

HƯỚNG DẪN THUẬT TOÁN TÌM KIẾM TRÊN ĐỒ THỊ THEO BFS

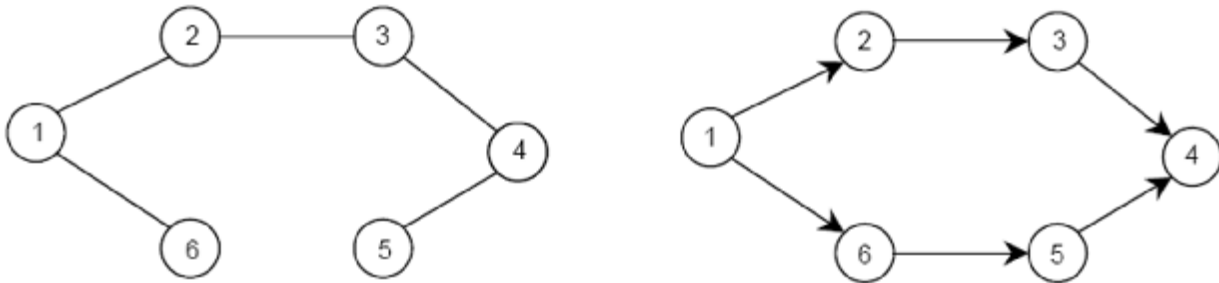
1. Lý thuyết

Cho đồ thị $G = (V, E)$. u và v là hai đỉnh của G . Một **đường đi (path)** độ dài k từ đỉnh u đến đỉnh v là dãy $(u = x_0, x_1, \dots, x_k = v)$ thỏa mãn $(x_i, x_{i+1}) \in E$ với $\forall i: (0 \leq i < k)$.

Đường đi nói trên còn có thể biểu diễn bởi dãy các cạnh: $(u = x_0, x_1), (x_1, x_2), \dots, (x_{k-1}, x_k = v)$.

Đỉnh u được gọi là đỉnh đầu, đỉnh v được gọi là đỉnh cuối của đường đi. Đường đi có đỉnh đầu trùng với đỉnh cuối gọi là **chu trình (Circuit)**, đường đi không có cạnh nào đi qua hơn 1 lần gọi là **đường đi đơn**, tương tự ta có khái niệm **chu trình đơn**.

Ví dụ: Xét một đồ thị vô hướng và một đồ thị có hướng dưới đây:



Trên cả hai đồ thị, $(1, 2, 3, 4)$ là đường đi đơn độ dài 3 từ đỉnh 1 tới đỉnh 4. Bởi $(1, 2)$, $(2, 3)$ và $(3, 4)$ đều là các cạnh (hay cung). $(1, 6, 5, 4)$ không phải đường đi bởi $(6, 5)$ không phải là cạnh (hay cung).

Một bài toán quan trọng trong lý thuyết đồ thị là bài toán duyệt tất cả các đỉnh có thể đến được từ một đỉnh xuất phát nào đó. Vấn đề này đưa về một bài toán liệt kê mà yêu cầu của nó là không được bỏ sót hay lặp lại bất kỳ đỉnh nào. Chính vì vậy mà ta phải xây

dùng những thuật toán cho phép **duyệt một cách hệ thống** các đỉnh, những thuật toán như vậy gọi là những thuật toán **tìm kiếm trên đồ thị** và ở đây ta quan tâm đến hai thuật toán cơ bản nhất: **thuật toán tìm kiếm theo chiều sâu (DFS_Depth First Search)** và **thuật toán tìm kiếm theo chiều rộng (BFS_ Breadth First Search)** cùng với một số ứng dụng của chúng.

2. Thuật toán tìm kiếm theo chiều sâu BFS _ Breadth First Search

Cho $G=(X, E)$ là một đồ thị gồm n đỉnh. Thuật toán tìm kiếm đường đi từ đỉnh S đến đỉnh F theo chiều sâu BFS dùng Queue như sau:

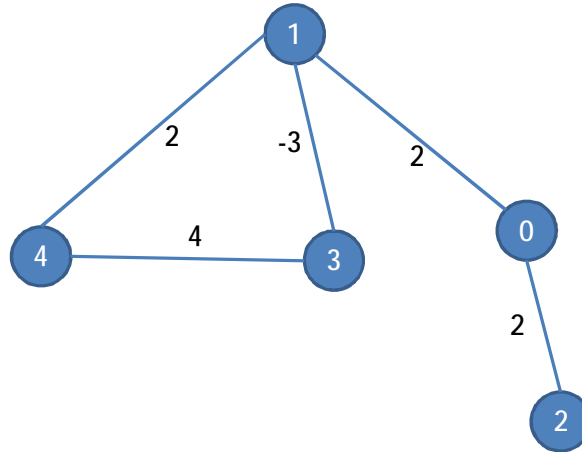
- Bước 1: Tạo ra 2 mảng 1 chiều **LuuVet** và **ChuaXet** với có kích thước là n .
- Bước 2: Gán
 - o **LuuVet**[i] = -1; // Vì ban đầu chưa chạy thuật toán nên các đỉnh i đều chưa có vết đi đến đó nên đặt giá trị là -1
 - o **ChuaXet**[i] = 0; // Vì ban đầu chưa chạy thuật toán nên các đỉnh i trong đồ thị g đều chưa được xét đến nên gán giá trị 0
- Bước 3: xây dựng với **QUEUE** các hàm cần thiết như
 - o **KhoiTaoQueue.**
 - o **DayGiaTriVaoQueue**
 - o **LayGiaTriRaKhoiQueue**
 - o **KiemTraQueueRong.**
- Bước 4: Bắt đầu duyệt và xét 1 đỉnh v nào đó (bắt đầu bước này đầu tiên thì đỉnh v là đỉnh S)
 - o **KhoiTaoQueue(Q);** // khởi tạo hàng đợi vì ban đầu thuật toán hàng đợi ta chưa có gì
 - o **DayGiaTriVaoQueue(Q,v);** // đẩy đỉnh v vào hàng đợi theo.
 - o Trong khi hàm đợi Q chưa rỗng thì tiến hành các bước sau:

- § LayGiaTriRaKhoiQueue(Q,v); //lấy phần tử đầu tiên trong hàng đợi ra và gán giá trị đó vào v.
- § ChuaXet[v] = 1; // gán lại giá trị đỉnh v trong mảng ChuaXet là 1, điều này có nghĩa là đỉnh v đã và đang được xét hay duyệt đến theo thuật toán.
- § Từ đỉnh v ta xem xét có đường đi đến đỉnh nào.
 - Nếu như tồn tại một đỉnh u mà đỉnh này chưa được xét đến tức ChuaXet[u] = - 1.
 - Ta tiến hành LuuVet[u] = v; // đánh dấu lại đỉnh u được đi đến từ đỉnh v trong quá trình duyệt theo thuật toán DFS.
 - DayGiaTriVaoQueue(Q,u); // đẩy đỉnh u vào hàng đợi Q.
 - Ngược lại hàng đợi Q rỗng thì dừng thuật toán.
- Bước 5: Sau khi tiến hành xong bước 3. Kiểm tra
 - if (ChuaXet[F] == 1) // sau khi thuật toán duyệt xong mà đỉnh F được xét hay duyệt đến thì nhãn của nó = 1 điều này có nghĩa là có đường đi từ S → F. Tiến hành xuất đường đi.
 - § Dựa vào mảng LuuVet tiến hành truy vết là xuất ra đường đi.
 - Ngược lại sau khi thuật toán duyệt xong mà đỉnh F chưa được xét hay duyệt đến thì nhãn của nó = 0 điều này có nghĩa là không có đường đi từ S → F. Thông báo không có đường đi từ S → F.

3. Ví dụ minh họa

Giả sử ta có đồ thị $G=(X, E)$ gồm có 5 đỉnh như sau

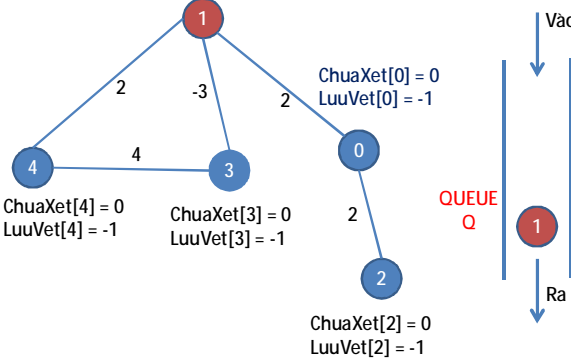
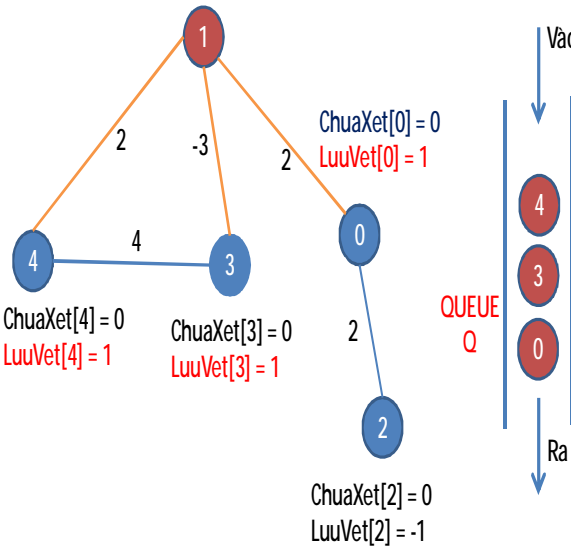
Hướng dẫn thuật toán tìm kiếm trên đồ thị BFS



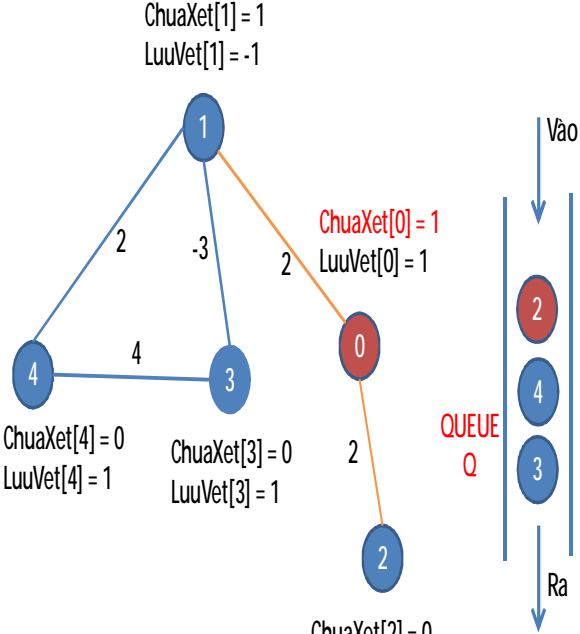
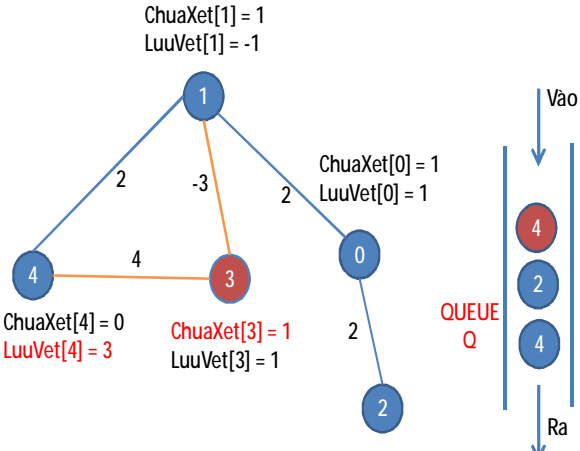
Thực hành thuật toán duyệt theo DFS tìm đường đi từ đỉnh 1 đến các đỉnh còn lại của đồ thị như sau:

Bước	Trạng thái đồ thị	Xử lý
1	<p>ChuaXet[1] = 0 LuuVet[1] = -1</p> <p>ChuaXet[4] = 0 LuuVet[4] = -1</p> <p>ChuaXet[3] = 0 LuuVet[3] = -1</p> <p>ChuaXet[0] = 0 LuuVet[0] = -1</p> <p>ChuaXet[2] = 0 LuuVet[2] = -1</p>	<p>Bước 1 và 2: Tạo ra 2 mảng 1 chiều LuuVet và ChuaXet với có kích thước là n.</p> <p>Và Gán:</p> <ul style="list-style-type: none">- LuuVet[i] = -1; // Vì ban đầu chưa chạy thuật toán nên các đỉnh i đều chưa có vết đi đến đó nên đặt giá trị là -1- ChuaXet[i] = 0; // Vì ban đầu chưa chạy thuật toán nên các đỉnh i trong đồ thị g đều chưa được xét đến nên gán giá trị 0

Hướng dẫn thuật toán tìm kiếm trên đồ thị BFS

2		<p>Xây dựng với QUEUE các hàm cần thiết như</p> <ul style="list-style-type: none"> - KhoiTaoQueue. - DayGiaTriVaoQueue - LayGiaTriRaKhoiQueue - KiemTraQueueRong. <p>Đẩy đỉnh 1 (tức là đỉnh S) vào hàng đợi Q</p>
3		<p>Kiểm tra hàng đợi Queue Q có rỗng ko. Hàng đợi Queue Q ko rỗng. Lấy điểm dưới cùng ra. Là đỉnh 1 và tiến hành xét đỉnh 1.</p> <ul style="list-style-type: none"> - ChuaXet[1] = 1; // gán lại giá trị đỉnh 1 trong mảng ChuaXet là 1, điều này có nghĩa là đỉnh 1 đã và đang được xét hay duyệt đến theo thuật toán. - Từ đỉnh 1 ta xem xét có đường đi đến đỉnh 0, 3, 4. Và 3 đỉnh này đều có giá trị ChuaXet là 0. <ul style="list-style-type: none"> - Đẩy 3 đỉnh 0, 3, 4 vào Queue Q. - LuuVet[0] = 1; LuuVet[3] = 1; LuuVet[4] = 1; // đánh dấu lại đỉnh 0, 3, 4 được đi đến từ đỉnh 1 trong quá trình duyệt theo thuật toán BFS <p>Sau đó ta quay lại đầu bước này</p>

Hướng dẫn thuật toán tìm kiếm trên đồ thị BFS

<p>4</p>	 <p>ChuaXet[1] = 1 LuuVet[1] = -1</p> <p>ChuaXet[0] = 1 LuuVet[0] = 1</p> <p>ChuaXet[2] = 0 LuuVet[2] = 0</p> <p>ChuaXet[4] = 0 LuuVet[4] = 1</p> <p>ChuaXet[3] = 0 LuuVet[3] = 1</p> <p>QUEUE Q</p> <p>Vào</p> <p>Ra</p>	<p>Kiểm tra hàng đợi Queue Q có rỗng ko. Hàng đợi Queue Q ko rỗng.</p> <p>Lấy điểm dưới cùng ra. Là đỉnh 0 và tiến hành xét đỉnh 0.</p> <ul style="list-style-type: none"> - ChuaXet[0] = 1; // gán lại giá trị đỉnh 0 trong mảng ChuaXet là 1, điều này có nghĩa là đỉnh 0 đã và đang được xét hay duyệt đến theo thuật toán. - Từ đỉnh 0 ta xem xét có đường đi đến đỉnh 1, 2. Và đỉnh 1 thì xét rồi (ChuaXet[1] = 1) nên bỏ qua, đỉnh 2 thì chưa xét nên: <ul style="list-style-type: none"> - Đẩy đỉnh 2 vào Queue Q. - LuuVet[2] = 0; // đánh dấu lại đỉnh 2 được đi đến từ đỉnh 0 trong quá trình duyệt theo thuật toán BFS <p>Sau đó ta quay lại đầu bước này</p>
<p>5</p>	 <p>ChuaXet[1] = 1 LuuVet[1] = -1</p> <p>ChuaXet[0] = 1 LuuVet[0] = 1</p> <p>ChuaXet[2] = 0 LuuVet[2] = 0</p> <p>ChuaXet[4] = 0 LuuVet[4] = 3</p> <p>ChuaXet[3] = 1 LuuVet[3] = 1</p> <p>QUEUE Q</p> <p>Vào</p> <p>Ra</p>	<p>Kiểm tra hàng đợi Queue Q có rỗng ko. Hàng đợi Queue Q ko rỗng.</p> <p>Lấy điểm dưới cùng ra. Là đỉnh 3 và tiến hành xét đỉnh 3.</p> <ul style="list-style-type: none"> - ChuaXet[3] = 1; - Từ đỉnh 3 ta xem xét có đường đi đến đỉnh 1, 4. Và đỉnh 1 thì xét rồi (ChuaXet[1] = 1) nên bỏ qua, đỉnh 4 thì chưa xét nên:

Hướng dẫn thuật toán tìm kiếm trên đồ thị BFS

		<ul style="list-style-type: none"> - Đẩy đỉnh 4 vào Queue Q. - $LuuVet[4] = 3$; // đánh dấu lại đỉnh 4 được đi đến từ đỉnh 3 trong quá trình duyệt theo thuật toán BFS <p>Sau đó ta quay lại đầu bước này</p>
6		<p>Kiểm tra hàng đợi Queue Q có rỗng ko. Hàng đợi Queue Q ko rỗng. Lấy điểm dưới cùng ra. Là đỉnh 4 và tiến hành xét đỉnh 4.</p> <ul style="list-style-type: none"> - $ChuaXet[4] = 1$; - Từ đỉnh 4 ta xem xét có đường đi đến đỉnh 1, 3. Và đỉnh 1, 3 thì xét rồi ($ChuaXet[1] = 1$, $ChuaXet[3] = 1$) nên bỏ qua. Vậy ko có đỉnh nào mới đưa vào QUEUE Q. <p>Sau đó ta quay lại đầu bước này.</p>
7		<p>Kiểm tra hàng đợi Queue Q có rỗng ko. Hàng đợi Queue Q ko rỗng. Lấy điểm dưới cùng ra. Là đỉnh 2 và tiến hành xét đỉnh 2.</p> <ul style="list-style-type: none"> - $ChuaXet[2] = 1$; - Từ đỉnh 2 ta xem xét có đường đi đến đỉnh 0. Và đỉnh 0 thì xét rồi ($ChuaXet[0] = 1$) nên bỏ qua. Vậy ko có đỉnh nào mới đưa vào QUEUE Q.

Hướng dẫn thuật toán tìm kiếm trên đồ thị BFS

		Sau đó ta quay lại đầu bước này.
8		<p>Kiểm tra hàng đợi Queue Q có rỗng ko. Hàng đợi Queue Q ko rỗng. Lấy điểm dưới cùng ra. Là đỉnh 4 và tiến hành xét đỉnh 4. Mà đỉnh 4 có giá trị ChuaXet[4] = 1; nên không cần phải xét nữa. Tới đây, ta thấy Queue Q rỗng nên thuật toán dừng.</p>

Sau khi duyệt BFS xong, bạn muốn biết đường đi từ đỉnh S (là đỉnh 1 trong trường hợp này) thì bạn căn cứ vào mảng **LuuVet** và **ChuaXet**.

LuuVet[0]	LuuVet[1]	LuuVet[2]	LuuVet[3]	LuuVet[4]
1	-1	0	1	3
ChuaXet[0]	ChuaXet[0]	ChuaXet[0]	ChuaXet[0]	ChuaXet[0]
1	1	1	1	1

Giả sử, bạn muốn tìm đường đi từ đỉnh 1 đến đỉnh 4. Thì đầu tiên căn cứ vào giá trị của mảng ChuaXet.

- Ta thấy ChuaXet[4] = 1 **è** có đường đi từ 1 đến 4.
- Ta sẽ truy vết đường đi dựa vào mảng LuuVet.
 - o Ta thấy LuuVet[4] = 3 **è** đỉnh 4 được đi đến từ đỉnh 3 trong đường đi duyệt theo DFS.
 - o Rồi qua xét LuuVet[3] = 1 **è** đỉnh 3 được đi đến từ đỉnh 1 trong đường đi duyệt theo DFS.
 - o Ta thấy đỉnh 1 là đỉnh xuất phát đường đi nên dừng.
- Vậy đường đi từ 1 **à** 4 là 4 **B** 3 **B** 1.

Tương tự hỏi bạn đường đi từ 1 **à** 2, 1 **à** 3 là đường nào ???

Chúc các bạn may mắn và học tốt môn này

GOOD LUCK TO U

-----HẾT-----