CHƯƠNG 4

ĐỒ THỊ EULER,
ĐỒ THỊ HAMILTON

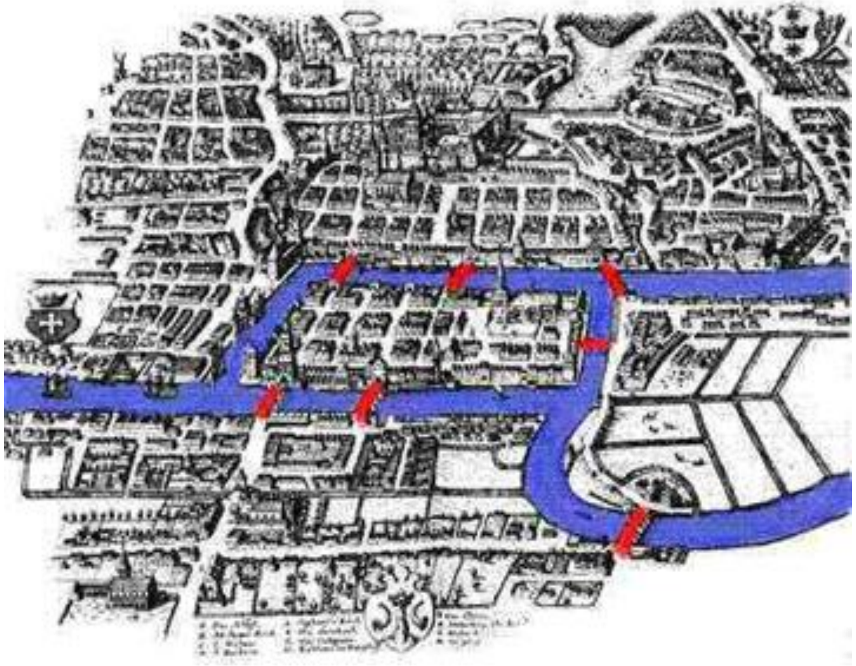
Nội dung

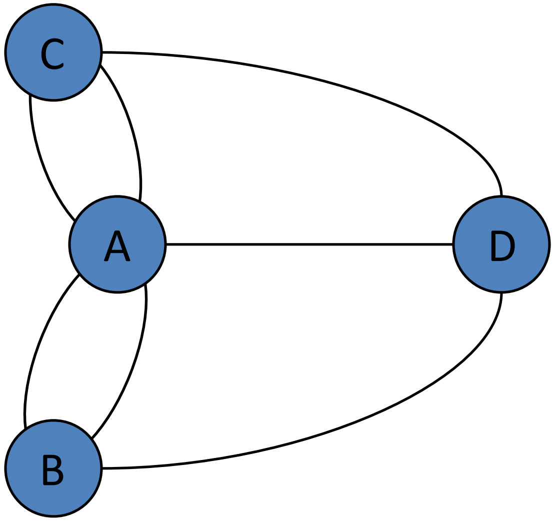
- **Đồ thị Euler**
  - Định nghĩa
  - Định lý
  - Thuật toán
- **Đồ thị Hamilton**
  - Định nghĩa
  - Qui tắc tìm chu trình Hamilton
  - Một số Định lý
  - Thuật toán tìm mọi chu trình Hamilton

ĐỒ THỊ EULER

Bài toán

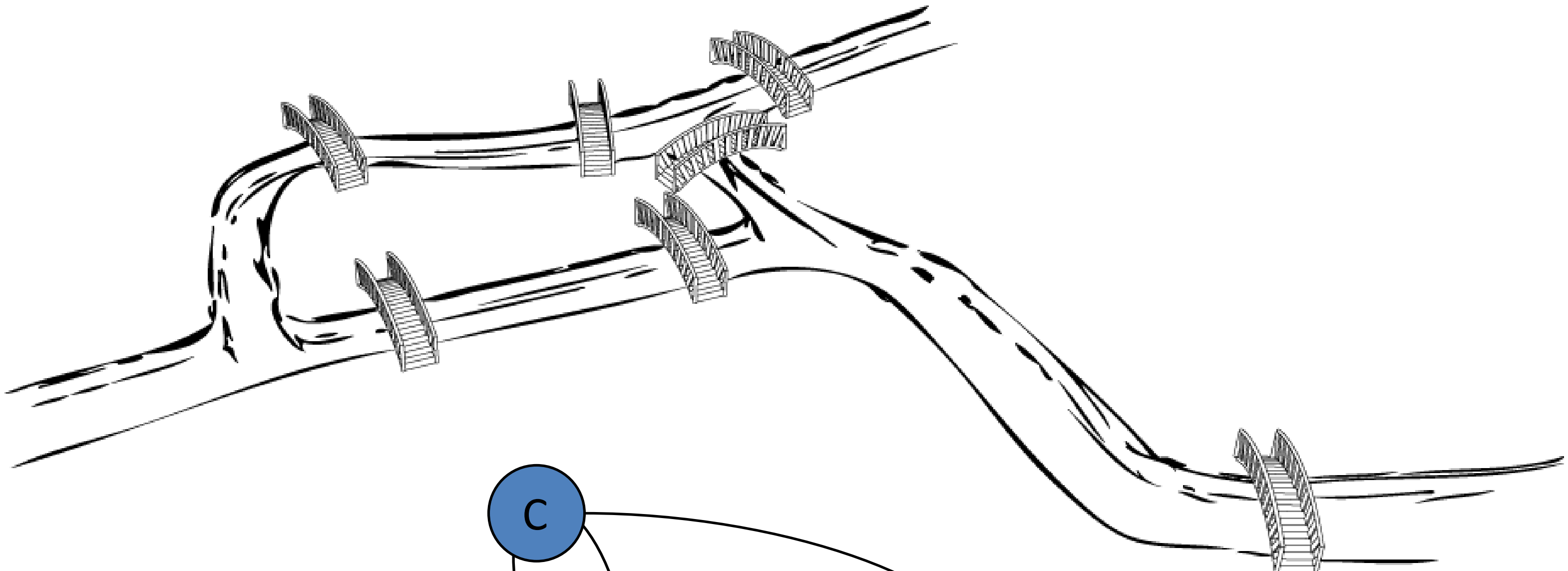
- Bài toán Tìm đường đi qua 7 cái cầu trong thành phố Konigsberg:
  - Làm sao xuất phát từ 1 vị trí, di chuyển qua tất cả các cầu (mỗi cầu qua 1 lần) và trở về vị trí xuất phát

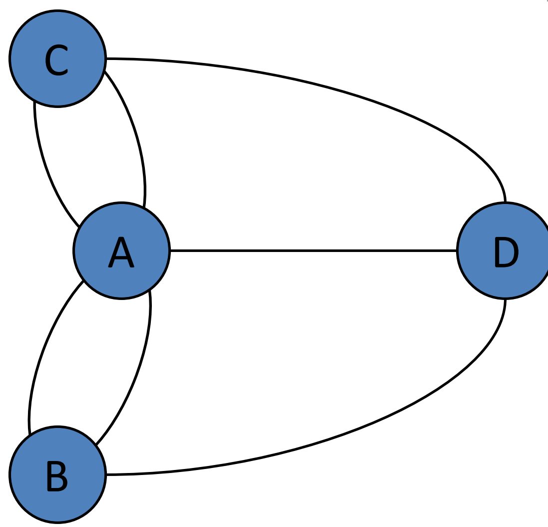




Mô hình đồ thị

Bài toán





Mô hình đồ thị

Các Định nghĩa

- **Chu trình Euler**: là chu trình qua tất cả các cạnh của đồ thị, mỗi cạnh đi qua đúng 1 lần
- **Đường đi Euler**: là đường đi qua tất cả các cạnh của đồ thị, mỗi cạnh đi qua đúng 1 lần
- **Đồ thị Euler**: Là đồ thị có chu trình Euler
- **Đồ thị nửa Euler**: Là đồ thị có đường đi Euler

Định lý

- **Định lý Euler 1:** (Điều kiện cần và đủ để đồ thị là Euler)
  - Đồ thị vô hướng G là đồ thị Euler khi và chỉ khi
    - (1) G liên thông và
    - (2) Mọi đỉnh của G đều có bậc chẵn

Thuật toán

- **Thuật toán kiểm tra đồ thị có Euler hay không**
  - **Input**: G(V, E)
  - **Output**: true/false

  - Bước 1: Kiểm tra tính liên thông của đồ thị
    - Nếu đồ thị không liên thông → Đồ thị không Euler → Kết thúc thuật toán
    - Nếu đồ thị liên thông qua bước 2
  - Bước 2: Tính bậc của mọi đỉnh
  - Bước 3: Kiểm tra bậc chẵn/lẻ của đỉnh
    - Nếu có đỉnh bậc lẻ thì đồ thị G không phải đồ thị Euler
    - Ngược lại G là đồ thị Euler

Cài đặt

```
bool IsEulerGraph()
{








}
```

Chứng minh

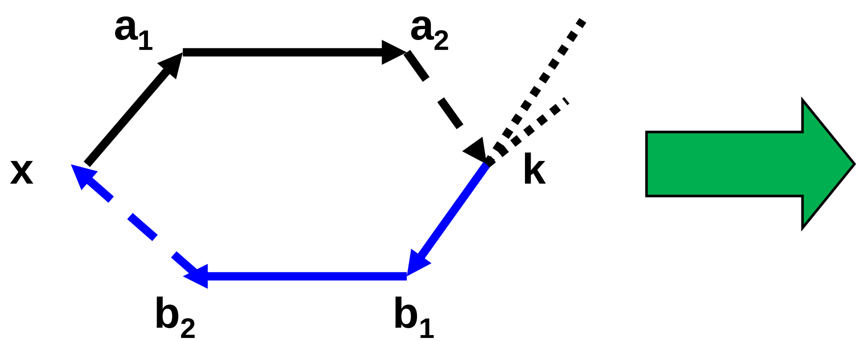
- **Điều kiện Cần**: Cho G=(V,E) là đồ thị Euler, CMR:
  - (1) G liên thông
  - (2) deg(v) chẵn $\forall v \in V$

Chứng minh

- **Điều kiện Đủ**: Ta chứng minh điều kiện đủ bằng cách chỉ ra thuật toán tìm chu trình Euler của đồ thị thỏa 2 tính chất: liên thông và bậc của mọi đỉnh là chẵn
  - Bước 1: Chọn đỉnh x bất kỳ
  - Bước 2 (Tạo chu trình con): Từ đỉnh x chúng ta đi ngẫu nhiên theo các cạnh của đồ thị, mỗi cạnh đi qua đúng 1 lần. Đi cho đến khi tắt đường tại đỉnh y. Lúc này x trùng y (Vì sao?),
    Ta được 1 chu trình C.

Chứng minh

- Bước 3 (Kiểm tra chu trình Euler):
  - Nếu chu trình C chứa mọi cạnh của đồ thị thì ta được chu trình Euler
  - Nếu chu trình C không phải là chu trình Euler thì tồn tại 1 đỉnh k trên chu trình còn cạnh chưa đi qua (Vì sao?)
- Bước 4: (Đảo chu trình): Đảo chu trình hiện tại sao cho lúc đầu chu trình bắt đầu từ x, bây giờ chu trình bắt đầu từ k



(x, a_1, a_2, …, k, b_1, b_2, …, x)

(k, b_1, b_2, …, x, a_1, a_2, …, k)

Chứng minh

- Bước 5: đặt x=k, quay lại bước 2 để tìm chu trình bắt đầu từ k.

- Quá trình này sẽ dừng sau 1 số hữu hạn các bước. Vậy chu trình cuối cùng chứa mọi cạnh của đồ thị là chu trình Euler

Thuật toán

- Thuật toán Tìm chu trình Euler (Tóm tắt)
  - **Input**: Đồ thị Euler G=(V, E)
  - **Output**: Chu trình Euler: euler[]

  - Bước 1: Chọn 1 đỉnh x bất kỳ
  - Bước 2: Lặp
    - Đi ngẫu nhiên từ x theo các cạnh chưa đi qua cho đến khi tắt đường (tại x)
    - Tìm đỉnh k đã đi qua mà còn cạnh chưa đi qua
      - Nếu tìm thấy k thì đảo chu trình hiện tại bắt đầu đi từ k. Sau đó đặt x= k
      - Ngược lại dừng thuật toán

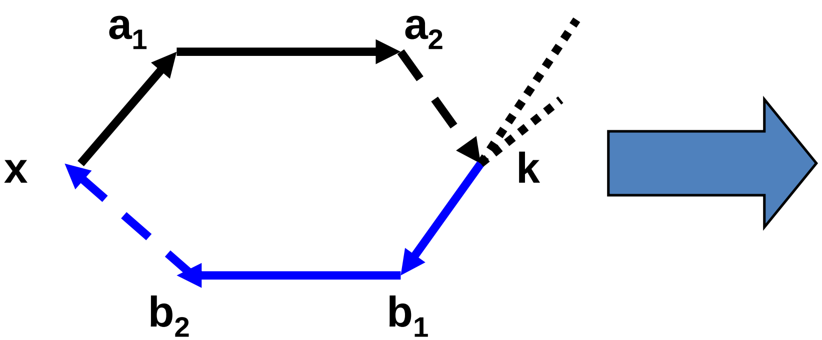
Cài đặt

- Cài đặt
  - Không dùng stack
  - Dùng Stack
  - Đệ quy

CÀI ĐẶT KHÔNG DÙNG STACK

Cài đặt

- Chúng ta phải giải quyết vấn đề đảo chu trình



$(k, b_1, b_2, \ldots, b_m, x, a_1, a_2, \ldots, a_n, k)$

$(x, a_1, a_2, \ldots, a_n, k, b_1, b_2, \ldots b_m, x)$

Cài đặt

- Thuật toán đối xứng gương

$$\left(a_1, a_2, \ldots, a_n, \underbrace{b_1, b_2, \ldots b_m}\right) \Rightarrow \left(\underbrace{b_1, b_2, \ldots, b_m}, a_1, a_2, \ldots, a_n\right)$$

  - Bước 1: Đảo đoạn đầu [a_1, …, a_n]
  - Bước 2: Đảo đoạn cuối [b_1, …, b_m]
  - Bước 3: Đảo nguyên cả mảng

Cài đặt

- Cấu trúc dữ liệu

```
// Input
int[,] a;
int n;

// Output
List<int> euler;

// Hỗ trợ
int[] deg;
```

Cài đặt

- Một số hàm cần cài đặt

```
// Hàm chính
void FindEulerCycle() {
  int x=0;
  while(true) {
    GoFrom(x);
    int i = FindVertex();
    if (i==-1) break;
    ReverseCycle(i);
  }
}
```

Cài đặt

- Một số hàm cần cài đặt

```
// Đi từ x cho đến khi tắt đường
void GoFrom(int x) {
}

// Tìm vị trí của đỉnh k đã đi qua mà còn cạnh chưa đi qua
int FindVertex() {
}

// Đảo chu trình: Chu trình bắt đầu từ vị trí k
void ReverseCycle(int k) {
}

// Đảo đoạn [k1, k2] của mảng a
void Reverse(int k1, int k2) {
}
```

CÀI ĐẶT DÙNG STACK

Cài đặt

- Cài đặt bằng Stack
  - Bước 1: s.Push(x) , x là đỉnh bắt đầu
  - Bước 2: while (s.Count!=0)
    - Lấy đỉnh **u** từ stack ra (không xóa u khỏi stack)
    - Nếu **u** không còn cạnh để đi thì đưa **u** vào chu trình Euler và **xóa u khỏi stack**
    - Ngược lại, tìm đỉnh v kề với u:
      - s.Push(v)
      - Xóa cạnh (u, v)

Cài đặt

- Cài đặt bằng Stack

```
public void Euler(int x)
{









}
```

CÀI ĐẶT BẰNG ĐỆ QUY

Cài đặt

- Cài đặt bằng Đệ quy

```
void Euler(int u) {
  for (int v=0; v<n; v++)
    if (a[u,v]==1) {
      a[u,v] = -1;
      a[v,u] = -1;
      Euler(v);
    }

  euler.Add(u);
}
```

Định lý

- **Định lý Euler 2:** (Điều kiện cần và đủ có đường đi Euler)
  - Đồ thị vô hướng G là đồ thị nửa Euler khi và chỉ khi
    - (1) G liên thông và
    - (2) Có đúng 2 đỉnh có bậc lẻ

Định lý

- **Điều kiện Cần:** Cho G=(V,E) là đồ thị nửa Euler, CMR:
  - (1) G liên thông
  - (2) Có đúng 2 đỉnh có bậc lẻ

Định lý

- **Điều kiện Đủ**:
  - Giả sử G có 2 đỉnh bậc lẻ là a, b. Tạo một đỉnh mới gọi là x và nối đỉnh x đến 2 đỉnh a và b, ta được đồ thị G’ có bậc mọi đỉnh là chẵn.
  - Vậy G’ có chu trình Euler. Trên chu trình Euler chúng ta loại bỏ đỉnh x và 2 cạnh liên thuộc ta được đường đi Euler trong G.

Thuật toán

- **Thuật toán kiểm tra đồ thị có nửa Euler hay không**
  - **Input**: G(V, E)
  - **Output**: true/false

  - Bước 1: Kiểm tra tính liên thông của đồ thị
    - Nếu đồ thị liên thông qua bước 2
    - Nếu đồ thị không liên thông → Đồ thị không nửa Euler → Kết thúc thuật toán
  - Bước 2: Tính bậc của mọi đỉnh
  - Bước 3:
    - Nếu có đúng 2 đỉnh bậc lẻ thì đồ thị G không phải đồ thị nửa Euler
    - Ngược lại G là đồ thị nửa Euler

Cài đặt

```
bool IsSemiEulerGraph()
{
}
```

Thuật toán

- **Thuật toán Tìm đường đi Euler**
  - **Input**: G=(V, E)
  - **Output**: Đường đi Euler
  - Bước 1: Tạo thêm đỉnh mới (đỉnh n), nối đỉnh này với 2 đỉnh bậc lẻ
  - Bước 2: Tìm chu trình Euler
  - Bước 3: Đường Euler là đường đi sinh ra từ chu trình Euler bằng cách bỏ đi 2 cạnh đã thêm vào.

Định lý

- **Định lý Euler 3**: Cho đồ thị có hướng G=(V, E). G có chu trình Euler nếu và chỉ nếu G cân bằng

- **Định lý Euler 4**: Cho đồ thị có hướng G=(V, E). G có đường đi Euler nếu và chỉ nếu G có 2 đỉnh u, v sao cho:
  - $deg^{+}(u) = deg^{-}(u) + 1$
  - $deg^{-}(v) = deg^{+}(v) + 1$
  - Và mọi đỉnh còn lại đều cân bằng

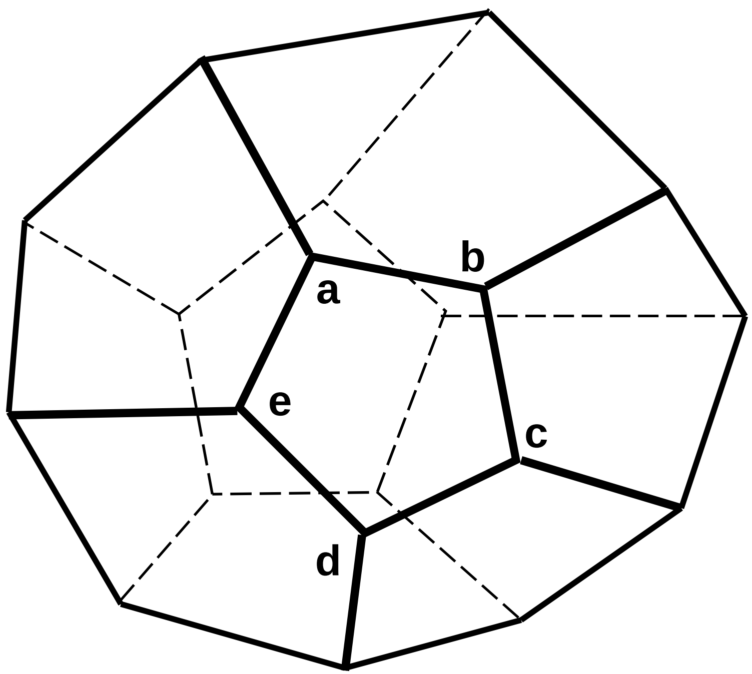
ĐỒ THỊ HAMILTON

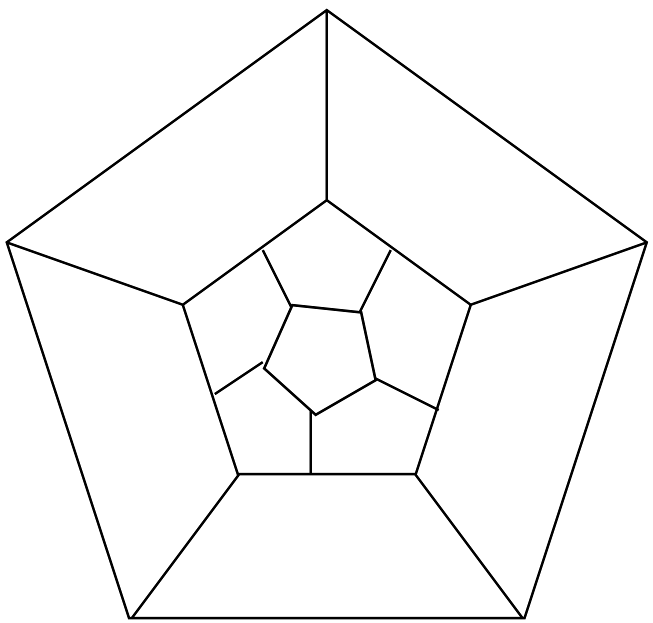
Nội dung

- Đồ thị Hamilton
  - Định nghĩa
  - Qui tắc tìm chu trình Hamilton
  - Một số Định lý
  - Thuật toán tìm mọi chu trình Hamilton

Bài toán

- **Bài toán hình khối (Hamilton 1857):** Cho một khối 12 mặt, mỗi mặt là một ngũ giác. Hỏi xem có thể xuất phát từ 1 đỉnh nào đó thông qua các cạnh để đi qua mọi đỉnh của khối và chỉ đi qua mỗi đỉnh 1 lần





Các định nghĩa

- **Chu trình Hamilton**: Là chu trình đi qua mọi đỉnh của G, mỗi đỉnh đúng 1 lần

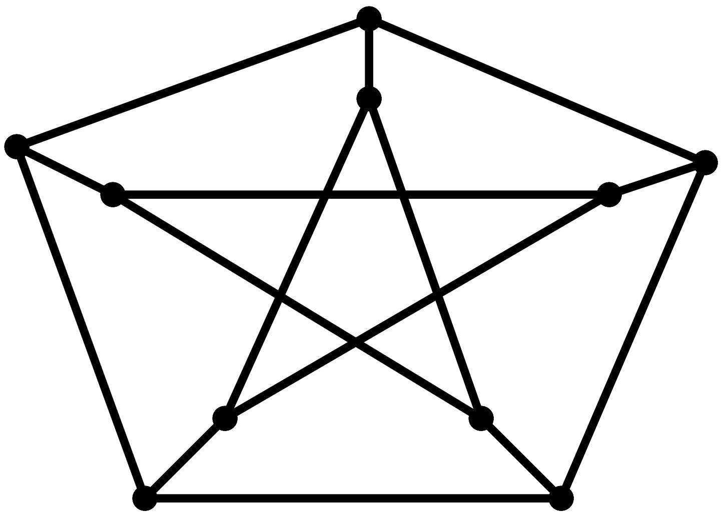
- **Đường đi Hamilton**: Là đường đi, đi qua mọi đỉnh của G, mỗi đỉnh đúng 1 lần

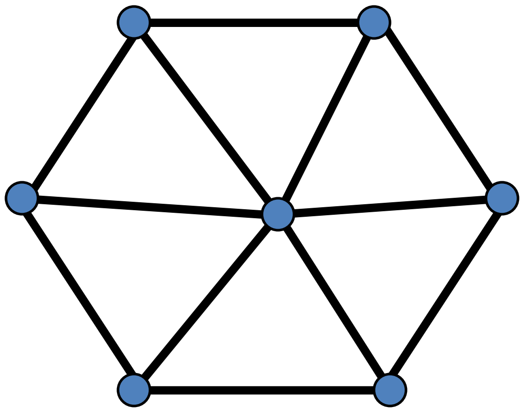
- **Đồ thị Hamilton**: Là đồ thị có chu trình Hamilton

- **Đồ thị nửa Hamilton**: Là đồ thị có đường đi Hamilton

Các định nghĩa

- Ví dụ 1: Bài toán du lịch vòng quanh thế giới
- Ví dụ 2: Kiểm tra xem đồ thị sau có chu trình Hamilton không





Các quy tắc tìm chu trình Hamilton

- Các quy tắc sau dùng để xây dựng chu trình Hamilton H hoặc chỉ ra đồ thị vô hướng không là đồ thị Hamilton
  - **Qui tắc 1**: Tất cả các cạnh kề với đỉnh bậc 2 phải ở trong H.
  - **Qui tắc 2**: Khi chu trình Hamilton mà chúng ta đang xây dựng đi qua đỉnh i thì xóa tất cả các cạnh kề với i mà chưa dùng (vì không còn dùng đến nữa). Điều này có thể cho chúng ta một số đỉnh bậc 2 và lại áp dụng quy tắc 1.

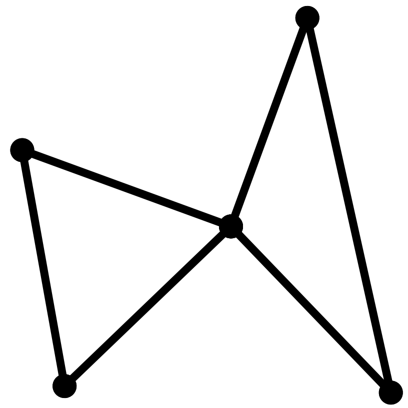
Các quy tắc tìm chu trình Hamilton

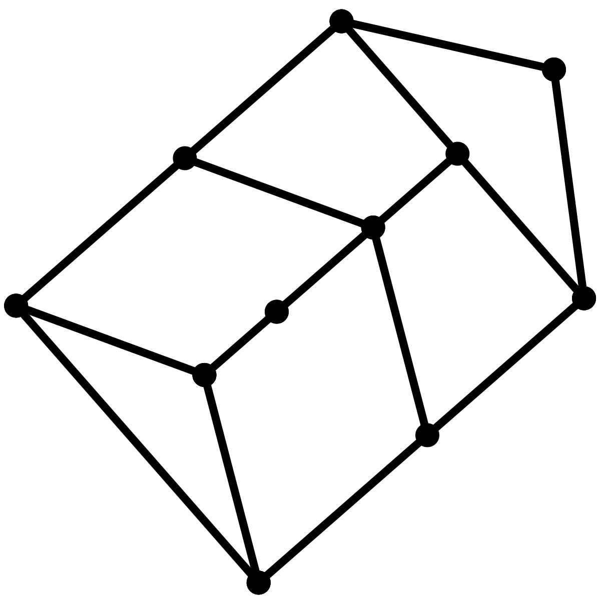
- **Quy tắc 3**: Không có chu trình con nào được tạo ra trong quá trình xây dựng H (nếu không thì không có chu trình Hamilton)

- **Quy tắc 4**: Không có đỉnh cô lập hay đỉnh treo nào được ta ra sau khi áp dụng quy tắc 2 (nếu không thì không có chu trình Hamilton)

Các quy tắc tìm chu trình Hamilton

- Ví dụ: Hãy áp dụng các quy tắc trên tìm chu trình Hamilton trong các đồ thị sau





Định lý

- Khác với đồ thị Euler, đến bây giờ
  - Chưa tìm được điều kiện cần và đủ cho biết một đồ thị có chu trình Hamilton hay không
  - Chưa có thuật toán hiệu quả để tìm chu trình Hamilton
- Các kết quả thu được ở dạng điều kiện đủ, nghĩa là “Nếu đồ thị G có số cạnh đủ lớn thì G là Hamilton”

Định lý

- **Định lý 1 (Dirac, 1952):**
  Cho đơn đồ thị vô hướng G=(V, E) liên thông có n đỉnh (n≥3).
  - Nếu $\deg(u) \geq \frac{n}{2}\ \forall u \in V$ thì G có chu trình Hamilton

Đồ thị Hamilton

- **Định lý 2 (Dirac tổng quát):**
  Cho đồ thị có hướng G=(V, E) liên thông mạnh có n đỉnh.
  - Nếu $deg^{+}(u) \geq \frac{n}{2}$ và $deg^{-}(u) \geq \frac{n}{2}$ thì G có chu trình Hamilton

Đồ thị Hamilton

- **Định lý 3:** Đồ thị đầy đủ K_n với $n \geq 3$ đều có chu trình Hamilton

- **Định lý 4:** Mọi đồ thị đầy đủ liên thông mạnh đều có chu trình Hamilton

Thuật toán tìm mọi chu trình Hamilton

- Thuật toán tìm mọi chu trình Hamilton (n≤20)
  - Chúng ta dùng phương pháp quay lui đề tìm mọi chu trình Hamilton xuất phát từ đỉnh v
  - Trong chu trình Hamilton, mỗi đỉnh chỉ xuất hiện 1 lần, cho nên chúng ta phải đánh dấu những đỉnh đã có trong chu trình trong quá trình tìm kiếm

```
int[] x; // size=n+1
vector<bool> visited;
```

Thuật toán tìm mọi chu trình Hamilton

```
void FindHamiltonCycle(int i) {
  if (i>n và x[i-1]==x[0])
    <Tìm được 1 chu trình Hamilton>

  else
    for (xét mọi đỉnh j kề x[i-1])
      if (visited[j]==false) {
        x[i] = j;
        visted[j]=true;
        FindHamiltonCycle(i+1);
        visted[j]=false;
      }
}
```

Tóm tắt chương 4

- Cạnh
  - Chu trình Euler, Đồ thị Euler
  - Đường đi Euler, Đồ thị nửa Euler
- Đỉnh
  - Chu trình Hamilton, Đồ thị Hamilton
  - Đường đi Hamilton, Đồ thị nửa Hamilton