ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN



**Bản báo cáo**

backtracking algorithm

Môn: Phân tích thiết kế thuật toán

Mã lớp: CS112.L12.KHCL

GVHD: Nguyễn Thanh Sơn

Nguyễn Quang Hiếu

18520748

Nguyễn Thanh Điền

18520600

Lê Đình Trọng Nghĩa

17520798

1. **TỔNG QUAN VỀ BACKTRACKING**
2. Giới thiệu

Backtracking hay quay lui là một phương pháp chung để giải quyết các bài toán liên quan đến liệt kê cấu trúc, các cấu trúc này là tổ hợp của từng trường hợp khả dĩ của bài toán. Người đầu tiên đề ra thuật ngữ này (backtrack) là nhà toán học người Mỹ D. H. Lehmer vào những năm 1950.

1. Ý tưởng

Về bản chất, tư tưởng của phương pháp là thử từng khả năng cho đến khi tìm thấy lời giải đúng. Đó là một quá trình *tìm kiếm theo độ sâu* trong một tập hợp các lời giải. Trong quá trình tìm kiếm, nếu ta gặp một hướng lựa chọn không thỏa mãn, ta quay lui về điểm lựa chọn nơi có các hướng khác và thử hướng lựa chọn tiếp theo. Khi đã thử hết các lựa chọn xuất phát từ điểm lựa chọn đó, ta quay lại điểm lựa chọn trước đó và thử hướng lựa chọn tiếp theo tại đó. Quá trình tìm kiếm thất bại khi không còn điểm lựa chọn nào nữa.

Quy trình đó thường được cài đặt bằng một hàm đệ quy(hoặc có thể là vòng lặp) mà trong đó mỗi thể hiện của hàm lấy thêm một biến và lần lượt gán tất cả các giá trị có thể cho biến đó, với mỗi lần gán trị lại gọi chuỗi đệ quy tiếp theo để thử các biến tiếp theo. Chiến lược quay lui tương tự với tìm kiếm theo độ sâu nhưng sử dụng ít không gian bộ nhớ hơn, nó chỉ lưu giữ trạng thái của một lời giải hiện tại và cập nhật nó.

1. Ưu nhược điểm

* Ưu điểm: tự động thử tất cả các tổ hợp để tìm được lời giải, giảm thiểu việc phải thử các tổ hợp đó bằng tay. Đồng thời thế mạnh của phương pháp này là nhiều cài đặt tránh được việc phải thử nhiều trường hợp chưa hoàn chỉnh, nhờ đó giảm thời gian chạy.
* Nhược điểm: Trong trường hợp xấu nhất độ phức tạp của quay lui vẫn là cấp số mũ. Vì nó mắc phải các nhược điểm sau:
  + Rơi vào tình trạng "thrashing": qúa trình tìm kiếm cứ gặp phải bế tắc với cùng một nguyên nhân.
  + Thực hiện các công việc dư thừa: Mỗi lần chúng ta quay lui, chúng ta cần phải đánh giá lại lời giải trong khi đôi lúc điều đó không cần thiết.
  + Không sớm phát hiện được các khả năng bị bế tắc trong tương lai. Quay lui chuẩn, không có cơ chế nhìn về tương lai để nhận biết đc nhánh tìm kiếm sẽ đi vào bế tắc.

1. **MỘT SỐ VÍ DỤ SỬ DỤNG BACKTRACKING**
2. N – Queen

Bài toán N quân hậu là bài toán xếp N quân hậu trên bàn cờ N × N sao cho không có hai quân hậu tấn công nhau.

Input: Số nguyên N

Output: Ma trận NxN có vị trí của các quân hậu

Tóm tắt giải thuật:

* Đầu tiên đặt quân hậu thứ 1 vào dòng thứ nhất của bàn cờ tại cột 1.
* Sau đó, tiếp tục đặt quân hậu thứ 2 vào dòng thứ 2 của bàn cờ sao cho con hậu thứ 2 không bị con hậu trước ăn mất.
* Cứ tiếp tục quá trình cho đến khi:
  + Trường hợp 1: đạt đến dòng n và đặt thành công con hậu thứ n. => in ra kết quả.
  + Trường hợp 2: đạt đến dòng thứ k nhưng không còn chỗ đặt con hậu thứ k:
    - quay lại dòng thứ k-1, tìm bên phải con hậu thứ k-1 một ô hợp lệ gần nhất, đặt con hậu k-1 vào đó.
    - sau đó tiếp tục tìm kiếm ô hợp lệ để đặt cho con hậu thứ k ở dòng thứ k.
    - nếu không tìm thấy ô hợp lệ cho k-1, tiếp tục quay lui về con thứ k-2 và lặp lại tương tự.

Phân tích độ phức tạp:

* **Độ phức tạp thời gian**: O(n!). Lời giải của bài toán N Queen bằng cách sử dụng kiểm tra backtracking cho tất cả các sắp xếp có thể có của N Queen trên bàn cờ. Và sau đó kiểm tra tính hợp lệ của giải pháp. Bây giờ số cách sắp xếp có thể có của N quân hậu vào N cột trên bàn cờ N x N là N! (bạn đang bỏ qua hàng hoặc cột, đã có một quân hậu được đặt.).
* **Độ phức tạp không gian**: O(1). Ma trận không thay đổi kích thước trong quá trình thực hiện.

1. Rat in maze

Maze được cho dưới dạng ma trận nhị phân N \* N trong đó nguồn là Maze [0] [0] và đích là là Maze [N- 1] [N-1] . Một con chuột bắt đầu từ nguồn và phải đến đích. Con chuột chỉ có thể di chuyển theo hai hướng: xuống và rẽ phải. Trong ma trận Maze, 0 có nghĩa là ngõ cụt và 1 có nghĩa là khối có thể được sử dụng trong đường dẫn từ nguồn đến đích.

Input: Ma trận NxN.

Output: Ma trận NxN với các phần tử có giá trị 1 tạo thành đường đi.

Tóm tắt giải thuật:

* Tạo một ma trận giải pháp, ban đầu được điền bằng 0.
* Tạo một phép đệ quy, lấy ma trận ban đầu, ma trận đầu ra và vị trí của chuột (i, j).
* Nếu vị trí nằm ngoài ma trận hoặc vị trí không hợp lệ thì quay trở lại.
* Đánh dấu đầu ra vị trí [i] [j] là 1 và kiểm tra xem vị trí hiện tại có phải là đích hay không. Nếu đến đích, in ma trận đầu ra và trả về.
* Gọi đệ quy cho vị trí (i + 1, j) và (i, j + 1).
* Bỏ đánh dấu vị trí (i, j), tức là đầu ra [i] [j] = 0.

Phân tích độ phức tạp:

* **Độ phức tạp thời gian**: O(2^(n^2)). Đệ quy có thể chạy hướng trên 2^(n^2) lần.
* **Độ phức tạp không gian**: O(n^2). Ma trận đầu ra là bắt buộc nên cần thêm một khoảng trống có kích thước n \* n.

1. **TÀI LIỆU THAM KHẢO**

[1] A Multipurpose Backtracking Algorithm, H.A. Priestley, M.P. Ward, January 17, 2003

[2] <https://www.geeksforgeeks.org/n-queen-problem-backtracking-3/>

[3] https://www.geeksforgeeks.org/rat-in-a-maze-backtracking-2/