Đệ quy - Quay lui

Một số thuật toán nâng cao

Ngày 23 tháng 4 năm 2025

Nội dung

1. Đệ quy

2. Phương pháp quay lui

Nội dung

Khái niệm

- **Giải thuật đệ quy**
  - Nếu lời giải của một bài toán P được thực hiện bằng lời giải của bài toán P' có dạng giống P thì đó là một lời giải đệ quy. Giải thuật ứng với lời giải như vậy được gọi là giải thuật đệ quy.
  - P' phải "nhỏ" hơn P và việc giải nó không cần dùng đến P.
  - P' giải được (không cần đệ quy) trong một số trường hợp nào đó (suy biến).
- **Đặc trưng của các bài toán có thể giải bằng đệ quy**
  - Các bài toán phụ thuộc tham số
  - Ứng với một giá trị đặc biệt, bài toán có thể giải trực tiếp.
  - Trong trường hợp tổng quát, bài toán có thể giải được bằng cách giải các bài toán con.

Lược đồ

```
Recursion (p_n)
  if (p_n == p_0)
          <Thực hiện giải thuật trong TH suy biến ;>
  else
     <Lệnh;>
     Recursion(p_{n-1});
     <Lệnh;>
  endif
End
```

Ví dụ

- **Tính n!**

```
Factorial (n)
  if (n ==1)
     Factorial = 1;
  else
     Factorial = n*Factorial(n-1));
  endif
End
```

Nội dung

Ý tưởng

Bạn đứng trước 5 đường hầm, có một túi vàng ở cuối một đường hầm nào đó, nhưng bạn không biết đó là đường hầm nào cả. Bạn sẽ làm thế nào để lấy được túi vàng?

Ý tưởng

- Phương pháp dùng để giải bài toán liệt kê các cấu hình (lời giải). Mỗi cấu hình được xây dựng bằng cách xây dựng từng phần tử, mỗi phần tử được chọn bằng cách thử tất cả các khả năng;
- Tại mỗi bước nếu có một bước lựa chọn **chấp thuận** thì ghi nhận lại lựa chọn này và tiến hành các bước thử tiếp theo;
- Nếu không có lựa chọn nào thích hợp thì làm lại bước trước, xóa bỏ sự ghi nhận;
- Độ dài cấu hình tùy thuộc bài toán
  - Xác định trước: bài toán sinh dãy nhị phân độ dài n, bài toán xếp hậu
  - Không xác định trước: bài toán phân tích số

Mô hình (1)

Giả sử cấu hình cần liệt kê có dạng $(x_1, x_2, ..., x_n)$, $x_i \in D_i$, khi đó thuật toán quay lui thực hiện như sau:

- Xét tất cả các giá trị $x_1 \in D_1$, thử cho x_1 nhận các giá trị có thể. Với mỗi giá trị thử gán cho x_1 ta sẽ:
- Xét tất cả các giá trị của $x_2 \in D_2$, thử cho x_2 nhận lần lượt các giá trị có thể. Với mỗi giá trị thử gán cho x_2 lại xét tiếp các khả năng chọn x_3, tiếp tục quá trình;
- ...
- Xét tất cả các giá trị $x_n \in D_n$, thử cho x_n nhận lần lượt các giá trị có thể. Thông báo cấu hình tìm được $(x_1, x_2, ..., x_n)$;

Mô hình (2)

Tóm lại, để tìm ra lời giải $x = (x_1, x_2, ..., x_n)$, ta phải xây dựng các lời giải x_i. Tại bước thứ i:

- Đã xây dựng được lời giải thành phần $(x_1, x_2, ..., x_{i-1})$.
- Xây dựng x_i bằng cách thử tất cả các khả năng mà $x_i \in D_i$:
  - Nếu $j \in D_i$ mà phù hợp cho x_i, thì $x_i = j$. Ghi nhận trạng thái mới của bài toán. Nếu $i = n$, ta có được một lời giải, ngược lại thì tiến hành bước $i + 1$ để xác định x_{i+1};
  - Nếu không có khả năng nào chấp nhận cho x_i thì lùi lại bước trước (bước $i - 1$) để xác định lại thành phần x_{i-1}

Mô hình (3)

Để giải một bài toán theo phương pháp quay lui ta cần biết:

Mô hình (3)

Để giải một bài toán theo phương pháp quay lui ta cần biết:

- Dạng của cấu hình cần tìm

Mô hình (3)

Để giải một bài toán theo phương pháp quay lui ta cần biết:

- Dạng của cấu hình cần tìm
- Tập các khả năng của từng phần tử

Mô hình (3)

Để giải một bài toán theo phương pháp quay lui ta cần biết:

- Dạng của cấu hình cần tìm
- Tập các khả năng của từng phần tử
- Điều kiện chấp nhận các khả năng

Lược đồ

```
backtracking (i)
  for (j thuộc tập khả năng D_i) {
      if (j chấp nhận được ){
         Xác định x_i theo j;
         Ghi nhận trạng thái mới (nếu có);
         if (i==n)
             Cập nhật lời giải;
         else
             backtracking (i+1);
          Bỏ ghi nhận (nếu có);
      }
  }
```

Đánh giá độ phức tạp

- Công thức truy hồi

$$T(n) = \begin{cases} 1 & n \leq 1 \\ dT(n-1) & n > 1 \end{cases} \quad (1)$$

d: max (hoặc giá trị trung bình) lực lượng của tập khả năng của các thành phần nghiệm x_i
Giải công thức truy hồi $T(n) = O(d^n)$

Một số bài toán điển hình

Bài toán liệt kê dãy nhị phân độ dài n

Bài toán

Cho một số nguyên dương n, hãy liệt kê tất cả các dãy nhị phân độ dài n.

Một số bài toán điển hình

Bài toán liệt kê dãy nhị phân độ dài n

Bài toán

Cho một số nguyên dương n, hãy liệt kê tất cả các dãy nhị phân độ dài n.

- Biểu diễn dãy nhị phân độ dài n dưới dạng $x = (x_1, x_2, ..., x_n)$
- Các khả năng của x_i : $x_i \in \{0, 1\}$

Một số bài toán điển hình

Bài toán liệt kê dãy nhị phân độ dài n

```
generate (i)
  for (j = 0; j<=1; j++) {
      x_i = j ;
      if (i==n)
          printSolution();
      else
          generate (i+1);
   }
```

Một số bài toán điển hình

Bài toán xếp hậu

Bài toán

Cho một bàn cờ kích thước $n \times n$. Một quân hậu trên bàn cờ có thể ăn được các quân khác tại các ô cùng hàng, cùng cột, hoặc cùng đường chéo. Hãy tìm cách xếp n quân hậu trên bàn cờ sao cho không quân hậu nào ăn quân hậu nào.

	1	2	3	4	5	6	7	8
1				q_1				
2						q_2		
3								q_3
4		q_4						
5							q_5	
6	q_6							
7			q_7					
8					q_8			

Một số bài toán điển hình

Bài toán xếp hậu

Phân tích

Trong một ma trận vuông

- Các phần tử nằm trên cùng hàng có chỉ số hàng bằng nhau
- Các phần tử nằm trên cùng cột có chỉ số cột bằng nhau
- Đường chéo thuận (song song với đường chéo chính). Các ô (x, y) bất kì nằm trên đường chéo thuận có tính chất $x - y = const$.
- Đường chéo nghịch (vuông góc với đường chéo chính). Các ô (x, y) bất kì nằm trên đường chéo thuận có tính chất $x + y = const$.
- n quân hậu, mỗi quân chỉ được đặt trên một hàng (vì hậu ăn ngang)

Một số bài toán điển hình

Bài toán xếp hậu

Phân tích

- Dạng nghiệm $x = (x_1, x_2, ..., x_n)$, x_i là quân hậu ở hàng thứ $i; i = \overline{1, n}$.
- $x_i = j$: quân hậu ở hàng i được đặt ở cột j, $j = \overline{1, n}$.
- Quân hậu được đặt ở ô (i, j) nếu ô này chưa bị khống chế
- Quân hậu đặt ở ô (i, j) sẽ khống chế
    - Toàn bộ hàng i
    - Toàn bộ cột j
    - Hai đường chéo

Một số bài toán điển hình

Bài toán xếp hậu

Phân tích

Sử dụng 3 mảng để đánh dấu:

- Mảng $a = (a_1, a_2, ..., a_n)$, trong đó:

$$a_j = \begin{cases} \mathit{true} & \text{nếu cột } j \text{ chưa bị khống chế} \\ \mathit{false} & \text{nếu cột } j \text{ đã bị khống chế} \end{cases} \tag{2}$$

Một số bài toán điển hình

Bài toán xếp hậu

Phân tích

Sử dụng 3 mảng để đánh dấu:

- Mảng $a = (a_1, a_2, ..., a_n)$, trong đó:

$$a_j = \begin{cases} \textit{true} & \text{nếu cột } j \text{ chưa bị khống chế} \\ \textit{false} & \text{nếu cột } j \text{ đã bị khống chế} \end{cases} \quad (2)$$

- Mảng $b = (b_1, b_2, ..., b_{2n-1})$, trong đó:

$$b_{i-j+n} = \begin{cases} \textit{true} & \text{nếu đường chéo thuận qua ô } (i,j) \text{ chưa bị khống chế} \\ \textit{false} & \text{nếu đường chéo thuận qua ô } (i,j) \text{ đã bị khống chế} \end{cases} \quad (3)$$

- Mảng $c = (c_1, c_2, ..., c_{2n-1})$, trong đó:

$$c_{i+j-1} = \begin{cases} \textit{true} & \text{nếu đường chéo nghịch qua ô } (i,j) \text{ chưa bị khống chế} \\ \textit{false} & \text{nếu đường chéo nghịch qua ô } (i,j) \text{ đã bị khống chế} \end{cases} \quad (4)$$

Một số bài toán điển hình

Bài toán xếp hậu

- Thử đặt quân hậu 1 vào một cột, với mỗi cách đặt như vậy, xét tất cả các cách đặt quân hậu 2 không bị quân hậu 1 ăn, chọn thử 1 cách đặt,...Mỗi cách đặt đến quân hậu n cho ta 1 nghiệm.
- Chọn cột j để đặt quân hậu thứ i khi:
  - $a_j = \textbf{true}$; $b_{i-j+n} = \textbf{true}$; $c_{i+j-1} = \textbf{true}$;
- Khi thử đặt quân hậu thứ i vào cột j, nếu đó là quân hậu cuối cùng thì ta có 1 nghiệm, ngược lại:
  - **Trước khi gọi đệ quy** tìm đặt quân hậu $i + 1$:

Một số bài toán điển hình

Bài toán xếp hậu

- Thử đặt quân hậu 1 vào một cột, với mỗi cách đặt như vậy, xét tất cả các cách đặt quân hậu 2 không bị quân hậu 1 ăn, chọn thử 1 cách đặt,...Mỗi cách đặt đến quân hậu n cho ta 1 nghiệm.
- Chọn cột j để đặt quân hậu thứ i khi:
  - a_j = **true**; b_{i-j+n} = **true**; c_{i+j-1} = **true**;
- Khi thử đặt quân hậu thứ i vào cột j, nếu đó là quân hậu cuối cùng thì ta có 1 nghiệm, ngược lại:
  - **Trước khi gọi đệ quy** tìm đặt quân hậu $i+1$: đánh dấu cột và 2 đường chéo bị quân hậu vừa thử đặt khống chế.
  - **Sau khi gọi đệ quy** tìm đặt quân hậu $i+1$:

Một số bài toán điển hình

Bài toán xếp hậu

- Thử đặt quân hậu 1 vào một cột, với mỗi cách đặt như vậy, xét tất cả các cách đặt quân hậu 2 không bị quân hậu 1 ăn, chọn thử 1 cách đặt,...Mỗi cách đặt đến quân hậu n cho ta 1 nghiệm.
- Chọn cột j để đặt quân hậu thứ i khi:
  - a_j = **true**; b_{i-j+n} = **true**; c_{i+j-1} = **true**;
- Khi thử đặt quân hậu thứ i vào cột j, nếu đó là quân hậu cuối cùng thì ta có 1 nghiệm, ngược lại:
  - **Trước khi gọi đệ quy** tìm đặt quân hậu $i+1$: đánh dấu cột và 2 đường chéo bị quân hậu vừa thử đặt khống chế.
  - **Sau khi gọi đệ quy** tìm đặt quân hậu $i+1$: bỏ đánh dấu cột và 2 đường chéo bị quân hậu vừa thử đặt khống chế.

Một số bài toán điển hình

Bài toán xếp hậu

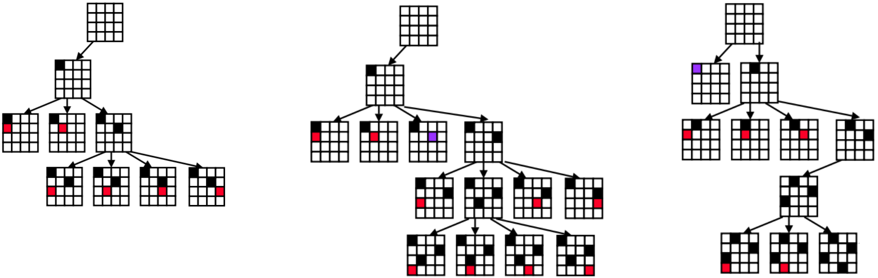
Mã giả

```
sol_nQueen (i)
  for (j=1;j<=n;j++){
     if(a[j]  &&  b[i-j+n]  && c[i+j-1]){
        x[i] = j;
        a[j] = false; b[i-j+n] = false; c[i+j-1] = false;
        if (i==n)
           printSolution();
        else
           sol_nQueen(i+1);
        a[j] = true; b[i-j+n] = true; c[i+j-1] = true;
     }
  }
```

Một số bài toán điển hình

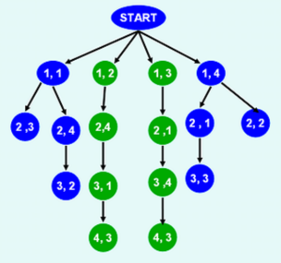
Bài toán xếp hậu - Xếp 4 hậu lên bàn cờ 4×4

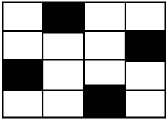
Ví dụ

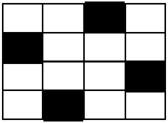


Một số bài toán điển hình

Bài toán xếp hậu - Xếp 4 hậu lên bàn cờ 4×4







Một số bài toán điển hình

Bài toán mã đi tuần

Bài toán

Cho bàn cờ vua kích thước $n \times n$. Một con mã được phép đi theo luật cờ vua, đầu tiên con mã được đặt ở ô (x_0, y_0). Hãy chỉ ra hành trình (nếu có) của con mã để con mã đi qua các ô của bàn cờ vua, mỗi ô đi qua đúng một lần.

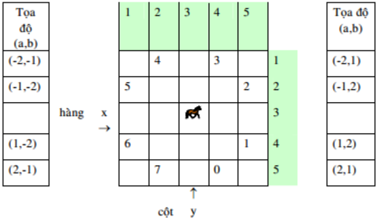
Một số bài toán điển hình

Bài toán mã đi tuần - Phân tích

- Biểu diễn bàn cờ bằng một ma trận vuông cấp n: $grid[n][n]$. Quy ước:

$$grid[x][y] = \begin{cases} 0 & \text{ô } (x, y) \text{ mã chưa đi qua} \\ i & \text{ô } (x, y) \text{ mã đã đi qua ở bước thứ } i \end{cases} \quad (5)$$

- Với cặp tọa độ (x, y) ban đầu đầu, có 8 ô (u, v) mà con mã có thể đi đến. Ta dùng 2 mảng $xMove$, $yMove$ để lưu trữ sự khác biệt về tọa độ.



Một số bài toán điển hình

Bài toán mã đi tuần - Phân tích

- Mã từ ô (x, y) có thể chuyển sang ô (u, v) nếu:
  - Ô (u, v) thuộc bàn cờ $(1 \leq u, v \leq n)$
  - Ô (u, v) chưa được đi qua, có nghĩa là $grid[u][v] = 0$.
- Để ghi nhận bước đi hợp lệ ở bước thứ i ta gán $grid[u][v] = i$, để hủy một nước đi ta gán $grid[u][v] = 0$;
- Ma trận kết quả nghiệm. Nếu có $grid[x][y] = 0$ thì đó không phải là lời giải của bài toán, ngược lại chứa đường đi của con mã.

	n=5	x=1	y=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

Hình 1: Một hành trình của con mã với kích thước bàn cờ là 5×5, ô bắt đầ (1,1)

Một số bài toán điển hình

Bài toán mã đi tuần - Mã giả

```
int xMove[8] = {2,1,-1,-2,-2,-1,1,2}
int yMove[8] = {1,2,2,1,-1,-2,-2,-1}
sol_knightTour (i, x, y)
   for (k=1;k<=8;k++){
      u = x + xMove[k];
      v = y + yMove[k];
      if (1 <=u, v <=n && grid[u][v] ==0){
         grid[u][v] = i;
         if (i < n*n)
            sol_knightTour (i+1,u,v);
         else
            printSolution();
         grid[u][v] = 0;
      }
   }
```

Một số bài toán điển hình

Bài toán mã đi tuần - Cài đặt

- Thủ tục đệ quy được khởi động bằng một lệnh gọi các tọa độ khởi đầu (x_0, y_0) là tham số. Ô xuất phát có giá trị 1, còn các ô còn lại đánh dấu còn trống.
    - $grid[x_0][y_0] = 1;$
    - sol_knightTour (2, x_0, y_0).