HUST

ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

ONE LOVE. ONE FUTURE.

ĐẠI HỌC
BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY
OF SCIENCE AND TECHNOLOGY

Applied Algorithm Lab

Strongly connected component

ONE LOVE. ONE FUTURE.

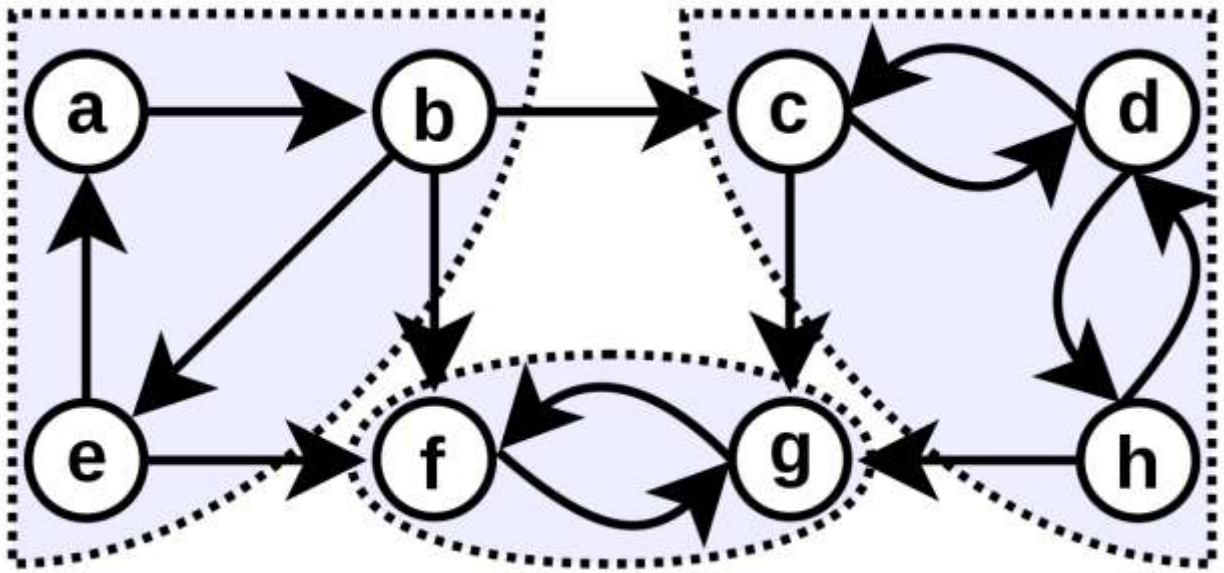
Strongly connected component

- Count #strongly connected components in a directed graph
  - Definition: A subset with maximal number of vertices that between 2 arbitrary vertices, there exists a path from a vertex to the other and vice versa.

  Thành phần liên thông mạnh là một tập con tối đa các đỉnh sao cho giữa 2 đỉnh bất kỳ luôn có đường đi từ đỉnh này đến đỉnh kia và ngược lại.
- **Input:** Edge list
  - Line 1 contains N, M
  - M lines follow, containing a pair of 2 integers a, b which is an undirected edge from a to b.
- **Output:**
  - Number of strongly connected components.

Strongly Connected Components

- BFS and DFS can find all connected component in undirected graph.
- However, in directed graph, finding all strongly connected components is not trivial.
- Can we use DFS tree to find all strongly connected components ?



Strongly connected component

- Idea to solve: DFS on residual graph. Algorithm:
  - Run DFS on G -> compute the finishing time f(v) of each node v of G
  - Build residual graph G^T of G: reverse direction of all edges
  - Run DFS on G^T:
    - the nodes are considered in a decreasing order of finishing time f:
    - Each run DFS(u) will visit all nodes of the strongly connected component containing u
  - Number of connected components on G = number of times call DFS on residual graph of G

Strongly connected component - Implementation

```cpp
// Đọc thêm tại https://cp-algorithms.com/graph/strongly-connected-components.html
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;
const int maxN = 1e5 + 5;
int n, m;
vector<int> adj[maxN], adj_trans[maxN];
vector<int> toposort;
bool visited[maxN];
int numSCC = 0;
// Tạo mảng toposort, trong đó các nút hiện đang được xếp tăng dần theo t_out
// (xem định nghĩa trong link bên trên)
void dfs1(int node){
    if (visited[node])
        return;
    visited[node] = true;
    for (auto u: adj[node])
        dfs1(u);
    toposort.push_back(node);
}
```

Strongly connected component - Implementation

```
23  // Tìm thành phần liên thông mạnh ứng với node
24  void dfs2(int node){
25      if (visited[node])
26          return;
27      visited[node] = true;
28      for (auto u: adj_trans[node])
29          dfs2(u);
30  }
```

Strongly connected component - Implementation

```
32  int main(){
33      cin >> n >> m;
34      for (int i = 0; i < m; i++){
35          int a, b;
36          cin >> a >> b;
37          adj[a].push_back(b);
38          adj_trans[b].push_back(a);
39      }
40      for (int i = 1; i <= n; i++)
41          if (!visited[i])
42              dfs1(i);
43      for (int i = 1; i <= n; i++)
44          visited[i] = false;
45      reverse(toposort.begin(), toposort.end());
46      for (int i = 0; i < n; i++){
47          if (!visited[toposort[i]]){
48              dfs2(toposort[i]);
49              numSCC++;
50          }
51      }
52      cout << numSCC << endl;
53      return 0;
54  }
```

THANK YOU !