

Phần 2

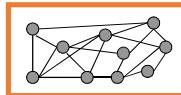
ĐỒ THỊ VÀ BIỂU DIỄN ĐỒ THỊ

Các ứng dụng thực tế của đồ thị

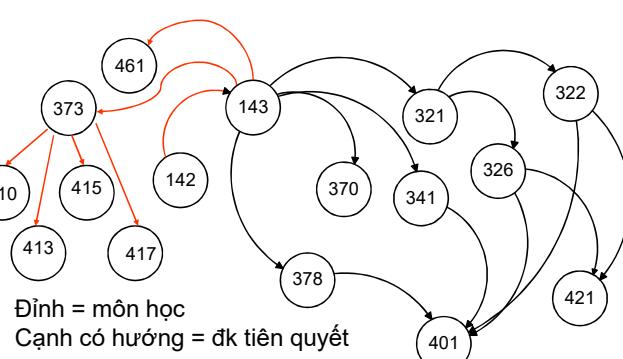
- Có tiềm năng ứng dụng trong nhiều lĩnh vực (Đồ thị có thể dùng để biểu diễn các quan hệ. Nghiên cứu quan hệ giữa các đối tượng là mục tiêu của nhiều lĩnh vực khác nhau).
- Ứng dụng trong mạng máy tính, mạng giao thông, mạng cung cấp nước, mạng điện,...) lập lịch, tối ưu hóa luồng, thiết kế mạch, quy hoạch phát triển...
- Các ứng dụng khác: Phân tích gen, trò chơi máy tính, chương trình dịch, thiết kế hướng đối tượng, ...

Đồ thị là gì?

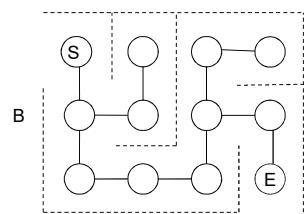
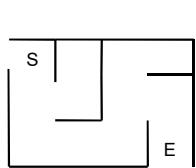
- Trong toán học đời thường hiểu là:
Bản vẽ hay sơ đồ biểu diễn dữ liệu như sau
Không phải cái này
Không phải cái ta muốn đề cập
- Trong toán rời rạc:
Đây là cấu trúc rời rạc có tính trực quan cao, rất tiện ích để biểu diễn các quan hệ.



Mối liên hệ giữa các môn học



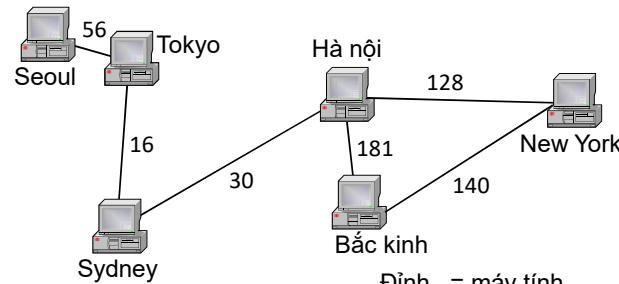
Biểu diễn mê cung



Đỉnh = phòng
Cạnh = cửa thông phòng hoặc hành lang

Truyền thông trong mạng máy tính

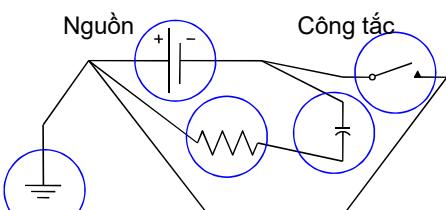
(Information Transmission in a Computer Network)



Đỉnh = máy tính
Cạnh = tốc độ truyền thông

Biểu diễn mạch điện

(Electrical Circuits)



Đỉnh = nguồn, công tắc, điện trở, ...
Cạnh = đoạn dây nối

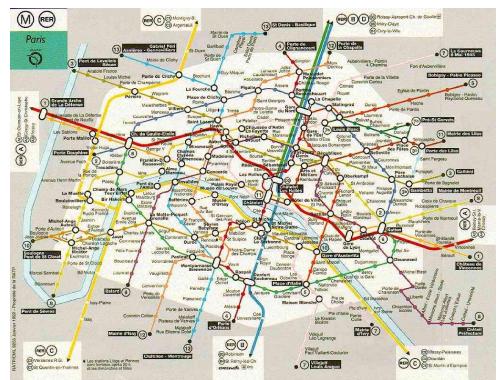
Luồng giao thông trên xa lộ

(Traffic Flow on Highways)

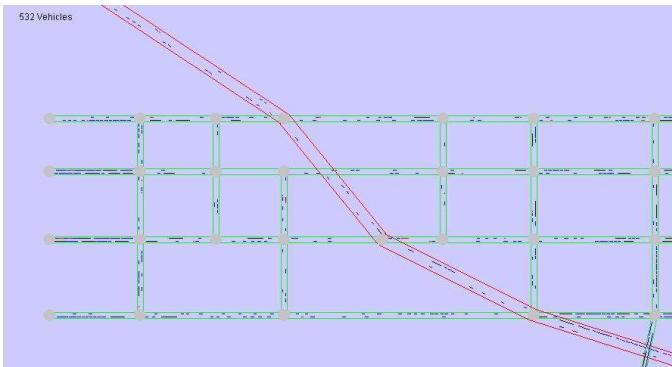


Đỉnh = thành phố
Cạnh = lượng xe cộ trên
tuyến đường cao tốc kết
nối giữa các thành phố

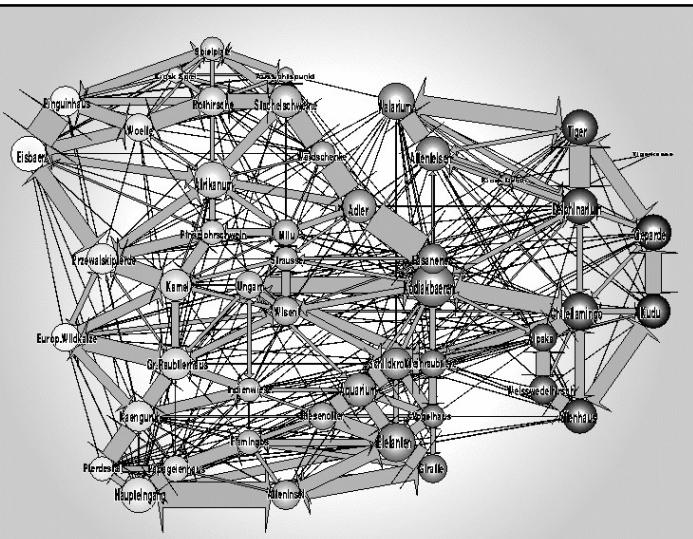
Mạng xe buýt

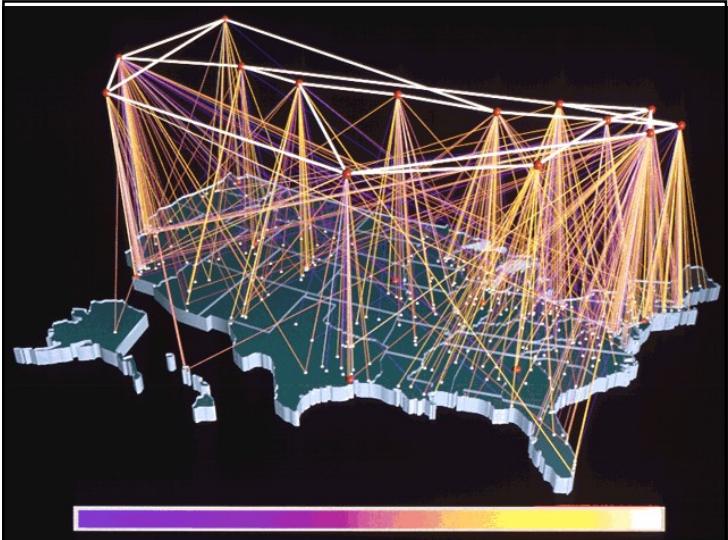
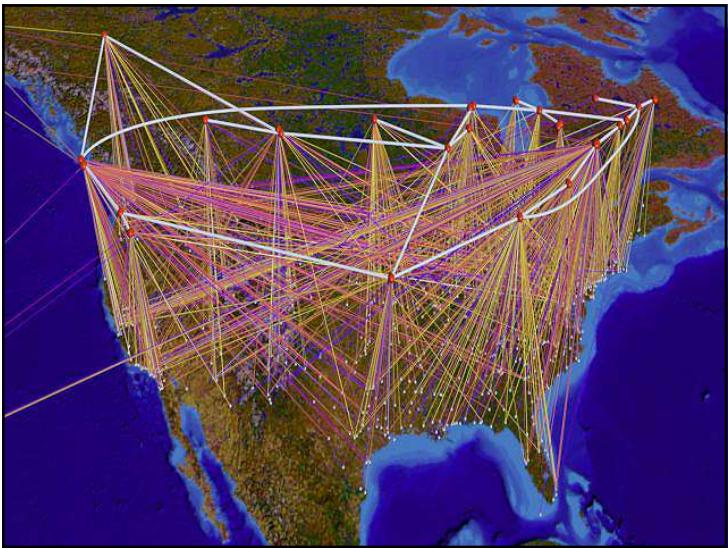


Sơ đồ đường phố



Mạng tàu điện ngầm





Chương 1 CÁC KHÁI NIỆM CƠ BẢN

- 1.1. Đồ thị trong thực tế
- 1.2. Các loại đồ thị**
- 1.3. Độ của đỉnh
- 1.4. Đồ thị con
- 1.5. Đồ thị đẳng cấu
- 1.6. Đường đi và chu trình
- 1.7. Tính liên thông
- 1.8. Một số loại đồ thị đặc biệt
- 1.9. Tô màu đồ thị

15

Đồ thị vô hướng

(Undirected Graphs)

Định nghĩa. Đơn (đa) đồ thị vô hướng $G = (V, E)$ là cặp gồm:

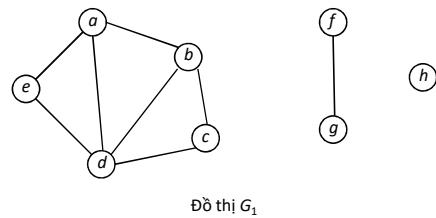
- Tập đỉnh V là tập hữu hạn phần tử, các phần tử gọi là các **đỉnh**
- Tập cạnh E là các cặp không có thứ tự của V gọi là các **cạnh**
- $(u, v), u, v \in V, u \neq v$

16

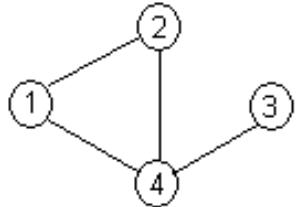
Đơn đồ thị vô hướng

(Simple Graph)

- Ví dụ: Đơn đồ thị $G_1 = (V_1, E_1)$, trong đó
 $V_1 = \{a, b, c, d, e, f, g, h\}$,
 $E_1 = \{(a,b), (b,c), (c,d), (a,d), (d,e), (a,e), (d,b), (f,g)\}$.



17



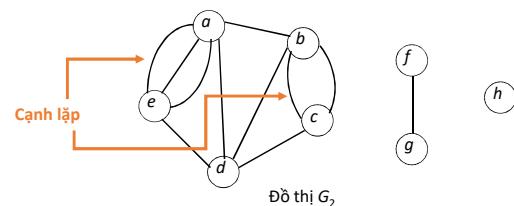
Không phải đơn đồ thị vô hướng do có các cặp cạnh nối cùng một cặp đỉnh

18

Đa đồ thị vô hướng

(Multi Graphs)

- Ví dụ: Đa đồ thị $G_2 = (V_2, E_2)$, trong đó
 $V_2 = \{a, b, c, d, e, f, g, h\}$,
 $E_2 = \{(a,b), (b,c), (b,c), (c,d), (a,d), (d,e), (a,e), (a,e), (a,e), (d,b), (f,g)\}$.



18

Đồ thị có hướng

(Directed Graph)

Định nghĩa. Đơn (đa) đồ thị có hướng $G = (V, E)$ là cặp gồm:

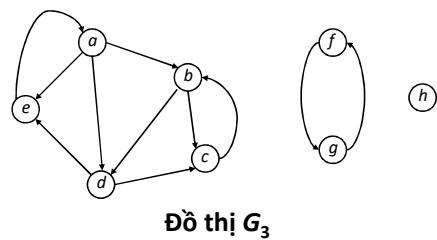
- Tập đỉnh V là tập hữu hạn phần tử, các phần tử gọi là các **đỉnh**
- Tập cung E là tập (**họ**) các bộ có thứ tự dạng (u, v) , $u, v \in V$, $u \neq v$

20

Đơn đồ thị có hướng

(Simple digraph)

- Ví dụ: Đơn đồ thị có hướng $G_3 = (V_3, E_3)$, trong đó
 $V_3 = \{a, b, c, d, e, f, g, h\}$,
 $E_3 = \{(a,b), (b,c), (c,b), (d,c), (a,d), (b,d), (a,e), (d,e), (e,a), (f,g), (g,f)\}$



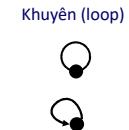
21

Các loại đồ thị: Tóm tắt

Loại	Kiểu cạnh	Có cạnh lặp?
Đơn đồ thị vô hướng	Vô hướng	Không
Đa đồ thị vô hướng	Vô hướng	Có
Đơn đồ thị có hướng	Có hướng	Không
Đa đồ thị có hướng	Có hướng	Có

Chú ý:

- Một dạng đồ thị ít sử dụng hơn, đó là giả đồ thị. **Giả đồ thị** là đa đồ thị mà trong đó có các **khuyên** (cạnh nối 1 đỉnh với chính nó).
- Cách phân loại đồ thị dùng ở đây chưa chắc đã được chấp nhận trong các tài liệu khác...

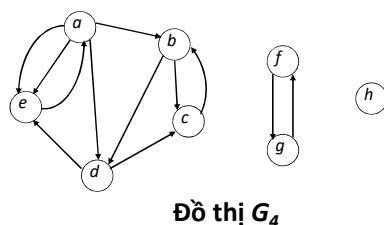


23

Đa đồ thị có hướng

(Multi Graphs)

- Ví dụ: Đa đồ thị có hướng $G_4 = (V_4, E_4)$, trong đó
 $V_4 = \{a, b, c, d, e, f, g, h\}$,
 $E_4 = \{(a,b), (b,c), (c,b), (d,c), (a,d), (b,d), (a,e), (a,e), (d,e), (e,a), (f,g), (g,f)\}$



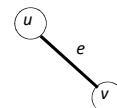
22

Các thuật ngữ

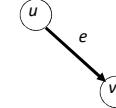
Graph Terminology

Chúng ta cần các thuật ngữ liên quan đến mối quan hệ giữa các đỉnh và các cạnh của đồ thị sau:

- Kề nhau, nối, đầu mút, bậc, bắt đầu, kết thúc, bán bậc vào, bán bậc ra...**



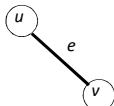
Cạnh vô hướng $e = (u, v)$



Cạnh có hướng (cung) $e = (u, v)$

24

Kè (Adjacency)



Cho G là đồ thị vô hướng với tập cạnh E . Giả sử $e \in E$ là cặp (u, v) . Khi đó ta nói:

- u, v là kè nhau/lân cận/nối với nhau (adjacent / neighbors / connected).
- Cạnh e là liên thuộc với hai đỉnh u và v .
- Cạnh e nối (connect) u và v .
- Các đỉnh u và v là các đầu mút (endpoints) của cạnh e .

25

1.1. Đồ thị trong thực tế

1.2. Các loại đồ thị

1.3. Bậc của đỉnh

1.4. Đồ thị con

1.5. Đồ thị đẳng cấu

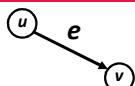
1.6. Đường đi và chu trình

1.7. Tính liên thông

1.8. Một số loại đồ thị đặc biệt

27

Tính kè trong đồ thị có hướng



- Cho G là đồ thị có hướng (có thể là đơn hoặc đa) và giả sử $e = (u, v)$ là cạnh của G . Ta nói:
 - u và v là kè nhau, u là kè tới v , v là kè từ u
 - e đi ra khỏi u , e đi vào v .
 - e nối u với v , e đi từ u tới v
 - Đỉnh đầu (initial vertex) của e là u
 - Đỉnh cuối (terminal vertex) của e là v

26

Bậc của đỉnh (Degree of a Vertex)

• Giả sử G là đồ thị vô hướng, $v \in V$ là một đỉnh nào đó.

• **Bậc** của đỉnh v , $\deg(v)$, là số cạnh kè với nó.

• Đỉnh bậc 0 được gọi là **đỉnh cô lập** (isolated).

• Đỉnh bậc 1 được gọi là **đỉnh treo** (pendant).

• Các ký hiệu thường dùng:

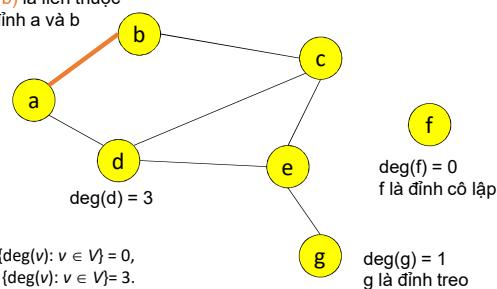
$$\delta(G) = \min \{\deg(v) : v \in V\},$$

$$\Delta(G) = \max \{\deg(v) : v \in V\}.$$

28

Ví dụ

Cạnh (a,b) là liên thuộc với hai đỉnh a và b



$$\delta(G) = \min \{\deg(v): v \in V\} = 0,$$
$$\Delta(G) = \max \{\deg(v): v \in V\} = 3.$$

29

Định lý về các cái bắt tay

(Handshaking Theorem)

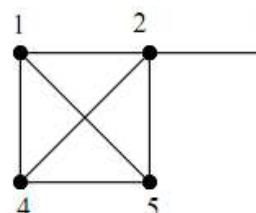
- Định lý. Giả sử G là đồ thị vô hướng (đơn hoặc đa) với tập đỉnh V và tập cạnh E . Khi đó

$$\sum_{v \in V} \deg(v) = 2|E|$$

CM: Trong tổng σ về trái mỗi cạnh $e=(u,v) \in E$ được tính hai lần: trong $\deg(u)$ và $\deg(v)$.

- Hệ quả: Trong một đồ thị vô hướng bất kỳ, số lượng đỉnh bậc lẻ (đỉnh có bậc là số lẻ) bao giờ cũng là số chẵn.

31



Bậc của các đỉnh:

30

Ví dụ.

Biết rằng mỗi đỉnh của đồ thị vô hướng $G=(V,E)$ với 14 đỉnh và 25 cạnh đều có bậc là 3 hoặc 5.

Hỏi G có bao nhiêu đỉnh bậc 3?

Giải. Giả sử G có x đỉnh bậc 3.

Khi đó có $14-x$ đỉnh bậc 5.

Do $|E| = 25$, nên tổng tất cả các bậc là 50.

Từ đó, $3x + 5(14-x) = 50$

Suy ra $x = 10$.

32

Bậc của đỉnh của đồ thị có hướng

- Cho G là đồ thị có hướng, v là đỉnh của G .
- Bán bậc vào (in-degree) của v , $\deg^-(v)$, là số cạnh đi vào v .
- Bán bậc ra (out-degree) của v , $\deg^+(v)$, là số cạnh đi ra khỏi v .
- Bậc của v , $\deg(v) := \deg^-(v) + \deg^+(v)$, là tổng của bán bậc vào và bán bậc ra của v .

33

Định lý về các cái bắt tay có hướng

Directed Handshaking Theorem

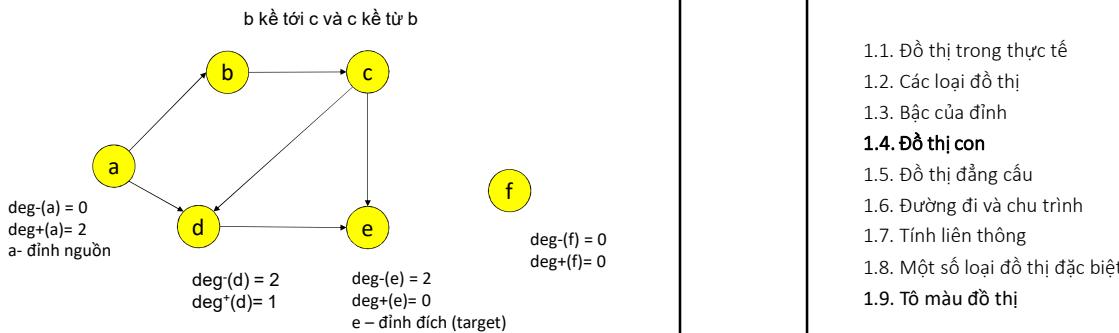
- Định lý. Giả sử G là đồ thị có hướng (có thể là đơn hoặc đa) với tập đỉnh V và tập cạnh E . Khi đó:

$$\sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = \frac{1}{2} \sum_{v \in V} \deg(v) = |E|$$

- Chú ý là khái niệm bậc của đỉnh là không thay đổi cho dù ta xét đồ thị vô hướng hay có hướng.

34

Ví dụ



34

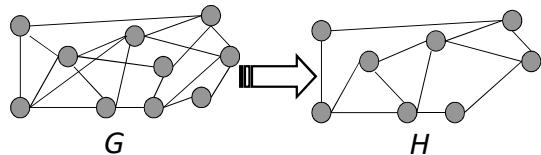
Chương 1 CÁC KHÁI NIỆM CƠ BẢN

1. Đồ thị trong thực tế
2. Các loại đồ thị
3. Bậc của đỉnh
4. Đồ thị con
5. Đồ thị đẳng cấu
6. Đường đi và chu trình
7. Tính liên thông
8. Một số loại đồ thị đặc biệt
9. Tô màu đồ thị

35

Đồ thị con (Subgraphs)

- **Định nghĩa.** Đồ thị $H=(W,F)$ được gọi là đồ thị con của đồ thị $G=(V,E)$ nếu $W \subseteq V$ và $F \subseteq E$.
- Ký hiệu: $H \subseteq G$.



37

Chương 1 CÁC KHÁI NIỆM CƠ BẢN

- 1.1. Đồ thị trong thực tế
- 1.2. Các loại đồ thị
- 1.3. Bậc của đỉnh
- 1.4. Đồ thị con
- 1.5. Đồ thị đẳng cấu
- 1.6. Đường đi và chu trình**
- 1.7. Tính liên thông
- 1.8. Một số loại đồ thị đặc biệt
- 1.9. Tô màu đồ thị

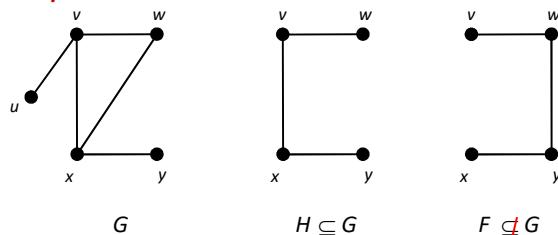
39

Ví dụ

Definition.

A graph H is a subgraph of a graph G if $V(H) \subseteq V(G)$ and $E(H) \subseteq E(G)$ (denote $H \subseteq G$).

Ví dụ



Đường đi, Chu trình

- **Định nghĩa.** Đường đi P độ dài n từ đỉnh u đến đỉnh v , trong đó n là số nguyên dương, trên đồ thị $G=(V,E)$ là dãy

$P: x_0, x_1, \dots, x_{n-1}, x_n$
trong đó $u = x_0, v = x_n, (x_i, x_{i+1}) \in E, i = 0, 1, 2, \dots, n-1$.

Đường đi nói trên còn có thể biểu diễn dưới dạng dãy các cạnh:

$(x_0, x_1), (x_1, x_2), \dots, (x_{n-1}, x_n)$.

Đỉnh u gọi là **đỉnh đầu**, còn đỉnh v gọi là **đỉnh cuối** của đường đi.

40

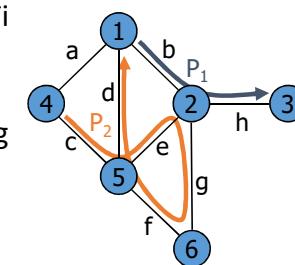
Đường đi, Chu trình

- Đường đi gọi là **đường đi đơn** nếu không có đỉnh nào bị lặp lại trên nó.
- Đường đi gọi là **đường đi cơ bản** nếu không có cạnh nào bị lặp lại trên nó.
- Nếu có đường đi từ u đến v thì ta nói đỉnh v **đạt đến được** từ đỉnh u. Ta quan niệm rằng một đỉnh v luôn đạt đến được từ chính nó.

41

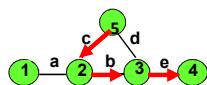
Ví dụ (cont.)

- $P_1 = (1, b, 2, h, 3)$ là đường đi đơn
- $P_2 = (4, c, 5, e, 2, g, 6, f, 5, d, 1)$ là đường đi nhưng không là đường đi đơn



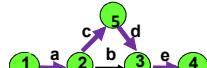
41

Đường đi (Path)



Ví dụ: 5, 2, 3, 4 hoặc 5, 2, b, 3, e, 4.

Không có đỉnh lặp nên là đường đi đơn



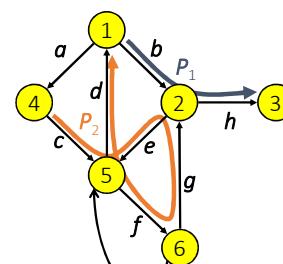
Ví dụ: 1, 2, 5, 3, 4 hoặc 1, a, 2, c, 5, d, 3, e, 4

• Là đường đi đơn

42

Ví dụ (cont.)

- $P_1 = (1, b, 2, h, 3)$ là đường đi đơn
- $P_2 = (4, c, 5, e, 2, g, 6, f, 5, d, 1)$ là đường đi nhưng không là đường đi đơn



44

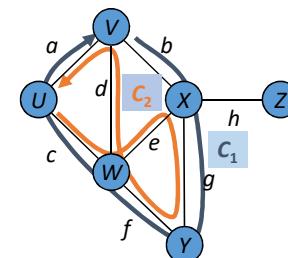
Chu trình

- Đường đi cơ bản có đỉnh đầu trùng với đỉnh cuối (tức là $u = v$) được gọi là **chu trình**.
- Chu trình được gọi là **đơn** nếu như ngoại trừ đỉnh đầu trùng với đỉnh cuối, không có đỉnh nào bị lặp lại.

45

Ví dụ: Chu trình trên đồ thị vô hướng

- $C_1=(V,b,X,g,Y,f,W,c,U,a,V)$ là chu trình đơn
- $C_2=(U,c,W,e,X,g,Y,f,W,d,V,a,U)$ là chu trình nhưng không là chu trình đơn

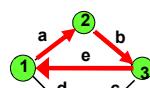


47

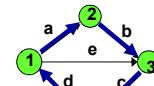
Chu trình (Cycle)

Chu trình

- 1, 2, 3, 1. (hay 1, a, 2, b, 3, e)
• Chu trình đơn



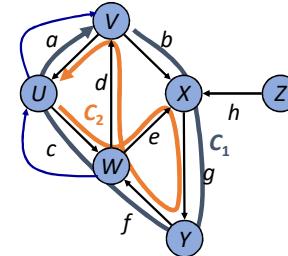
- Chu trình: (1, 2, 3, 4, 1) hay
1, a, 2, b, 3, c, 4, d, 1
• Chu trình đơn



46

Ví dụ: Chu trình trên đồ thị có hướng

- $C_1=(V,b,X,g,Y,f,W,c,U,a,V)$ là chu trình đơn
- $C_2=(U,c,W,e,X,g,Y,f,W,d,V,a,U)$ là chu trình nhưng không là chu trình đơn



48

Chương 1 CÁC KHÁI NIỆM CƠ BẢN

- 1.1. Đồ thị trong thực tế
- 1.2. Các loại đồ thị
- 1.3. Độ của đỉnh
- 1.4. Đồ thị con
- 1.5. Đồ thị đẳng cấu
- 1.6. Đường đi và chu trình
- 1.7. Tính liên thông**
- 1.8. Một số loại đồ thị đặc biệt
- 1.9. Tô màu đồ thị

49

Tính liên thông (Connectedness)

- **Mệnh đề:** Luôn tìm được đường đi đơn nối hai đỉnh bất kỳ của đồ thị vô hướng liên thông.

• **Chứng minh.**

Theo định nghĩa, luôn tìm được đường đi nối hai đỉnh bất kỳ của đồ thị liên thông. Gọi P là đường đi ngắn nhất nối hai đỉnh u và v . Rõ ràng P phải là đường đi đơn.

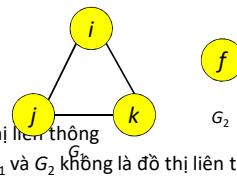
51

Tính liên thông (Connectedness)

- **Đồ thị vô hướng** được gọi là *liên thông* nếu luôn tìm được đường đi nối hai đỉnh bất kỳ của nó.

• **Ví dụ**

- G_1 và G_2 là các đồ thị *liên thông*
- G_1 và G_2 không là đồ thị *liên thông*

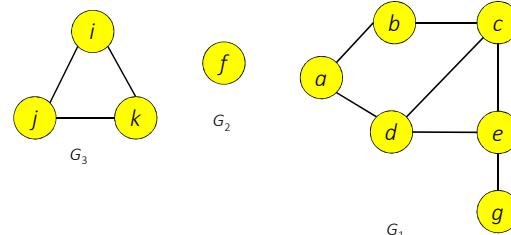


50

Tính liên thông (Connectedness)

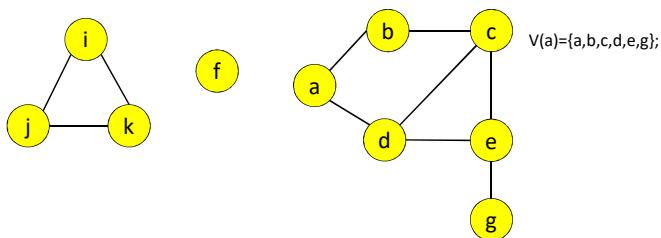
- **Thành phần liên thông (Connected component):** Đồ thị con liên thông cực đại của đồ thị vô hướng G được gọi là thành phần liên thông của nó.

• **Ví dụ:** Đồ thị G có 3 thành phần liên thông G_1 , G_2 , G_3



52

Thành phần liên thông



Giả sử $v \in V$. Gọi

- $V(v)$ – tập các đỉnh của đồ thị đạt đến được từ v ,
- $E(v)$ – tập các cạnh có ít nhất một đầu mút trong $V(v)$.

Khi đó $G(v) = (V(v), E(v))$ là đồ thị liên thông và được gọi là thành phần liên thông sinh bởi đỉnh v . Để thấy $G(v)$ là thành phần liên thông sinh bởi mọi đỉnh $u \in V(v)$.

53

Ví dụ

Mệnh đề. Cạnh e của đồ thị liên thông G là cầu iff e không thuộc bất cứ chu trình nào trên G .

Chứng minh

(\Rightarrow) Cho e là cầu của G .

Giả sử $e = (u, v)$, và giả sử ngược lại là e nằm trên chu trình

$$C : u, v, w, \dots, x, u.$$

Khi đó

$$C - e : v, w, \dots, x, u$$

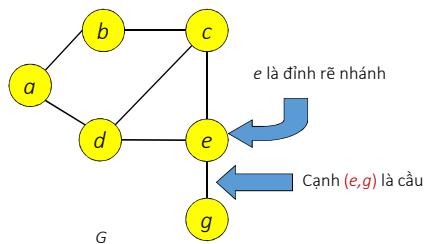
là đường đi từ u đến v trên đồ thị $G - e$.

Ta sẽ chứng minh: $G - e$ là liên thông.

(Điều đó sẽ mâu thuẫn với giả thiết e là cầu)

Đỉnh rẽ nhánh và cầu (Connectedness)

- Đỉnh rẽ nhánh (cut vertex):** là đỉnh mà việc loại bỏ nó làm tăng số thành phần liên thông của đồ thị
- Cầu (bridge):** Cạnh mà việc loại bỏ nó làm tăng số thành phần liên thông của đồ thị .
- Ví dụ:**



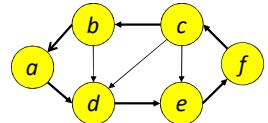
54

Tính liên thông của Đồ thị có hướng

- Đồ thị có hướng được gọi là *liên thông mạnh (strongly connected)* nếu như luôn tìm được đường đi nối hai đỉnh bất kỳ của nó.
- Đồ thị có hướng được gọi là *liên thông yếu (weakly connected)* nếu như đồ thị vô hướng thu được từ nó bởi việc bỏ qua hướng của tất cả các cạnh của nó là đồ thị vô hướng liên thông.
- Để thấy là nếu G là liên thông mạnh thì nó cũng là liên thông yếu, nhưng điều ngược lại không luôn đúng.

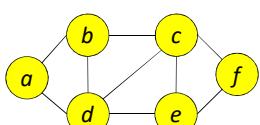
55

Ví dụ



Đồ thị liên thông mạnh

Đồ thị liên thông yếu



57

BIỂU DIỄN ĐỒ THỊ

Representations of Graphs

58

Chương 1 CÁC KHÁI NIỆM CƠ BẢN

- 1.1. Đồ thị trong thực tế
- 1.2. Các loại đồ thị
- 1.3. Độ bắc của đỉnh
- 1.4. Đồ thị con
- 1.5. Đồ thị đẳng cấu
- 1.6. Đường đi và chu trình
- 1.7. Tính liên thông
- 1.8. Một số loại đồ thị đặc biệt**
- 1.9. Tô màu đồ thị

58

Biểu diễn đồ thị

- Có nhiều cách biểu diễn. Việc lựa chọn cách biểu diễn phụ thuộc vào từng bài toán cụ thể cần xét, thuật toán cụ thể cần cài đặt.
- Có hai vấn đề chính cần quan tâm khi lựa chọn cách biểu diễn:
 - Bộ nhớ mà cách biểu diễn đó đòi hỏi
 - Thời gian cần thiết để trả lời các truy vấn thường xuyên đối với đồ thị trong quá trình xử lý đồ thị:
 - Chẳng hạn:
 - Có cạnh nối hai đỉnh u, v ?
 - Liệt kê các đỉnh kề của đỉnh v ?

60

Ma trận kề

(Adjacency Matrix)

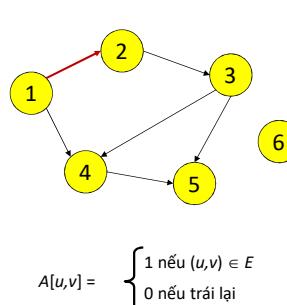
- $|V| \times |V|$ ma trận A .
- Các đỉnh được đánh số từ 1 đến $|V|$ theo 1 thứ tự nào đó.
- A xác định bởi:

$$A[i, j] = a_{ij} = \begin{cases} 1 & \text{nếu } (i, j) \in E \\ 0 & \text{nếu trái lại} \end{cases}$$

• $n = |V|$; $m = |E|$

61

Ma trận kề của đồ thị có hướng

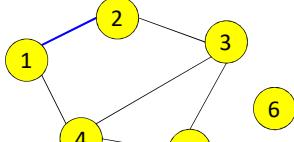


$$A[u, v] = \begin{cases} 1 & \text{nếu } (u, v) \in E \\ 0 & \text{nếu trái lại} \end{cases}$$

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 1 & 1 & 0 \\ 4 & 0 & 0 & 0 & 0 & 1 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

61

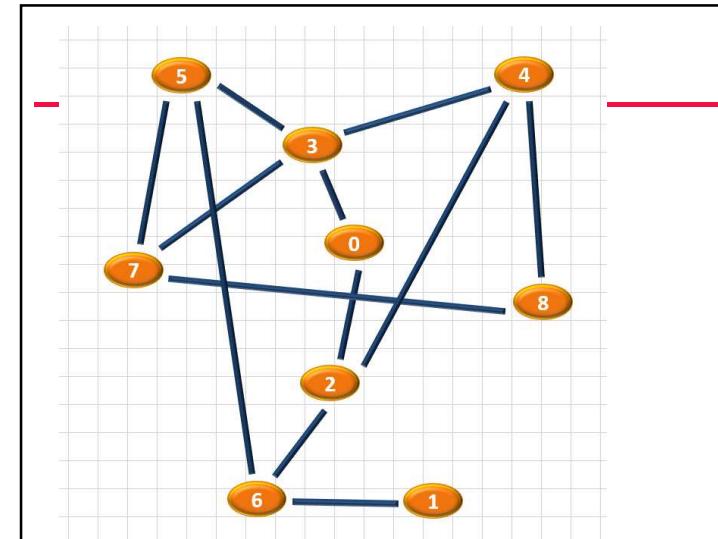
Ma trận kề của đồ thị vô hướng



$$A[u, v] = \begin{cases} 1 & \text{nếu } (u, v) \in E \\ 0 & \text{nếu trái lại} \end{cases}$$

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 \\ 2 & 1 & 0 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 0 & 1 & 1 & 0 \\ 4 & 1 & 0 & 1 & 0 & 1 & 0 \\ 5 & 0 & 0 & 1 & 1 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

62



64

Tính chất của ma trận kề

- Gọi A là ma trận kề của đồ thị vô hướng:

- A là ma trận đối xứng: $A = A^T$ ($a_{ij} = a_{ji}$)
- $\deg(v) =$ Tổng các phần tử trên dòng v của A
- Nếu ký hiệu $A^k = (a^{(k)}[u,v])$ thì $a^{(k)}[u,v]$ là số lượng đường đi từ u đến v đi qua không quá $k-1$ đỉnh trung gian.
- Khái niệm ma trận kề có thể mở rộng để biểu diễn đa đồ thị vô hướng: a_{uv} – số lượng cạnh nối hai đỉnh u và v .

65

Ma trận liên thuộc đỉnh cạnh

- Xét $G = (V, E)$, ($V = \{1, 2, \dots, n\}$, $E = \{e_1, e_2, \dots, e_m\}$), là đồ thị có hóng.

- Ma trận liên thuộc đỉnh cạnh $A = (a_{ij}; i = 1, 2, \dots, n; j = 1, 2, \dots, m)$

$$a_{ij} = \begin{cases} 1, & \text{nếu đỉnh } i \text{ là đỉnh đầu của cung } e_j \\ -1, & \text{nếu đỉnh } i \text{ là đỉnh cuối của cung } e_j \\ 0, & \text{nếu đỉnh } i \text{ không là đầu/mút của cung } e_j \end{cases}$$

- Ma trận liên thuộc đỉnh-cạnh là một trong những cách biểu diễn rất hay được sử dụng trong các bài toán liên quan đến đồ thị có hóng mà trong đó phải xử lý các cung của đồ thị.

67

Phân tích chi phí

- Bộ nhớ (Space)

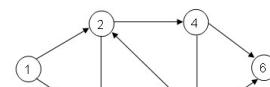
- $|V|^2$ bits
- Các thông tin bổ sung, chẳng hạn chi phí trên cạnh, cần được cất giữ dưới dạng ma trận. Một cách làm khác là cất giữ con trỏ đến các thông tin này.

- Thời gian trả lời các truy vấn

- Hai đỉnh i và j có kề nhau? $O(1)$
- Bổ sung hoặc loại bỏ cạnh $O(1)$
- Bổ sung đỉnh: tăng kích thước ma trận
- Liệt kê các đỉnh kề của v $O(|V|)$ (ngay cả khi v là đỉnh cô lập).

66

Ma trận liên thuộc đỉnh cạnh



(1,2) (1,3) (2,3) (2,4) (3,5) (4,5) (4,6) (5,2) (5,6)

$$A = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & -1 & 0 & 1 & 1 & 0 & 0 & 0 & -1 \\ 3 & 0 & -1 & -1 & 0 & 1 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 \\ 5 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \end{bmatrix}$$

68

Ma trận trọng số

- Trong trường hợp đồ thị có trọng số trên cạnh, thay vì ma trận kề, để biểu diễn đồ thị ta sử dụng **ma trận trọng số**

$$C = c[i, j], \quad i, j = 1, 2, \dots, n,$$

với

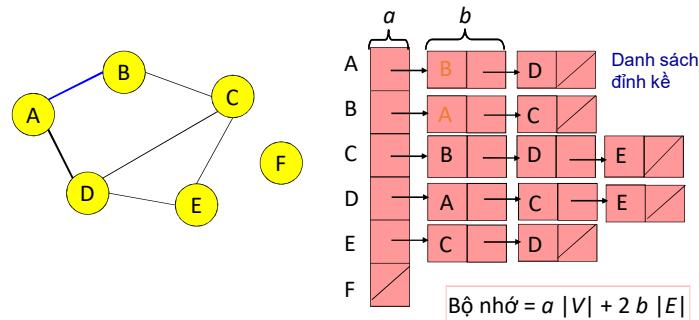
$$c[i, j] = \begin{cases} c(i, j), & \text{nếu } (i, j) \in E \\ \theta, & \text{nếu } (i, j) \notin E, \end{cases}$$

trong đó θ là giá trị đặc biệt để chỉ ra một cặp (i, j) không là cạnh, tùy từng trường hợp cụ thể, có thể đặt bằng một trong các giá trị sau: $0, +\infty, -\infty$.

69

Danh sách kề của đồ thị vô hướng

Với mỗi $v \in V$, $Ke(v) =$ danh sách các đỉnh $u: (v, u) \in E$



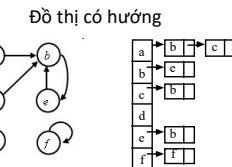
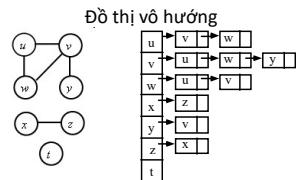
71

Danh sách kề

- Danh sách kề** (Adjacency Lists): Với mỗi đỉnh v cất giữ danh sách các đỉnh kề của nó.

- Là mảng Ke gồm $|V|$ danh sách.
- Mỗi đỉnh có một danh sách.
- Với mỗi $u \in V$, $Ke[u]$ bao gồm tất cả các đỉnh kề của u .

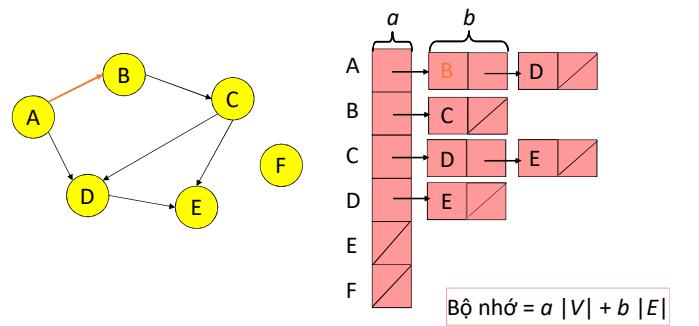
Ví dụ:



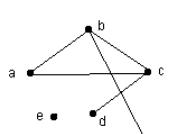
70

Danh sách kề của đồ thị có hướng

Với mỗi $v \in V$, $Ke(v) = \{ u: (v, u) \in E \}$



72



Đỉnh	Định kè
a	b, c, •
b	a, c, f, •
c	a, b, d, •
d	c, •
e	•
f	b, •

73

Biểu diễn đồ thị

• Thời gian trả lời các truy vấn:

- Thêm cạnh $O(1)$
- Xoá cạnh Duyệt qua danh sách kè của mỗi đầu mút.
- Thêm đỉnh Phụ thuộc vào cài đặt.
- Liệt kê các đỉnh kè của v: $O(<\text{số đỉnh kè}>)$ (tốt hơn ma trận kè)
- Hai đỉnh i, j có kè nhau?
 - Tìm kiếm trên danh sách: $\Theta(\text{degree}(i))$. Đánh giá trong tình huống tồi nhất là $O(|V|) \Rightarrow$ không hiệu quả (tốt hơn ma trận kè)

75

Yêu cầu bộ nhớ

• Tổng cộng bộ nhớ: $\Theta(|V| + |E|)$

- Thường là nhỏ hơn nhiều so với $|V|^2$, nhất là đối với đồ thị thưa (sparse graph).
- Đồ thị thưa là đồ thị mà $|E| = k |V|$ với $k < 10$.
- **Chú ý:**
 - Phần lớn các đồ thị trong thực tế ứng dụng là đồ thị thưa!
 - Cách biểu diễn này được sử dụng nhiều nhất trong ứng dụng

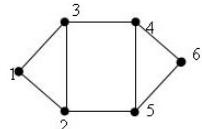
74

Danh sách cạnh (hoặc cung).

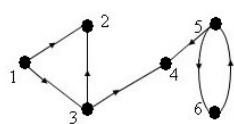
- Trong trường hợp thưa (tức là có số cạnh ít, thường là $m < 6n$, với m là số cạnh và n là số đỉnh của đồ thị), người ta thường dùng phương pháp biểu diễn đồ thị dưới dạng Danh sách cạnh: Mỗi cạnh (hoặc cung) $e = \langle x, y \rangle$ sẽ tương ứng với hai biến: Đầu[e] và Cuối[e].
- Như vậy để lưu trữ đồ thị ta cần sử dụng $2m$ đơn vị bộ nhớ. Nếu đồ thị có trọng số, ta cần thêm m đơn vị bộ nhớ để lưu trữ trọng số của các cạnh.
- Nhược điểm của phương pháp: Để xác định những đỉnh nào của đồ thị là kè của một đỉnh cho trước ta phải làm cỡ m phép so sánh (duyệt qua danh sách tất cả các cạnh của đồ thị).

76

Danh sách cạnh (hoặc cung).



Dau	Cuoi
1	2
1	3
2	3
2	5
3	4
4	5
4	6
5	6



Dau	Cuoi
1	2
1	3
3	2
3	4
5	4
5	6
6	5

77

BÀI TOÁN 7 CHIẾC CẦU

Thành phố Konigsberg (Đức) bị chia thành 4 vùng do 2 nhánh của 1 dòng sông. Có 7 chiếc cầu nối những vùng này với nhau.

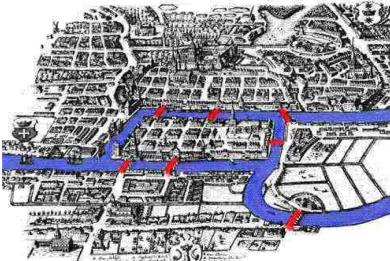
Bài toán: xuất phát từ một vùng đi dạo qua mỗi chiếc cầu đúng một lần và trở về nơi xuất phát.

Năm 1736, nhà toán học Euler đã mô hình bài toán này bằng một đồ thị vô hướng với mỗi đỉnh ứng với một vùng, mỗi cạnh ứng với một chiếc cầu

79



Konigsberg,
Hmmm

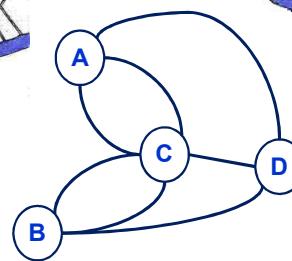
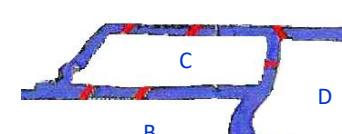
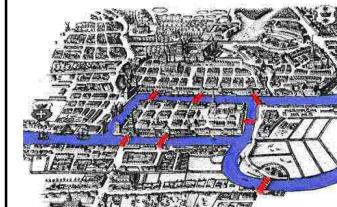


Leonhard Euler
(1707 – 1783)

ĐỒ THỊ EULER

78

BÀI TOÁN 7 CHIẾC CẦU



Lý thuyết đồ thị - chương 3 - Nguyễn Thành Sơn

80

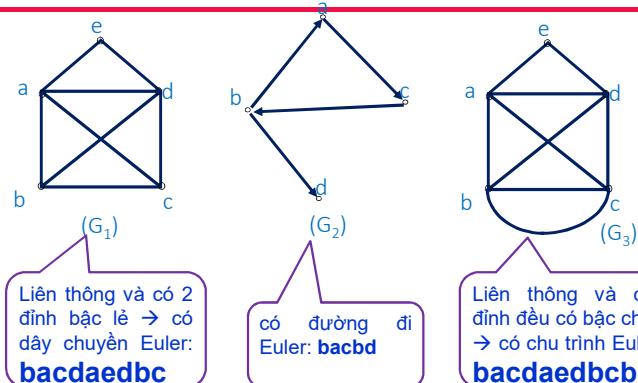
ĐỊNH NGHĨA

- DÂY CHUYỀN EULER: dây chuyền đi qua tất cả các cạnh trong đồ thị, mỗi cạnh đúng một lần.
- CHU TRÌNH EULER: dây chuyền Euler có đỉnh đầu trùng với đỉnh cuối.
- ĐƯỜNG ĐI EULER: đường đi qua tất cả các cạnh của đồ thị, mỗi cạnh đúng một lần.
- ĐỒ THỊ EULER VÔ HƯỚNG: đồ thị vô hướng có chứa một chu trình Euler.
- ĐỒ THỊ EULER CÓ HƯỚNG: đồ thị có hướng có chứa một mạch Euler.

Lý thuyết đồ thị - chương 3 - Nguyễn Thành Sơn

81

VÍ DỤ



83

ĐỊNH LÝ EULER

Đồ thị vô hướng $G=(X, E)$

1. G là đồ thị Euler $\Leftrightarrow G$ liên thông và $d(x)$ chẵn $\forall x \in X$.
2. G có chứa dây chuyền Euler và không chứa chu trình Euler $\Leftrightarrow G$ liên thông có chứa đúng hai đỉnh bậc lẻ.

Đồ thị có hướng $G=(X, E)$

1. G là đồ thị Euler $\Leftrightarrow G$ liên thông và $d^+(x)=d^-(x) \forall x \in X$.

82

GIẢI THUẬT FLEURY

- Cạnh e của đồ thị G được gọi là CẦU nếu xóa e khỏi đồ thị thì làm tăng số thành phần liên thông của G .

Giải thuật

Gọi chu trình cần tìm là C .

1. Khởi tạo: Chọn một đỉnh bất kỳ cho vào C .

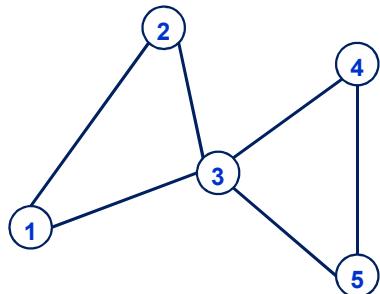
2. Lặp trong khi G vẫn còn cạnh

- Chọn cạnh e nối đỉnh vừa chọn với một đỉnh kề với nó theo nguyên tắc: chỉ chọn cầu nếu không còn cạnh nào khác để chọn.
- Bổ sung e và đỉnh cuối của nó vào C .
- Xóa e khỏi G .

84

VÍ DỤ

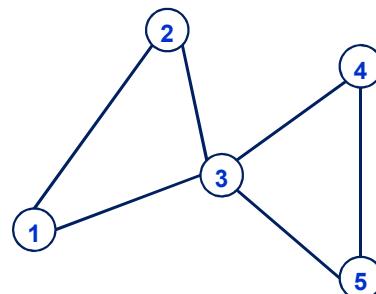
1 2 3 4 5 3 1



85

VÍ DỤ

1 2 3 1
3 4 5 3



87

GIẢI THUẬT XÁC ĐỊNH CÁC CHU TRÌNH THÀNH PHẦN

Input: đồ thị Euler $G(X, E)$

Output: chu trình Euler C của G

1. Chọn đỉnh $v \in X$; $C = \{v\}$

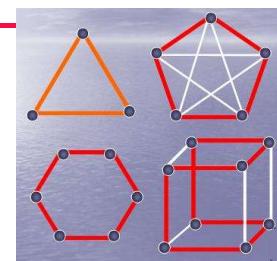
2. Lặp trong khi G còn cạnh

1. Chọn đỉnh $v \in C$ còn cạnh trong G
2. Tìm chu trình C' xuất phát từ v .
3. Ghép C' vào C
4. Loại bỏ các cạnh của C' khỏi G

86



Sir William Rowan Hamilton
(1805-1865)



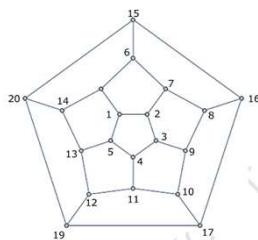
ĐỒ THỊ HAMILTON

88

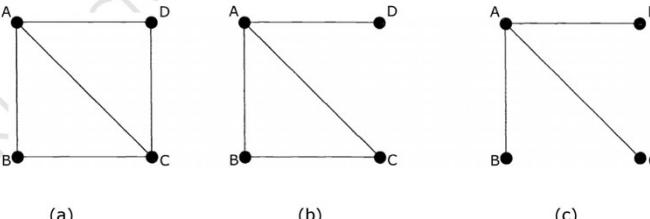
BÀI TOÁN KHỞI ĐIỂM

"Xuất phát từ một đỉnh của khối thập nhị diện đều, hãy đi dọc theo các cạnh của khối đó sao cho đi qua tất cả các đỉnh khác, mỗi đỉnh qua đúng một lần, sau đó trở về đỉnh xuất phát".

Bài toán này được nhà toán học Hamilton đưa ra vào năm 1859



89



(a)

(b)

(c)

ĐỊNH NGHĨA

Đồ thị vô hướng $G(V, E)$

- **Đường đi HAMILTON:** là một đường đi đơn qua tất cả các đỉnh của đồ thị mỗi đỉnh đúng một lần.
 - **CHU TRÌNH HAMILTON:** chu trình đơn đi qua tất cả các đỉnh của đồ thị đúng 1 lần và quay về đỉnh xuất phát.
 - **ĐỒ THỊ HAMILTON:** đồ thị có chứa ít nhất một chu trình Hamilton.
 - **ĐỒ THỊ NỬA HAMILTON:** là đồ thị có chứa đường Hamilton

90

MỘT SỐ KẾT QUẢ

Định lý 7.5 (Ore, 1960): Nếu G là đơn đồ thị vô hướng có n đỉnh ($n \geq 3$) và với mọi cặp đỉnh u, v không kề nhau, tổng bậc của u và bậc của v lớn hơn hoặc bằng n thì G là đồ thị Hamilton.

Bố đề 7.6 (Dirac, 1952): Nếu G là đơn đồ thị vô hướng có n đỉnh ($n \geq 3$) và mọi đỉnh v của G đều có bậc lớn hơn hoặc bằng $n/2$ thì G là đồ thị Hamilton.

Bổ đề 7.7 (Ghouila-Houiri, 1960): Nếu G là đồ thị có hướng liên thông mạnh có n đỉnh và mọi đỉnh của G có bậc lớn hơn hoặc bằng n thì G là đồ thị Hamilton.

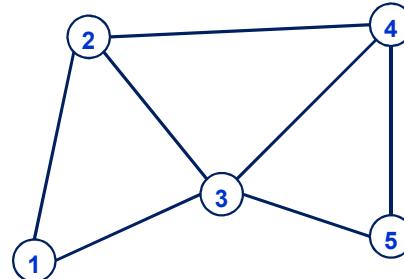
Định lý 7.8 (Meyniel, 1973): Nếu G là đồ thị có hướng liên thông mạnh có n đỉnh và với mọi cặp đỉnh u, v không kề nhau, tổng bậc của u và bậc của v lớn hơn hoặc bằng $(2n-1)$ thì G là đồ thị Hamilton.

- Đồ thị vô hướng đơn G gồm n đỉnh với $n \geq 3$
 - Nếu $d(x) \geq n/2 \forall x$ của G thì G là đồ thị Hamilton.
 - Nếu $d(x) \geq (n-1)/2 \forall x$ của G thì G có đường Hamilton.
 - Nếu $d(x)+d(y) \geq n$ với mọi cặp đỉnh x, y không kề nhau của G thì G là đồ thị Hamilton.

Lý thuyết đồ thị - chương 3 - Nguyễn Thành Sơn

93

VÍ DỤ



95

QUI TẮC XÁC ĐỊNH

- Nếu G có đỉnh bậc < 2 thì G không có chu trình Hamilton
- Nếu đỉnh có bậc 2 thì 2 cạnh kề với nó phải nằm trong chu trình Hamilton
- Các cạnh thừa (ngoài 2 cạnh đã chọn trong chu trình Hamilton) phải được bỏ đi trong quá trình xác định chu trình
- Nếu quá trình xây dựng tạo nên một chu trình con thì đồ thị không có chu trình Hamilton

94

CÁC THUẬT TOÁN DUYỆT ĐỒ THỊ

(Graph Searching, Graph Traversal)

96

Các thuật toán duyệt đồ thị

- Duyệt đồ thị: Graph Searching hoặc Graph Traversal
 - Duyệt qua mỗi đỉnh và mỗi cạnh của đồ thị
- Ứng dụng:
 - Cần để khảo sát các tính chất của đồ thị
 - Là thành phần cơ bản của nhiều thuật toán trên đồ thị
- Hai thuật toán duyệt cơ bản:
 - Tìm kiếm theo chiều rộng (Breadth First Search – BFS)
 - Tìm kiếm theo chiều sâu (Depth First Search – DFS)

97

Tìm kiếm theo chiều rộng

Breadth-first Search (BFS)

98

Ý tưởng chung của các thuật toán duyệt

Ý tưởng chung:

Giả sử $G = (V, E)$ là đồ thị với tập đỉnh V và tập cạnh E , v_0 là một đỉnh bất kỳ của G . Thuật toán duyệt đồ thị tổng quát được phát biểu như sau:

Bước 1: Khởi tạo cấu trúc dữ liệu kiểu danh sách để chứa các đỉnh cần duyệt: $L \leftarrow \{v_0\}$.

Bước 2: Lấy đỉnh u ra khỏi đầu danh sách L .

Bước 3: Duyệt đỉnh u .

Bước 4: Thêm các đỉnh liên kề với u vào danh sách L .

Bước 5: Nếu $L \neq \emptyset$, quay lại bước 2. Ngược lại, dừng thuật toán.

98

Tìm kiếm theo chiều rộng ~~Breadth-first Search~~

Trong giải thuật duyệt theo chiều rộng (Breath First Search), cấu trúc danh sách L mô tả trong giải thuật duyệt đồ thị tổng quát ở mục 8.1 được tổ chức theo kiểu hàng đợi (queue). Với cách tổ chức danh sách L như vậy, việc duyệt cây có tính chất lan rộng. Nghĩa là đỉnh gần với đỉnh xuất phát sẽ được xét trước, đỉnh xa đỉnh xuất phát sẽ được xét sau.

Thuật toán duyệt đồ thị theo chiều rộng (BFS) được phát biểu như sau:

Dữ liệu vào: Đồ thị $G = (V, E)$, đỉnh xuất phát v .

Kết quả: Danh sách các đỉnh của đồ thị G .

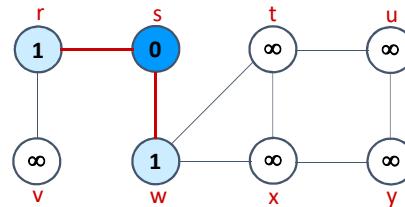
100

Thuật giải duyệt đồ thị bắt đầu từ đỉnh S theo BFS:

- Thăm đỉnh S
- Đánh dấu đỉnh S “DaTham”
- Cho S vào mảng TTDuyet
- Cho S vào hàng đợi
- Bắt đầu vòng lặp while (Trong khi hàng đợi còn phần tử, thực hiện các lệnh dưới)
 - v là đỉnh lấy từ hàng đợi
 - Xét tất cả các đỉnh kề của v
 - Nếu u là đỉnh kề v và chưa thăm thì
 - Đánh dấu u DaTham
 - LuuVet[u]=v;
 - Cho u vào mảng TTDuyet
 - Cho u vào hàng đợi
- Vòng lặp while kết thúc khi hàng đợi rỗng: **Đã duyệt tất cả các đỉnh liên thông với S**

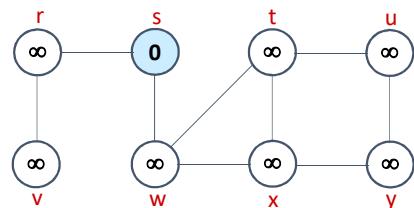
101

Ví dụ (BFS)



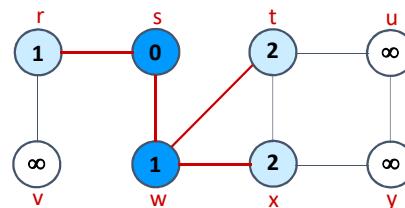
101

Ví dụ (BFS)



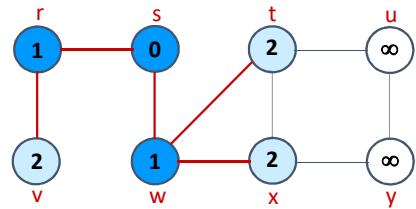
102

Ví dụ (BFS)



104

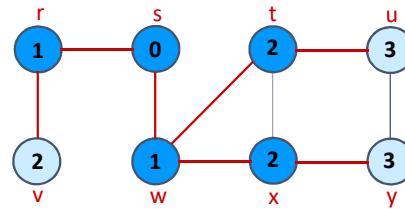
Ví dụ (BFS)



Q: t x v
2 2 2

105

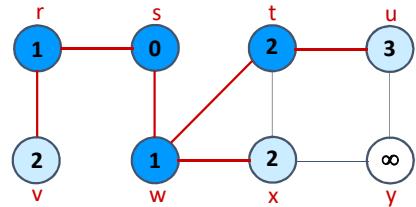
Ví dụ (BFS)



Q: v u y
2 3 3

107

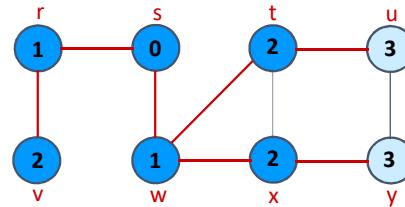
Ví dụ (BFS)



Q: x v u
2 2 3

106

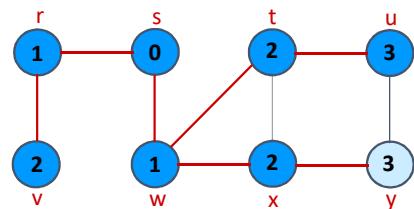
Ví dụ (BFS)



Q: u y
3 3

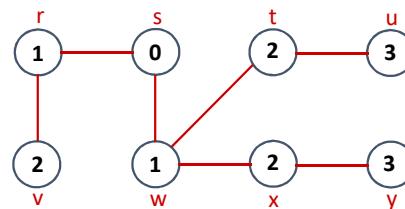
108

Ví dụ (BFS)



109

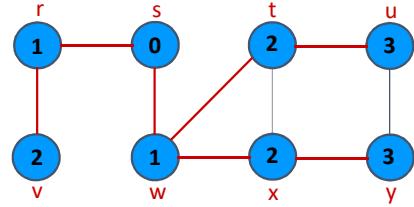
Ví dụ (BFS)



Cây BFS(s)

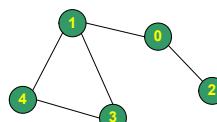
111

Ví dụ (BFS)



110

Ví dụ



- là: 0 1 2 3 4 Duyệt bắt đầu từ đỉnh 0, thứ tự duyệt sẽ
- là: 1 0 3 4 2 Duyệt bắt đầu từ đỉnh 1, thứ tự duyệt sẽ
- là: 2 0 1 3 4 Duyệt bắt đầu từ đỉnh 2, thứ tự duyệt sẽ
- là: 3 1 4 0 2 Duyệt bắt đầu từ đỉnh 3, thứ tự duyệt sẽ
- Duyệt bắt đầu từ đỉnh 4, thứ tự duyệt sẽ là:

4 1 3 0 2

112

Ứng dụng trực tiếp của BFS

- Sử dụng BFS để kiểm tra tính liên thông của đồ thị vô hướng:
 - Mỗi lần gọi đến BFS ở trong chương trình chính sẽ sinh ra một thành phần liên thông
- Xét sự tồn tại đường đi từ đỉnh s đến đỉnh t :
 - Thực hiện BFS(s).
- Chú ý: BFS tìm được đường đi ngắn nhất theo số cạnh.

113

Ý tưởng của tìm kiếm theo chiều sâu

- Trước hết, mọi đỉnh x kề với S tất nhiên sẽ đến được từ S .
- Với mỗi đỉnh x kề với S đó thì tất nhiên những đỉnh y kề với x cũng đến được từ S ...
- Điều đó gợi ý cho ta viết một thủ tục đệ quy DFS(u) mô tả việc duyệt từ đỉnh u bằng cách thông báo thăm đỉnh u và tiếp tục quá trình duyệt DFS(v) với v là một đỉnh chưa thăm kề với u .

114

Tìm kiếm theo chiều sâu

Depth-first Search (DFS)

114

Mô tả DFS

Trong giải thuật duyệt theo chiều sâu (Depth First Search), cấu trúc danh sách L mô tả trong giải thuật duyệt đồ thị tổng quát ở mục 8.1 được tổ chức theo kiểu ngăn xếp (stack). Với cách tổ chức danh sách L như vậy, mỗi lần duyệt một đỉnh, ta duyệt đến tận cùng của mỗi nhánh rồi mới duyệt sang nhánh tiếp theo.

Thuật toán duyệt đồ thị theo chiều sâu (DFS) được phát biểu như sau:

Dữ liệu vào: Đồ thị $G = (V, E)$, đỉnh xuất phát v .

Kết quả: Danh sách các đỉnh của đồ thị G .

115

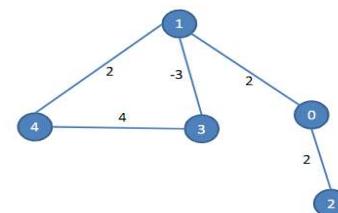
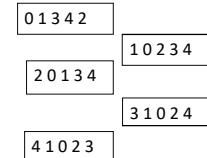
Depth-First Search: Code

```
Function DFS(v)
{
    Khởi tạo ngăn xếp L chỉ chứa đỉnh xuất phát v
    visited[v] = true
    for u ∈ V do
        visited[u] = false
    while (L ≠ ∅)
    {
        Loại u ra khỏi đầu hàng đợi L.
        Thăm đỉnh u.
    }
}
```

117

Ví dụ

- Duyệt bắt đầu từ đỉnh 0, thứ tự duyệt sẽ là:
- Duyệt bắt đầu từ đỉnh 1, thứ tự duyệt sẽ là:
- Duyệt bắt đầu từ đỉnh 2, thứ tự duyệt sẽ là:
- Duyệt bắt đầu từ đỉnh 3, thứ tự duyệt sẽ là:
- Duyệt bắt đầu từ đỉnh 4, thứ tự duyệt sẽ là:



118

Depth-First Search: Code

Đồ thị được duyệt bằng hàm DFS theo cách sau:

```
Function main()
{
    for v ∈ V do visited[v] = false
    for v ∈ V do
        if (visited[v] == false) then DFS(v)
}
```

119

CÁC ỨNG DỤNG CỦA DFS

- Tính liên thông của đồ thị
- Tìm đường đi từ s đến t
- Phát hiện chu trình
- Kiểm tra tính liên thông mạnh
- Định hướng đồ thị

120

Bài toán về tính liên thông

- **Bài toán:** Cho đồ thị vô hướng $G = (V, E)$. Hỏi đồ thị gồm bao nhiêu thành phần liên thông, và từng thành phần liên thông gồm các đỉnh nào?
- Giải: Sử dụng DFS (BFS):
 - Mỗi lần gọi đến DFS (BFS) ở trong chương trình chính sẽ sinh ra một thành phần liên thông

121

CÂY BAO TRÙM VÀ CÂY BAO TRÙM NHỎ NHẤT

121

Tìm đường đi

- Bài toán tìm đường đi
 - Input: Đồ thị $G = (V, E)$ xác định bởi danh sách kề và hai đỉnh s và t .
 - Đầu ra: Đường đi từ đỉnh s đến đỉnh t , hoặc khẳng định không tồn tại đường đi từ s đến t .
- Thuật toán: Thực hiện DFS(s) (hoặc BFS(s)).

122

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Định nghĩa và các tính chất cơ bản

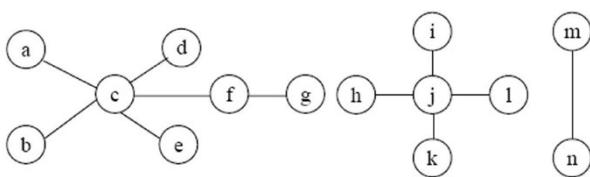
Định nghĩa:

- Cây là một đồ thị vô hướng liên thông, không chứa chu trình và có ít nhất hai đỉnh.
- Một đồ thị vô hướng không chứa chu trình và có ít nhất hai đỉnh gọi là một rừng.
- Trong một rừng, mỗi thành phần liên thông là một cây.

60

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Ví dụ: Đây là một rừng gồm 3 cây



61

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Cây khung và bài toán tìm cây khung nhỏ nhất:

Định nghĩa:

- Trong đồ thị liên thông G , nếu ta loại bỏ cạnh nằm trên chu trình nào đó thì ta sẽ được đồ thị vẫn là liên thông. Nếu cứ loại bỏ các cạnh ở các chu trình khác cho đến khi nào đồ thị không còn chu trình (vẫn liên thông) thì ta thu được một cây nối các đỉnh của G .
- Cây đó gọi là **cây khung** hay **cây bao trùm** của đồ thị G .

63

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Định lý: Cho T là một đồ thị có $n \geq 2$ đỉnh. Các điều sau là tương đương:

- 1) T là một cây.
- 2) T liên thông và có $n-1$ cạnh.
- 3) T không chứa chu trình và có $n-1$ cạnh.
- 4) T liên thông và mỗi cạnh là cầu.
- 5) Giữa hai đỉnh phân biệt bất kỳ của T luôn có duy nhất một đường đi đơn.
- 6) T không chứa chu trình nhưng khi thêm một cạnh mới thì có được một chu trình duy nhất.

62

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Một số tính chất:

1. Đồ thị vô hướng G có cây bao trùm khi và chỉ khi G liên thông.
2. Số cây bao trùm (cây khung) của đồ thị vô hướng đầy đủ n đỉnh là n^{n-2} .
3. Đồ thị vô hướng liên thông G có ít nhất một cây bao trùm.
4. Mọi cây bao trùm của đồ thị G có cùng số lượng đỉnh và cạnh.

62

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

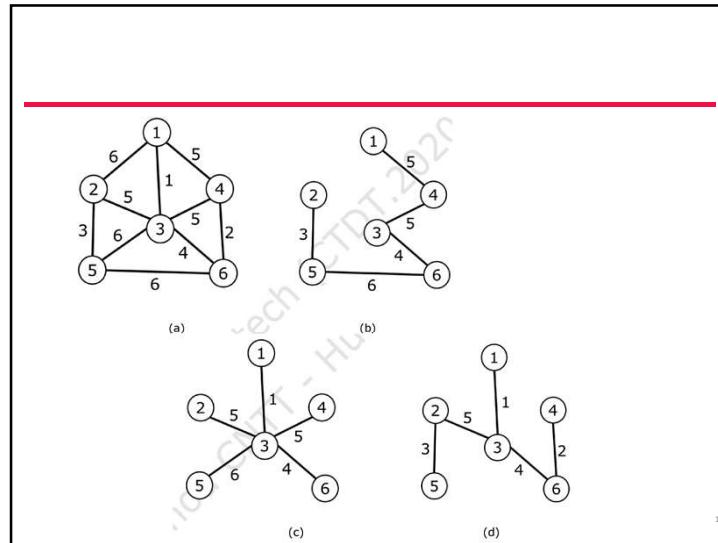
Cây khung và bài toán tìm cây khung nhỏ nhất:

Bài toán được phát biểu như sau:

Cho $G=(V,E)$ là đồ thị vô hướng liên thông có trọng số, mỗi cạnh e thuộc E có trọng số $m(e) \geq 0$. Giả sử $T=(V_T, E_T)$ là cây khung của đồ thị G ($V_T = V$). Ta gọi độ dài $m(T)$ của cây khung T là tổng trọng số của các cạnh của nó. Khi đó:

$$m(T) = \sum_{e \in E_T} m(e)$$

64



131

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Cây khung và bài toán tìm cây khung nhỏ nhất:

- Bài toán đặt ra là trong số tất cả các cây khung của đồ thị G , hãy tìm cây khung có độ dài nhỏ nhất.
- Cây khung như vậy được gọi là **cây khung nhỏ nhất** của đồ thị.
- Bài toán được gọi là "**bài toán tìm cây khung nhỏ nhất**".

- Hai mô hình thực tế tiêu biểu:
 - * *Bài toán xây dựng hệ thống đường sắt.*
 - * *Bài toán nối mạng máy tính.*

65

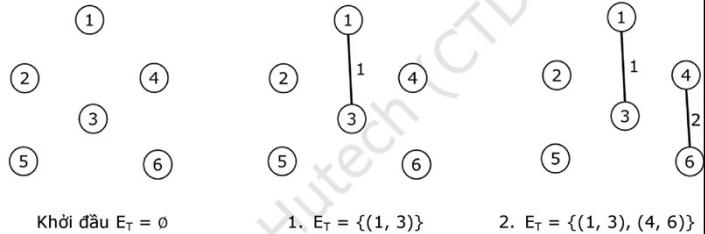
6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Thuật toán Kruskal: Thuật toán sẽ xây dựng tập cạnh E_T của cây khung nhỏ nhất $T = (V_T, E_T)$ theo từng bước:

1. Bắt đầu từ đồ thị rỗng T có n đỉnh.
2. Sắp xếp các cạnh của G theo thứ tự **không giảm** của trọng số.
3. Bắt đầu từ cạnh đầu tiên của dãy này, thêm dần các cạnh của dãy đã được xếp vào T theo nguyên tắc cạnh thêm vào không được tạo thành chu trình trong T .
4. Lặp lại Bước 3 cho đến khi nào số cạnh trong T bằng $n-1$, ta thu được cây khung nhỏ nhất cần tìm.

66

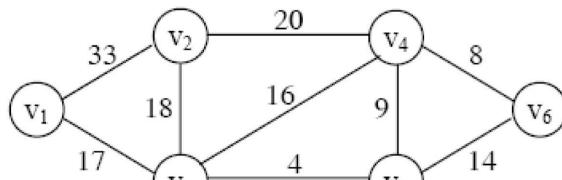
Các bước của thuật toán có thể được minh họa bằng hình ảnh như sau:



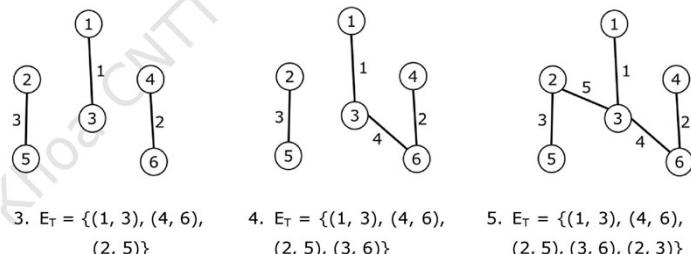
133

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Ví dụ: Tìm cây khung nhỏ nhất cho bối đồ thị sau



67



134

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Cho đồ thị liên thông, vô hướng, có trọng số $G = (V, E)$. Thuật toán Prim tìm cây khung nhỏ nhất:

Bước 1: Chọn tùy ý $v_o \in V$, và khởi tạo $V_T := \{v_o\}$, $E_T := \emptyset$. Trong đó, V_T là tập các đỉnh được chọn vào cây khung nhỏ nhất và E_T là tập các cạnh của cây này.

Bước 2: Trong số những cạnh nối đỉnh $v \in V_T$ với đỉnh $s \in V \setminus V_T$ chọn cạnh e có trọng số bé nhất.

Bước 3: Gán $V_T := V_T \cup \{s\}$ và $E_T := E_T \cup \{e\}$

Bước 4: Nếu số phần tử của E_T là $(n-1)$ thì kết thúc, nếu chưa thì quay lại thực hiện bước B2.

68

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Ví dụ: Tìm cây khung nhỏ nhất bằng thuật toán Prim cho bối đồ thị sau

	A	B	C	D	E	F	H	I
A	∞	15	16	19	23	20	32	18
B	15	∞	33	13	34	19	20	12
C	16	33	∞	13	29	21	20	19
D	19	13	13	∞	22	30	21	11
E	23	34	29	22	∞	34	23	21
F	20	19	21	30	34	∞	17	18
H	32	20	20	21	23	17	∞	14
I	18	12	19	11	21	18	14	∞

69

ĐƯỜNG ĐI NGẮN NHẤT

139

6. CÂY VÀ CÂY KHUNG CỦA ĐỒ THỊ

Bảng nhãn của các đỉnh:

V.lắp	A	B	C	D	E	F	H	I	V_T	E_T
K.tạo	-	[A,15]	[A,16]	[A,19]	[A,23]	[A,20]	[A,32]	[A,18]	A	\emptyset
1	-	-	[A,16]	[B,13]	[A,23]	[B,19]	[B,20]	[B,12]	A, B	(A,B)
2	-	-	[A,16]	[I,11]	[L,21]	[I,18]	[I,14]	-	A, B, I	(A.B), (B.I)
3	-	-	[D,13]	-	[I,21]	[I,18]	[I,14]	-	A, B, I, D	(A.B), (B.I), (I.D)
4	-	-	-	-	[I,21]	[I,18]	[I,14]	-	A, B, I, D, C	(A.B), (B.I), (I.D), (D.C)
5	-	-	-	-	[I,21]	[H,17]	-	-	A, B, I, D, C, H	(A.B), (B.I), (I.D), (D.C), (I.H)
6	-	-	-	-	[I,21]	-	-	-	A, B, I, D, C, H, F	(A.B), (B.I), (I.D), (D.C), (I.H), (H.F)
7	-	-	-	-	-	-	-	-	A, B, I, D, C, H, F, E	(A.B), (B.I), (I.D), (D.C), (I.H), (H.F), (I.E)

70

Vậy độ dài cây khung nhỏ nhất là:

$$15 + 12 + 11 + 13 + 14 + 17 + 21 = 103.$$

Bài toán đường đi ngắn nhất

• Cho đơn đồ thị có hướng $G = (V,E)$ với hàm trọng số $w: E \rightarrow R$ ($w(e)$ được gọi là độ dài hay trọng số của cạnh e)

• Độ dài của đường đi $P = v_1 \rightarrow v_2 \rightarrow \dots \rightarrow v_k$ là số

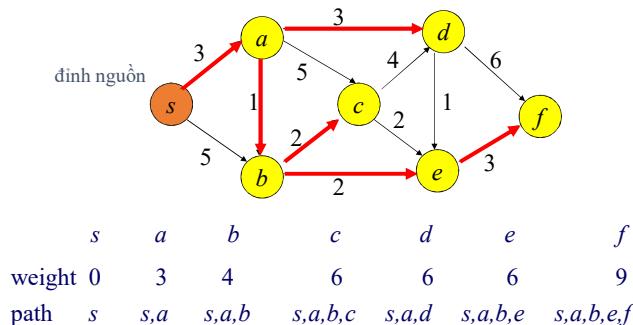
$$w(P) = \sum_{i=1}^{k-1} w(v_i, v_{i+1})$$

• Đường đi ngắn nhất từ đỉnh u đến đỉnh v là đường đi có độ dài ngắn nhất trong số các đường đi nối u với v .

• Độ dài của đường đi ngắn nhất từ u đến v còn được gọi là khoảng cách từ u tới v và ký hiệu là $\delta(u,v)$.

Ví dụ

Cho đồ thị có trọng số $G = (V, E)$, và đỉnh nguồn $s \in V$, hãy tìm đường đi ngắn nhất từ s đến mỗi đỉnh còn lại.



Các dạng bài toán ĐĐNN

- Bài toán một nguồn một đích:** Cho hai đỉnh s và t , cần tìm đường đi ngắn nhất từ s đến t .
- Bài toán một nguồn nhiều đích:** Cho s là đỉnh nguồn, cần tìm đường đi ngắn nhất từ s đến tất cả các đỉnh còn lại.
- Bài toán mọi cặp:** Tìm đường đi ngắn nhất giữa mọi cặp đỉnh của đồ thị.
 - Đường đi ngắn nhất theo số cạnh - BFS.

Các ứng dụng thực tế

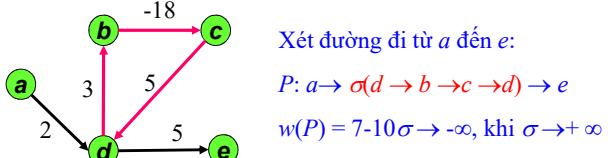
- Giao thông (Transportation)
- Truyền tin trên mạng (Network routing) (cần hướng các gói tin đến đích trên mạng theo đường nào?)
- Truyền thông (Telecommunications)
- Speech interpretation (best interpretation of a spoken sentence)
- Điều khiển robot (Robot path planning)
- Medical imaging
- Giải các bài toán phức tạp hơn trên mạng
- ...

Nhận xét

- Các bài toán được xếp theo thứ tự từ đơn giản đến phức tạp
- Hỗn hợp các bài toán có thể giải một trong ba bài toán thì thuật toán đó cũng có thể sử dụng để giải hai bài toán còn lại

Giả thiết cơ bản

- Nếu đồ thị có chu trình âm thì độ dài đường đi giữa hai đỉnh nào đó có thể làm nhỏ tùy ý:



Giả thiết:

Đồ thị không chứa chu trình độ dài âm (gọi tắt là chu trình âm)

Toán rời rạc, Fall 2005

Bài toán đường đi ngắn nhất 145

Thuật toán Dijkstra

- Trong trường hợp trọng số trên các cung là không âm, thuật toán do Dijkstra đề nghị hữu hiệu hơn rất nhiều so với thuật toán Ford-Bellman.

- Thuật toán được xây dựng dựa trên thủ tục gán nhãn. Thoát tiên nhãn của các đỉnh là tạm thời. Ở mỗi một bước lặp có một nhãn tạm thời trở thành nhãn cố định. Nếu nhãn của một đỉnh u trở thành cố định thì $d[u]$ sẽ cho ta độ dài của đường từ đỉnh s đến u . Thuật toán kết thúc khi nhãn của tất cả các đỉnh trở thành cố định.

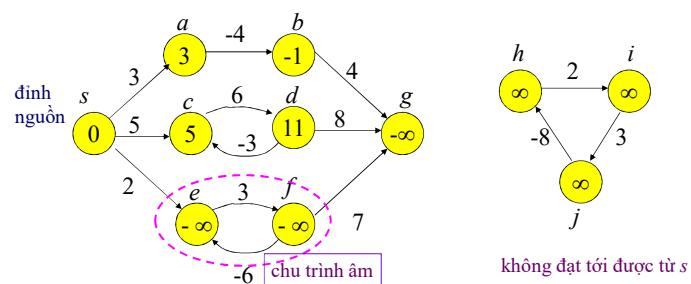


Edsger W.Dijkstra
(1930-2002)



Trọng số âm

Độ dài của đường đi ngắn nhất có thể là ∞ hoặc $-\infty$.



Thuật toán Dijkstra

- Đầu vào:** Đồ thị có hướng $G=(V,E)$ với n đỉnh, $s \in V$ là đỉnh xuất phát, $w[u,v], u,v \in V$ - ma trận trọng số;
- Giả thiết:** $w[u,v] \geq 0, u, v \in V$.
- Đầu ra:** Với mỗi $v \in V$
 - $d[v] = \delta(s, v)$;
 - $p[v]$ - đỉnh đi trước v trong ddnn từ s đến v .

Toán rời rạc, Fall 2005

Bài toán đường đi ngắn nhất 148

Thuật toán Dijkstra

[Tập S: Chỉ cần cho chứng minh định lý]

```

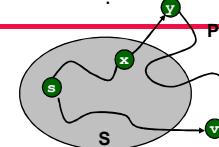
procedure Dijkstra;
begin
    for v ∈ V do begin (* Khởi tạo *)
        d[v] := w[s, v] ; p[v]:=s;
    end;
    d[s] := 0; S := {s};          (* S – tập đỉnh có nhãn cố định *)
    T := V \ {s};                (* T là tập các đỉnh có nhãn tạm thời *)
    while T ≠ ∅ do              (* Bắt đầu lặp *)
        begin
            Tim đỉnh u ∈ T thoả mãn d[u] = min{ d[z] : z ∈ T};
            T := T \ {u}; S := S ∪ {u}; (* Cố định nhãn của đỉnh u *)
            for v ∈ T do             (* Gán nhãn lại cho các đỉnh trong T *)
                if d[v] > d[u] + w[u,v] then begin
                    d[v] := d[u] + w[u,v]; p[v] := u ;
                end;
        end;
    end;
end;

```

Chứng minh tính đúng đắn của Thuật toán Dijkstra

• Ta sẽ CM với mỗi $v \in S$, $d(v) = \delta(s, v)$.

- Qui nạp theo $|S|$.
- Cơ sở qui nạp: Với $|S| = 1$, rõ ràng là đúng.



• Chuyển qui nạp:

- giả sử thuật toán Dijkstra bổ sung v vào S
 - $d(v)$ là độ dài của một đường đi từ s đến v
 - nếu $d(v)$ không là độ dài ngắn từ s đến v, thì gọi P^* là ngắn từ s đến v
 - P^* phải sử dụng cạnh ra khỏi S, chẳng hạn (x, y)
 - khi đó $d(v) > \delta(s, v)$
 - $= \delta(s, x) + w(x, y) + \delta(y, v)$ giả thiết
 - $\geq \delta(s, x) + w(x, y)$ tính chất 3
 - $= d(x) + w(x, y)$ $\delta(y, v)$ là không âm
 - $\geq d(y)$ giả thiết quy nạp
- vi thế thuật toán Dijkstra phải chọn y thay vì chọn v ?!

Toán rời rạc, Fall 2005

Bài toán đường đi ngắn nhất 151

Thuật toán Dijkstra

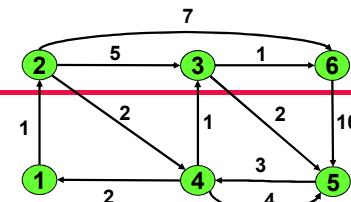
- Chú ý:** Nếu chỉ cần tìm đường đi ngắn nhất từ s đến t thì có thể chấm dứt thuật toán khi đỉnh t trở thành có nhãn cố định.
- Định lý 1.** *Thuật toán Dijkstra tìm được đường đi ngắn nhất từ đỉnh s đến tất cả các đỉnh còn lại trên đồ thị sau thời gian $O(n^2)$.*
- CM: Rõ ràng thời gian tính là $O(n^2)$

Toán rời rạc, Fall 2005

Bài toán đường đi ngắn nhất 150

Ví dụ

Tìm đường đi ngắn nhất từ đỉnh 1 đến tất cả các đỉnh còn lại

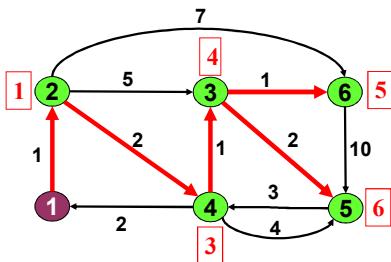


	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh 5	Đỉnh 6
Khởi tạo	[0, 1]	[1, 1]*	[∞, 1]	[∞, 1]	[∞, 1]	[∞, 1]
1	-	-	[6, 2]	[3, 2]*	[∞, 1]	[8, 2]
2	-	-	[4, 4]*	-	[7, 4]	[8, 2]
3	-	-	-	-	[6, 3]	[5, 3]*
4	-	-	-	-	[6, 3]*	-
5	-	-	-	-	-	-

Cây đường đi ngắn nhất

- Tập cạnh $\{(p(v), v) : v \in V \setminus \{s\}\}$ tạo thành cây có gốc tại đỉnh nguồn s được gọi là cây đđnn xuất phát từ đỉnh s .

- Các cạnh màu đỏ tạo thành cây đđnn xuất phát từ đỉnh 1
- Số màu đỏ viết bên cạnh mỗi đỉnh là độ dài đường đi ngắn nhất từ 1 đến nó.



Thuật toán Ford-Bellman

- Thuật toán Ford - Bellman tìm đường đi ngắn nhất từ đỉnh s đến tất cả các đỉnh còn lại của đồ thị.
- Thuật toán làm việc trong trường hợp trọng số của các cung là tùy ý.
- Giả thiết rằng trong đồ thị không có chu trình âm.**
- Đầu vào:** Đồ thị $G = (V, E)$ với n đỉnh xác định bởi ma trận trọng số $w[u, v]$, $u, v \in V$, đỉnh nguồn $s \in V$;
- Đầu ra:** Với mỗi $v \in V$
 - $d[v] = \delta(s, v)$;
 - $p[v] =$ đỉnh đi trước v trong đđnn từ s đến v .

Toán rời rạc, Fall 2005

Bài toán đường đi ngắn nhất 155

Thuật toán Ford-Bellman



Richard Bellman
1920-1984



Lester R. Ford, Jr.
1927~

Mô tả thuật toán

```

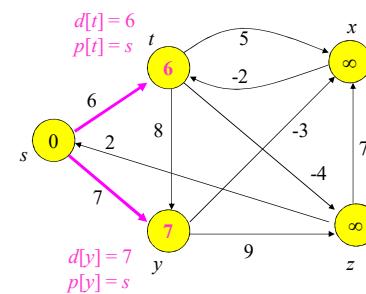
procedure Ford_Bellman;
begin
  for v ∈ V do begin (* Khởi tạo *)
    d[v] := w[s, v] ; p[v]:=s;
  end;
  d[s]:=0; p[s]:=s;
  for k := 1 to n-2 do      (* n = |V| *)
    for v ∈ V \ {s} do
      for u ∈ V do
        if d[v] > d[u] + w[u, v] then
          begin d[v] := d[u] + w[u, v] ;
            p[v] := u ;
          end;
    end;
end;

```

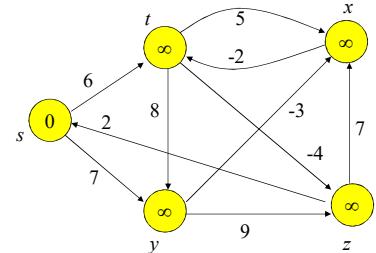
Nhận xét

- Tính đúng đắn của thuật toán có thể chứng minh trên cơ sở nguyên lý tối ưu của quy hoạch động.
- Độ phức tạp tính toán của thuật toán là $O(n^3)$.
- Có thể chấm dứt vòng lặp theo k khi phát hiện trong quá trình thực hiện hai vòng lặp trong không có biến $d[v]$ nào bị đổi giá trị. Việc này có thể xảy ra đối với $k < n-2$, và điều đó làm tăng hiệu quả của thuật toán trong việc giải các bài toán thực tế.

Lần 1



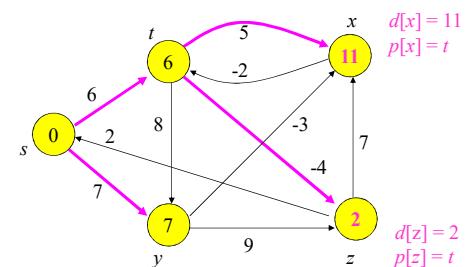
Ví dụ



Source: s

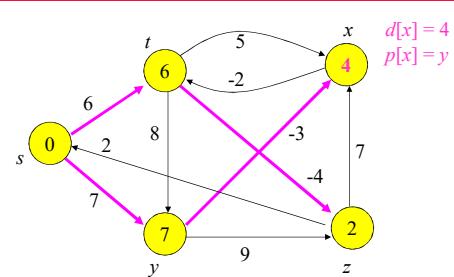
Trình tự duyệt cạnh để giảm cận: $(t, x), (t, y), (t, z), (x, t), (y, x), (y, z), (z, x), (z, s), (s, t), (s, y)$

Lần 2



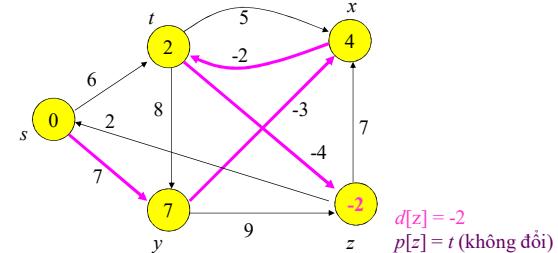
Relax $(t, x), (t, y), (t, z), (x, t)$.

Lần 2 (tiếp)

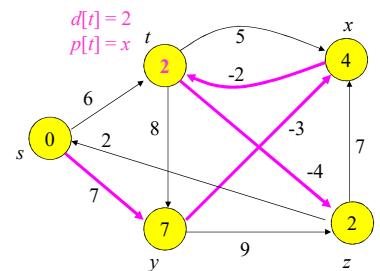


Relax $(y, x), (y, z), (z, x), (z, s), (z, 0), (s, t), (s, y)$.

Lần 4



Lần 3



Nhận xét

- Đối với đồ thi tha tốt hơn là sử dụng danh sách kề $Ke^-(v)$, $v \in V$, để biểu diễn đồ thi, khi đó vòng lặp theo u cần viết lại dưới dạng

```

for  $u \in Ke^-(v)$  do
  if  $d[v] > d[u] + w[u,v]$  then
    begin
       $d[v] := d[u] + w[u,v]$  ;
       $p[v] := u$  ;
    end;
  
```

- Thuật toán có độ phức tạp $O(n.m)$.

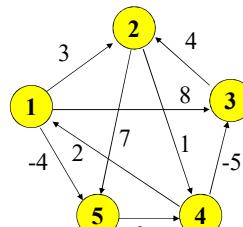
ĐƯỜNG ĐI NGẮN NHẤT GIỮA MỌI CẶP ĐỈNH

All-Pairs Shortest Paths

Toán rời rạc, Fall 2005

Bài toán đường đi ngắn nhất 165

Ví dụ



Đầu vào

$n \times n$ ma trận $W = (w_{ij})$ với
 $w_{ij} = \begin{cases} 0 & \text{nếu } i=j \\ w(i,j) & \text{nếu } i \neq j \text{ & } (i,j) \in E \\ \infty & \text{còn lại} \end{cases}$

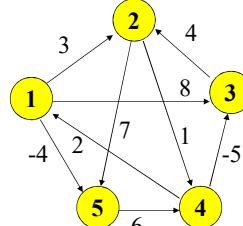
$$\begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

Đường đi ngắn nhất giữa mọi cặp đỉnh

Bài toán Cho đồ thị $G = (V, E)$, với trọng số trên cạnh e là $w(e)$, đối với mỗi cặp đỉnh u, v trong V , tìm đường đi ngắn nhất từ u đến v .

- ✳ Đầu vào: *ma trận trọng số*.
- ✳ Đầu ra *ma trận*: phần tử ở dòng u cột v là độ dài đường đi ngắn nhất từ u đến v .
- ✳ Cho phép có trọng số âm
- ✳ **Giả thiết:** Đồ thị không có chu trình âm.

Tiếp



Đầu ra

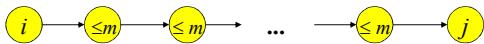
Đường đi: 1 - 5 - 4 - 3 - 2

$$\begin{pmatrix} 0 & 1 & -3 & 2 & -4 \\ 3 & 0 & -4 & 1 & -1 \\ 7 & 4 & 0 & 5 & 3 \\ 2 & -1 & -5 & 0 & -2 \\ 8 & 5 & 1 & 6 & 0 \end{pmatrix}$$

5 - 4 - 1 4 - 1 - 5

Thuật toán Floyd-Warshall

$d_{ij}^{(m)}$ = độ dài đường đi ngắn nhất từ i đến j sử dụng các đỉnh trung gian trong tập đỉnh $\{1, 2, \dots, m\}$.



Khi đó độ dài đường đi ngắn nhất từ i đến j là $d_{ij}^{(n)}$

Thuật toán Floyd-Warshall

```

Floyd-Warshall( $n, W$ )
 $D^{(0)} \leftarrow W$ 
for  $k \leftarrow 1$  to  $n$  do
    for  $i \leftarrow 1$  to  $n$  do
        for  $j \leftarrow 1$  to  $n$  do
             $d_{ij}^{(k)} \leftarrow \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)})$ 
return  $D^{(n)}$ 

```

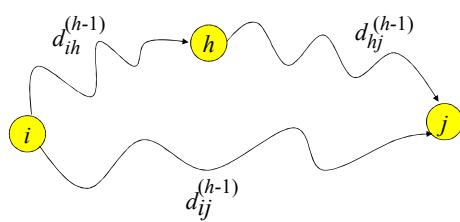
Thời gian tính $\Theta(n^3)$!

Công thức đệ quy tính $d^{(h)}$

• $d_{ij}^{(0)} = w_{ij}$

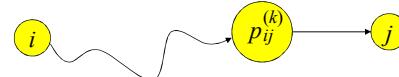


• $d_{ij}^{(h)} = \min(d_{ij}^{(h-1)}, d_{ih}^{(h-1)} + d_{hj}^{(h-1)})$ nếu $h \geq 1$



Xây dựng đường đi ngắn nhất

Predecessor matrix $P^{(k)} = (p_{ij}^{(k)})$:

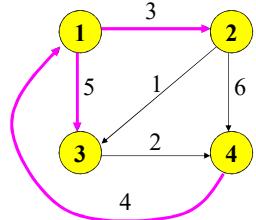


đường đi ngắn nhất từ i đến j chỉ qua các đỉnh trung gian trong $\{1, 2, \dots, k\}$.

$$p_{ij}^{(0)} = \begin{cases} i, & \text{nếu } (i, j) \in E \\ \text{NIL}, & \text{nếu } (i, j) \notin E \end{cases}$$

$$p_{ij}^{(k)} = \begin{cases} p_{ij}^{(k-1)}, & \text{nếu } d_{ij}^{(k-1)} \leq d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \\ p_{kj}^{(k-1)}, & \text{trái lại} \end{cases}$$

Ví dụ



$$D^{(0)} \begin{pmatrix} 0 & 3 & 5 & \infty \\ \infty & 0 & 1 & 6 \\ \infty & \infty & 0 & 2 \\ 4 & \infty & \infty & 0 \end{pmatrix}$$

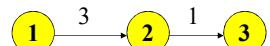
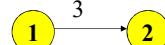
$$P^{(0)} \begin{pmatrix} \text{NIL} & 1 & 1 & \text{NIL} \\ \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & \text{NIL} & \text{NIL} & 3 \\ 4 & \text{NIL} & \text{NIL} & \text{NIL} \end{pmatrix}$$

Có thể sử dụng 1 là đỉnh trung gian:

$$D^{(1)} \begin{pmatrix} 0 & 3 & 5 & \infty \\ \infty & 0 & 1 & 6 \\ \infty & \infty & 0 & 2 \\ 4 & 7 & 9 & 0 \end{pmatrix}$$

$$P^{(1)} \begin{pmatrix} \text{NIL} & 1 & 1 & \text{NIL} \\ \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & \text{NIL} & \text{NIL} & 3 \\ 4 & 1 & 1 & \text{NIL} \end{pmatrix}$$

Ví dụ (tiếp)



Ví dụ (tiếp)

$$D^{(2)} \begin{pmatrix} 0 & 3 & 4 & 9 \\ \infty & 0 & 1 & 6 \\ \infty & \infty & 0 & 2 \\ 4 & 7 & 8 & 0 \end{pmatrix}$$

$$P^{(2)} \begin{pmatrix} \text{NIL} & 1 & 2 & 2 \\ \text{NIL} & \text{NIL} & 2 & 2 \\ \text{NIL} & \text{NIL} & \text{NIL} & 3 \\ 4 & 1 & 2 & \text{NIL} \end{pmatrix}$$

$$D^{(3)} \begin{pmatrix} 0 & 3 & 4 & 6 \\ \infty & 0 & 1 & 3 \\ \infty & \infty & 0 & 2 \\ 4 & 7 & 8 & 0 \end{pmatrix}$$

$$P^{(3)} \begin{pmatrix} \text{NIL} & 1 & 2 & 3 \\ \text{NIL} & \text{NIL} & 2 & 3 \\ \text{NIL} & \text{NIL} & \text{NIL} & 3 \\ 4 & 1 & 2 & \text{NIL} \end{pmatrix}$$

$$D^{(4)} \begin{pmatrix} 0 & 3 & 4 & 6 \\ 7 & 0 & 1 & 3 \\ 6 & 9 & 0 & 2 \\ 4 & 7 & 8 & 0 \end{pmatrix}$$

$$P^{(4)} \begin{pmatrix} \text{NIL} & 1 & 2 & 3 \\ 4 & \text{NIL} & 2 & 3 \\ 4 & 1 & \text{NIL} & 3 \\ 4 & 1 & 2 & \text{NIL} \end{pmatrix}$$

Thuật toán Floyd-Warshall

```
Floyd-Warshall( $n, W$ )
 $D \leftarrow W$ 
for  $k \leftarrow 1$  to  $n$  do
    for  $i \leftarrow 1$  to  $n$  do
        for  $j \leftarrow 1$  to  $n$  do
             $d_{ij} \leftarrow \min(d_{ij}, d_{ik} + d_{kj})$ 
return  $D$ 
```

Thời gian tính $\Theta(n^3)$!

Robert W. Floyd, 1936-2001



- Born in New York, Floyd finished school at age 14. At the University of Chicago, he received a Bachelor's degree in liberal arts in 1953 (when still only 17) and a second Bachelor's degree in physics in 1958.
- Becoming a computer operator in the early 1960s, he began publishing many noteworthy papers and was appointed an associate professor at Carnegie Mellon University by the time he was 27 and became a full professor at Stanford University six years later. He obtained this position without a Ph.D.
- Turing Award, 1978.

177

Thuật toán Floyd-Warshall

✳ Ma trận xuất phát là ma trận kè

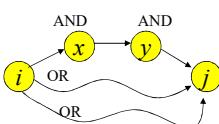
$$a(i, j) = \begin{cases} 1 & \text{nếu } i = j \text{ hoặc có cạnh nối 2 đỉnh } i \text{ và } j \\ 0 & \text{trái lại} \end{cases}$$



Nếu

✳ Thuật toán Floyd-Warshall thay

$$\begin{array}{l} \min \xrightarrow{\quad} \text{boolean OR} \\ + \xrightarrow{\quad} \text{boolean AND} \end{array}$$



✳ Thời gian tính $\Theta(n^3)$