

VNU-HUS MAT3500: Toán rời rạc

Lý thuyết đồ thị III

Cây

Hoàng Anh Đức

Bộ môn Tin học, Khoa Toán-Cơ-Tin học
Đại học KHTN, ĐHQG Hà Nội
hoanganhduc@hus.edu.vn



Nội dung



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Giới thiệu

Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

2 Giới thiệu

Một số tính chất của cây

Cây có gốc

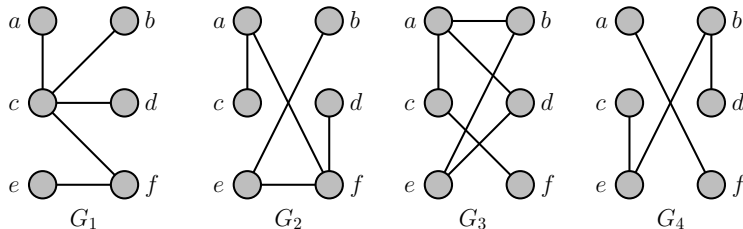
Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

- Đơn đồ thị vô hướng $T = (V, E)$ được gọi là một **cây (tree)** nếu T là đồ thị liên thông và không có chu trình
- Đơn đồ thị vô hướng $T = (V, E)$ được gọi là một **rừng (forest)** nếu T là đồ thị không có chu trình



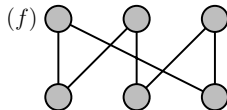
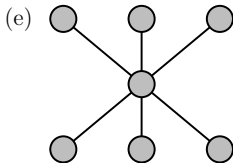
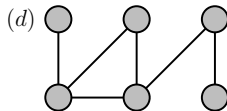
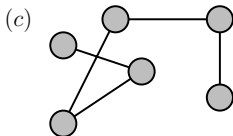
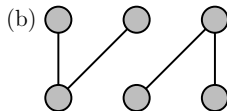
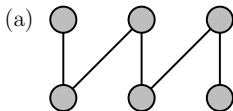
Hình: G_1, G_2 là cây. G_3, G_4 không là cây. G_4 là rừng. G_3 không là rừng

Bài tập 1

Các đồ thị hai phần đầy đủ $K_{m,n}$ ($m, n \in \mathbb{Z}^+$) nào là cây?

Bài tập 2

Đồ thị nào trong các đồ thị sau là cây?



3 Giới thiệu

Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Định lý 1

Một đồ thị vô hướng G là một cây khi và chỉ khi giữa hai đỉnh bất kỳ của G tồn tại một đường đi đơn duy nhất

Chứng minh.

(\Rightarrow) Giả sử G là cây

- Do G liên thông, với hai đỉnh bất kỳ $u, v \in V$, **tồn tại** một đường đi đơn giữa u và v
- Ta chứng minh đường đi này là **duy nhất**. Giả sử phản chứng rằng có ít nhất hai đường đi đơn $P = v_0, v_1, \dots, v_k$ và $Q = w_0, w_1, \dots, w_\ell$ khác nhau giữa hai đỉnh u và v , với $v_0 = w_0 = u$ và $v_k = w_\ell = v$
 - Do $v_0 = w_0$, tồn tại i thỏa mãn $v_j = w_j$ với mọi $0 \leq j \leq i$ và $v_{i+1} \neq w_{i+1}$
 - Do $v_k = w_\ell$, tồn tại $p \geq i + 1$ nhỏ nhất thỏa mãn $v_p \in \{w_i, \dots, w_\ell\}$ hoặc $w_p \in \{v_i, \dots, v_k\}$. Giả sử $v_p = w_q$ với $q \geq i$ nhỏ nhất có thể
 - v_i, v_{i+1}, \dots, v_p và w_q, \dots, w_{i+1}, w_i tạo thành một chu trình trong G , mâu thuẫn với giả thiết G là cây

Cây

Một số tính chất của cây



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

5 Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

Chứng minh (tiếp).

(\Leftarrow) Giả sử với mọi cặp đỉnh $u, v \in V$, tồn tại một đường đi đơn duy nhất giữa u và v trong G

- Theo định nghĩa, G là **liên thông**
- Ta chứng minh G **không có chu trình**. Giả sử phản chứng rằng tồn tại một chu trình C trong G . Do đó, với hai đỉnh u, v bất kỳ thuộc C , có hai đường đi đơn khác nhau nối u và v , mâu thuẫn với giả thiết tồn tại duy nhất một đường đi đơn giữa hai đỉnh bất kỳ trong G



Cây

Một số tính chất của cây



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

6 Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

Định lý 2

Mọi cây $T = (V, E)$ với $|V| \geq 2$ có ít nhất hai đỉnh bậc 1

Chứng minh.

Lấy một đường đi đơn **dài nhất** v_0, v_1, \dots, v_m trong T . Bậc của v_0 và v_m đều bằng 1, nếu không ta có thể tìm được một đường đi dài hơn \square

Định lý 3

Mọi cây gồm n đỉnh có chính xác $n - 1$ cạnh

Chứng minh.

Quy nạp theo n

- **Bước cơ sở:** Với $n = 1$, cây gồm 1 đỉnh có 0 cạnh
- **Bước quy nạp:** Giả sử mọi cây gồm k đỉnh có chính xác $k - 1$ cạnh, với $k \geq 1$. Ta chứng minh mọi cây T gồm $k + 1$ đỉnh có chính xác k cạnh. Thật vậy, do T là một cây, T có một đỉnh u bậc 1 nào đó. $T - u$ là một cây gồm k đỉnh và theo giả thiết quy nạp có $k - 1$ cạnh. Do đó, T có $(k - 1) + 1 = k$ cạnh \square

Cây

Một số tính chất của cây



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

7 Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

Bài tập 3

Chứng minh rằng một đồ thị $G = (V, E)$ là một cây khi và chỉ khi G không có chu trình và với mọi cặp đỉnh $u, v \in V$ thỏa mãn $uv \notin E$, đồ thị $G + uv$ có chính xác một chu trình

Bài tập 4

Giả sử một cây $T = (V, E)$ có số các cạnh là một số chẵn. Chứng minh rằng tồn tại một đỉnh trong T có bậc chẵn

Bài tập 5

Cho $T = (V, E)$ là một cây. Gọi Δ là bậc lớn nhất của một đỉnh trong T , nghĩa là, $\Delta = \max_{v \in V} \deg_T(v)$. Chứng minh rằng T có ít nhất Δ đỉnh bậc 1

Cây

Một số tính chất của cây



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

8 Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

Bài tập 6

Cho $T = (V, E)$ là một cây với $|V| > 1$. Giả sử nếu v là một đỉnh liền kề với một đỉnh bậc 1 trong T thì $\deg_T(v) \geq 3$. Chứng minh rằng tồn tại hai đỉnh bậc 1 trong T có chung một đỉnh liền kề với hai đỉnh đó

Bài tập 7

Một rừng bao gồm n đỉnh và t cây (mỗi cây là một thành phần liên thông của rừng) có bao nhiêu cạnh?

Cây

Cây có gốc



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

9 Cây có gốc

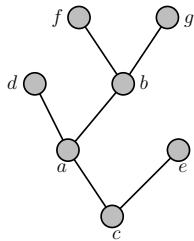
Duyệt cây có gốc

Cây bao trùm

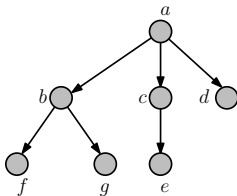
Cây bao trùm nhỏ nhất

Tài liệu tham khảo

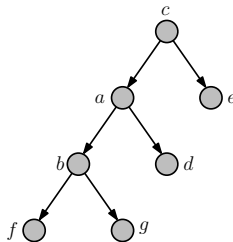
Một **cây có gốc (rooted tree)** là một cây trong đó một đỉnh được coi là **đỉnh gốc (root)** và mọi cạnh được định hướng xa khỏi đỉnh gốc. Ta ký hiệu một cây T với gốc r bằng cặp (T, r)



Cây T



Cây T với gốc a



Cây T với gốc c

Thông thường, một cây có gốc (T, r) được vẽ với **đỉnh gốc ở trên đỉnh của đồ thị**. Khi đó, ta có thể **bỏ qua các hướng của các cạnh**, do việc lựa chọn đỉnh gốc đã xác định các hướng này



- Với một đỉnh v khác đỉnh gốc của một cây có gốc T
 - **Đỉnh cha (parent)** của v là đỉnh u duy nhất sao cho có một cạnh có hướng từ u đến v . Ta cũng gọi v là **đỉnh con (child)** của u .
 - Đỉnh v được gọi là **đỉnh lá (leaf)** nếu v không có đỉnh con. Các đỉnh có đỉnh con được gọi là **các đỉnh trong (internal vertices)**
 - Các đỉnh có chung một đỉnh cha được gọi là **các đỉnh anh em (siblings)**
 - **Các đỉnh tổ tiên (ancestors)** của v là các đỉnh trên đường đi từ gốc tới đỉnh đó. **Các đỉnh con cháu (descendants)** của v là các đỉnh có v là một đỉnh tổ tiên
 - **Cây con (subtree) với gốc v** là đồ thị con của T cảm sinh bởi v và các đỉnh con cháu của nó
- Đỉnh gốc là đỉnh duy nhất trong cây **không có đỉnh cha** và là **tổ tiên của mọi đỉnh còn lại**. Đỉnh gốc luôn là một đỉnh trong, trừ trường hợp cây chỉ có một đỉnh duy nhất là đỉnh gốc thì đỉnh gốc là một đỉnh lá

Cây

Cây có gốc



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

11 Cây có gốc

Duyệt cây có gốc

Cây bao trùm

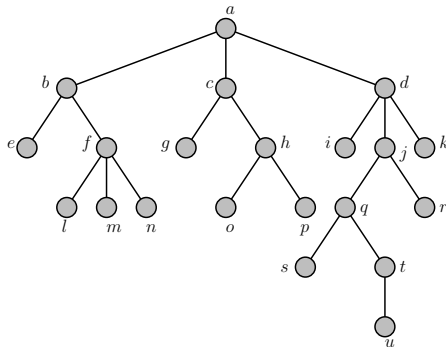
Cây bao trùm nhỏ nhất

Tài liệu tham khảo

- **Mức (level)** của một đỉnh trong một cây có gốc là độ dài của đường đi duy nhất từ gốc đến đỉnh đó. Mức của đỉnh gốc là 0
- **Độ cao (height)** h của một cây có gốc là độ dài lớn nhất của một đường đi từ gốc đến đỉnh lá. Nói cách khác, độ cao của một cây là mức lớn nhất của một đỉnh trong cây đó

Bài tập 8

Hãy trả lời các câu hỏi về cây có gốc được minh họa như hình vẽ dưới đây



(a) Đỉnh nào là đỉnh gốc?

(b) Các đỉnh nào là đỉnh trong?

(c) Các đỉnh nào là đỉnh lá?

(d) Các đỉnh nào là đỉnh con của j ?

(e) Đỉnh nào là đỉnh cha của h ?

(f) Các đỉnh nào là đỉnh anh em của o ?

(g) Các đỉnh nào là đỉnh tổ tiên của m ?

(h) Các đỉnh nào là đỉnh con cháu của b ?

(i) Mức của mỗi đỉnh là bao nhiêu?

(j) Vẽ các cây con gốc c và gốc e

Cây

Cây có gốc



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

13 Cây có gốc

Duyệt cây có gốc

Cây bao trùm

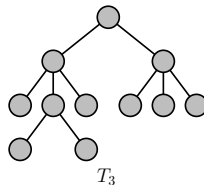
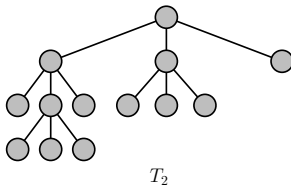
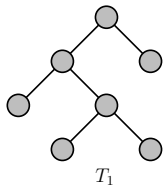
Cây bao trùm nhỏ nhất

Tài liệu tham khảo

- Một cây có gốc T được gọi là một **cây m -phân (m -ary tree)** ($m \geq 1$) nếu tất cả các đỉnh trong của T có tối đa là m đỉnh con. Nếu như mỗi đỉnh trong của T có chính xác m đỉnh con, ta gọi T là một **cây m -phân đầy đủ ($\text{full } m$ -ary tree)**
- Cây m -phân với $m = 2$ được gọi là **cây nhị phân (binary tree)**
- Một cây m -phân có độ cao h được gọi là **cân đối (balanced)** nếu tất cả các đỉnh lá có mức $h - 1$ hoặc h

Bài tập 9

Trong số các cây sau, với m là số nguyên dương nào đó, cây nào là cây m -phân? Cây nào là cây m -phân đầy đủ?



Cây

Cây có gốc



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

14 Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

- Một *cây có gốc được sắp thứ tự (ordered rooted tree)* là một cây có gốc trong đó các đỉnh con của mỗi đỉnh được sắp xếp theo một thứ tự nào đó
- Ta thường vẽ cây có gốc được sắp thứ tự theo cách thông thường với hiểu ngầm rằng *các đỉnh con của mỗi đỉnh được sắp xếp theo thứ tự từ trái sang phải*
- Với một cây nhị phân được sắp thứ tự, nếu một đỉnh trong u có hai đỉnh con thì đỉnh thứ nhất được gọi là *đỉnh con trái (left child)*, đỉnh thứ hai được gọi là *đỉnh con phải (right child)*. Tương tự, cây con có gốc là đỉnh con trái được gọi là *cây con trái (left subtree)* và cây con có gốc là đỉnh con phải được gọi là *cây con phải (right subtree)* của u
- Các cây có gốc sắp thứ tự được ứng dụng rộng rãi trong khoa học máy tính, ví dụ như các cây phân tích cú pháp (parse trees), cấu trúc cây trong các tài liệu XML (XML documents), cấu trúc cây mô tả các thư mục và tệp (directories and file system), v.v...

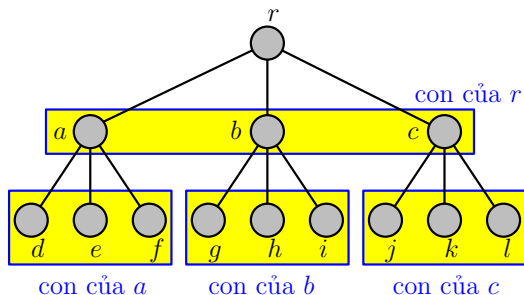


Định lý 4

Mọi cây m -phân đầy đủ ($m \geq 1$) với i đỉnh trong có chính xác $n = m \cdot i + 1$ đỉnh

Chứng minh.

Mọi đỉnh khác đỉnh gốc là con của một đỉnh trong nào đó. Do đó có $m \cdot i$ đỉnh con (của các đỉnh trong) và 1 đỉnh gốc





Định lý 5

Một cây m -phân đầy đủ với

- (i) n đỉnh có $i = (n - 1)/m$ đỉnh trong và $\ell = ((m - 1)n + 1)/m$ đỉnh lá;
- (ii) i đỉnh trong có $n = mi + 1$ đỉnh và $\ell = (m - 1)i + 1$ đỉnh lá;
- (iii) ℓ đỉnh lá có $n = (m\ell - 1)/(m - 1)$ đỉnh và $i = (\ell - 1)/(m - 1)$ đỉnh trong

Bài tập 10

Sử dụng Định lý 4, hãy chứng minh Định lý 5

Bài tập 11

Chứng minh rằng với mọi $m \geq 1$, có nhiều nhất m^h đỉnh lá trong một cây m -phân có độ cao h . (**Gợi ý:** Quy nạp theo h)



Một ứng dụng của cây nhị phân là cấu trúc dữ liệu **cây tìm kiếm nhị phân (binary search tree)** sử dụng để **xây dựng các thuật toán tìm kiếm hiệu quả** khi **các phần tử trong tập hợp cần tìm kiếm đã được sắp xếp theo thứ tự nào đó**

Cây tìm kiếm nhị phân

- Là cây nhị phân: mỗi đỉnh có **tối đa 2 đỉnh con** (đỉnh con trái và đỉnh con phải)
- Mỗi đỉnh được gán nhãn bằng một **khóa (key)** tương ứng với phần tử trong tập hợp cần tìm kiếm
- Khóa ứng với một đỉnh **lớn hơn tất cả các khóa của đỉnh trong các cây con trái** và **nhỏ hơn tất cả các khóa của đỉnh trong các cây con phải**

Cây

Cây có gốc



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

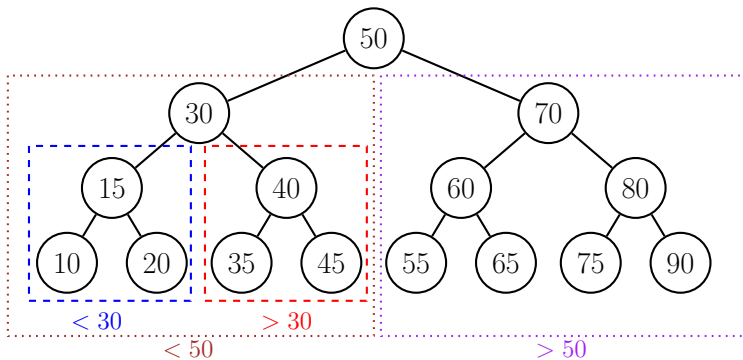
18 Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Hình: Ví dụ một cây tìm kiếm nhị phân



Thuật toán xây dựng cây tìm kiếm nhị phân

Đầu vào: Danh sách các phần tử $L = [k_1, k_2, \dots, k_n]$

Đầu ra: Cây tìm kiếm nhị phân T

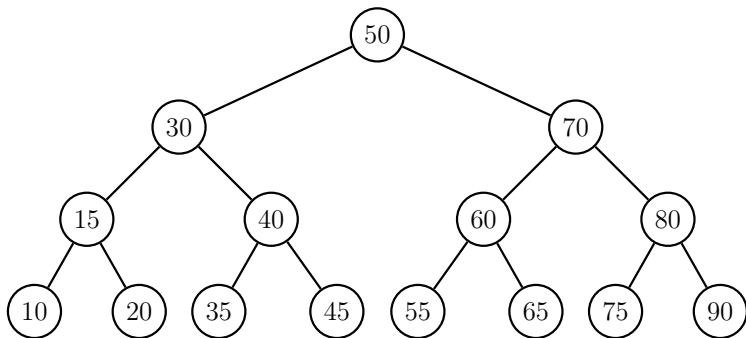
1. **Thủ tục $\text{CreateBST}(L)$:** (Tạo cây tìm kiếm nhị phân từ danh sách L)

- Khởi tạo T là cây rỗng với một đỉnh gốc r
- Gán $\text{key}(r) = k_1$
- Với mỗi i từ 2 đến n : Gọi thủ tục $\text{Insert}(T, r, k_i)$
- Trả về T

2. **Thủ tục $\text{Insert}(T, v, k)$:** (Chèn phần tử (có khóa bằng) k vào cây T với gốc v)

- Nếu $k < \text{key}(v)$:
 - Nếu v có con trái: Gọi đệ quy $\text{Insert}(T, \text{con trái của } v, k)$
 - Ngược lại: Tạo đỉnh mới w với $\text{key}(w) = k$ và thêm w làm con trái của v
- Nếu $k > \text{key}(v)$:
 - Nếu v có con phải: Gọi đệ quy $\text{Insert}(T, \text{con phải của } v, k)$
 - Ngược lại: Tạo đỉnh mới w với $\text{key}(w) = k$ và thêm w làm con phải của v

Ví dụ 1



Hình: Minh họa các bước xây dựng cây tìm kiếm nhị phân với danh sách các phần tử $L = [50, 30, 70, 15, 40, 60, 80, 10, 20, 35, 45, 55, 65, 75, 90]$

Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

20 Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Bài tập 12

Xây dựng cây tìm kiếm nhị phân với danh sách các phần tử:

(a) $L = [10, 15, 20, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 90]$

(b) $L = [50, 35, 75, 10, 60, 45, 20, 80, 55, 30, 90, 40, 70, 15, 65]$

Bài tập 13

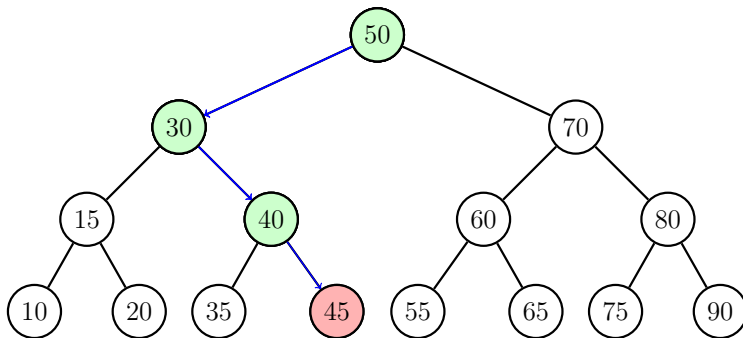
Sử dụng *thứ tự bảng chữ cái (alphabetical order)*¹, hãy xây dựng một cây tìm kiếm nhị phân cho các từ trong câu “The quick brown fox jumps over the lazy dog”

Bài tập 14

Từ thủ tục $\text{Insert}(T, v, k)$, hãy viết một thuật toán đệ quy tìm kiếm một phần tử có khóa bằng k trong cây tìm kiếm nhị phân T với gốc v và nếu không tìm được thì chèn một phần tử mới có khóa bằng k vào cây T và trả lại vị trí của phần tử mới này. (**Gợi ý:** Một phiên bản không đệ quy của thuật toán này là [Rosen 2012], Thuật toán 1, Trang 795)

¹Thứ tự bảng chữ (alphabetical order) cái chính là thứ tự từ điển (lexicographic order) được áp dụng với các chữ cái

Ví dụ 2



$45 > 30$: đi sang cây con phải

Hình: Tìm kiếm phần tử có giá trị 45 trong cây tìm kiếm nhị phân

Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

22 Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Một ứng dụng khác của cây nhị phân được thể hiện trong định lý sau

Định lý 6

Mọi thuật toán sắp xếp n phần tử nào đó dựa trên các phép so sánh hai phần tử bất kỳ cần sử dụng ít nhất $\lceil \log(n!) \rceil$ phép so sánh trong trường hợp xấu nhất

- Do $\log(n!) = \Theta(n \log n)$, *một thuật toán sắp xếp n phần tử trong trường hợp xấu nhất cần $\Theta(n \log n)$ phép so sánh là một thuật toán tối ưu*, theo nghĩa là *không tồn tại thuật toán sắp xếp nào khác có thời gian chạy trong thời gian xấu nhất tốt hơn*
- **Ý tưởng:** Mô hình các thuật toán bằng cây nhị phân đầy đủ với *các đỉnh trong là các phép so sánh* và *các đỉnh lá là các kết quả thu được* khi đi từ gốc (gọi là *cây quyết định (decision tree)*). *Số phép so sánh* sử dụng để ra một kết quả chính là *độ cao của cây quyết định*

Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

23 Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

Cây

Cây có gốc



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

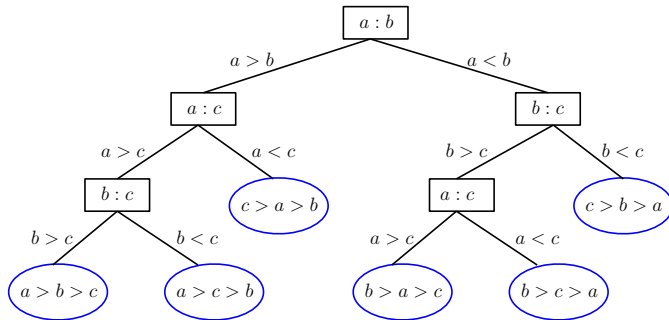
24 Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Hình: Mô hình thuật toán sắp xếp dãy ba phần tử a, b, c bằng cây quyết định

Bài tập 15

Với mọi $m \geq 1$, chứng minh rằng trong một cây m -phân có ℓ đỉnh lá và có độ cao h , ta có $h \geq \lceil \log_m \ell \rceil$



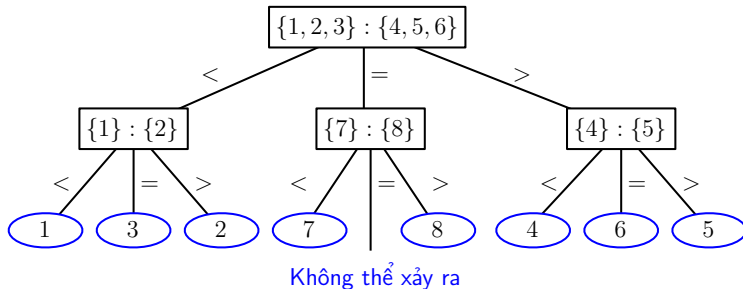
Ví dụ 3 (Ứng dụng khác của cây quyết định)

Ta sử dụng cây quyết định để giải câu đố tìm đồng xu giả quen thuộc sau: Giả sử có bảy đồng xu, tất cả có cùng trọng lượng, và một đồng xu giả có trọng lượng nhẹ hơn các đồng xu khác. Cần bao nhiêu lần cân bằng cách sử dụng một cân đĩa để xác định đồng xu nào là đồng xu giả? Hãy đưa ra thuật toán để tìm đồng xu giả này

- Một cân đĩa cho ra ba kết quả: bên trái nhẹ hơn, bên phải nhẹ hơn hoặc hai bên bằng nhau \Rightarrow Cây quyết định là cây 3-phân
- Có tất cả 8 đồng xu \Rightarrow Cây quyết định có ít nhất 8 đỉnh lá (mỗi đồng xu đều có thể là giả)
- Cây quyết định có độ cao $h \geq \lceil \log_3(8) \rceil = 2$ (Bài tập 15) \Rightarrow Cần tối thiểu 2 lần cân để xác định đồng xu giả



Ví dụ 3 (tiếp)



Hình: Cây quyết định tìm đồng xu giả trong số 8 đồng xu đánh số $1, 2, \dots, 8$. Chỉ cần 2 lần cân là đủ để xác định đồng xu giả

Bài tập 16

Cần ít nhất bao nhiêu lần cân bằng cân đĩa để tìm một đồng xu giả trong số bốn đồng xu nếu đồng xu giả có thể nặng hơn hoặc nhẹ hơn các đồng xu khác? Mô tả một thuật toán để tìm đồng xu giả bằng số lần cân này

Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

26 Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

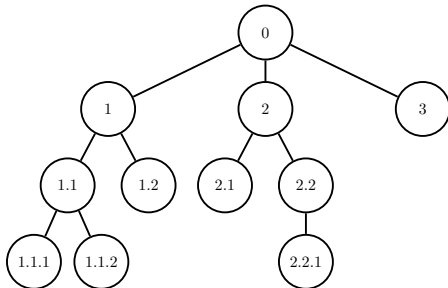


- Cây có gốc được sắp thứ tự thường được sử dụng để **lưu trữ thông tin**. Để truy cập dữ liệu, cần có các thuật toán để thăm (duyet) mỗi đỉnh của cây có gốc được sắp thứ tự
- Một số **thuật toán quan trọng** để **duyet tất cả các đỉnh của một cây có gốc được sắp thứ tự**
 - Duyệt tiền thứ tự (Preorder traversal)
 - Duyệt trung thứ tự (Inorder traversal)
 - Duyệt hậu thứ tự (Postorder traversal)
- Cây có gốc được sắp thứ tự cũng có thể được sử dụng để **biểu diễn các loại biểu thức khác nhau**, chẳng hạn như các biểu thức số học bao gồm số, biến và các phép toán. Thứ tự duyệt cây có gốc được sắp thứ tự cũng có thể được sử dụng trong việc **đánh giá (evaluation) các biểu thức biểu diễn bởi cây đó**



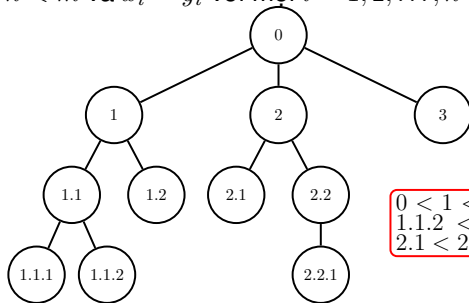
Hệ thống địa chỉ phổ quát (universal address system)

- Gốc có nhãn 0
- Các đỉnh con của gốc được gán nhãn $1, 2, \dots, k$ từ trái sang phải
- Với mỗi đỉnh v ở mức n với nhãn A , gán nhãn cho k_v đỉnh con của nó theo thứ tự từ trái sang phải, với các nhãn $A.1, A.2, \dots, A.k_v$



Trong hệ thống địa chỉ phổ quát:

- Với $n \geq 1$, đỉnh v ở mức n có nhãn $x_1 \dots x_n$, thể hiện đường đi duy nhất từ gốc đến v đi qua đỉnh x_1 ở mức 1, x_2 ở mức 2, \dots , x_n ở mức n
- Các đỉnh có thể được sắp xếp theo **thứ tự từ điển** (*lexicographic order*): Đỉnh có nhãn $x_1.x_2 \dots x_n$ nhỏ hơn đỉnh có nhãn $y_1.y_2 \dots y_m$ nếu tồn tại i , $0 \leq i \leq n$, sao cho $x_1 = y_1, x_2 = y_2, \dots, x_{i-1} = y_{i-1}$ và $x_i < y_i$; hoặc nếu $n < m$ và $x_i = y_i$ với mọi $i = 1, 2, \dots, n$



$0 < 1 < 1.1 < 1.1.1 < 1.1.2 < 1.2 < 2 < 2.1 < 2.2 < 2.2.1 < 3$



Bài tập 17

Giả sử địa chỉ của đỉnh v trong cây có gốc được sắp thứ tự T là 3.4.5.2.4.

- (a) v nằm ở mức nào?
- (b) Địa chỉ của đỉnh cha của v là gì?
- (c) Số lượng đỉnh anh em ít nhất mà v có thể có là bao nhiêu?
- (d) Nếu v có địa chỉ như trên, số lượng đỉnh nhỏ nhất có thể có trong T là bao nhiêu?
- (e) Tìm các địa chỉ khác mà bắt buộc phải xuất hiện trong cây.



Thuật toán 1: Duyệt tiền thứ tự (preorder traversal)

Input: T : cây có gốc sắp được thứ tự

Output: Danh sách các đỉnh của T

1 **procedure** preorder(T):

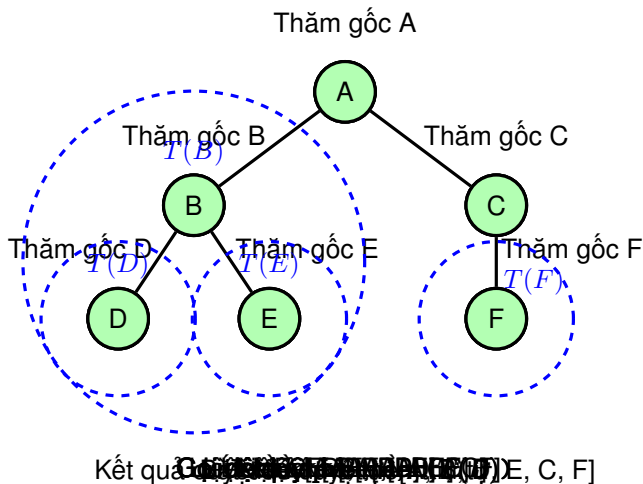
2 $r :=$ gốc của cây T

3 liệt kê r

4 **for** mỗi đỉnh con c của r từ trái sang phải **do**

5 $T(c) :=$ cây con gốc c

6 preorder($T(c)$)





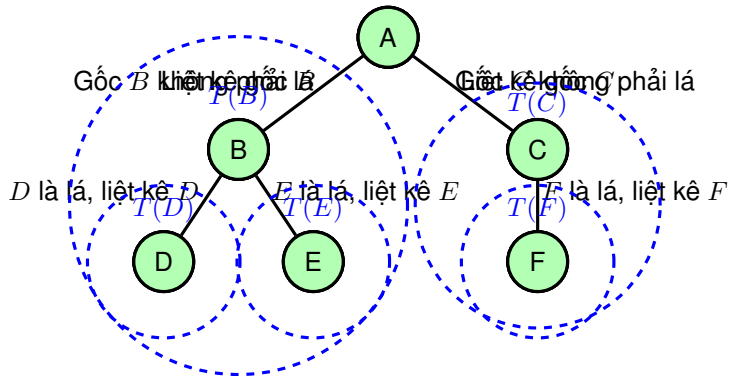
Thuật toán 2: Duyệt trung thứ tự (inorder traversal)

Input: T : cây có gốc sắp được thứ tự

Output: Danh sách các đỉnh của T

```
1 procedure inorder( $T$ ):  
2    $r :=$  gốc của cây  $T$   
3   if  $r$  là một nút lá then  
4     liệt kê  $r$   
5   else  
6      $l :=$  nút con đầu tiên của  $r$  từ trái sang phải  
7      $T(l) :=$  cây con gốc  $l$   
8     inorder( $T(l)$ )  
9     liệt kê  $r$   
10    for mỗi đỉnh con  $c$  của  $r$  từ trái sang phải, ngoại  
        trừ  $l$  do  
11       $T(c) :=$  cây con gốc  $c$   
12      inorder( $T(c)$ )
```

Gốc Liệt kê gốc phải lá



Kết quả của duyệt cây có gốc là dãy $[A, B, D, E, C, F]$



Thuật toán 3: Duyệt hậu thứ tự (postorder traversal)

Input: T : cây có gốc sắp được thứ tự

Output: Danh sách các đỉnh của T

1 **procedure** postorder(T):

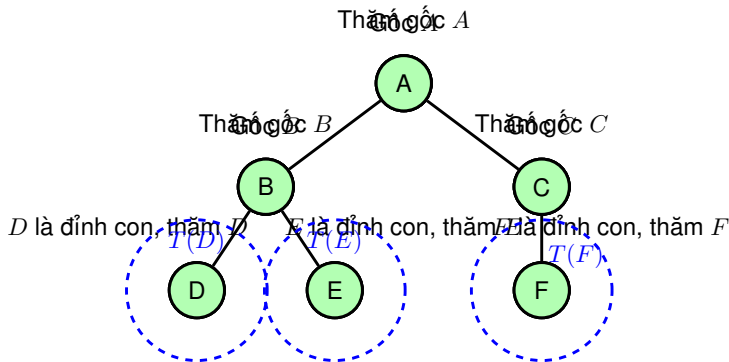
2 $r :=$ gốc của cây T

3 **for** mỗi đỉnh con c của r từ trái sang phải **do**

4 $T(c) :=$ cây con gốc c

5 preorder($T(c)$)

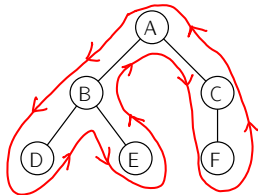
6 liệt kê r



Hình 1.1. Duyệt cây có gốc A

Cách đơn giản để liệt kê các đỉnh của một cây có gốc được sắp thứ tự theo tiền thứ tự, trung thứ tự, và hậu thứ tự

- Vẽ một **đường cong bao quanh cây có gốc được sắp thứ tự**, **bắt đầu từ gốc** và **di chuyển dọc theo các cạnh**
- Liệt kê các đỉnh theo **tiền thứ tự** bằng cách **liệt kê mỗi đỉnh ngay lần đầu tiên đường cong đi qua nó**
- Liệt kê các đỉnh theo **trung thứ tự** bằng cách **liệt kê một đỉnh lá ngay lần đầu tiên đường cong đi qua nó và liệt kê mỗi đỉnh trong vào lần thứ hai đường cong đi qua nó**
- Liệt kê các đỉnh theo **hậu thứ tự** bằng cách **liệt kê mỗi đỉnh vào lần cuối cùng nó được đường cong đi qua khi quay trở lại đỉnh cha của nó**



Tiền thứ tự: A, B, D, E, C, F

Trung thứ tự: D, B, E, A, F, C

Hậu thứ tự: D, E, B, F, C, A



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

Cây có gốc

38 Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

Có thể *biểu diễn các biểu thức phức tạp*, như các mệnh đề hợp thành, các phép kết hợp tập hợp, và các biểu thức số học bằng cách *sử dụng cây có gốc được sắp thứ tự*. Ví dụ, xét việc *biểu diễn một biểu thức số học (arithmetic expression)* liên quan đến các phép toán $+$ (cộng), $-$ (trừ), $*$ (nhân), $/$ (chia), và \uparrow (lũy thừa)

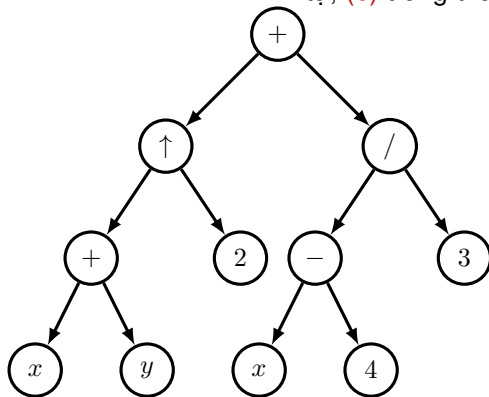
- Các dấu ngoặc đơn được sử dụng để chỉ ra thứ tự của các phép toán
- Một cây có gốc được sắp thứ tự có thể được sử dụng để biểu diễn các biểu thức như vậy, trong đó các đỉnh trong biểu diễn các phép toán, và các đỉnh lá biểu diễn các biến hoặc số
- Mỗi phép toán thực hiện trên cây con trái và cây con phải của nó (theo thứ tự đó)

Ví dụ 4

Bài tập 18

Liệt kê các đỉnh của cây trong Ví dụ

2. Xây dựng cây biểu diễn $((x + y) \uparrow 2) + ((x - 4)/3)$
4. Bằng cách duyệt theo (a) tiền thứ tự, (b) trung thứ tự, và (c) hậu thứ tự



Hình: Xây dựng cây biểu diễn biểu thức $((x + y) \uparrow 2) + ((x - 4)/3)$

Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

Cây có gốc

39 Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Cây nhị phân tương ứng với biểu thức số học (như trong Ví dụ 4) có thể được sử dụng để biểu diễn ***các dạng khác nhau của biểu thức*** đó dựa vào các cách duyệt cây

Dạng trung tố (infix form)

- Được tạo bằng cách duyệt cây theo ***trung thứ tự***: Toán hạng bên trái của toán tử được duyệt trước, sau đó là toán tử, và cuối cùng là toán hạng bên phải của toán tử
- ***Tính toán*** một biểu thức ở dạng trung tố:
 - Thực hiện ***từ trái sang phải***
 - Khi ***gặp một toán tử***, thực hiện phép toán tương ứng với ***hai toán hạng ngay bên trái và bên phải của toán tử này***
 - Khi ***một phép toán được thực hiện***, ***kết quả sẽ được xem như một toán hạng mới***

Ví dụ 5

- Dạng quen thuộc: $(x + y)^2 + \frac{x - 4}{3}$
- Đối với cây trong Ví dụ 4: $((x + y) \uparrow 2) + ((x - 4)/3)$ (cần thêm các ngoặc đơn mỗi khi gặp một toán tử để tránh nhầm lẫn)

Cây

Duyệt cây có gốc



Cây nhị phân tương ứng với biểu thức số học (như trong Ví dụ 4) có thể được sử dụng để biểu diễn *các dạng khác nhau của biểu thức* đó dựa vào các cách duyệt cây

Dạng tiền tố (prefix form)

- Được tạo bằng cách duyệt cây theo *tiền thứ tự*: Toán tử được duyệt trước các toán hạng của nó
- Còn được gọi là *ký pháp Ba Lan (Polish notation)* (đặt theo tên nhà logic học Ba Lan Jan Łukasiewicz)
- *Tính toán* một biểu thức ở dạng tiền tố:
 - Thực hiện *từ phải sang trái*
 - Khi *gặp một toán tử*, thực hiện phép toán tương ứng với *hai toán hạng ngay bên phải của toán tử này*
 - Khi *một phép toán được thực hiện*, *kết quả sẽ được xem như một toán hạng mới*

Ví dụ 6

- Đối với cây trong Ví dụ 4: $+ \uparrow +xy2 / - x43$ (không cần ngoặc đơn)

Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

Cây có gốc

41 Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Cây nhị phân tương ứng với biểu thức số học (như trong Ví dụ 4) có thể được sử dụng để biểu diễn các dạng khác nhau của biểu thức đó dựa vào các cách duyệt cây

Dạng hậu tố (postfix form)

- Được tạo bằng cách duyệt cây theo *hậu thứ tự*: Toán tử được duyệt sau các toán hạng của nó
- Còn được gọi là ký pháp Ba Lan nghịch đảo (Reverse Polish notation)
- *Tính toán* một biểu thức ở dạng hậu tố:
 - Thực hiện *từ trái sang phải*
 - Khi *gặp một toán tử*, thực hiện phép toán tương ứng với *hai toán hạng ngay bên trái của toán tử này*
 - Khi *một phép toán được thực hiện*, *kết quả sẽ được xem như một toán hạng mới*

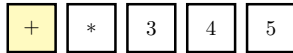
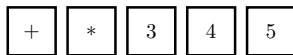
Ví dụ 7

- Đối với cây trong Ví dụ 4: $xy + 2 \uparrow x4 - 3 / +$ (không cần ngoặc đơn)



Ví dụ 8 (Tính toán biểu thức dạng tiền tố)

Biểu thức tiền tố: $+ * 345$



Bước 1: Gặp toán tử $+$, cần hai toán hạng



Bước 2: Tính $*34 = 3 \times 4 = 12$

$$3 \times 4 = 12$$



Bước 3: Tính $+125 = 12 + 5 = 17$

$$12 + 5 = 17$$

17

Kết quả cuối cùng

Hình: Minh họa quá trình tính toán biểu thức dạng tiền tố $+ * 345$



Ví dụ 9 (Tính toán biểu thức dạng hậu tố)

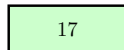
Biểu thức hậu tố: $3\ 4\ *\ 5\ +$



$$3 \times 4 = 12$$



$$12 + 5 = 17$$



Bước 1: Gặp $*$, áp dụng với hai toán hạng trước nó

Bước 2: Gặp $+$, áp dụng với hai toán hạng trước nó

Kết quả cuối cùng

Hình: Minh họa quá trình tính toán biểu thức dạng hậu tố $3\ 4\ *\ 5\ +$



Bài tập 19

- (a) Biểu diễn biểu thức $((x + 2) \uparrow 3) * (y - (3 + x)) - 5$ bằng cây nhị phân
- (b) Viết biểu thức này ở dạng tiền tố
- (c) Viết biểu thức này ở dạng hậu tố
- (d) Viết biểu thức này ở dạng trung tố

Bài tập 20

Vẽ cây có gốc được sắp thứ tự tương ứng với mỗi biểu thức số học được viết ở dạng tiền tố sau đây. Sau đó viết mỗi biểu thức ở dạng trung tố.

- (a) $+ * + - 53214$
- (b) $\uparrow +23 - 51$
- (c) $*/93 + *24 - 76$



Bài tập 21

Giá trị của mỗi biểu thức tiền tố sau là bao nhiêu?

(a) $- * 2 / 843$

(b) $\uparrow - * 33 * 425$

(c) $+ - \uparrow 32 \uparrow 23 / 6 - 42$

(d) $* + 3 + 3 \uparrow 3 + 333$

Bài tập 22

Giá trị của mỗi biểu thức hậu tố sau là bao nhiêu?

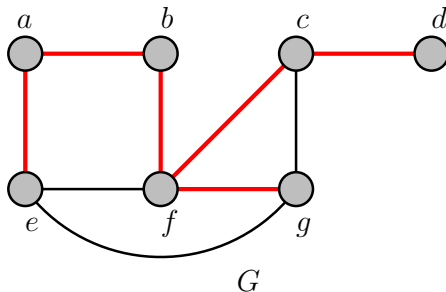
(a) $521 - -314 + +*$

(b) $93 / 5 + 72 - *$

(c) $32 * 2 \uparrow 53 - 84 / * -$



Cho $G = (V, E)$ là một đơn đồ thị. Một **cây bao trùm (spanning tree)** của G là một đồ thị con T của G thỏa mãn điều kiện T là một cây và T chứa tất cả các đỉnh của G





Định lý 7

Mọi đơn đồ thị liên thông G có một cây bao trùm

Chứng minh.

Ta xây dựng một cây bao trùm của G như sau

- Lấy một chu trình trong G nếu có
- Xóa một cạnh từ chu trình đó. Đồ thị mới thu được vẫn là liên thông
- Lặp lại các bước trên đến khi không có chu trình trong G

Do G có hữu hạn số đỉnh, cuối cùng ta thu được một cây bao trùm của G (Tại sao?) □

Bài tập 23

- Đơn đồ thị vô hướng liên thông nào có chính xác một cây bao trùm?
- Khi nào thì một cạnh của một đơn đồ thị vô hướng liên thông phải thuộc mọi cây bao trùm của đồ thị đó?

Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

48 Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Cho $G = (V, E)$ là một đơn đồ thị vô hướng liên thông. Ta trình bày thuật toán **duyet theo chiều sâu (depth-first search, DFS)** để tìm một cây bao trùm của G

Thuật toán 4: Duyệt theo chiều sâu

Input: G : đơn đồ thị vô hướng liên thông với các đỉnh

v_1, v_2, \dots, v_n

Output: Cây bao trùm T

1 **procedure** dfs(G):

2 $T :=$ cây có chính xác một đỉnh v_1

3 visit(v_1)

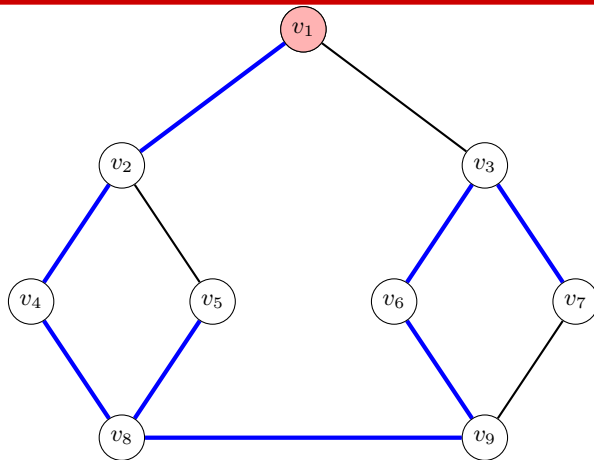
4 **procedure** visit(v : một đỉnh của G):

5 **for** mỗi đỉnh w liền kề với v và chưa thuộc T **do**

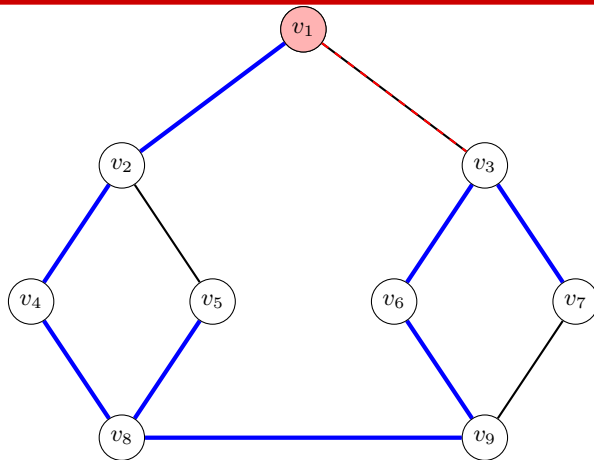
6 Thêm w và cạnh vw vào T

7 visit(w)

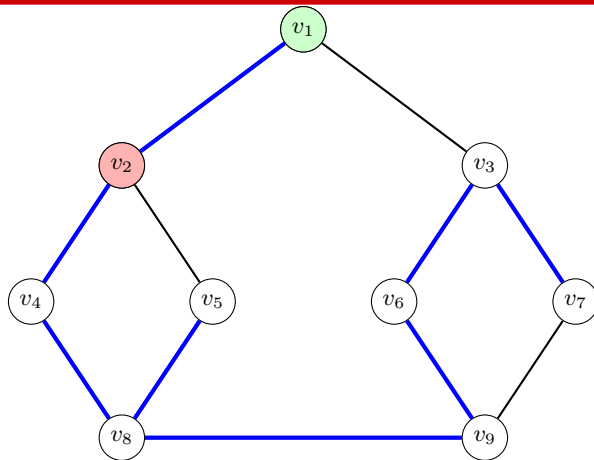
8 **return** T



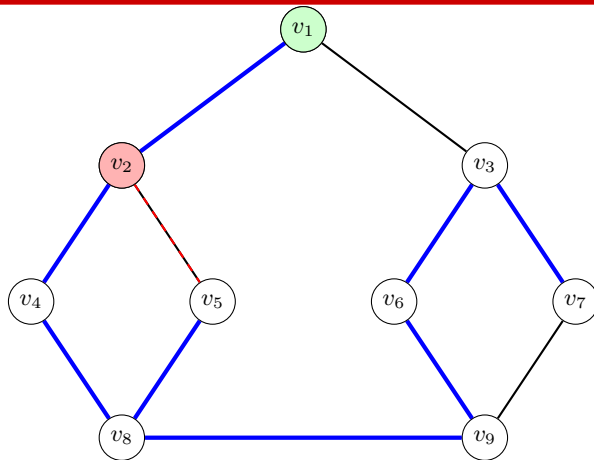
Bắt đầu thuật toán: Khởi tạo T với đỉnh v_1



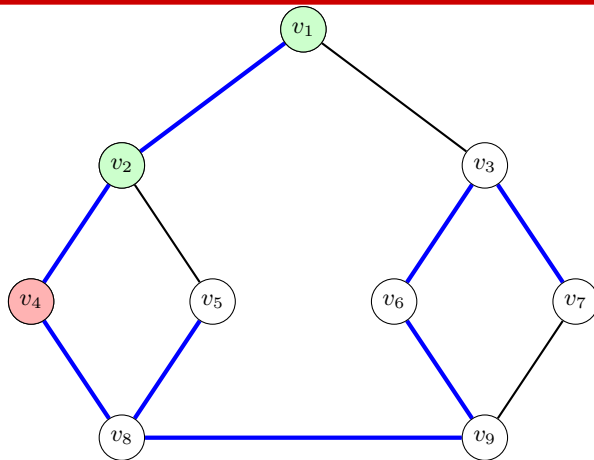
Gọi $\text{visit}(v_1)$, xem xét các đỉnh liền kề v_2 và v_3



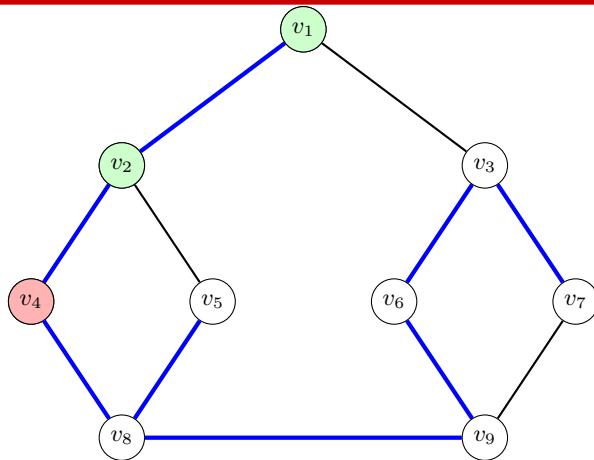
Thêm v_2 và cạnh v_1v_2 vào T



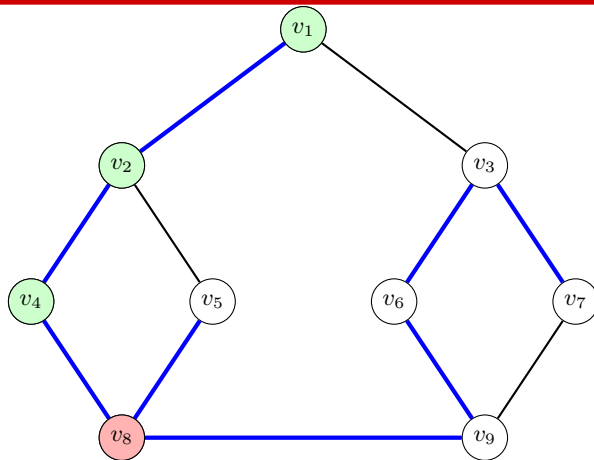
Gọi $\text{visit}(v_2)$, xem xét các đỉnh liền kề v_4 và v_5



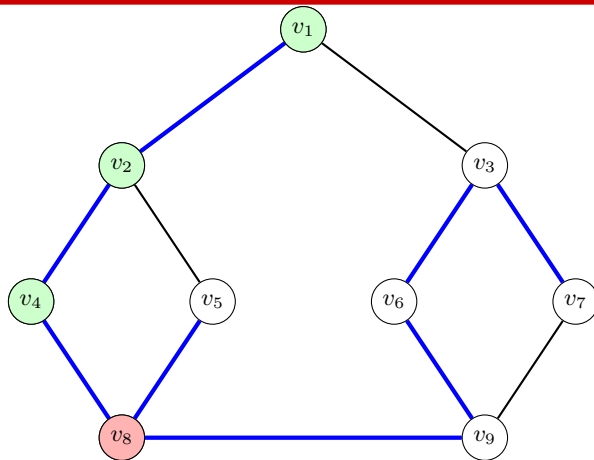
Thêm v_4 và cạnh v_2v_4 vào T



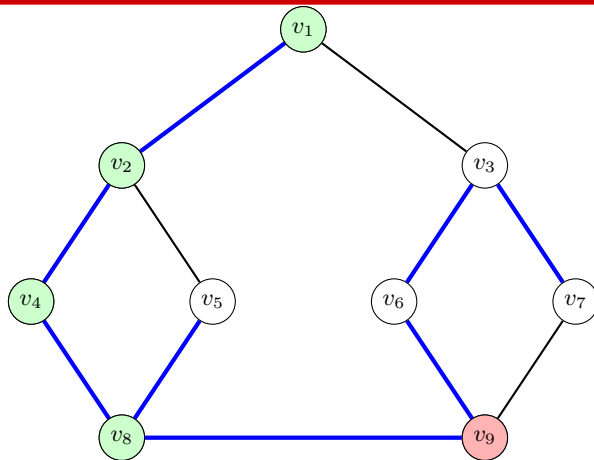
Gọi $\text{visit}(v_4)$, xem xét đỉnh liền kề v_8



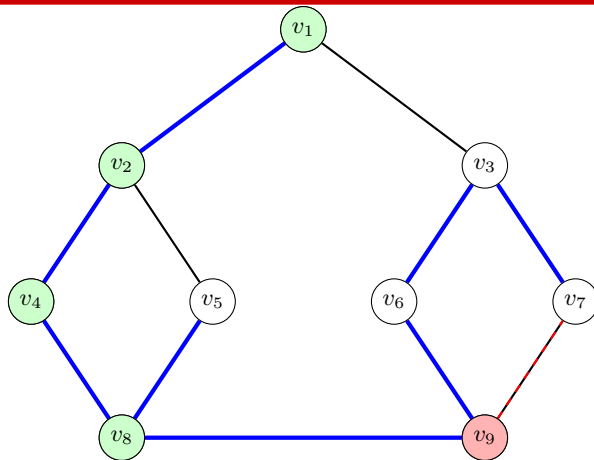
Thêm v_8 và cạnh v_4v_8 vào T



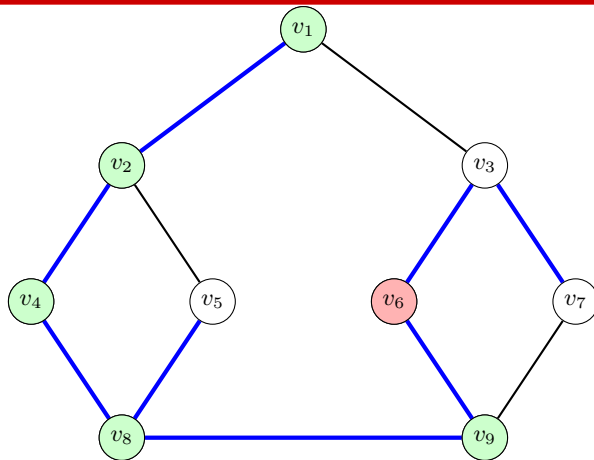
Gọi $\text{visit}(v_8)$, xem xét đỉnh liền kề v_5 và v_9



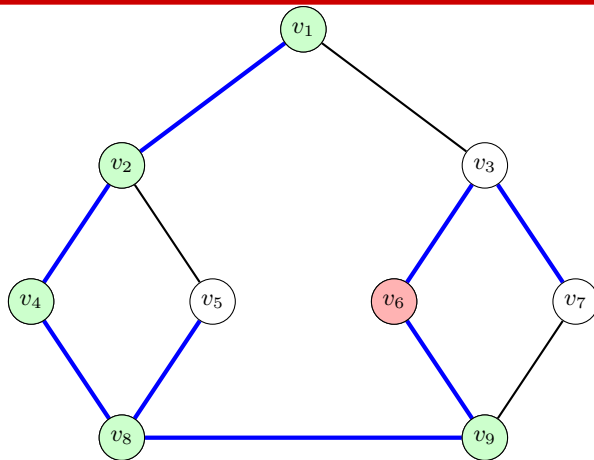
Thêm v_9 và cạnh v_8v_9 vào T



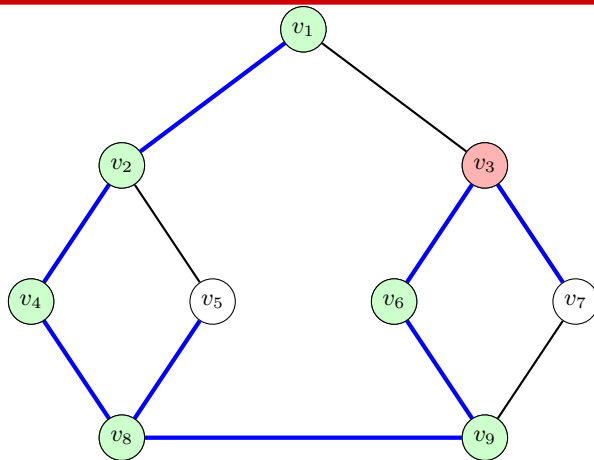
Gọi $\text{visit}(v_9)$, xem xét đỉnh liền kề v_6 và v_7 chưa thuộc T



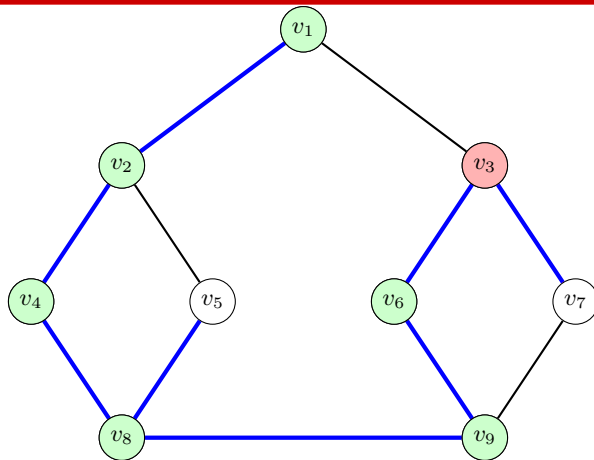
Thêm v_6 và cạnh v_9v_6 vào T



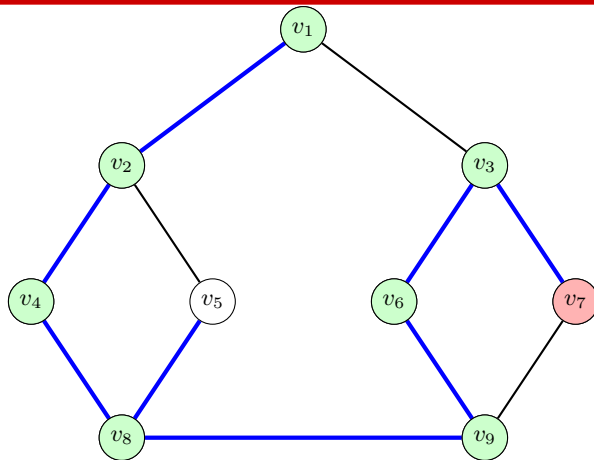
Gọi $\text{visit}(v_6)$, xem xét đỉnh liền kề v_3



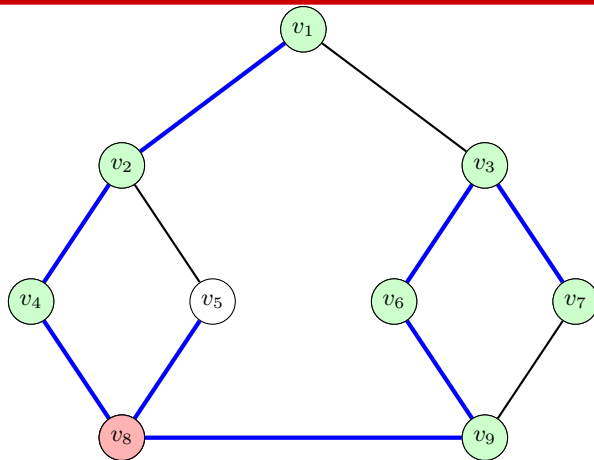
Thêm v_3 và cạnh v_6v_3 vào T



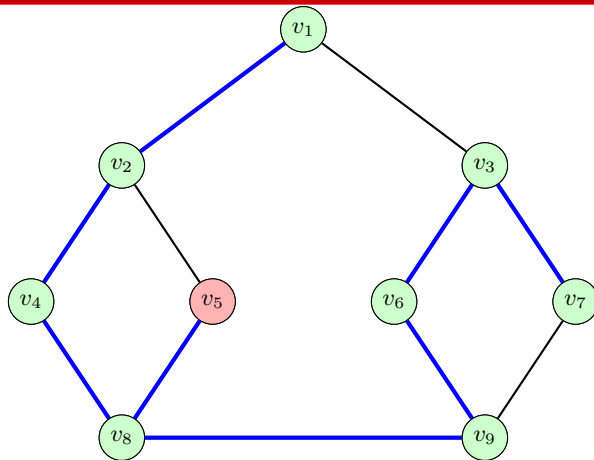
Gọi $\text{visit}(v_3)$, xem xét đỉnh liền kề v_7 chưa thuộc T



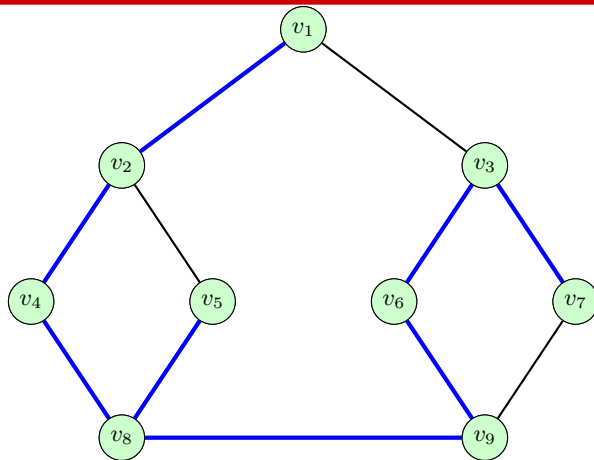
Thêm v_7 và cạnh v_3v_7 vào T



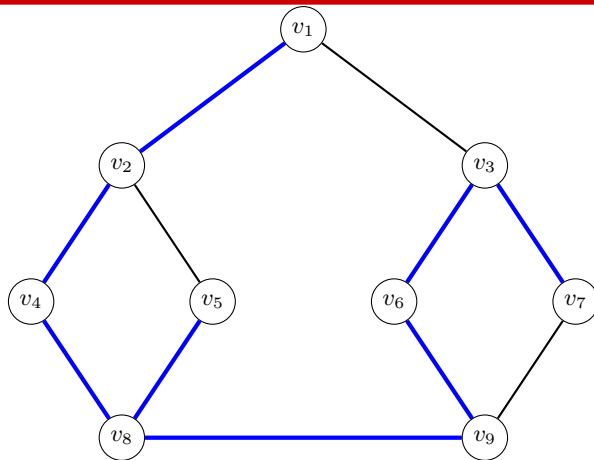
Quay lui đến $\text{visit}(v_8)$, xét đỉnh v_5 chưa thuộc T



Thêm v_5 và cạnh v_8v_5 vào T



Thuật toán hoàn thành: Cây bao trùm T được tìm thấy



Thứ tự duyệt: $v_1 \rightarrow v_2 \rightarrow v_4 \rightarrow v_8 \rightarrow v_9 \rightarrow v_6 \rightarrow v_3 \rightarrow v_7 \rightarrow v_5$

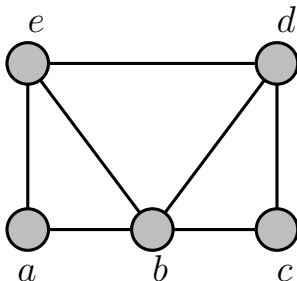


- Trước mỗi lần lặp trong thủ tục `visit()`, T là cây chứa toàn bộ các đỉnh đã được gọi bởi thủ tục `visit()`
- Thuật toán DFS chạy trong thời gian $O(n^2)$ với đơn đồ thị vô hướng liên thông $G = (V, E)$ bất kỳ, trong đó $n = |V|$
 - `visit()` được gọi cho mỗi đỉnh chính xác một lần
 - Một cạnh e trong G được xét tối đa hai lần để xác định xem e và một đầu mút của e có thể được thêm vào cây T hay không
 - Do đó, thuật toán DFS chạy trong thời gian $O(|E|)$. Chú ý rằng $|E| \leq n(n-1)/2 = O(n^2)$
- Thuật toán DFS có thể được sử dụng như là cơ sở để giải quyết nhiều bài toán khác, ví dụ như bài toán tìm các đường đi và chu trình trong đồ thị, bài toán xác định các thành phần liên thông, bài toán tìm đỉnh cắt, v.v... DFS cũng là cơ sở cho *kỹ thuật quay lui (backtracking technique)* được sử dụng để thiết kế các giải thuật cho nhiều bài toán khó



- Có những bài toán chỉ có thể giải được bằng cách tìm kiếm trên tập toàn bộ các lời giải. Để tiến hành tìm kiếm lời giải một cách có hệ thống, một hướng tiếp cận là sử dụng cây quyết định với mỗi đỉnh trong mô tả một quyết định và mỗi đỉnh lá ứng với một lời giải nào đó
- Trong *kỹ thuật quay lui (backtracking)*, để tìm kiếm lời giải mong muốn, đầu tiên ta tiến hành một dãy các quyết định sao cho ta có thể tìm đến các lời giải ở càng xa đỉnh gốc càng tốt. Một dãy các quyết định này tương ứng với một đường đi trong cây quyết định từ gốc đến lá
- Một khi ta biết rằng không thể tìm đến một lời giải mới xa hơn thông qua dãy các quyết định, ta lui lại xét đỉnh cha của đỉnh đang xét hiện tại, và tiếp tục tìm đến một lời giải khác dựa trên một dãy các quyết định mới nếu có thể
- Quá trình này được lặp lại cho đến khi tìm được lời giải mong muốn hoặc kết luận rằng không có lời giải nào như thế tồn tại

Bài tập 24 (★)



Hãy tô màu đồ thị trên bằng 3 màu đỏ (**R**), xanh lá cây (**G**), và xanh da trời (**B**) dựa trên kỹ thuật quay lui đã mô tả. (**Gợi ý:** Xem [Rosen 2012], Mục 11.4.4)

Cho $G = (V, E)$ là một đơn đồ thị vô hướng liên thông. Ta trình bày thuật toán **duyệt theo chiều rộng (breadth-first search, BFS)** để tìm một cây bao trùm của G

Thuật toán 5: Duyệt theo chiều rộng

Input: G : đơn đồ thị vô hướng liên thông với các đỉnh v_1, v_2, \dots, v_n

Output: Cây bao trùm T

```
1  $T :=$  cây chỉ chứa duy nhất một đỉnh  $v_1$ 
2  $L :=$  danh sách rỗng
3 Thêm  $v_1$  vào danh sách  $L$  các đỉnh chưa xét
4 while  $L$  khác rỗng do
5     |   Bỏ đi đỉnh thứ nhất  $v$  từ  $L$ 
6     |   for mỗi đỉnh  $w$  liền kề với  $v$  do
7     |   |   if  $w$  không thuộc  $L$  và  $w$  không thuộc  $T$  then
8     |   |   |   Thêm  $w$  vào cuối danh sách  $L$ 
9     |   |   |   Thêm  $w$  và cạnh  $vw$  vào  $T$ 
10 return  $T$ 
```

Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

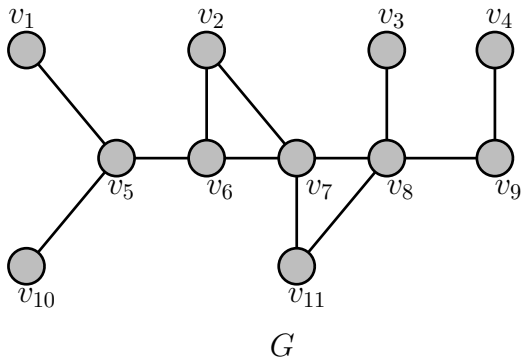
54 Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo

Ví dụ 10

Tìm một cây bao trùm T của đồ thị $G = (V, E)$ sau bằng thuật toán duyệt theo chiều rộng



Cây

Cây bao trùm



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

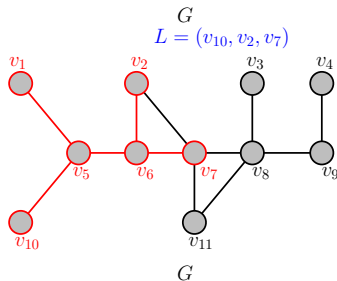
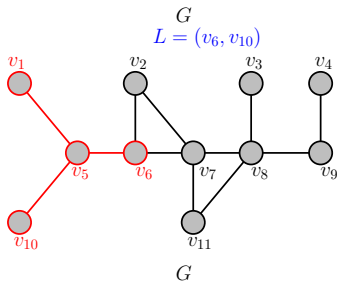
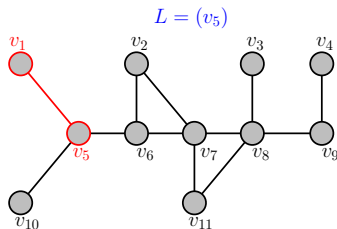
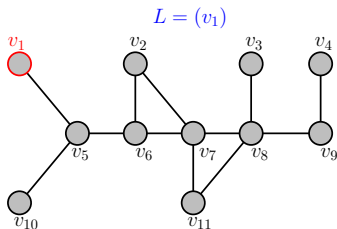
Cây có gốc

Duyệt cây có gốc

56 Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Cây

Cây bao trùm



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

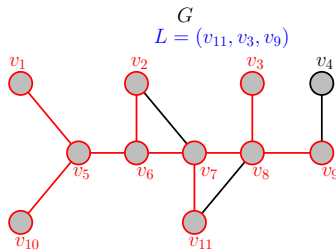
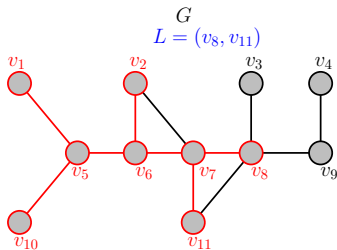
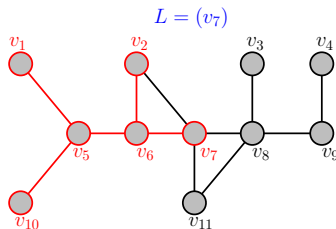
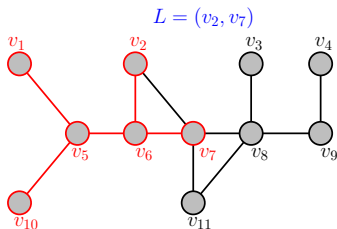
Cây có gốc

Duyệt cây có gốc

57 Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Cây

Cây bao trùm



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

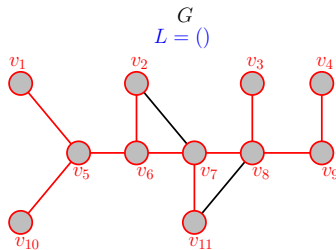
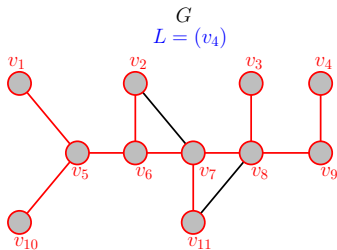
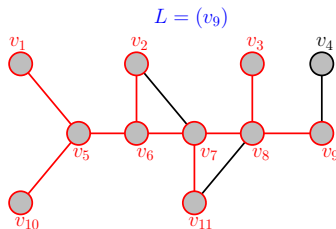
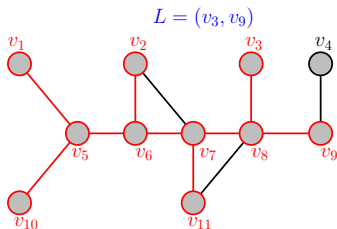
Cây có gốc

Duyệt cây có gốc

58 Cây bao trùm

Cây bao trùm nhỏ nhất

Tài liệu tham khảo



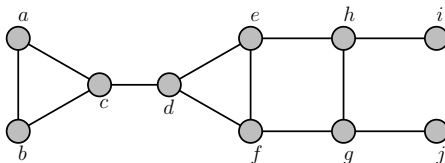


- Trước mỗi lần lặp vòng **while**, T là cây chứa các đỉnh đã xét (= các đỉnh đã bỏ đi từ danh sách L)
- Thuật toán BFS chạy trong thời gian $O(n^2)$ với đơn đồ thị vô hướng liên thông $G = (V, E)$ bất kỳ, trong đó $n = |V|$
 - Với mỗi đỉnh v , thuật toán xét các đỉnh liền kề với v và thêm các đỉnh chưa xét vào cuối danh sách L
 - Mỗi cạnh được xét nhiều nhất hai lần để xác định xem có cần thêm cạnh đó và một đỉnh đầu mút vào cây T hay không
 - Do đó, thuật toán BFS chạy trong thời gian $O(|E|)$, hay nói cách khác $O(n^2)$
- Thuật toán BFS là cơ sở để thiết kế các thuật toán giải nhiều bài toán khác nhau, ví dụ như bài toán tìm các thành phần liên thông, bài toán xác định xem một đồ thị có phải đồ thị hai phần hay không, bài toán tìm đường đi ngắn nhất giữa hai đỉnh, v.v...

Bài tập 25

Tìm một cây bao trùm của đồ thị sau bằng

- (1) thuật toán tìm kiếm theo chiều sâu (DFS);
 - (2) thuật toán tìm kiếm theo chiều rộng (BFS);
- bắt đầu từ đỉnh a



Bài tập 26

Hãy mô tả các cây bao trùm xuất ra bởi thuật toán tìm kiếm theo chiều rộng và thuật toán tìm kiếm theo chiều sâu khi chạy trên đồ thị đầy đủ K_n , với n là số nguyên dương nào đó

Cây

Cây bao trùm nhỏ nhất



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

Cây bao trùm

61 Cây bao trùm nhỏ nhất

Tài liệu tham khảo

Cho $G = (V, E, w)$ là một đồ thị liên thông vô hướng có trọng số. Một **cây bao trùm nhỏ nhất (minimum spanning tree)** của G là một cây bao trùm của G có tổng trọng số các cạnh của cây là nhỏ nhất có thể

- **Input:** Đồ thị liên thông vô hướng có trọng số $G = (V, E, w)$
- **Output:** Một cây bao trùm nhỏ nhất T của G

- Thuật toán Prim
- Thuật toán Kruskal



Thuật toán 6: Thuật toán Prim

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

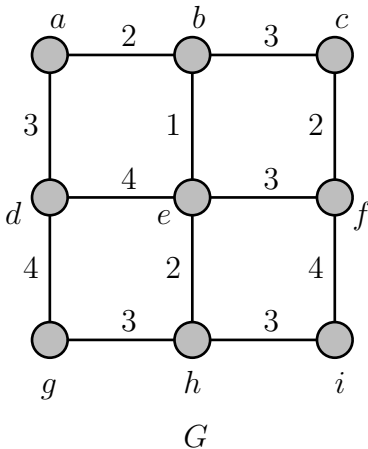
Output: Một cây bao trùm nhỏ nhất T của G

```
1  $T :=$  một cạnh có trọng số nhỏ nhất
2 for  $i := 1$  to  $n - 2$  do
3    $e :=$  một cạnh có trọng số nhỏ nhất liên thuộc với một
   đỉnh của  $T$  thỏa mãn  $T + e$  không có chu trình
4    $T := T + e$ 
5 return  $T$ 
```



Ví dụ 11

Tìm một cây bao trùm nhỏ nhất T của đồ thị $G = (V, E, w)$ sau bằng thuật toán Prim



Cây

Cây bao trùm nhỏ nhất



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

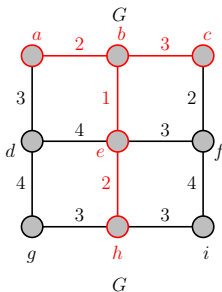
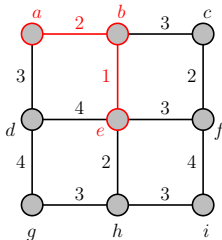
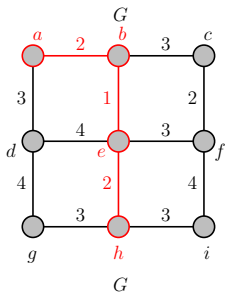
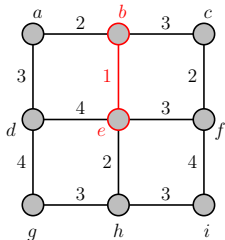
Cây có gốc

Duyệt cây có gốc

Cây bao trùm

64 Cây bao trùm nhỏ nhất

Tài liệu tham khảo



Cây

Cây bao trùm nhỏ nhất



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

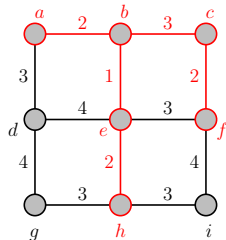
Cây có gốc

Duyệt cây có gốc

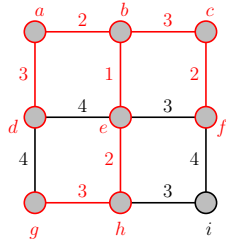
Cây bao trùm

65 Cây bao trùm nhỏ nhất

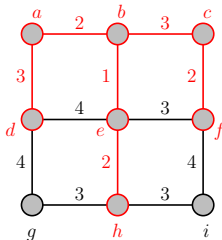
Tài liệu tham khảo



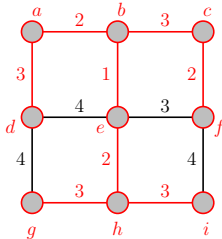
G



G



G



G



Định lý 8

Nếu T là một cây bao trùm xuất ra bởi thuật toán Prim với đồ thị đầu vào $G = (V, E, w)$ thì T là một cây bao trùm nhỏ nhất của G

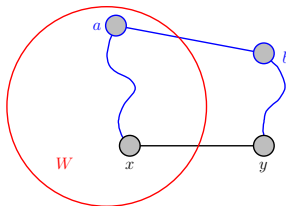
Chứng minh.

- Thuật toán Prim luôn xuất ra một cây bao trùm
 - Ở các bước trung gian, T luôn là cây
 - Mỗi bước trung gian, thuật toán thêm một cạnh và một đỉnh mới vào T
- Ta chứng minh bằng phản chứng. Giả sử T không có tổng trọng số nhỏ nhất
- Gọi $ET = (e_1, e_2, \dots, e_m)$ là dãy các cạnh được chọn theo thứ tự thực hiện của thuật toán Prim
- Gọi U là một cây bao trùm nhỏ nhất thỏa mãn điều kiện U có chứa k cạnh đầu tiên của ET với k lớn nhất có thể, nghĩa là e_1, e_2, \dots, e_k thuộc U và $e_{k+1} = xy$ không thuộc U



Chứng minh (tiếp).

- Gọi W là tập các đỉnh của T tính đến thời điểm *ngay trước khi* $e_{k+1} = xy$ *được chọn*
- Gọi P là một đường đi giữa x và y trong U (chú ý rằng xy không là một cạnh của U) và gọi ab là cạnh đầu tiên của P có một đầu mút (a) thuộc W và một đầu mút (b) không thuộc W
- Xét cây bao trùm $S = U - ab + xy$
 - Nếu $w(a, b) > w(x, y)$: Tổng trọng số của S nhỏ hơn của U , mâu thuẫn với giả thiết U là cây bao trùm nhỏ nhất
 - Nếu $w(a, b) = w(x, y)$: S là một cây bao trùm nhỏ nhất và S có chứa nhiều cạnh của ET hơn U , mâu thuẫn với định nghĩa của U
 - Nếu $w(a, b) < w(x, y)$: Thuật toán Prim sẽ chọn cạnh ab thay vì cạnh xy , mâu thuẫn với giả thiết ban đầu là xy là cạnh được chọn





- Thuật toán Prim là một thuật toán tham lam
 - Ở mỗi bước xây dựng cây bao trùm nhỏ nhất T , thuật toán Prim chỉ thêm các cạnh có trọng số nhỏ nhất liên kết với một đỉnh của cây T tính đến hiện tại mà không tạo thành chu trình mới
- Thuật toán Prim có thể được lập trình để chạy trong thời gian $O(m \log n)$ với đồ thị $G = (V, E, w)$ thỏa mãn $m = |E|$ và $n = |V|$
- Thuật toán Prim tương tự như thuật toán Dijkstra tìm đường đi ngắn nhất giữa hai đỉnh đã giới thiệu trước đó



Thuật toán 7: Thuật toán Kruskal

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

Output: Một cây bao trùm nhỏ nhất T của G

```
1  $T :=$  đồ thị rỗng (không có đỉnh và cạnh)
2 for  $i := 1$  to  $n - 1$  do
3    $e :=$  một cạnh có trọng số nhỏ nhất thỏa mãn  $T + e$ 
   không có chu trình
4    $T := T + e$ 
5 return  $T$ 
```

Cây

Cây bao trùm nhỏ nhất



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

Cây có gốc

Duyệt cây có gốc

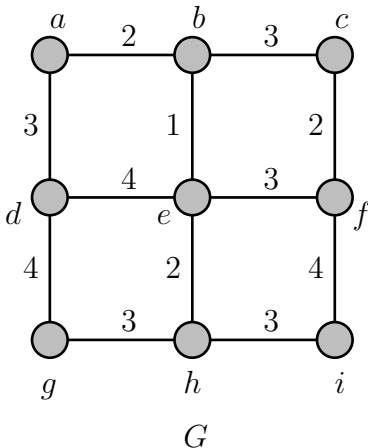
Cây bao trùm

70 Cây bao trùm nhỏ nhất

Tài liệu tham khảo

Ví dụ 12

Tìm một cây bao trùm nhỏ nhất T của đồ thị $G = (V, E, w)$ sau bằng thuật toán Kruskal



Cây

Cây bao trùm nhỏ nhất



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

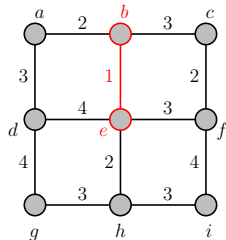
Cây có gốc

Duyệt cây có gốc

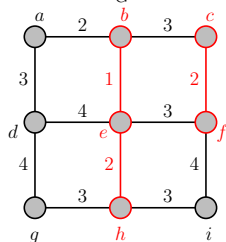
Cây bao trùm

71 Cây bao trùm nhỏ nhất

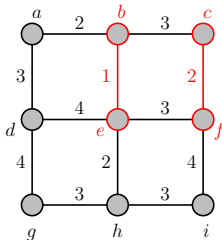
Tài liệu tham khảo



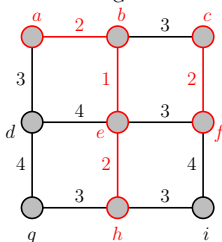
G



G



G



G

Cây

Cây bao trùm nhỏ nhất



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của cây

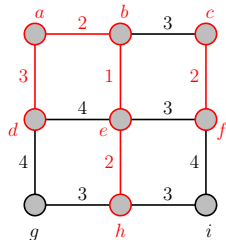
Cây có gốc

Duyệt cây có gốc

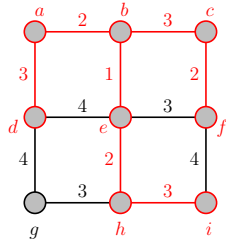
Cây bao trùm

72 Cây bao trùm nhỏ nhất

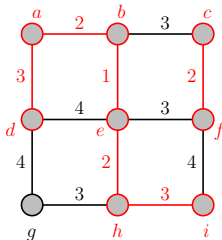
Tài liệu tham khảo



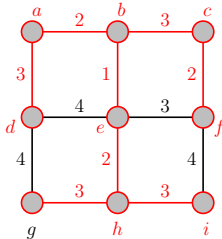
G



G



G



G



Bài tập 27

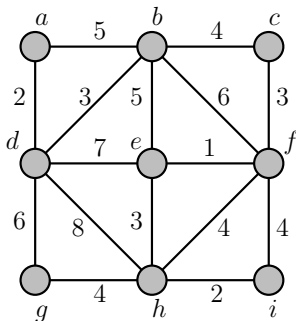
Chứng minh rằng nếu T là một cây bao trùm của $G = (V, E, w)$ xuất ra bởi thuật toán Kruskal thì T là cây bao trùm nhỏ nhất (**Gợi ý:** xem lại chứng minh tính đúng đắn của thuật toán Prim)

- Thuật toán Kruskal là một thuật toán tham lam
 - Ở mỗi bước xây dựng cây bao trùm nhỏ nhất T , thuật toán Kruskal chỉ thêm các cạnh có trọng số nhỏ nhất mà không tạo thành chu trình mới
 - Khác với thuật toán Prim, *ở các bước trung gian của thuật toán Kruskal, T có thể không là cây*
- Thuật toán Kruskal có thể được lập trình để chạy trong thời gian $O(m \log m)$ với đồ thị $G = (V, E, w)$ thỏa mãn $m = |E|$



Bài tập 28

Tìm một cây bao trùm nhỏ nhất của đồ thị sau



bằng cách sử dụng

(a) thuật toán Prim

(b) thuật toán Kruskal

Tài liệu tham khảo



Lý thuyết đồ thị III

Hoàng Anh Đức

Giới thiệu

Một số tính chất của
cây


Cây có gốc

Duyệt cây có gốc

Cây bao trùm

Cây bao trùm nhỏ nhất

75 Tài liệu tham khảo

 Rosen, Kenneth (2012). *Discrete Mathematics and Its Applications*. 7th. McGraw-Hill.