Advanced DP

Hello, and welcome to Competitive Programming! Today we are going to talk about some extra techniques you can apply to dynamic programming to make it more effective, especially if you are low on memory.

Bitmasks

You have already seen the bitmask technique used for recursive backtracking, but it can be used in dynamic programming as well.

Consider a problem where we have a collection of items that could be paired together. Each paring has a different cost associated with it, and we want to find the minimum cost for pairing all items together.

We can use a bitmask to represent the items that have been paired together. To pair (i) and (j), we simply set the bits (i) and (j) in a bitmask integer. It will be easiest to model this if we let the zero bit be the right most one.

The resulting bitmask now is a index into our DP array. You can either start with 0 and do a bottom up DP, or else start with all the bits set and do a top-down approach.

Other Tricks

There are a few other tricks that you may need to use.

One situation that sometimes comes up is that the state you need to use for your DP index could have negative entries. This is usually not a problem, all you have to do is have apply an offset. For example, suppose you had a problem in which you has a set of numbers and for each one you could either add or subtract it. If you needed to use this as a state index, you would first find out the largest negative number and then add that to the index for your array.

Sometimes the array you need is too big for the memory limits, but also sparse. In other words, you are not going to need every one of the entries. In that case, using a tree based map will save memory for a small time complexity hit.

In general, the hardest part about solving DP problems is making sure you have the correct representation for your state.

Dropping One Parameter

Sometimes you have a DP problem where the state space is too large because there are multiple parameters that cause your array to be multidimensional. Sometimes its possible to compute one of those dimensions in terms of the other, and therefore you don't need to store it in the array.

b

Next
Transcript — Basic Graphs →