TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI
KHOA CÔNG NGHỆ THÔNG TIN

TOÁN RỜI RẠC

Chương 4: Lý thuyết đồ thị

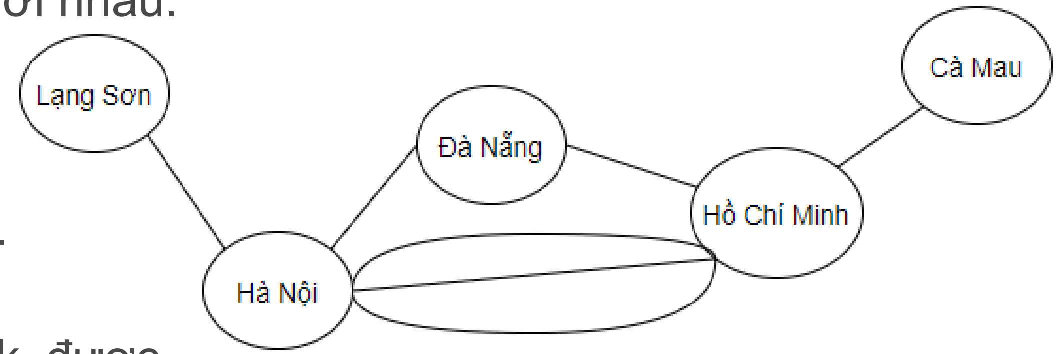
GV: KS. Bùi Minh Thảo
BM: KHMT

Nội dung

1. Khái niệm đồ thị
2. Các loại đồ thị cơ bản
3. Các thuật ngữ về đồ thị
4. Một số đồ thị đặc biệt và ứng dụng
5. Biểu diễn đồ thị và đồ thị đẳng cấu
6. Tính liên thông trong đồ thị
7. Chu trình Euler và Hamilton
8. Bài toán tìm đường đi ngắn nhất
9. Cây và các khái niệm cơ bản
10. Cây khung và các thuật toán

1. Khái niệm đồ thị

- ❖ **Định nghĩa đồ thị**: cấu trúc rời rạc gồm
  - ➢ Các đỉnh
  - ➢ Các cạnh nối các đỉnh với nhau.
- ❖ **Biểu diễn**:
  - ➢ Đỉnh: các điểm
  - ➢ Cạnh: đường thẳng/cong.
- ❖ **VD**: Giả sử rằng 1 network được tạo bởi các trung tâm dữ liệu và các đường kết nối giữa các máy tính.



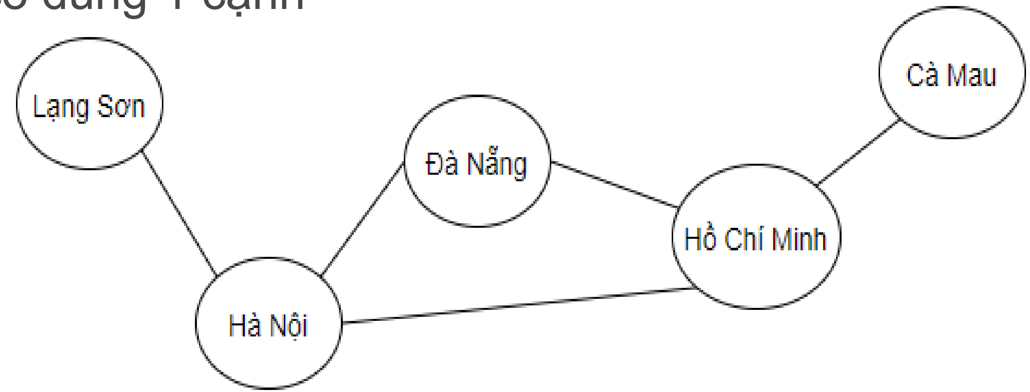
2. Các loại đồ thị cơ bản

❖ **Đơn đồ thị**: G = (V,E)

- V: tập hợp không rỗng của các đỉnh.
- E: tập hợp, mỗi phần tử là một cặp đỉnh không sắp thứ tự (u,v) của hai đỉnh thuộc V.

☞ Các cạnh nối các đỉnh lại với nhau.

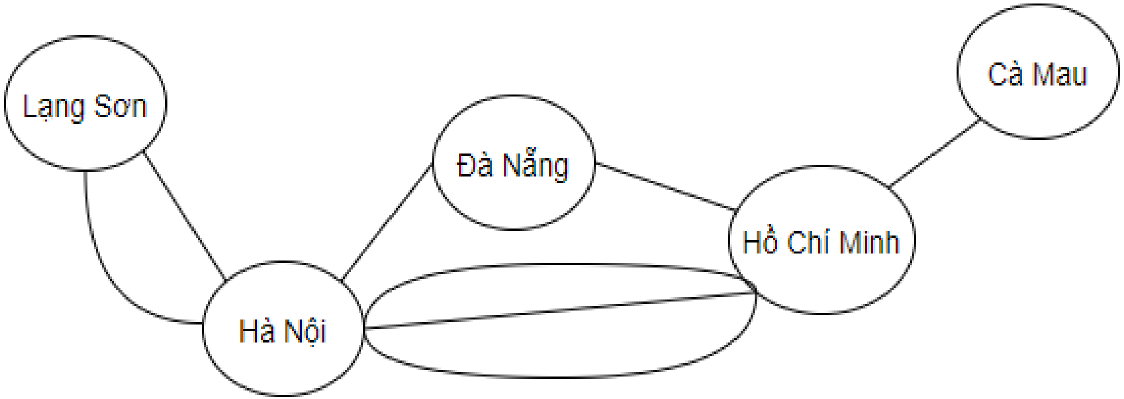
☞ Giữa 2 đỉnh chỉ có đúng 1 cạnh



2. Các loại đồ thị cơ bản

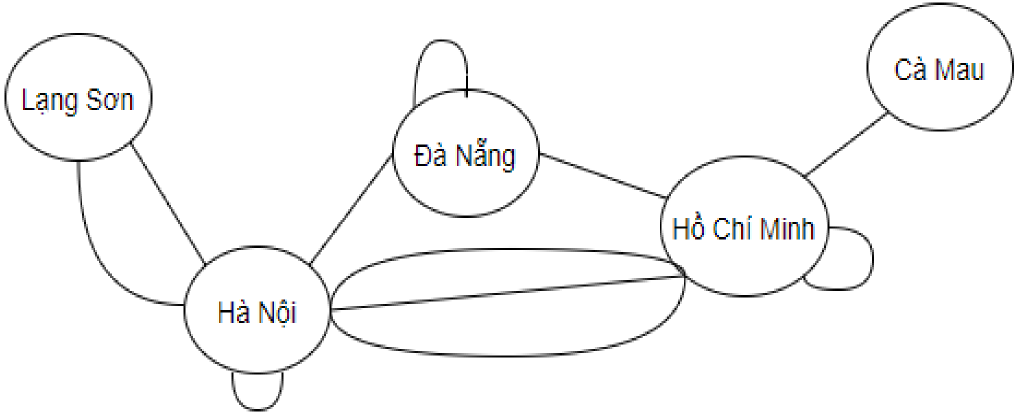
❖ **Đa đồ thị**: G = (V,E)

☞ E: cho phép nhiều cạnh nối 1 cặp đỉnh



2. Các loại đồ thị cơ bản

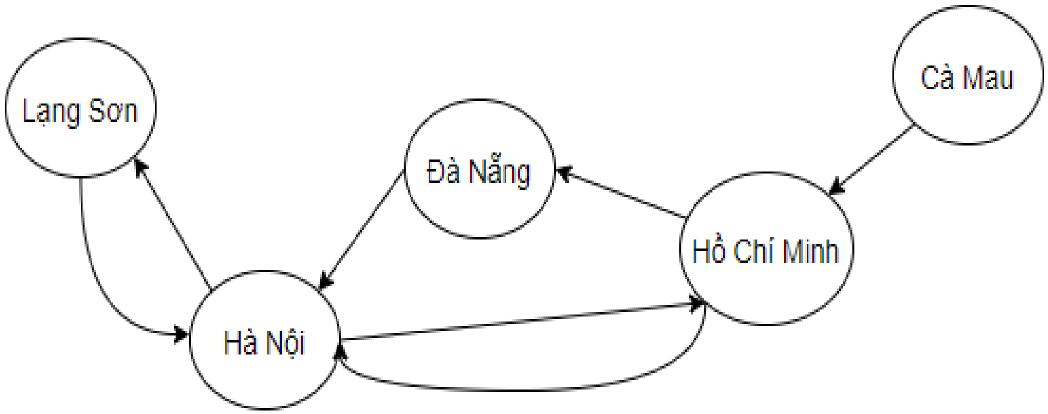
- ❖ **Khuyên** là cạnh có 2 đỉnh trùng nhau.
- ❖ **Giả đồ thị**: G = (V,E)
  - ☞ E: cho phép lặp tại các đỉnh.
    (Còn gọi là chứa các khuyên)



2. Các loại đồ thị cơ bản

❖ **Đồ thị có hướng**: G(V,E)

- V: 1 tập hợp không rỗng của các đỉnh.
- E: tập các cặp đỉnh có thứ tự.



2. Các loại đồ thị cơ bản

- **Đơn đồ thị có hướng**: một đồ thị có hướng không có khuyên và không có đa cạnh. Bởi một đơn đồ thị có hướng có nhiều nhất một cạnh gắn với một cặp thứ tự đỉnh (u,v), chúng ta gọi (u,v) là một cạnh nếu có một cạnh gắn với nó trong đồ thị.
- **Đa đồ thị có hướng**: một đồ thị có hướng có đa cạnh có hướng từ một đỉnh đến đỉnh thứ hai. Khi có m cạnh có hướng, mỗi cạnh gắn với một cặp thứ tự các đỉnh (u,v), chúng ta nói rằng (u,v) là một cạnh có bội m.

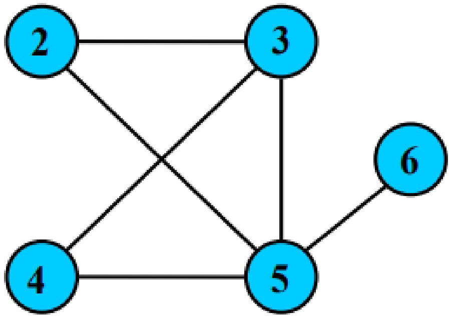
2. Các loại đồ thị cơ bản

❖ **Bài tập**

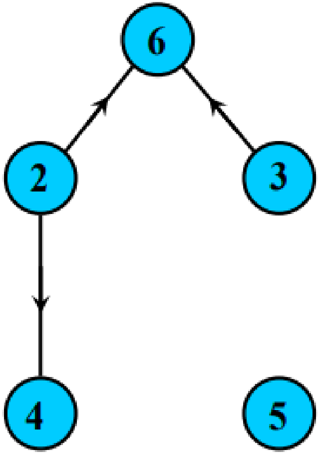
1. Cho tập V = {2, 3, 4, 5, 6}. Hãy biểu diễn quan hệ nguyên tố cùng nhau của tập trên.
2. Cho tập V = {2, 3, 4, 5, 6}. Hãy biểu diễn quan hệ aRb ⇔ a là ước của b và a ≠ b.

❖ **Đáp án**

1.



2.



3. Các thuật ngữ về đồ thị

❖ *Định nghĩa 1:* Cho G = (V,E) là đồ thị vô hướng và e = (u,v) $\in$ E.

- u và v gọi là 2 **đỉnh liền kề**.
- e gọi là **cạnh kề** của u và v.

❖ *Định nghĩa 2:*

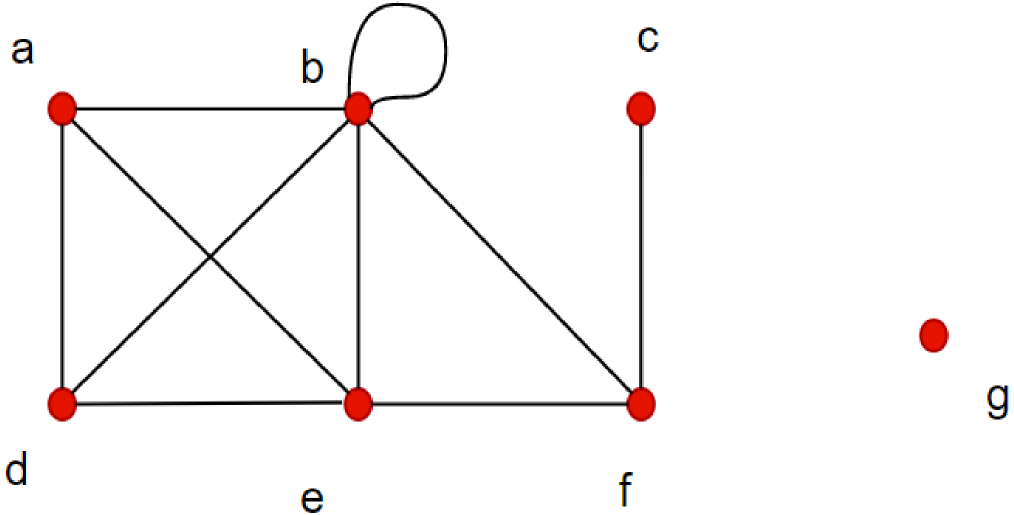
Tập tất cả các đỉnh kề của 1 đỉnh v của G = (V,E) kí hiệu là N(v), được gọi là hàng xóm của v.

❖ *Định nghĩa 3:*

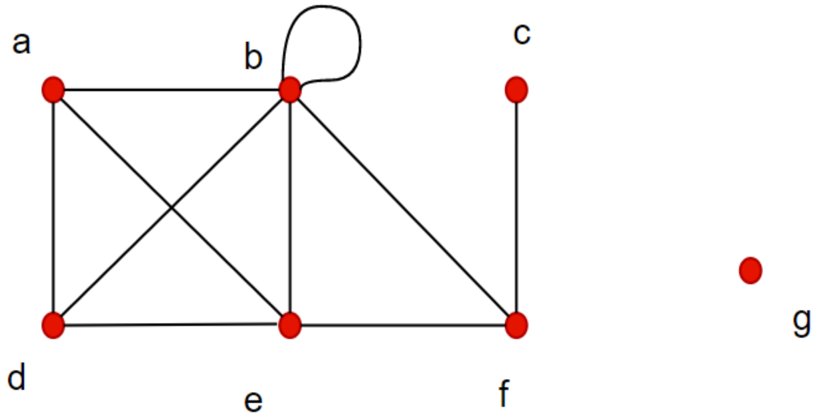
- Bậc của 1 đỉnh trong 1 đồ thị vô hướng là số cạnh kề với nó.
- Bậc của đỉnh v được kí hiệu là **deg(v)**.
- Một đỉnh có bậc 0 được gọi là đỉnh cô lập.
- Một đỉnh là lá nếu bậc của nó là 1.

3. Các thuật ngữ về đồ thị

- **VD**: Tìm bậc và hàng xóm của các đỉnh trong đồ thị sau:



3. Các thuật ngữ về đồ thị



Lời giải:

Trong đồ thị trên,

Bậc của các đỉnh:

deg(a) = deg(d) = deg(f) =3, deg(b) = 6, deg(c) =1, deg(e) =4, deg(g) =0.

Hàng xóm của các đỉnh:

N(a) = {b,d,e}, N(b) = {a,b,d,e,f}, N(c) = {f},

N(d) = {a,b,e}, N(e) = {a,b,d,f}, N(f) = {b,c,e}, N(g) = ∅.

3. Các thuật ngữ về đồ thị

❖ *Định lý 1: Định lý bắt tay*

Cho G = (V,E) là một đồ thị vô hướng với m cạnh thì:

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

❖ *Định lý 2:*

Một đồ thị vô hướng có một số chẵn đỉnh có bậc lẻ.

- **VD**: Trong đồ thị có 10 đỉnh, mỗi đỉnh có bậc bằng 6. Hỏi đồ thị có bao nhiêu cạnh.

Số cạnh m = (10 x 6) : 2 = 30 (cạnh)

3. Các thuật ngữ về đồ thị

- *Định nghĩa 4:*

  Cho G = (V,E) là đồ thị có hướng và e = (u,v) $\in$ E.

  - u gọi là kề tới v, v gọi là kề từ u.
  - u gọi là đỉnh đầu, v gọi là đỉnh cuối.

  - $deg^-(v)$: bậc “vào” của v.
  - $deg^+(v)$: bậc “ra” của v.

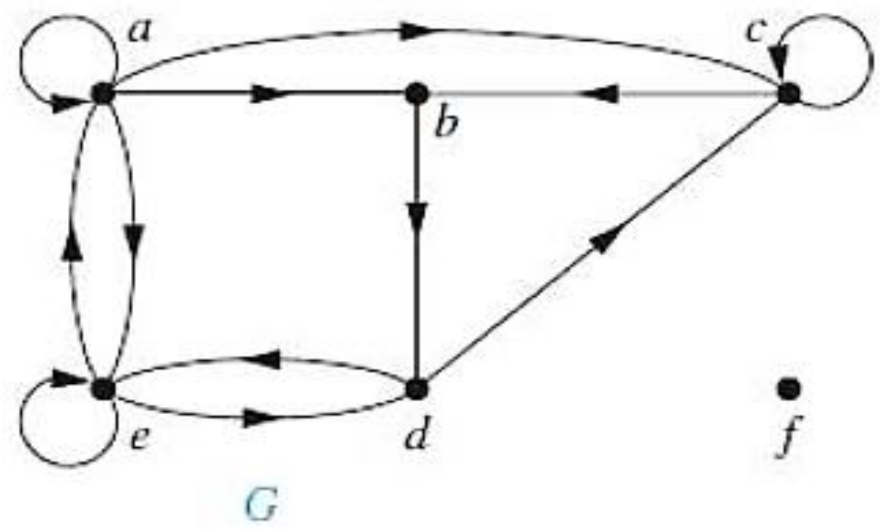
- *Định lý 3:*

  Gọi G = (V,E) là một đồ thị có hướng thì:

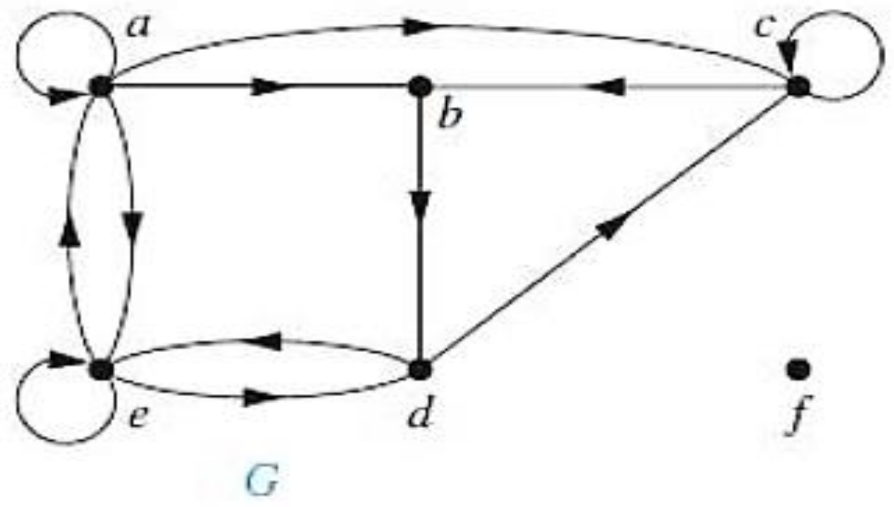
$$\sum_{v \in V} deg^-(v) = \sum_{v \in V} deg^+(v) = |E|.$$

3. Các thuật ngữ về đồ thị

VD: Tìm bậc vào và bậc ra của mỗi đỉnh trong đồ thị G dưới đây:



3. Các thuật ngữ về đồ thị



Các bậc vào trong đồ thị là:

deg-(a)=2, deg-(b)=2, deg-(c)=3, deg-(d)=2, deg-(e)=3, deg-(f)=0

Các bậc ra của đồ thị là:

deg+(a)=4, deg+(b)=1, deg+(c)=2, deg+(d)=2, deg+(e)=3, deg+(f)=0

4. Một số đồ thị đặc biệt và ứng dụng

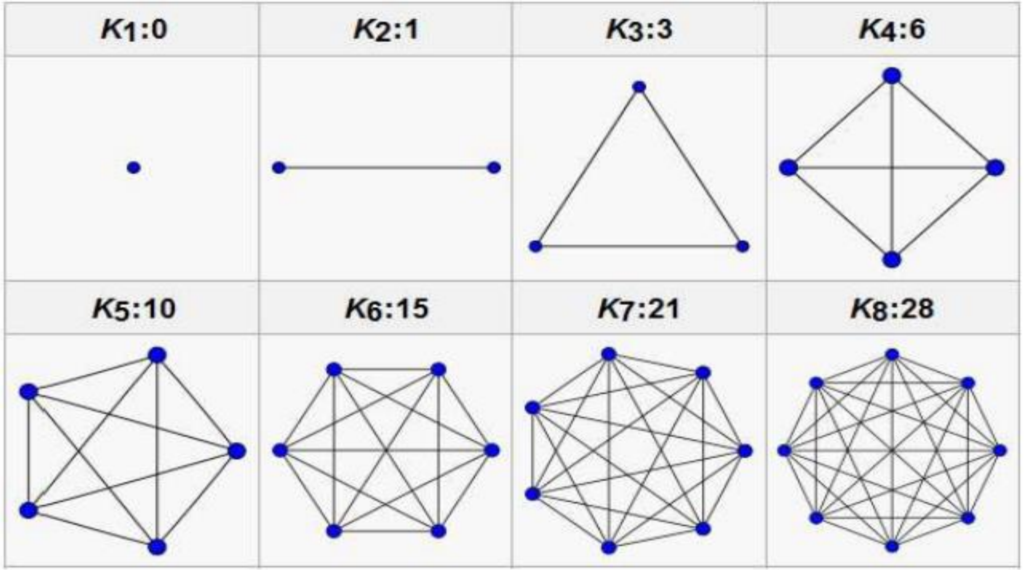
❖ **Đồ thị đầy đủ**: *n* đỉnh

Mỗi cặp đỉnh đều có đúng 1 cạnh nối. Kí hiệu: $\boldsymbol{K_n}$

Đồ thị K_n có:

- n đỉnh
- Deg(u) = n-1
- số cạnh là $\frac{n(n-1)}{2}$

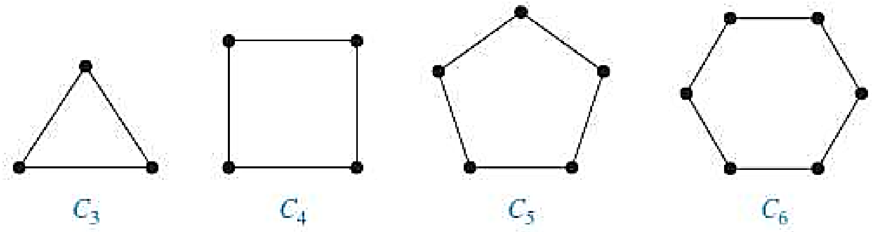
➢ Trong thực tế, mô hình đồ thị đầy đủ dùng để biểu diễn các cặp đấu trong 1 giải đấu mà các đội thi đấu vòng tròn 1 lượt.



4. Một số đồ thị đặc biệt và ứng dụng

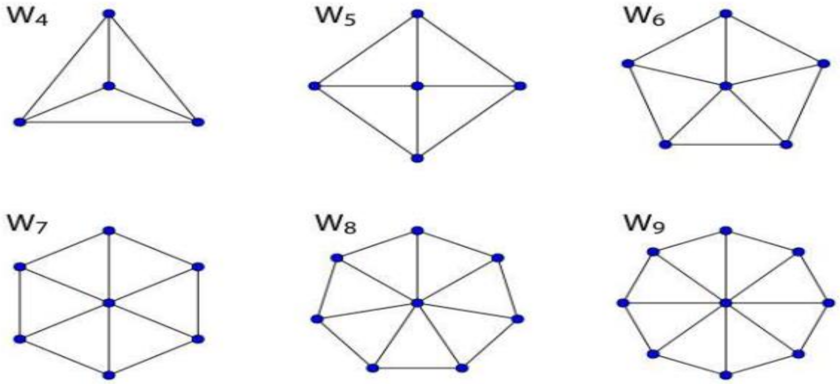
❖ **Đồ thị chu trình**: *n* ≥ *3* đỉnh

Kí hiệu: $\boldsymbol{C_n}$



❖ **Đồ thị bánh xe**: *n* ≥ *3* đỉnh và 1 đỉnh ở giữa nối với các đỉnh kia.

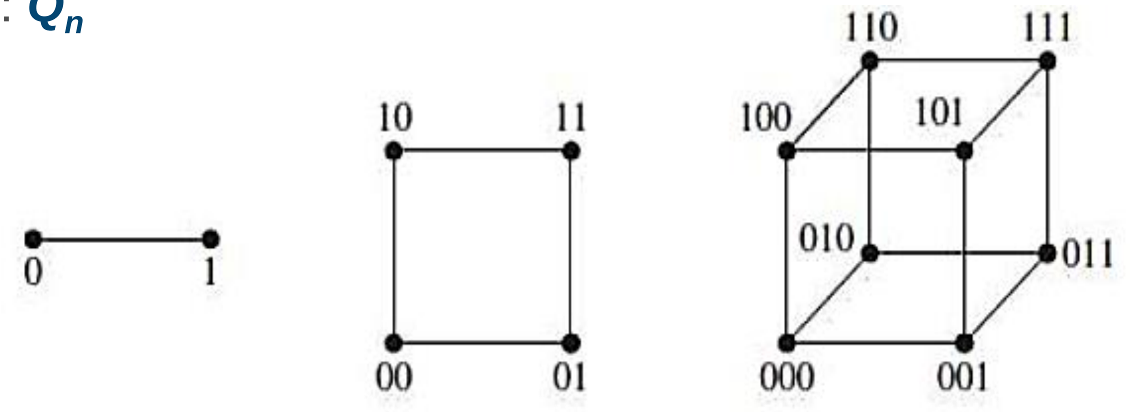
Kí hiệu: $\boldsymbol{W_n}$



4. Một số đồ thị đặc biệt và ứng dụng

❖ **Siêu khối n chiều**: đồ thị có các đỉnh biểu diễn 2^n chuỗi bit có độ dài n.

Kí hiệu: Q_n



Q_1 Q_2 Q_3

4. Một số đồ thị đặc biệt và ứng dụng

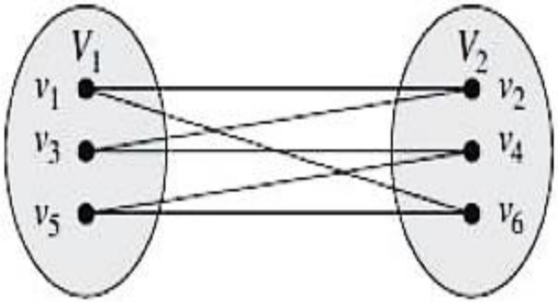
- **Đồ thị phân đôi**:
  - Các đỉnh của 1 đồ thị chia làm 2 tập con.
  - Mỗi cạnh nối 1 đỉnh từ tập này đến 1 đỉnh ở tập kia.

Ví dụ:
- Quan hệ hôn nhân trong một làng gồm 2 tập con là phái nam và phái nữ.
- Quan hệ "gán" giữa danh sách các công việc và danh sách các nhân viên.

Định nghĩa đồ thị phân đôi: G = (V,E)

- $V = V_1 \cup V_2,\ V_1, V_2 \neq \varnothing$ và $V_1 \cap V_2 = \varnothing$
- $(u,v) \in E,\ u \in V_1,\ v \in V_2$

Ví dụ: C_6 là 1 phân đôi

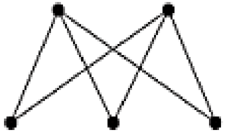


4. Một số đồ thị đặc biệt và ứng dụng

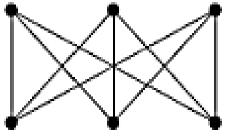
❖ **Đồ thị phân đôi đầy đủ**:

G = (V,E) là phân đôi đầy đủ nếu:

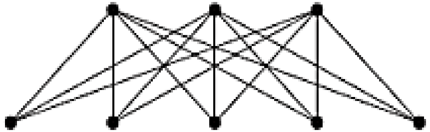
- ➢ G là đồ thị phân đôi.
- ➢ $\forall$ u $\in$ $V_{1,}$ v $\in$ $V_{2,}$ (u,v) $\in$ E.
- ➢ Kí hiệu: $\boldsymbol{K_{m,n}}$ với $|V_1| = m$, $|V_2| = n$.



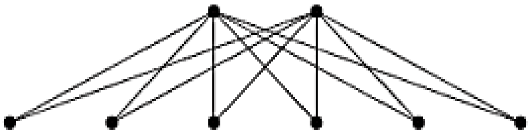
$K_{2,3}$



$K_{3,3}$



$K_{3,5}$

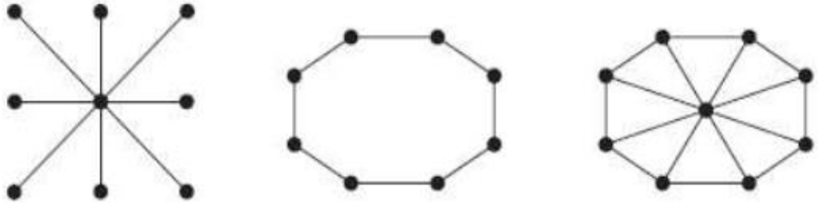


$K_{2,6}$

4. Một số đồ thị đặc biệt và ứng dụng

❖ **Ứng dụng:**

➢ Mạng cục bộ (LAN): hình sao, hình vòng, hình lai (có dư thừa nhưng tăng độ tin cậy)



➢ Các mạng kết nối cho tính toán song song: 1 chiều, lưới, siêu khối.

FIGURE 12 A Linear Array for Six Processors.

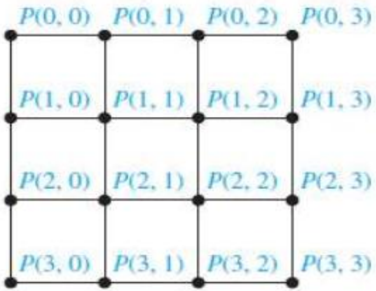


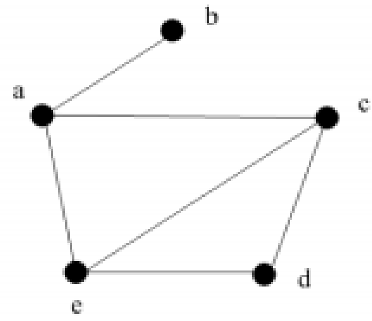
FIGURE 13 A Mesh Network for 16 Processors.

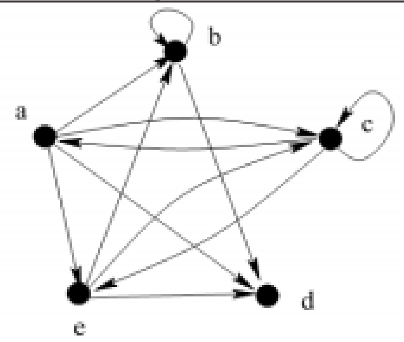
5. Biểu diễn đồ thị và đồ thị đẳng cấu

❖ **Danh sách kề:**

Vertex	Adjacency vertices
a	b, c, e
b	a
c	a, d, e
d	c, e
e	a, c, d

Initial vertex	Adjacency vertices
a	b, c, d, e
b	b, d
c	a, c, e
d	c, e
e	b, c, d





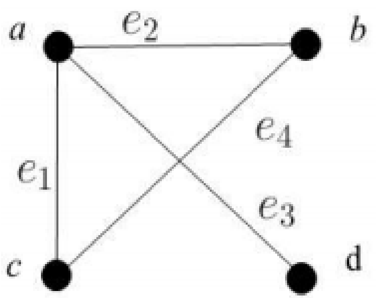
5. Biểu diễn đồ thị và đồ thị đẳng cấu

❖ **Ma trận kề:**

Cho đồ thị G = (V,E). Ma trận kề A_G

- ➢ Kích thước: |V| * |V|
- ➢ Giá trị các phần tử: $a_{ij} = \begin{cases} 1 \text{ nếu } (v_i, v_j) \in E, \\ 0, \text{ngược lại.} \end{cases}$

Ví dụ:



$$\begin{array}{c} \\ a \\ b \\ c \\ d \end{array} \begin{array}{c} \begin{array}{cccc} a & b & c & d \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

5. Biểu diễn đồ thị và đồ thị đẳng cấu

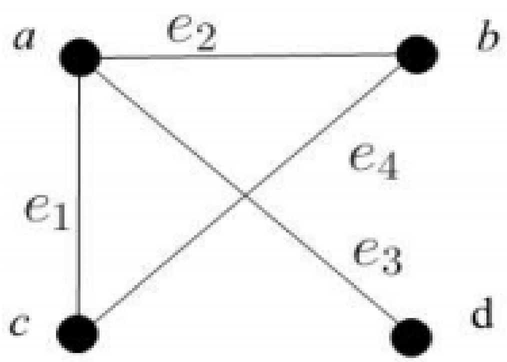
Nhận xét:

- Một ma trận kề của 1 đồ thị dựa trên thứ tự lựa chọn các đỉnh. Do đó, có thể có n! ma trận kề khác nhau cho 1 đồ thị có n đỉnh.
- Ma trận kề của 1 đơn đồ thị là đối xứng, nghĩa là $a_{ij} = a_{ji}$.
- Bởi 1 đơn đồ thị thì không có khuyên, mỗi vị trí $a_{ii} = 0$, i=1,2,3,..,n.
- Các ma trận kề cũng được sử dụng để biểu diễn các đồ thị vô hướng có khuyên và đa cạnh.
- Một khuyên tại đỉnh v_i được biểu diễn bởi giá trị 1 tại vị trí (i,j) của ma trận kề.
- Khi đa cạnh nối cùng cặp đỉnh v_i và v_j hoặc đa cạnh tại cùng đỉnh thì ma trận kề không còn là ma trận 0-1, bởi vị trí (i,j) của ma trận này bằng số cạnh nối cặp đỉnh { v_i , v_j}.
- Tất cả đồ thị vô hướng, bao gồm cả đa đồ thị và giả đồ thị đều có ma trận kề đối xứng.

5. Biểu diễn đồ thị và đồ thị đẳng cấu

❖ **Ma trận kề:**

Bài tập: Cho đồ thị đơn vô hướng, dữ liệu lấy từ input. Dòng đầu nhập n,m là số đỉnh, số cạnh của đồ thị, m dòng tiếp theo mỗi dòng gồm u,v thể hiện cạnh nối giữa 2 đỉnh. Output là ma trận kề của đồ thị trên.




```
Nhap so dinh, so canh cua do thi: 4 4

Nhap cac canh cua do thi:
1 2
1 3
1 4
2 3

Ma tran ke cua do thi:
0 1 1 1
1 0 1 0
1 1 0 0
1 0 0 0
```

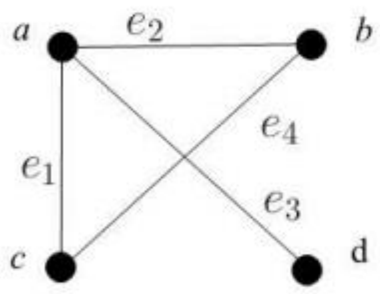
5. Biểu diễn đồ thị và đồ thị đẳng cấu

❖ **Ma trận liên thuộc:**

Cho đồ thị G = (V,E). Ma trận liên thuộc M_G

- Kích thước: |V| * |E|
- Giá trị các phần tử: $m_{ij} = \begin{cases} 1 \text{ khi cạnh } e_j \text{ kề với } v_i, \\ 0, \text{ngược lại.} \end{cases}$

Ví dụ:

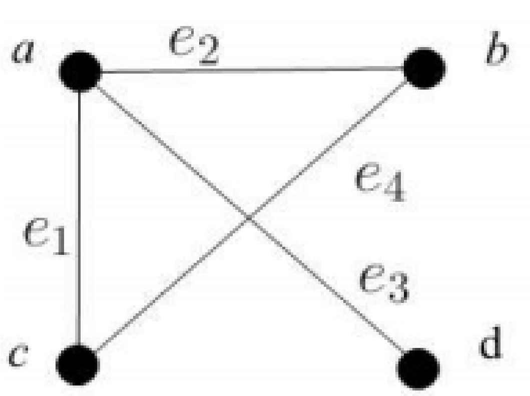


$$\begin{array}{c c} & \begin{array}{cccc} e_1 & e_2 & e_3 & e_4 \end{array} \\ \begin{array}{c} a \\ b \\ c \\ d \end{array} & \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \end{array}$$

5. Biểu diễn đồ thị và đồ thị đẳng cấu

❖ **Ma trận liên thuộc:**

Bài tập: Cho đồ thị đơn vô hướng, dữ liệu lấy từ input. Dòng đầu nhập n,m là số đỉnh, số cạnh của đồ thị, m dòng tiếp theo mỗi dòng gồm u,v thể hiện cạnh nối giữa 2 đỉnh. Output là ma trận liên thuộc của đồ thị trên.



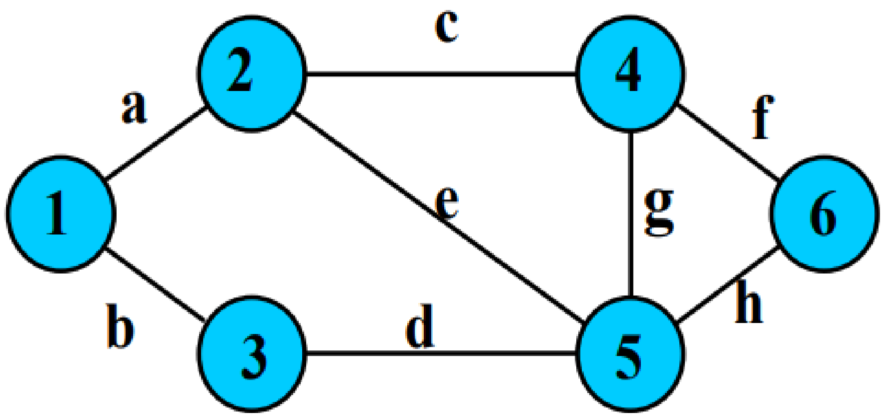

```
Nhap so dinh, so canh cua do thi: 4 4

Nhap cac canh cua do thi:
1 3
1 2
1 4
2 3

Ma tran lien thuoc cua do thi:
1 1 1 0
0 1 0 1
1 0 0 1
0 0 1 0
```

5. Biểu diễn đồ thị và đồ thị đẳng cấu

Bài tập: Biểu diễn đồ thị sau bằng ma trận kề và ma trận liên thuộc



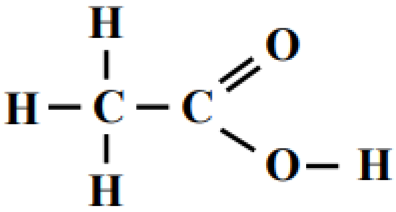
5. Biểu diễn đồ thị và đồ thị đẳng cấu

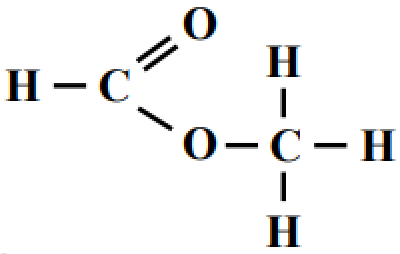
❖ **Đồ thị đẳng cấu:**

☞ Có thể vẽ được 2 đồ thị theo cùng 1 cách?

☞ Ví dụ:

- Trong hóa học, có nhiều chất có cùng công thức phân tử nhưng cấu trúc khác nhau.
- Xét công thức phân tử C2H4O2:





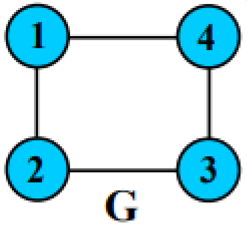
⇨ Các đồ thị có cùng cấu trúc được gọi là các đồ thị đẳng cấu biểu diễn mô hình của cùng một chất.

5. Biểu diễn đồ thị và đồ thị đẳng cấu

❖ **Đồ thị đẳng cấu:**

- Hai đồ thị được gọi là đẳng cấu nếu có một song ánh giữa tập đỉnh của hai đồ thị đảm bảo quan hệ liền kề.
- Cho 2 đồ thị đơn *G1 = (V1 , E1)* và *G2 = (V2 , E2)*. *G1* và *G2* là đẳng cấu nếu tồn tại song ánh f sao cho:
  - ➢ Hai đỉnh *a* và *b* là kề nhau trong *G1*.
  - ➢ Hai đỉnh *f(a)* và *f(b)* là kề nhau trong *G2*.

☞ *f* được gọi là một phép đẳng cấu.



Ví dụ: G và H là đẳng cấu

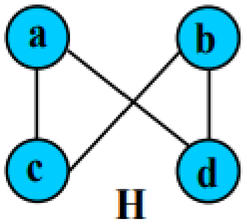
Ta dễ dàng chỉ ra song ánh f như sau:

f: {1, 2, 3, 4} → {a, b, c, d}

f(1) = a f(2) = c f(3) = b f(4) = d

(1, 2) – (a, c) (1, 4) – (a, d)

(2, 3) – (c, b) (3, 4) – (b, d)



5. Biểu diễn đồ thị và đồ thị đẳng cấu

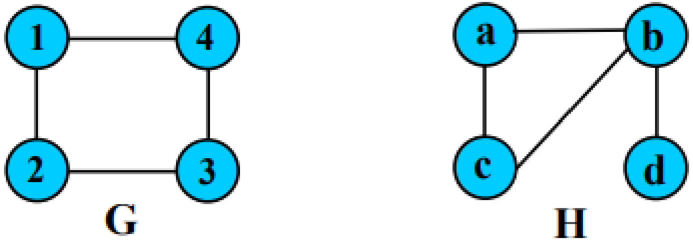
❖ **Đồ thị đẳng cấu:**

- ❑ Việc xác định 2 đồ thị đồng cấu:
  - ➢ Rất khó khăn.
  - ➢ Có *n!* phép tương đương *một – một* 2 tập đỉnh của 2 đồ thị có *n* đỉnh.
- ❑ Thông thường: chứng minh 2 đồ thị không đẳng cấu
  - ☞ Chỉ ra chúng không có 1 tính chất chung.
- ❑ Các tính chất chung của 2 đơn đồ thị đẳng cấu:
  - ➢ Cùng số đỉnh.
  - ➢ Cùng số cạnh.
  - ➢ Bậc của các đỉnh tương ứng của các đơn đồ thị đẳng cấu phải giống nhau.

5. Biểu diễn đồ thị và đồ thị đẳng cấu

❖ **Đồ thị đẳng cấu:**

Xác định 2 đồ thị sau có đẳng cấu không?



- Số đỉnh: cùng là 4
- Số cạnh: cùng là 4
- Bậc của đỉnh:
  - G có 4 đỉnh bậc 2
  - H có 2 đỉnh bậc 2, 1 đỉnh bậc 3, 1 đỉnh bậc 1.

⇨ H và G không đẳng cấu.

6. Tính liên thông trong đồ thị

❖ **Đường đi:**

Cho $G = (V,E)$ là đồ thị vô hướng hoặc có hướng.
Đường đi độ dài *n* (nguyên dương) từ *u* tới *v* là một dãy các cạnh $\{x_0, x_1\}, \{x_1, x_2\}, \ldots, \{x_{n-1}, x_n\}$ sao cho $x_0 = u$ và $x_n = v$.

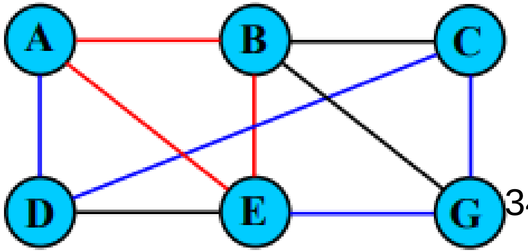
❖ **Đường đi trên đồ thị đơn:**

- Có thể kí hiệu bằng dãy các đỉnh $x_0, x_1, \ldots, x_n$.
- Đường đi đơn: không chứa 1 cạnh quá 1 lần.

❖ **Đường đi là chu trình:**

- Bắt đầu tại *u*, kết thúc tại *u* (quay trở lại).
- Chu trình đơn: không chứa 1 cạnh quá 1 lần.

❖ **Ví dụ:** A,D,C,G,E: đường đi độ dài 4.
D,E,C,A: không là đường đi.
A,B,E,A: là chu trình có độ dài 3.

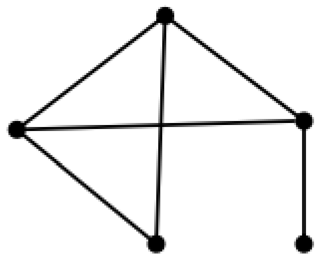


6. Tính liên thông trong đồ thị

6.1. Tính liên thông trong đồ thị vô hướng

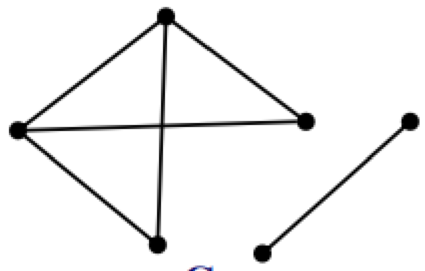
- ❖ **Định nghĩa:** Một đồ thị vô hướng là liên thông nếu tồn tại đường đi giữa mọi cặp đỉnh của đồ thị.
- ❖ ***Định lý***: Có một đường đi đơn giữa mọi cặp đỉnh phân biệt trong một đồ thị vô hướng liên thông.

Ví dụ:



G_1

G1 liên thông



G_2

G2 không liên thông

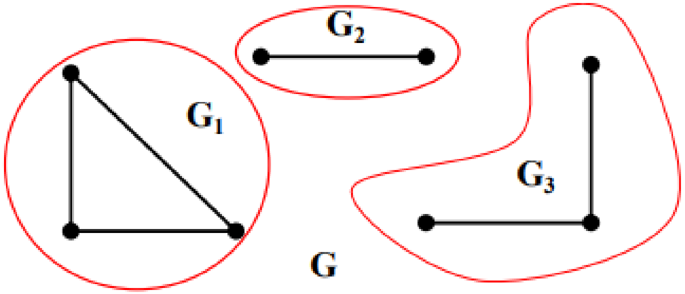
6. Tính liên thông trong đồ thị

6.1. Tính liên thông trong đồ thị vô hướng

❖ Một tập đỉnh con cùng với các cạnh nối các đỉnh của chúng tạo thành một đồ thị thành phần. Và được gọi là **thành phần liên thông** của đồ thị đã cho.

⇨ Một đồ thị không liên thông được chia thành các đồ thị thành phần liên thông.

Ví dụ:

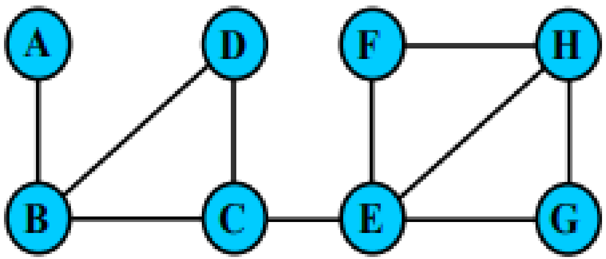


6. Tính liên thông trong đồ thị

6.1. Tính liên thông trong đồ thị vô hướng

- u được gọi là **đỉnh cắt** nếu như bỏ nó và các cạnh liên thuộc với nó đi thì sẽ làm tăng thành phần liên thông của đồ thị con.
- e được gọi là **cạnh cắt** nếu như xóa nó đi thì sẽ làm tăng thành phần liên thông của đồ thị con.

Ví dụ: Tìm đỉnh cắt và cạnh cắt



Đỉnh cắt là B, C, E.
Cạnh cắt là (A, B); (C, E)

6. Tính liên thông trong đồ thị

6.2. Tính liên thông trong đồ thị có hướng

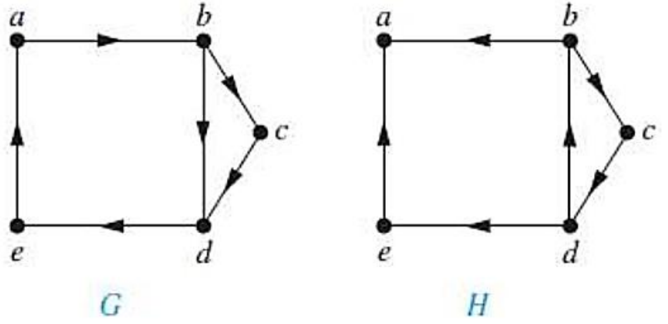
❖ Tính liên thông mạnh:

Nếu tồn tại đường đi giữa mọi cặp đỉnh u,v (2 chiều).

❖ Tính liên thông yếu:

Nếu tồn tại đường đi giữa 2 đỉnh bất kỳ trên đồ thị vô hướng cơ sở.

Ví dụ: Đồ thị G và H có là liên thông mạnh, liên thông yếu?



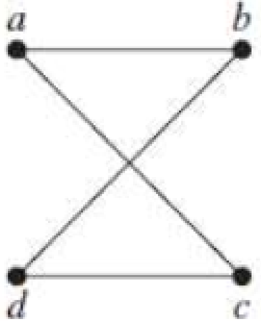
❖ Các đồ thị con của đồ thị có hướng G là liên thông mạnh nhưng không được chứa trong đồ thị con liên thông mạnh lớn hơn, nghĩa là đồ thị con liên thông lớn nhất, được gọi là thành phần liên thông mạnh của G.

6. Tính liên thông trong đồ thị

6.3. Đếm đường đi giữa các đỉnh

- ❖ Số đường đi giữa hai đỉnh trong một đồ thị có thể được xác định bằng việc sử dụng ma trận kề của nó
- ❖ **Định lý**: Gọi G là một đồ thị có ma trận kề A với thứ tự đỉnh là $v_1, v_2, \ldots, v_n$. Số đường đi khác nhau độ dài r từ v_i *đến* v_j *, trong đó r là một số nguyên dương, bằng phần tử (i,j) của ma trận* A^r

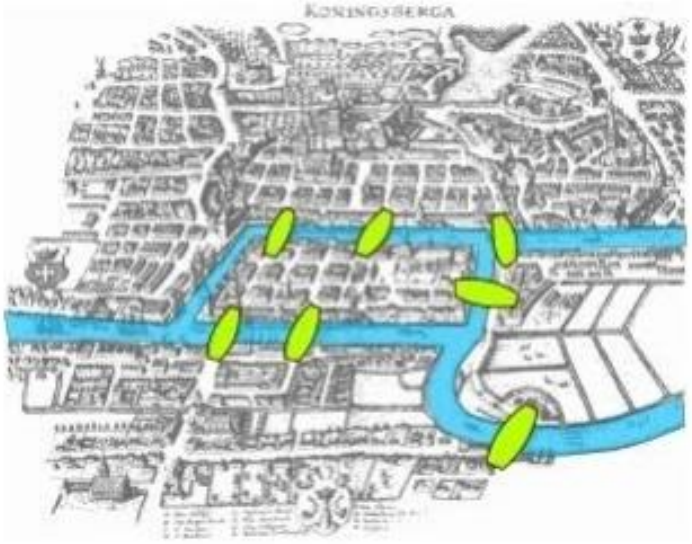
Ví dụ: Có bao nhiêu đường đi độ dài 4 từ a đến d trong đơn đồ thị sau?

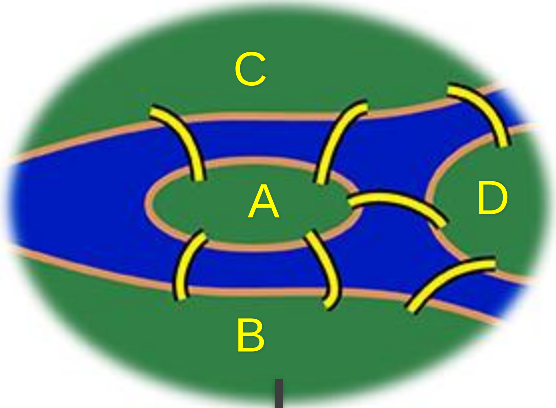


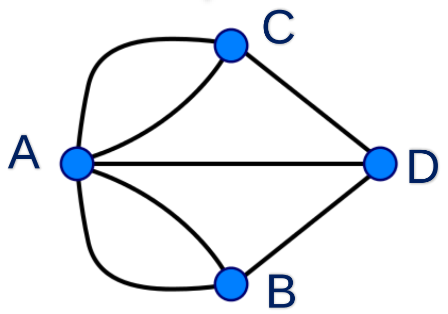
$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \qquad \mathbf{A}^4 = \begin{bmatrix} 8 & 0 & 0 & 8 \\ 0 & 8 & 8 & 0 \\ 0 & 8 & 8 & 0 \\ 8 & 0 & 0 & 8 \end{bmatrix}$$

7. Chu trình Euler và Hamilton

7.1. Đường đi và chu trình Euler







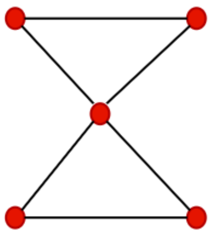
Bài toán 7 cây cầu tại thị trấn Konigsberg

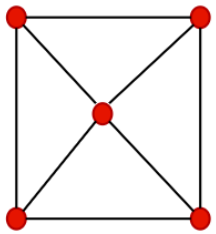
7. Chu trình Euler và Hamilton

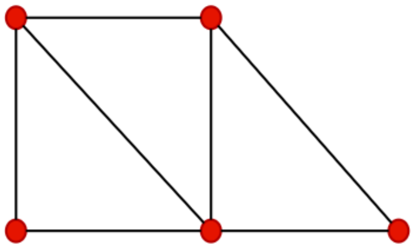
7.1. Đường đi và chu trình Euler

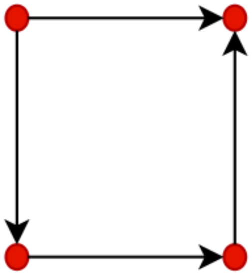
- **Chu trình Euler**: chu trình đơn đi qua tất cả các cạnh của đồ thị G đúng một lần.
- **Đường đi Euler**: đường đi đơn đi qua tất cả các cạnh của đồ thị G đúng một lần.

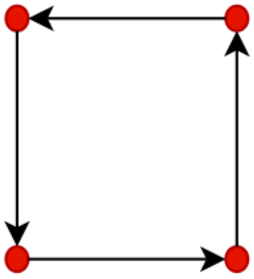
Ví dụ:

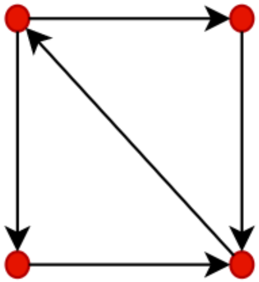












7. Chu trình Euler và Hamilton

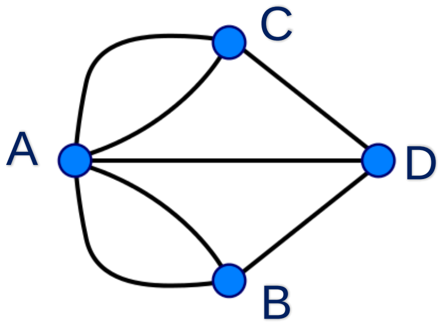
7.1. Đường đi và chu trình Euler

Điều kiện cần và đủ:

- ❖ **Định lý 1:** Một đa đồ thị liên thông có chu trình Euler khi và chỉ khi mỗi đỉnh của nó đều có bậc chẵn.
- ❖ **Định lý 2**: Một đa đồ thị liên thông có đường đi Euler nhưng không có chu trình Euler khi và chỉ khi nó có đúng hai đỉnh bậc lẻ.

Ứng dụng: Tìm hành trình ngắn nhất cho người đưa thư, xe thu rác, cảnh sát tuần tra,…

VD: Trả lời bài toán 7 cây cầu



7. Chu trình Euler và Hamilton

7.1. Đường đi và chu trình Euler

Thuật toán tìm chu trình Euler:

- ❖ **Bước 1:** Tạo mảng CE để ghi đường đi và một stack để xếp các đỉnh ta sẽ xét. Xếp vào đó 1 đỉnh u tùy ý nào đó của đồ thị, nghĩa là đỉnh u sẽ được xét đầu tiên.
- ❖ **Bước 2:** Xét đỉnh đầu ngăn xếp, giả sử đỉnh đó là v và thực hiện:
  - Nếu v là đỉnh cô lập thì lấy v khỏi ngăn xếp và đưa vào CE.
  - Nếu v là liên thông với đỉnh x thì xếp x vào ngăn xếp sau đó xóa bỏ cạnh (v,x).
  - Lặp lại cho đến khi ngăn xếp rỗng.
- ❖ **Bước 3:** Kết quả chu trình Euler được chứa trong CE theo thứ tự ngược lại.

7. Chu trình Euler và Hamilton

7.1. Đường đi và chu trình Euler

Bài tập: Cho ma trận kề của các đồ thị. Tìm chu trình Euler

$$\begin{vmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{vmatrix}$$

G

$$\begin{vmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{vmatrix}$$

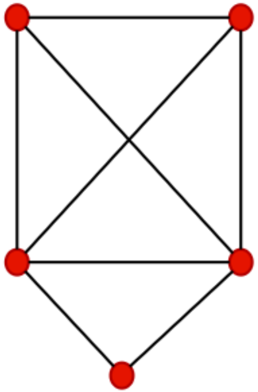
H

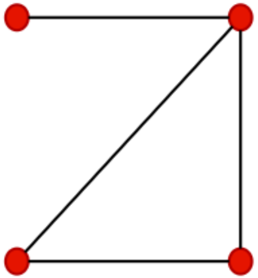
7. Chu trình Euler và Hamilton

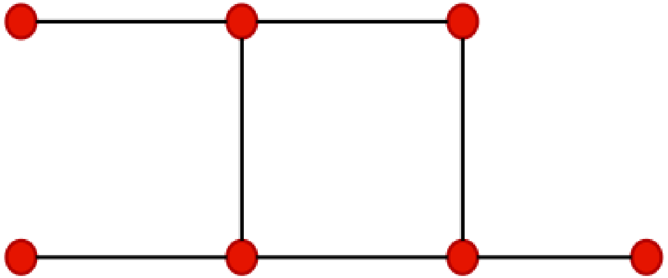
7.2. Đường đi và chu trình Hamilton

- ❖ **Chu trình Hamilton**: chu trình đơn đi qua tất cả các đỉnh của đồ thị G đúng một lần.
- ❖ **Đường đi Hamilton**: đường đi đơn đi qua tất cả các đỉnh của đồ thị G đúng một lần.

Ví dụ:







7. Chu trình Euler và Hamilton

7.2. Đường đi và chu trình Hamilton

- ☞ Không có chu trình Hamilton nếu:

  $\exists v \in V: deg(v) = 1$

- ☞ Định lý DIRAC: Đồ thị đơn G với *n* đỉnh ($n \geq 3$) có một chu trình Hamilton nếu:

  $deg(v) \geq n/2, \ \forall v \in V$

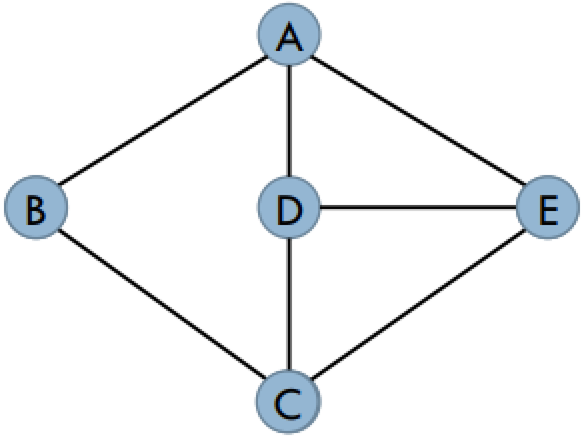
- ☞ Định lý ORE: Đồ thị đơn G với *n* đỉnh ($n \geq 3$) có một chu trình Hamilton nếu:

  $deg(u) + deg(v) \geq n, \ \forall u,v \in V$ và $(u,v) \notin E$

7. Chu trình Euler và Hamilton

7.2. Đường đi và chu trình Hamilton

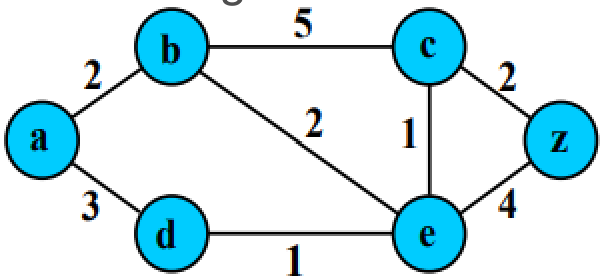
Bài tập: Tìm chu trình Hamilton



8. Bài toán tìm đường đi ngắn nhất

❖ **Bài toán**

- Có 6 điểm du lịch trong 1 khu sinh thái là a,b,c,d,e,z. Giữa 2 điểm có thể có hoặc không có đường đi trực tiếp.
  - Hãy tìm đường đi có khoảng cách ngắn nhất từ điểm a đến z.
- Bài toán được mô hình hóa bằng đồ thị có trọng số như sau:
  - Mỗi đỉnh biểu diễn một điểm du lịch.
  - Hai đỉnh có cạnh nối đều có đường đi trực tiếp.
  - Trọng số của cạnh được gán là khoảng cách từ điểm này sang điểm kia.
- Đồ thị mô hình bài toán:



- Đường đi ngắn nhất là đường đi có tổng trọng số các cạnh của nó là nhỏ nhất.

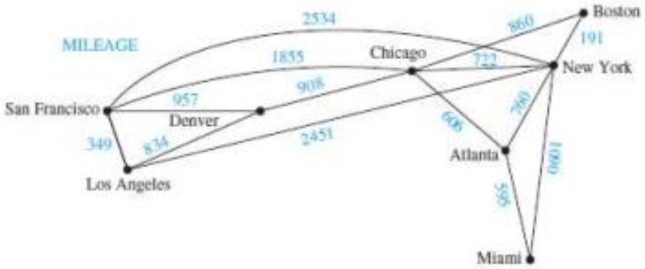
8. Bài toán tìm đường đi ngắn nhất

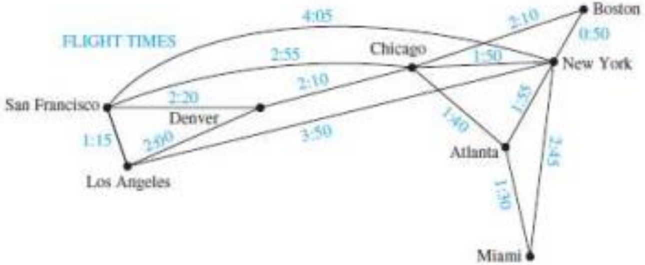
❖ **Đồ thị có trọng số:** là đồ thị mà mỗi cạnh được gán một số (nguyên hoặc thực) với ngụ ý nào đó.

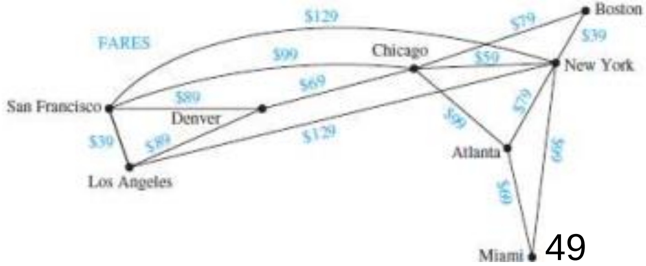
❖ Trọng số có thể biểu diễn thời gian, khoảng cách, chi phí,…

❖ Độ dài của đường đi có trọng số: Tổng trọng số của các cạnh trên đường đi.

→ Đường đi ngắn nhất: đường đi có độ dài nhỏ nhất trong số các đường đi có thể có.







8. Bài toán tìm đường đi ngắn nhất

❖ Thuật toán Dijkstra

- **Bài toán**: Tìm đường đi ngắn nhất từ đỉnh a đến z của đồ thị có trọng số liên thông G = (V,E).
- Gọi L(v) là độ dài đường đi ngắn nhất từ đỉnh a đến đỉnh v.
- S là tập các đỉnh đã tìm được đường đi ngắn nhất từ a đến nó.
- **Ý tưởng**:
  - Bước 1: $L(a) = 0$, $S = \varnothing$, $\forall v \in V, v \neq a: L(v) = \infty$
  - Bước 2: Nếu $z \in S$ thì kết thúc.
  - Bước 3: Chọn $v \notin S$ sao cho L(v) là nhỏ nhất. Đưa v vào S.
  - Bước 4: Với mỗi đỉnh x liền kề v và $x \notin S$ thì đặt:

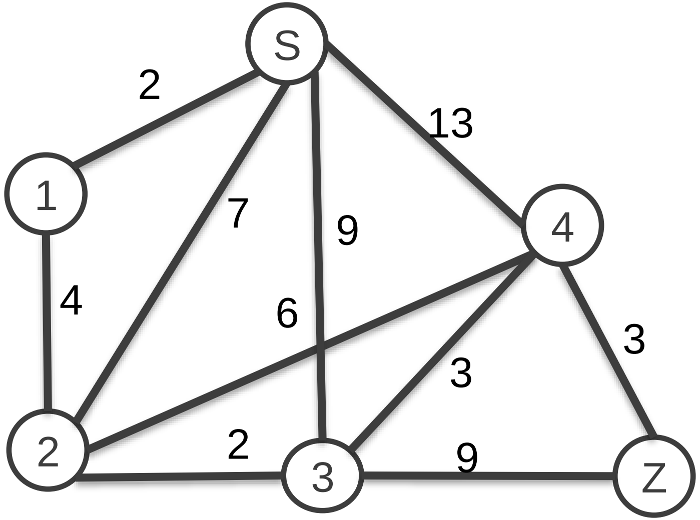
    $L(x) = \min\{L(x), L(v) + w(v,x)\}$

    Quay lại bước 2.

8. Bài toán tìm đường đi ngắn nhất

❖ **Thuật toán Dijkstra**

Ví dụ : Tìm đường đi ngắn nhất từ đỉnh S đến Z bằng thuật toán Dijkstra.



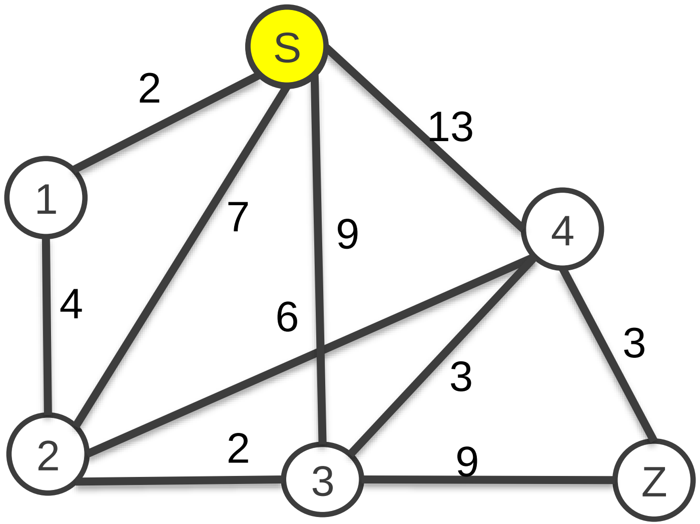
Bước 1: $L(a) = 0$, $S = \varnothing$, $\forall v \in V$, $v \neq a$: $L(v) = \infty$
Bước 2: Nếu $z \in S$ thì kết thúc.
Bước 3: Chọn $v \notin S$ sao cho $L(v)$ là nhỏ nhất. Đưa v vào S.
Bước 4: Với mỗi đỉnh x liền kề v và $x \notin S$ thì đặt:
$L(x) = \min\{L(x), L(v) + w(v,x)\}$
Quay lại bước 2.

8. Bài toán tìm đường đi ngắn nhất

❖ Thuật toán Dijkstra

Ví dụ: Tìm đường đi ngắn nhất từ đỉnh S đến Z bằng thuật toán Dijkstra.

k	Đỉnh S	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh Z
0	(0,S)*	(∞,S)	(∞,S)	(∞,S)	(∞,S)	(∞,S)



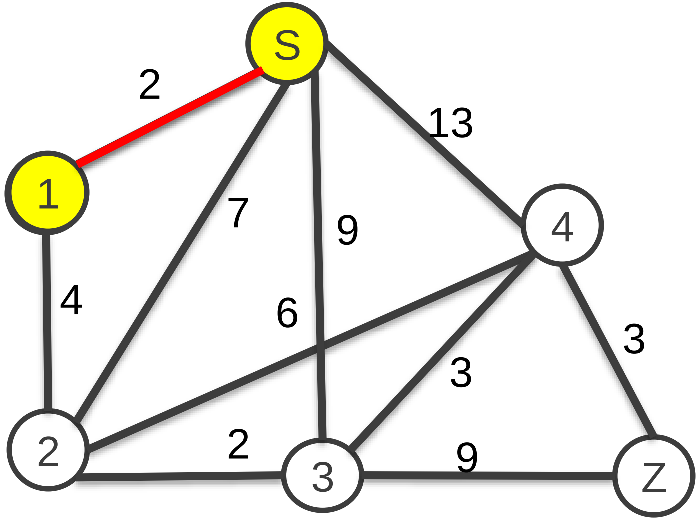
Bước 1: L(a) = 0, S = ∅, ∀v∈V, v ≠ a: L(v) = ∞
Bước 2: Nếu z ∈ S thì kết thúc.
Bước 3: Chọn v ∉ S sao cho L(v) là nhỏ nhất. Đưa v vào S.
Bước 4: Với mỗi đỉnh x liền kề v và x ∉ S thì đặt:
L(x) = min{L(x), L(v) + w(v,x)}
Quay lại bước 2.

8. Bài toán tìm đường đi ngắn nhất

❖ **Thuật toán Dijkstra**

Ví dụ: Tìm đường đi ngắn nhất từ đỉnh S đến Z bằng thuật toán Dijkstra.

k	Đỉnh S	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh Z
0	(0,S)*	(∞,S)	(∞,S)	(∞,S)	(∞,S)	(∞,S)
1	-	(2,S)*	(7,S)	(9,S)	(13,S)	(∞,S)



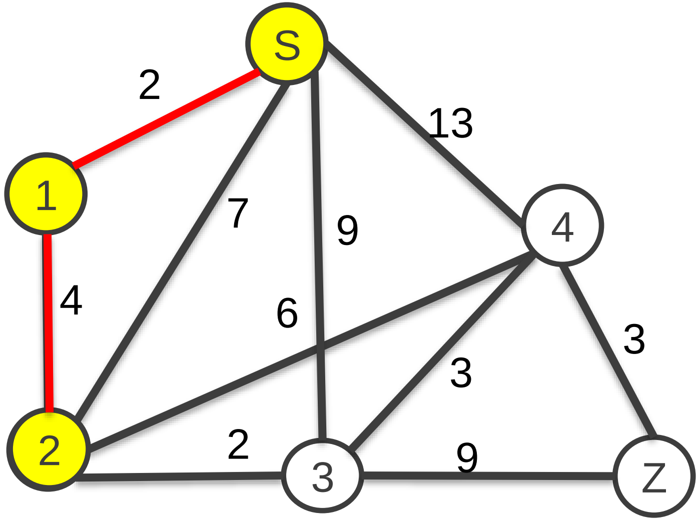
Bước 1: L(a) = 0, S = ∅, ∀v∈V, v ≠ a: L(v) = ∞
Bước 2: Nếu z ∈ S thì kết thúc.
Bước 3: Chọn v ∉ S sao cho L(v) là nhỏ nhất. Đưa v vào S.
Bước 4: Với mỗi đỉnh x liền kề v và x ∉ S thì đặt:
L(x) = min{L(x), L(v) + w(v,x)}
Quay lại bước 2.

8. Bài toán tìm đường đi ngắn nhất

❖ **Thuật toán Dijkstra**

Ví dụ: Tìm đường đi ngắn nhất từ đỉnh S đến Z bằng thuật toán Dijkstra.

k	Đỉnh S	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh Z
0	(0,S)*	(∞,S)	(∞,S)	(∞,S)	(∞,S)	(∞,S)
1	-	(2,S)*	(7,S)	(9,S)	(13,S)	(∞,S)
2	-	-	(6,1)*	(9,S)	(13,S)	(∞,S)



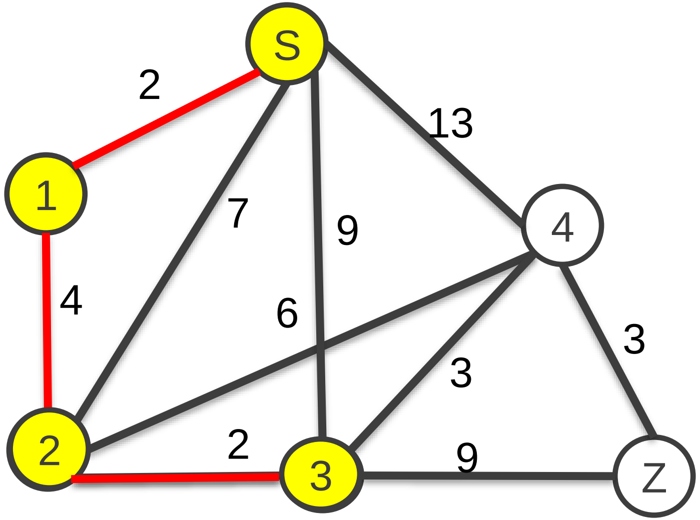
Bước 1: L(a) = 0, S = ∅, ∀v∈V, v ≠ a: L(v) = ∞
Bước 2: Nếu z ∈ S thì kết thúc.
Bước 3: Chọn v ∉ S sao cho L(v) là nhỏ nhất. Đưa v vào S.
Bước 4: Với mỗi đỉnh x liền kề v và x ∉ S thì đặt:
L(x) = min{L(x), L(v) + w(v,x)}
Quay lại bước 2.

8. Bài toán tìm đường đi ngắn nhất

❖ **Thuật toán Dijkstra**

Ví dụ: Tìm đường đi ngắn nhất từ đỉnh S đến Z bằng thuật toán Dijkstra.

k	Đỉnh S	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh Z
0	(0,S)*	(∞,S)	(∞,S)	(∞,S)	(∞,S)	(∞,S)
1	-	(2,S)*	(7,S)	(9,S)	(13,S)	(∞,S)
2	-	-	(6,1)*	(9,S)	(13,S)	(∞,S)
3	-	-	-	(8,2)*	(12,2)	(∞,S)



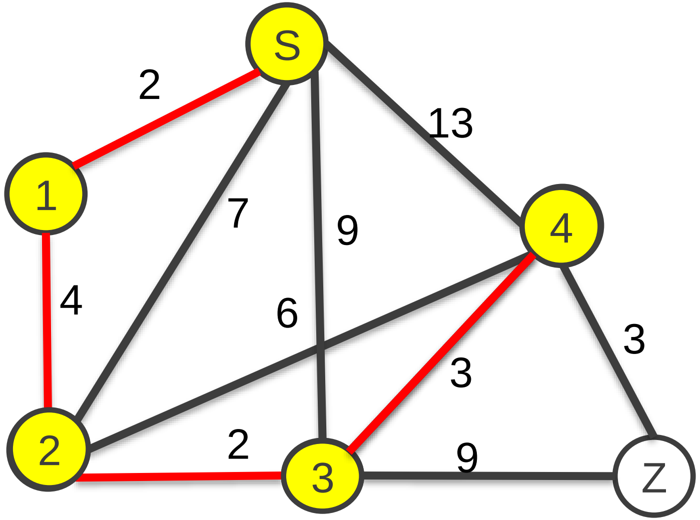
Bước 1: L(a) = 0, S = ∅, ∀v∈V, v ≠ a: L(v) = ∞
Bước 2: Nếu z ∈ S thì kết thúc.
Bước 3: Chọn v ∉ S sao cho L(v) là nhỏ nhất. Đưa v vào S.
Bước 4: Với mỗi đỉnh x liền kề v và x ∉ S thì đặt:
L(x) = min{L(x), L(v) + w(v,x)}
Quay lại bước 2.

8. Bài toán tìm đường đi ngắn nhất

❖ **Thuật toán Dijkstra**

Ví dụ: Tìm đường đi ngắn nhất từ đỉnh S đến Z bằng thuật toán Dijkstra.

k	Đỉnh S	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh Z
0	(0,S)*	(∞,S)	(∞,S)	(∞,S)	(∞,S)	(∞,S)
1	-	(2,S)*	(7,S)	(9,S)	(13,S)	(∞,S)
2	-	-	(6,1)*	(9,S)	(13,S)	(∞,S)
3	-	-	-	(8,2)*	(12,2)	(∞,S)
4	-	-	-	-	(11,3)*	(17,3)



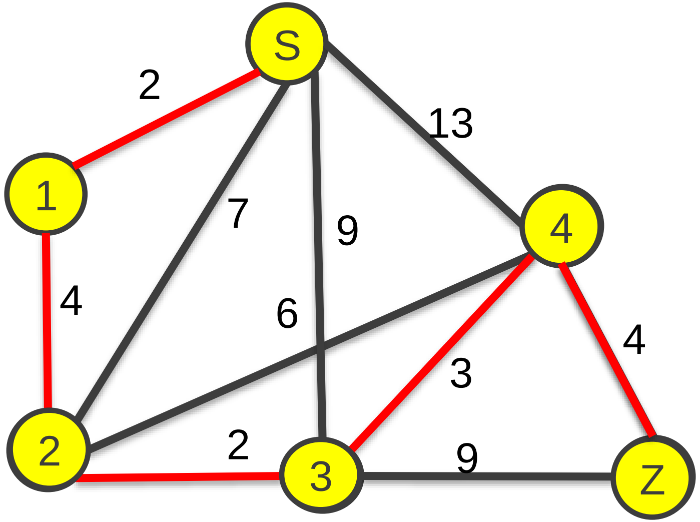
Bước 1: L(a) = 0, S = ∅, ∀v∈V, v ≠ a: L(v) = ∞
Bước 2: Nếu z ∈ S thì kết thúc.
Bước 3: Chọn v ∉ S sao cho L(v) là nhỏ nhất. Đưa v vào S.
Bước 4: Với mỗi đỉnh x liền kề v và x ∉ S thì đặt:
L(x) = min{L(x), L(v) + w(v,x)}
Quay lại bước 2.

8. Bài toán tìm đường đi ngắn nhất

❖ **Thuật toán Dijkstra**

Ví dụ: Tìm đường đi ngắn nhất từ đỉnh S đến Z bằng thuật toán Dijkstra.

k	Đỉnh S	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh Z
0	(0,S)*	(∞,S)	(∞,S)	(∞,S)	(∞,S)	(∞,S)
1	-	(2,S)*	(7,S)	(9,S)	(13,S)	(∞,S)
2	-	-	(6,1)*	(9,S)	(13,S)	(∞,S)
3	-	-	-	(8,2)*	(12,2)	(∞,S)
4	-	-	-	-	(11,3)*	(17,3)
5	-	-	-	-	-	(15,4)

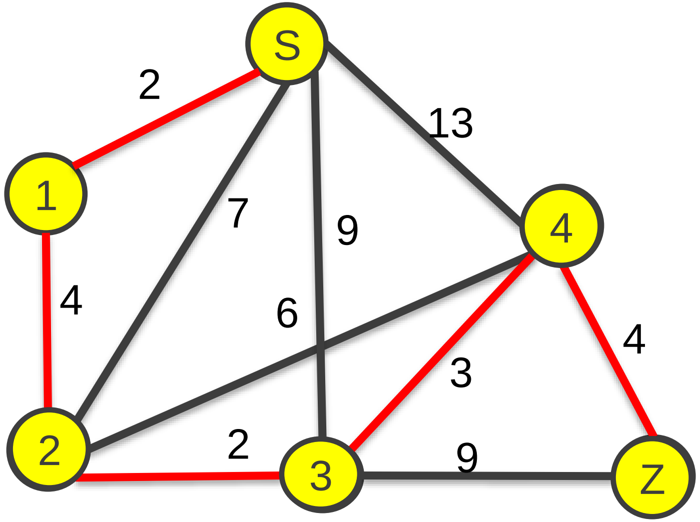


8. Bài toán tìm đường đi ngắn nhất

❖ **Thuật toán Dijkstra**

Ví dụ: Tìm đường đi ngắn nhất từ đỉnh S đến Z bằng thuật toán Dijkstra.

k	Đỉnh S	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh Z
0	(0,S)*	(∞,S)	(∞,S)	(∞,S)	(∞,S)	(∞,S)
1	-	(2,S)*	(7,S)	(9,S)	(13,S)	(∞,S)
2	-	-	(6,1)*	(9,S)	(13,S)	(∞,S)
3	-	-	-	(8,2)*	(12,2)	(∞,S)
4	-	-	-	-	(11,3)*	(17,3)
5	-	-	-	-	-	(15,4)
	-	-	-	-	-	-

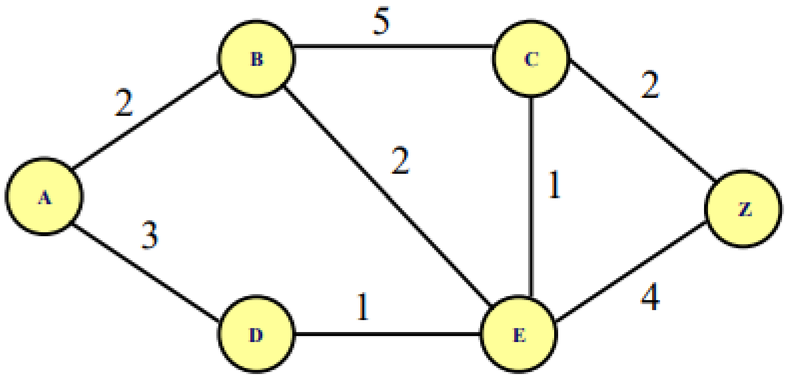


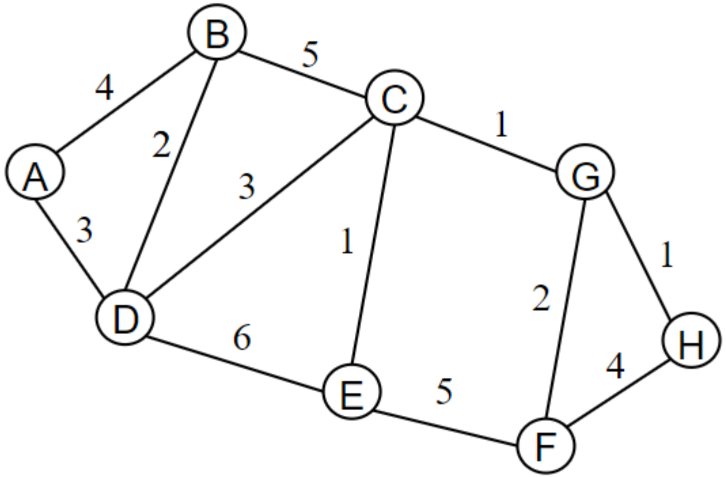
Đường đi ngắn nhất: S->1->2->3->4->Z. Độ dài đường đi là 15

8. Bài toán tìm đường đi ngắn nhất

❖ **Bài tập**

Tìm đường đi ngắn nhất từ đỉnh A đến Z bằng thuật toán Dijkstra.

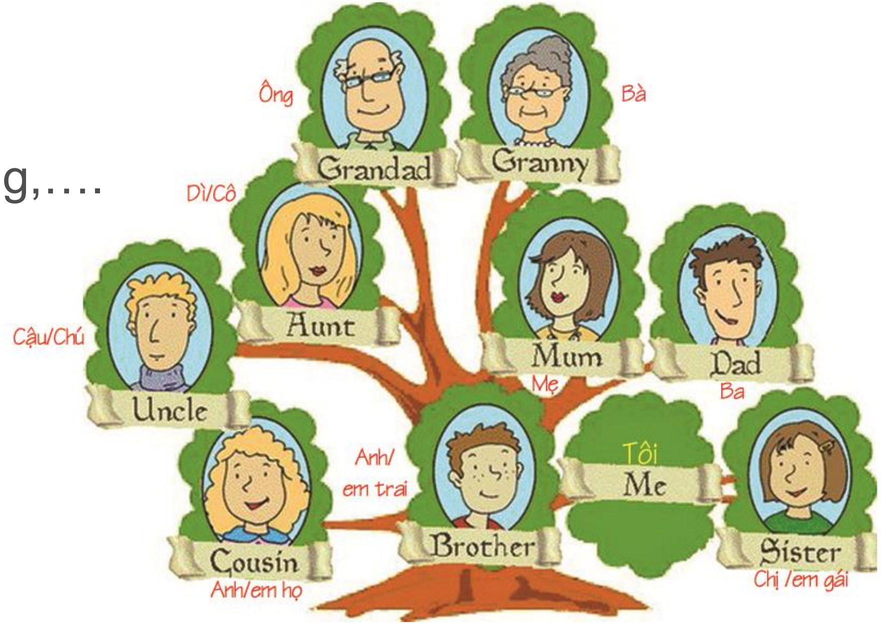




9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

- ❖ **Cây** là một đồ thị vô hướng, liên thông và không chứa chu trình đơn.
- ❖ Ứng dụng trong KHMT:
  - ➢ Các thuật toán tìm kiếm.
  - ➢ Thiết kế mạng máy tính.
  - ➢ Lưu trữ dữ liệu trên ổ cứng,….
- ❖ Khác:
  - ➢ Sơ đồ tổ chức hoạt động.
  - ➢ Công thức hóa học.

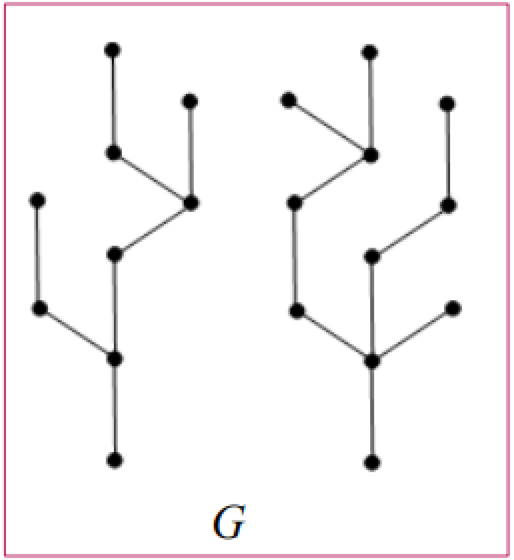


9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

- **Rừng:** là đồ thị vô hướng, không liên thông và không chứa chu trình đơn.

→ Rừng gồm nhiều cây.

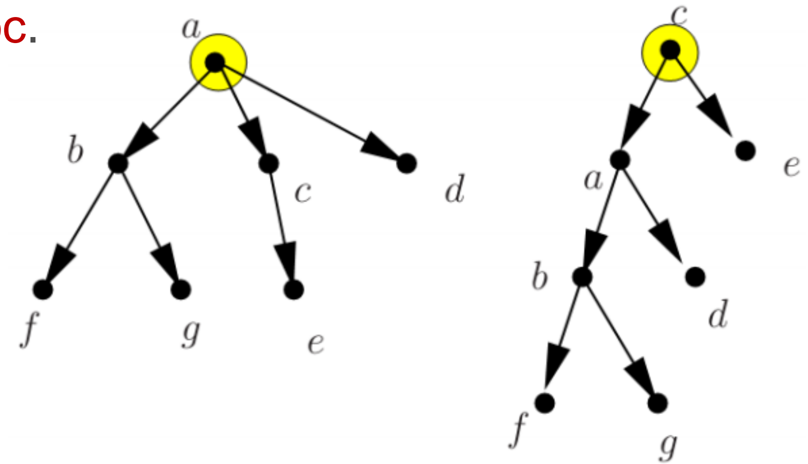


9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

- **Tính chất**:
  - Giữa 2 đỉnh bất kỳ có duy nhất 1 đường đi đơn.
  - Cây n đỉnh sẽ có n-1 cạnh.

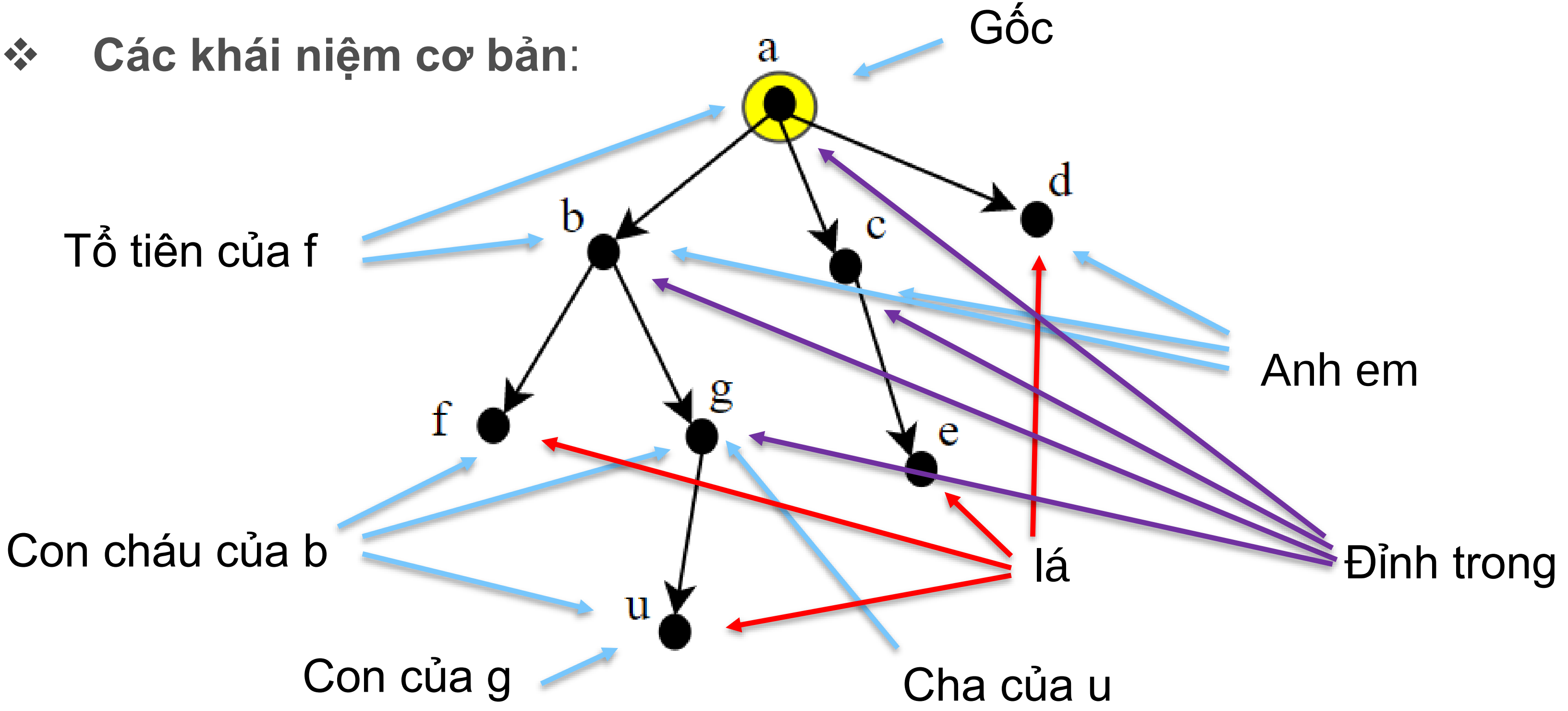
    Nếu thêm 1 cạnh tùy ý -> Tạo ra 1 chu trình.
- **Cây có gốc:** Một đỉnh được chỉ định là gốc, các cạnh đều có hướng và hướng này đi ra xa gốc.



9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

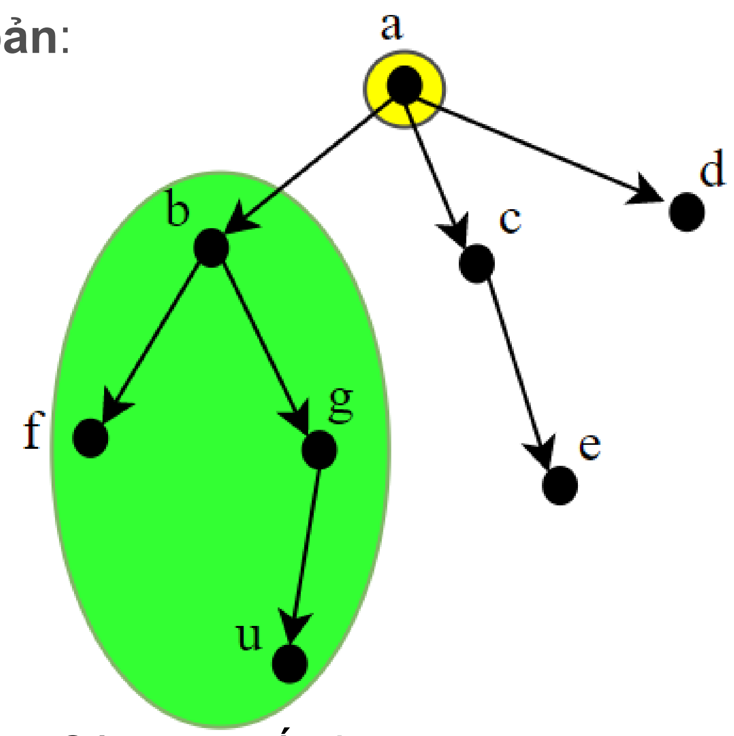
❖ **Các khái niệm cơ bản:**



9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

❖ **Các khái niệm cơ bản:**



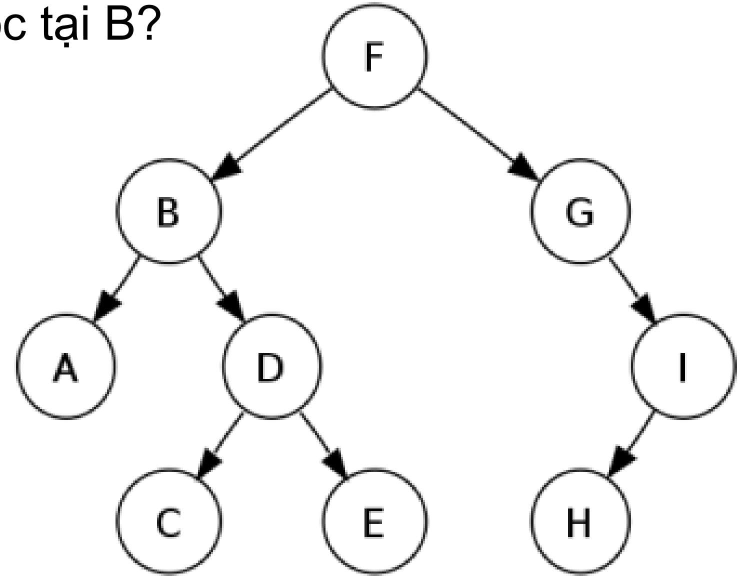
Cây con gốc b

9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

❖ **Bài tập:**

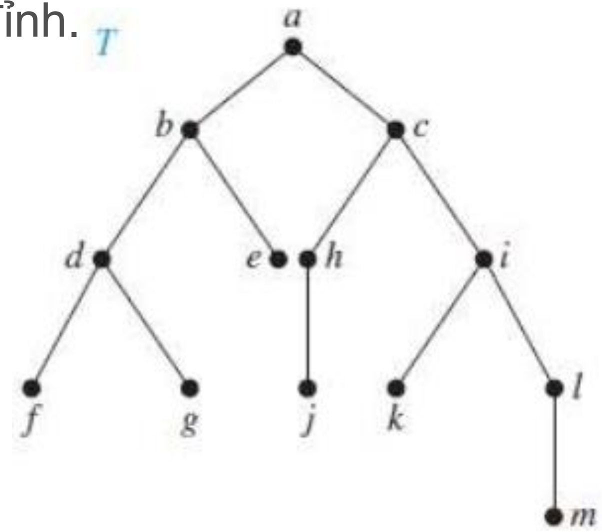
Trong cây dưới đây (với gốc F) tìm cha của A, con của D, anh em của I, tất cả tổ tiên của H, tất cả con cháu của G, tất cả đỉnh trong, tất cả lá. Tìm cây con có gốc tại B?



9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

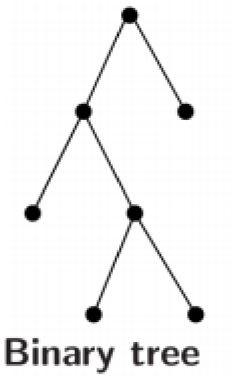
- ❖ **Cây có gốc được sắp xếp:** là một cây có gốc trong đó các con của mỗi đỉnh trong được sắp xếp theo thứ tự từ trái sang phải.
- ❖ Trong cây nhị phân được sắp xếp, nếu đỉnh trong có 2 con, con thứ nhất gọi là **con trái**, con thứ hai gọi là **con phải**. Cây có gốc tại con trái của một đỉnh gọi là **cây con trái** của đỉnh và cây có gốc tại con phải của một đỉnh gọi là **cây con phải** của đỉnh.
- ❖ **Ví dụ:** Tìm con trái và con phải của d?
  Tìm cây con trí và cây con phải của c?

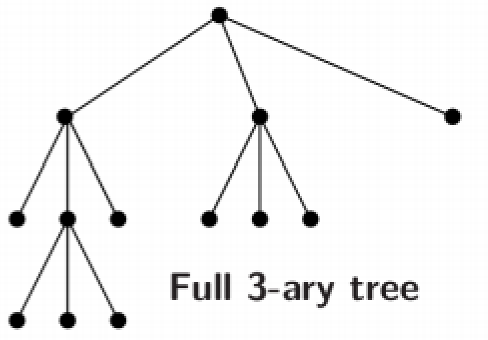


9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

- Một cây có gốc gọi là *phân m* nếu mọi đỉnh trong của nó có không ít hơn *m* con.
- Cây là *phân m* đầy đủ nếu mọi đỉnh trong của nó có chính xác *m* con.
- Một cây phân m đầy đủ có:
  - n đỉnh có $i = (n-1)/m$ đỉnh trong và $l = [(m-1)n + 1]/m$ lá
  - i đỉnh trong có $n = mi + 1$ đỉnh và $l = (m-1)i + 1$ lá
  - l lá có $n = (ml - 1)/(m-1)$ đỉnh và $i = (l-1)/(m-1)$ đỉnh trong
- **Ví dụ:**



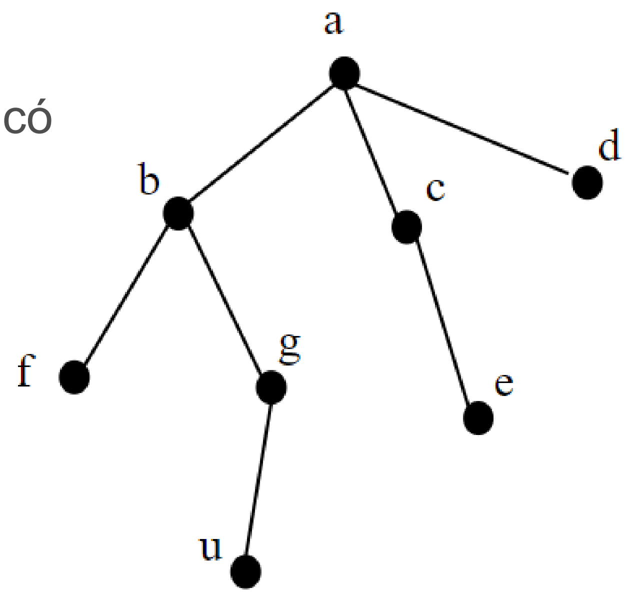


9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

- Bậc của đỉnh v trong cây có gốc là độ dài của đường đi duy nhất từ gốc đến đỉnh này. Bậc của gốc là 0.
- Độ cao của một cây có gốc là giá trị lớn nhất của tất cả bậc của các đỉnh.

- **Ví dụ:** Tìm bậc của mỗi đỉnh trong cây có gốc sau. Cho biết độ cao của cây này?

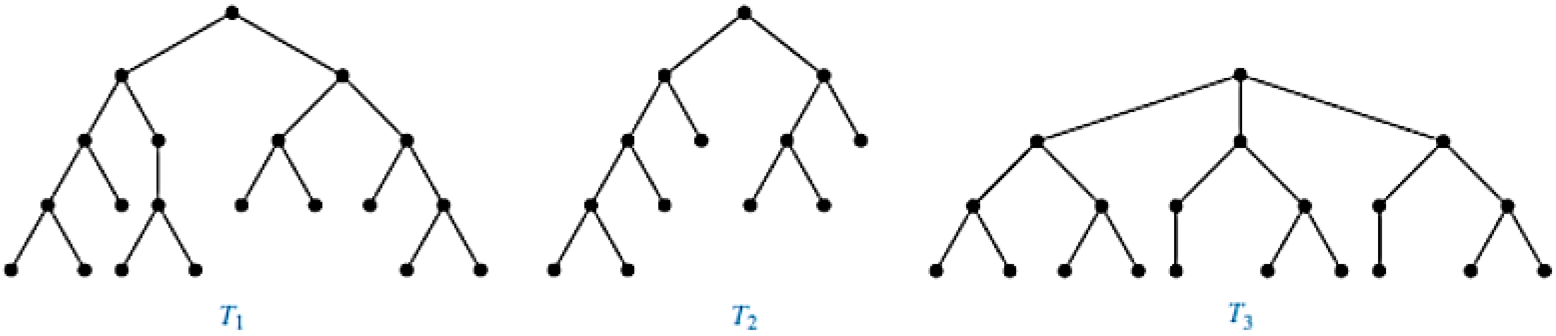


  - Gốc a có bậc 0
  - Các đỉnh b, c, d có bậc 1.
  - Các đỉnh f, g, e có bậc 2.
  - Đỉnh u có bậc 3.
  - Độ cao của cây là 4.

9. Cây và các khái niệm cơ bản

9.1. Giới thiệu về cây

- ❖ Cây chia m cân bằng: Một cây chia m có gốc có độ cao h được gọi là cân bằng nếu tất cả lá có bậc h hoặc h-1.
- ❖ Có nhiều nhất m^h lá trong một cây phân m có độ cao h.
- ❖ Nếu một cây chia m có độ cao h có l lá, thì $h \geq \lceil \log_m l \rceil$
- ❖ Nếu cây phân m là đầy đủ và cân bằng thì $h = \lceil \log_m l \rceil$
- ❖ Ví dụ: Cây có gốc nào là cân bằng?



Cân bằng | Không cân bằng | Cân bằng

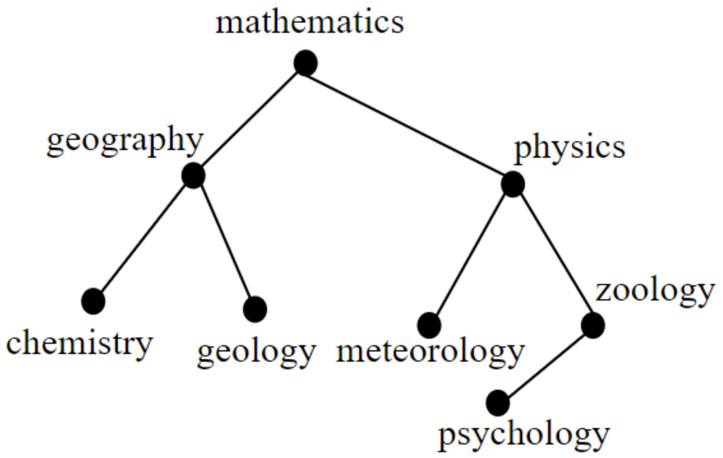
9. Cây và các khái niệm cơ bản

9.2. Ứng dụng của cây

Cây tìm kiếm nhị phân

- Là một cây nhị phân trong đó mỗi con của một đỉnh gọi là con trái và con phải, không có đỉnh nào có nhiều hơn 2 con.
- Mỗi đỉnh được gán nhãn bằng 1 key, là một trong các phần tử.
- Các đỉnh được gán nhãn sao cho key của 1 đỉnh lớn hơn các key của mọi đỉnh trong cây con trái và nhỏ hơn các key của mọi đỉnh trong cây con phải.

- Vd: Tạo 1 cây tìm kiếm nhị phân cho từ các từ mathematics, physics, geography, zoology, meteorology, geology, psychology, chemistry (theo thứ tự abc).

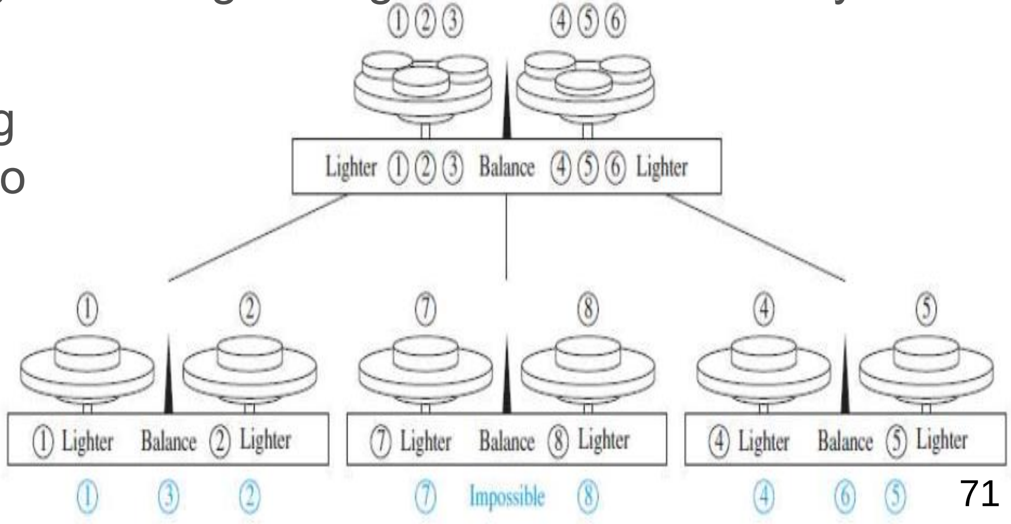


9. Cây và các khái niệm cơ bản

9.2. Ứng dụng của cây

Cây quyết định

- ❖ Một cây có gốc mà mỗi đỉnh trong tương ứng với một quyết định.
- ❖ Một cây con tại một đỉnh sẽ cho kết quả tương ứng với quyết định tại đỉnh đó, nó được gọi là cây quyết định.
- ❖ Lời giải bài toán tương ứng với những đường đi đến các lá của cây.
- ❖ Vd: Giả sử có 7 đồng xu cùng trọng lượng và 1 đồng xu giả nhẹ hơn. Hỏi cần bao nhiêu lần sử dụng một cân cân bằng để xác định đồng xu nào là giả. Đưa ra thuật toán tìm đồng xu giả?



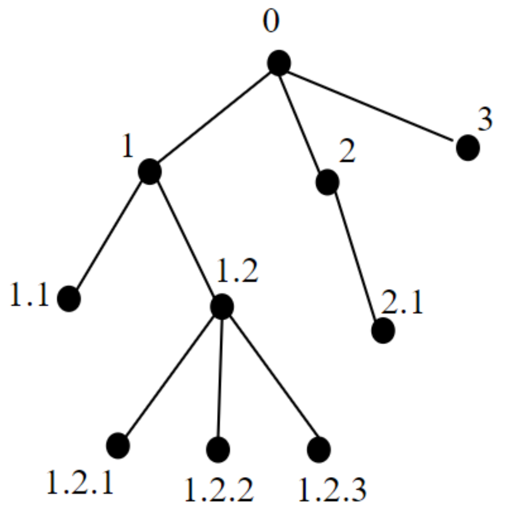
9. Cây và các khái niệm cơ bản

9.3. Duyệt cây

Hệ thống địa chỉ toàn bộ

- ❖ Các thủ tục duyệt toàn bộ đỉnh của một cây có gốc được sắp thứ tự phụ thuộc vào sự sắp xếp của các con của nó.
- ❖ Các con của một đỉnh trong được biểu diễn từ trái sang phải.
- ❖ Để tạo ra sự sắp xếp này, phải gán nhãn cho toàn bộ đỉnh. Thực hiện đệ quy sau:

1. Gán nhãn cho gốc là 0.
2. Gán nhãn k cho các con của nó (tại mức 1) từ trái sang phải các số 1,2,…,k.
3. Với mỗi đỉnh v ở mức n có nhãn A, gán nhãn k_v cho con của nó từ trái sang phải các số A.1,A.2,…,A.k_v

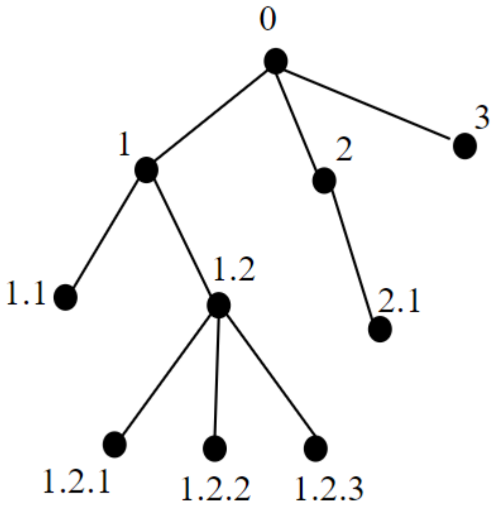


9. Cây và các khái niệm cơ bản

9.3. Duyệt cây

Hệ thống địa chỉ toàn bộ

- Có thể sắp toàn bộ các đỉnh sử dụng cách sắp xếp dựa trên nhãn của chúng trong hệ thống địa chỉ toàn bộ.
- Đỉnh có nhãn $x_1, x_2,\ldots x_n$ là nhỏ hơn đỉnh có nhãn $y_1, y_2,\ldots, y_m$ nếu có 1 giá trị I, $0\le i \le n$ với $x_1 = y_1, x_2 = y_2,\ldots, x_{i-1} = y_{i-1}$ và $x_i < y_i$ hoặc nếu $n < m$ và $x_i < y_i$ với I = 1,2,…,n.

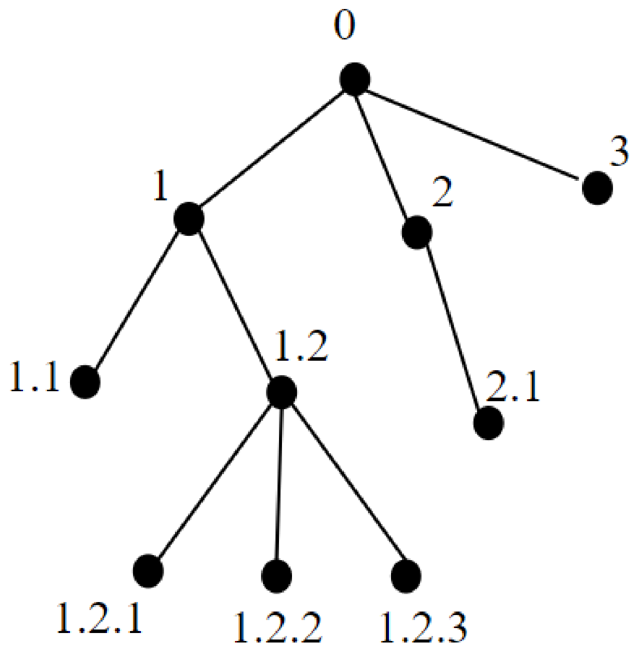


9. Cây và các khái niệm cơ bản
9.3. Duyệt cây

Hệ thống địa chỉ toàn bộ

❖ Ví dụ: Hãy cho biết thứ tự các nhãn có trong hệ thống địa chỉ của cây có trong hình.

0 < 1 < 1.1 < 1.2 < 1.2.1 < 1.2.2 < 1.2.3
< 2 < 2.1
<3



9. Cây và các khái niệm cơ bản

9.3. Duyệt cây

Các thuật toán duyệt cây

- ❖ Các thủ tục nhằm thăm mọi đỉnh của một cây có gốc được sắp xếp thứ tự một cách có hệ thống được gọi là các thuật toán duyệt cây.
- ❖ Các thuật toán sử dụng: duyệt trước, duyệt giữa, duyệt sau,…
- ❖ Mỗi thuật toán có thể được định nghĩa đệ quy.

9. Cây và các khái niệm cơ bản

9.3. Duyệt cây

Định nghĩa 1: Duyệt trước

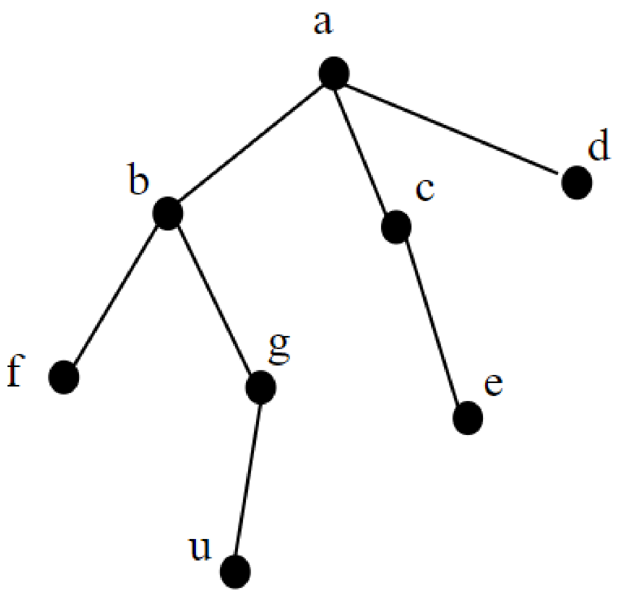
- ❖ Gọi T là một cây có gốc được sắp thứ tự với gốc r.
- ❖ Nếu T chỉ chứa r, thì r là kết quả duyệt trước của T.
- ❖ Ngược lại, giả sử T1, T2,…, Tn là các cây con của r từ trái sang phải trong T.
- ❖ Duyệt trước bắt đầu bằng việc thăm r.
- ❖ Sau đó tiếp tục duyệt T1 theo thứ tự trước. Tiếp duyệt T2 theo thứ tự trước. Và tiếp tục đến khi Tn được duyệt theo thứ tự trước.

9. Cây và các khái niệm cơ bản

9.3. Duyệt cây

Định nghĩa 1: Duyệt trước

❖ **Ví dụ**: Cho biết kết quả duyệt trước các đỉnh cây sau theo thứ tự trước.



a – b – f – g – u – c – e – d.

9. Cây và các khái niệm cơ bản

9.3. Duyệt cây

Định nghĩa 2: Duyệt giữa

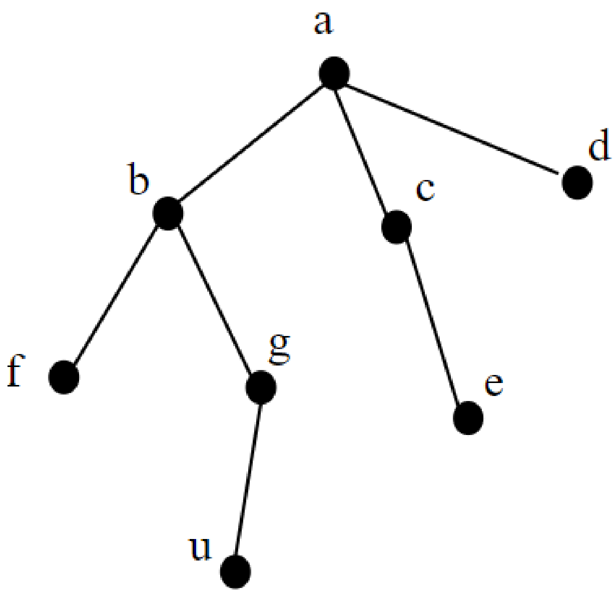
- Gọi T là một cây có gốc được sắp thứ tự với gốc r.
- Nếu T chỉ chứa r, thì r là kết quả duyệt giữa của T.
- Ngược lại, giả sử T1, T2,…, Tn là các cây con của r từ trái sang phải trong T.
- Duyệt giữa bắt đầu bằng duyệt T1 theo thứ tự giữa sau đó thăm r.
- Tiếp duyệt T2 theo thứ tự giữa. Và tiếp tục đến khi Tn được duyệt theo thứ tự giữa.

9. Cây và các khái niệm cơ bản

9.3. Duyệt cây

Định nghĩa 2: Duyệt giữa

❖ **Ví dụ**: Cho biết kết quả duyệt giữa các đỉnh cây sau theo thứ tự giữa.



f – b – u – g – a – e – c – d.

9. Cây và các khái niệm cơ bản

9.3. Duyệt cây

Định nghĩa 3: Duyệt sau

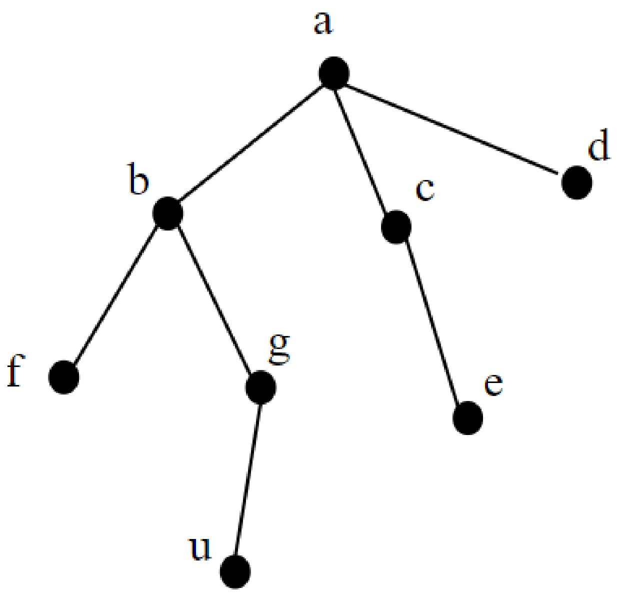
- Gọi T là một cây có gốc được sắp thứ tự với gốc r.
- Nếu T chỉ chứa r, thì r là kết quả duyệt sau của T.
- Ngược lại, giả sử T1, T2,…, Tn là các cây con của r từ trái sang phải trong T.
- Duyệt sau bắt đầu bằng duyệt T1 theo thứ tự sau.
- Tiếp duyệt T2 theo thứ tự sau. Và tiếp tục đến khi Tn được duyệt theo thứ tự sau.
- Kết thúc bằng thăm r.

9. Cây và các khái niệm cơ bản

9.3. Duyệt cây

Định nghĩa 3: Duyệt sau

- **Ví dụ**: Cho biết kết quả duyệt sau các đỉnh cây sau theo thứ tự sau.



f – u – g – b – e – c – d – a.

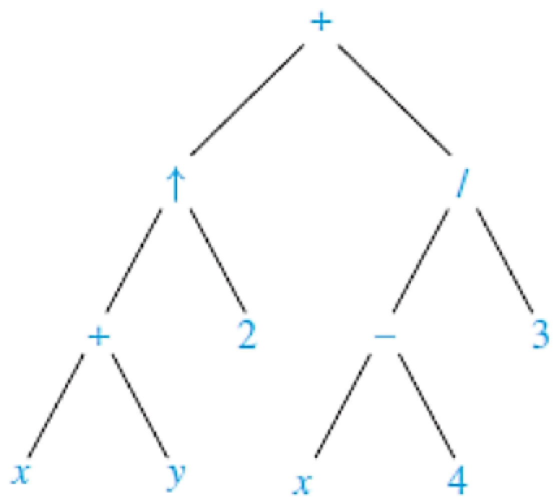
9. Cây và các khái niệm cơ bản

9.4. Kí hiệu trung tố, tiền tố và hậu tố

❖ Có thể biểu diễn các biểu thức phức tạp chẳng hạn như các mệnh đề phức, kết hợp các tập và các biểu thức toán học sử dụng cây có gốc được sắp xếp.

❖ VD: Tìm cây biểu thức biểu diễn biểu thức ((x+y)↑2) + ((x-4)/3)

➢ Các đỉnh là phép toán
➢ Các lá là các biến hoặc số
➢ Mỗi phép toán thực hiện trên các cây con trái và phải (theo thứ tự đó).



9. Cây và các khái niệm cơ bản

9.4. Kí hiệu trung tố, tiền tố và hậu tố

Dạng trung tố của biểu thức

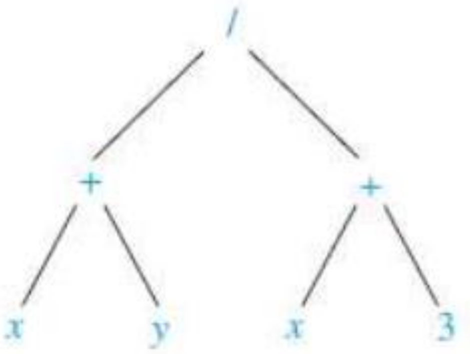
- ❖ Một phép duyệt giữa của một cây nhị phân biểu diễn một biểu thức sinh ra biểu thức ban đầu với các phần tử và phép toán cùng thứ tự như chúng xuất hiện trong biểu thức ban đầu, ngoại trừ phép toán một ngôi.
- ❖ Để không gây nhầm lẫn giữa các biểu thức, cần sử dụng dấu đóng mở ngoặc đơn trong phép duyệt giữa.
- ❖ Biểu thức được đóng mở ngoặc đơn đầy đủ thi được theo cách này gọi là dạng trung tố.

9. Cây và các khái niệm cơ bản

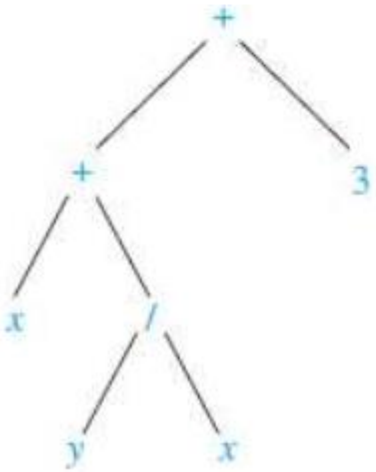
9.4. Kí hiệu trung tố, tiền tố và hậu tố

Dạng trung tố của biểu thức

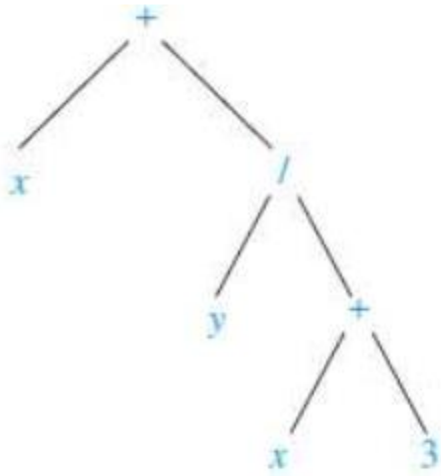
❖ Ví dụ: Hãy tìm các biểu thức dạng trung tố của các cây sau?



(x+y)/(x+3)



(x+(y/x))+3



x+(y/(x+3))

9. Cây và các khái niệm cơ bản

9.4. Kí hiệu trung tố, tiền tố và hậu tố

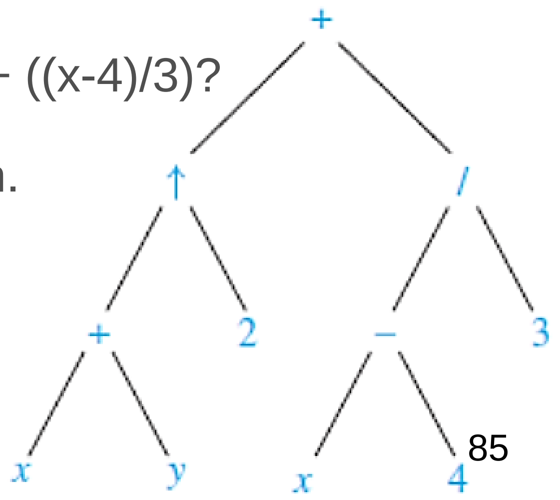
Dạng tiền tố của biểu thức

❖ Dạng tiền tố của biểu thức thu được bằng cách duyệt cây nhị phân biểu diễn nó theo thứ tự duyệt trước.

❖ Phép toán hai ngôi, chẳng hạn +, đứng trước 2 toán hạng của nó. Do đó có thể tính giá trị biểu thức bằng cách thực hiện từ phải sang trái.

❖ VD: Cho biết dạng tiền tố của biểu thức ((x+y)↑2) + ((x-4)/3)?

- Xây dựng cây nhị phân biểu diễn biểu thức trên.
- Duyệt cây nhị phân theo thứ tự duyệt trước
- Kết quả: + ↑ + x y 2 / - x 4 3



9. Cây và các khái niệm cơ bản

9.4. Kí hiệu trung tố, tiền tố và hậu tố

Dạng tiền tố của biểu thức

❖ Ví dụ: Tính giá trị của biểu thức tiền tố + - * 2 3 5 / ↑ 2 3 4?

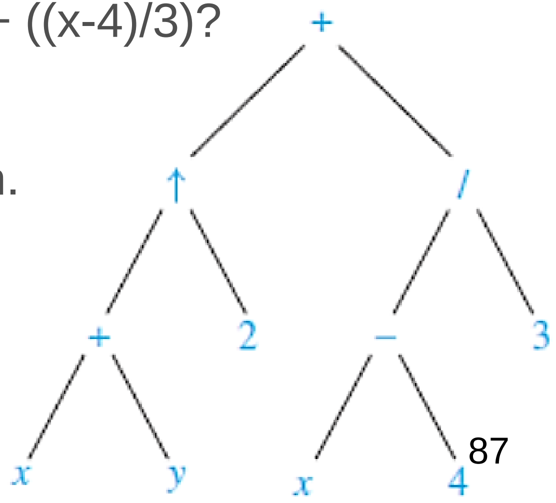
- + - * 2 3 5 / ↑ 2 3 4
- + - * 2 3 5 / 8 4
- + - * 2 3 5 2
- + - 6 5 2
- + 1 2
- Kết quả là 3

9. Cây và các khái niệm cơ bản

9.4. Kí hiệu trung tố, tiền tố và hậu tố

Dạng hậu tố của biểu thức

- ❖ Dạng hậu tố của biểu thức thu được bằng cách duyệt cây nhị phân biểu diễn nó theo thứ tự duyệt sau.
- ❖ Phép toán hai ngôi đứng sau 2 toán hạng của nó. Do đó có thể tính giá trị biểu thức bằng cách thực hiện từ trái sang phải.
- ❖ VD: Cho biết dạng tiền tố của biểu thức ((x+y)↑2) + ((x-4)/3)?



- Xây dựng cây nhị phân biểu diễn biểu thức trên.
- Duyệt cây nhị phân theo thứ tự duyệt sau
- Kết quả: x y + 2 ↑ x 4 – 3 / +

10. Cây khung và các thuật toán

- Cây khung: Cho G là một đồ thị. Một đồ thị con của G và chứa tất cả các đỉnh của G gọi là cây khung của G.

☞ Định lý: Đồ thị G có cây khung nếu và chỉ nếu nó liên thông.

- Xác định cây khung là việc xây dựng một cây chứa tất cả các đỉnh của đồ thị.
- Hai thuật toán xác định cây khung là:
  - Xác định ưu tiên theo chiều rộng
  - Xác định ưu tiên theo chiều sâu

10. Cây khung và các thuật toán

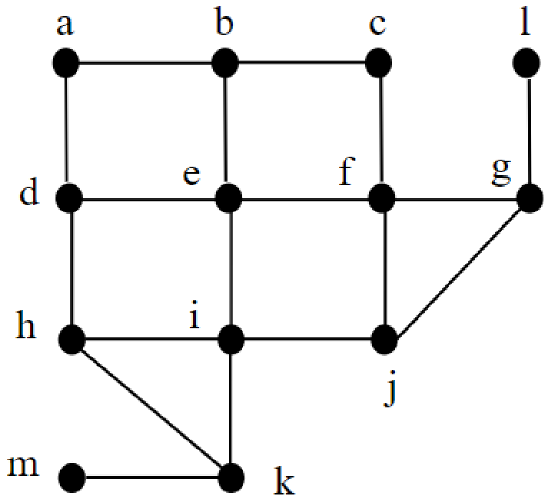
Theo chiều rộng (BFS)

- Bước 1: Lấy một đỉnh a làm gốc của cây khung.
- Bước 2: Ghép các cạnh liên thuộc với gốc. Các đỉnh kề với gốc trong bước này có mức là 1.
- Bước 3: Tiếp tục ghép các cạnh liên thuộc đỉnh mức 1 sao cho không tạo chu trình. Các đỉnh được đưa vào ở bước này có mức là 2.
- Bước 4: Tiếp tục quá trình khi tất cả các đỉnh đã được ghép vào cây.

10. Cây khung và các thuật toán

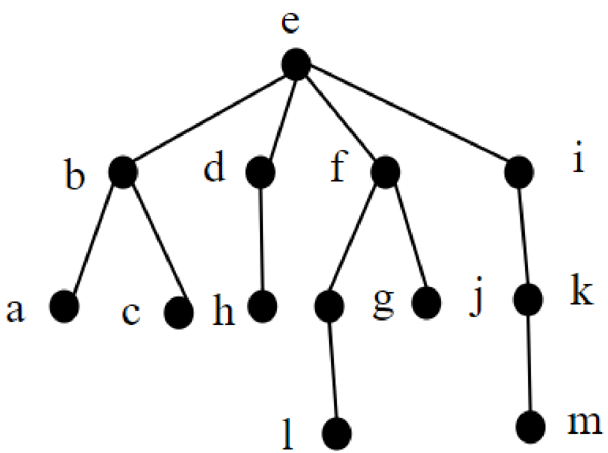
Theo chiều rộng (BFS)

- ❖ Ví dụ: Tìm một cây khung cho đồ thị bên bằng thuật toán tìm kiếm theo chiều rộng



Lời giải:

Giả sử chọn đỉnh e đầu tiên



10. Cây khung và các thuật toán

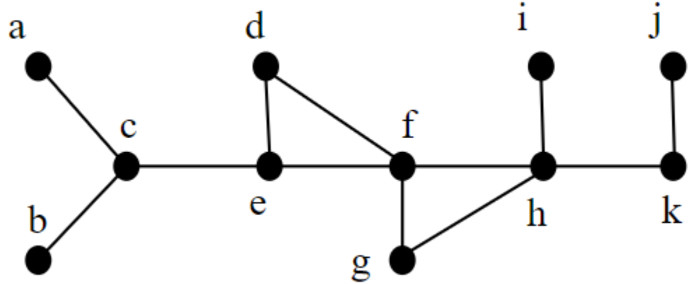
Theo chiều sâu (DFS)

- ❖ Bước 1: Lấy một đỉnh a làm gốc của cây khung.
- ❖ Bước 2: Xây dựng đường đi từ đỉnh này bằng cách ghép lần lượt các cạnh vào. Mỗi cạnh được ghép vào nối đỉnh cuối cùng của đường đi và một đỉnh chưa thuộc đường đi. Thực hiện đến khi không ghép được thêm cạnh nào nữa
- ❖ Bước 3: Nếu đường đi chứa tất cả các đỉnh của đồ thị thì đó chính là cây khung. Nếu không thì chuyển sang bước 4.
- ❖ Bước 4: Quay lui lại đỉnh ngay trước đỉnh cuối cùng của đường đi và xây dựng đường đi mới bắt đầu từ đỉnh này. Nếu không được thì lùi tiếp đỉnh nữa.

  ➢ Thuật toán này được gọi là backtracking (thuật toán quay lui)

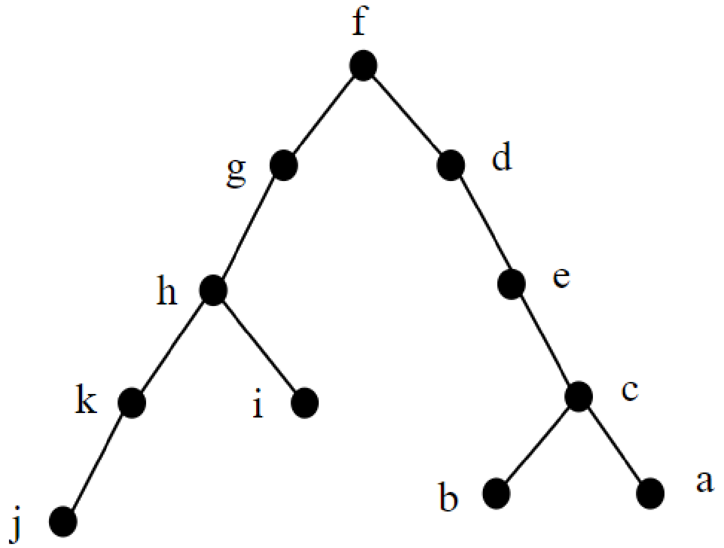
10. Cây khung và các thuật toán

Theo chiều sâu (DFS)



❖ **Ví dụ**: Tìm cây khung cho đồ thị bên bằng thuật toán tìm kiếm chiều sâu.

Lời giải:
Giả sử chọn đỉnh f đầu tiên



10. Cây khung và các thuật toán

- Cây khung nhỏ nhất: trong một đồ thị liên thông, có trọng số là một cây khung có tổng trọng số trên các cạnh của nó là nhỏ nhất.
- Thuật toán tìm cây khung nhỏ nhất:
  - Thuật toán Prim
  - Thuật toán Kruskal

10. Cây khung và các thuật toán

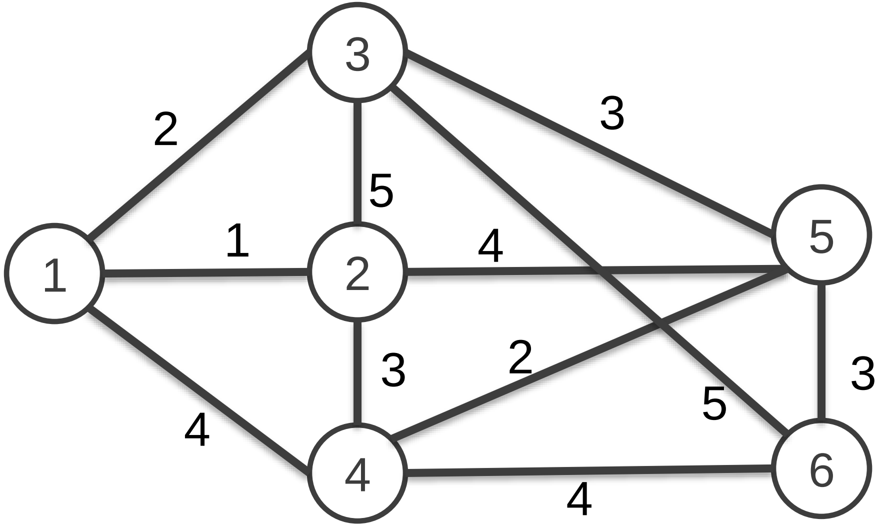
Thuật toán Prim

- Đồ thị G = (V,E) liên thông, có n đỉnh.
  - Bước 1: Chọn một cạnh bất kỳ có trọng số nhỏ nhất, đặt nó vào cây khung.
  - Bước 2: Lần lượt ghép vào cây các cạnh có trọng số nhỏ nhất liên thuộc với một đỉnh của cây và không tạo ra chu trình trong cây.
  - Bước 3: Thuật toán dừng lại khi (n-1) cạnh được ghép vào cây.

10. Cây khung và các thuật toán

Thuật toán Prim

❖ Ví dụ: TÌm cây khung nhỏ nhất theo thuật toán Prim

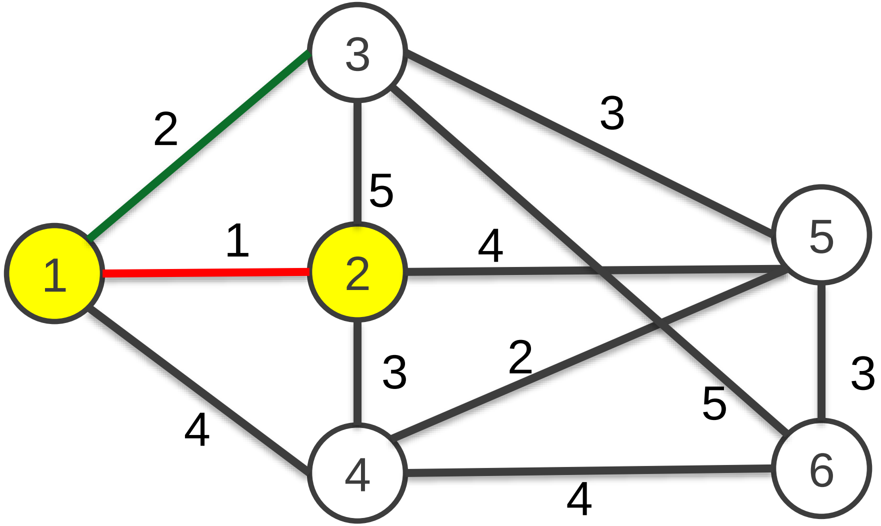


10. Cây khung và các thuật toán

Thuật toán Prim

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Prim

Bước chọn	Cạnh	Trọng số
1	1,2	1

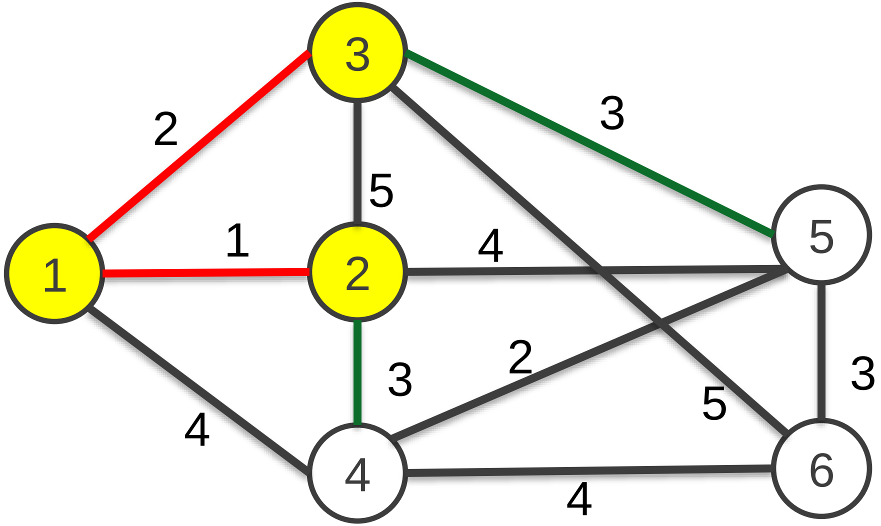


10. Cây khung và các thuật toán

Thuật toán Prim

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Prim

Bước chọn	Cạnh	Trọng số
1	1, 2	1
2	1, 3	2

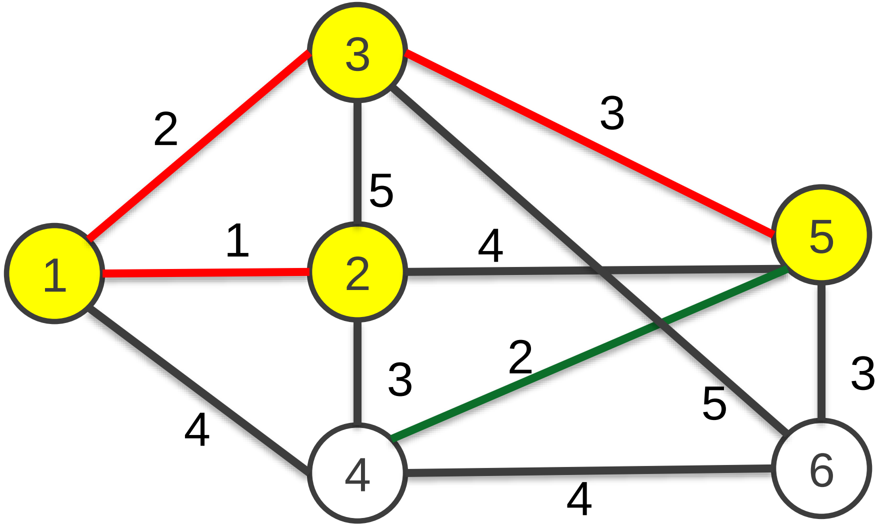


10. Cây khung và các thuật toán

Thuật toán Prim

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Prim

Bước chọn	Cạnh	Trọng số
1	1, 2	1
2	1, 3	2
3	3, 5	3

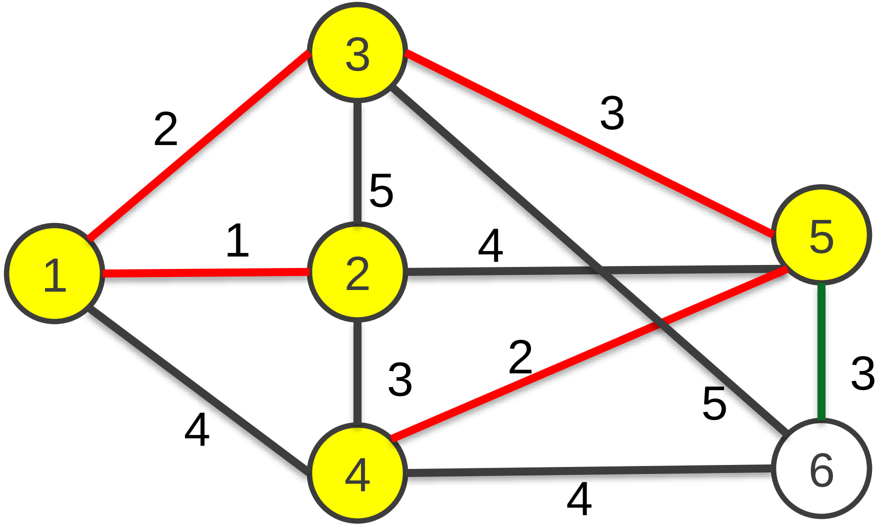


10. Cây khung và các thuật toán

Thuật toán Prim

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Prim

Bước chọn	Cạnh	Trọng số
1	1, 2	1
2	1, 3	2
3	3, 5	3
4	4, 5	2

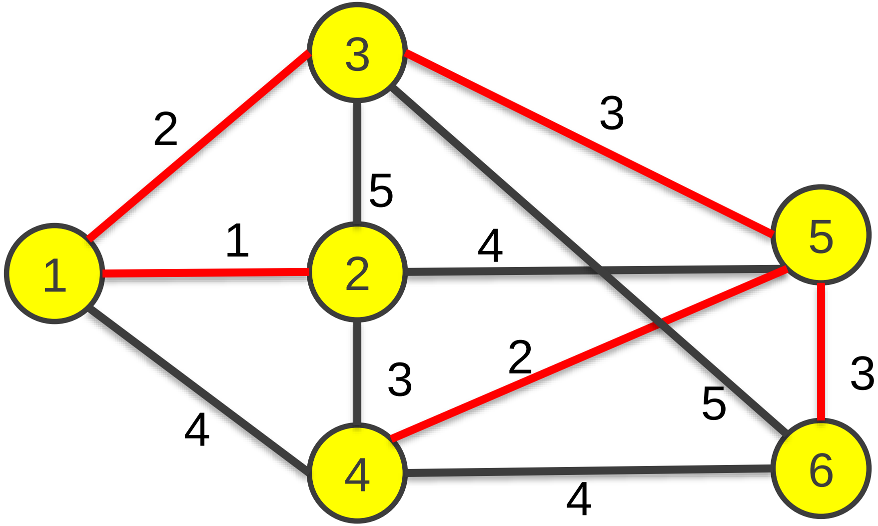


10. Cây khung và các thuật toán

Thuật toán Prim

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Prim

Bước chọn	Cạnh	Trọng số
1	1, 2	1
2	1, 3	2
3	3, 5	3
4	4, 5	2
5	5, 6	3



Tổng trọng số là 1+2+3+2+3 = 11

10. Cây khung và các thuật toán

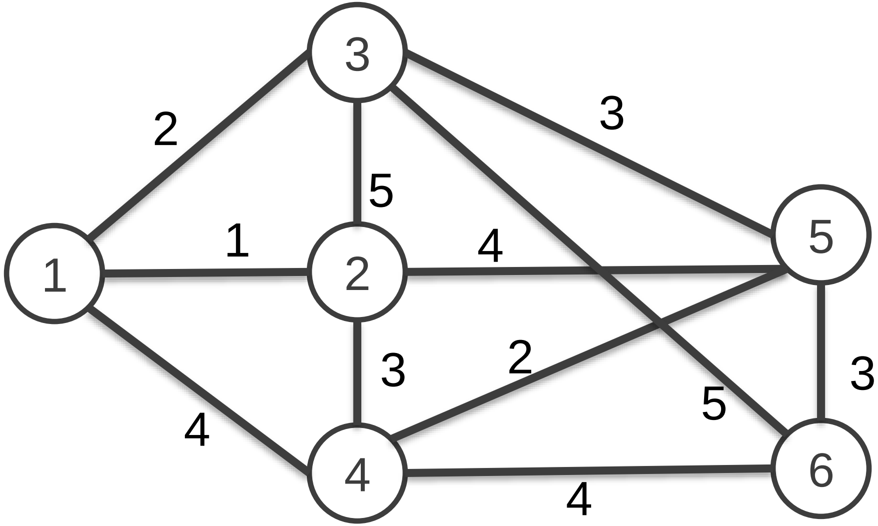
Thuật toán Kruskal

- Đồ thị G = (V,E) liên thông, có n đỉnh.
  - Bước 1: Chọn một cạnh bất kỳ có trọng số nhỏ nhất, đặt nó vào cây khung.
  - Bước 2: Lần lượt ghép vào cây các cạnh có trọng số nhỏ nhất mà không tạo ra chu trình trong cây.
  - Bước 3: Thuật toán dừng lại khi (n-1) cạnh được ghép vào cây.

10. Cây khung và các thuật toán

Thuật toán Kruskal

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Kruskal

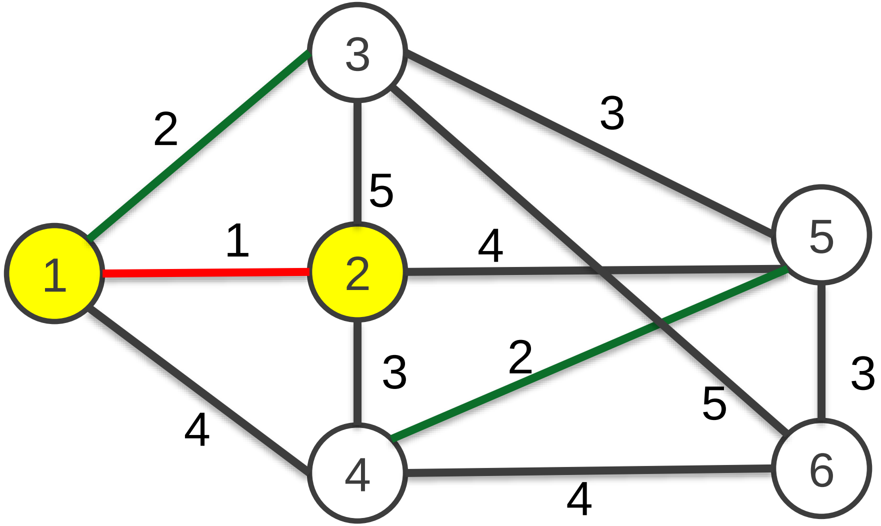


10. Cây khung và các thuật toán

Thuật toán Kruskal

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Kruskal

Bước chọn	Cạnh	Trọng số
1	1,2	1

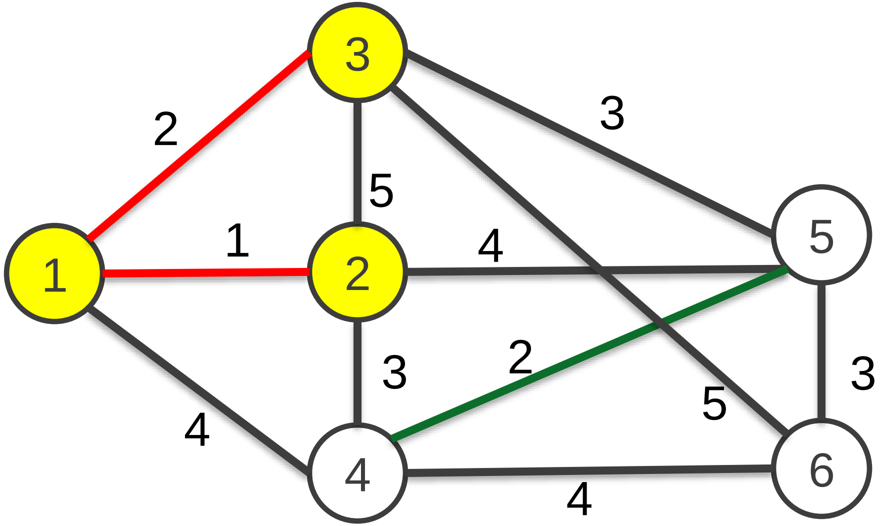


10. Cây khung và các thuật toán

Thuật toán Kruskal

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Kruskal

Bước chọn	Cạnh	Trọng số
1	1,2	1
2	1, 3	2

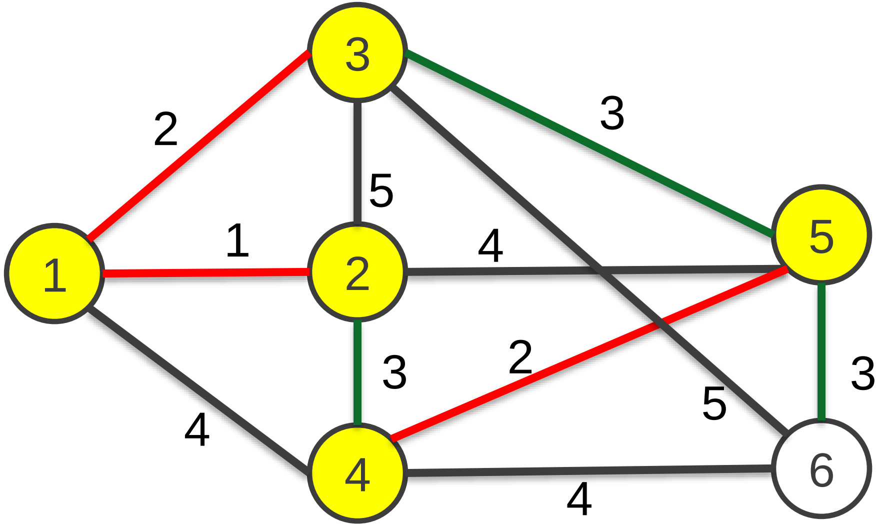


10. Cây khung và các thuật toán

Thuật toán Kruskal

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Kruskal

Bước chọn	Cạnh	Trọng số
1	1,2	1
2	1, 3	2
3	4, 5	2

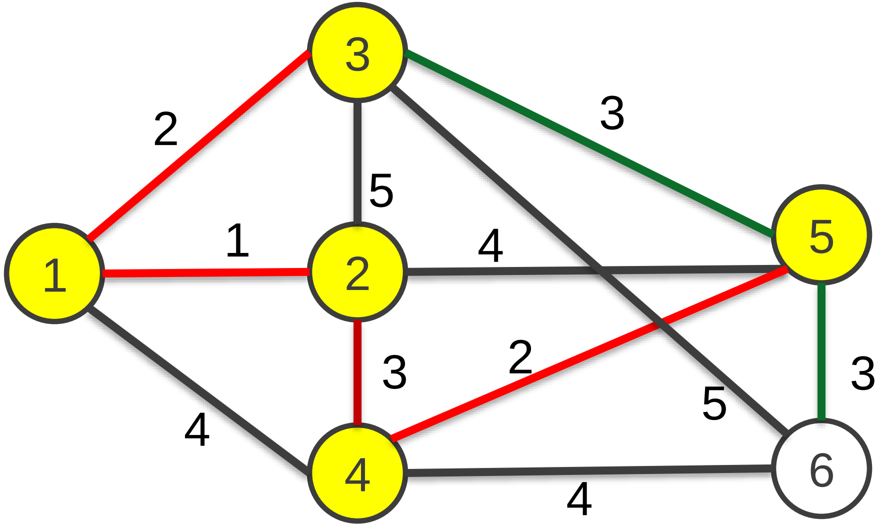


10. Cây khung và các thuật toán

Thuật toán Kruskal

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Kruskal

Bước chọn	Cạnh	Trọng số
1	1,2	1
2	1, 3	2
3	4, 5	2
4	2, 4	3

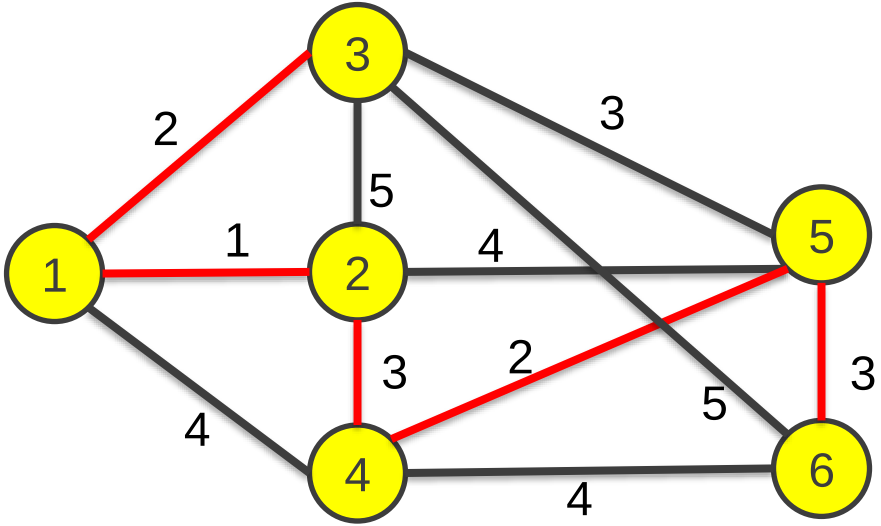


10. Cây khung và các thuật toán

Thuật toán Kruskal

❖ Ví dụ: Tìm cây khung nhỏ nhất theo thuật toán Kruskal

Bước chọn	Cạnh	Trọng số
1	1,2	1
2	1, 3	2
3	4, 5	2
4	2, 4	3
5	5, 6	3



Tổng trọng số là 1+2+2+3+3 = 11

10. Cây khung và các thuật toán

Sự khác nhau giữa thuật toán Prim và Kruskal

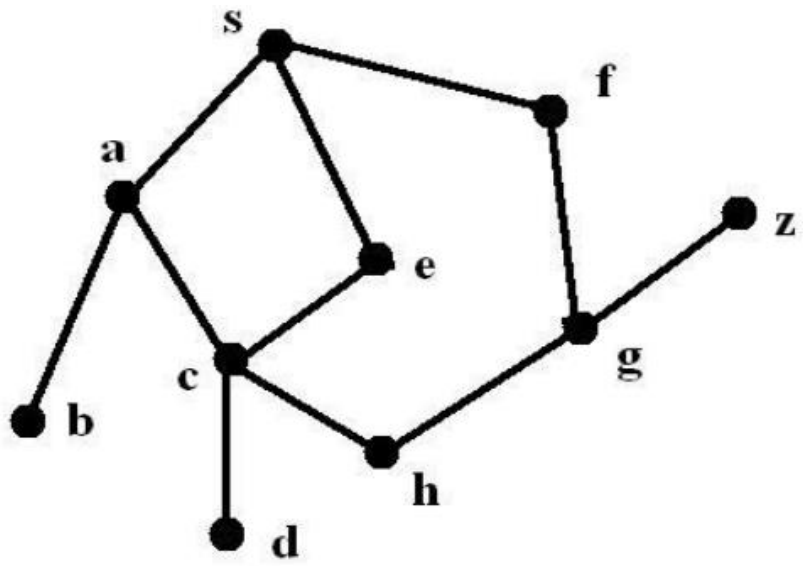
- Với thuật toán Prim các cạnh có trọng số nhỏ nhất phải kề với một đỉnh đã có trong cây. Với Kruskal các cạnh có trọng số nhỏ nhất không cần thiết phải kề với một đỉnh trong cây.
- Giống nhau:
  - Thực hiện bằng cách bổ sung những cạnh kế tiếp có trọng số nhỏ nhất từ những cạnh chưa được sử dụng.
  - Dùng thuật toán tham lam.

10. Cây khung và các thuật toán

Bài tập:

1. Hãy mô tả các bước xét các đỉnh trong quá trình tìm đường đi từ đỉnh s tới z trong đồ thị bên.

   a. Theo chiều rộng
   b. Theo chiều sâu



10. Cây khung và các thuật toán

Bài tập:

2. Hãy tìm cây khung nhỏ nhất của đồ thị G có trọng số trong hình bên bằng thuật toán tìm kiếm:

   a. Thuật toán Prim
   b. Thuật toán Kruskal