Cây

Biên soạn: TS.Nguyễn Viết Đông

Cây

1. ĐN và tính chất
2. Cây khung ngắn nhất
3. Cây có gốc
4. Phép duyệt cây

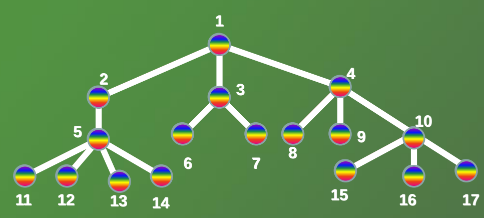
Định nghĩa và tính chất

Định nghĩa Cây.

a) Cho G là đồ thị vô hướng. G được gọi là *một cây* nếu G liên thông và không có chu trình sơ cấp.

b) *Rừng* là đồ thị mà mỗi thành phần liên thông của nó là một cây.

Định nghĩa và tính chất



Định nghĩa và tính chất

Điều kiện cần và đủ.

Cho T là đồ thị vô hướng có n đỉnh. Các phát biểu sau đây là tương đương:

i. T là cây.

ii. T liên thông và có n-1 cạnh.

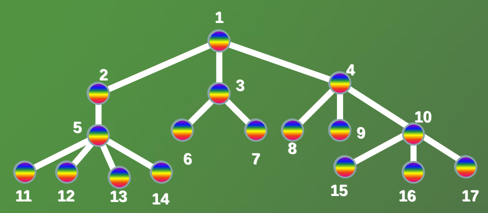
iii. T không có chu trình sơ cấp và có n-1 cạnh .

iv. T liên thông và mỗi cạnh là một cầu.

v. Giữa hai đỉnh bất kỳ có đúng một đường đi sơ cấp nối chúng với nhau.

vi. T không có chu trình sơ cấp và nếu thêm vào một cạnh giữa hai đỉnh không kề nhau thì có một chu trình sơ cấp duy nhất.

Định nghĩa và tính chất



Định nghĩa và tính chất

Định nghĩa cây khung.

Cho G = (V,E) là đồ thị vô hướng.

T là đồ thị con khung của G.

Nếu T là một cây thì T được gọi là *cây khung*(hay *cây tối đại*, hay *cây bao trùm*) của đồ thị G.

Thuật toán tìm cây khung.

Breadth-first Search Algorithm .Thuật toán ưu tiên chiều rộng

Cho *G* là đồ thị liên thông với tập đỉnh $\{v_1, v_2, \ldots, v_n\}$

Bước 0:thêm v_1 như là gốc của cây rỗng.

Bước 1: thêm vào các đỉnh kề v_1 làm con của nó và các cạnh nối v_1 với chúng.

Những đỉnh này là đỉnh mức 1 trong cây.

Bước 2: đối với mọi đỉnh *v* mức1, thêm vào các cạnh kề với *v* vào cây sao cho không tạo nên chu trình đơn.

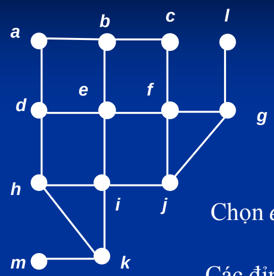
Thu được các đỉnh mức 2.

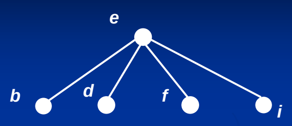
………………………………………………………

Tiếp tục quá trình này cho tới khi tất cả các đỉnh của đồ thị được ghép vào cây.

Cây*T* thu được là cây khung của đồ thị.

Ví dụ. Xét đồ thị liên thông *G*.

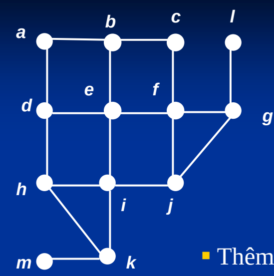


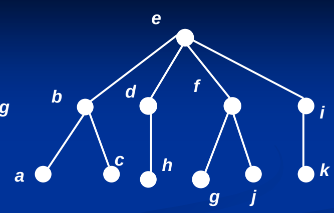


Chọn *e* làm gốc

Các đỉnh kề với *e* là con của nó.

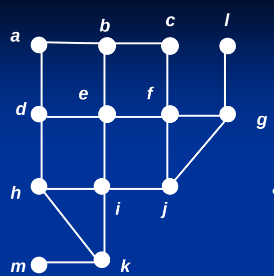
Các đỉnh mức 1 là: *b, d, f, i*

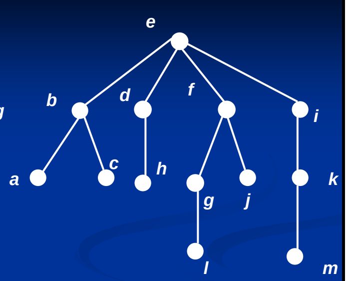




- Thêm *a* và *c* làm con của *b*,
- *h* là con duy nhất của *d*,
- *g* và *j* là con của *f*,
- *k* là con duy nhất của *i*,

Các đỉnh mức 2 là: *a, c, h, g, j, k*





- Cuối cùng thêm *l* và *m* là con của *g* và *k* tương ứng

Các đỉnh mức 3 là: *l, m*

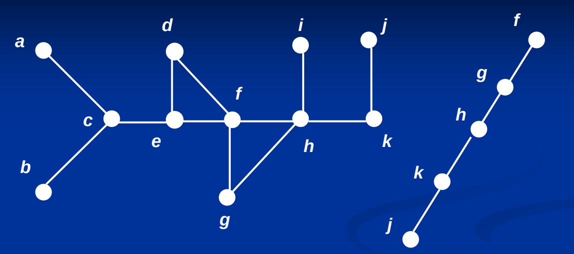
Depth-first Search Algorithm

(Thuật toán ưu tiên chiều sâu)

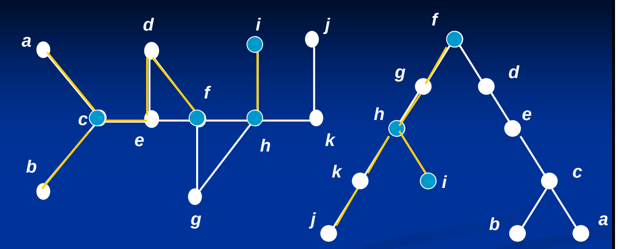
Cho *G* là đồ thị liên thông với tập đỉnh $\{v_1, v_2, \ldots, v_n\}$

Chọn một đỉnh tùy ý của đồ thị làm gốc. Xây dựng đường đi từ đỉnh này bằng cách lượt ghép các cạnh sao cho mỗi cạnh mới ghép sẽ nối đỉnh cuối cùng trên đường đi với một đỉnh còn chưa thuộc đường đi.Tiếp tục ghép thêm cạnh vào đường đi chừng nào không thể thêm được nữa. Nếu đường đi qua tất cả các đỉnh của đồ thị thì cây do đường đi này tạo nên là cây khung. Nếu chưa thì lùi lại đỉnh trước đỉnh cuối cùng của đường đi và xây dựng đường đi mới xuất phát từ đỉnh này đi qua các đỉnh còn chưa thuộc đường đi. Nếu điều đó không thể làm được thì lùi thêm một đỉnh nữa trên đường đi, tức là lùi hai đỉnh trên đường đi và thử xây dựng đường đi mới.

Ví dụ. Tìm cây bao trùm của đồ thị *G*.



Giải. Bắt đầu chọn đỉnh *f* làm gốc và
Thêm các hậu duệ của *f* : *g, h, k, j*
Lùi về *k* không thêm được cạnh nào, tiếp tục lùi về *h*



- Thêm *i* làm con thứ hai của *h* và lùi về *f.*
- Lại thêm các hậu duệ của *f* : *d, e, c, a*
- Lùi về *c* và thêm *b* làm con thứ hai của nó .
- Cây thu được là cây khung của đồ thị đã cho

Định nghĩa và tính chất

Định nghĩa Cây khung ngắn nhất.

Cho G là đồ thị có trọng số. Cây khung T của G được gọi là *cây khung ngắn nhất (cây tối đại ngắn nhất,cây bao trùm ngắn nhất, cây khung tối tiểu)* nếu nó là cây khung của G mà có trọng lượng nhỏ nhất.

Cây khung ngắn nhất

Thuật toán tìm cây khung ngắn nhất

a)Thuật toán Kruscal

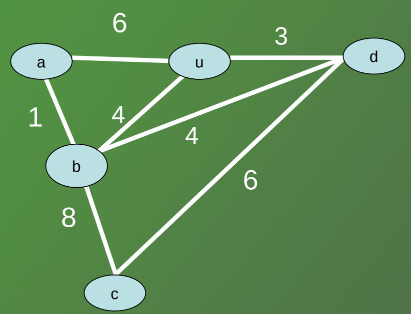
Cho G là đồ thị liên thông, có trọng số, *n* đỉnh.

Bước 1. Trước hết chọn cạnh ngắn nhất e_1 trong các cạnh của G.

Bước 2. Khi đã chọn *k* cạnh $e_1, e_2, \ldots e_k$ thì chọn tiếp cạnh e_{k+1} ngắn nhất trong các cạnh còn lại của G sao cho không tạo thành chu trình với các cạnh đã chọn trước.

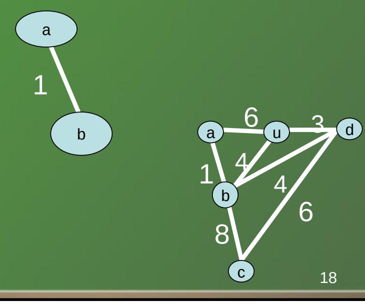
Bước 3. Chọn đủ *n*-1 cạnh thì dừng.

Cây khung ngắn nhất

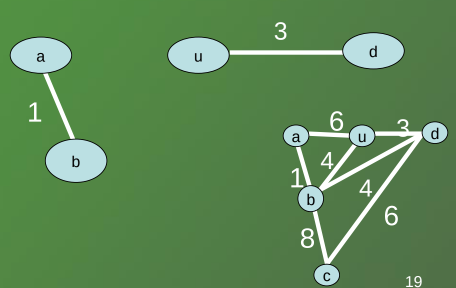


Cây khung ngắn nhất

S_1

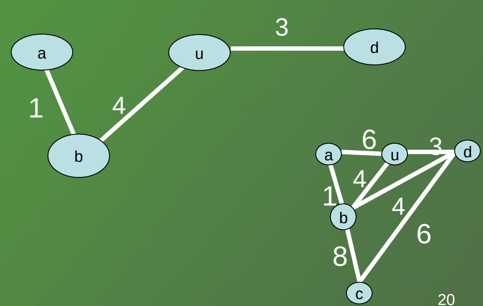


Cây khung ngắn nhất



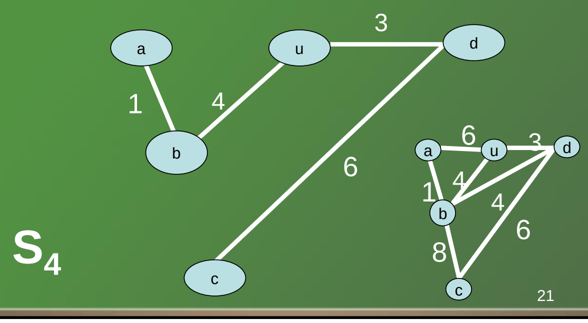
S_2

Cây khung ngắn nhất

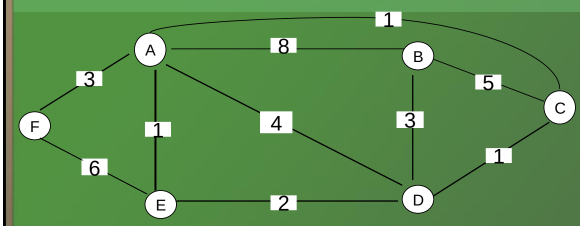


S_3

Cây khung ngắn nhất



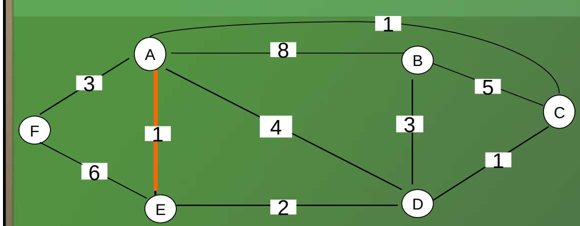
Thuật toán Krusal



AE	DC	AC	ED	BD	AF	AD	BC	FE	AB
1	1	1	2	3	3	4	5	6	8

0. $S_0=\{\}$

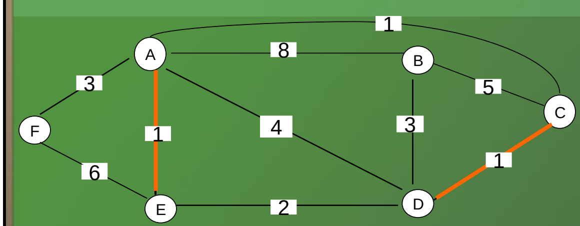
Thuật toán Krusal



AE	DC	AC	ED	BD	AF	AD	BC	FE	AB
1	1	1	2	3	3	4	5	6	8

0. $S_0=\{\}$
1. $S_1=S_0$ + AE = {AE}

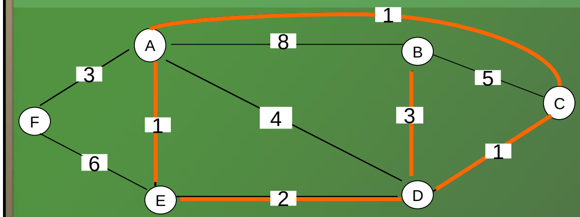
Thuật toán Krusal



AE	DC	AC	ED	BD	AF	AD	BC	FE	AB
1	1	1	2	3	3	4	5	6	8

1. S_1= {AE}
2. $S_2=S_1$ + DC = {AE,DC}

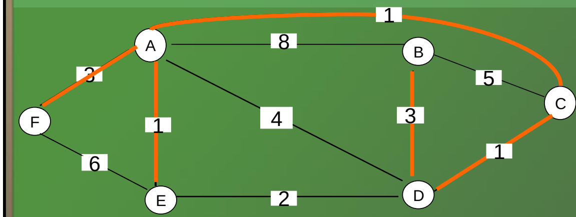
Thuật toán Krusal



AE	DC	AC	ED	BD	AF	AD	BC	FE	AB
1	1	1	2	3	3	4	5	6	8

3. S_3= {AE,DC,AC}
4. S_4=S_3 + BD= {AE,DC,AC,BD}

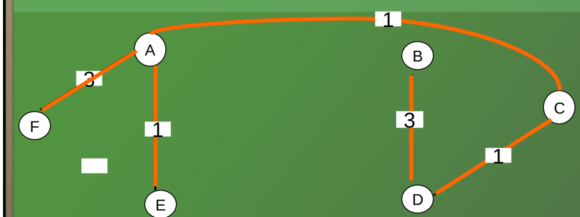
Thuật toán Krusal



AE	DC	AC	ED	BD	AF	AD	BC	FE	AB
1	1	1	2	3	3	4	5	6	8

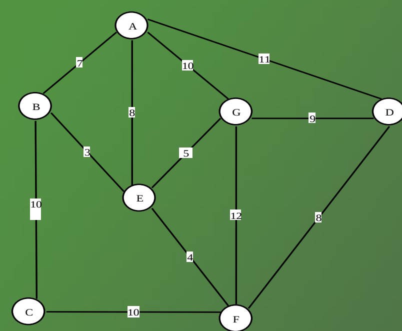
4. S_4= {AE,DC,AC,BD}
5. S_5=S_4 + AF= {AE,DC,AC,BD,AF}

Thuật toán Krusal



Kết luận: S_5= {AE,DC,AC,BD,AF}
là cây bao trùm tối tiểu cần tìm
Trọng lượng: 9

Ví dụ



Cây khung ngắn nhất

Thuật toán tìm cây khung ngắn nhất

b)Thuật toán Prim.

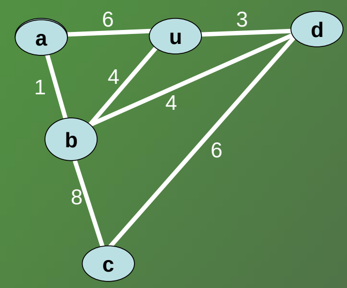
Bước 1. Chọn 1 đỉnh bất kỳ v_1 để có cây T_1 chỉ gồm 1 đỉnh.

Bước 2. Khi đã chọn cây T_k thì chọn tiếp cây $T_{k+1} = T_k \cup e_{k+1}$. Trong đó e_{k+1} là cạnh ngắn nhất trong các cạnh có một đầu mút thuộc T_k và đầu mút kia không thuộc T_k

Bước 3. Chọn được cây T_n thì dừng.

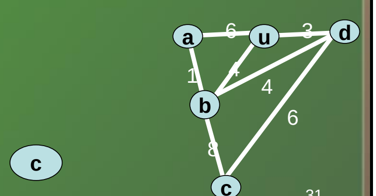
Cây khung ngắn nhất

Thuật toán tìm cây khung ngắn nhất



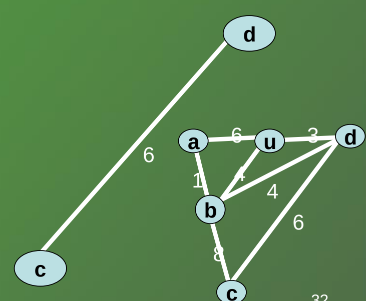
Cây khung ngắn nhất

T_1

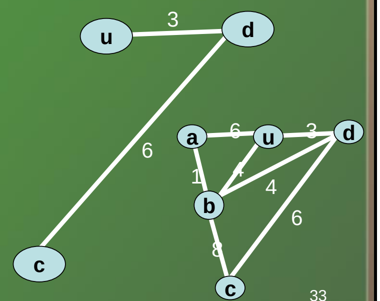


Cây khung ngắn nhất

T_2

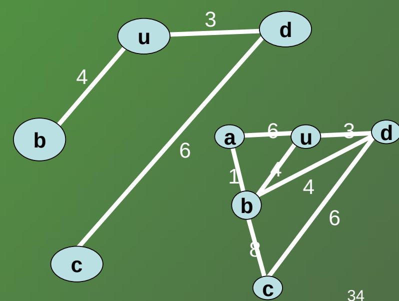


Cây khung ngắn nhất



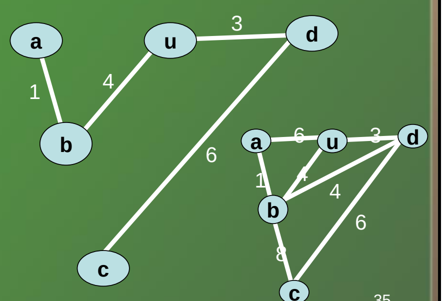
T_3

Cây khung ngắn nhất



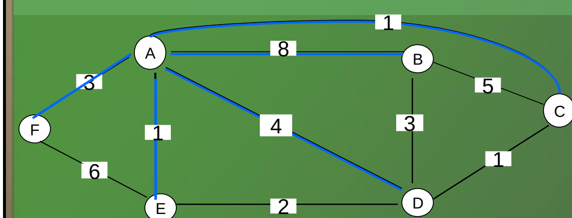
T_4

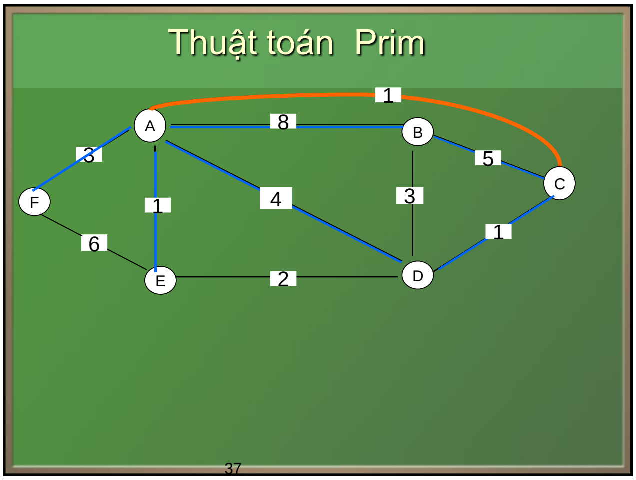
Cây khung ngắn nhất

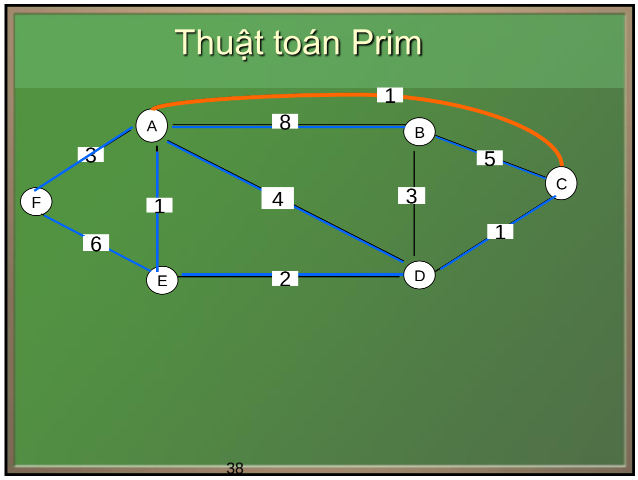


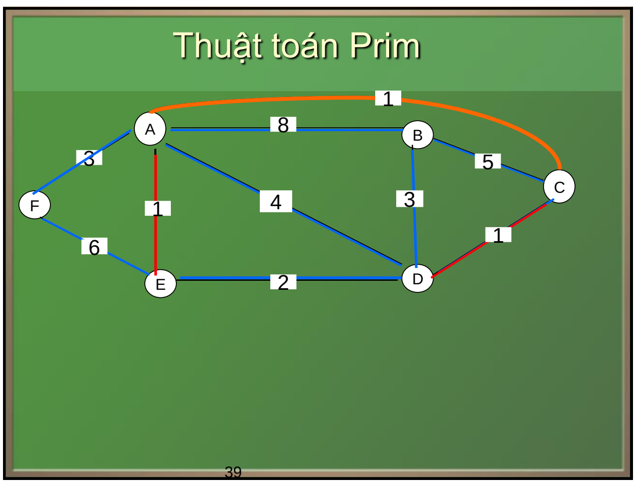
T_5

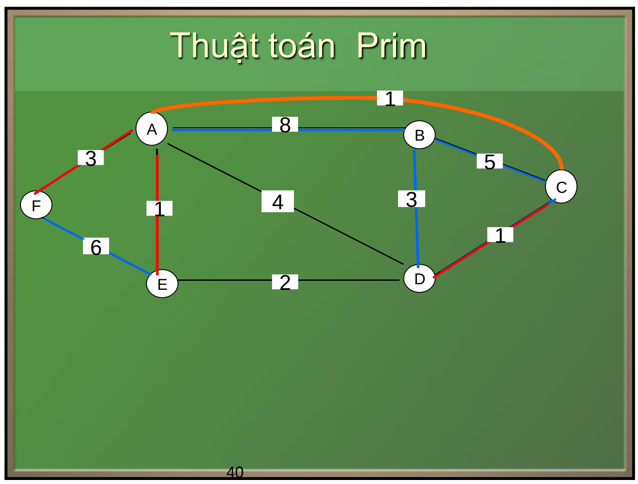
Thuật toán Prim



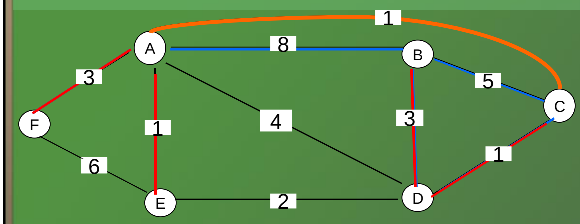

Thuật toán Prim
1
A
8
B
3
5
C
F
1
4
3
1
6
E
2
D
37


Thuật toán Prim
1
A
8
B
3
5
C
F
1
4
3
1
6
E
2
D
38

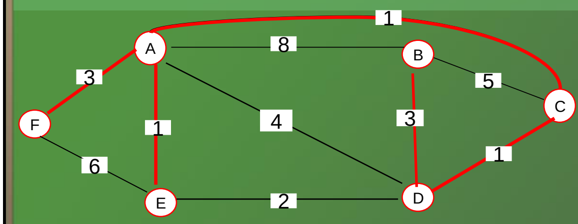

Thuật toán Prim
1
A
8
B
3
5
C
F
1
4
3
1
6
E
2
D
39


Thuật toán Prim
1
A
8
B
3
5
C
F
1
4
3
1
6
E
2
D
40

Thuật toán Prim

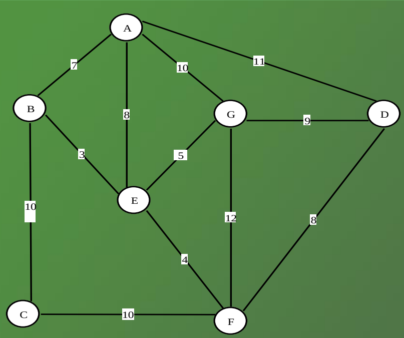


Thuật toán Prim



Cây T_5 = {AC, AE,CD,AF,DB} là cây bao trùm tối tiểu cần tìm với $w(T_5) = 9$

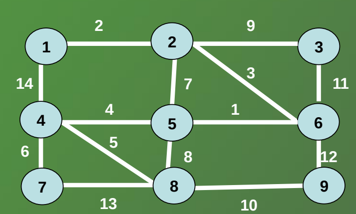
Ví dụ



Cây khung ngắn nhất

Đề thi 2004

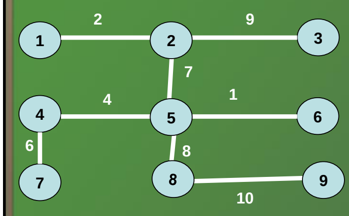
Hãy trình bày thuật toán tìm cây khung ngắn nhất của G chứa cạnh 58 nhưng không chứa cạnh 26

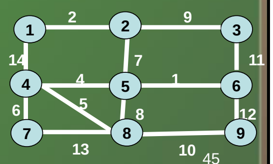


Cây khung ngắn nhất

Giải

- Đặt G’= G -26 thì cây khung phải tìm là ở trong G’. Đầu tiên chọn cạnh 58 sau đó áp dụng Kruscal như thông thường





Cây có gốc

Định nghĩa.

Cho *T* là một cây. Chọn một đỉnh *r* của cây gọi là *gốc* . Vì có đường đi sơ cấp duy nhất từ gốc tới mỗi đỉnh của đồ thị nên ta định hướng mỗi cạnh là hướng từ gốc đi ra . Cây cùng với gốc sinh ra một đồ thị có hướng gọi là *cây có gốc*

Trong một cây có gốc *r* thì $\deg^-(r) = 0$,

$\deg^-(v) = 1$ với mọi đỉnh không phải là gốc.

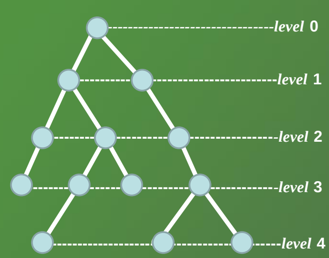
Cây có gốc

Định nghĩa

Cho cây có gốc *r*.

- Gốc *r* được gọi là *đỉnh mức 0* (level 0).
- Các đỉnh kề với gốc *r* được xếp ở phía dưới gốc và gọi là *đỉnh mức* 1(level 1).
- Đỉnh sau của đỉnh mức 1(xếp phía dưới đỉnh mức1)gọi là *đỉnh mức 2.* ……
- Level (*v*) = *k* ⇔ đường đi từ gốc *r* đến *v* qua *k* cung.
- *Độ cao của cây* là mức cao nhất của các đỉnh.

Cây có gốc



Cây có gốc

Định nghĩa

Cho cây có gốc *r*

a) Nếu *uv* là một cung của *T* thì *u* được gọi là *cha của v*, còn *v* gọi là *con của u*.

b) Đỉnh không có con gọi là *lá*(hay *đỉnh ngoài*). Đỉnh không phải là lá gọi là *đỉnh trong*.

c) Hai đỉnh có cùng cha gọi là *anh em*.

Cây có gốc

Định nghĩa

Cho cây có gốc *r*

d) Nếu có đường đi $v_1v_2\dots v_k$ thì $v_1, v_2,.., v_{k-1}$ gọi là *tổ tiên của* v_k. Còn v_k gọi là *hậu duệ* của $v_1, v_2,.., v_{k-1}$.

e) Cây con tại đỉnh *v* là cây có gốc là *v* và tất cả các đỉnh khác là mọi hậu duệ của *v* trong cây *T* đã cho.

Cây có gốc

Định nghĩa

Cho *T* là cây có gốc.

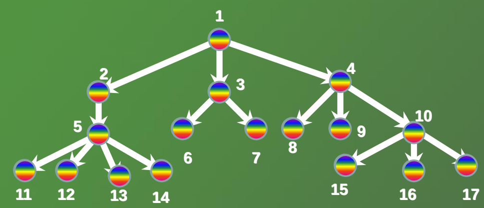
a) T được gọi là *cây k-phân* nếu mỗi đỉnh của *T* có nhiều nhất là *k* con.

b) Cây 2-phân được gọi là *cây nhị phân*.

c) Cây k-phân đủ là cây mà mọi đỉnh trong có đúng *k* con.

d) Cây *k*- phân với độ cao *h* được gọi là *cân đối* nếu các lá đều ở mức *h* hoặc *h* – 1 .

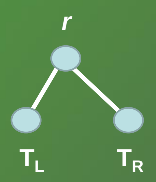
Cây có gốc



Cây có gốc

Định nghĩa

Cho T là cây nhị phân có gốc là r. Ta có thể biểu diễn T như hình vẽ dưới với hai cây con tại r là T_L và T_R ,chúng lần lượt được gọi là *cây con bên trái và cây con bên phải* của T.



Cây có gốc

Định nghĩa

Độ dài đường đi trong và độ dài đường đi ngoài

Cho T là cây nhị phân **đủ**.

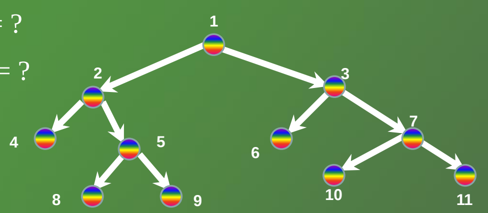
a) Độ dài đường đi trong là tổng tất cả các mức của các đỉnh trong, ký hiệu IP(T).

b) Độ dài đường đi ngoài là tổng tất cả các mức của các lá, ký hiệu EP(T).

Cây có gốc

IP(T) = ?

EP(T) = ?



Cây có hướng

Định lí

Cho T là cây nhị phân đủ với k đỉnh trong và s lá.

Ta có:

$$\boxed{s = k+1 \text{ và } EP=IP+2k}$$

Cây có hướng

Định nghĩa

Cho *T* là cây nhị phân không đủ. Lập *T'* là cây có được bằng cách sau:

i. Thêm vào mỗi lá của *T* hai con.

ii. Thêm vào *v* một con nếu *v* là đỉnh trong của *T* mà chỉ có một con. Ta đặt:

$$\boxed{IP(T) := IP(T') \ \& \ EP(T) := EP(T')}$$

Phép duyệt cây(Tree travesal)

Định nghĩa

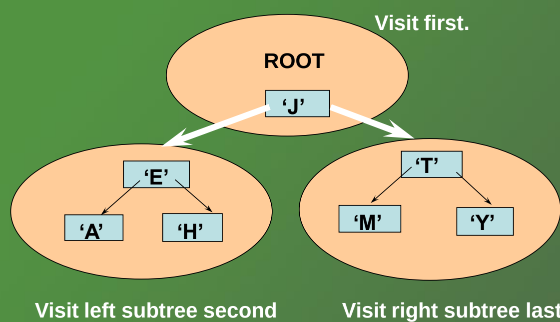
Duyệt cây là liệt kê tất các đỉnh của cây theo một thứ tự nào đó thành một dãy, mỗi đỉnh chỉ xuất hiện một lần .

Phép duyệt cây

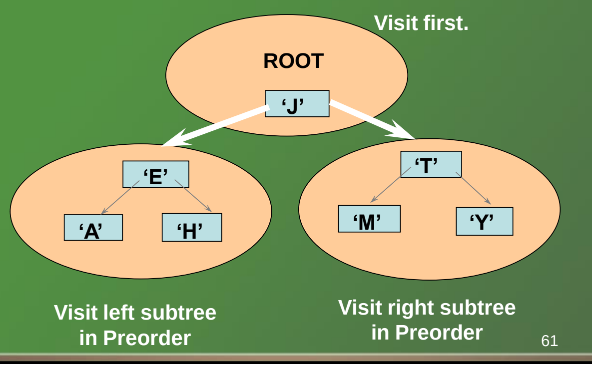
Phép duyệt tiền thứ tự (Preoder traversal)

1. Đến gốc *r*.
2. Dùng phép duyệt tiền thứ tự để duyệt các cây con T_1 rồi cây con T_2 …từ trái sang phải.

Preorder Traversal: J E A H T M Y



Preorder Traversal: J E A H T M Y

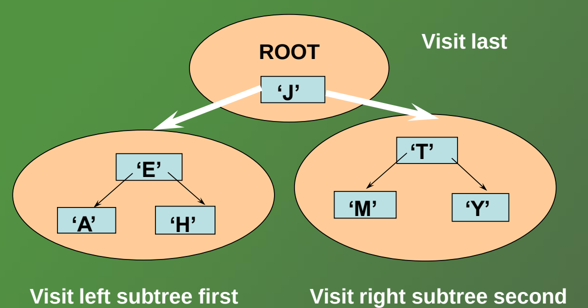


Phép duyệt cây

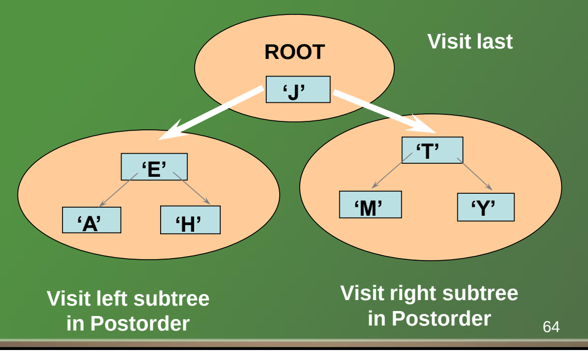
Phép duyệt hậu thứ tự (Posoder traversal).

1. Dùng phép duyệt hậu thứ tự để lần lượt duyệt cây con T_1, T_2,…. từ trái sang phải.
2. Đến gốc *r*.

Postorder Traversal: A H E M Y T J



Postorder Traversal: A H E M Y T J

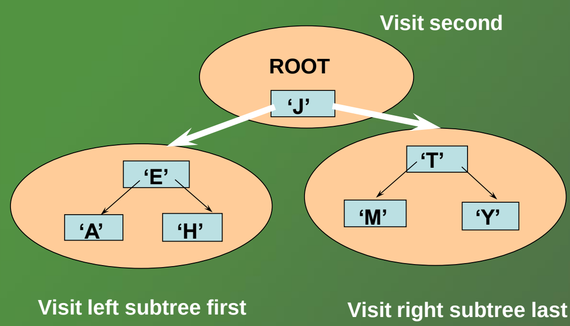


Phép duyệt cây

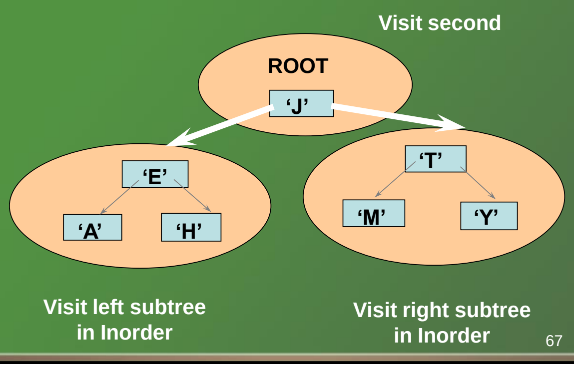
Phép duyệt trung thứ tự cho *cây nhị phân* (Inorder traversal)

1. Duyệt cây con bên trái T_L theo trung thứ tự.
2. Đến gốc *r*.
3. Duyệt cây con bên phải theo trung thứ tự.

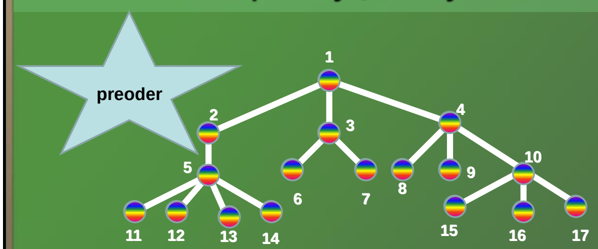
Inorder Traversal: A E H J M T Y



Inorder Traversal: A E H J M T Y

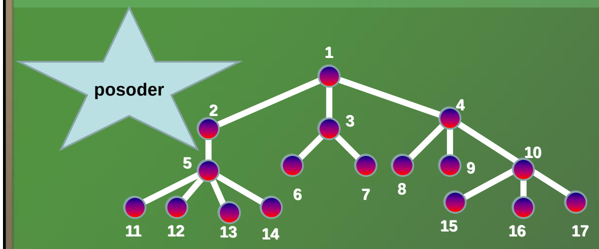


Phép duyệt cây



Preoder:1,2,5,11,12,13,14,3,6,7,4,8,9,10,15,16,17

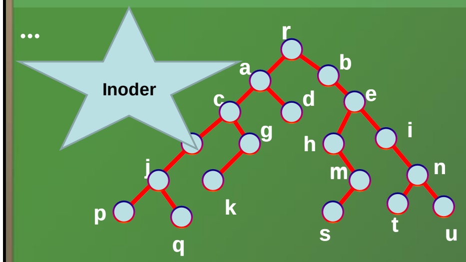
Phép duyệt cây



Posoder:11,12,13,14,5,2,6,7,3,8,9,15,16,17,10,4,1

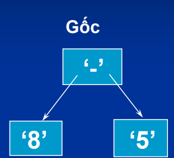
Phép duyệt cây

...



Inoder :p,j,q,f,c,k,g,a,d,r,b,h,s,m,e,i,t,n,u

Cây nhị phân của biểu thức



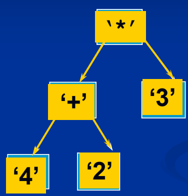
INORDER TRAVERSAL :	8 - 5 có giá trị 3
PREORDER TRAVERSAL:	- 8 5
POSTORDER TRAVERSAL:	8 5 -

Định nghĩa

Cây nhị phân của biểu thức là cây nhị phân mà

1. Mỗi biến số được biểu diễn bởi một lá.
2. Mỗi đỉnh trong biểu diễn một phép toán với các thành tố là cây con tại đỉnh ấy.
3. Cây con bên trái và bên phải của một đỉnh trong biểu diễn cho biểu thức con, giá trị của chúng là thành tố mà ta áp dụng cho phép toán tại gốc của cây con.

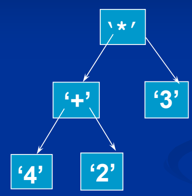
Cây nhị phân của biểu thức



Kết quả?

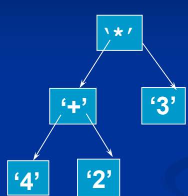
(4 + 2) * 3 = 18

Cây nhị phân của biểu thức



Dạng trung tố, tiền tố, hậu tố?

Cây nhị phân của biểu thức



Infix: ((4 + 2) * 3)
Prefix: * + 4 2 3 Ký pháp Ba lan : *từ phải sang trái*
Postfix: 4 2 + 3 * Ký pháp BL đảo : *từ trái sang phải*

Giải thích

Để có biểu thức theo ký pháp Ba lan, ta duyệt cây nhị phân của biểu thức bằng phép duyệt tiền thứ tự.

Thực hiện biểu thức từ phải sang trái:

Bắt đầu từ bên phải, khi gặp một phép toán thì phép toán này được thực hiện cho 2 thành tố ngay bên phải nó, kết quả này là thành tố cho phép toán tiếp theo.

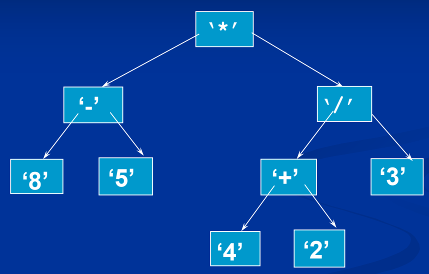
Giải thích

Để có biểu thức theo ký pháp Ba lan ngược, ta duyệt cây nhị phân của biểu thức bằng phép duyệt hậu thứ tự.

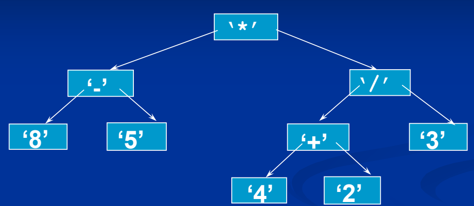
Thực hiện biểu thức từ trái sang phải:

Bắt đầu từ bên trái, khi gặp một phép toán thì phép toán này được thực hiện cho 2 thành tố ngay bên trái nó, kết quả này là thành tố cho phép toán tiếp theo.

Ví dụ



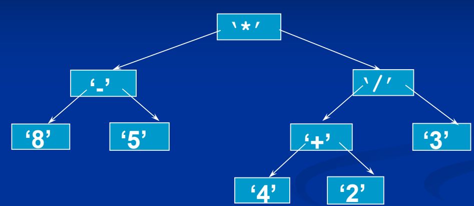
Kết quả của infix, prefix, postfix?



Infix: ((8 - 5) * ((4 + 2) / 3))

Prefix: * - 8 5 / + 4 2 3

Postfix: 8 5 - 4 2 + 3 / *

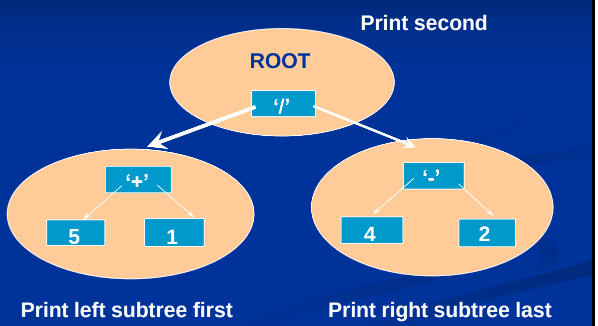


Infix: ((8 - 5) * ((4 + 2) / 3))

Prefix: * - 8 5 / + 4 2 3 ***Thực hiện từ phải sang***

Postfix: 8 5 - 4 2 + 3 / * ***Thực hiện từ trái sang***

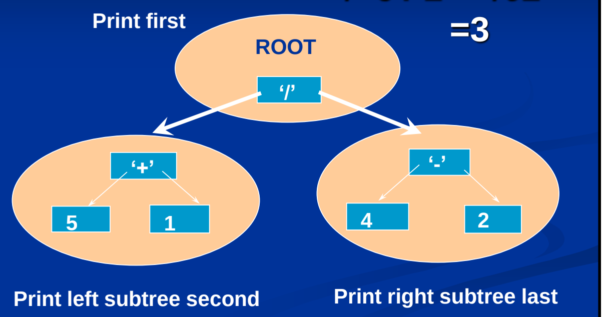
Inorder Traversal: (5 + 1) / (4 - 2) = 3



Preorder Traversal: / + 5 1 – 4 2

=/+51 2 = /62

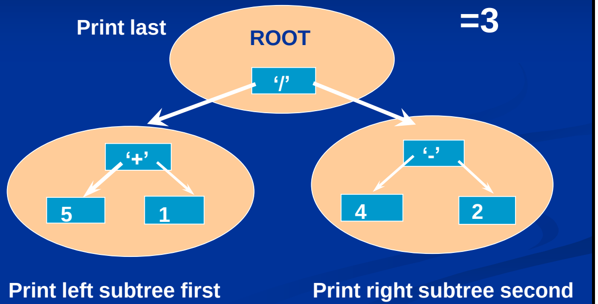
=3



Postorder Traversal: 5 1 + 4 2 – /=

= 6 4 2 –/ =6 2 /

=3



Cây khung có hướng

Định nghĩa

Cho G =(V,E) là đồ thị có hướng và

T = (V,F) là đồ thị con khung của G. Nếu T là cây có hướng thì T gọi là *cây khung có hướng(hay cây có hướng tối đại)* của G.

Cây khung có hướng

Matrận Kirchhoff (G không khuyên)

a) Nếu G là đồ thị có hướng thì K(G) =(k_{ij})

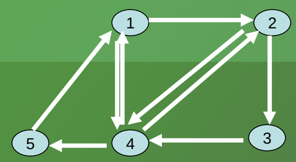
$$k_{ij} = \begin{cases} \deg^{-}(i) & khi\ i = j \\ -B_{ij} & khi\ i \neq j \end{cases}$$

trong đó B_{ij} là số cung đi từ i đến j

b) Nếu G là đồ thị vô hướng thì K(G) =(k_{ij})

$$k_{ij} = \begin{cases} \deg(i) & khi\ i = j \\ -B_{ij} & khi\ i \neq j \end{cases}$$

trong đó B_{ij} là số cung đi từ i đến j



$$\begin{pmatrix} 2 & -1 & 0 & -1 & 0 \\ 0 & 2 & -1 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ -1 & -1 & 0 & 3 & -1 \\ -1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Cây khung có hướng

Định lý

Cho G là đồ thị không khuyên. Đặt K_q(G) là phần phụ của k_{qq}(Ma trận có được từ K(G) bằng cách xoá dòng q và cột q).

Số cây khung có hướng trong G có gốc là đỉnh q bằng detK_q(G).

Đề thi

Đề thi 2003

➢Cho đồ thị có hướng G = (V,E) với V={1,2,3,4,5} xác định bởi ma trận kề sau:

$$\begin{pmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

a) Tìm số liên thông đỉnh của G

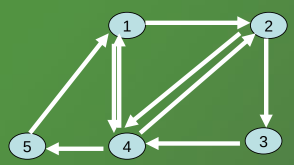
b) G có là đồ thị Euler không? Tại sao?

c) Tìm số cậy có hướng tối đại của G có gốc là đỉnh 1

d) Vẽ các cây trong câu c)

Đề thi

...



89

Đề thi

Giải

a) Với A $\subseteq$ V ký hiệu G-A để chỉ đồ thị có được từ G bằng cách xoá các đỉnh thuộc A và các cung kề với nó.Ta thấy G - A vẫn liên thông nếu A chỉ gồm một đỉnh. G - A không liên thông nếu

A ={1,4}. Vậy v(G) = 2

b) G liên thông và cân bằng nên G là Euler.

90

Đề thi

Giải

c)Ma trận Kirchhoff của G là ma trận sau

$$\begin{pmatrix} 2 & -1 & 0 & -1 & 0 \\ 0 & 2 & -1 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ -1 & -1 & 0 & 3 & -1 \\ -1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

91

Đề thi

...

$$K_1(G) = \begin{pmatrix} 2 & -1 & -1 & 0 \\ 0 & 1 & -1 & 0 \\ -1 & 0 & 3 & -1 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

92

Đề thi

…

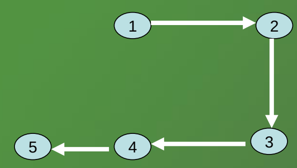
$$\det K_1(G) = \begin{vmatrix} 2 & -1 & -1 \\ 0 & 1 & -1 \\ -1 & 0 & 3 \end{vmatrix} = 4$$

Vậy G có 4 cây có hướng tối đại .

Đó là các cây sau đây

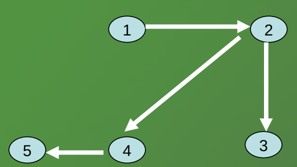
Đề thi

…



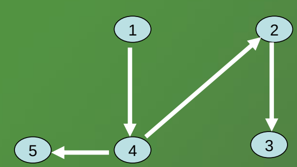
Đề thi

…



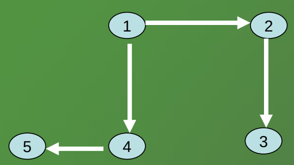
Đề thi

…



Đề thi

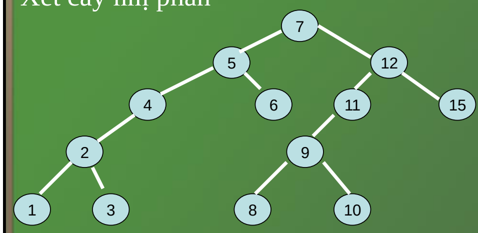
...



Đề thi

Đề thi 2001

Xét cây nhị phân



Đề thi

Đề thi 2001

a) Hãy duyệt cây theo thứ tự giữa (trung thứ tự). Có nhận xét gì về giá trị của các khoá khi duyệt theo thứ tự giữa.

b) Hãy chèn lần lượt các khoá 13,14 vào cây mà vẫn duy trì được nhận xét trên.

Đề thi

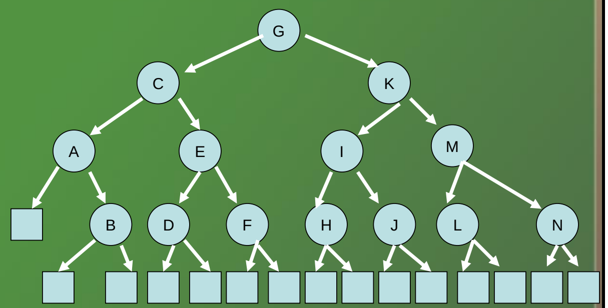
Giải

a) Duyệt theo thứ tự giữa các khoá sẽ có giá trị tăng dần 1,2,3,4,5,6,7,8,9,10,11,12,15.

b) Khoá 13 được chèn thành nút con bên trái của nút 15 và khoá 14 được chèn thành nút con bên phải của nút 13.

Đề thi

Đề thi 2002



101

Đề thi

Đề thi 2002

a) Tìm độ dài đường đi trong và độ dài đường đi ngoài của cây.

b) Cho biết kết quả duyệt cây theo thứ tự sau.

c) Xây dựng cây biểu diễn cho thuật toán tìm kiếm nhị phân trên mảng a sắp thứ tự tăng gồm 14 phần tử. Suy ra số lần so sánh khoá trung bình khi dùng thuật toán tìm kiếm nhị phân để tìm xem một phần tử x có nằm trong mảng a hay không.

102

Đề thi

Giải

a) Độ dài đường đi trong IP=0+2.1+4.2+7.3=31.

Độ dài đường đi ngoài EP=IP+2n=31+2.14=59.

b) Kết quả dyệt cây theo thứ tự sau:

B,A,D,F,E,C,H,J,I,L,N,M,K.

c) Là cây trong đề bài bằng cách thay tương ứng A,B,C,… bởi 1,2,3,…

103

Đề thi

...

Số phép so sánh khoá trung bình

➢Tìm thành công (dừng tại nút trong):

$(IP+n)/n = (31 + 14) /14 \approx 3.21$

➢Tìm không có (dừng tại nút ngoài):

$EP/(n+1) = 59/15 \approx 3,93$

104

Đề thi

Đề thi 2008

Bài 5.Một cạnh e của đồ thị đơn, liên thông G được gọi là cầu nếu G không còn liên thông khi ta xóa e. Chứng minh rằng e là cầu nếu và chỉ nếu mọi cây tối đại của G đều chứa e.

Đề thi

Giải:- Giả sử e là cầu.Khi đó G – e không liên thông.Giả sử T là một cây không chứa e.Do T liên thông nó sẽ nằm trong một thành phần liên thông của G – e , vì vậy T không phải là cây tối đại của G.

- Đảo lại:Giả sử e nằm trong mọi cây tối đại. Nếu G – e liên thông thì nó sẽ chứa một cây tối đại T. Rõ ràng T cũng là một cây tối đại của G, mà T không chứa e, mâu thuẫn.Vậy G – e không liên thông, do đó e là cầu.

Đề thi

- Đề 2008.

Bài 6.

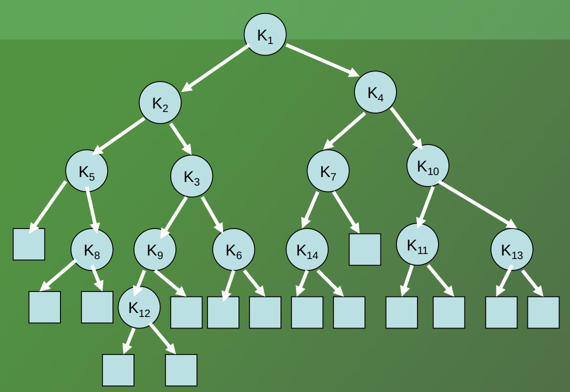
a) Vẽ cây nhị phân có được bằng cách chèn lần lượt các khóa $K_1,K_2,\ldots,K_{14}$ sao cho khóa ở mỗi nút lớn hơn khóa của các nút thuộc cây con bên trái và bé hơn khóa của các các nút thuộc cây con bên phải.Thứ tự của các khóa như sau:

Đề thi

$K_5 < K_8 < K_2 < K_{12} < K_9 < K_3 < K_6 < K_1 < K_{14} < K_7 < K_4 < K_{11} < K_{10} < K_{13}$

b) Nếu tìm ngẫu nhiên một khóa K đã có trong cây thì số phép so sánh trung bình là bao nhiêu? Ta giả thiết rằng xác suất để K bằng một trong các khóa trong cây là như nhau.

$K_5 < K_8 < K_2 < K_{12} < K_9 < K_3 < K_6 < K_1 < K_{14} < K_7 < K_4 < K_{11} < K_{10} < K_{13}$



109

Đề thi

- Độ dài đường đi trong :

I = 0+2.1+4.2 + 6.3+ 4 = 2 +8+18+4 = 32

Số phép so sánh trung bình cho tìm kiếm thành công:

(I + n)/n = 46/14 = 3,29

110

Đề thi

Đề thi ĐHBK 2000.

a) Xây dựng cây biểu diễn cho thuật toán tìm kiếm nhị phân trên mảng sắp thứ tự tăng gồm 13 phần tử.

b) Tìm độ dài đường đi trong và độ dài đường đi ngoài của cây.

c) Cho biết kết quả duyệt cây theo thứ tự trước.

111

Appendix

Thuật toán tìm kiếm nhị phân(binary search):

❑Tìm phần tử x trong dãy số tăng dần.

➢Nhập: dãy $a_1, a_2, \ldots, a_n$ tăng dần và phần tử x.

➢Xuất :vị trí của x trong dãy hoặc 0.

112

Appendix

Thuật toán

l:=1,r:= n

repeat

  i:=(l+r)div2;

  if a_i<x then l:=i+1;

  if a_i>x then r : = i-1:

utill(x = a_i or (l >r);

if(x =a_i)then

  xuất i (tìm thấy x ở vị trí i)

else

  xuất 0(không tìm thấy x trong dãy)

113