



CHUYÊN MỤC

- Cấu trúc dữ liệu
 - BIT
 - Deque
 - Hashing
 - Heap
 - IT
 - Queue
 - Set
 - Stack
- Cloud Platform
 - Server
- Khác
- Kiểm tiền Online
- Lập trình Mobile
 - Lập trình Android
- Lập trình Web
 - CSS
 - HTML
- MongoDB
- Ngôn ngữ lập trình
 - .NET
 - C++
 - Go
 - Java
 - MySQL
 - NodeJS
 - Pascal
 - PHP
- OJ
 - Codeforces
 - Kattis
 - PTIT
 - SPOJ
- Phương pháp
 - Đệ quy có nhớ
 - Nén số (rời rạc hóa mảng)
 - Nhân ma trận
 - Quy hoạch động
 - Quy hoạch động trạng thái
- Thủ thuật máy tính
- Thư viện
 - Đệ thi
 - Giải thuật
 - Tài liệu
- Thuật toán
 - Bitmask
 - Duyệt phân tập
 - Đệ quy
 - Đồ thị
 - BFS



CẢNH BÁO: SAU KHÓA HỌC NÀY BẠN SẼ CUỒNG EXCEL

Chỉ cần 30 phút mỗi ngày ngay tại nhà bạn sẽ giỏi Excel sau 1 tuần

Thuật toán Kruskal tìm cây khung nhỏ nhất

05/10/2017 BY KHANH NGUYEN [LEAVE A COMMENT](#)

Định nghĩa cây khung – Cây khung nhỏ nhất

Cho đồ thị vô hướng, cây khung (spanning tree) của đồ thị là một cây con chứa tất cả các đỉnh của đồ thị. Nói cách khác, cây khung là một tập hợp các cạnh của đồ thị, không chứa chu trình và kết nối tất cả các đỉnh của đồ thị.

Trong đồ thị có trọng số, cây khung nhỏ nhất là cây khung có tổng trọng số các cạnh nhỏ nhất. Định nghĩa tương tự với cây khung lớn nhất.

Thuật toán Kruskal

Ban đầu mỗi đỉnh là một cây riêng biệt, ta tìm cây khung nhỏ nhất bằng cách duyệt các cạnh theo trọng số từ nhỏ đến lớn, rồi hợp nhất các cây lại với nhau.

Cụ thể hơn, giả sử cạnh đang xét nối 2 đỉnh u và v nếu 2 đỉnh này nằm ở 2 cây khác nhau thì ta thêm cạnh này vào cây khung, đồng thời hợp nhất 2 cây chứa u và v .

Để thực hiện thao tác trên một cách nhanh chóng, ta sử dụng cấu trúc [Disjoint Set](#), độ phức tạp của toàn bộ thuật toán là $O(M \log M)$ với M là số cạnh.

Cài đặt

Đoạn code dưới cài đặt thuật toán Kruskal, có thể dùng để nộp bài

QBMST.

```
#include <iostream>
#include <vector>
#include <algorithm> // Hàm sort
using namespace std;

// Cấu trúc để Lưu cạnh đồ thị,
// u, v là 2 đỉnh, w là trọng số cạnh
struct edge {
    int u, v, w;
};

// Hàm so sánh để dùng trong hàm sort ở dưới
bool cmp(const edge &a, const edge &b) {
    return a.w < b.w;
}

// Số đỉnh tối đa trong đề bài
#define N 10005

// 2 mảng sử dụng trong Disjoint Set
int cha[N], hang[N];

// Tìm xem u thuộc cây nào
int find(int u) {
    if (cha[u] != u) cha[u] = find(cha[u]);
    return cha[u];
}

// Hợp nhất 2 cây chứa u và v,
// Trả về false nếu không thể hợp nhất
bool join(int u, int v) {
    u = find(u); v = find(v);
    if (u == v) return false;
    if (hang[u] == hang[v]) hang[u]++;
    if (hang[u] < hang[v]) cha[u] = v;
}
```

- Cặp ghép
- DFS
- Dijkstra
- Floyd
- Kruskal
- LCA
- Luồng
- Prim
- Tree
- Hình học
- Convex Hull
- Khác
- Tham lam
- Tìm kiếm nhị phân
- Toán học
- Sàng nguyên tố
- Vết cạn
- Xử lý số lớn
- Toán rời rạc
- Trí tuệ nhân tạo
- Học máy

THEO DÕI CHÚNG TÔI TRÊN FACEBOOK

Like fanpage để nhận thông tin mới nhất trên newsfeed

<https://www.facebook.com/yeulaptrinh.pw>

```

else cha[v]=u;
return true;
}

int main() {
// Thêm dòng này để cin, cout chạy nhanh
ios::sync_with_stdio(false); cin.tie(0);

// Nhập vào số đỉnh và số cạnh
int n, m; cin >> n >> m;

// Nhập danh sách các cạnh
vector<edge> edges(m);
for (edge &e: edges) cin >> e.u >> e.v >> e.w;

// Sắp xếp lại các cạnh theo trọng số tăng dần
sort(edges.begin(), edges.end(), cmp);

// Khởi tạo cấu trúc Disjoint Set
for (int i=1; i<=n; i++) {
    cha[i] = i;
    hang[i] = 0;
}

// Lưu tổng trọng số các cạnh trong cây khung nhỏ nhất
int mst_weight = 0;

// Duyệt qua các cạnh theo thứ tự đã sắp xếp
for (edge &e: edges) {
    // Thử hợp nhất 2 cây chứa u và v
    if (join(e.u, e.v)) {
        // Hợp nhất thành công, ta thêm e và kết quả
        mst_weight += e.w;
    }
}

// Xuất kết quả
cout << mst_weight;
return 0;
}

```

GIẢI THUẬT , KRUSKAL MỨC ĐỘ: NORMAL

ên dùng



Khóa Học Laravel 5 Từ A-Z

Vietpro Academy

Lập Trình Web Thi
Tế Dạng Laravel
Là
Lar

ak Your Mind

CẢNH BÁO: SAU KHÓA HỌC NÀY SẼ BỊ CUỒNG EX

Chỉ cần 30 phút mỗi ngày ngay
bạn sẽ giỏi Excel sau 1 tu:

edit mail

Tìm Hiểu I

CHỌN MỤC BẠN MUỐN XEM!

basic bit cặp ghép cơ bản code

codeforces CTDI đáp án day con day con tang

DEMSO spoj deque de quy de thi dfs dhfrbus

dhlock spoj dhloco dijkstra do thi duyen hai

duyet duyet phan tap easy GRAPH_ spoj

hard HBTLCa spoj heap heap max hình học

Name *

Email *

Website

PHẢN HỒI

hsggg ican4 IT kruskal lca liq lis luong
 normal PreVOI QHĐ quite
 hard spoj tknp trình soạn thảo c++

BÀI VIẾT MỚI

- Giải thuật tìm kiếm Fibonacci
- Heroku là gì?
- GSON là gì?
- Cài đặt NodeJS và một số lỗi thường gặp
- Giới thiệu về MongoDB
- COMNET – spoj
- Tổng hợp tài liệu, đề thi môn Cơ Nhiệt
- LUBENICA – spoj

BÌNH LUẬN MỚI

- hieu4 trong SEQ198 – spoj
- Nguyễn Ngọc Trung trong Top 5 trường đại học tốt nhất để học công nghệ thông tin
- Phạm Văn Khánh trong SEQ198 – spoj
- Hoàng trong PBCDEM – SPOJ
- vũ long trong LINEGAME – SPOJ
- vũ long trong LINEGAME – SPOJ
- INFORMAC – spoj trong Tổng hợp tài liệu về thuật toán cặp ghép

QUẢN LÝ BLOG

- Đăng kí
- Đăng nhập
- RSS cho bài viết
- Dòng thông tin các phản hồi.
- WordPress.org

Đây là kho tri thức mở, bất kỳ ai cũng có thể viết bài về bất kỳ chủ đề gì.

Hãy [đăng ký thành viên](#) để có thể viết bài.

Đôi điều về YeulapTrinh.pw

Website được viết bởi nhiều thành viên trên mọi miền tổ quốc với mục đích chia sẻ, trao đổi, giúp đỡ lẫn nhau trong học lập trình. Hi vọng được sự ủng hộ của các bạn đọc thân mến <3

Các chủ đề:

- Thuật toán
- Cấu trúc dữ liệu
- Ngôn ngữ lập trình
- Lập trình Web
- Thủ thuật máy tính
- Kiểm tiền Online

Sử dụng Website

Viết bài:

Để viết bài bạn phải đăng ký thành viên. Bài viết bạn viết sẽ được kiểm duyệt và đăng lên trang Web

Mọi vấn đề thắc mắc, đóng góp xin liên hệ:

yeulaptrinh.pw@gmail.com