Chương5
CÂY KHUNG CỦA ĐỒ THỊ

Toán rời rạc 2

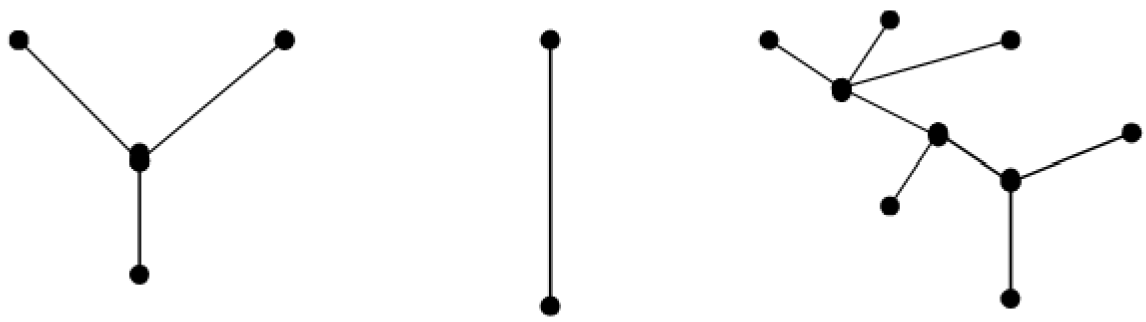
Nội dung

- Cây và các tính chất cơ bản của cây.
- Cây khung của đồ thị & các thuật toán cơ bản xây dựng cây khung của đồ thị.
- Bài toán tìm cây khung nhỏ nhất & các thuật toán tìm cây khung nhỏ nhất.
- Thuật toán Kruskal tìm cây khung nhỏ nhất.
- Thuật toán Prim tìm cây khung nhỏ nhất.

Cây và các tính chất cơ bản của cây

Định nghĩa

- **Định nghĩa 1**: Ta gọi cây là một đồ thị vô hướng, liên thông, không có chu trình.
- **Định nghĩa 2**: Ta gọi rừng là một đồ thị vô hướng, không có chu trình.
  - Như vậy rừng là một đồ thị mà mỗi thành phần liên thông của nó là một cây.
- **Ví dụ**: một rừng có 3 cây

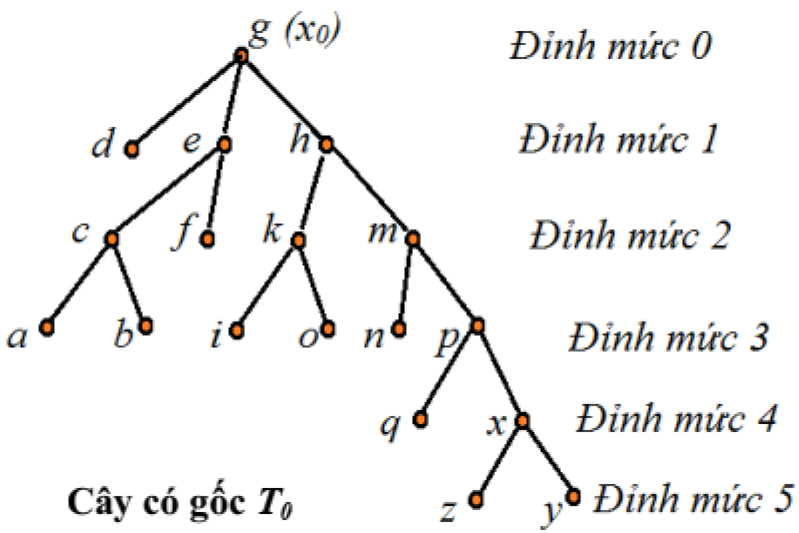


Một số tính chất của cây

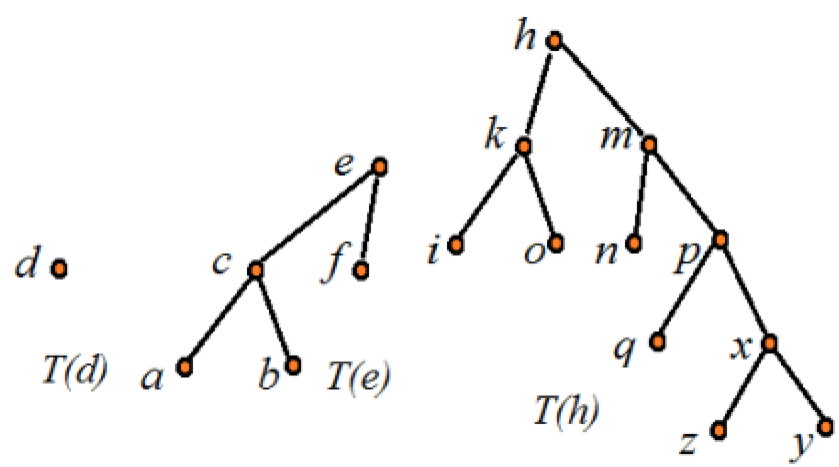
- **Định lý:** Giả sử T= <V, E> là đồ thị vô hướng *n* đỉnh. Khi đó những khẳng định sau là tương đương:
  a) T là một cây;
  b) T không có chu trình và có n-1 cạnh;
  c) T liên thông và có đúng n-1 cạnh;
  d) T liên thông và mỗi cạnh của nó đều là cầu;
  e) Giữa hai đỉnh bất kỳ của T được nối với nhau bởi đúng một đường đi đơn;
  f) T không chứa chu trình nhưng hễ cứ thêm vào nó một cạnh ta thu được đúng một chu trình;

Cây có gốc

- **Định nghĩa 1:** Cho cây T = (X, U). Trong cây T, ta tiến hành xác định mức cho từng đỉnh như sau:
  - (1) Chọn một đỉnh tùy ý làm gốc của cây, kí hiệu x_0, có mức là 0.
  - (2) Đỉnh kề với gốc gọi là đỉnh mức 1. Cứ như vậy, đỉnh chưa xác định mức nhưng kề với đỉnh mức k-1 gọi là đỉnh mức k.
- Cây đã xác định mức cho các đỉnh như trên gọi là cây có gốc.



Cây có gốc T_0



Các cây con tại g của cây có gốc T_0

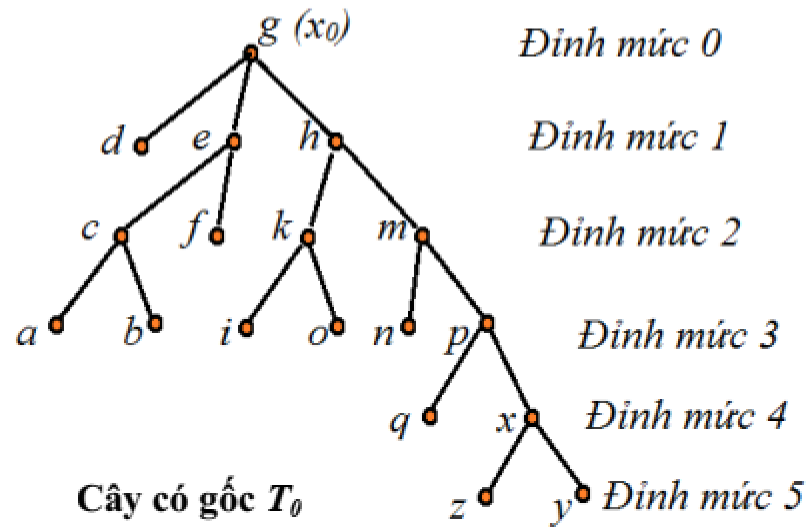
Cây có gốc

Định nghĩa 2: Cho cây có gốc *T*.

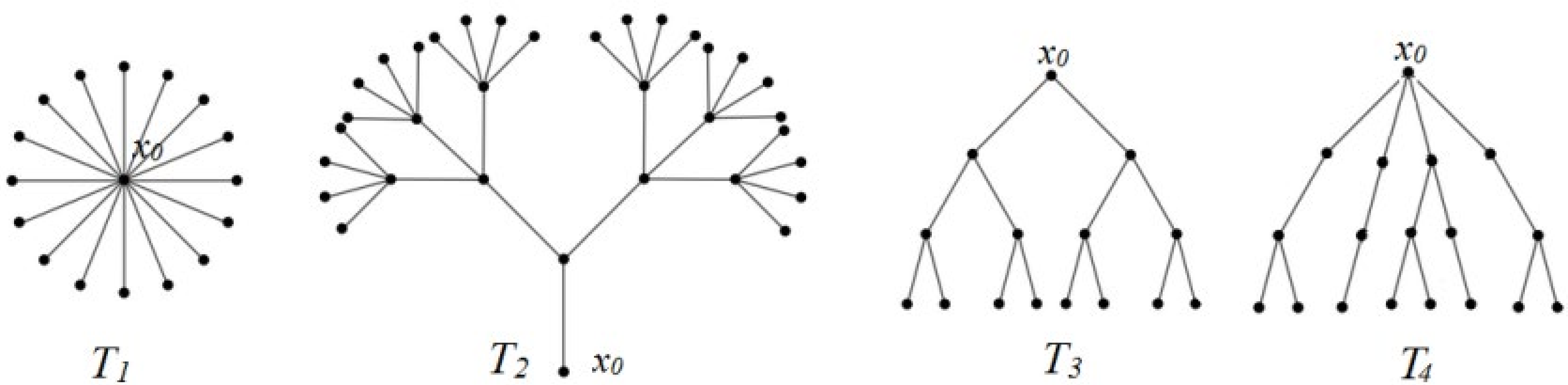
1) Nếu *v* là một đỉnh khác gốc *x0* thì *có một đường đi đơn duy nhất* từ gốc *x0* đến *v*. Trên đường đi này: ta gọi đỉnh *u* kề với *v* là *cha của v* và ngược lại *v* gọi là *con* của *u*; tất cả các đỉnh không kề với *v* gọi là *tổ tiên* của *v* và ngược lại *v* được gọi là *con cháu* của các đỉnh này.
2) Các đỉnh có cùng cha gọi là *anh em*.
3) Cho *y* là một đỉnh của cây *T*. Kí hiệu *T(y)* là đồ thị con của cây *T* đang xét, bao gồm đỉnh *y* và các con cháu của nó cùng tất cả các cạnh kề các con cháu của *y*. Khi đó *T(y)* được gọi là một *cây con tại đỉnh r* của cây *T* ban đầu (ở đây *r* là cha của *y*).
4) Các đỉnh của cây không có con được gọi là *lá*. Hay nói cách khác, mọi đỉnh treo không có con (không phải là gốc) là *lá* của cây. Mọi đỉnh không phải là lá gọi là *đỉnh trong* của cây (gốc *x0* cũng là một đỉnh trong). *Chiều cao* h(*T*) của cây là mức lớn nhất của một lá trong cây.
5) Một cây có gốc *T* gọi là *đối xứng (hay cân đối)* nếu mọi đỉnh là lá đều có mức bằng chiều cao h(*T*) của cây.

Cây có gốc

Cây T_0 là cây có gốc không cân đối.



Cây có gốc T_0



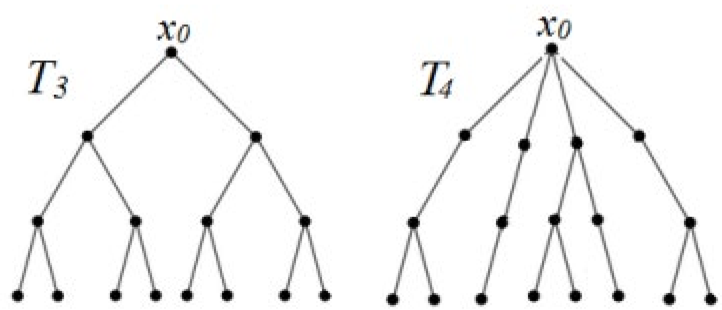
Các cây cân đối với gốc là đỉnh x0

Cây *m*-phân

Định nghĩa:

- (1) Một cây có gốc *T* được gọi là *cây m-phân*, nếu mỗi đỉnh trong của *T* có nhiều nhất *m* con và có ít nhất một đỉnh có m con.
- (2) Cây có gốc *T* gọi là cây *m-phân đầy đủ* nếu mọi đỉnh trong của nó đều có *m* con.
- (3) Cây *m*-phân với $m = 2$ gọi là *cây nhị phân*.

Định lý 1: Một cây *m*-phân đầy đủ có *k* đỉnh trong thì có $m.k + 1$ đỉnh và $(m – 1).k + 1$ lá.



Định lý 2: Một cây *m*-phân có chiều cao *h* và số lá *s* thì: $s \le m^h$ hay $h \ge log_m(s)$.

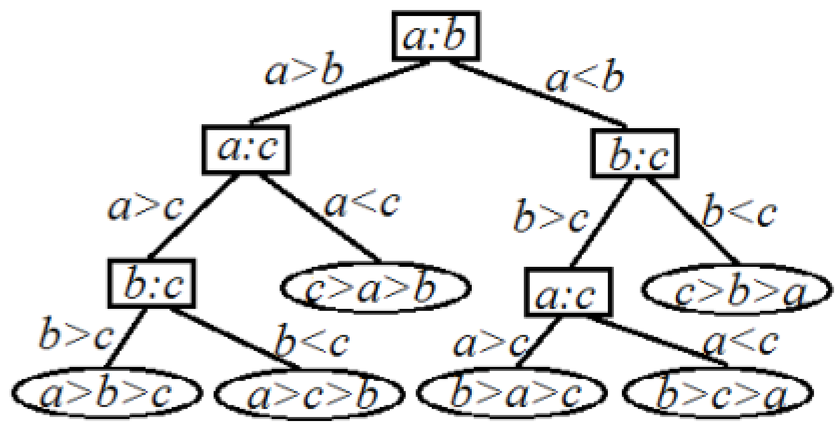
- Ví dụ: *Cho cây nhị phân T có 18 lá. Ta có* $log_2 18 = 4,17$ *nên chiều cao nhỏ nhất của T là:* $h = 5$.

Cây quyết định

Định nghĩa:

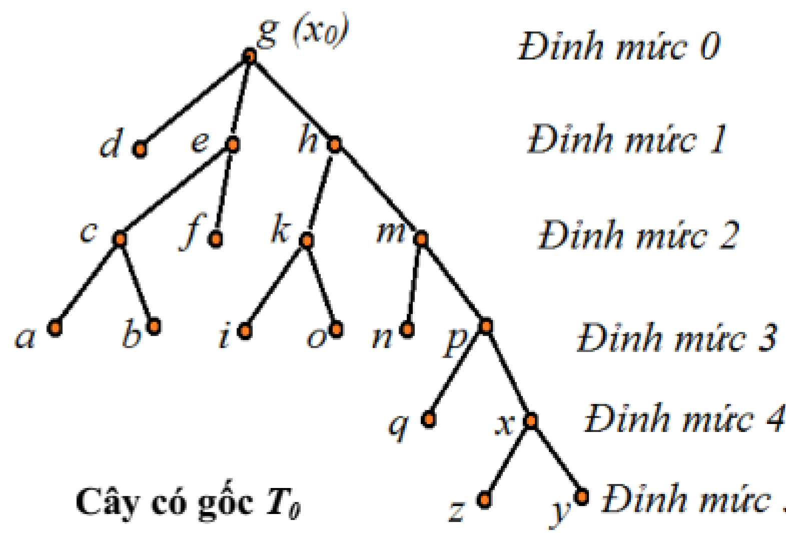
- *Cây quyết định* là một cây có gốc có thể dùng để mô hình các bài toán, trong đó có một dãy các quyết định dẫn đến lời giải.
- Mỗi *đỉnh trong* ứng với *một quyết định* và *mỗi cây con* của nó ứng với *mỗi một kết cục* có thể của quyết định cha.
- Những lời giải có thể có của bài toán tương ứng với mỗi đường đi từ gốc đến lá của cây quyết định này.

Ví dụ*: Cần thực hiện tối đa bao nhiêu phép so sánh nhị nguyên (mỗi lần so sánh hai phần tử với nhau) để sắp xếp n số thực phân biệt a1, a2, …, an theo thứ tự giảm dần*.



Các thuật toán duyệt cây

- ***Định nghĩa:*** Cho *T* = *T(r)* là một cây có gốc và được sắp thứ tự với gốc *r*. Gọi *T1*, *T2*, …, *Tn* là các cây con tại *r* của *T* theo thứ tự từ trái qua phải.
  - (1) *Duyệt tiền thứ tự:* Thăm gốc *r* / Duyệt cây con *T1* /…/ Duyệt cây con *Tn*.
  - (2) *Duyệt trung thứ tự:* Duyệt cây con *T1* / Thăm gốc *r* / Duyệt cây con *T2* / Duyệt cây con *T3* /…/ Duyệt cây con *Tn*.
  - (3) *Duyệt hậu thứ tự:* Duyệt cây con *T1* / …/ Duyệt cây con *Tn* / Thăm gốc *r*.



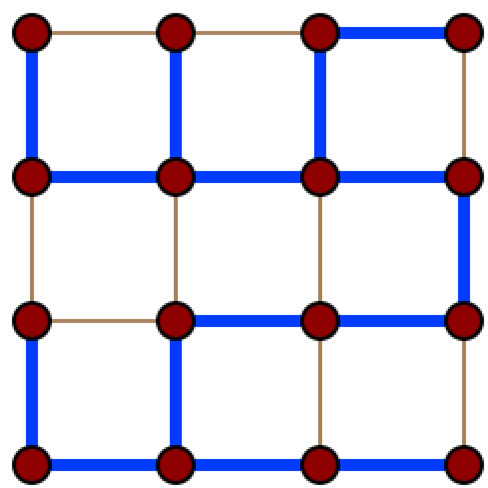
Cây có gốc T_0

Duyệt tiền thứ tự	*g, d, e, c, a, b, f, h, k, i, o, m, n, p, q, x, z, y*
Duyệt trung thứ tự	*d, g, a, c, b, e, f, i, k, o,h, n, m, q, p, z, x, y.*
Duyệt hậu thứ tự	*d, a, b, c, f, e, i, o, k, n, q, z, y, x, p, m, h, g.*

Cây khung của đồ thị

Định nghĩa Cây khung

- **Định nghĩa:** Cho G là đồ thị vô hướng liên thông. Ta gọi đồ thị con T của G là một cây khung (cây bao trùm) của G nếu T thoả mãn hai điều kiện:
  a) T là một cây;
  b) Tập đỉnh của T bằng tập đỉnh của G.
- **Ví dụ:**



Định nghĩa Cây khung

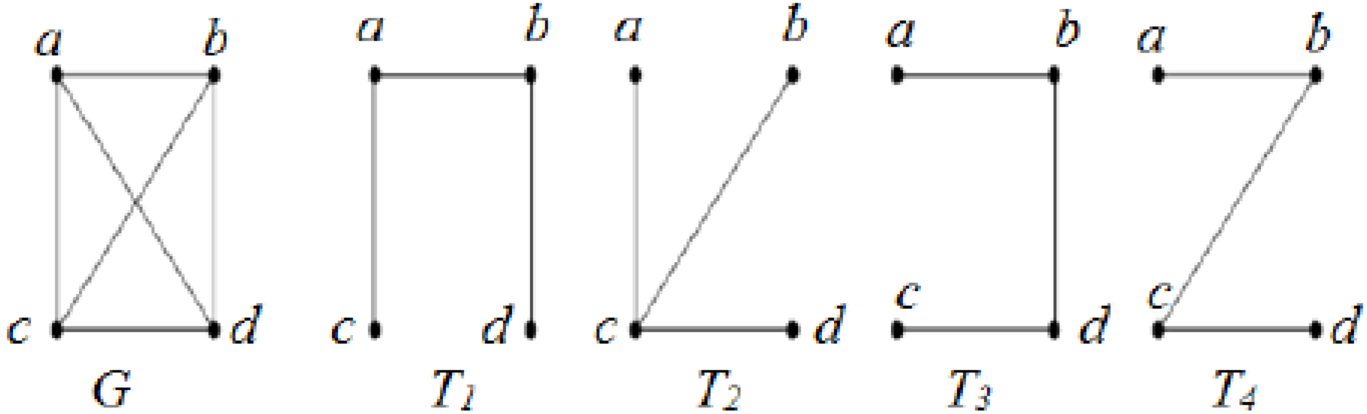
Định Nghĩa:

Cho đồ thị vô hướng $G = (X, U)$ có n đỉnh $(n \geq 2)$. Đồ thị T gọi là một *cây khung* của G nếu T là một cây và T là đồ thị bộ phận của G.

Nhận xét: Đồ thị $T = (E, V)$ là một cây khung của đồ thị $G = (X, U)$ thì *T liên thông, không có chu trình, có n đỉnh và có n – 1 cạnh* (tức là $N(E) = N(X)$ và $N(V) = n - 1$).

Định lý: Điều kiện cần và đủ để đồ thị G có cây khung là G liên thông.

Ví dụ:



(Số đỉnh n = 4)

Cây khung T_1 T_2 T_3 và T_4 của đồ thị G với số đỉnh n = 4

Hai bài toán cơ bản về cây

- **Bài toán 1.** Cho đồ thị vô hướng G=<V,E>. Hãy xây dựng một cây khung của đồ thị bắt đầu tại đỉnh u$\in$V.
  - Giải quyết bằng các thuật toán tìm kiếm cơ bàn: thuật toán DFS hoặc BFS.
- **Bài toán 2.** Cho đồ thị vô hướng G=<V,E> có trọng số. Hãy xây dựng cây khung có độ dài nhỏ nhất.
  - Giải quyết bằng thuật toán Kruskal hoặc PRIM.

Hai bài toán cơ bản về cây

- **Bài toán 1.** Cho đồ thị vô hướng G=<V,E>. Hãy xây dựng một cây khung của đồ thị bắt đầu tại đỉnh u∈V.
  - Giải quyết bằng các thuật toán tìm kiếm cơ bản: thuật toán DFS hoặc BFS.

Thuật toán tìm kiếm ưu tiên chiều sâu.

(1) : Chọn một đỉnh tùy ý làm gốc của cây.

(2) : Từ đỉnh vừa chọn, xây dựng một đường đi bằng cách ghép lần lượt các cạnh mới nối tiếp các cạnh trước đó, sao cho không tạo thành chu trình (Cạnh ghép thêm phải với tới các đỉnh mới).

(3) : Nếu đường đi vừa lập đã đi qua tất cả các đỉnh của *G* (hoặc đã đủ $n - 1$ cạnh) thì ta được cây khung cần tìm.

Ngược lại, nếu đường đi này chưa đi qua mọi đỉnh của *G* thì từ đỉnh cuối cùng của đường đi này, quay lại đỉnh trước nó và lặp lại bước (2).

Hai bài toán cơ bản về cây

- **Bài toán 1.** Cho đồ thị vô hướng G=<V,E>. Hãy xây dựng một cây khung của đồ thị bắt đầu tại đỉnh u∈V.
  - Giải quyết bằng các thuật toán tìm kiếm cơ bàn: thuật toán DFS hoặc BFS.

Thuật toán tìm kiếm ưu tiên chiều rộng.

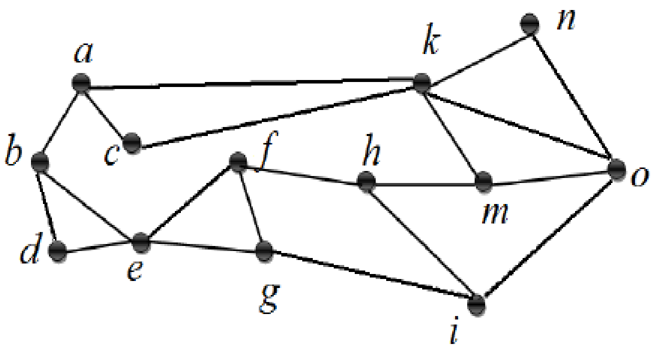
(1) : Chọn một đỉnh tùy ý làm gốc của cây, ghép tất cả các cạnh kề với gốc ta được đỉnh mức 1.

(2) : Từ mỗi đỉnh mức *k*, ghép tất cả các cạnh kề với đỉnh này sao cho không tạo thành chu trình, ta được đỉnh mức $k + 1$.

Lặp lại (2). Thuật toán dừng khi tất cả các đỉnh được ghép vào cây (hoặc đã ghép đủ $n - 1$ cạnh)

Hai bài toán cơ bản về cây

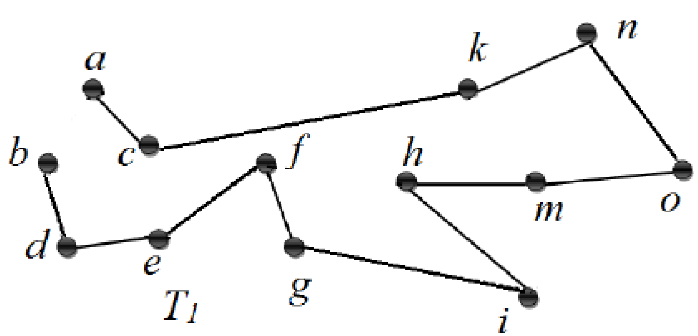
Ví dụ: Cho đồ thị sau, tìm cây khung theo DFS và BFS



Giải. $n = 13$. *Cây khung của G có 12 cạnh.*

- *Theo thuật toán tìm kiếm ưu tiên chiều sâu:*

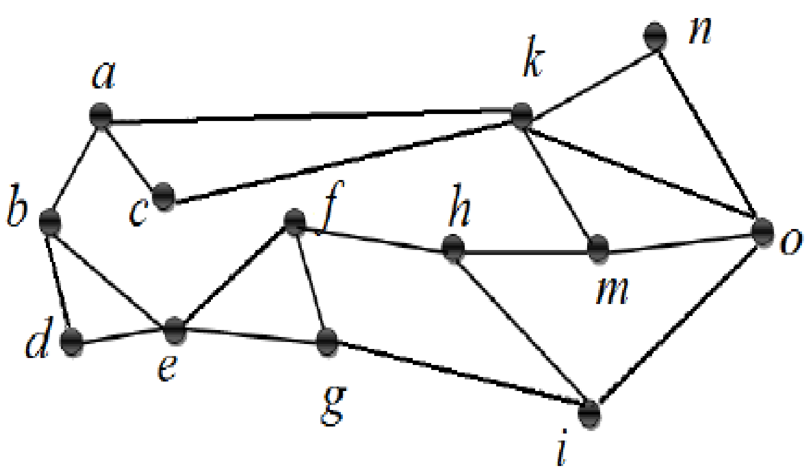
$$\begin{cases} \text{Chọn } a \text{ làm gốc} \\ \text{Từ } a \text{ có đường đi } < a; c; k; n; o; m; h; i; g; f; e; d; b > \end{cases}$$ *(đã ghép đủ 12 cạnh).*

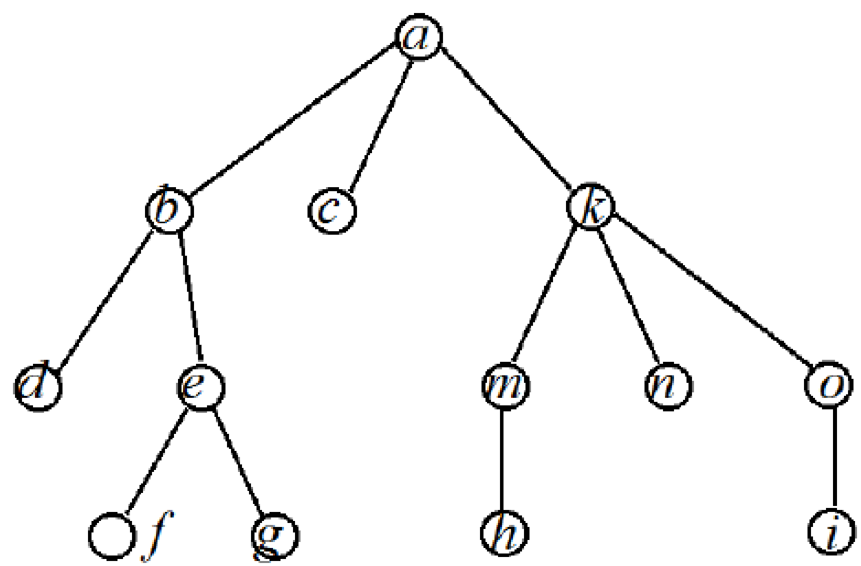


Hai bài toán cơ bản về cây

Theo thuật toán tìm kiếm ưu tiên chiều rộng:

$$\begin{cases} \textit{Chọn đỉnh } a \textit{ làm gốc} \quad (\textit{mức } 0) \\ \textit{Các đỉnh mức 1 kề với gốc } a \textit{ là: } b, c, k. \\ \textit{Các đỉnh mức 2 kề với đỉnh } b \textit{ là: } d, e; \textit{ đỉnh mức 2 kề với } k \textit{ là: } m, n, o. \\ \textit{Các đỉnh mức 3 kề với } e \textit{ là: } f, g; \textit{ kề } m \textit{ là: } h; \textit{ kề } o \textit{ là: } i. \end{cases}$$





Cây khung theo BFS: Ghép đủ (n-1) cạnh

Hai bài toán cơ bản về cây

Ví dụ: Cho ma trận đỉnh-cạnh của đồ thị sau, tìm cây khung của nó
n = 5; => cây khung có 4 cạnh

$$A = \begin{array}{c c} & \begin{array}{ccccc} a & b & c & d & e \end{array} \\ \begin{array}{c} a \\ b \\ c \\ d \\ e \end{array} & \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

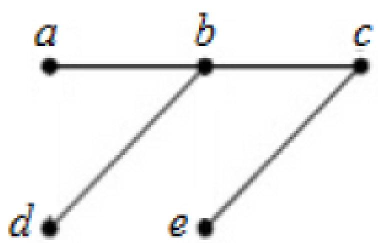
✓ *Theo thuật toán tìm kiếm ưu tiên chiều sâu:*

- *Chọn a làm gốc.*

- *Từ a có đường đi <a; b; c; e>.*

- *Quay lại b có đường đi <b; d>.*

Đã chọn đủ 4 cạnh, ta được cây khung T_1



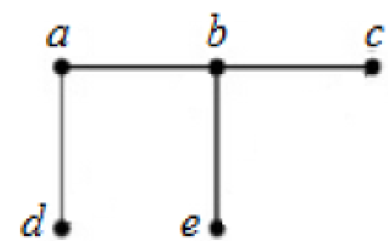
✓ *Theo thuật toán tìm kiếm ưu tiên chiều rộng:*

- *Chọn a làm gốc (mức 0).*

- *Đỉnh mức 1 kề gốc a là: b ; d.*

- *Đỉnh mức 2 kề b là : c ; e.*

Đã đủ 5 đỉnh, ta được cây khung T_2.



Bài toán 2: Tìm cây khung nhỏ nhất & các thuật toán tìm cây khung nhỏ nhất trong đồ thị có trọng số. Thuật toán Kruskal và thuật toán Prim

Xây dựng cây khung có độ dài nhỏ nhất

- Mô tả:
  - Cho G=<V, E> là đồ thị vô hướng liên thông với tập đỉnh V = {1, 2, . . ., n } và tập cạnh E gồm **m** cạnh.
  - Mỗi cạnh **e** của đồ thị được gán với một số không âm ***c(e)*** được gọi là độ dài cạnh.
  - Giả sử H=<V, T> là một cây khung của đồ thị G. Ta gọi độ dài ***c(H)*** của cây khung H là tổng độ dài các cạnh:

$$c(H) = \sum_{e \in T} c(e)$$

- Bài toán:
  - Trong số các cây khung của đồ thị hãy tìm *cây khung có độ dài nhỏ nhất* của đồ thị.

Hai mô hình thực tế của bài toán

- **Bài toán nối mạng máy tính**
  - Một mạng máy tính gồm **n** máy tính được đánh số từ 1, 2, . . ., n. Biết chi phí nối máy ***i*** với máy **j** là ***c[i, j]***, i, j = 1, 2, . . ., n.
  - Hãy tìm cách nối mạng sao cho chi phí là nhỏ nhất.
- **Bài toán xây dựng hệ thống cable**
  - Giả sử ta muốn xây dựng một hệ thống cable điện thoại nối **n** điểm của một mạng viễn thông sao cho điểm bất kỳ nào trong mạng đều có đường truyền tin tới các điểm khác.
  - Biết chi phí xây dựng hệ thống cable từ điểm ***i*** đến điểm ***j*** là ***c[i,j]***.
  - Hãy tìm cách xây dựng hệ thống mạng cable sao cho chi phí là nhỏ nhất.

Thuật toán Kruskal (1/2)

Thuật toán Kruskal: Thuật toán Kruskal tìm cây khung nhỏ nhất dựa trên nguyên tắc: Chọn lần lượt đủ $n - 1$ cạnh bằng cách mỗi lần chọn thêm một cạnh có trọng số nhỏ nhất, trong số các cạnh còn lại chưa được chọn của G, sao cho các cạnh đã chọn không tạo thành chu trình. Mô tả như sau:

(1) : Sắp xếp tập cạnh U của G theo thứ tự trọng số không giảm: $l(u_1) \le l(u_2) \le \cdots \le l(u_m)$

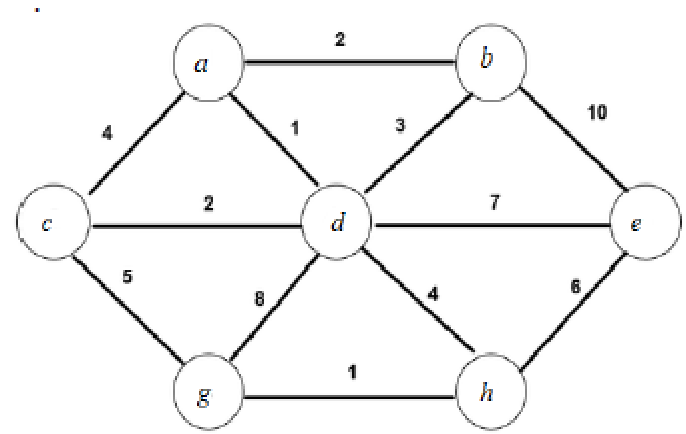
(2) : Chọn lần lượt các cạnh của dãy trên sao cho các cạnh đã chọn không tạo thành chu trình.

Dừng khi chọn đủ $n - 1$ cạnh.

Thuật toán Kruskal (2/2)

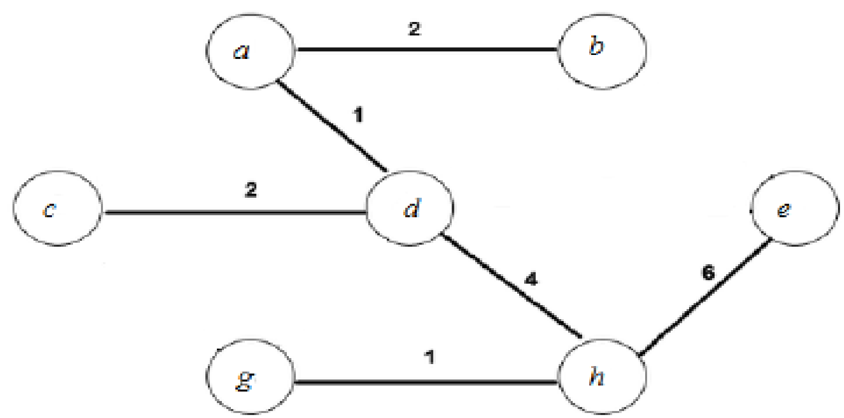
Ví dụ: Tìm Cây khung nhỏ nhất (CKNN) cho đồ thị G sau, sử dụng Kruskal:

1/ Các cạnh chọn: không hình thành chu trình
2/ Cây khung nhỏ nhất thỏa: Tổng số cạnh chọn: (n-1) và đủ n đỉnh.



Có n = 7. Cây khung của G có 6 cạnh. Theo thuật toán Kruskal:

TT	Trọng số	Bước chọn	Đỉnh đã chọn	Cây khung nhỏ nhất
1	1 = (a,d)	1	a; d	
2	1 = (g,h)	2	g; h	
3	2 = (c,d)	3	c	
4	2 = (a,b)	4	b	
5	3 = (b,d)	-------------	------------------	
6	4 = (a,c)	-------------	------------------	
7	4 = (d,h)	5		
8	5 = (c,g)	-------------	------------------	
9	6 = (e,h)	6	e	
10	7 = (d,e)	-------------	------------------	
11	8 = (d,g)	-------------	------------------	
12	10 = (b,e)	-------------	------------------	



T_0

Trọng số của CKNN là: $l(T_0) = 16$.

Thuật toán Prim(1/2)

Thuật toán Prim (tiếng anh: Prim's algorithm) là một thuật toán tham lam được dùng để tìm cây khung nhỏ nhất (Minimum Spanning Tree – MST) của một đồ thị liên thông có trọng số. Thuật toán được tìm ra vào năm 1975 và được đặt tên theo nhà nghiên cứu khoa học máy tính Robert C. Prim.

Một số thuật ngữ liên quan:
Spanning Tree : là cây khung của đồ thị liên thông và vô hướng, (Cho nên thuật toán này chỉ sử dụng cho đồ thị vô hướng).
Minimum Spanning Tree (MST): là cây khung nhỏ nhất, (Có tổng trọng số của các cạnh là nhỏ nhất).

Thuật toán Prim(1/2)

Ý tưởng thuật toán Prim

Thuật toán Prim sẽ bắt đầu bằng việc chọn ngẫu nhiên một đỉnh bất kì và tiếp tục thêm các cạnh có trọng số nhỏ nhất cho đến khi có đủ tập hợp các đỉnh. Các bước để thực hiện:

- Bước 1. Khởi tạo tập hợp đỉnh V rỗng, tập hợp cạnh E rỗng.

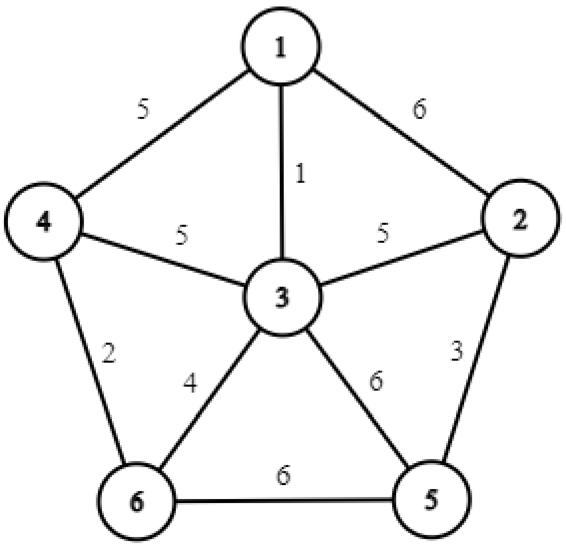
- Bước 2. Chọn ngẫu nhiên 1 đỉnh và thêm vào tập hợp các đỉnh V.

- Bước 3. Chọn 1 đỉnh chưa có trong tập V mà có kết nối với 1 đỉnh trong V, cạnh tạo từ 2 đỉnh đó phải là cạnh có trọng số nhỏ nhất và thêm vào tập hợp các cạnh E.

- Bước 4. Lặp lại bước 3 cho đến khi cây khung hoàn thành (Cách nhận biết cây khung hoàn thành là tất cả các đỉnh của cây khung đều đã xuất hiện trong V), cây khung nhỏ nhất là cây khung được tạo từ tập các cạnh trong E.

Thuật toán Prim(1/2)

Ví dụ:



•Có đồ thị G=(V, E) Có V đỉnh và E cạnh.

•Tập hợp các đỉnh của đồ thị V = {1, 2, 3, 4, 5, 6} (Viết tắt của vertex nghĩa là đỉnh).

•Tập hợp các cạnh của E = {(1, 2), (1, 3), (1, 4), (2, 3), (2, 5), (3, 4), (3, 5), (3, 6), (4, 6), (5, 6)} (Viết tắt của edge nghĩa là cạnh).

•Ta tạo ra 2 tập hợp U và T, U là tập hợp các đỉnh và T là tập hợp các cạnh tạo ra cây khung nhỏ nhất.

•Ban đầu U={Ø} (rỗng) và T={Ø}.

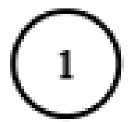
Thuật toán Prim(1/2)

Bước 1. Khởi tạo

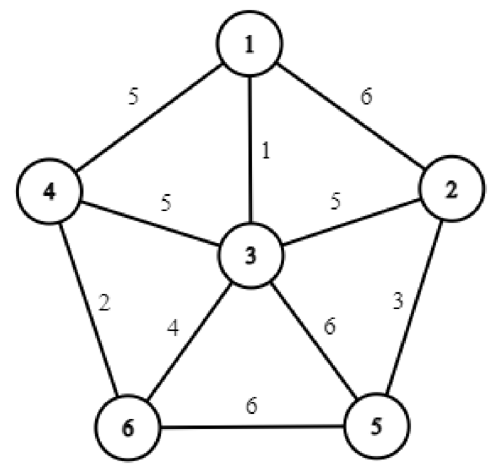
•Ta tạo ra 2 tập hợp U và T, U là tập hợp các đỉnh và T là tập hợp các cạnh tạo ra cây khung nhỏ nhất.

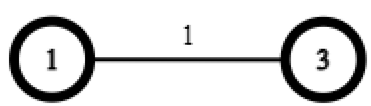
•Ban đầu U={Ø} (rỗng) và T={Ø}.

Bước 2. Chọn 1 đỉnh ngẫu nhiên từ cây khung trên, ví dụ chúng ta chọn đỉnh 1. Sau bước này, ta có U={1} và T={Ø}.

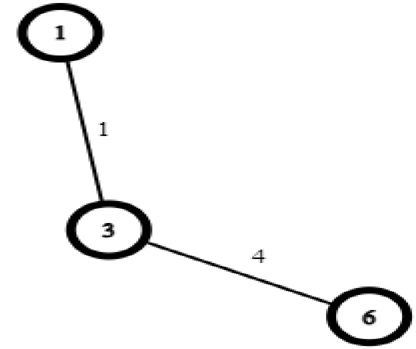


Bước 3. Tìm tất cả các cạnh có 1 đỉnh ở trong U. được cạnh (1, 3) do khoảng cách nhỏ nhất với gi này ta có U={1, 3} và T={(1, 3)}.

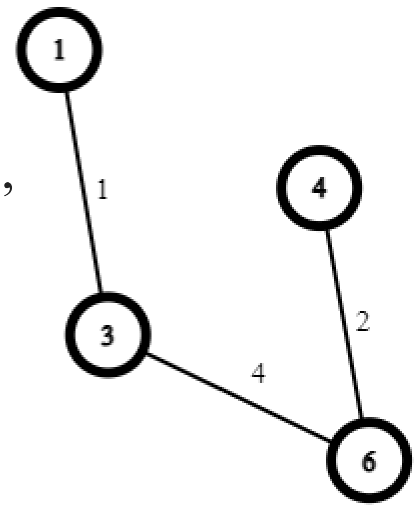




Thuật toán Prim(1/2)

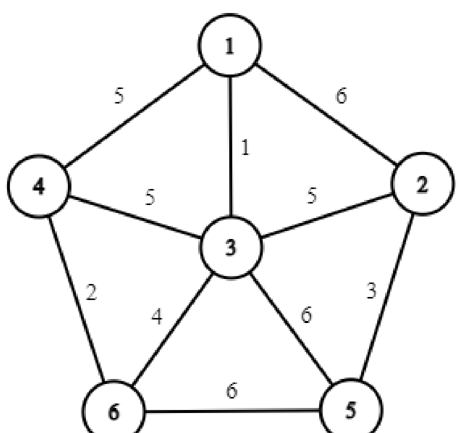


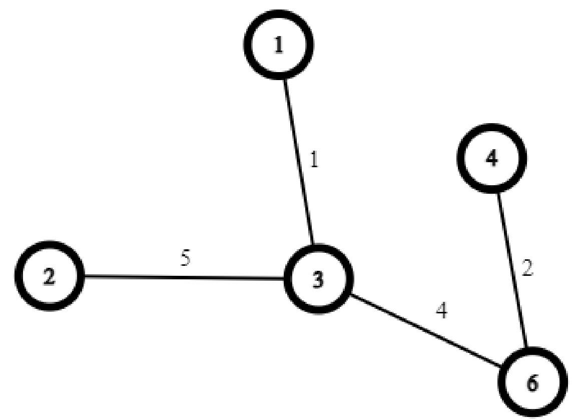
Bước 4. Tương tự bước 3, ta đi tìm cạnh nhỏ nhất có 1 đỉnh trong U mà cạnh đó chưa có trong V. Kết quả là ta tìm ra cạnh (3, 6) với giá trị 4. Sau bước này ta có U={1, 3, 6} và T={(1, 3), (3, 6)}.



Bước 5. Do tập U mới chỉ có 3/6 đỉnh, ta tiếp tục đi tìm cạnh nhỏ nhất chưa khám phá mà 1 đỉnh của nó trong U. Kết quả là ta tìm được cạnh (6, 4). Bây giờ U={1, 3, 6, 4} và T={(1, 3), (3, 6), (6, 4)}.

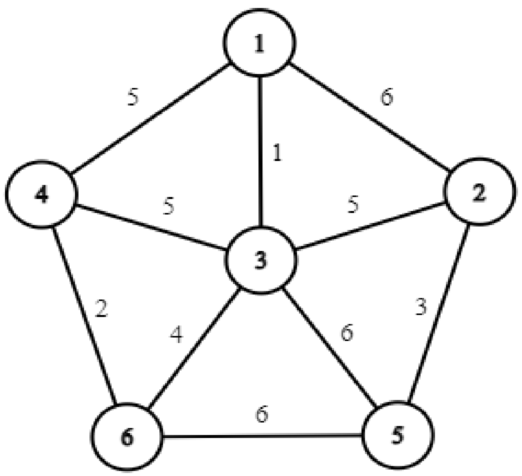
Bước 6. Do tập U mới chỉ có 4/6 đỉnh, tiếp tục tìm cạnh nhỏ nhất chưa có trong T mà 1 đỉnh của nó trong U. Ta tìm được cạnh (3,2) với giá trị 5. Sau bước này U={1, 3, 6, 4, 2} và T={(1, 3), (3, 6), (6, 4), (3, 2)}.

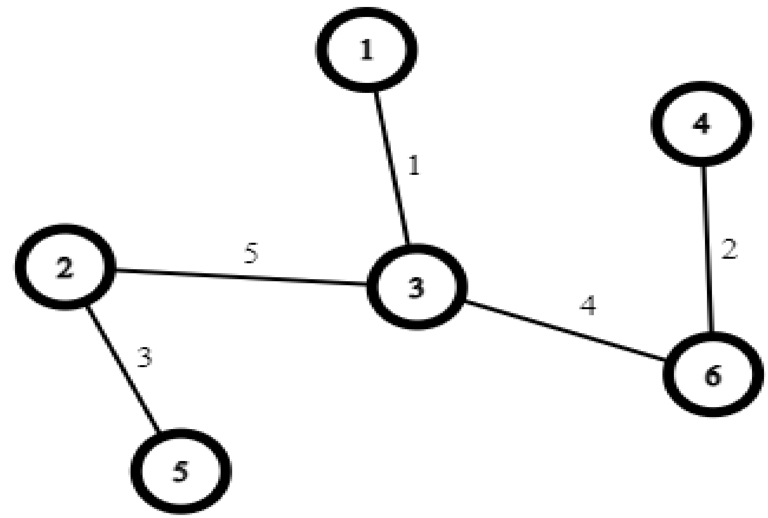




Thuật toán Prim(1/2)

Bước 7. Do tập U mới chỉ có 5/6 đỉnh, tiếp tục tìm cạnh nhỏ nhất chưa có trong T mà 1 đỉnh của nó trong U. Ta tìm được cạnh (2, 5) với giá trị 3. Lúc này tập U đã có 6/6 đỉnh nên thuật toán kết thúc.



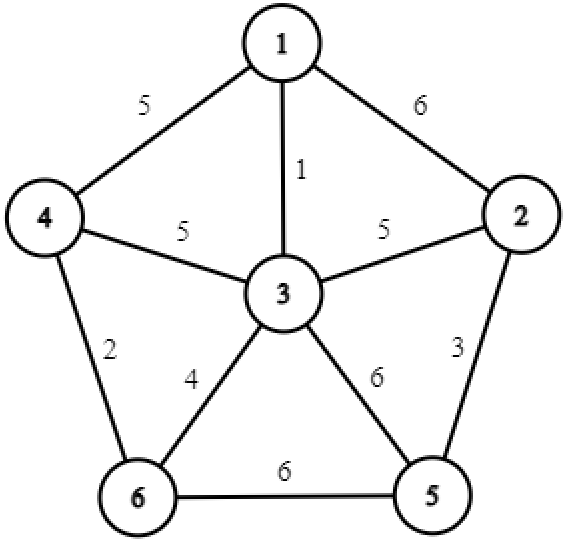


Cây khung nhỏ nhất mà chúng ta tìm được bằng thuật toán Prim.

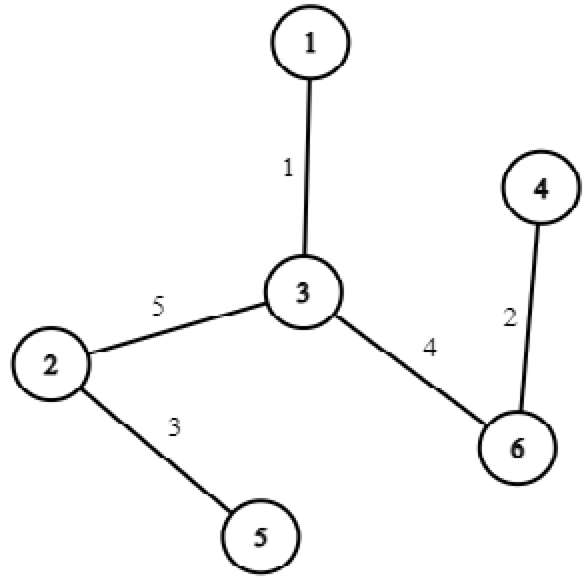
Sau khi áp dụng xong thuật toán Prim, ta có U={1, 3, 6, 4, 2, 5} và T={(1, 3), (3, 6), (6, 4), (3, 2), (2, 5)}.

•Tập hợp các cạnh T={(1, 3), (3, 6), (6, 4), (3, 2), (2, 5)} tạo thành cây khung có tổng trọng số nhỏ nhất. Tổng trọng số của cây khung nhỏ nhất là:1 + 4 + 2 + 5 + 3 = 15

Thuật toán Prim(1/2)



Spanning Tree



Minimum Spanning Tree

So sánh 2 cây khung (ban đầu, bên trái) và cây khung nhỏ nhất tìm được bằng thuật toán Prim (bên phải).

Ứng dụng của cây khung nhỏ nhất

Với bài toán tìm cây khung nhỏ nhất, chúng ta có thể áp dụng để giải quyết & tối ưu rất nhiều bài toán trong thực tế. Ví dụ với bài toán lắp đặt hệ thống mạng cho 1 trường học:

- Tất cả các phòng cần có kết nối internet.
- Biết trước khoảng cách giữa 2 phòng bất kỳ

- Bài toán: Làm sao lắp đặt hệ thống mạng cho trường học này với chi phí tối thiểu (giả sử chi phí tỉ lệ thuận với khoảng cách).

Thuật toán Prim(1/2)

Thuật toán Prim: Thuật toán Prim tìm cây khung nhỏ nhất dựa trên nguyên tắc: Từ một đỉnh tùy của *G*, chọn lần lượt đủ $n - 1$ cạnh bằng cách mỗi lần thêm một cạnh kề có trọng số nhỏ nhất (trong số tất cả các cạnh kề với các đỉnh đã được chọn) sao cho các cạnh đã chọn không tạo thành chu trình. Mô tả như sau:

(1) : Chọn một đỉnh tùy ý làm gốc. Gọi Y = Tập đỉnh đã chọn.

(2) : Gọi K = Tập các cạnh kề với các đỉnh đã chọn, mà đỉnh kia $\notin Y$.

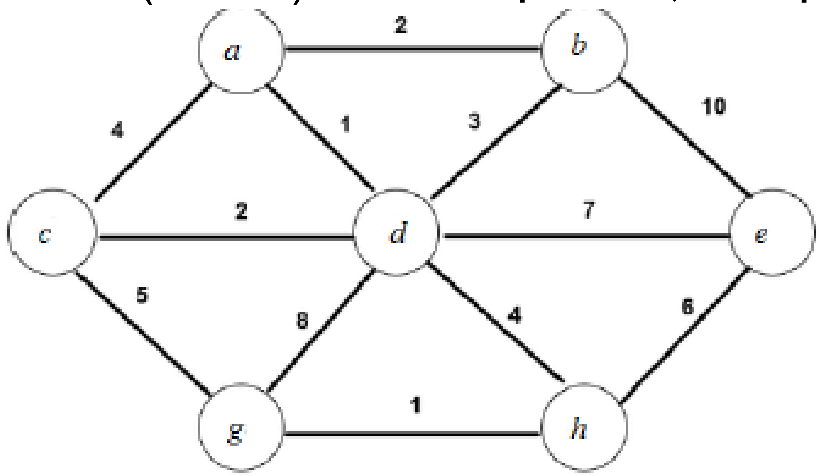
Gọi v_0 là cạnh có *trọng số nhỏ nhất* trong các cạnh kề thuộc K, tức $l(v_0) = \min_{u \in K} l(u)$.

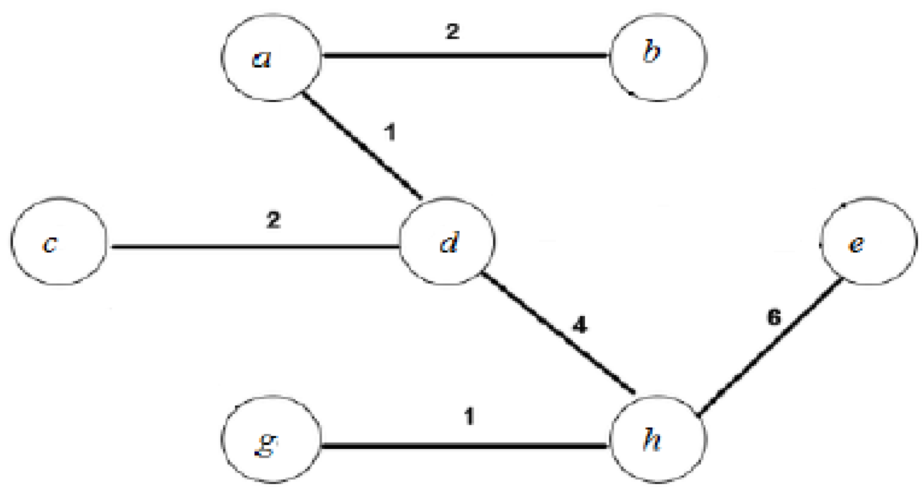
Giả sử $v_0 = (y_0, z_0)$ với $y_0 \in Y$, $z_0 \notin Y$. Khi đó: v_0 là cạnh tiếp theo được chọn và $Y := Y \cup \{z_0\}$.

(3) : Lặp lại B2. Dừng khi $Y = X$.

Thuật toán Prim(1/2)

Ví dụ: Tìm Cây khung nhỏ nhất (CKNN) cho đồ thị G sau, sử dụng Prim:



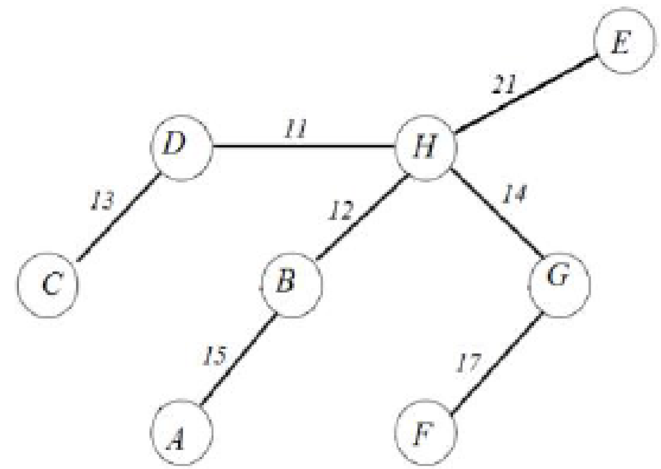
TT	*Các cạnh kề*	*Cạnh kề nhỏ nhất*	*Đỉnh đã chọn*	*Cây khung nhỏ nhất*
0			*a*	
1	*(a,d) = 1*	*(a,d) = 1*	*d*	
2	*(a,b) = 2* *(d,c) = 2…*	*(a,b) = 2*	*b*	
3	*(d,c) = 2* *(b,e) = 10…*	*(d,c) = 2*	*c*	
4	*(c,g) = 5* *(d,h) = 4…*	*(d,h) = 4*	*h*	
5	*(h,g) = 1* *(c,g) = 5…*	*(h,g) = 1*	*g*	
6	*(h,e) = 6…*	*(h,e) = 6*	*e*	*T_0* *Trọng số của CKNN là: $l(T_0) = 16$.*

Ví dụ: *Tìm cây khung nhỏ nhất của đồ thị có ma trận kề trọng số là :*

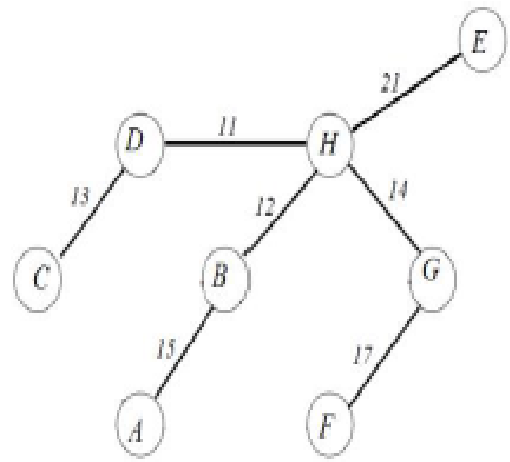
$$
\begin{array}{c c}
 & \begin{array}{cccccccc} A & B & C & D & E & F & G & H \end{array} \\
\begin{array}{c} A \\ B \\ C \\ D \\ E \\ F \\ G \\ H \end{array} &
\begin{bmatrix}
\infty & 15 & 16 & 19 & 23 & 20 & 32 & 18 \\
15 & \infty & 33 & 13 & 34 & 19 & 20 & 12 \\
16 & 33 & \infty & 13 & 29 & 21 & 20 & 19 \\
19 & 13 & 13 & \infty & 22 & 30 & 21 & 11 \\
23 & 34 & 29 & 22 & \infty & 34 & 23 & 21 \\
20 & 19 & 21 & 30 & 34 & \infty & 17 & 18 \\
32 & 20 & 20 & 21 & 23 & 17 & \infty & 14 \\
18 & 12 & 19 & 11 & 21 & 18 & 14 & \infty
\end{bmatrix}
\end{array}
$$

Giải. n = 8 (đỉnh) => cây khung có 7 cạnh.

Thuật toán Kruskal:

TT	A	B	C	D	E	F	G	Cạnh chọn	Cạnh xóa
B	15								 Cây khung nhỏ nhất T_0
C	16	33							
D	19	13	13						
E	23	34	29	22					
F	20	19	21	30	34				
G	32	20	20	21	23	17			
H	18	12	19	11	21	18	14		
Trọng số nhỏ nhất trên từng cột	15	12	13	11	21	17	14	DH (11)	
				21				BH (12)	BD
		19						DC (13)	CH, CB
		19	20					GH (14)	GB, GD, GC
		19	21	22			∞	AB (15)	AH, AD, AC, AG
	20							GF (17)	FA, FB, FC, FD, FH
	23	34	29			∞		EH (21)	Vậy: $l(T_0) = 103$

Thuật toán Prim:

TT	A	B	C	D	E	F	G	H	Cạnh chọn	Đỉnh chọn	Cạnh xóa
B	15										
C	16	33									
D	19	13	13								
E	23	34	29	22							
F	20	19	21	30	34						
G	32	20	20	21	23	17					
H	18	12	19	11	21	18	14	∞			
Khởi tạo										A	
Trọng số nhỏ nhất kề với đỉnh đã chọn	15								AB = 15	B	
	16	12							BH = 12	H	HA
	16	13						11	HD =11	D	DB, DA
	16	19		13				14	DC =13	C	CA, CB, CH
	20	19	20	21				14	HG =14	G	GA, GB, GC, GD
	20	19	21	22			17	18	GF = 17	F	FA, FB, FC, FD, FH
	23	34	29	22		34	23	21	HE = 21	E	Vậy: $l(T_0) = 103$

Kiểm nghiệm thuật toán Kruskal(1/4)

- Áp dụng thuật toán Kruskal tìm cây khung nhỏ nhất cho đồ thị được biểu diễn bằng ma trận trọng số như hình bên ?

$$
\begin{vmatrix}
\infty & 2 & 1 & 3 & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
2 & \infty & 2 & \infty & \infty & 5 & 5 & \infty & \infty & \infty & \infty & \infty & \infty \\
1 & 2 & \infty & 4 & \infty & 5 & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
3 & \infty & 4 & \infty & 5 & 5 & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & 5 & \infty & 6 & \infty & \infty & \infty & 6 & \infty & \infty & \infty \\
\infty & 5 & 5 & 5 & 6 & \infty & 6 & 6 & 6 & 6 & \infty & \infty & \infty \\
\infty & 5 & \infty & \infty & \infty & 6 & \infty & 6 & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 6 & 6 & \infty & 7 & \infty & \infty & 7 & 7 \\
\infty & \infty & \infty & \infty & \infty & 6 & \infty & 7 & \infty & 7 & 7 & \infty & \infty \\
\infty & \infty & \infty & \infty & 6 & 6 & \infty & \infty & 7 & \infty & 7 & 7 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & \infty & 7 & 7 & \infty & 8 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 7 & \infty & 7 & 8 & \infty & 8 \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 7 & \infty & \infty & \infty & 8 & \infty
\end{vmatrix}
$$

Kiểm nghiệm thuật toán (2/4)

- **Bước 1.** T = ∅; D(T) = 0;
- **Bước 2.** Sắp xếp các cạnh theo thứ tự tăng dần của trọng số

Đầu	Cuối	Tr.Số
1	2	2
1	3	1
1	4	3
2	3	2
2	6	5
2	7	5
3	4	4
3	6	5
4	5	5
4	6	5
5	6	6
5	10	6
6	7	6
6	8	6
6	9	6
6	10	6
7	8	6
8	9	7
8	12	7
8	13	7
9	10	7
9	11	7
10	11	7
10	12	7
11	12	8
12	13	8

Đầu	Cuối	Tr.Số
1	3	1
1	2	2
2	3	2
1	4	3
3	4	4
2	6	5
2	7	5
3	6	5
4	5	5
4	6	5
5	6	6
5	10	6
6	7	6
6	8	6
6	9	6
6	10	6
7	8	6
8	9	7
8	12	7
8	13	7
9	10	7
9	11	7
10	11	7
10	12	7
11	12	8
12	13	8

Kiểm nghiệm thuật toán (3/4)

- **Bước 3.** (lặp).

STT	Cạnh được xét	$T \cup e$
1	E \(1,3)	$T = T \cup (1,3)$; $D(T) = 1$
2	E = E\(1,2)	$T = T \cup (1,2)$; $D(T) = 1+2 = 3$
3	E = E\(2,3)	*Tạo nên chu trình*
4	E = E\(1,4)	$T = T \cup (1,4)$; $D(T) = 3 + 3 = 6$
5	E = E\(3,4)	*Tạo nên chu trình*
6	E = E\(2,6)	$T = T \cup (2,6)$; $D(T) = 6+5=11$
7	E = E\(2,7)	$T = T \cup (2,7)$; $D(T) = 11+5 = 16$
8	E = E\(3,6)	*Tạo nên chu trình*
9	E = E\(4,5)	$T = T \cup (4,5)$; $D(T) = 16+5 = 21$
10	E = E\(4,6)	*Tạo nên chu trình*
11	E = E\(5,6)	*Tạo nên chu trình*
12	E = E\(5,10)	$T = T \cup (5,10)$; $D(T) = 21+6 = 27$
13	E = E\(6,7)	*Tạo nên chu trình*
14	E = E\(6,8)	$T = T \cup (6,8)$; $D(T) = 27+6 = 33$
15	E = E\(6,9)	$T = T \cup (6,9)$; $D(T) = 33+6 = 39$
16	E = E\(6,10)	*Tạo nên chu trình*
17	E = E\(7,8)	*Tạo nên chu trình*
18	E = E\(8,9)	*Tạo nên chu trình*
19	E = E\(8,12)	$T = T \cup (8,12)$; $D(T) = 39+7 = 46$
20	E = E\(8,13)	$T = T \cup (8,13)$; $D(T) = 46+7 = 53$
21	E = E\(9,10)	*Tạo nên chu trình*
22	E = E\(9,11)	$T = T \cup (9,11)$; $D(T) = 53+7 = 60$
Bước lặp kết thúc vì $\|T\| = N-1 = 12$		

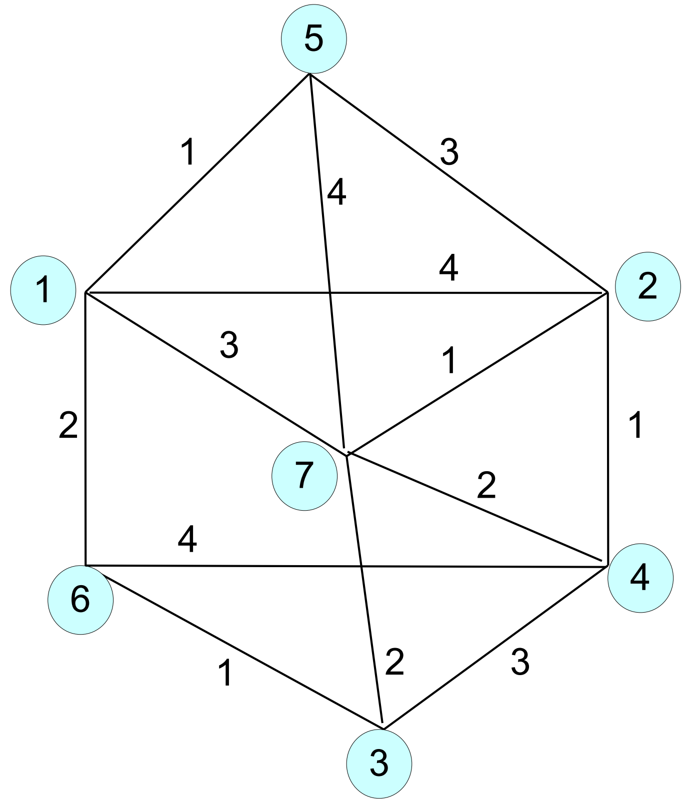
Kiểm nghiệm thuật toán (4/4)

- **Bước 4 :** Trả lại kết quả.

T = { (1,3), (1,2), (1,4), (2,6), 2,7), (4,5), (5,10), (6,8),(6,9), (8,12), (8,13), (9,11) }

D(T) = 1 + 2 + 3 + 5 + 5 + 5 + 6 + 6 + 6 + 7 + 7 +7 = 60

Ví dụ



- Vẽ bảng ghi nhận lại nhãn của các đỉnh trong các bước lặp của thuật toán Prim và cho biết tập cạnh cùng với chiều dài của cây khung nhỏ nhất trên đồ thị sau (giả sử đỉnh xuất phát là đỉnh 1)

Ví dụ (tt)

Bước lặp	**Đỉnh 1**	**Đỉnh 2**	**Đỉnh 3**	**Đỉnh 4**	**Đỉnh 5**	**Đỉnh 6**	**Đỉnh 7**	**T**
Khởi tạo	[0,1]	[4,1]	[,1]	[,1]	[1 ,1]	[2 ,1]	[3 ,1]	
1	-	[3,5]	[,1]	[,1]	-	[2,1]	[3,1]	(5,1)
2	-	[3,5]	[1,6]	[4,6]	-	-	[3,1]	(5,1), (1,6)
3	-	[3,5]	-	[3,3]	-	-	[2,3]	(5,1), (1,6), (3,6)
4	-	[1,7]	-	[2,7]	-	-	-	(5,1), (1,6), (3,6), (3,7)
5	-	-	-	[1,2]	-	-	-	(5,1), (1,6), (3,6), (3,7), (2,7)
6	-	-	-	-	-	-	-	(5,1), (1,6), (3,6), (3,7), (2,7), (4,2)

- Tập cạnh của cây khung nhỏ nhất: (5,1), (1,6), (3,6), (3,7), (2,7), (4,2)
- Chiều dài cây khung nhỏ nhất: 8

Bài tập 1

Cho đồ thị vô hướng được biểu diễn dưới dạng ma trận kề như Hình bên phải. Hãy thực hiện:

a) Trình bày thuật toán xây dựng một cây khung của đồ thị bắt đầu tại đỉnh u∈V dựa vào thuật toán BFS(u)?

b) Kiểm nghiệm thuật toán BFS(u) bắt đầu tại đỉnh u=1? Chỉ rõ kết quả trung gian theo mỗi bước thực hiện của thuật toán.

c) Kiểm nghiệm thuật toán BFS(u) bắt đầu tại đỉnh u=7? Chỉ rõ kết quả trung gian theo mỗi bước thực hiện của thuật toán.

d) Viết chương trình xây dựng cây khung của đồ thị bắt đầu tại đỉnh u∈V dựa vào thuật toán BFS(u)?

0	1	1	1	1	0	0	0	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	0	1	0	1	0	1	0	0	0	0
0	0	0	0	1	1	0	1	0	1	0	0	0
0	0	0	0	1	0	1	0	1	0	0	0	0
0	0	0	0	1	1	0	1	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	1	1	1
0	0	0	0	0	0	0	0	0	1	0	1	1
0	0	0	0	0	0	0	0	0	1	1	0	1
0	0	0	0	0	0	0	0	0	1	1	1	0

Bài tập 2

Ke(1) = { 2, 3, 4, 5 }.
Ke(2) = { 1, 3, 4 }.
Ke(3) = { 1, 2, 4 }.
Ke(4) = { 1, 2, 3 }.

Ke(5) = { 1, 6, 7, 8, 9 }.
Ke(6) = { 5, 7, 9 }.
Ke(7) = { 5, 6, 8, 10 }.
Ke(8) = { 5, 7, 9 }.

Ke(9) = { 5, 6, 8 }.
Ke(10) = { 7, 11, 12, 13 }.
Ke(11) = { 10, 12, 13 }.
Ke(12) = { 10, 11, 13 }.
Ke(13) = { 10, 11, 12 }.

Cho đồ thị vô hướng được biểu diễn dưới dạng danh sách kề như Hình bên trên. Hãy thực hiện:

a) Trình bày thuật toán xây dựng một cây khung của đồ thị bắt đầu tại đỉnh $u \in V$ dựa vào thuật toán DFS(u)?

b) Xây dựng cây khung của đồ thị bắt đầu tại đỉnh u=3? Chỉ rõ kết quả theo mỗi bước thực hiện của thuật toán?

c) Viết chương trình xây dựng cây khung của đồ thị bắt đầu tại đỉnh $u \in V$ dựa vào thuật toán DFS(u)?.

Bài tập 3

Cho đồ thị vô hướng có trọng số được biểu diễn dưới dạng ma trận kề như Hình bên phải. Hãy thực hiện:

a) Trình bày thuật toán Prim tìm cây khung nhỏ nhất trên đồ thị vô hướng có trọng số?

b) Áp dụng thuật toán, tìm cây khung nhỏ nhất tại đỉnh số 1 của đồ thị G, chỉ rõ kết quả theo từng bước thực hiện của thuật toán?

c) Viết chương trình tìm cây khung nhỏ nhất của đồ thị bằng thuật toán PRIM?

∞	2	1	3	∞	∞	∞	∞	∞	∞	∞	∞	∞
2	∞	2	∞	∞	5	5	∞	∞	∞	∞	∞	∞
1	2	∞	4	∞	5	∞	∞	∞	∞	∞	∞	∞
3	∞	4	∞	5	5	∞	∞	∞	∞	∞	∞	∞
∞	∞	∞	5	∞	6	∞	∞	∞	6	∞	∞	∞
∞	5	5	5	6	∞	6	6	6	6	∞	∞	∞
∞	5	∞	∞	∞	6	∞	6	∞	∞	∞	∞	∞
∞	∞	∞	∞	∞	6	6	∞	7	∞	∞	7	7
∞	∞	∞	∞	∞	6	∞	7	∞	7	7	∞	∞
∞	∞	∞	∞	6	6	∞	∞	7	∞	7	7	∞
∞	∞	∞	∞	∞	∞	∞	∞	7	7	∞	8	∞
∞	∞	∞	∞	∞	∞	∞	7	∞	7	8	∞	8
∞	∞	∞	∞	∞	∞	∞	7	∞	∞	∞	8	∞

Bài tập 4

Cho đồ thị vô hướng có trọng số được biểu diễn dưới dạng ma trận kề như Hình bên phải. Hãy thực hiện:

a) Trình bày thuật toán Kruskal tìm cây khung nhỏ nhất trên đồ thị vô hướng có trọng số?

b) Áp dụng thuật toán, tìm cây khung nhỏ nhất của đồ thị G, chỉ rõ kết quả theo từng bước thực hiện của thuật toán?

c) Viết chương trình tìm cây khung nhỏ nhất của đồ thị bằng thuật toán Kruskal?

∞	2	1	3	∞	∞	∞	∞	∞	∞	∞	∞	∞
2	∞	2	∞	∞	5	5	∞	∞	∞	∞	∞	∞
1	2	∞	4	∞	5	∞	∞	∞	∞	∞	∞	∞
3	∞	4	∞	5	5	∞	∞	∞	∞	∞	∞	∞
∞	∞	∞	5	∞	6	∞	∞	∞	6	∞	∞	∞
∞	5	5	5	6	∞	6	6	6	6	∞	∞	∞
∞	5	∞	∞	∞	6	∞	6	∞	∞	∞	∞	∞
∞	∞	∞	∞	∞	6	6	∞	7	∞	∞	7	7
∞	∞	∞	∞	∞	6	∞	7	∞	7	7	∞	∞
∞	∞	∞	∞	6	6	∞	∞	7	∞	7	7	∞
∞	∞	∞	∞	∞	∞	∞	∞	7	7	∞	8	∞
∞	∞	∞	∞	∞	∞	∞	7	∞	7	8	∞	8
∞	∞	∞	∞	∞	∞	∞	7	∞	∞	∞	8	∞