Code Jam 2011 - World Finals

Analysis: Rains Over Atlantis

One obvious solution (which works for small input) is to simulate. The trouble starts when the heights are large, and **M** is small. Thus, we need to be able to do multiple steps at once.

Begin by noting that, given the current map of heights, we can determine the water levels (i.e., what areas are submerged and what areas are not) using a single run of Dijkstra's shortest path algorithm, in RC log(RC) time.

We will have a single step procedure. What it does is:

1. Calculate water levels for all fields.
2. For each field find out if it is submerged; if not, find the lowest water level in the adjacent fields.
3. Create a new map in which all the heights are lessened by the appropriate erosion, and increase the day counter by one.

This is one step of the simulation, which is also needed for the naive solution.

We will also have a multistep procedure, which tries to perform multiple single steps at once, as long as all of the following are true:

- all non-submerged fields erode at maximum speed,
- the water levels of all submerged fields decrease at the same speed, and
- no submerged field becomes un-submerged during these steps.

The multistep procedure will work as follows:

1. Calculate the water levels for all fields.
2. For each field find out whether it is submerged. If not, find out how much it will erode. If this is not **M**, break the multistep. Note that we can choose to erode fields to below sea level for simplicity without changing the result.
3. If all fields erode at speed **M**, then the water levels of all "lakes" also decrease at this speed. Note that each lake has at least one field on the border which is not submerged, but determines the water level of the lake, and this level will decrease by **M**. Thus, we can continue repeating steps until some field that was submerged surfaces.
4. Thus, we find the submerged field with the smallest amount of water on top, and batch the appropriate number of steps together.
5. If there are no submerged fields, then we increment the day counter by ceil(maxheight / M) and finish the algorithm.

Note that in each multistep one field that was submerged becomes uncovered, and it's easy to see that a field that is not submerged will never become submerged in the future - thus there are at most RC multisteps in the algorithm (actually, fewer because the fields on the edge of the board are never submerged).

Now we want to know how many steps are possible before we can perform a multistep. Define an auxilliary graph that has a node for each un-submerged field. For each lake we join all the fields in this lake to an arbitrary field on the boundary of the lake which defines the water level for the lake (the outlet of the lake). We define natural edges in the graph -- two nodes have an edge if any of the two fields merged into the two nodes were adjacent. For each nodes we can define its "parent" to be one of the nodes with the lowest level adjacent to it. Thus, we have a tree, rooted in the external sea. We define any path going upward in the tree to be "fixed" if all of the edges along the path have a height difference of at least **M**.

We can now prove that each day either

- a submerged becomes uncovered (thus decreasing the number of multisteps available), or
- the number of vertices that have fixed paths to the root increases, or
- all vertices lie on fixed paths.

If all vertices lie on fixed paths, then we can perform a multistep. As there can be at most RC increases before all vertices are on fixed paths, this means we get at most RC steps and at most one multistep before some field is uncovered, meaning at most $(RC)^2$ steps and RC multisteps in total, giving a running time of $(RC)^3 \log(RC)$. The constants are very small here, so this will easily run in time.

Now consider any fixed path. We assume the node distribution remains the same (i.e. the lakes remain lakes, and nothing gets uncovered). First notice that it remains a fixed path after one day - all of the fields on the path decrease by the same amount, so the differences remain the same. Now take any vertex that is not on a fixed path, but its parent is. Note that the sea level is on a fixed path by definition, so if no such vertex exists, it means that all vertices are on a fixed path. Then after one day the height of this vertex decreases to the height of the parent, while the height of the parent decreases by **M**. So the new difference is **M**, meaning that the vertex has joined the fixed path. This ends the proof.

Note that the upper bound can actually be (more or less) hit in the following map with M = 2: take a winding path of length that is on the order of RC, that does not touch itself, like so:

```
X X X X X X X
P P P P P P X
X X X X X P X
X P P P P P X
X P X X X X X
X P P P P P X
X X X X X X X
```

Where X represent very large values, and P represents the path. Thus, for the foreseeable future, the tree described above is actually the single path marked by Ps.

Now take the following values on the path: A, A-2L-1, A+1, A - 4L - 1, A+1, A - 6L - 1, A+1, A - 8L - 1, etc., where L is the length of the path. It is not difficult to check that "off by one" changes get propagated up the path with speed one, and each change gets to be fully propagated before the next one kicks in (due to the uncovering of another lake).