GeeksforGeeks

A computer science portal for geeks

GeeksQuiz

- Home
- Algorithms
- DS
- GATE
- Interview Corner
- Q&A
- C
- C++
- Java
- Books
- Contribute
- Ask a Q
- About

Array
Bit Magic
C/C++
Articles
GFacts
Linked List
MCQ
Misc
Output
String
Tree
Graph

Dynamic Programming | Set 37 (Boolean Parenthesization Problem)

Given a boolean expression with following symbols.

```
Symbols
    'T' ---> true
    'F' ---> false
```

And following operators filled between symbols

```
Operators
    &   ---> boolean AND
    |   ---> boolean OR
    ^   ---> boolean XOR
```

Count the number of ways we can parenthesize the expression so that the value of expression evaluates to true.

Let the input be in form of two arrays one contains the symbols (T and F) in order and other contains operators (&, | and ^}

Examples:

```
Input: symbol[]    = {T, F, T}
       operator[]  = {^, &}
Output: 2
The given expression is "T ^ F & T", it evaluates true
in two ways "((T ^ F) & T)" and "(T ^ (F & T))"

Input: symbol[]    = {T, F, F}
       operator[]  = {^, |}
Output: 2
The given expression is "T ^ F | F", it evaluates true
in two ways "( (T ^ F) | F )" and "( T ^ (F | F) )".

Input: symbol[]    = {T, T, F, T}
       operator[]  = {|, &, ^}
Output: 4
The given expression is "T | T & F ^ T", it evaluates true
in 4 ways ((T|T)&(F^T)), (T|(T&(F^T))), (((T|T)&F)^T)
and (T|((T&F)^T)).
```

Solution:
Let **T(i, j)** represents the number of ways to parenthesize the symbols between i and j (both inclusive) such that the subexpression between i and j evaluates to true.

$$T(i,j) = \sum_{k=i}^{j-1} \begin{cases} T(i,k) * T(k+1,j) & \text{If operator[k] is '\&'} \\ Total(i,k) * Total(k+1,j) - F(i,k) * F(k+1,j) & \text{If operator[k] is '|'} \\ T(i,k) * F(k+1,j) + F(i,k) * T(k+1) & \text{If operator[k] is '\^{}'} \end{cases}$$
$$\text{Total(i, j)} = \text{T(i, j)} + \text{F(i, j)}$$

Let **F(i, j)** represents the number of ways to parenthesize the symbols between i and j (both inclusive) such that the subexpression between i and j evaluates to false.

$$F(i,j) = \sum_{k=i}^{j-1} \begin{cases} Total(i,k) * Total(k+1,j) - T(i,k) * T(k+1,j) & \text{If operator[k] is '\&'} \\ F(i,k) * F(k+1,j) & \text{If operator[k] is '|'} \\ T(i,k) * T(k+1,j) + F(i,k) * F(k+1) & \text{If operator[k] is '\^{}'} \end{cases}$$
$$\text{Total(i, j)} = \text{T(i, j)} + \text{F(i, j)}$$

Base Cases:

```
T(i, i) = 1 if symbol[i] = 'T'
T(i, i) = 0 if symbol[i] = 'F'

F(i, i) = 1 if symbol[i] = 'F'
F(i, i) = 0 if symbol[i] = 'T'
```

If we draw recursion tree of above recursive solution, we can observe that it many overlapping subproblems. Like other dynamic programming problems, it can be solved by filling a table in bottom up manner. Following is C++ implementation of dynamic programming solution.

```
#include<iostream>
#include<cstring>
using namespace std;

// Returns count of all possible parenthesizations that lead to
// result true for a boolean expression with symbols like true
// and false and operators like &, | and ^ filled between symbols
int countParenth(char symb[], char oper[], int n)
{
    int F[n][n], T[n][n];

    // Fill diaginal entries first
    // All diagonal entries in T[i][i] are 1 if symbol[i]
    // is T (true).  Similarly, all F[i][i] entries are 1 if
    // symbol[i] is F (False)
    for (int i = 0; i < n; i++)
    {
        F[i][i] = (symb[i] == 'F')? 1: 0;
        T[i][i] = (symb[i] == 'T')? 1: 0;
    }

    // Now fill T[i][i+1], T[i][i+2], T[i][i+3]... in order
    // And F[i][i+1], F[i][i+2], F[i][i+3]... in order
    for (int gap=1; gap<n; ++gap)
    {
        for (int i=0, j=gap; j<n; ++i, ++j)
        {
            T[i][j] = F[i][j] = 0;
            for (int g=0; g<gap; g++)
            {
                // Find place of parenthesization using current value
                // of gap
                int k = i + g;

                // Store Total[i][k] and Total[k+1][j]
                int tik = T[i][k] + F[i][k];
                int tkj = T[k+1][j] + F[k+1][j];

                // Follow the recursive formulas according to the current
                // operator
                if (oper[k] == '&')
                {
                    T[i][j] += T[i][k]*T[k+1][j];
                    F[i][j] += (tik*tkj - T[i][k]*T[k+1][j]);
                }
                if (oper[k] == '|')
                {
                    F[i][j] += F[i][k]*F[k+1][j];
                    T[i][j] += (tik*tkj - F[i][k]*F[k+1][j]);
                }
                if (oper[k] == '^')
                {
                    T[i][j] += F[i][k]*T[k+1][j] + T[i][k]*F[k+1][j];
                    F[i][j] += T[i][k]*T[k+1][j] + F[i][k]*F[k+1][j];
                }
            }
        }
    }
    return T[0][n-1];
}

// Driver program to test above function
int main()
{
    char symbols[] = "TTFT";
    char operators[] = "|&^";
    int n = strlen(symbols);

    // There are 4 ways
    // ((T|T)&(F^T)), (T|(T&(F^T))), (((T|T)&F)^T) and (T|((T&F)^T))
    cout << countParenth(symbols, operators, n);
    return 0;
}
```

Output:

```
4
```

Time Complexity: O(n^3)
Auxiliary Space: O(n^2)

References:
http://people.cs.clemson.edu/~bcdean/dp_practice/dp_9.swf

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)

Tweet 3 g+1 1

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

11 Comments GeeksforGeeks Login

11 Comments GeeksforGeeks Login

Recommend Share Sort by Newest

Join the discussion…

khan • 3 months ago

this program uses lots of time and space complexity

• Reply • Share ›

tourist • 3 months ago

Can we not compute
T(i,j) of length 2
T(i,j) of length 3 and so on instead ?

• Reply • Share ›

helper • 6 months ago

a very nice code.
a small suggestion:
int tik = T[i][k] + F[i][k];
int tkj = T[k+1][j] + F[k+1][j];
we can put these two assignments in that these two assignments in second if. another suggestion. the later two if's can be made else if.
all this will save some time.

• Reply • Share ›

aa1992 • 7 months ago

excellent program.

• Reply • Share ›

anuj • 7 months ago

what is the application on boolean parenthesization??

• Reply • Share ›

Karshit Jaiswal • 10 months ago

I think a dry run example and proper explanation for the formula is required for this topic.
Although the video is fine to understand. its really a good problem.
Thanks guys.

• Reply • Share ›

Bhagwat Singh • a year ago

ok i understand with the help of video

• Reply • Share ›

Reply • Share ›

Bhagwat Singh • a year ago

kindly provide the logic behind the mathematical formula.

Reply • Share ›

Rainer Hoffmann • a year ago

Hello,
running the program with this input from top of article:

Input: symbol[] = {T, F, F}
operator[] = {^, |}
gives Output = 2 instead of above mentioned
Output: 1
Could you please check?
Thank you!

Reply • Share ›

> **GeeksforGeeks** Mod → Rainer Hoffmann • a year ago
>
> Thanks for pointing this out. The code seems to be giving the correct output. The output mentioned in example was incorrect. We have updated the example now.
>
> Reply • Share ›

GOPI GOPINATH • a year ago

Wow the explanation given in the reference is pretty OSM..Nice one

Reply • Share ›

Subscribe   Add Disqus to your site   Privacy

-
-
-
-
    - 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

- lt_k

  i need help for coding this function in java...

  Java Programming Language · 1 hour ago

- Piyush

  What is the purpose of else if (recStack[*i])...

  Detect Cycle in a Directed Graph · 1 hour ago

- Andy Toh

  My compile-time solution, which agrees with the...

  Dynamic Programming | Set 16 (Floyd Warshall Algorithm) · 1 hour ago

- lucy

  because we first fill zero in first col and...

  Dynamic Programming | Set 29 (Longest Common Substring) · 2 hours ago

- lucy

  @GeeksforGeeks i don't n know what is this long...

Dynamic Programming | Set 28 (Minimum insertions to form a palindrome) · 3 hours ago

- manish

  Because TAN is not a subsequence of RANT. ANT...

  Given two strings, find if first string is a subsequence of second · 3 hours ago

-