

## CHƯƠNG 5. CÂY VÀ CÂY KHUNG CỰC TIỂU

---

### Contents

|  |            |
|--|------------|
| <b>5.1 Cây .....</b>   | <b>89</b>  |
| 5.1.1 Định lý Daisy Chain (Điều kiện cần và đủ) .....                      | 89         |
| 5.1.2 Cây có gốc .....   | 90         |
| 5.1.3 Cây tối đại (Cây khung) .....  | 90         |
| 5.1.3.1 Số lượng cây khung của đồ thị .....                                | 91         |
| 5.1.3.2 Finding (a) spanning tree.....                                     | 91         |
| 5.1.3.3 Đếm số cây khung .....   | 92         |
| <b>5.2 Bài toán cây khung cực tiểu (MST – Minimum spanning tree) .....</b> | <b>94</b>  |
| 5.2.1 Ứng dụng .....   | 94         |
| 5.2.2 Phát biểu bài toán .....   | 95         |
| 5.2.3 Thuật toán Kruskal (Avoid Cycles).....                               | 95         |
| 5.2.4 Thuật toán Prim (Build Tree) .....                                   | 103        |
| <b>5.3 So sánh Prim và Kruskal .....</b>                                   | <b>109</b> |

## 5.1 Cây

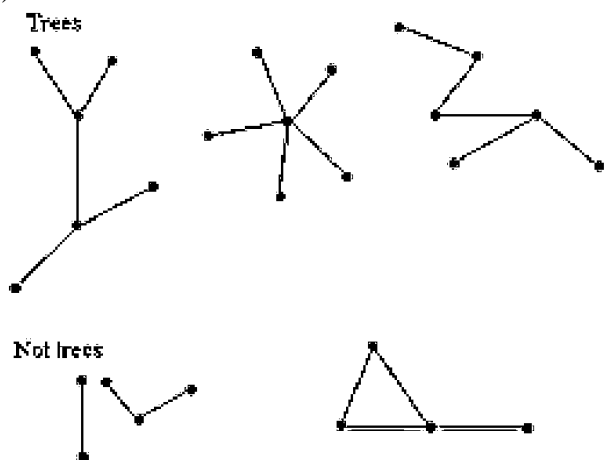
- Cây là một **đơn đồ thị** mà trong đó hai đỉnh bất kì đều được nối với nhau bằng đúng một đường đi. Nói cách khác, **đồ thị liên thông bất kỳ không có chu trình là một cây**.

- Rừng là hợp (disjoint union) của các cây. Rừng có thể có nhiều thành phần liên thông. Mỗi thành phần liên thông là một cây.

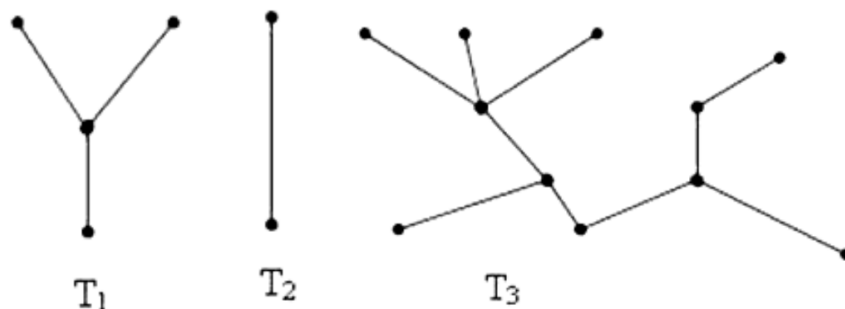
- In addition, a vertex of degree 1 is called a **leaf**.

**Ví dụ:**

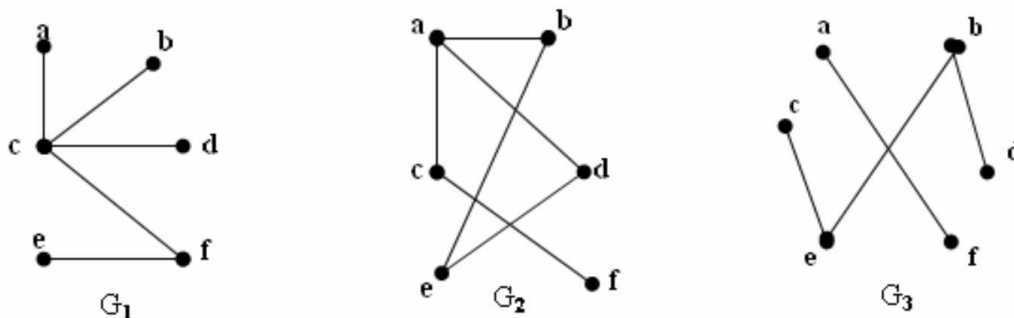
a)



b) Rừng gồm có 3 cây  $T_1, T_2, T_3$ .



c)



$G_1$  là cây;  $G_2$  không là cây (do chứa chu trình);  $G_3$  không là cây (do không liên thông)

- **Chú ý:** Định nghĩa cây hàm ý nói rằng mọi cây đều không chứa khuyên, cũng không chứa cạnh song song.

### 5.1.1 Định lý Daisy Chain (Điều kiện cần và đủ)

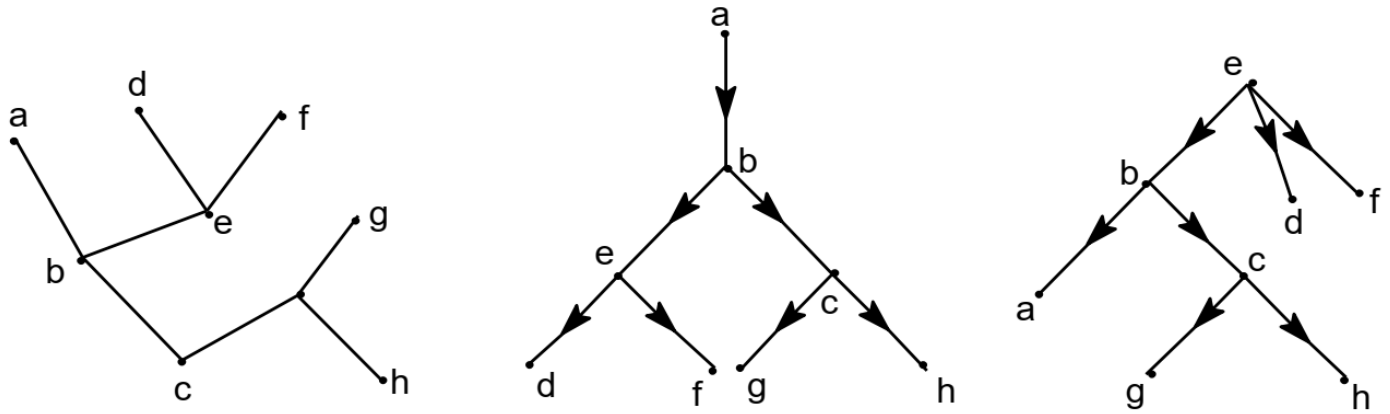
- Cho  $T = (V, E)$  là đồ thị vô hướng có  $n$  đỉnh. Các phát biểu sau đây là tương đương:

- (1)  $T$  là cây ( $T$  liên thông và không chứa chu trình)
- (2)  $T$  không chứa chu trình và có  $n-1$  cạnh
- (3)  $T$  liên thông và có  $n-1$  cạnh
- (4)  $T$  liên thông và mỗi cạnh của nó đều là cầu
- (5) Hai đỉnh bất kỳ của  $T$  được nối với nhau bởi đúng một đường đi đơn
- (6)  $T$  không có chu trình đơn nhưng nếu thêm vào một cạnh giữa hai đỉnh không kề nhau thì lại có một chu trình đơn duy nhất.

### 5.1.2 Cây có gốc

- Một cây với một đỉnh được chọn làm gốc. Định hướng các cạnh trên cây từ gốc đi ra

*Ví dụ*



- Cùng một cây, nếu chọn gốc khác nhau thì cây có gốc thu được sẽ khác nhau

### 5.1.3 Cây tối đại (Cây khung)

- **Định nghĩa:** Giả sử  $G = (V, E)$  là đồ thị vô hướng liên thông.

Cây  $T = (V, F)$  với  $F \subseteq E$  được gọi là cây tối đại (cây phủ, cây bao trùm hay cây khung) của đồ thị  $G$ .

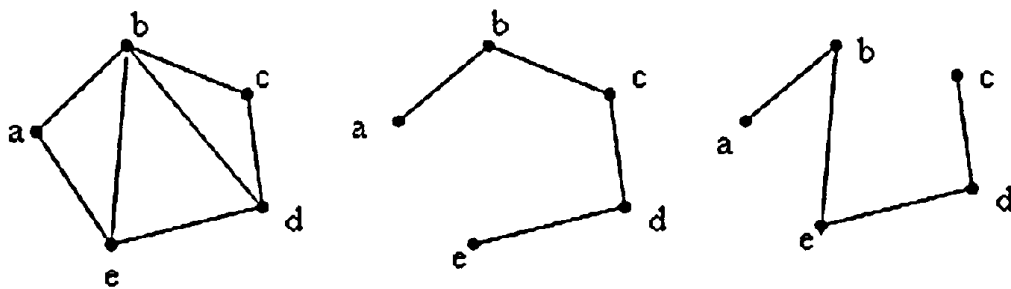
+  $T$  liên thông và không có chu trình.

+  $T$  là đồ thị con của  $G$ , *chứa tất cả các đỉnh của  $G$  (tức là nó "spans" tất cả các đỉnh trong  $G$ )*.

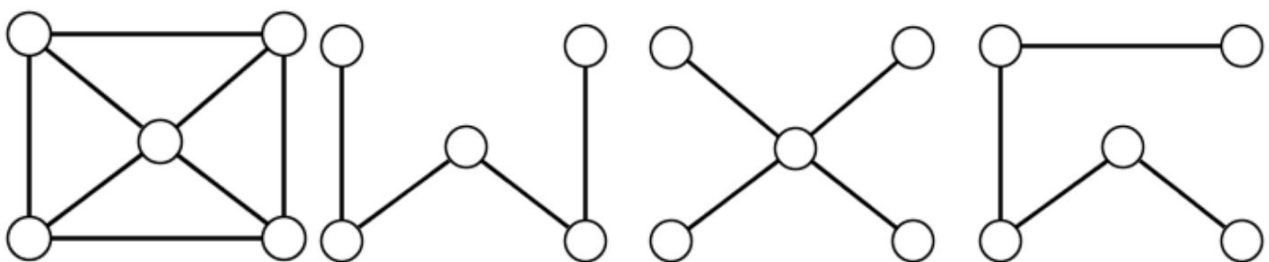
+ Một đồ thị  $G$  có thể có nhiều cây khung.

*Ví dụ:*

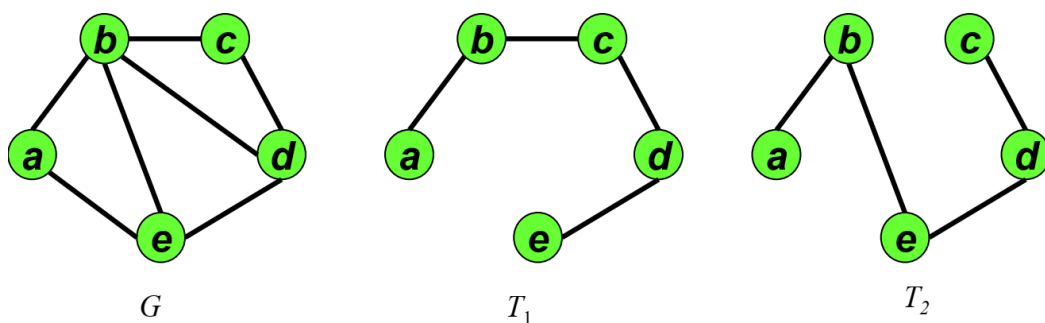
a)



b)



c)



**Lý do tại sao gọi là "cây khung" (spanning tree) là bởi vì:**

- **Cây (Tree):** T là một cây bởi vì nó là đồ thị liên thông không có chu trình.
- **Khung (Spanning):** T bao trùm tất cả các đỉnh của đồ thị G. Điều này có nghĩa là mỗi đỉnh trong G sẽ được bao gồm trong T, và T sẽ "khung" toàn bộ đồ thị G theo nghĩa là nó bao phủ toàn bộ không gian đỉnh của G mà không cần thêm bất kỳ đỉnh nào khác ngoài các đỉnh đã có trong G.

Cụm từ "cây khung" (spanning tree) xuất phát từ việc T là một cây bao trùm (spanning) toàn bộ các đỉnh của đồ thị gốc G. Cây khung đại diện cho một cách kết nối tất cả các đỉnh trong đồ thị gốc mà không có chu trình và chỉ sử dụng số lượng cạnh nhỏ nhất có thể.

### 5.1.3.1 Số lượng cây khung của đồ thị

- Gọi  $t(G)$  là số các cây bao trùm của đồ thị liên thông G.

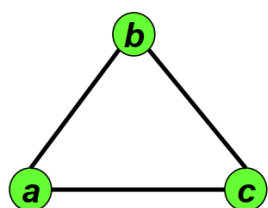
- Trong một số trường hợp, số  $t(G)$  có thể tính trực tiếp. Chẳng hạn nếu G là một cây, khi đó  $t(G)=1$ , còn khi G là một đồ thị vòng  $C_n$  với  $n$  đỉnh thì  $t(G)=n$ .

Với đồ thị G bất kỳ, số  $t(G)$  có tính nhờ [Định lý Kirchhoff](#).

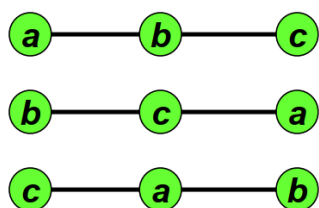
- **Định lý (sự tồn tại của cây tối đại):** Mọi đồ thị liên thông đều chứa ít nhất một cây tối đại.

- **Định lý Cayley:** Số cây khung của đồ thị  $K_n$  là  $n^{n-2}$ .

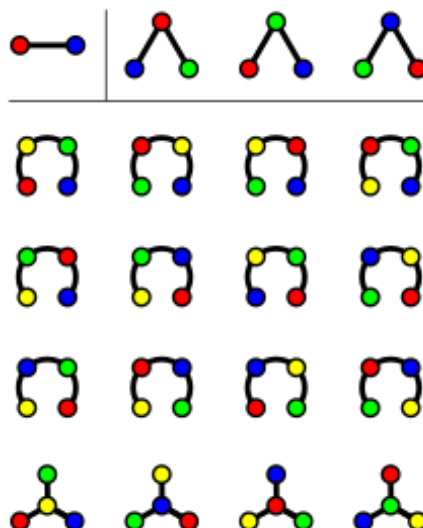
Có tất cả  $2^{2-2}=1$  cây bao trùm của  $K_2$ ,  $3^{3-2}=3$  cây bao trùm của  $K_3$ , và  $4^{4-2}=16$  cây bao trùm của  $K_4$ .



$K_3$



Ba cây khung của  $K_3$



### 5.1.3.2 Finding (a) spanning tree

- When graph G is connected, a depth first or breadth first search starting at any vertex will visit all vertices in G

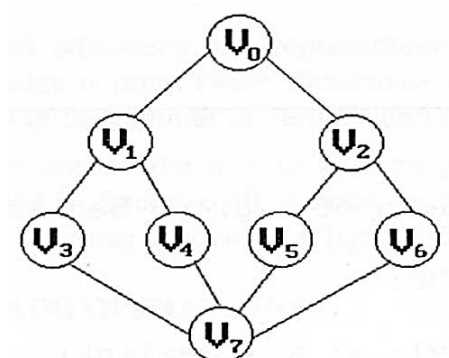
- We may use DFS or BFS to create a spanning tree

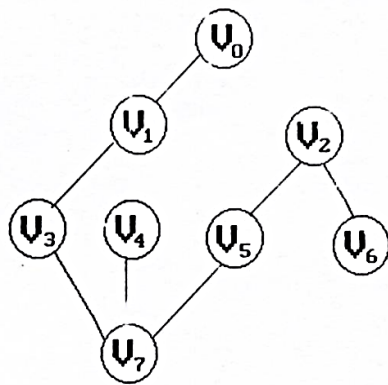
+ Depth first spanning tree when DFS is used

+ Breadth first spanning tree when BFS is used.

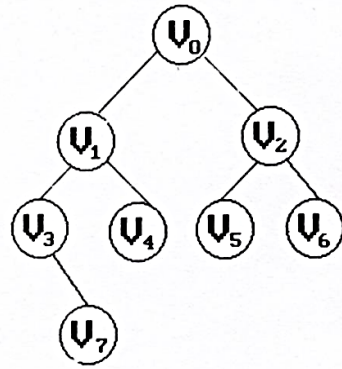
**Ví dụ:**

a)



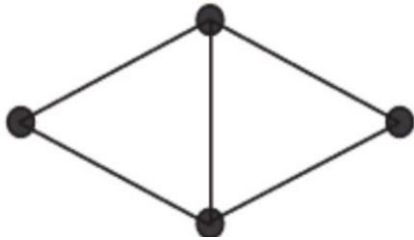


(a) dfs(0) spanning tree

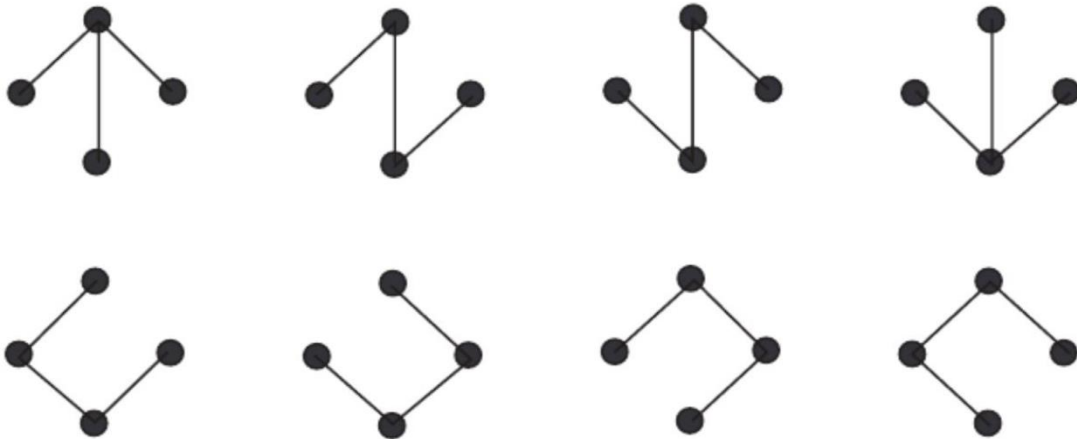


(b) bfs(0) spanning tree

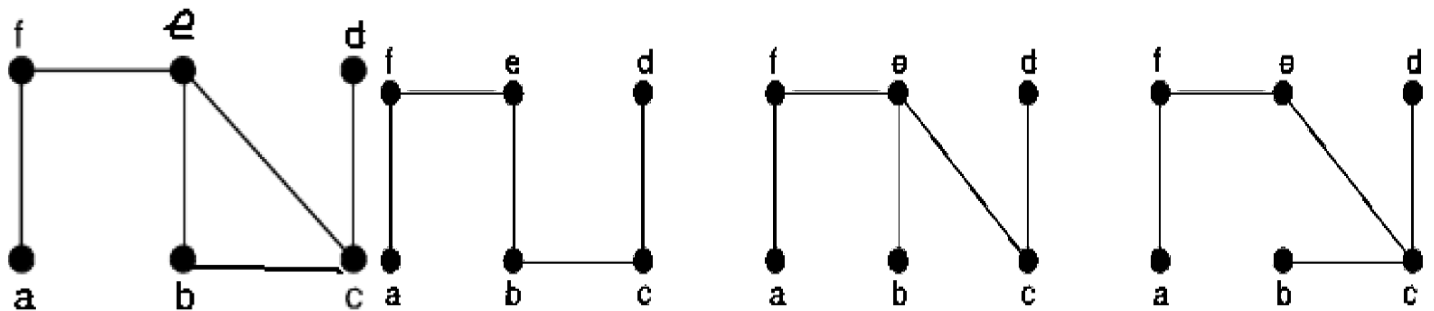
b)



The graph has 8 spanning trees which are shown below:



c)



### 5.1.3.3 Đếm số cây khung

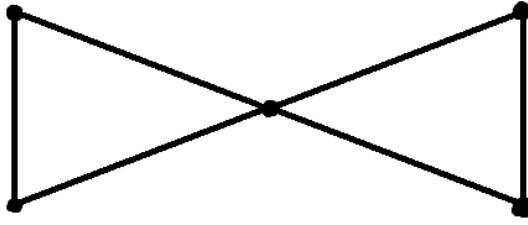
- Áp dụng công thức tính ở phần 5.1.3.1.

- Nếu đồ thị không thuộc các dạng đã nêu, ta tính như sau: <https://www.youtube.com/watch?v=dr8JOxrnBoY>

Mục tiêu để hình thành cây khung từ một đồ thị là ta xóa bỏ các cạnh làm nên chu trình trong đồ thị, và làm sao để số cạnh trong cây đúng bằng  $n-1$  (với  $n$  là số cạnh của đồ thị).

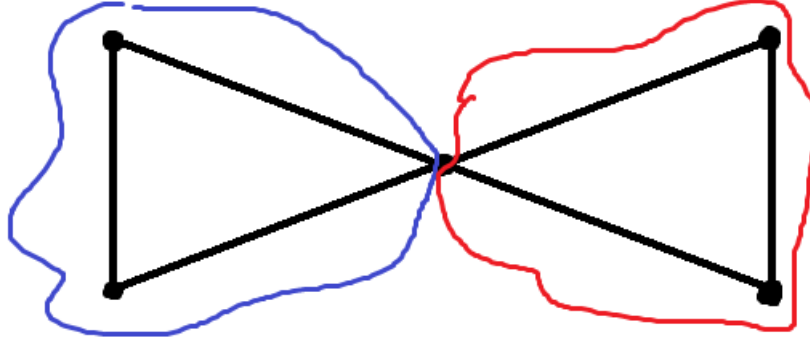
Giả sử có đồ thị như sau:

**Ví dụ 1:**



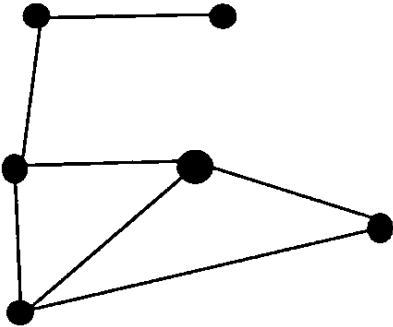
Đồ thị này có 5 đỉnh & 6 cạnh.

- + Vậy mọi cây khung của đồ thị này đều cần có 5 đỉnh, và số cạnh là 4. Tức là ta cần xóa đi 2 cạnh của đồ thị, mà 2 cạnh đó làm nên chu trình.
- + Nhận thấy 2 bên đồ thị đều có chu trình, nên mỗi bên ta xóa đi một cạnh.
- + Vậy số cây khung được tạo ra chính là số cách để xóa đi 2 cạnh làm nên chu trình.

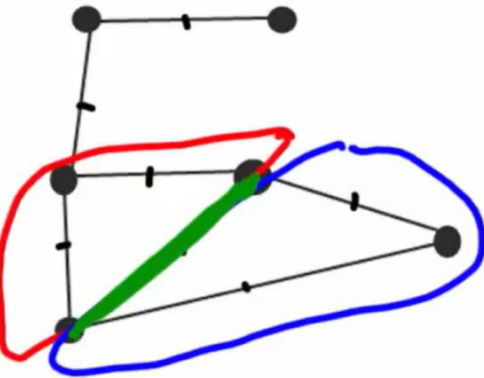


- + Xét bên trái: số cách xóa chu trình = số cách xóa đi một cạnh = 3. Tương tự xét bên phải ta cũng được 3.
- + Theo quy tắc nhân, số cây khung =  $3 \times 3 = 9$ .

**Ví dụ 2:**



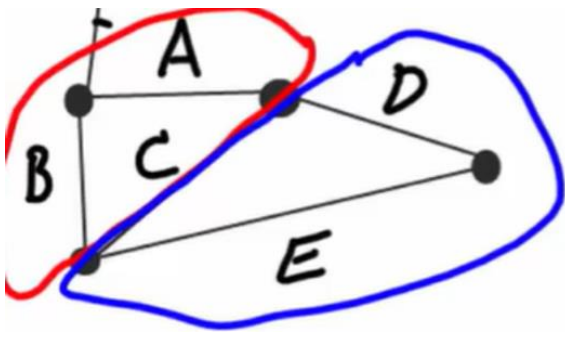
Đồ thị có 6 đỉnh & 7 cạnh. Vậy cây khung cần có 6 đỉnh & 5 cạnh.



Có 2 chu trình xanh và đỏ. Vậy mỗi chu trình ta cần loại bỏ đi 1 cạnh.

Vậy số cách loại bỏ chu trình là:  $3 \times 3 - 1 = 8 =$  số cây khung (-1 là do ta tính cạnh chung 2 lần).

8 trường hợp bỏ đi 2 cạnh:



A, D; B, D; C, D; A, E; B, E; C, E; A, C; B, C.

## 5.2 Bài toán cây khung cực tiểu (MST – Minimum spanning tree)

### 5.2.1 Ứng dụng

- Làm sao xây dựng mạng giao thông nối các thành phố với chi phí **xây dựng và vận hành thấp nhất**?
- Trong lý thuyết mạch, làm thế nào để xây dựng một mạch điện tử có **kích thước, chi phí thấp nhất và tốc độ truyền tín hiệu nhanh nhất**?
- Trong mạng máy tính, đòi hỏi xây dựng hệ thống mạng có **chi phí kết nối thấp nhất và tốc độ truyền dữ liệu cao nhất**?

Công ty truyền thông AT&T cần xây dựng mạng truyền thông kết nối  $n$  khách hàng. Chi phí thực hiện kênh nối  $i$  và  $j$  là  $c_{ij}$ . Hỏi chi phí nhỏ nhất để thực hiện việc kết nối tất cả các khách hàng là bao nhiêu?

Giả thiết là: Chỉ có cách kết nối duy nhất là đặt kênh nối trực tiếp giữa hai nút.

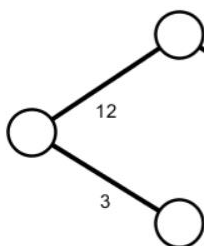
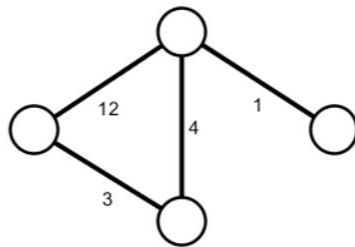
- Bài toán xây dựng hệ thống đường sắt

❖ Giả sử ta muốn xây dựng một hệ thống đường sắt nối  $n$  thành phố sao cho hành khách có thể đi lại giữa hai thành phố bất kỳ đồng thời tổng chi phí xây dựng phải là nhỏ nhất.

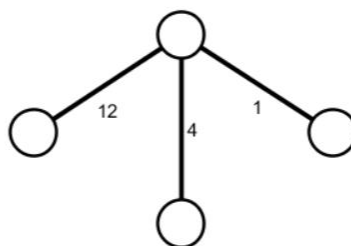
❖ Rõ ràng là đồ thị mà đỉnh là các thành phố còn các cạnh là các tuyến đường sắt nối các thành phố tương ứng với phương án xây dựng tối ưu phải là cây.

❖ Vì vậy, bài toán đặt ra dẫn về bài toán tìm cây khung nhỏ nhất trên đồ thị đầy đủ  $n$  đỉnh, mỗi đỉnh tương ứng với một thành phố, với độ dài trên các cạnh chính là chi phí xây dựng đường ray nối hai thành phố tương ứng

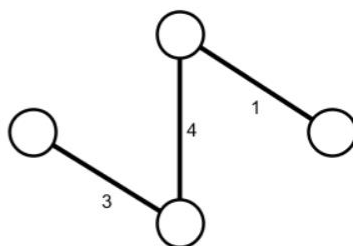
❖ **Chú ý:** Trong bài toán này ta giả thiết là không được xây dựng tuyến đường sắt có các nhà ga phân tuyến nằm ngoài các thành phố.



$C(T)=16$



$C(T)=17$



**$C(T^*)=8$**

### 5.2.2 Phát biểu bài toán

- Cho đồ thị vô hướng, liên thông có trọng số  $G(V, E, C)$ . Tìm cây khung của đồ thị thỏa mãn điều kiện: **Tổng trọng số của nó  $\rightarrow$  MIN.**

- **Why Not Try All Possibilities? (Tại sao phải đi tìm mà không thử tất cả các trường hợp?)**

- Suppose the graph has  $n$  vertices. Then the number of possible spanning trees can be as large as  $n^{n-2}$ .
- When  $n = 75$ , this means that the number of spanning trees can be as large as

7576562804644601479086318651590413464814067833088403392470432810180242799713568047081935219466686248779296875

### 5.2.3 Thuật toán Kruskal (Avoid Cycles)

**Input:**  $G = (V, E)$  liên thông, vô hướng, có trọng số

**Output:** Cây khung nhỏ nhất

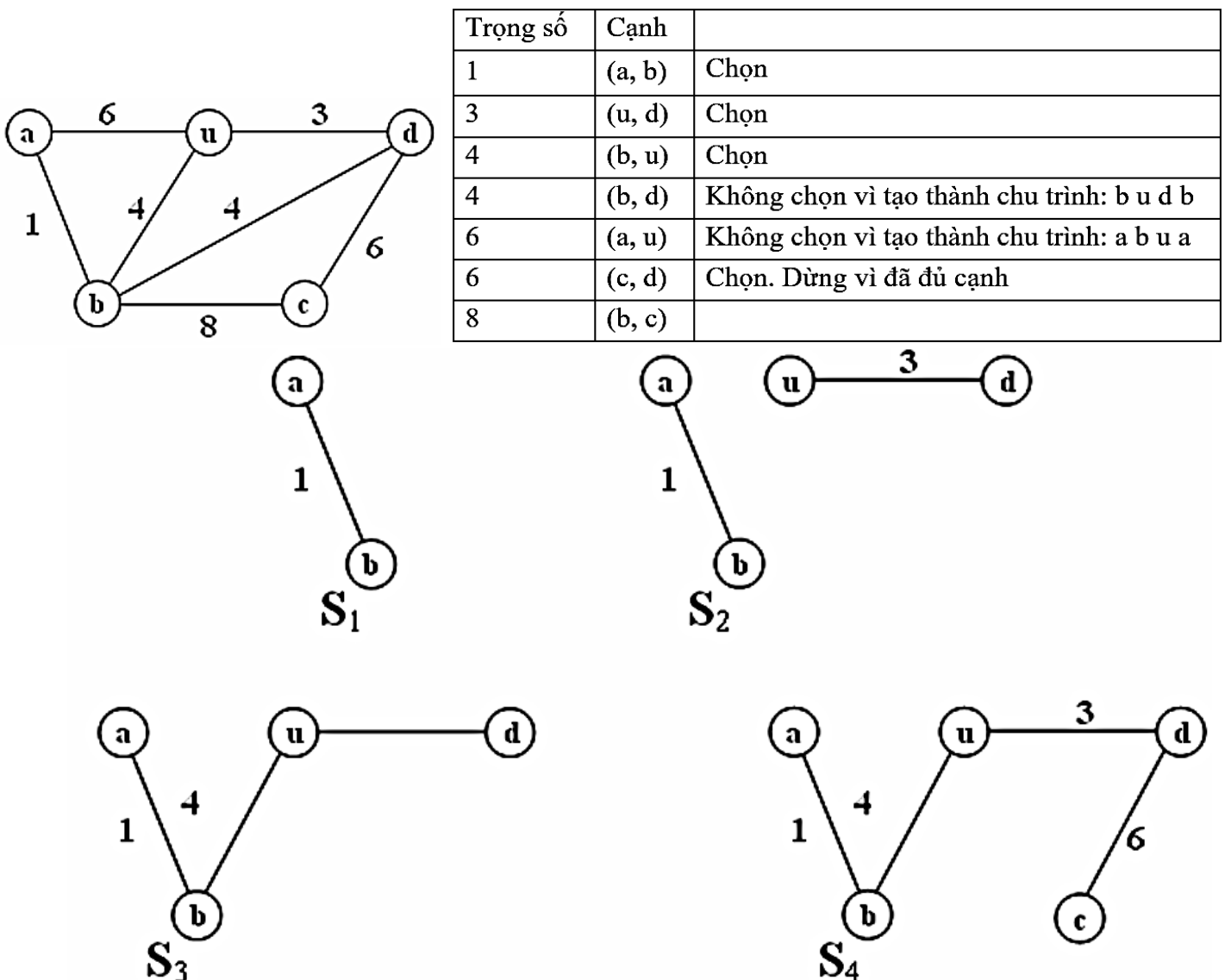
**B1:** Sort các cạnh của  $E$  tăng dần theo trọng số. Đưa vào list, đặt là  $L$ .

**B2:** Lấy một cạnh trong  $L$  thêm vào cây khung cực tiểu sao cho không tạo chu trình. Lặp lại B2 cho tới khi số cạnh của cây khung cực tiểu  $= |V| - 1$ .

• **Làm thế nào để biết việc bổ sung cạnh  $(u,v)$  vào cây  $T$  có tạo ra chu trình hay không?** Gán nhãn (make-set):

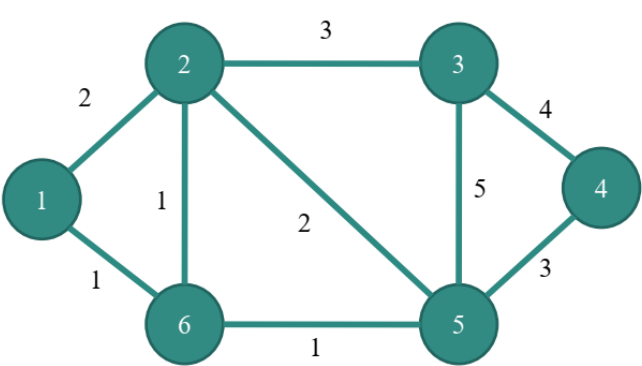
- + Nhãn là một số nguyên được gán cho các đỉnh.
- + Các đỉnh thuộc cùng thành phần liên thông trong  $T$  sẽ có cùng nhãn.
- + Mỗi thành phần liên thông của  $T$  có 1 nhãn khác nhau.

**Ví dụ 1:**



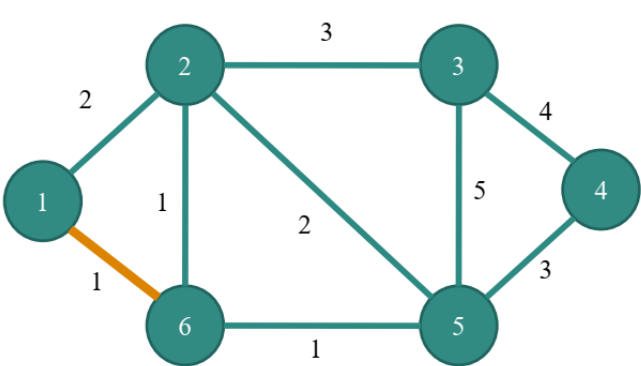


Ví dụ 2:



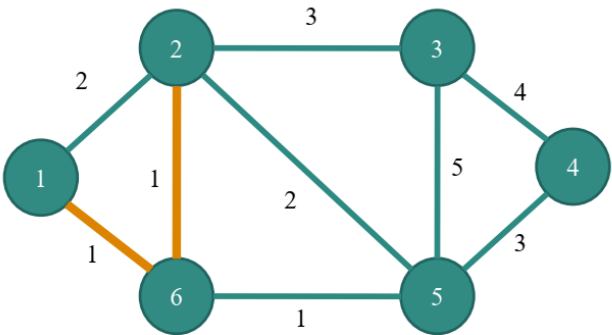
|      |   |   |   |   |   |   |
|------|---|---|---|---|---|---|
| Đỉnh | 1 | 2 | 3 | 4 | 5 | 6 |
| Nhãn | 1 | 2 | 3 | 4 | 5 | 6 |

| Cạnh  | Trọng số | Chọn |
|-------|----------|------|
| (1,6) | 1        |      |
| (2,6) | 1        |      |
| (5,6) | 1        |      |
| (1,2) | 2        |      |
| (2,5) | 2        |      |
| (2,3) | 3        |      |
| (4,5) | 3        |      |
| (3,4) | 4        |      |
| (3,5) | 5        |      |



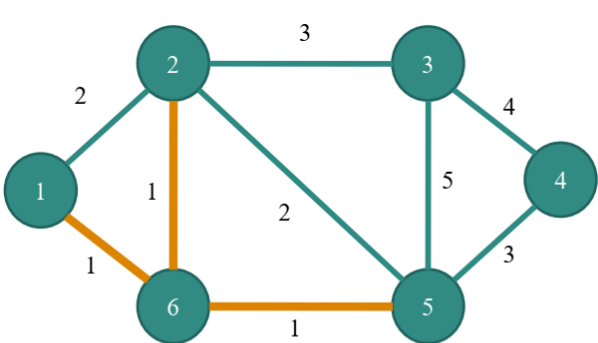
|   |   |   |   |   |   |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 2 | 3 | 4 | 5 | 1 |

| Cạnh  | Trọng số | Chọn |
|-------|----------|------|
| (1,6) | 1        | x    |
| (2,6) | 1        |      |
| (5,6) | 1        |      |
| (1,2) | 2        |      |
| (2,5) | 2        |      |
| (2,3) | 3        |      |
| (4,5) | 3        |      |
| (3,4) | 4        |      |
| (3,5) | 5        |      |



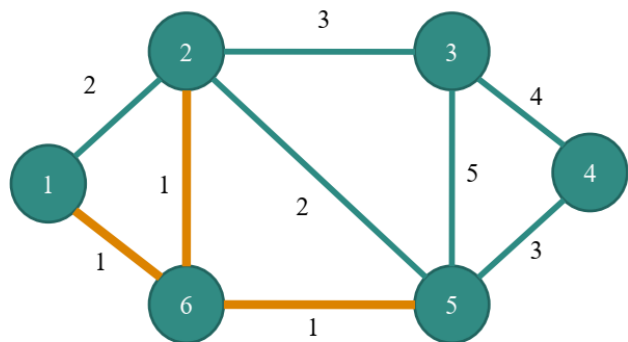
|   |   |   |   |   |   |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 1 | 3 | 4 | 5 | 1 |

| Cạnh  | Trọng số | Chọn |
|-------|----------|------|
| (1,6) | 1        | x    |
| (2,6) | 1        | x    |
| (5,6) | 1        |      |
| (1,2) | 2        |      |
| (2,5) | 2        |      |
| (2,3) | 3        |      |
| (4,5) | 3        |      |
| (3,4) | 4        |      |
| (3,5) | 5        |      |



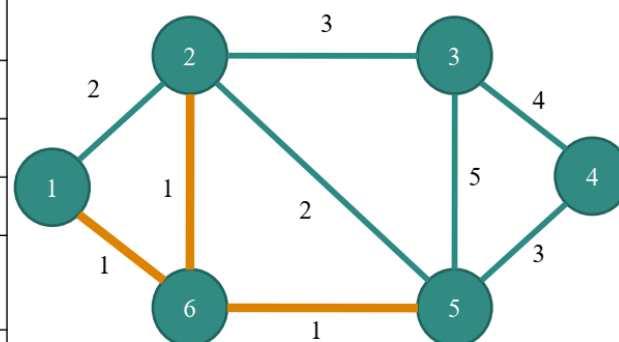
|   |   |   |   |   |   |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 1 | 3 | 4 | 1 | 1 |

| Cạnh  | Trọng số | Chọn |
|-------|----------|------|
| (1,6) | 1        | x    |
| (2,6) | 1        | x    |
| (5,6) | 1        | x    |
| (1,2) | 2        |      |
| (2,5) | 2        |      |
| (2,3) | 3        |      |
| (4,5) | 3        |      |
| (3,4) | 4        |      |
| (3,5) | 5        |      |



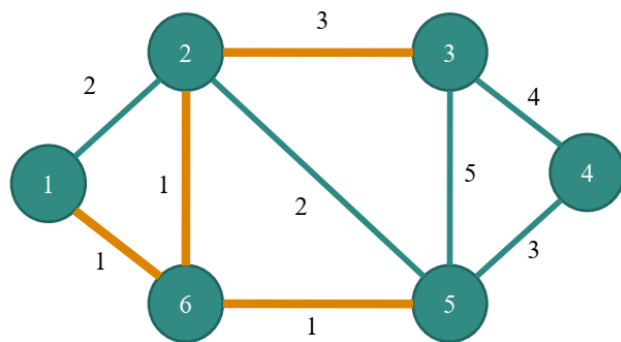
| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|
| 1 | 1 | 3 | 4 | 1 | 1 |

| Cạnh  | Trọng số | Chọn          |
|-------|----------|---------------|
| (1,6) | 1        | x             |
| (2,6) | 1        | x             |
| (5,6) | 1        | x             |
| (1,2) | 2        | tạo chu trình |
| (2,5) | 2        |               |
| (2,3) | 3        |               |
| (4,5) | 3        |               |
| (3,4) | 4        |               |
| (3,5) | 5        |               |



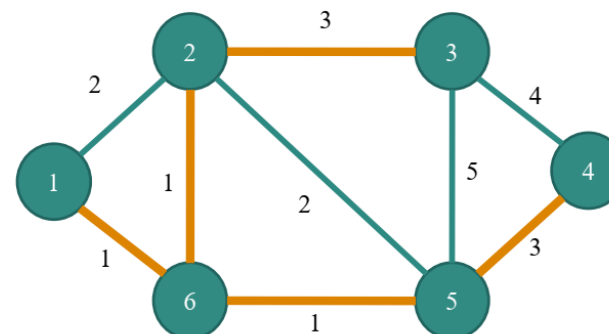
| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|
| 1 | 1 | 3 | 4 | 1 | 1 |

| Cạnh  | Trọng số | Chọn          |
|-------|----------|---------------|
| (1,6) | 1        | x             |
| (2,6) | 1        | x             |
| (5,6) | 1        | x             |
| (1,2) | 2        | tạo chu trình |
| (2,5) | 2        | tạo chu trình |
| (2,3) | 3        |               |
| (4,5) | 3        |               |
| (3,4) | 4        |               |
| (3,5) | 5        |               |



| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 4 | 1 | 1 |

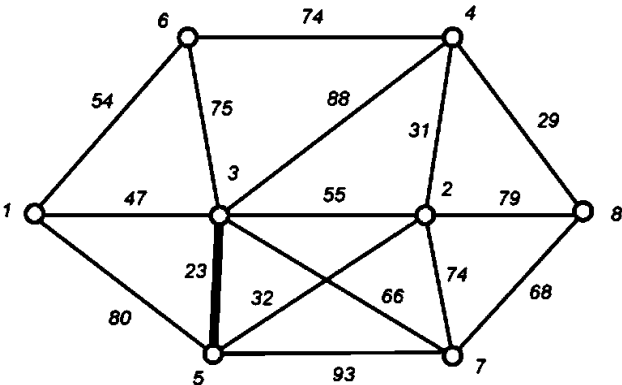
| Cạnh  | Trọng số | Chọn          |
|-------|----------|---------------|
| (1,6) | 1        | x             |
| (2,6) | 1        | x             |
| (5,6) | 1        | x             |
| (1,2) | 2        | tạo chu trình |
| (2,5) | 2        | tạo chu trình |
| (2,3) | 3        | x             |
| (4,5) | 3        |               |
| (3,4) | 4        |               |
| (3,5) | 5        |               |



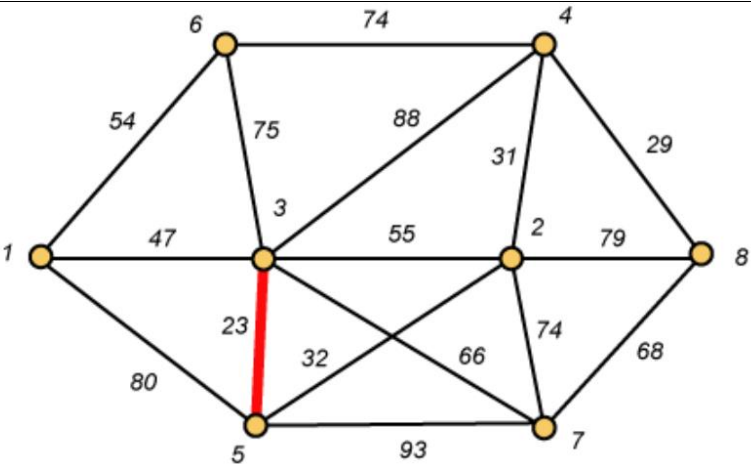
| 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 | 1 |

| Cạnh  | Trọng số | Chọn          |
|-------|----------|---------------|
| (1,6) | 1        | x             |
| (2,6) | 1        | x             |
| (5,6) | 1        | x             |
| (1,2) | 2        | tạo chu trình |
| (2,5) | 2        | tạo chu trình |
| (2,3) | 3        | x             |
| (4,5) | 3        | x             |
| (3,4) | 4        |               |
| (3,5) | 5        |               |

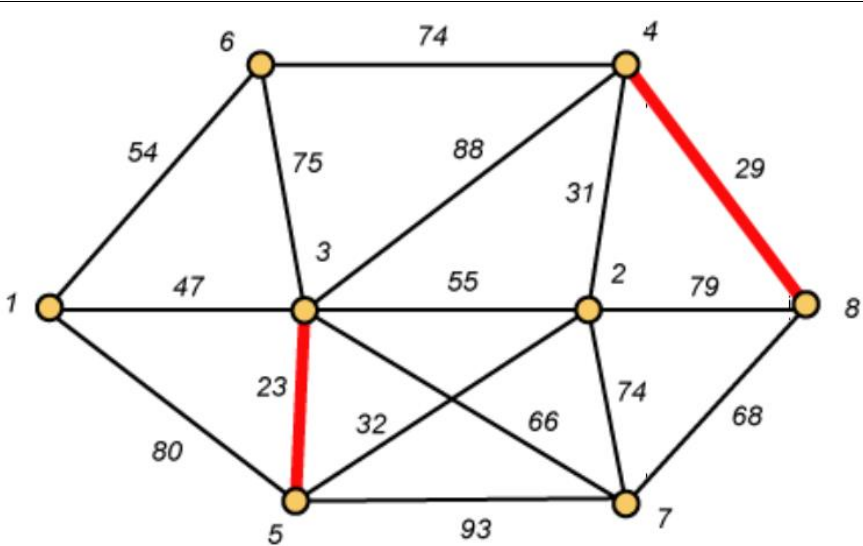
Ví dụ 3:



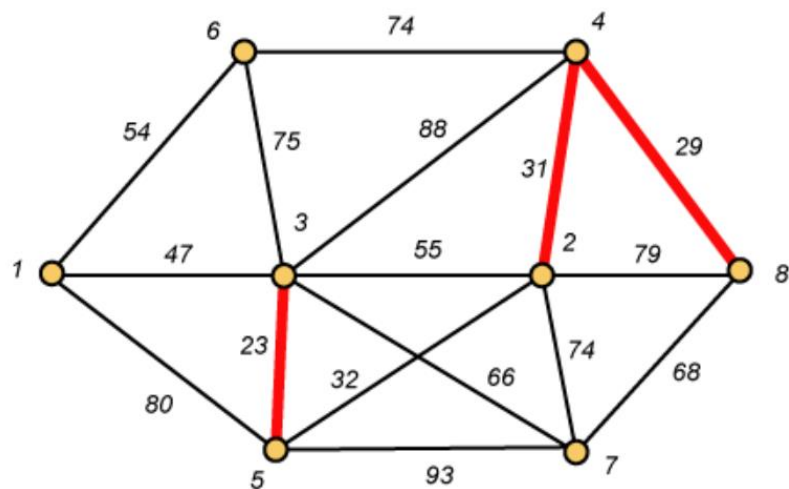
|    |     |   |
|----|-----|---|
| 23 | 3,5 | x |
| 29 | 4,8 |   |
| 31 | 2,4 |   |
| 32 | 2,5 |   |
| 47 | 1,3 |   |
| 54 | 1,6 |   |
| 55 | 2,3 |   |
| 66 | 3,7 |   |
| 68 | 7,8 |   |
| 74 | 4,6 |   |
| 74 | 2,7 |   |
| 75 | 3,6 |   |
| 79 | 2,8 |   |
| 80 | 1,5 |   |
| 88 | 3,4 |   |
| 93 | 5,7 |   |



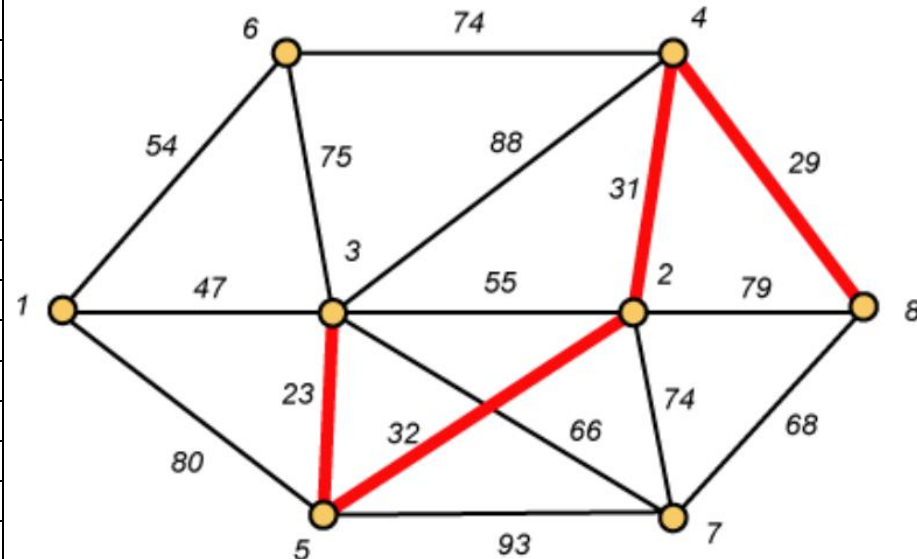
|    |     |   |
|----|-----|---|
| 23 | 3,5 | x |
| 29 | 4,8 | x |
| 31 | 2,4 |   |
| 32 | 2,5 |   |
| 47 | 1,3 |   |
| 54 | 1,6 |   |
| 55 | 2,3 |   |
| 66 | 3,7 |   |
| 68 | 7,8 |   |
| 74 | 4,6 |   |
| 74 | 2,7 |   |
| 75 | 3,6 |   |
| 79 | 2,8 |   |
| 80 | 1,5 |   |
| 88 | 3,4 |   |
| 93 | 5,7 |   |



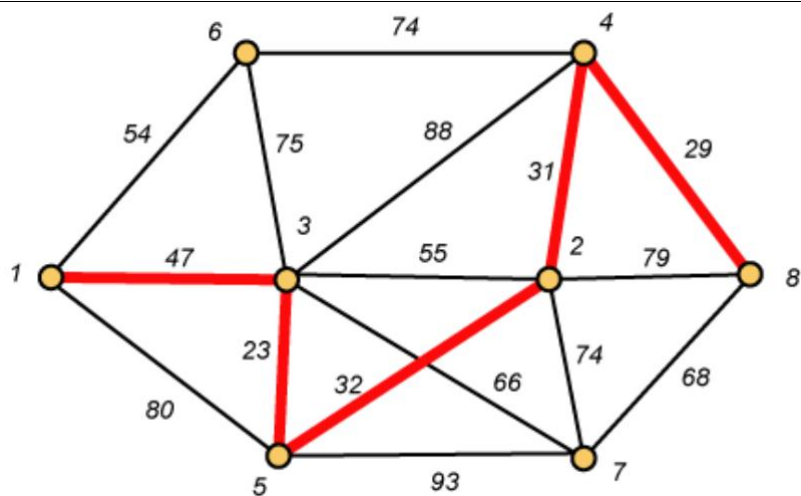
|    |     |   |
|----|-----|---|
| 23 | 3,5 | x |
| 29 | 4,8 | x |
| 31 | 2,4 | x |
| 32 | 2,5 |   |
| 47 | 1,3 |   |
| 54 | 1,6 |   |
| 55 | 2,3 |   |
| 66 | 3,7 |   |
| 68 | 7,8 |   |
| 74 | 4,6 |   |
| 74 | 2,7 |   |
| 75 | 3,6 |   |
| 79 | 2,8 |   |
| 80 | 1,5 |   |
| 88 | 3,4 |   |
| 93 | 5,7 |   |



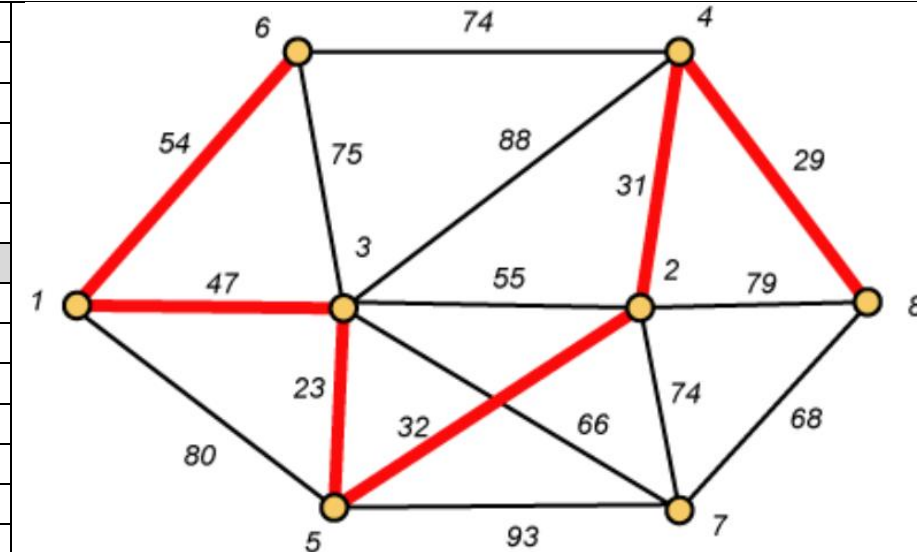
|    |     |   |
|----|-----|---|
| 23 | 3,5 | x |
| 29 | 4,8 | x |
| 31 | 2,4 | x |
| 32 | 2,5 | x |
| 47 | 1,3 |   |
| 54 | 1,6 |   |
| 55 | 2,3 |   |
| 66 | 3,7 |   |
| 68 | 7,8 |   |
| 74 | 4,6 |   |
| 74 | 2,7 |   |
| 75 | 3,6 |   |
| 79 | 2,8 |   |
| 80 | 1,5 |   |
| 88 | 3,4 |   |
| 93 | 5,7 |   |



|    |     |   |
|----|-----|---|
| 23 | 3,5 | x |
| 29 | 4,8 | x |
| 31 | 2,4 | x |
| 32 | 2,5 | x |
| 47 | 1,3 | x |
| 54 | 1,6 |   |
| 55 | 2,3 |   |
| 66 | 3,7 |   |
| 68 | 7,8 |   |
| 74 | 4,6 |   |
| 74 | 2,7 |   |
| 75 | 3,6 |   |
| 79 | 2,8 |   |
| 80 | 1,5 |   |
| 88 | 3,4 |   |
| 93 | 5,7 |   |

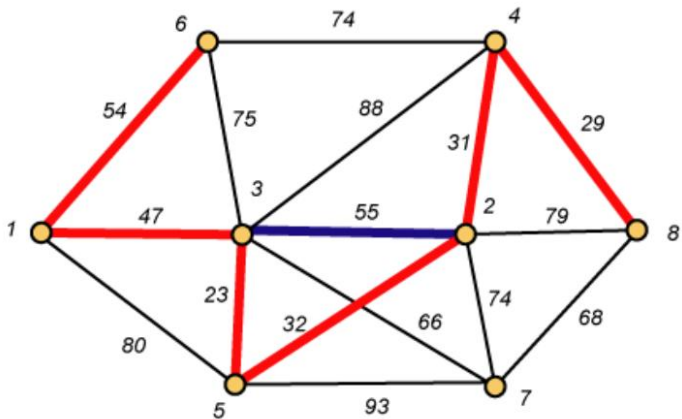


|    |     |   |
|----|-----|---|
| 23 | 3,5 | x |
| 29 | 4,8 | x |
| 31 | 2,4 | x |
| 32 | 2,5 | x |
| 47 | 1,3 | x |
| 54 | 1,6 | x |
| 55 | 2,3 |   |
| 66 | 3,7 |   |
| 68 | 7,8 |   |
| 74 | 4,6 |   |
| 74 | 2,7 |   |
| 75 | 3,6 |   |
| 79 | 2,8 |   |
| 80 | 1,5 |   |
| 88 | 3,4 |   |
| 93 | 5,7 |   |

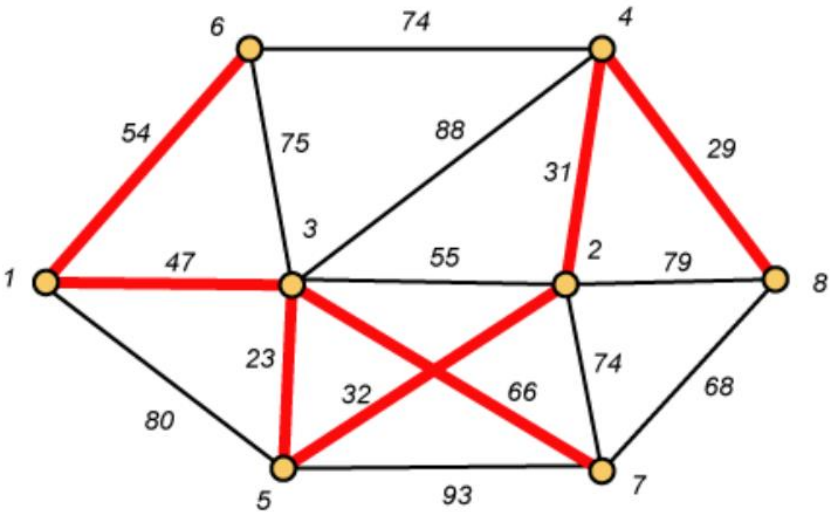


Up to this point, we have simply taken the edges in order of their weight. But now we will have to reject an edge since it forms a cycle when added to those already chosen.

So we cannot take the blue edge having weight 55.



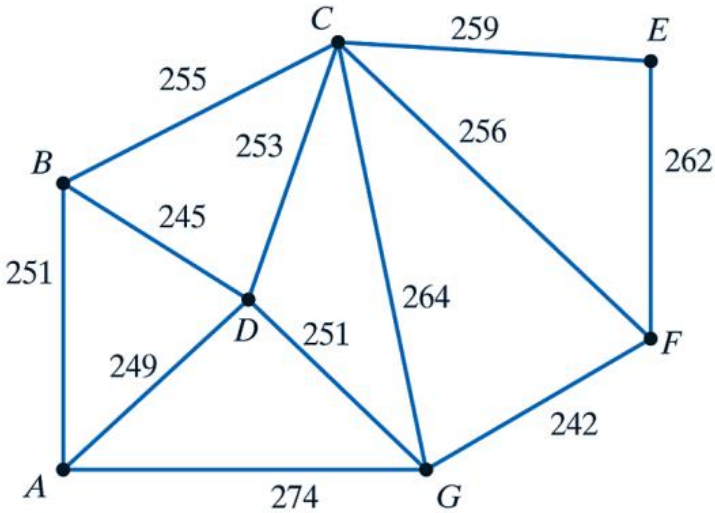
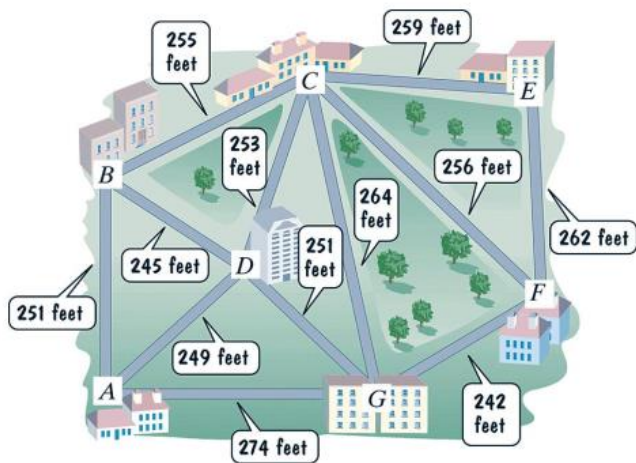
|    |     |   |
|----|-----|---|
| 23 | 3,5 | x |
| 29 | 4,8 | x |
| 31 | 2,4 | x |
| 32 | 2,5 | x |
| 47 | 1,3 | x |
| 54 | 1,6 | x |
| 55 | 2,3 |   |
| 66 | 3,7 | x |
| 68 | 7,8 |   |
| 74 | 4,6 |   |
| 74 | 2,7 |   |
| 75 | 3,6 |   |
| 79 | 2,8 |   |
| 80 | 1,5 |   |
| 88 | 3,4 |   |
| 93 | 5,7 |   |



$$\text{Weight (T)} = 23 + 29 + 31 + 32 + 47 + 54 + 66 = \mathbf{282}$$

**Ví dụ 4:**

❖ Seven buildings on a college are connected by the sidewalks shown in the figure.



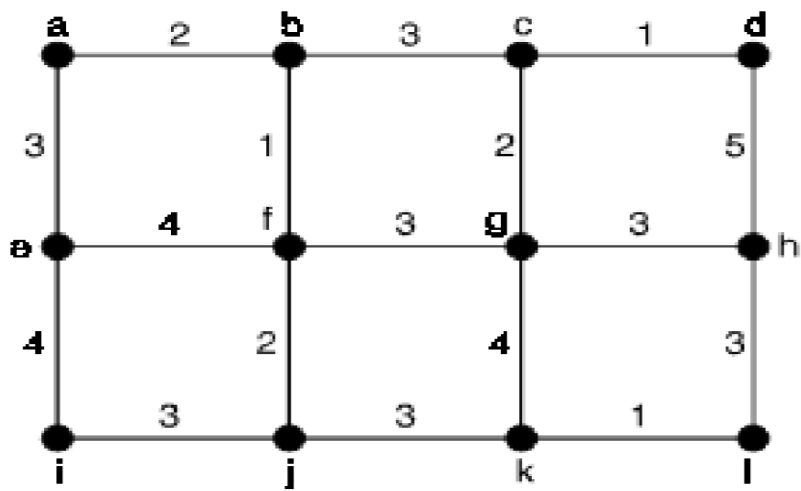
- ❖ The weighted graph represents buildings as vertices, sidewalks as edges, and sidewalk lengths as weights.
- ❖ A heavy snow has fallen and the sidewalks need to be cleared quickly. Campus decides to clear as little as possible and still ensure that students walking from building to building will be able to do so along cleared paths.
- ❖ Determine the shortest series of sidewalks to clear. What is the total length of the sidewalks that need to be cleared?

Using Kruskal’s Algorithm, we complete minimizing the spanning tree in a series of steps given below. Refer to the next graph coinciding with the steps.

|     |    |                |
|-----|----|----------------|
| 242 | GF | Step: 1        |
| 245 | BD | 2              |
| 249 | AD | 3              |
| 241 | AB | Make circuit   |
| 251 | DG | 4              |
| 253 | CD | 5              |
| 255 | BC | Make circuit   |
| 256 | CF | Make circuit   |
| 259 | CE | 6              |
| 262 | EF | Full of edges. |
| 274 | AG |                |

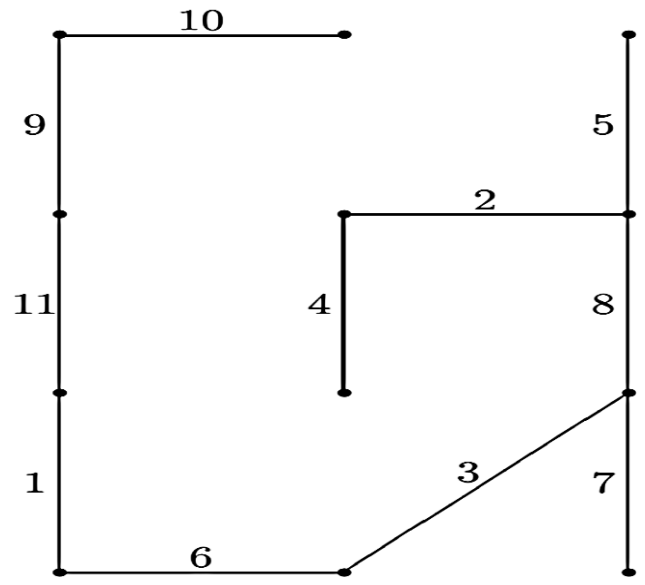
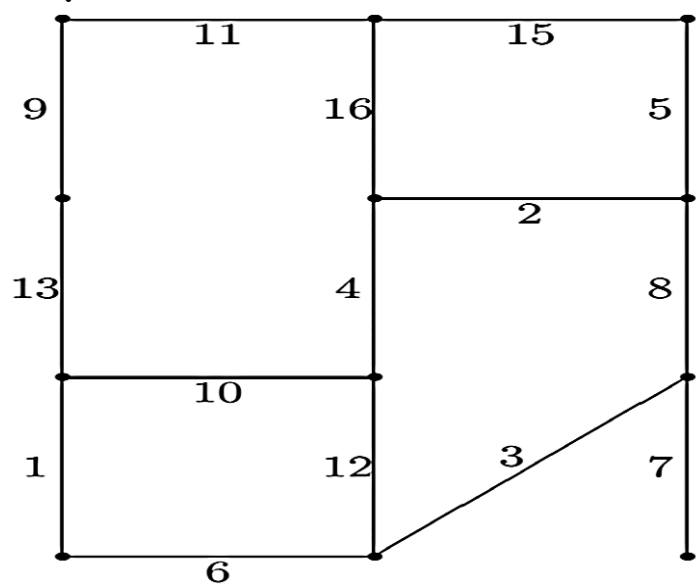
- ❖ **Step 1.** Find the edge with the smallest weight. Select edge  $GF$  by marking it in red.
- ❖ **Step 2.** Find the next-smallest edge in the graph. Select  $BD$  by marking it in red.
- ❖ **Step 3.** Find the next-smallest edge in the graph. Select  $AD$  by marking it in red.
- ❖ **Step 4.** Find the next-smallest edge in the graph that does not create a circuit. Select  $DG$ , since it does not create a circuit, by marking it in red.
- ❖ **Step 5.** Find the next-smallest edge in the graph that does not create a circuit. Select  $CD$ , since it does not create a circuit, by marking it in red.
- ❖ **Step 6.** Find the next-smallest edge in the graph that does not create a circuit. Select  $CE$ , since it does not create a circuit, by marking it in red.
- ❖ The minimum spanning tree is completed. The red subgraph contains all seven vertices, six edges, and no circuits.
- ❖ The total length of the sidewalks that need to be cleared is  $242 + 245 + 249 + 251 + 253 + 259$ , or 1499 feet.

**Ví dụ 5:**



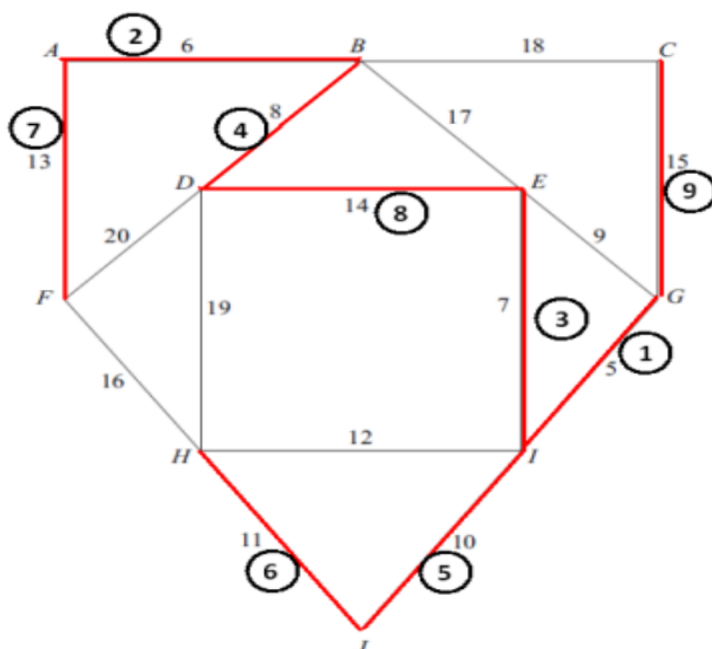
| Choice | Edge       | Weight |
|--------|------------|--------|
| 1      | $\{c, d\}$ | 1      |
| 2      | $\{k, l\}$ | 1      |
| 3      | $\{b, f\}$ | 1      |
| 4      | $\{c, g\}$ | 2      |
| 5      | $\{a, b\}$ | 2      |
| 6      | $\{f, j\}$ | 2      |
| 7      | $\{b, c\}$ | 3      |
| 8      | $\{j, k\}$ | 3      |
| 9      | $\{g, h\}$ | 3      |
| 10     | $\{i, j\}$ | 3      |
| 11     | $\{a, e\}$ | 3      |
| Total: |            | 24     |

Ví dụ 6:

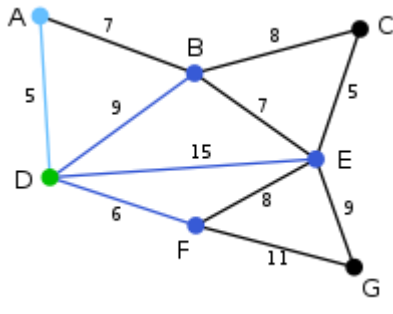
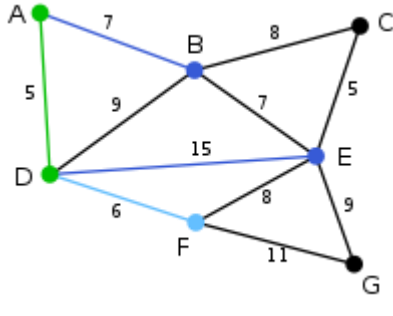
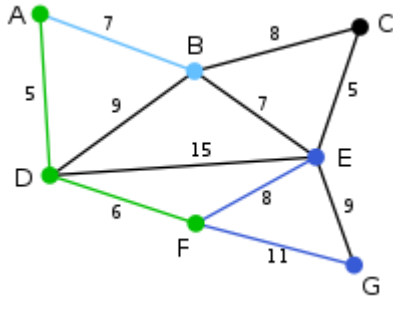
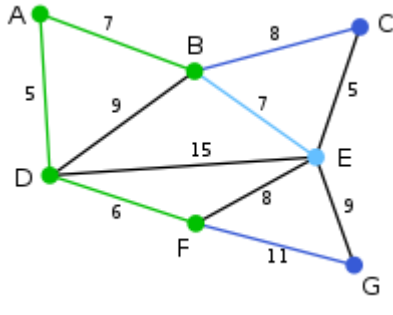
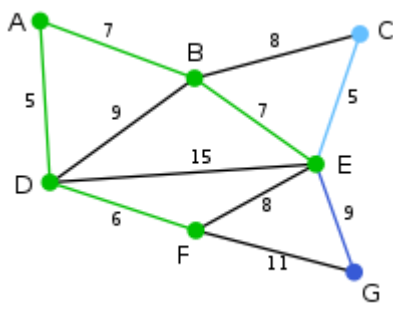


Ví dụ 7:



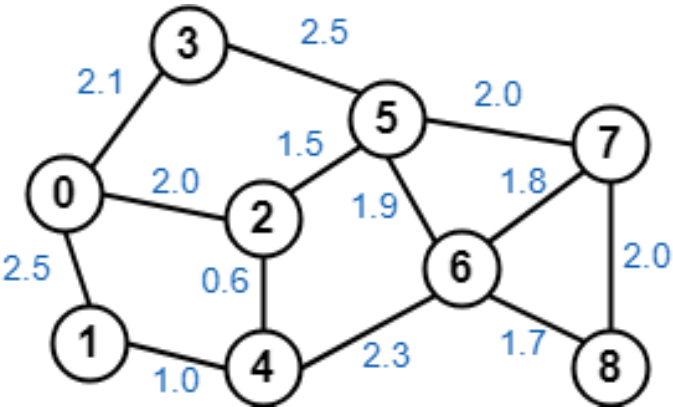




|  |             |  |               |  |
|--|-------------|--|---------------|--|
|     | {D}         | (D,A) = 5 <b>V</b><br>(D,B) = 9<br>(D,E) = 15<br>(D,F) = 6   | {A,B,C,E,F,G} | Chọn một cách tùy ý đỉnh <b>D</b> là đỉnh bắt đầu. Các đỉnh <b>A, B, E</b> và <b>F</b> đều được nối trực tiếp tới <b>D</b> bằng cạnh của đồ thị. <b>A</b> là đỉnh gần <b>D</b> nhất nên ta chọn <b>A</b> là đỉnh thứ hai của cây và thêm cạnh <b>AD</b> vào cây.   |
|    | {A,D}       | (D,B) = 9<br>(D,E) = 15<br>(D,F) = 6 <b>V</b><br>(A,B) = 7   | {B,C,E,F,G}   | Đỉnh được chọn tiếp theo là đỉnh gần <b>D</b> hoặc <b>A</b> nhất. <b>B</b> có khoảng cách tới <b>D</b> bằng 9 và tới <b>A</b> bằng 7, <b>E</b> có khoảng cách tới cây hiện tại bằng 15, và <b>F</b> có khoảng cách bằng 6. <b>F</b> là đỉnh gần cây hiện tại nhất nên chọn đỉnh <b>F</b> và cạnh <b>DF</b> . |
|   | {A,D,F}     | (D,B) = 9<br>(D,E) = 15<br>(A,B) = 7 <b>V</b><br>(F,E) = 8<br>(F,G) = 11   | {B,C,E,G}     | Thuật toán tiếp tục tương tự như bước trước. Chọn đỉnh <b>B</b> có khoảng cách tới <b>A</b> bằng 7.  |
|  | {A,B,D,F}   | (B,C) = 8<br>(B,E) = 7 <b>V</b><br>(D,B) = 9 chu trình<br>(D,E) = 15<br>(F,E) = 8<br>(F,G) = 11                                  | {C,E,G}       | Ở bước này ta chọn giữa <b>C, E</b> , và <b>G</b> . <b>C</b> có khoảng cách tới <b>B</b> bằng 8, <b>E</b> có khoảng cách tới <b>B</b> bằng 7, và <b>G</b> có khoảng cách tới <b>F</b> bằng 11. <b>E</b> là đỉnh gần nhất, nên chọn đỉnh <b>E</b> và cạnh <b>BE</b> .   |
|  | {A,B,D,E,F} | (B,C) = 8<br>(D,B) = 9 chu trình<br>(D,E) = 15 chu trình<br>(E,C) = 5 <b>V</b><br>(E,G) = 9<br>(F,E) = 8 chu trình<br>(F,G) = 11 | {C,G}         | Ở bước này ta chọn giữa <b>C</b> và <b>G</b> . <b>C</b> có khoảng cách tới <b>E</b> bằng 5, và <b>G</b> có khoảng cách tới <b>E</b> bằng 9. Chọn <b>C</b> và cạnh <b>EC</b> .  |

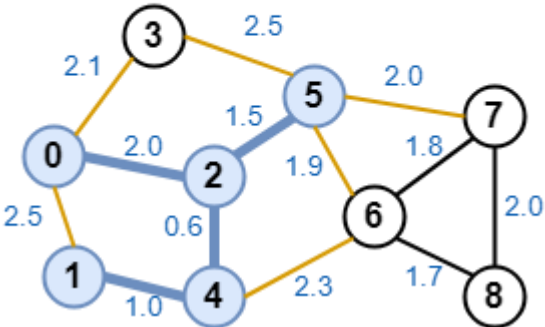
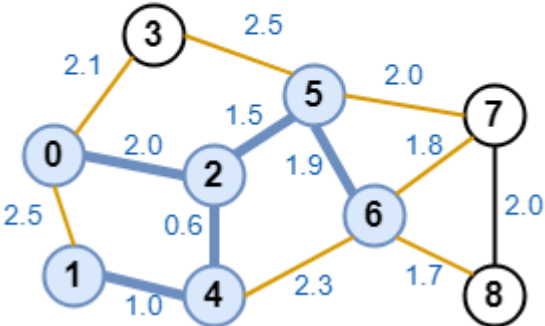
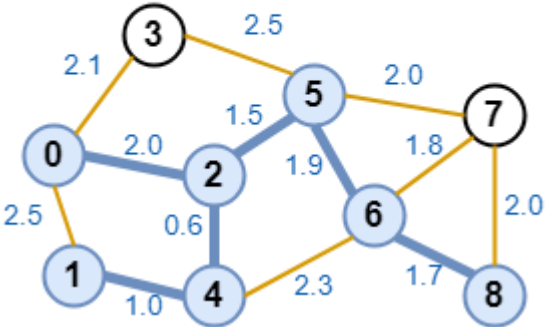
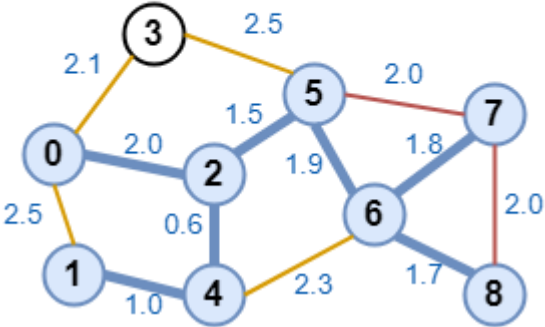
|  |                 |   |     |   |
|--|-----------------|---|-----|---|
|  | {A,B,C,D,E,F}   | (B,C) = 8 chu trình<br>(D,B) = 9 chu trình<br>(D,E) = 15 chu trình<br>(E,G) = 9 <b>V</b><br>(F,E) = 8 chu trình<br>(F,G) = 11 | {G} | Đỉnh <b>G</b> là đỉnh còn lại duy nhất. Nó có khoảng cách tới <b>F</b> bằng 11, và khoảng cách tới <b>E</b> bằng 9. <b>E</b> ở gần hơn nên chọn đỉnh <b>G</b> và cạnh <b>EG</b> . |
|  | {A,B,C,D,E,F,G} | (B,C) = 8 chu trình<br>(D,B) = 9 chu trình<br>(D,E) = 15 chu trình<br>(F,E) = 8 chu trình<br>(F,G) = 11 chu trình             | {}  | Hiện giờ tất cả các đỉnh đã nằm trong cây và <b>cây bao trùm nhỏ nhất</b> được tô màu xanh lá cây. Tổng trọng số của cây là 39.   |

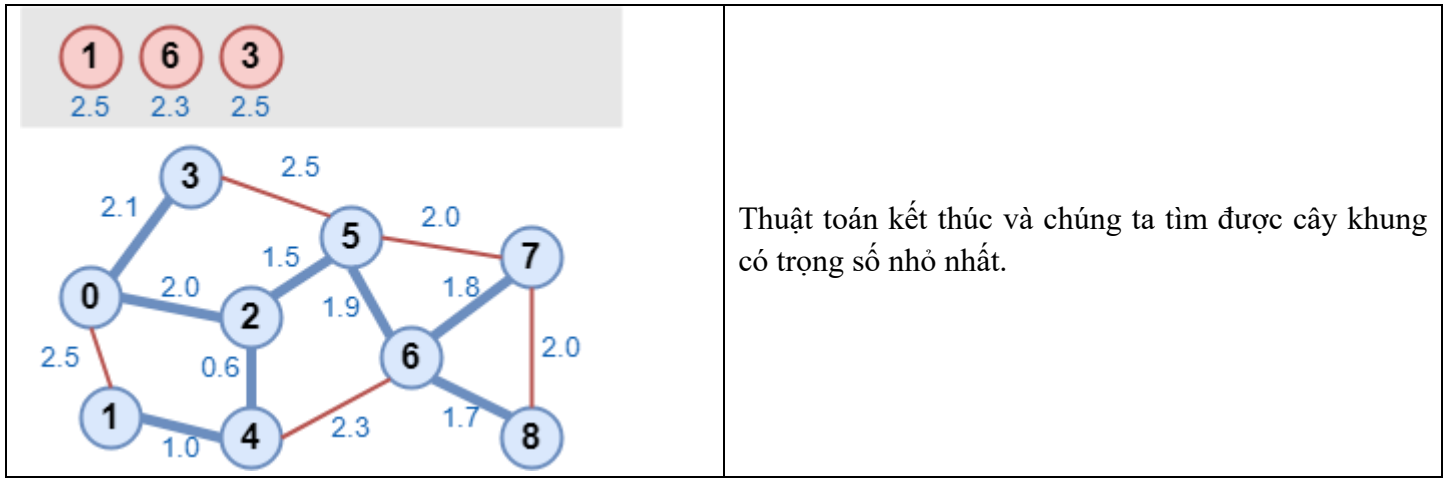
Ví dụ 2:



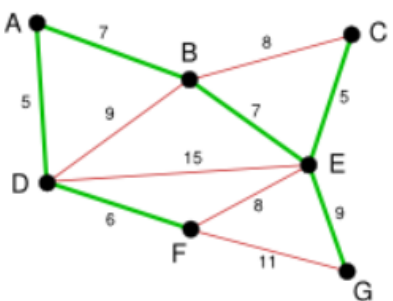
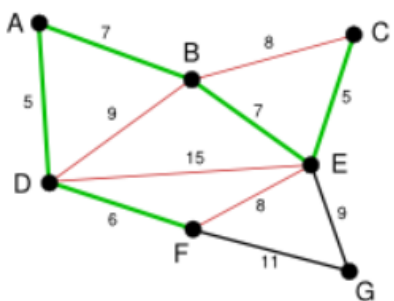
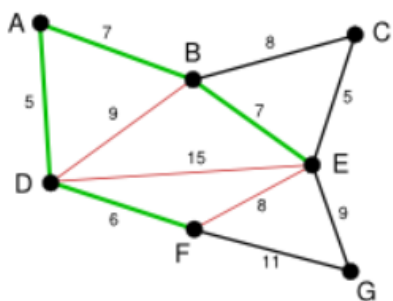
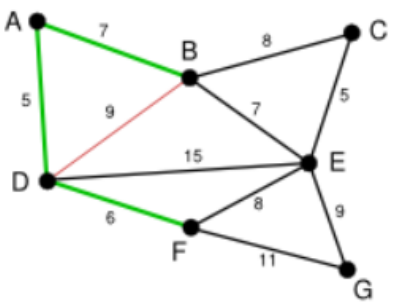
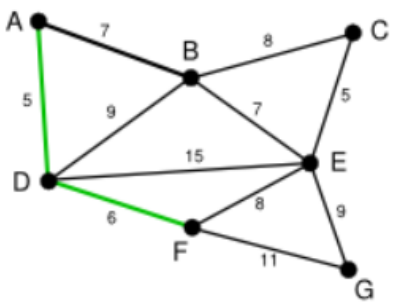
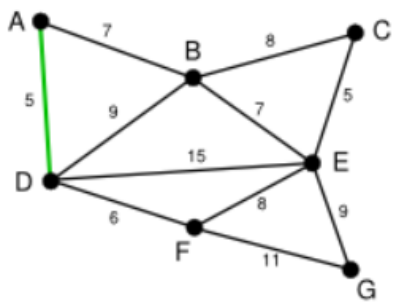
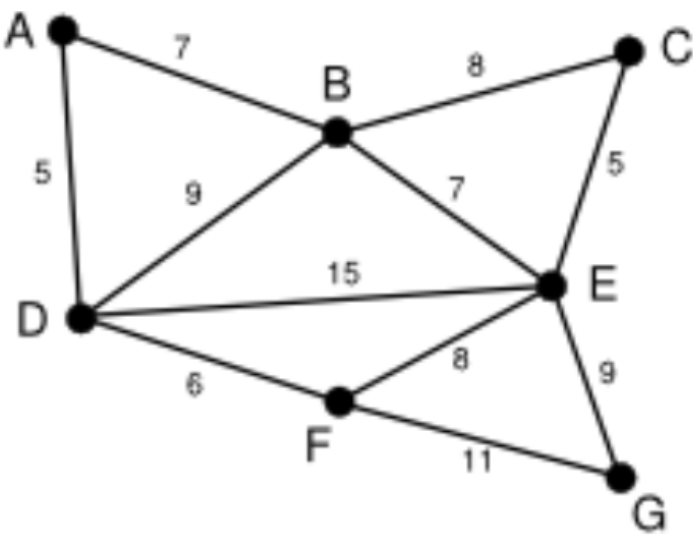
|  |  |
|--|--|
| <div> <div> <div>1</div> <div>2</div> <div>3</div> </div> <div> <div>2.5</div> <div>2.0</div> <div>2.1</div> </div> </div> | <p>Chọn đỉnh 0 làm đỉnh bắt đầu, đưa đỉnh này vào cây khung (màu xanh). Đưa các cạnh 0-1, 0-2, 0-3 vào danh sách cạnh đang xét (màu cam). Ta thấy cạnh 0-2 là cạnh có trọng số nhỏ nhất.</p> |
|--|--|



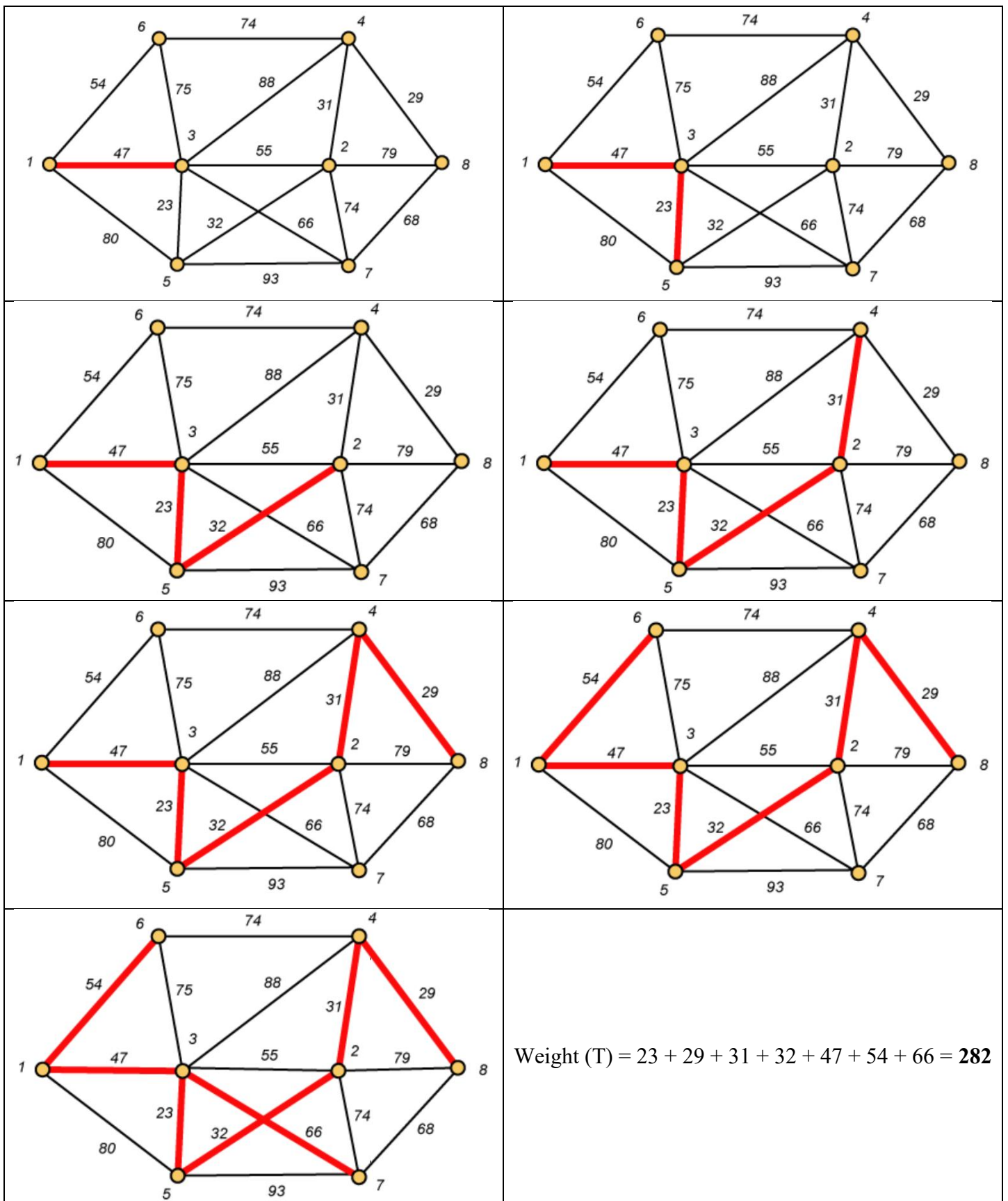
|  |  |
|--|--|
| <div data-bbox="92 40 694 159"> <div> <div>1</div> <div>3</div> <div>6</div> <div>3</div> <div>6</div> <div>7</div> </div> <div> <div>2.5</div> <div>2.1</div> <div>2.3</div> <div>2.5</div> <div>1.9</div> <div>2.0</div> </div> </div>                                  | <p>Đưa cạnh 5-6 và đỉnh 6 vào cây khung. Tiếp tục đưa cạnh 6-7 và 6-8 vào danh sách đang xét, không đưa cạnh 6-4 vì đỉnh 4 đã nằm trong cây khung. Cạnh 6-8 lúc này có trọng số nhỏ nhất.</p>  |
| <div data-bbox="92 515 694 633"> <div> <div>1</div> <div>3</div> <div>6</div> <div>3</div> <div>7</div> <div>7</div> <div>8</div> </div> <div> <div>2.5</div> <div>2.1</div> <div>2.3</div> <div>2.5</div> <div>2.0</div> <div>1.8</div> <div>1.7</div> </div> </div>     | <p>Đưa cạnh 6-8 và đỉnh 8 vào cây khung. Đưa cạnh 8-7 vào danh sách đang xét. Lúc này danh sách có cạnh 6-7 có trọng số nhỏ nhất.</p>  |
| <div data-bbox="92 985 694 1104"> <div> <div>1</div> <div>3</div> <div>6</div> <div>3</div> <div>7</div> <div>7</div> <div>7</div> </div> <div> <div>2.5</div> <div>2.1</div> <div>2.3</div> <div>2.5</div> <div>2.0</div> <div>1.8</div> <div>2.0</div> </div> </div>  | <p>Đưa cạnh 6-7 và đỉnh 7 vào cây khung.<br/>Không có cạnh nào được đưa tiếp vào danh sách đang xét vì các đỉnh kề của 7 đều đã nằm trong cây khung. Lúc này cạnh 5-7 và 8-7 đều có trọng số nhỏ nhất. Tuy nhiên, cả 3 đỉnh 5, 7, 8 đều đã nằm trong cây khung nên sẽ bị loại bỏ. Vậy cạnh 0-3 là cạnh có trọng số nhỏ nhất.</p> |
| <div data-bbox="92 1456 694 1574"> <div> <div>1</div> <div>3</div> <div>6</div> <div>3</div> <div>7</div> <div>7</div> </div> <div> <div>2.5</div> <div>2.1</div> <div>2.3</div> <div>2.5</div> <div>2.0</div> <div>2.0</div> </div> </div>                             | <p>Các cạnh còn lại được lấy ra nhưng các đỉnh đều đã nằm trong cây khung nên sẽ bị loại bỏ.</p>   |



Ví dụ 3:



Ví dụ 4:



### 5.3 So sánh Prim và Kruskal

- Cả hai thuật toán đều sử dụng phương pháp tham lam.
- Prim chọn cạnh có trọng số nhỏ nhất liên thuộc với một đỉnh đã thuộc cây và không tạo ra chu trình
- Kruskal chọn cạnh có trọng số nhỏ nhất miễn là không tạo ra chu trình
- Thuật toán Prim hiệu quả hơn đối với các đồ thị dày (số cạnh nhiều)