Code Jam 2012 - World Finals

Analysis: Shifting Paths

For the small dataset, we can directly simulate the process of walking through the forest. How many steps can this take? There are 2^{10} states the trees in the clearings can be in, and 10 possible states for the clearing we are standing in. So if we reach the final clearing, it can't take more than 10×2^{10} steps, and if we take that many steps without reaching the final clearing we know the answer is Infinity.

For the large dataset, 40×2^{40} steps is too many to simulate directly, but can an input case approach this many?

A test case can be designed to make the clearings simulate an **N**-1 bit counter. The first clearing has both paths leading to the second clearing. Each clearing after that forms a chain where one path leads to the next clearing, and one path leads back to the first clearing. Whenever the trail leads back to the first clearing, the states of the clearings give the next **N**-1 bit number, and after all of those numbers have been produced, it can reach the final clearing. This will take at least 2^{39} steps.

So we need a solution that will not simulate every step individually. In the previous example, we spend 2^{21}-2 steps in the first 20 clearings between each time we visit any of the last 20 clearings. If our program could detect that this always happens, it could simulate those 2^{21}-2 steps without having to do them all individually. We will only take 2^{20}-3 steps from any of the second twenty clearings, so the total runtime would be reasonable.

An implementation of this approach is to split the clearings into two sets A and B of roughly equal size. We then use dynamic programming to compute, for each location in A and each of the $2^{|A|}$ states of the clearings in A, how many steps it will take to leave A, which clearing in B we will be leaving towards, and what state the clearings in A will be left in.

After this DP is done, our solution will take an amount of time proportional to the number of steps we take from clearings in B -- for each of those, we simulate that step, and if that takes us to a clearing in A, we look up the appropriate result of the DP from a table. So to make this solution efficient, we need to choose A and B so that only a small number of steps are taken from clearings in B.

Find the distance from each clearing to the final clearing, where distance is defined as the number of paths you need to take, assuming the state of the clearings is optimal. If the final clearing cannot be reached from some clearings, put those in a separate set. If we ever reach one of those, the answer is Infinity.

Choose B to be the closest half of the remaining clearings. There can be at most $B \times 2^{|B|}$ steps taken from clearings in B, because we cannot repeat a state of B without reaching the final clearing. This can be only at most 20×2^{20}, so this solution is sufficiently fast.