a bit of cs

an exploration on computer science and mathematics

Tuesday, December 2, 2014

Dynamic Programming on Tree: Forming up Subtrees with One Black Node

This problem is from Codeforces Round #263 (Div. 1) Problem B:

Problem Statement:
461B - Appleman and Tree

Solution:
The dynamic programming approach to this tree problem is quite interesting. The focus of the problem is to find the number of ways such that the subtrees formed have one and only one black node each. The idea behind the DP might not be that intuitive though, and in a sense its implementation is also not as straight forward, although its final implementation looks very simple.

Let's consider node u. Node u has a few children v_i. To solve the problem, we need to keep track of two information:
1. The number of ways such that the subtree that is rooted at u has no black nodes at all, while the remaining subtrees formed have exactly one black node each. Call this W[u].
2. Symmetrically, the number of ways such that the subtree that is rooted at u has exactly one black nodes, and the remaining subtrees formed have also exactly one black node each. Call this B[u].

Now, to compute W[u] and B[u], initialize W[u] = 1, B[u] = 0 if color[u] is white, otherwise W[u] = 0, B[u] = 1. Then, we iterate through the children of u one by one, for each v_i for $i = 1 \ldots n$:
1. Suppose we already know what is W[v] and B[v].
2. Let W' and B' be W[u] and B[u] after we consider node v. Set W' = 0 and B' = 0.
3. We have two case to consider:
Case 1: v is included in the subtree rooted at u.
    then W' += W[u] * W[v] (so that the subtree has no black node)
    and B' += B[u] * W[v] + B[v] * W[u] (since the subtree must only have one black node).
Case 2: v forms a separate subtree rooted at v that is not a part of subtree rooted at u.
    then W' += W[u] * B[v] (since the subtree rooted at v must have one black node)
    and B' += B[u] * B[v] (same reason. all other subtree must have one black node).
4. Finally update W[u] = W', B[u] = B'.

Implementation:

```
        dp[u][1] %= MOD;

        dp[u][0] += zero * dp[v][0];
        dp[u][0] %= MOD;
        dp[u][1] += one * dp[v][0] + zero * dp[v][1];
        dp[u][1] %= MOD;

    }
}

int main(){
    scanf("%d", &N);
    adj = vector<vector<int> >(N+3);
    int v;
```

Find

Search

Total Pageviews

121548

Articles On

- Algorithm
- CodeChef
- Codeforces
- Data Structure
- Dynamic Programming
- Greedy Algorithm
- SPOJ
- Statistics
- TopCoder
- UVa

Past Articles

December 2014 (25) ▼

About me

Prajogo Tio

G+ Follow 91

```
    for(int i=0;i<N-1;++i){
        scanf("%d", &v);
        adj[i+1].push_back(v);
        adj[v].push_back(i+1);
    }
    for(int i=0;i<N;++i){
        scanf("%d", &col[i]);
    }
    for(int i=0;i<N;++i){
        vis[i] = 0;
    }
    dfs(0);
    printf("%d\n", (int) dp[0][1]);
    return 0;
```

View my complete profile

Posted by Prajogo Tio at 4:56 AM G+

Labels: Codeforces, Combinatorics, Dynamic Programming, Tree Coloring

1 comment

Add a comment

Top comments

Prajogo Tio via Google+ 3 years ago - Shared publicly

Dynamic Programming on Tree: Forming up Subtrees with One Black Node

This problem is from Codeforces Round #263 (Div. 1) Problem B: Problem Statement: 461B - Appleman and Tree Solution: The dynamic programming approach to this tree problem is quite interesting. The focus of the problem is to find the number of ways such that...

1 · Reply

Newer Post Home Older Post

Subscribe to: Post Comments (Atom)