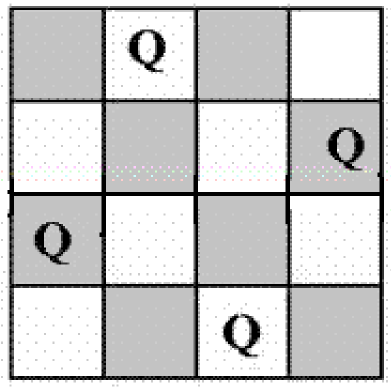
Backtracking | Set 3 (N Queen Problem)

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])
{
    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][col]
   Note that this function is called when "col" queens are already placeed
   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 solution
               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 mainly uses
solveNQUtil() to solve the problem. It returns false if queens cannot be placed,
otherwise return true and prints placement of queens in the form of 1s. Please
note that there may be more than one solutions, this function prints one of the
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;
}

// 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 Topics:

- Linearity of Expectation
- Iterative Tower of Hanoi
- Count possible ways to construct buildings
- Build Lowest Number by Removing n digits from a given number
- Set Cover Problem | Set 1 (Greedy Approximate Algorithm)
- Find number of days between two given dates
- How to print maximum number of A's using given four keys
- Write an iterative O(Log y) function for pow(x, y)

Tags: Backtracking

Tweet 8+1 5

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

47 Comments GeeksforGeeks 1 Login

Recommend 2 Share Sort by Newest

Join the discussion…

Truong Khanh Nguyen · 2 months ago

I found that isSafe is not good for N = 11. Just recursive and check whether it is a solution when we have found N positions. isSafe is called too much and makes the program slower. Find my blog here http://www.capacode.com/?p=682

Reply · Share ›

Sundar · 2 months ago

friends here is my code...

http://ideone.com/fB587a
This one is very Simple

Reply · Share ›

helper · 6 months ago

the power of recursion is astonishing... my solution to this problem http://ideone.com/oJCa92

4 · Reply · Share ›

Pushkar · 7 months ago

What is the time complexity of this Solution??

1 · Reply · Share ›

Rakesh Mondal Pushkar · 7 months ago

Exponential :)

1 • Reply • Share ›

Pushkar Rakesh Mondal • 7 months ago

I got it.. But exactly what in terms of n??

• Reply • Share ›

Rakesh Mondal Pushkar • 7 months ago

An algorithm is said to be exponential time, if T(n) is upper bounded by 2^poly(n), where poly(n) is some polynomial in n. More formally, an algorithm is exponential time if T(n) is bounded by O((2^n)^k) for some constant k.

Source - Wikipedia

4 • Reply • Share ›

joshua fernandus • 8 months ago

Hi I got a program of N Queen in Java,Its working nice I understood the concept of N Queen But some part of program i didn't undestood,In the method enumerate method of the the program,when i gets 3 where i < 4 n gets reset to its previous state for example iteration 1) i =3 and n = 2 then in the next iteration 2) i = 2 and n =1,My question is How n is decremented to its previous state please help me.

source: http://introcs.cs.princeton.ed...

• Reply • Share ›

ntk18 • 8 months ago

Here is my code : http://ideone.com/mFYndK
Let me know if there is any error in the code.

• Reply • Share ›

np • 10 months ago

watch 13 minutes video you will be able to code for the NQUEEN problem
https://www.youtube.com/watch?...

• Reply • Share ›

Sumit Gulati • 10 months ago

More Simple safe function :
int issafe(int row,int col,int sol[N][N])

{

int i,j;

for(int i=0;i<col;i++) {="" if(sol[row][i])="" return="" false;="" }="" int="" diff="row-col;"

for(i="0;i&lt;N;i++)" {="" for(j="0;j&lt;N;j++)" {="" if((i-j)="=diff)" {="" if(sol[i][j])="" return="" false;="" }="" }="" }="" int="" sum="row+col;" for(i="0;i&lt;N;i++)" {="" for(j="0;j&lt;N;j++)" {="" if((i+j)="=sum)" if(sol[i][j])="" return="" false;="" }="" }="" return="" true;="" }="">

• Reply • Share ›

typing.. • a year ago

what is time complexty of this program?

5 • Reply • Share ›

gandhi_rahul • a year ago

This code is not working. It is always printing "Solution does not exist" . WHY SO ??

1 • Reply • Share ›

guest gandhi_rahul • 8 months ago

apne naam pe khare utre ho beta. achcha hai

15 • Reply • Share ›

Mayank Koul • a year ago

THIS CODE IS SAME AS PREVIOUS BUT HAS BEEN PROPERLY ARRANGED AND READY TO USE

#include<stdio.h>

#include<math.h>

int a[30],count=0;

int place(int pos)

{

int i;

for(i=1;i<pos;i++) {="" if((a[i]="=a[pos])||((abs(a[i]-a[pos])==abs(i-pos))))" return="" 0;="" }="" return="" 1;="" }="" void="" print_sol(int="" n)="" {="" int="" i,j;="" count++;="" printf("\n\nsolution="" #%d:\n",count);="" for(i="1;i&lt;=n;i++)" {="" for(j="1;j&lt;=n;j++)" {="" if(a[i]="=j)" printf("q\t");="" else="" printf("*\t");="" }="" printf("\n");="" }="" }="" void="" queen(int="" n)="" {="" int="" k="1;" a[k]="0;" while(k!="0)" {="" a[k]="a[k]+1;" while((a[k] <="n)&amp;&amp;!place(k))" a[k]++;="" if(a[k]<="n)" {="" if(k="=n)" print_sol(n);="" else="" {="" k++;="" a[k]="0;" }="" }="" else="" k--;="" }="" }="" void="" main()="" {="" int="" i,n;="" printf("enter="" the="" number="" of="" queens\n");="" scanf("%d",&n);="" queen(n);="" printf("\ntotal="" solutions="%d\n",count);" }="">

1 • Reply • Share ›

Mayank Koul • a year ago

checkout this code

#include<stdio.h>

#include<math.h>

int a[30],count=0;

int place(int pos)

{

int i;

for(i=1;i<pos;i++) {="" if((a[i]="=a[pos])||((abs(a[i]-a[pos])==abs(i-pos))))" return="" 0;="" }="" return="" 1;="" }="" void="" print_sol(int="" n)="" {="" int="" i,j;="" count++;="" printf("\n\nsolution="" #%d:\n",count);="" for(i="1;i&lt;=n;i++)" {="" for(j="1;j&lt;=n;j++)" {="" if(a[i]="=j)" printf("q\t");="" else="" printf("*\t");="" }="" printf("\n");="" }="" }="" void="" queen(int="" n)="" {="" int="" k="1;" a[k]="0;" while(k!="0)" {="" a[k]="a[k]+1;" while((a[k] <="n)&amp;&amp;!place(k))" a[k]++;="" if(a[k]<="n)" {="" if(k="=n)" print_sol(n);="" else="" {="" k++;="" a[k]="0;" }="" }="" else="" k--;="" }="" }="" void="" main()="" {="" int="" i,n;="" printf("enter="" the="" number="" of="" queens\n");="" scanf("%d",&n);="" queen(n);="" printf("\ntotal="" solutions="%d\n",count);" }="">

2 ^ | ⌄ • Reply • Share ›

Luna Ram • a year 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,j=col;i&lt;N&amp;&amp;j">=0;i++,j--)
if(board[i][j])
return false;

1 ^ | ⌄ • Reply • Share ›

HeyM • a year 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 beginner.)

2 ^ | ⌄ • Reply • Share ›

Darshak Mehta ➦ HeyM • a year ago

This is right solution

• Reply • Share ›

Guest • a year 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&lt;n;j++)" {="" r1="j;" c1="solve[j];"
if(r1="=r2||c1==c2||abs(r1-r2)==abs(c1-c2))" break;="" }="" if(j="=n)" {="" solve[n]="i;"
if(nqueens(solve,n+1))="" return="" true;="" }="" }="" return="" false;="" }="" int="" main()="" {=""
int="" solve[n];="" if(nqueens(solve,0))="" {="" cout<<"row="" column\n";="" for(int=""
j="0;j&lt;N;j++)" {="" cout<<j<<"="" "<<solve[j]<<endl;="" }="" }="" else="" {="" cout<<"\n\nno=""
such="" combination="" is="" possible";="" }="" return="" 0;="" }="">
```

• Reply • Share ›

Guest • a year ago

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

• Reply • Share ›

Guest • a year 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&lt;n;j++)" {="" r1="j;" c1="solve[j];"
if(r1="=r2||c1==c2||abs(r1-r2)==abs(c1-c2))" break;="" }="" if(j="=n)" {="" solve[n]="i;"
if(nqueens(solve,n+1))="" return="" true;="" }="" }="" return="" false;="" }="" int="" main()="" {=""
int="" solve[n];="" if(nqueens(solve,0))="" {="" cout<<"row="" column\n";="" for(int=""
```

j="0;j&lt;N;j++)" {="" cout<<j<<"="" "<<solve[j]<<endl;="" }="" }="" else="" {="" cout<<"\n\nno="" such="" combination="" is="" possible";="" }="" return="" 0;="" }="">

^ | ˅ • Reply • Share ›

Byanjati • a year ago

for the higher queen problem , we could use 2-Swap Operator for the best Complexity , and it implement the backtracking method too

^ | ˅ • Reply • Share ›

virat • 2 years ago

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

^ | ˅ • Reply • Share ›

Guduru Siva Reddy • 2 years 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);
}
}
}
}
```

see more

6 ^ | ˅ • Reply • Share ›

Guduru Siva Reddy ➦ Guduru Siva Reddy • a year ago

This solution prints all possible solutions.

^ | ˅ • Reply • Share ›

Faizan Ayubi • 2 years ago

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

^ | ˅ • Reply • Share ›

Vikash Verma · 2 years ago

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

2 ^ | ⌄ · Reply · Share ›

hh · 2 years ago

What is complexity of your soln?

4 ^ | ⌄ · Reply · Share ›

Nikunj Bhartia · 2 years 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

^ | ⌄ · Reply · Share ›

Crescent Bokaro · 2 years ago

this will surely help u simply go through it

^ | ⌄ · Reply · Share ›

Crescent Bokaro · 2 years 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

Reply • Share ›

Kavish Dwivedi • 2 years 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;
        for(i=0;i&lt;col;i++)
                if(sol[row][i]==1)
                {
                        //printf(&quot;False
&quot;);
                        return 0;
                }
        for(i=row,j=col ; i&gt;= 0 &amp;&amp; j&gt;=0 ; i--,j--)
        {
                if(sol[i][j]==1)
```

see more

Reply • Share ›

Kavish Dwivedi → Kavish Dwivedi • 2 years 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

• Reply • Share ›

Bhuvana Nagaraj • 2 years ago

Hi.
Can u pls post the n-queens code in opengl where the user inputs the number of queens? (Max no of queens=12).

• Reply • Share ›

Zeenat Islam • 2 years ago

this solution is getting hanged when N is 16... what changes to make to get this code to work for large values of N??

• Reply • Share ›

GeeksforGeeks Zeenat Islam • 2 years ago

Could you please post the code that you tried?

• Reply • Share ›

Ajinkya • 3 years ago

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

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

10 • Reply • Share ›

sk007 • 3 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 • Reply • Share ›

Anand • 4 years ago

http://anandtechblog.blogspot....

• Reply • Share ›

Anand • 4 years ago

http://anandtechblog.blogspot....

• Reply • Share ›

Doom • 4 years ago

heres the code to solve sudoku using same technique

http://ideone.com/vQ7Ej

• Reply • Share ›

Nitish Garg • 4 years ago

What update will be required to print all the possible ways to place N queens on an N X N chessboard, like for 8 queens, we have 92 different solutions?

• Reply • Share ›

kartik ➦ Nitish Garg • 4 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 board[row][col]
```

see more

^ | v • Reply • Share ›

reema ➦ kartik • 4 years ago

@GeeksForGeeks Please Don't Forgot to Analyze and mention Time,Space Complexity

1 ^ | v • Reply • Share ›

kartik ➦ reema • 4 years ago

@rahul: Time complexity is exponential in worst case. Same is the case with all other Backtracking algos like Rat in a Mzae, Knight Tour, Subseet Sum.. etc

^ | v • Reply • Share ›

hh ➦ kartik • 2 years ago

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

^ | v • Reply • Share ›

Google™ Custom Search

-
-
-
-
    - Interview Experiences
    - Advanced Data Structures
    - Dynamic Programming
    - Greedy Algorithms
    - Backtracking
    - Pattern Searching
    - Divide & Conquer
    - Mathematical Algorithms
    - Recursion
    - Geometric Algorithms
-

- ## 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
- Follow @GeeksforGeeks

- ## Recent Comments

    - Ashish Aggarwal

Try Data Structures and Algorithms Made Easy -...

Algorithms · 17 minutes ago

- Vlad

  Thanks. Very interesting lectures.

  Expected Number of Trials until Success · 1 hour ago

- cfh

  My implementation which prints the index of the...

  Longest Even Length Substring such that Sum of First and Second Half is same · 1 hour ago

- Gaurav pruthi

  forgot to see that part ;)

  Bloomberg Interview | Set 1 (Phone Interview) · 1 hour ago

- saeid aslami

  thanks

  Greedy Algorithms | Set 7 (Dijkstra's shortest path algorithm) · 1 hour ago

- Cracker

  Implementation:...

  Implement Stack using Queues · 2 hours ago

-