CÂY

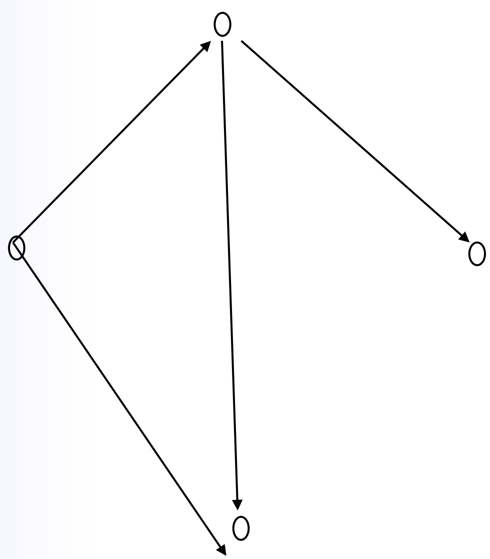
Định nghĩa

- Định nghĩa cây:
  - Cây là đồ thị liên thông và không có chu trình
  - Một rừng p cây là một đồ thị gồm p thành phần liên thông, trong đó mỗi thành phần liên thông là một cây

- Ghi chú: Định nghĩa cây hàm ý rằng mọi cây đều không chứa khuyên cũng không chứa cạnh song song.

Định nghĩa

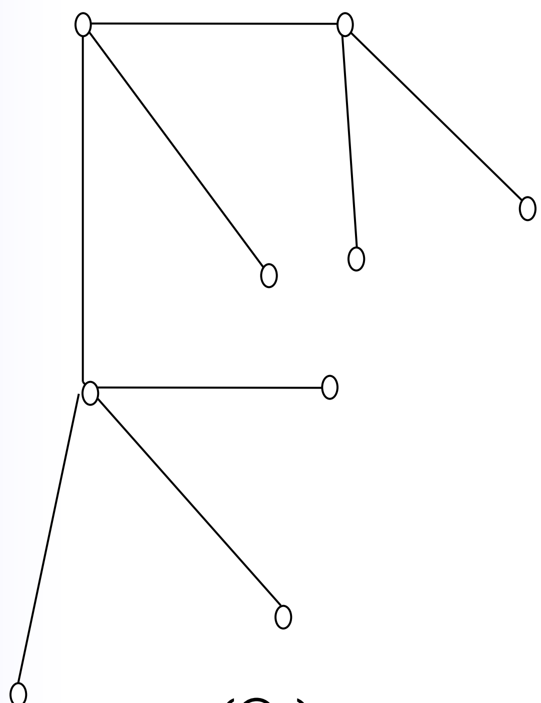
- <u>Ví dụ.</u> (G1) không là cây



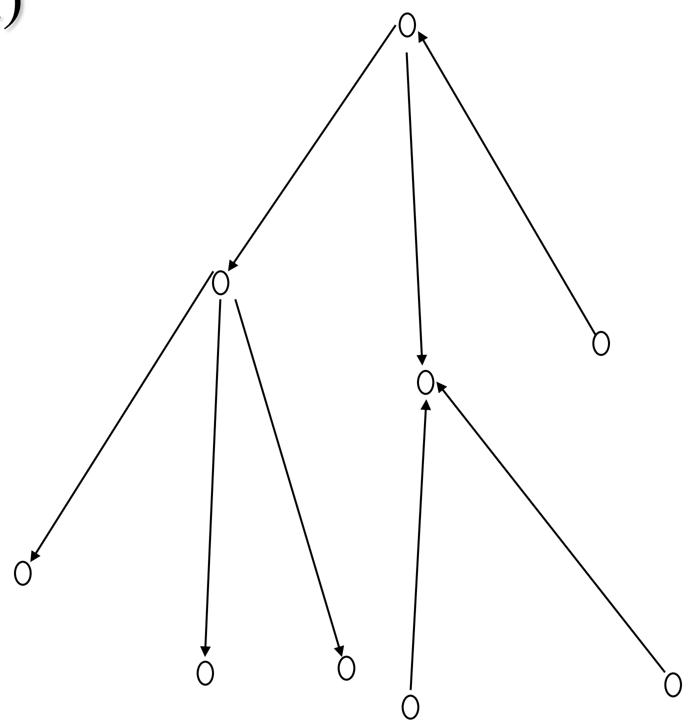
(G_1)

Định nghĩa

- <u>Ví dụ.</u>(G2) và (G3) là cây (chú ý định nghĩa chu trình của đồ thị có hướng trong chương I)



(G_2)



(G_3)

Định lý về sự tồn tại các đỉnh treo

- Nếu một cây T gồm n đỉnh với $n \geq 2$ thì T chứa ít nhất hai đỉnh treo

Định lý
về các định nghĩa tương đương

- Xét một đồ thị G gồm n đỉnh, các điều sau đây tương đương.
  - (a) Đồ thị G là cây.
  - (b) Giữa hai đỉnh bất kỳ của G, tồn tại duy nhất một dây chuyền nối chúng với nhau.
  - (c) G liên thông tối tiểu (nghĩa là G liên thông và nếu xóa đi bất kỳ một cạnh nào của G thì nó không còn liên thông nữa).

Định lý
về các định nghĩa tương đương

- Xét một đồ thị G gồm n đỉnh, các điều sau đây tương đương.
  - (d) Thêm một cạnh nối 2 đỉnh bất kỳ của G thì G sẽ chứa một chu trình duy nhất.
  - (e) G liên thông và có n-1 cạnh
  - (f) G không có chu trình và có n-1 cạnh

Cây tối đại
(cây phủ, cây bao trùm, cây khung)

- **Định nghĩa**:
  - Cho G=(X, E) là một đồ thị liên thông và T=(X, F) là một đồ thị bộ phận của G. Nếu T là cây thì T được gọi là một cây tối đại của G.
- **Định lý (sự tồn tại cây tối đại)**
  - Mọi đồ thị liên thông đều có chứa ít nhất một cây tối đại

Cây tối đại (cây phủ, cây bao trùm, cây khung)

- **Thuật toán (tìm một cây tối đại của đồ thị G)**
  - Cho G=(X, E) là một đồ thị liên thông gồm n đỉnh. Thuật toán sau đây cho phép tìm ra được một cây tối đại của G.
  - Bước1. Chọn tùy ý $v \in X$ và khởi tạo $V := \{ v \}$; $T := \varnothing$
  - Bước 2. Chọn $w \in X \setminus V$ sao cho có một cạnh e nào đó của G nối w với một đỉnh trong V
  - Bước 3. Gán $V := V \cup \{w\}$ và $T := T \cup \{e\}$
  - Bước 4. Nếu T đủ n-1 phần tử thì dừng, ngược lại làm tiếp tục bước 2.

Cây tối đại ngắn nhất

- **Định nghĩa**: Cho đồ thị G=(X, E).
  - (a) Đồ thị G được gọi là có trọng nếu mỗi cạnh của G được tương ứng với một số thực dương, nghĩa là có một ánh xạ như sau:

$$L: E \longrightarrow \mathbb{R}^+$$

$$e \longmapsto L(e)$$

Cây tối đại ngắn nhất

- **Định nghĩa**: Cho đồ thị G=(X, E).
  - (b) Trọng lượng (hay giá) của một cây tối đại T trong đồ thị liên thông có trọng bằng với tổng trọng lượng các cạnh trong cây:

$$L(T) = \sum (e \in T)\, L(e)$$

  - (c) Giả sử G liên thông có trọng. Cây tối đại ngắn nhất của G là cây tối đại có trọng lượng nhỏ nhất khi xét trong tập hợp tất cả các cây tối đại có thể có của G.

Thuật toán Prim

- Cho G=(X, E) là một đồ thị liên thông có trọng gồm n đỉnh. Thuật toán Prim được dùng để tìm ra cây tối đại ngắn nhất của G.
- Bước 1. Chọn tùy ý $v \in X$ và khởi tạo $Y := \{ v \}$; $T := \emptyset$
- Bước 2. Trong số những cạnh e nối đỉnh w với v mà $w \in X \backslash V$ và $v \in Y$ ta chọn cạnh có trọng lượng nhỏ nhất.
- Bước 3. Gán $Y := Y \cup \{w\}$ và $T := T \cup \{e\}$
- Bước 4. Nếu T đủ n-1 phần tử thì dừng, ngược lại làm tiếp tục bước 2.

Thuật toán Prim

- Cài đặt thuật toán Prim
- Trong các thuật toán tìm cây tối đại ngắn nhất chúng ta có thể bỏ đi hướng các cạnh và các khuyên; đối với các cạnh song song thì có thể bỏ đi và chỉ để lại một cạnh trọng lượng nhỏ nhất trong chúng. Vì vậy dữ liệu nhập cho thuật toán thường là ma trận trọng lượng được qui ước như sau:
- $$Lij = \begin{cases} \text{trọng lượng cạnh nhỏ nhất nối i đến j nếu có} \\ \text{0 nếu không có cạnh nối i đến j} \end{cases}$$

Thuật toán Kruskal

- Cho đồ thị G=(X, E).
- Bước 1. Sắp xếp các cạnh theo thứ tự trọng lượng tăng dần và khởi tạo T := ∅.
- Bước 2. Lần lượt lấy từng cạnh e thuộc danh sách đã sắp xếp. Nếu T+{e} không chứa chu trình thì gán T := T+{e}.
- Bước 3. Nếu T đủ n-1 phần tử thì dừng, ngược lại làm tiếp tục bước 2.