

CHƯƠNG 1

Các khái niệm cơ bản về đồ thị

1

Các khái niệm cơ bản về đồ thị

- Định nghĩa đồ thị
- Độ của đỉnh
- Một số dạng đồ thị đặc biệt
- Đồ thị con, đồ thị bộ phận
- Đồ thị đẳng cấu
- Biểu diễn đồ thị

2

Định nghĩa đồ thị

- Đồ thị là một cấu trúc rời rạc gồm tập hợp các đỉnh và tập hợp các cạnh. Tuỳ vào kiểu và số lượng cạnh, ta có các loại đồ thị khác nhau.
 - Đồ thị có hướng
 - Đồ thị vô hướng

Giới thiệu

- LDTT là lĩnh vực nghiên cứu đã có từ lâu và có nhiều ứng dụng trong CNTT. Nó được đề xuất vào những năm đầu của thế kỷ 18 bởi nhà toán học lõi lạc người Thụy Sỹ: Leonhard Euler
- Những ứng dụng cơ bản của đồ thị:
 - Xác định tính liên thông trong một mạng máy tính
 - Tìm đường đi ngắn nhất
 - Giải các bài toán tối ưu
 - Giải bài toán tô màu trên bản đồ, ...

3

4

Đồ thị có hướng

- **ĐN1:** Đồ thị có hướng $G=(V,E)$ là đồ thị gồm:
 - V : tập hợp hữu hạn gồm các **đỉnh** của đồ thị.
 - E : tập hợp các cặp có thứ tự (u,v) với $u, v \in V$. Mỗi phần tử của E được gọi là 1 **cung** của G .
 - Cung $(u,v) \neq$ cung (v,u) có thể kí hiệu là **uv**

5

Đồ thị có hướng (tt)

- Cho (u,v) là cung của G :
 - Cung uv đi từ $u \rightarrow v$
 - Đỉnh u là gốc (đỉnh đầu) của cung.
 - Đỉnh v là ngọn (đỉnh cuối) của cung.
 - u, v là 2 đỉnh kè nhau
 - u là đỉnh trước của v
 - v là đỉnh sau của u

6

Đồ thị có hướng (tt)

- Hai cung uv , vu được gọi là 2 cung song song cùng chiều.



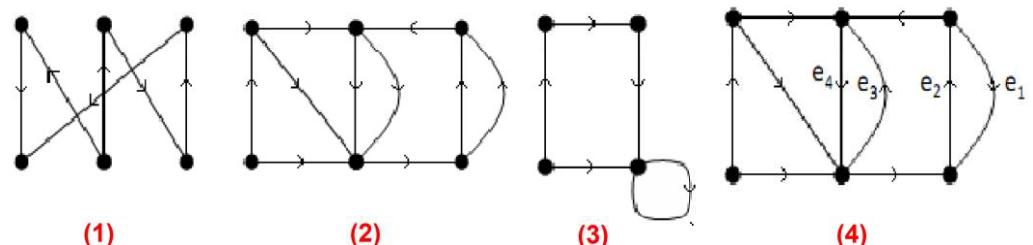
- Cung uu , được gọi là khuyên



7

Đồ thị có hướng (tt)

- **ĐN2:** Đồ thị có hướng, không có cung song song cùng chiều và không có khuyên gọi là đơn đồ thị có hướng.
- **Ví dụ:** đơn đồ thị có hướng?



8

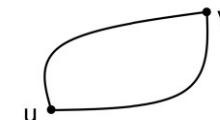
Đồ thị vô hướng

- **ĐN1:** Đồ thị vô hướng $G=(V,E)$ là đồ thị gồm:
 - V : tập hợp hữu hạn gồm các **đỉnh** của đồ thị.
 - E : tập hợp các cặp không kể thứ tự của 2 đỉnh. Mỗi phần tử của E được gọi là 1 **cạnh** của G .
 - Cạnh (u,v) có thể kí hiệu là **uv**. Cạnh uv và vu là như nhau.

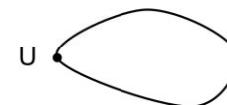
9

Đồ thị vô hướng (tt)

- Hai cạnh uv , uv được gọi là 2 cạnh song song.



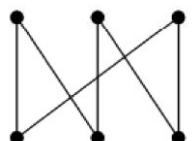
- Cạnh uu được gọi là khuyên



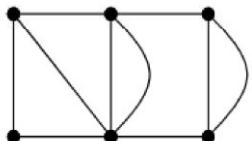
10

Đồ thị vô hướng (tt)

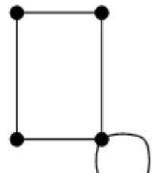
- **ĐN2:** Đồ thị vô hướng, không có cạnh song song và không có khuyên gọi là đơn đồ thị vô hướng.
- **Ví dụ:** đơn đồ thị vô hướng?



(1)



(2)



(3)

11

Định nghĩa (tt)

Loại đồ thị	Cạnh	cạnh/cung song song	Khuyên
Đơn đồ thị vô hướng			
Đa đồ thị vô hướng			
Giả đồ thị vô hướng			
Đồ thị có hướng			
Đa đồ thị có hướng			

12

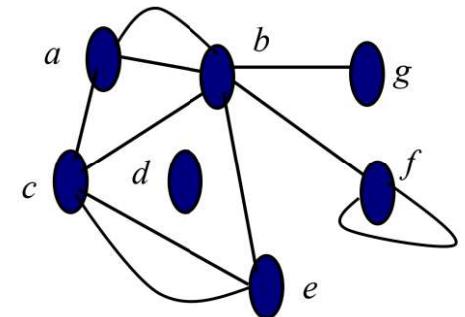
Bậc của đỉnh

- **ĐN1:** Cho $G=(V,E)$, vô hướng, $v \in V$.
 - Nếu $e = (u,v)$ là một cạnh của G thì:
 - Hai đỉnh u, v được gọi là hai đỉnh kề nhau
 - Cạnh e được gọi là cạnh kề với đỉnh u và đỉnh v
 - u, v được gọi là đỉnh đầu của cạnh e
 - $\deg(v) =$ số cạnh kề với v .
- Mỗi khuyên tại v được tính 2 lần cho v

13

Ví dụ:

- $\deg(a) =$
- $\deg(b) =$
- $\deg(c) =$
- $\deg(d) =$
- $\deg(e) =$
- $\deg(f) =$
- $\deg(g) =$



Chú ý:

- Đỉnh bậc 0 gọi là đỉnh cô lập
- Đỉnh bậc 1 gọi là đỉnh treo

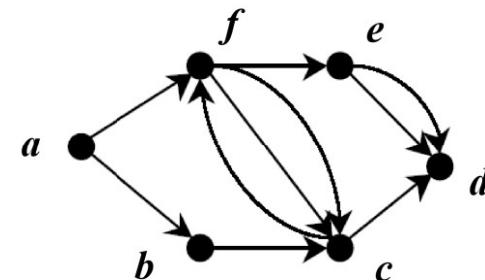
14

Bậc của đỉnh (tt)

- **ĐN2:** Cho $G=(V,E)$, có hướng, $v \in V$.
 - Bán bậc vào: $\deg^-(v)$
 - $\deg^-(v) =$ số cung nhận v là ngọn (đỉnh cuối)
 - Bán bậc ra: $\deg^+(v)$
 - $\deg^+(v) =$ số cung nhận v là gốc (đỉnh đầu)
 - $\deg(v) = \deg^+(v) + \deg^-(v)$

15

Ví dụ:



$$\begin{aligned}\deg^-(a) &= 0, \quad \deg^-(b) = 1, \quad \deg^-(c) = 3, \quad \deg^-(d) = ?, \quad \deg^-(e) = ?, \quad \deg^-(f) = ? \\ \deg^+(a) &= 2, \quad \deg^+(b) = 1, \quad \deg^+(c) = 1, \quad \deg^+(d) = ?, \quad \deg^+(e) = ?, \quad \deg^-(f) = ?\end{aligned}$$

16

Bậc của đỉnh (tt)

- **Định lý 1:** Cho $G=(V,E)$ có m cạnh (cung)

$$1. \sum_{v \in V} \deg(v) = 2m$$

- 2. Nếu G là đồ thị có hướng

$$\sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = m$$

Bậc của đỉnh (tt)

- **Định lý 2:** Số đỉnh bậc lẻ trong G phải là số chẵn

- **Định lý 3:** Trong 1 đồ thị với n đỉnh ($n \geq 2$) có ít nhất 2 đỉnh cùng bậc

- **Định lý 4:** Nếu đồ thị có n đỉnh ($n > 2$) có đúng 2 đỉnh cùng bậc thì 2 đỉnh này không thể đồng thời có bậc 0 hoặc $n-1$

17

18

Một số dạng đồ thị đặc biệt

- Đồ thị đầy đủ (đồ thị đủ)
- Đồ thị vòng
- Đồ thị bánh xe
- Đồ thị lưỡng phân (đồ thị hai phía)

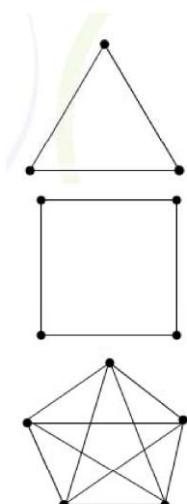
Đồ thị đầy đủ

- **Kí hiệu:** K_n
- **Đặc điểm:** là đơn đồ thị vô hướng, n đỉnh, 2 đỉnh bất kì đều kề nhau.

- **Tính chất:**

- Số cạnh $\frac{n(n-1)}{2}$
- Bậc của đỉnh

$$\deg(v) = n - 1 \quad \forall v$$

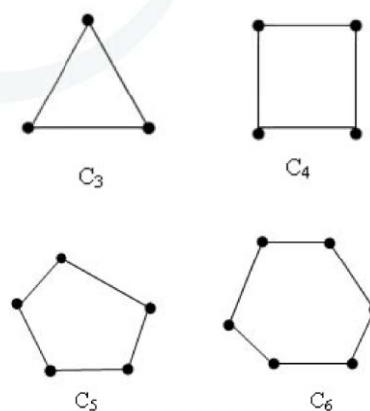


19

20

Đồ thị vòng

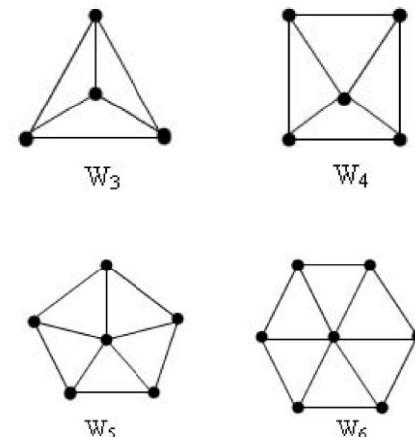
- Kí hiệu: C_n
- Đặc điểm
 - Đồ thị vô hướng n đỉnh
 - Các đỉnh nối với nhau theo vòng tròn
- Tính chất
 - Các đỉnh đều có bậc 2
 - Số cạnh: n



21

Đồ thị bánh xe

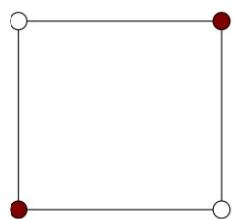
- Kí hiệu: W_n
- Đặc điểm
 - Đồ thị vô hướng
 - Hai đỉnh bất kỳ luôn kề nhau
- Tính chất
 - n+1 đỉnh
 - n đỉnh bậc 3
 - 1 đỉnh bậc n
 - Số cạnh: 2n



22

Đồ thị lưỡng phân

- Cho $G=(V,E)$
 $V = V_1 \cup V_2$
 $V_1 \cap V_2 = \emptyset$
Mỗi cạnh có 1 đỉnh $\in V_1$ và đỉnh kia $\in V_2$.
- Định lý: Đơn đồ thị là đồ thị lưỡng phân khi và chỉ khi nó không chứa chu trình độ dài lẻ.



23

Giải thuật xác định đồ thị lưỡng phân

- Đặt $X= \{v\}$ với v là đỉnh bất kỳ của đồ thị
- $Y= \{\text{tập các đỉnh kề}\} \text{ của các đỉnh trong } X\}$
- $T= \{\text{tập các đỉnh kề}\} \text{ của các đỉnh trong } Y\}$

⇒ Nếu $T \cap Y \neq \emptyset \rightarrow$ không là đồ thị lưỡng phân
⇒ Ngược lại $X=X \cup T$, tiếp tục giải thuật

24

Đồ thị lưỡng phân đú

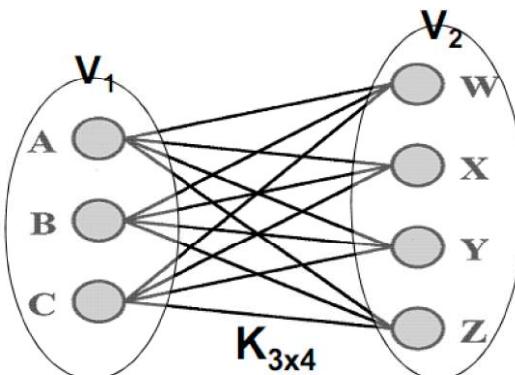
■ Kí hiệu: $K_{m,n}$

■ Đặc điểm

- Đồ thị lưỡng phân
- Mọi cặp đỉnh giữa hai tập V_1 và V_2 đều nối với nhau.

■ Tính chất

- $m + n$ đỉnh
- $m \times n$ cạnh.
- m đỉnh bậc n
- n đỉnh bậc m



25

Đẳng cấu đồ thị

■ Dùng để chỉ các đồ thị có cùng cấu trúc

■ Định nghĩa

- Cho 2 đồ thị $G_1 = (V_1, E_1)$ và $G_2 = (V_2, E_2)$ là 2 đồ thị cùng có hướng hoặc cùng vô hướng. Ta nói $G_1 \cong G_2$ nếu tồn tại song ánh

$f: V_1 \rightarrow V_2$ sao cho:

uv là cạnh của $G_1 \Leftrightarrow f(u)f(v)$ là cạnh của G_2

uv là cung của $G_1 \Leftrightarrow f(u)f(v)$ là cung của G_2

26

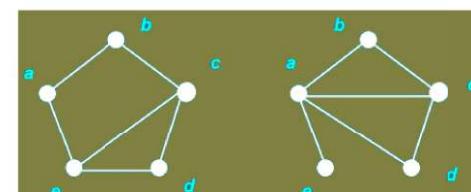
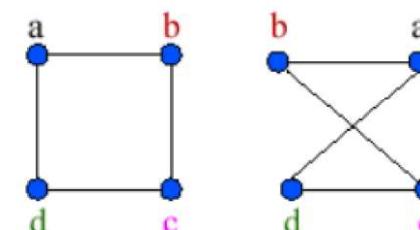
Đẳng cấu đồ thị (tt)

Chú ý:

■ Nếu $G_1 \cong G_2$ thì chúng có:

- Cùng số đỉnh
- Cùng số cạnh
- Cùng số đỉnh với bậc cho sẵn
- $\deg v = \deg f(v)$

VD: Đẳng cấu đồ thị

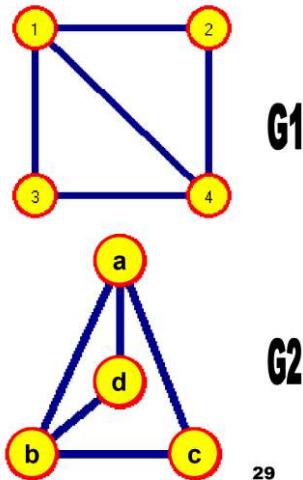


27

28

VD: Đẳng cấu đồ thị (tt)

- Xét xem 2 đồ thị G_1 , G_2 có đẳng cấu không?



29

Đồ thị con, đồ thị bộ phận

- ĐN1:** Cho $G=(V,E)$ và $G'=(V',E')$ là hai đồ thị cùng có hướng hoặc cùng vô hướng. Nếu $V' \subseteq V$, $E' \subseteq E$ thì G' là đồ thị con của G . Kí hiệu $G' \leq G$.
- ĐN2:** Cho $G' \leq G$, nếu $V'=V$ thì G' được gọi là đồ thị bộ phận của G .
- VD:** Mạng giao thông đường bộ cả nước
 - Mạng xe buýt cả nước: đồ thị bộ phận
 - Mạng giao thông kv phía nam: đồ thị con

30

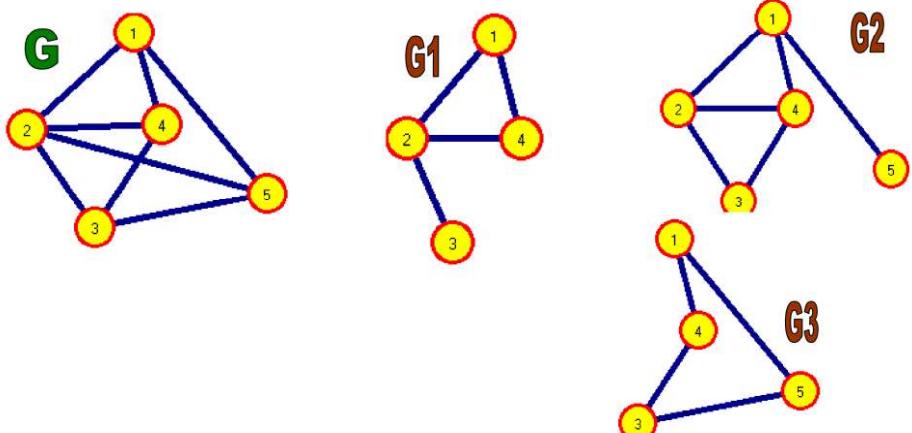
Đồ thị con sinh bởi tập đỉnh

- Cho $G=(V,E)$ và $A \subseteq V$, đồ thị con sinh bởi tập A , ký hiệu là $\langle A \rangle_G$ được định nghĩa là $\langle A \rangle_G = (A, U)$ trong đó:
 - Tập cạnh $U \subseteq E$
 - Gọi $e=(i,j) \in E$ là 1 cạnh của G , nếu $i,j \in A$ thì $e \in U$

31

VD:

- Cho biết đâu là đồ thị con; đồ thị bộ phận; đồ thị con sinh bởi tập đỉnh $\{1,3,4,5\}$ của G ?



32

Biểu diễn đồ thị trên máy tính

- Ma trận kề, ma trận trọng số
- Ma trận đỉnh – cạnh, đỉnh – cung
- Danh sách cạnh – cung
- Danh sách kề

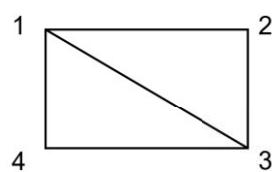
Ma trận kề

- Cho đơn đồ thị $G=(V,E)$. Lập ma trận $A=(a_{ij})_n$, với:
 - $a_{ij} = 1$ nếu có cạnh nối từ $i \rightarrow j$
 - $a_{ij} = 0$ nếu ngược lại
- Đa đồ thị:
 - $a[i,j] =$ số cạnh (cung) nối hai đỉnh i, j .

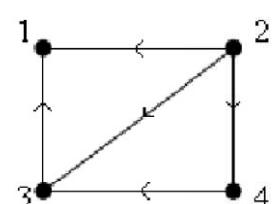
33

34

Ma trận kề (tt)

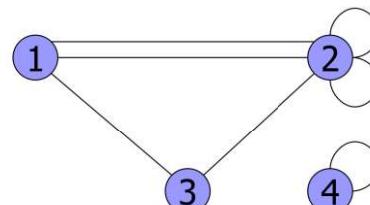


$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

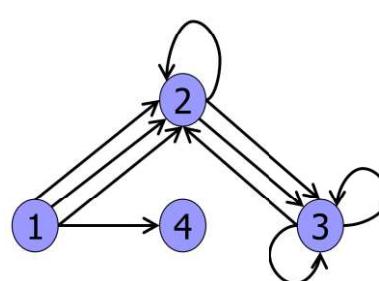


$$A = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

Ma trận kề (tt)



$$\begin{pmatrix} 0 & 2 & 1 & 0 \\ 2 & 2 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



$$\begin{pmatrix} 0 & 3 & 0 & 1 \\ 0 & 1 & 2 & 0 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

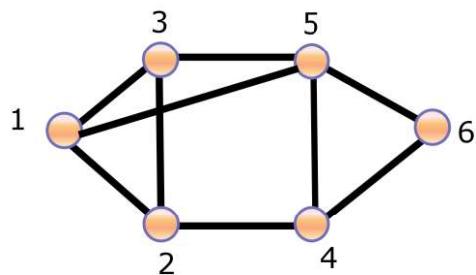
35

36

Ma trận kề (tt)

■ Tính chất của ma trận kề của đồ thị vô hướng:

- Đối xứng
- Tổng các phần tử trên dòng i (cột j) bằng bậc của đỉnh i (đỉnh j)

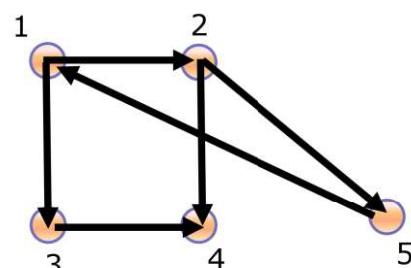


37

Ma trận kề (tt)

■ Tính chất của ma trận kề của đồ thị có hướng:

- Không đối xứng
- Tổng các phần tử trên dòng i bằng bán bậc ra của đỉnh i và tổng các phần tử trên cột j bằng bán bậc vào của đỉnh j.

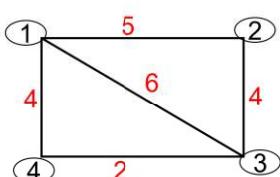


38

Ma trận trọng số

■ Lập ma trận trọng số $C=(c_{ij})_n$, với:

- $c_{ij} = c(i,j)$ nếu $(i,j) \in E$
- $c_{ij} = \theta$ nếu $(i,j) \notin E$
- θ có thể được đặt bằng một trong các giá trị sau: 0, $+\infty$, $-\infty$



$$C = \begin{bmatrix} 0 & 5 & 6 & 4 \\ 5 & 0 & 4 & 0 \\ 6 & 4 & 0 & 2 \\ 4 & 0 & 2 & 0 \end{bmatrix}$$

39

Ma trận đỉnh – cạnh, đỉnh - cung

Dòng \leftrightarrow đỉnh
Cột \leftrightarrow cạnh (cung)

■ Xét $G=(V, E)$ có hướng. Ma trận $A=(a_{ij})$ được định nghĩa như sau:

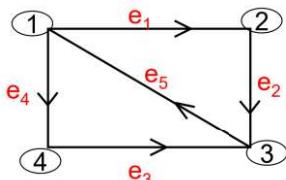
- Với cung $e(i,j) \in E$ thì $a_{ie}=1$, $a_{je}=-1$
- Ngược lại =0

■ Xét $G=(V, E)$ vô hướng. Ma trận $A=(a_{ij})$ được định nghĩa như sau:

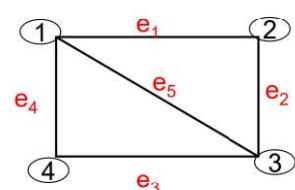
- Với cạnh $e(i,j) \in E$ thì $a_{ie}=a_{je}=1$,
- Ngược lại =0

40

Ma trận đỉnh – cạnh, đỉnh – cung (tt)



	e1	e2	e3	e4	e5
1	1	0	0	1	-1
2	-1	1	0	0	0
3	0	-1	-1	0	1
4	0	0	1	-1	0

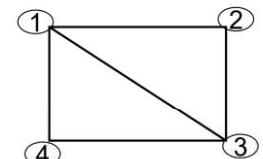


	e1	e2	e3	e4	e5
1	1	0	0	1	1
2	1	1	0	0	0
3	0	1	1	0	1
4	0	0	1	1	0

41

Danh sách cạnh - cung

- Mỗi cạnh (cung) e(x,y) tương ứng với 2 biến : **dau[e]**, **cuoi[e]**



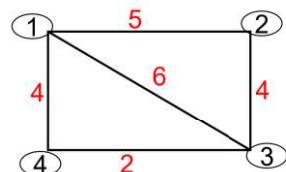
Dau	Cuoi
1	2
1	4
2	3
3	1
4	3

Dau	Cuoi
1	2
1	3
1	4
2	3
3	4

42

Danh sách cạnh – cung (tt)

- Nếu đồ thị có trọng số



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

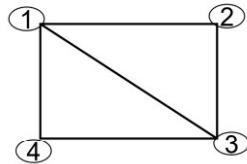
43

Danh sách kề

- Với mỗi đỉnh v của đồ thị, ta lưu trữ danh sách các đỉnh kề với nó.
- Để lưu trữ danh sách các đỉnh kề của v, ta có thể dùng **mảng** hoặc **dslk**.

44

Danh sách kề (tt)



ke = {2, 3, 4, 1, 3, 1, 2, 4, 1, 3}

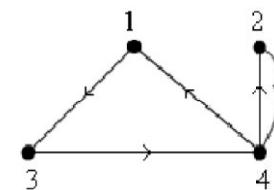
tro = {1, 4, 6, 9, 11}

Vị trí bắt đầu của ds các đỉnh kề của i

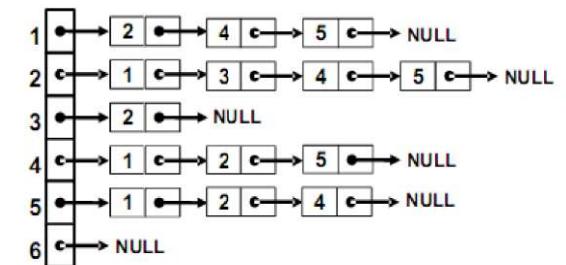
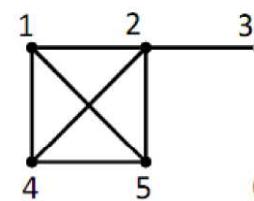
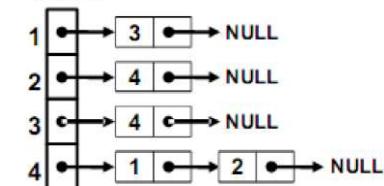
ke[tro[i]] → **ke[tro[i+1]-1]** ~ ds các đỉnh kề với i

45

Danh sách kề (tt)



ke[v]



Cài đặt danh sách kề

//Định nghĩa 1 nút

```
struct node  
{int i;  
 struct node * p;  
 } nut;
```

// Số đỉnh tối đa

```
const max=100;
```

// Mảng lưu các con trỏ của các đỉnh đầu
nut * ke[max];

Câu hỏi???

47

48

CHƯƠNG 2

Duyệt đồ thị và tính liên thông của đồ thị

1

Nội dung

■ Duyệt đồ thị

- Khái niệm duyệt đồ thị
- Thuật toán duyệt đồ thị
- Duyệt đồ thị theo chiều sâu
- Duyệt đồ thị theo chiều rộng

■ Đồ thị liên thông

- Các định nghĩa
- Tìm đường đi giữa 2 đỉnh
- Kiểm tra tính liên thông của đồ thị

2

Duyệt đồ thị theo chiều sâu (Depth-First Search)

Procedure DFS(v)

Begin

 Tham_dinh(v);
 Chuaxet[v]:=false;
 For u ∈ Ke(v) do
 If Chuaxet[u] then DFS(u)

End

void DFS(v)

{

 printf("%d", v);
 chuaxet[v]=false;
 for (i=1; i<=n; i++)
 If (a[v][i]==1 &&
 chuaxet[i]==1)
 DFS(i);

}

3

4

Ví dụ:

Duyệt đồ thị theo chiều rộng (Breadth-First Search)

Procedure BFS(v)

Begin

QUEUE := \emptyset ;

QUEUE $\leftarrow v$;

Chuaxet[v] := false;

While QUEUE $\neq \emptyset$ do

Begin

$p \leftarrow$ QUEUE;

Tham_dinh(p);

For $u \in K_e(p)$ do

if chuaxet[u] then

Begin

QUEUE $\leftarrow u$;

Chuaxet[v] := false;

End;

End;

End

5

6

Ví dụ:

Nội dung

■ Duyệt đồ thị

- Khái niệm duyệt đồ thị
- Thuật toán duyệt đồ thị
- Duyệt đồ thị theo chiều sâu
- Duyệt đồ thị theo chiều rộng

■ Đồ thị liên thông

- Các định nghĩa
- Tìm đường đi giữa 2 đỉnh
- Kiểm tra tính liên thông của đồ thị

7

8

Tính liên thông của đồ thị

- Các định nghĩa
 - Đường đi, chu trình
 - Liên thông
- Tìm đường đi giữa 2 đỉnh
- Kiểm tra tính liên thông của đồ thị

9

Đường đi, chu trình

- **ĐN1:** đường đi độ dài k từ đỉnh u → v trên đồ thị $G=(V,E)$ là dãy các đỉnh $u = x_0, x_1, \dots, x_k = v$, mà các cạnh (cung) $(x_i, x_{i+1}) \in E$.
Đỉnh u gọi là **đỉnh đầu** (đỉnh xuất phát), đỉnh v gọi là **đỉnh cuối** (đỉnh đích) của đường đi.
- Đường đi có **đỉnh đầu trùng với đỉnh cuối** gọi là **chu trình**.

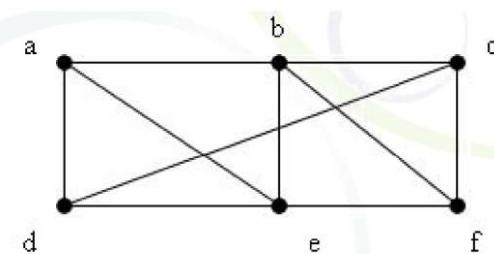
10

Đường đi, chu trình

- Đường đi đgl **đơn** nếu đường đi không có **cạnh / cung** nào bị lặp lại.
- Chu trình đgl **đơn** nếu chu trình không có **cạnh / cung** nào bị lặp lại (trừ đỉnh đầu và đỉnh cuối).
- Đường đi hay chu trình đgl **sơ cấp** nếu không có **đỉnh** nào bị lặp lại (trừ đỉnh đầu và đỉnh cuối).

11

Ví dụ



- a, d, c, f, e là đường đi đơn độ dài 4
d, e, c, a không là đường đi, do $(c,e) \notin E$
b, c, f, e, b là chu trình độ dài 4
a, b, e, d, a, b là đường đi độ dài là 5 (không đơn)

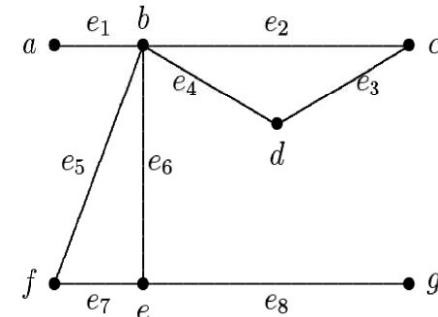
12

Đồ thị liên thông

- **ĐN2:** Đồ thị vô hướng $G=(V,E)$ đgl liên thông nếu luôn tìm được đường đi giữa 2 đỉnh bất kỳ của nó.
- **ĐN3:** Cho $G'=(V',E')$ là đồ thị con của $G=(V,E)$. G' đgl thành phần liên thông của G nếu
 - G' liên thông
 - không tồn tại đường đi nào từ 1 đỉnh thuộc G' tới 1 đỉnh không thuộc G'
- Vậy, nếu G chỉ có 1 thành phần liên thông thì G liên thông

13

Ví dụ

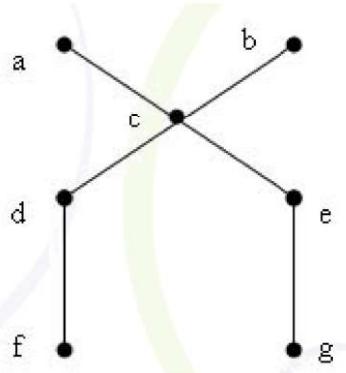


Hãy xác định:

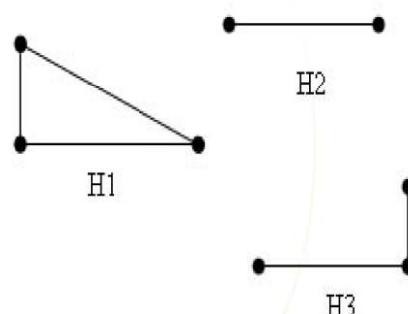
1. Đường đi đơn độ dài 3; 4
2. Đường đi sơ cấp độ dài 3; 4
3. Chu trình
4. Chu trình đơn
5. Chu trình sơ cấp

14

Ví dụ:



Hình 1



Hình 2

Đồ thị liên thông (tt)

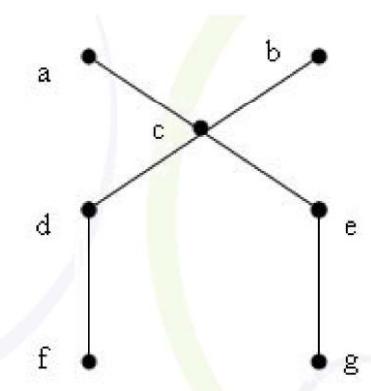
- **ĐN4:** Cho $G=(V,E)$ là đồ thị vô hướng, liên thông. Đỉnh **u** đgl **đỉnh rẽ nhánh** (đỉnh khớp, đỉnh cắt) của G , nếu như xoá đỉnh u và các cạnh kề với u thì đồ thị có được là không liên thông.
- Cạnh **e** đgl **cầu** (cạnh cắt), nếu như xoá nó khỏi G thì đồ thị có được không liên thông

15

16

Ví dụ

- Đỉnh **d** và **e** là đỉnh rẽ nhánh.
- Cạnh **(d,f)** và **(e,g)** là cầu.



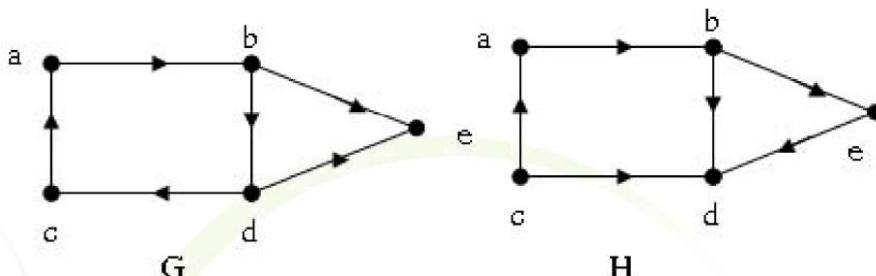
17

Đồ thị liên thông (tt)

- ĐN5: Cho $G=(V,E)$ có hướng, G đgl **liên thông mạnh** nếu giữa 2 đỉnh u, v bất kỳ luôn luôn có đường đi từ $u \rightarrow v$ và ngược lại.
- ĐN5: Cho $G=(V,E)$ có hướng, G đgl **liên thông yếu** nếu khi coi các cung là các cạnh thì đồ thị vô hướng tương ứng là liên thông.

Ví dụ

- Đồ thị liên thông mạnh G
- Đồ thị liên thông yếu H



19

Tìm đường đi giữa 2 đỉnh

- Bài toán: Cho s, t là 2 đỉnh của đồ thị. Tìm đường đi từ $s \rightarrow t$.
- Phương pháp:
 - Dùng DFS(s) và BFS(s)
 - Nếu t được “thăm” \rightarrow tồn tại đường đi. Ngược lại, không tồn tại đường đi từ $s \rightarrow t$.
 - Thêm 1 mảng để ghi nhận đường đi
`truoc[i]=j // trước đỉnh i là đỉnh j`

20

Tìm các thành phần liên thông

- Nếu đồ thị liên thông \leftrightarrow số thành phần liên thông =1.
- Nếu đồ thị không liên thông \leftrightarrow số thành phần liên thông >1 .

21

Tìm các thành phần liên thông (tt)

- **B1:** Khởi tạo biến $solt=0$, đánh dấu tất cả các đỉnh đều là 0.
- **B2:** Với mọi đỉnh $i \in V$
 Nếu đỉnh i được đánh dấu =0
 $solt = solt+1$
 Thăm và đánh dấu đỉnh i
 Cuối nếu
Cuối với mọi

22

Tìm các thành phần liên thông (tt)

Hàm thăm và đánh dấu đỉnh i:

```
DFS(i, solt)      // hoặc BFS(i, solt)
{
    ....
    chuaxet[i]=solt; // thay vì chuaxet[i]=true;
    ....
}
```

23

Câu hỏi???

24

CHƯƠNG 3

Cây và cây khung

1

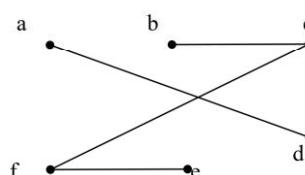
Cây và cây khung

- Giới thiệu
- Cây là gì?
- Tính chất cơ bản của cây
- Cây khung
- Cây khung nhỏ nhất

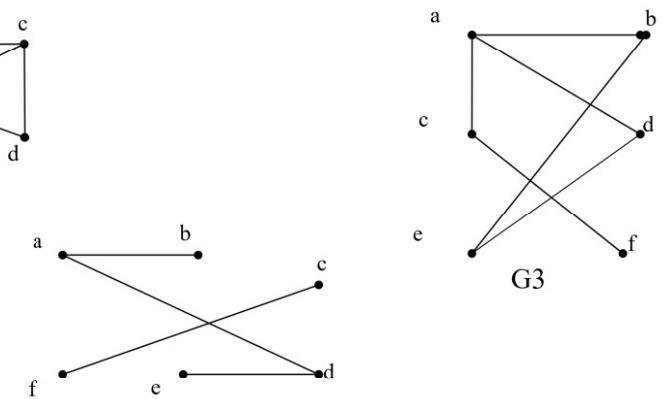
2

Cây là gì? (tt)

- Ví dụ: Đồ thị nào dưới đây là cây?



G1



Cây là gì?

- ĐN1: Cây là đồ thị vô hướng liên thông không có chu trình.
- ĐN2: Rừng là ĐT mà mỗi thành phần liên thông của nó là một cây.

3

4

Cấu trúc cây (cây có gốc)

- Là ĐT có hướng, mỗi đỉnh đều có 1 đỉnh trước trừ 1 phần tử duy nhất không có gọi là **GỐC**.
- Xét một đỉnh x của một cây T có gốc là r :
 - Đỉnh y nằm trên đường hướng từ gốc đến x, gọi là **ĐỈNH TRƯỚC** của x, và x là **ĐỈNH SAU** của y.
 - Nếu (x,y) là một cạnh của T, ta gọi x là **CHA** của y và y là **CON** của x.
 - Hai đỉnh cùng cha gọi là **ANH EM**.
 - Đỉnh không có con gọi là **LÁ**.
 - Đỉnh không là lá gọi là **ĐỈNH TRONG**.

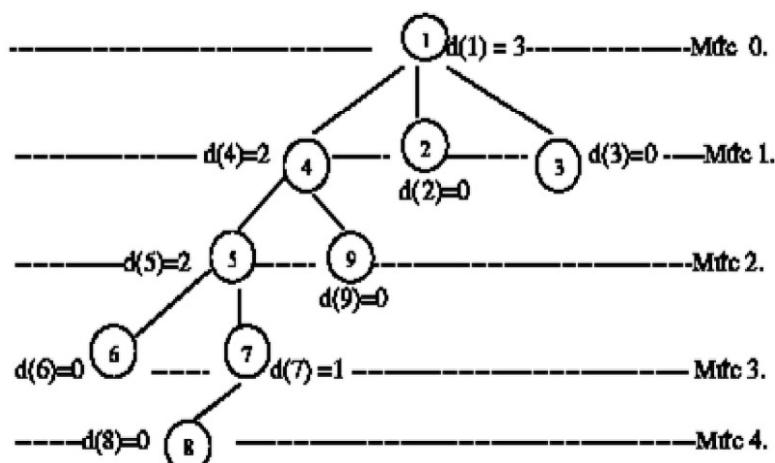
5

Cấu trúc cây (tt)

- Mức của đỉnh là khoảng cách từ gốc đến đỉnh.
 - Mức của nút gốc = 0.
- Chiều cao của cây = mức lớn nhất của đỉnh.
- Độ sâu của nút & độ sâu của cây .
 - Độ sâu của nút là số con của nút đó.
 - Độ sâu của cây là độ sâu lớn nhất của các nút của cây.
 - Nếu cây có độ sâu là k, ta gọi là cây k-phân.
 - Nếu mỗi đỉnh trong cây có tối đa hai con, thì ta gọi đó là cây nhị phân.

6

Cấu trúc cây (tt)



7

Các tính chất cơ bản của cây

- Cho $G=(V,E)$ là đồ thị vô hướng n đỉnh, **các mệnh đề sau đây là tương đương**:
 - (1). T là cây.
 - (2). T không chứa chu trình và có $n-1$ cạnh.
 - (3). T liên thông và có $n-1$ cạnh.
 - (4). T liên thông và mỗi cạnh của nó đều là cầu.
 - (5). 2 đỉnh bất kỳ của T được nối với nhau bởi đúng 1 đường đi đơn.
 - (6). T không chứa chu trình nhưng nếu thêm vào 1 cạnh ta thu được đúng 1 chu trình.

8

Một số ứng dụng của cây

■ Cây tìm kiếm nhị phân:

- Tìm 1 phần tử trong ds.

■ Cây quyết định:

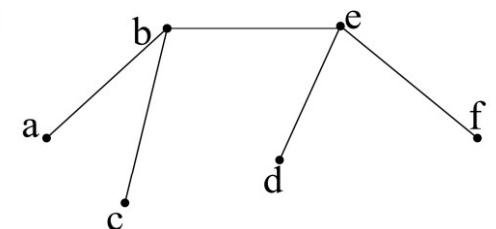
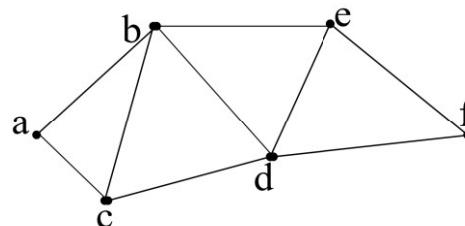
- Mỗi **đỉnh trong** ứng với 1 quyết định
- Mỗi cây con tại các đỉnh này ứng với 1 kết quả của quyết định.
- Những lời giải tương ứng với các đường đi tới các lá của cây.

9

Cây khung

■ Giới thiệu bài toán:

Hệ thống giao thông ở thành phố A



10

Cây khung (tt)

■ Định nghĩa:

- Cho G là một đơn đồ thị. Một cây được gọi là **cây khung của G** nếu nó là một **đồ thị con** của G và **chứa tất cả các đỉnh** của G .

■ Định lý (sự tồn tại cây khung)

- Mọi đồ thị liên thông đều chứa ít nhất một cây khung

11

Cây khung (tt)

■ Tìm cây khung

- **B1:** Chọn tùy ý 1 đỉnh của G đặt vào H
- **B2:** Nếu mọi đỉnh của G đều nằm trong H thì dừng
- **B3:** Nếu không, tìm 1 đỉnh $\in G$ nhưng $\notin H$, sao cho 1 cạnh e nào đó của G nối nó với 1 đỉnh của H . Thêm đỉnh và cạnh này vào H .
Quay về B2

12

Cây khung (tt)

- Giải thuật tìm cây khung
 - Tìm cây khung bằng DFS
 - Tìm cây khung bằng BFS

Cây khung (tt)

- Tìm cây khung của đồ thị dưới đây bằng giải thuật DFS và BFS?

13

14

Cây khung nhỏ nhất

- ĐN: Cho đồ thị G vô hướng, liên thông và có trọng số không âm. Cây khung nhỏ nhất của đồ thị G là cây khung có tổng trọng số trên các cạnh của nó nhỏ nhất.
- Giải thuật:
 - Giải thuật **Kruskal**
 - Giải thuật **Prim**

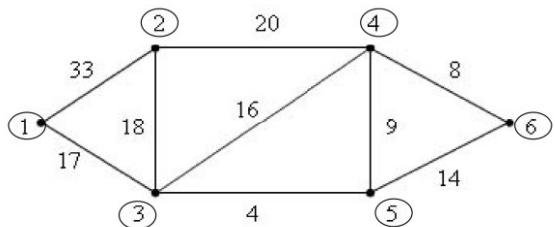
Giải thuật **Kruskal**

- **B1.** Sắp xếp các cạnh có trọng số *tăng dần*
- **B2.** Lần lượt chọn cạnh có *trọng số nhỏ nhất* trong số các cạnh còn lại để đưa vào cây nếu sau khi đưa vào *không tạo thành chu trình*.
- **B3.** Chọn đủ $n-1$ cạnh thì dừng.

15

16

Giải thuật Kruskal (tt)



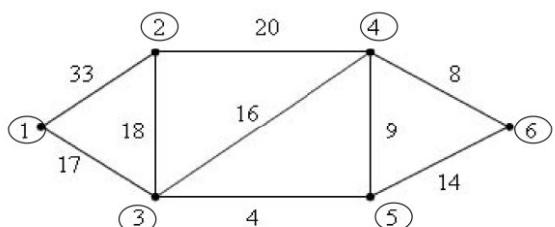
17

Giải thuật Prim

- **B1.** Chọn đỉnh đầu tiên *tuỳ ý* đưa vào cây (nếu không yêu cầu thì có thể chọn **đỉnh 1**).
- **B2.** Lần lượt chọn $n-1$ đỉnh còn lại (tương ứng $n-1$ cạnh) vào cây khung bằng cách: mỗi lần *chọn 1 cạnh có trọng số nhỏ nhất* mà **một đầu của cạnh đã thuộc cây, đầu kia chưa thuộc cây**.

18

Giải thuật Prim



19

Giải thuật Prim (tt)

```
//tim cạnh có trọng số nhỏ nhất trong các cạnh có 1 đầu đã nạp và 1 đầu chưa nạp vào cây khung nhỏ nhất đang hình thành
void timcanh (int *x, int *y)
{min=32000;
for (i=1; i<=n; i++)
    if (chuaxet[i]==1)           //đỉnh i đã đưa vào cây khung
        for (j=1; j<=n; j++)
            if (chuaxet[j]==0)     //đỉnh chưa xét
                if (a[i][j]>0 and a[i][j]<min)
                    { *x=i;
                      *y=j;
                      min=a[i][j];
                    }
    }
```

20

Giải thuật Prim (tt)

- Trong quá trình cài đặt, để chọn đỉnh và cạnh cần bổ sung vào cây khung, ta có thể dùng **phương pháp gán nhãn cho đỉnh**:
- Mỗi đỉnh **s** được gán cho 1 nhãn là **($d[s]$, $near[s]$)** với ý nghĩa:
 - $near[s]$ là đỉnh đã nạp vào cây khung mà **kè và gần s nhất**. $near[s]$ có thể gọi là đỉnh trước của s
- $d[s]$ là độ dài cạnh ($near[s]$, s)

21

Giải thuật Prim (tt)

- B1.** Khởi tạo và gán nhãn ban đầu.
 - Lấy 1 đỉnh **s** tuỳ ý đưa vào cây khung. Khi đó tập đỉnh của cây khung có duy nhất 1 đỉnh là $VH = \{s\}$. Tập cạnh T là rỗng.
 - Với mỗi đỉnh **v** kề s thì gán nhãn $(d[v], s)$, với $d[v]$ là trọng số cạnh (s, v) .
 - Các đỉnh **v** không kề s thì gán nhãn (∞, s)

22

Giải thuật Prim (tt)

- B2.** Vòng lặp cho đến khi chọn đủ $n-1$ cạnh (hoặc xét hết các đỉnh)
- Vòng lặp thực hiện các thao tác sau:**
 - Đưa đỉnh và cạnh và cây: Chọn đỉnh **u** ngoài tập **D**, có nhãn $d[u]$ nhỏ nhất, nạp đỉnh **u** đó vào tập **D**. Vậy $D=D+\{u\}$. nạp cạnh (s, u) vào tập cạnh **T**.
 - Nếu số cạnh (hoặc đỉnh) thoả \rightarrow kết thúc vòng lặp
 - Sửa nhãn: sau khi đỉnh **u** nạp vào cây khung, nó sẽ tạo cho 1 số đỉnh kề với nó gần cây khung hơn, nên phải sửa lại nhãn. Cụ thể: xét các đỉnh **v** $\notin D$, nếu $d[v] > c[u][v]$ thì $d[v] = c[u][v]$.

23

Câu hỏi???

24

Chương 4

ĐỒ THỊ PHẲNG VÀ TÔ MÀU ĐỒ THỊ

1

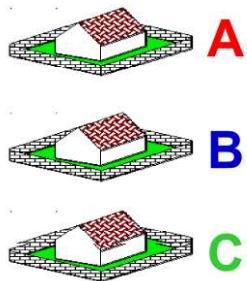
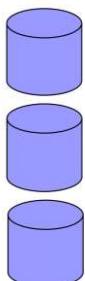
Nội dung

- Bài toán
- Khái niệm đồ thị phẳng
- Công thức Euler
- Định lý Kuratowski
- Tô màu đồ thị
- Một số ứng dụng

2

Khái niệm đồ thị phẳng

- Mô hình hóa bài toán bằng đồ thị lưỡng phân đủ $K_{3,3}$
- Câu hỏi: “Có hay không cách vẽ đồ thị $K_{3,3}$ trên một mặt phẳng sao cho không có hai cạnh nào cắt nhau?”



3

Bài toán

- Có 3 nhà và 3 cái giếng. Các nhà đều muốn thiết kế đường dây nối từ nhà đến mỗi giếng nhưng không được cắt nhau. Hỏi phải thiết kế làm sao cho các dây nối không cắt nhau?

4

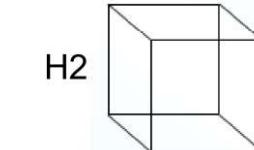
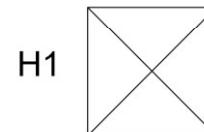
Khái niệm đồ thị phẳng (tt)

- **Định nghĩa:** Đồ thị vô hướng G là **đồ thị phẳng** nếu ta có thể biểu diễn nó trên một mặt phẳng sao cho không có cạnh nào cắt nhau.
- **Chú ý:** một đồ thị có thể được vẽ bằng nhiều cách khác nhau. Nếu tồn tại một cách vẽ thỏa mãn định nghĩa trên thì nó là đồ thị phẳng

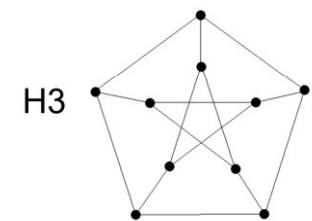
5

Khái niệm đồ thị phẳng (tt)

- VD:



Đồ thị
phẳng



Không là đồ
thị phẳng

6

Công thức Euler

- **Định lý:** Cho G là đồ thị phẳng, liên thông với n đỉnh và e cạnh. Gọi d là số miền trong biểu diễn phẳng của G . Khi đó, ta có:

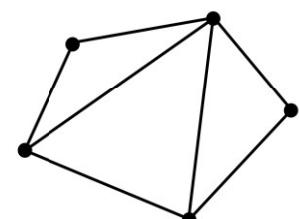
$$d = e - n + 2$$

7

Công thức Euler (tt)

- VD:

Số miền biểu diễn
phẳng của đồ thị?



8

Công thức Euler (tt)

- VD: CMR $K_{3,3}$ không là đồ thị phẳng.
- Giải

Công thức Euler (tt)

- **Hệ quả 1.** Nếu G là đơn đồ thị phẳng liên thông với e cạnh, n đỉnh, trong đó $n \geq 3$. Khi đó ta có:

$$e \leq 3n - 6$$

- **Hệ quả 2.** Trong một đồ thị phẳng liên thông tùy ý, luôn tồn tại ít nhất một đỉnh có bậc không vượt quá 5.

9

10

Công thức Euler (tt)

- **Hệ quả 3.** Giả sử G là đơn đồ thị phẳng liên thông với n đỉnh và e cạnh và *không chứa chu trình đơn độ dài 3*. Khi đó

$$e \leq 2n - 4$$

Định lý Kuratowski

- **Định lý:** Đồ thị G không là đồ thị phẳng khi và chỉ khi G chứa đồ thị con đồng phôi với K_5 hoặc $K_{3,3}$

- K_5 và $K_{3,3}$ là các đồ thị không phẳng đơn giản nhất, vì khi xóa bất kỳ đỉnh hoặc cạnh của các đồ thị này sẽ nhận được đồ thị phẳng.
- K_5 là đồ thị không phẳng ít đỉnh nhất.
- $K_{3,3}$ là đồ thị không phẳng ít cạnh nhất.

11

12

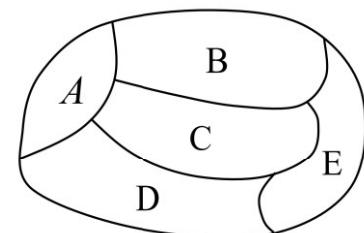
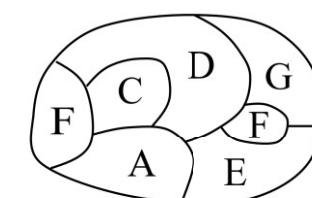
Tô màu đồ thị



13

Tô màu đồ thị (tt)

- Để phân biệt các miền trên bản đồ ta phải tô màu chúng bằng các màu khác nhau
- Cần ít nhất bao nhiêu màu để tô một bản đồ bất kỳ sao cho các miền kề nhau không cùng một màu?



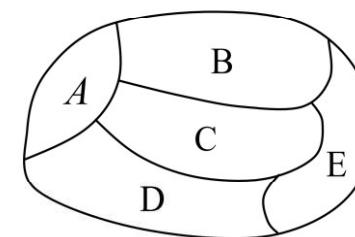
14

Tô màu đồ thị (tt)

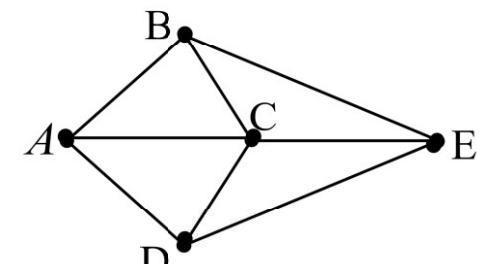
Mô hình bài toán:

- Mỗi miền tương ứng một đỉnh của đồ thị
- Hai đỉnh có cạnh nối nếu chúng là hai miền có chung biên giới
- Đồ thị nhận được gọi là đồ thị đối ngẫu của bản đồ. Đồ thị đối ngẫu của bản đồ là đồ thị phẳng.

Tô màu đồ thị (tt)



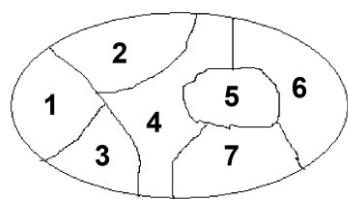
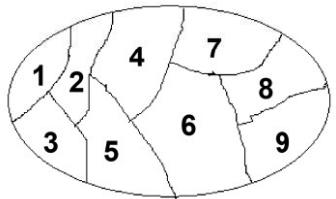
Bản đồ



Đồ thị

Tô màu đồ thị (tt)

- Hãy biểu diễn đồ thị cho các bản đồ sau?



17

Tô màu đồ thị (tt)

- Định lý 1:** Mọi chu trình độ dài lẻ đều có sắc số là 3.
- Định lý 2:** Đồ thị G có sắc số là 2 khi và chỉ khi G không có chu trình độ dài lẻ.
- Định lý 3: (Định lý 5 màu của Kempe-Heawood):** Mọi đồ thị phẳng đều có thể tô đúng bằng 5 màu.

18

Tô màu đồ thị (tt)

- Định lý 4 (Định lý 4 màu của Appel-Haken, 1976):** Mọi đồ thị phẳng đều có sắc số không lớn hơn 4.
- Lưu ý**

Định lý 4 màu chỉ đúng với đồ thị phẳng, với đồ thị không phẳng: **có thể cần nhiều hơn 4 màu.**

19

Tô màu đồ thị (tt)

- Sắc số của một số đồ thị cơ bản**
 - Đồ thị lưỡng phân đủ có sắc số là **2**.
 - Đồ thị vòng có sắc số là **2** nếu số đỉnh n chẵn, là **3** nếu số đỉnh n lẻ.
 - Đồ thị bánh xe ($n \geq 4$) có sắc số là **4** nếu số đỉnh n chẵn, là **3** nếu số đỉnh n lẻ.
 - Đồ thị đầy đủ K_n có sắc số là **n**.

20

Thuật toán tô màu đồ thị

Liệt kê các tập V' các đỉnh của đồ thị theo thứ tự
bậc giảm dần.

k=1

while $V' \neq \emptyset$ do

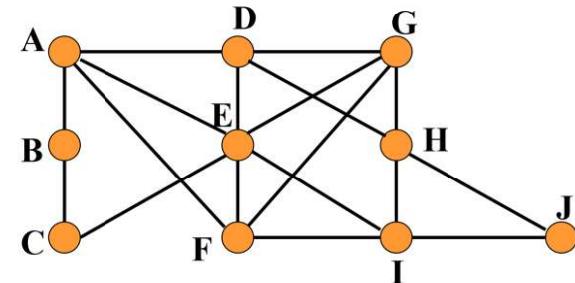
- { Tô màu k cho đỉnh v đầu tiên trong danh sách
- Tô màu k cho các đỉnh tiếp theo không kề v
- Loại khỏi V' các đỉnh đã tô màu
- k=k+1**

}

21

Thuật toán tô màu đồ thị (tt)

- **VD:** tô màu cho đồ thị dưới đây



22

Thuật toán tô màu đồ thị (tt)

■ Giải

Một số bài toán ứng dụng

- Các bài toán xếp lịch
 - Lịch thi
 - Lịch công tác
 - Thời khoá biểu
- Bài toán điều khiển đèn các nút giao thông

23

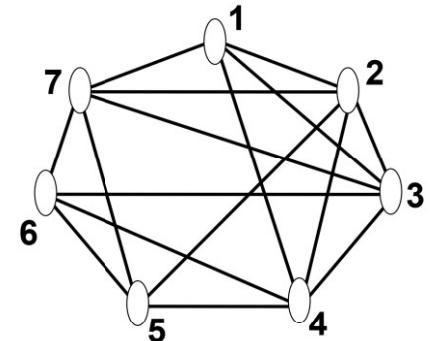
24

Một số bài toán ứng dụng (tt)

- Xếp lịch: xếp lịch thi trong một trường sao cho không có sinh viên (lớp) nào thi hai môn cùng một lúc.
- **Giải pháp:**
 - Mỗi môn học là một đỉnh
 - Nếu 2 môn học nào được dự thi bởi cùng 1 sinh viên thì sẽ nối bằng 1 cạnh.
 - Cách lập lịch sẽ tương ứng với bài toán tô màu của đồ thị này.

Một số bài toán ứng dụng (tt)

- **VD:** Có 7 môn thi với thông tin SV dự thi như sau:
 - Môn 1: có sv A, B, C và D
 - Môn 2: có sv A, E, F, G và H
 - Môn 3: có sv B, E, I, J và K
 - Môn 4: có sv B, F, L và M
 - Môn 5: có sv G, L, N và O
 - Môn 6: có sv J, M, N và P
 - Môn 7: có sv D, H, K, O và P



Câu hỏi???

CHƯƠNG 5

Đường đi ngắn nhất

Đường đi ngắn nhất

- Bài toán tìm đường đi
- Các thuật toán tìm đường đi ngắn nhất
 - Thuật toán Dijkstra
 - Thuật toán Ford-Bellman
 - Thuật toán Floyd

1

2

Bài toán tìm đường đi

- **Bài toán:** Cho đồ thị $G = (V, E)$ và hai đỉnh a, b . Tìm đường đi ngắn nhất (*nếu có*) đi từ đỉnh a đến đỉnh b trong đồ thị G .

- **Ý nghĩa thực tế:**

Bài toán giúp chúng ta chọn các hành trình tiết kiệm nhất (quãng đường, thời gian, chi phí ...) trong giao thông, lập lịch thi công công trình một cách tối ưu, xử lý trong truyền tin ...

Bài toán tìm đường đi (tt)

- Xét mô hình hệ thống đường hàng không:
 - Thành phố \rightarrow đỉnh
 - Chuyến bay \rightarrow cạnh
 - Trọng số: khoảng cách giữa các TP, thời gian của chuyến bay, tiền vé
- Xét mô hình mạng máy tính:
 - Máy tính \rightarrow đỉnh
 - Đường truyền \rightarrow cạnh
 - Trọng số: khoảng cách giữa các MT, thời gian đáp ứng qua mạng của máy.

3

4

Bài toán tìm đường đi (tt)

- Xác định đường đi ngắn nhất giữa 2 đỉnh của đồ thị là xác định tổng trọng số của các cạnh trên đường đi.

5

Thuật toán Dijkstra

- Do E.Dijkstra người Hà Lan đề xuất
- **Bài toán:** Cho đồ thị có trọng số không âm. Tìm đường đi (sơ cấp) ngắn nhất từ đỉnh s đến các đỉnh còn lại.

6

Thuật toán Dijkstra (tt)

■ Tổ chức dữ liệu:

- **a[n][n]**: ma trận trọng số.
- **d[]**: lưu khoảng cách từ đỉnh s đến các đỉnh còn lại.
- **chuaxet[]**: đánh dấu các đỉnh đã có độ dài tối ưu từ đỉnh bắt đầu s hay chưa.
- **truoc[]**: nếu **truoc[i]=j** thì đỉnh j là đỉnh ngay trước đỉnh i trên hành trình đường đi ngắn nhất.

7

Thuật toán Dijkstra (tt)

■ Thuật toán

B1: Khởi tạo

- $d[i] = a[s][i]$
 - $Truoc[i] = s$
 - $Chuaxet[v] = false$
 - Gán $d[s] = 0$, $chuaxet[s] = true$
- } với $i=1, n$

8

Thuật toán Dijkstra (tt)

B2. Vòng lặp

- Tìm đỉnh u chưa xét và có $d[u]$ nhỏ nhất.
- Nếu không tìm được đỉnh u thoả đk (hoặc $u=y$ là đỉnh đích của đường đi) \rightarrow thoát vòng lặp. Ngược lại:
 - đánh dấu $chuaxet[u]=true$
 - sửa lại giá trị lưu trong mảng d cho các đỉnh v kề với đỉnh u vừa tìm được theo công thức:
nếu $d[v] > d[u]+a[u][v]$ thì:
$$d[v]=d[u]+a[u][v]$$

$$truoc[v]=u$$

9

Thuật toán Dijkstra (tt)

B3. Tìm và ghi kết quả

- $D[i]$ chính là độ dài đường đi ngắn nhất từ đỉnh s đến i . Lần ngược theo mảng $truoc$ để in ra kết quả đường đi.

10

Thuật toán Dijkstra (tt)

■ VD

Thuật toán Ford-Bellman

- **Bài toán:** Cho đồ thị có trọng số, không có chu trình âm.
- Tìm đường đi ngắn nhất từ đỉnh s đến tất cả các đỉnh còn lại của đồ thị.

11

12

Thuật toán Ford-Bellman (tt)

- B1: Khởi tạo
- B2: Sửa nhãn khoảng cách d[]
 - Thực hiện n-2 lần sửa nhãn cho các đỉnh. Trong mỗi lần sửa, nếu (i,j) là một cạnh/cung của đồ thị và $d[j] > d[i] + a[i,j]$ thì sửa lại $d[j] = d[i] + a[i,j]$
 - Sau n-2 lần sửa nhãn khoảng cách thì $d[i]$ chính là độ dài đường đi ngắn nhất từ đỉnh s đến i.
- B3: Tìm và ghi kết quả
- Ví dụ: giáo trình tr103

13

Thuật toán Ford-Bellman (tt)

- VD

14

Thuật toán Floyd

- Tổ chức dữ liệu và khởi tạo
 - $A[n][n]$: ma trận trọng số
 - $A_{ij} = 0$ nếu $i = j$
 - $A_{ij} = a[i,j]$ nếu $(i,j) \in E$
 - $A_{ij} = \infty$ nếu $(i,j) \notin E$
 - $P[n][n]$: ma trận các đỉnh trước
 - $P_{ij} = i$, trong đó P_{ij} là đỉnh trước của đỉnh j trên đường đi từ i đến j

15

Thuật toán Floyd (tt)

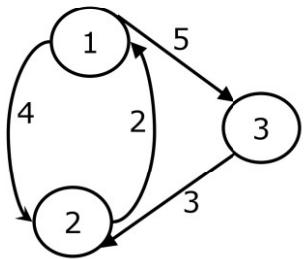
- Vòng lặp: thực hiện n lần....

```
for (k = 1; k ≤ n ; k++)
    for (i = 1 ; i ≤ n ; i++)
        for (j = 1 ; j ≤ n ; j++)
            if (a[i,k] + a[k,j] < a[i,j])
            {
                a[i,j] = a[i,k] + a[k,j];
                p[i,j] = p[k,j];
            }
```
- Kết thúc thuật toán:
 - P_{ij} = đỉnh trước của j trên đường đi ngắn nhất từ i đến đỉnh j, với chiều dài tương ứng là A_{ij} .

16

Thuật toán Floyd (tt)

■ Ví dụ:



$$A_0 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 5 \\ \hline 2 & 2 & 0 & \infty \\ \hline 3 & \infty & 3 & 0 \\ \hline \end{array}$$

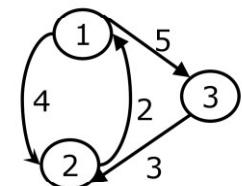
$$P_0 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 1 & 1 & 1 \\ \hline 2 & 2 & 2 & 2 \\ \hline 3 & 3 & 3 & 3 \\ \hline \end{array}$$

17

Thuật toán Floyd (tt)

$$A_0 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 5 \\ \hline 2 & 2 & 0 & \infty \\ \hline 3 & \infty & 3 & 0 \\ \hline \end{array}$$

$k = 1$



18

$$A_1 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 5 \\ \hline 2 & 2 & 0 & 7 \\ \hline 3 & \infty & 3 & 0 \\ \hline \end{array}$$

$$P_1 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 1 & 1 & 1 \\ \hline 2 & 2 & 2 & 1 \\ \hline 3 & 3 & 3 & 3 \\ \hline \end{array}$$

Thuật toán Floyd (tt)

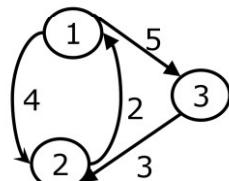
$$A_1 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 5 \\ \hline 2 & 2 & 0 & 7 \\ \hline 3 & \infty & 3 & 0 \\ \hline \end{array}$$

$k = 2$

$$A_2 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 5 \\ \hline 2 & 2 & 0 & 7 \\ \hline 3 & 5 & 3 & 0 \\ \hline \end{array}$$

$$P_2 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 1 & 1 & 1 \\ \hline 2 & 2 & 2 & 1 \\ \hline 3 & 2 & 3 & 3 \\ \hline \end{array}$$

19



Thuật toán Floyd (tt)

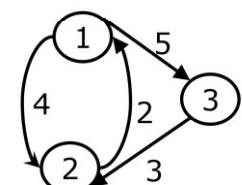
$$A_2 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 5 \\ \hline 2 & 2 & 0 & 7 \\ \hline 3 & 5 & 3 & 0 \\ \hline \end{array}$$

$k = 3$

$$A_3 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 5 \\ \hline 2 & 2 & 0 & 7 \\ \hline 3 & 5 & 3 & 0 \\ \hline \end{array}$$

$$P_3 = \begin{array}{|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 1 & 1 & 1 \\ \hline 2 & 2 & 2 & 1 \\ \hline 3 & 2 & 3 & 3 \\ \hline \end{array}$$

20



Thuật toán Floyd (tt)

■ Đường đi ngắn nhất từ i đến j:

- A_{ij} cho biết **độ dài đường đi**
- Dòng thứ i của ma trận P cho biết đường đi

■ VD: đđnnn từ 2 → 3 ?

- $P[2,3]=1$: 1 là đỉnh trước của 3
- $P[2,1]=2$: 2 là đỉnh trước của 1
- Kết quả đđnnn là: $2 \rightarrow 1 \rightarrow 3$

$$A_3 = \begin{array}{|c|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 0 & 4 & 5 \\ \hline 2 & 2 & 0 & 7 \\ \hline 3 & 5 & 3 & 0 \\ \hline \end{array}$$
$$P_3 = \begin{array}{|c|c|c|c|} \hline & 1 & 2 & 3 \\ \hline 1 & 1 & 1 & 1 \\ \hline 2 & 2 & 2 & 1 \\ \hline 3 & 2 & 3 & 3 \\ \hline \end{array}$$

Câu hỏi???

CHƯƠNG 6

Đồ thị Euler và Hamilton

1

Đồ thị Euler và Hamilton

■ Đồ thị Euler

- Giới thiệu, định nghĩa
- Các định lý và hệ quả
- Thuật toán tìm chu trình Euler

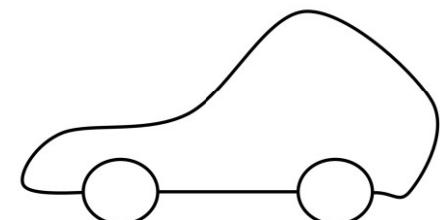
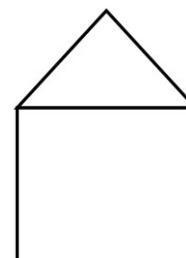
■ Đồ thị Hamilton

- Giới thiệu, định nghĩa
- Điều kiện đủ để là đồ thị Hamilton
- Qui tắc chỉ ra chu trình Hamilton hay không Hamilton
- Thuật toán tìm chu trình Hamilton

2

Câu hỏi???

■ Hình nào dưới đây có thể vẽ bằng 1 nét bút?



Đồ thị Euler

■ Đồ thị Euler

- Giới thiệu, định nghĩa
- Các định lý và hệ quả
- Thuật toán tìm chu trình Euler

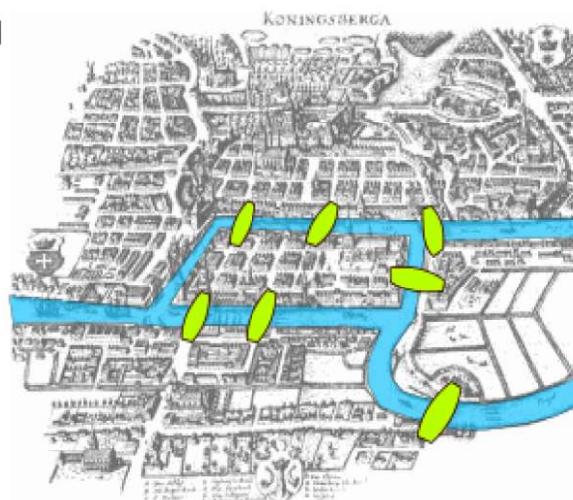
3

4

Giới thiệu

■ Bài toán 7 cây cầu

Hỏi có thể đi qua cả 7 cây cầu, mỗi cầu đúng một lần, rồi quay về chỗ xuất phát được hay không?



5

Giới thiệu (tt)

- 1736, nhà toán học Thụy Sỹ, Leonhard Euler đã giải bài toán này.
- Lời giải của Ông cho bài toán bảy cây cầu ở Königsberg được coi là **định lý đầu tiên** của lý thuyết đồ thị.



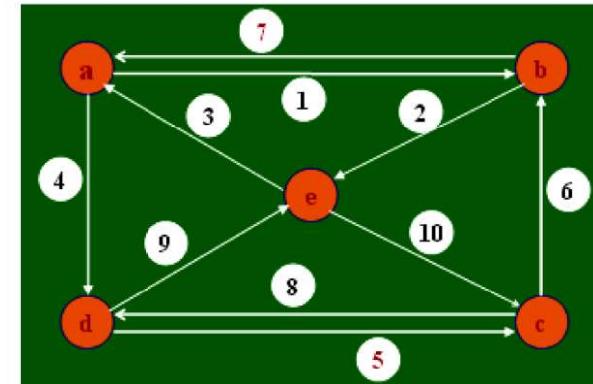
Leonhard Euler
(April 1707 – September 1783)

6

Các định nghĩa

- Đường đi qua tất cả các **cạnh/cung** của đồ thị, mỗi cạnh/cung **đúng một lần** gọi là đường đi Euler.
- Đường đi Euler có điểm đầu trùng với điểm cuối gọi là chu trình Euler.
- Đồ thị có đường đi Euler là **đồ thị nửa Euler**.
- Đồ thị có chu trình Euler gọi là **đồ thị Euler**.

Ví dụ



Chu trình Euler: $E = [1, 2, 3, 4, 5, 8, 9, 10, 6, 7]$

7

8

Các định lý, hệ quả

- **Định lý 1:** Đồ thị vô hướng, liên thông G là đồ thị Euler khi và chỉ khi mọi đỉnh của G đều có bậc chẵn.
- **Hệ quả 1:** Đồ thị vô hướng, liên thông G là nửa Euler khi và chỉ khi nó có không quá 2 đỉnh bậc lẻ.

9

Các định lý, hệ quả (tt)

- **Định lý 2:** Đồ thị có hướng liên thông mạnh G là đồ thị Euler khi và chỉ khi $\deg^+(v) = \deg^-(v) \forall v \in V$
- **Hệ quả 2:** Đồ thị có hướng liên thông yếu, nếu tồn tại 2 đỉnh u và v : $\deg^-(u) < \deg^+(u)$ 1 đơn vị, $\deg^-(v) > \deg^+(v)$ 1 đơn vị, mọi đỉnh khác đều có $\deg^- = \deg^+$ thì có đường đi Euler từ u tới v .

10

Thuật toán tìm chu trình Euler

(Thuật toán Fleury)

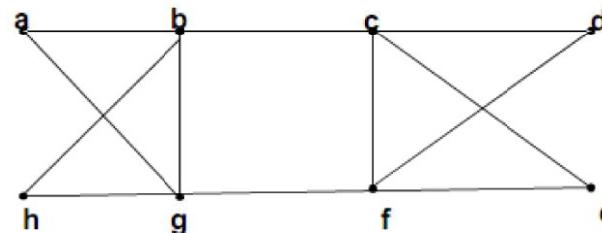
Xuất phát từ một đỉnh u nào đó của G , ta đi theo các cạnh của nó một cách tùy ý, chỉ cần tuân theo 2 qui tắc sau:

- Khi đi qua 1 cạnh thì xoá cạnh đó đi và xoá luôn đỉnh cô lập nếu có.
- Không bao giờ đi qua cầu (cạnh cắt) trừ khi không còn cách nào khác.

11

Ví dụ

- Tìm chu trình Euler cho đồ thị:



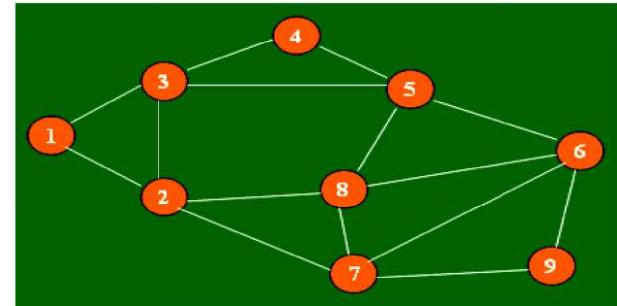
12

Cài đặt

- **B1.** Nạp 1 đỉnh u tuỳ ý của đồ thị vào stack (thường là đỉnh 1)
- **B2.** Thực hiện vòng lặp đến khi stack rỗng
 - Xét đỉnh x trên đỉnh stack:
 - Nếu là đỉnh cô lập thì lấy khỏi stack → kết quả.
 - Nếu còn đỉnh y kè với x thì nạp y vào stack và xoá cạnh xy.

Ví dụ

- Tìm chu trình Euler cho đồ thị?



13

14

Đồ thị Hamilton

- Đồ thị Hamilton
 - Giới thiệu, định nghĩa
 - Điều kiện đủ để là đồ thị Hamilton
 - Qui tắc chỉ ra chu trình Hamilton hay không Hamilton
 - Thuật toán tìm chu trình Hamilton

Giới thiệu



- **William Rowan Hamilton** là một nhà toán học, vật lý và thiên văn học người Ireland.
- Có 1 khối 12 mặt, mỗi mặt hình ngũ giác đều. Mỗi đỉnh trong 20 đỉnh của khối là tên của 1 TP. Hãy tìm đường xuất phát từ 1 TP, đi dọc theo các cạnh của khối ghé thăm mỗi TP đúng 1 lần, cuối cùng trở về TP ban đầu.

15

16

Giới thiệu

- *Tổ chức tour du lịch*: sao cho người du lịch thăm quan mỗi thăng cảnh trong thành phố đúng một lần
- *Bài toán mã đi tuần*: cho con mã đi trên bàn cờ vua sao cho nó đi qua mỗi ô đúng một lần.

1	2	3	4
5	6	7	8
9	10	11	12

H = [8, 10, 1, 7, 9, 2, 11, 5, 3, 12, 6, 4]

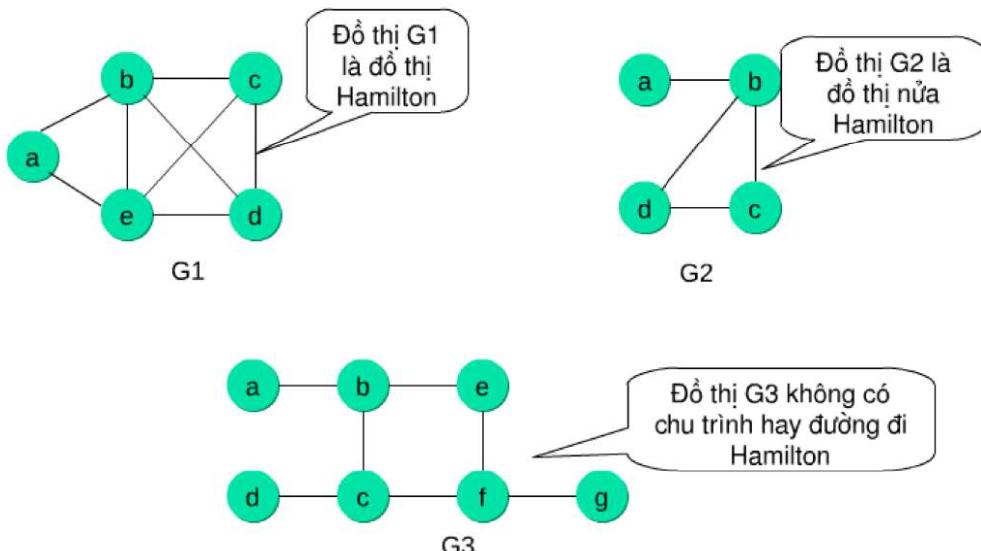
17

Các định nghĩa

1. Đường đi qua tất cả các **đỉnh** của đồ thị, mỗi đỉnh đúng một lần gọi là đường đi Hamilton.
2. Chu trình đi qua tất cả các đỉnh, mỗi đỉnh đúng 1 lần (trừ đỉnh đầu trùng đỉnh cuối) gọi là chu trình Hamilton.
3. Đồ thị có đường đi Hamilton là đồ thị **nửa Hamilton**.
4. Đồ thị có chu trình Hamilton gọi là **đồ thị Hamilton**.

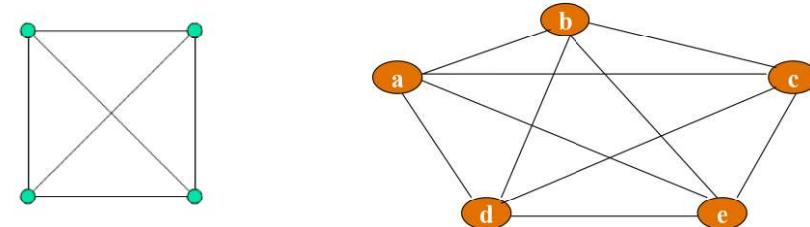
18

Ví dụ



Tính chất HAMILTON trong đồ thị đầy đủ

- Đồ thị đầy đủ luôn là đồ thị Hamilton.
- Với n lẻ và $n \geq 3$ thì K_n có $(n-1)/2$ chu trình Hamilton đôi một không có cạnh chung



20

Điều kiện đủ

1. **Định lý 1 (Dirak):** Đơn đồ thị **vô hướng** G với $n > 2$ đỉnh, mỗi đỉnh có bậc không nhỏ hơn $n/2$ là đồ thị Hamilton.
2. **Định lý 3:** Nếu G là đơn đồ thị **có hướng** liên thông mạnh n đỉnh và $\deg^+(v) \geq n/2$, $\deg^-(v) \geq n/2 \forall v$ thì G là đồ thị Hamilton.

21

Qui tắc

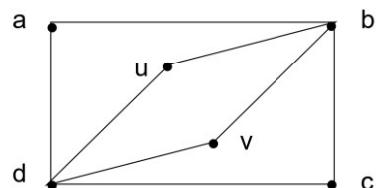
Chỉ ra chu trình Hamilton H hay chỉ ra G không là Hamilton (đồ thị vô hướng)

- **Qt1.** Mọi cạnh kề với đỉnh bậc 2 đều phải thuộc chu trình H .
- **Qt2.** Không có chu trình con nào được tạo thành trong khi xây dựng H .
- **Qt3.** Sau khi đã lấy 2 cạnh tới đỉnh x đặt vào chu trình H rồi thì xoá tất cả những cạnh còn lại mà kề với x . Khi đó có thể tạo ra những đỉnh bậc 2 mới (áp dụng Qt1).
- **Qt4.** Nếu \exists 1 đỉnh của G có bậc ≤ 1 thì G không có chu trình Hamilton.

22

Ví dụ

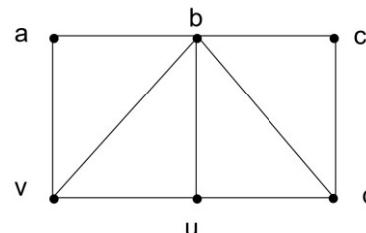
- Xét xem đồ thị sau có là đồ thị Hamilton không?



23

Ví dụ

- Xét xem đồ thị sau có là đồ thị Hamilton không?



24

Thuật toán tìm chu trình Hamilton

- Có thể sử dụng thuật toán quay lui để tìm chu trình Hamilton
 - Lưu trữ đồ thị đã cho dưới dạng danh sách $Ke(v)$
 - Liệt kê các chu trình Hamilton thu được bằng việc phát triển dãy các đỉnh ($X[1], \dots, X[k-1]$)

25

Thuật toán tìm chu trình Hamilton (tt)

- B1: Bắt đầu từ đỉnh 1, $x[1]=1$
- B2: Tìm và lưu đỉnh có cạnh nối với $x[i]$ và và đỉnh j này chưa thăm trước đó
- B3: Nếu đỉnh j này là $x[n]$ và $x[1]$ có cạnh nối với nó thì xuất ra đồ thị Hamilton
Nếu đỉnh j vẫn chưa phải là $x[n]$ thì tiếp tục B2

26

Thuật toán tìm chu trình Hamilton (tt)

```
Procedure Hamilton(k);
BEGIN
  for y ∈ Ke(X[k-1]) do
    if (k = N+1) and (y=v0) then
      Ghinhan(X[1], . . . , X[n], v0)
    else if Chuaxet[y] then
      begin
        X[k]:=y;
        Chuaxet[y]:=false;
        Hamilton(k+1);
        Chuaxet[y]:=true;
      end;
END;
```

27

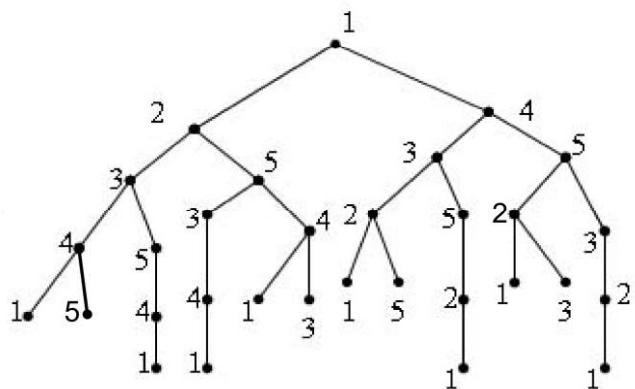
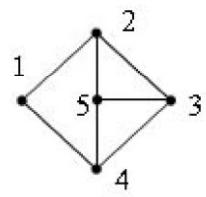
Thuật toán tìm chu trình Hamilton (tt)

```
(* Main program*)
BEGIN
  for v ∈ V do
    Chuaxet[v]:=true;
  X[1]:=v0; (* v0: 1 đỉnh bất kỳ *)
  Chuaxet[v0]:=false;
  Hamilton(2);
END
```

28

Thuật toán tìm chu trình Hamilton (tt)

Câu hỏi???



29

30

CHƯƠNG 7

Luồng cực đại trong mạng

Luồng cực đại trong mạng

- Giới thiệu
- Các định nghĩa, định lý
- Bài toán luồng cực đại
- Thuật toán Ford - Fulkerson

Giới thiệu

- Luồng cực đại là một trong những bài toán tối ưu của Lý thuyết Đồ thị, được đề xuất vào đầu những năm 1950.
- Bài toán gắn liền và trở nên nổi tiếng với tên tuổi của hai nhà toán học Mỹ là Ford và Fulkerson

1

2

Các định nghĩa, định lý

ĐN1 (Mạng): Đồ thị có hướng $G=(V,E)$ là mạng khi và chỉ khi:

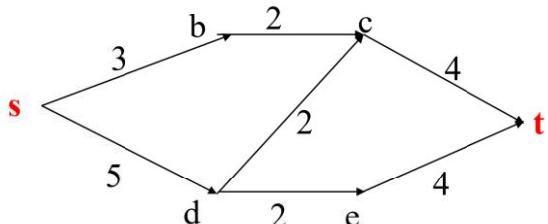
- Tồn tại duy nhất một đỉnh $s \in V$, mà tại s không có cung đi vào. Điểm s gọi là điểm phát
- Tồn tại duy nhất một đỉnh $t \in V$, mà tại t không có cung đi ra. Điểm t gọi là điểm thu
- Mỗi cung $u(i,j) \in E$ đều được gán 1 giá trị nguyên không âm $c(i,j)$ và gọi là khả năng thông qua của cung.

3

4

Các định nghĩa, định lý (tt)

- Đồ thị sau là mạng với điểm phát là đỉnh s và điểm thu là t



Ta có khả năng thông qua

$$C(s,b) = 3, C(b,c) = 2, C(s,d) = 5, C(d,c) = 2, \\ C(d,e) = 2, C(c,t) = 4, C(e,t) = 4$$

5

Các định nghĩa, định lý (tt)

- Với một mạng $G = (V, E, c)$, ta ký hiệu:
 - $W^-(x) = \{ (a, x) \in E \mid a \in V \}$ - tập các cạnh đi vào đỉnh x .
 - $W^+(x) = \{ (x, b) \in E \mid b \in V \}$ - tập các cạnh đi ra khỏi đỉnh x .

6

Các định nghĩa, định lý (tt)

- ĐN2 (Luồng trên mạng):** Hàm $f: E \rightarrow N$ là một luồng đi qua mạng (G, c) nếu:
 - $\forall e \in E : f(e) \leq c(e)$: luồng trên mỗi cạnh không được vượt quá khả năng thông qua của cạnh đó.
 - $\forall x \neq s \text{ và } t : f(W^-(x)) = f(W^+(x))$: luồng trên các đỉnh phải cân bằng
 - Giá trị của luồng f là số $Val(f) = f(W^+(s)) = f(W^-(t))$

7

Các định nghĩa, định lý (tt)

- ĐN3 (lát cắt):** Ta gọi lát cắt (X, X^*) là một cách phân hoạch tập đỉnh V của mạng ra thành hai tập X và $X^* = V \setminus X$, trong đó $s \in X, t \in X^*$. Khả năng thông qua của lát cắt (X, X^*) là số:

$$c(X, X^*) = \sum_{\substack{v \in X \\ w \in X^*}} c(v, w)$$

- Lát cắt với khả năng thông qua nhỏ nhất được gọi là **lát cắt hẹp nhất**.

8

Các định nghĩa, định lý (tt)

- **Bổ đề 1:** Giá trị của luồng f trong mạng luôn \leq khả năng thông qua của lát cắt (X, X^*) bất kỳ trong nó.
- **Hệ quả 1:** Giá trị luồng cực đại trong mạng không vượt quá khả năng thông qua của lát cắt hẹp nhất trong mạng

9

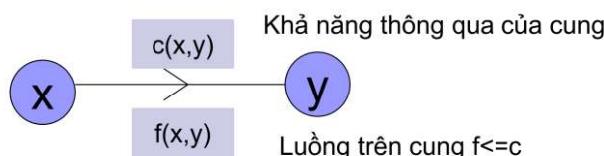
Bài toán luồng cực đại

- **Phát biếu:** Cho mạng G với nguồn s , đích t và khả năng thông qua $C(i,j)$. Trong số các luồng trên mạng G , tìm luồng có giá trị lớn nhất.
- **Ý tưởng:** xuất phát từ luồng nào đó, ta tìm đường đi từ s đến t , cho phép hiệu chỉnh giá trị luồng f trên đường đi đó, sao cho luồng mới có giá trị lớn hơn. Nếu không tìm được đường đi như vậy thì ta có luồng cực đại.
- Đồ thị G_f đó được gọi là đồ thị tăng luồng.

10

Định lý

- Gọi f là luồng trên mạng $G=(V,E)$. Các mệnh đề sau đây là tương đương:
 - f là luồng cực đại trong mạng
 - Không tìm được đường tăng luồng f
 - \exists một lát cắt (X, X^*) mà $\text{val}(f) = c(X, X^*)$



11

Thuật toán Ford - Fulkerson

1. Gán nhãn cho các đỉnh

- Mỗi đỉnh sẽ có 1 trong 3 trạng thái
 - Đỉnh chưa có nhãn
 - Đỉnh có nhãn nhưng chưa xét
 - Đỉnh có nhãn đã xét xong
- Nhãn của đỉnh y sẽ có dạng $y: [\pm x, \sigma(y)]$
 - $+x$: cần tăng luồng theo cung (x,y)
 - $-x$: cần giảm luồng theo cung (y,x)
 - $\sigma(y)$: là lượng dùng để tăng/giảm
- Đầu tiên, các đỉnh đều chưa có nhãn

12

Thuật toán Ford – Fulkerson (tt)

■ B1:

- Gán nhãn cho đỉnh phát s : $[+s, \infty]$
- Đỉnh s có nhãn nhưng chưa xét
- Tất cả các đỉnh khác chưa có nhãn

13

Thuật toán Ford – Fulkerson (tt)

■ B2:

- Xét 1 đỉnh có nhãn nhưng chưa xét, giả sử đó là x : $[\pm y, \sigma(y)]$
- Với mỗi đỉnh u chưa có nhãn, là ảnh của x (đỉnh cuối) và $f(x,u) < c(x,u)$, được gán nhãn
 - u : $[+x, \sigma(u)]$ với $\sigma(u) = \min(\sigma(x), c(x,y)-f(x,y))$
- Với mỗi đỉnh v chưa có nhãn, là tạo ảnh của x (đỉnh đầu) và $f(v,x) > 0$, được gán nhãn
 - v : $[-x, \sigma(v)]$ với $\sigma(v) = \min(\sigma(x), f(v,x))$
- x có nhãn đã xét; u, v có nhãn nhưng chưa xét

14

Thuật toán Ford – Fulkerson (tt)

■ B3:

- Lặp lại B2 cho đến khi:
 - Hoặc là đỉnh thu T được gán nhãn \rightarrow B4
 - Hoặc là đỉnh T không được gán nhãn và cũng không thể gán nhãn. Trường hợp này giải thuật kết thúc với luồng cực đại.

Gọi X_0 là tập hợp các đỉnh có nhãn

Gọi Y_0 là tập hợp các đỉnh không có nhãn

Thì (X_0, Y_0) là lát cắt hẹp nhất

15

Thuật toán Ford – Fulkerson (tt)

2. Tăng luồng

■ B4: Đặt $x = t$

■ B5

- Nếu nhãn x : $[+y, \sigma(x)]$ thì tăng luồng từ $y \rightarrow x$ là $\sigma(t)$
- Nếu nhãn x : $[-y, \sigma(x)]$ thì giảm luồng từ $x \rightarrow y$ là $\sigma(t)$

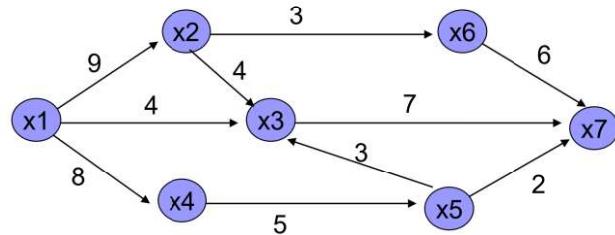
■ B6

- Nếu $x = s$ (điểm phát) thì xoá tất cả các nhãn, quay lại B1 với luồng đã được điều chỉnh ở B5
- Nếu $x \neq s$ thì đặt $x = y$ và quay lại B5

16

Ví dụ:

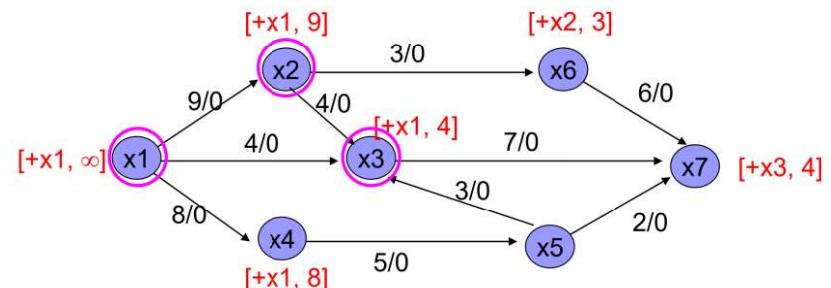
- Tìm luồng cực đại trong mạng sau đây



17

Giải

- Lần lặp thứ 1 (luồng =0)

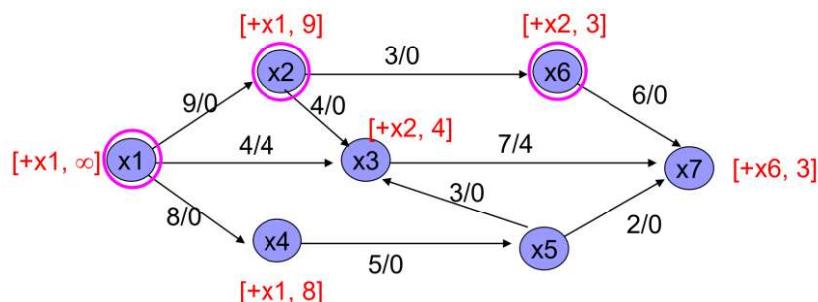


- Chọn x2
- Chọn x3
- Đặt x=x7: [+x3, 4] → tăng luồng (x3, x7) lên 4
- Đặt x=x3: [+x1, 4] → tăng luồng (x1, x3) lên 4

18

Giải

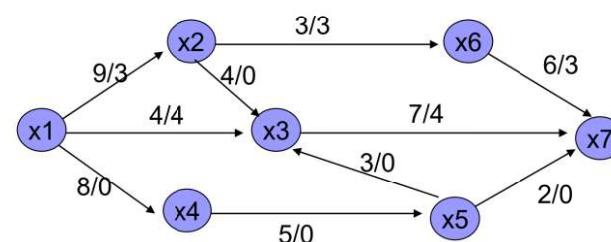
- Lần lặp thứ 2 (với luồng đã điều chỉnh)



19

Giải

- Lần lặp thứ 3 (với luồng đã điều chỉnh):



20

- Đặt x=x7: [+x6, 3] → tăng luồng (x6, x7) lên 3
- Đặt x=x6: [+x2, 3] → tăng luồng (x2, x6) lên 3
- Đặt x=x2: [+x1, 9] → tăng luồng (x1, x2) lên 3

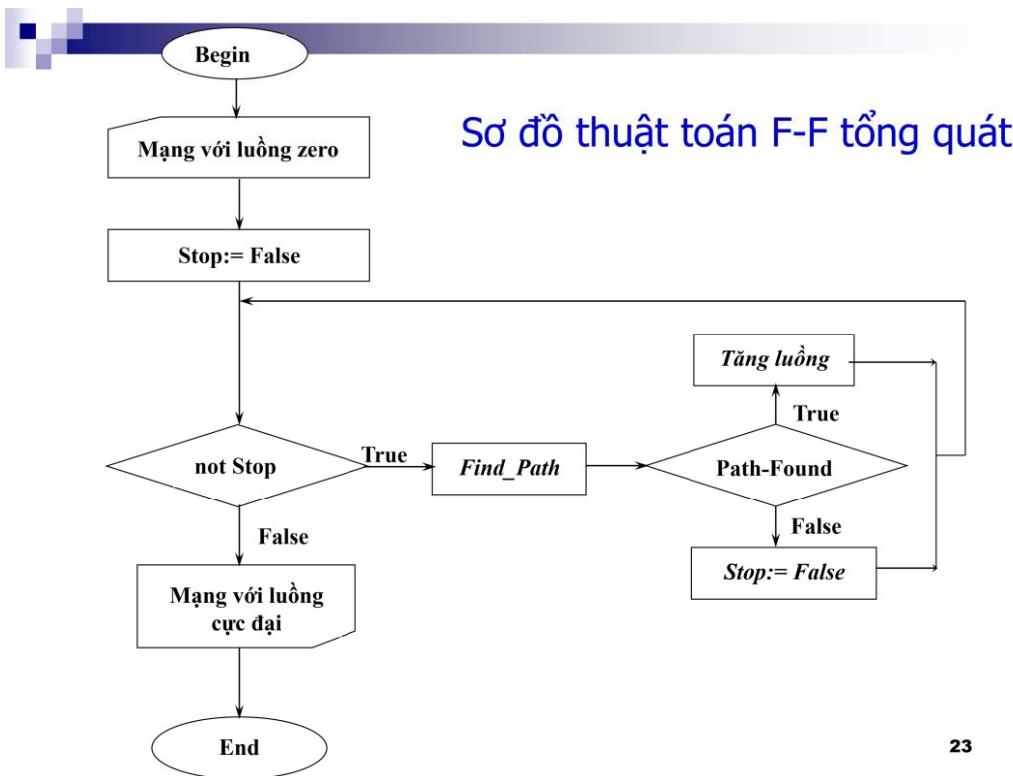
Giải

- Lần lặp thứ 4 (với luồng đã điều chỉnh):

Giải

- Lần lặp thứ 5 (với luồng đã điều chỉnh):

21



Sơ đồ thuật toán F-F tổng quát

22

Câu hỏi???

23

24