

### 3 坐标型动规

笔记本: DP Note

创建时间: 10/13/2019 3:14 PM

更新时间: 10/21/2019 2:45 PM

作者: tanziqi1756@outlook.com

机器人和金字塔 详见第0讲

机器人从左上角往右下角走

类似金字塔从顶往下走

## Leetcode 361 Bomb enemy

### 361. Bomb Enemy

难度 中等

8

8

收藏

分享

切换为中文

关注

题目描述

评论 (8)

题解 (8) <sup>New</sup>

提交记录

Given a 2D grid, each cell is either a wall 'W', an enemy 'E' or empty '0' (the number zero), return the maximum enemies you can kill using one bomb.

The bomb kills all the enemies in the same row and column from the planted point until it hits the wall since the wall is too strong to be destroyed.

**Note:** You can only put the bomb at an empty cell.

**Example:**

Input: `[["0","E","0","0"],["E","0","W","E"],["0","E","0","0"]]`

Output: 3

Explanation: For the given grid,

`0 E 0 0`

`E 0 W E`

`0 E 0 0`

Placing a bomb at (1,1) kills 3 enemies.

### 题目分析

九章算法

- 每个炸弹可以往四个方向传播爆炸力
- 我们可以分析一个方向，然后举一反三
- 即如果在一个空地放一个炸弹，最多向上能炸死多少敌人
- 可以直接枚举，即向上枚举到碰到墙为止
- 时间复杂度 $O(MN*M)$
- 用动态规划思想加速


	0	1	2	3
0	0	E	0	0
1	E	W	E	
2	0	E	0	0

暴力法 $O(N*N*2N)$

- 设  $Up[i][j]$  表示  $(i, j)$  格放一个炸弹向上能炸死的敌人数

$$Up[i][j] = \begin{cases} Up[i-1][j], & \text{如果}(i, j)\text{格是空地} \\ Up[i-1][j] + 1, & \text{如果}(i, j)\text{格是敌人} \\ 0, & \text{如果}(i, j)\text{格是墙} \end{cases}$$

- 初始条件：第0行的Up值和格子内容相关
  - $Up[0][j] = 0$ , 如果  $(0, j)$  格不是敌人
  - $Up[0][j] = 1$ , 如果  $(0, j)$  格是敌人

	0	1	2	3
0	0	E	0	0
1	E		W	E
2	0	E	0	0

### 361. Bomb Enemy

难度 中等

8

8

收藏

分享

切换为中文

关注

题目描述

评论 (8)

题解 (8) <sup>New</sup>

提交记录

执行结果: 通过 [显示详情](#)

执行用时: **12 ms** , 在所有 java 提交中击败了 **81.58%** 的用户

内存消耗: **39.8 MB** , 在所有 java 提交中击败了 **100.00%** 的用户

```

1  class Solution {
2      public int maxKilledEnemies(char[][] grid) {
3          if( grid.length == 0 || grid[0].length == 0) {
4              return 0;
5          }
6
7          int x = grid.length;
8          int y = grid[0].length;
9
10         // We can divide this problem into four subproblems and conquer
11         // If the bomb can only kill the upper enemies
12         int[][] up = new int[x][y];
13         int[][] down = new int[x][y];
14         int[][] left = new int[x][y];
15         int[][] right = new int[x][y];
16
17         // up
18         // initialize the first row
19         for( int j = 0; j < y; j++ ) {
20             if( grid[0][j] == 'E' ) {
21                 up[0][j] = 1;
22             }
23         }
24         // dp
25         for( int i = 1; i < x; i++ ) {
26             for( int j = 0; j < y; j++ ) {
27                 if( grid[i][j] == 'E' ) {
28                     up[i][j] += up[i-1][j] + 1;
29                 }
30                 else if( grid[i][j] == '0' ) {
31                     up[i][j] += up[i-1][j];
32                 }
33             }
34         }

```

```

36 // down
37 // initialize
38 for( int j = 0; j < y; j++ ) {
39     if( grid[x - 1][j] == 'E' ) {
40         down[x - 1][j] = 1;
41     }
42 }
43 // dp
44 for( int i = x - 2; i >= 0; i-- ) {
45     for( int j = 0; j < y; j++ ) {
46         if( grid[i][j] == 'E' ) {
47             down[i][j] += down[i+1][j] + 1;
48         }
49         else if( grid[i][j] == '0' ) {
50             down[i][j] += down[i+1][j];
51         }
52     }
53 }
54
55 // left
56 // initialize
57 for( int i = 0; i < x; i++ ) {
58     if( grid[i][0] == 'E' ) {
59         left[i][0] = 1;
60     }
61 }
62 // dp
63 for( int i = 0; i < x; i++ ) {
64     for( int j = 1; j < y; j++ ) {
65         if( grid[i][j] == 'E' ) {
66             left[i][j] += left[i][j-1] + 1;
67         }
68         else if( grid[i][j] == '0' ) {
69             left[i][j] += left[i][j-1];
70         }
71     }
72 }
73

```

```

74 // right
75 // initialize
76 for( int i = 0; i < x; i++ ) {
77     if( grid[i][y-1] == 'E' ) {
78         right[i][y-1] = 1;
79     }
80 }
81 // dp
82 for( int i = 0; i < x; i++ ) {
83     for( int j = y - 2; j >= 0; j-- ) {
84         if( grid[i][j] == 'E' ) {
85             right[i][j] += right[i][j+1] + 1;
86         }
87         else if( grid[i][j] == '0' ) {
88             right[i][j] += right[i][j+1];
89         }
90     }
91 }
92
93 // answer
94 int ans = 0;
95 for( int i = 0; i < x; i++ ) {
96     for( int j = 0; j < y; j++ ) {
97         if( grid[i][j] == '0' ) {
98             int temp = up[i][j] + down[i][j] + left[i][j] + right[i][j];
99             if( temp > ans ) {
100                 ans = temp;
101             }
102         }
103     }
104 }
105 return ans;
106 }
107
108 }

```