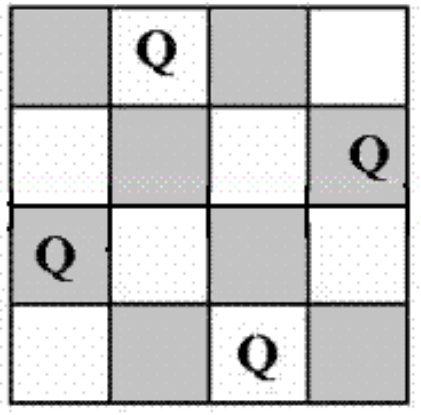
Backtracking | Set 3 (N Queen Problem)

We have discussed Knight's tour and Rat in a Maze problems in Set 1 and Set 2 respectively. Let us discuss N Queen as another example problem that can be solved using Backtracking.

The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. For example, following is a solution for 4 Queen problem.

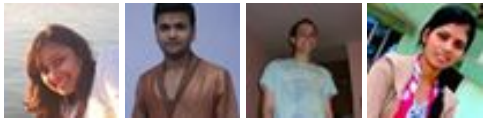


The expected output is a binary matrix which has 1s for the blocks where queens are placed. For example following is the output matrix for above 4 queen solution.

```
{ 0,  1,  0,  0}
{ 0,  0,  0,  1}
{ 1,  0,  0,  0}
{ 0,  0,  1,  0}
```

Naive Algorithm

Generate all possible configurations of queens on board and print a configuration that satisfies



the given constraints.

```
while there are untried conflagrations
{
   generate the next configuration
   if queens don't attack in this configuration then
   {
      print this configuration;
   }
}
```

Backtracking Algorithm

The idea is to place queens one by one in different columns, starting from the leftmost column. When we place a queen in a column, we check for clashes with already placed queens. In the current column, if we find a row for which there is no clash, we mark this row and column as part of the solution. If we do not find such a row due to clashes then we backtrack and return false.

```
1) Start in the leftmost column
2) If all queens are placed
    return true
3) Try all rows in the current column.  Do following for every tried row.
    a) If the queen can be placed safely in this row then mark this [row,
        column] as part of the solution and recursively check if placing
        queen here leads to a solution.
    b) If placing queen in [row, column] leads to a solution then return
        true.
    c) If placing queen doesn't lead to a solution then umark this [row,
        column] (Backtrack) and go to step (a) to try other rows.
3) If all rows have been tried and nothing worked, return false to trigger
    backtracking.
```

Implementation of Backtracking solution

```
#define N 4
#include<stdio.h>

/* A utility function to print solution */
void printSolution(int board[N][N])
```


Popular Posts

All permutations of a given string

Memory Layout of C Programs

Understanding “extern” keyword in C

Median of two sorted arrays

Tree traversal without recursion and without stack!

Structure Member Alignment, Padding and Data Packing

Intersection point of two Linked Lists

Lowest Common Ancestor in a BST.

Check if a binary tree is BST or not

Sorted Linked List to Balanced BST

```
{
    for (int i = 0; i < N; i++)
    {
        for (int j = 0; j < N; j++)
            printf(" %d ", board[i][j]);
        printf("\n");
    }
}

/* A utility function to check if a queen can be placed on board[row][
   Note that this function is called when "col" queens are already pla
   in columns from 0 to col -1. So we need to check only left side for
   attacking queens */
bool isSafe(int board[N][N], int row, int col)
{
    int i, j;

    /* Check this row on left side */
    for (i = 0; i < col; i++)
    {
        if (board[row][i])
            return false;
    }

    /* Check upper diagonal on left side */
    for (i = row, j = col; i >= 0 && j >= 0; i--, j--)
    {
        if (board[i][j])
            return false;
    }

    /* Check lower diagonal on left side */
    for (i = row, j = col; j >= 0 && i < N; i++, j--)
    {
        if (board[i][j])
            return false;
    }

    return true;
}

/* A recursive utility function to solve N Queen problem */
bool solveNQUtil(int board[N][N], int col)
{
    /* base case: If all queens are placed then return true */
    if (col >= N)
        return true;
```

```
    /* Consider this column and try placing this queen in all rows
       one by one */
    for (int i = 0; i < N; i++)
    {
        /* Check if queen can be placed on board[i][col] */
        if ( isSafe(board, i, col) )
        {
            /* Place this queen in board[i][col] */
            board[i][col] = 1;

            /* recur to place rest of the queens */
            if ( solveNQUtil(board, col + 1) == true )
                return true;

            /* If placing queen in board[i][col] doesn't lead to a sol
               then remove queen from board[i][col] */
            board[i][col] = 0; // BACKTRACK
        }
    }

     /* If queen can not be place in any row in this colum col
        then return false */
    return false;
}

/* This function solves the N Queen problem using Backtracking.  It ma
solveNQUtil() to solve the problem. It returns false if queens cannot
otherwise return true and prints placement of queens in the form of 1s
note that there may be more than one solutions, this function prints o
feasible solutions.*/
bool solveNQ()
{
    int board[N][N] = { {0, 0, 0, 0},
        {0, 0, 0, 0},
        {0, 0, 0, 0},
        {0, 0, 0, 0}
    };

    if ( solveNQUtil(board, 0) == false )
    {
      printf("Solution does not exist");
      return false;
    }

    printSolution(board);
    return true;
```


Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 8 minutes ago

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 47 minutes ago

kzs please provide solution for the problem...

Backtracking | Set 2 (Rat in a Maze) · 51 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 1 hour ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago

```
}

// driver program to test above function
int main()
{
    solveNQ();

    getchar();
    return 0;
}
```

Sources:
http://see.stanford.edu/materials/icspacs106b/H19-RecBacktrackExamples.pdf
http://en.literateprograms.org/Eight_queens_puzzle_%28C%29
http://en.wikipedia.org/wiki/Eight_queens_puzzle

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Related Tpoics:

- Backtracking | Set 8 (Solving Cryptarithmetic Puzzles)

- Tail Recursion
- Find if two rectangles overlap
- Analysis of Algorithm | Set 4 (Solving Recurrences)
- Print all possible paths from top left to bottom right of a mXn matrix
- Generate all unique partitions of an integer
- Russian Peasant Multiplication
- Closest Pair of Points | O(nlogn) Implementation

Writing code in comment? Please use **ideone.com** and share the link here.

31 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

Luna Ram • 18 days ago

Given solution is wrong if change the value of N

Plz Add in isSafe function

for(i=row,j=col;i>=0&&j<n;i--,j++) if(board[i][j])="" return="" false;="" for(i="row,
if(board[i][j])
return false;

HeyM • a month ago

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

what is wrong with this solution. And this would come before given one. (I am

Darshak Mehta ➦ HeyM • 24 days ago

This is right solution

Guest • 4 months ago

How about this code:- http://atiqwhiz.blogspot.in/20...

#include <iostream>

#define N 8

using namespace std;

bool nQueens(int solve[],int n)

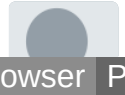
{

int j,r1,r2,c1,c2;

if(n==N)

return true;

for(int i=0;i<n;i++) {="" r2="n;" c2="i;" for(j="0;j<n;j++)" {="" r1="j;" c1="solve[
r2)==abs(c1-c2))" break;="" }="" if(j="n)" {="" solve[n]="i;" if(nqueens(solve,n·
return="" false;="" }="" int="" main()="" {="" int="" solve[n];="" if(nqueens(solve
column\n";="" for(int="" j="0;j<N;j++)" {="" cout<<j<<"="" "<<solve[j]<<endl;="
cout<<"\n\nno="" such="" combination="" is="" possible";="" }="" return="" 0;=



Guest • 4 months ago

How about this code:- http://atiqwhiz.blogspot.in/20...

Guest • 4 months ago

How about this code:- http://atiqwhiz.blogspot.in/20...

#define N 8
using namespace std;
bool nQueens(int solve[],int n)
{
int j,r1,r2,c1,c2;
if(n==N)
return true;
for(int i=0;i<n;i++) {="" r2="n;" c2="i;" for(j="0;j<n;j++)" {="" r1="j;" c1="solve[
r2)==abs(c1-c2))" break;="" }="" if(j="n)" {="" solve[n]="i;" if(nqueens(solve,n-
return="" false;="" }="" int="" main()="" {="" int="" solve[n];="" if(nqueens(solve
column\n";="" for(int="" j="0;j<N;j++)" {="" cout<<j<<"="" "<<solve[j]<<endl;=
cout<<"\n\nno="" such="" combination="" is="" possible";="" }="" return="" 0;=

Byanjati • 5 months ago

for the higher queen problem , we could use 2-Swap Operator for the best Co
backtracking method too

virat • 7 months ago

this will only print only one feasible solution....what about other combinations

Guduru Siva Reddy • 8 months ago

public class Nqueens {
public static void nqueens(int k, int n, int[] a) {

```
for (int i = 1; i <= n; i++) {
if (place(k, i, a) == true) {
a[k] = i;
if (k == n) {
System.out.println();
for (int f = 1; f < a.length; f++) {
System.out.println(a[f]);
}

} else {
nqueens(k + 1, n, a);
}
}
}
}

public static boolean place(int k, int i, int[] a) {// helper function to
```

see more

4 ^ | v •

Guduru Siva Reddy ➦ Guduru Siva Reddy • 6 months ago

This solution prints all possible solutions.

^ | v •

Faizan Ayubi • 9 months ago

what does this mean graphicall "/* Check upper diagonal on left side */";.

^ | v •

Vikash Verma • 9 months ago

You can reduce N xN space with mere N space... :D
Have a look... http://goo.gl/8gNxxb

1 ^ | v •

hh • 10 months ago

What is complexity of your soln?

4

Nikunj Bhartia • 11 months ago

```
 #include <stdio.h>
#include <stdlib.h>
void swap(int *,int*);
int checklist(int *,int );
void display(int *,int );
int count=0,ans=1;

permute(int list[],int i,int n)
{   int j;
    if(i==n){
        if(checklist(list,n))
          {printf("\n soltion no. %d\n\n",count);display(list,n);}
    }
    for(j=i ; j<=n ; j++){
        swap(&list[i],&list[j]);
        permute(list,i+1,n);
        swap(&list[i],&list[j]);
      }
```

see more

Crescent Bokaro • a year ago

this will surely help u simply go through it

Crescent Bokaro • a year ago

```
#include<conio.h>
#include<math.h>
#include<time.h>
#include<stdlib.h>
/* For printing the Grid */
void print_grid(int n,int x[])
{

char arr[100][100];

int i,j;

for(i=1;i<=n;i++)

{
for(j=1; j<=n; j++)
{
arr[i][j]=&#039*&#039;
}
```

see more

Kavish Dwivedi • a year ago

Here is my solution for 16 queen problem

```
 #include&lt;stdio.h&gt;
#define N 16
int sol[N][N];
int check(int row,int col)
{
        int i,j;
```

```
                if(sol[row][i]==1)
                {
                        //printf(&amp;quot;False
&amp;quot;);
                        return 0;
                }
        for(i=row,j=col ; i&gt;= 0 &amp;amp;&amp;amp; j&gt;=0 ; i--,j--)
        {
                if(sol[i][j]==1)
```

see more

Kavish Dwivedi Kavish Dwivedi • a year ago

Sorry , some typo error came unnoticed.

[sourcecode]

```
#include<stdio.h>
#define N 8
int sol[N][N];
int check(int row,int col)
{
int i,j;
for(i=0;i<col;i++)
if(sol[row][i]==1)
{
//printf("False\n");
return 0;
}
for(i=row,j=col ; i>= 0 && j>=0 ; i--,j--)
{
if(sol[i][j]==1)
```

return 0;

see more

Bhuvana Nagaraj • a year ago

Hi.
Can u pls post the n-queens code in opengl where the user inputs the number

Zeenat Islam • a year ago

this solution is getting hanged when N is 16... what changes to make to get thi

GeeksforGeeks ➦ Zeenat Islam • a year ago

Could you please post the code that you tried?

Ajinkya • 2 years ago

What is the time complexity of this approach? and of backtracking in general..
exponential... cna someone work out the time complexity in detail?
Thanks

```
/* Paste your code here (You may delete these lines if not writing c
```

5

sk007 • 2 years ago

Here is another solution for 8x8 board using the backtracking principle:

```
int column[8];


void NQueen(int row){
```

```
if(row==8){
printBoard();
return;
}
for(i=0;i<8;i++){
   column[row]=i;
   if(check(row))
       NQueen(row+1);
}

}
```

see more

1 ^ | v •

Anand • 3 years ago

http://anandtechblog.blogspot....

^ | v •

Anand • 3 years ago

http://anandtechblog.blogspot....

^ | v •

Doom • 3 years ago

heres the code to solve sudoku using same technique

http://ideone.com/vQ7Ej

^ | v •

Nitish Garg • 3 years ago

What update will be required to print all the possible ways to place N queens c
queens, we have 92 different solutions?

kartik ➚ Nitish Garg • 3 years ago

See the following modified code.

```
 #define N 4
#include<stdio.h>

/* A utility function to print solution */
void printSolution(int board[N][N])
{
    for (int i = 0; i < N; i++)
    {
        for (int j = 0; j < N; j++)
            printf(" %d ", board[i][j]);
        printf("\n");
    }
    printf("\n");
}

/* A utility function to check if a queen can be placed on boar
```

see more

reema ➚ kartik • 3 years ago

@GeeksForGeeks Please Don't Forgot to Analyze and mentior

1

kartik ➚ reema • 3 years ago

@rahul: Time complexity is exponential in worst case. S
Backtracking algos like Rat in a Mzae, Knight Tour, Sub

hh kartik • 10 months ago

how come exponential for this ..I thought it should be O(

Subscribe Add Disqus to your site