Report AI Lab6

Nguyễn Quốc Bảo

18110053

1 Mở đầu

Bài toán tám nữ hoàng là một ví dụ về "bài toán n nữ hoàng" tổng quát hơn - bài toán tìm số nghiệm hợp lệ để đặt n nữ hoàng trên một bảng có kích thước n x n. Lưu ý rằng chữ n đại diện cho cùng một số trong cả hai trường hợp - số quân hậu phải bằng số hàng / cột trên bàn cờ.

Mục tiêu là tính toán có bao nhiêu cách khác nhau để đặt 8 quân hậu trên bàn cờ 8 x 8, sao cho không ai trong số họ đang đe dọa lẫn nhau (nghĩa là - không có quân nào trong số quân hậu đặt cùng một hàng, cột hoặc đường chéo, sao cho 1 quân hậu không thể bắt được quân hậu khác trong một lần di chuyển).

2 Thuật Toán

1. Bắt đầu ở cột ngoài cùng bên trái.

2. Nếu tất cả các quân hậu đã được đặt trả về giá trị true và xuất vị trí của từng quân hậu.

3. Thử tất cả các hàng trong cột hiện tại. Làm theo cho mỗi hàng đã thử.

(a) Nếu quân hậu có thể được đặt an toàn vào hàng này thì hãy đánh dấu [hàng, cột] này như một phần của giải pháp và kiểm tra đệ quy xem việc đặt quân hậu ở đây có dẫn đến giải pháp hay không.

(b) Nếu đặt quân hậu vào [hàng, cột] dẫn đến một nghiệm thì trả về true.

(c) Nếu việc đặt quân hậu không dẫn đến giải pháp thì hãy bỏ đánh dấu [hàng, cột] này (lùi lại) và chuyển sang bước (a) để thử các hàng khác.

4. Nếu tất cả các hàng đã được thử và không có gì hoạt động, hãy trả về false để quay ngược lại trạng thái trước.

3 Code

Hướng dẫn cài đặt

```
def solve(boardsize, board=[], solutions_found=0):
    ''' A recursive utility function to solve N
    Queen problem.
    '''
    # If all queens are placed
    if len(board) == boardsize:
        solutions_found += 1
        display(solutions_found, board)
    else:
        # Consider this column and try placing this queen in all rows one by one
        for q in [col for col in range(1,boardsize+1) if col not in board]:
            if Issafe(q,board):
                solutions_found = solve(boardsize, board + [q], solutions_found)
    # If queen can not be place in any row in this column col then return board before
    return solutions_found
```

Một giải pháp được xây dựng bắt đầu bằng một bảng(board) trống [] bằng cách đặt một Q ở hàng tiếp theo ở vị trí cột mà nó không thể lấy được. Các vị trí có thể có là các cột chưa được lấy trước đó (đây là vòng lặp for trong hàm giải quyết). Đối với mỗi vị trí cột có thể có này, hàm Issafe sẽ kiểm tra xem vị trí đó có an toàn không để bị lấy theo đường chéo bởi các Q đã có trên bảng. Nếu vị trí an toàn, bảng giải được mở rộng thêm một hàng khác và giải pháp lặp lại cho đến khi bảng được lấp đầy (len (board) == boardize), lúc này số giải

được tăng lên và bảng được hiển thị.

```
def Issafe(q, board):
    ''' checking if placing a queen in a space would place it
    in the same row, column, or diagonal as any other queens
    '''
    x = len(board)
    for col in board:
        if col in [q+x,q-x]: return False
        x -= 1
    return True
```

Hàm Issafe, kiểm tra xem việc đặt một quân hậu trong một khoảng trống sẽ đặt nó vào cùng một hàng, cột hoặc đường chéo như bất kỳ quân hậu nào khác.

```
def display(solution_number, board):
    ''' Showing results through the position of placing Queen for each column'''
    row = '|   ' * len(board) + '|'
    hr  = '+---' * len(board) + '+'
    for col in board:
        print(hr)
        print(row[:col*4-3],'Q',row[col*4:])
    print(f'{hr}\n{board}\nSolution - {solution_number}\n')
```

Hàm display, hiển thị kết quả thông qua vị trí đặt Nữ hoàng cho mỗi cột.

```
if __name__ == '__main__':
    solutions = solve(8)
    print(f'{solutions} solutions found')
```