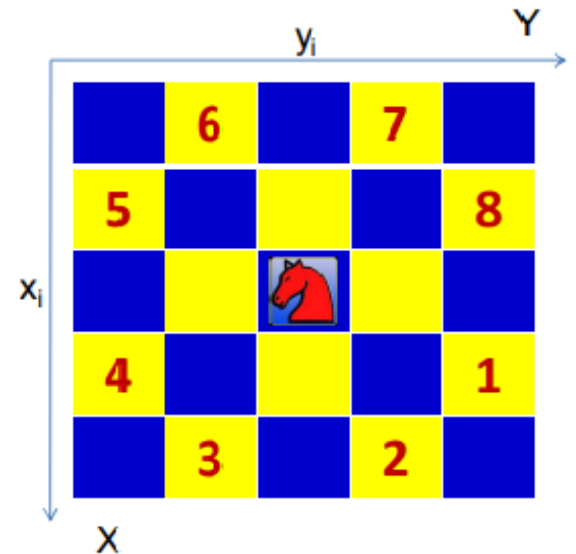
□ Ứng viên

- Tại bước i:

- Vị trí mã đang đứng là (x_i, y_i)
- Mã có thể di chuyển tối đa 8 ô (hình bên)
- Tọa độ 8 vị trí so với vị trí hiện tại (x_i, y_i) lần lượt là:

$(x_i + 1, y_i + 2), (x_i + 2, y_i + 1), (x_i + 2, y_i - 1), (x_i + 1, y_i - 2)$

$(x_i - 1, y_i - 2), (x_i - 2, y_i - 1), (x_i - 2, y_i + 1), (x_i - 1, y_i + 2)$



Backtracking

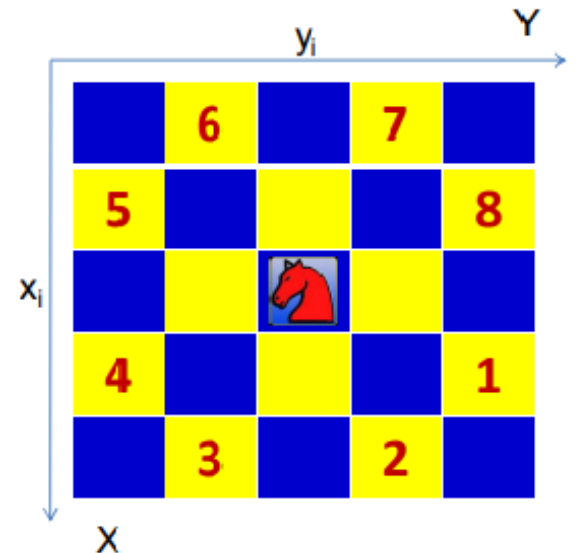
□ Ứng viên

- Tại bước i:

- Dùng mảng $a[1..8]$ mô tả độ lệch tọa độ X so với x_i theo trên ta có:

$$a = (1, 2, 2, 1, -1, -2, -2, -1)$$

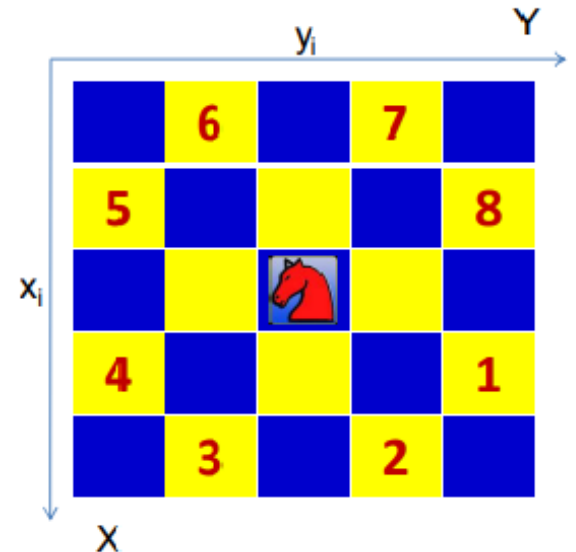
- Dùng mảng $b[1..8]$ mô tả độ lệch tọa độ Y so với y_i theo trên ta có
- $$b = (2, 1, -1, -2, -2, -1, 1, 2)$$



Backtracking

□ Ứng viên

- Tại bước i :
 - Ứng viên của bước $i+1$ được xác định tại tọa độ $(x_i + a[k], y_i + b[k])$ với $k=1..8$
- Tính hợp lệ:
 - Ứng viên tại tọa độ $(x_i + a[k], y_i + b[k])$ với $k=1..8$ được chấp nhận nếu $h[x_i + a[k], y_i + b[k]] = 0$
 - Ngoài ra $(x_i + a[k], y_i + b[k])$ phải nằm trong bàn cờ



Backtracking

```
Try(i, x, y) {  
    for(k=1; k<=8; k++) {  
        u = x + a[k]; v = y + b[k];  
        if (1 <= u, v <= n && h[u][v] == 0) {  
            h[u][v] = i;  
            if (I < n*n)  
                Try(i+1, u, v);  
            else  
                xuất_nghiệm(); // in ma trận h  
        }  
        h[u][v] = 0;  
    }  
}
```

Backtracking

□ Minh họa

- Với $N=5$, mã xuất phát tại (1,1)

1	6	15	10	21
14	9	20	5	16
19	2	7	22	11
8	13	24	17	4
25	18	3	12	23

Backtracking

□ Kết quả

- Độ phức tạp thuật toán: $T(n) = ?$
- Viết hàm `xuat_nghiem()`: in phương án nghiệm
- Code, chạy thử và submit trên SPOJ
- Lưu ý: tùy vào kích thước bàn cờ, bài toán có lời giải ở một số vị trí bắt đầu (x_0, y_0) nhất định

Nội dung

□ Backtracking

- Lược đồ chung
- Bài toán 8 hậu
- Bài toán mã đi tuần
- **Liệt kê các hoán vị**
- Liệt kê dãy nhị phân độ dài N
- Duyệt đồ thị

Backtracking

□ Liệt kê các hoán vị

- Có N đối tượng ($1..N$), hãy liệt kê tất cả các hoán vị có thể của N đối tượng đó
- Bài toán trên có thể quy về bài toán: Liệt kê tất cả các hoán vị của N số nguyên đầu tiên
- Ví dụ: Các hoán vị của 3 số: 1, 2, 3
 - 123, 132, 213, 231, 312, 321 (thứ tự từ điển)
 - 321, 312, 231, 213, 132, 123 (thứ tự TĐ ngược)

Backtracking

□ Ý tưởng thuật toán (Thử và sai)

1. Cần xếp các số từ 1-N vào N vị trí (khác nhau từng đôi một)
2. Giả sử đã xếp được đến vị trí thứ $i-1$
3. Tìm 1 giá trị thích hợp (chưa được dùng) trong khoảng từ 1 đến N cho vị trí thứ i . Lặp lại bước 3 khi $i < N$

Backtracking

□ Phương án nghiệm

- Bộ gồm N số từ 1 đến N khác nhau từng đôi một.
- Ứng viên: các giá trị từ 1 đến N
- Tính hợp lệ: $x[i]$ nhận giá trị j nếu j chưa được dùng cho các $x[1]$ đến $x[i-1]$

Backtracking

□ Cài đặt

- Dùng mảng $b[j]$: $i=1..N$ để đánh dấu giá trị j đã được dùng hay chưa
 - $b[j] = 0$: j đã được dùng
 - $b[j] = 1$: j chưa được dùng

Backtracking

```
Try(i, x, y) {  
    for(j=1; k<=n; k++) {  
        if (b[j]) {  
            a[i] = j;  
            b[j] = 0;    // Ghi nhận trạng thái  
            if(i < n)  
                Try(i + 1);  
            else  
                Xuat();  
            b[j] = 1;    // Trả lại trạng thái cũ  
        }  
    }  
}
```

Backtracking

□ Kết quả

- Độ phức tạp thuật toán: $T(n) = ?$
- Viết hàm xuất(): in phương án nghiệm
- Code, chạy thử và submit trên SPOJ

Nội dung

□ Backtracking

- Lược đồ chung
- Bài toán 8 hậu
- Bài toán mã đi tuần
- Liệt kê các hoán vị
- Liệt kê dãy nhị phân độ dài N
- Duyệt đồ thị

Backtracking

□ Liệt kê dãy nhị phân độ dài N

- Bài toán 1: Liệt kê tất cả các số nhị phân có độ dài N. Ví dụ với $N=3$, ta có các liệt kê sau:
000, 001, 010, 011, 110, 101, 110, 111
- Bài toán 2: Liệt kê tập tất cả các tập con của tập N phần tử
- Nhận xét: Bài toán 2 có thể chuyển về bài toán 1

Backtracking

□ Ý tưởng thuật toán (Thử và sai)

- Ta có thể sử dụng sơ đồ tìm tất cả các lời giải của bài toán. Hàm Try(i) xác định x_i , trong đó x_i chỉ có 1 trong 2 giá trị 0 hay 1.
- Các giá trị này mặc nhiên được chấp nhận mà không cần phải thỏa mãn điều kiện gì

Backtracking

❑ Phương án nghiệm

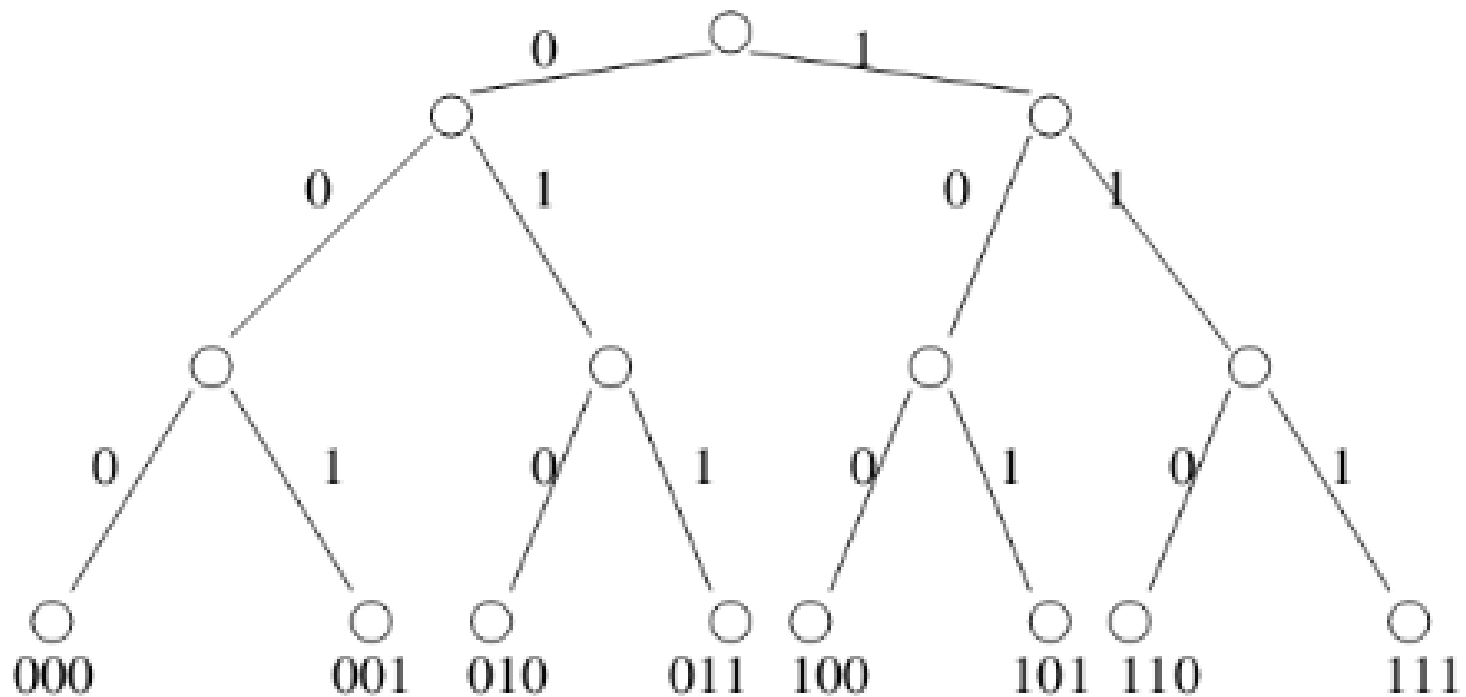
- Dãy N các giá trị 0, 1 (đảm bảo không có 2 nghiệm trùng nhau)

Backtracking

```
Try(i, x, y) {  
    for(j = 0; j<=1; j++) {  
        x[i] = j;  
        if(i < n)  
            Try(i + 1);  
        else  
            Xuat(x);           // Xuat nghiem  
    }  
}
```

Backtracking

□ Minh họa



Backtracking

□ Kết quả

- Độ phức tạp thuật toán: $T(n) = ?$
- Viết hàm xuất(): in phương án nghiệm
- Code, chạy thử và submit trên SPOJ

Nội dung

□ Backtracking

- Lược đồ chung
- Bài toán 8 hậu
- Bài toán mã đi tuần
- Liệt kê các hoán vị
- Liệt kê dãy nhị phân độ dài N
- Duyệt đồ thị

Nội dung

□ Duyệt đồ thị

- DFS
- BFS
- 2 nội dung này sẽ được tìm hiểu trong phần:
Thuật toán về đồ thị và ứng dụng



