

CHƯƠNG 2.

CẤU TRÚC CÂY.

2.1 ĐỊNH NGHĨA & THÍ DỤ.

2.1.1 CÂY.

Cây là một đồ thị không định hướng, liên thông và không có chu trình.

THÍ DỤ.

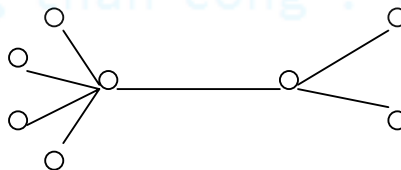


FIG. 2.1. Cây.

- Chiều dài của đường nối hai đỉnh lại với nhau được gọi là khoảng cách giữa hai đỉnh.

TÍNH CHẤT.

- Giữa hai đỉnh bất kỳ của một cây sẽ có duy nhất một dây chuyền nối chúng lại với nhau.
- Một cây n đỉnh sẽ có $n - 1$ cạnh. Cộng thêm vào cây một cạnh giữa hai đỉnh bất kỳ sẽ tạo nên một chu trình duy nhất.

2.1.2 RỪNG.

Là một đồ thị không định hướng và không có chu trình (Không liên thông mạnh) Mỗi thành phần liên thông của một rừng là một cây.

2.1.3 CẤU TRÚC CÂY (CÂY CÓ GỐC).

Là một đồ thị có định hướng sao cho mỗi đỉnh đều có một đỉnh trước trừ một phần tử duy nhất không có, được gọi là **GỐC**. Với mọi đỉnh x thì có duy nhất một đường từ gốc đi đến x .

Xét một đỉnh x của một cây T có gốc là r :

- Một đỉnh y bất kỳ nằm trên đường hướng từ gốc đến x , được gọi là các **ĐỈNH TRƯỚC** (ANCETRE) của x , và x được là **ĐỈNH SAU** (DESCENDANT) 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 được gọi là **ANH EM**. Một đỉnh không có con được gọi là **LÁ**. Những đỉnh không là **LÁ** được gọi là **ĐỈNH TRONG**.
- Chiều dài của đường từ gốc đến đỉnh được gọi là **độ sâu** của đỉnh đó.
- **Mức (Niveau)** của một đỉnh trong T là khoảng cách từ gốc đến x .
 - ❖ Mức của nút gốc = 0.
 - ❖ Mức của nút khác gốc = Mức của cây con nhỏ nhất chứa nó + 1.
- **Chiều cao hay độ sâu** (Hauteur, profondeur) của cây là giá trị lớn nhất của mức của các đỉnh trong cây.
- Nếu mỗi đỉnh trong cây có tối đa hai con, thì ta gọi đó là **cây nhị phân**.
- **Bậc của nút & bậc của cây (Degrée).**
 - ❖ Bậc của nút là số cây con của nút đó.
 - ❖ Bậc của cây là bậc lớn nhất của các nút của cây. Nếu cây có bậc là n , ta gọi là cây n -cành.

THÍ DỤ . Cây 3 – cành có gốc, với 8 đỉnh và có độ cao là 4.

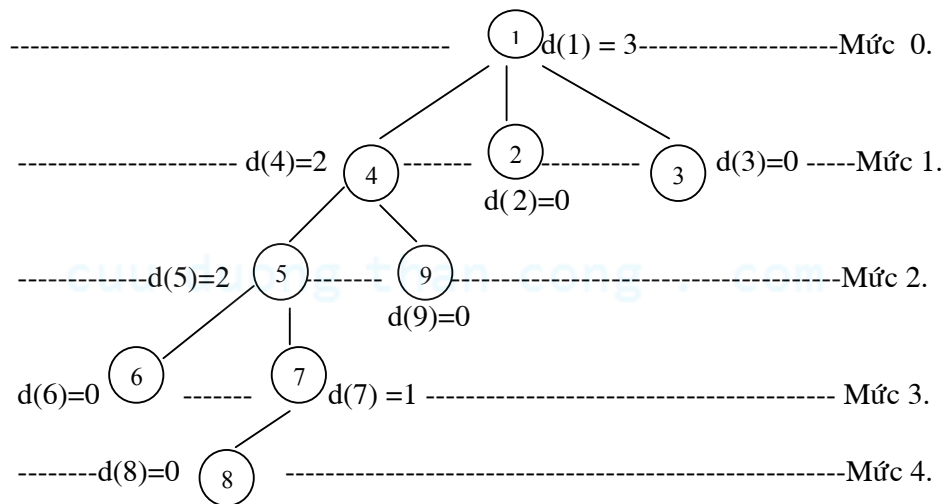
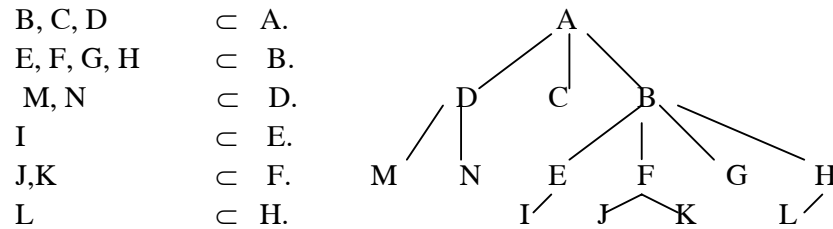


FIG.2.2. Cây có gốc.

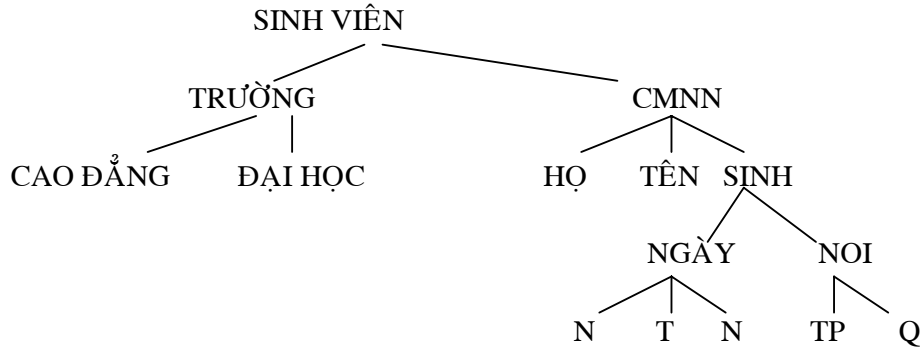
2.1.4. THÍ DỤ.

- Đôi khi ta có thể biểu diễn một quan hệ bao hàm thức của nhiều tập hợp bằng một cấu trúc cây.

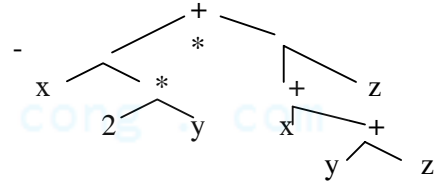
Thí dụ. Bao hàm của các tập hợp sau có thể biểu diễn thành cấu trúc cây như sau :



- Một Biến có cấu trúc** có thể biểu diễn dưới dạng cây.

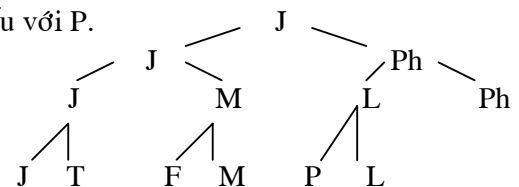


- Biểu thức số học.** Biểu thức $X = (x - (2 * y) + ((x + (y + z)) * z))$ có thể biểu diễn thành hình cây như sau :



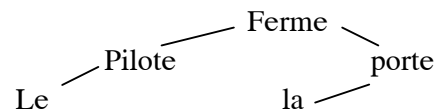
- Vòng loại** trong một cuộc thi đấu bóng bàn.

- ❖ Vòng 1. J đấu với T, F đấu với M, L đấu với P.
- ❖ Vòng 2. J đấu với M, L đấu với Ph
- ❖ Vòng 3. J đấu Ph.
- ❖ Cuối cùng J thắng.



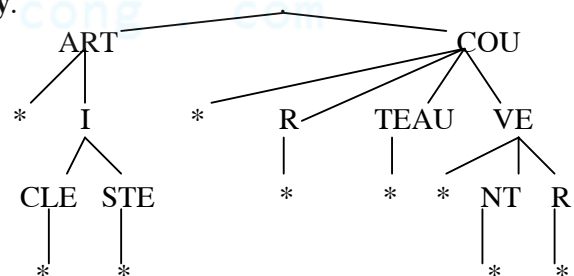
- Câu trong ngôn ngữ tự nhiên** (hay trong ngôn ngữ lập trình)

Đối với câu « Le Pilote ferme la porte »
Có thể biểu diễn dưới dạng



- Tự điển có thể tổ chức theo hình cây.**

Chẳng hạn tự điển gồm các từ ART, ARTICLE, ASTISTE, COU, COUR, COUTEAU, COUVE, COUVENT, COUVER có thể biểu diễn theo hình vẽ sau. Ký tự «*» chỉ chấm dứt một từ. Chú ý, thứ tự ALPHABET theo thứ tự từ phải sang trái.



2.2 TÍNH CHẤT CƠ BẢN.

2.2.1 ĐỊNH LÝ 1.

Cho G là một cây bậc $n > 1$. Các tính chất sau đây tương đương với nhau :

1. G liên thông và không có chu trình.
2. G liên thông và có $n - 1$ cạnh.
3. G không có chu trình và có $n - 1$ cạnh.
4. G không có chu trình và nếu thêm vào một cạnh giữa hai đỉnh không kề sẽ tạo một chu trình duy nhất giữa chúng.
5. G liên thông tối thiểu (có nghĩa là nếu xóa đi một cạnh bất kỳ thì G không còn liên thông nữa)
6. Mọi cặp đỉnh có duy nhất dây chuyền nối chúng.

CHỨNG MINH. Bài tập.

2.2.2 ĐỊNH LÝ 2.

Một đồ thị $G = (X, U)$ là một đồ thị có chứa một đồ thị riêng phần nếu và chỉ nếu G liên thông.

CHỨNG MINH. Bài tập.

2.2.3 ĐỊNH LÝ 3.

Mọi Cấu trúc cây là một cây.

CHỨNG MINH. Bài tập.

2.3 CÂY NHỊ PHÂN.

2.3.1. ĐỊNH NGHĨA (THEO ĐỆ QUI).

Một cây nhị phân B hoặc là \emptyset hoặc có dạng :

$B = \langle O, B_1, B_2 \rangle$ trong đó :

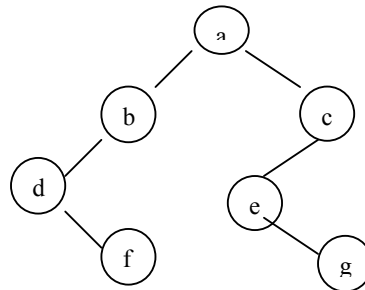
O : gốc,

B_1 : cây con trái và

B_2 : cây con phải.

2.3.2. BIỂU DIỄN CÂY NHỊ PHÂN.

THÍ DỤ.



❖ **SỬ DỤNG BẢNG.** Có thể định nghĩa kiểu dữ liệu như sau :

Type Arctab = Array [1..n] of Record $v : t$;

$G : integer$;

$D : integer$;

End ;

Với thí dụ ở trên, ta có :

		Trái	Phải
1			
2	d	0	8
3	a	5	6
4	e	0	9
5	b	2	0
6	c	4	0
7			
8	f	0	0
9	g	0	0
10			

❖ **SỬ DỤNG CON TRỎ.** Có thể định nghĩa kiểu dữ liệu như sau :

Type Pt = ^nut ;

nut = Record

$G : Pt$;

$Val : t$;

$D : Pt$;

End ;

2.3.3. DUYỆT MỘT CÂY NHỊ PHÂN.

Có 3 cách duyệt một cây nhị phân (phụ thuộc theo gốc).

1. THỨ TỰ TRƯỚC (PREFIXÉ).

- Xử lý gốc.
- Duyệt cây con trái.
- Duyệt cây con phải.

2. THỨ TỰ GIỮA (INFIXÉ).

- Duyệt cây con trái.
- Xử lý gốc.
- Duyệt cây con phải.

3. THỨ TỰ SAU (POSTFIXÉ).

- Duyệt cây con trái.
- Duyệt cây con phải.
- Xử lý gốc.

THÍ DỤ. Theo cây ở thí dụ trên , ta có :

- **Trước** : a b d f c e g.
- **Giữa** : d f b a e g c.
- **Sau** : f d b g e c a.

2.4 CÂY PHỦ.

2.4.1. ĐỊNH NGHĨA.

Cho một đồ thị vô hướng G . Một cây H gọi là cây phủ của G nếu H là cây riêng phần của G chứa mọi đỉnh của G .

2.4.2. ĐỊNH LÝ.

Đồ thị G có cây phủ nếu và chỉ nếu G liên thông.

2.4.3. GIẢI THUẬT TÌM CÂY PHỦ.

Xét một đồ thị G .

GIẢI THUẬT.

- **Bước 1.** Chọn tùy ý một đỉnh của G đặt vào H .
- **Bước 2.** Nếu mọi đỉnh của G đều nằm trong H thì dừng.
- **Bước 3.** Nếu không, tìm một đỉnh của G không nằm trong H mà nó có thể nối nó với một đỉnh của H bằng một cạnh. Thêm đỉnh và cạnh này vào H . Quay về bước 2.

THÍ DỤ . Cho đồ thị G theo hình vẽ sau :

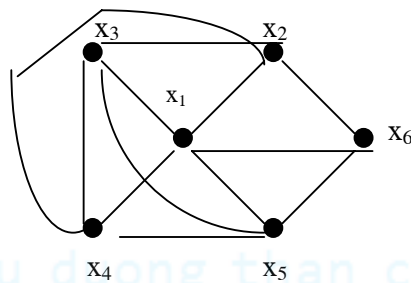


FIG. 2.3.

- ❖ Khởi từ x_1 . $T = \emptyset$.
- ❖ Bước 1. Chọn x_2 , $T = \{(x_1, x_2)\}$.
- ❖ Bước 2. Chọn x_3 , $T = \{(x_1, x_2), (x_2, x_3)\}$.
- ❖ Bước 3. Chọn x_4 , $T = \{(x_1, x_2), (x_2, x_3), (x_3, x_4)\}$.
- ❖ Bước 4. Chọn x_5 , $T = \{(x_1, x_2), (x_2, x_3), (x_3, x_4), (x_4, x_5)\}$.
- ❖ Bước 5. Chọn x_6 , $T = \{(x_1, x_2), (x_2, x_3), (x_3, x_4), (x_4, x_5), (x_5, x_6)\}$.

Kết quả : T là cây phủ của G .

2.4.4. ĐỊNH LÝ.

Coi một cây phủ H của G .

Thêm vào H một cạnh của G (không thuộc H), ta được một chu trình trong H . Hủy một cạnh bất kỳ trên chu trình này ra khỏi H , ta nhận được một cây phủ mới của G .

2.4.5. GIẢI THUẬT KIỂM TRA TÍNH LIÊN THÔNG.

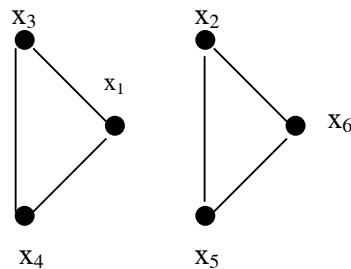
Xét một đồ thị không định hướng G .

Áp dụng giải thuật trên vào G . Khi giải thuật dừng.

- Nếu H chứa mọi đỉnh của G thì G liên thông và H là một cây phủ của G .
- Nếu H không chứa mọi đỉnh của G thì G không liên thông và H là một cây phủ của một thành phần liên thông của G .

THÍ DỤ 1. Trường hợp đồ thị G ở hình FIG. 2.3. thì ta có G liên thông.

THÍ DỤ 2. Cho đồ thị G theo hình vẽ sau :



- ❖ Khởi từ x_1 . $T = \emptyset$.
- ❖ Bước 1. Chọn x_3 , $T = \{(x_1, x_3)\}$.
- ❖ Bước 2. Chọn x_4 , $T = \{(x_1, x_3), (x_3, x_4)\}$.

Thuật toán dừng. T là cây phủ của một thành phần liên thông của G mà thôi.

2.4.6. GIẢI THUẬT TÌM THÀNH PHẦN LIÊN THÔNG THEO CÁCH DUYỆT T THEO CHIỀU SÂU.

Do thủ tục duyệt theo chiều sâu PROF(s) cho phép thăm tất cả các đỉnh thuộc cùng một thành phần liên thông với đỉnh s , nên số thành phần liên thông của đồ thị chính bằng số lần gọi đến thủ tục này. Vấn đề còn lại là cách ghi nhận các đỉnh trong từng thành phần liên thông bằng cách cải tiến thủ tục chiều theo chiều sâu PROF(s) như sau :

THỦ TỤC DFS(int k) ;

//Duyệt theo chiều sâu bắt đầu từ đỉnh k

```
{
    Mark[k] = socomp;
    For (int i = 1; i ≤ n ; i++)
        if (a[i][k] == 1 && (Mark[i] == 0)) DFS(i);
}
```

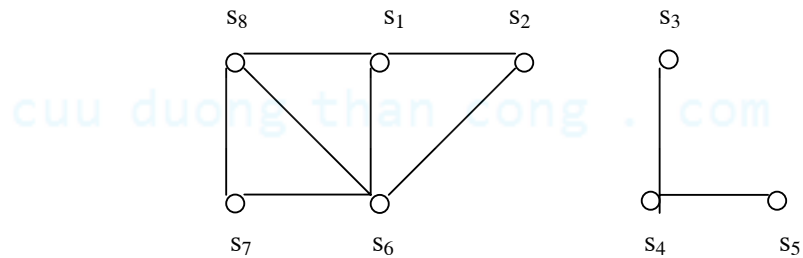

THỦ TỤC CONNEXE ;

```

{
    // Khởi tạo số liệu ban đầu cho Mark (các đỉnh đã duyệt rồi) và socomp (số
    // thành phần liên thông
    For (int j= 1 ;j≤ n ;j++) { Mark[j] =0 ; Socomp =0 ; }
    //Gọi thủ tục để xác định các thành phần liên thông
    For (int i= 1 ;i≤n ;i++)
        If Mark [i] ==0
        {
            Socomp = Socomp +1 ;
            DFS(i) ;
        }
    //In kết quả
}

```

THÍ DỤ.



- ❖ Khởi từ s_1 . Gọi DFS(1), ta có Tập đánh dấu $\{s_1, s_2, s_6, s_7, s_8\}$.
- ❖ $i= 3$ Gọi DFS(3), ta có Tập đánh dấu $\{s_3, s_4, s_5\}$.

- ❖ **Kết quả.** Có 2 thành phần liên thông.

$$C_1 = \{s_1, s_2, s_6, s_7, s_8\}.$$

$$C_2 = \{s_3, s_4, s_5\}.$$

2.5 CÂY PHỦ TỐI THIỂU.

BÀI TOÁN 1. Cho một đồ thị liên thông $G = (X, U)$, và, với mọi cạnh u liên kết với một con số $l(u)$ mà ta gọi là chiều dài (trong lượng). Vấn đề đặt ra là tìm một cây riêng phần $H = (X, V)$ của G sao cho tổng chiều dài $\sum_u l(u)$ là nhỏ nhất.

THÍ DỤ. Bài toán này thường gặp trong viễn thông và trong nhiều trường hợp khác. Chẳng hạn, bài toán đặt ra cho chúng ta là Tìm đường dây cáp ngắn nhất để nối n thành phố lại với nhau? Các thành phố được biểu diễn là đỉnh của một đồ thị và $l(x, y)$ là khoảng cách giữa thành phố x và y . Mạng dây cáp nối bắt buộc phải liên thông. Ở đây, vấn đề là tìm cây riêng phần có tổng chiều dài nhỏ nhất nối tất cả các đỉnh?

BỔ ĐỀ. Nếu $G = (X, U)$ là một đồ thị đầy đủ và nếu tất cả các chiều dài $l(u)$ tương ứng của các cạnh đều phân biệt thì khi ấy, Bài toán 1 có một lời giải duy nhất (X, V) . Tập $V = \{v_1, v_2, \dots, v_{n-1}\}$ nhận được theo cách sau đây :

- Chọn v_1 là cạnh có chiều dài nhỏ nhất.
- v_2 là cạnh có chiều dài nhỏ nhất sao cho $v_2 \neq v_1$ và $V_2 = \{v_1, v_2\}$ không chứa chu trình.
- v_3 là cạnh nhỏ nhất sao cho $v_3 \neq v_2 \neq v_1$ và $V_3 = \{v_1, v_2, v_3\}$ không chứa chu trình.
- Cứ thế, tiếp tục.

2.5.1. THUẬT TOÁN PRIM.**Ký hiệu :**

- ♦ A = Ma trận kề biểu diễn đồ thị, có trọng lượng, được định nghĩa như sau :

$$A = [a_{i,j}] = \begin{cases} l(i,j) & \text{chiều dài của cạnh ứng } u=(i,j) \in U \\ \infty & u=(i,j) \notin U \\ 0, & i=j \end{cases}$$

- ♦ M = Tập đỉnh chưa đánh dấu (có số phần tử là n_0).
- ♦ $Pr(p)$ = Đỉnh trước đỉnh p .
- ♦ d = Tập chiều dài của Cây phủ có chiều dài ngắn nhất.
- ♦ $Mark$ = Tập đỉnh đã đánh dấu (đã xét rồi), định nghĩa như sau :
 $Mark[i] = \begin{cases} 1, & \text{nếu đỉnh đã xét rồi,} \\ 0, & \text{ngược lại.} \end{cases}$

NGUYÊN LÝ THUẬT TOÁN.1. Khởi tạo : Xuất phát từ đỉnh 1. $T = \emptyset$,

$$M = \{2, \dots, n\}$$

$$Pr = [1, 1, \dots, 1]$$

$$d = a[1, j], j=1..n \text{ (Dòng đầu của ma trận kề } A)$$

$$Mark = [1, 0, \dots, 0]$$

2. Ở mỗi bước lặp, chọn đỉnh đánh dấu là đỉnh có độ dài ngắn nhất.

$$\diamond k = \text{Argmin}_{x \in M} d[x].$$

$$\diamond Mark[k]=1.$$

♦ Cập nhật lại $d[i]$, $Pr[i]$ với $i \in M \setminus \{k\}$ theo công thức:

- $d[i] = a[k, i]$ nếu $d[i] > a[k, i]$.
- $Pr[i] = k$.

$$\diamond \text{Thay } M := M \setminus \{k\}.$$

Nếu $M = \emptyset$. Dừng. Nếu không, quay lại 2.**Độ phức tạp :** $O(m \log n)$.

THÍ DỤ. Ta có Ma trận kề A, biểu diễn Đồ thị ở FIG. 2.3., như sau :

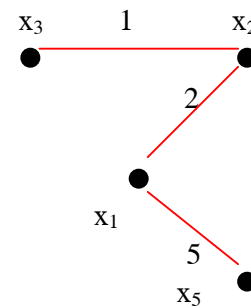
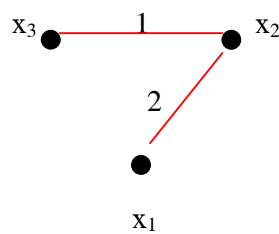
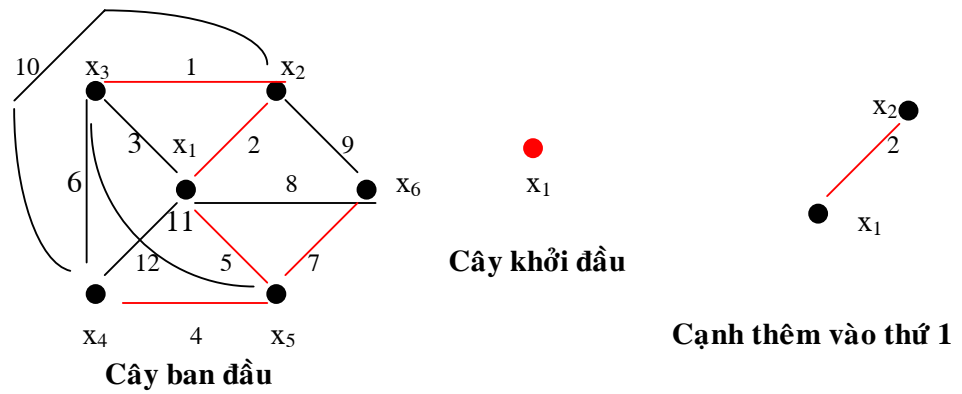
		1	2	3	4	5	6
	1	0	2	3	11	5	8
	2	2	0	1	10	∞	9
A =	3	3	1	0	6	12	∞
	4	11	10	6	0	4	∞
	5	5	∞	12	4	0	7
	6	8	9	∞	∞	7	0

Các bước của thuật toán thực hiện như sau :

- Gán ban đầu cho : M, d, Pr :
 $M = \{ \quad, 2, 3, 4, 5, 6 \}$
 $d = [0, 2, 3, 11, 5, 8]$
 $Pr = [1, 1, 1, 1, 1, 1]$
- **Bước 1.** Chọn **đỉnh** s_2 . Cập nhật M, d, Pr :
 $M = \{ \quad, 3, 4, 5, 6 \}$
 $d = [0, 2, 1, 10, 5, 8]$
 $Pr = [1, 1, 2, 2, 1, 1]$
- **Bước 2.** Chọn **đỉnh** s_3 . Cập nhật M, d, Pr :
 $M = \{ \quad, \quad, 4, 5, 6 \}$
 $d = [0, 2, 1, 6, 5, 8]$
 $Pr = [1, 1, 2, 3, 1, 1]$
- **Bước 3.** Chọn **đỉnh** s_5 . Cập nhật M, d, Pr :
 $M = \{ \quad, \quad, 4, \quad, 6 \}$
 $d = [0, 2, 1, 4, 5, 7]$
 $Pr = [1, 1, 2, 5, 1, 5]$
- **Bước 4.** Chọn **đỉnh** s_4 . Cập nhật M, d, Pr :
 $M = \{ \quad, \quad, \quad, \quad, 6 \}$
 $d = [0, 2, 1, 4, 5, 7]$
 $Pr = [1, 1, 2, 5, 1, 5]$

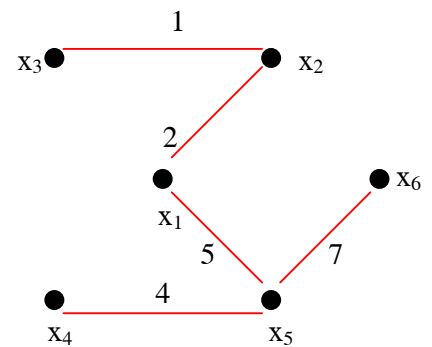
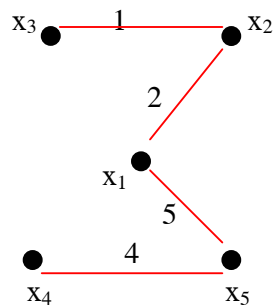
Ta có Kết quả như sau :

- Cây Phủ có độ dài ngắn nhất theo các Bước lặp :
 $T = (x_1, x_2), (x_2, x_3), (x_1, x_5), (x_5, x_4), (x_5, x_6)$ và có độ dài $l(T) = 19$
- Cây Phủ có độ dài ngắn nhất đọc kết quả theo d và Pr :
 $T = (x_1, x_2), (x_2, x_3), (x_5, x_4), (x_1, x_5), (x_5, x_6)$ và có độ dài $l(T) = 19$



Cạnh thêm vào thứ 2

Cạnh thêm vào thứ 3



Cạnh thêm vào thứ 4

Cạnh thêm vào thứ 5.

FIG. 2.3. Tìm Cây phủ có độ dài ngắn nhất theo PRIM ($s=1$).

2.5.2. THUẬT TOÁN KRUSKAL (1956).

Cho đồ thị $G = (X, U)$ là đồ thị liên thông không định hướng, có trọng lượng. Giả sử đã sắp xếp các cạnh của đồ thị theo thứ tự không giảm theo chiều dài.

Ý tưởng của thuật toán KRUSKAL ở mỗi bước lặp, ta bổ sung vào tập cạnh của cây phủ $H = (X, T)$ sao cho không tạo thành chu trình.

Thuật toán dừng khi tất cả các đỉnh của đồ thị đều được nối, nghĩa là số cạnh của H bằng $n - 1$. Đây là thuật toán « háu ăn », theo nghĩa là ở mỗi bước, ta chọn một lời giải tối ưu địa phương và mong muốn lời giải tối ưu địa phương này là tối ưu toàn cục.

Cây nhận được là duy nhất nếu tất cả các cạnh có chiều dài khác nhau.

Độ phức tạp : $O(m \log m)$.

THỦ TỤC KRUSKAL ;

Begin

$T := \{\emptyset\}$;

While $\text{Card}(T) < (n-1)$ and $(U \neq \emptyset)$ Do Begin

 Chọn u là cạnh có độ dài nhỏ nhất trong U ;

$U := U \setminus \{u\}$;

 If $(T \cup \{u\})$ không chứa chu trình then $T := T \cup \{u\}$;

End ;

If $(\text{Card}(T) < n-1)$ Then Đồ thị không liên thông.

End ;

THÍ DỤ 1. Xem hình FIG. 2.3. Ta có :

$U = \{(x_2, x_3), (x_1, x_2), (x_1, x_3), (x_4, x_5), (x_1, x_5), (x_3, x_4), (x_5, x_6), (x_1, x_6), (x_2, x_6), (x_2, x_4), (x_1, x_4), (x_3, x_5)\}$

$L(U) = \{ 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12 \}$

Các bước của thuật toán thực hiện như sau :

- Bước 1. $T = \{(x_2, x_3)\}$,
 $L(T) = \{ 1 \}$
- Bước 2. $T = \{(x_2, x_3), (x_1, x_2)\}$,
 $L(T) = \{ 1, 2 \}$
- Bước 3. $T = \{(x_2, x_3), (x_1, x_2), (x_4, x_5)\}$,
 $L(T) = \{ 1, 2, 4 \}$
- Bước 4. $T = \{(x_2, x_3), (x_1, x_2), (x_4, x_5), (x_1, x_5)\}$,
 $L(T) = \{ 1, 2, 4, 5 \}$
- Bước 5. $T = \{(x_2, x_3), (x_1, x_2), (x_4, x_5), (x_1, x_5), (x_5, x_6)\}$
Kết thúc vì $\text{Card}(T) = 5 = 6$ (đỉnh) $- 1$. Tổng chiều dài nhỏ nhất = 19.

Chú ý. Trong thí dụ này, ta tìm lại cây phủ giống như trong thuật toán PRIM. Nhưng, trong trường hợp tổng quát, ta có thể tìm thấy một cây phủ khác nhưng có cùng tổng trọng lượng.

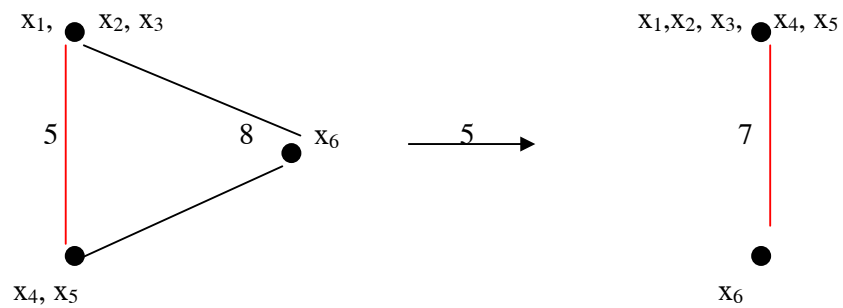
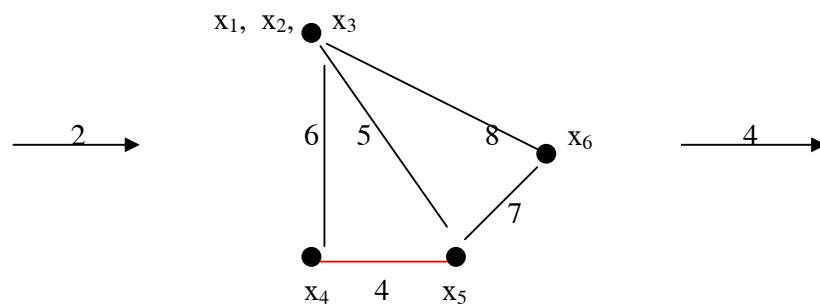
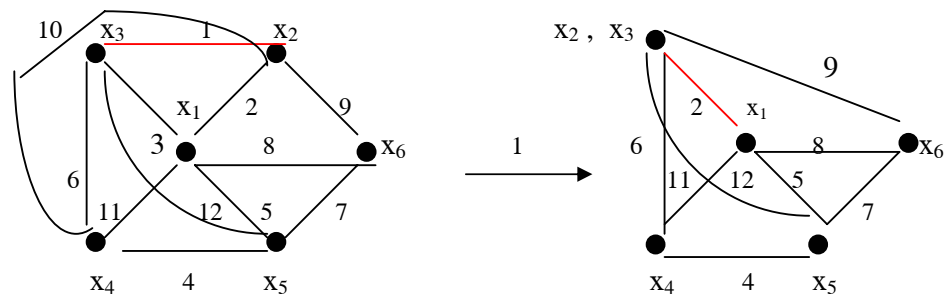


FIG. 2.4. Tìm cây phủ có chiều dài ngắn nhất theo thuật toán KRUSKAL.