Bài tập SamSung phần B

8)

Thuật toán không đảm bảo kết quả cho một thứ tự topo vì vd ở các đỉnh ở khoảng cách n hoàn toàn có thể có đường đi đến đến 1 đỉnh ở khoảng cách bé hơn n. Mà theo thuật toán thì các đỉnh ở khoảng cách bé hơn n được liệt kê trước.

10)

+)Mô tả thuật toán tuyến tính tính thành phần liên thông mạnh chứa đồ thị v từ trước:

B1: DFS bắt đầu từ v, tạo 1 stack để thêm các đỉnh duyệt vào theo thứ tự.

B2: tạo ra một đồ thị G’ với các cạnh đổi hướng với đồ thị ban đầu.

B3: lấy lần lượt các đỉnh trong stack để dfs nó trên G’ nếu có v thì đó chính là thành phần liên thông cần tìm.

+)Mô tả thuật toán thời gian bậc hai đơn giản tính số thành phần liên thông mạnh

B1: DFS tất cả các đỉnh trên đồ thị G rồi lưu vào 1 stack

B2: xây dựng đồ thị G’ với các cạnh đổi hướng

B3: Pop các đỉnh trong stack ở b1 và gọi dfs trên đồ thị G’

#include<iostream>

#include<vector>

#include<string.h>

using namespace std;

bool visited[100];

vector <int> adj[100];

void dfs(int x)

{

visited[x]= true;

for (int it : adj[x])

{

if (!visited[it])

{

dfs(it);

}

}

}

void conectedComponents(int x)

{

int ans = 0;

for (int i = 1;i<=x;i++)

{

if(!visited[i])

{

ans++;

dfs(i);

}

}

cout << ans;

}

int main()

{

int n,m;

cin>> n >> m;

for (int i=0;i<m;i++)

{

int x,y; cin>> x >> y;

adj[x].push\_back(y);

adj[y].push\_back(x);

}

memset(visited,false,sizeof(visited));

conectedComponents(n);

}

11)

B1:Thực hiện thuật toán sắp xếp topo đưa các cạn vào 1 mảng

B2: Duyệt từng đỉnh từ mảng. với mỗi đỉnh kiểm tra trong đó xem nếu có đỉnh nào đứng liền sau trong thứ tự tôp hay không nếu có tiếp tục duyệt nếu không thì ta có thể kết luận đồ thị không có đường đi hamilton.

13)

Giả sử với mọi số nguyên k nhỏ hơn V, số lượng đồ thị có hướng với k đỉnh là |G(k)| = 2^(k choose 2).

Bây giờ, chúng ta thêm 1 đỉnh vào đồ thị có k đỉnh trước đó để tạo ra đồ thị mới có V = k + 1 đỉnh.

Với mỗi trường hợp thêm 1 đỉnh vào đồ thị có k đỉnh (có |G(k)| trường hợp), lại có 2^k cách để nối đỉnh mới vào đồ thị trước đó.

Do đó, khi thêm 1 đỉnh vào, chúng ta có tổng cộng 2^(k choose 2) \* 2^k cách để xây dựng đồ thị mới.

Tổng số đồ thị mới cho V = k + 1 là 2^(k + 1 choose 2) cách.