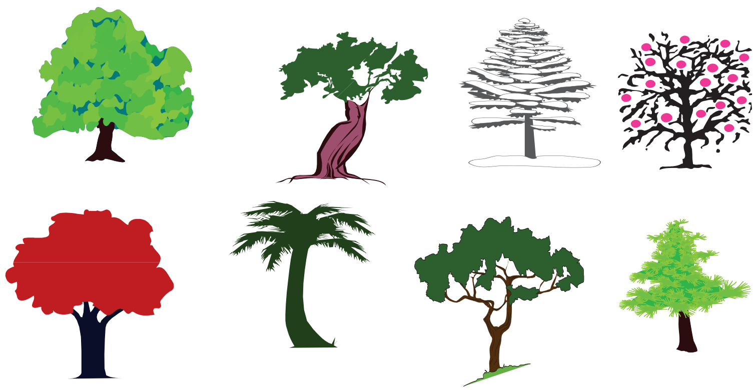
Chương 4. CÂY



NỘI DUNG

4.1. Cây và rừng

4.2. Cây có gốc

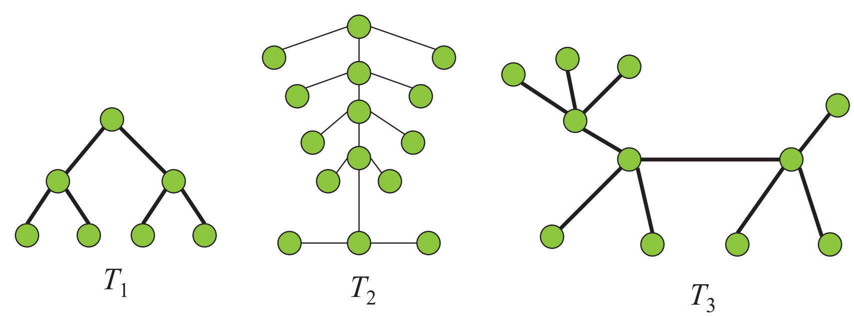
4.3. Bài toán cây khung nhỏ nhất

4.1. Cây và rừng

Định nghĩa 1. *Ta gọi cây là đồ thị vô hướng liên thông không có chu trình. Đồ thị không có chu trình được gọi là rừng.*

• Như vậy, rừng là đồ thị mà mỗi thành phần liên thông của nó là một cây.

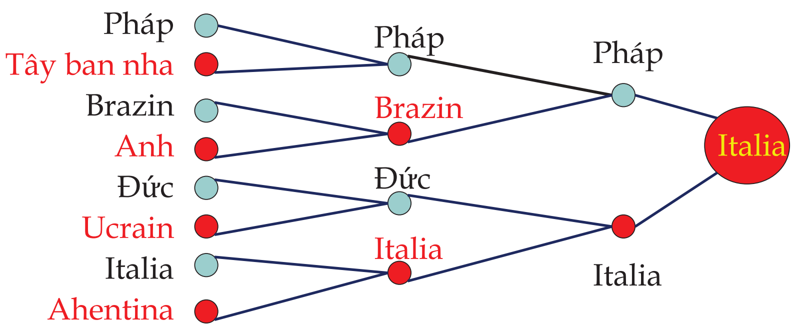
Ví dụ



Rừng F gồm 3 cây T_1, T_2, T_3

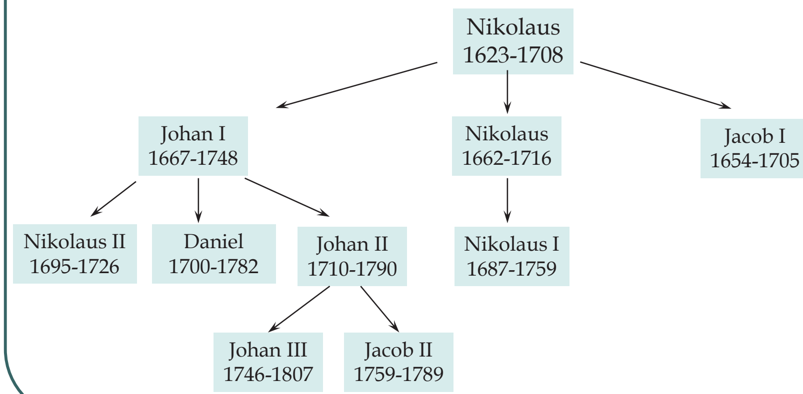
Cây lịch thi đấu

Trong đời thường cây rất hay được sử dụng để diễn tả lịch thi đấu của các giải thể thao theo thể thức đấu loại trực tiếp, chẳng hạn vòng 2 của World Cub

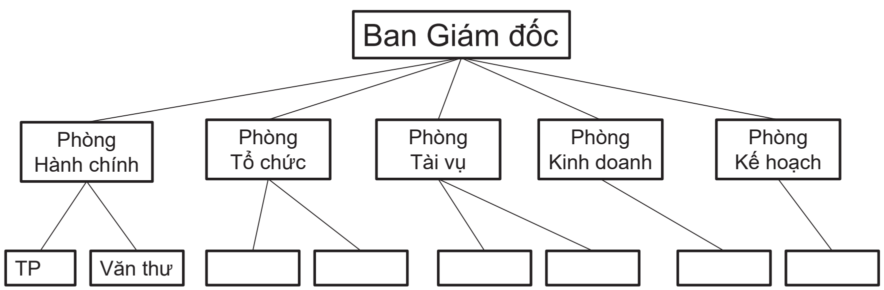


Cây gia phả

Cây gia phả của các nhà toán dòng họ Bernoulli



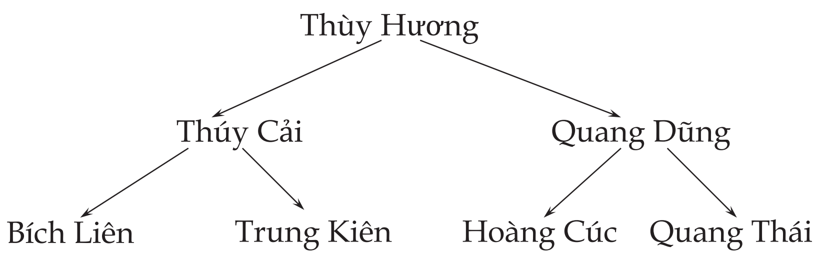
Cây phân cấp quản lý



Cây thư mục

Cây gia phả ngược (Ancestor Tree)

Cây phả hệ ngược: mỗi người đều có bố mẹ. Cây này là cây nhị phân (*binary* tree).



Các tính chất của cây

Định lý 1. *Giả sử T=(V,E) là đồ thị vô hướng n đỉnh. Khi đó các mệnh đề sau đây là tương đương:*

(1) *T là cây;*

(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 chứa chu trình nhưng hễ cứ thêm vào nó một cạnh ta thu được đúng một chu trình.*

Chương 4

4.1. Cây và rừng

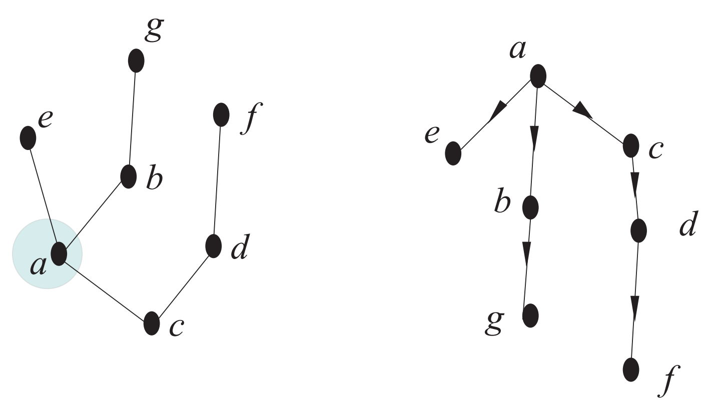
4.2. Cây có gốc

4.3. Bài toán cây khung nhỏ nhất

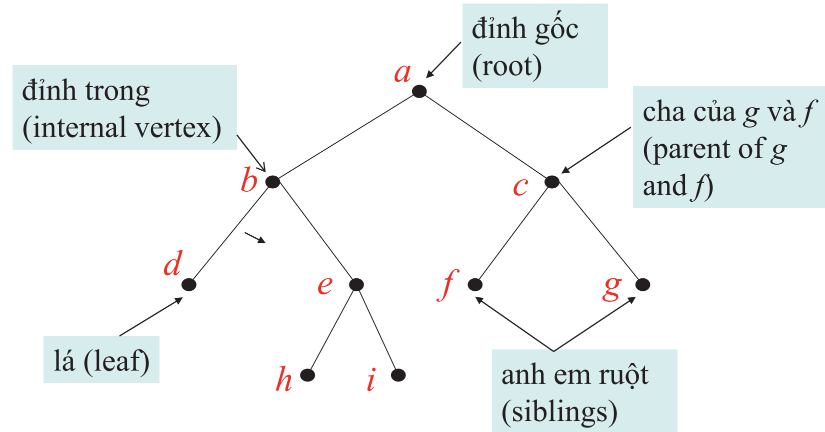
4.2. Cây có gốc (Rooted Trees)

Khi một đỉnh của cây được chọn làm gốc của cây, ta có thể định hướng cho các cạnh để thu được cây có gốc định hướng.

Cây có gốc

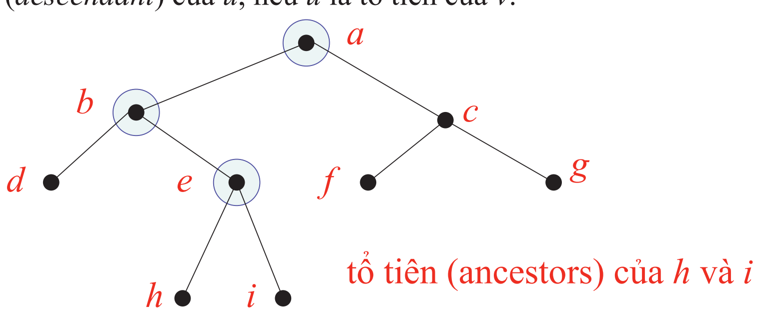


Thuật ngữ

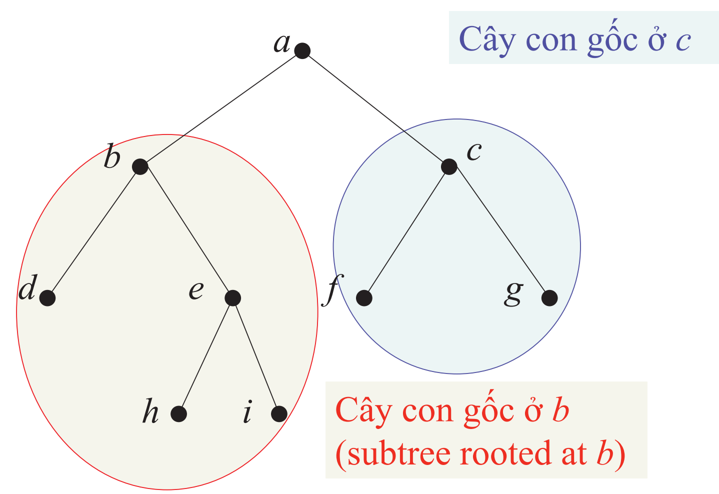


Thuật ngữ

Định nghĩa. Đỉnh *u* được gọi là *tổ tiên* (*ancestor*) của *v* nếu có đường đi từ gốc qua *u* đến *v*. Đỉnh *v* được gọi là *con cháu/hậu duệ* (*descendant*) của *u*, nếu *u* là tổ tiên của *v*.



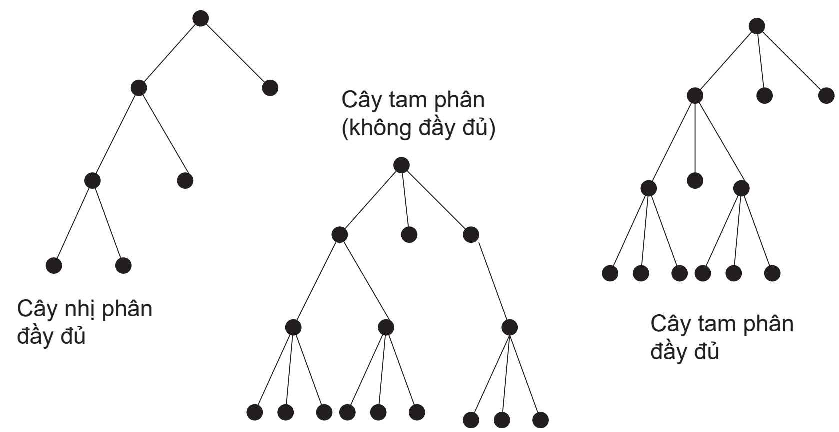
Thuật ngữ



Cây m-phân (*m*-ary trees)

Cây có gốc được gọi là cây *m*-phân (*m-ary tree*) nếu mỗi đỉnh trong có không quá *m* con. Cây được gọi là *m*-phân đầy đủ (*full m-ary tree*) nếu mỗi nút trong có đúng *m* con. Cây *m*-phân với *m*=2 được gọi là cây nhị phân (*binary tree*).

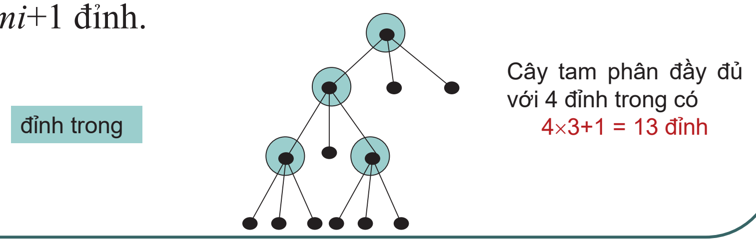
Ví dụ



Tính chất của cây có gốc

Mệnh đề 1. Cây *m*-phân đầy đủ với *i* đỉnh trong có $n = mi+1$ đỉnh.

CM. Mỗi đỉnh, ngoại trừ gốc, đều là con của một đỉnh trong. Mỗi một trong số *i* đỉnh trong có *m* con, suy ra *mi* là số lượng đỉnh của cây không kể đỉnh gốc. Suy ra, cây chứa $mi+1$ đỉnh.



Tính chất của cây m-phân

Mệnh đề 2.

(*i*) Cây *m*-phân với *n* đỉnh có

$i = (n-1)/m$ đỉnh trong

và $l = [(m-1)n+1]/m$ lá.

(*ii*) Cây *m*-phân với *i* đỉnh trong có

$n = mi + 1$ đỉnh và $l = (m-1)i + 1$ lá.

(*iii*) Cây *m*-phân với *l* lá có

$n = (ml - 1)/(m-1)$ đỉnh

và $i = (l-1)/(m-1)$ đỉnh trong.

Chứng minh mệnh đề 2

- Ta có: $n = l+i$, và theo mệnh đề 1 $n = mi+1$, trong đó
  - n – số đỉnh,
  - i – số lượng đỉnh trong
  - l – số lượng lá
- Từ đó ta có thể chứng minh mệnh đề 2 như sau:
- i) Từ $n = mi+1$ suy ra $i = (n-1)/m$.

  Từ $n = mi+1$ và $n = l+i$ suy ra

  $l = n - i = (mi+1)-i = (m-1)i+1$

Chứng minh mệnh đề 2 (tiếp)

- ii) Từ đẳng thức $n = mi+1$ suy ra

  $l = n - i = (mi+1)-i = (m-1)i+1.$
- iii) Cho m và l. Do $n = mi+1$ và $n = l+i$, ta suy ra $l + i = mi + 1$. Vì thế $i = (l-1)/(m-1)$.
- Do đó

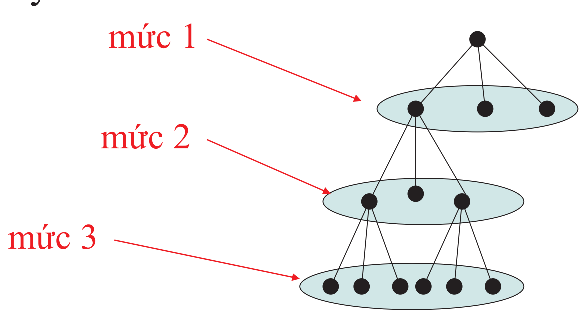
  $n = mi + 1$

  $= m(l-1)/(m-1) + 1$

  $= (ml-1)/(m-1).$

Mức (Level)

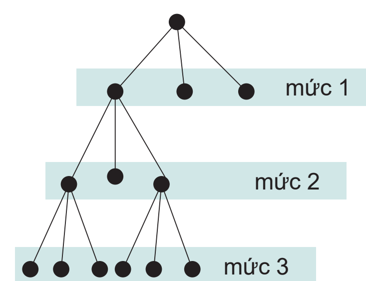
Ta gọi ***mức*** của đỉnh v trong cây có gốc là độ dài (số cạnh) của đường đi từ gốc đến đỉnh này.



Chiều cao của cây

Ta gọi ***chiều cao*** (*height*) của cây có gốc là giá trị lớn nhất của mức của các đỉnh.

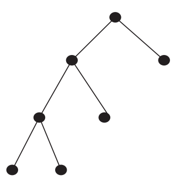
Chiều cao của cây là 3



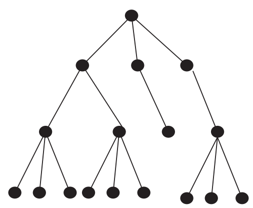
Cây cân bằng

Cây có gốc *m*-phân chiều cao *h* được gọi là ***cân bằng* (*balanced*)** nếu mọi lá đều ở mức *h* hoặc *h*-1.

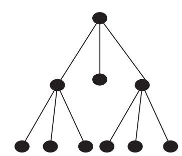
Cây nhị phân đầy đủ không cân bằng



Cây tam phân cân bằng (nhưng không đầy đủ)



Cây tam phân đầy đủ cân bằng



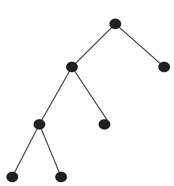
Tính chất

Mệnh đề 3. Có không quá m^h lá trong cây *m*-phân chiều cao *h*.

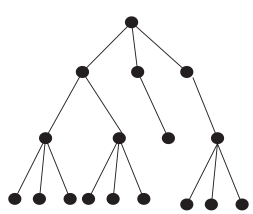
CM. Qui nạp theo chiều cao *h*. Với *h*=1, cây *m*-phân rõ ràng có không quá m^1 lá. Giả sử khẳng định đúng với cây *m*-phân độ cao *h*-1. Xét cây *m*-phân độ cao *h*. Loại bỏ gốc của cây này, ta thu được không quá *m* cây con, mỗi cây có độ cao không quá *h*-1. Vì thế số lượng lá của mỗi cây con, theo giả thiết qui nạp, là không vượt quá m^{h-1}. Suy ra, tổng số lượng lá của các cây con là không vượt quá $m*m^{h-1}=m^h$. Do số lượng lá của cây không vượt quá tổng số lượng lá của các cây con của nó, nên ta có đpcm.

Tính chất

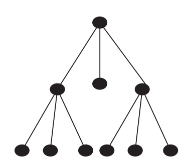
Có không quá m^h lá trong cây *m*-phân chiều cao *h*.



lá ≤ 2^3=8



lá ≤ 3^3=27



lá ≤ 3^2=9

Tính chất

Mệnh đề 4. Nếu cây *m*-phân chiều cao *h* có *l* lá thì

$$h \geq \lceil \log_m l \rceil$$

Nếu thêm vào đó cây là đầy đủ và cân bằng thì ta có dấu đẳng thức.

CM. Ta đã chứng minh $l \leq m^h$. Lấy log cơ số $m>1$ hai vế bất đẳng thức suy ra bất đẳng thức cần chứng minh.

Phần CM còn lại coi là bài tập.

Chương 4

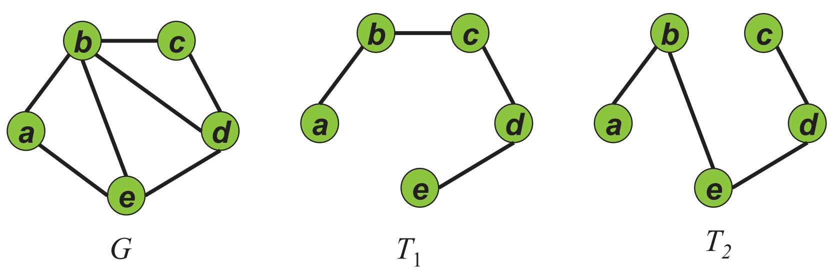
- 4.1. Cây và rừng
- 4.2. Cây có gốc
- **4.3. Bài toán cây khung nhỏ nhất**

4.3. Bài toán cây khung nhỏ nhất
(Minimum Spanning Trees)

- 4.3.1. Cây khung của đồ thị
- 4.3.2. Bài toán cây khung nhỏ nhất
- 4.3.3. Thuật toán Kruskal
- 4.3.4. Thuật toán Prim

4.3.1. Cây khung của đồ thị

- **Định nghĩa.** *Giả sử $G=(V,E)$ là đồ thị vô hướng liên thông. Cây $T=(V,E_T)$ với $E_T \subset E$ được gọi là cây khung của đồ thị G.*

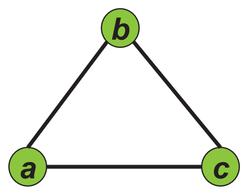


Đồ thị G và hai cây khung T_1 và T_2 của nó

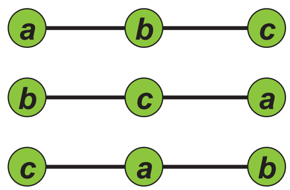
CÂY KHUNG

Số lượng cây khung của đồ thị

- Định lý sau đây cho biết số lượng cây khung của đồ thị đầy đủ K_n:
- Định lý 2 (Cayley). *Số cây khung của đồ thị K_n là n^{n-2}*
.

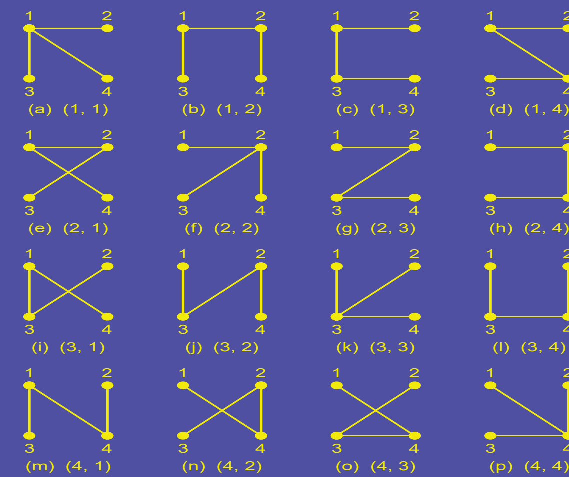


K_3



Ba cây khung của K_3

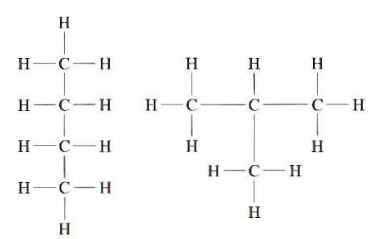
16 cây khung của K_4



Arthur Cayley
1821 - 1895

Bài toán trong hoá học hữu cơ

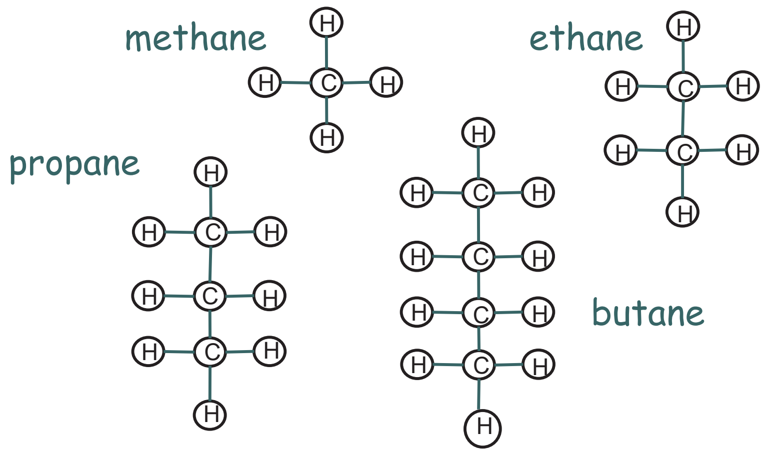
- Biểu diễn cấu trúc phân tử:
  - Mỗi đỉnh tương ứng với một nguyên tử
  - Cạnh – thể hiện liên kết giữa các nguyên tử



Butane   Isobutane

- Bài toán: Đếm số đồng phân của cacbua hydro no chứa một số nguyên tử cácbon cho trước

Saturated hydrocarbons C_nH_{2n+2}



Chứng minh định lý Cayley

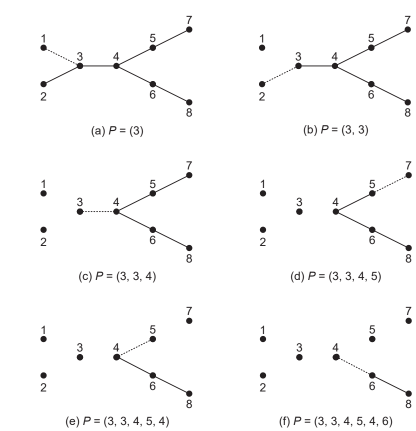
- Năm1918, Prüfer đưa ra chứng minh xây dựng cho định lý Cayley.
- **Định nghĩa.** Dãy Prüfer độ dài n-2 ($n \geq 2$) là một chỉnh hợp lặp chập n-2 từ các số 1, 2, ..., n.
- Có tất cả n^{n-2} dãy Prüfer độ dài n-2.
- Ta sẽ chỉ ra tương ứng một-một giữa một dãy Prüfer và một cây cây khung của đồ thị đầy đủ K_n.

Ernst Paul Heinz Prüfer
(1896–1934)
German mathematician

Thuật toán mã hóa Prüfer (Prüfer_Encoding)

Đầu vào: Cây T với các đỉnh được đánh số bởi 1, 2, …, n
Đầu ra: Dãy Prüfer tương ứng với T
P = ( ); // Khởi tạo dãy rỗng
for k = 1.. n–2 {
  Chọn v là đỉnh có chỉ số nhỏ nhất trong số các đỉnh có bậc bằng 1;
  P = (P, neigh(v)); // Nạp neigh(v) - là đỉnh kề với v vào dãy đầu ra;
  Loại v khỏi cây;
}
return P;

Ví dụ: Mã hóa cây gồm 8 đỉnh



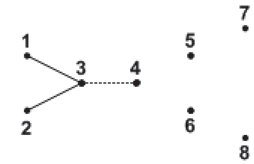
- Mã Prüfer của cây khung là

  P = (3, 3, 4, 5, 4, 6)

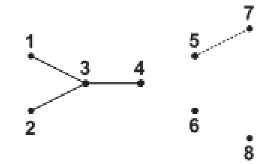
Thuật toán giải mã Prüfer (Prüfer_Decoding)

Đầu vào: Dãy Prüfer $P = (p_1, p_2, \ldots, p_{n-2})$.
Đầu ra: Cây T với các đỉnh 1, 2, …, n tương ứng với dãy P.
$n = | P | + 2$;
Đặt $L = (1, 2, ..., n)$;
Bắt đầu với đồ thị T gồm n đỉnh cô lập được gán nhãn từ 1 đến n.
for $i = 1.. n-2$
  Tìm v là chỉ số nhỏ nhất trong L không có mặt trong P;
  Thêm cạnh (v, p_i) vào đồ thị T;
  Loại bỏ v khỏi L; // L còn n-i phần tử
  Loại bỏ p_i khỏi P; // tại đây $P = \{p_{i+1}, p_{i+2}, \ldots, p_{n-2}\}$
Kết thúc vòng lặp, trong danh sách L còn hai số, chẳng hạn u và v. Thêm cạnh (u, v) vào T.
return T;

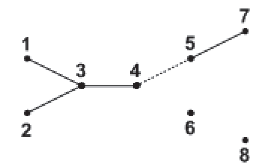
Ví dụ: Giải mã dãy *P* = (3, 3, 4, 5, 4, 6)



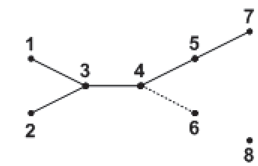
(c) P = (4, 5, 4, 6); L={3, 4, 5, 6, 7, 8}



(d) P = (5, 4, 6); L={4, 5, 6, 7, 8}

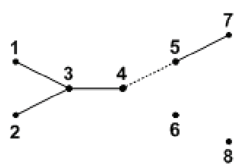


(e) P = (4, 6); L={4, 5, 6, 8}

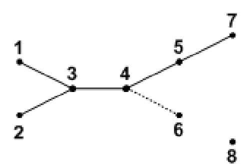


(f) P = (6); L={4, 6, 8}

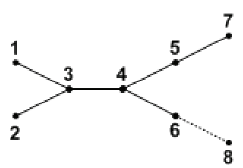
Ví dụ: Giải mã dãy *P* = (3, 3, 4, 5, 4, 6)



(e) P = (4, 6); L={4, 5, 6, 8}



(f) P = (6); L={4, 6, 8}



(g) P = ( ); L={6, 8}

Chứng minh định lý Cayley

- Dễ dàng chứng minh được rằng: Mã hóa và giải mã Prüfer là hai phép toán đảo ngược nhau, nghĩa là có một tương ứng một - một giữa một cây có nhãn gồm n đỉnh với một dãy Prüfer độ dài n–2.
- Từ đó suy ra định lý Cayley

4.3. Bài toán cây khung nhỏ nhất
(Minimum Spanning Trees)

4.3.1. Cây khung của đồ thị

4.3.2. Bài toán cây khung nhỏ nhất

4.3.3. Thuật toán Kruskal

4.3.4. Thuật toán Prim

Bài toán cây khung nhỏ nhất

- 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 G được gán với một số thực $c(e)$, gọi là độ dài của nó.
- Ta gọi độ dài $c(T)$ của cây khung $T=(V,E_T)$ là tổng độ dài của các cạnh của nó:

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

Bài toán cây khung nhỏ nhất

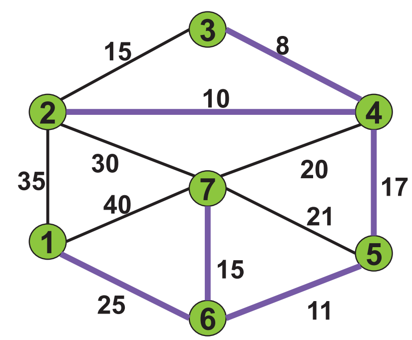
- ***Bài toán cây khung nhỏ nhất:*** Trong số tất cả các cây khung của G hãy tìm cây khung với độ dài nhỏ nhất. Cây khung như vậy được gọi là ***cây khung nhỏ nhất*** của đồ thị.
- Có thể phát biểu dưới dạng bài toán tối ưu tổ hợp:
- Tìm cực tiểu

$$c(T) = \sum_{e \in E_T} c(e) \rightarrow \min$$

với điều kiện $T=(V,E_T)$ là cây khung của G.

- Do số lượng của cây khung của G là rất lớn (xem định lý Caylley), nên không thể giải được nhờ duyệt toàn bộ các cây khung.

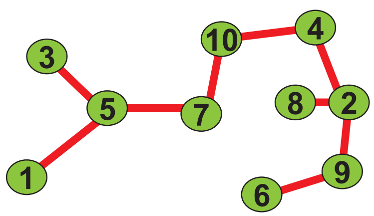
Ví dụ



- Cây khung nhỏ nhất gồm các cạnh màu tím
- Độ dài: 8+10+11+15+17+25 = 86

Ứng dụng thực tế: Mạng truyền thông

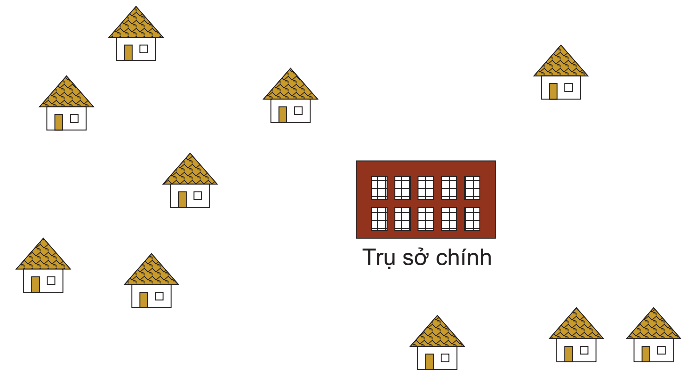
Công ty truyền thông AT&T cần xây dựng mạng truyền thông kết nối n khách hàng. Chi phí thực hiện kênh nối i và j là c_{ij}. Hỏi chi phí nhỏ nhất để thực hiện việc kết nối tất cả các khách hàng là bao nhiêu?



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

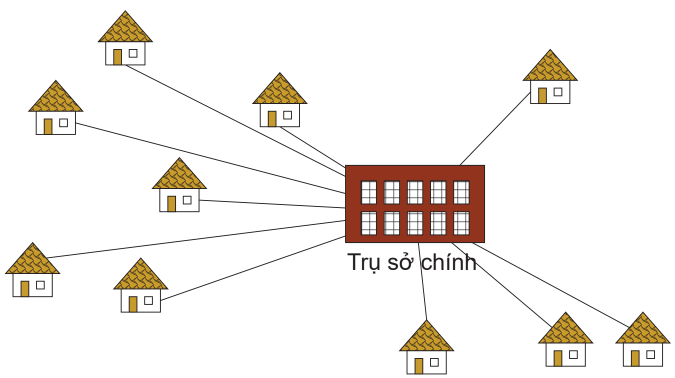
Ứng dụng: Kết nối mạng điện thoại

(Laying Telephone Wire)



Kết nối: Tiếp cận trực tiếp

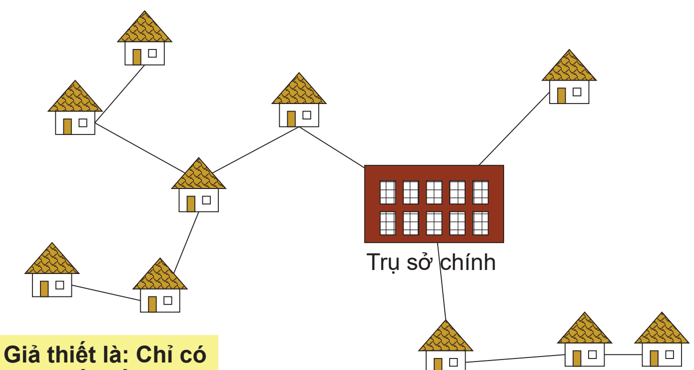
(Naive Approach)



Tốn kém!
Expensive!

Kết nối: Cách làm tốt hơn

(Better Approach)



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

Cực tiểu hóa tổng độ dài đường dây kết nối

4.3. Bài toán cây khung nhỏ nhất
(Minimum Spanning Trees)

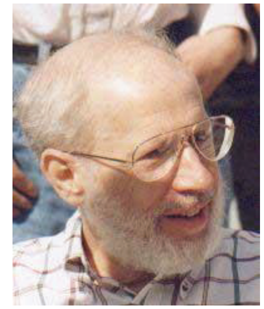
4.3.1. Cây khung của đồ thị

4.3.2. Bài toán cây khung nhỏ nhất

4.3.3. Thuật toán Kruskal

4.3.4. Thuật toán Prim

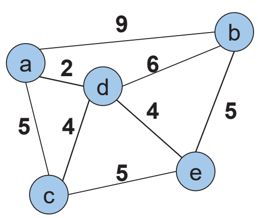
Thuật toán Kruskal

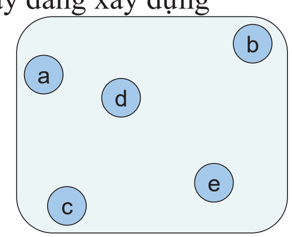


Joseph Bernard Kruskal, Jr.
(born January 29, 1928)
is an American mathematician, statistician, and psychometrician.

Ý tưởng thuật toán

- Tạo rừng gồm n cây, mỗi cây gồm một đỉnh của đồ thị
- Lặp lại thao tác nối các cây bằng cách bổ sung cạnh "an toàn" cho đến khi thu được một cây duy nhất.
- Cạnh “an toàn” là cạnh có trọng số nhỏ nhất mà việc bổ sung nó không tạo thành chu trình trong cây đang xây dựng





Rừng: {a}, {b}, {c}, {d}, {e}

Thuật toán Kruskal

Đầu vào:

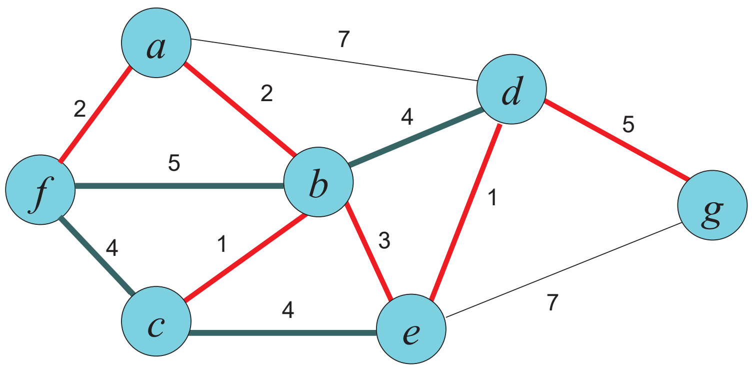
E = tập cạnh của đồ thị;
n = số đỉnh của đồ thị.

Đầu ra: E_T = tập cạnh của cây khung nhỏ nhất cần xây dựng.

Thuật toán Kruskal

$E_T = \varnothing$;
while E_T có ít hơn n - 1 cạnh {
  Loại cạnh (v,w) có trọng số nhỏ nhất khỏi E;
  if $E_T \cup \{(v,w)\}$ không chứa chu trình
  then $E_T = E_T \cup \{(v, w)\}$ //bổ sung (v,w) vào E_T
}

Thuật toán Kruskal – Ví dụ

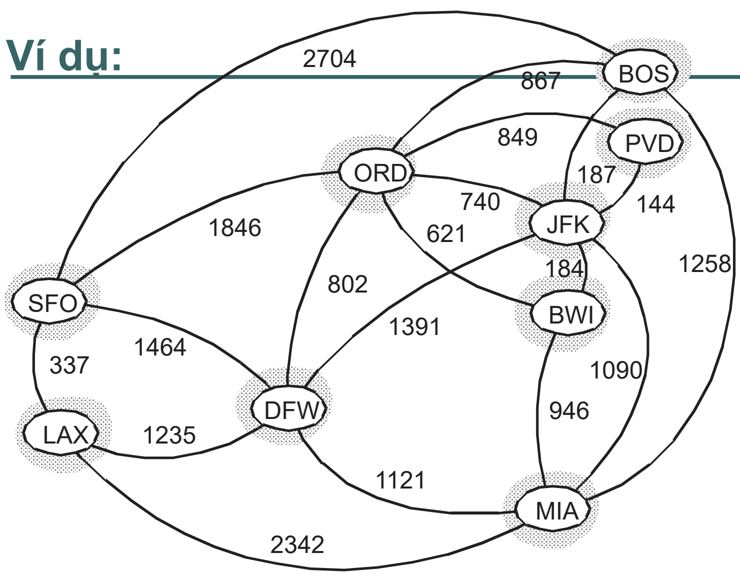


- Tập cạnh của cây khung nhỏ nhất:
  $E_T = \{(b,c),(e,d), (a,b),(a,f),(b,e),(d,g)\}$
- Độ dài của CKNN: 14

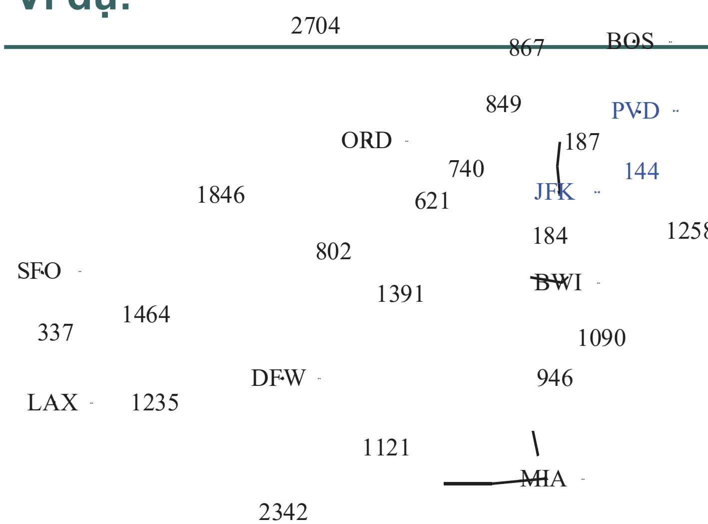
Cài đặt thuật toán Kruskal

- Để có thể cài đặt hiệu quả thuật toán Kruskal cần chú ý việc cài đặt hai thao tác:
  - Thao tác tìm cạnh có trọng số nhỏ nhất: Để thực hiện hiệu quả thao tác này, thông thường ta sắp xếp các cạnh theo thứ tự không giảm của trọng số
  - Kiểm tra $E_T \cup \{(v,w)\}$ không chứa chu trình: Có thể thực hiện hiệu quả nhờ sử dụng cấu trúc dữ liệu các tập không giao nhau (Disjoint Set) và thuật toán Nối-Tìm (UNION-FIND).

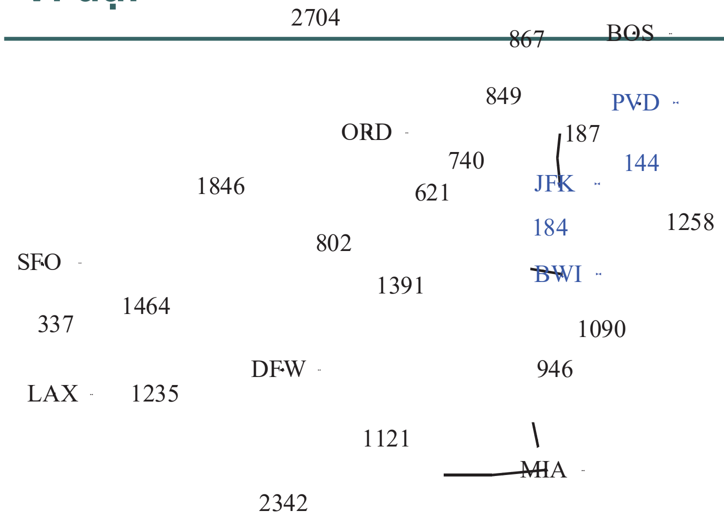
Ví dụ:



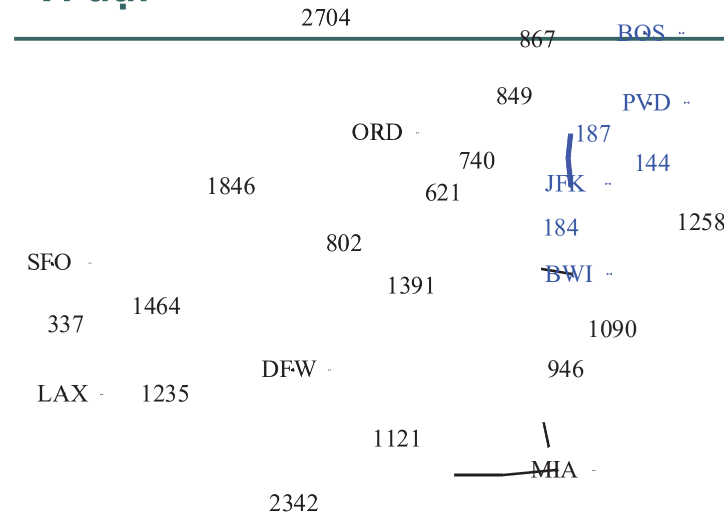
Ví dụ:



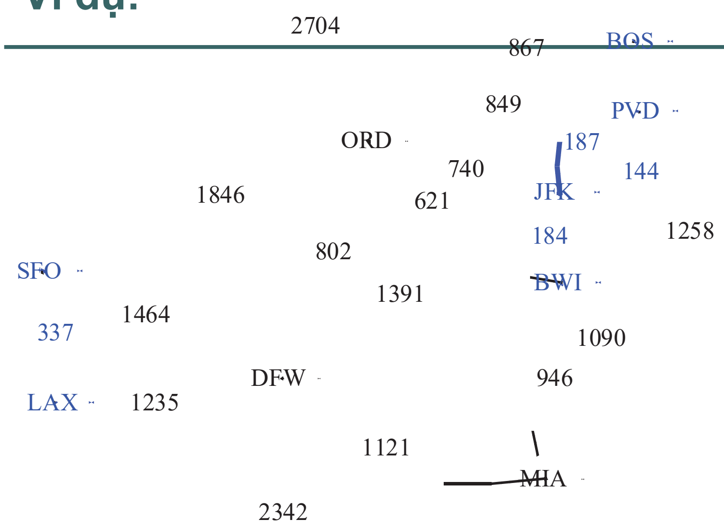
Ví dụ:



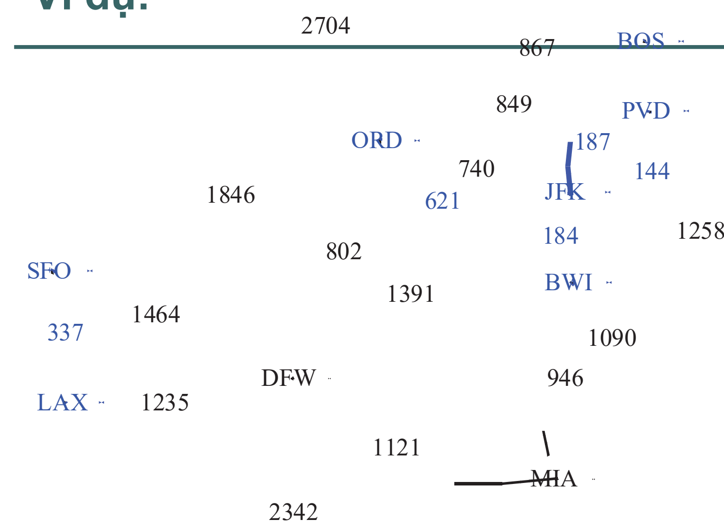
Ví dụ:



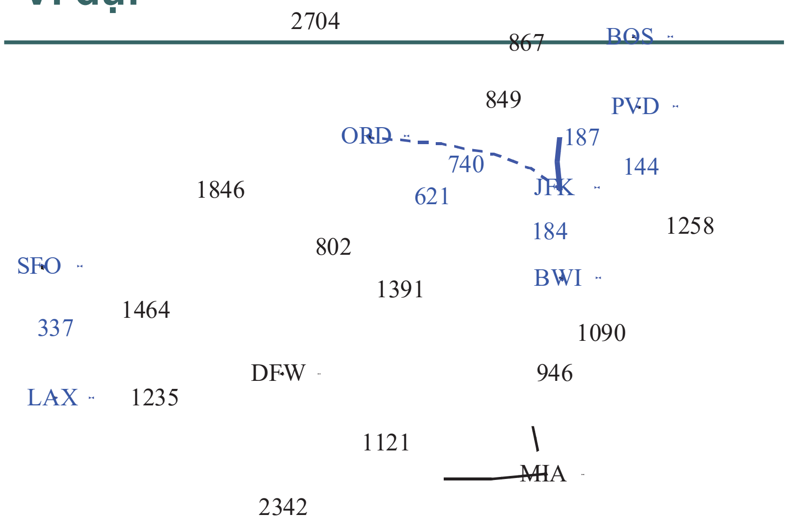
Ví dụ:



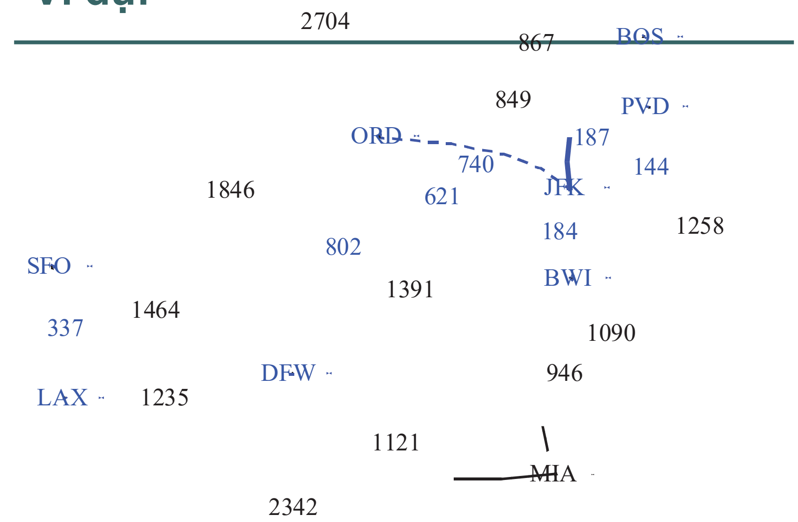
Ví dụ:



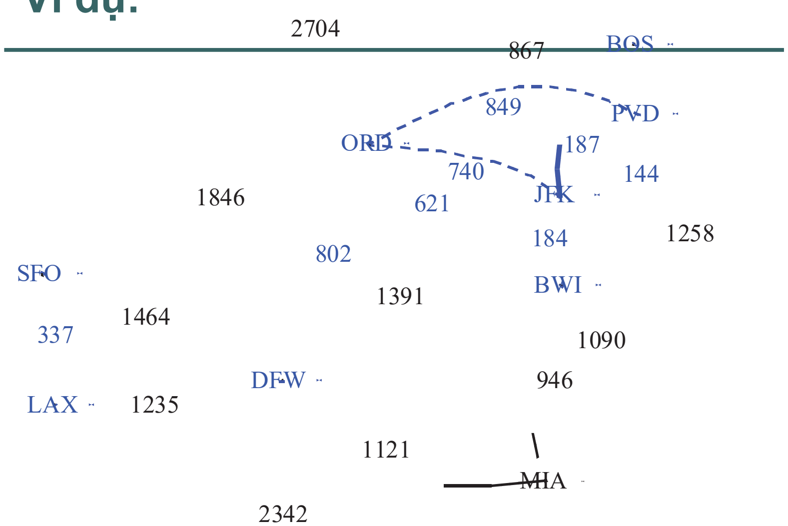
Ví dụ:



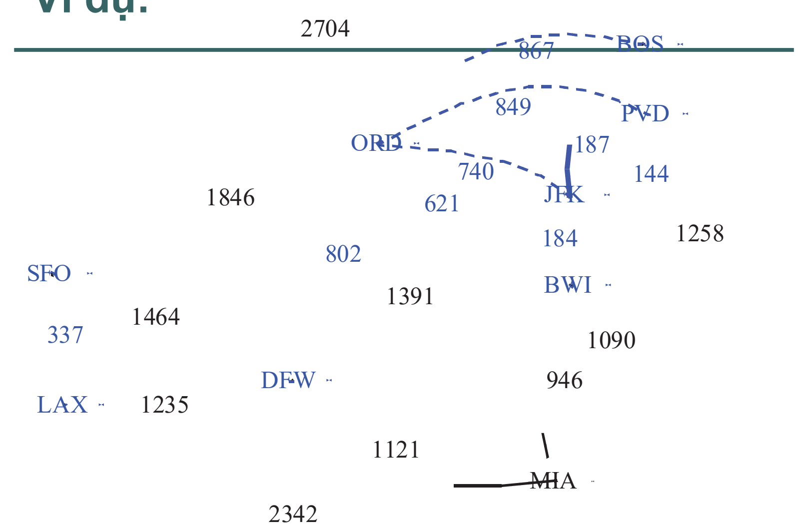
Ví dụ:



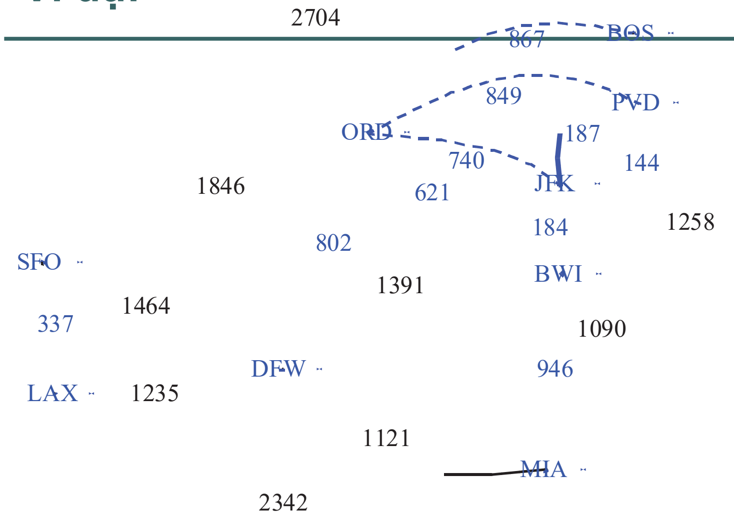
Ví dụ:



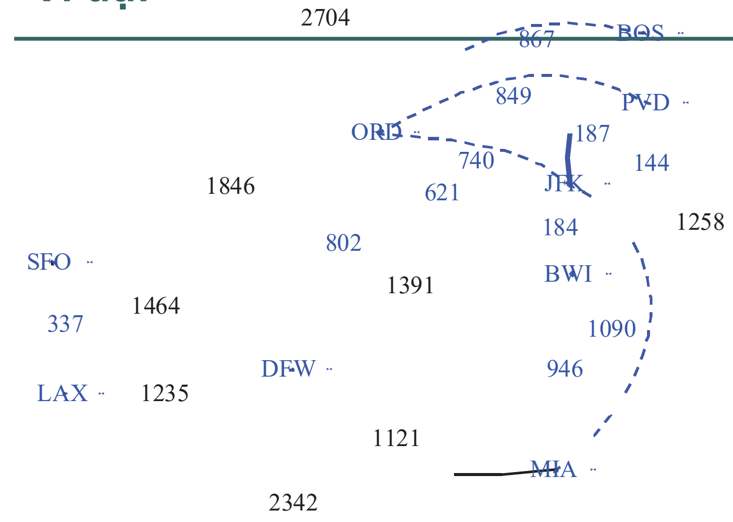
Ví dụ:



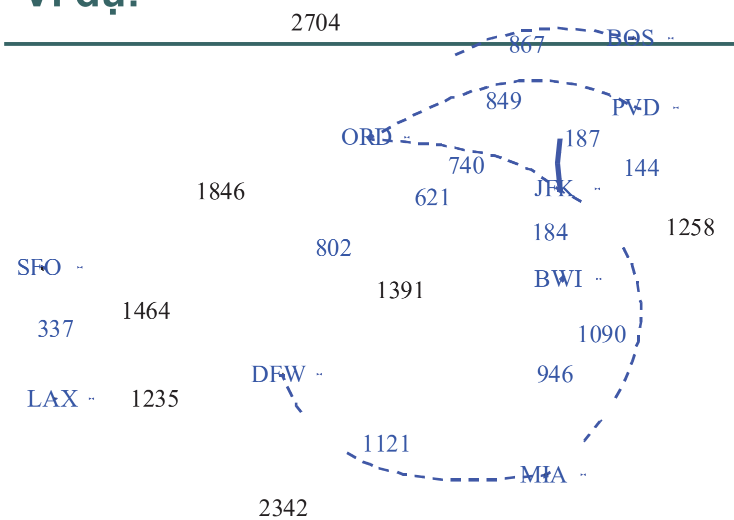
Ví dụ:



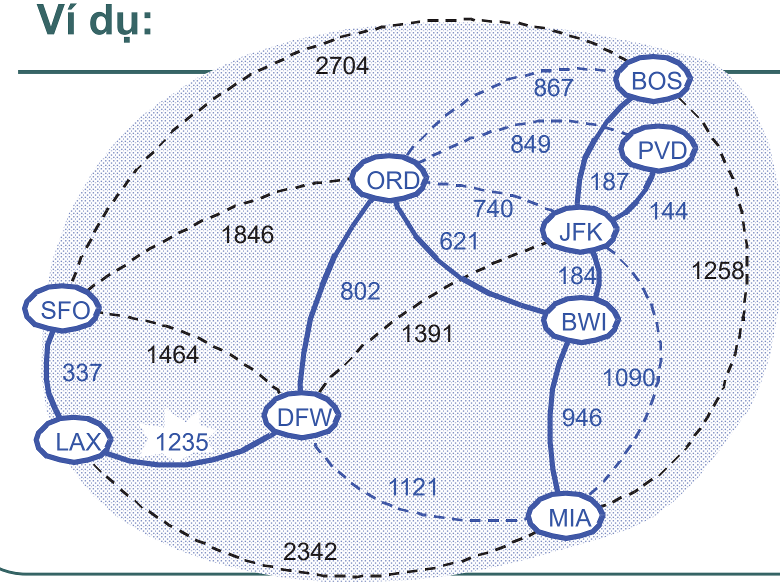
Ví dụ:



Ví dụ:



Ví dụ:



Kết quả tính toán

- Tập cạnh của cây khung nhỏ nhất:

  Các cạnh tô xanh đậm

- Độ dài của cây khung nhỏ nhất:

144 + 184 + 187 + 337 + 621 + 802 + 946 + 1235 = 4456

4.3. Bài toán cây khung nhỏ nhất
(Minimum Spanning Trees)

4.3.1. Cây khung của đồ thị

4.3.2. Bài toán cây khung nhỏ nhất

4.3.3. Thuật toán Kruskal

4.3.4. Thuật toán Prim

Thuật toán Prim

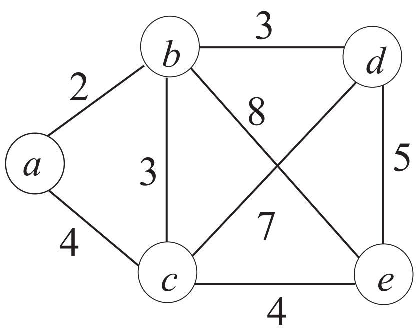
Robert Clay Prim
(born 1921 in Sweetwater, Texas)
is an American mathematician and computer scientist.

Thuật toán Prim

- **Bước khởi tạo.** Chọn v là một đỉnh tuỳ ý. Gọi T_0 là cây gồm đỉnh v và không có cạnh.
- **Bước k=1, 2, ..., n-1.** Trong số các cạnh có một đầu mút thuộc T_{k-1} còn đầu mút kia không thuộc T_{k-1} chọn cạnh có trọng số nhỏ nhất. Gọi cạnh tìm được là e_k. Bổ sung cạnh e_k vào T_{k-1} ta thu được cây con T_k.
- Kết thúc thuật toán ta thu được cây T_{n-1} là cây khung nhỏ nhất cần tìm.

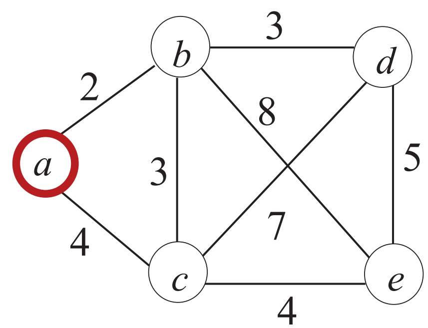
Ví dụ

- Tìm cây khung nhỏ nhất của đồ thị có trọng số trên cạnh sau đây theo thuật toán Prim



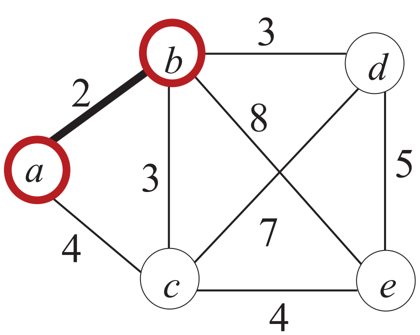
Bước khởi tạo

- Chọn đỉnh a. Cây T_0 gồm duy nhất đỉnh a và không có cạnh.



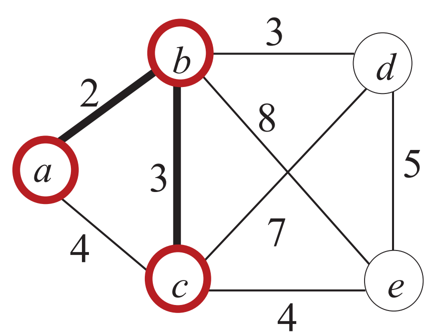
Bước 1

- Trong 2 cạnh (a,b) và (a,c) chọn cạnh có trọng số nhỏ hơn (a,b) bổ sung vào T_0 thu được cây T_1.



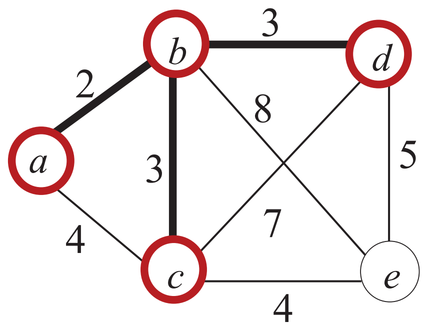
Bước 2

- Trong số các cạnh có một đầu mút trong T_1: (a,c), (b,c), (b,d), (b,e) chọn cạnh có trọng số nhỏ nhất (b,c) bổ sung vào T_1 thu được cây T_2.



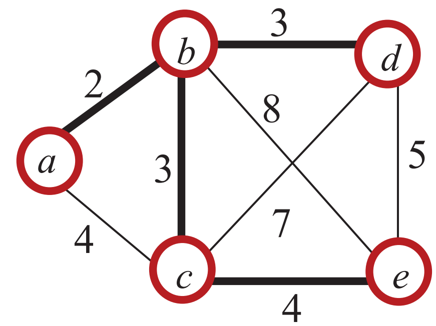
Bước 3

- Trong số các cạnh có một đầu mút trong T_2: (b,d), (b,e), (c,d), (c,e) chọn cạnh có trọng số nhỏ nhất (b,d) bổ sung vào T_2 thu được cây T_3.



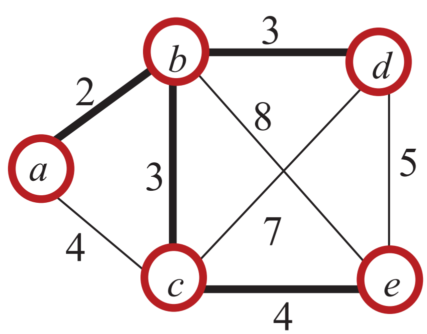
Bước 4

- Trong số các cạnh có một đầu mút trong T_3: (b,e), (c,e), (d,e) chọn cạnh có trọng số nhỏ nhất (c,e) bổ sung vào T_3 thu được cây T_4.



Kết quả

- Thu được cây T_4 là cây khung nhỏ nhất



Mô tả thuật toán Prim

procedure Prim(*G*, *c*)
begin
  Chọn đỉnh tuỳ ý $r \in V$;
  Khởi tạo cây $T=(V(T), E(T))$ với $V(T)=\{ r \}$ và $E(T)=\varnothing$;
  while *T* có $< n$ đỉnh **do**
  begin
    Gọi (u, v) là cạnh nhẹ nhất với $u \in V(T)$ và $v \in V(G) - V(T)$
    $E(T) \leftarrow E(T) \cup \{ (u, v) \}$; $V(T) \leftarrow V(T) \cup \{ v \}$
  end
end;

Cài đặt thuật toán Prim đối với đồ thị dày

- Giả sử đồ thị cho bởi ma trận trọng số $C=\{c[i,j],\ i, j = 1, 2,\ldots, n\}$.
- Ở mỗi bước để nhanh chóng chọn đỉnh và cạnh cần bổ sung vào cây khung, các đỉnh của đồ thị sẽ được gán cho các nhãn.
- Nhãn của một đỉnh $v \in V\text{-}S$ có dạng $[d[v], near[v]]$:

  $d[v]$ dùng để ghi nhận khoảng cách từ đỉnh v đến tập đỉnh S:

  $$d[v] := \min\{\ c[v, w] :\ w \in S\ \}\ (= c[v, z]),$$

  $near[v] := z$ ghi nhận đỉnh của cây khung gần v nhất

Thuật toán Prim

```
procedure  Prim;
    (*   Bước khởi tạo    *)
    S := { r }; T := ∅ ; d[r] := 0; near[r] := r.
    for  v ∈ V \ S  do  begin
        d[v] := c[r,v];  near[v] := r;
    end;
    (*   Bước lặp    *)
    for k:=2 to n do
    begin
        Tìm  u ∈ V\ S  thoả mãn:  d[u] = min { d[v] : v ∈ V\ S };
        S :=  S ∪ { u };   T := T ∪ { ( u, near[u] ) } ;
        for  v∈ V\ S   do
            if  d[v] > c[u,v]  then   begin
                                  d[v] := c[u,v] ;  near[v] :=  u;
                                end;
    end;
    H = ( S , T )  là cây khung nhỏ nhất ;
```

Thời gian tính: $O(|V|^2)$

Thuật toán Prim – Ví dụ

- Ví dụ: Tìm CKNN cho đồ thị cho bởi ma trận trọng số

$$C = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left| \begin{array}{cccccc} 0 & 33 & 17 & \infty & \infty & \infty \\ 33 & 0 & 18 & 20 & \infty & \infty \\ 17 & 18 & 0 & 16 & 4 & \infty \\ \infty & 20 & 16 & 0 & 9 & 8 \\ \infty & \infty & 4 & 9 & 0 & 14 \\ \infty & \infty & \infty & 8 & 14 & 0 \end{array} \right| \end{array}$$

Thuật toán Prim: Ví dụ

Bước	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh 5	Đỉnh 6	S
Khởi tạo							
1							
2							
3							
4							
5							

$$C = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left| \begin{array}{cccccc} 0 & 33 & 17 & \infty & \infty & \infty \\ 33 & 0 & 18 & 20 & \infty & \infty \\ 17 & 18 & 0 & 16 & 4 & \infty \\ \infty & 20 & 16 & 0 & 9 & 8 \\ \infty & \infty & 4 & 9 & 0 & 14 \\ \infty & \infty & \infty & 8 & 14 & 0 \end{array} \right| \end{array}$$

Thuật toán Prim: Ví dụ

	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh 5	Đỉnh 6	S
Khởi tạo	[0, 1]	[33, 1]	[17, 1]*	[∞, 1]	[∞, 1]	[∞, 1]	1
1							
2							
3							
4							
5							

$$C = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left| \begin{array}{cccccc} 0 & 33 & 17 & \infty & \infty & \infty \\ 33 & 0 & 18 & 20 & \infty & \infty \\ 17 & 18 & 0 & 16 & 4 & \infty \\ \infty & 20 & 16 & 0 & 9 & 8 \\ \infty & \infty & 4 & 9 & 0 & 14 \\ \infty & \infty & \infty & 8 & 14 & 0 \end{array} \right| \end{array}$$

Thuật toán Prim: Ví dụ

	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh 5	Đỉnh 6	S
Khởi tạo	[0, 1]	[33, 1]	[17, 1]*	[∞, 1]	[∞, 1]	[∞, 1]	1
1	-	[18, 3]	-	[16, 3]	[4, 3]*	[∞, 1]	1, 3
2							
3							
4							
5							

```
for  v∈ V\ S   do
     if  d[v] > c[u,v]  then
                    d[v] := c[u,v] ;
                    near[v] :=  u;
```

$$C = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left| \begin{array}{cccccc} 0 & 33 & 17 & \infty & \infty & \infty \\ 33 & 0 & 18 & 20 & \infty & \infty \\ 17 & 18 & 0 & 16 & 4 & \infty \\ \infty & 20 & 16 & 0 & 9 & 8 \\ \infty & \infty & 4 & 9 & 0 & 14 \\ \infty & \infty & \infty & 8 & 14 & 0 \end{array} \right| \end{array}$$

Thuật toán Prim: Ví dụ

	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh 5	Đỉnh 6	S
Khởi tạo	[0, 1]	[33, 1]	[17, 1]*	[∞, 1]	[∞, 1]	[∞, 1]	1
1	-	[18, 3]	-	[16, 3]	[4, 3]*	[∞, 1]	1, 3
2	-	[18, 3]	-	[9,5]*	-	[14, 5]	1, 3, 5
3							
4							
5							

```
for  v∈ V\ S   do
     if  d[v] > c[u,v]  then
                    d[v] := c[u,v] ;
                    near[v] :=  u;
```

$$C = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left| \begin{array}{cccccc} 0 & 33 & 17 & \infty & \infty & \infty \\ 33 & 0 & 18 & 20 & \infty & \infty \\ 17 & 18 & 0 & 16 & 4 & \infty \\ \infty & 20 & 16 & 0 & 9 & 8 \\ \infty & \infty & 4 & 9 & 0 & 14 \\ \infty & \infty & \infty & 8 & 14 & 0 \end{array} \right| \end{array}$$

Thuật toán Prim: Ví dụ

	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh 5	Đỉnh 6	S
Khởi tạo	[0, 1]	[33, 1]	[17, 1]*	[∞, 1]	[∞, 1]	[∞, 1]	1
1	-	[18, 3]	-	[16, 3]	[4, 3]*	[∞, 1]	1, 3
2	-	[18, 3]	-	[9,5]*	-	[14, 5]	1, 3, 5
3	-	[18,3]	-	-	-	[8,4]*	1,3,5,4
4							
5							

```
for  v∈ V\ S   do
     if  d[v] > c[u,v]  then
                    d[v] := c[u,v] ;
                    near[v] :=  u;
```

$$C = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \left| \begin{array}{cccccc} 0 & 33 & 17 & \infty & \infty & \infty \\ 33 & 0 & 18 & 20 & \infty & \infty \\ 17 & 18 & 0 & 16 & 4 & \infty \\ \infty & 20 & 16 & 0 & 9 & 8 \\ \infty & \infty & 4 & 9 & 0 & 14 \\ \infty & \infty & \infty & 8 & 14 & 0 \end{array} \right| \end{array}$$

Thuật toán Prim: Ví dụ

	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh 5	Đỉnh 6	S
Khởi tạo	[0, 1]	[33, 1]	[17, 1]*	[∞, 1]	[∞, 1]	[∞, 1]	1
1	-	[18, 3]	-	[16, 3]	[4, 3]*	[∞, 1]	1, 3
2	-	[18, 3]	-	[9,5]*	-	[14, 5]	1, 3, 5
3	-	[18,3]	-	-	-	[8,4]*	1,3,5,4
4	-	[18,3]*	-	-	-	-	1,3,5,4,6
5							

```
for  v∈ V\ S   do
      if  d[v] > c[u,v]  then
                    d[v] := c[u,v] ;
                    near[v] :=  u;
```

$$C = \begin{array}{c|cccccc|} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 0 & 33 & 17 & \infty & \infty & \infty \\ 2 & 33 & 0 & 18 & 20 & \infty & \infty \\ 3 & 17 & 18 & 0 & 16 & 4 & \infty \\ 4 & \infty & 20 & 16 & 0 & 9 & 8 \\ 5 & \infty & \infty & 4 & 9 & 0 & 14 \\ 6 & \infty & \infty & \infty & 8 & 14 & 0 \end{array}$$

Thuật toán Prim: Ví dụ

	Đỉnh 1	Đỉnh 2	Đỉnh 3	Đỉnh 4	Đỉnh 5	Đỉnh 6	S
Khởi tạo	[0, 1]	[33, 1]	[17, 1]*	[∞, 1]	[∞, 1]	[∞, 1]	1
1	-	[18, 3]	-	[16, 3]	[4, 3]*	[∞, 1]	1, 3
2	-	[18, 3]	-	[9,5]*	-	[14, 5]	1, 3, 5
3	-	[18,3]	-	-	-	[8,4]*	1,3,5,4
4	-	[18,3]*	-	-	-	-	1,3,5,4,6
5	-	-	-	-	-	-	1,3,5,4,6,2

Độ dài của CKNN : 18 + 17 + 9 + 4 + 8 = 56

Tập cạnh của CKNN: {(2,3), (3,1), (4,5), (5,3), (6,4)}

Người đề xuất bài toán MST

Otakar Borůvka

- Nhà khoa học Séc (Czech)
- Người đề xuất bài toán
- Đề xuất thuật toán thời gian *O*(*m* log *n*)
- Bài báo được xuất bản ở Séc từ năm 1926.
- Ứng dụng vào việc phát triển hệ thống mạng điện ở Bohemia.

Tăng tốc

- $O(m \log n)$ Borůvka, Prim, Dijkstra, Kruskal,…
- $O(m \log \log n)$ Yao (1975), Cheriton-Tarjan (1976)
- $O(m\, \beta(m, n))$ Fredman-Tarjan (1987)
- $O(m \log \beta(m, n))$ Gabow-Galil-Spencer-Tarjan (1986)
- $O(m\, \alpha(m, n))$ Chazelle (*JACM* 2000)
- Optimal Pettie-Ramachandran (*JACM* 2002)

QUESTION?