Minimum effort algorythm

Mariusz Figiel

1 Task to acomplish

There's a map in shape of matrix (n x n size). We wanna get from top left corner to bottom right corner but:

- we can go only down or right
- every value we step on is the value of effort we make to get there
- we look for minimum effort way of getting there

For example in matrix:

$$\begin{matrix} 2 & 4 \\ 1 & 1 \end{matrix}$$

Minimum effort equals

$$2 + 1 + 1 = 4$$

.

2 Solution idea

2.1 Problems

When we consider options for 1x1 it's going easy - only one way, because we are already in the end. Also - 2x2 and 3x3 matrixes we can consider to be really easy problem, because there are few possibilities for each of them:

size	**number of possible ways**
1x1	1
2x2	2
3x3	6

but counting possibilities for greater matrixes, we see that they are growing in unusual way

size	**number of possible ways**
4x4	20
5x5	70

It shows up, that they are growing in (some kind of) controlled way.

To see this, we need to know what is "Pascal Triangle". After that, when we look at it, and use some "trick" everything looks obviously clear.

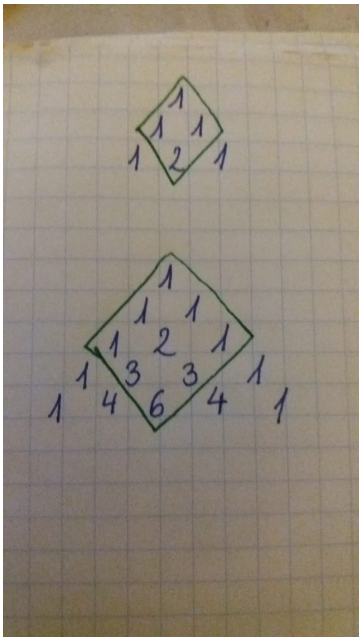

Figure 1: Fragments of Pascal Triangle

On the figure 1 we can see 2 fragments of Pascal Triangle, with marked some numbers in green diamonds. When we sum up them (and add 1) we will receive number of possible ways in matrixes 3x3 and 4x4. It shows up that, for greater matrixes dependency stays the same (only size of triangle and diamond grows).

That knowledge firstly gives us information that, making program checking EVERY possible way (especially for huge maps) will consume huge amounts of computer resources.

2.2 Suggested solution

After think through problem mathematically, we need to look closer at problem assumptions.

If effort is counted as value from every single matrix cell we step on, then we always are on first (top left) and last (bottom right) cell. So they shouldn't be our problem.

After that observation we should look at 2x2 matrix. It looks like we should only be aware of 2 cells, and choose one that has less value. So the solution look like:

$$\text{first cell} + \text{last cell} + \text{the one with less value} = \text{minimum effort}$$

Let's expand observation to 3x3 matrix. If we cut out first/last cell we are left with 2 2x2 matrixes overlapping each other (Look figure 2).

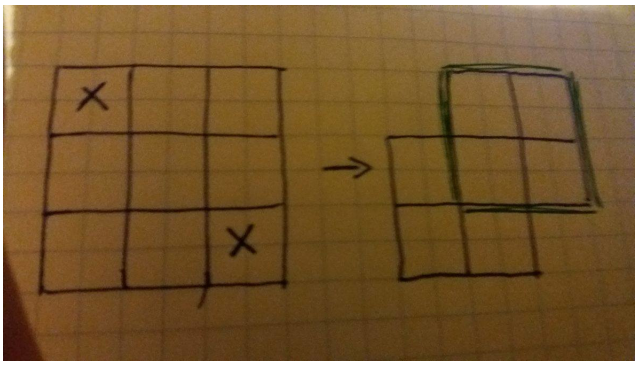

Figure 2: 3x3 matrix with cut out first/last cell and marked one of overlapping 2x2 matrixes

So we have brought 3x3 matrix problem to 2x2 matrix problem, which one we have done already. There's just need to check which 2x2 matrix needs less minimal effort and choose that one. What's next? :) Expand it further like on figure 3...

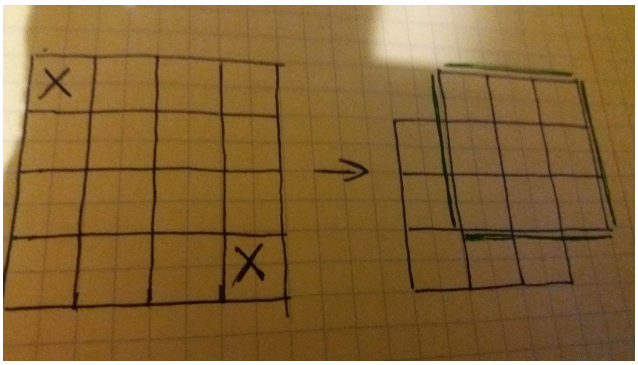

Figure 3: 4x4 matris with cut out first/last cell and marked one of overlapping 3x3 matrixes

So general suggested solution for n x n matrix looks like:

$$\text{n x n} \Rightarrow \text{(n-1)x(n-1)} \Rightarrow \ldots \Rightarrow \text{3x3} \Rightarrow \text{2x2}$$

3 Program task

- gather maps (matrixes) from file,

- calculate minimum effort way,
- print results in the same order as maps in source file.