CHƯƠNG 6:

LÝ THUYẾT ĐỒ THỊ

Giảng Viên: Ths Nguyễn Thị Quỳnh Như

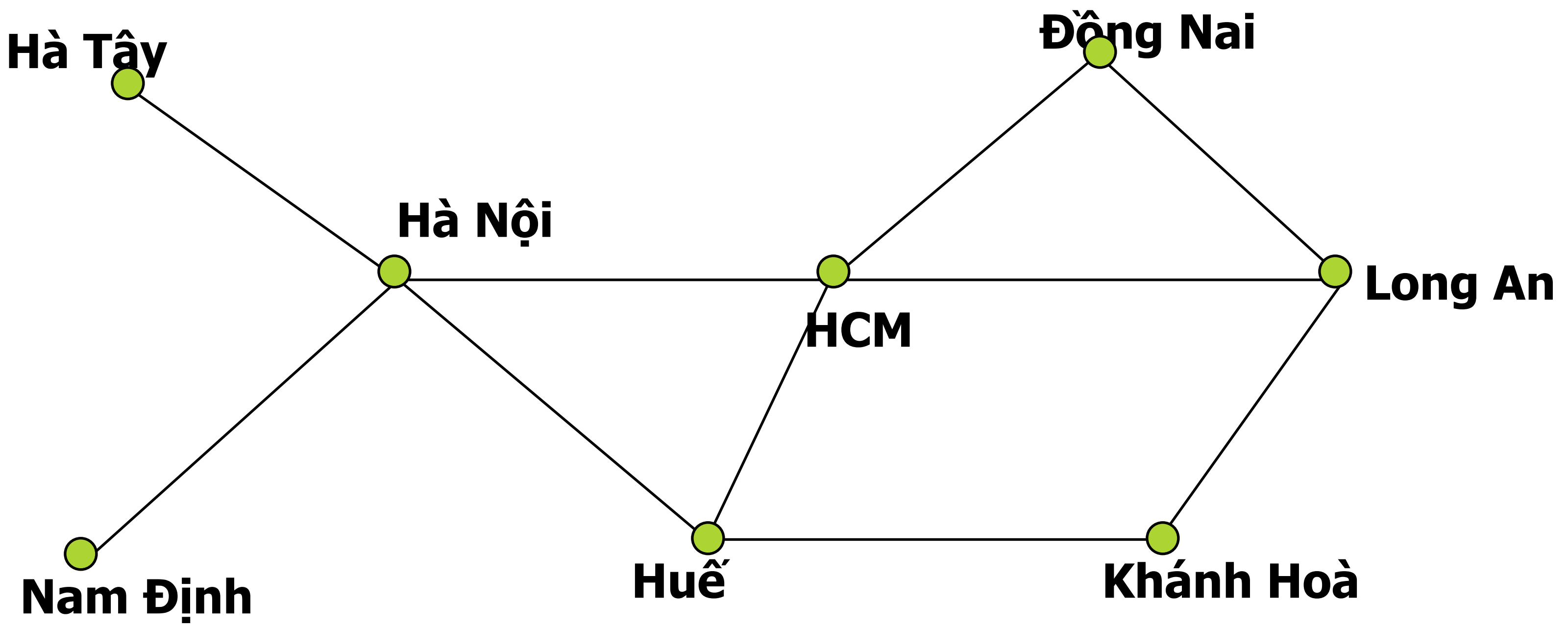
Email: ntqnhu@hou.edu.vn

NỘI DUNG BÀI HỌC

- Định nghĩa đồ thị
- Các thuật ngữ cơ bản
- Đường đi, Chu trình, đồ thị liên thông
- Các dạng đồ thị đặc biệt
- Biểu diễn đồ thị
- Các thuật toán duyệt trên đồ thị

I. Định nghĩa:

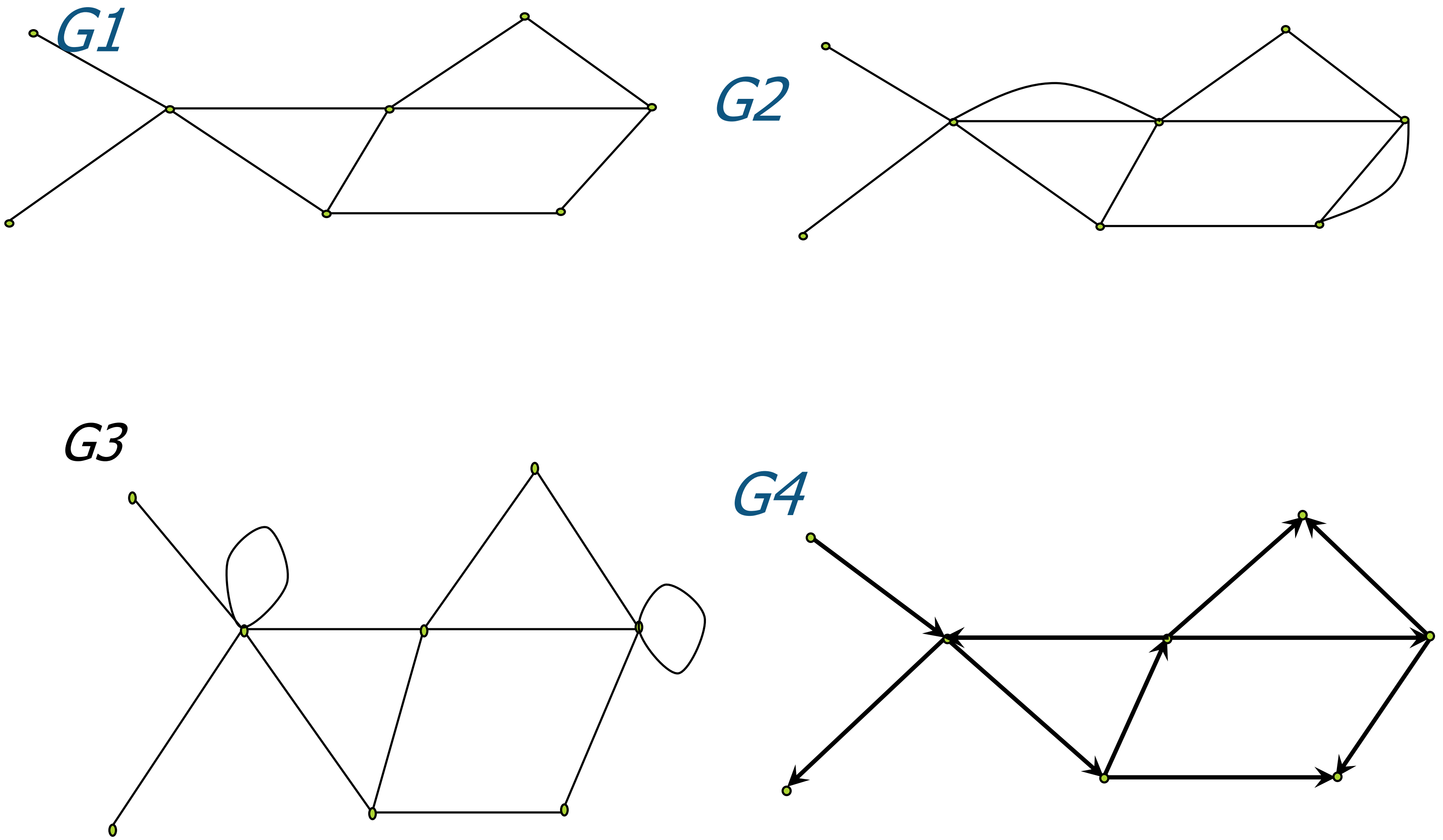
- Định nghĩa: Đồ thị G là một cấu trúc rời rạc bao gồm:
  - tập các đỉnh V và
  - tập các cạnh nối các đỉnh E
- Ví dụ:



Các khái niệm liên quan

- **Đồ thị vô hướng:** Mọi cung thuộc E không phân biệt thứ tự.
- **Đồ thị có hướng:** Mọi cung thuộc E có phân biệt thứ tự
- **Đơn đồ thị:** Giữa 2 đỉnh phân biệt bất kỳ chỉ có 1 cung nối.
- **Đa đồ thị:** Giữa 2 đỉnh phân biệt bất kỳ có thể có nhiều hơn 1 cung nối
- **Giả đồ thị:** Tồn tại ít nhất một khuyên trong đồ thị (Khuyên là cung mà đỉnh đầu và cuối trùng nhau)

Ví dụ

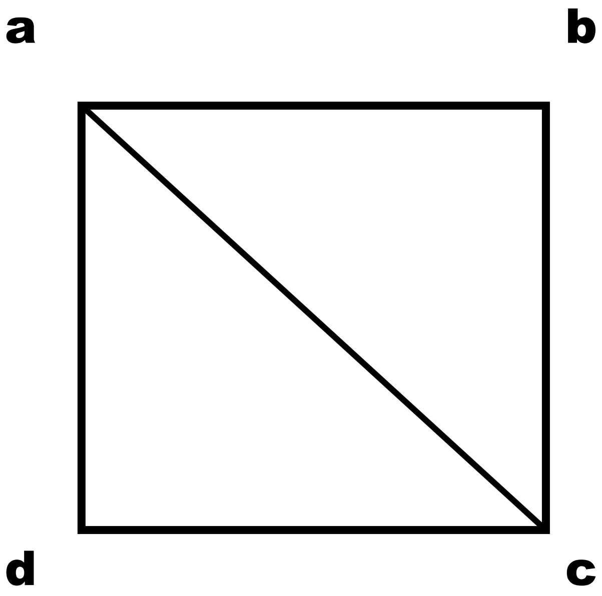


II. Các thuật ngữ cơ bản

- Hai đỉnh u và v của đồ thị vô hướng G được gọi là **kề nhau** nếu (u,v) là một cạnh của đồ thị G.
- Nếu e=(u,v) là một cạnh của đồ thị thì ta nói cạnh e **liên thuộc** với hai đỉnh u và v.
- Còn đỉnh u và đỉnh v được gọi là **đỉnh đầu** và **đỉnh cuối** của cạnh (u,v).
- Bậc của một đỉnh trong đồ thị vô hướng là **số cạnh liên thuộc với đỉnh đó**, riêng khuyên tại một đỉnh được tính hai lần cho bậc của đỉnh đó.
  - Ký hiệu bậc của đỉnh v là **deg(v)**

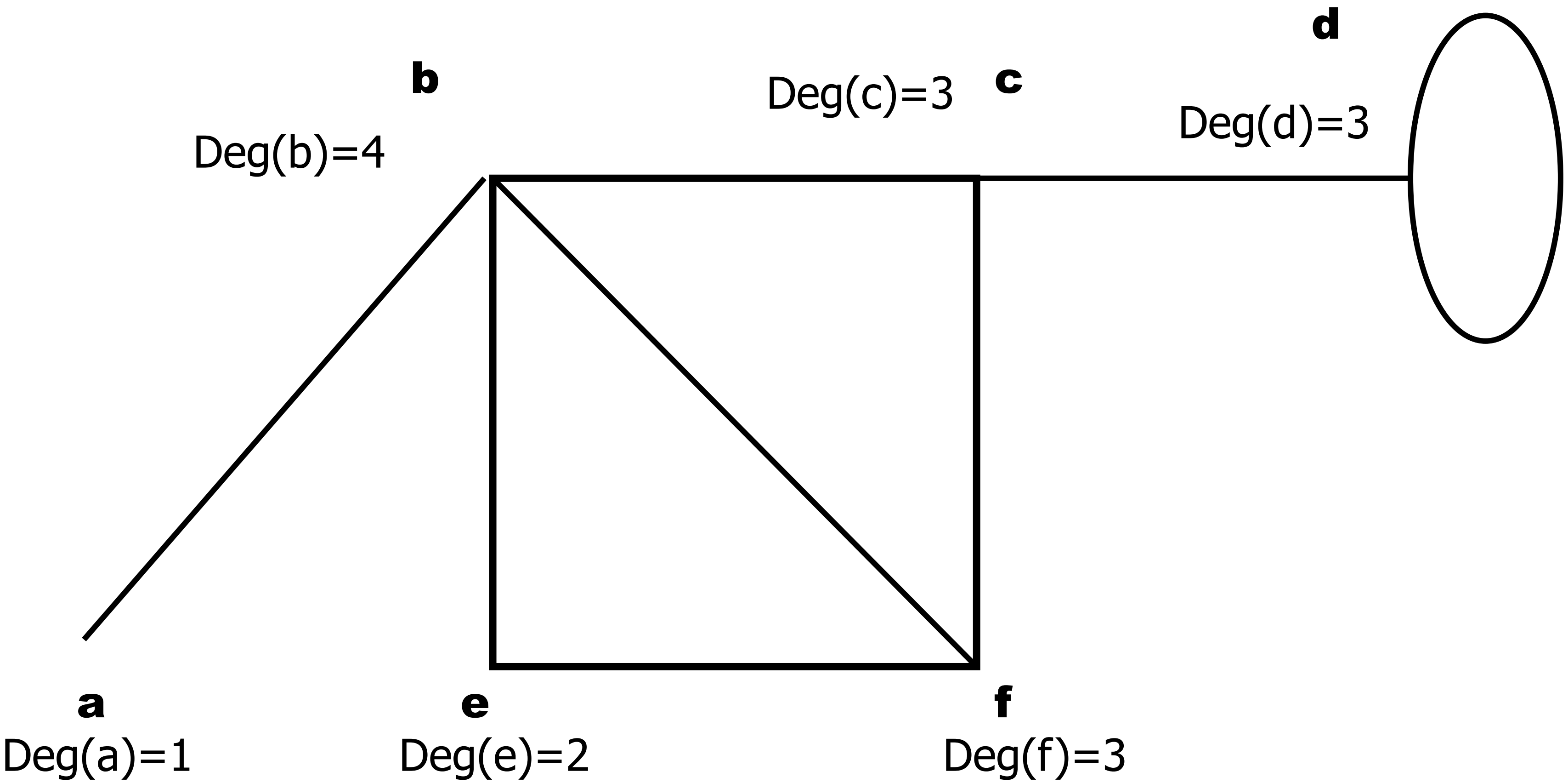
Ví dụ

- Cho đồ thị sau, liệt kê các cặp đỉnh kề nhau



Ví dụ

- Cho đồ thị sau, tìm các bậc tương ứng của các đỉnh của đồ thị



Định lý

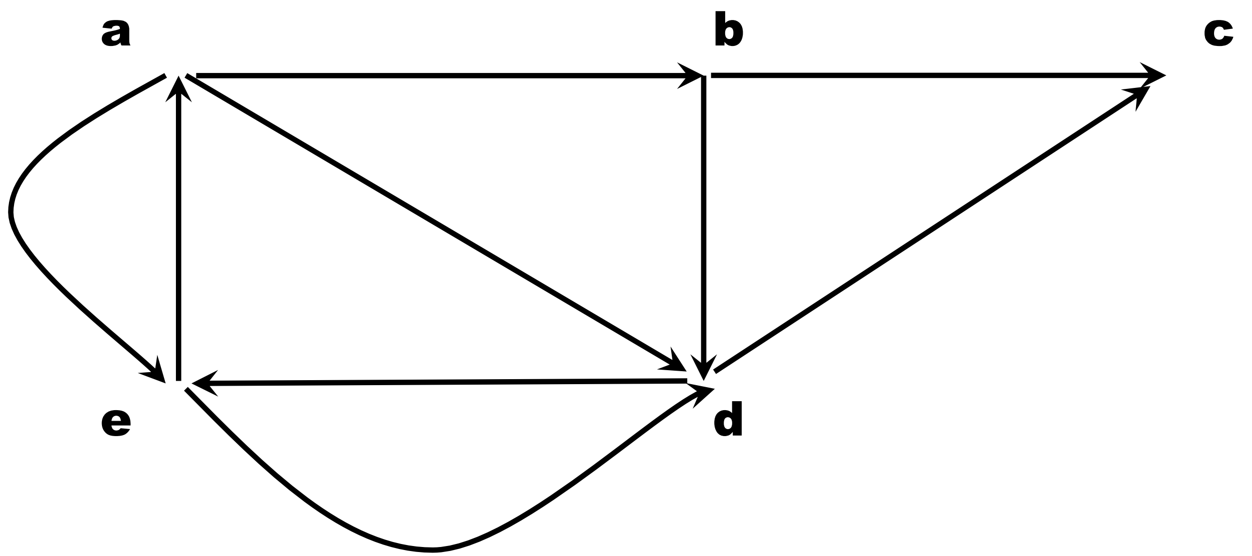
- **Định lý 2.1:** Giả sử G=(V,E) là đồ thị vô hướng với m cạnh. Khi đó tổng số bậc của tất cả các đỉnh bằng hai lần số cạnh

$$\boldsymbol{\Sigma \deg(v)=2m}$$

- **Hệ quả 2.1:** Trong đồ thị vô hướng, số đỉnh bậc lẻ (có bậc là số lẻ) là một số chẵn
- **Chú ý:**
  - Đỉnh có số bậc bằng 0 được gọi là đỉnh cô lập
  - Đỉnh có bậc bằng 1 được gọi là đỉnh treo

Bán bậc ra-Bán bậc vào

- Ta gọi bán bậc ra của đỉnh v trong đồ thị có hướng là số cung của đồ thị đi ra khỏi đỉnh v, ký hiệu deg+(v). Bán bậc vào của đỉnh v trong đồ thị có hướng là số cung của đồ thị đi vào đỉnh v, ký hiệu deg-(v).



Định lý 2.2

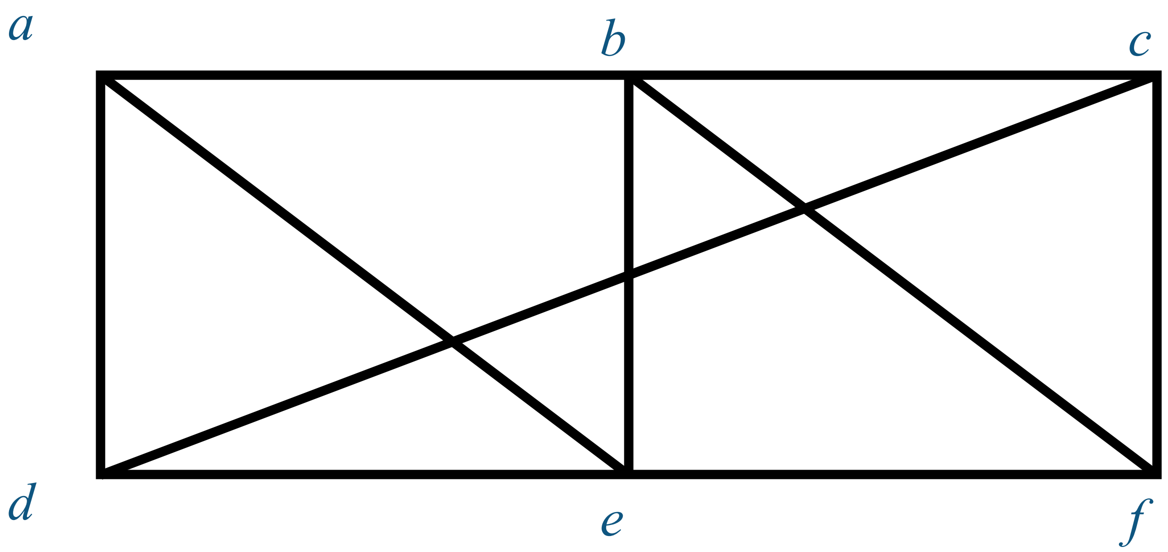
- Giả sử G=(V,E) là đồ thị có hướng. Khi đó với m là số cung của đồ thị thì ta có công thức sau:

$$2m = \sum_{v \in V} \deg^{+}(v) + \sum_{v \in V} \deg^{-}(v)$$

III. Đường đi. Chu trình. Đồ thị liên thông

- Định nghĩa: Đường đi có độ dài n từ đỉnh u đến đỉnh v, trong đó n là số nguyên dương, trên đồ thị vô hướng G=(V,E) là dãy liên tiếp được ký hiệu:
- x0,x1,x2,…..,xn-1,xn
- hoặc (x0,x1), (x1,x2), …., (xn-1,xn)
- Trong đó:
  - (xi,xi+1) là một cạnh của đồ thị với i=0 ÷ n-1
  - Đỉnh u=x0 được gọi là đỉnh đầu, đỉnh v=xn được gọi là đỉnh cuối của đường đi.
  - Đường đi có đỉnh đầu trùng với đỉnh cuối (u≡v) được gọi là chu trình
  - Đường đi hay chu trình được gọi là đơn nếu như không có cạnh nào được lặp lại trong đường đi, chu trình đó.

Ví dụ về đường đi



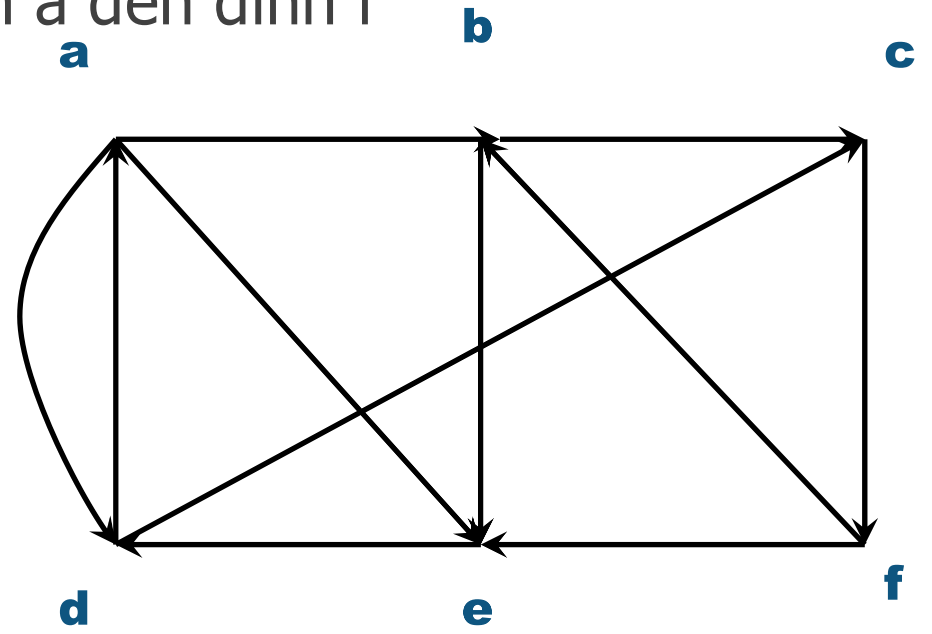
- *Trả lời các câu hỏi sau:*
  - a,d,c,f,e có phải là đường đi hay không? độ dài bằng bao nhiêu?
  - d,e,c,a có phải là đường đi hay không? độ dài bằng bao nhiêu?
  - b,c,f,b được gọi là gì?
  - a,b,e,d,a,b là đường đi đơn hay không?

Đường đi trong đồ thị có hướng

- Đường đi có độ dài n từ đỉnh u đến đỉnh v, n nguyên dương, trong đồ thị có hướng G=(V,E) là dãy
  - x0,x1,x2,.....,xn-1,xn
  - hoặc (x0,x1),(x1,x2),....,(xn-1,xn)
- **Trong đó:**
  - (xi,xi+1) là một cung của đồ thị i=0÷n-1
  - Với đỉnh u=x0 được gọi là đỉnh đầu, đỉnh v=xn được gọi là đỉnh cuối của đường đi.

Ví dụ

- Liệt kê đường đi từ đỉnh a đến đỉnh f

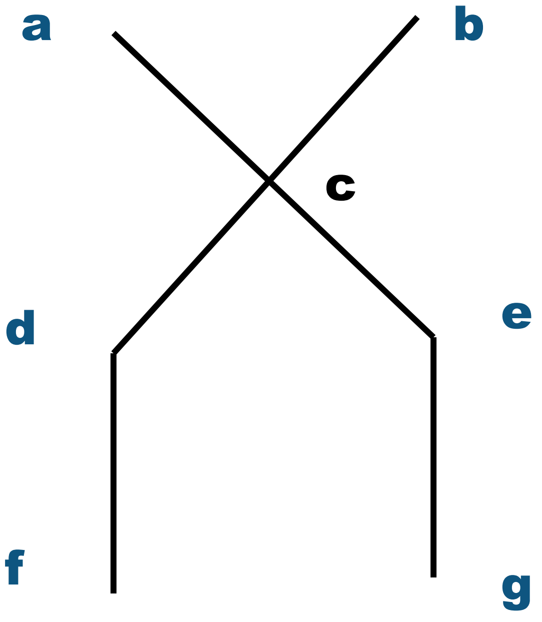


- Có các đường đi sau:
  - a-> b -> c-> f
  - a->b->e->d->c->f
  - a->d->c->f
  - a->e->d->c->f

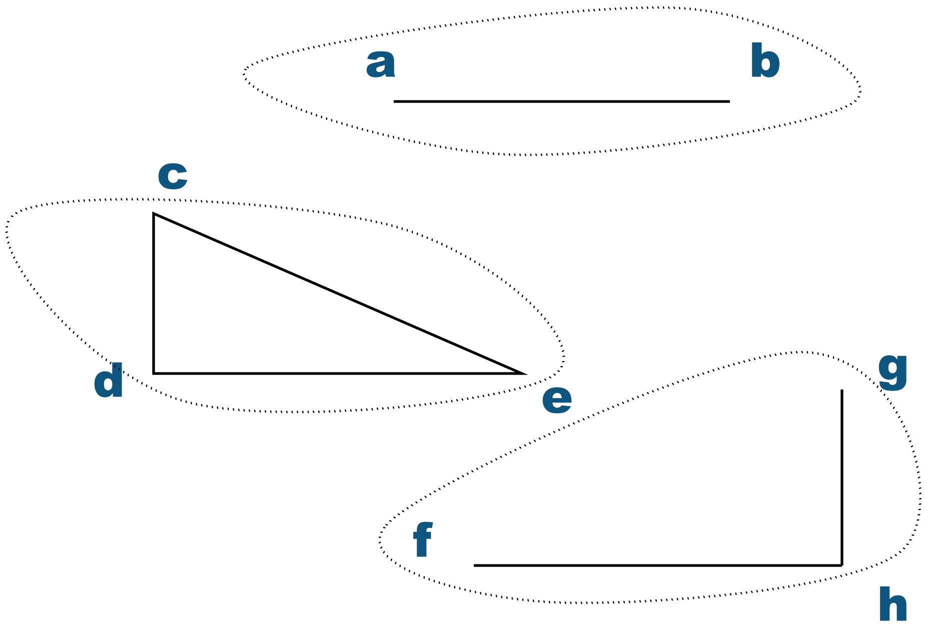
Đồ thị liên thông

- Đồ thị vô hướng G=(V,E) được gọi là liên thông nếu luôn tìm được đường đi giữa hai đỉnh bất kỳ phân biệt của đồ thị.
- Nếu G không liên thông thì chắc chắn G là tập hợp của hai hay nhiều đồ thị con liên thông

Ví dụ



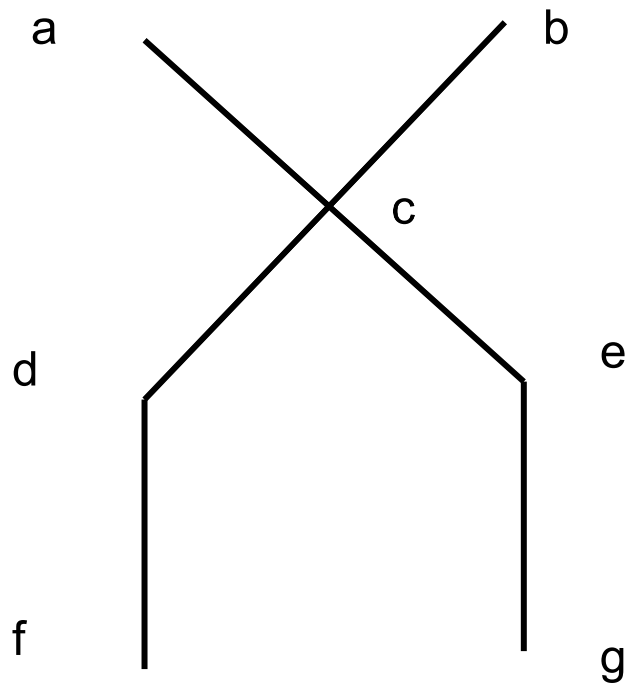
G1



G2

Định lý 3.1

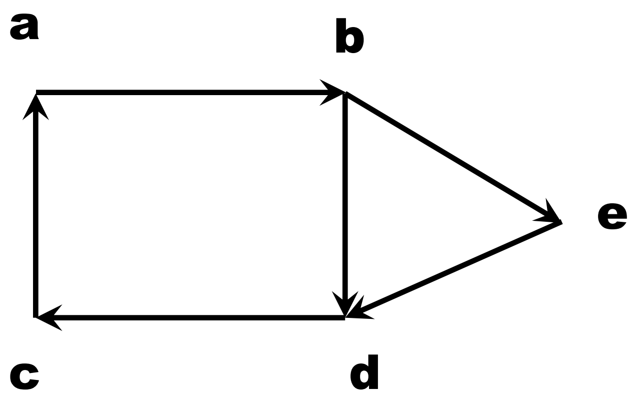
- Giữa mọi cặp đỉnh phân biệt của đồ thị vô hướng liên thông luôn có đường đi đơn (là đường đi qua mỗi cạnh đúng một lần)



G

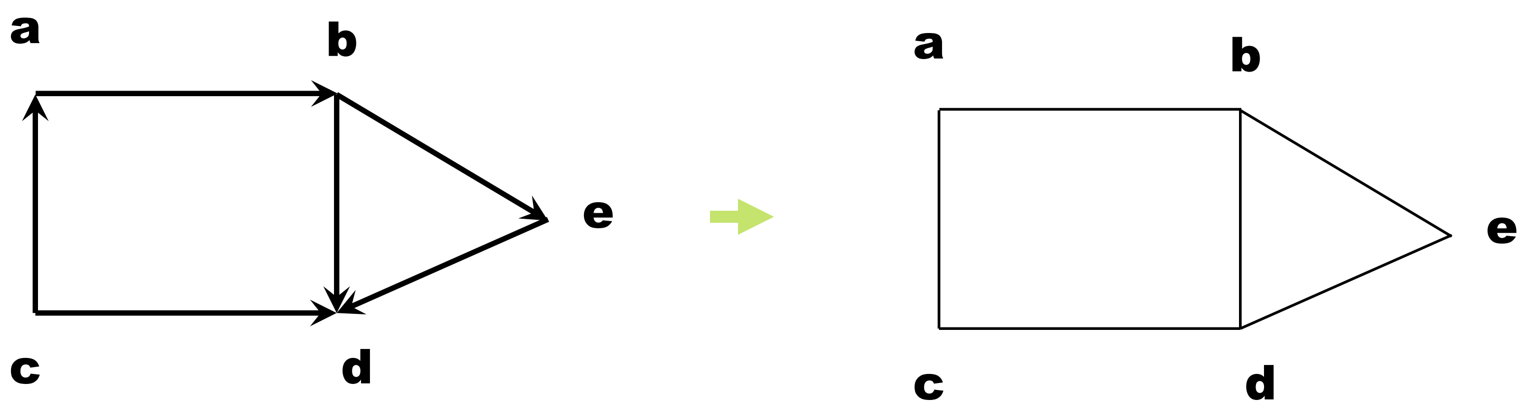
Đồ thị có hướng liên thông mạnh

- Đồ thị có hướng G=(V,A) được gọi là liên thông mạnh nếu luôn tìm được đường đi giữa hai đỉnh bất kỳ của đồ thị
- **Ví dụ:**



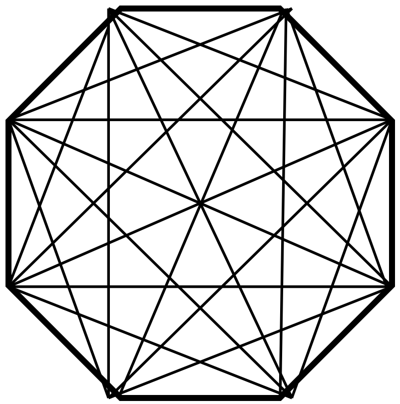
Đồ thị có hướng liên thông yếu

- Đồ thị có hướng G=(V,A) được gọi là liên thông yếu nếu đồ thị vô hướng tương ứng với nó là đồ thị liên thông
- Ví dụ:



IV. Các dạng đồ thị đặc biệt

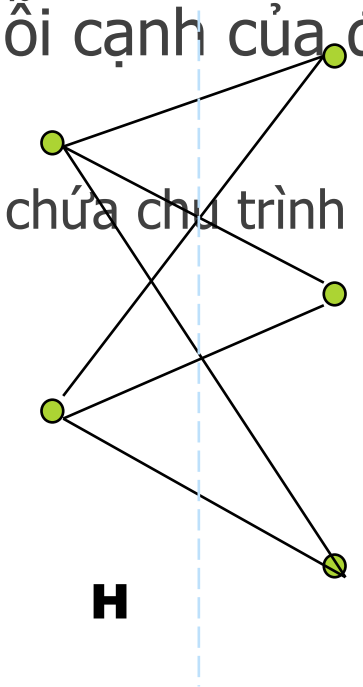
- **Đồ thị đầy đủ:** Đồ thị đầy đủ n đỉnh, ký hiệu Kn, là đơn đồ thị vô hướng mà giữa hai đỉnh bất kỳ của đồ thị luôn có cạnh nối
- **Chú ý:** đồ thị đầy đủ Kn có n(n-1)/2 cạnh



G

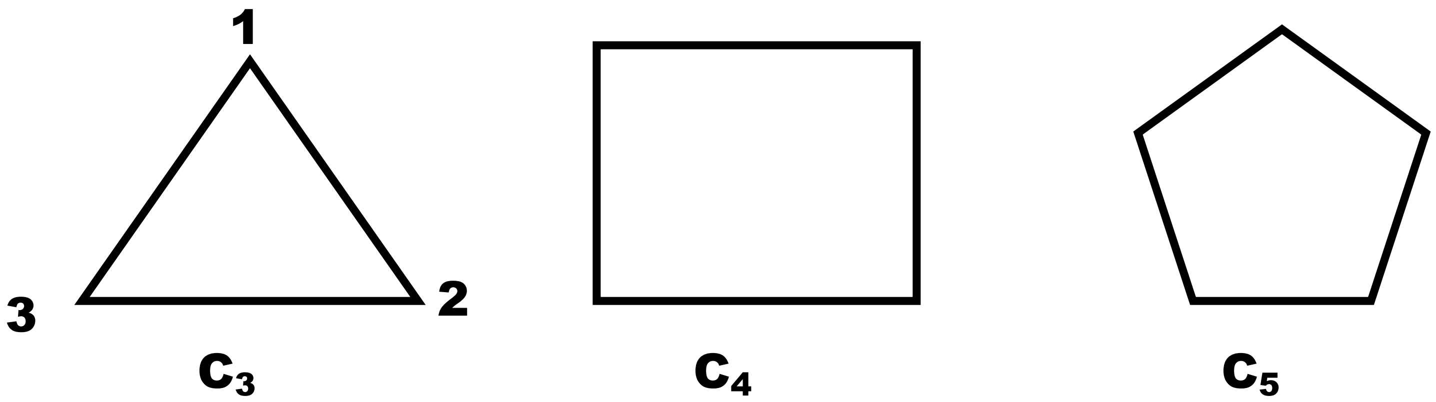
IV. Các dạng đồ thị đặc biệt

- Đồ thị hai phía: Đơn đồ thị G=(V,E) được gọi là hai phía nếu tập đỉnh V của đồ thị có thể phân hoạch thành hai tập X và Y sao cho mỗi cạnh của đồ thị chỉ nối một đỉnh nào đó trong X với đỉnh nào đó trong Y
  - Định lý 5.1: Đơn đồ thị là đồ thị hai phía khi và chỉ khi nó không chứa chu trình có độ dài lẻ



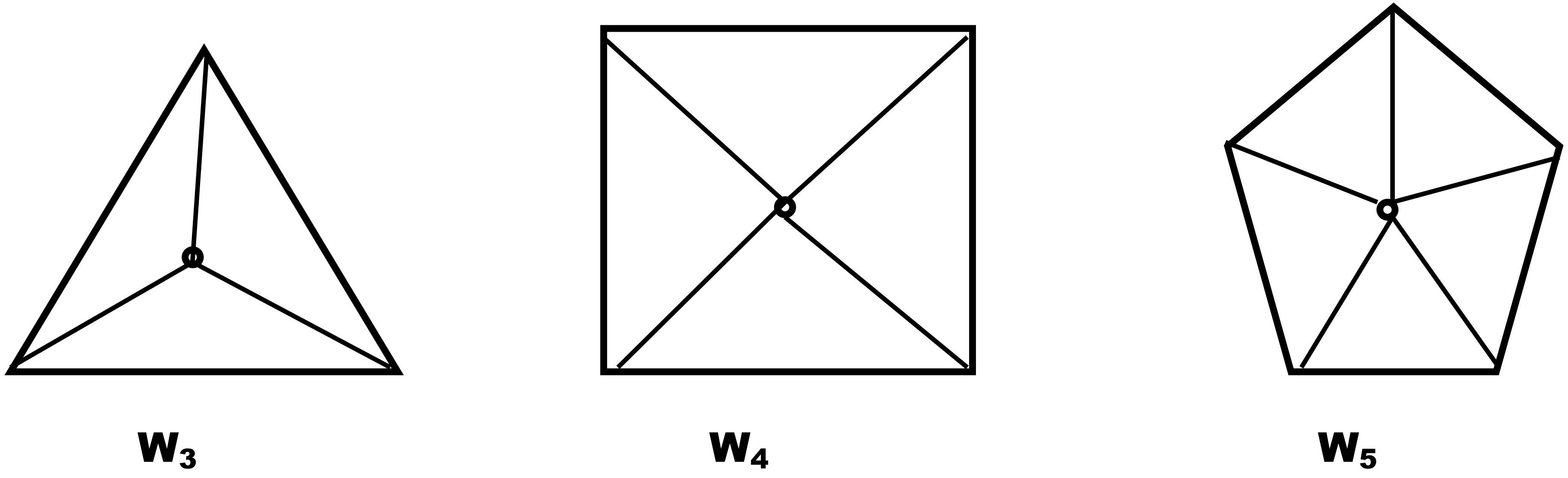
IV. Các dạng đồ thị đặc biệt (2)

- **Đồ thị vòng:** Đồ thị vòng Cn, n>=3, gồm n đỉnh v1, v2,…,vn và các cạnh liên tiếp nhau (v1,v2), (v2,v3),…, (vn-1,vn), (vn,v1)
- **Ví dụ:**



IV. Các dạng đồ thị đặc biệt (3)

- **Đồ thị bánh xe:** Đồ thị Wn được gọi là đồ thị bánh xe nếu Wn thu được từ đồ thị vòng Cn bằng cách bổ sung vào một đỉnh giữa nối tất cả các đỉnh của đồ thị vòng Cn



V. Biểu diễn đồ thị

- Biểu diễn bằng ma trận kề
- Biểu diễn bằng danh sách cạnh
- Biểu diễn bằng danh sách kề

1. Ma trận kề

- Cho G=(V,E) là một đơn đồ thị, với

  V={1,2,3,…,n} là tập tất cả các đỉnh ⇔ |V|=n

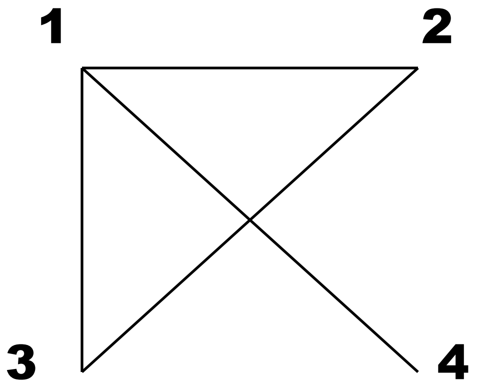
  E={e_1,e_2,…,e_m} là tập các cạnh ⇔ |E|=m

- Ma trận kề A của đồ thị G(A_G) là ma trận cấp nxn có phần tử tại hàng i, cột j bằng 1 nếu đỉnh i và j là liền kề nhau, và giá trị bằng 0 nếu hai đỉnh i và j không được nối với nhau

$$\Leftrightarrow A_G = [a_{ij}]_{i,j=1\ldots n}$$

Ví dụ

- Cho đồ thị theo hình sau.
  - Tìm ma trận kề biểu diễn đồ thị



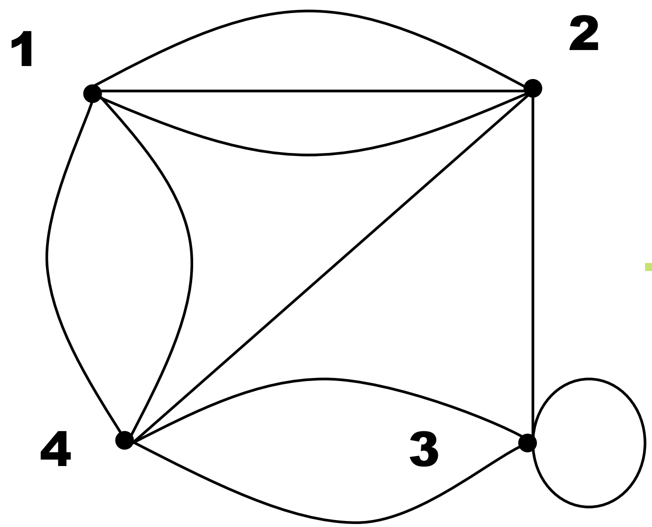
	1	2	3	4
1	0	1	1	1
2	1	0	1	0
3	1	1	0	0
4	1	0	0	0

Các tính chất của ma trận kề

- Ma trận kề của đồ thị vô hướng là ma trận đối xứng ⇔ a[i,j]=a[j,i]
- Số phần tử trên dòng i(hoặc cột j) của ma trận kề mà a[i,j]=1 chính bằng bậc của đỉnh i(hay đỉnh j)
- Ma trận kề cũng có thể dùng để biểu diễn đồ thị vô hướng có khuyên hoặc đa đồ thị. Khuyên tại đỉnh ai được biểu diễn là 1 tại a[i,i]. Còn trong đa đồ thị thì vị trí aij là số cạnh nối hai đỉnh i và j

Ví dụ

- Tìm ma trận kề của đồ thị sau



	1	2	3	4
1	0	3	0	2
2	3	0	1	1
3	0	1	1	2
4	2	1	2	0

Ma trận trọng số

- Khi trên mỗi cạnh e của đồ thị được gán một giá trị c(e) gọi là trọng số của cạnh e ➔ Đồ thị được gọi là đồ thị có trọng số => Khi đó đồ thị được biểu diễn dưới dạng ma trận trọng số C

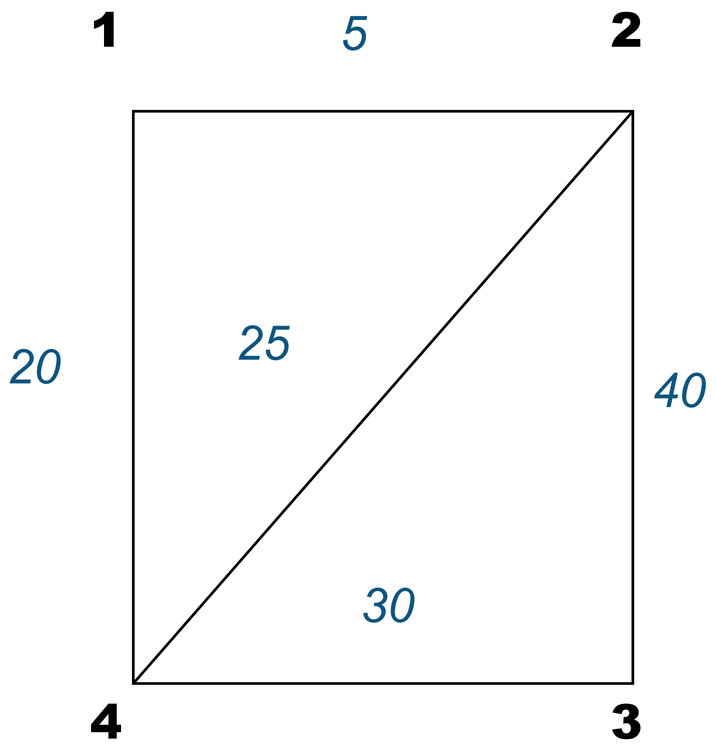
$$C = [cij] \; i, j=1\text{-}n$$

- Trong đó:

$$cij = \begin{cases} c(i,j) \text{ nếu } (i,j) \text{ là một cạnh(trọng số cạnh } i,j) \\ 0 \text{ hoặc } \infty \text{ nếu } (i,j) \text{ không là cạnh} \end{cases}$$

Ví dụ

- Tìm ma trận trọng số của đồ thị sau

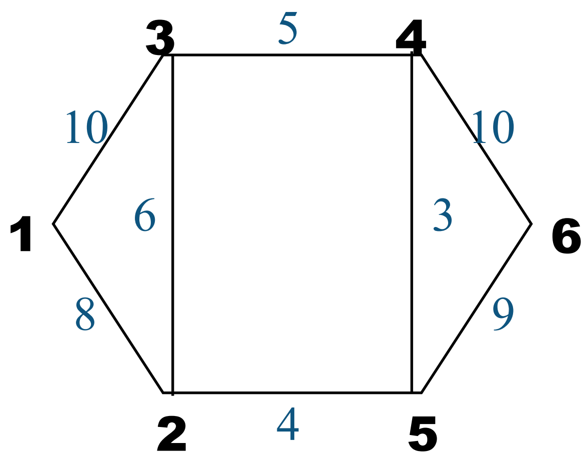


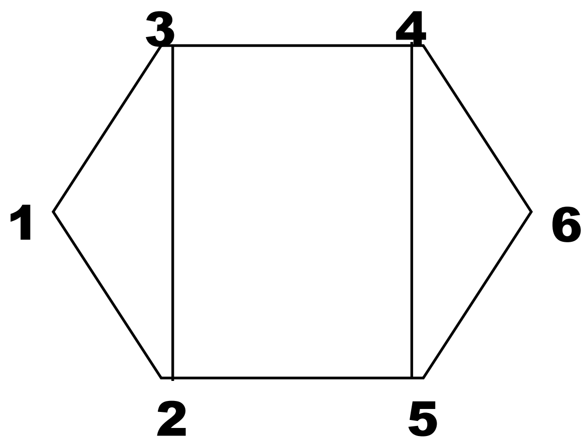
	1	2	3	4
1	0	5	0	20
2	5	0	40	25
3	0	40	0	30
4	20	25	30	0

2. Danh sách cạnh

- Trong trường hợp đồ thị thưa (m<6n) người ta thường dùng cách biểu diễn đồ thị dưới dạng danh sách cạnh
- Sử dụng danh sách chỉ lưu trữ tất cả các cạnh hiện có của đồ thị. Một cạnh e của đồ thị được lưu tương ứng với ba thành phần: Đầu, Cuối, Trọng số (nếu có)

Ví dụ





	D	C	TS
1	1	2	8
2	1	3	10
3	2	3	6
4	2	5	4
5	3	4	5
6	4	5	3
7	4	6	10
8	5	6	9

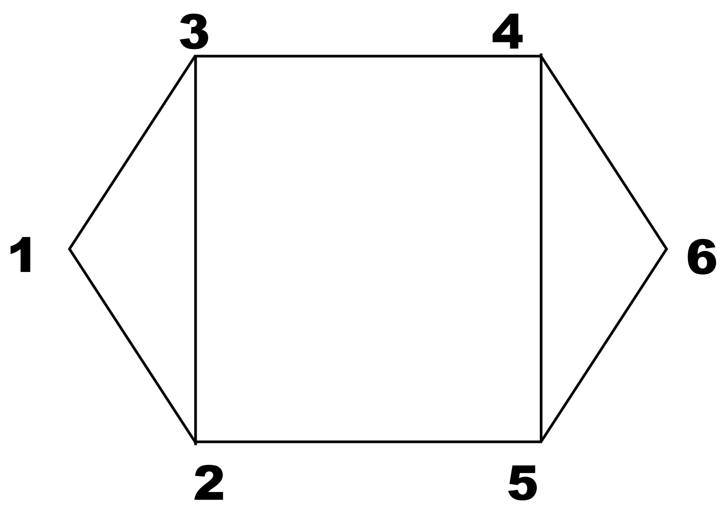
	D	C
1	1	2
2	1	3
3	2	3
4	2	5
5	3	4
6	4	5
7	4	6
8	5	6

3. Danh sách kề

- Sử dụng danh sách, với mỗi đỉnh v của đồ thị lưu trữ một danh sách các đỉnh liền kê với đỉnh v đó
- Đồ thị có bao nhiêu đỉnh có bấy nhiêu danh sách kề biểu diễn các đỉnh kề tương ứng
- Có thể biểu diễn danh sách kề theo dạng mảng hoặc danh sách móc nối.

Ví dụ

- Biểu diễn danh sách kề dưới dạng mảng

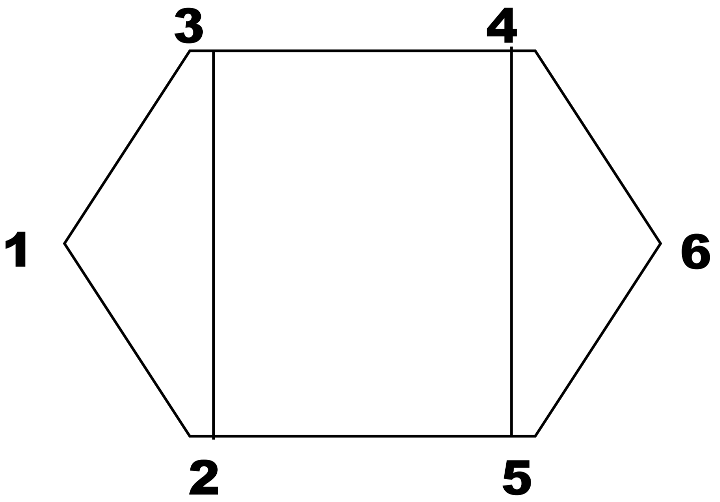


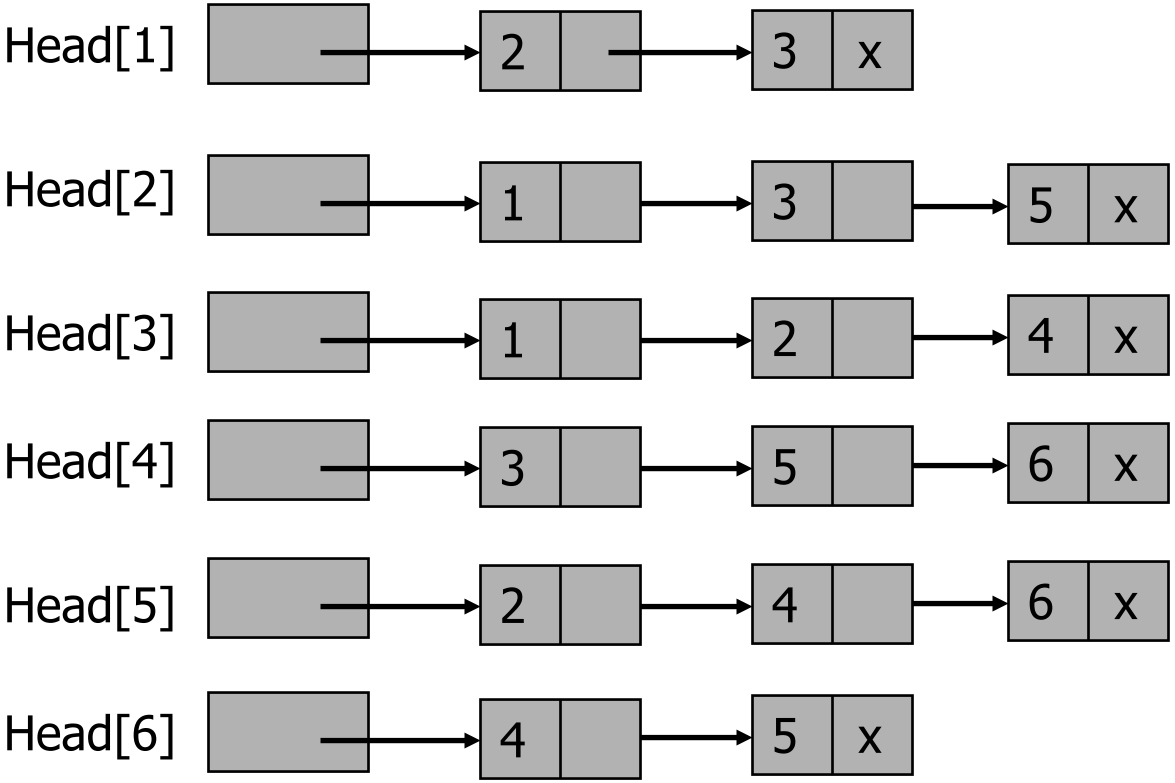
1	2	3	
2	1	3	5
3	1	2	4
4	3	5	6
5	2	4	6
6	4	5	

Biểu diễn danh sach kề

- Biểu diễn danh sách kề dưới dạng danh sách móc nối
  - N hàng của ma trận được thay bằng n danh sách móc nối.
  - Mỗi đỉnh của G có 1 danh sách kề tương ứng.
    - Mỗi danh sách có 1 nút đầu danh sách Head[i] ứng với đỉnh thứ i.
    - Mỗi nút trên danh sách có 2 trường:
      - vertext (chứa chỉ số của đỉnh lân cận với đỉnh i)
      - link trỏ tới nút tiếp theo trong danh sách

Ví dụ





Head[i] là con trỏ đầu danh sách trỏ đến danh sách kề của đỉnh thứ i

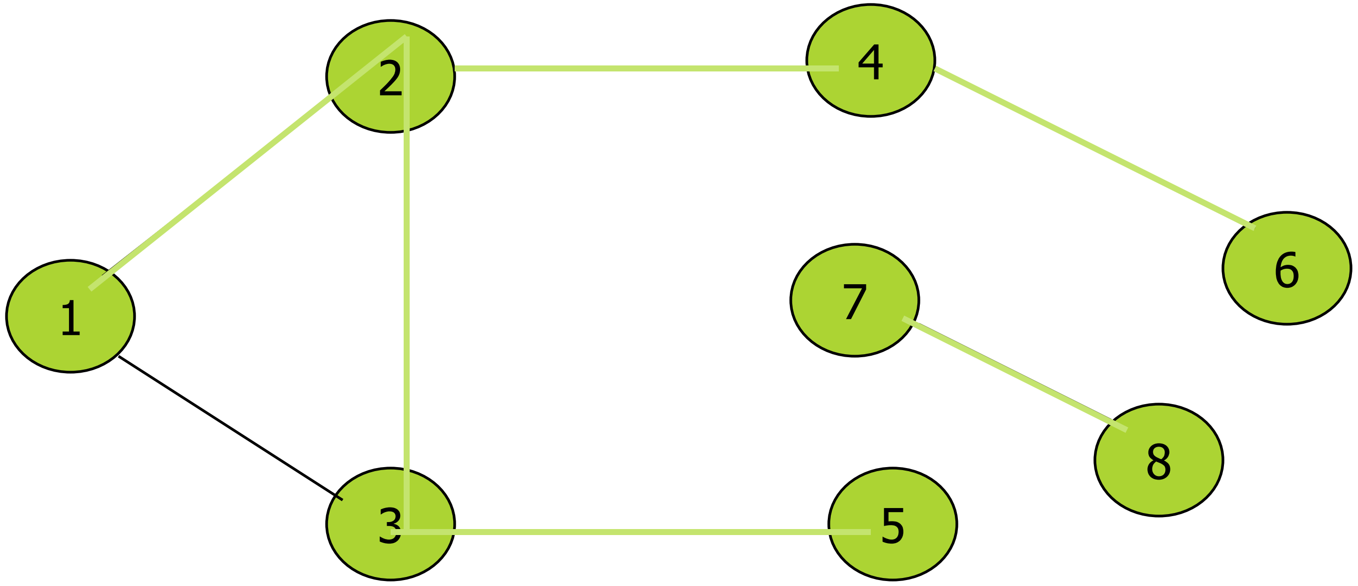
Một số ứng dụng cơ bản trên đồ thị

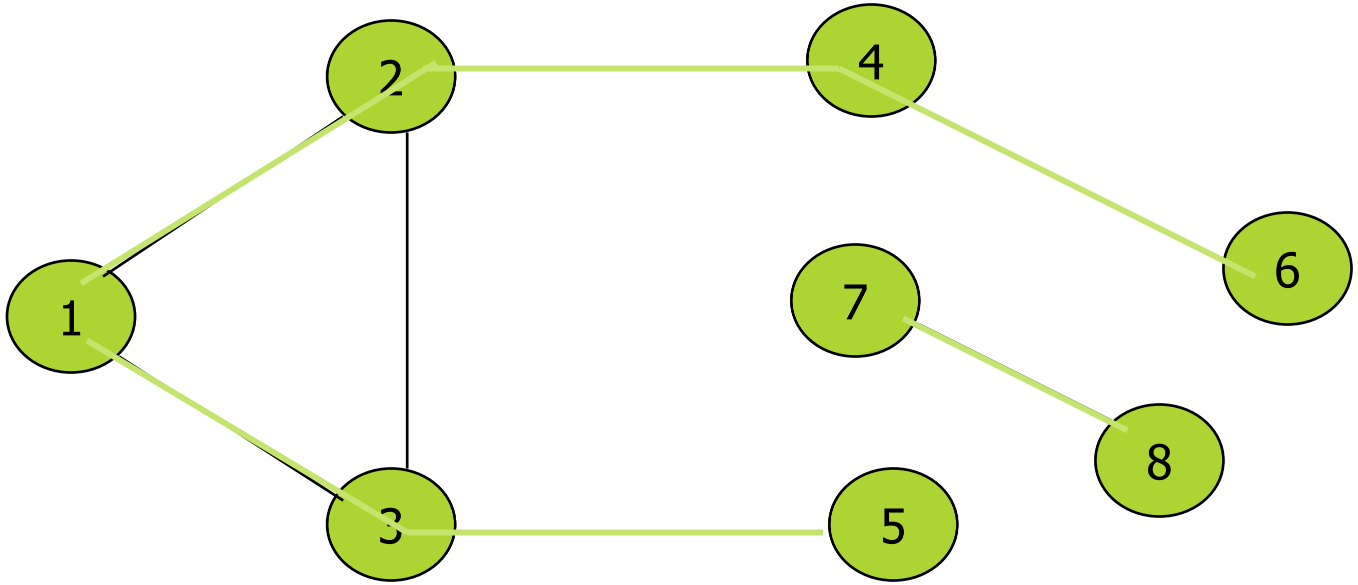
- Xác định có đường đi giữa 2 đỉnh
- Xác định các thành phần liên thông trên đồ thị
- Xác định chu trình
- Duyệt đồ thị
  - Duyệt sâu
  - Duyệt rộng

Các thuật toán duyệt trên đồ thị

- Duyệt sâu:
  - Đỉnh xuất phát u được thăm, tiếp tục thăm đỉnh v với v là một đỉnh chưa thăm kề với u.
  - Khi một đỉnh s nếu được "với tới" mà mọi đỉnh kề với nó đều đã được thăm rồi thì ta quay ngược lên trên đỉnh cuối cùng vừa được thăm, tìm đỉnh w kề với đỉnh đó mà w chưa được thăm.
  - Sử dụng kỹ thuật đánh dấu để tránh thăm 2 lần 1 đỉnh.
- Duyệt rộng:
  - Xuất phát từ một đỉnh u, tìm thăm tất cả các đỉnh v kề với u.
  - Sau khi thăm hết tất cả các đỉnh v kề với u, ta quay lại tìm thăm tất cả các đỉnh w kề với v mà mà đỉnh v gần với u nhất.

Duyệt sâu- Duyệt rộng





Thuật toán duyệt sâu

```
void duyetsau()
{//đánh dấu chưa duyệt tất cả các đỉnh
   for (v =1; v <=n; v++)
        mark[v-1]=unvisited;
   //duyệt theo chiều sâu từ đỉnh thứ 1
   for (v = 1; v<=n; v++)
    if (mark[v-1] == unvisited)
    dfs(v); //duyệt theo chiều sâu đỉnh v
}
```

Thuật toán duyệt sâu

```
void dfs(vertex v) // v ∈[1..n]
{
vertex w;
mark[v-1]=visited;
for (mỗi đỉnh w là đỉnh kề với v)
  if (mark[w-1] == unvisited)
   dfs(w);
}
```

Thuật toán duyệt rộng

```
void duyetrong()
{//đánh dấu chưa duyệt tất cả các đỉnh
   for (v = 1; v<= n; v++)
   mark[v-1] = unvisited;
   //n là số đỉnh của đồ thị
   //duyệt theo chiều rộng từ đỉnh đánh số 1
   for (v = 1; v<=n; v++)
      if (mark[v-1] == unvisited)
      bfs(v);
}
```

Thuật toán duyệt rộng

```
void bfs(vertex v) // v thuoc [1..n]
{ QUEUE of vertex Q; // định nghĩa 1 queue
   vertex x,y;
   mark[v-1] = visited;
   Insert(v,Q); // nạp v vào Queue Q
 while !(EMPTY_QUEUE(Q)) //queue khác rỗng
  { remove(x,Q); // lấy phần tử x từ queue
    for (mỗi đỉnh y kề với x)
        if (mark[y-1] == unvisited)
        {   mark[y-1] = visited; //duyệt y
            Insert(y,Q); //Nạp y vào Queue
        }
}  }
```

Bài tập áp dụng

- Cho đồ thị G.
  - Mô tả thuật toán xác định các thành phần liên thông của đồ thị (nộp file .docx)
  - Cài đặt thuật toán xác định các thành phần liên thông của đồ thị