**CHƯƠNG VI**

# CÂY

Một đồ thị liên thông và không có chu trình được gọi là cây. Cây đã được dùng từ năm 1857, khi nhà toán học Anh tên là Arthur Cayley dùng cây để xác định những dạng khác nhau của hợp chất hoá học. Từ đó cây đã được dùng để giải nhiều bài toán trong nhiều lĩnh vực khác nhau. Cây rất hay được sử dụng trong tin học. Chẳng hạn, người ta dùng cây để xây dựng các thuật toán rất có hiệu quả để định vị các phần tử trong một danh sách. Cây cũng dùng để xây dựng các mạng máy tính với chi phí rẻ nhất cho các đường điện thoại nối các máy phân tán. Cây cũng được dùng để tạo ra các mã có hiệu quả để lưu trữ và truyền dữ liệu. Dùng cây có thể mô hình các thủ tục mà để thi hành nó cần dùng một dãy các quyết định. Vì vậy cây đặc biệt có giá trị khi nghiên cứu các thuật toán sắp xếp.

**6.1. ĐỊNH NGHĨA VÀ CÁC TÍNH CHẤT CƠ BẢN.**

**6.1.1. Định nghĩa:** Cây là một đồ thị vô hướng liên thông, không chứa chu trình và có ít nhất hai đỉnh.

Một đồ thị vô hướng không chứa chu trình và có ít nhất hai đỉnh gọi là một rừng. Trong một rừng, mỗi thành phần liên thông là một cây.

**Thí dụ 1:** Rừng sau có 3 cây:

**6.1.2. Mệnh đề:** Nếu T là một cây có n đỉnh thì T có ít nhất hai đỉnh treo.

**Chứng minh:** Lấy một cạnh (a,b) tuỳ ý của cây T. Trong tập hợp các đường đi sơ cấp chứa cạnh (a,b), ta lấy đường đi từ u đến v dài nhất. Vì T là một cây nên u ≠ v. Mặt khác, u và v phải là hai đỉnh treo, vì nếu một đỉnh, u chẳng hạn, không phải là đỉnh treo thì u phải là đầu mút của một cạnh (u,x), với x là đỉnh không thuộc đường đi từ u đến v. Do đó, đường đi sơ cấp từ x đến v, chứa cạnh (a,b), dài hơn đường đi từ u đến v, trái với tính chất đường đi từ u đến v đã chọn.

**6.1.3. Định lý:** Cho T là một đồ thị có n ≥ 2 đỉnh. Các điều sau là tương đương:

**1)** T là một cây.

**2)** T liên thông và có n−1 cạnh.

**3)** T không chứa chu trình và có n−1 cạnh.

**4)** T liên thông và mỗi cạnh là cầu.

**5)** Giữa hai đỉnh phân biệt bất kỳ của T luôn có duy nhất một đường đi sơ cấp.

**6)** T không chứa chu trình nhưng khi thêm một cạnh mới thì có được một chu trình duy nhất.

**Chứng minh:** **1)⇒2)** Chỉ cần chứng minh rằng một cây có n đỉnh thì có n−1 cạnh. Ta chứng minh bằng quy nạp. Điều này hiển nhiên khi n=2. Giả sử cây có k đỉnh thì có k−1 cạnh, ta chứng minh rằng cây T có k+1 đỉnh thì có k cạnh. Thật vậy, trong T nếu ta xoá một đỉnh treo và cạnh treo tương ứng thì đồ thị nhận được là một cây k đỉnh, cây này có k−1 cạnh, theo giả thiết quy nạp. Vậy cây T có k cạnh.

**2)⇒3)** Nếu T có chu trình thì bỏ đi một cạnh trong chu trình này thì T vẫn liên thông. Làm lại như thế cho đến khi trong T không còn chu trình nào mà vẫn liên thông, lúc đó ta được một cây có n đỉnh nhưng có ít hơn n−1 cạnh, trái với 2).

**3)⇒4)** Nếu T có k thành phần liên thông T­1, ..., Tk lần lượt có số đỉnh là n1, ..., nk (với n1+n2+ … +nk=n) thì mỗi Ti là một cây nên nó có số cạnh là ni−1. Vậy ta có

n−1=(n1−1)+(n2−1)+ ... +(nk−1)=(n1+n2+ … +nk)−k=n−k.

Do đó k=1 hay T liên thông. Hơn nữa, khi bỏ đi một cạnh thì T hết liên thông, vì nếu còn liên thông thì T là một cây n đỉnh với n−2 cạnh, trái với điều đã chứng minh ở trên.

**4)⇒5)** Vì T liên thông nên giữa hai đỉnh phân biệt bất kỳ của T luôn có một đường đi sơ cấp, nhưng không thể được nối bởi hai đường đi sơ cấp vì nếu thế, hai đường đó sẽ tạo ra một chu trình và khi bỏ một cạnh thuộc chu trình này, T vẫn liên thông, trái với giả thiết.

**5)⇒6)** Nếu T chứa một chu trình thì hai đỉnh bất kỳ trên chu trình này sẽ được nối bởi hai đường đi sơ cấp. Ngoài ra, khi thêm một cạnh mới (u,v), cạnh này sẽ tạo nên với đường đi sơ cấp duy nhất nối u và v một chu trình duy nhất.

**6)⇒1)** Nếu T không liên thông thì thêm một cạnh nối hai đỉnh ở hai thành phần liên thông khác nhau ta không nhận được một chu trình nào. Vậy T liên thông, do đó nó là một cây.

**6.2. CÂY KHUNG VÀ BÀI TOÁN TÌM CÂY KHUNG NHỎ NHẤT.**

**6.2.1. Định nghĩa:** Trong đồ thị liên thông G, nếu ta loại bỏ cạnh nằm trên chu trình nào đó thì ta sẽ được đồ thị vẫn là liên thông. Nếu cứ loại bỏ các cạnh ở các chu trình khác cho đến khi nào đồ thị không còn chu trình (vẫn liên thông) thì ta thu được một cây nối các đỉnh của G. Cây đó gọi là cây khung hay cây bao trùm của đồ thị G.

Tổng quát, nếu G là đồ thị có n đỉnh, m cạnh và k thành phần liên thông thì áp dụng thủ tục vừa mô tả đối với mỗi thành phần liên thông của G, ta thu được đồ thị gọi là rừng khung của G. Số cạnh bị loại bỏ trong thủ tục này bằng m−n+k, số này ký hiệu là ν(G) và gọi là chu số của đồ thị G.

**6.2.2. Bài toán tìm cây khung nhỏ nhất:** Bài toán tìm cây khung nhỏ nhất của đồ thị là một trong số những bài toán tối ưu trên đồ thị tìm được ứng dụng trong nhiều lĩnh vực khác nhau của đời sống. Trong phần này ta sẽ có hai thuật toán cơ bản để giải bài toán này. Trước hết, nội dung của bài toán được phát biểu như sau.

Cho G=(V,E) là đồ thị vô hướng liên thông có trọng số, mỗi cạnh e∈E có trọng số m(e)≥0. Giả sử T=(VT,ET) là cây khung của đồ thị G (VT=V). Ta gọi độ dài m(T) của cây khung T là tổng trọng số của các cạnh của nó:

m(T)=.

Bài toán đặt ra là trong số tất cả các cây khung của đồ thị G, hãy tìm cây khung có độ dài nhỏ nhất. Cây khung như vậy được gọi là cây khung nhỏ nhất của đồ thị và bài toán đặt ra được gọi là bài toán tìm cây khung nhỏ nhất.

Để minh hoạ cho những ứng dụng của bài toán cây khung nhỏ nhất, dưới đây là hai mô hình thực tế tiêu biểu cho nó.

*Bài toán xây dựng hệ thống đường sắt:* Giả sử ta muốn xây dựng một hệ thống đường sắt nối n thành phố sao cho hành khách có thể đi từ bất cứ một thành phố nào đến bất kỳ một trong số các thành phố còn lại. Mặt khác, trên quan điểm kinh tế đòi hỏi là chi phí về xây dựng hệ thống đường phải là nhỏ nhất. Rõ ràng là đồ thị mà đỉnh là các thành phố còn các cạnh là các tuyến đường sắt nối các thành phố tương ứng, với phương án xây dựng tối ưu phải là cây. Vì vậy, bài toán đặt ra dẫn về bài toán tìm cây khung nhỏ nhất trên đồ thị đầy đủ n đỉnh, mỗi đỉnh tương ứng với một thành phố với độ dài trên các cạnh chính là chi phí xây dựng hệ thống đường sắt nối hai thành phố.

*Bài toán nối mạng máy tính:* Cần nối mạng một hệ thống gồm n máy tính đánh số từ 1 đến n. Biết chi phí nối máy i với máy j là m(i,j) (thông thường chi phí này phụ thuộc vào độ dài cáp nối cần sử dụng). Hãy tìm cách nối mạng sao cho tổng chi phí là nhỏ nhất. Bài toán này cũng dẫn về bài toán tìm cây khung nhỏ nhất.

Bài toán tìm cây khung nhỏ nhất đã có những thuật toán rất hiệu quả để giải chúng. Ta sẽ xét hai trong số những thuật toán như vậy: thuật toán Kruskal và thuật toán Prim.

**6.2.3. Thuật toán Kruskal:**Thuật toán sẽ xây dựng tập cạnh ET của cây khung nhỏ nhất T=(VT, ET) theo từng bước. Trước hết sắp xếp các cạnh của đồ thị G theo thứ tự không giảm của trọng số. Bắt đầu từ ET=∅, ở mỗi bước ta sẽ lần lượt duyệt trong danh sách cạnh đã sắp xếp, từ cạnh có độ dài nhỏ đến cạnh có độ dài lớn hơn, để tìm ra cạnh mà việc bổ sung nó vào tập ET không tạo thành chu trình trong tập này. Thuật toán sẽ kết thúc khi ta thu được tập ET gồm n−1 cạnh. Cụ thể có thể mô tả như sau:

**1.** Bắt đầu từ đồ thị rỗng T có n đỉnh.

**2.** Sắp xếp các cạnh của G theo thứ tự không giảm của trọng số.

**3.** Bắt đầu từ cạnh đầu tiên của dãy này, ta cứ thêm dần các cạnh của dãy đã được xếp vào T theo nguyên tắc cạnh thêm vào không được tạo thành chu trình trong T.

**4.** Lặp lại Bước 3 cho đến khi nào số cạnh trong T bằng n−1, ta thu được cây khung nhỏ nhất cần tìm.

**Thí dụ 2:** Tìm cây khung nhỏ nhất của đồ thị cho trong hình dưới đây:

Bắt đầu từ đồ thị rỗng T có 6 đỉnh.

Sắp xếp các cạnh của đồ thị theo thứ tự không giảm của trọng số:

{(v3, v5), (v4, v6), (v4, v5), (v5, v6), (v3, v4), (v1, v3), (v2, v3), (v2, v4), (v1, v2)}.

Thêm vào đồ thị T cạnh (v3, v5).

Do số cạnh của T là 1<6−1 nên tiếp tục thêm cạnh (v4, v6) vào T. Bây giờ số cạnh của T đã là 2 vẫn còn nhỏ hơn 6, ta tiếp tục thêm cạnh tiếp theo trong dãy đã sắp xếp vào T. Sau khi thêm cạnh (v4, v5) vào T, nếu thêm cạnh (v5, v6) thì nó sẽ tạo thành với 2 cạnh (v4, v5), (v4, v6) đã có trong T một chu trình. Tình huống tương tự cũng xãy ra đối với cạnh (v3, v4) là cạnh tiếp theo trong dãy. Tiếp theo ta bổ sung cạnh (v1, v3), (v2, v3) vào T và thu dược tập ET gồm 5 cạnh:

{(v3, v5), (v4, v6), (v4, v5), (v1, v3), (v2, v3)}.

**Tính đúng đắn của thuật toán:** Rõ ràng đồ thị thu được theo thuật toán có n−1 cạnh và không có chu trình. Vì vậy theo Định lý 6.1.3, nó là cây khung của đồ thị G. Như vậy chỉ còn phải chỉ ra rằng T có độ dài nhỏ nhất. Giả sử tồn tại cây khung S của đồ thị mà m(S)<m(T). Ký hiệu ek là cạnh đầu tiên trong dãy các cạnh của T xây dựng theo thuật toán vừa mô tả không thuộc S. Khi đó đồ thị con của G sinh bởi cây S được bổ sung cạnh ek sẽ chứa một chu trình duy nhất C đi qua ek. Do chu trình C phải chứa cạnh e thuộc S nhưng không thuộc T nên đồ thị con thu được từ S bằng cách thay cạnh e của nó bởi ek, ký hiệu đồ thị này là S’, sẽ là cây khung. Theo cách xây dựng, m(ek)≤m(e), do đó m(S’)≤m(S), đồng thời số cạnh chung của S’ và T đã tăng thêm một so với số cạnh chung của S và T. Lặp lại quá trình trên từng bước một, ta có thể biến đổi S thành T và trong mỗi bước tổng độ dài không tăng, tức là m(T)≤m(S). Mâu thuẩn này chứng tỏ T là cây khung nhỏ nhất của G.

Độ phức tạp của thuật toán Kruskal được đánh giá như sau. Trước tiên, ta sắp xếp các cạnh của G theo thứ tự có chiều dài tăng dần; việc sắp xếp này có độ phức tạp O(p2), với p là số cạnh của G. Người ta chứng minh được rằng việc chọn ei+1 không tạo nên chu trình với i cạnh đã chọn trước đó có độ phức tạp là O(n2). Do p≤n(n−1)/2, thuật toán

Kruskal có độ phức tạp là O(p2).

**6.2.4. Thuật toán Prim:** Thuật toán Kruskal làm việc kém hiệu quả đối với những đồ thị dày (đồ thị có số cạnh m ≈ n(n−1)/2). Trong trường hợp đó, thuật toán Prim tỏ ra hiệu quả hơn. Thuật toán Prim còn được gọi là phương pháp lân cận gần nhất.

**1.** VT:={v\*}, trong đó v\* là đỉnh tuỳ ý của đồ thị G.

ET:=∅.

**2.** Với mỗi đỉnh vj∉VT, tìm đỉnh wj∈VT sao cho

m(wj,vj) = min m(xi, vj)=:βj

xi∈VT

và gán cho đỉnh vj nhãn [wj, βj]. Nếu không tìm đuợc wj như vậy (tức là khi vj không kề với bất cứ đỉnh nào trong VT) thì gán cho vj nhãn [0, ∞].

**3.** Chọn đỉnh vj\* sao cho

βj\* = min βj

vj∉VT

VT := VT ∪ {vj\*},

ET := ET ∪ {(wj\*, vj\*)}.

Nếu |VT| = n thì thuật toán dừng và (VT, ET) là cây khung nhỏ nhất.

Nếu |VT| < n thì chuyển sang Bước 4.

**4.** Đối với tất cả các đỉnh vj∉VT mà kề với vj\*, ta thay đổi nhãn của chúng như sau:

Nếu βj > m(vj\*, vj) thì đặt βj:=m(vj\*, vj) và nhãn của vj là [vj\*, βj]. Ngược lại, ta giữ nguyên nhãn của vj. Sau đó quay lại Bước 3.

**Thí dụ 3:** Tìm cây khung nhỏ nhất bằng thuật toán Prim của đồ thị gồm các đỉnh A, B, C, D, E, F, H, I được cho bởi ma trận trọng số sau.

A

B

C

D

E

F

H

I

A

B

.

C

D

E

F

H

I

Yêu cầu viết các kết quả trung gian trong từng bước lặp, kết quả cuối cùng cần đưa ra tập cạnh và độ dài của cây khung nhỏ nhất.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| V.lặp | A | B | C | D | E | F | H | I | **V­T** | **ET** |
| K.tạo | − | **[**A,15**]** | [A,16] | [A,19] | [A,23] | [A,20] | [A,32] | [A,18] | A | ∅ |
| 1 | − | − | [A,16] | [B,13] | [A,23] | [B,19] | [B,20] | **[**B,12**]** | A, B | (A,B) |
| 2 | − | − | [A,16] | **[**I,11**]** | [I,21] | [I,18] | [I,14] | − | A, B, I | (A,B), (B,I) |
| 3 | − | − | **[**D,13**]** | − | [I,21] | [I,18] | [I,14] | − | A, B, I, D | (A,B), (B,I), (I,D) |
| 4 | − | − | − | − | [I,21] | [I,18] | **[**I,14**]** | − | A, B, I, D, C | (A,B), (B,I), (I,D), (D,C) |
| 5 | − | − | − | − | [I,21] | **[**H,17**]** | − | − | A, B, I, D, C, H | (A,B), (B,I), (I,D), (D,C), (I,H) |
| 6 | − | − | − | − | **[**I,21**]** | − | − | − | A, B, I, D, C, H, F | (A,B), (B,I), (I,D), (D,C), (I,H), (H,F) |
| 7 | − | − | − | − | − | − | − | − | A, B, I, D, C, H, F, E | (A,B), (B,I), (I,D), (D,C), (I,H), (H,F), (I,E) |

Vậy độ dài cây khung nhỏ nhất là:

15 + 12 + 11 + 13 + 14 + 17 + 21 = 103.

**Tính đúng đắn của thuật toán:** Để chứng minh thuật toán Prim là đúng, ta chứng minh bằng quy nạp rằng T(k) (k=1, 2, ...,n), đồ thị nhận được trong vòng lặp thứ k, là một đồ thị con của cây khung nhỏ nhất của G, do đó T(n) chính là một cây khung nhỏ nhất của G.

T(1) chỉ gồm đỉnh v\* của G, do đó T(1) là đồ thị con của mọi cây khung của G. Giả sử T(i) (1≤i<n) là một đồ thị con của một cây khung nhỏ nhất của G. Ta chứng minh rằng T(i+1) cũng là đồ thị con của một cây khung nhỏ nhất.

Thật vậy, theo thuật toán Prim ET(i+1)=ET(i) ∪ {ei+1}, với ei+1 là cạnh ngắn nhất trong tất cả các cạnh có một đầu mút thuộc VT(i), đầu mút kia không thuộc VT(i).

Nếu ei+1 là một cạnh của T thì Ti+1 là đồ thị con của T.

Nếu ei+1 không phải là một cạnh của T thì Ti+1 là đồ thị con T’=(VT, ET∪{ei+1}). Đồ thị T’ chứa một chu trình sơ cấp duy nhất C (theo tính chất 6 của định lý về cây). Ta chọn trong C một cạnh ej có một đỉnh thuộc T(i) và đỉnh kia không thuộc T(i) và ej≠ei+1. Ta bỏ ej trong C. Khi đó

T’’=(VT, ET’ \ {ej})

là một cây khung của G và T(i+1) là đồ thị con của T’ nên cũng là đồ thị con của T’’. Theo cách chọn ei+1 của thuật toán Prim, ta có

m(ei+1) ≤ m(ej) do đó m(T’’) ≤ m(T).

Nhưng T’’ là một cây khung của G, còn T là cây khung nhỏ nhất, vì vậy phải có m(T’’)=m(T), tức là T’’ cũng là cây khung nhỏ nhất của G.

Độ phức tạp của thuật toán Prim là O(n3). Thật vậy, nếu T(k) có k đỉnh thì có n−k đỉnh không thuộc T(k), do đó ta phải chọn chiều dài nhỏ nhất của nhiều nhất là k(n−k) cạnh. Do k(n−k) < (n−1)2, nên độ phức tạp của bước chọn ek+1 là O(n2). Vì phải chọn n−1 cạnh, nên độ phức tạp của thuật toán Prim là O(n3).

**6.3. CÂY CÓ GỐC.**

**6.3.1. Định nghĩa:** Cây có hướng là đồ thị có hướng mà đồ thị vô hướng nền của nó là một cây.

Cây có gốc là một cây có hướng, trong đó có một đỉnh đặc biệt, gọi là gốc, từ gốc có đường đi đến mọi đỉnh khác của cây.

**Thí dụ 4:**

Trong cây có gốc thì gốc r có bậc vào bằng 0, còn tất cả các đỉnh khác đều có bậc vào bằng 1.

Một cây có gốc thường được vẽ với gốc r ở trên cùng và cây phát triển từ trên xuống, gốc r gọi là đỉnh mức 0. Các đỉnh kề với r được xếp ở phía dưới và gọi là đỉnh mức 1. Đỉnh ngay dưới đỉnh mức 1 là đỉnh mức 2, ...

Tổng quát, trong một cây có gốc thì v là đỉnh mức k khi và chỉ khi đường đi từ r đến v có độ dài bằng k.

Mức lớn nhất của một đỉnh bất kỳ trong cây gọi là chiều cao của cây.

Cây có gốc ở hình trên thường được vẽ như trong hình dưới đây để làm rõ mức của các đỉnh.

Trong cây có gốc, mọi cung đều có hướng từ trên xuống, vì vậy vẽ mũi tên để chỉ hướng đi là không cần thiết; do đó, người ta thường vẽ các cây có gốc như là cây nền của nó.

**6.3.2. Định nghĩa:** Cho cây T có gốc r=v0. Giả sử v0, v1, ..., vn-1, vn là một đường đi trong T. Ta gọi:

− vi+1 là con của vi và vi là cha của vi+1.

− v0, v1, ..., vn-1 là các tổ tiên của vn và vn là dòng dõi của v0, v1, ..., vn-1.

− Đỉnh treo vn là đỉnh không có con; đỉnh treo cũng gọi là lá hay đỉnh ngoài; một đỉnh không phải lá là một đỉnh trong.

**6.3.3. Định nghĩa:** Một cây có gốc T được gọi là cây m-phân nếu mỗi đỉnh của T có nhiều nhất là m con. Với m=2, ta có một cây nhị phân.

Trong một cây nhị phân, mỗi con được chỉ rõ là con bên trái hay con bên phải; con bên trái (t.ư. phải) được vẽ phía dưới và bên trái (t.ư. phải) của cha.

Cây có gốc T được gọi là một cây m-phân đầy đủ nếu mỗi đỉnh trong của T đều có m con.

**6.3.4. Mệnh đề:** Một cây m-phân đầy đủ có i đỉnh trong thì có mi+1 đỉnh và có (m−1)i+1 lá.

**Chứng minh:** Mọi đỉnh trong của cây m-phân đầy đủ đều có bậc ra là m, còn lá có bậc ra là 0, vậy số cung của cây này là mi và do đó số đỉnh của cây là mi+1. Gọi *l* là số lá thì ta có *l*+i=mi+1, nên *l*=(m−1)i+1.

**6.3.5. Mệnh đề:** **1)** Một cây m-phân có chiều cao h thì có nhiều nhất là mh lá.

**2)** Một cây m-phân có *l* lá thì có chiều cao h ≥ [logm*l*].

**Chứng minh:** **1)** Mệnh đề được chứng minh bằng quy nạp theo h. Mệnh đề hiển nhiên đúng khi h=1. Giả sử mọi cây có chiều cao k ≤ h−1 đều có nhiều nhất mk-1 lá (với h≥2). Xét cây T có chiều cao h. Bỏ gốc khỏi cây ta được một rừng gồm không quá m cây con, mỗi cây con này có chiều cao ≤ h−1. Do giả thiết quy nạp, mỗi cây con này có nhiều nhất là mh-1 lá. Do lá của những cây con này cũng là lá của T, nên T có nhiều nhất là m.mh-1=mh lá.

**2)** *l* ≤ mh ⇔ h ≥ [logm*l*].

**6.4. DUYỆT CÂY NHỊ PHÂN.**

**6.4.1. Định nghĩa:** Trong nhiều trường hợp, ta cần phải “điểm danh” hay “thăm” một cách có hệ thống mọi đỉnh của một cây nhị phân, mỗi đỉnh chỉ một lần. Ta gọi đó là việc duyệt cây nhị phân hay đọc cây nhị phân.

Có nhiều thuật toán duyệt cây nhị phân, các thuật toán đó khác nhau chủ yếu ở thứ tự thăm các đỉnh.

Cây nhị phân T có gốc r được ký hiệu là T(r). Giả sử r có con bên trái là u, con bên phải là v. Cây có gốc u và các đỉnh khác là mọi dòng dõi của u trong T gọi là cây con bên trái của T, ký hiệu T(u). Tương tự, ta có cây con bên phải T(v) của T. Một cây T(r) có thể không có cây con bên trái hay bên phải.

Sau đây là ba trong các thuật toán duyệt cây nhị phân T(r). Các thuật toán đều được trình bày đệ quy. Chú ý rằng khi cây T(x) chỉ là môt đỉnh x thì “duyệt T(x)” có nghĩa là “thăm đỉnh x”.

**Thí dụ 5:**

**6.4.2. Các thuật toán duyệt cây nhị phân:**

**1) Thuật toán tiền thứ tự:**

**1.** Thăm gốc r.

**2.** Duyệt cây con bên trái của T(r) theo tiền thứ tự.

**3.** Duyệt cây con bên phải của T(r) theo tiền thứ tự.

Duyệt cây nhị phân T(a) trong hình trên theo tiền thứ tự:

1. Thăm a

2. Duyệt T(b)

2.1. Thăm b

2.2. Duyệt T(d)

2.2.1. Thăm d

2.2.2. Duyệt T(g)

2.2.2.1. Thăm g

2.2.2.3. Duyệt T(*l*): Thăm *l*

2.2.3. Duyệt T(h): Thăm h

2.3. Duyệt T(e)

2.3.1. Thăm e

2.3.2. Duyệt T(i)

2.3.2.1. Thăm i

2.3.2.2. Duyệt T(m): Thăm m

2.3.2.3. Duyệt T(n): Thăm n

3. Duyệt T(c)

3.1. Thăm c

3.3. Duyệt T(f)

3.3.1.Thăm f

3.3.2. Duyệt T(j)

3.3.2.1. Thăm j

3.3.2.2. Duyệt T(o): Thăm o

3.3.2.3. Duyệt T(p): Thăm p

3.3.3. Duyệt T(k)

3.3.3.1. Thăm k

3.3.3.2. Duyệt T(q): Thăm q

3.3.3.3. Duyệt T(s): Thăm s

Kết quả duyệt cây T(a) theo tiền thứ tự là:

a, b, d, g, *l*, h, e, i, m, n, c, f, j, o, p, k, q, s.

**2) Thuật toán trung thứ tự:**

**1.** Duyệt cây con bên trái của T(r) theo trung thứ tự.

**2.** Thăm gốc r.

**3.** Duyệt cây con bên phải của T(r) theo trung thứ tự.

Duyệt cây nhị phân T(a) trong hình trên theo trung thứ tự:

1. Duyệt T(b)

1.1. Duyệt T(d)

1.1.1. Duyệt T(g)

1.1.1.2. Thăm g

1.1.1.3. Duyệt T(*l*): thăm *l*

1.1.2. Thăm d

1.1.3. Duyệt T(h): Thăm h

1.2. Thăm b

1.3. Duyệt T(e)

1.3.1. Duyệt T(i)

1.3.1.1. Duyệt T(m): Thăm m

1.3.1.2. Thăm i

1.3.1.3. Duyệt T(n): Thăm n

1.3.2. Thăm e

2. Thăm a

3. Duyệt T(c)

3.2. Thăm c

3.3. Duyệt T(f)

3.3.1. Duyệt T(j)

3.3.1.1. Duyệt T(o): Thăm o

3.3.1.2. Thăm j

3.3.1.3. Duyệt T(p): Thăm p

3.3.2. Thăm f

3.3.3. Duyệt T(k)

3.3.3.1. Duyệt T(q): Thăm q

3.3.3.2. Thăm k

3.3.3.3. Duyệt T(s): Thăm s

Kết quả duyệt cây T(a) theo trung thứ tự là:

g, *l*, d, h, b, m, i, n, e, a, c, o, j, p, f, q, k, s.

**3) Thuật toán hậu thứ tự:**

**1.** Duyệt cây con bên trái của T(r) theo hậu thứ tự.

**2.** Duyệt cây con bên phải của T(r) theo hậu thứ tự.

**3.** Thăm gốc r.

Duyệt cây nhị phân T(a) trong hình trên theo hậu thứ tự:

1. Duyệt T(b)

1.1. Duyệt T(d)

1.1.1. Duyệt T(g)

1.1.1.2. Duyệt T(*l*): thăm *l*

1.1.1.3. Thăm g

1.1.2. Duyệt T(h): thăm h

1.1.3. Thăm d

1.2. Duyệt T(e)

1.2.1. Duyệt T(i)

1.2.1.1. Duyệt T(m): Thăm m

1.2.1.2. Duyệt T(n): Thăm n

1.2.1.3. Thăm i

1.2.3. Thăm e

1.3. Thăm b

2. Duyệt T(c)

2.2. Duyệt T(f)

2.2.1. Duyệt T(j)

2.2.1.1. Duyệt T(o): Thăm o

2.2.1.2. Duyệt T(p): Thăm p

2.2.1.3. Thăm j

2.2.2. Duyệt T(k)

2.2.2.1. Duyệt T(q): Thăm q

2.2.2.2. Duyệt T(s): Thăm s

2.2.2.3. Thăm k

2.2.3. Thăm f

2.3. Thăm c

3. Thăm a

Kết quả duyệt cây T(a) theo trung thứ tự là:

*l*, g, h, d, m, n, i, e, b, o, p, j, q, s, k, f, c, a.

**6.4.3. Ký pháp Ba Lan:**

Xét biểu thức đại số sau đây:

(a+b)(c−) (1)

Ta vẽ một cây nhị phân như hình dưới đây, trong đó mỗi đỉnh trong mang dấu của một phép tính trong (1), gốc của cây mang phép tính sau cùng trong (1), ở đây là dấu nhân, ký hiệu là , mỗi lá mang một số hoặc một chữ đại diện cho số.

Duyệt cây nhị phân trong hình trên theo trung thứ tự là:

a + b  c **−** d / 2 (2)

và đây là biểu thức (1) đã bỏ đi các dấu ngoặc.

Ta nói rằng biểu thức (1) được biểu diễn bằng cây nhị phân T() trong hình trên, hay cây nhị phân T() này tương ứng với biểu thức (1). Ta cũng nói: cách viết (ký pháp) quen thuộc trong đại số học như cách viết biểu thức (1) là ký pháp trung thứ tự kèm theo các dấu ngoặc.

Ta biết rằng các dấu ngoặc trong (1) là rất cần thiết, vì (2) có thể hiểu theo nhiều cách khác (1), chẳng hạn là

(a + b  c) **−** d / 2 (3)

hoặc là a + (b  c **−** d) / 2 (4)

Các biểu thức (3) và (4) có thể biểu diễn bằng cây nhị phân trong các hình sau. Hai cây nhị phân tương ứng là khác nhau, nhưng đều được duyệt theo trung thứ tự là (2).

Đối với cây trong hình thứ nhất, nếu duyệt theo tiền thứ tự, ta có

 + a b − c / d 2 (5)

và nếu duyệt theo hậu thứ tự, ta có:

a b + c d 2 / −  (6)

Có thể chứng minh được rằng (5) hoặc (6) xác định duy nhất cây nhị phân trong hình thứ nhất, do đó xác định duy nhất biểu thức (1) mà không cần dấu ngoặc. Chẳng hạn cây nhị phân trên hình thứ hai được duyệt theo tiền thứ tự là

− + a  b c / d 2 khác với (5).

và được duyệt theo hậu thứ tự là

a b c  + d 2 / − khác với (6).

Vì vậy, nếu ta viết các biểu thức trong đại số, trong lôgic bằng cách duyệt cây tương ứng theo tiền thứ tự hoặc hậu thứ tự thì ta không cần dùng các dấu ngoặc mà không sợ hiểu nhầm.

Người ta gọi cách viết biểu thức theo tiền thứ tự là ký pháp Ba Lan, còn cách viết theo hậu thứ tự là ký pháp Ba Lan đảo, để ghi nhớ đóng góp của nhà toán học và lôgic học Ba Lan Lukasiewicz (1878-1956) trong vấn đề này.

Việc chuyển một biểu thức viết theo ký pháp quen thuộc (có dấu ngoặc) sang dạng ký pháp Ba Lan hay ký pháp Ba Lan đảo hoặc ngược lại, có thể thực hiện bằng cách vẽ cây nhị phân tương ứng, như đã làm đối với biểu thức (1). Nhưng thay vì vẽ cây nhị phân, ta có thể xem xét để xác định dần các công thức bộ phận của công thức đã cho. Chẳng hạn cho biểu thức viết theo ký pháp Ba Lan là

−  ↑ / − − a b  5 c 2 3 ↑ − c d 2  − − a c d / ↑ − b  3 d 3 5

Trước hết, chú ý rằng các phép toán +, −, \*, /, ↑ đều là các phép toán hai ngôi, vì vậy trong cây nhị phân tương ứng, các đỉnh mang dấu các phép toán đều là đỉnh trong và có hai con. Các chữ và số đều đặt ở lá. Theo ký pháp Ba Lan (t.ư. Ba Lan đảo) thì T a b (t.ư. a b T) có nghĩa là a T b, với T là một trong các phép toán +, −, \*, /, ↑.













**BÀI TẬP CHƯƠNG VI:**

**1.** Vẽ tất cả các cây (không đẳng cấu) có:

**a)** 4 đỉnh **b)** 5 đỉnh **c)** 6 đỉnh

**2.** Một cây có n­2 đỉnh bậc 2, n3 đỉnh bậc 3, …, nk đỉnh bậc k. Hỏi có bao nhiêu đỉnh bậc 1?

**3.** Tìm số tối đa các đỉnh của một cây m-phân có chiều cao h.

**4.** Có thể tìm được một cây có 8 đỉnh và thoả điều kiện dưới đây hay không? Nếu có, vẽ cây đó ra, nếu không, giải thích tại sao:

**a)** Mọi đỉnh đều có bậc 1.

**b)** Mọi đỉnh đều có bậc 2.

**c)** Có 6 đỉnh bậc 2 và 2 đỉnh bậc 1.

**d)** Có đỉnh bậc 7 và 7 đỉnh bậc 1.

**5.** Chứng minh hoặc bác bỏ các mệnh đề sau đây.

**a)** Trong một cây, đỉnh nào cũng là đỉnh cắt.

**b)** Một cây có số đỉnh không nhỏ hơn 3 thì có nhiều đỉnh cắt hơn là cầu.

**6.** Có bốn đội bóng đá A, B, C, D lọt vào vòng bán kết trong giải các đội mạnh khu vực. Có mấy dự đoán xếp hạng như sau:

**a)** Đội B vô địch, đội D nhì.

**b)** Đội B nhì, đội C ba.

**c)** Đọi A nhì, đội C tư.

Biết rằng mỗi dự đoán trên đúng về một đội. Hãy cho biết kết quả xếp hạng của các đội.

**7.** Cây *Fibonacci* có gốc Tn đuợc dịnh nghĩa bằng hồi quy như sau. T1 và T2 đều là cây có gốc chỉ gồm một đỉnh và với n=3,4, … cây có gốc Tn được xây dựng từ gốc với Tn-1 như là cây con bên trái và Tn-2 như là cây con bên phải.

**a)** Hãy vẽ 7 cây *Fibonacci* có gốc đầu tiên.

**b)** Cây *Fibonacci* T­n có bao nhiêu đỉnh, lá và bao nhiêu đỉnh trong. Chiều cao của nó bằng bao nhiêu?

**8.** Hãy tìm cây khung của đồ thị sau bằng cách xoá đi các cạnh trong các chu trình đơn.

**a)**

**b)**

**9.** Hãy tìm cây khung cho mỗi đồ thị sau.

**a)** K5 **b)** K4,4 **c)** K1,6

**d)** Q3 **e)** C5 **f)** W­­5.

**10.** Đồ thị Kn với n=3, 4, 5 có bao nhiêu cây khung không đẳng cấu?

**11.** Tìm cây khung nhỏ nhất của đồ thị sau theo thuật toán Kruskal và Prim.

**12.** Tìm cây khung nhỏ nhất bằng thuật toán Prim của đồ thị gồm các đỉnh A, B, C, D, E, F, H, I được cho bởi ma trận trọng số sau.

F

G

E

D

C

B

H

A

.

H

G

F

E

D

C

A

B

Yêu cầu viết các kết quả trung gian trong từng bước lặp, kết quả cuối cùng cần đưa ra tập cạnh và độ dài của cây khung nhỏ nhất.

**13.** Duyệt các cây sau đây lần lượt bằng các thuật toán tiền thứ tự, trung thứ tự và hậu thứ tự.

**a)** **b)**

**14.** Viết các biểu thức sau đây theo ký pháp Ba Lan và ký pháp Ba Lan đảo.

**a)** .

**b)** .

**15.** Viết các biểu thức sau đây theo ký pháp quen thuộc.

**a)** x y + 2 ↑ x y − 2 ↑ − x y \* /.

**b)** −  ↑ / − − a b  3 c 2 4 ↑ − c d 5  − − a c d / ↑ − b  2 d 4 3.