130. Surrounded Regions

Medium

644388FavoriteShare

Given a 2D board containing 'X' and 'O' (the letter O), capture all regions surrounded by 'X'.

A region is captured by flipping all 'O's into 'X's in that surrounded region.

Example:

X X X X  
X O O X  
X X O X  
X O X X

After running your function, the board should be:

X X X X  
X X X X  
X X X X  
X O X X

Explanation:

Surrounded regions shouldn’t be on the border, which means that any 'O' on the border of the board are not flipped to 'X'. Any 'O' that is not on the border and it is not connected to an 'O' on the border will be flipped to 'X'. Two cells are connected if they are adjacent cells connected horizontally or vertically.

class Solution {

public:

void solve(vector<vector<char>>& board) {

if(board.size()==0) return;

int rows=board.size();

int cols=board[0].size();

for(int i=0;i<cols;i++){

if(board[0][i]=='O') dfs(board,0,i);

}

for(int i=0;i<cols;i++){

if(board[rows-1][i]=='O') dfs(board,rows-1,i);

}

for(int i=0;i<rows;i++){

if(board[i][0]=='O') dfs(board,i,0);

}

for(int i=0;i<rows;i++){

if(board[i][cols-1]=='O') dfs(board,i,cols-1);

}

for(int i=0;i<board.size();i++){

for(int j=0;j<board[0].size();j++){

if(board[i][j]=='O') board[i][j]='X';

else if(board[i][j]=='A') board[i][j]='O';

}

}

}

void dfs(vector<vector<char>>& board, int i, int j){

if(i<0||j<0||i>=board.size()||j>=board[0].size()||board[i][j]!='O') return;

board[i][j]='A';

dfs(board,i,j+1);

dfs(board,i,j-1);

dfs(board,i+1,j);

dfs(board,i-1,j);

}

};

Success

[Details](https://leetcode.com/submissions/detail/211156764/)

Runtime: 32 ms, faster than 100.00% of C++ online submissions for Surrounded Regions.

Memory Usage: 14.6 MB, less than 58.74% of C++ online submissions for Surrounded Regions.