

**BỘ GIÁO DỤC VÀ ĐÀO TẠO  
TRƯỜNG: ĐH GIAO THÔNG VẬN TẢI  
KHOA: CNTT**



# **BÀI TẬP LỚN**

*Môn Học : Thuật toán và ứng dụng*

*Đề tài: Diện tích vùng số 1 lớn nhất*

*Topic: Unit Area of largest region of 1's*

Giáo viên hướng dẫn: Phạm Xuân Tích

Sinh viên thực hiện: Ngô Văn Khải \_ 191200552

Lớp: Công nghệ thông tin 2

Khóa: 60

Hà Nội, tháng 11 năm 2021

# Mục lục

1.	Nêu bài toán.....	3
2.	Nêu ý tưởng .....	3
3.	Mô phỏng tính toán bằng tay .....	3
4.	Code.....	4
5.	Đánh giá độ phức tạp.....	6

## 1. Nêu bài toán

- Cho một ma trận  $n \times m$  chỉ gồm 2 số 0 và 1. Tìm diện tích vùng số 1 lớn nhất.
- Vùng của một nhóm số 1 được kết nối theo 8 hướng (Theo chiều ngang, chiều dọc và đường chéo)

$1 \leq n, m \leq 500$

## 2. Nêu ý tưởng

**B1:** Ta duyệt lần lượt các vị trí có trong ma trận  $n \times m$ .

B2: Kiểm tra xem vị trí đó có bằng 1 và chưa đi qua hay không, nếu có đó ta sẽ đánh dấu vị trí là đã đi qua đồng thời khởi tạo biến đếm bằng 1.

B3: Duyệt 8 vị trí xung quanh của vị trí ở B2 và kiểm tra xem các vị trí đó có bằng 1 hay không nếu có thì ta đánh dấu vị trí đó là đã đi qua và tăng biến đếm thêm 1 đơn vị. Nếu các vị trí xung quanh của điểm ở B2 không thỏa mãn điều kiện ta tiến hành thực hiện so sánh max với biến đếm và cập nhật lại max rồi quay trở về B1.

### 3. Mô phỏng tính toán bằng tay

- Ma trận ban đầu:

1	0	0	0	1
0	0	0	1	1
1	1	0	0	0
0	1	1	1	1
0	0	0	0	0
0	1	0	1	1
0	0	1	1	0
1	0	0	0	1

- Ma trận sau khi tính toán

[illegible]

- Xét vị trí  $i = 0$  và  $j = 0 \Rightarrow$  Ta có  $a[0][0]$  bằng 1 và chưa đi qua. Ta đánh dấu điểm đã đi qua bằng cách gán  $a[0][0] = 0$  rồi xét các vị trí xung quanh của  $a[0][0]$  là 3 vị trí  $a[1][0]$ ,  $a[0][1]$  và  $a[1][1]$  thì cả 3 vị trí này đều có giá trị bằng 0 nên ta cập nhật lại  $max = 1$  (max khởi tạo bằng 0)

- Xét vị trí  $i = 0$  và  $j = 4 \Rightarrow$  Ta có  $a[0][4]$  bằng 1 và chưa đi qua. Ta đánh dấu điểm đã đi qua bằng cách gán  $a[0][4] = 0$  rồi xét các vị trí xung quanh của  $a[0][4]$  là 3 vị trí  $a[0][3]$ ,  $a[1][3]$  và  $a[1][4]$  thì có 2 vị trí  $a[1][3]$  và  $a[1][4]$  là có giá trị bằng 1 và chưa đi qua. Nên ta lần lượt duyệt 2 vị trí đó và đổi giá trị của 2 vị trí đó về 0. Nhận thấy là xung quanh cả 2 vị trí đó đều không đi được nữa nên ta cập nhật lại  $max = 3$  (Do có 3 vị trí liên nhau và max hiện tại đang bằng 1)

- Làm tương tự ta sẽ được kết quả là  $max = 6$  là vùng màu xanh dương và vùng màu xanh lá.

## 4. Code

- Code C++:

```
#include<iostream>
#include<vector>
#include<queue>
#include<utility>
#define x first
#define y second
using namespace std;
int n, m, **a;
pair<int, int> d[8] = {{-1, -1}, {-1, 0}, {0, -1}, {1, -1}, {-1, 1}, {0, 1}, {1, 0}, {1, 1}};
bool check(int i, int j, int n, int m) {
    if (i >= 0 and i < n and j >= 0 and j < m)
        return true;
    return false;
}
int sol(){
    int res = 0;
    queue<pair<int, int>>q;
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            if (a[i][j] == 1)
            {
                a[i][j] = 0;
                int cnt = 1;
                q.push({ i, j });
                while (!q.empty())
                {
                    int u = q.front().x;
                    int v = q.front().y;
                    q.pop();
                    for (int k = 0; k < 8; k++)
                    {
                        pair<int, int> z = { u + d[k].x, v + d[k].y };
                        if (check(z.x, z.y, n, m) and a[z.x][z.y] == 1)
                        {
                            a[z.x][z.y] = 0;
                            cnt++;
                            q.push({ z.x, z.y });
                        }
                    }
                }
            }
        }
    }
}
```

```

        res = max(res, cnt);
    }
}
return res;
}
int main()
{
    cin >> n >> m;
    a = new int* [n];
    for (int i = 0; i < n; i++) {
        a[i] = new int[m];
    }
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            cin >> a[i][j];
        }
    }
    int res = sol();
    cout << res;
    return 0;
}

```

- Code Python:

```

import queue

def check(i, j, n, m):
    if i >= 0 and i < n and j >= 0 and j < m:
        return True
    return False

d = [[-1,-1],[-1, 0],[0, -1],[1, -1],[-1, 1],[0, 1],[1, 0],[1, 1]]
def sol(a, n, m):
    res = 0
    q = queue.Queue(maxsize=m * n)
    for i in range(n):
        for j in range(m):
            if a[i][j] == 1:
                a[i][j] = 0
                cnt = 1
                q.put((i, j))
                while (False == q.empty()):
                    u, v = q.get()
                    for k in range(0, 8):
                        x, y = d[k]
                        z = (u + x, v + y)
                        if check(u + x, v + y, n, m) and a[u + x][v + y] == 1:
                            a[u + x][v + y] = 0
                            cnt+=1
                            q.put((u + x, v + y))
                res = max(res, cnt)
    return res

if __name__ == '__main__':
    n, m = map(int, input().split())
    a = []
    for i in range(0, n):
        a.append([])
        x = list(input().split())
        for j in range(0, m):

```

```

        a[i].append(int(x[j]))
    print(sol(a, n, m))

```

## 5. Đánh giá độ phức tạp

Hàm bool có độ phức tạp là:  $9 + 1 = 10$

```

bool check(int i, int j, int n, int m) {
    if (i >= 0 and i < n and j >= 0 and j < m)
        return true;
    return false;
}

```

Hàm sol có độ phức tạp:  $2 + 2 + n * m + 1 = n * m$

```

int sol(){
    int res = 0;
    queue<pair<int, int>>>q;
    for (int i = 0; i < n; i++)
    {
        for (int j = 0; j < m; j++)
        {
            if (a[i][j] == 1)
            {
                a[i][j] = 0;
                int cnt = 1;
                q.push({ i, j });
                while (!q.empty())
                {
                    int u = q.front().x;
                    int v = q.front().y;
                    q.pop();
                    for (int k = 0; k < 8; k++)
                    {
                        pair<int, int> z = { u + d[k].x, v + d[k].y };
                        if (check(z.x, z.y, n, m) and a[z.x][z.y] == 1)
                        {
                            a[z.x][z.y] = 0;
                            cnt++;
                            q.push({ z.x, z.y });
                        }
                    }
                }
                res = max(res, cnt);
            }
        }
    }
    return res;
}

```

Hàm Main có độ phức tạp:  $2 + 2 + (n + 2) * 3 + 3 * m * n + 3 + 1 = n * m$

```

int main()
{
    cin >> n >> m;
    a = new int* [n];
    for (int i = 0; i < n; i++) {
        a[i] = new int[m];
    }
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            cin >> a[i][j];

```

```
    }  
}  
int res = sol();  
cout << res;  
return 0;  
  
}
```

Tổng độ phức tạp là :  $10 + n * m + n * m = 10 + 2 * n * m = n * m$

=> Độ phức tạp của bài toán là:  $O(n * m)$