Solutions to previous round!

11 March 2016

Counting binary numbers

How many binary numbers of length N have maximum K consecutive 1s ?

E.g. for N = 3, K = 1 the answer is 5

000, 001, 010, 100, 101

Counting binary numbers solution

Let's calculate the following way, num[i][j] = how many binary numbers of length i exist such that they currently have exactly j 1s at the end.

The recurrence is:

- if the last bit is 0, num[i][0] = sum(num[i-1][j]) for all j from 0 to k
- if the last bit is 1, num[i][j] = num[i-1][j-1] for all j from 1 to k

Trap coins

You have an array of values v. Select a subset so that their sum is as big as possible and no 2 consecutive elements are taken.

Trap coins solution

Let’s do the following dynamic programming, best[i] = how many coins can we get if we only consider the first i elements.

Well if we want to take the current coin we can have a maximum of v[i] + best[i-2] (we can’t get best[i-1] since it may include v[i-1] which is forbidden).

If we don’t take the current position we simply take best[i-1].

So best[i] = max(best[i-1], best[i-2] + v[i])

Tree coins

You are given a tree with values in each node, select a subset of nodes so that their sum is as big as possible, and no two selected nodes are neighbours.

Tree coins solution

This is very similar to the previous problem (the previous was a corner case where the tree is a line).

We'll root the tree at 1 to make it easier for us and do the following two dynamic programming: take[i] -> the maximum value of i's subtree if we take i, and notake [i] the maximum value of i's subtree if we don't take i. The final solution will be max(take[i],notake[i])

take[i] will be equal to v[i] plus the sum of the best notakes of its children.

notake[i] will be equal to the sum of best value of its children (either taken or not taken since both are allowed)

Tree coins solution

So take[i] = v[i] + sum(notake[c]) for all c children of i

notake[i] = sum(max(notake[c], take[c])) for all c children of i.

Chocolate squares

You have an N x M chocolate, you can split it vertically or horizontally. How many splits will it take to reach only square pieces ?

Chocolate squares solution

Let’s do the following dynamic programming best[i][j], how many will it take if we have an i * j matrix.

If i equals j, the result is 0.

Otherwise we’ll just simulate all the vertical and horizontal splits and pick the one that minimizes the sum of splits needed in the 2 pieces.

best[i][j] = 1 + min(min(best[i][k] + best[i][j-k]), min(best[k][j] + best[i-k][j]))

South-East Counting

You have a N x M matrix with free cells (0) or occupied cells (1). How many ways are there to get from (1,1) to (N,M) ?

South-East Counting solution

Let’s count how many ways there are to get from 1,1 to i, j : num[i][j]

Well if (i, j) is occupied, the result is 0.

If it’s free then we just add it’s north and west neighbour, since those are the only 2 possible paths we can continue:

```
if(free) num[ i ][ j ] = num[i-1][ j ] + num[ i ][j-1];

else num[ i ][ j ] = 0;
```