|  |  |
| --- | --- |
| Buổi 06 | **ỨNG DỤNG THUẬT TOÁN DFS (tiếp)** |

**Bài 4** : **Đồ thị phân đôi**

Cho đồ thị vô hướng 𝐺 = (𝑉, 𝐸) có 𝑛 đỉnh (𝑛 ≤ 105 ), các đỉnh được đánh số từ 1 đến 𝑛. 𝐺 được gọi là đồ thị phân đôi nếu chúng ta có thể chia tập đỉnh 𝑉 thành 2 tập đỉnh 𝑉1, 𝑉2 không giao nhau sao cho các cạnh của 𝐺 nối một đỉnh trong 𝑉1 với một đỉnh trong 𝑉2 (không có cạnh nối hai đỉnh trong cùng một tập đỉnh). Hãy kiểm tra đồ thị 𝐺 có thể phân đôi không.

|  |  |  |
| --- | --- | --- |
| Đồ thị | Input : Bipartite.txt | Output |
| 1. 1   2 3  4 5 | 6  1 3 5  0 2 4  1 3 5  0 2 4  1 3 5  0 2 4 | Trả lời :  Đồ thị phân đôi hay không ? |

*Hướng dẫn* :

* Dùng một array color[] để tô màu cho các đỉnh với 2 màu 0 và 1. Ban đầu color[i] = -1, ∀i
* Trong class AdjList.cs viết 2 phương thức sau :

public void Bipartite()

{

Khởi tạo visited[] và gán giá trị ban đầu : visited[i] = false, ∀i

Khởi tạo color[] và gán giá trị ban đầu : color[i] = -1 , ∀i

Chọn một đỉnh bất kỳ (giả sử chọn đỉnh 0) và tô màu 1 : color[0] = 1;

Nếu (**IsBipartite**(0) == true) 🡪 "Đồ thị 2 phía"

Ngược lại : “Không phải Đồ thị 2 phía"

}

public bool IsBipartite(int s)

{

Duyệt các đỉnh u kề với s : (int u in v[s])

{

Nếu u chưa duyệt

{

Đánh dấu duyệt u

Tô màu cho u khác màu với s : color[u] = 1 - color[s];

Nếu (!IsBipartite(u)) // Dừng đệ qui khi gặp lỗi : return false

return false;

}

Ngược lại, Nếu 2 đỉnh kề nhau s và u cùng màu :

return false;

}

return true;

}

**Bài 5** : **kiểm tra đồ thị có chu trình**

|  |  |  |
| --- | --- | --- |
| Đồ thị | Input : Cycle.txt | Output |
|  | 7  1 4  2  3  1  5 6  3 5 | Trả lời :  Đồ thị có chu trình hay không? |

*Hướng dẫn* :

* Trong class AdjList.cs khai báo thêm thành phần : int[] color; // Dùng tô màu
* Dùng giải thuật sau :

Bước 1 : Khởi tạo: Ban đầu các đỉnh có màu 0 : color[i] = 0

Bước 2 : Với mỗi đỉnh 𝑖 có color[i] = 0 ta chạy DFS tại đỉnh 𝑖. Trong DFS(i) :

* Đánh dấu color[i] = 1 nếu bắt đầu thăm đỉnh 𝑖
* Đánh dấu color[i] = 2 nếu đã thăm xong đỉnh 𝑖
* Nếu DFS di chuyển đến đỉnh 𝑘 có color[𝑘] = 1 thì chúng ta có 1 chu trình
* Trong class AdjList.cs viết 2 phương thức :

// Khai báo môi trường ban đầ và gọi DFS

public void DFSTestCycle()

{

Khởi tạo và gán giá trị ban đầu cho color[] : color[i] = 0,∀i

Duyệt từng đỉnh i (0..n-1) và nhận phản hồi từ DFSCycle(i) cho mỗi đỉnh

Nếu DFSCycle(i) = true (-> có chu trình)

Xuất : " Đồ thị có chu trình"

Dừng xét : return;

Xuất : " Đồ thị không có chu trình"

}

// phương thức DFS xét chu trình

public bool DFSCycle(int s)

{

Đang duyệt, tô màu 1 cho đỉnh s : color[s] = 1;

Duyệt tiếp các đỉnh kề của s (int u in v[s])

{ Nếu chưa duyệt u (u có màu 0) :

gọi đệ qui với u : DFSCycle(u);

Ngược lại, nếu u có màu 1 :

trả về true (có chu trình)

}

// Ðến đây đã duyệt xong s -> tô màu 2 : color[s] = 2;

return false;

}

**Bài 6** : **Sắp xếp topo**

Cho **đồ thị có hướng, không có chu trình** 𝐺 = (𝑉, 𝐸) gồm 𝑛 đỉnh (𝑛 ≤ 105 ), các đỉnh được đánh số từ 1 đến 𝑛. Hãy sắp xếp các đỉnh sao cho nếu có cung (𝑢, 𝑣) thì 𝑢 phải đứng trước 𝑣 trong thứ tự đó.

|  |  |  |
| --- | --- | --- |
| Đồ thị | Input : Topo.txt | Output : xuất lên màn hình |
|  | 7  1 4  2  3  5 6    3 5 | **0 4 6 5 1 2 3** |

*Hướng dẫn* :

Thực hiện DFS trên đồ thị 𝐺, mỗi khi đỉnh được thăm xong ta đưa nó vào đầu danh sách liên kết (đều đó có nghĩa là những đỉnh kết thúc thăm càng muộn thì sẽ càng ở gần đầu danh sách liên kết). Danh sách liên kết thu được khi kết thúc DFS sẽ cho ta thứ tự cần tìm.

**Bài tập thêm** :

Một dự án chung cư gồm các công trình sau :

1/ San lấp mặt bằng

2/ Đào và xây móng, được thực hiện sau khi đã san lấp mặt bằng

3/ Làm đường nội bộ, thực hiện sau khi đào và xây móng

4/ Xây tòa nhà ở, sau khi đã san lấp mặt bằng và xây móng

5/ Làm tường rào và hệ thống cây xanh, sau khi đã làm đường nội bộ và xây xong tòa nhà

6/ Lắp đặt hệ thống điện nhà ở, sau khi đã xây xong tòa nhà

a/ Hãy xây dựng đồ thị có hướng để mô tả các công trình của dự án trên.

b/ Viết chương trình đáp ứng hai yêu cầu sau :

* Chứng tỏ đồ thị có hướng xây dựng ở câu a không có chu trình
* Cho biết thứ tự các công trình được thực thi sao cho thỏa các yêu cầu trên.