

Tên bài giảng

Phương pháp phân tích thiết kế quay lui

Môn học: **Phân tích thiết kế thuật toán**

Chương: 4

Hệ: Đại học

Giảng viên: TS. Phạm Đình Phong

Email: phongpd@utc.edu.vn

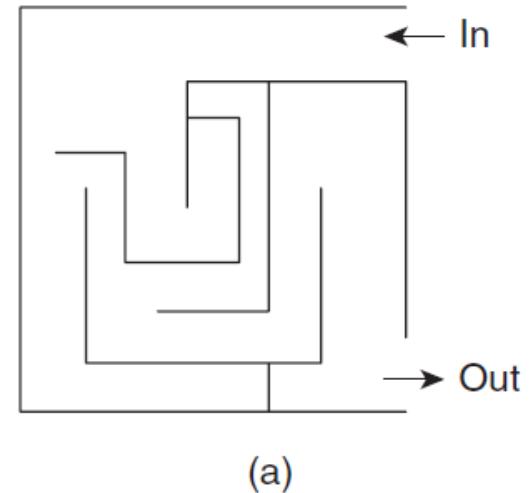
Nội dung bài học

- 1. Phân tích thiết kế, đánh giá thuật toán quay lui**
- 2. Một số thuật toán quay lui vét cạn**
- 3. Một số thuật toán quay lui trên dữ liệu nhiều chiều**

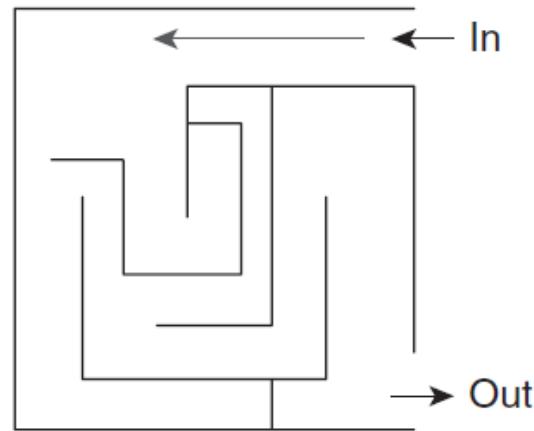
Phương pháp quay lui

- **Ý tưởng**

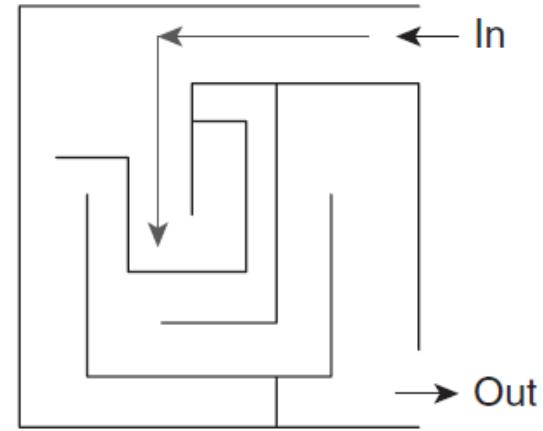
Một người đi vào căn phòng bên trong phòng có nhiều vách ngăn từ cửa vào (In) và tìm cửa ra (Out)



(a)

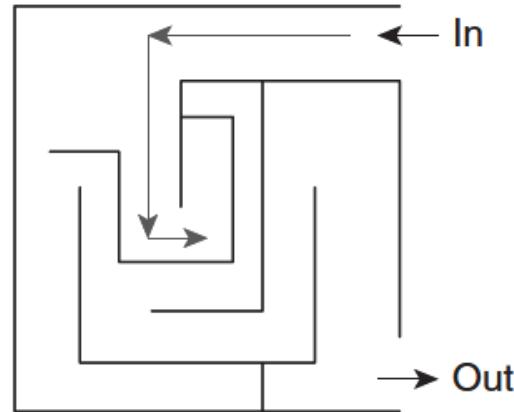


(b)

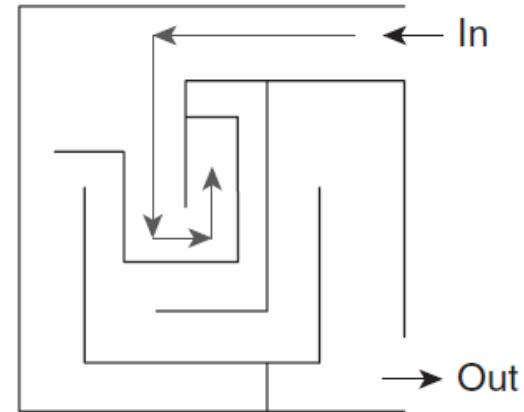


(c)

Phương pháp quay lui

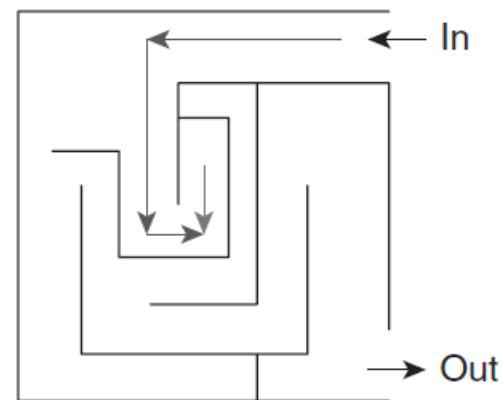


(d)



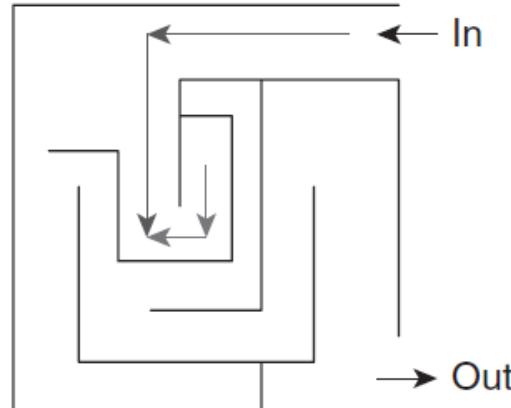
(e)

Gặp ngõ cùt nên quay trở ra

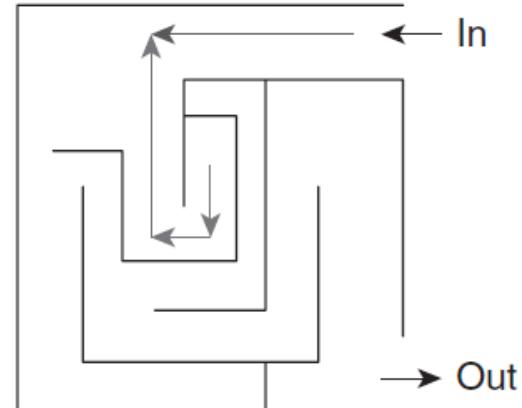


(f)

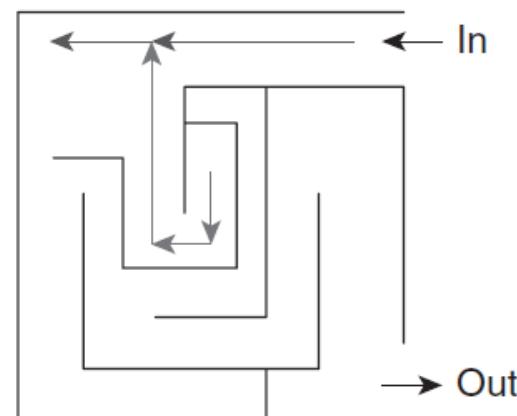
Phương pháp quay lui



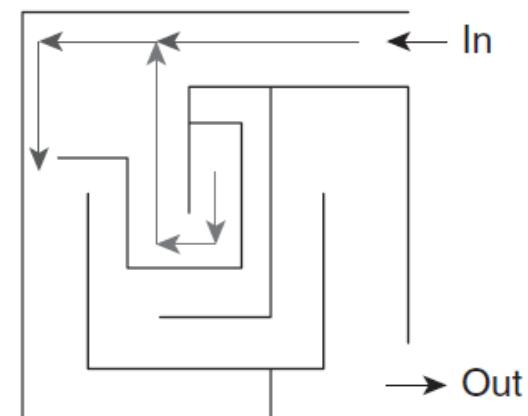
(g)



(h)

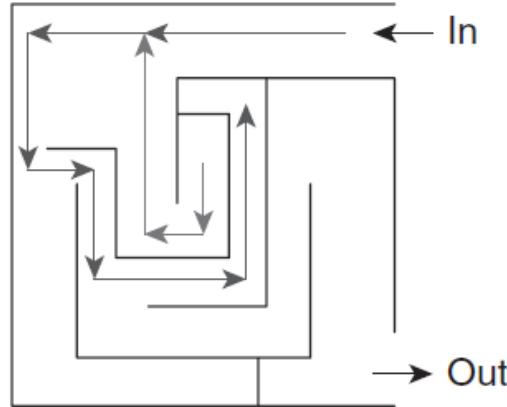


(i)

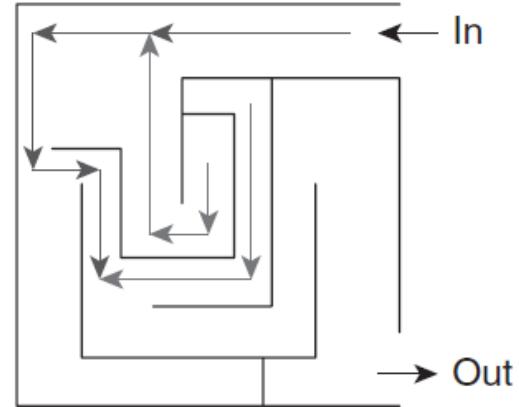


(j)

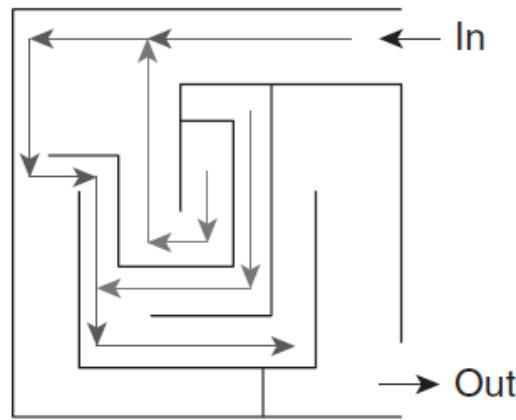
Phương pháp quay lui



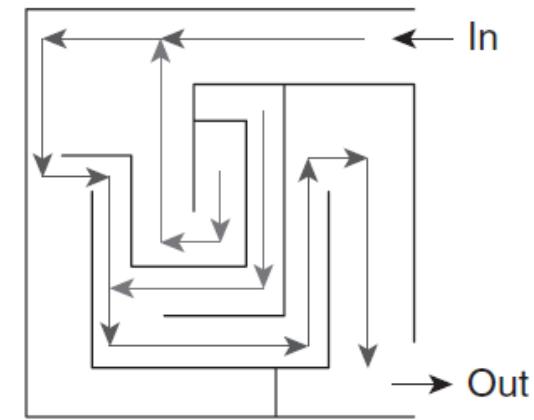
(k)



(l)



(m)



(n)

Phương pháp quay lui

- **Ý tưởng**
 - Kỹ thuật thiết kế thuật toán dựa trên đệ quy
 - Tìm lời giải từng bước, tại mỗi bước
 - Nếu có một lựa chọn chấp nhận được → ghi nhận và tiến hành thử bước tiếp theo
 - Nếu không có lựa chọn nào khả thi
 - Quay lại bước trước
 - Xóa bỏ ghi nhận
 - Thủ các lựa chọn còn lại tại bước này
 - Phù hợp với bài toán liệt kê cấu hình dạng $X[1, \dots, n]$

Phương pháp quay lui

- **Bài toán liệt kê**

Cho A_1, A_2, \dots, A_n là các tập hữu hạn. Ký hiệu

$$X = A_1 \times A_2 \times \dots \times A_n = \{(x_1, x_2, \dots, x_n) : x_i \in A_i, i = 1, 2, \dots, n\}$$

Giả sử P là tính chất cho trên X . Vấn đề đặt ra là liệt kê tất cả các phần tử của X thoả mãn tính chất P :

$$D = \{x = (x_1, x_2, \dots, x_n) \in X : x \text{ thoả tính chất } P\}$$

Các phần tử của tập D được gọi là các ***lời giải chấp nhận được***

Phương pháp quay lui

- Ví dụ

- Bài toán liệt kê chuỗi nhị phân độ dài n dẫn về việc liệt kê các phần tử của tập

$$B^n = \{(x_1, \dots, x_n) : x_i \in \{0, 1\}, i = 1, 2, \dots, n\}$$

- Tập các hoán vị của các số tự nhiên 1, 2, ..., n là tập

$$I^n = \{(x_1, \dots, x_n) \in N^n : x_i \neq x_j ; i \neq j\}$$

Phương pháp quay lui

- Lời giải bộ phận

- Ta gọi lời giải bộ phận cấp k ($0 \leq k \leq n$) là bộ có thứ tự gồm k thành phần (x_1, x_2, \dots, x_k) , trong đó $x_i \in A_i$, $i = 1, 2, \dots, k$
- Khi $k = 0$, lời giải bộ phận cấp 0 được ký hiệu là () và còn được gọi là lời giải rỗng
- Nếu $k = n$, ta có lời giải đầy đủ hay đơn giản là một lời giải của bài toán

Phương pháp quay lui

- Mô hình thuật toán
 - Tại mỗi bước i
 - Đã xây dựng xong lời giải bộ phận cấp $i - 1$ (gồm các thành phần x_1, \dots, x_{i-1})
 - Xây dựng thành phần thứ i với tất cả các khả năng A_i
 - Nếu trên cơ sở tính chất P , có một phần tử $a_j \in A_i$ phù hợp (thỏa tính chất P) với vị trí thứ $i \rightarrow$ chọn a_j vào vị trí thứ i của lời giải. Tập các $\{a_j\}$ như vậy được gọi là tập ứng cử viên (UCV) vào vị trí thứ i của lời giải, ký hiệu là S_i
 - Nếu $i = n$ (có lời giải đầy đủ) \rightarrow ghi nhận một lời giải
 - Ngược lại, tiến hành bước $i + 1$ để xác định x_{i+1}
 - Nếu trên cơ sở tính chất P , không có $a_j \in A_i$ nào phù hợp với vị trí thứ i thì lùi lại bước $i-1$ để xác định lại x_{i-1} rồi tiếp tục xây dựng thành phần thứ i

Phương pháp quay lui

```
backtracking(i) {  
    for (Mỗi j ∈ Ai) {  
        if (j thỏa tính chất P) {  
            Chọn j cho xi;  
            if (i == n) { //Đã xác định được đủ n thành phần  
                Đưa ra kết quả;  
            } else {  
                backtracking(i + 1);  
                Bỏ chọn j cho xi;  
            }  
        }  
    }  
}
```

Lệnh gọi để thực hiện thuật toán quay lui là: backtracking(1)

Phương pháp quay lui

- Nhận xét
 - Để xây dựng thuật toán quay lui ta cần:
 - Biết dạng của cấu hình cần tìm
 - Độ dài của lời giải là không biết trước
 - Các lời giải không nhất thiết phải có cùng độ dài
 - Cần kiểm tra (x_1, x_2, \dots, x_k) đã là lời giải hay chưa
 - Xác định các khả năng của x_i (số phần tử của tập A_i) \rightarrow ít nhất có thể. Khi đó, ta có $A_i = S_i$.
 - Xác định điều kiện thỏa tính chất P cho vị trí thứ i (điều kiện “chấp nhận khả năng j ” cho x_i) \rightarrow đơn giản nhất có thể

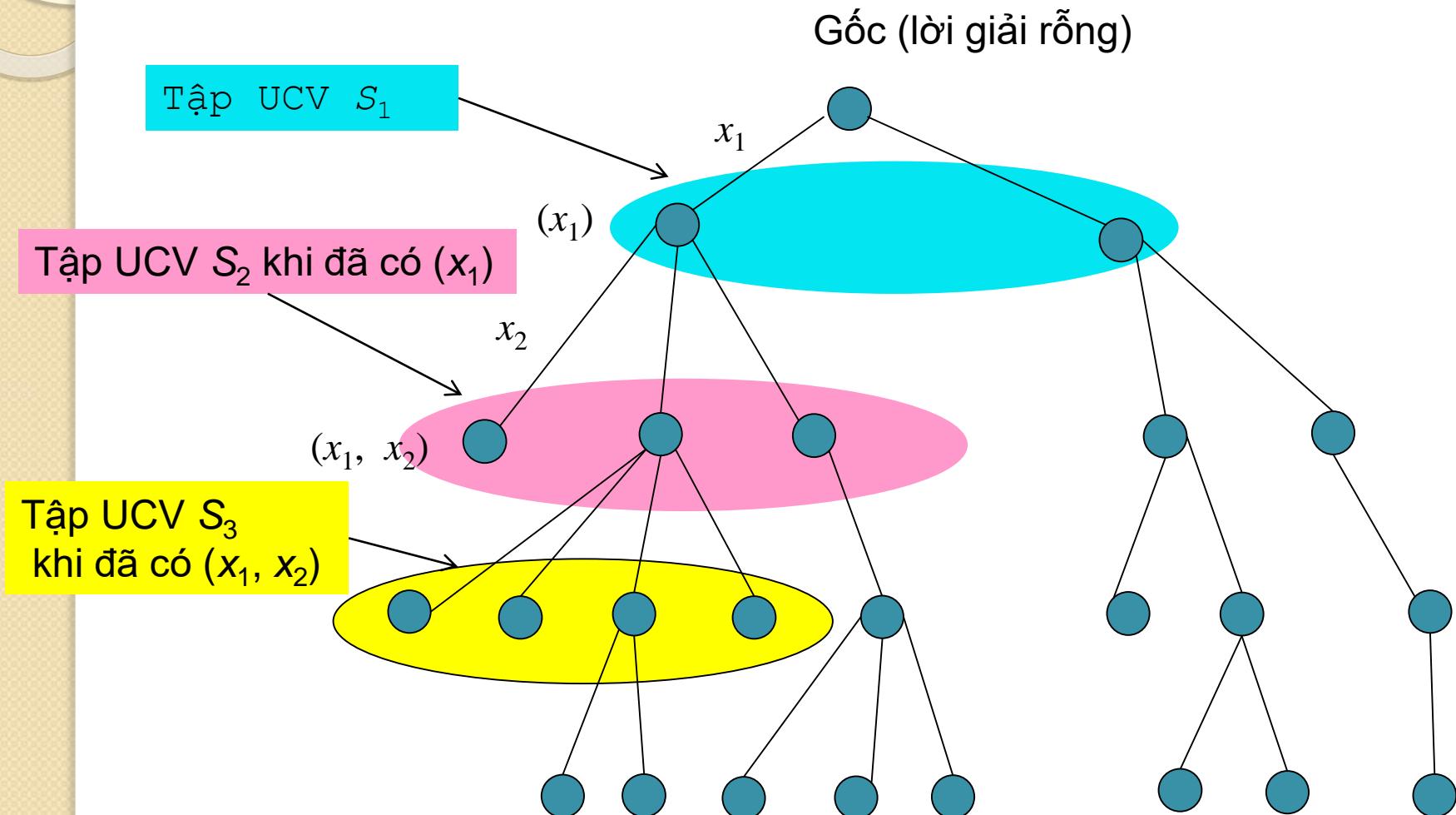
Phương pháp quay lui

- Nhận xét

- Nếu chỉ cần tìm một lời giải thì cần tìm cách chấm dứt các thủ tục gọi đệ qui lồng nhau sinh bởi lệnh gọi Backtracking(1) sau khi ghi nhận được lời giải đầu tiên
- Nếu kết thúc thuật toán mà ta không thu được một lời giải nào thì điều đó có nghĩa là bài toán không có lời giải

Phương pháp quay lui

- Cây liệt kê lời giải



Liệt kê dãy nhị phân có độ dài n

- Dãy nhị phân có độ dài n là dãy số có số chữ số bằng n và các chữ số trong dãy chỉ gồm 0 và 1
- Ví dụ: các dãy nhị phân có độ dài 3

0 0 0
0 0 1
0 1 0
0 1 1
1 0 0
1 0 1
1 1 0
1 1 1

Liệt kê dãy nhị phân có độ dài n

- Bài toán dẫn về việc liệt kê các phần tử của tập
$$B^n = \{(x_1, \dots, x_n) : x_i \in \{0, 1\}, i = 1, 2, \dots, n\}$$

→ Cấu hình cần tìm có dạng: (x_1, x_2, \dots, x_n)
- Giả sử đã có xâu nhị phân cấp $i - 1$ (x_1, \dots, x_{i-1}), khi đó $x_i = \{0, 1\}$

→ Các khả năng của x_i là 0 và 1 hay $S_i = \{0, 1\}$
- Chấp nhận khả năng j của x_i : luôn chấp nhận

0 0 0

0 0 1

0 1 0

0 1 1

1 0 0

...

Liệt kê dãy nhị phân có độ dài n

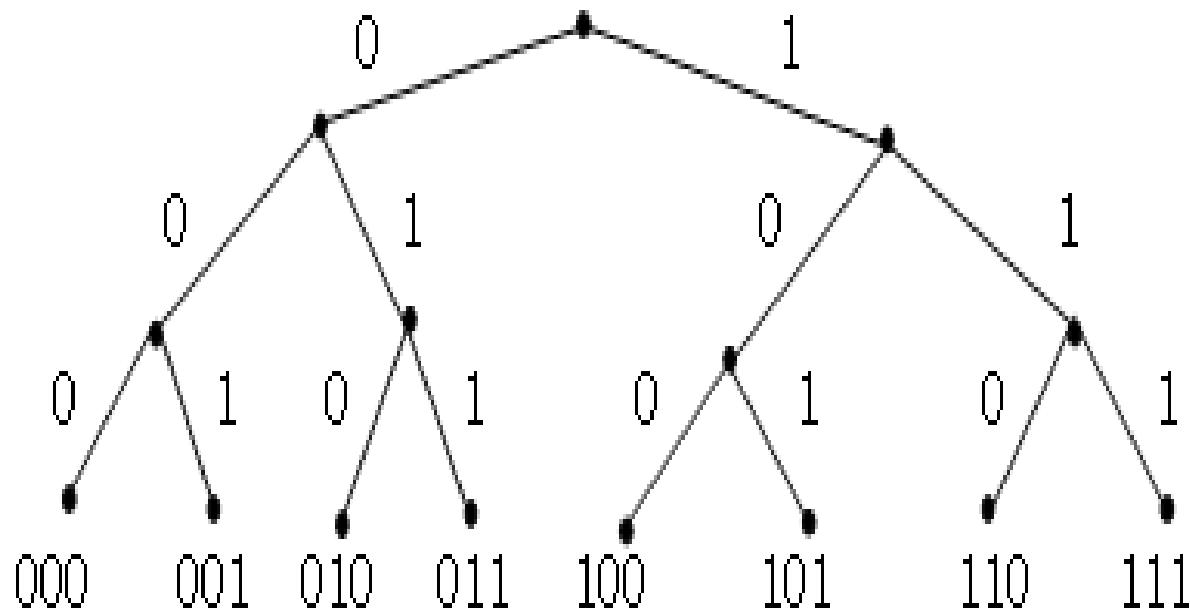
- Mã giả của thuật toán

```
void backtracking(int i ) {  
    for (int j=0; j<=1; j++) {  
        x[i] = j;  
        if (i == n) //nếu tìm đến  $x_i$  cuối cùng thì xuất ra kết quả  
            output();  
        else  
            backtracking(i+1); // chưa tìm đến i cuối thì tăng i lên  
    }  
}
```

Lệnh gọi để thực hiện thuật toán quay lui là: backtracking(1)

Liệt kê dãy nhị phân có độ dài n

- Cây liệt kê dãy nhị phân độ dài 3



Bài toán liệt kê tổ hợp chập k của n

- Mô tả bài toán

- Tổ hợp chập k phần tử của $X = \{1, \dots, n\}$ là một tập con k phần tử của X
- Ví dụ: $k = 3, n = 4, X = \{1, 2, 3, 4\}$, ta có các tổ hợp: $(1, 2, 3), (1, 2, 4), (1, 3, 4), (2, 3, 4)$

Bài toán liệt kê tổ hợp chập k của n

- Bài toán dẫn về bài toán liệt kê các phần tử của tập:
 $S(k, n) = \{(x_1, \dots, x_k) \in N^k: 1 \leq x_1 < \dots < x_k \leq n\}$
→ **Cấu hình cần tìm dạng:** (x_1, x_2, \dots, x_k) với $x_{i-1} < x_i$
- Từ điều kiện $1 \leq x_1 < x_2 < \dots < x_k \leq n$ ta có tập ứng cử viên cho vị trí thứ nhất (các khả năng của x_1):
 $S_1 = \{1, 2, \dots, n-k+1\}$

Giả sử đã xây dựng được lời giải bộ phận cấp $i-1$ (x_1, \dots, x_{i-1}). Từ điều kiện $x_{i-1} < x_i < \dots < x_k \leq n$, ta có **tập ứng cử viên cho vị trí thứ i** (các khả năng của x_i) là:
 $S_i = \{x_{i-1}+1, x_{i-1}+2, \dots, n-k+i\}$

- **Điều kiện chấp nhận khả năng j :** luôn chấp nhận

Bài toán liệt kê tổ hợp chập k của n

Ví dụ: với $k = 3$, $n = 4$ thì các khả năng của

$$x_1 = \{1, \dots, 4 - 3 + 1\} = \{1, 2\}$$

$$x_2 = \{x_1 + 1, \dots, 4 - 3 + 2\} = \{2, 3\}$$

$$x_3 = \{x_2 + 1, \dots, 4 - 3 + 3\} = \{3, 4\}$$

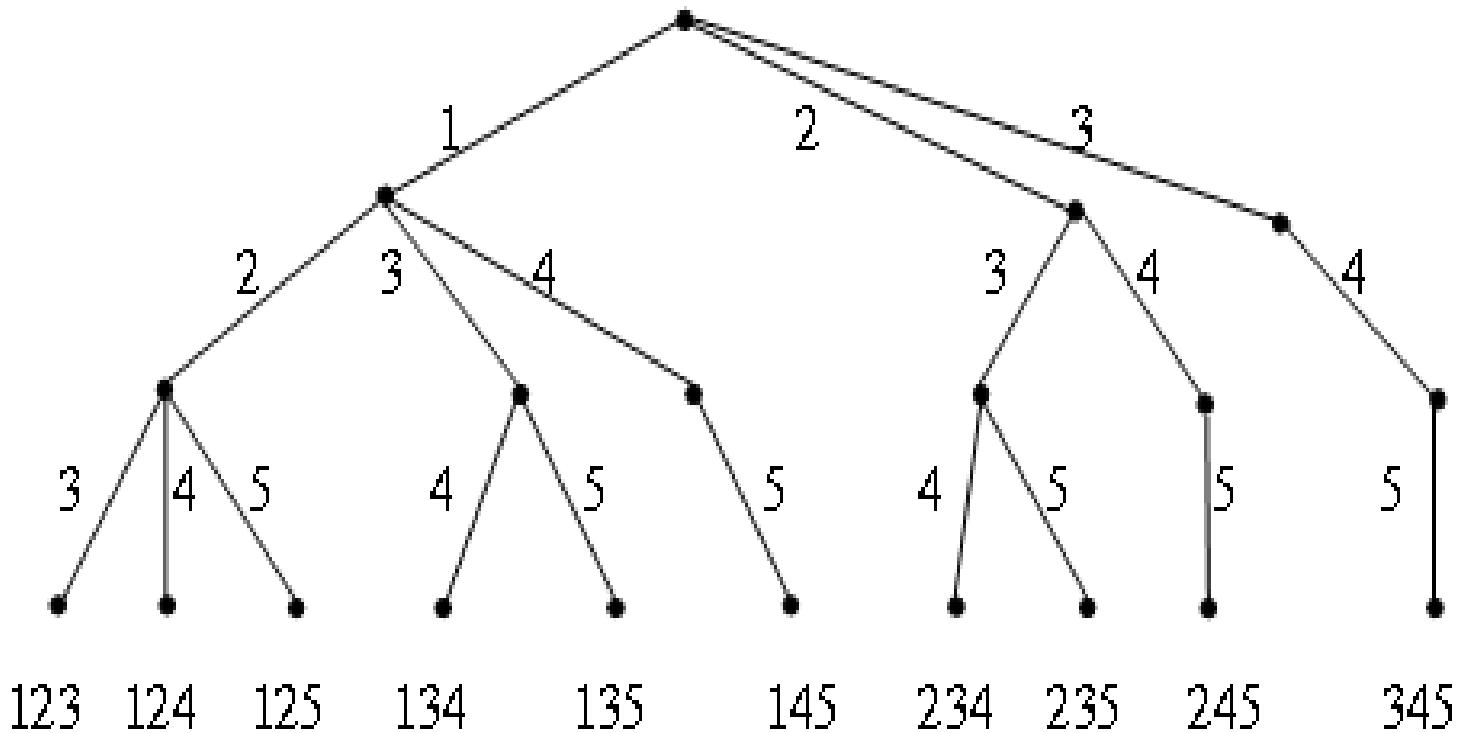
Bài toán liệt kê tổ hợp chập k của n

- Mã giả của thuật toán

```
void backtracking(int i ) { // hàm quay lui  
    for(int j = x[i-1]+1; j<=n-k+i ; j++ ) { // xét các khả năng của i  
        x[i] = j ; // ghi nhận một giá trị của i  
        if (i==k) { // nếu cấu hình đã đủ k phần tử  
            printResult() ; // in một cấu hình  
        } else {  
            backtracking(i+1) ; // quay lui  
        }  
    }  
}
```

Bài toán liệt kê tổ hợp chập k của n

- Cây liệt kê kê lời giải với $k = 3, n = 5$



Bài toán liệt kê hoán vị

- Mô tả bài toán

- Cho một tập hợp gồm n phần tử có giá trị từ 1 đến n . Xuất ra các hoán vị của tập hợp này
- Ví dụ: $n = 3$, các hoán vị là:

1 2 3

1 3 2

2 1 3

2 3 1

3 1 2

3 2 1

Bài toán liệt kê hoán vị

- Dẫn đến bài toán liệt kê tất cả các thành phần tập $\Pi^n = \{(x_1, \dots, x_n) \in N^n: x_i \neq x_j, i \neq j\}$
→ Cấu hình cần tìm dạng: (x_1, x_2, \dots, x_n) với $x_i \neq x_j$
 - Hiển nhiên cấu hình thứ nhất là $(1, 2, \dots, n)$. Giả sử ta có hoán vị bộ phận $(x_1, x_2, \dots, x_{i-1})$, từ điều kiện $x_i \neq x_j$ với mọi $i \neq j$ ta có:
→ Các khả năng của x_i là: $A_i = \{N \setminus \{x_1, x_2, \dots, x_{i-1}\}\}$
 - Điều kiện chấp nhận khả năng j của x_i : khi x_i chưa được sử dụng (x_i khác x_1, \dots, x_{i-1} và $x_i \leq n$)

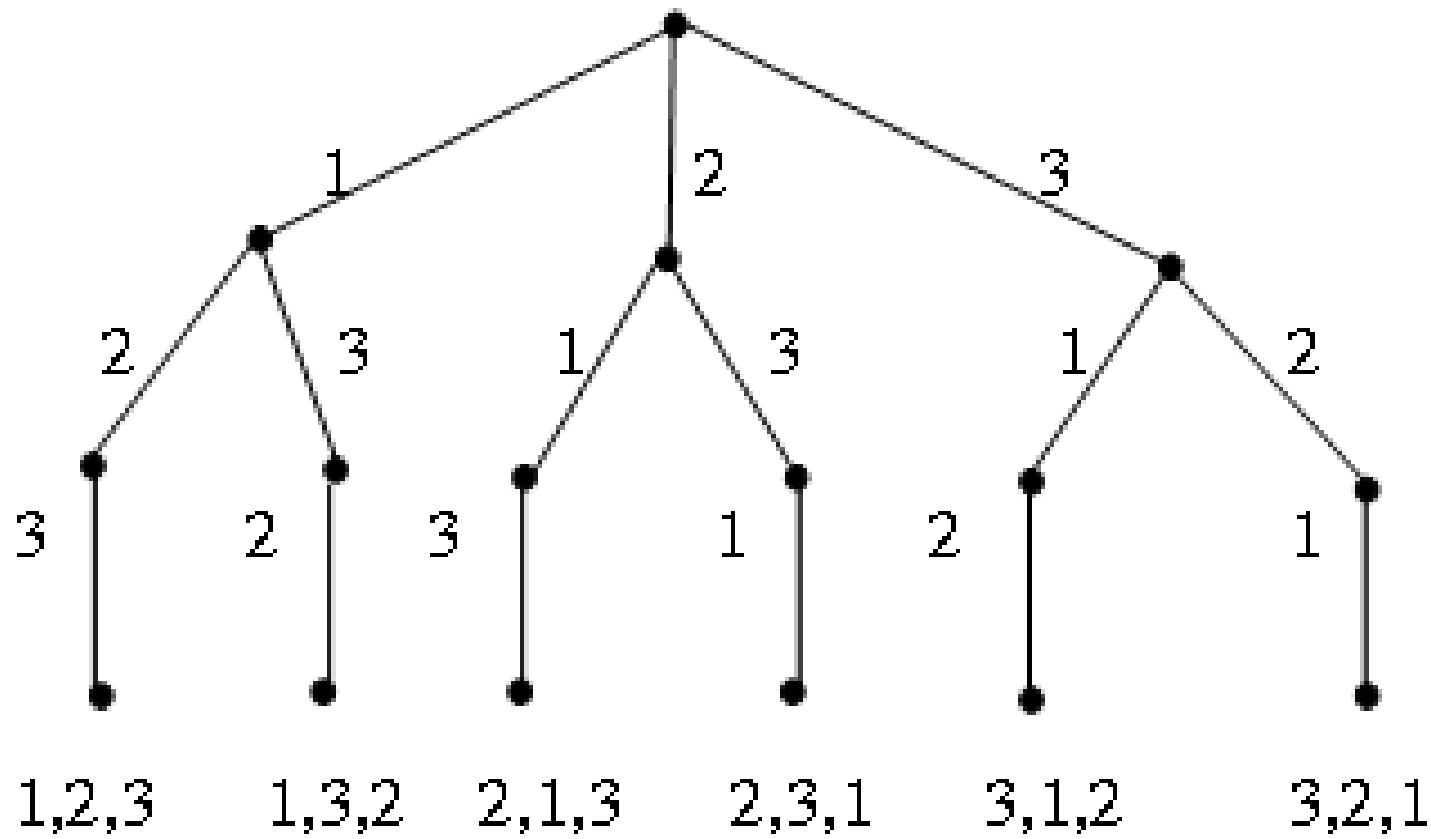
Bài toán liệt kê hoán vị

• Mã giả của thuật toán

```
void hoanvi(int i) {  
    for (int j = 0; j < n; j++) {  
        if (b[j] == 0) {          //Khi số này chưa được chọn  
            x[i] = j + 1;        //Do chỉ số j chạy từ 0  
            b[j] = 1;            //Đánh dấu đã sử dụng số này  
            if (i == n)  
                output();  
            else  
                hoanvi(i + 1);  
            b[j] = 0;  
        }  
    }  
}
```

Bài toán liệt kê hoán vị

- Cây liệt kê hoán vị của $\{1, 2, 3\}$



Bài toán tổng tập con (Subset Sum)

- Mô tả bài toán

- Cho một mảng n phần tử $X[1, \dots, n]$ không âm và một số T . Tồn tại tập con của X có tổng là T ?
- **Ví dụ 1:** $X = \{8, 6, 7, 5, 3, 10, 9\}$ và $T = 12$
→ tồn tại tập con $\{7, 5\}$ của X có tổng là 12
- **Ví dụ 2:** $X = \{3, 34, 4, 12, 5, 2\}$ và $T = 9$
→ tồn tại tập con $\{4, 5\}$ của X có tổng là 9

Bài toán tổng tập con (Subset Sum)

- Ý tưởng:
 - Xét $x \in X$, tồn tại một dãy con có tổng bằng T nếu một trong hai điều kiện sau là đúng
 1. Tồn tại một tập con của $X \setminus \{x\}$ có tổng bằng $T - x$
 2. Tồn tại một tập con của $X \setminus \{x\}$ có tổng bằng T

Bài toán tổng tập con (Subset Sum)

- Cách tiếp cận:
 - **Với hai trường hợp trên ta có:**
 1. Bỏ phần tử cuối cùng và giờ đây
Tổng = T – giá trị của phần tử cuối cùng
và số phần tử = tổng số phần tử – 1
 2. Giữ lại phần tử cuối cùng và giờ đây
Tổng = T và **số phần tử = tổng số phần tử – 1**

Bài toán tổng tập con (Subset Sum)

- Công thức đệ quy cho hàm **isSubsetSum**

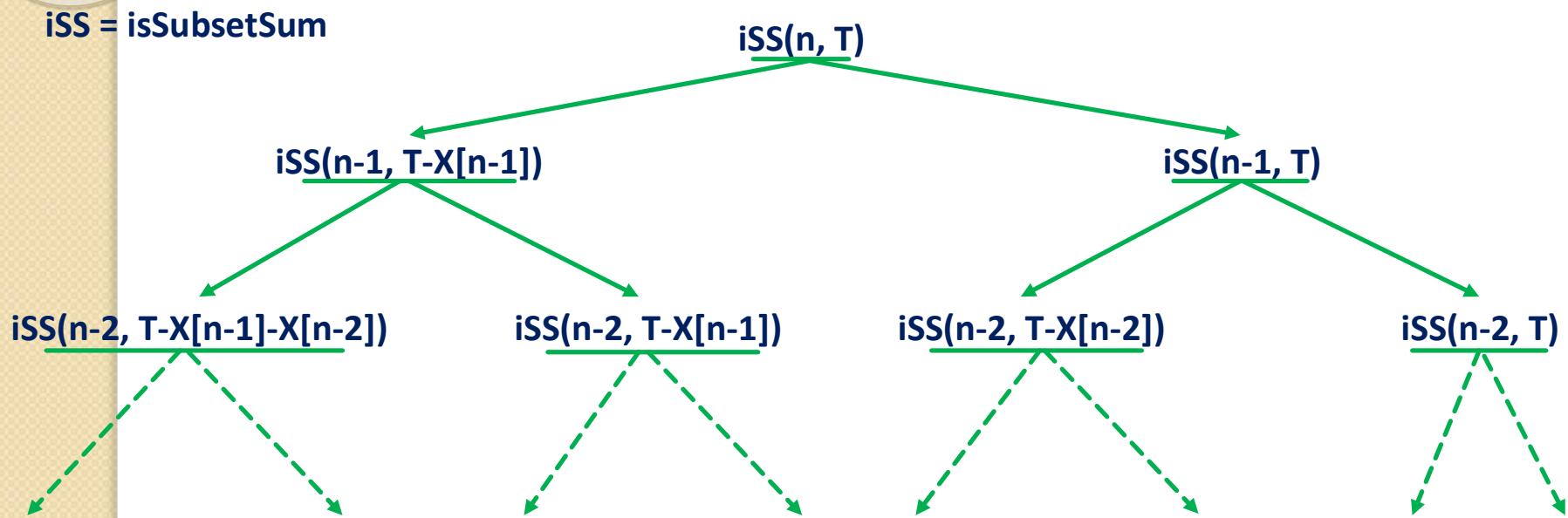
```
isSubsetSum(X, n, T)
= isSubsetSum(X, n - 1, T) ||
  isSubsetSum(X, n - 1, T - X[n - 1])
```

Cơ sở của đệ quy:

$\text{isSubsetSum}(X, n, T) = \text{false}$, if $T > 0$ and $n == 0$
 $\text{isSubsetSum}(X, n, T) = \text{true}$, if $T == 0$

Bài toán tổng tập con (Subset Sum)

- Cách tiếp cận:

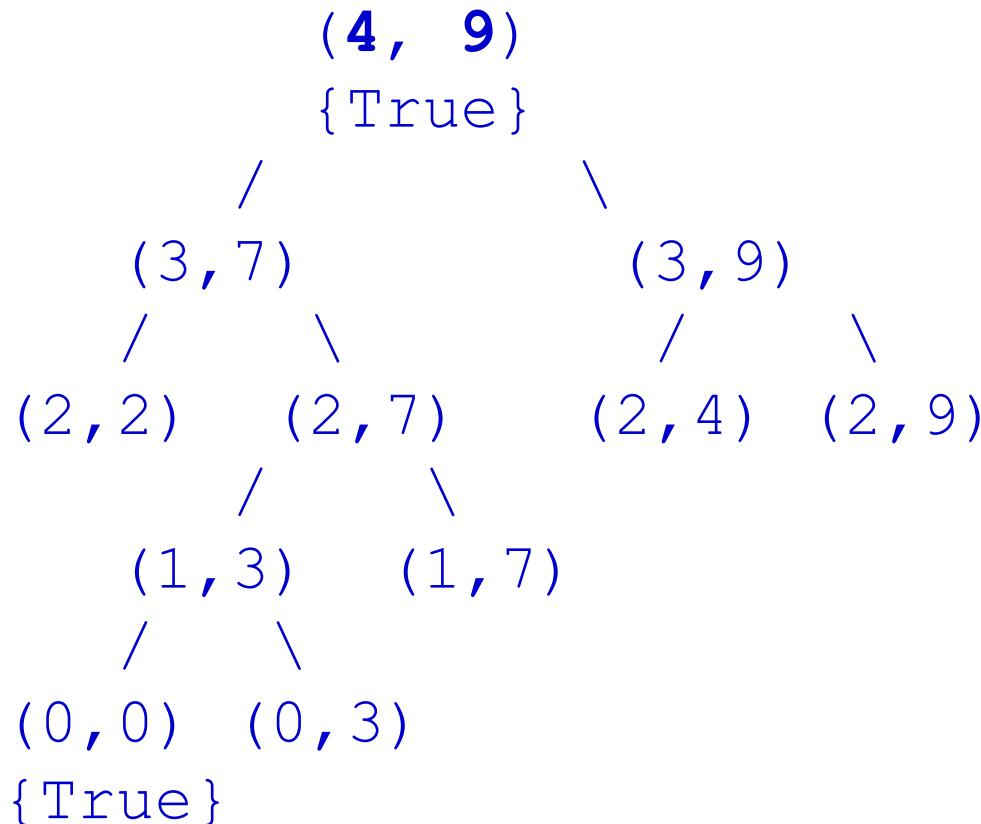


Bài toán tổng tập con (Subset Sum)

- Ví dụ:

$X[] = \{3, 4, 5, 2\}$ và $T = 9$

(x, y) = 'x' là số phần tử còn lại và 'y' là tổng hiện tại



Bài toán tổng tập con (Subset Sum)

- Mã giả của thuật toán

```
bool isSubsetSum(int X[], int n, int T) {  
    if (T == 0) return true;  
    if (n == 0 && T != 0) return false;  
    // Trường hợp phần tử cuối lớn hơn T thì bỏ qua  
    if (X[n - 1] > T)  
        return isSubsetSum(X, n - 1, T);  
    return isSubsetSum(X, n - 1, T) ||  
           isSubsetSum(X, n - 1, T - X[n-1]);  
}
```

Bài toán tổng tập con (Subset Sum)

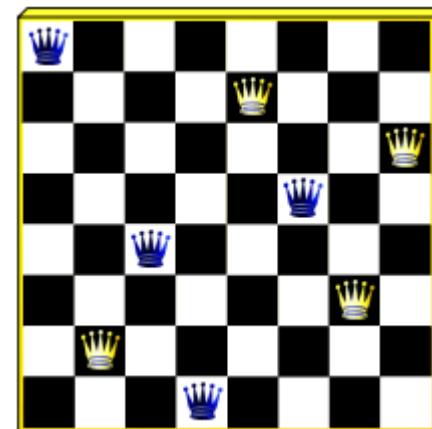
- Độ phức tạp:
 - Ở mỗi bước, gọi đệ quy hai lần trên mảng con kích thước nhỏ hơn một đơn vị
 $\rightarrow T(n) = 2T(n-1) + O(1)$
 - Giải phương trình đệ quy:
$$\begin{aligned} T(n) &= 2[2T(n-2) + 1] + 1 \\ &= 2^2[2T(n-2) + 1] + 3 \\ &= 2^3T(n-3) + 2^2 + 2^1 + 2^0 \\ &= 2^nT(0) + 2^{n-1} + \dots + 2^1 + 2^0 \\ &= 2^nT(0) + 2^n - 1 \text{ với } T(0) = c \end{aligned}$$
$$\rightarrow T(n) = O(2^n)$$

Bài toán n quân hậu

- Mô tả bài toán

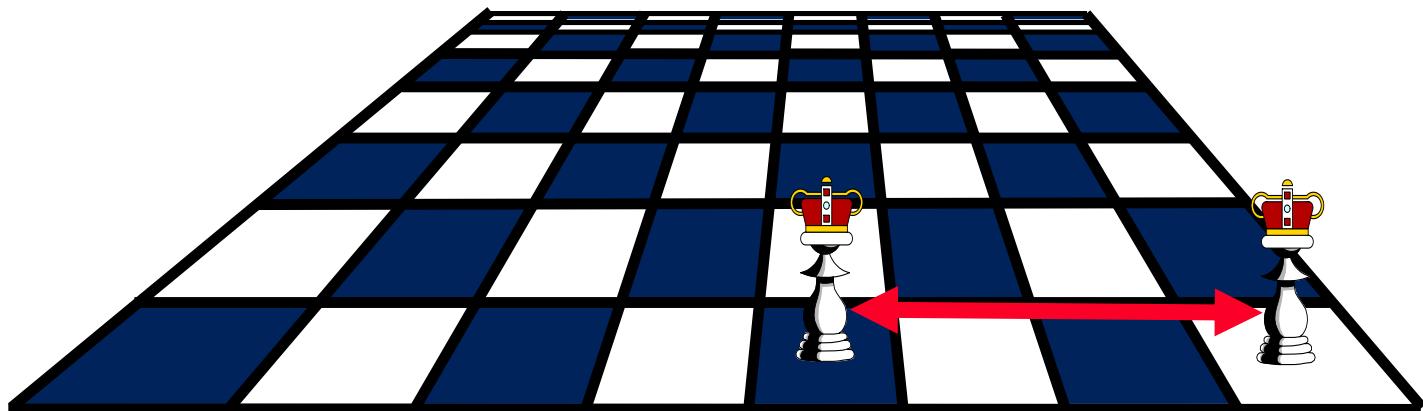
- Hãy tìm cách xếp n quân hậu trên bàn cờ kích thước $n \times n$ sao cho không quân nào ăn được quân nào

Ví dụ: một cách xếp 8 quân hậu trên bàn cờ 8×8



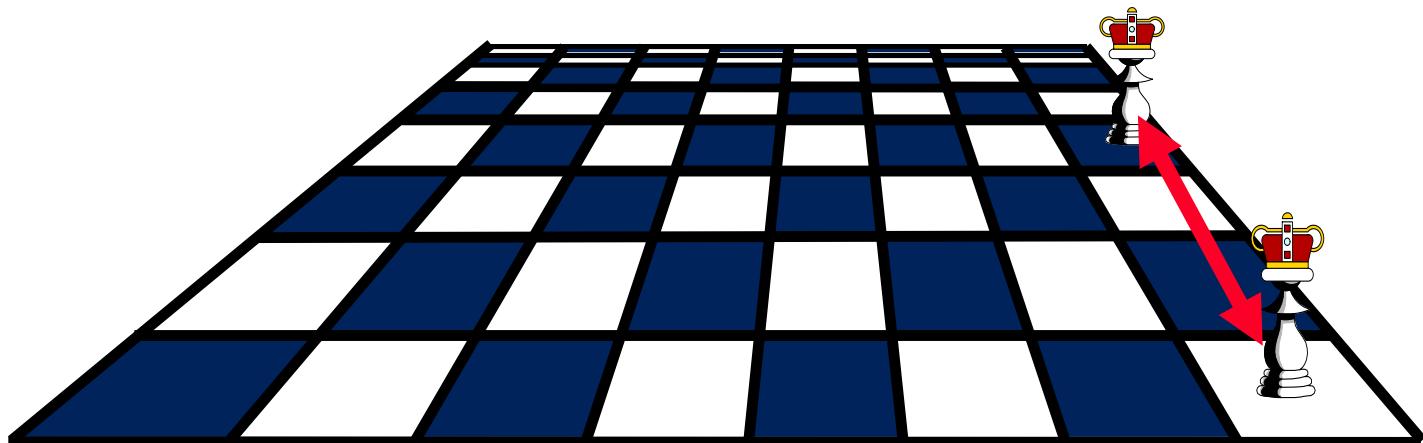
Bài toán n quân hậu

Hai quân hậu bất kỳ không được
xếp trên cùng một hàng



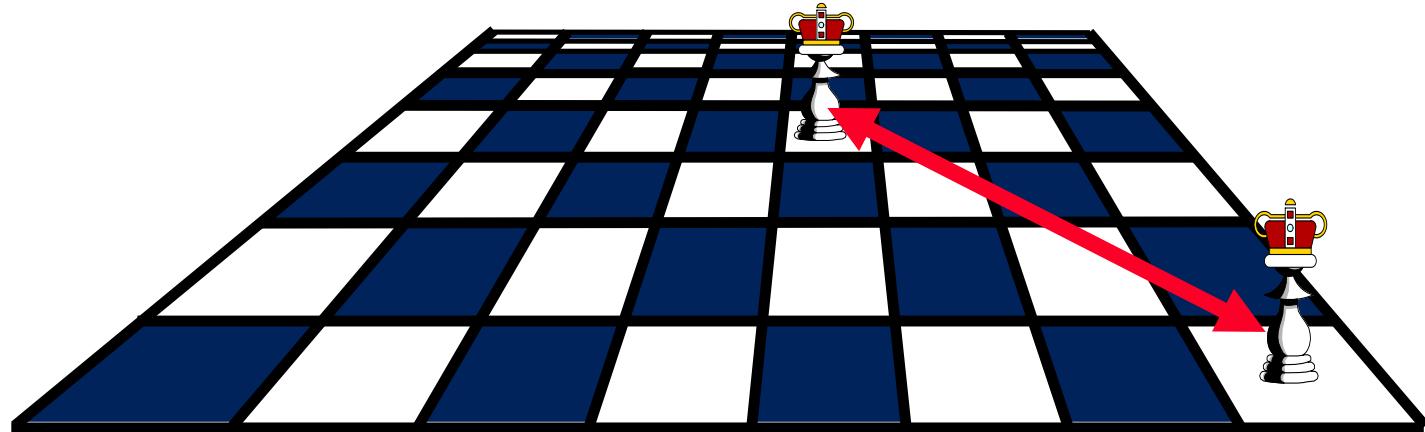
Bài toán n quân hậu

Hai quân hậu bất kỳ không được
xếp trên cùng một cột



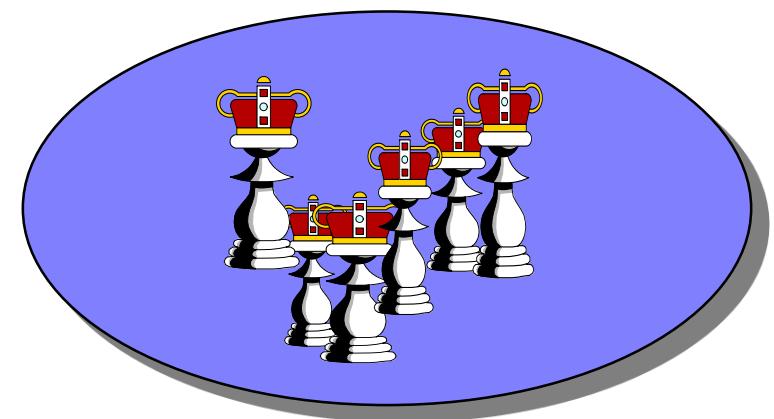
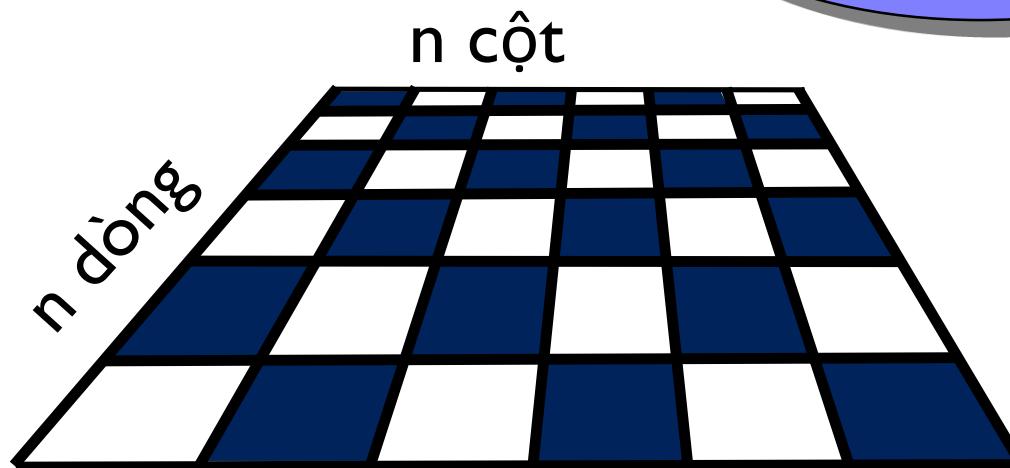
Bài toán n quân hậu

Hai quân hậu bất kỳ không được xếp trên cùng một đường chéo



Bài toán n quân hậu

Kích thước $n \times n$



Bài toán n quân hậu

- Đánh số các cột và dòng của bàn cờ từ 1 đến n . Một cách xếp hậu có thể biểu diễn bởi bộ có n thành phần (x_1, x_2, \dots, x_n) , trong đó x_i là toạ độ cột của con Hậu ở dòng i .
→ **Cấu hình cần tìm dạng:** (x_1, x_2, \dots, x_n)
- Các điều kiện đặt ra đối với bộ (x_1, x_2, \dots, x_n)
 - $x_i \neq x_j$, với mọi $i \neq j$ (hai quân Hậu ở hai dòng i và j không được nằm trên cùng một cột)
 - Hai quân Hậu ở hai vị trí (i, j) và (a, b) ăn được nhau theo đường chéo khi và chỉ khi $i - j = a - b = \text{hằng số}$ (đ/chéo ĐN-TB) hoặc $i + j = a + b = \text{hằng số}$ (đ/chéo ĐB-TN)
→ $|x_i - x_j| \neq |i - j|$, với mọi $i \neq j$ (hai quân Hậu ở hai ô (x_i, i) và (x_j, j) không được nằm trên cùng một đường chéo)

Bài toán n quân hậu

- Như vậy bài toán xếp Hậu dẫn về bài toán liệt kê các phần tử của tập:

$D = \{(x_1, x_2, \dots, x_n) \in N^n$ với

$x_i \neq x_j$ và $|x_i - x_j| \neq |i - j|, i \neq j\}$

Bài toán n quân hậu

- Mã giả của thuật toán

```
int XepHau(int i) {
    int j, k;
    for (j=1; j<=n; j++) {
        legal = true;
        for (k = 1; k < i; k++)
            if ((j==x[k]) || (fabs(j-x[k])==i-k))
                legal = false;
        if (legal) {
            x[i] = j;
            if (i == n) Ghinhau();
            else XepHau(i+1);
        }
    }
}
```

Bài toán n quân hậu

● Độ phức tạp thời gian

- Vòng lặp ngoài gọi đệ quy n lần, mỗi vòng giảm kích thước 1 do đặt được một quân Hậu
- Vòng lặp kiểm tra tính hợp lệ là $O(n)$

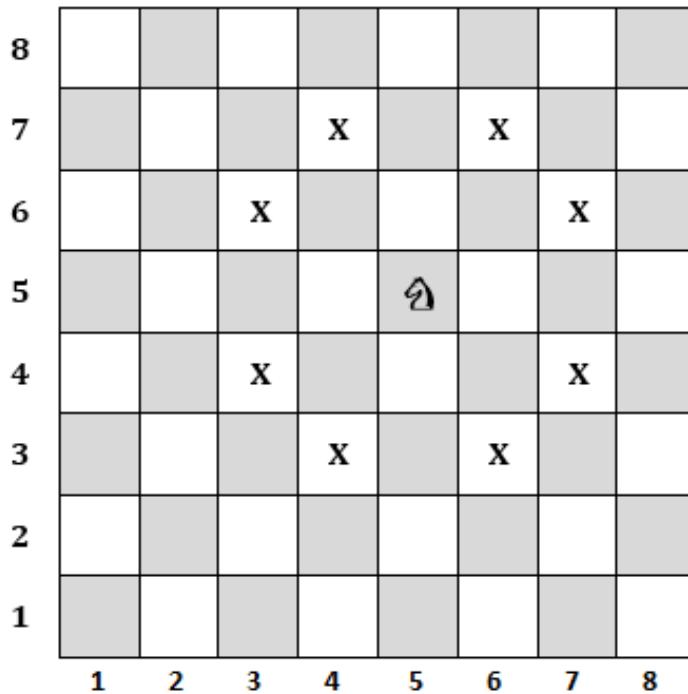
$$\begin{aligned}T(n) &= nT(n - 1) + O(n) \\&= n[nT(n - 2) + n] + n \\&= n^2[nT(n - 3) + n] + n^2 + n \\&= n^3T(n - 3) + n^3 + n^2 + n \\&= n^nT(0) + n^n + \dots + n^2 + n \\&= n^nT(0) + \sum_{i=1}^n n^i \text{ với } T(0) = c\end{aligned}$$

$$\rightarrow T(n) = O(n^n)$$

Bài toán mã đi tuần

- Mô tả bài toán

- Di chuyển một quân mã trên bàn cờ vua trống kích thước (8x8) qua mỗi ô trên bàn cờ đúng một lần
- Nếu vị trí bắt đầu bằng vị trí kết thúc thì đó là một hành trình đóng



Bài toán mã đi tuần

- Áp dụng thuật toán quay lui
 - Cấu hình cần tìm dạng: $0 \leq A[x, y] \leq 64$ thể hiện thứ tự bước đi của quân mã, cặp (x, y) là tọa độ bàn cờ
 - Các khả năng của x, y : $0 \leq x, y \leq 7$
 - Tính hợp lệ của bước đi x, y :
 - Một bước đi: $|\Delta x| + |\Delta y| = 3$, với $x, y > 0$
 - Từ một vị trí bất kỳ có tối đa 8 đường di chuyển. Các bước đi là: $(-2, -1), (-2, 1), (-1, -2), (-1, 2), (1, -2), (1, 2), (2, -1), (2, 1)$
 - $X = \{-2, -2, -1, -1, 1, 1, 2, 2\}$
 - $Y = \{-1, 1, -2, 2, -2, 2, -1, 1\}$
 - Với x, y là vị trí hiện tại của quân mã \rightarrow thử tất cả các bước đi kế tiếp. Một bước đi hợp lệ thỏa: $0 \leq x + X_i \leq n - 1$ và $0 \leq y + Y_i \leq n - 1$ và $A[x + X_i, y + Y_i] = 0$
 \rightarrow Lưu thứ tự bước đi vào ô nhớ mảng $A[x + X_i, y + Y_i]$

Bài toán mã đi tuần

• Áp dụng thuật toán quay lui

```
void diChuyen (int x, int y) {  
    ++dem; //Tăng giá trị bước đi  
    A[x][y] = dem; //Đánh dấu đã đi  
    for (int i = 0; i < 8; i++) {  
        if (dem == n * n) { //Kiểm tra xem mã đã đi hết bàn cờ chưa  
            cout << "Cac buoc di la: \n";  
            xuat();  
            exit(0); //kết thúc chương trình  
        }  
        //Nếu chưa đi hết bàn cờ thì tạo bước đi mới  
        int u = x + X[i]; //tạo một vị trí x mới  
        int v = y + Y[i]; //tạo một vị trí y mới  
        //Nếu hợp lệ thì tiến hành di chuyển  
        if (u >= 0 && u < n && v >= 0 && v < n && A[u][v] == 0)  
            diChuyen(u, v);  
    }  
    //Nếu không tìm được bước đi thì ta phải trả lại các giá trị ban đầu  
    --dem; A[x][y] = 0;  
}
```

Bài toán mã đi tuần

- Áp dụng thuật toán quay lui

```
D:\PhongPD\DHGTVT\Phan_th

Nhap n: 8
Nhap vi tri ban dau.
x: 0
y: 0
Cac buoc di la:
  1 12  9  6  3 14 17 20
10  7  2 13 18 21  4 15
31 28 11  8  5 16 19 22
64 25 32 29 36 23 48 45
33 30 27 24 49 46 37 58
26 63 52 35 40 57 44 47
53 34 61 50 55 42 59 38
62 51 54 41 60 39 56 43
```

Trò chơi Sudoku

- Mô tả bài toán

- Có một hình vuông được chia thành 9×9 ô vuông con. Mỗi ô vuông con có giá trị trong khoảng từ 1 đến 9. Ban đầu hình vuông có một số ô vuông con cho trước (có điền sẵn số) và còn lại là trống. Hãy điền các số từ 1-9 vào các ô còn lại sao cho: hàng ngang là các số khác nhau từ 1 đến 9, hàng dọc là các số khác nhau từ 1 đến 9, và mỗi khối 3×3 chính là các số khác nhau từ 1 đến 9

Trò chơi Sudoku

- Ví dụ

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

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

TỔNG KẾT

1. Tìm hiểu thuật toán quy lui
2. Minh họa một số bài toán minh họa
3. Ưu điểm:
 - ❖ Thủ tất cả các tổ hợp để tìm được một lời giải
 - ❖ Nhiều cài đặt tránh được việc phải thử nhiều trường hợp chưa hoàn chỉnh, nhờ đó giảm thời gian chạy
4. Nhược điểm:
 - ❖ Quá trình tìm kiếm cứ gấp phải bế tắc với cùng một nguyên nhân
 - ❖ Thực hiện các công việc dư thừa (mỗi lần quay lui lại phải đánh giá lại lời giải)
 - ❖ Không sớm phát hiện được các khả năng bị bế tắc trong tương lai

Bài tập

- **Bài 1.** Sửa thuật toán quay lui isSubsetSum ở trên để in ra các dãy số trong X có tổng bằng T.
- **Bài 2.** Sửa thuật toán quay lui của bài toán mã đi tuần ở trên để in ra một chu trình đóng.
- **Bài 3.** Cho một số tự nhiên $N \leq 9$. Giữa các số từ 1 đến N hãy thêm vào các dấu + và - sao cho kết quả thu được bằng 0. Hãy viết chương trình tìm tất cả các khả năng có thể.
- **Bài 4.** Cho một tự nhiên $N \leq 30$. Tìm tất cả các cách phân tích số N thành tổng của các số nguyên dương. Các cách phân tích là hoán vị của nhau thì chỉ tính là một cách.