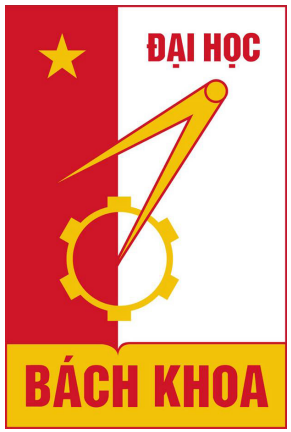
Thuật toán ứng dụng
Bài thực hành số 1.2: Cấu trúc dữ liệu

TS. Đinh Viết Sang, TA. Đặng Xuân Vương



Trường Đại học Bách khoa Hà Nội
Viện Công nghệ thông tin và Truyền thông

Ngày 20 tháng 12 năm 2020

Mục lục

Mục lục

02. LOCATE

- Cho T test, mỗi test gồm 2 bản đồ kích thước LxC, thể hiện cùng một địa điểm tại 2 thời điểm khác nhau.
- Mỗi bản đồ biểu diễn bởi các số 0, 1. 1 ứng với vị trí có vật thể bay (có thể là chim hoặc chiến đấu cơ), 0 là vị trí không có vật thể nào.
- Biết rằng tất cả các chiến đấu cơ trên bản đồ di chuyển theo cùng một quy luật.
- Tính số chiến đấu cơ tối đa có thể xuất hiện trong cả hai bản đồ.
- Biết rằng: $1 \leq L, C \leq 1000$. Tổng số các số 1 không quá 10000.

Thuật toán

- Tổng số các số 1 không quá 10000 nên có thể lưu toạ độ các đỉnh 1 của mỗi trạng thái vào 2 mảng.
- So sánh từng cặp đỉnh của mỗi mảng để đếm số khoảng cách có thể.

Code

```
const int N = 1010;

int n, m;
vector<pair<int, int>> a, b;
int cnt[N * 2][N * 2];

int main() {
    //freopen("test.in", "r", stdin);
    ios_base::sync_with_stdio(0); cin.tie(0);
    int tc;
    cin >> tc;
    while (tc--) {
        memset(cnt, 0, sizeof cnt);
        cin >> n >> m;
        a.clear(); b.clear();
        ...
```

Code

```
...
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= m; j++) {
            int u;
            cin >> u;
            if (u == 1) a.push_back({i, j});
        }
    }
    for (int i = 1; i <= n; i++) {
        for (int j = 1; j <= m; j++) {
            int u;
            cin >> u;
            if (u == 1) b.push_back({i, j});
        }
    }
...
```

Code

```
        ...
        for (auto u : a) {
            for (auto v : b) {
                pair<int, int> w =
                    {u.first - v.first + N,
                     u.second - v.second + N};
                cnt[w.first][w.second]++;
            }
        }
        int res = 0;
        for (int i = 0; i < N * 2; i++) {
            res = max(res,
                  *max_element(cnt[i], cnt[i] + N * 2));
        }
        cout << res << '\n';
    }
    return 0;
}
```

Mục lục

02. WATERJUG-BFS

- Có hai bình đựng nước, một bình có dung tích a lít, bình còn lại dung tích b lít. Tìm cách để có thể đong được chính xác c lít nước.
- Đầu vào: Dòng đầu tiên cho một số nguyên $T \leq 1000$ là số lượng test, với mỗi test sẽ gồm 3 số nguyên dương $a, b, c \leq 10^3$

Thuật toán

Nhận xét

- Kí hiệu (X, Y) tương ứng với bình 1 có X lít nước, bình 2 có Y lít nước, với mỗi trạng thái như vậy ta có thể thực hiện các cách đổ sau:
  1. Làm trống 1 bình, $(X, Y) \rightarrow (0, Y)$, đổ hết bình 1.
  2. Đổ đầy một bình, $(0, 0) \rightarrow (X, 0)$, đổ đầy bình 1.
  3. Đổ nước từ một bình sang một bình còn lại cho đến khi một trong hai bình đầy hoặc hết nước, $(X, Y) \rightarrow (X + d, Y - d)$

Thuật toán

1. Bắt đầu với trạng thái (0, 0), chúng ta chạy thuật toán duyệt theo chiều rộng (BFS) với mỗi đỉnh duyệt là một trạng thái đã có, và các đỉnh khác sẽ được sinh ra bằng 3 cách đổ nước bên trên từ những đỉnh trước đó.
2. Thuật toán sẽ dừng khi đã tìm được trạng thái có chứa c lít trong đó hoặc đã duyệt hết các trạng thái có thể sinh ra.

Code

```
#include <bits/stdc++.h>
#define pii pair<int, int>
#define mp make_pair
using namespace std;

// level is number of steps to (X,Y) State
map<pii, int> level;
// queue to maintain states
queue<pii> q;

// Changing state of jugs from (u1, u2) to (a,b)
void Pour(int a, int b, pii u){
    // if this state isn't visited
    if (level[{ a, b }] == 0)
    {
        // Save
        q.push({a, b });
        level[{a, b}] = level[{u.first, u.second}] + 1;
    }
```

Code

```
void BFS(int a, int b, int target)
{
    // Map is used to store the states, every
    bool isSolvable = false;
    level.clear();
    q = queue<pii>();
    // queue to maintain states
    // Initialing with initial state
    q.push({ 0, 0 });
    level[{0,0}] = 1;

    while (!q.empty()) {
        pii u = q.front(); // current state
        q.pop(); // pop off used state
        // if we reach solution state
        if(u.first==target||u.second==target){
            isSolvable = true;
            cout<<level[{u.first, u.second}]-1;
            break;
```

Code

```
        Pour(u.first, b, u); // fill Jug2
        Pour(a, u.second, u);// fill Jug1
        Pour(u.first, 0, u);// Empty Jug2
        Pour(0, u.second, u);// Empty Jug1

        for (int ap=0; ap<=max(a, b); ap++) {
            // pour amount ap from Jug2 to Jug1
            int c = u.first + ap;
            int d = u.second - ap;
            // check if this state is possible
            if ((c == a && d >= 0)
                    || (d == 0 && c <= a))
                Pour(c, d, u);
            // Pour amount ap from Jug 1 to Jug2
            c = u.first - ap;
            d = u.second + ap;
            // check if this state is possible
            if ((c == 0 && d <= b)
                    || (d == b && c >= 0))
```

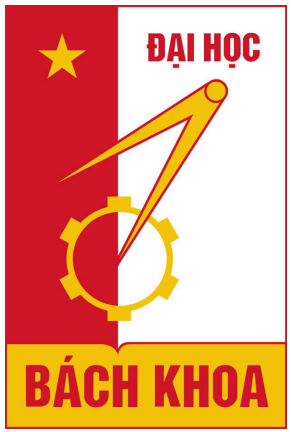
Code

```
        }
    }

    // No, solution exists if ans=0
    if (!isSolvable)
        cout << -1 << endl;
}
int main()
{
    int T;
    cin >> T;
    while (T > 0)
    {
        int Jug1, Jug2, target;
        cin >> Jug1 >> Jug2 >> target;
        BFS(Jug1, Jug2, target);
        T --;
    }
    return 0;
```

Thuật toán ứng dụng
Bài thực hành số 1.2: Cấu trúc dữ liệu

TS. Đinh Viết Sang, TA. Đặng Xuân Vương



Trường Đại học Bách khoa Hà Nội
Viện Công nghệ thông tin và Truyền thông

Ngày 20 tháng 12 năm 2020