434 · Number of Islands II(LintCode)

Description

Given a n,m which means the row and column of the 2D matrix and an array of pair A(size k). Originally, the 2D matrix is all 0 which means there is only sea in the matrix. The list pair has k operator and each operator has two integer A[i].x, A[i].y means that you can change the grid matrix[A[i].x][A[i].y] from sea to island. Return how many island are there in the matrix after each operator.You need to return an array of size K.

0 is represented as the sea, 1 is represented as the island. If two 1 is adjacent, we consider them in the same island. We only consider up/down/left/right adjacent.

Example

Example 1:

```
Input: n = 4, m = 5, A = [[1,1],[0,1],[3,3],[3,4]]
Output: [1,1,2,2]
Explanation:
0.  00000
    00000
    00000
    00000
1.  00000
    01000
    00000
    00000
2.  01000
    01000
    00000
    00000
3.  01000
    01000
    00000
    00010
4.  01000
    01000
    00000
    00011
```

Example 2:

```
Input: n = 3, m = 3, A = [[0,0],[0,1],[2,2],[2,1]]
Output: [1,1,2,2]
```

```
"""
Definition for a point.
class Point:
    def __init__(self, a=0, b=0):
        self.x = a
        self.y = b
"""

class Solution:
    """
    @param n: An integer
    @param m: An integer
    @param operators: an array of point
    @return: an integer array
    """
    def numIslands2(self, n, m, operators):
        # write your code here
        # matrix = [[0]*m for i in range(n)]
        countOfIsland = 0
        ans = []
        parent = [-1 for i in range(n*m)]
        rank = [-1 for i in range(n*m)]
        for point in operators:
            x = point.x
            y = point.y
            cellNumber = n*x+y
            if parent[cellNumber]!=-1:
                ans.append(countOfIsland)
                continue
            parent[cellNumber] = cellNumber
            rank[cellNumber] = 1
            countOfIsland+=1
            for dx,dy in ([1,0],[0,1],[-1,0],[0,-1]):
                xx = x+dx
                yy = y+dy
                cellDash = xx*n+yy
                if xx<0 or yy<0 or xx>=n or yy>=m or parent[cellDash]==-1:
                    continue
                lx = self.find(cellDash,parent)
                ly = self.find(cellNumber,parent)
                if lx!=ly:
                    if rank[lx]>rank[ly]:
                        parent[ly] = lx
                    elif rank[lx]<rank[ly]:
                        parent[lx] = ly
                    else:
                        parent[ly] = lx
                        rank[lx]+=1
                    countOfIsland-=1
            ans.append(countOfIsland)
        return ans

    def find(self,x,parent):
        if parent[x]==x:
            return x
        temp = self.find(parent[x],parent)
        parent[x] = temp
        return temp
```

We used Union find and not DFS because this is dynamic graph. For Dyamic graph we use Union Find