CHƯƠNG 8

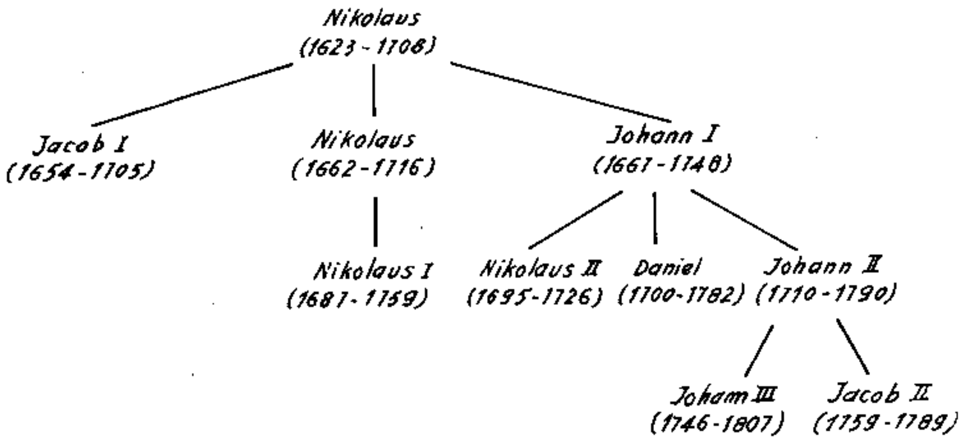
CÂY

Một đồ thị liên thông và không có chu trình đơn 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 hóa học. Từ đó cây đã được dùng để giải nhiều bài toán trong nhiều lĩnh vực khác nhau, như sẽ chỉ ra trong chương này.

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.

8.1. MỞ ĐẦU VỀ CÂY

Biểu đồ phả hệ của dòng họ Bernoulli, một gia đình toán học nổi tiếng người Thuỵ sĩ được biểu thị trên Hình 1. Biểu đồ như vậy cũng được gọi là cây phả hệ. Cây phả hệ là một đồ thị trong đó các đỉnh biểu thị các thành viên, các cạnh biểu thị mối quan hệ cha–con. Đồ thị vô hướng biểu diễn các biểu đồ phả hệ là một ví dụ về một loại đồ thị đặc biệt gọi là **cây**.

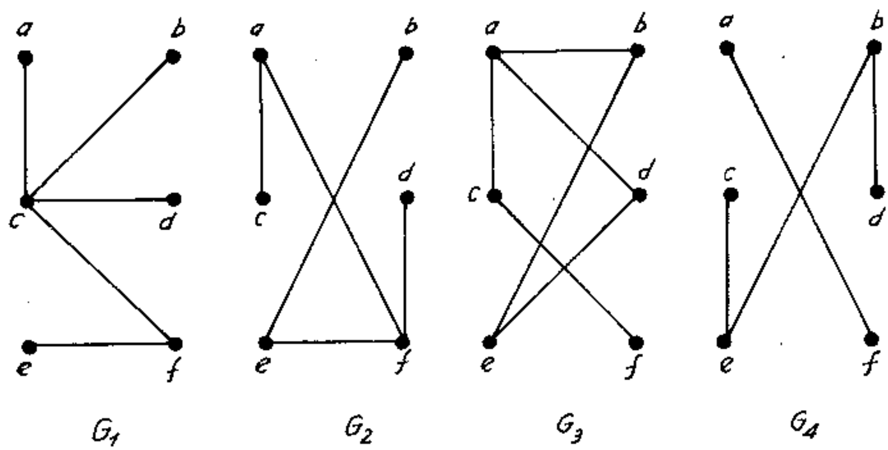


Hình 1. Phả hệ nhà toán học Bernoulli.

ĐỊNH NGHĨA 1. *Cây* là một đồ thị vô hướng, liên thông và không có chu trình đơn.

Vì cây không thể có chu trình đơn, nên cây không thể có cạnh bội và khuyên. Vậy mọi cây đều là đồ thị đơn.

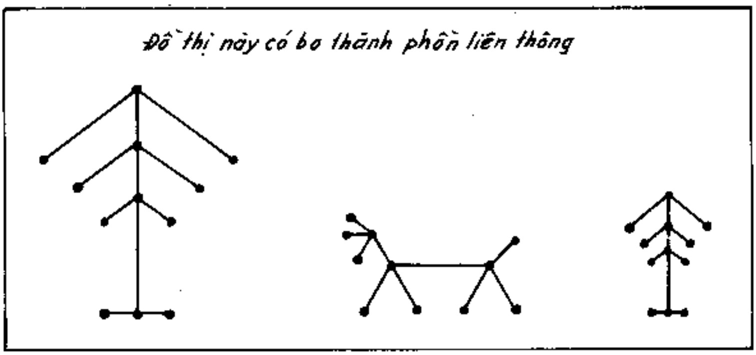
Ví dụ 1. Đồ thị nào trong các đồ thị trên Hình 2 là cây?



Hình 2. G_1 và G_2 là cây, G_3 và G_4 không là cây.

Giải: G_1, G_2 là các cây, vì chúng đều là các đơn đồ thị liên thông và không có chu trình đơn. G_3 không là cây vì *e, b, a, d, e* là một chu trình đơn của đồ thị này. Cuối cùng G_4 không là cây bởi vì nó không liên thông.

Mọi đồ thị liên thông và không có chu trình đơn đều là cây. Ta có thể nói gì về đồ thị không có chu trình đơn nhưng không liên thông? Các đồ thị như vậy gọi là **rừng**. Vậy rừng là một đồ thị mà mỗi thành phần liên thông của nó là một cây. Hình 3 biểu diễn một rừng.



Hình 3. Ví dụ về rừng.

Cây thường được định nghĩa như một đồ thị vô hướng, trong đó giữa mọi cặp đỉnh của nó luôn tồn tại đường đi đơn duy nhất. Định lý sau cho thấy định nghĩa này tương đương với định nghĩa của cây cho ở trên.

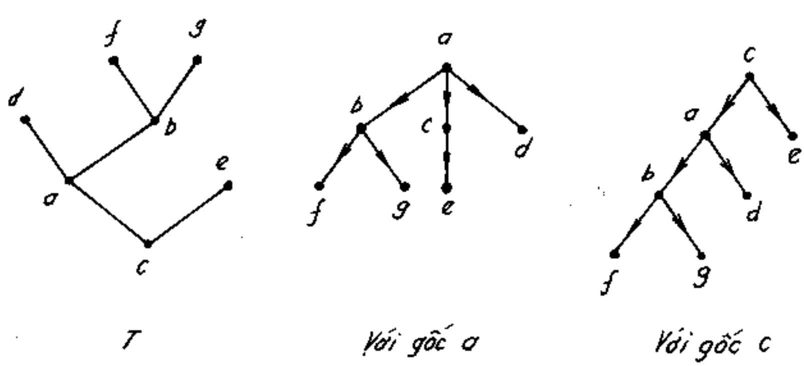
ĐỊNH LÝ 1. Một đồ thị vô hướng là một cây nếu giữa mọi cặp đỉnh của nó luôn tồn tại đường đi đơn duy nhất.

Chứng minh. Trước tiên giả sử T là một cây. Khi đó T là một đồ thị liên thông không có chu trình. Gọi x và y là hai đỉnh của T. Vì T là liên thông nên theo Định lý 1 của Tiết 7.4 có một đường đi đơn giữa hai đỉnh này. Đường đi này là duy nhất vì nếu có đường đi thứ hai từ x tới y thì đường đi tạo bởi hợp của đường đi thứ nhất từ x tới y và đường đi từ y tới x nhận được bằng cách đảo ngược đường đi thứ hai từ x tới y sẽ tạo thành một chu trình. Từ đó theo Bài tập 35 của Tiết 7.4, suy ra có chu trình đơn trong T. Vì thế giữa hai đỉnh bất kỳ của cây luôn có đường đi đơn duy nhất.

Bây giờ giả sử ngược lại, giữa hai đỉnh bất kỳ của đồ thị T luôn có đường đi đơn duy nhất. Khi đó T là liên thông. Tiếp theo, T không thể có chu trình đơn. Giả sử ngược lại là có chu trình đơn chứa hai đỉnh x

và y của T. Khi đó có hai đường đi giữa x và y, vì đường đi thứ nhất chính là phần của chu trình từ x tới y, đường thứ hai là phần còn lại của chu trình nhưng theo thứ tự ngược lại, tức là giữa x và y có hai đường đi đơn. Vì vậy, T là đồ thị liên thông không có chu trình đơn hay nó là một cây.

Trong rất nhiều ứng dụng, một đỉnh đặc biệt của cây được gọi là **gốc**. Một khi đã định rõ gốc, ta có thể gán cho mỗi cạnh một hướng như sau. Vì có đường đi duy nhất từ gốc tới mỗi đỉnh của đồ thị (Định lý 1), nên ta định hướng mỗi cạnh bằng hướng từ gốc đi ra. Như vậy cây cùng với gốc sinh ra một đồ thị có hướng gọi là **cây có gốc**. Ta có thể chuyển cây không gốc thành cây có gốc bằng cách chọn một đỉnh bất kỳ làm gốc. Lưu ý rằng việc chọn gốc khác nhau sẽ tạo ra các cây có gốc khác nhau. Ví dụ, Hình 4 biểu diễn các cây có gốc khác nhau được tạo ra từ đồ thị T bằng cách chọn a và sau đó là c làm gốc. Thường người ta vẽ cây có gốc cho gốc ở phía trên của đồ thị. Và có thể bỏ mũi tên chỉ hướng trên các cạnh của cây có gốc vì việc chọn gốc đã xác định hướng của các cạnh rồi.

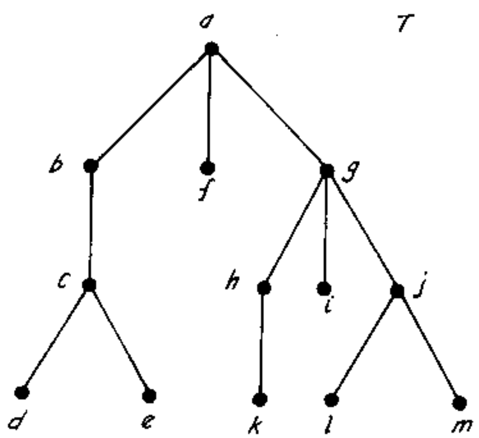


Hình 4. Cây và các cây có gốc.

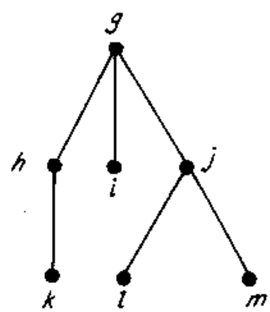
Các thuật ngữ đối với cây có nguồn gốc thực vật học hay nguồn gốc phả hệ. Giả sử T là cây có gốc. Nếu v là một đỉnh khác gốc của T, khi đó cha của v là đỉnh u duy nhất sao cho có một cạnh có hướng từ u đến v. (Độc giả chứng minh có duy nhất một đỉnh như vậy). Khi đó u được gọi là **cha** của v và v là **con** của u. Các đỉnh có cùng cha được gọi là **anh em**. **Tổ tiên** của một đỉnh khác với gốc là các đỉnh trên đường đi từ gốc tới đỉnh này (tức là, cha của nó, ông của nó, v.v., cho tới khi đến gốc). **Con cháu** của đỉnh v là các đỉnh có v như là tổ tiên. Các đỉnh của cây gọi là **lá** nếu nó không có con. Các đỉnh có con được gọi là **đỉnh trong**. Gốc là một đỉnh trong trừ khi nó là một đỉnh duy nhất của đồ thị, trong trường hợp đó nó là lá.

Nếu a là một đỉnh của một cây, thì **cây con** với gốc a là đồ thị con của cây đang xét, bao gồm a và các con cháu của nó cùng tất cả các cạnh liên thuộc với các con cháu của a.

Ví dụ 2. Trong cây T có gốc a trên Hình 5, hãy tìm cha của c, con của g, anh em của h, các tổ tiên của e, con cháu của b tất cả các đỉnh trong và các lá. Đâu là cây con với gốc tại g?



Hình 5. Cây có gốc T.



Hình 6. Cây con có gốc tại g.

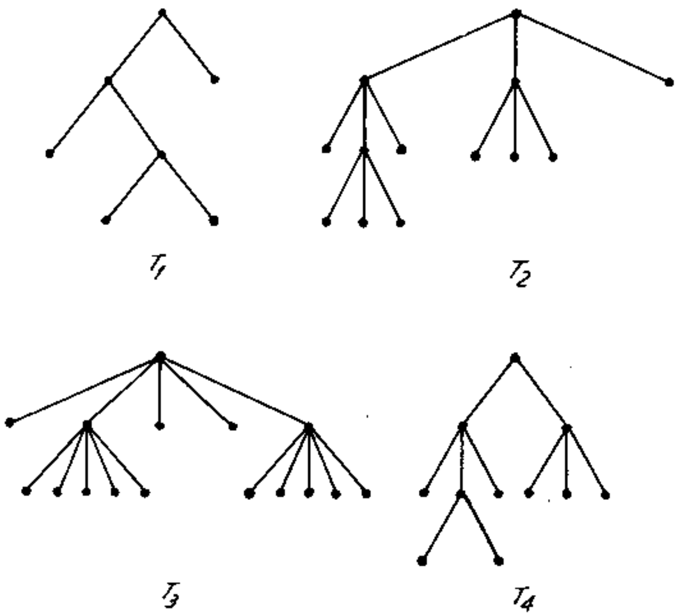
Giải: Cha của c là b. Con của g là h, i và j. Anh em của h là i và j. Tổ tiên của e là c, b và a. Con cháu của b là c, d và e. Các đỉnh trong là a, b, c, g, h và j. Các lá là d, e, f, i, k, l và m. Cây con có gốc tại g được biểu thị trên Hình 6.

■

Cây có gốc với tất cả các đỉnh trong đều có cùng số con có nhiều ứng dụng khác nhau. Dưới đây, trong chương này, chúng ta sẽ sử dụng các cây để như vậy nghiên cứu các bài toán tìm kiếm, sắp xếp và mã hóa.

ĐỊNH NGHĨA 2. Cây có gốc được gọi là *cây m-phân* nếu tất cả các đỉnh trong của nó không có hơn m con. Cây được gọi là *m-phân đầy đủ* nếu mọi đỉnh trong có đúng m con. Cây *m-phân* với $m = 2$, được gọi là cây nhị phân.

Ví dụ 3. Các cây trên Hình 7 có là cây m – phân đầy đủ với m là một số nguyên dương nào đó không?



Hình 7. Bốn cây có gốc.

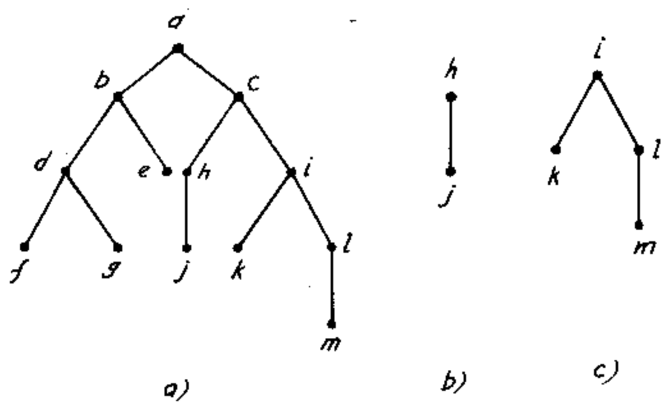
Giải: T_1 là cây nhị phân đầy đủ, vì mỗi đỉnh trong của nó đều có hai con. T_2 là cây tam phân đầy đủ vì mỗi đỉnh trong của nó có ba con. Trong T_3 mỗi đỉnh trong đều có 5 con, vì thế mà T_3 được gọi là cây ngũ phân đầy đủ. T_4 không là cây m–phân đầy đủ với một m nào đó vì một số đỉnh trong có hai con một số đỉnh khác lại có ba con.

■

Cây có gốc được sắp (hay **có thứ tự**) là cây có gốc trong đó các con của mỗi đỉnh trong được sắp xếp theo một thứ tự nhất định. Cây có gốc được sắp được vẽ sao cho các con của mỗi đỉnh trong được sắp từ trái qua phải. Nhớ rằng biểu diễn của cây có gốc theo cách truyền thống xác định thứ tự của các cạnh của nó. Chúng ta sẽ dùng thứ tự các cạnh như thế trong hình vẽ với ngầm ý là ta đang xét cây có gốc được sắp.

Trong cây nhị phân có thứ tự, các đỉnh trong có hai con, con thứ nhất gọi là **con bên trái** và con thứ hai là **con bên phải**. Cây có gốc tại con bên trái của một đỉnh gọi là cây con bên trái của đỉnh này, và cây có gốc tại con bên phải của một đỉnh gọi là cây con bên phải của đỉnh này. Độc giả cần chú ý rằng trong một số áp dụng, mọi đỉnh của cây nhị phân khác gốc được gọi tên là con bên trái hoặc con bên phải.

Ví dụ 4. Xác định con bên trái và con bên phải của d trong cây nhị phân T trên Hình 8a (trong đó thứ tự được suy ra từ hình vẽ). Hãy chỉ ra cây con bên trái và cây con bên phải của c.



Hình 8. Cây nhị phân T và các cây con bên trái và bên phải của đỉnh c.

Giải: Con bên trái của d là f và con bên phải của d là g. Chúng ta biểu thị cây con bên trái và cây con bên phải của đỉnh c trên Hình 8b và 8c.

∎

Cũng hoàn toàn giống như trong đồ thị, không có những thuật ngữ chuẩn để mô tả cây, cây có gốc, cây có gốc có thứ tự, cây nhị phân. Sở dĩ không có các thuật ngữ chuẩn vì cây được dùng rất rộng rãi trong tin học, một ngành khoa học tương đối trẻ. Độc giả hãy kiểm tra cẩn thận ý nghĩa của các từ liên quan tới cây mỗi khi chúng xuất hiện.

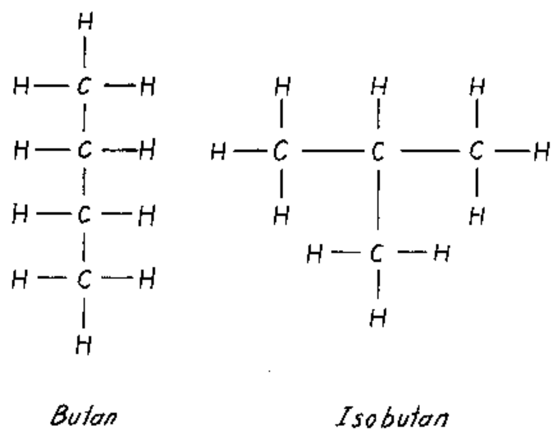
CÂY NHƯ LÀ CÁC MÔ HÌNH

Cây được dùng để mô hình các bài toán trong nhiều lĩnh vực khác nhau như tin học, thực vật học và tâm lý học. Chúng ta sẽ mô tả một số mô hình khác nhau có dùng cây.

Ví dụ 5. *Hydrocarbon no và Cây.* Đồ thị có thể dùng để biểu diễn một phân tử, trong đó nguyên tử được biểu thị bởi các đỉnh, các liên kết giữa chúng bằng các cạnh. Nhà toán học Anh, Arthur Cayley đã dùng cây vào năm 1857 khi ông ta tìm cách đánh số các đồng phân của hợp chất có dạng C_nH_{2n+2}, có tên là các *hydrocarbon no.*

Trong mô hình đồ thị của hydrocarbon no, mỗi nguyên tử các bon được biểu diễn bởi một đỉnh bậc bốn và mỗi nguyên tử hydro được biểu diễn bằng một đỉnh bậc một. Có $3n + 2$ đỉnh trong biểu diễn đồ thị của hợp chất có dạng C_nH_{2n+2}. Số các cạnh trong đồ thị như thế bằng một nửa tổng các bậc của các đỉnh. Vì thế có $(4n + 2n + 2)/2 = 3n + 1$ cạnh trong đồ thị này. Vì đồ thị là liên thông và số cạnh nhỏ hơn số đỉnh một đơn vị nên nó là một cây (xem Bài tập 9 ở cuối tiết này).

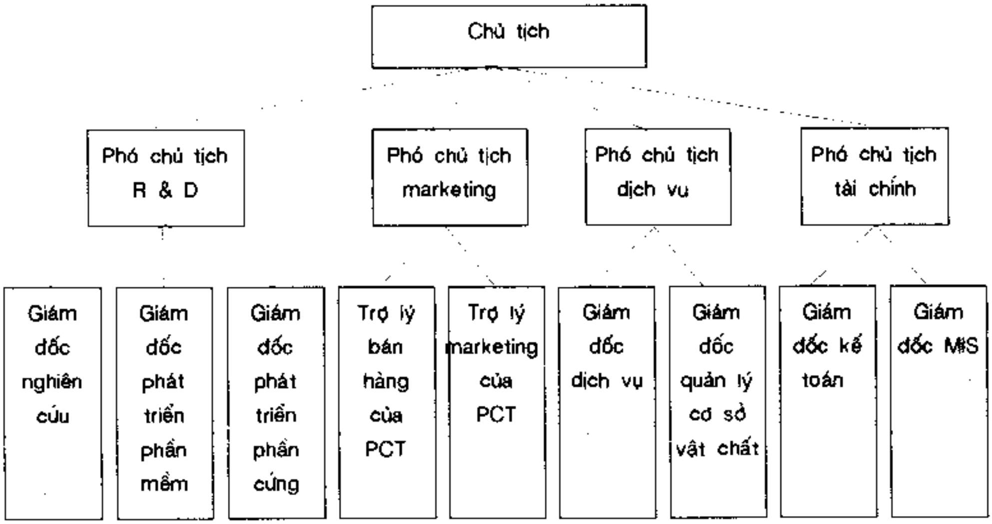
Các cây không đẳng cấu với n đỉnh bậc 4 và $2n + 2$ đỉnh bậc 1 biểu diễn các đồng phân khác nhau dạng C_nH_{2n+2}. Ví dụ khi $n = 4$, có đúng 2 cây không đẳng cấu có dạng như thế (độc giả tự kiểm tra). Vì thế có đúng hai đồng phân dạng C_4H_{10}. Cấu trúc của nó được biểu thị trên Hình 9. Các đồng phân này được gọi là butane và isobutane.



Hình 9. Hai đồng phân của Butane.

■

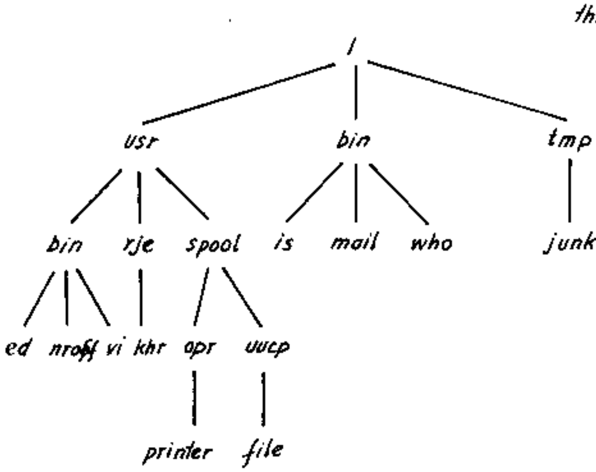
Ví dụ 6. *Biểu diễn các tổ chức.* Cấu trúc của một tổ chức lớn có thể mô hình bằng cây có gốc. Mỗi đỉnh biểu thị một chức vụ trong tổ chức này. Một cạnh từ một đỉnh tới một đỉnh khác chỉ ra rằng người biểu thị bằng đỉnh đầu là chủ (lãnh đạo) của người biểu thị bằng đỉnh cuối. Trên Hình 10 là một cây như thế.



Hình 10. Cây tổ chức của một công ty máy tính.

Ví dụ 7. *Hệ thống các tệp tin* trong máy tính có thể được tổ chức thành các thư mục. Một thư mục có thể chứa các tệp tin và các thư mục con. Thư mục gốc chứa toàn bộ hệ thống tệp tin. Như vậy hệ các tệp tin có thể biểu diễn bằng cây thư mục, trong đó gốc của cây là thư

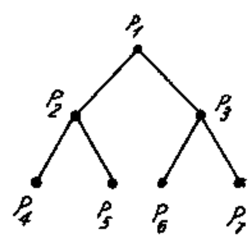
Gốc là thư mục gốc/ các đỉnh trong là các thư mục. Lá là tệp



Hình 11. Hệ các tệp tin trong máy tính.

mục gốc, các đỉnh trong là các thư mục con, và các lá là các tệp tin hay thư mục rỗng. Một hệ các tệp tin như thế được biểu diễn trên Hình 11, trong đó tệp tin *khr* ở trong thư mục *rje*.

Ví dụ 8. *Các bộ xử lý song song kết nối kiểu cây.* Trong ví dụ 13 của Tiết 7.2 chúng ta đã mô tả một số mạng liên kết để xử lý song song. **Mạng kết nối kiểu cây** là một cách quan trọng khác để nối các bộ xử lý với nhau. Đồ thị biểu diễn mạng như thế là một cây nhị phân đầy đủ. Các mạng này liên kết $n = 2^k - 1$ bộ xử lý với nhau, trong đó k là một số dương. Bộ xử lý biểu diễn bằng đỉnh v không là gốc hoặc lá có 3 liên kết hai chiều – một nối với bộ xử lý được biểu diễn bởi bố của v và hai nối với các bộ xử lý biểu thị bởi hai con của v. Bộ xử lý biểu diễn bởi gốc có hai liên kết hai chiều với các bộ xử lý biểu điễn bởi hai con của nó. Bộ xử lý biểu điễn bằng một lá v chỉ có một liên kết hai chiều với cha của v. Hình 12 giới thiệu một mạng liên kết kiểu cây với 7 bộ xử lý.



Hình 12. Mạng kết nối kiểu cây với 7 bộ xử lý.

Bây giờ ta sẽ minh họa cách dùng mạng kết nối kiểu cây để tính toán song song. Đặc biệt ta sẽ chỉ ra cách dùng mạng trên Hình 12 để cộng 8 số bằng ba bước. Trong bước đầu tiên ta dùng P_4 để cộng x_1 với x_2, dùng P_5 để cộng x_3 với x_4, dùng P_6 để cộng x_5 với x_6, dùng P_7 để cộng x_7 với x_8. Bước thứ hai dùng P_2 để cộng $x_1 + x_2$ với $x_3 + x_4$, dùng P_3 để cộng $x_5 + x_6$ với $x_7 + x_8$. Cuối cùng, dùng P_1 để cộng $x_1 + x_2 + x_3 + x_4$ với $x_5 + x_6 + x_7 + x_8$. Ba bước để cộng 8 số thật có lợi so với phải dùng bảy bước để cộng liên tiếp 8 số, trong đó mỗi bước cộng một số với tổng các số đã cộng trước đó.

■

NHỮNG TÍNH CHẤT CỦA CÂY

Chúng ta thường cần các kết quả liên quan tới số đỉnh và số cạnh của các loại cây.

ĐỊNH LÝ 2. Cây với n đỉnh có đúng $n - 1$ cạnh.

Chứng minh. Chọn đỉnh r làm gốc của cây. Ta sẽ xây dựng phép tương ứng một - một giữa các cạnh với các đỉnh khác r bằng cách gán đỉnh cuối của một cạnh với chính cạnh này. Vì có $n - 1$ đỉnh khác r nên ta có $n - 1$ cạnh.

■

Số các đỉnh trong một cây m - phân đầy đủ với một số xác định các đỉnh trong sẽ được cho trong định lý sau.

ĐỊNH LÝ 3. Cây m - phân đầy đủ với i đỉnh trong sẽ có tất cả $n = m.i + 1$ đỉnh.

Chứng minh. Mỗi đỉnh trừ gốc là con của một đỉnh trong. Vì mỗi một trong i đỉnh trong có m con nên có $m.i$ đỉnh khác gốc. Do đó cây có tất cả $n = m.i + 1$ đỉnh.

■

Giả sử T là cây m - phân đầy đủ. Gọi i là số các đỉnh trong và l là số các lá của cây này. Khi biết một trong các đại lượng n, i và l thì hai đại lượng kia cũng được xác định. Chúng ta có định lý sau.

ĐỊNH LÝ 3. Cây m - phân đầy đủ với

(i) n đỉnh có $i = \dfrac{n-1}{m}$ đỉnh trong và $l = \dfrac{(m-1)n+1}{m}$ lá,

(ii) i đỉnh trong có $n = mi + 1$ đỉnh và $l = (m - 1)i + 1$ lá,

(iii) có $n = \dfrac{ml-1}{m-1}$ đỉnh và $i = \dfrac{l-1}{m-1}$ đỉnh trong.

Chứng minh. Gọi n là số đỉnh, i số đỉnh trong và l là số lá. Ba phần của định lý này có thể chứng minh bằng cách dùng đẳng thức trong Định lý 3, tức là $n = mi + 1$, cùng với đẳng thức $n = l + i$, sở dĩ có điều này vì mỗi đỉnh hoặc là lá hoặc là đỉnh trong. Bây giờ chúng ta sẽ chứng minh (i).

Hai phần sau độc giả tự chứng minh lấy.

Giải đối với i từ đẳng thức $n = mi + 1$ ta được $i = \dfrac{n-1}{m}$. Thế biểu thức này của i vào phương trình $n = l + i$ ta có

$$l = n - i = n - \frac{n-1}{m} = \frac{(m-1)n+1}{m}.$$

■

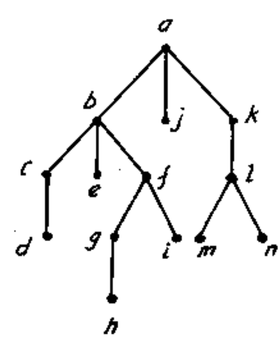
Ví dụ sau minh họa cách dùng Định lý 4.

Ví dụ 9. Giả sử ta có trò chơi viết thư dây chuyền. Ban đầu có một người nhận được một bức thư và giả sử rằng mỗi người khi nhận được một bức thư hoặc sẽ viết thư cho 4 người khác hoặc không viết thư cho ai cả. Hỏi có bao nhiêu người nhận được thư kể cả người đầu tiên nếu không có ai nhận được nhiều hơn một bức thư và trò chơi kết thúc khi có 100 người nhận thư mà không viết cho ai?

Giải: Trò chơi gửi thư này có thể biểu diễn bằng cây tứ phân. Các đỉnh trong ứng với những người gửi thư cho người khác còn lá là những người nhận thư mà không viết cho ai. Vì có 100 người không viết thư nên số lá của cây có gốc này là $l = 100$. Vì thế theo phần (iii) của Định lý 4 ta có số người nhận thư $n = (4.100 - 1)/(4 - 1) = 133$. Số các đỉnh trong là $133 - 100 = 33$, tức là có 33 người viết thư.

■

Trong ứng dụng ta thường gặp cây có gốc "cân đối". Đó là cây mà các cây con tại mỗi đỉnh có đường đi với độ dài gần như nhau. Một vài định nghĩa sẽ làm rõ hơn khái niệm này. **Mức** của đỉnh v trong cây có gốc là độ dài của đường đi duy nhất từ gốc tới nó. Mức của gốc được định nghĩa bằng không. **Độ cao** của cây là mức cao nhất của tất cả các đỉnh. Nói cách khác độ cao của cây có gốc là chiều dài của đường đi dài nhất từ gốc tới một đỉnh bất kỳ.



Hình 13. Cây có gốc.

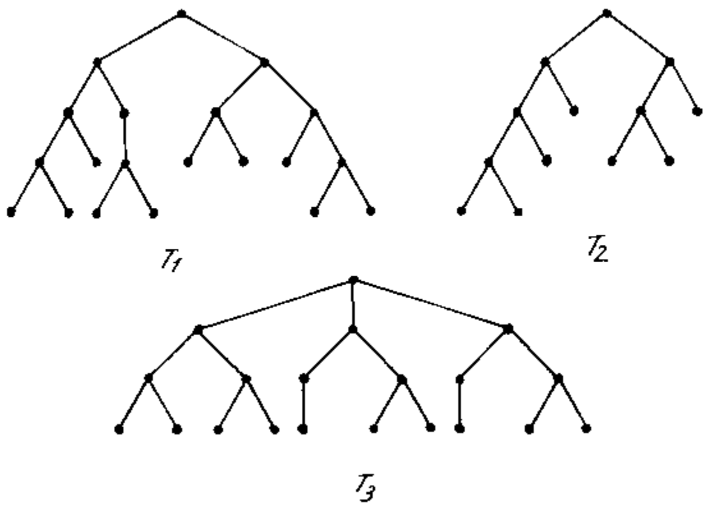
Ví dụ 10. Hãy tìm mức của mỗi đỉnh trong cây có gốc trên Hình 13. Độ cao của cây này bằng bao nhiêu?

Giải: Gốc a có mức bằng 0. Đỉnh b, j và k có mức bằng 1, các đỉnh c, e, f và l có mức bằng 2. Các đỉnh d, g, i, m và n ở mức 3. Cuối cùng, mức của đỉnh h bằng 4. Vì mức lớn nhất của tất cả các đỉnh là 4 nên độ cao của cây là 4.

■

Cây m – phân có gốc và độ cao h được gọi là **cân đối** nếu tất cả các lá đều ở mức h hoặc $h - 1$.

Ví dụ 11. Cây nào trong các cây có gốc trên Hình 14 là cân đối?



Hình 14. Một vài cây có gốc.

Giải: T_1 là cân đối vì các lá của nó đều ở mức 3 và 4. Tuy vậy, T_2 là không cân đối vì nó có các lá ở mức 2, 3 và 4. Cuối cùng, T_3 là cân đối vì tất cả các lá đều ở mức 3.

■

Những kết quả sau đây liên quan tới độ cao và số lá của cây m – phân.

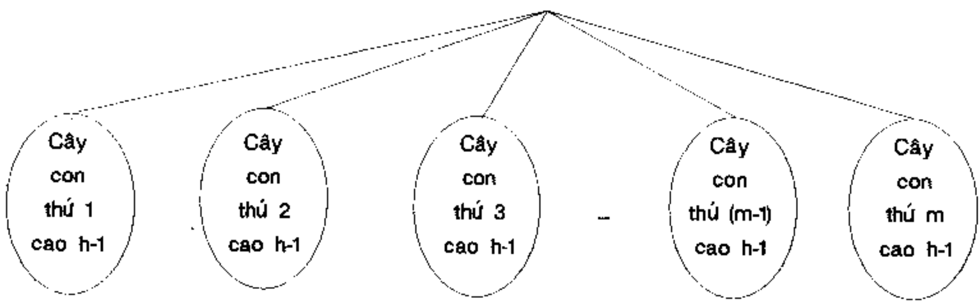
ĐỊNH LÝ 5. Có nhiều nhất m^h lá trong cây m–phân với độ cao h.

Chứng minh. Ta sẽ chứng minh bằng quy nạp theo chiều cao. Trước tiên, hãy xét cây *m-phân* có chiều cao 1. Những cây này có gốc và không quá m con, mỗi con là một lá. Vì thế có không quá $m^1 = m$ lá trong cây m–phân có chiều cao $h = 1$. Đó chính là bước cơ sở của chứng minh quy nạp.

Bây giờ ta giả sử kết quả là đúng với cây m–phân có chiều cao nhỏ hơn h. Đây là giả thiết quy nạp. Giả sử T là cây m – phân cao h. Các lá của T là các lá của các cây con nhận được từ T bằng cách xóa các cạnh nối từ gốc tới các đỉnh ở mức 1, như đã chỉ ra trên Hình 15.

Mỗi một trong các cây con này có chiều cao không quá $h - 1$. Vì vậy theo giả thiết quy nạp mỗi cây con có nhiều nhất m^{h-1} lá. Vì có nhiều nhất m cây con như thế, mỗi cây có nhiều nhất m^{h-1} lá, nên có nhiều nhất $m.m^{h-1} = m^h$ lá trong cây có gốc T. Đó là điều cần chứng minh.

■



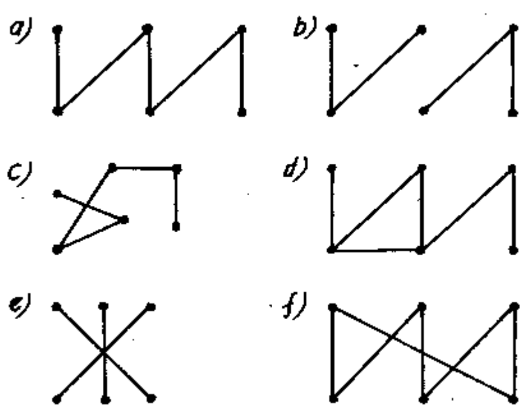
Hình 15. Bước qui nạp trong chứng minh.

HỆ QUẢ 1. Nếu cây m-phân cao h có l lá, khi đó $h \geqslant \lceil \log_m l \rceil$. Nếu cây m-phân đầy đủ và cân đối, khi đó $h = \lceil \log_m l \rceil$ (Nhớ lại là $\lceil x \rceil$ là số nguyên nhỏ nhất lớn hơn hay bằng x).

Chứng minh. Từ Định lý 5 ta có $l \leqslant m^h$. Lấy lôgarit cơ số m ta được $\log_m l \leqslant h$. Vì h là số nguyên dương nên $h \geqslant \lceil \log_m l \rceil$. Bây giờ giả sử cây là cân đối. Khi đó mọi lá đều ở mức h hoặc $h - 1$, và vì chiều cao của nó là h, nên có ít nhất một lá ở mức h. Từ đó suy ra phải có hơn m^{h-1} lá (xem Bài tập 24 ở cuối tiết). Vì $l \leqslant m^h$ ta có $m^{h-1} < l \leqslant m^h$. Lấy lôgarit cơ số m ta được $h - 1 < \log_m l \leqslant h$. Vậy $h = \lceil \log_m l \rceil$ ■

BÀI TẬP

1. Trong các đồ thị sau đồ thị nào là cây?



2. Hãy trả lời các câu hỏi sau về cây có gốc cho ở hình bên.

a) Đỉnh nào là gốc?

b) Đỉnh nào là đỉnh trong?

c) Đỉnh nào là lá?

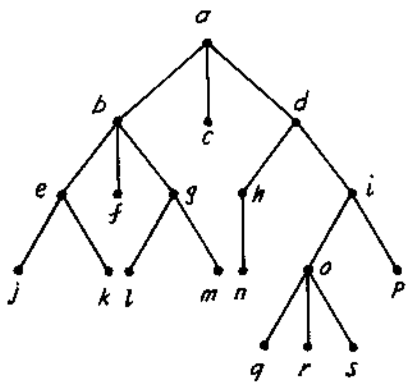
d) Đỉnh nào là con của i?

e) Đỉnh nào là cha của h?

f) Đỉnh nào là anh em của o?

g) Đỉnh nào là tổ tiên của m?

h) Đỉnh nào là con cháu của b?



3. Cây có gốc trong Bài tập 2 có là cây m-phân đầy đủ với một số dương m nào đó không?

4. Tìm mức của mỗi đỉnh trong cây ở Bài tập 2.

5. Vẽ cây con của cây trong Bài tập 2 có gốc tại

a) a. b) c. c) e.

6*. Có bao nhiêu cây không gốc không đẳng cấu với n đỉnh nếu

a) $n = 3$? b) $n = 4$? c) $n = 5$?

7*. Câu hỏi như Bài tập 6 với cây có gốc (dùng sự đẳng cấu của các đồ thị có hướng).

8*. Chỉ ra rằng một đồ thị đơn là cây nếu và chỉ nếu nó liên thông nhưng khi xóa một cạnh bất kỳ sẽ nhận được một đồ thị không liên thông.

9*. Gọi G là một đơn đồ thị với n đỉnh. Chỉ ra rằng G là cây nếu và chỉ nếu G liên thông và có $n-1$ cạnh.

10. Trong các đồ thị phân đôi đầy đủ $K_{m,n}$, với m,n nguyên dương, đồ thị nào là cây?

11. Cây với 10000 đỉnh có bao nhiêu cạnh?

12. Cây ngũ phân đầy đủ với 100 đỉnh trong có bao nhiêu đỉnh?

13. Cây nhị phân đầy đủ với 1000 đỉnh trong có bao nhiêu cạnh?

14. Cây tam phân đầy đủ với 100 đỉnh có bao nhiêu lá?

15. Giả sử có 1000 người tham gia vào một cuộc đấu cờ. Hãy dùng cây lập mô hình cuộc thi đấu và xác định xem có bao nhiêu trận đấu xảy ra để chọn người giữ chức vô địch, nếu một vận động viên sẽ bị loại sau một trận thua và các trận đấu được tiến hành cho tới khi chỉ có một người không thua. (Giả sử không có trận hòa).

16. Trò chơi gửi thư dây chuyền bắt đầu khi một người nhận được một bức thư và giả sử rằng mỗi người khi nhận được một bức thư hoặc sẽ viết thư cho 5 người khác chưa bao giờ nhận được thư hoặc không viết thư cho ai cả. Giả sử rằng có 10 000 người gửi thư trước khi trò chơi kết thúc và không có ai nhận được nhiều hơn một bức thư. Có bao nhiêu người nhận thư và bao nhiêu người không gửi thư cho người khác?

17. Trò chơi gửi thư dây chuyền bắt đầu khi một người gửi thư cho 10 người khác. Giả sử rằng mỗi người được yêu cầu viết thư cho 10 người khác nữa và mỗi bức thư có một danh sách gồm 6 người trước đó trong dây chuyền. Trừ khi có ít hơn 6 người trong danh sách, còn mỗi người sẽ gửi một đô-la cho người đầu tiên trong danh sách và xoá tên người đó đi, dịch 5 tên còn lại mỗi tên lên một vị trí, rồi chèn tên của mình vào cuối danh sách. Nếu không có ai làm gián đoạn dây chuyền và không có ai nhận được hơn một bức thư, thì người trong dây chuyền sẽ nhận được tối hậu bao nhiêu đô la?

18*. Hoặc là vẽ cây m-phân đầy đủ với 76 lá và có chiều cao bằng 3 trong đó m là số nguyên dương hoặc chỉ ra rằng không tồn tại cây như thế.

19*. Hoặc là vẽ cây m-phân đầy đủ với 84 lá và có chiều cao bằng 3 trong đó m là số nguyên dương hoặc chỉ ra rằng không tồn tại cây như thế.

20*. Cây m-phân đầy đủ T với 81 lá và có chiều cao bằng 4 :

a) Hãy tìm cận trên và cận dưới của m.

b) Giá trị của m bằng bao nhiêu nếu T cũng là cây cân đối.

Cây m **– phân hoàn toàn** *là cây* m*-phân đầy đủ trong đó mọi lá ở cùng một mức.*

21 Xây dựng cây nhị phân hoàn toàn có chiều cao bằng 4, và cây tam phân hoàn toàn có chiều cao bằng 3.

22. Cây m–phân hoàn toàn có chiều cao bằng h có bao nhiêu đỉnh và bao nhiêu lá?

23. Hãy chứng minh :

a) phần (ii) của Định lý 4.

b) phần (iii) của Định lý 4.

24. Hãy chỉ ra rằng cây m–phân đầy đủ cân đối chiều cao h có hơn m^{h-1} lá.

25. Có bao nhiêu cạnh trong rừng với t cây và có tất cả n đỉnh?

26. Nói rõ cách dùng cây để biểu diễn bảng mục lục của một cuốn sách được tổ chức thành các chương trong đó mỗi chương được tổ chức thành các tiết, mỗi tiết được tổ chức thành các mục.

27. Các hydrocacbon sau đây có bao nhiêu đồng phân khác nhau?

a) C_3H_8 b) C_5H_{12} c) C_6H_{14}

28. Mỗi một trong các phần sau đây biểu diễn cái gì trong một cây biểu thị một tổ chức?

a) Cha của một đỉnh,

b) Con của một đỉnh,

c) Anh em của một đỉnh,

d) Tổ tiên của một đỉnh,

e) Con cháu của một đỉnh,

f) Mức của một đỉnh,

g) Chiều cao của một cây.

29. Hãy trả lời những câu hỏi như trong Bài tập 28 đối với một cây có gốc biểu diễn một hệ các tệp tin lưu trữ trên máy tính.

30. a) Hãy vẽ cây nhị phân hoàn toàn có 15 đỉnh biểu thị mạng nối kết kiểu cây có 15 bộ xử lý.

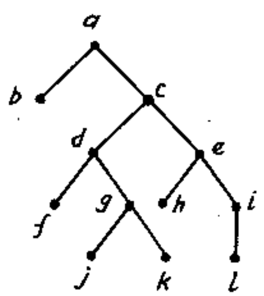
b) Hãy chứng tỏ chỉ cần 4 bước có thể cộng 16 số bằng 15 bộ xử lý được tổ chức như trong phần a).

31. Gọi n là lũy thừa của 2. Chỉ ra rằng có thể cộng n số bằng logn bước khi dùng một mạng nối kết kiểu cây gồm $n - 1$ bộ xử lý.

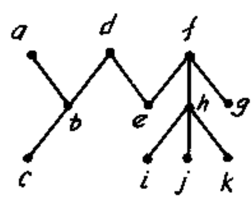
32*. **Cây có gán nhãn** là cây trong đó mỗi đỉnh được gán một nhãn. Hai cây có gán nhãn được coi là đẳng cấu nếu giữa chúng có phép đẳng cấu bảo toàn nhãn của các đỉnh. Có bao nhiêu cây không đẳng cấu có ba đỉnh được gán nhãn bằng các số nguyên khác nhau của tập hợp {1, 2, 3} ? Có bao nhiêu cây không đẳng cấu có bốn đỉnh được gán nhãn bằng các số nguyên khác nhau của tập hợp {1, 2, 3, 4}?

Tâm sai *của một đỉnh trong cây không gốc là độ dài của đường đi đơn dài nhất bắt đầu từ đỉnh này. Một đỉnh gọi là tâm nếu không có đỉnh nào trong cây có tâm sai nhỏ hơn tâm sai của đỉnh này. Trong các Bài tập từ 33 – 35 hãy tìm mọi đỉnh là tâm trong cây đã cho.*

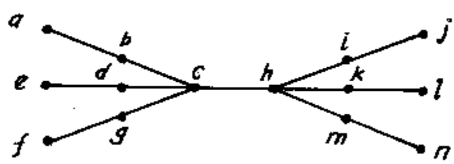
33.



34.



35.



36. Chứng minh rằng nếu chọn tâm làm gốc thì cây có gốc đó có chiều cao bé nhất so với tất cả các cây có gốc nhận được từ một cây không gốc.

37*. Chứng tỏ rằng một cây hoặc có một tâm hoặc có hai tâm liền kề nhau.

38. Chứng tỏ ràng mọi cây đều có thể tô bằng hai màu.

Cây Fibonacci có gốc T_n *được định nghĩa bằng hồi quy như sau.* T_1 *và* T_2 *đều là cây có gốc chỉ gồm một đỉnh và với* $n = 3,4, \ldots$ *cây có gốc* T_n *được xây dựng từ gốc với* T_{n-1} *như là cây con bên trái và* T_{n-2} *như là cây con bên phải.*

39. Hãy vẽ bảy cây Fibonacci có gốc đầu tiên.

40*. Cây Fibonacci có gốc T_n với n là số nguyên dương, có bao nhiêu đỉnh, lá, và bao nhiêu đỉnh trong. Chiều cao của nó bằng bao nhiêu?

8.2. CÁC ỨNG DỤNG CỦA CÂY

MỞ ĐẦU

Chúng ta sẽ nghiên cứu ba bài toán bằng mô hình cây. Bài toán đầu như sau : Các phần tử trong một danh sách được lưu trữ như thế nào để có thể dễ dàng định vị được chúng? Bài toán thứ hai : Hãy xác định dãy các quyết định để tìm một đối tượng có tính chất nào đó trong tập hợp các đối tượng thuộc một loại nào đó. Bài toán thứ ba : Cần phải mã hóa tập các chữ cái bằng các dãy nhị phân như thế nào để có hiệu quả nhất?

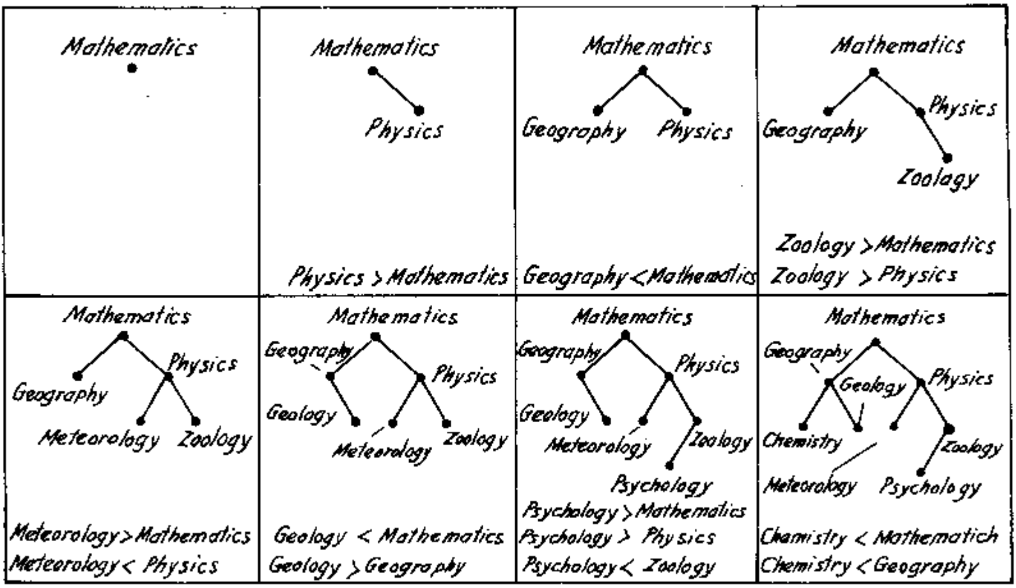
CÂY TÌM KIẾM NHỊ PHÂN

Tìm kiếm một phần tử trong một danh sách là một trong những công việc quan trọng nhất trong tin học. Mục đích hàng đầu của chúng ta là đưa ra một thuật toán tìm kiếm có hiệu quả nhất một phần tử khi các phần tử được sắp xếp theo một thứ tự nào đó. Điều đó có thể thực hiện được bằng **cây tìm kiếm nhị phân**. Đó là một cây nhị phân trong đó mỗi con của một đỉnh hoặc là con bên phải hoặc là con bên trái, không có đỉnh nào có hơn một con bên phải hay hơn một con bên trái, và mỗi đỉnh được gán một khóa (mỗi giá trị của khóa xác định chỉ một phần tử). Hơn thế nữa, các đỉnh được gán khóa sao cho khóa của đỉnh lớn hơn khóa của tất cả các đỉnh thuộc cây con bên trái, và nhỏ hơn khóa của tất cả các đỉnh thuộc cây con bên phải của nó.

Thủ tục đệ quy sau đây dùng để tạo lập cây tìm kiếm nhị phân cho một danh sách các phần tử. Bắt đầu là cây có đúng một đỉnh, tức là gốc. Phần tử đầu tiên trong danh sách được dùng làm khóa của gốc. Để thêm một phần tử mới ta so sánh nó với khóa của các đỉnh đã có trên cây, bắt đầu từ gốc và đi sang trái nếu phần tử nhỏ hơn khóa của đỉnh tương ứng nếu đỉnh này có con bên trái hoặc đi sang bên phải nếu phần tử lớn hơn khóa của đỉnh tương ứng nếu đỉnh này có con bên phải. Khi phần tử nhỏ hơn khóa của đỉnh tương ứng và đỉnh này không có con bên trái khi đó ta tạo một đỉnh mới cho phần tử này như là con bên trái của đỉnh đang xét. Phần tử được chọn làm khóa của đỉnh mới này. Tương tự khi phần tử lớn hơn khóa của đỉnh đang xét và đỉnh này không có con bên phải ta tạo một đỉnh mới cho phần tử này như là con bên phải với khóa là phần tử đang xét. Ta minh họa thủ tục này bằng ví dụ sau đây.

Ví dụ 1. Hãy tạo cây tìm kiếm nhị phân cho các từ sau : *mathematics, physics, geography, zoology, meteology, geology, psychology* và *chemistry* (dùng thứ tự từ điển).

Giải. Hình 1 cho thấy các bước xây dựng tìm kiếm nhị phân. Từ *mathematics* là khóa của gốc. Vì *physics* đi sau *mathematics* (theo thứ tự từ điển) nên ta thêm một con bên phải với khoá là *physics*. Vì *geography* đứng trước *mathematics*, ta thêm một con bên trái của gốc với khóa là *geography*. Tiếp theo ta thêm vào con bên phải của đỉnh với khóa *physics* và gán cho nó khóa là *zoology* vì *zoology* đi sau *mathematics* và *physics*. Tương tự, thêm con bên trái của đỉnh với khóa là *physics* và gán cho đỉnh mới này khóa *meteology*. Thêm vào phải của đỉnh với khóa *geography* gán cho nó khóa là *geology* Thêm vào một con bên trái của đỉnh với khóa *zoology* và gán cho nó khóa là *psychology*. Thêm vào con bên trái của đỉnh với khóa *geography* cho nó khóa là *chemistry*. (Độc giả tự kiểm tra việc so sánh tại mỗi bước).



Hình 1. Xây dựng cây tìm kiếm nhị phân.

Để định vị một phần tử ta thử thêm nó vào cây tìm kiếm nhị phân. Chúng ta sẽ định vị được nó nếu nó đã có trong cây. Thuật toán 1 dưới dạng giả mã cho phép định vị một phần tử trong cây tìm kiếm nhị phân và thêm đỉnh mới với phần tử này là khoá của nó nếu nó chưa có trong cây. Thuật toán sẽ định vị được x nếu nó là khóa của một đỉnh. Khi x không là khóa của bất cứ đỉnh nào thì một đỉnh mới với khóa x được thêm vào cây. Trong dạng giả mã đỉnh v có khóa là x và label (v) biểu thị khóa của đỉnh v.

THUẬT TOÁN 1. THUẬT TOÁN TÌM KIẾM NHỊ PHÂN.

procedure *insertion* (T : cây tìm kiếm nhị phân, x : phần tử)

v : = gốc của T

{đỉnh không có trong T sẽ có giá trị bằng *null*}

while $v \neq$ *null* và *label(v)* $\neq x$

begin

    if $x <$ label(v) **then**

if con bên trái của $v \neq$ null **then** v : = con bên trái của v

else thêm *đỉnh mới* là con trái của v và đặt v : = *null*

else

if con bên phải của $v \neq$ *null* **then** v : = con bên phải của v

else thêm *đỉnh mới* là con bên phải của v và đặt v : = *null*

end

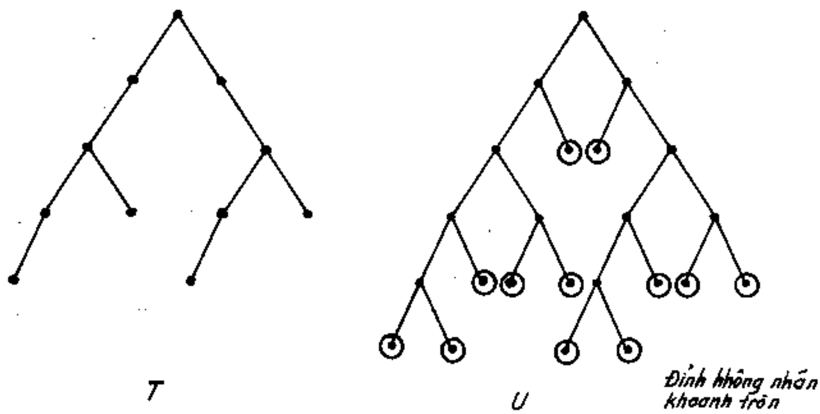
if gốc của T = *null* **then** thêm đỉnh r vào cây và gán cho nó nhãn là x

else if label (v) $\neq x$ **then** gán nhãn cho *đỉnh mới* là x.

$\{v$ = vị trí của $x\}$

Bây giờ ta sẽ xác định độ phức tạp tính toán của thủ tục này. Giả sử ta có cây tìm kiếm nhị phân T ứng với danh sách n phần tử. Ta có thể xây dựng cây nhị phân đầy đủ U từ T bằng cách thêm vào các đỉnh không có nhãn, nếu cần, sao cho mọi đỉnh có khóa đều có hai con. Điều này được minh họa trên hình 2. Làm như vậy ta có thể dễ dàng định vị hoặc thêm một phần tử mới như là khóa của đỉnh mà không phải thêm vào một đỉnh mới.

Số phép so sánh nhiều nhất cần có để thêm một phần tử mới là độ dài của đường đi dài nhất trong U từ gốc tới một lá. Các đỉnh trong của U là các đỉnh của T. Vậy U có n đỉnh trong. Bây giờ ta có thể dùng phần (ii) của Định lý 4 ở Tiết 8.1 để kết luận rằng U có $n + 1$ lá. Sử dụng Hệ quả 1 của Tiết 8.1 ta thấy chiều cao của U lớn hay bằng $h = \lceil log(n+1) \rceil$. Vậy phải thực hiện ít nhất $\lceil log(n + 1) \rceil$ phép so sánh để thêm một phần tử mới vào cây. Lưu ý rằng U là cân đối và chiều cao của nó là $\lceil log(n + 1) \rceil$ (theo Hệ quả 1 của Tiết 8.1). Như vậy nếu cây tìm kiếm nhị phân là cân đối việc định vị hay thêm một phần tử đòi hỏi không quá $\lceil log(n+1) \rceil$ phép so sánh. Cây tìm kiếm nhị phân có thể trở thành không cân đối khi các phần tử được thêm vào. Vì cây tìm kiếm nhị phân cân đối là trường hợp có độ phức tạp tối ưu trong những trường hợp tồi tệ nhất, nên những thuật toán được tạo ra đã tái cân đối các cây tìm kiếm nhị phân khi các phần tử được thêm vào. Những độc giả quan tâm vấn đề này có thể tham khảo các tài liệu về cấu trúc dữ liệu liên quan tới các thuật toán như thế.



Hình 2. Thêm các đỉnh không nhãn để tạo cây tìm kiếm nhị phân đầy đủ.

CÂY QUYẾT ĐỊNH

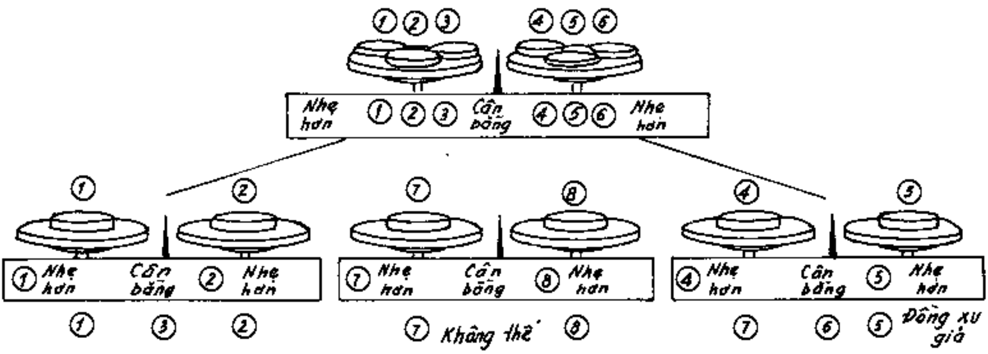
Các cây có gốc có thể dùng để mô hình các bài toán trong đó có một dãy các quyết định dẫn đến lời giải. Chẳng hạn, cây tìm kiếm nhị phân có thể dùng để định vị các phần tử dựa trên một loạt các so sánh, trong đó mỗi so sánh cho biết ta có định vị được phần tử đó hay chưa, hoặc ta sẽ đi theo cây con bên phải hay bên trái. Cây có gốc trong đó mỗi đỉnh trong ứng với một quyết định và mỗi cây con tại các đỉnh này ứng với mỗi một kết cục có thể của quyết định được gọi là **cây quyết định**. Những lời giải có thể của bài toán tương ứng với các đường đi tới các lá của cây có gốc này. Ví dụ sau sẽ minh họa một áp dụng của cây quyết định.

Ví dụ 2. Giả sử có bảy đồng xu, tất cả có trọng lượng như nhau, và một đồng giả có trọng lượng nhỏ hơn các đồng khác. Nếu dùng một chiếc cân có hai đĩa cân thì phải cần bao nhiêu lần cân để xác định đồng xu nào trong tám đồng xu này là đồng xu giả. Hãy đề xuất một thuật toán tìm đồng xu giả.

Giải: Có ba khả năng xảy ra đối với mỗi lần cân. Hai đĩa có trọng lượng bằng nhau, đĩa thứ nhất nặng hơn, đĩa thứ hai nặng hơn. Do đó cây quyết định cho một dãy các lần cân là cây tam phân. Có ít nhất tám lá trong cây quyết định vì có tám kết cục có thể (vì mỗi một trong tám đồng xu có thể là đồng xu giả) và mỗi kết cục có thể cần phải được biểu diễn bằng ít nhất một lá. Số lần cân nhiều nhất để xác định đồng xu giả là chiều cao của cây quyết định. Từ Hệ quả 1 trong Tiết 8.1 ta suy ra chiều cao của cây quyết định là $\lceil log_3 8 \rceil = 2$. Vì thế cần ít nhất hai lần cân.

Có thể xác định đồng xu giả bằng hai lần cân. Cây quyết định biểu diễn trên Hình 3.

■



Hình 3. Cây quyết định để xác định đồng xu giả.

Trong Tiết 4 của chương này chúng ta sẽ nghiên cứu thuật toán sắp xếp bằng cây quyết định.

CÁC MÃ TIỀN TỐ

Bây giờ chúng ta sẽ nghiên cứu bài toán mã hóa các chữ cái tiếng Anh bằng các dãy nhị phân (trong đó không phân biệt chữ thường với chữ hoa). Dễ thấy là chúng ta có thể biểu diễn mỗi chữ cái bằng một xâu nhị phân độ dài bằng 5 vì chỉ có 26 chữ cái mà ta có 32 xâu nhị phân độ dài 5. Tổng số các bit dùng để mã hóa các dữ liệu bằng năm lần số các ký tự dùng trong văn bản khi mỗi ký tự được mã bằng 5 bit. Liệu có thể tìm được một lược đồ mã hóa sao cho khi dữ liệu được mã hóa, thì chỉ cần dùng một số ít bit hơn không? Nếu câu trả lời là khẳng định thì ta có thể tiết kiệm bộ nhớ và giảm thời gian truyền dữ liệu.

Ta sẽ nghiên cứu cách dùng các xâu nhị phân có độ dài khác nhau để mã hóa các chữ cái tiếng Anh. Các chữ cái xuất hiện thường xuyên hơn sẽ được mã hóa bằng các xâu nhị phân ngắn, các xâu nhị phân dài hơn dùng để mã các chữ xuất hiện ít hơn. Khi các chữ được mã bằng số bit thay đổi cần phải có cách xác định xem các bit ứng với mỗi chữ bắt đầu và kết thúc ở đâu. Chẳng hạn, nếu chữ *e* được mã bằng 0, chữ *a* bằng 1 và *t* bằng 01, khi đó xâu nhị phân 0101 có thể tương ứng với *eat*, *eaea* hoặc *tt*.

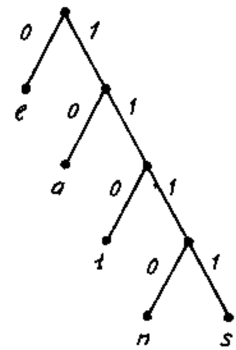
Để đảm bảo không có xâu nhị phân nào ứng với hơn một dãy các chữ cái, xâu nhị phân ứng với một chữ không bao giờ xuất hiện như là phần đầu của xâu nhị phân ứng với chữ khác. Mã có tính chất như vậy gọi là **mã tiền tố**. Ví dụ, mã *e* bằng 0, *a* bằng 10 còn *t* bằng 11 là mã tiền tố. Một từ có thể tìm lại được từ một xâu nhị phân mã các chữ cái của nó. Ví dụ xâu 10110 là mã của từ *ate*. Để giải mã ta thấy số 1 đầu tiên không biểu diễn một chữ nào nhưng 10 lại biểu diễn chữ *a* (và không thể là phần đầu của xâu nhị phân biểu diễn chữ khác). Khi đó số 1 tiếp theo không biểu diễn một chữ, nhưng 11 là chữ t. Số 0 cuối cùng biểu diễn chữ *e*.

Mã tiền số có thể biểu diễn bằng cây nhị phân, trong đó các ký tự là nhãn của lá trên cây. Các cạnh của cây được gán nhãn sao cho cạnh dẫn tới con bên trái được gán số 0, còn cạnh dẫn tới con bên phải được gán số 1. Xâu nhị phân mã hóa một chữ là dãy các nhãn của các cạnh thuộc đường đi duy nhất từ gốc tới lá có nhãn là chữ cái này. Ví dụ cây trên Hình 4 biểu diễn mã của các chữ *e bằng 0*, *a* bằng 10, *t* bằng 110, *n* bằng 1110 và *s* bằng 1111.

Cây biểu diễn mã được dùng để giải mã một dãy nhị phân. Ví dụ, ta nghiên cứu một từ được mã bởi 11111011100 nhờ mã trên Hình 4. Có thể giải mã xâu này bắt đầu từ gốc, dùng dãy các bit để tạo đường đi khi tới lá thì kết thúc. Mỗi khi gặp bit 0 ta chọn đường đi xuống theo cạnh dẫn tới con trái của đỉnh vừa đi qua và khi gặp bit 1 ta đi tới con phải của đỉnh này. Do đó, 1111 ứng với đường đi từ gốc đi theo nhánh phải bốn lần tới lá có nhãn là *s*, vì xâu 1111 là mã của chữ *s*. Tiếp tục, với bit thứ năm, ta đi theo cạnh phải rồi rẽ trái, tới lá có nhãn là chữ *a* là chữ có mã là 10. Xuất phát từ bit thứ bảy chúng ta đi tới lá tiếp theo sau khi đi theo nhánh phải ba lần liên tiếp, sau đó rẽ trái tới đỉnh có nhãn là *n* được mã bằng 1110. Bit 0 cuối cùng dẫn tới chữ *e* theo nhánh trái. Do đó từ được mã là *sane.*

Chúng ta có thể xây dựng mã tiền tố bằng bất kỳ cây nhị phân nào có cạnh trái của mỗi đỉnh trong được gán nhãn 0, cạnh phải gán nhãn 1 và các lá là các chữ cái. Các ký tự được mã bằng các xâu nhị phân tạo thành bởi các cạnh của đường đi duy nhất từ gốc tới lá.

Có các thuật toán như mã Huffman được dùng để tạo ra các mã có hiệu quả cao dựa trên tần xuất của các ký tự. Chúng ta không trình bày chi tiết của thuật toán này ở đây.



Hình 4. Cây nhị phân với mã tiền tố

BÀI TẬP

1. Hãy xây dựng cây tìm kiếm nhị phân cho các từ *banana, peach, apple, pear, coconut, mango* và *papaya* theo thứ tự từ điển.

2. Hãy xây dựng cây tìm kiếm nhị phân cho các từ *oenology, phrenology, campanology, ornithology, ichthyology, limnology, alchemy* và *astrology* theo thứ tự từ điển.

3. Cần bao nhiêu phép so sánh để định vị hay thêm các từ sau đây vào cây nhị phân trong Bài tập 1, với mỗi lần bắt đầu tìm kiếm từ đầu?

   a) *pear* b) *banana*

   c) *kumquat* d) *orange*.

4. Cần bao nhiêu phép so sánh để định vị hay thêm các từ sau đây vào cây nhị phân trong Bài tập 2 với mỗi lần bắt đầu tìm kiếm từ đầu?

   a) *palmistry* b) *etymology*

   c) *paleontology* d) *glaciology*.

5. Dùng thứ tự từ điển, hãy xây dựng cây tìm kiếm nhị phân cho câu "*The quick brown fox jumps over the lazy dog*".

6. Cần phải cân bao nhiêu lần bằng một chiếc cân hai đĩa để tìm một đồng xu giả trong bốn đồng xu? Biết rằng đồng xu giả có trọng lượng nhẹ hơn các đồng xu thật. Mô tả thuật toán tìm đồng xu nhẹ với số lần cân tìm được.

7. Cần phải cân bao nhiêu lần bằng một chiếc cân hai đĩa để tìm một đồng xu giả trong bốn đồng xu? Biết rằng đồng xu giả có trọng lượng nhẹ hơn hoặc nặng hơn các đồng xu thật. Mô tả thuật toán tìm đồng xu giả với số lần cân tìm được.

8*. Cần phải cân bao nhiêu lần bằng một chiếc cân hai đĩa để tìm một đồng xu giả trong 8 đồng xu? Biết rằng đồng xu giả có trọng lượng nhẹ hơn hoặc nặng hơn các đồng xu thật. Mô tả thuật toán tìm đồng xu giả với số lần cân tìm được.

9*. Cần phải cân bao nhiêu lần cân bằng một chiếc cân hai đĩa để tìm đồng xu giả trong mười hai đồng xu? Biết rằng đồng xu giả có trọng lượng nhẹ hơn các đồng xu thật. Mô tả thuật toán tìm đồng xu giả với số lần cân tìm được.

10*. Một trong bốn đồng xu là giả thì nó phải có trọng lượng nhẹ hơn hoặc nặng hơn các đồng xu thật. Cần phải cân bao nhiêu lần bằng một chiếc cân hai đĩa để xác định xem có đồng xu giả không, nếu có thì nó nhẹ hơn hay nặng hơn các đồng xu thật? Mô tả thuật toán tìm đồng xu giả với số lần cân tìm được và xác định xem nó nhẹ hơn hay nặng hơn các đồng xu thật.

11. Xác định cái nào là mã tiền tố trong các sơ đồ mã sau đây :

a) a : 11, e : 00, t : 10, s : 01

b) a : 0, e : 1, t : 01, s : 001

c) a : 101, e : 11, t : 001, s : 011, n : 010

d) a : 010, e : 11, t : 011, s : 1011, n : 1001, i : 10101

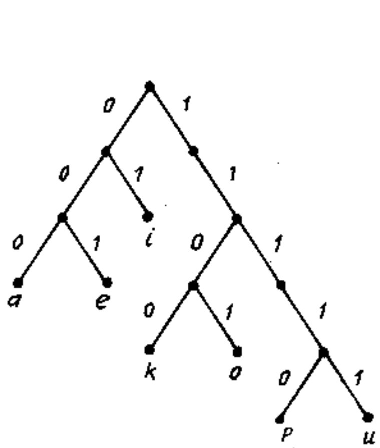
12. Dựng cây nhị phân với mã tiền tố biểu diễn các lược đồ mã hoá sau :

a) a : 11, e : 0, t : 101, s :100

b) a : 1, e : 01, t : 001, s = 0001, n = 00001

c) a : 1010, e : 0, t : 11, s : 1011, n : 1001, i : 100001

13. Hãy xác định mã của a, e, i, k, o, p và u nếu sơ đồ mã được biểu diễn bằng cây ở hình bên.



14. Cho sơ đồ mã a : 001, b : 0001, e : 1, r : 0000, s : 0100, t : 011,

x : 01010 hãy tìm các từ được biểu diễn bởi

a) 01110100011.

b) 0001110000.

c) 0100101010.

d) 01100101010.

8.3. CÁC PHƯƠNG PHÁP DUYỆT CÂY

MỞ ĐẦU

Cây có gốc và được sắp thứ tự thường được dùng để lưu trữ thông tin. Chúng ta cần có các thủ tục "viếng thăm" các đỉnh của cây để truy nhập dữ liệu. Sau đây chúng ta sẽ mô tả một số thuật toán viếng thăm tất cả các đỉnh của cây. Cây có gốc và được sắp thứ tự cũng có thể dùng để biểu diễn các loại hiểu thức khác nhau, như biểu thức số học chứa các số, các hiến và các phép toán. Những cách liệt kê khác nhau các đỉnh của cây có gốc và được sắp biểu diễn các biểu thức sẽ rất có ích khi tính giá trị của các biểu thức này.

HỆ ĐỊA CHỈ PHỔ DỤNG

Các thủ tục duyệt tất cả các đỉnh của cây có gốc và được sắp thứ tự đều dựa trên việc sắp thứ tự các đỉnh con. Trong các cây có gốc và được sắp thứ tự, khi vẽ đồ thị có hướng của chúng, các con của một đỉnh trong được thể hiện từ trái sang phải.

Dưới đây sẽ giới thiệu một cách sắp thứ tự toàn bộ các đỉnh của một cây có gốc và được sắp. Để làm điều này trước tiên ta cần phải gán nhãn cho tất cả các đỉnh bằng phương pháp truy hồi như sau :

1. Gán nhãn cho gốc bằng số nguyên 0. Sau đó k đỉnh con của nó (ở mức 1) từ trái sang phải được gán các nhãn là 1, 2, 3,..., k.

2. Với mọi đỉnh v ở mức n có nhãn là A, thì k_v đỉnh con của nó từ trái qua phải được gán nhãn là $A.1$, $A.2$,..., $A.k_v$.

Theo thủ tục này, đỉnh v ở mức n với $n \geq 1$, có nhãn là $x_1x_2. \ldots x_n$, trong đó đường đi duy nhất từ gốc tới v sẽ đi qua đỉnh thứ x_1 ở mức 1, đỉnh thứ x_2 ở mức 2, v.v. Cách gán nhãn như vậy được gọi là **hệ địa chỉ phổ dụng** của một cây có gốc và được sắp.

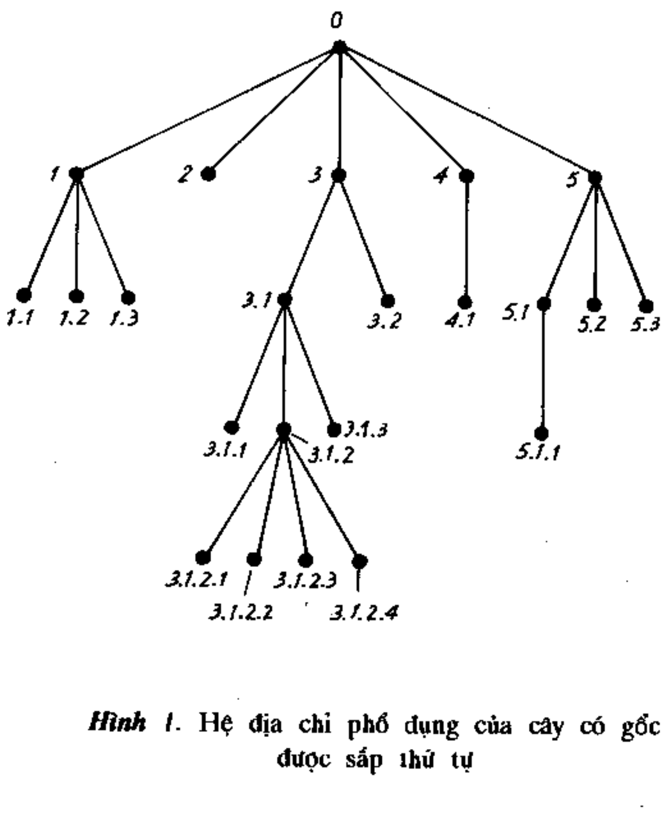
Bây giờ ta có thể sắp tất cả các đỉnh của cây theo thứ tự từ điển của các nhãn của chúng trong hệ địa chỉ phổ dụng. Đỉnh có nhãn $x_1x_2 \ldots x_n$ là nhỏ hơn đỉnh có nhãn $y_1y_2 \ldots y_m$ nếu có một giá trị của i, $0 \leq i \leq n$, sao cho $x_1 = y_1$, $x_2 = y_2, \ldots$, $x_{i-1} = y_{i-1}$ và $x_i < y_i$ hoặc nếu $n < m$ và $x_i = y_i$ với $i = 1,2,\ldots,n$.

Ví dụ 1. Chúng ta sẽ gán nhãn theo địa chỉ phổ dụng cho tất cả các đỉnh của cây trên Hình 1. Thứ tự từ điển của các nhãn là :

0 < 1 < 1.1 < 1.2 < 1.3 < 2 < 3 < 3.1 < 3.1.1. < 3.1.2 <

< 3.1.2.1 < 3.1.2.2 < 3.1.2.3 < 3.1.2.4 < 3.1.3 < 3.2 < 4 < 4.1 <

< 5 < 5.1 < 5.1.1 < 5.2 < 5.3



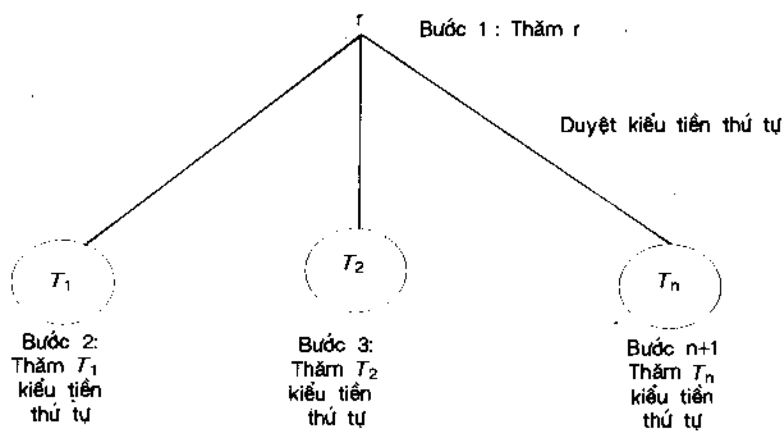
Hình 1. Hệ địa chỉ phổ dụng của cây có gốc được sắp thứ tự

CÁC THUẬT TOÁN DUYỆT CÂY

Các thủ tục viếng thăm một cách có hệ thống tất cả các đỉnh của một cây có gốc và được sắp thứ tự được gọi là các **thuật toán duyệt cây**. Dưới đây sẽ giới thiệu ba thuật toán được sử dụng thường xuyên nhất : **duyệt tiền thứ tự, duyệt trung thứ tự** và **duyệt hậu thứ tự**.

ĐỊNH NGHĨA 1. Giả sử T là một cây có gốc và được sắp thứ tự với gốc r. Nếu T chỉ có r thì r là cách *duyệt tiền thứ tự* của T. Nếu không thì gọi T_1, T_2 ... T_n là các cây con tại r từ trái qua phải của T. *Duyệt tiền thứ tự* sẽ viếng thăm r đầu tiên. Tiếp tục duyệt T_1 theo kiểu tiền thứ tự, sau đó duyệt T_2 theo kiểu tiền thứ tự, cứ như vậy cho đến khi T_n được duyệt theo kiểu tiền thứ tự.

Độc giả hãy tự kiểm tra rằng cách duyệt kiểu tiền thứ tự một cây có gốc và được sắp cho ta một thứ tự các đỉnh hệt như là thứ tự nhận được theo hệ địa chỉ phổ dụng. Hình 2 biểu thị cách duyệt theo kiểu tiền thứ tự.

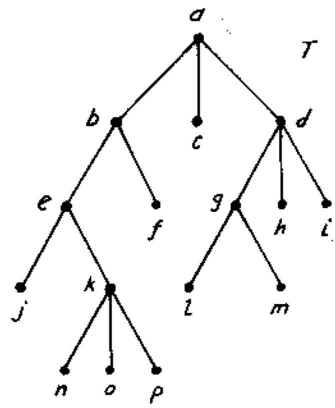


Hình 2. Duyệt cây kiểu tiền thứ tự.

Ví dụ 2. Cách duyệt tiền thứ tự sẽ viếng thăm các đỉnh của cây có gốc và được sắp trên Hình 3 theo thứ tự nào?

Giải: Các bước duyệt tiền thứ tự cây T được biểu thị trên Hình 4. Chúng ta duyệt T theo cách tiền thứ tự bằng việc viếng thăm gốc a đầu tiên. Sau đó là duyệt tiền thứ tự cây con có gốc b, duyệt tiền thứ tự cây con có gốc c (nó chỉ gồm c) và duyệt tiền thứ tự cây con có gốc d.

Duyệt tiền thứ tự cây con có gốc b bắt đầu bằng cách liệt kê b sau đó là các đỉnh của cây con có gốc e theo kiểu tiền thứ tự và sau đó là cây con có gốc f theo kiểu tiền thứ tự (nó chỉ có f). Liệt kê theo kiểu tiền thứ tự của cây con gốc d bắt đầu bằng việc liệt kê d sau đó là duyệt cây con có gốc g theo kiểu tiền thứ tự, tiếp theo là cây con có gốc h (nó chỉ có h) và duyệt cây con có gốc i (nó chính là i).



Hình 3. Cây có gốc và được sắp T

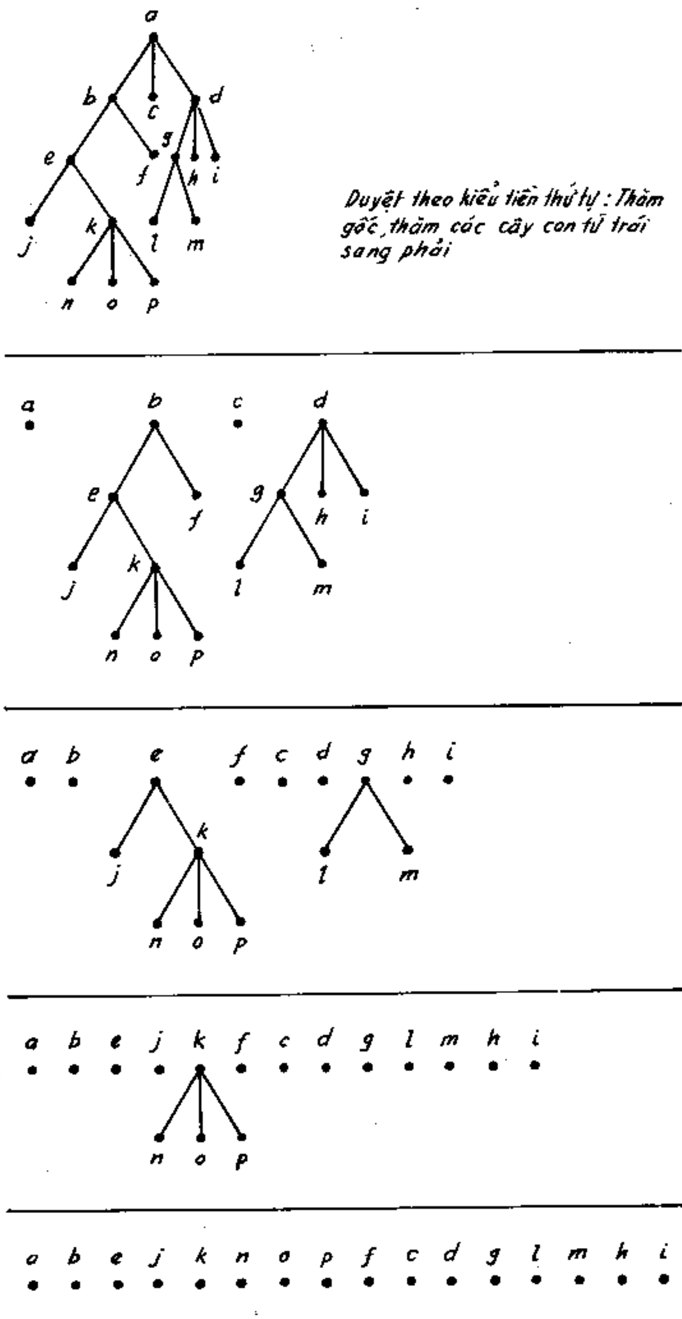
Duyệt theo kiểu tiền thứ tự cây con với gốc e bắt đầu bằng liệt kê e, tiếp theo là duyệt theo kiểu tiền thứ tự cây con có gốc j (nó chính là j), tiếp sau là duyệt theo kiểu tiền thứ tự cây con có gốc k. Liệt kê theo kiểu tiền thứ tự cây con có gốc g là g tiếp sau là l, rồi m. Duyệt theo kiểu tiền thứ tự cây con có gốc k là k, n, o, p. Do đó duyệt theo kiểu tiền thứ tự cây T là a, b, e, j, k, n, o, p, f, c, d, g, l, m, h, i.

ĐỊNH NGHĨA 2. Giả sử T là một cây có gốc và được sắp với gốc r. Nếu T chỉ có r thì r là cách *duyệt trung thứ tự* của T. Nếu không, thì gọi T_1, T_2,... T_n là các cây con tại r từ trái qua phải của T. *Duyệt trung thứ tự* sẽ bắt đầu bằng việc duyệt T_1 theo kiểu trung thứ tự, sau đó viếng thăm r. Tiếp tục duyệt T_2 theo kiểu trung thứ tự, tiếp tục duyệt T_3 theo kiểu trung thứ tự, và cứ tiếp tục cho đến khi T_n được duyệt theo kiểu trung thứ tự.

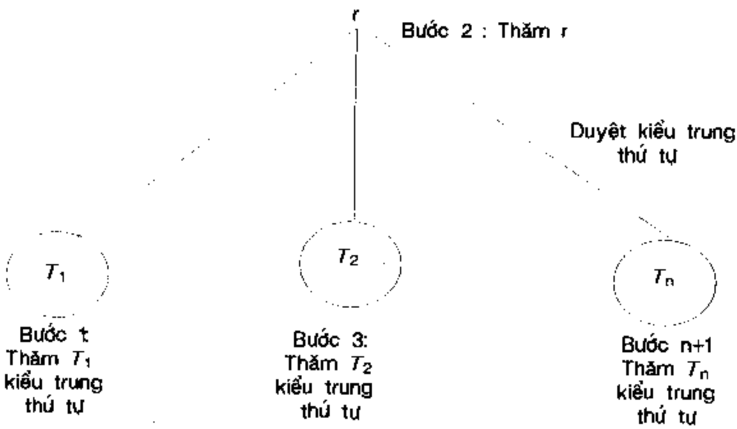
Hình 5 biểu thị cách duyệt theo kiểu trung thứ tự.

Ví dụ 3. Cách duyệt trung thứ tự sẽ viếng thăm các đỉnh của cây có gốc và được sắp trên Hình 3 theo thứ tự nào?

Giải: Các bước duyệt kiểu trung thứ tự cây có gốc và được sắp T được biểu diễn trên Hình 6. Duyệt kiểu trung thứ tự bắt đầu bằng cách duyệt kiểu trung thứ tự cây con với gốc b, sau đó là gốc a, duyệt kiểu trung thứ tự cây con với gốc c (chính là c), và duyệt kiểu trung thứ tự cây con gốc d.



Hình 4. Duyệt cây T theo kiểu tiền thứ tự.



Hình 5. Duyệt cây theo kiểu trung thứ tự.

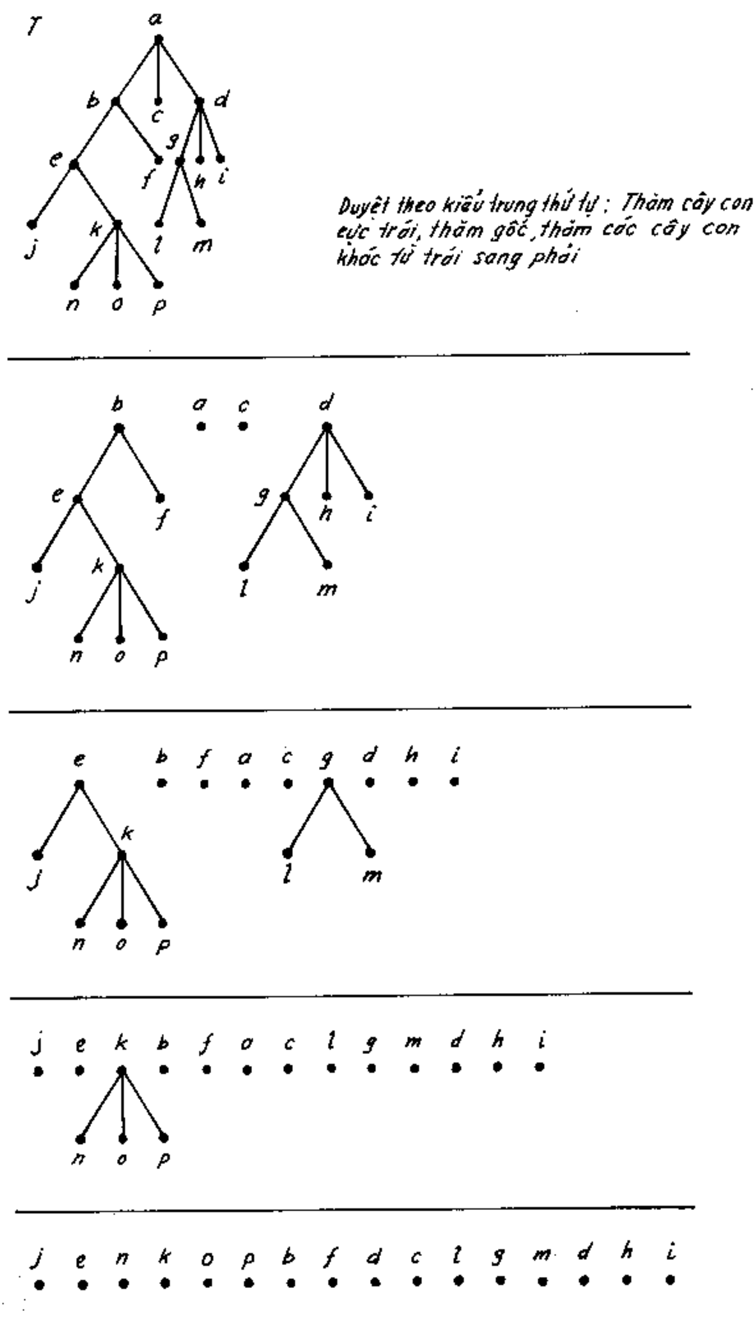
Duyệt kiểu trung thứ tự cây con với gốc b bắt đầu bằng duyệt trung tự cây con với gốc e, gốc b và f. Duyệt kiểm trung thứ tự cây con d bắt đầu bằng liệt kê trung thứ tự cây con với gốc g, tiếp theo là gốc d, sau nữa là h và i.

Duyệt kiểu trung thứ tự cây con với gốc e là j sau là gốc e, tiếp theo liệt kê kiểu trung thứ tự cây con với gốc k. Duyệt kiểu trung thứ tự cây con với gốc k là n, k, o, p. Do đó, danh sách liệt kê theo kiểu trung thứ tự của cây con có gốc và được sắp T là j, e, n, k, o, p, b, f, a, c, l, g, m, d, h, i.

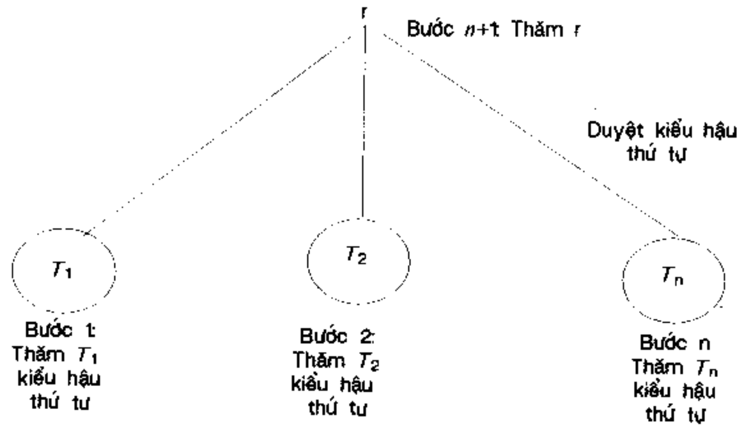
■

Định nghĩa duyệt theo kiểu hậu thứ tự như sau.

ĐỊNH NGHĨA 3. Giả sử T là một cây có gốc và được sắp với gốc r. Nếu T chỉ có r thì r là cách *duyệt hậu thứ tự* của T. Nếu không, thì gọi T_1, T_2, ..., T_n là các cây con tại r từ trái qua phải của T. *Duyệt hậu thứ tự* sẽ bắt đầu bằng việc duyệt T_1 theo kiểu hậu thứ tự, sau đó duyệt T_2 theo kiểu hậu thứ tự và cứ tiếp tục cho đến khi T_n được duyệt theo kiểu hậu thứ tự, và cuối cùng kết thúc bằng việc viếng thăm r.



Hình 6. Duyệt cây T theo kiểu trung thứ tự.



***Hình** 7.* Duyệt cây theo kiểu hậu thứ tự.

Hình 7 biểu thị cách duyệt theo kiểu hậu thứ tự.

Ví dụ 4. Cách duyệt kiểu hậu thứ tự sẽ viếng thăm các đỉnh của cây có gốc và được sắp trên Hình 3 theo thứ tự nào?

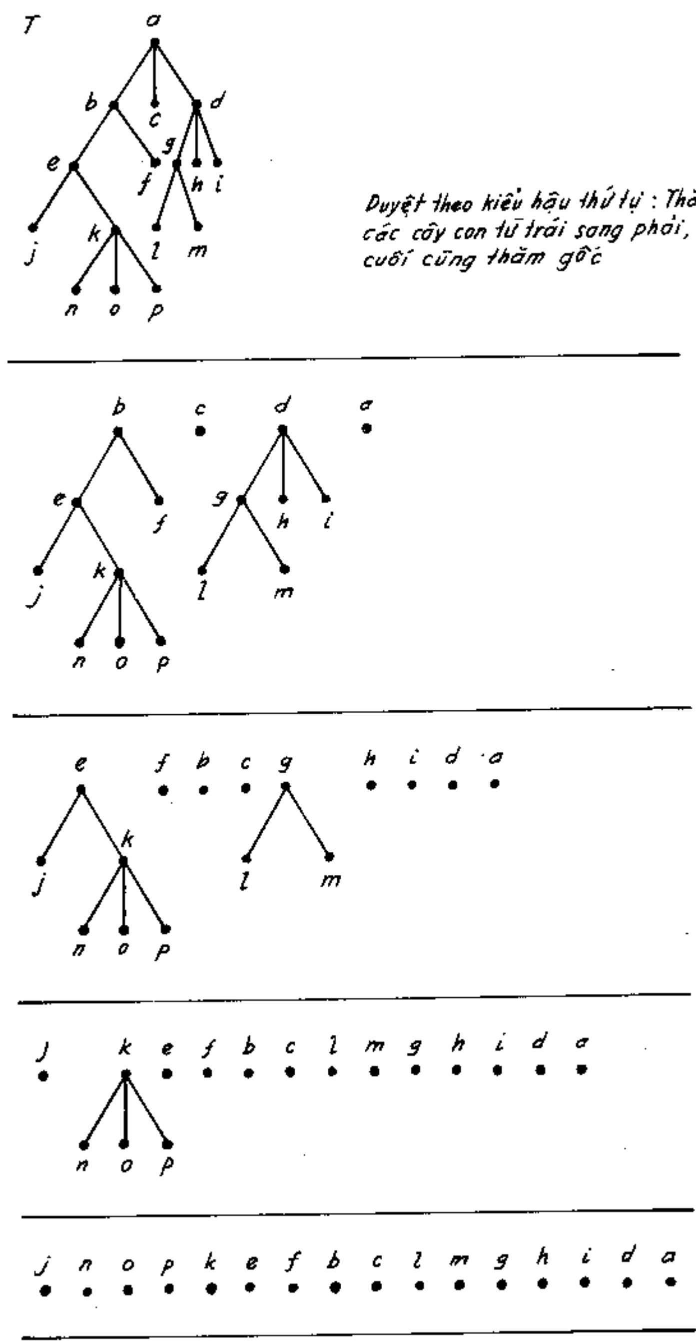
Giải: Các bước duyệt kiểu hậu thứ tự cây có gốc và được sắp T được biểu diễn trên Hình 8. Duyệt kiểu hậu thứ tự bắt đầu bằng cách duyệt hậu thứ tự cây con với gốc b, duyệt kiểu hậu thứ tự cây con với gốc c (chính là c), và duyệt kiểu hậu thứ tự cây con gốc d, cuối cùng là gốc a.

Duyệt kiểu hậu thứ tự cây con với gốc b bắt đầu bằng duyệt hậu thứ tự cây con với gốc e, f và tiếp theo là gốc b. Duyệt kiểu hậu thứ tự cây con d bắt đầu bằng liệt kê kiểu hậu thứ tự cây con với gốc g, *tiếp theo là* h và i, sau nữa là gốc d.

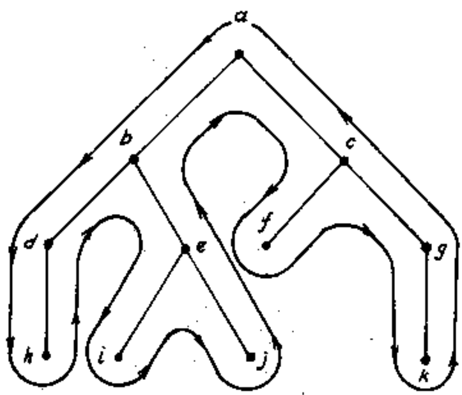
Duyệt kiểu hậu thứ tự cây con với gốc e bắt đầu bằng j sau đó liệt kê kiểu hậu thứ tự cây con với gốc k, tiếp theo là gốc e. Duyệt kiểu hậu thứ tự cây con với gốc g là l, m, g. Duyệt kiểu hậu thứ tự cây con với gốc k là n, o, p, k. Do đó, danh sách liệt kê theo kiểu hậu thứ tự của cây con có gốc và được sắp T là j, n, o, p, k, e, f, b, c, l, m, g, h, i, d, a.

■

Có nhiều cách dễ dàng liệt kê các đỉnh của cây có gốc và được sắp theo kiểu tiền, trung hay hậu thứ tự. Trước tiên ta vẽ một đường cong bao quanh cây đang xét, xuất phát từ gốc chuyển động dọc theo các cạnh như trên Hình vẽ 9. Ta có thể liệt kê các đỉnh của cây theo kiểu tiền



Hình 8. Duyệt cây T theo kiểu hậu thứ tự.



Hình 9. Cách dễ nhớ các cách duyệt cây theo các kiểu tiền, trung, hậu thứ tự.

thứ tự bằng cách liệt kê mỗi đỉnh khi đường cong đi qua nó. Ta có thể nhận được danh sách các đỉnh theo kiểu trung thứ tự bằng cách liệt kê các lá khi đi ngang qua nó lần đầu và liệt kê các đỉnh trong khi đi ngang qua nó lần thứ hai. Ta có thể nhận được danh sách các đỉnh theo kiểu hậu thứ tự bằng cách liệt kê các đỉnh khi đi ngang qua nó lần thứ hai để trở về cha của nó. Theo quy tắc này với cây có gốc trên Hình 9 kiểu duyệt tiền thứ tự cho ta a, b, d, h, e, i, j, c, f, g, k, kiểu duyệt trung thứ tự cho ta h, d, b, i, e, j, a, f, c, k, g còn theo kiểu duyệt hậu thứ tự thì ta được h, d, i, j, e, b, f, k, g, c, a.

THUẬT TOÁN 1. DUYỆT KIỂU TIỀN THỨ TỰ.

procedure *preorder* (*T* : cây có gốc và được sắp)

r : = gốc của *T*

liệt kê *r*

for mỗi cây con *c* của *r* từ trái sang phải

begin

  T(c) : = Cây con với gốc *c*

  preorder (*T(c)*)

end

THUẬT TOÁN 2. DUYỆT KIỂU TRUNG THỨ TỰ.

procedure *inorder (T)* : cây có gốc và được sắp) ;

r : = gốc của *T*
if *t* là lá **then** liệt kê *r*

else

begin

l : = con đầu tiên từ trái sang phải của *r*
T *(l)* : = Cây con với gốc *l*
inorder *(T(l))*

liệt kê *r*

for mỗi cây con *c* của *r* từ trái sang phải trừ *l*

T (c) : = Cây con với gốc *c*

inorder (T(c))

end.

THUẬT TOÁN 3. DUYỆT KIỂU HẬU THỨ TỰ.
procedure *postorder (T* : cây có gốc và được sắp)
r : = gốc của *T*

for mỗi cây con *c* của *r* từ trái sang phải

begin

T (c) : = Cây con với gốc *c*
postorder(T(c))

end.

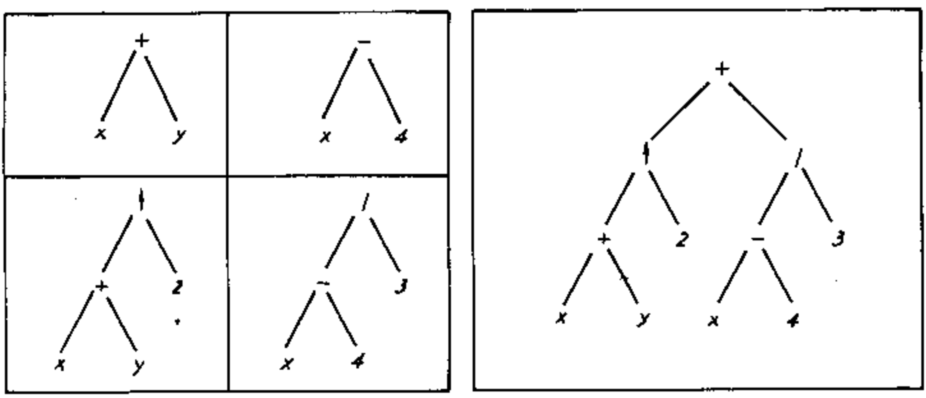
liệt kê *r*

CÁC KÝ PHÁP TRUNG TỐ, TIỀN TỐ VÀ HẬU TỐ

Chúng ta có thể biểu diễn các biểu thức phức tạp như các mệnh đề phức hợp, những tổ hợp của các tập hợp, các biểu thức số học bằng cây có gốc và được sắp. Ví dụ, chúng ta nghiên cứu cách biểu diễn các biểu thức số học có chứa các toán tử + (cộng), − (trừ), * (nhân), / (chia) và ↑ (lũy thừa). Chúng ta sẽ dùng các dấu ngoặc để biểu thị thứ tự các phép toán. Những biểu thức như thế có thể được biểu diễn bằng các cây có gốc và được sắp, trong đó các đỉnh trong biểu thị các phép toán các lá biểu thị các số hay các biến. Mỗi một phép toán tác động lên các cây con bên trái và cây con bên phải của nó (theo thứ tự này).

Ví dụ 5. Tìm cây có gốc biểu diễn biểu thức

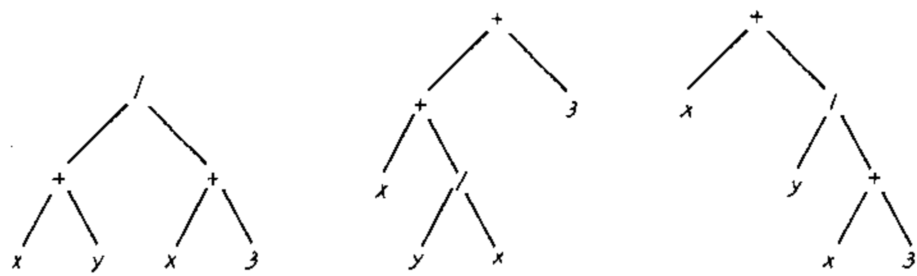
$$((x + y)\uparrow 2) + ((x - 4)/3).$$



Hình 10. Cây nhị phân biểu diễn $((x + y)\uparrow 2) + ((x - 4)/3)$.

Giải: Cây nhị phân cho biểu thức này có thể được xây dựng từ dưới lên. Trước tiên xây dựng cây con cho biểu thức $x + y$. Sau đó kết hợp thành cây con lớn hơn $(x + y)\uparrow 2$. Cũng như vậy, ta xây dựng cây con $(x - 4)$ và sau đó kết hợp thành cây con $(x - 4)/3$. Cuối cùng hai cây con biểu thị $(x + y)\uparrow 2$ và $(x - 4)/3$ kết hợp với nhau để nhận được cây có gốc và được sắp biểu diễn $((x + y)\uparrow 2) + ((x - 4)/3)$. Các bước này được thể hiện trên Hình 10. ■

Cách duyệt cây nhị phân biểu diễn biểu thức đã cho theo kiểu trung thứ tự sẽ tạo ra biểu thức có các số hạng và các phép toán theo đúng thứ tự như là đã có trong biểu thức ban đầu trừ các phép toán một ngôi thay vì đi ngay sau các toán hạng của chúng. Ví dụ, duyệt kiểu trung thứ tự các cây nhị phân trên Hình 11 biểu diễn các biểu thức $(x + y)/(x + 3)$, $(x + (y/x)) + 3$ và $x + (y/(x + 3))$ tất cả đều dẫn tới biểu thức trung tố $x + y/a + 3$. Để cho các biểu thức này rõ ràng cần phải dùng các dấu ngoặc trong cách duyệt trung thứ tự mỗi khi ta gặp một phép toán. Biểu thức có đầy đủ dấu ngoặc đơn nhận được bằng cách như vậy được gọi là ở **dạng trung tố.**



Hình 11. Cây có gốc biểu diễn $(x + y)/(x + 3)$, $(x + (y/x)) + 3$, và $x + (y/x + 3)$.

Chúng ta nhận được **dạng tiền tố** của biểu thức khi ta duyệt cây có gốc theo kiểu tiền thứ tự. Các biểu thức được viết dưới dạng tiền tố được gọi là **ký pháp Ba lan**. Một biểu thức ở dạng tiền tố (trong đó mỗi phép toán đều có một số xác định các toán hạng) là rõ ràng tức là không cần dùng các dấu ngoặc trong các biểu thức này. Độc giả tự kiểm tra lại điều này như một bài tập.

Ví dụ 6. Tìm dạng tiền tố của biểu thức $((x + y) \uparrow 2) + ((x - 4)/3)$.

Giải: Chúng ta nhận được dạng tiền tố của biểu thức này bằng cách duyệt cây nhị phân biểu diễn nó trên Hình 10 theo kiểu tiền thứ tự. Từ đó

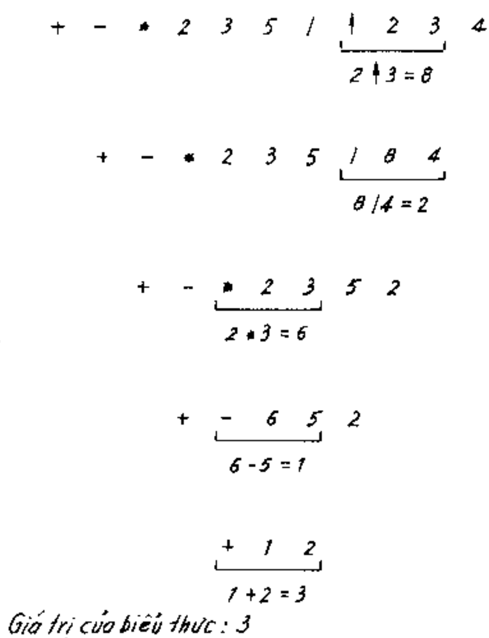
nhận được $+ \uparrow + x\ y\ 2 / - x\ 4\ 3$.

Trong dạng tiền tố của một biểu thức một toán tử hai ngôi, như phép cộng, đi trước hai toán hạng của nó. Vì thế, chúng ta có thể đánh giá một biểu thức ở dạng tiền tố bằng cách đi từ phải sang trái. Khi chúng ta gặp một toán tử ta thực hiện phép toán tương ứng có hai toán hạng đi liền bên phải của toán tử này. Cũng vậy mỗi khi một phép toán được thực hiện chúng ta coi kết quả như một toán hạng mới.

Ví dụ 7. Tính giá trị của biểu thức tiền tố $+ - * 2\ 3\ 5 / \uparrow 2\ 3\ 4$.

Giải: Các bước để tính giá trị của biểu thức này được tiến hành từ phải sang trái và thực hiện các phép toán với các toán hạng ở bên phải nó, như chỉ ra trên Hình 12. Kết quả nhận được là 3.

Chúng ta nhận được **dạng hậu tố** của một biểu thức bằng cách duyệt cây nhị phân theo kiểu hậu thứ tự. Biểu thức viết dưới dạng hậu tố được gọi là **ký pháp Ba lan ngược**. Các biểu thức dưới dạng ký pháp Ba lan ngược là rõ ràng (không mập mờ), vì vậy không cần dùng dấu ngoặc. Độc giả tự kiểm tra điều này.



Hình 12. Tính giá trị của biểu thức tiền tố.

Ví dụ 8. Tìm dạng hậu tố của biểu thức $((x + y) \uparrow 2) + ((x - 4)/3)$.

Giải. Dạng hậu tố của biểu thức này nhận được bằng cách duyệt theo kiểu hậu thứ tự cây nhị phân ứng với biểu thức này, như trên Hình 10. Ta nhận được biểu thức hậu tố $x\ y + 2 \uparrow\ x\ 4 - 3\ / +$.

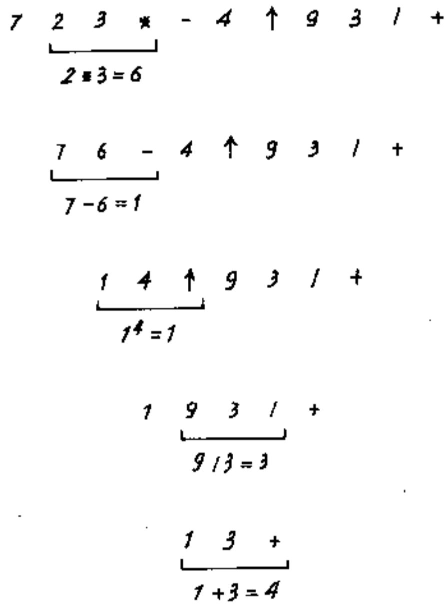
Trong dạng hậu tố này của một biểu thức các toán tử hai ngôi đi sau hai toán hạng của chúng. Vì thế để đánh giá một biểu thức ở dạng hậu tố ta phải tiến hành từ trái sang phải và thực hiện một phép toán mỗi khi có một toán tử đi sau hai toán hạng. Sau khi mỗi phép toán được thực hiện kết quả của phép toán này trở thành toán hạng của phép toán mới.

■

Ví dụ 9. Tìm giá trị của biểu thức hậu tố $7\ 2\ 3 * - 4 \uparrow 9\ 3\ / +$.

Giải. Các bước tính giá trị của biểu thức này được tiến hành từ bên trái và thực hiện các phép toán với hai toán hạng ở bên trái toán tử, như chỉ ra trên Hình 13. Kết quả nhận được là 4.

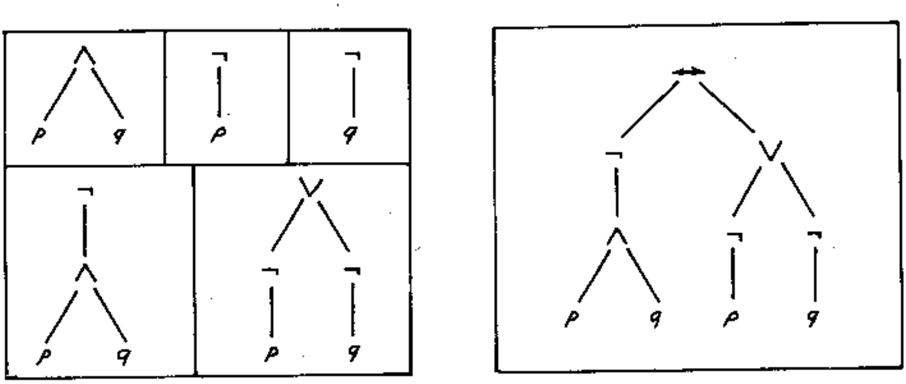
Cây có gốc có thể dùng để biểu diễn các biểu thức có dạng khác, ví dụ như các mệnh đề phức hợp hay tổ hợp các tập hợp. Trong các ví dụ này có các toán tử một ngôi, chẳng hạn như phép phủ định một mệnh đề. Để biểu diễn các toán tử và các toán hạng loại này, chúng ta coi các đỉnh là các toán tử, con của các đỉnh này là các toán hạng mà nó sẽ sử dụng.



Hình 13. Tính giá trị của biểu thức hậu tố.

Ví dụ 10. Hãy tìm cây có gốc và được sắp biểu diễn mệnh đề lôgic phức hợp : $(\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$. Sau đó dùng cây nhị phân này hãy tìm dạng tiền tố, hậu tố và trung tố của biểu thức này.

Giải: Cây có gốc ứng với mệnh đề phức hợp này được xây dựng từ dưới lên. Trước tiên ta tạo các cây con $\neg p$ và $\neg q$ (trong đó $\neg$ là toán tử phủ định). Tương tự ta xây dựng cây con $p \wedge q$. Sau đó là các cây con $\neg(p \wedge q)$ và $(\neg p) \vee (\neg q)$ được xây dựng. Cuối cùng dùng hai cây con này để tạo cây có gốc cần tìm. Từng bước của thủ này được thể hiện trên Hình 14.



Hình 14. Xây dựng cây có gốc cho mệnh đề phức hợp.

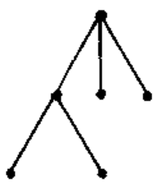
Dạng tiền tố, hậu tố và trung tố của biểu thức này được tìm bằng cách duyệt cây có gốc vừa xây dựng ở trên tương ứng theo kiểu tiền, hậu và trung thứ tự (kể cả các dấu ngoặc). Kết quả chúng ta nhận được $\leftrightarrow \neg \wedge pq \vee \neg p \neg q$, $pq \wedge \neg p \neg q \neg \vee \leftrightarrow$, và $(\neg(p \wedge q)) \leftrightarrow ((\neg p) \vee (\neg q)$, tương ứng.

Vì các biểu diễn tiền tố và hậu tố là rõ ràng, không nhập nhằng nước đôi và bởi vì có thể dễ dàng tính giá trị của chúng mà không phải quét tới quét lui, nên chúng được sử dụng rất nhiều trong tin học. Các biểu diễn như thế đặc biệt có lợi khi xây dựng các bộ dịch.

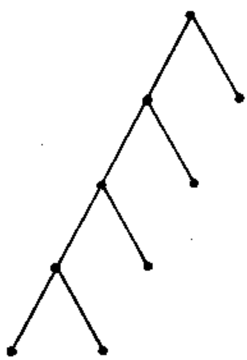
BÀI TẬP

Trong các bài tập 1-3 hãy xây dựng các hệ địa chỉ phổ dụng cho các cây có gốc được sắp đã cho. Sau đó dùng chúng để sắp xếp các đỉnh của cây theo thứ tự từ điển của các nhãn.

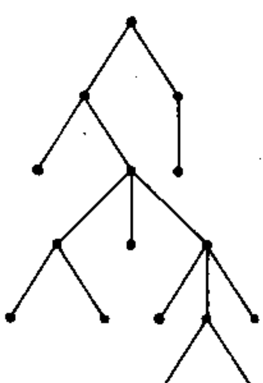
1.



2.



3.



4. Giả sử địa chỉ của đỉnh v trong cây T là 3.4.5.2.4.

a) v ở mức nào?

b) Tìm địa chỉ của cha của v.

c) v có tối thiểu mấy anh em?

d) T có ít nhất bao nhiêu đỉnh?

e) Hãy tìm các địa chỉ khác phải có.

5. Giả sử đỉnh có địa chỉ lớn nhất trong cây T có địa chỉ là 2.3.4.3.1. Có thể xác định được số đỉnh trong T hay không?

6 Các lá của cây có gốc và được sắp có thể có danh sách địa chỉ phổ dụng như sau được không? Nếu có, hãy xây dựng cây có gốc đó.

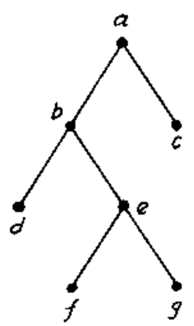
a) 1.1.1, 1.1.2, 1.2, 2.1.1.1, 2.1.2, 2.1.3, 2.2, 3.1.1, 3.1.2.1, 3.1.2.2, 3.2

b) 1.1, 1.2.1, 1.2.2, 1.2.3, 2.1, 2.2.1, 2.3.1, 2.3.2, 2.4.2.1, 2.4.2.2, 3.1, 3.2.1, 3.2.2

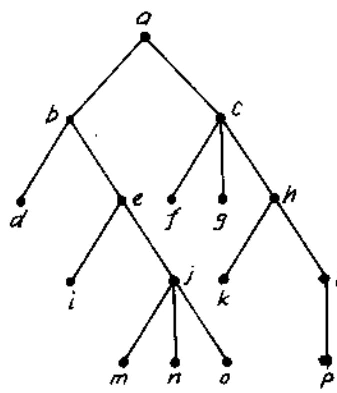
c) 1.1, 1.2.1, 1.2.2, 1.2.2.1, 1.3, 1.4, 2, 3.1, 3.2, 4.1.1.1

Trong các Bài tập từ 7-9 hãy xác định thứ tự mà các đỉnh của cây có gốc được viếng thăm nếu ta duyệt nó theo kiểu tiền thứ tự.

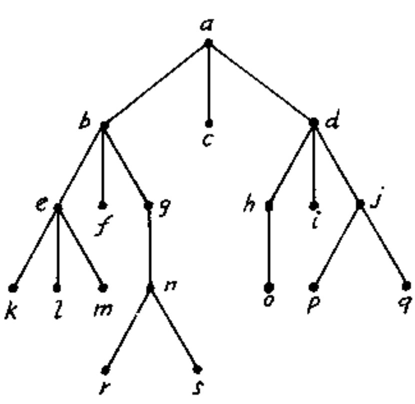
7.



8.



9.



10. Xác định thứ tự mà các đỉnh của cây có gốc trong Bài tập 7 được viếng thăm nếu ta duyệt nó theo kiểu trung thứ tự.

11. Xác định thứ tự mà các đỉnh của cây có gốc trong Bài tập 8 được viếng thăm nếu ta duyệt nó theo kiểu trung thứ tự.

12. Xác định thứ tự mà các đỉnh của cây có gốc trong Bài 9 được viếng thăm nếu ta duyệt nó theo kiểu trung thứ tự.

13. Xác định thứ tự mà các đỉnh của cây có gốc trong Bài tập 7 được viếng thăm nếu ta duyệt nó theo kiểu hậu thứ tự.

14. Xác định thứ tự mà các đỉnh của cây có gốc trong Bài tập 8 được viếng thăm nếu ta duyệt nó theo kiểu hậu thứ tự.

15. Xác định thứ tự mà các đỉnh của cây có gốc trong Bài tập 9 được viếng thăm nếu ta duyệt nó theo kiểu hậu thứ tự.

16. Hãy biểu diễn biểu thức $((x + 2) \uparrow 3) * (y - (3 + x)) - 5$ bằng cây nhị phân.

17. Hãy viết biểu thức trong Bài tập 16 dưới dạng :

a) ký pháp tiền tố

b) ký pháp hậu tố

c) ký pháp trung tố.

18. Hãy biểu diễn các biểu thức $(x + xy) + (x/y)$ và $x + ((xy + x)/y)$ bằng cây nhị phân.

19. Hãy viết biểu thức trong Bài 18 dưới dạng :

a) ký pháp tiền tố

b) ký pháp hậu tố

c) ký pháp trung tố.

20. Hãy biểu diễn các mệnh đề phức hợp $(\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$ và $(\neg p \wedge (q \leftrightarrow \neg p)) \vee \neg q$ bằng cây có gốc và được sắp.

21. Hãy viết biểu thức trong Bài tập 20 dưới dạng :

a) ký pháp tiền tố

b) ký pháp hậu tố

c) ký pháp trung tố.

22. Hãy biểu diễn $(A \cap B) - (A \cup (B - A))$ bằng cây có gốc được sắp.

23. Hãy viết hiểu thức trong Bài tập 22 dưới dạng :

a) ký pháp tiền tố

h) ký pháp hậu tố

c) ký pháp trung tố.

24*. Xâu $\neg p \wedge q \leftrightarrow \neg p \vee \neg q$ có thể được đặt trong ngoặc đơn theo bao nhiêu cách để sinh ra một biểu thức trung tố?

25*. Xâu $A \cap B - A \cap B - A$ có thể được đặt trong ngoặc đơn theo bao nhiêu cách để sinh ra một biểu thức trung tố?

26. Hãy vẽ cây có gốc được sắp xếp ứng với mỗi biểu thức số học được viết ở dạng ký pháp tiền tố sau đây. Sau đó hãy viết biểu thức bằng ký pháp trung tố.

a) + * + − 5 3 2 1 4

b) ↑ + 2 3 − 5 1

c) */9 3 + * 2 4 − 7 6

27. Hãy tính giá trị của các biểu thức tiền tố sau đây :

a) − * 2/8 4 3

b) ↑ * 3 3 * 4 2 5

c) + − ↑ 3 2 ↑ 2 3 / 6 − 4 2

d) * + 3 + 3 ↑ 3 + 3 3 3

28. Hãy tính giá trị của các biểu thức hậu tố sau đây :

a) 5 2 1 − − 3 1 4 + + *

b) 9 3 / 5 + 7 2 − *

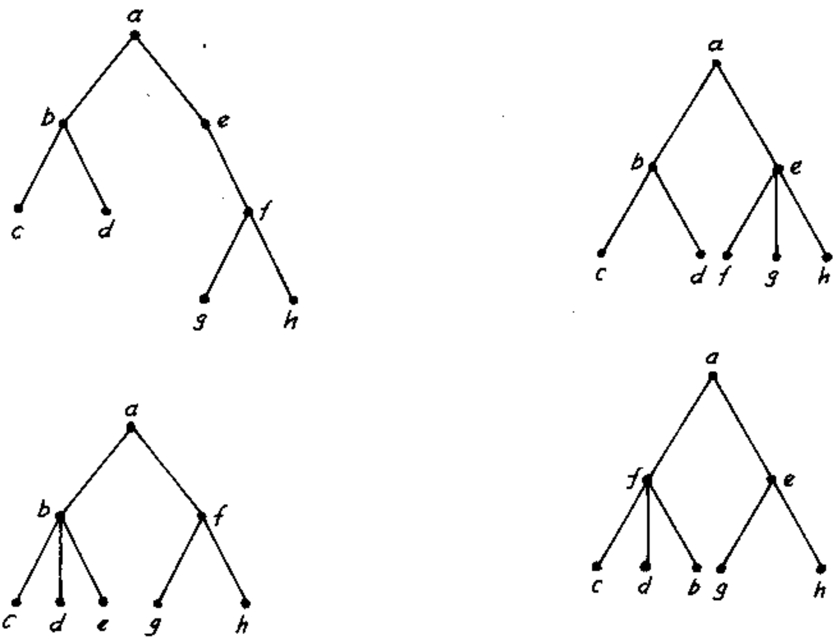
c) 3 2 * 2 ↑ 5 3 − 8 4 / * −

29. Hãy xây dựng cây có gốc được sắp sao cho khi duyệt nó theo kiểu tiền thứ tự ta được *a, b, f, c, g, h, i, d, e, j, k, l* trong đó *a* có 4 con, *c* có 3 con, *j* có 2 con, *b* và *e* đều có 1 con còn tất cả các đỉnh khác đều là lá.

30*. Hãy chứng tỏ rằng cây có gốc được sắp là xác định duy nhất khi cho danh sách các đỉnh của nó sinh ra bằng cách duyệt tiền thứ tự và số con của mỗi đỉnh là được cho trước.

31*. Hãy chứng tỏ rằng cây có gốc được sắp là xác định duy nhất khi cho danh sách các đỉnh của nó sinh ra bằng cách duyệt hậu thứ tự và số con của mỗi đỉnh là được cho trước.

32. Hãy chỉ ra rằng khi duyệt hai cây dưới đây (hình bên trái) theo kiểu tiền thứ tự sẽ tạo ra cùng một danh sách các đỉnh. Lưu ý là điều này không mâu thuẫn với Bài tập 30, vì số con của các đỉnh trong của hai cây này là khác nhau.



33. Hãy chỉ ra rằng khi duyệt hai cây (hình trên, bên phải) theo kiểu hậu thứ tự sẽ tạo ra cùng một danh sách các đỉnh. Lưu ý là điều này không mâu thuẫn với Bài tập 31, vì số con của các đỉnh trong của hai cây này là khác nhau.

Các công thức được tạo đúng *ở dạng ký pháp tiền tố từ tập các ký hiệu và tập các toán tử hai ngôi được định nghĩa dệ quy theo quy tắc sau đây :*

(i) Nếu x là một ký hiệu, thì x là công thức được tạo dúng ở dạng ký pháp tiền tố ;

*(ii) Nếu X và Y là các công thức được tạo đúng và * là một toán tử thì * XY là một công thức được tạo đúng.*

34. Trong các công thức sau đây công thức nào là được tạo đúng từ tập các ký hiệu $\{x, y, z\}$ và tập các toán tử hai ngôi $\{\times, +, \circ\}$?

a) $\times + + x\ y\ x$ b) $\circ\ x\ y \times x\ z$

c) $\times \circ x\ z \times \times x\ y$ d) $\times + \circ x\ x \circ x\ x\ x$

35*. Chỉ ra rằng một công thức được tạo đúng bất kỳ ở dạng ký pháp tiền tố từ tập các ký hiệu và tập các toán tử hai ngôi có số ký hiệu nhiều hơn số toán tử đúng một đơn vị.

36. Đưa ra định nghĩa công thức được tạo đúng ở dạng ký pháp hậu tố từ một tập các ký hiệu và tập các toán tử hai ngôi.

37. Hãy đưa ra sáu ví dụ về công thức được tạo đúng có ba hay nhiều hơn các toán tử ở dạng hậu tố từ tập các ký hiệu $\{x, y, z\}$ và tập các toán tử hai ngôi $\{\times, +, \circ\}$.

38. Hãy mở rộng định nghĩa công thức được tạo đúng ở dạng ký pháp tiền tố từ một tập các ký hiệu và tập các toán tử trong đó các toán tử có thể không là hai ngôi.

8.4. CÂY VÀ BÀI TOÁN SẮP XẾP

MỞ ĐẦU

Bài toán sắp xếp các phần tử của một tập hợp xuất hiện trong rất nhiều lĩnh vực. Ví dụ, để lập danh bạ điện thoại cần phải sắp xếp tên của những người thuê bao theo thứ tự từ điển.

Giả sử cần sắp xếp toàn bộ các phần tử của một tập hợp. Ban đầu các phần tử của tập có thể được sắp đặt theo một trật tự nào đó. **Sắp xếp** (sorting) là sự sắp đặt lại các phần tử này vào một danh sách theo thứ tự tăng dần. Ví dụ, sắp xếp danh sách 7, 2, 1, 4, 5, 9 sẽ tạo ra danh sách 1, 2, 4, 5, 7, 9, Sắp xếp danh sách *d, h, c, a, f* (theo thứ tự từ điển) sẽ cho danh sách *a, c, d, f, h*.

Phần lớn các công việc của máy tính là dành cho việc sắp xếp các đối tượng thuộc các loại khác nhau. Vì thế, người ta đã dành rất nhiều công sức cho việc phát triển các thuật toán sắp xếp có hiệu quả. Trong mục này chúng ta sẽ trình bày một vài thuật toán sắp xếp và độ phức tạp tính toán của chúng. Cũng trong tiết này chúng ta sẽ thấy cây được dùng để mô tả các thuật toán sắp xếp và dùng để phân tích độ phức tạp của chúng như thế nào.

ĐỘ PHỨC TẠP CỦA SẮP XẾP

Hiện nay có khá nhiều thuật toán sắp xếp. Để đánh giá tính hiệu quả của mỗi thuật toán người ta cần xác định độ phức tạp của nó. Bằng mô hình cây có thể tìm được cận dưới cho trường hợp có độ phức tạp tồi tệ nhất của thuật toán sắp xếp.

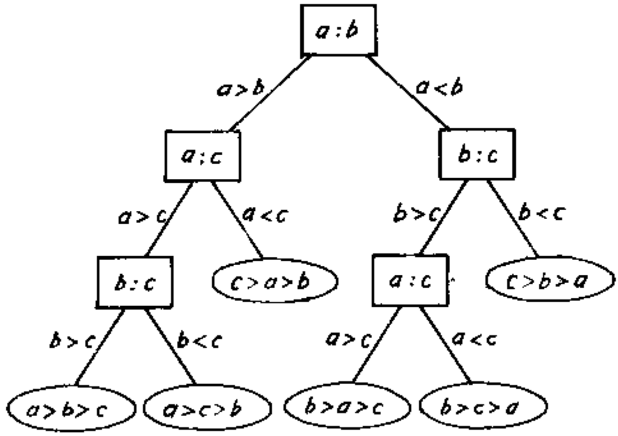
Giả sử ta có một tập gồm n phần tử. Khi đó ta sẽ có $n!$ cách sắp xếp các phần tử này, đó là số hoán vị của chúng. Các thuật toán sắp xếp mà chúng ta sẽ nghiên cứu đều dựa trên các phép so sánh nhị nguyên, tức là mỗi lần so sánh hai phần tử với nhau. Kết quả của mỗi phép so sánh như thế sẽ thu hẹp tập các cách sắp xếp có thể. Như vậy, thuật toán sắp xếp dựa trên so sánh nhị nguyên có thể được biểu diễn bằng cây quyết định nhị phân với các đỉnh trong là các phép so sánh hai phần tử, mỗi lá là một trong $n!$ hoán vị của n phần tử.

Ví dụ 1. Trên Hình 1 chúng ta biểu diễn cây quyết định để sắp xếp các phần tử của danh sách a, b, c.

Độ phức tạp sắp xếp dựa trên so sánh nhị nguyên được đo bằng số các phép so sánh được sử dụng. Số tối đa các phép so sánh cần dùng để sắp xếp một danh sách có n phần tử chính là trường hợp tồi tệ nhất của thuật toán. Số tối đa các phép so sánh cần dùng sẽ bằng độ dài của đường đi dài nhất trong cây quyết định biểu diễn thủ tục sắp xếp. Nói cách khác số tối đa các phép so sánh cần dùng sẽ bằng chiều cao của cây quyết định. Vì chiều cao của cây nhị phân với $n!$ lá tối thiểu bằng $\lceil \log n! \rceil$ (dùng Hệ quả 1 trong Tiết 8.1) nên cần ít nhất $\lceil \log n! \rceil$ phép so sánh, như Định lý 1 sau khẳng định.

ĐỊNH LÝ 1. Thuật toán sắp xếp dựa trên so sánh nhị nguyên đòi hỏi ít nhất $\lceil \log n! \rceil$ phép so sánh.

Theo Ví dụ 6 của Tiết 1.8 ta suy ra $\lceil \log n! \rceil$ là $O(n\log n)$. Thật vậy, vì $\log n!$ lớn hơn $(n\log n)/4$ với mọi $n > 4$ (xem Bài tập 18) ta suy ra không có thuật toán sắp xếp nào dùng các phép so sánh như là một phương pháp sắp xếp có thể có độ phức tạp về thời gian trong trường hợp tồi tệ nhất lại tốt hơn $O(n\log n)$. Do đó thuật toán sắp xếp là có hiệu quả tốt nhất có thể nếu độ phức tạp thời gian của nó là $O(n\log n)$.



Hình 1. Cây quyết định để sắp xếp ba phần tử khác nhau.

SẮP XẾP KIỂU NỔI BỌT

Sắp xếp kiểu nổi bọt là một thuật toán sắp xếp đơn giản nhất nhưng không phải là một trong những thuật toán có hiệu quả nhất. Thuật toán này đặt danh sách theo thứ tự tăng dần bằng cách so sánh liên tiếp các phần tử kề nhau, đổi chỗ chúng cho nhau nếu chúng chưa có thứ tự tăng dần. Để tiến hành sắp xếp kiểu nổi bọt chúng ta thực hiện một thao tác cơ bản, đó là sự đổi chỗ phần tử lớn hơn với phần tử nhỏ hơn đi sau, bắt đầu từ đầu danh sách và duyệt qua toàn bộ danh sách. Chúng ta lặp thủ tục này cho tới khi việc sắp xếp được hoàn thành. Ta hãy tưởng tượng các phần tử được đặt vào một cột. Trong sắp xếp kiểu nổi bọt các phần tử nhỏ hơn sẽ "nổi" lên trên vì chúng đổi chỗ với các phần tử lớn hơn. Các phần tử lớn hơn sẽ "chìm" xuống đáy. Điều này được minh họa trong ví dụ sau.

Ví dụ 2. Hãy sắp xếp 3, 2, 4, 1, 5 theo thứ tự tăng dần bằng cách dùng thuật toán nổi bọt.

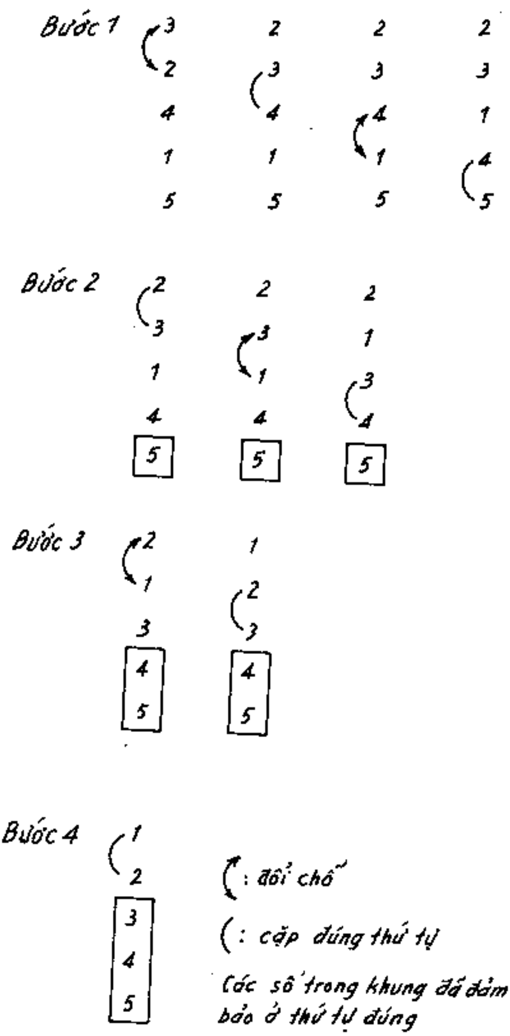
Giải: Trước tiên ta so sánh hai phần tử 3 và 2. Vì 3 > 2 nên đổi chỗ 3 với 2, ta được danh sách 2, 3, 4, 1, 5. vì 3 < 4 ta tiếp tục so sánh 4 với 1. Vì 4 > 1 nên đổi chỗ 4 với 1, ta nhận được danh sách 2, 3, 1, 4, 5. Vì 4 < 5 nên vòng duyệt thứ nhất được hoàn thành. Vòng này đảm bảo phần tử lớn nhất, 5, được đặt vào đúng vị trí của nó.

Vòng duyệt thứ hai bắt đầu bằng việc so sánh 2 và 3. Vì chúng ở đúng thứ tự cần sắp nên ta so sánh 3 và 1. Vì 3 > 1 nên đổi chỗ chúng cho nhau ta được danh sách 2, 1, 3, 4, 5. Vì 3 < 4 nên các số này đã ở đúng thứ tự. Không cần so sánh thêm vì 5 đã được đặt đúng vị trí.

Vòng thứ ba bắt đầu bằng việc so sánh 2 với 1. Cần phải đổi chỗ chúng cho nhau vì 2 > 1. kết quả ta được 1, 2, 3, 4, 5. Vì 2 < 3 nên hai phần tử này đã ở đúng thứ tự. Không cần so sánh tiếp vì các phần tử 4 và 5 đã ở đúng vị trí của chúng.

Vòng thứ tư chỉ cần một phép so sánh giữa 1 và 2. Vì 1 < 2 nên chúng đã ở đúng thứ tự. Thuật toán sắp xếp kiểu nổi bọt kết thúc.

Các bước của thuật toán này được biểu thị trên Hình 2. ■



***Hình** 2.* Các bước sắp xếp kiểu nổi bọt

Mô tả giả mã của cách sắp xếp kiểu nổi bọt được cho trong Thuật toán 1.

Bây giờ ta xem xét mức độ hiệu quả của sắp xếp kiểu nổi bọt. Vì trong vòng thứ i cần phải thực hiện $(n - i)$ phép so sánh, nên tổng các pbép so sánh trong sắp xếp kiểu nổi bọt là :

$$(n - 1) + (n - 2) + \dots + 2 + 1.$$

Dễ dàng thấy tổng này có giá trị bằng $n(n - 1)/2$. Do đó, sắp xếp kiểu nổi bọt dùng $n(n - 1)/2$ phép so sánh để sắp thứ tự một danh sách n

phần tử. (Ta nhận thấy rằng sắp xếp kiểu nổi bọt dùng quá nhiều phép so sánh vì nó tiếp tục cả khi danh sách đã được sắp hoàn toàn tại một bước trung gian nào đó). Vì thế thuật toán sắp xếp nổi bọt có độ phức tạp trong trường hợp xấu nhất là $O(n^2)$. Vì với mọi số dương thực c, ta luôn có $n(n-1)/2 > cnlogn$, với n đủ lớn, ta suy ra sắp xếp kiểu nổi bọt không có $O(nlogn)$ như là độ phức tạp thời gian trong trường hợp xấu nhất.

THUẬT TOÁN 1. SẮP XẾP NỔI BỌT.

procedure *bubblesort* $(a_1, a_2, \ldots, a_n)$

for $i := 1$ **to** $n - 1$

begin

  for $j := 1$ *to* $n - i$

    if $a_j > a_{j+1}$ **then** đổi chỗ a_j và a_{j+1}

end

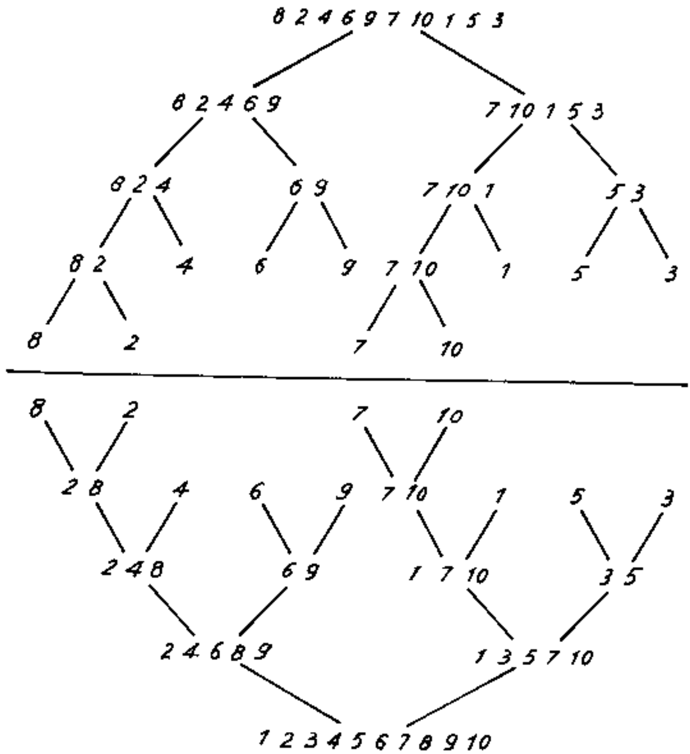
  $\{a_1, a_2, \ldots, a_n$ được sắp theo thứ tự tăng dần$\}$

SẮP XẾP KIỂU HOÀ NHẬP

Nhiều thuật toán sắp xếp có độ phức tạp trong trường hợp xấu nhất đạt được giá trị tối ưu, tức là $O(nlogn)$ phép so sánh để sắp xếp n phần tử. Chúng ta sẽ trình bày một trong những thuật toán này, đó là thuật toán sắp xếp kiểu hoà nhập. Trước tiên ta sẽ minh họa thuật toán này bằng một ví dụ.

Ví dụ 3. Chúng ta sẽ sắp xếp danh sách 8, 2, 4, 6, 9, 7, 10, 1, 5, 3 theo thuật toán sắp xếp kiểu hoà nhập. Theo thuật toán này trước tiên ta chia đôi liên tiếp danh sách đã cho thành hai danh sách con. Dãy các danh sách con ứng với ví dụ này được biểu diễn bằng cây nhị phân cân đối có chiều cao hằng 4 (Xem nửa trên của Hình 3).

Việc sắp xếp được tiến hành bằng cách hoà nhập lần lượt các cặp danh sách. Ở bước đầu tiên, hai phần tử được hòa vào một danh sách theo thứ tự tăng dần. Sau đó sẽ lần lượt hoà nhập các cặp danh sách cho tới khi toàn bộ danh sách được xếp đặt theo thứ tự tăng dần. Dãy các danh sách được hoà nhập theo thứ tự tăng dần được hiểu diễn bằng cây nhị phân cân đối chiều cao bằng 4 (Cây này được trình diễn lộn ngược ở nửa dưới của Hình 3).



Hình 3. Sắp xếp kiểu hoà nhập danh sách 8, 2, 4, 6, 9, 7, 10, 1, 5, 3.

Nói chung, sắp xếp kiểu hoà nhập được thực hiện bằng cách phân đôi liên tiếp các danh sách thành hai danh sách con có độ dài bằng nhau (hoặc hơn kém nhau một phần tử) cho tới khi mỗi danh sách con chỉ gồm một phần tử. Dãy danh sách con này có thể biểu diễn bằng cây nhị phân cân đối. Thủ tục tiếp tục bằng cách hoà nhập lần lượt các cặp danh sách đã có thứ tự tăng dần thành một danh sách lớn với các phần tử được sắp xếp theo thứ tự tăng dần cho tới khi toàn bộ danh sách ban đầu được sắp theo thứ tự tăng dần. Dãy danh sách hoà nhập được biểu diễn bằng cây nhị phân cân đối.

Chúng ta có thể mô tả bằng đệ quy thuật toán sắp xếp kiểu hoà nhập. Để sắp xếp theo kiểu hoà nhập, ta chia danh sách thành hai danh sách con có số phần tử bằng nhau hoặc gần bằng nhau, sắp xếp mỗi danh sách con bằng thuật toán sắp xếp kiểu hoà nhập và sau đó hoà nhập hai danh sách này lại. Chúng tôi để lại cho đọc giả hoàn thành phiên bản hồi quy của thủ tục sắp xếp kiểu hoà nhập.

Thuật toán có hiệu quả để hoà nhập hai danh sách có thứ tự thành một danh sách có thứ tự lớn hơn là rất cần cho thuật toán sắp xếp kiểu hoà nhập. Bây giờ ta sẽ mô tả thuật toán như vậy.

Ví dụ 4. Chúng ta sẽ mô tả cách hoà nhập danh sách có thứ tự 2, 3, 5, 6 và 1, 4. Bảng 1 minh họa các bước chúng ta sử dụng.

Trước tiên so sánh hai phần tử nhỏ nhất trong hai danh sách là 2 và 1. Vì 1 nhỏ hơn nên đặt nó vào đầu của danh sách hoà nhập và xóa nó khỏi danh sách thứ hai. Cuối giai đoạn này danh sách đầu là 2, 3, 5, 6 và danh sách thứ hai là 4 còn danh sách trộn là 1.

Tiếp theo so sánh 2 và 4, hai phần tử nhỏ nhất của hai danh sách. Vì 2 là nhỏ hơn nên gộp 2 vào danh sách hoà nhập và xóa nó khỏi danh sách thứ nhất. Cuối giai đoạn này danh sách đầu là 3, 5, 6 và danh sách thứ hai là 4 còn danh sách hoà nhập là 1, 2.

BẢNG 1. Hoà nhập hai danh sách 2, 3, 5, 6 và 1, 4

Danh sách 1	Danh sách 2	Danh sách hoà nhập	So sánh
2 3 5 6	1 4		$1 < 2$
2 3 5 6	4	1	$2 < 4$
3 5 6	4	1 2	$3 < 4$
5 6	4	1 2 3	$4 < 5$
5 6		1 2 3 4	
		1 2 3 4 5 6	

Tiếp tục so sánh 3 và 4 là hai phần tử nhỏ nhất của hai danh sách. Vì 3 nhỏ hơn nên thêm 3 vào danh sách hoà nhập, rồi xóa nó khỏi danh sách 1. Cuối giai đoạn này danh sách đầu là 5, 6 và danh sách thứ hai là 4 còn danh sách hòa nhập là 1, 2, 3.

Bây giờ tiếp tục so sánh 4 và 5 là hai phần tử nhỏ nhất của hai danh sách. Vì 4 là nhỏ hơn nên thêm 4 vào danh sách hoà nhập, rồi xóa nó khỏi danh sách 2. Cuối giai đoạn này danh sách đầu là 5, 6, danh sách thứ hai rỗng còn danh sách hoà nhập là 1, 2, 3, 4.

Cuối cùng vì danh sách thứ hai rỗng nên tất cả các phần tử của danh sách 1 được nối vào cuối của danh sách hoà nhập theo thứ tự mà chúng có ở danh sách 1. Kết quả chúng ta nhận được danh sách được sắp theo thứ tự tăng dần 1, 2, 3, 4, 5, 6.

■

Bây giờ chúng ta nghiên cứu bài toán tổng quát về việc hoà nhập hai danh sách có thứ tự L_1 và L_2 vào một danh sách có thứ tự L. Thủ tục hoà nhập như sau. L được khởi tạo là một danh sách rỗng. So sánh hai phần tử nhỏ nhất của hai danh sách. Đặt phần tử nhỏ hơn trong hai phần tử này vào cuối danh sách L, và xóa nó khỏi danh sách mà nó đã có mặt. Tiếp theo, nếu một trong hai danh sách L_1 và L_2 là rỗng thì nối danh sách kia (danh sách không rỗng) vào cuối L. Thủ tục hoà nhập kết thúc. Nếu cả L_1 và L_2 đều không rỗng thì lặp lại quá trình trên. Dạng giả mã của thủ tục này được cho trong Thuật toán 2.

Bây giờ chúng ta sẽ đánh giá số phép so sánh cần dùng trong thủ tục hoà nhập hai danh sách có thứ tự, một thủ tục cơ bản của thuật toán sắp xếp hoà nhập. Mỗi lần có một phép so sánh một phần tử của L_1 với một phần tử của L_2 và ghi một phần tử vào danh sách hoà nhập L. Tuy nhiên khi L_1 hoặc L_2 là rỗng thì không cần phải so sánh nữa. Vì thế Thuật toán có 2 hiệu lực kém nhất khi có $m + n - 2$ phép so sánh được thực hiện, trong đó m và n tương ứng là số các phần tử của L_1 và L_2, mà mỗi danh sách vẫn còn một phần tử. Ta cần phải làm một so sánh nữa thì sẽ có một danh sách rỗng. Vì thế Thuật toán 2 dùng không quá $m + n - 1$ phép so sánh. Tất cả những điều vừa nói trên là nội dung của bổ đề sau.

BỔ ĐỀ 1. Hai danh sách được sắp với m và n phần tử có thể được hoà nhập vào một danh sách được sắp khi dùng không quá $m + n - 1$ phép so sánh.

Đôi khi hai danh sách được sắp với độ dài m và n có thể được hoà nhập với nhau mà chỉ cần số phép so sánh ít hơn $m + n$ rất nhiều. Chẳng hạn khi $m = 1$ thủ tục tìm kiếm nhị phân có thể dùng để đặt một phần tử của danh sách đầu vào danh sách thứ hai. Khi đó chỉ cần $\lceil \log n \rceil$ phép so sánh, nhỏ hơn nhiều so với $m + n - 1 = n$, với $m = 1$. Mặt khác, với một số giá trị của m và n, Bổ đề 1 cho ta một giới hạn khá đỉ tốt nhất. Điều này có nghĩa là có những danh sách với những giá trị của m và n nào đó không thể sắp xếp theo kiểu hoà nhập với ít hơn $(m + n - 1)$ phép so sánh. (Xem Bài tập 7 ở cuối tiết này).

THUẬT TOÁN 2. HOÀ NHẬP HAI DANH SÁCH.

procedure *merge* (L_1, L_2 : danh sách)

L : = danh sách rỗng.

while cả L_1 và L đều không rỗng

begin

xóa phần tử nhỏ hơn trong hai phần tử đầu của L_1 va L_2 khỏi danh sách chứa nó, và đặt nó vào cuối của danh sách hoà nhập L.

if việc xóa một phần tử làm cho một danh sách trở thành rỗng

then xóa tất cả các phần tử khỏi danh sách kia, và nối chúng vào cuối L.

end

{L là danh sách hoà nhập với các phần tử được sắp theo thứ tăng dần}

Bây giờ chúng ta có thể phân tích độ phức tạp của thuật toán sắp xếp kiểu hoà nhập. Thay cho việc nghiên cứu bài toán tổng quát chúng ta giả sử số phần tử của danh sách là lũy thừa của 2, tức là $n = 2^m$. Điều đó làm cho việc phân tích bớt cồng kềnh hơn, nhưng khi $n \neq 2^m$, dùng những thay đổi khác nhau cũng sẽ cho một đánh giá như thế.

Đầu tiên ta chia danh sách làm hai danh sách con ở mức 1 của cây nhị phân, mỗi danh sách có 2^{m-1} phần tử. Tiếp tục chia hai danh sách con này thành bốn danh sách con ở mức 2, mỗi danh sách có 2^{m-2} phần tử, cứ như thế ta lại phân đôi tiếp. Nói chung , sẽ có 2^{k-1} danh sách ở mức $k - 1$, mỗi danh sách có 2^{m-k+1} phần tử. Các danh sách ở mức $k - 1$ được phân chia thành 2^k danh sách ở mức k, mỗi danh sách có 2^{m-k} phần tử. Kết thúc quá trình phân chia, ở mức m ta có 2^m danh sách, mỗi danh sách có đúng một phần tử.

Chúng ta bắt đầu hoà nhập mỗi cặp của 2^m danh sách có 1 phần tử thành 2^{m-1} danh sách ở mức $(m - 1)$ mỗi danh sách có 2 phần tử. Việc hoà nhập mỗi cặp cần đúng 1 phép so sánh.

Thủ tục tiếp tục sao cho ở mức k ($k = m, m - 1, m - 2, \ldots, 3, 2, 1$), 2^k danh sách, mỗi danh sách có 2^{m-k} phần tử, được hoà nhập thành 2^{k-1} danh sách mỗi danh sách có 2^{m-k+1} phần tử ở mức $k - 1$. Khi đó ta cần tất cả 2^{k-1} phép hoà nhập hai danh sách, mỗi danh sách có 2^{m-k} phần tử. Theo bổ đề 1 mỗi phép hoà nhập cần nhiều nhất $2^{m-k} + 2^{m-k} - 1 = 2^{m-k+1} - 1$ phép so sánh. Vì thế để chuyển từ mức k xuống mức $k - 1$ cần $2^{k-1}(2^{m-k+1} - 1)$ phép so sánh. Lấy tổng tất cả các đánh giá này ta sẽ nhận được số các phép so sánh cần thiết cho thuật toán hoà nhập, nhiều nhất là

$$\sum_{k=1}^{m} 2^{k-1}(2^{m-k+1} - 1) = \sum_{k=1}^{m} 2^m - \sum_{k=1}^{m} 2^{k-1}$$

$$= m\ 2^m - (2^m - 1)$$

$$= n\log n - n + 1.$$

vì $m = \log n$ và $n = 2^m$.

Vậy thuật toán hoà nhập đạt được đánh giá big – O tốt hơn về số các phép so sánh cần thiết. Điều này được phát biểu bằng định lý sau.

ĐỊNH LÝ 2. Số các phép so sánh cần thiết trong thuật toán sắp xếp kiểu hoà nhập một danh sách n phần tử là $O(n\log n)$.

Trong phần bài tập chúng ta sẽ mô tả thuật toán nữa, cũng rất có hiệu quả, đó là thuật toán sắp xếp nhanh (quick sort).

BÀI TẬP

1. Dùng thuật toán sắp xếp kiểu nổi bọt hãy sắp xếp danh sách 3, 1, 5, 7, 4. Hãy chỉ rõ các danh sách nhận được ở mỗi bước.

2. Dùng thuật toán sắp xếp kiểu nổi bọt hãy sắp xếp danh sách *d, f, m, k, a, b*. Hãy chỉ rõ các danh sách nhận được ở mỗi bước.

3*. Hãy sửa lại thuật toán sắp xếp kiểu nổi bọt sao cho nó dừng khi không cần sự đổi chỗ nào nữa. Hãy chỉ ra rằng phiên bản này hiệu quả hơn phiên bản của thuật toán đã cho ở dạng giả mã.

4. Dùng thuật toán sắp xếp kiểu hoà nhập hãy sắp xếp danh sách 4, 3, 2, 5, 1, 8, 7, 6. Hãy chỉ rõ tất cả các bước được sử dụng trong thuật toán.

5. Dùng thuật toán sắp xếp kiểu hoà nhập hãy sắp xếp danh sách *b, d, a, f, g, h, z, p, o, k*. Hãy chỉ rõ tất cả các bước được sử dụng trong thuật toán.

6. Cần bao nhiêu phép so sánh để hoà nhập các danh sách sau đây bằng Thuật toán 2?

a) 1, 3, 5, 7, 9 ; 2, 4, 6, 8, 10

b) 1, 2, 3, 4, 5 ; 6, 7, 8, 9, 10

c) 1, 5, 6, 7, 8 ; 2, 3, 4, 9 10

7. Chỉ ra rằng có những danh sách với m và n phần tử sao cho chúng không thể hoà nhập thành một danh sách được sắp bằng Thuật toán 2 với ít hơn $m + n - 1$ phép so sánh.

8*. Tính số các phép so sánh ít nhất cần thiết để hoà nhập hai danh sách bất kỳ đã có thứ tự tăng dần thành một danh sách cũng theo thứ tự tăng dần, nếu số phần tử của chúng là :

a) 1, 4? b) 2, 4?

c) 3, 4? d) 4, 4?

Sắp xếp kiểu chọn lọc *bắt đầu bằng việc tìm phần tử nhỏ nhất trong danh sách. Phần tử này được chuyển lên đầu danh sách. Sau đó ta lại tìm phần tử nhỏ nhất trong các phần tử còn lại rồi đặt nó ở vị trí thứ hai. Thủ tục này được lặp lại cho tới khi toàn bộ danh sách được sắp xếp.*

9. Sắp xếp danh sách sau đây bằng thuật toán sắp xếp kiểu chọn lọc :

a) 3, 5, 4, 1, 2. b) 5, 4, 3, 2, 1.

c) 1, 2, 3, 4, 5.

10. Hãy viết thuật toán sắp xếp kiểu chọn lọc ở dạng giả mã.

11. Cần bao nhiêu phép so sánh để sắp xếp n phần tử bằng cách sắp xếp kiểu chọn lọc.

Sắp xếp nhanh *là một thuật toán có hiệu quả tốt. Để sắp xếp* $a_1, a_2, \ldots, a_n$, *thuật toán này bắt đầu bằng việc lấy phần tử đầu tiên* a_1 *và tạo hai danh sách con, danh sách đầu chứa các phần tử nhỏ hơn* a_1 *theo thứ tự xuất hiện của chúng, danh sách thứ hai chứa các phần tử lớn hơn* a_1 *theo thứ tự xuất hiện của chúng. Khi đó* a_1 *được đặt ở cuối của danh sách đầu. Thủ tục này được lặp lại một cách đệ quy cho mỗi danh sách con cho tới khi mỗi danh sách chứa chỉ một phần tử theo thứ tự xuất hiện của chúng.*

12. Sắp xếp danh sách 3, 5, 7, 8, 1, 9, 2, 4, 6 bằng thuật toán sắp xếp nhanh.

13. Giả sử a_1, a_2, ..., a_n là danh sách n số thực phân biệt. Cần bao nhiêu phép so sánh để tạo ra hai danh sách con từ danh sách ban đầu sao cho danh sách con thứ nhất chứa các phần tử nhỏ hơn a_1, danh sách con thứ hai bao gồm các phần tử lớn hơn a_1?

14. Mô tả thuật toán sắp xếp nhanh dưới dạng giả mã.

15. Tính số lớn nhất các phép so sánh cần thiết để sắp xếp một danh sách có 4 phần tử bằng thuật toán sắp xếp nhanh.

16. Tìm số ít nhất các phép so sánh cần thiết để sắp xếp một danh sách có 4 phần tử bằng thuật toán sắp xếp nhanh.

17. Xác định độ phức tạp trong trường hợp xấu nhất của thuật toán sắp xếp nhanh phụ thuộc vào số các phép so sánh được dùng.

18*. Hãy chỉ ra rằng $\log n!$ là lớn hơn $(n\log n)/4$ với $n > 4$. (*Gợi ý* :Bắt đầu hằng bất đẳng thức $n! > n(n-1)(n-2) \ldots \lceil n/2 \rceil$).

19*. Hãy viết thuật toán sắp xếp kiểu hoà nhập dưới dạng giả mã.

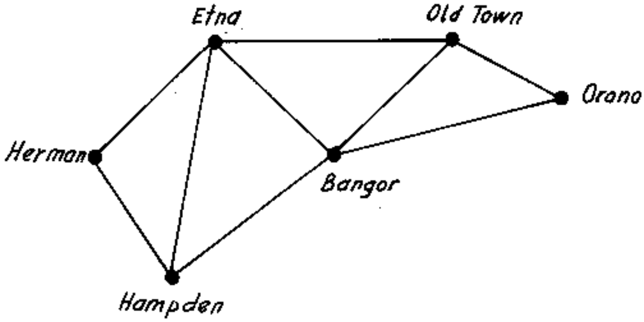
8.5. CÂY KHUNG

MỞ ĐẦU

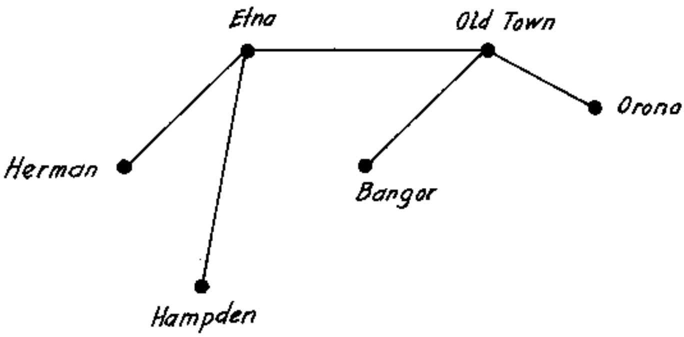
Hệ thống đường giao thông ở Maine được hiểu thị bằng đồ thị đơn trên Hình 1(a). Cách duy nhất để những con đường có thể đi lại được vào mùa đông là phải cào tuyết thường xuyên. Chính quyền địa phương muốn cào tuyết một số ít nhất các con đường sao cho luôn luôn có đường thông suốt nối hai thành phố bất kỳ. Có thể làm điều đó bằng cách nào?

Cần phải cào tuyết ít nhất năm con đường mới đảm bảo có đường đi giữa hai thành phố bất kỳ. Hình 1(b) biểu thị một tập hợp các con đường như vậy. Ta nhận thấy đồ thị con biểu diễn các con đường này là một cây, vì nó liên thông và chứa sáu đỉnh, năm cạnh.

Bài toán trên được giải bằng một đồ thị con có một số tối thiểu các cạnh và chứa tất cả các đỉnh của đồ thị xuất phát. Đồ thị như thế phải là một cây.



a)



b)

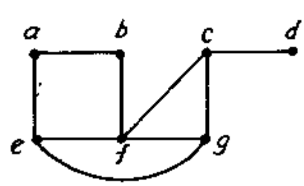
Hình 1. (a) Hệ thống đường và (b) Tập các con đường phải cào tuyết

ĐỊNH NGHĨA 1. Cho G là một đơn đồ thị. Một cây được gọi là *cây khung* của G nếu nó là một đồ thị con của G và chứa tất cả các đỉnh của G.

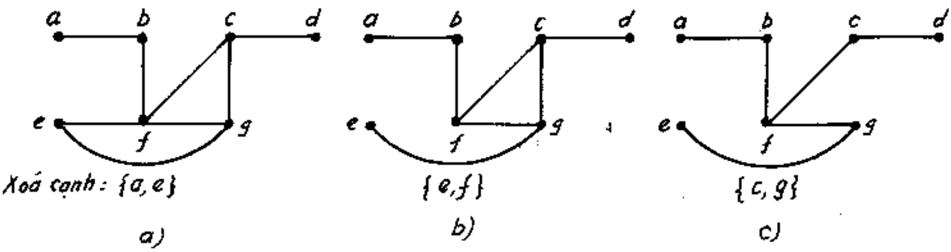
Một đơn đồ thị có cây khung sẽ là một đồ thị liên thông vì có đường đi trong cây khung giữa hai đỉnh bất kỳ. Điều ngược lại cũng đúng, tức là mọi đồ thị liên thông đều có cây khung. Chúng ta xét một ví dụ trước khi chứng minh kết quả này.

Ví dụ 1. Tìm cây khung của đồ thị G trên Hình 2.

Giải: Đồ thị G liên thông, nhưng không là một cây vì nó chứa chu trình đơn. Xóa cạnh $\{a, e\}$. Điều đó loại được một chu trình, đồ thị con nhận được vẫn còn liên thông và chứa tất cả các đỉnh của G. Tiếp theo, xóa cạnh $\{e, f\}$ sẽ loại được một chu trình nữa. Cuối cùng xóa cạnh $\{c, g\}$ sẽ sinh được một đơn đồ thị không có chu trình. Đồ thị này



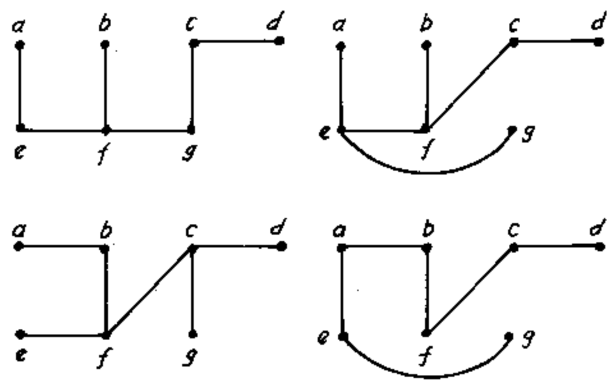
Hình 2. Đồ thị đơn G



Hình 3. Tạo ra cây khung của G bằng cách xóa các cạnh tạo ra chu trình đơn.

là cây khung vì nó là cây và chứa tất cả các đỉnh của G. Dãy các cạnh bị xóa đi khi tạo ra cây khung được minh họa trên Hình 3.

Cây trên Hình 3 không phải là cây khung duy nhất của G. Ví dụ, mỗi một trong bốn cây trên Hình 4 đều là cây khung của G.



Hình 4. Các cây khung của G.

ĐỊNH LÝ 1. Một đơn đồ thị là liên thông nếu và chỉ nếu nó có cây khung.

Chứng minh. Trước tiên giả sử đồ thị G có cây khung $T.T$ chứa tất cả các đỉnh của G. Hơn nữa có đường đi trong T giữa hai đỉnh bất kỳ. Vì T là đồ thị con của G nên có đường đi trong G giữa hai đỉnh bất kỳ của nó. Do đó, G là liên thông.

Bây giờ chúng ta sử dụng G là liên thông. Nếu G không phải là một cây thì nó phải có chu trình đơn. Xóa đi một cạnh của một trong các chu trình đơn này. Đồ thị nhận được chứa một số cạnh ít hơn nhưng vẫn còn chứa tất cả các đỉnh của G và vẫn liên thông. Nếu đồ thị con này không là cây thì nó còn chứa chu trình đơn. Cũng giống như trên, ta lại xóa đi một cạnh của chu trình đơn. Lặp lại quá trình này cho đến khi không còn chu trình đơn. Điều này là có thể vì chỉ có một số hữu hạn các cạnh trong đồ thị. Quá trình kết thúc khi không còn chu trình đơn trong đồ thị nhận được. Cây được tạo ra vì đồ thị vẫn còn liên thông khi xóa đi các cạnh. Cây này là cây khung vì nó chứa tất cả các đỉnh của G.

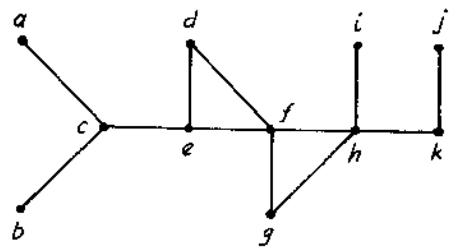
NHỮNG THUẬT TOÁN XÂY DỰNG CÂY KHUNG

Cách chứng minh Định lý 1 cho ta một thuật toán tìm cây khung bằng cách xóa đi các cạnh khỏi các chu trình đơn. Thuật toán này là không hiệu quả vì nó đòi hỏi phải nhận biết được các chu trình đơn. Thay cho việc xây dựng cây khung bằng cách loại bỏ các cạnh, cây khung có thể được xây dựng bằng cách lần lượt ghép các cạnh. Chúng ta sẽ giới thiệu hai thuật toán dựa trên nguyên tắc này.

Chúng ta sẽ xây dựng cây khung của một đồ thị liên thông bằng phương pháp **tìm kiếm ưu tiên chiều sâu**. Chúng ta sẽ tạo một cây có gốc và cây khung sẽ là đồ thị vô hướng nền của cây có gốc này. Chọn tùy ý một đỉnh của đồ thị làm gốc. Xây dựng đường đi từ đỉnh này bằng cách lần lượt ghép các cạnh vào 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 các cạnh vào đường đi chừng nào không thể thêm được nữa thì thôi. Nếu đường đi qua tất cả các đỉnh của đồ thị thì cây do đường đi này tạo nên sẽ là cây khung. Nhưng nếu đường đi không đi qua tất cả các đỉnh thì cần thêm các cạnh khác vào đường đi. Lùi lại đỉnh trước đỉnh cuối cùng của đường đi, và nếu có thể, 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 lại hai đỉnh trên đường đi và thử xây dựng đường đi mới. Lập lại thủ tục này, bắt đầu từ đỉnh cuối cùng được viếng thăm lùi theo đường đi mỗi lần một đỉnh, xây dựng đường đi mới càng dài càng tốt cho tới khi nào không thể thêm được một cạnh nào nữa. Vì đồ thị có hữu hạn cạnh và là liên thông nên quá trình đó sẽ kết thúc và tạo được cây khung. Mỗi đỉnh mà tại đó đường đi kết thúc ở mỗi giai đoạn của thuật toán sẽ là lá trong cây có gốc. Mỗi đỉnh tại đó đường đi bắt đầu từ đó sẽ là một đỉnh trong. Độc giả cần nhận thấy bản chất đệ quy của thủ tục này. Cũng lưu ý là nếu các đỉnh của đồ thị là được sắp thì việc chọn các cạnh ở mỗi giai đoạn của thủ tục là hoàn toàn xác định khi chúng ta luôn chọn đỉnh đầu tiên của dãy sắp nếu có thể. Tuy nhiên, chúng ta thường không sắp xếp các đỉnh của đồ thị.

Tìm kiếm ưu tiên chiều sâu cũng được gọi là thủ tục **quay lui** (hay **lần ngược**), vì nó quay lại đỉnh đã viếng thăm trước trên đường đi. Ví dụ sau đây minh họa thủ tục quay lui.

Ví dụ 2. Dùng thuật toán ưu tiên chiều sâu, tìm cây khung của đồ thị G trên Hình 5.

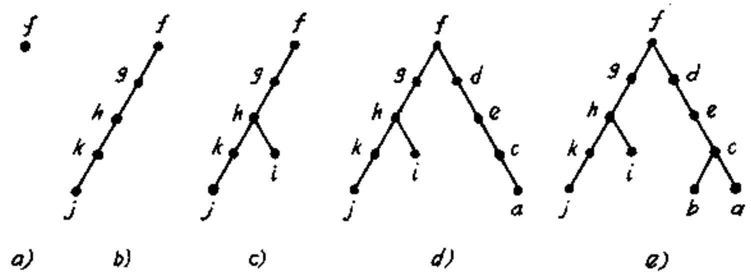


***Hình** 5.* Đồ thị G

Giải. Các bước dùng trong thuật toán ưu tiên chiều sâu để xây dựng cây khung của G được biểu thị trên Hình 6. Chúng ta xuất phát từ một đỉnh tùy ý chẳng hạn đỉnh f. Đường đi được xây dựng bằng cách lần lượt ghép, càng nhiều càng tốt, các cạnh liên thuộc với các đỉnh còn chưa thuộc đường đi. Điều đó tạo ra được đường đi f, g, h, k, j (lưu ý là có thể đường đi khác được xây dựng). Lùi lại k. Không có đường đi bắt đầu từ k chứa các đỉnh chưa được viếng thăm. Vì thế lùi tới h. Từ h có đường đi h, i. Sau đó lùi về h và tiếp tục lùi về f. Từ f có đường đi f, d, e, c, a. Ta lại lùi về c và xây dựng đường đi c, b. Thủ tục này đã xây dựng được cây khung.

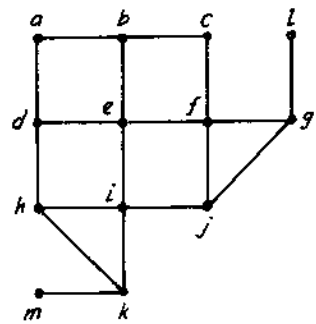
■

Chúng ta có thể xây dựng cây khung của một đơn đồ thị bằng thuật toán **tìm kiếm ưu tiên chiều rộng**. Một lần nữa, cây có gốc sẽ được xây dựng, và đồ thị vô hướng nền của cây có gốc sẽ tạo nên cây khung. Chọn một đỉnh bất kỳ của đồ thị làm gốc. Sau đó ghép vào tất cả các cạnh liên thuộc với đỉnh này. Các đỉnh mới ghép vào trong giai đoạn này trở thành các đỉnh ở mức 1 của cây khung. Sắp xếp chúng theo một thứ tự tùy ý. Tiếp theo, với mỗi đỉnh ở mức 1, được viếng thăm theo thứ tự vừa sắp ở trên, ta ghép tất cả các cạnh liên thuộc với nó vào cây mà không tạo ra chu trình. Sắp xếp các đỉnh con của mỗi đỉnh ở mức 1 theo một trật tự nào đó. Quá trình này tạo ra các đỉnh ở mức 2 của cây. Tiếp tục làm lại thủ tục này cho tới khi tất cả các đỉnh của đồ thị được ghép vào cây. Thủ tục này kết thúc vì chỉ có một số hữu hạn các cạnh của đồ thị. Cây khung được tạo ra vì xây dựng được cây chứa tất cả các đỉnh của đồ thị. Dưới đây là một ví dụ minh họa thuật toán tìm kiếm ưu tiên chiều rộng.



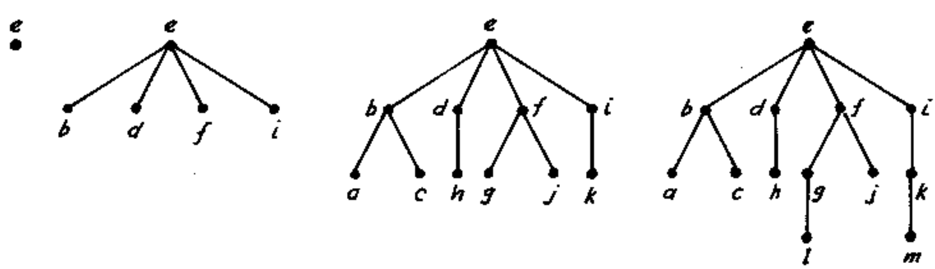
Hình 6. Tìm kiếm ưu tiên chiều sâu của G.

Ví dụ 3. Dùng thuật toán ưu tiên chiều rộng, tìm cây khung của đồ thị trên Hình 7.



Hình 7. Đồ thị G.

Giải: Các bước của thủ tục tìm kiếm ưu tiên chiều rộng được biểu diễn trên Hình 8. Ta chọn đỉnh e làm gốc của cây. Sau đó ta thêm các cạnh liên thuộc và tất cả các đỉnh liền kề với e, tức là các cạnh từ e tới b, d, f, và i được ghép vào. Vậy ở mức 1 của cây có 4 đỉnh. Tiếp theo ghép các cạnh từ các đỉnh ở mức 1 nối với các đỉnh còn chưa ở trong cây. Vì thế các cạnh từ b tới a và c được ghép vào, cũng như thế các cạnh từ d tới h, từ f tới j và g, và từ i tới k. Các đỉnh mới a, c, h, g, j, k ở mức 2 của cây. Tiếp theo ghép các cạnh từ các đỉnh này nối với các đỉnh còn chưa thuộc vào cây. Tức là ghép thêm các cạnh từ g tới l và từ k tới m.



Hình 8. Tìm kiếm ưu tiên chiều rộng của G.

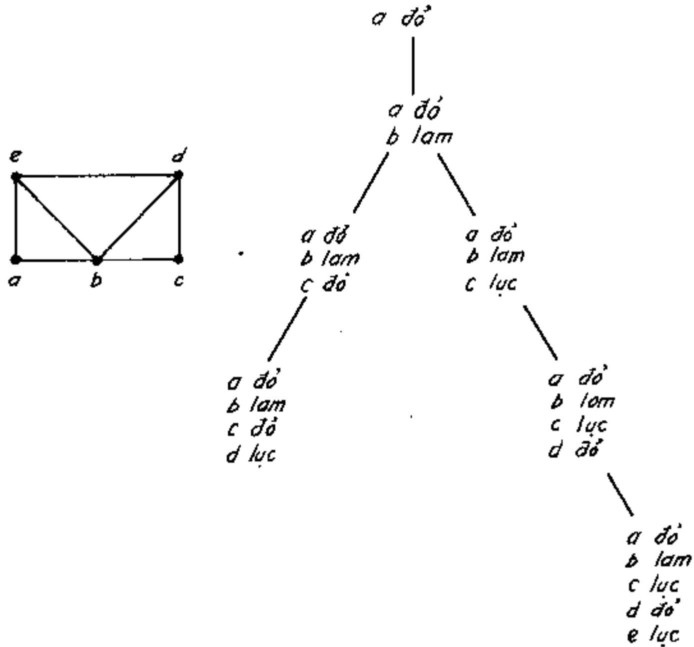
KỸ THUẬT QUAY LUI

Có nhiều bài toán chỉ có thể giải được bằng cách nghiên cứu thấu đáo tất cả các lời giải có thể. Một phương pháp nghiên cứu một cách có hệ thống các lời giải là dùng cây quyết định, trong đó mỗi đỉnh trong biểu thị một quyết định, và mỗi lá biểu thị một lời giải có thể. Để tìm nghiệm bằng kỹ thuật quay lui, trước tiên ta tạo ra một dãy quyết định, càng dài càng tốt, để tiến tới lời giải. Dãy các quyết định có thể được biểu diễn bằng một đường đi trong cây quyết định. Mỗi khi biết được không thể có lời giải từ bất kỳ dãy quyết định tiếp theo nào đó, ta quay lui lại đỉnh cha của đỉnh hiện thời, hướng tới lời giải bằng dãy các quyết định khác, nếu có thể, Thủ tục tiếp tục cho tới khi tìm được lời giải hoặc là kết luận không có lời giải. Những ví dụ sau đây minh họa ích lợi của kỹ thuật quay lui.

Ví dụ 4. *Tô màu đồ thị*. Kỹ thuật quay lui có thể được sử dụng như thế nào để khẳng định xem một đồ thị có thể được tô bằng n màu hay không ?

Giải: Ta có thể giải bài toán này bằng kỹ thuật quay lui như sau. Trước tiên ta chọn một đỉnh a và gán cho nó màu 1. Sau đó chọn đỉnh b và nếu b không liền kề với a thì gán cho nó màu 1. Còn không thì gán màu 2 cho b. Tiếp theo ta đi tới đỉnh c. Dùng màu 1 cho c nếu có thể, còn không thì dùng màu 2 nếu có thể. Chỉ khi cả màu 1 và màu 2 đều không thể dùng được thl ta sẽ dùng màu 3. Tiếp tục quá trình này tới chừng nào còn có thể, để gán một trong n màu cho mỗi đỉnh mới lấy thêm, luôn dùng màu đầu tiên có thể dùng được trong danh sách. Nếu một đỉnh không thể tô bằng bất cứ màu nào trong n màu thì lùi lại đỉnh vừa được gán màu, thay đổi màu tô bằng màu có thể tiếp theo trong danh sách. Nếu không thể thay đổi cách tô màu cho đỉnh này thì ta lại lùi tới đỉnh được tô màu trước đó, mỗi lần lùi một bước, cho tới khi còn có thể thay đổi cách tô màu cho một đỉnh. Nếu tồn tại cách tô bằng n màu thì kỹ thuật quay lui sẽ sinh ra nó. (Tiếc thay thủ tục này lại cực kỳ không hiệu quả)

Đặc biệt, ta xem xét bài toán tô màu đồ thị trên Hình 9 bằng ba màu. Cây trên Hình 9 minh họa cách dùng kỹ thuật quay lui để tạo ra cách tô bằng ba màu. Theo thủ tục này màu đỏ được dùng đầu tiên, sau đó là màu lục cuối cùng là màu lam. Ví dụ đơn giản này tất nhiên là có thể giải mà không dùng thủ tục quay lui, nhưng nó là một minh họa tốt cho kỹ thuật này.



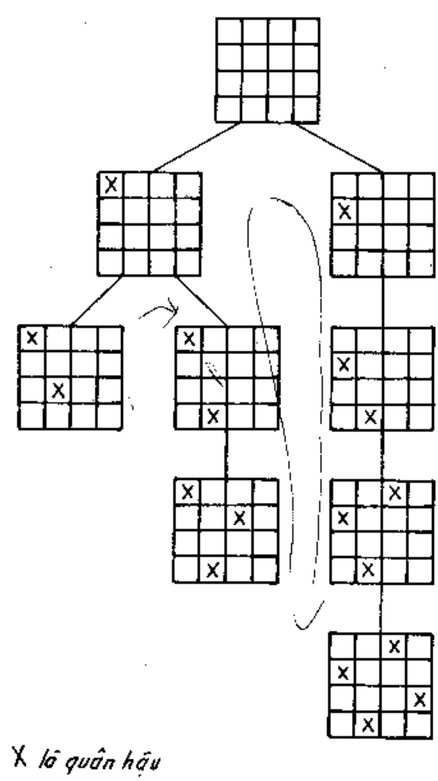
Hình 9. Tô màu đồ thị bằng kỹ thuật lần ngược.

Trong cây này đường đi xuất phát từ gốc a được gán màu đỏ, dẫn tới cách tô a đỏ, b lam, c đỏ, d lục. Không thể tô e bằng bất kỳ màu nào trong ba màu này khi a, b, c và d đã được tô màu bằng cách đó. Vì thế ta quay lui lại cha của đỉnh biểu thị cách tô này, nhưng không có cách nào khác để tô d, ta lần lên mức trên nữa. Khi đó thay đổi cách tô đỉnh c bằng màu lục. Chúng ta nhận được cách tô màu đồ thị bằng cách tô d đỏ và e lục.

Ví dụ 5. *Bài toán n quân hậu.* Hãy đặt n quân hậu lên bàn cờ $n \times n$ sao cho không có hai quân nào tấn công lẫn nhau. Có thể dùng kỹ thuật quay lui để giải bài toán này như thế nào?

Giải: Để giải bài toán này ta phải tìm n vị trí trên bàn cờ $n \times n$ sao cho không có hai trong các vị trí này trên cùng một hàng, cùng một cột hoặc trên cùng một đường chéo (đường chéo bao gồm tất cả các vị trí

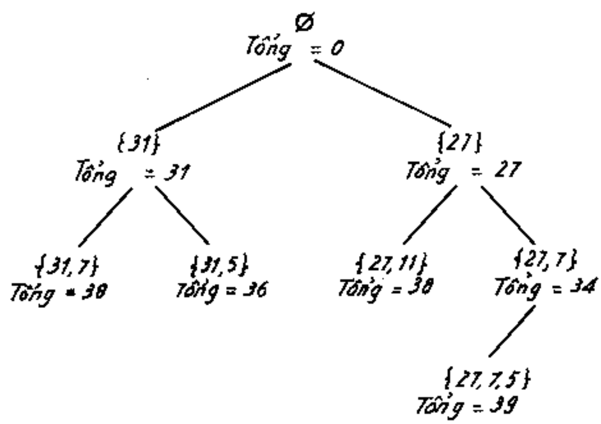
(i, j) trong đó $i + j = m$ với m nào đó, hoặc $i - j = m$ với m nào đó). Chúng ta sẽ dùng kỹ thuật quay lui để giải bài toán này. Xuất phát từ bàn cờ trống. Tại bước $k + 1$ ta thử đặt thêm một quân hậu vào cột $k + 1$ của bàn cờ, trong đó k quân hậu đã được đặt vào k cột đầu tiên. Trong các ô vuông ở cột $k + 1$ bắt đầu từ ô ở hàng 1, chúng ta tìm vị trí để đặt quân hậu này sao cho nó không nằm cùng hàng hoặc cùng đường chéo với các quân hậu đã có ở trên bàn cờ. (Ta đã biết chúng không thể nằm trên cùng cột). Nếu không thể tìm được vị trí để đặt quân hậu vào cột $k + 1$ thì ta trở lại cách đặt quân hậu ở cột k. Ta đặt nó vào vị trí có thể tiếp theo thuộc cột đó nếu một ô như vậy tồn tại. Nếu không tìm được một ô như thế ta tiếp tục quay lui lại cột trước nữa. Ví dụ, trên Hình 10 biểu diễn quá trình tìm nghiệm bằng kỹ thuật quay lui, trong trường hợp có 4 quân hậu. Trước tiên ta đặt một quân hậu vào hàng 1 cột 1. Sau đó đặt quân hậu thứ 2 vào cột 2 hàng 3. Khi đó không thể đặt quân hậu vào cột 3 được. Vì thế ta quay lại cột 2 và đặt lại quân hậu thứ 2 vào hàng 4 cột 2. Và do vậy có thể đặt quân hậu thứ 3 vào hàng 2 cột 3. Nhưng không thể tìm được chỗ cho quân hậu thứ 4. Điều này chứng tỏ không thể nhận được lời giải nào nếu đặt quân hậu thứ nhất ở hàng 1, cột 1. Quay lại bàn cờ trống, và đặt quân hậu ở hàng 2, cột 1. Tiếp tục quá trình ta tìm được lời giải như trên Hình 10.



Hình 10. Giải bài toán 4 quân hậu bằng kỹ thuật lần ngược.

■

Ví dụ 6. *Tổng các tập con.* Ta xét bài toán sau. Cho tập các số nguyên dương $x_1, x_2, \ldots, x_n$, hãy tìm tập con của tập này sao cho tổng các phần tử của nó bằng M. Giải bài toán này bằng kỹ thuật quay lui.



Hình 11. Tìm tập con có tổng bằng 39 nhờ kỹ thuật lần ngược.

Giải: Chúng ta xuất phát từ tập con chưa có số hạng nào. Tiếp tục xây dựng tập con này bằng cách lần lượt thêm các số hạng. Một số nguyên trong dãy đã cho được gộp vào tập con nếu tổng các phần tử của nó khi thêm phần tử này vào vẫn còn nhỏ hơn M. Nếu tổng đạt tới giá trị sao cho việc thêm bất kỳ phần tử nào nữa vào tập con ta đều nhận được tổng lớn hơn M, thì ta quay lui trở lại bằng cách trả lại phần tử cuối cùng vừa đặt vào tập con.

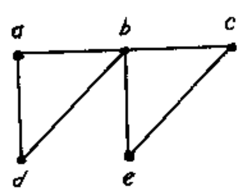
Hình 11 biểu thị kỹ thuật quay lui tìm tập con của dãy {31, 27, 15, 11, 7, 5} có tổng bằng 39.

BÀI TẬP

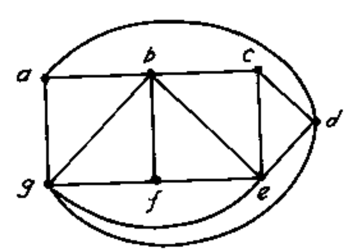
1. Cần phải xóa đi bao nhiêu cạnh khỏi đồ thị liên thông với n đỉnh và m cạnh để nhận được một cây khung?

Trong các Bài tập 2–6 hãy tìm cây khung của đồ thị đã cho bằng cách xóa đi các cạnh trong các chu trình đơn.

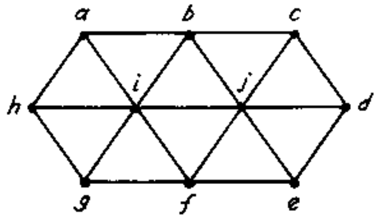
2.



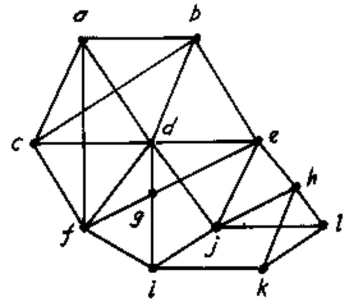
3.



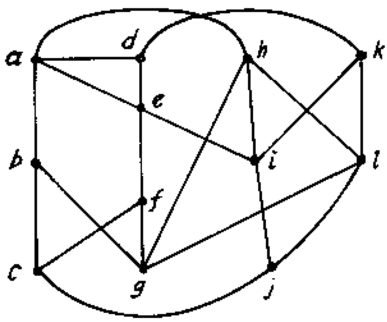
4.



5.



6.



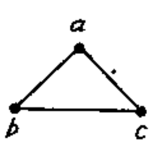
7. Hãy tìm cây khung cho mỗi đồ thị sau.

a) K_5 b) $K_{4,4}$ c) $K_{1,6}$

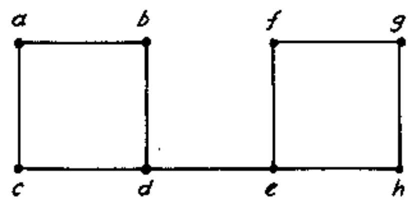
d) Q_3 e) C_5 f) W_5

Trong các Bài tập 8–10 hãy vẽ tất cả các cây khung của đồ thị đơn tương ứng.

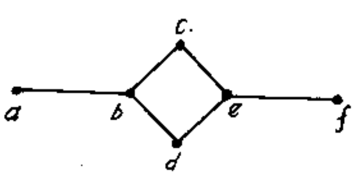
8.



9.



10.



11*. Mỗi đồ thị sau có bao nhiêu cây khung khác nhau?

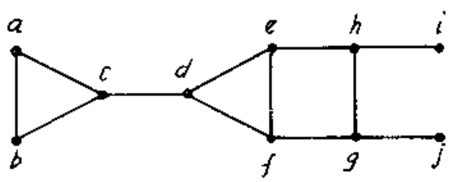
a) K_3 b) K_4 c) $K_{2,2}$ d) C_5

12*. Mỗi đồ thị sau có bao nhiêu cây khung không đẳng cấu?

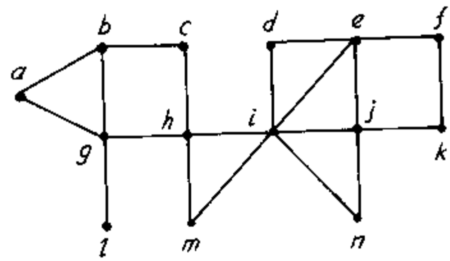
a) K_3 b) K_4 c) K_5

Trong các Bài tập 13–15 dùng kỹ thuật tìm kiếm ưu tiên chiều sâu hãy xác định cây khung cho các đồ thị đơn đã cho. Chọn a làm gốc của cây, và giả sử rằng các đỉnh được sắp theo thứ tự từ điển.

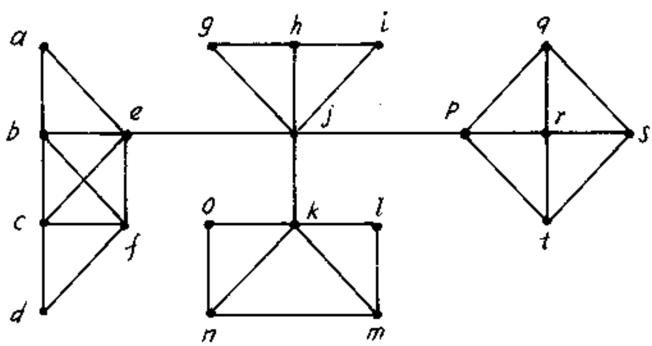
13.



14.

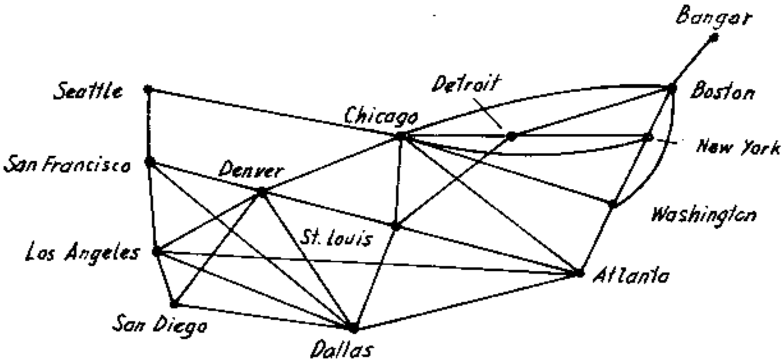


15.



16. Dùng kỹ thuật tìm kiếm ưu tiên chiều rộng hãy tạo cây khung cho các đồ thị đơn đã cho trong các Bài tập 13–15. Chọn a làm gốc của cây.

17. Giả sử rằng hãng hàng không cần giảm bớt lịch bay để tiết kiệm tiền. Nếu ban đầu các đường bay được minh họa trên hình vẽ dưới đây. Có thể hủy bỏ các chuyến bay nào mà vẫn giữ được giao thông giữa hai thành phố bất kỳ. (Nếu cần có thể tổ hợp các chuyến bay từ thành phố này đến thành phố khác)?



18. Khi nào một cạnh của đơn đồ thị liên thông cần phải có trong mọi cây khung của đồ thị này?

19. Đơn đồ thị liên thông nào có đúng một cây khung?

20. Hãy giải thích cách dùng kỹ thuật tìm kiếm ưu tiên chiều sâu và tìm kiếm ưu tiên chiều rộng để sắp xếp các đỉnh của một đồ thị liên thông.

21*. Hãy viết thủ tục tìm kiếm ưu tiên chiều sâu dưới dạng giả mã.

22*. Hãy viết thủ tục tìm kiếm ưu tiên chiều rộng dưới dạng giả mã.

23*. Hãy chứng minh rằng độ dài của đường đi ngắn nhất giữa các đỉnh v và u trong một đơn đồ thị liên thông bằng số mức của u trong cây khung ưu tiên chiều rộng của G với gốc v.

24. Dùng kỹ thuật quay lui, hãy tìm cách tô mỗi một trong các đồ thị trong các Bài tập 5–7 của Tiết 7.8 bằng ba màu.

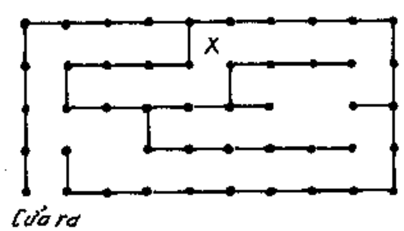
25. Dùng kỹ thuật quay lui để giải bài toán n quân hậu với các giá trị sau của n :

a) $n = 3$ b) $n = 5$ c) $n = 6$.

26. Dùng kỹ thuật quay lui tìm tập con, nếu nó tồn tại, của tập {27, 24, 19, 14, 11, 8} có tổng bằng :

a) 20 b) 41 c) 60

27. Hãy giải thích cách dùng kỹ thuật quay lui để tìm đường đi hoặc chu trình Hamilton trong một đồ thị.



28. a) Hãy giải thích cách dùng kỹ thuật quay lui để tìm đường ra khỏi mê cung, nếu cho biết vị trí xuất phát và vị trí cửa ra. Coi mê cung được chia thành các vị trí, trong đó tại mỗi vị trí tập các chuyển động có thể là {lên, xuống, phải, trái}.

b) Hãy tìm đường đi từ vị trí xuất phát được đánh dấu bằng X tới cửa ra (exit) trong mê cung ở hình bên.

Rừng khung *của đồ thị* G *là rừng chứa mọi đỉnh của* G *sao cho hai đỉnh thuộc cùng một cây của rừng nếu giữa chúng có một đường đi trong G.*

29. Chứng minh rằng mọi đơn đồ tbị hữu hạn đều có rừng khung.

30. Trong rừng khung của một đồ thị có bao nhiêu cây?

31. Cần phải bỏ đi bao nhiêu cạnh để tạo ra rừng khung của một đồ thị có n đỉnh m cạnh và c thành phần liên thông?

32. Hãy đề xuất một thuật toán xây dựng rừng khung của một đồ thị bằng cách xóa các cạnh tạo thành các chu trình đơn.

33. Hãy đề xuất một thuật toán xây dựng rừng khung của một đồ thị bằng kỹ thuật tìm kiếm ưu tiên chiều sâu.

34. Hãy đề xuất một thuật toán xây dựng rừng khung của một đồ thị bằng kỹ thuật tìm kiếm ưu tiên chiều rộng.

Gọi T_1 và T_2 là hai cây khung của một đồ thị. **Khoảng cách** *giữa T_1 và T_2 được định nghĩa là số các cạnh trong T_1 và T_2 mà không là cạnh chung của chúng.*

35. Hãy tìm khoảng cách giữa mỗi cặp cây khung trên Hình 3 và Hình 4 của đồ thị G.

36*. Giả sử T_1, T_2 và T_3 là các cây khung của đơn đồ thị G. Chứng minh rằng khoảng cách giữa T_1 và T_3 không vượt quá tổng khoảng cách giữa T_1 và T_2 và khoảng cách giữa T_2 và T_3.

37.** Giả sử rằng T_1 và T_2 là cây khung của đơn đồ thị G, còn e_1 là một cạnh trong T_1 mà không là một cạnh trong T_2. Chứng minh rằng trong T_2 có cạnh e_2 không thuộc T_1 sao cho T_1 vẫn còn là một cây nếu xóa e_1 khỏi nó và thêm e_2 vào nó, và T_2 vẫn còn là một cây nếu xóa e_2 khỏi nó và thêm e_1 vào nó.

38*. Chỉ ra rằng có thể tìm được một dãy cây khung sao cho từ một cây khung bất kỳ này chuyển sang cây khung khác bằng cách lần lượt xóa một cạnh và ghép vào một cạnh khác.

Cây khung có gốc *của một đồ thị có hướng được định nghĩa là cây có gốc chứa các cạnh của đồ thị sao cho mọi đỉnh của đồ thị đều là một điểm đầu mút của một trong các cạnh của cây.*

39. Trong mỗi một đồ thị có hướng của các Bài tập 24–28 của Tiết 7.5 hoặc là tìm cây khung có gốc của các đồ thị đó hoặc chứng tỏ rằng không tồn tại cây như thế.

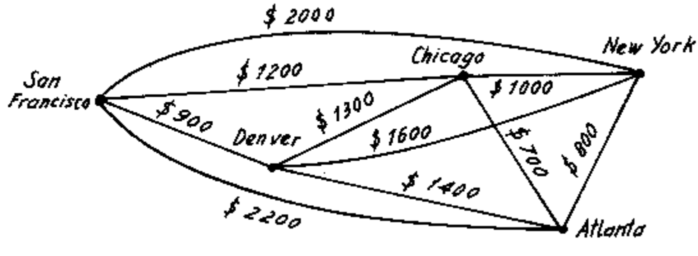
40*. Chỉ ra rằng đồ thị có hướng liên thông trong đó mỗi đỉnh có bậc–ra và bậc–vào như nhau, sẽ có cây khung có gốc.
(*Gợi ý* : Dùng chu trình Euler).

41*. Hãy đề xuất một thuật toán xây dựng cây khung có gốc của một đồ thị có hướng liên thông trong đó mỗi đỉnh có bậc ra và bậc vào bằng nhau.

8.6. CÂY KHUNG NHỎ NHẤT

MỞ ĐẦU

Một công ty lập kế hoạch xây dựng một mạng truyền thông nối năm trung tâm máy tính với nhau. Bất kỳ hai trung tâm nào cũng có thể được nối kết với nhau bằng đường điện thoại. Cần phải kết nối như thế nào để đảm bảo giữa hai trung tâm máy tính bất kỳ luôn có đường truyền thông



Hình 1. Đồ thị có trọng số biểu thị tiền thuê bao hàng tháng đường truyền thông trong mạng máy tính.

sao cho tổng số tiền thuê bao của toàn mạng là tối thiểu? Chúng ta cần mô hình bài toán này bằng đồ thị có trọng số như trên Hình 1, trong đó mỗi đỉnh là một trung tâm máy tính, mỗi cạnh là một đường truyền thông được thuê bao, còn trọng số của mỗi cạnh là tiền thuê bao hàng tháng của đường truyền thông được biểu thị bằng cạnh đó. Chúng ta có thể giải bài toán này bằng cách tìm cây khung sao cho tổng các trọng số của các cạnh của cây đạt cực tiểu. Cây khung như thế được gọi là **cây khung nhỏ nhất.**

THUẬT TOÁN TÌM CÂY KHUNG NHỎ NHẤT

Một lớp rất rộng các bài toán có thể giải bằng cách tìm cây khung nhỏ nhất trong một đồ thị có trọng số sao cho tổng trọng số của các cạnh của cây là nhỏ nhất.

ĐỊNH NGHĨA 1. *Cây khung nhỏ nhất* trong một đồ thị liên thông, có trọng số là một cây khung có tổng trọng số trên các cạnh của nó là nhỏ nhất.

Chúng ta sẽ giới thiệu hai thuật toán xây dựng cây khung nhỏ nhất. Cả hai đều được tiến hành bằng cách ghép các cạnh có trọng số nhỏ nhất trong số các cạnh có một tính chất nào đó mà chưa được dùng. Những thuật toán này là những ví dụ về các **thuật toán tham lam**. Thuật toán tham lam là một thủ tục thực hiện một lựa chọn tối ưu ở mỗi giai đoạn. Tối ưu hóa ở mỗi giai đoạn của thuật toán không đảm bảo tạo ra lời giải tối ưu toàn cục. Nhưng hai thuật toán giới thiệu trong mục này để xây dựng cây khung nhỏ nhất là các thuật toán tham lam tạo ra các lời giải tối ưu.

Thuật toán đầu tiên mà chúng ta sẽ thảo luận bây giờ là do Robert Prim đưa ra vào năm 1957, mặc dù ý tưởng cơ bản của nó đã có từ sớm hơn rất nhiều. Để thực hiện **thuật toán Prim**, ta bắt đầu bằng việc chọn một cạnh bất kỳ có trọng số nhỏ nhất, đặt nó vào cây khung. Lần lượt ghép vào cây các cạnh có trọng số tối thiểu liên thuộc với một đỉnh của cây và không tạo ra chu trình trong cây. Thuật toán sẽ dừng khi $n - 1$ cạnh đã được ghép vào cây.

Cuối mục này, chúng ta chứng minh rằng thuật toán này tạo ra cây khung nhỏ nhất cho đồ thị liên thông có trọng số. Thuật toán 1 mô tả thuật toán Prim ở dạng giả mã.

Lưu ý rằng việc chọn một cạnh ghép vào cây trong mỗi giai đoạn của thuật toán là không xác định khi có nhiều hơn một cạnh cùng trọng số và thỏa mãn những tiêu chuẩn nào đó. Chúng ta cần sắp xếp các cạnh theo một thứ tự nào đó để việc chọn một cạnh được xác định. Khi đó chúng ta sẽ không phải lo lắng gì về điều này. Cũng vậy cần lưu ý là có nhiều hơn một cây khung nhỏ nhất ứng với một đồ thị liên thông và có trọng số. (Xem Bài tập 9). Ví dụ sau minh họa cách dùng thuật toán. Prim.

THUẬT TOÁN 1. THUẬT TOÁN PRIM.

procedure *Prim* (*G* : đồ thị liên thông có trọng số với *n* đỉnh)

T : = cạnh có trọng số nhỏ nhất.

for *i* := 1 **to** *n* − 2

begin

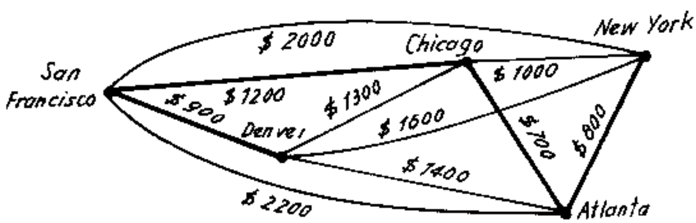
    e : = cạnh có trọng số tối thiểu liên thuộc với một đỉnh trong *T* và không tạo ra chu trình trong *T* nếu ghép nó vào *T*

    T : = *T* với *e* được ghép vào.

end {*T* là cây khung nhỏ nhất của *G*}

Ví dụ 1. Dùng thuật toán Prim để thiết kế một mạng truyền thông có giá tối thiểu để nối các trung tâm máy tính được biểu diễn trên Hình 1.

Giải: Chúng ta sẽ giải bài toán này bằng cách tìm cây khung nhỏ nhất trong đồ thị trên Hình 1. Thuật toán Prim được tiến hành bằng cách chọn cạnh đầu tiên là cạnh có trọng số nhỏ nhất và lần lượt ghép thêm một cạnh có trọng số nhỏ nhất trong số những cạnh nối với một dỉnh của cây và không tạo thành một chu trình. Các cạnh được

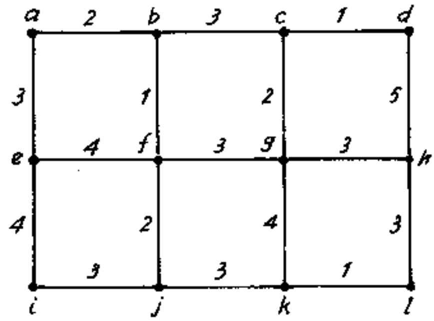


Bước chọn	Cạnh	Phí tổn
1	{Chicago, Atlanta}	$ 700
2	{Atlanta, New York}	$ 800
3	{Chicago, San Francisco}	$ 1200
4	{San Francisco, Denver}	$ 900
	Tổng cộng	$ 3600

***Hình** 2* Cây khung nhỏ nhất đối với đồ thị có trọng số ở hình 1 tô đậm trên Hình 2 là cây khung nhỏ nhất nhận được bằng thuật toán Prim, từng bước chọn cạnh của cây cũng được biểu diễn trên hình đó. ■

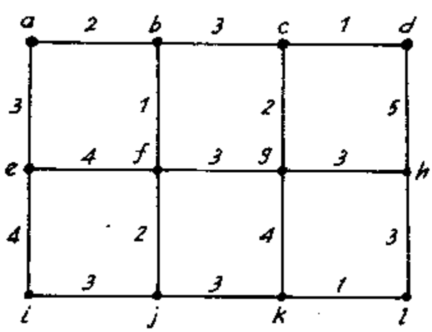
Ví dụ 2. Dùng thuật toán Prim, hãy tìm cây khung nhỏ nhất của đồ thị trên Hình 3.

Giải: Cây khung nhỏ nhất được xây dựng bằng thuật toán Prim được thể hiện trên Hình 4. Thứ tự chọn các cạnh cũng được biểu diễn trong bảng bên cạnh.



Hình 3. Đồ thị có trọng số.

Thuật toán thứ hai mà chúng ta sẽ thảo luận do Joseph Kruskal phát minh vào năm 1956, mặc dù ý tưởng cơ bản của nó đã được biết từ sớm hơn nhiều. Để thực hiện **thuật toán Kruskal** ta chọn cạnh có trọng số nhỏ nhất của đồ thị. Lần lượt ghép thêm vào cạnh có trọng số tối thiểu, và không tạo thành chu trình với các cạnh đã được chọn. Thuật toán dừng sau khi $n - 1$ cạnh đã được chọn.



a)

Bước chọn	Cạnh	Trọng số
1	b, f	1
2	a, b	2
3	f, j	2
4	a, e	3
5	i, j	3
6	f, g	3
7	c, g	2
8	c, d	1
9	g, h	3
10	h, l	3
11	k, l	1
	Tổng cộng	24

b)

Hình 4. Cây khung nhỏ nhất nhận được bằng thuật toán Prim.

Trong một bài tập ở cuối Tiết này chúng tôi yêu cầu độc giả chứng minh rằng thuật toán Kruskal sẽ tạo ra cây khung nhỏ nhất cho một đồ thị liên thông, có trọng số. Dạng giả mã của thuật toán này được cho trong Thuật toán 2.

THUẬT TOÁN 2. THUẬT TOÁN KRUSKAL.

procedure *Kruskal* (*G* : đồ thị *n* đỉnh, liên thông có trọng số)

T : = đồ thị rỗng

for *i* := 1 **to** *n* – 1

begin

  e : = một cạnh bất kỳ của *G* với trọng số nhỏ nhất và không tạo ra chu trình trong *T*, khi ghép nó vào *T*.

  T : = *T* với cạnh *e* đã được ghép thêm vào.

end { *T* là cây khung nhỏ nhất}

Độc giả sẽ nhận thấy sự khác nhau giữa thuật toán Prim và thuật toán Kruskal. Trong thuật toán Prim ta chọn các cạnh có trọng số tối thiểu, liên thuộc với các đỉnh đã thuộc cây và không tạo ra chu trình. Trong khi đó theo thuật toán Kruskal sẽ là chọn các cạnh có trọng số tối thiểu mà không nhất thiết phải liên thuộc với các đỉnh của cây và không tạo ra chu trình. Chú ý rằng cũng như trong thuật toán Prim nếu các cạnh là không được sắp thứ tự có thể có nhiều cách chọn trong mỗi bước của thuật toán này. Do đó để cho thủ tục xác định cần sắp xếp các cạnh theo một trật tự nào đó. Ví dụ sau đây sẽ minh họa cách dùng thuật toán Kruskl.

Ví dụ 3. Dùng thuật toán Kruskal, hãy tìm cây khung nhỏ nhất của đồ thị trên Hình 3.

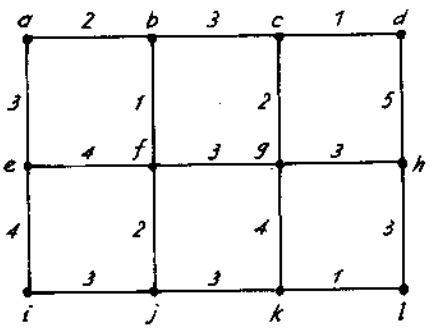
Giải: Cây khung nhỏ nhất và cách chọn các cạnh trong mỗi bước của thuật toán Kruskal được thể hiện trên Hình 5.

■

Bây giờ chúng ta sẽ chứng minh rằng thuật toán Prim tạo ra cây khung nhỏ nhất của đồ thị liên thông có trọng số.

Chứng minh: Gọi G là một đồ thị liên thông có trọng số. Giả sử các cạnh lần lượt được chọn theo thuật toán Prim là e_1, e_2, ..., e_{n-1}. Gọi S là cây với e_1, e_2, ..., e_{n-1} là các cạnh của nó, và S_k là cây với các cạnh e_1, e_2, ..., e_k. Gọi T là cây khung nhỏ nhất của G có chứa các cạnh e_1, e_2, ..., e_k, trong đó k là số nguyên lớn nhất sao cho tồn tại cây khung nhỏ nhất có chứa k cạnh đầu tiên được chọn bằng thuật toán Prim. (*Chú ý* : Vì T là cây khung của G nên có n đỉnh. Do đó theo Định lý 2 ở Tiết 8.1

T có n–1 cạnh – ND). Định lý được chứng minh nếu ta chỉ ra được $S = T$.



a)

b)

Bước chọn	Cạnh	Trọng số
1	c, d	1
2	k, l	1
3	b, f	1
4	c, g	2
5	a, b	2
6	f, j	2
7	b, c	3
8	j, k	3
9	g, h	3
10	i, j	3
11	a, e	3
	Tổng cộng	24

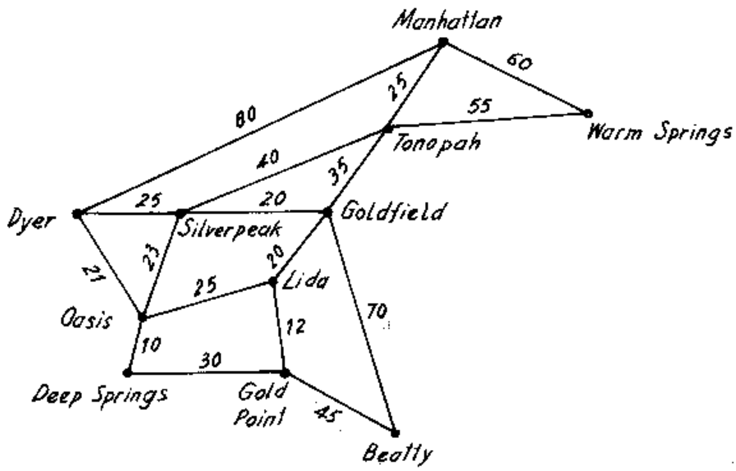
***Hình** 5*. Cây khung nhỏ nhất nhận được bằng thuật toán Kruskal.

Giả sử $S \neq T$, với $k < n - 1$. Do đó, T chứa $e_1, e_2, ..., e_k$ nhưng không chứa e_{k+1}. Xét đồ thị tạo bởi T cùng với e_{k+1}. Vì đồ thị này liên thông và có n cạnh, quá nhiều cạnh để là một cây, vậy tồn tại một chu trình đơn. Chu trình này phải chứa cạnh e_{k+1} vì trong cây T không có chu trình. Hơn nữa trong chu trình đơn này phải có một cạnh không thuộc S_{k+1} vì S_{k+1} là cây. Xuất phát từ điểm đầu mút của cạnh e_{k+1} và cũng là điểm đầu mút của một trong các cạnh $e_1, e_2, ... e_k$, đi dọc theo chu trình cho tới khi gặp một cạnh không thuộc S_{k+1}. Bằng cách đó chúng ta có thể tìm được cạnh e không thuộc S_{k+1} và có một đầu mút là đầu mút của một trong các cạnh $e_1, e_2, ... e_k$. Xóa e khỏi T và thêm vào e_{k+1} chúng ta nhận được cây T' có $n - 1$ cạnh (nó là cây vì nó không có chu trình). Chú ý là T' chứa các cạnh $e_1, e_2, ..., e_k, e_{k+1}$. Hơn nữa ta thấy vì e_{k+1} được chọn bằng thuật toán Prim ở bước $k + 1$ và e cũng có thể được chọn tại bước này nên trọng số của e_{k+1} là nhỏ hơn hay bằng trọng số của e. Vì thế ta suy ra rằng T' cũng là cây khung nhỏ nhất vì tổng trọng số các cạnh của nó không vượt quá tổng trọng số của các cạnh của T. Điều này mâu thuẫn với cách chọn k như là số nguyên lớn nhất sao cho cây khung nhỏ nhất chứa $e_1, e_2, ..., e_k$. Vì thế, $k = n - 1$ và $S = T$, tức là thuật toán Prim tạo ra cây khung nhỏ nhất.

BÀI TẬP

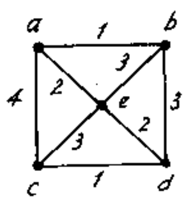
1. Các con đường được biểu diễn trên đồ thị sau là hoàn toàn chưa được trải nhựa. Độ dài của các con đường được biểu thị bằng trọng số của các cạnh. Cần phải trải nhựa những đường nào để vẫn có

đường đi được trải nhựa giữa hai thành phố bất kỳ mà độ dài trải nhựa là tối thiểu?

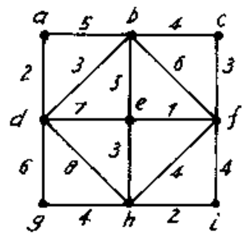


Trong các Bài tập 2-4 hãy dùng thuật toán Prim tìm cây khung nhỏ nhất của các đồ thị đã cho.

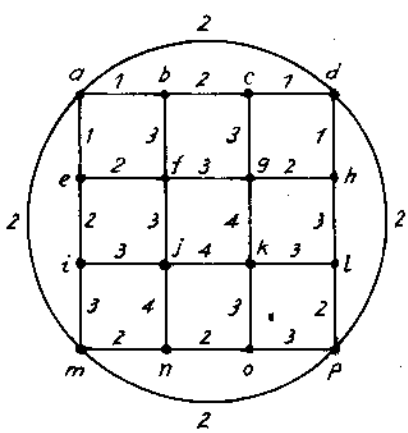
2.



3.



4.



5. Dùng thuật toán Kruskal hãy thiết kế mạng truyền thông được mô tả ở đầu Tiết này.

6. Dùng thuật toán Kruskal hãy tìm cây khung nhỏ nhất cho đồ thị có trọng số trong Bài tập 2.

7. Dùng thuật toán Kruskal hãy tìm cây khung nhỏ nhất cho đồ thị có trọng số trong Bài tập 3.

8. Dùng thuật toán Kruskal hãy tìm cây khung nhỏ nhất cho đồ thị có trọng số trong Bài tập 4.

9. Hãy tìm một đơn đồ thị liên thông có trọng số với một số tối thiểu các cạnh sao cho nó có nhiều hơn một cây khung nhỏ nhất.

10. **Rừng khung nhỏ nhất** trong một đồ thị có trọng số là một rừng khung có trọng số nhỏ nhất. Hãy sửa đổi các thuật toán Prim và thuật toán Kruskal để xây dựng rừng khung nhỏ nhất.

Cây khung cực đại *của một đồ thị vô hướng, liên thông, có trọng số là cây khung có trọng số lớn nhất.*

11. Hãy đề xuất một thuật toán tương tự thuật toán Prim xây dựng cây khung cực đại của một đồ thị liên thông có trọng số.

12. Hãy đề xuất một thuật toán tương tự thuật toán Kruskal xây dựng cây khung cực đại của một đồ thị liên thông có trọng số.

13. Hãy tìm cây khung cực đại cho đồ thị có trọng số trong Bài tập 2.

14. Hãy tìm cây khung cực đại cho đồ thị có trọng số trong Bài tập 3.

15. Hãy tìm cây khung cực đại cho đồ thị có trọng số trong Bài tập 4.

16. Hãy tìm mạng truyền thông rẻ thứ hai nối các trung tâm máy tính trong bài toán ở đầu Tiết này.

17*. Hãy đề xuất một thuật toán tìm cây khung ngắn thứ hai trong một đồ thị liên thông có trọng số.

18*. Hãy chỉ ra rằng cạnh với trọng số nhỏ nhất trong một đồ thị liên thông có trọng số phải có mặt trong một cây khung nhỏ nhất bất kỳ.

19. Chỉ ra rằng có duy nhất một cây khung nhỏ nhất trong một đồ thị liên thông có trọn số nếu trọng số của tất cả các cạnh đều khác nhau.

20. Giả sử mạng máy tính nối các thành phố trên Hình 1 phải chứa một đường trực tiếp giữa New York và Denver. Cần phải thêm các kết nối nào để có đường truyền thông giữa hai trung tâm máy tính bất kỳ và giá thành thấp nhất.

21. Hãy tìm cây khung có tổng trọng số tối thiểu và chứa các cạnh $\{e, i\}$ *và* $\{g, k\}$ trong đồ thị có trọng số trên Hình 3.

22. Hãy mô tả thuật toán tìm cây khung với trọng số tối thiểu chứa một tập xác định các cạnh trong một đồ thị đơn vô hướng liên thông và có trọng số.

23. Hãy viết thuật toán đề xuất trong Bài tập 22 dưới dạng giả mã.

Thuật toán Sollin *sinh ra cây khung nhỏ nhất từ một đơn đồ thị liên thông có trọng số $G = (V, E)$, bằng cách lần lượt ghép một nhóm các cạnh. Giả sử các đỉnh trong V là được sắp thứ tự. Điều đó tạo ra một cách sắp xếp các cạnh trong đó $\{u_0, v_0\}$ đi trước $\{u_1, v_1\}$ nếu u_0 đi trước u_1, hoặc nếu $u_0 = u_1$ và v_0 đi trước v_1. Thuật toán bắt đầu bằng việc chọn đồng thời cạnh có trọng số nhỏ nhất liên thuộc với mỗi đỉnh. Cạnh đầu tiên trong thứ tự sẽ được chọn trong trường hợp ngang nhau. Thủ tục này tạo được một đồ thị không có chu trình, tức là được một rừng. Tiếp theo, chọn đồng thời cho mỗi cây của rừng một cạnh ngắn nhất nối một đỉnh của cây này với một đỉnh ở cây khác. Một lần nữa ta lại lấy cạnh đầu tiên trong thứ tự nếu có sự ngang nhau. (Điều đó tạo ra đồ thị không có chu trình đơn, và chứa ít cây hơn so với các đồ thị tạo ra ở bước trước, xem Bài tập 24). Tiếp tục quá trình ghép đồng thời các cạnh nối các cây cho tới khi chọn được $n - 1$ cạnh. Cây khung nhỏ nhất được tạo ra ở cuối bước này.*

24*. Chứng tỏ rằng việc ghép thêm mỗi cạnh ở mỗi giai đoạn của thuật toán Sollin tạo ra rừng.

25. Dùng thuật toán Sollin hãy tạo ra cây khung nhỏ nhất cho đồ thị có trọng số cho trên.

a) Hình 1 b) Hình 3.

26*. Viết thuật toán Sollin dưới dạng giả mã.

27.** Chứng tỏ rằng thuật toán Sollin tạo ra cây khung nhỏ nhất trong đồ thị liên thông vô hướng và có trọng số.

28*. Chỉ ra rằng bước 1 của thuật toán Sollin tạo ra rừng chứa ít nhất $\lceil n/2 \rceil$ cạnh.

29*. Chứng tỏ rằng nếu có r cây trong một rừng ở một bước trung gian nào đó của thuật toán Sollin thì ít nhất có $\lceil r/2 \rceil$ cạnh được ghép thêm vào sau bước lặp tiếp theo của thuật toán Sollin.

30*. Chỉ ra rằng có không quá $\lceil n/2^k \rceil$ cây còn lại sau khi bước lặp thứ nhất được thực hiện và sau khi bước lặp thứ hai được thực hiện $k - 1$ lần.

31*. Chỉ ra rằng thuật toán Sollin đòi hỏi nhiều nhất $log\ n$ phép lặp để tạo ra cây khung nhỏ nhất từ một đồ thị liên thông, vô hướng và có trọng số với n đỉnh.

32. Chứng minh rằng thuật toán Kruskal tạo ra cây khung nhỏ nhất.

CÂU HỎI ÔN TẬP

1. a) Định nghĩa cây, b) Định nghĩa rừng.

2. Có thể có hai đường đi đơn khác nhau giữa hai đỉnh của một cây không?

3. Hãy đưa ra ít nhất ba ví dụ dùng cây để mô hình bài toán thực tế.

4. a) Định nghĩa cây có gốc và gốc của một cây như thế.

b) Hãy định nghĩa đỉnh cha và đỉnh con của một đỉnh trong cây có gốc.

c) Đỉnh trong, lá, đồ thị con của một cây có gốc là gì?

d) Hãy vẽ cây có gốc có ít nhất 10 đỉnh, trong đó bậc của mỗi đỉnh không vượt quá 3. Hãy định rõ gốc, cha của mỗi đỉnh, con của mỗi đỉnh, các đỉnh trong và lá.

5. a) Cây với n đỉnh có bao nhiêu lá?

b) Để xác định số cạnh trong rừng với n đỉnh bạn cần phải biết những gì?

6. a) Định nghĩa cây m-phân đầy đủ.

b) Cây m - phân đầy đủ có bao nhiêu đỉnh nếu nó có i đỉnh trong? Cây này có bao nhiêu lá?

7. a) Chiều cao của một cây có gốc là gì?

b) Cây cân đối là gì?

c) Cây m-phân với chiều cao h có bao nhiêu lá?

8. a) Cây tìm kiếm nhị phân là gì?

b) Hãy mô tả thuật toán xây dựng cây tìm nhị phân.

c) Hãy xây dựng cây tìm kiếm nhị phân cho các từ *vireo, warbler, egret, grosbeak, nuthatch*, và *kingfisher*.

9. a) Mã tiền tố là gì?

b) Mã tiền tố có thể biểu diễn bằng cây nhị phân như thế nào?

10. a) Hãy định nghĩa cách duyệt cây theo kiểu tiền thứ tự, trung thứ tự và hậu thứ tự.

b) Hãy đưa ra ví dụ về các cách duyệt theo kiểu tiền thứ tự, trung thứ tự và hậu thứ tự một cây nhị phân mà bạn chọn với ít nhất 12 đỉnh.

11. a) Hãy giải thích cách dùng các kiểu duyệt tiền thứ tự, trung thứ tự và hậu thứ tự để tìm dạng tiền tố, trung tố và hậu tố của một biểu thức số học.

b) Vẽ cây có gốc và được sắp biểu diễn biểu thức :

$((x - 3) + ((x/4) + (x - y) \uparrow 3))$.

c) Tìm dạng tiền tố và hậu tố của biểu thức trong phần *b*).

12. Chứng tỏ rằng số phép so sánh dùng thuật toán sắp xếp ít nhất bằng $\lceil \log n\,! \rceil$.

13. a) Mô tả thuật toán sắp xếp kiểu nổi bọt.

b) Dùng thuật toán sắp xếp kiểu nổi bọt hãy sắp danh sách 5, 2, 4, 1, 3 theo thứ tự tăng dần.

c) Hãy đưa ra một đánh giá bậc lớn big – O cho các phép so sánh dùng trong thuật toán sắp xếp kiểu nổi bọt.

14. a) Mô tả thuật toán sắp xếp kiểu hoà nhập.

b) Dùng thuật toán sắp xếp kiểu hoà nhập hãy sắp danh sách 5, 2, 4, 1, 3 theo thứ tự tăng dần.

c) Hãy đưa ra một dánh giá bậc lớn big – O cho các phép so sánh dùng trong thuật toán sắp xếp kiểu hoà nhập.

15. a) Cây khung của một đơn đồ thị là gì?

b) Những đơn đồ thị nào có cây khung?

c) Hãy mô tả ít nhất hai ứng dụng khác nhau yêu cầu tìm cây khung của một đơn đồ thị.

16. a) Hãy mô tả hai thuật toán khác nhau tìm cây khung của một đơn đồ thị.

b) Hãy minh họa cách dùng hai thuật toán mà bạn mô tả trong câu a) để tìm cây khung của một đơn đồ thị, dùng đồ thị mà bạn chọn có ít nhất 8 đỉnh, 15 cạnh.

17. a) Hãy giải thích cách dùng kỹ thuật quay lui để xác định xem có thể tô một đơn đồ thị bằng n màu hay không?

b) Hãy chỉ ra, bằng ví dụ, rằng kỹ thuật quay lui có thể dùng để chứng minh một đồ thị với số màu bằng 4 không thể tô bằng ba màu, nhưng có thể tô bằng bốn màu.

18. a) Thế nào là cây khung nhỏ nhất của một đơn đồ thị?

b) Hãy đưa ra ít nhất hai ứng dụng khác nhau yêu cầu tìm cây khung nhỏ nhất của một đơn đồ thị.

19. a) Hãy mô ta thuật toán Prim và thuật toán Kruskal tìm cây khung nhỏ nhất.

b) Hãy minh họa cách dùng thuật toán Prim và thuật toán Kruskal để tìm cây khung nhỏ nhất cho một đồ thị có trọng số và có ít nhất 8 đỉnh và 15 cạnh.

BÀI TẬP BỔ SUNG

1*. Chỉ ra rằng một đơn đồ thị là cây nếu và chỉ nếu nó không chứa chu trình đơn và khi thêm một cạnh nối hai đỉnh không liền kề sẽ tạo ra một đồ thị mới có đúng một chu trình đơn (trong đó những chu trình chứa các cạnh như nhau không coi là khác nhau).

2*. Có bao nhiêu cây có gốc không đẳng cấu với 6 đỉnh.

3. Chỉ ra rằng mọi cây có ít nhất một cạnh phải có ít nhất hai đỉnh treo.

4. Chỉ ra rằng một cây với n đỉnh có $n - 1$ đỉnh treo sẽ đẳng cấu với $K_{1,n-1}$.

5. Tính tổng bậc của các đỉnh của một cây với n đỉnh.

6*. Giả sử rằng $d_1, d_2, ..., d_n$ là các số nguyên dương với tổng là $2n-2$. Hãy chỉ ra rằng tồn tại một cây có n đỉnh sao cho bậc của mỗi đỉnh là $d_1, d_2, ..., d_n$.

7. Chứng tỏ rằng mọi cây là một đồ thị phẳng.

8. Chỉ ra rằng mọi cây là đồ thị phân đôi.

9. Chỉ ra rằng mọi rừng đều có thể tô bằng hai màu.

Một B-cây bậc k *là một cây có gốc sao cho tất cả các lá của nó ở cùng một mức, gốc của nó có ít nhất hai con, nhiều nhất k con trừ khi nó là lá và mỗi đỉnh trong không phải gốc có ít nhất* $\lceil k/2 \rceil$, *nhưng không hơn k con. Các tệp tin có thể truy nhập rất hiệu quả khi dùng các B-cây để biểu diễn chúng.*

10. Hãy vẽ ba B−cây bậc 3 khác nhau và có chiều cao bằng 4.

11*. Hãy đưa ra cận trên và cận dưới của số các lá trong B−cây bậc k và có chiều cao h.

12*. Hãy đưa ra cận trên và cận dưới của chiều cao của B−cây bậc k và có n lá.

Cây có gốc T được gọi là **S_k-cây** *nếu nó thỏa mãn định nghĩa truy hồi sau đây. Cây chỉ có một đỉnh là S_0-cây. Với $k > 0$, T là S_k-cây nếu có thể được xây dựng từ hai S_{k-1}-cây bằng cách lấy gốc của một S_{k-1}-cây làm gốc của S_k-cây và lấy gốc của cây kia là con của gốc của S_k-cây đầu tiên đó.*

13. Hãy vẽ S_k−cây với $k = 0, 1, 2, 3, 4$.

14. Hãy chỉ ra S_k−cây có 2^k đỉnh và một đỉnh duy nhất ở mức k. Đỉnh duy nhất ở mức k này gọi là *tay lái*.

15*. Giả sử T là S_k−cây với *tay lái* v. Hãy chỉ ra rằng T có thể nhận được từ các cây rời nhau $T_0, T_1, \ldots, T_{k-1}$, trong đó v không thuộc bất kỳ cây nào trong các cây này, và T_i là S_i−cây ($i = 1, 2, \ldots, k-1$) bằng cách nối v với r_0 và r_i với r_{i+1} ($i = 0, 1, 2, \ldots, k-2$).

Liệt kê các đỉnh của một cây có gốc được sắp theo **thứ tự mức**, *bắt đầu từ gốc sau đó là các đỉnh ở mức 1, mức 2,..., và trong cùng một mức thì các đỉnh được liệt kê từ trái sang phải.*

16. Hãy liệt kê của cây có gốc được sắp trên Hình 3 và 9 của Tiết 8.3 theo thứ tự mức.

17. Hãy đề xuất một thuật toán để liệt kê các đỉnh của cây có gốc được sắp theo thứ tự mức.

18*. Hãy đề xuất một thuật toán để xác định một tập các địa chỉ phổ dụng có thể là địa chỉ các lá của một cây có gốc hay không.

19. Hãy đề xuất một thuật toán để xây dựng một cây có gốc từ địa chỉ phổ dụng của các lá của nó.

Sắp xếp kiểu chèn *thực hiện bằng cách xét các phần tử của danh sách, mỗi lần một phần tử bắt đầu từ phần tử thứ hai. Mỗi phần tử được so sánh với những phần tử trước nó trong danh sách, đó là những phần tử đã được đặt theo đúng thứ tự. Phần tử đang xét sẽ được đặt vào vị trí đúng của nó bằng cách dịch chuyển phần tử đang ở vị trí này và tất cả các phần tử bên phải nó về bên phải một vị trí.*

20. Sắp xếp danh sách 3, 2, 4, 5, 2, 1 bằng thuật toán sắp xếp kiểu chèn.

21. Hãy viết thuật toán sắp xếp kiểu chèn bằng giả mã.

22. Hãy xác định độ phức tạp trong trường hợp xấu nhất của thuật toán sắp kiểu chèn theo số phép so sánh cần dùng.

23. Giả sử trong một đơn đồ thị, e là một cạnh liên thuộc với một đỉnh treo. Chứng minh rằng e phải nằm trong một cây khung nào đó.

Tập cắt *của một đồ thị là tập các cạnh sao cho việc xóa chúng sẽ tạo ra một đồ thị con có số thành phần liên thông nhiều hơn đồ thị xuất phát, nhưng một tập con thực sự của nó không có tính chất này.*

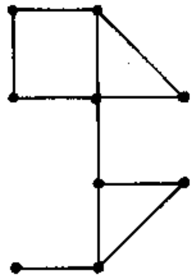
24. Chỉ ra rằng tập cắt của đồ thị phải có ít nhất một cạnh chung với bất kỳ cây khung nào của đồ thị này.

Cactus *là đồ thị liên thông trong đó không có cạnh nào thuộc nhiều hơn một chu trình đơn không đi qua một đỉnh bất kỳ khác đỉnh xuất phát nhiều hơn một lần hoặc đỉnh ban đầu của nó khác với đỉnh kết thúc (trong đó hai chu trình chứa cùng số cạnh như nhau không được coi là khác nhau).*

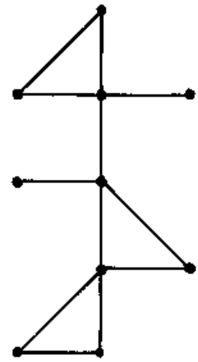
25. Đồ thị nào trong các đồ thị sau là cactus?

a)

b)



c)



26. Một cây có nhất thiết là cactus không?

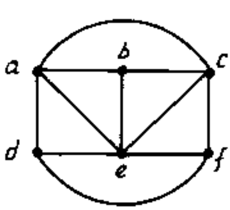
27. Hãy chỉ ra rằng một cactus sẽ được tạo thành nếu ta thêm một chu trình chứa các cạnh mới bắt đầu và kết thúc tại một đỉnh của cây.

28*. Chỉ ra rằng nếu mọi chu trình không đi qua bất kỳ đỉnh nào khác với đỉnh đầu tiên của nó quá một lần trong một đồ thị liên thông chứa một số lẻ các cạnh thì đồ thị này phải là một cactus.

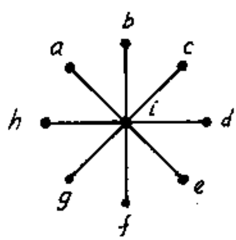
Cây khung với bậc hạn chế *của một đồ thị đơn* G *là cây khung có tính chất sau. Bậc của một đỉnh trong cây này không thể vượt quá một giới hạn xác định nào đó. Cây khung với bậc hạn chế rất có lợi khi mô hình các hệ thống vận tải trong đó số các con đường cắt nhau là có giới hạn, hay mô hình các mạng truyền thông trong đó số các kết nối đi tới một nút là bị hạn chế, v.v.*

Trong các Bài tập 29-31 hãy tìm cây khung với bậc hạn chế của các đồ thị đã cho trong đó mỗi đỉnh có bậc nhỏ hơn hay bằng 3, hoặc chỉ ra rằng không tồn tại cây khung như thế.

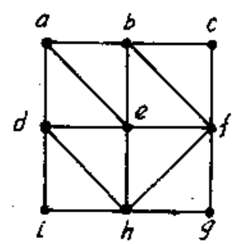
29.



30.

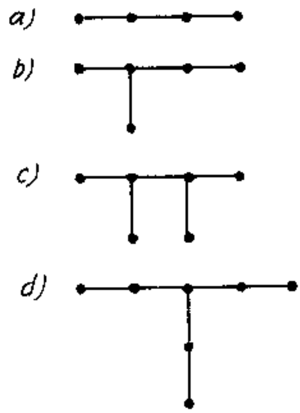


31.



32. Hãy chỉ ra rằng cây khung có số bậc hạn chế của một đơn đồ thị trong đó mỗi đỉnh có bậc không quá 2 bao gồm một đường đi Hamilton trong đồ thị.

33. Một cây với n đỉnh được gọi là **duyên dáng** nếu các đỉnh của nó có thể gán nhãn bằng các số tự nhiên 1, 2, ..., n sao cho giá trị tuyệt đối của hiệu các nhãn tại các đỉnh liền kề đều khác nhau. Hãy chỉ ra các cây ở hình bên là các cây duyên dáng.



Cây dây xích *là cây chứa đường đi đơn sao cho mọi đỉnh không thuộc đường đi là liền kề với đỉnh thuộc đường đi*

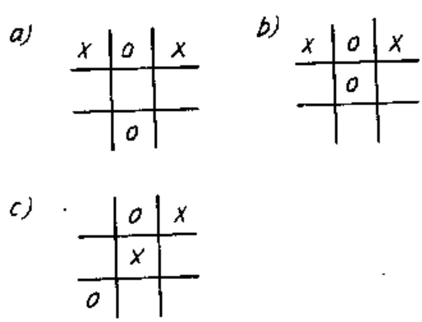
34. Đồ thị nào trong Bài tập 33 là dây xích?

35. Có bao nhiêu cây dây xích không đẳng cấu với 6 đỉnh?

36*. a) Chứng minh hoặc bác bỏ rằng tất cả các cây mà các cạnh của chúng tạo thành một đường đi đơn đều là cây duyên dáng.

** b) Chứng minh hoặc bác bỏ rằng tất cả các cây dây xích đều là các cây duyên dáng.

37. Giả sử rằng bốn nước đi đầu tiên của trò chơi cờ carô như đã chỉ ra trên hình vẽ. Hãy giải thích cách dùng cây để chỉ ra những nước đi kế tiếp của trò chơi này. Nếu người chơi dùng dấu X đi trước, anh ta có chiến thuật để luôn luôn thắng không?



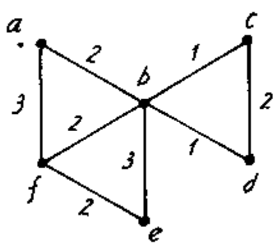
38. Ba cặp vợ chồng đi tới bờ một con sông. Mỗi bà vợ đều hay ghen và không tin chồng mình khi để anh ta đứng một mình với một trong các bà kia, mà không cùng với mình. Làm thế nào sáu người có thể qua sông bằng một chiếc thuyền chỉ chở được không quá hai người sao cho không có ông chồng nào ở một mình với các bà không là vợ mình? Hãy dùng lý thuyết đồ thị để lập mô hình.

39*. Giả sử e là một cạnh của một đồ thị có trọng số, cạnh này liên thuộc với đỉnh v sao cho trọng số của e không vượt quá trọng số của bất kỳ cạnh nào khác liên thuộc với v. Chỉ ra rằng có cây khung nhỏ nhất chứa cạnh này.

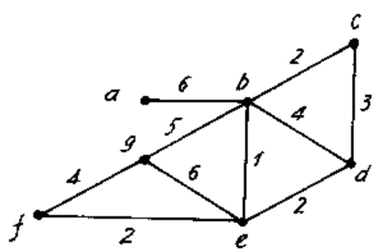
40*. Hãy chỉ ra rằng trong đồ thị có trọng số, nếu không có hai cạnh nào có cùng trọng số khi đó cạnh với trọng số nhỏ nhất liên thuộc với đỉnh v đều được gộp vào mọi cây khung nhỏ nhất.

41. Hãy tìm cây khung nhỏ nhất của mỗi một trong các đồ thị có trọng số sau đây, trong đó bậc của mỗi đỉnh trong cây khung nhỏ nhất không vượt quá 2.

a)



b)



BÀI TẬP TRÊN MÁY TÍNH

Viết chương trình với các input và output cho dưới đây.

1. Cho ma trận liền kề của một đơn đồ thị vô hướng, hãy xác định đồ thị nào là cây.

2. Cho ma trận liền kề của một cây có gốc và một đỉnh của cây, hãy tìm cha, con, tổ tiên, con cháu và mức của đỉnh này.

3. Cho danh sách các cạnh của một cây có gốc và một đỉnh của cây, hãy tìm cha, con, tổ tiên, con cháu và mức của đỉnh này.

4. Cho danh sách các phần tử, xây dựng cây tìm kiếm nhị phân chứa các phần tử này.

5. Cho cây tìm kiếm nhị phân và một phần tử hãy định vị nó hoặc thêm nó vào cây tìm kiếm nhị phân này.

6. Cho danh sách có thứ tự các cạnh của một cây có gốc và được sắp, hãy tìm địa chỉ phổ dụng các đỉnh của nó.

7. Cho danh sách có thứ tự các cạnh của một cây có gốc và được sắp, hãy liệt kê các đỉnh của nó theo kiểu tiền thứ tự, trung thứ tự, và hậu thứ tự.

8. Cho một biểu thức số học dưới dạng tiền tố, hãy tìm giá trị của nó.

9. Cho một biểu thức số học dưới dạng hậu tố, hãy tìm giá trị của nó.

10. Cho một tập n số nguyên, hãy sắp xếp chúng bằng thuật toán sắp xếp kiiểu nổi bọt.

11. Cho hai danh sách được sắp các số nguyên hãy hoà nhập chúng thành một danh sách được sắp, và theo dõi số các phép so sánh đã dùng.

12. Cho một tập n số nguyên, hãy sắp xếp chúng bằng thuật toán sắp xếp kiểu hoà nhập.

13. Cho ma trận liền kề của một đơn đồ thị vô hướng liên thông, hãy tìm cây khung của đồ thị này bằng kỹ thuật tìm kiếm ưu tiên chiều sâu.

14. Cho ma trận liền kề của một đơn đồ thị vô hướng liên thông, hãy tìm cây khung của đồ thị này bằng kỹ thuật tìm kiếm ưu tiên chiều rộng.

15. Cho một tập các số nguyên dương và một số nguyên dương N, hãy dùng kỹ thuật quay lui tìm tập con của các số nguyên này có tổng bằng N.

16*. Cho ma trận liền kề của một đơn đồ thị vô hướng, hãy dùng kỹ thuật quay lui tô đồ thị này bằng ba màu nếu có thể.

17*. Cho só dương n hãy giải bài toán n quân hậu bằng kỹ thuật quay lui.

18. Cho danh sách các cạnh và trọng số của chúng trong một đồ thị liên thông có trọng số, hãy dùng thuật toán Prim tìm cây khung nhỏ nhất của đồ thị này.

19. Cho danh sách các cạnh và trọng số của chúng trong một đồ thị liên thông có trọng số, hãy dùng thuật toán Kruskal tìm cây khung nhỏ nhất của đồ thị này.

TÍNH TOÁN VÀ KHÁM PHÁ

Dùng các chương trình mà bạn đã viết giải các bài tập sau.

1. Hãy biểu diễn tất cả các cây có sáu đỉnh.

2. Hãy biểu diễn tất cả các cây không đẳng cấu có bảy đỉnh.

3*. Hãy xây dựng mã Huffman cho các chữ cái tiếng Anh dựa trên tần số xuất hiện của chúng trong một văn bản tiếng Anh thông thường.

4. Hãy tính số cây khung khác nhau của K_n với $n = 1, 2, 3, 4, 5, 6$. Hãy phỏng đoán công thức cho số cây khung như thế với n là một số nguyên dương tùy ý.

5. Hãy so sánh số các phép so sánh cần thiết để sắp xếp một danh sách n phần tử với $n = 100, 1000, 10\ 000$ trong đó các phần tử là các số nguyên dương được chọn ngẫu nhiên bằng sắp xếp kiểu chèn, sắp xếp kiểu hoà nhập, và sắp xếp nhanh.

6. Tính số các cách khác nhau để sắp n quân hậu trên một bàn cờ $n \times n$ sao cho không có hai quân nào tấn công lẫn nhau, với n nguyên dương không vượt quá 10.

7*. Tìm cây khung nhỏ nhất của một đồ thị nối thủ đô của 50 bang của nước Mỹ với nhau, trong đó trọng số của mỗi cạnh là khoảng cách giữa các thành phố đó.

VIẾT TIỂU LUẬN

Dùng các tư liệu ngoài cuốn sách này viết các tiểu luận trả lời những câu hỏi sau :

1. Hãy trình bày cách Cayley dùng cây để tính số các đồng phân của hydrocarbon.

2. Định nghĩa cây *AVL* (đôi khi cũng được gọi là cây có độ cao cân đối). Mô tả cách dùng và tại sao *AVL* – cây được dùng trong nhiều thuật toán.

3. Định nghĩa cây quad và giải thích cách dùng cây quad để biểu diễn hình ảnh. Hãy mô tả cách quay hình ảnh, cách phóng to thu nhỏ và biến đổi hình ảnh bằng các thao tác đối với cây quad tương ứng.

4. Hãy định nghĩa *heap* và giải thích cây có thể biến thành heap như thế nào, tại sao heap rất có lợi khi sắp xếp?

5. Hãy mô tả các thuật toán nén dữ liệu dựa trên tần số chữ cái, kể cả mã Huffman, và các thuật toán liên quan dựa trên tần số của các khối chữ cái.

6. Hãy bàn về việc cây đã được dùng như thế nào để lập mô hình các trò chơi và tìm chiến lược thắng. Hãy giải thích người ta đã nghiên cứu các trò chơi như cờ caro, nim, hex, và một vài trò chơi khác bằng cây như thế nào?

7. Hãy định nghĩa một loại đồ thị có tên là *mạng các cây*. Hãy giải thích cách dùng đồ thị này trong các ứng dụng của hệ thống tích hợp cực lớn và tính toán song song rất lớn.

8. Mô tả thuật toán tìm cây khung nhỏ nhất của một đồ thị sao cho bậc cực đại của một đỉnh bất kỳ trong cây khung nhỏ nhất đó không vượt quá hằng số k cố định.

9. Hãy so sánh và đối chiếu một số thuật toán sắp xếp quan trọng nhất về độ phức tạp của chúng và khi dùng chúng.

10. Hãy thảo luận về lịch sử và nguồn gốc của các thuật toán xây dựng cây khung nhỏ nhất.

11. Hãy mô tả các thuật toán tạo cây ngẫu nhiên.

CHƯƠNG 9

ĐẠI SỐ BOOLE

Các mạch điện trong máy tính và các dụng cụ điện tử khác đều có các đầu vào, mỗi đầu vào là số 0 hoặc số 1, và tạo ra các đầu ra cũng là các số 0 và 1. Các mạch điện đó đều có thể được xây dựng bằng cách dùng bất kỳ một phần tử cơ bản nào có hai trạng thái khác nhau. Chúng bao gồm các chuyển mạch có thể ở hai vị trí mở hoặc đóng và các dụng cụ quang học có thể là sáng hoặc tối. Năm 1938 Claude Shannon chứng tỏ rằng có thể dùng các qui tắc cơ bản của logic do George Boole đưa ra vào năm 1854 trong cuốn "*Các qui luật của tư duy*" của ông để thiết kế các mạch điện. Các qui tắc này đã tạo nên cơ sở của đại số Boole. Trong chương này chúng ta sẽ phát triển các tính chất cơ bản của đại số Boole. Sự hoạt động của một mạch điện được xác định bởi một hàm Boole chỉ rõ giá trị của đầu ra đối với mỗi tập đầu vào. Bước đầu tiên trong việc xây dựng một mạch điện là biểu diễn hàm Boole của nó bằng một biểu thức được lập bằng cách dùng các phép toán cơ bản của đại số Boole. Chúng ta cũng sẽ đưa ra một thuật toán để tạo các biểu thức như vậy. Biểu thức mà chúng ta sẽ nhận được có thể chứa nhiều phép toán hơn mức cần thiết để biểu diễn hàm đó. Ở phần cuối của chương chúng ta sẽ mô tả các phương pháp để tìm một biểu thức với số tối thiểu các phép tổng và tích được dùng để biểu diễn một hàm Boole. Các thủ tục mà chúng ta sẽ mô tả – đó là các bản đồ Karnaugh và các phương pháp Quine – McCluskey – là rất quan trọng trong việc thiết kế các mạch điện có hiệu quả cao.

9.1. HÀM BOOLE

MỞ ĐẦU

Đại số Boole đưa ra các phép toán và qui tắc làm việc với tập {0, 1}. Các chuyển mạch điện tử và quang học có thể được nghiên cứu bằng cách dùng tập này và các qui tắc của đại số Boole. Ba phép toán trong đại số Boole mà chúng ta sẽ dùng nhiều nhất, đó là phép lấy phần bù, phép lấy tổng Boole và tích Boole. **Phần bù** của một phần tử được ký hiệu bằng một gạch ngang trên đầu và được định nghĩa bởi $\overline{0} = 1$ và $\overline{1} = 0$. Tổng Boole được ký hiệu là + hoặc *OR* (hoặc) có các giá trị sau :

$$1 + 1 = 1 \; ; \; 1 + 0 = 1, \; 0 + 1 = 1, \; 0 + 0 = 0$$

Tích Boole được ký hiệu là . hoặc AND (và) có các giá trị như sau :

$$1.1 = 1 \; ; \; 1.0 = 0 \; ; \; 0.1 = 0 \; ; \; 0.0 = 0$$

Khi không có nguy cơ nhầm lẫn ký hiệu dấu• có thể được bỏ đi như cách viết các tích đại số thông thường. Nếu không dùng các dấu ngoặc thì thứ tự thực hiện các phép toán Boole như sau : trước hết, thực hiện tất cả các phép lấy phần bù, sau đó đến tích Boole rồi mới đến tổng Boole. Điều này được minh họa trong ví dụ sau :

Ví dụ 1. Tìm giá trị của $1.0 + \overline{(0 + 1)}$

Giải. Dùng các định nghĩa của phép lấy phần bù, phép lấy tổng và tích Boole, ta suy ra :

$$(1.0) + \overline{(0 + 1)} = 0 + \overline{1}$$

$$= 0 + 0 = 0$$

■

Phép lấy phần bù, lấy tổng và tích Boole tương ứng với các toán tử logic ¬, ∨ và ∧, trong đó 0 tương ứng với *F* (sai) và 1 tương ứng với *T* (đúng). Các kết quả của đại số Boole có thể được dịch trực tiếp thành các kết quả về các mệnh đề. Ngược lại, các kết quả về các mệnh đề cũng có thể được dịch thẳng thành các khẳng định của đại số Boole.

BIỂU THỨC BOOLE VÀ HÀM BOOLE

Cho $B = \{0, 1\}$. Biến x được gọi là một **biến Boole** nếu nó nhận các giá trị chỉ từ B. Một hàm từ B^n – tức là từ tập $\{(x_1, x_2, ..., x_n) \mid x_i \in B, 1 \leq i \leq n\}$ – tới B được gọi là **hàm Boole bậc n**. Các giá trị của hàm Boole thường được cho trong các bảng. Ví dụ, hàm Boole $F(x, y)$ với giá trị bằng 1 khi $x = 1$ và $y = 0$ và bằng 0 với mọi lựa chọn khác đối với các giá trị của x và y có thể được biểu diễn bởi Bảng 1.

BẢNG 1

x	y	F(x,y)
1	1	0
1	0	1
0	1	0
0	0	0

Các hàm Boole cũng có thể được biểu diễn bằng cách dùng các biểu thức được tạo bởi các biến và các phép toán Boole. Các **biểu thức Boole** với các biến $x_1, x_2, ..., x_n$ được định nghĩa một cách đệ qui như sau :

0, 1, $x_1, x_2, ..., x_n$ là các biểu thức Boole.

Nếu E_1 và E_2 là các biểu thức Boole thì $\overline{E}$, $(E_1 E_2)$ và $(E_1 + E_2)$ cũng là các biểu thức Boole.

Mỗi một biểu thức Boole biểu diễn một hàm Boole. Các giá trị của hàm này nhận được bằng cách thay 0 và 1 cho các biến trong biểu thức đó. Trong Tiết 9.2 chúng ta sẽ chứng minh rằng mỗi một hàm Boole đều có thể được biểu diễn bằng một biểu thức Boole.

Ví dụ 2. Tìm các giá trị của hàm Boole được biểu diễn bởi $F(x,y,z) = xy + \bar{z}$

Giải. Các giá trị của hàm này được cho trong Bảng 2

BẢNG 2

x	y	z	xy	$\bar{z}$	$F(x,y,z) = xy + \bar{z}$
1	1	1	1	0	1
1	1	0	1	1	1
1	0	1	0	0	0
1	0	0	0	1	1
0	1	1	0	0	0
0	1	0	0	1	1
0	0	1	0	0	0
0	0	0	0	1	1

Hai hàm n biến F và G được gọi là bằng nhau nếu $F(b_1, b_2, \ldots b_n) = G(b_1, b_2, \ldots b_n)$ với mọi $b_1, b_2, \ldots, b_n$ thuộc B. Hai biểu thức Boole khác nhau biểu diễn cùng một hàm được gọi là **tương đương**. Ví dụ, các biểu thức Boole xy, $xy + 0$ và $xy.1$ là tương đương. **Phần bù** của hàm Boole F là hàm $\overline{F}$ với $\overline{F}(x_1, \ldots, x_n) = \overline{F(x_1, \ldots, x_n)}$. Giả sử F và G là các hàm Boole bậc n. **Tổng Boole** $F + G$ và **Tích Boole** FG được định nghĩa bởi :

$$(F + G)(x_1, \ldots, x_n) = F(x_1, \ldots, x_n) + G(x_1, \ldots, x_n)$$

$$(FG)(x_1, \ldots, x_n) = F(x_1, \ldots, x_n)\ G(x_1, \ldots, x_n).$$

Một hàm Boole bậc 2 là hàm từ tập 4 phần tử, cụ thể là các cặp phần tử từ tập $B = \{0,1\}$, đến B là tập có hai phần tử. Từ đó suy ra có 16 hàm Boole bậc 2. Bảng 2 cho giá trị của 16 hàm Boole bậc 2 khác nhau mà ta ký hiệu là $F_1, F_2, \ldots, F_{16}$.

BẢNG 3

x	y	F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_6	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}	F_{16}
1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
1	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
0	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
0	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

Ví dụ 3. Có bao nhiêu hàm Boole khác nhau bậc n ?

Giải. Theo qui tắc nhân của phép đếm ta suy ra rằng có 2^n bộ n phần tử khác nhau gồm các số 0 và 1. Vì hàm Boole là sự gán 0 hoặc 1 cho mỗi bộ trong số 2^n bộ n phần tử đó, nên lại theo qui tắc nhân sẽ có 2^{2^n} các hàm Boole khác nhau.

BẢNG 4. Số các hàm Boole bậc n

Bậc	Số các hàm Boole
1	4
2	16
3	256
4	65.536
5	4.294.967.296
6	18.446.744.073.709.551.616

Bảng 4 cho số các hàm Boole khác nhau từ bậc 1 cho tới bậc 6. Số các hàm này tăng cực kỳ nhanh.

CÁC HẰNG ĐẲNG THỨC CỦA ĐẠI SỐ BOOLE

Trong đại số Boole có nhiều hằng đẳng thức. Các hằng đẳng thức quan trọng nhất được cho trong bảng 5. Các hằng đẳng thức này đặc biệt tiện ích trong việc làm đơn giản hóa việc thiết kế các mạch. Mỗi một hằng đẳng thức trong bảng 5 đều có thể được chứng minh bằng cách lập bảng. Ta sẽ chứng minh một trong số hai luật phân phối bằng cách đó trong ví dụ dưới đây. Việc chứng minh các hằng đẳng thức còn lại xin dành cho bạn đọc.

BẢNG 5 CÁC HẰNG ĐẲNG THỨC BOOLE

Hằng đẳng thức	**Tên gọi**
$x = x$	Luật phần bù kép
$x + x = x$ $x \cdot x = x$	Luật lũy đẳng (idempotent)
$x + 0 = x$ $x \cdot 1 = x$	Luật đồng nhất
$x + 1 = 1$ $x \cdot 0 = 0$	Luật nuốt
$x + y = y + x$ $xy = yx$	Luật giao hoán
$x + (y+z) = (x+y) + z$ $x(yz) = (xy)z$	Luật kết hợp
$x + yz = (x + y)(x + z)$ $x(y+z) = xy + xz$	Luật phân phối
$\overline{(xy)} = \overline{x} + \overline{y}$ $\overline{(x + y)} = \overline{xy}$	Luật De Morgan

Ví dụ 4. Chứng minh sự đúng đắn của luật phân phối $x(y+z) = xy + xz$.

Giải. Sự chứng minh hằng đẳng thức này được cho trong Bảng 6. Hằng đẳng thức này đúng vì hai cột sau cùng của bảng hoàn toàn phù hợp với nhau.

BẢNG 6							
x	y	z	y+z	xy	xz	x (y+z)	xy + xz
1	1	1	1	1	1	1	1
1	1	0	1	1	0	1	1
1	0	1	1	0	1	1	1
1	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0
0	1	0	1	0	0	0	0
0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0

Các hằng đẳng thức trong Bảng 5 cũng có thể được dùng để chứng minh các hằng đẳng thức khác. Ta sẽ minh họa điều này bằng ví dụ sau.

Ví dụ 5. Chứng minh **luật hút thu** $x(x+y) = x$ bằng cách dùng các hằng đẳng thức của đại số Boole. (Hằng đẳng thức này được gọi là luật hút thu vì sự hút thu của $x + y$ vào x để cho x không thay đổi).

Giải. Các bước được dùng để rút ra hằng đẳng thức trên và các luật được sử dụng ở mỗi bước đó như sau :

$x(x + y) = (x + 0)(x + y)$ – luật đồng nhất đối với tổng Boole

$= x + 0.y$ – luật phân phối của tổng Bool đối với tích Boole

$= x + y.0$ – luật giao hoán của tích Boole

$= x + 0$ – luật nuốt đối với tích Boole

$= x$ – luật đồng nhất đối với tổng Boole.

■

TÍNH ĐỐI NGẪU

Các hằng đẳng thức trong Bảng 5 xuất hiện theo từng cặp (trừ luật phần bù kép). Để giải thích mối quan hệ giữa hai hằng đẳng thức trong mỗi cặp đó chúng ta phải dùng khái niệm đối ngẫu. **Đối ngẫu** của một biểu thức Boole nhận được bằng cách các tổng và tích Boole đổi chỗ cho nhau, các số 0 và 1 đổi chỗ cho nhau.

Ví dụ 6. Tìm các đối ngẫu của $x(y + 0)$ và $\overline{x} . 1 + (\overline{y} + z)$.

Giải: Đổi chỗ các dấu . và + cho nhau, các số 0 và 1 cho nhau trong các biểu thức trên ta sẽ nhận được các đối ngẫu của chúng. Các đối ngẫu đó là $x + (y.1)$ và $(\overline{x} + 0)(\overline{y} \,.\, z)$, tương ứng.

■

Đối ngẫu của một hàm Boole được biểu diễn bởi một biểu thức Boole là một hàm Boole được biểu diễn bởi đối ngẫu của biểu thức đó. Hàm đối ngẫu này được ký hiệu bởi F^d – không phụ thuộc vào hiểu thức Boole đặc biệt nào đó được dùng để hiểu diễn F. Một hằng dẳng thức giữa các hàm được hiểu diễn bởi các hiểu thức Boole vẫn còn đúng nếu ta lấy đối ngẫu hai vế của nó (xem Bài tập 22). Kết quả này – được gọi là **nguyên lý đối ngẫu** – rất tiện ích để nhận được các hằng đẳng thức mới.

Ví dụ 7. Hãy lập một hằng đẳng thức từ luật hút thu $x(x + y) = x$ được cho trong Ví dụ 5 bằng cách lấy đối ngẫu.

Giải: Lấy đối ngẫu hai vế hằng đẳng thức trên ta được hằng đẳng thức $x + xy = x$. Hằng đẳng thức này cũng được gọi là luật hút thu.

■

ĐỊNH NGHĨA TRỪU TƯỢNG CỦA ĐẠI SỐ BOOLE

Trong tiết này ta sẽ tập trung xem xét các hàm và biểu thức Boole. Tuy nhiên, các kết quả mà chúng ta xác lập được có thể chuyển thành các kết quả cho các mệnh đề hoặc kết quả cho các tập hợp. Vì thế, sẽ rất tiện ích nếu chúng ta định nghĩa đại số Boole một cách trừu tượng. Một khi đã chứng minh được rằng một cấu trúc đặc biệt nào đó là một đại số Boole, thì khi đó mọi kết quả đã được thiết lập cho các đại số Boole tổng quát sẽ được áp dụng cho cấu trúc đặc biệt đó.

Các đại số Boole có thể được định nghĩa bằng nhiều cách. Tuy nhiên, cách phổ hiến nhất là chỉ ra những tính chất mà các phép toán cần phải thỏa mãn, như được làm trong định nghĩa dưới dây :

ĐỊNH NGHĨA 1. **Đại số Boole** là một tập B có hai phần tử 0 và 1 với hai phép toán hai ngôi ∨ và ∧, và một phép toán một ngôi ¯ sao cho các tính chất dưới đây đúng với mọi x, y, z thuộc B.

$$\left.\begin{array}{l} x \vee 0 = x \\ x \wedge 1 = x \end{array}\right\} \qquad \text{Luật đồng nhất}$$

$$\left.\begin{array}{l} x \vee \bar{x} = 1 \\ x \wedge \bar{x} = 0 \end{array}\right\} \quad \text{Luật nuốt}$$

$$\left.\begin{array}{l} (x \vee y) \vee z = x \vee (y \vee z) \\ (x \wedge y) \wedge z = x \wedge (y \wedge z) \end{array}\right\} \quad \text{Luật kết hợp}$$

$$\left.\begin{array}{l} x \vee y = y \vee x \\ x \wedge y = y \wedge x \end{array}\right\} \quad \text{Luật giao hoán}$$

$$\left.\begin{array}{l} x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z) \\ x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z) \end{array}\right\} \quad \text{Luật phân phối}$$

Dùng các luật được cho trong Định nghĩa 1 có thể chứng minh được nhiều luật khác đúng với mọi đại số Boole như luật lũy đẳng và luật De Morgan (xem các Bài tập 25 – 32).

Từ thảo luận ở trên của chúng ta, tập $B = \{0,1\}$ với các phép toán *OR và AND* cùng với toán tử bù thỏa mãn tất cả các tính chất đó. Tập hợp các mệnh đề n biến với các phép toán $\vee$ và $\wedge$, cùng với **F** (sai) và **T** (đúng), và toán tử phủ định cũng thỏa mãn tất cả các tính chất của đại số Boole như có thể thấy qua Bảng 5 trong Tiết 1.2. Tương tự, tập các tập con của tập vũ trụ U với các phép toán hợp và giao, cùng với tập rỗng và tập vũ trụ, và toán tử lấy phần bù của tập hợp cũng là một đại số Boole như dễ dàng thấy qua Bảng 1 của Tiết 1.5. Như vậy, để thiết lập các kết quả cho mỗi một biểu thức Boole, cho các mệnh đề hoặc tập hợp ta chỉ cần chứng minh các kết quả cho các đại số Boole trừu tượng.

BÀI TẬP

1. Tìm giá trị của các biểu thức sau :

a) $1 \cdot \overline{0}$ b) $1 + \overline{1}$

c) $\overline{0} \cdot 0$ d) $\overline{(1 + 0)}$

2. Tìm các giá trị, nếu có, của biến Boole x thỏa mãn các phương trình sau :

a) $x \cdot 1 = 0$ b) $x + x = 0$

c) $x \cdot 1 = x$ d) $x\,\overline{x} = 1.$

3. Tìm giá trị của các biến Boole x và y thỏa mãn phương trình $xy = x + y$

4. Có bao nhiêu hàm Boole bậc 7 khác nhau ?

5. Chứng minh luật hút thu $x + xy = x$ bằng cách dùng các hằng đẳng thức cho trong Bảng 5.

6. Chứng minh rằng $F(x,y,z) = xy + xz + yz$ có giá trị 1 nếu và chỉ nếu ít nhất hai trong số các biến x, y, z có giá trị 1.

7. Chứng minh rằng $x\bar{y} + y\bar{z} + \bar{x}z = \bar{x}y + \bar{y}z + x\bar{z}$

Các bài tập từ 8 đến 15 liên quan đến đại số Boole được định nghĩa bởi tổng Boole và tích Boole trên tập $\{0,1\}$.

8. Chứng minh luật phần bù kép

9. Chứng minh luật lũy đẳng

10. Chứng minh luật đồng nhất

11. Chứng minh luật nuốt

12. Chứng minh luật giao hoán

13. Chứng minh luật kết hợp

14. Chứng minh luật phân phối thứ nhất trong Bảng 5

15. Chứng minh luật De Morgan.

Toán tử Boole $\oplus$, được gọi là toán tử XOR, được định nghĩa như sau : $1 \oplus 1 = 0$, $1 \oplus 0 = 1$, $0 \oplus 1 = 1$ và $0 \oplus 0 = 0$.

16. Rút gọn các biểu thức sau :

a) $x \oplus 0$ b) $x \oplus 1$

c) $x \oplus x$ d) $x \oplus \bar{x}$

17. Chứng minh các hằng đẳng thức sau :

a) $x \oplus y = (x + y)(\overline{xy})$

b) $x \oplus y = (x\bar{y}) + (\bar{x}y)$

18. Chứng minh rằng $x \oplus y = y \oplus x$

19. Chứng minh hoặc bác bỏ các đẳng thức sau :

a) $x \oplus (y \oplus z) = (x \oplus y) \oplus z$

b) $x + (y \oplus z) = (x + y) \oplus (x + z)$

c) $x \oplus (y + z) = (x \oplus y) + (x \oplus z)$

20. Tìm đối ngẫu của các biểu thức sau :

a) $x + y$ b) $\bar{x}\,\bar{y}$

c) $xyz + \bar{x}\ \bar{y}\ \bar{z}$ d) $x\bar{z} + x\ .\ 0 + \bar{x}\ .\ 1$

21*. Cho F là hàm Boole được biểu diễn bởi một biểu thức Boole với các biến $x_1, ..., x_n$. Chứng minh rằng $F^d(x_1, ..., x_n) = \overline{F(\bar{x}_1, ..., \bar{x}_n)}$

22*. Chứng minh rằng nếu F và G là các hàm Boole được biểu diễn bởi các biểu thức Boole n biến và $F = G$ thì $F^d = G^d$ với F^d và G^d là các hàm Boole được biểu diễn bởi đối ngẫu của các biểu thức Boole biểu diễn các hàm F và G tương ứng (*Gợi ý* : dùng kết quả của Bài tập 21).

23*. Có bao nhiêu hàm Boole $F(x,y,z)$ khác nhau sao cho $F(\bar{x}, \bar{y}, \bar{z}) = F(x,y,z)$ đối với mọi giá trị của các biến Boole x, y, z ?

24*. Có bao nhiêu hàm Boole $F(x,y,z)$ khác nhau sao cho $F(\bar{x},y,z) = F(x,\bar{y},z) = F(x,y,\bar{z})$ đối với mọi giá trị của các biến Boole x, y, z ?

Trong các Bài tập từ 25 đến 32, hãy dùng các luật trong Định nghĩa 1 để chứng tỏ rằng các tính chất nêu trong đầu bài đúng với mọi đại số Boole.

25. Chứng minh rằng trong một đại số Boole các **tính chất lũy đẳng** $x \vee x = x$ và $x \wedge x = x$ đúng với mọi x.

26. Chứng minh rằng trong một đại số Boole, mọi phần tử x đều có một phần bù $\bar{x}$ duy nhất sao cho $x \vee \bar{x} = 1$ và $x \wedge \bar{x} = 0$.

27. Chứng minh rằng trong một đại số Boole, phần bù của phần tử 0 là phần tử 1 và ngược lại.

28. Chứng minh rằng trong một đại số Boole đúng **luật phần bù kép**, tức là $\bar{\bar{x}} = x$ với mọi x.

29. Chứng minh rằng trong một đại số Boole, **luật De Morgan** luôn luôn đúng, tức là $\overline{(x \vee y)} = \bar{x} \wedge \bar{y}$ và $\overline{(x \wedge y)} = \bar{x} \vee \bar{y}$ với mọi x và y.

30. Chứng minh rằng trong một đại số Boole các **tính chất modular** sau đây luôn luôn đúng :

+ $x \wedge (y \vee (x \wedge z)) = (x \wedge y) \vee (x \wedge z)$

+ $x \vee (y \wedge (x \vee z)) = (x \vee y) \wedge (x \vee z)$

31. Chứng minh rằng trong một đại số Boole, nếu $x \vee y = 0$ thì $x = 0$ và $y = 0$ và nếu $x \wedge y = 1$ thì $x = 1$ và $y = 1$.

32. Chứng minh rằng trong một đại số Boole **đối ngẫu** của một hằng đẳng thức nhận được bằng cách đổi chỗ các phép toán $\vee$ và $\wedge$ và đổi chỗ các phần tử 0 và 1 cũng là một hằng đẳng thức.

9.2. BIỂU DIỄN CÁC HÀM BOOLE

Hai bài toán quan trọng của đại số Boole sẽ được nghiên cứu trong tiết này. Bài toán thứ nhất là : cho các giá trị của một hàm Boole, làm thế nào tìm được biểu thức Boole biểu diễn hàm đó ? Bài toán này sẽ được giải bằng cách chứng minh rằng mọi hàm Boole đều có thể được biểu diễn bởi tổng các tích Boole của các biến và phần bù của chúng. Lời giải của bài toán này chứng tỏ rằng mọi hàm Boole đều có thể được biểu diễn bằng cách dùng ba toán tử Boole ., + và $^{-}$. Bài toán thứ hai là : liệu có thể dùng một tập nhỏ hơn các toán tử để biểu diễn các hàm Boole không ? Chúng ta sẽ trả lời bài toán này bằng cách chứng minh rằng mọi hàm Boole đều có thể được biểu diễn bằng cách dùng chỉ một toán tử. Cả hai bài toán trên đều có tầm quan trọng thực tiễn trong việc thiết kế các mạch.

KHAI TRIỂN TỔNG CÁC TÍCH

Chúng ta sẽ dùng các ví dụ để minh họa một phương pháp quan trọng để tìm biểu thức Boole biểu diễn một hàm Boole.

Ví dụ 1. Tìm các biểu thức Boole biểu diễn các hàm $F(x,y,z)$ và $G(x,y,z)$ có các giá trị được cho trong Bảng 1.

BẢNG 1

x	y	z	F	G
1	1	1	0	0
1	1	0	0	1
1	0	1	1	0
1	0	0	0	0
0	1	1	0	0
0	1	0	0	1
0	0	1	0	0
0	0	0	0	0

Giải: Cần phải có một biểu thức có giá trị 1 khi $x = z = 1$ và $y = 0$ và có giá trị 0 trong mọi trường hợp còn lại để biểu diễn hàm F. Có thể lập một biểu thức như vậy bằng cách lấy tích Boole của x, $\overline{y}$ và z. Tích này, tức $x\overline{y}z$, có giá trị 1 nếu và chỉ nếu $x = \overline{y} = z = 1$, mà điều này đúng nếu và chỉ nếu $x = z = 1$ và $y = 0$.

Để biểu diễn hàm G, ta cần có một biểu thức bằng 1 khi $x = y = 1$ và $z = 0$ *hoặc khi* $x = z = 0$ và $y = 1$. Chúng ta có thể lập một biểu thức với các giá trị đó bằng cách lấy tổng Boole của hai tích Boole khác nhau. Tích Boole $xy\overline{z}$ có giá trị 1 nếu và chỉ nếu $x = y = 1$ và $z = 0$. Tương tự, tích Boole $\overline{x}y\overline{z}$ có giá trị 1 nếu và chỉ nếu $x = z = 0$ và $y = 1$. Tổng Boole của hai tích này, tức $xy\overline{z} + \overline{x}y\overline{z}$, biểu diễn hàm G vì nó có giá trị 1 nếu và chỉ nếu $x = y = 1$ và $z = 0$ hoặc $x = z = 0$ và $y = 1$. ■

Ví dụ 1 minh họa một thủ tục xây dựng biểu thức Boole biểu diễn một hàm Boole có các giá trị đã cho. Mỗi một tổ hợp giá trị của các biến làm cho hàm có giá trị 1 sẽ dẫn tới một tích Boole của các biến hoặc các phần bù của chúng.

ĐỊNH NGHĨA 1. Một biến Boole hoặc phần bù của nó được gọi là một *tục biến*. Tích Boole $y_1y_2 \ldots y_n$ trong đó $y_i = x_i$ hoặc $y_i = \overline{x}_i$ với $x_1, x_2 \ldots x_n$ là các biến Boole được gọi là một *tiểu hạng* (minterm). Do đó, tiểu hạng là tích của n tục biến.

Một tiểu hạng có giá trị 1 đối với một và chỉ một tổ hợp giá trị của các biến của nó. Nói một cách chính xác hơn, tiểu hạng $y_1y_2 \ldots y_n$ bằng

1 nếu và chỉ nếu mọi $y_i = 1$ và điều này xảy ra nếu và chỉ nếu $x_i = 1$ khi $y_i = x_i$ và $x_i = 0$ khi $y_i = \bar{x}_i$

Ví dụ 2. Tìm tiểu hạng có giá trị bằng 1 nếu $x_1 = x_3 = 0$ và $x_2 = x_4 = x_5 = 1$ và bằng 0 trong mọi trường hợp còn lại.

Giải: Tiểu hạng $\bar{x}_1 x_2 \bar{x}_3 x_4 x_5$ có tập các giá trị đúng theo yêu cầu của đầu bài.

■

Bằng cách lấy tổng Boole của các tiểu hạng phân biệt chúng ta có thể lập được biểu thức Boole với tập các giá trị đã được cho trước. Đặc biệt, tổng Boole của các tiểu hạng có giá trị 1 chỉ khi một trong các tiểu hạng của tổng có giá trị bằng 1. Tiểu hạng đó có giá trị 0 đối với mọi tổ hợp giá trị còn lại của các biến. Do đó, với một hàm Boole đã cho, ta có thể lập một tổng Boole các tiểu hạng có giá trị 1 khi hàm đó có giá trị 1 và có giá trị 0 khi hàm đó có giá trị 0. Các tiểu hạng của tổng Boole này tương ứng với các tổ hợp giá trị làm cho hàm có giá trị 1. Tổng các tiểu hạng biểu diễn hàm được gọi là **khai triển tổng các tích** hay **dạng tuyển chuẩn tắc** của hàm Boole.

Ví dụ 3. Tìm khai triển tổng các tích của hàm $F(x,y,z) = (x + y)\bar{z}$

Giải: Bước đầu tiên là tìm các giá trị của hàm F. Các giá trị này được cho trong bảng 2.

BẢNG 2

x	y	z	$x + y$	$\bar{z}$	$(x + y)\bar{z}$
1	1	1	1	0	0
1	1	0	1	1	1
1	0	1	1	0	0
1	0	0	1	1	1
0	1	1	1	0	0
0	1	0	1	1	1
0	0	1	0	0	0
0	0	0	0	1	0

Khai triển tổng các tích của F là tổng của ba tiểu hạng tương ứng với ba dòng của bảng cho giá trị 1 của hàm đó. Từ đó, ta có :

$$F(x,y,z) = xy\bar{z} + x\bar{y}\bar{z} + \bar{x}y\bar{z}$$

Cũng có thể tìm biểu thức biểu diễn một hàm Boole bằng cách lấy tích Boole của các tổng Boole. Biểu thức tìm được được gọi là **dạng hội chuẩn tắc** hay **khai triển tích các tổng**. Các khai triển này có thể tìm được từ khai triển tổng các tích bằng cách lấy các đối ngẫu. Việc tìm các biểu thức này như thế nào sẽ được mô tả trực tiếp trong Bài tập 10 ở cuối tiết này.

TÍNH ĐẦY ĐỦ

Tất cả các hàm Boole đều có thể được biểu diễn như tổng Boole của các tiểu hạng. Mỗi tiểu hạng là một tích Boole của các biến Boole hoặc các phần bù của chúng. Điều này chứng tỏ rằng mỗi hàm Boole có thể được biểu diễn bằng cách dùng các phép toán Boole ., + và $\bar{}$. Vì tất cả các hàm Boole đều có thể được biểu diễn bằng cách dùng các phép toán đó, nên ta nói rằng tập hợp { ., +, $\bar{}$ } là **đầy đủ**. Liệu ta có thể tìm được một tập đầy đủ các phép toán nhỏ hơn thế không ? Chúng ta có thể làm được điều đó nếu một trong ba phép toán của tập đó có thể được biểu diễn qua hai phép toán kia. Và điều này có thể làm được bằng cách dùng một trong hai luật De Morgan. Chúng ta có thể loại tất cả các tổng Boole bằng cách dùng hằng đẳng thức :

$$x + y = \overline{\bar{x}\,\bar{y}}$$

hằng đẳng thức này nhận được bằng cách lấy phần bù cả hai vế của luật De Morgan thứ hai được cho trong Bảng 5 ở Tiết 9.1, rồi sau đó áp dụng luật phần bù kép. Điều này có nghĩa là tập { . ; $\bar{}$ } là đầy đủ. Tương tự, ta cũng có thể loại tất cả các tích Boole bằng cách dùng hằng đẳng thức :

$$xy = \overline{\bar{x} + \bar{y}},$$

hằng đẳng thức này nhận được bằng cách lấy phần bù cả hai vế của luật De Morgan thứ nhất được cho trong Bảng 5, Tiết 9.1 và sau đó áp dụng luật phần bù kép. Do đó, tập {+ ; $\bar{}$} cũng là đầy đủ. Chú ý rằng tập {+, .} không phải là đầy đủ vì nó không thể biểu diễn hàm Boole $F(x) = \bar{x}$ bằng cách dùng các phép toán đó (xem Bài tập 19).

Ở trên chúng ta đã tìm được các tập đầy đủ chứa hai phép toán. Liệu chúng ta còn có thể tìm được tập đầy đủ các phép toán nhỏ hơn nữa, cụ thể là chỉ chứa một phép toán thôi không ? Những tập như vậy có tồn tại. Ta sẽ định nghĩa hai phép toán : phép $|$ hay NAND và phép $\downarrow$ hay NOR như sau :

$$1 | 1 = 0,\ 1 | 0 = 0 | 1 = 0 | 0 = 1 \text{ và}$$

$$1 \downarrow 1 = 1 \downarrow 0 = 0 \downarrow 1 = 0,\ 0 \downarrow 0 = 1.$$

Cả hai tập $\{ | \}$ và $\{ \downarrow \}$ đều là đầy đủ.

Vì $\{ . ; \bar{\ } \}$ là đầy đủ, nên để thấy $\{ | \}$ là đầy đủ, tất cả những thứ mà ta cần phải làm là chứng tỏ rằng cả hai phép toán . và $\bar{\ }$ đều có thể được biểu diễn bằng cách chỉ dùng phép toán $|$. Điều này được làm như sau:

$$\bar{x} = x | x$$

$$xy = (x | y) | (x | y)$$

Độc giả nên tự chứng minh hai hằng đẳng thức đó (xem Bài tập 14).

Chúng tôi cũng dành việc chứng minh $\{ \downarrow \}$ là tập đầy đủ cho bạn đọc như một bài tập (xem các Bài tập 15 và 16).

BÀI TẬP

1. Tìm tích Boole của các biến x, y, z hoặc phần bù của chúng, biết rằng tích đó có giá trị 1 nếu và chỉ nếu :

a) $x = y = 0,\ z = 1$ b) $x = 0,\ y = 1,\ z = 0$

c) $x = 0,\ y = z = 1$ d) $x = y = z = 0$.

2. Tìm khai triển tổng các tích của các hàm Boole sau :

a) $F(x,y) = \bar{x} + y$ b) $F(x,y) = x\bar{y}$

c) $F(x,y) = 1$ d) $F(x,y) = \bar{y}$

3. Cũng hỏi như trên với các hàm Boole sau :

a) $F(x,y,z) = x + y + z$ b) $F(x,y,z) = (x + z)\ y$

c) $F(x,y,z) = x$ d) $F(x,y,z) = x\ \bar{y}$

4. Tìm khai triển tổng các tích của hàm Boole $F(x,y,z)$ biết rằng F bằng 1 nếu và chỉ nếu :

a) $x = 0$ b) $xy = 0$

c) $x + y = 0$ d) $xyz = 0$.

5. Tìm khai triển tổng các tích của hàm Boole $F(\omega,x,y,z)$, biết rằng F bằng 1 nếu và chỉ nếu một số lẻ của w, x, y, z có giá trị 1.

6. Tìm khai triển tổng các tích của hàm Boole $F(x_1, x_2, x_3, x_4, x_5)$ biết rằng F nhận giá trị 1 nếu và chỉ nếu ba hoặc nhiều hơn các biến x_1, x_2, x_3, x_4, x_5 có giá trị 1.

Có một cách khác để tìm biểu thức Boole biểu diễn một hàm Boole là lập tích các tổng Boole của các tục biến. Các Bài tập từ 7 đến 11 liên quan tới các biểu diễn thuộc loại đó.

7. Tìm một tổng Boole chứa x hoặc $\overline{x}$, y hoặc $\overline{y}$ và z hoặc $\overline{z}$ có giá trị 0 nếu và chỉ nếu

a) $x = y = 1, z = 0$

b) $x = y = z = 0$

c) $x = z = 0, y = 1$.

8. Tìm tích Boole các tổng Boole của các tục biến, biết rằng tích đó có giá trị 0 nếu và chỉ nếu $x = y = 1$ và $z = 0$ hoặc $x = z = 0$ và $y = 1$ hoặc $x = y = z = 0$ (*Gợi ý* : lấy tích Boole của các tổng Boole tìm được trong các câu (a), (b) và (c) của Bài tập 7).

9. Chứng minh rằng tổng $y_1 + y_2 + \ldots + y_n$ trong đó $y_i = x_i$ hoặc $y_i = \overline{x}_i$ có giá trị 0 đối với chỉ một tổ hợp giá trị của các biến, cụ thể là khi $x_i = 0$ nếu $y_i = x_i$ và $x_i = 1$ nếu $y_i = \overline{x}_i$. Tổng Boole này được gọi là một **đại hạng** (maxterm).

10. Chứng minh rằng một hàm Boole có thể được biểu diễn như tích các đại hạng. Biểu diễn này được gọi là **khai triển tích các tổng** hay **dạng hội chuẩn tắc** của hàm đó. (*Gợi ý* : Đưa một đại hạng vào tích này đối với mỗi tổ hợp giá trị của các biến làm cho hàm đó nhận giá trị 0).

11. Tìm khai triển tính các tổng của các hàm Boole cho trong Bài tập 3.

12. Biểu diễn các hàm Boole sau bằng cách dùng các phép toán . và $\overline{\ }$:

a) $x + y + z$ b) $x + \overline{y}(\overline{x} + z)$

c) $\overline{(x + \overline{y})}$ d) $\overline{x}\,(x + \overline{y} + \overline{z})$

13. Biểu diễn các hàm Boole cho trong Bài tập 12 bằng cách dùng các phép toán + và $\overline{\ }$.

14. Chứng minh rằng

a) $\bar{x} = x \mid x$

b) $xy = (x \mid y) \mid (x \mid y)$

c) $x + y = (x \mid x) \mid (y \mid y)$

15. Chứng minh rằng

a) $\bar{x} = x \downarrow x$

b) $xy = (x \downarrow x) \downarrow (y \downarrow y)$

c) $x + y = (x \downarrow y) \downarrow (x \downarrow y)$

16. Dùng kết quả của Bài tập 15 chứng minh rằng tập $\{\downarrow\}$ là đầy đủ.

17. Biểu diễn các hàm Boole cho trong Bài tập 3 bằng cách chỉ dùng phép toán $\mid$.

18. Cũng hỏi như trên nhưng đối với phép toán $\downarrow$.

19. Chứng minh rằng tập các phép toán $\{+, .\}$ là không đầy đủ.

20. Các tập sau có là đầy đủ không

a) $\{+, \oplus\}$ b) $\{\bar{\ }, \oplus\}$ c) $\{., \oplus\}$.

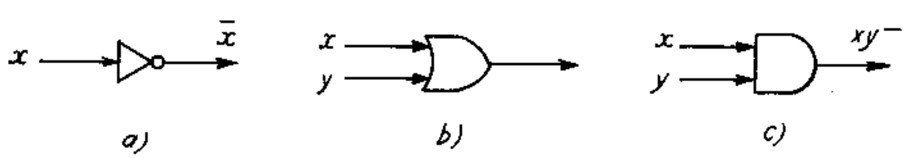
9.3. CÁC CỔNG LOGIC

MỞ ĐẦU

Đại số Boole được dùng để mô hình hóa sơ đồ các mạch trong các dụng cụ điện tử. Mỗi một đầu vào và mỗi một đầu ra của một dụng cụ như vậy có thể được xem như một phần tử của tập $\{0,1\}$. Một máy tính cũng như một dụng cụ điện tử khác được tạo bởi nhiều mạch. Mỗi một mạch có thể được thiết kế bằng cách dùng các qui tắc của đại số Boole đã được đề cập tới trong Tiết 9.1 và 9.2. Các phần tử cơ bản của các mạch được gọi là các **cổng**. Mỗi một loại cổng thực hiện một phép toán Boole. Trong tiết này, chúng ta sẽ định nghĩa một số loại cổng. Dùng các cổng này, chúng ta sẽ áp dụng các qui tắc của đại số Boole để thiết kế các mạch thực hiện các nhiệm vụ khác nhau. Các mạch mà chúng ta sẽ nghiên cứu trong chương này sẽ cho đầu ra chỉ phụ thuộc vào đầu vào chứ không phụ thuộc vào trạng thái hiện thời của mạch. Nói một cách khác, các mạch này không có khả năng nhớ. Những mạch như vậy được gọi là **mạch tổ hợp**.

Chúng ta sẽ xây dựng các mạch tổ hợp bằng cách dùng ba loại phần tử. Loại thứ nhất là **bộ đảo**, nó chấp nhận giá trị của một biến Boole như đầu vào và tạo phần bù của giá trị đó như đầu ra. Ký hiệu được dùng để biểu diễn bộ đảo được cho trên hình 1a. Đầu vào của bộ đảo được cho ở phía trái, đi vào và đầu ra được cho ở phía phải, đi ra từ bộ đảo đó.

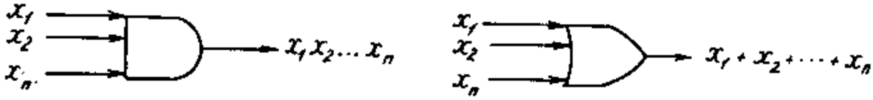
Loại phần tử thứ hai mà chúng ta sẽ dùng là cổng *OR*. Đầu vào cổng này là các giá trị của hai hoặc nhiều hơn biến Boole. Đầu ra là tổng Boole của các giá trị đó. Ký hiệu được dùng để biểu diễn cổng *OR* được cho trên hình 1b. Đầu vào cổng *OR* được cho ở phía trái, đi vào và đầu ra được cho ở phía phải, đi ra khỏi phần tử đó.



Hình 1. Các loại cổng cơ bản.

Loại phần tử thứ ba mà chúng ta sẽ dùng là cổng *AND*. Đầu vào cổng này là các giá trị của hai hoặc nhiều hơn biến Boole. Đầu ra là tích Boole của các giá trị đó. Ký hiệu được dùng để biểu diễn cổng *AND* được cho trên hình 1c. Đầu vào cổng *AND* được cho ở phía trái, đi vào và đầu ra được cho ở phía phải, đi ra từ cổng đó.

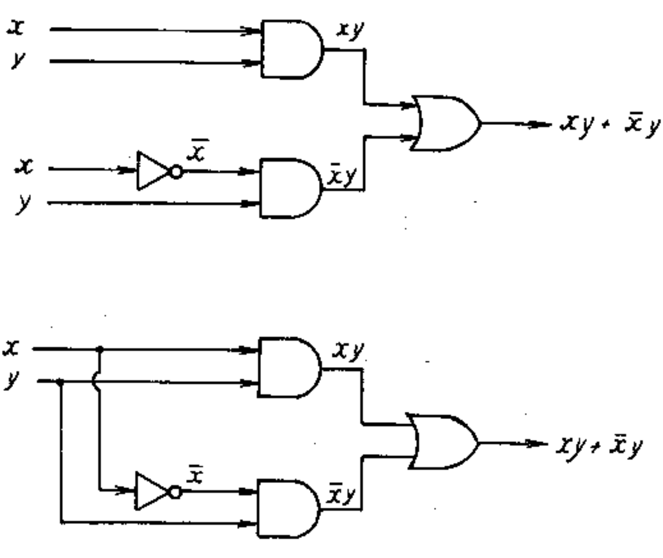
Chúng ta sẽ cho phép có nhiều đầu vào đối với các cổng *OR* và *AND*. Các đầu vào đối với mỗi cổng này được cho ở bên trái, đi vào cổng và đầu ra cho ở bên phải. Ví dụ về các cổng *AND* và *OR* với n đầu vào được cho trong Hình 2.



Hình 2. Các cổng có n đầu vào.

TỔ HỢP CÁC CỔNG

Các mạch tổ hợp có thể được xây dựng bằng cách dùng tổ hợp các bộ đảo, các cổng *OR* và *AND*. Khi lập tổ hợp các mạch, một số cổng có thể dùng chung đầu vào. Điều này được chỉ rõ ở một trong hai cách vẽ mạch dưới đây. Một cách dùng các phân nhanh để chỉ tất cả các cổng cùng dùng một đầu vào đã cho. Còn cách thứ hai chỉ đầu vào này một cách riêng biệt đối với mỗi cổng. Hình 3 minh họa hai cách biểu diễn các cổng cùng dùng chung các giá trị đầu vào. Cũng cần chú ý rằng đầu ra từ một cổng có thể được dùng như đầu vào đối với một hoặc nhiều phần tử như được chỉ rõ trên hình 3. Cả hai hình vẽ trên hình 3 đều vẽ mạch cho cùng đầu ra là $xy + \bar{x}y$.



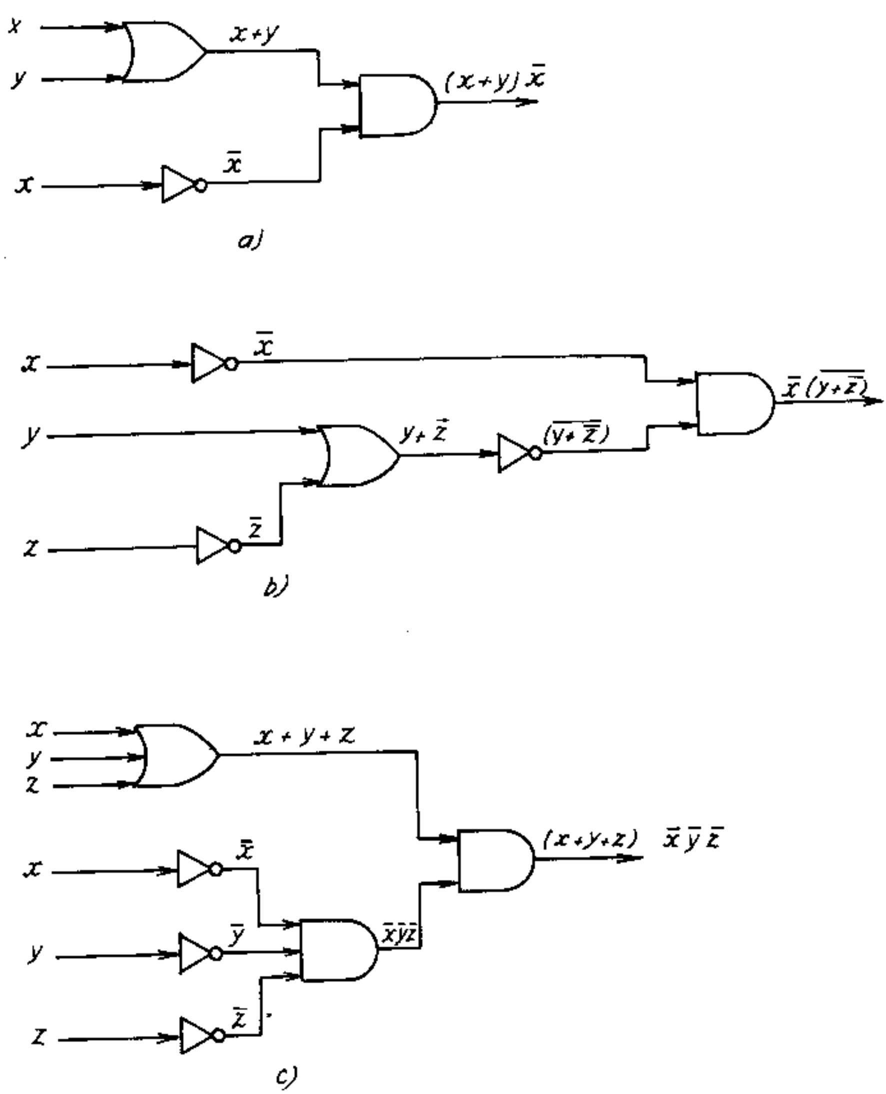
Hình 3. Hai cách vẽ cùng một mạch.

Ví dụ 1. Dựng các mạch tạo các đầu ra sau :

(a) $(x + y)\overline{x}$;

(b) $\overline{x}(\overline{y + \overline{z}})$ và (c) $(x + y + z)(\overline{x}\,\overline{y}\,\overline{z})$

Giải: Các mạch tạo các đầu ra như trên được cho trên Hình 4. ■



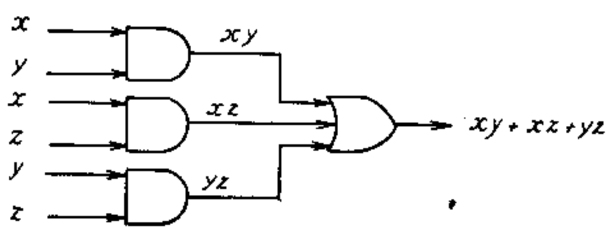
Hình 4. Các mạch tạo đầu ra cho trong ví dụ 1.

VÍ DỤ VỀ CÁC MẠCH

Chúng ta sẽ cho một số ví dụ về các mạch thực hiện một số hàm tiện ích.

Ví dụ 2. Một ủy ban gồm ba thành viên phải quyết định các vấn đề của một tổ chức. Mỗi một thành viên bỏ phiếu tán thành hoặc không cho mỗi một đề nghị được đưa ra. Một đề nghị sẽ được thông qua nếu nó nhận được ít nhất hai phiếu tán thành. Hãy thiết kế một mạch cho phép xác định được một đề nghị có được thông qua hay không.

Giải: Cho $x = 1$ nếu thành viên thứ nhất bỏ phiếu tán thành và $x = 0$ nếu thành viên đó không tán thành ; cho $y = 1$ nếu thành viên thứ hai bỏ phiếu tán thành và $y = 0$ nếu thành viên đó không tán thành ; cho $z = 1$ nếu thành viên thứ ba bỏ phiếu tán thành và $z = 0$ nếu thành viên đó không tán thành. Khi đó mạch cần được thiết kế sao cho nó tạo đầu ra bằng 1 từ các đầu vào x, y và z khi có hai hoặc nhiều hơn các biến x, y, z có giá trị là 1. Một biểu diễn của hàm Boole có giá trị đầu ra đó là $xy + xz + yz$ (xem Bài tập 6 ở Tiết 9.1). Mạch thực hiện hàm này được cho trên Hình 5.



***Hình** 5.* Mạch bỏ phiếu theo đa số.

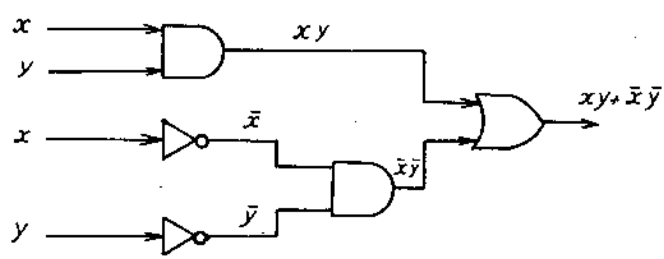
■

Ví dụ 3. Đôi khi các hệ thống đèn cố định được điều khiển bởi nhiều công tắc. Các mạch cần được thiết kế sao cho khi ấn (hoặc gạt) một công tắc bất kỳ hệ thống đèn đang tắt sẽ bật và đang bật sẽ tắt. Hãy thiết kế một mạch thực hiện điều đó khi có hai công tắc và khi có ba công tắc.

Giải: Chúng ta sẽ bắt đầu bằng việc thiết kế mạch điện điều khiển hệ thống đèn khi dùng hai công tắc khác nhau. Giả xử $x = 1$ khi công tắc thứ nhất đóng và $x = 0$ khi nó mở, giả sử $y = 1$ khi công tắc thứ hai đóng và $y = 0$ khi nó mở. Giả sử $F(x,y) = 1$ khi đèn sáng và $F(x,y) = 0$ khi đèn tắt. Chúng ta hoàn toàn có thể tùy chọn để đèn sẽ sáng khi hai công tắc đều đóng, tức là $F(1,1) = 1$. Điều này sẽ xác định các giá trị khác cùng hàm F. Khi một trong hai công tắc mở đèn sẽ tắt, tức là $F(1,0) = F(0,1) = 0$. Khi công tắc còn lại cũng mở nốt đèn lại sáng, tức $F(0,0) = 1$. Các giá trị đó được cho trong Bảng 1. Chúng ta thấy rằng $F(x,y) = xy + \bar{x}\,\bar{y}$. Hàm này được thực hiện bởi mạch được cho trên Hình 6.

BẢNG 1

x	y	F
1	1	1
1	0	0
0	1	0
0	0	1

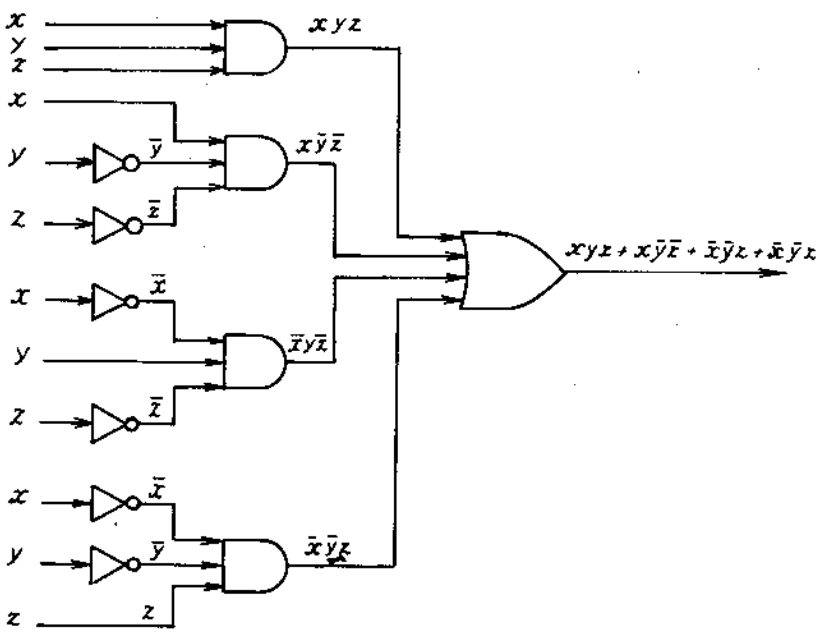


Hình 6. Mạch đèn được điều khiển bởi hai công tắc.

Bây giờ chúng ta sẽ thiết kế mạch dùng ba công tắc. Cho x, y, z là ba biến Boole chỉ sự đóng mở của các công tắc. Giả sử $x = 1$ khi công tắc một đóng và $x = 0$ khi nó mở ; giả sử $y = 1$ khi công tắc hai đóng và $y = 0$ khi nó mở ; giả sử $z = 1$ khi công tắc ba đóng và $z = 0$ khi nó mở. Giả sử $F(x,y,z) = 1$ khi đèn bật và $F(x,y,z) = 0$ khi đèn tắt. Điều này sẽ xác định các giá trị khác của F. Khi một công tắc mở đèn sẽ tắt, tức là $F(1,1,0) = F(1,0,1) = F(0,1,1) = 0$. Khi thêm một công tắc nữa mở đèn lại sáng, tức là $F(1,0,0) = F(0,1,0) = F(0,0,1) = 1$. Cuối cùng khi cả ba công tắc đều mở đèn lại tắt, tức là $F(0,0,0) = 0$. Các giá trị của hàm F được cho trong Bảng 2.

BẢNG 2			
x	y	z	F(x,y,z)
1	1	1	1
1	1	0	0
1	0	1	0
1	0	0	1
0	1	1	0
0	1	0	1
0	0	1	1
0	0	0	0

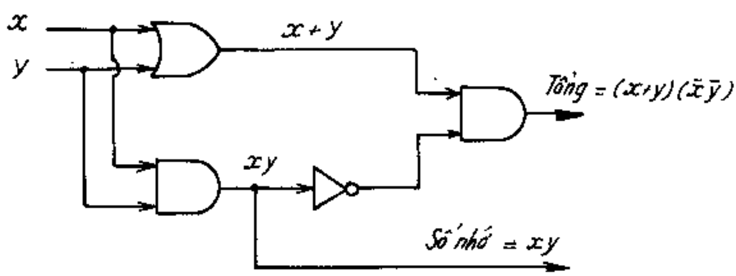
Hàm F có thể được biểu diễn bởi khai triển tổng các tích, cụ thể là $F(x,y,z) = xyz + x\,\bar{y}\,\bar{z} + \bar{x}\,y\,\bar{z} + \bar{x}\,\bar{y}\,z$. Mạch thực hiện hàm này được cho trên Hình 7.



Hình 7. Mạch đèn được điều khiển bởi ba công tắc.

BỘ CỘNG

Chúng ta sẽ minh họa các mạch logic có thể được dùng như thế nào để thực hiện phép cộng hai số nguyên dương từ các khai triển nhị phân của chúng. Chúng ta sẽ xây dựng sơ đồ mạch để làm việc này từ một số mạch thành phần. Trước hết, chúng ta sẽ xây dựng một mạch có thể được dùng để tìm $x + y$ với x và y là hai bit. Đầu vào mạch này sẽ là x và y vì mỗi chúng đều có giá trị 0 hoặc 1. Đầu ra sẽ gồm hai bit, cụ thể là s và c, trong đó s là bit tổng và c là bit nhớ. Mạch này được gọi là mạch **nhiều đầu ra** vì nó có hơn một đầu ra. Mạch mà chúng ta đang thiết kế được gọi là **bộ nửa cộng**, vì nó cộng hai bit mà không xét đến số nhớ từ phép cộng trước. Chúng ta biểu diễn đầu vào và đầu ra của bộ nửa cộng trong Bảng 3. Từ Bảng 3 ta thấy rằng $c = xy$ và $s = x\bar{y} + \bar{x}y = (x + y)(\overline{xy})$. Do đó, mạch cho trên Hình 8 sẽ tính bit tổng s và bit nhớ c từ các bit x và y.



Hình 8. Bộ nửa cộng.

BẢNG 3. Đầu vào và đầu ra đối với bộ nửa cộng

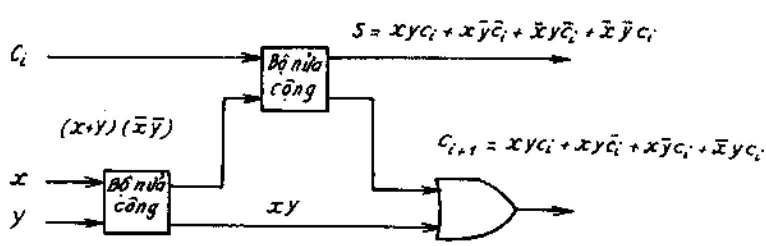
x	y	s	c
1	1	0	1
1	0	1	0
0	1	1	0
0	0	0	0

Chúng ta sẽ dùng **bộ cộng đầy đủ** để tính bit tổng và bit nhớ khi hai bit được cộng cùng với số nhớ. Đầu vào đối với bộ cộng đầy đủ là các bit x *và* y và số nhớ c_i. Đầu ra là bit tổng s và hit nhớ mới c_{i+1}. Đầu vào và đầu ra của bộ cộng đầy đủ được cho trong Bảng 4.

BẢNG 4. Đầu vào và đầu ra đối với bộ cộng đầy đủ

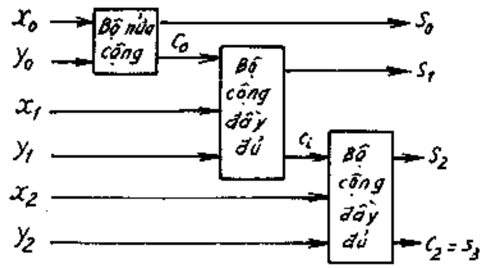
Đầu vào			Đầu ra	
x	y	c_i	s	c_{i+1}
1	1	1	1	1
1	1	0	0	1
1	0	1	0	1
1	0	0	1	0
0	1	1	0	1
0	1	0	1	0
0	0	1	1	0
0	0	0	0	0

Hai đầu ra của bộ cộng đầy đủ, tức là bit tổng và bit nhớ c_{i+1} được biểu diễn bởi khai triển tổng các tích $xyc_i + x\bar{y}\bar{c}_i + \bar{x}y\bar{c}_i + \bar{x}\bar{y}c_i$ và $xyc_i + xy\bar{c}_i + x\bar{y}c_i + \bar{x}yc_i$, tương ứng. Tuy nhiên, thay vì thiết kế bộ cộng đầy đủ từ các phần tử cơ bản, ta sẽ dùng các bộ nửa cộng để tạo các đầu ra mong muốn. Mạch bộ cộng đầy đủ dùng các bộ nửa cộng được cho trên Hình 9.



Hình 9. Bộ cộng đầy đủ

Cuối cùng Hình 10 cho thấy các bộ cộng đầy đủ và bộ nửa cộng đã được sử dụng để cộng hai số nguyên dương ba bit $(x_1\ x_1\ x_0)_2$ và $(y_2\ y_1\ y_0)_2$ để tạo tổng $(s_3\ s_2\ s_1\ s_0)_2$ như thế nào. Chú ý rằng s_3 là bit bậc cao nhất trong tổng được cho bởi số nhớ c_2.

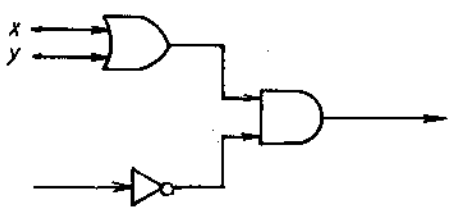


Hình 10. Cộng hai số nguyên ba bit bằng các bộ cộng đầy đủ và bộ nửa cộng.

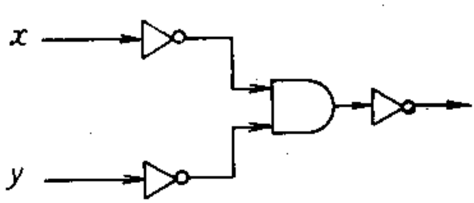
BÀI TẬP

Trong các bài tập từ 1 đến 5 hãy tìm đầu ra của các mạch đã cho

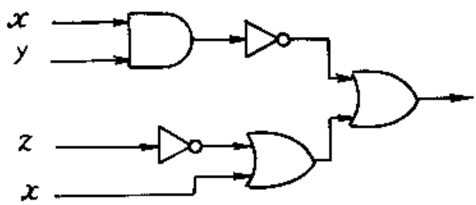
1.



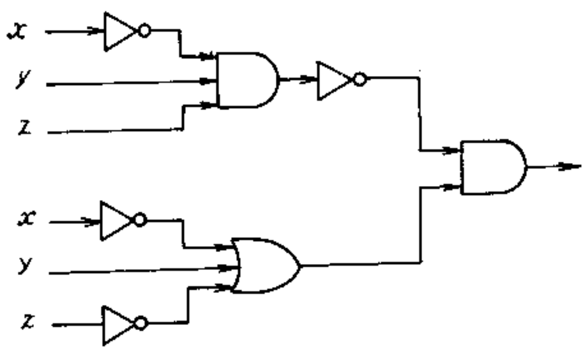
2.



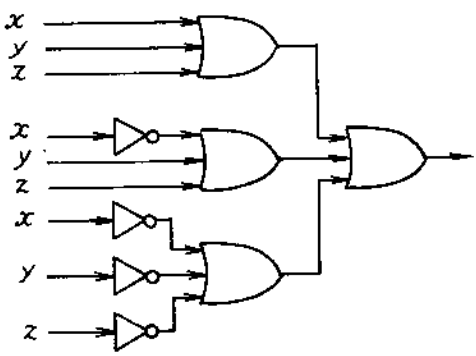
3.



4.



5.



6. Dựng các mạch gồm các bộ đảo, các cổng *AND* và *OR* để tạo các đầu ra sau :

a) $\overline{x} + y$ b) $\overline{(x + y)}\, x$

c) $xyz + \overline{x}\,\overline{y}\,\overline{z}$ d) $\overline{(\overline{x} + z)(y + \overline{z})}$

7. Thiết kế một mạch thực hiện sự bỏ phiếu theo đa số cho năm thành viên.

8. Thiết kế một mạch cho hệ thống đèn được điều khiển bởi bốn công tắc trong đó chỉ cần ấn (hoặc gạt) một công tắc bất kỳ cũng làm cho đèn đang tắt bật sáng và đèn đang sáng sẽ tắt.

9. Hãy chỉ ra cách tìm tổng của hai số nguyên năm bit khi dùng các bộ cộng đầy đủ và bộ nửa cộng.

10. Hãy xây dựng một mạch cho bộ nửa trừ bằng cách dùng các bộ đảo, các cổng *AND* và *OR*. Bộ nửa trừ có hai bit là đầu vào và tạo đầu ra gồm hiệu và phần mượn.

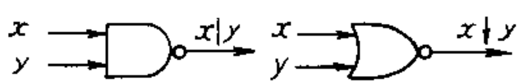
11. Hãy xây dựng một mạch cho **bộ trừ đầy đủ** bằng cách dùng các bộ đảo và các cổng *AND* và *OR*. Bộ **trừ đầy đủ** có đầu vào là hai bit và phần mượn và tạo đầu ra gồm bit hiệu và phần mượn.

12. Dùng các mạch từ các Bài tập 10 và 11 tìm hiệu của hai số nguyên bốn bit trong đó số nguyên thứ nhất lớn hơn số nguyên thứ hai.

13*. Xây dựng một mạch có nhiệm vụ so sánh hai số nguyên hai bit $(x_1x_0)_2$ và $(y_1y_0)_2$ và cho đầu ra bằng 1 khi số thứ nhất lớn hơn và bằng 0 trong trường hợp còn lại.

14*. Xây dựng một mạch tính tích của hai số nguyên hai bit $(x_1x_0)_2$ và $(y_1y_0)_2$. Mạch này cần có bốn bit đầu ra cho các bit trong tích đó.

Có hai cổng thường dược dùng trong các mạch là các cổng NAND và NOR. Khi dùng các cổng NAND hoặc NOR để biểu diễn các mạch, người ta không cần phải sử dụng các loại cổng khác. Ký hiệu dược dùng để biểu diễn các cổng này như sau :



15*. Hãy dùng các cổng *NAND* xây dựng các mạch với các đầu ra như sau :

a) $\bar{x}$ b) $x + y$

c) xy d) $x \oplus y$.

16*. Hãy dùng các cổng *NOR* để dựng các mạch có đầu ra được cho trong Bài tập 15.

17*. Hãy dùng các cổng *NAND* để dựng bộ nửa cộng.

18*. Hãy dùng các cổng *NOR* để dựng bộ nửa cộng.

Một bộ **dồn tín hiệu** *(multiplexer) là một mạch chuyển tạo một đầu ra từ tập các bit đầu vào dựa trên các giá trị của các bit điều khiển.*

19. Xây dựng một bộ dồn tín hiệu bằng cách dùng các bộ đảo, các cổng *AND* và *OR*, biết rằng nó có bốn bit đầu vào x_0, x_1, x_2 và x_3 và hai bit điều khiển c_0 và c_1. Lập một mạch sao cho x_i là đầu vào với i là giá trị thập phân của số nguyên hai bit $(c_1c_0)_2$.

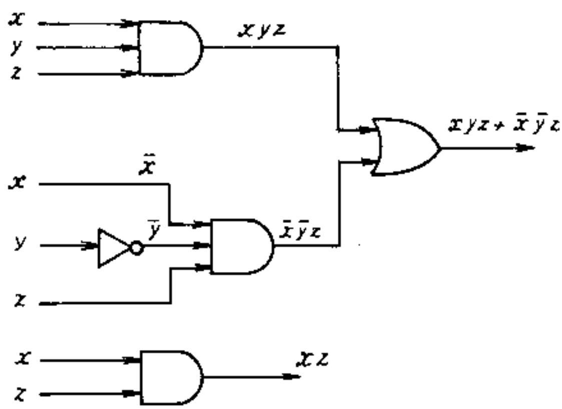
9.4. CỰC TIỂU HÓA CÁC MẠCH

MỞ ĐẦU

Hiệu quả của một mạch tổ hợp phụ thuộc vào số các cổng và sự bố trí các cổng đó. Quá trình thiết kế một mạch tổ hợp được bắt đầu bằng một bảng chỉ rõ các giá trị đầu ra đối với mỗi một tổ hợp các giá trị đầu vào. Chúng ta luôn luôn có thể sử dụng khai triển tổng các tích của mạch để tìm tập các cổng logic thực hiện mạch đó. Tuy nhiên, khai triển tổng các tích có thể chứa các số hạng nhiều hơn mức cần thiết. Các số hạng trong khai triển tổng các tích chỉ khác nhau ở một biến, sao cho trong số hạng này xuất hiện biến đó và trong số hạng kia xuất hiện phần bù của nó, đều có thể được tổ hợp lại. Ví dụ, xét mạch có đầu ra bằng 1 nếu và chỉ nếu $x = y = z = 1$ *hoặc* $x = z = 1$ và $y = 0$. Khai triển tổng các tích của mạch này là $xyz + x\bar{y}z$. Hai tích trong khai triển này chỉ khác nhau chỉ ở một biến, đó là biến y. Chúng có thể được tổ hợp lại như sau :

$$\begin{aligned} xyz + x\bar{y}z &= (y + \bar{y})xz \\ &= 1xz \\ &= xz. \end{aligned}$$

Do đó, xz là biểu thức với ít phép toán hơn biểu diễn mạch đã cho. Hai thực hiện khác nhau của mạch đó được cho trên Hình 1. Mạch thứ hai chỉ dùng một cổng, trong khi đó mạch thứ nhất phải dùng ba cổng và một bộ đảo.



Hình 1. Hai mạch có cùng đầu ra.

Ví dụ trên chứng tỏ rằng sự tổ hợp các số hạng trong khai triển tổng các tích sẽ dẫn đến một biểu thức đơn giản hơn đối với mạch. Chúng ta sẽ mô tả dưới đây hai thủ tục làm đơn giản hóa các khai triển tổng các tích. Mục đích của hai thủ tục này là tạo tổng Boole của các tích Boole chứa một số nhỏ nhất tích các tục biến sao cho các tích này lại chứa một số ít nhất các tục biến trong số tất cả những tổng các tích biểu diễn cùng một hàm Boole.

Các kỹ thuật được mô tả trong tiết này dùng để đơn giản hóa những khai triển tổng các tích vẫn còn giữ nguyên giá trị thực tiễn. Tuy nhiên, các mạch hiện đại thường được xây dựng từ các loại phần tử phức tạp hơn các cổng *AND*, *OR* và các bộ đảo. Và có rất nhiều thủ tục được dùng để đơn giản hóa các mạch được xây dựng từ các loại phần tử phức tạp đó. Tuy nhiên, nhiều phương pháp đó đều dùng những ý tưởng tương tự như những ý tưởng được mô tả trong tiết này.

BẢN ĐỒ KARNAUGH

Để làm giảm số các số hạng trong một biểu thức Boole biểu diễn một mạch, ta cần phải tìm các số hạng để tổ hợp lại. Có một phương pháp đồ thị – được gọi là **bản đồ Karnaugh** hay **bìa Karnaugh** – được dùng để tìm các số hạng tổ hợp được đối với các hàm Boole có số biến tương đối nhỏ. Phương pháp mà chúng ta sắp mô tả dưới đây đã được Maurice Karnaugh đưa ra vào năm 1953. Phương pháp của ông dựa trên một công trình trước đó của E.W.Veitch. (Phương pháp này thường chỉ được áp dụng khi hàm có sáu biến hoặc ít hơn). Các bản đồ Karnaugh cho chúng ta một phương pháp trực quan để rút gọn các khai triển tổng các tích, nhưng chúng không thích hợp với việc cơ khí hóa quá trình này. Trước hết chúng ta sẽ minh họa cách dùng các bản đồ Karnaugh để rút gọn biểu thức của các hàm Boole hai biến.

Có bốn tiểu hạng khả dĩ trong khai triển tổng các tích của một hàm Boole có hai biến x và y. Một bản đồ Karnaugh đối với một hàm Boole hai biến này gồm bốn ô vuông, trong đó hình vuông biểu diễn tiểu hạng có mặt trong khai triển được ghi số 1. Các hình ô được gọi là **kề nhau** nếu các tiểu hạng mà chúng biểu diễn chỉ khác nhau một tục biến. Ví dụ, ô vuông biểu diễn $\bar{x}y$ kề với các ô vuông biểu biểu diễn xy và $\bar{x}\,\bar{y}$. Bốn ô vuông và các tiểu hạng mà chúng biểu diễn được cho trên Hình 2.

	y	$\bar{y}$
x	xy	$x\bar{y}$
$\bar{x}$	$\bar{x}y$	$\bar{x}\,\bar{y}$

Hình 2. Bản đồ Karnaugh hai biến.

	y	$\bar{y}$
x	1	
$\bar{x}$	1	

a)

	y	$\bar{y}$
x		1
$\bar{x}$	1	

b)

	y	$\bar{y}$
x		1
$\bar{x}$	1	1

c

Hình 3. Các bản đồ Karnaugh cho những khai triển tổng các tích trong Ví dụ 1.

Ví dụ 1. Tìm các bản đồ Karnaugh cho

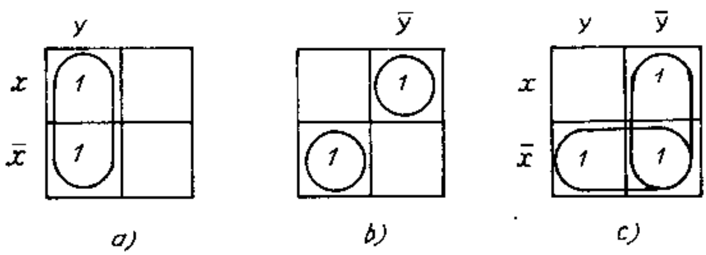
(a) $xy + \bar{x}y$ (b) $x\bar{y} + \bar{x}y$ và (c) $x\bar{y} + \bar{x}y + \bar{x}\,\bar{y}$

Giải: Chúng ta ghi số 1 vào ô vuông khi tiểu hạng được biểu diễn bởi ô đó có mặt trong khai triển tổng các tích. Ba bản đồ Karnaugh được cho trên Hình 3.

■

Chúng ta có thể nhận dạng được các tiểu hạng có thể tổ hợp được từ bản đồ Karnaugh. Bất cứ khi nào có các số 1 ở hai ô kề nhau, thì các tiểu hạng được biểu diễn bởi các ô đó đều có thể được tổ hợp lại thành một tích chỉ có một biến. Ví dụ, $x\overline{y}$ và $\overline{x}\,\overline{y}$ được biểu diễn bởi hai ô kề nhau và có thể tổ hợp lại thành $\overline{y}$, vì $x\overline{y} + \overline{x}\,\overline{y} = (x + \overline{x})\overline{y} = \overline{y}$. Hơn nữa, nếu các số 1 có trong tất cả bốn ô, thì bốn tiểu hạng có thể tổ hợp lại thành một số hạng, cụ thể là biểu thức Boole 1, không có liên quan đến một biến nào. Chúng ta sẽ khoanh các khối ô trong bản đồ Karnaugh, đó là những ô biểu diễn các tiểu hạng có thể tổ hợp lại và sau đó tìm tổng tương ứng của các tích. Mục tiêu là phải nhận dạng các khối khả dĩ lớn nhất và phủ tất cả các ô chứa số 1 bằng số ít nhất các khối bằng cách dùng trước hết các khối lớn nhất và luôn luôn dùng các khối khả dĩ lớn nhất.

Ví dụ 2. Rút gọn các khai triển tổng các tích được cho trong Ví dụ 1.



Hình 4. Đơn giản hóa khai triển tổng cách tích từ Ví dụ 1.

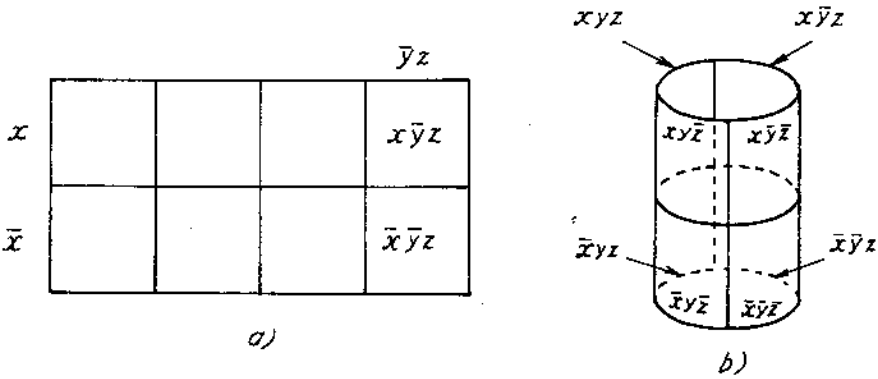
Giải: Việc nhóm các tiểu hạng được chỉ ra trong Hình 4 bằng cách sử dụng các bản đồ Karnaugh cho các khai triển đó. Khai triển cực tiểu của tổng các tích này tương ứng là :

(a) y ; (b) $x\overline{y} + \overline{x}y$ và (c) $\overline{x} + \overline{y}$.

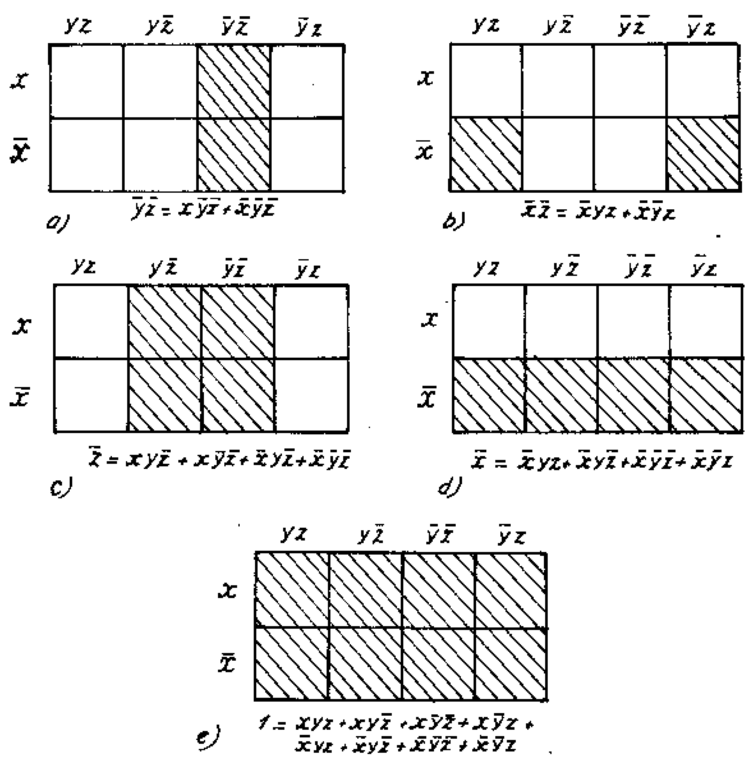
■

Bản đồ Karnaugh ba biến là một hình chữ nhật được chia thành tám ô. Các ô đó biểu diễn tám tiểu hạng ba biến khả dĩ. Hai ô được gọi là kề nhau nếu các tiểu hạng mà chúng biểu diễn chỉ khác nhau một tục biến.

Một trong các cách để lập bản đồ Karnaugh ba biến được cho trên hình 5(a). Bản đồ Karnaugh này có thể được xem như nằm trên một mặt trụ, như được biểu diễn trên hình 5(b). Trên mặt trụ, hai ô có biên chung nếu và chỉ nếu chúng là kề nhau.



Hình 5. Bản đồ Karnaugh ba biến.



Hình 6. Các khối trong bản đồ Karnaugh ba biến.

Để rút gọn khai triển tổng các tích ba biến, chúng ta sẽ dùng bản đồ Karnaugh để nhận dạng các tiểu hạng có thể tổ hợp lại. Các khối gồm hai ô kề nhau biểu diễn cặp các tiểu hạng có thể được tổ hợp lại thành một tích của hai tục biến ; các khối 2 × 2 và 4 × 1 biểu diễn các tiểu hạng có thể tổ hợp lại thành một tục biến duy nhất ; còn khối gồm tất cả tám ô biểu diễn một tích không có một tục biến nào, cụ thể, đây là biểu thức 1. Hình 6 cho biểu diễn các khối 1 × 2, 2 × 1, 2 × 2, 4 × 1 và 4 × 2 và các tích mà chúng biểu diễn.

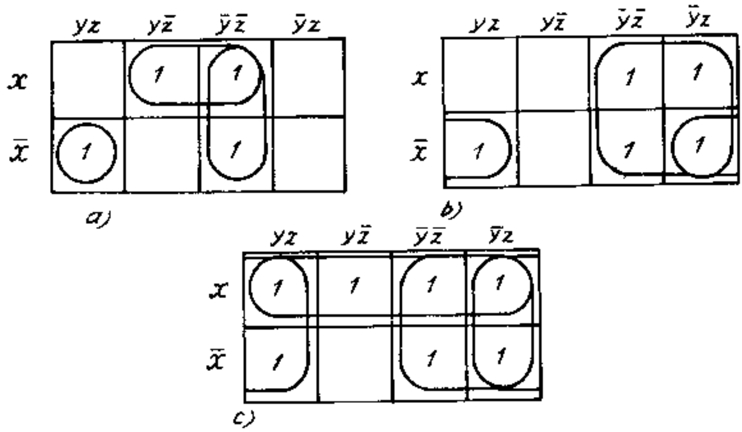
Mục tiêu là phải nhận dạng các khối khả dĩ lớn nhất trong bản đồ và phủ tất cả các ô chứa số 1 bằng một số nhỏ nhất các khối mà trước hết là khối lớn nhất. Luôn luôn phải chọn các ô khả dĩ lớn nhất. Chú ý rằng không chỉ có một cách để làm điều này. Ví dụ sau cho thấy các bản đồ Karnaugh ba biến được sử dụng như thế nào.

Ví dụ 3. Dùng các bản đồ Karnaugh ba biến để rút gọn các khai triển tổng các tích sau :

a) $xy\bar{z} + x\bar{y}\,\bar{z} + \bar{x}yz + \bar{x}\,\bar{y}\,\bar{z}$,

b) $x\bar{y}z + x\bar{y}\,\bar{z} + \bar{x}yz + \bar{x}\,\bar{y}\,z + \bar{x}\,\bar{y}\,\bar{z}$

c) $xyz + xy\bar{z} + x\bar{y}z + x\bar{y}\,\bar{z} + \bar{x}yz + \bar{x}\,\bar{y}\,z + \bar{x}\,\bar{y}\,\bar{z}$.



Hình 7. Dùng các bản đồ Karnaugh ba biến.

Giải: Bản đồ Karnaugh cho những khai triển tổng các tích này được cho trên hình 7. Việc nhóm thành các khối cho thấy rằng các khai triển cực tiểu thành các tổng Boole của các tích Boole là :

a) $x\bar{z} + \bar{y}\,\bar{z} + \bar{x}\,yz$, b) $\bar{y} + \bar{x}z$ và c) $x + \bar{y} + z$.

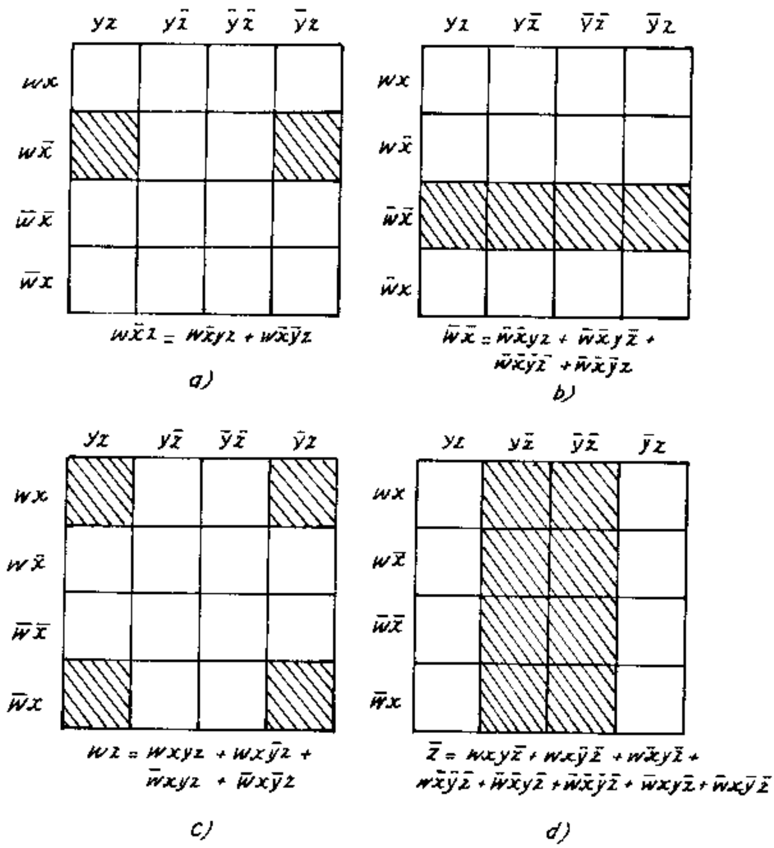
■

Bản đồ Karnaugh bốn biến là một hình vuông được chia làm 16 ô. Các ô này biểu diễn 16 tiểu hạng khả dĩ. Một trong những cách lập bản đồ Karnaugh bốn biến được cho trên Hình 8.

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
ωx	ωxyz	$\omega xy\bar{z}$	$\omega x\bar{y}\bar{z}$	$\omega x\bar{y}z$
$\omega\bar{x}$	$\omega\bar{x}yz$	$\omega\bar{x}y\bar{z}$	$\omega\bar{x}\bar{y}\bar{z}$	$\omega\bar{x}\bar{y}z$
$\bar{\omega}\bar{x}$	$\bar{\omega}\bar{x}yz$	$\bar{\omega}\bar{x}y\bar{z}$	$\bar{\omega}\bar{x}\bar{y}\bar{z}$	$\bar{\omega}\bar{x}\bar{y}z$
$\bar{\omega}x$	$\bar{\omega}xyz$	$\bar{\omega}xy\bar{z}$	$\bar{\omega}x\bar{y}\bar{z}$	$\bar{\omega}x\bar{y}z$

Hình 8. Bản đồ Karnaugh bốn biến

Hai ô được gọi là kề nhau nếu và chỉ nếu các tiểu hạng mà chúng biểu



Hình 9. Các khối trong bản đồ Karnaugh bốn biến.

diễn chỉ khác nhau một tục biến. Do đó, mỗi một ô kề với bốn ô khác. Bản đồ Karnaugh của một khai triển tổng các tích bốn biến có thể được xem như nằm trên mặt một hình xuyến (giống như chiếc xăm ô tô – ND), sao cho các ô kề nhau đều có biên chung (xem Bài tập 20). Sự rút gọn một khai triển tổng các tích bốn biến được thực hiện bằng cách nhận dạng các khối gồm 2, 4, 8 hoặc 16 ô biểu diễn các tiểu hạng có thể tổ hợp lại được. Mỗi ô biểu diễn một tiểu hạng hoặc được dùng để lập một tích có ít tục biến hơn hoặc được đưa vào trong khai triển. Hình 9 cho một số ví dụ về các khối biểu diễn các tích có ba tục biến, các tích có hai tục biến, và một tục biến duy nhất.

Cũng như trong trường hợp bản đồ Karnaugh hai và ba biến, mục tiêu là cần phải nhận dạng các khối lớn nhất có chứa các số 1 trong bản đồ và phủ tất cả các số 1 bằng cách dùng một số ít nhất các khối, mà trước hết là các khối lớn nhất. Ví dụ sau minh họa các bản đồ Karnaugh bốn biến được dùng như thế nào.

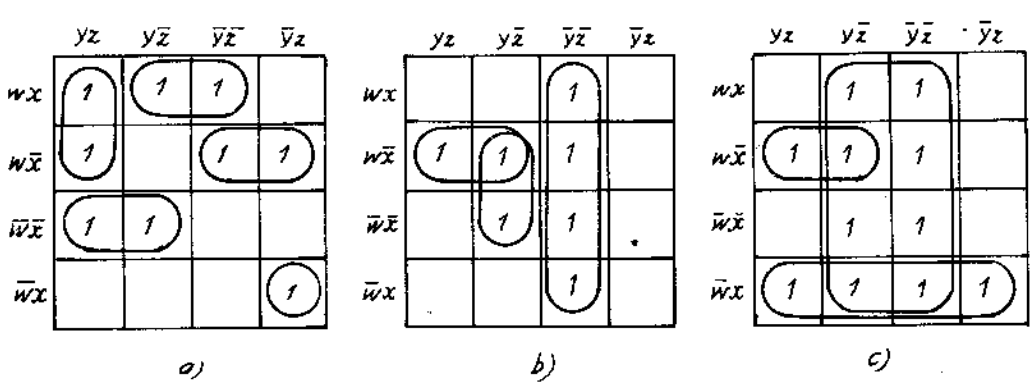
Ví dụ 4. Dùng các bản đồ Karnaugh để rút gọn các khai triển tổng các tích sau :

(a) $\omega xyz+\omega xy\bar{z}+\omega x\bar{y}\bar{z}+\omega\bar{x}yz+\omega\bar{x}\bar{y}z+\omega\bar{x}\bar{y}\bar{z}+\bar{\omega}x\bar{y}z+\bar{\omega}\bar{x}yz+\bar{\omega}\bar{x}y\bar{z}$

(b) $\omega x\bar{y}\bar{z}+\omega\bar{x}yz+\omega\bar{x}y\bar{z}+\omega\bar{x}\bar{y}\bar{z}+\bar{\omega}x\bar{y}\bar{z}+\bar{\omega}\bar{x}y\bar{z}+\bar{\omega}\bar{x}\bar{y}\bar{z}$

(c) $\omega xy\bar{z}+\omega x\bar{y}\bar{z}+\omega\bar{x}yz+\omega\bar{x}y\bar{z}+\omega\bar{x}\bar{y}\bar{z}+\bar{\omega}xyz+\bar{\omega}xy\bar{z}+\bar{\omega}x\bar{y}\bar{z}+\bar{\omega}x\bar{y}z+\bar{\omega}\bar{x}y\bar{z}+\bar{\omega}\bar{x}\bar{y}\bar{z}$

Giải: Bản đồ Karnaugh cho các khai triển trên được cho trên Hình 10. Dùng các khối đã vạch trên hình có thể dẫn tới tổng các tích sau



Hình 10. Dùng các bản đồ Karnaugh bốn biến.

a) $w yz + wx\bar{z} + w\bar{x}\,\bar{y} + \bar{w}\,\bar{x}y + \bar{w}\,x\,\bar{y}\,z$, b) $\bar{y}\,\bar{z} + w\bar{x}y + \bar{x}\,y\,\bar{z}$ và c) $\bar{z} + \bar{w}x + w\bar{x}y$. Độc giả nên xác định xem có còn cách chọn các khối khác trong mỗi câu, dẫn đến những tổng khác của các tích biểu diễn các hàm Boole hay không.

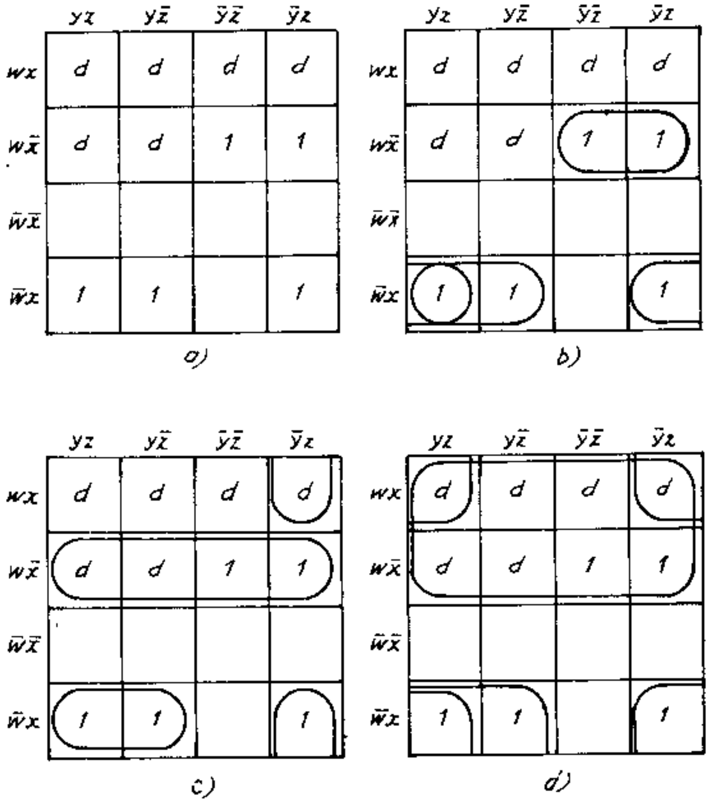
■

CÁC ĐIỀU KIỆN *"KHÔNG CẦN QUAN TÂM"*

Trong một số mạch, chúng ta chỉ quan tâm đầu ra đối với một số tổ hợp của các giá trị đầu vào, vì những tổ hợp khác của các giá trị đầu vào không bao giờ xảy ra. Điều này cho phép chúng ta tự do tạo một mạch đơn giản với đầu ra đúng như mong muốn, vì các giá trị đầu ra đối với tất cả các tổ hợp không bao giờ xảy ra có thể được chọn một cách tùy ý. Các giá trị của hàm đối với những tổ hợp này được gọi là các **điều kiện** *không cần quan tâm*. Người ta sẽ dùng chữ cái d (hoặc dấu $\times$) trong bản đồ Karnaugh để đánh dấu tổ hợp các giá trị của các biến mà đối với chúng hàm có thể được gán tùy ý. Trong quá trình rút gọn, chúng ta có thể gán giá trị 1 cho những tổ hợp các giá trị đầu vào đó và điều này sẽ dẫn tới các khối lớn nhất trong bản đồ Karnaugh. Điều này được minh họa trong ví dụ sau.

Ví dụ 5. Một cách để mã các khai triển thập phân là dùng bốn bit của khai triển nhị phân đối với mỗi chữ số trong khai triển thập phân. Ví dụ, số 873 được mã thành 1000 0111 0011. Sự mã hóa này của khai triển thập phân được gọi là **khai triển thập phân mã nhị phân**. Vì có 16 khối bốn bit mà chỉ có 10 chữ số thập phân, nên có sáu tổ hợp của bốn bit không được dùng để mã hóa các chữ số. Giả sử rằng cần phải dựng một mạch cho đầu ra là 1 nếu chữ số thập phân là 5 hoặc lớn hơn và cho đầu ra là 0 nếu chữ số thập phân nhỏ hơn 5. Làm thế nào có thể xây dựng một cách đơn giản mạch này bằng cách dùng các cổng OR, AND và các bộ đảo ?

Giải: Giả sử $F(w,x,y,z)$ là đầu ra của mạch, trong đó $wxyz$ là khai triển nhị phân của một chữ số thập phân. Các giá trị của F được cho trong Bảng 1. Bản đồ Karnaugh cho F với chữ d ghi ở các vị trí (ô) *không cần quan tâm* được cho trên Hình 11a. Chúng ta có thể bao hàm hoặc loại trừ các ô này ra khỏi các khối. Điều này cho phép chúng ta có nhiều khả năng lựa chọn các khối. Ví dụ, việc loại trừ tất cả các ô có ghi chữ d và tạo các khối như được chỉ ra trên Hình 11b sẽ cho ta biểu thức $w\bar{x}\,\bar{y} + \bar{w}xy + \bar{w}x\,z$. Còn việc bao hàm một số ô có ghi chữ d và loại đi



Hình 11. Bản đồ Karnaugh cho hàm F có chỉ rõ các vị trí *không cần quan tâm.*

BẢNG 1					
Chữ số	w	x	y	z	F
0	0	0	0	0	0
1	0	0	0	1	0
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	1
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	1

một số ô khác, rồi tạo các khối như được chỉ ra trên hình 11c sẽ cho ta biểu thức $\omega\bar{x}\,\bar{y} + \bar{\omega}xy + x\,\bar{y}\,z$. Cuối cùng, nếu bao hàm tất cả các ô có ghi chữ d và dùng các khối được chỉ ra trên hình 11d sẽ cho ta biểu thức đơn giản nhất có thể được, cụ thể là, $F(\omega,x,y,z) = \omega + xy + xz$. ∎

PHƯƠNG PHÁP QUINE - McCLUSKEY

Chúng ta đã thấy rằng các bản đồ Karnaugh có thể được dùng để tạo biểu thức cực tiểu của các hàm Boole như tổng của các tích Boole. Tuy nhiên, các bản đồ Karnaugh sẽ rất khó dùng khi số biến lớn hơn bốn. Hơn nữa, việc dùng các bản đồ Karnaugh lại dựa trên việc rà soát trực quan để nhận dạng các số hạng cần được nhóm lại. Vì những nguyên nhân đó, cần phải có một thủ tục rút gọn những khai triển tổng các tích có thể cơ khí hóa được. Phương pháp Quine–McCluskey là một thủ tục như vậy. Nó có thể được dùng cho các hàm Boole có số biến bất kỳ. Phương pháp này được W.V.Quine và E.J. McCluskey con phát triển vào những năm 1950. Về cơ bản, phương pháp Quine–McCluskey có hai phần. Phần đầu là tìm các số hạng là ứng viên để đưa vào khai triển cực tiểu như một tổng các tích Boole. Phần thứ hai là xác định xem trong số các ứng viên đó, các số hạng nào là thực sự dùng được. Chúng ta sẽ lấy một ví dụ để minh họa thủ tục này được tiến hành như thế nào.

Ví dụ 6. Chúng ta sẽ dùng phương pháp Quine–McCluskey để tìm biểu thức cực tiểu tương đương với

$$xyz + x\bar{y}z + \bar{x}yz + \bar{x}\,\bar{y}z + \bar{x}\,\bar{y}\,\bar{z}$$

Ta sẽ biểu diễn các tiểu hạng trong khai triển trên bằng các xâu bit. Bit đầu tiên sẽ là 1 nếu xuất hiện x và là 0 nếu xuất hiện $\bar{x}$. Bit thứ hai sẽ là 1 nếu xuất hiện y và là 0 nếu xuất hiện $\bar{y}$. Bit thứ ba sẽ là 1 nếu xuất hiện z và là 0 nếu xuất hiện $\bar{z}$. Sau đó chúng ta sẽ nhóm các số hạng theo số các số 1 trong các xâu bit tương ứng. Thông tin này được cho trong Bảng 2.

Các tiểu hạng có thể được tổ hợp lại là những số hạng chỉ khác nhau một tục biến. Do đó, hai số hạng có thể tổ hợp được sẽ chỉ khác nhau một con số 1 trong các xâu bit biểu diễn các số hạng đó. Khi hai tiểu hạng được tổ hợp thành một tích, tích này sẽ chứa hai tục biến. Tích có hai tục biến được biểu diễn bằng một dấu gạch ngang để chỉ biến không xuất hiện. Ví dụ, tiểu hạng $x\bar{y}z$ và $\bar{x}\bar{y}z$ được biểu diễn bằng các xâu bit 101 và 001 có thể được tổ hợp thành $\bar{y}z$ được biểu diễn bằng xâu −01. Tất cả các cặp tiểu hạng tổ hợp được và tích tạo thành từ các tổ hợp đó được cho trong Bảng 3.

BẢNG 2		
Tiểu hạng	**Xâu bit**	**Số các số 1**
xyz	111	3
$x\bar{y}z$ $\bar{x}yz$	101 011	2 2
$\bar{x}\bar{y}z$	001	1
$\bar{x}\bar{y}\bar{z}$	000	0

Tiếp theo, tất cả các cặp tích có hai tục biến có thể tổ hợp được sẽ được tổ hợp thành số hạng có một tục biến. Hai tích như vậy có thể tổ hợp được nếu chúng chứa tục hiến của cùng hai biến. Các tục biến này chỉ khác nhau đối với một trong hai biến đó. Nói theo ngôn ngữ các xâu biểu diễn các tích đó, thì hai xâu đó phải có cái gạch ngang ở cùng một vị trí và chỉ khác nhau ở một trong hai vị trí còn lại. Chúng ta có thể tổ hợp các tích yz và $\bar{y}z$ được biểu diễn bởi các xâu −11 và −01 thành z – được biểu diễn bởi xâu −−1. Tất cả những tổ hợp có thể được tạo theo cách đó được cho trong bảng 3.

BẢNG 3							
			Bước 1			**Bước 2**	
	Số Hạng	Xâu bit		Số Hạng	Xâu bit	Số Hạng	Xâu bit
1	xyz	111	(1,2)	xz	1-1	(1,2,3,4) z	--1
2	$x\bar{y}z$	101	(1,3)	yz	-11		
3	$\bar{x}yz$	011	(2,4)	$\bar{y}z$	-01		
4	$\bar{x}\bar{y}z$	001	(3,4)	$\bar{x}z$	0-1		
5	$\bar{x}\bar{y}\bar{z}$	000	(4,5)	$\bar{x}\bar{y}$	00-		

Trong Bảng 3 chúng ta cũng chỉ ra những số hạng đã được dùng để tạo ra các tích có số tục biến nhỏ hơn nhưng không nhất thiết có mặt trong biểu thức cực tiểu. Bước tiếp sau là nhận dạng tập cực tiểu các tích cần thiết để biểu diễn hàm Boole. Chúng ta sẽ bắt đầu với tất cả các tích chưa được dùng để xây dựng các tích có số tục biến ít hơn (Ví dụ, z và $\bar{x}\bar{y}$ trong ví dụ này). Tiếp sau, chúng ta lập Bảng 4, trong đó có một dòng dành cho mỗi tích ứng viên đã được tạo ra bằng cách tổ hợp các số hạng gốc (ban đầu) và một cột dành cho mỗi số hạng gốc. Chúng ta sẽ ghi dấu X ở vị trí nếu số hạng gốc trong khai triển tổng các tích đã được dùng để tạo tích ứng viên đó. Trong trường hợp này ta nói tích ứng viên đã **phủ** tiểu hạng gốc. Chúng ta cần phải bao hàm ít nhất một tích phủ mỗi một tiểu hạng gốc. Do đó, bất cứ khi nào chỉ có một dấu *X trong một cột trong bảng, thì tích tương ứng với hàng có X* đó sẽ cần phải được sử dụng. Từ Bảng 4 ta thấy rằng cả z lẫn $\bar{x}\bar{y}$ đều là cần thiết. Do đó đáp số cuối cùng sẽ là $z + \bar{x}\bar{y}$.

■

BẢNG 4

	xyz	$x\bar{y}z$	$\bar{x}yz$	$\bar{x}\bar{y}z$	$\bar{x}\bar{y}\bar{z}$
z	x	x	x	x	
$\bar{x}\bar{y}$				x	x

Như đã được minh họa trong ví dụ 6, phương pháp Quine – McCluskey dùng các bước theo trình tự sau để rút gọn một khai triển tổng các tích :

1. Biểu diễn mỗi tiểu hạng n bit bằng một xâu bit có chiều dài n với số 1 ở vị trí thứ i nếu x_i xuất hiện và với số 0 nếu $\bar{x}_i$ xuất hiện.

2. Nhóm các xâu bit theo số các số 1 trong chúng.

3. Xác định tất cả các tích $n - 1$ biến có thể tạo thành bằng cách lấy tổng Boole các tích trong khai triển đó. Các tiểu hạng có thể tổ hợp được biểu diễn bằng các xâu bit chỉ khác nhau ở một vị trí. Biểu diễn các tích $n - 1$ biến này bằng các xâu chuỗi có số 1 ở vị trí thứ i nếu ở đó có x_i hoặc số 0 nếu vị trí đó có $\bar{x}_i$ hoặc là một dấu gạch ngang nếu ở đó không có một tục biến nào liên quan đến biến x_i trong tích.

4. Xác định tất cả các tích $n-2$ biến có thể được tạo thành bằng cách lấy tổng Boole của các tích $n-1$ biến đã tìm được ở bước trước. Các tích $n-1$ biến có thể tổ hợp được biểu diễn bằng các xâu bit có dấu gạch ngang ở cùng vị trí và khác nhau chỉ ở một vị trí.

5. Tiếp tục tổ hợp các tích Boole thành các tích có số biến ít hơn dài mãi có thể được.

6. Tìm tất cả các tích Boole xuất hiện nhưng không được dùng để lập tích Boole với số tục biến bớt đi 1.

7. Tìm tập nhỏ nhất các tích Boole sao cho tổng các tích này biểu diễn được hàm Boole đã cho ban đầu. Điều này được làm bằng cách lập bảng chỉ rõ các tiểu hạng nào đã được phủ bởi các tích nào. Mỗi một tiểu hạng cần phải được phủ ít nhất bởi một tích. (Đây là phần khó khăn nhất của thủ tục. Nó có thể được cơ khí hóa bằng cách dùng thủ tục quay lui).

Ví dụ cuối cùng sẽ minh họa thủ tục này được dùng như thế nào để rút gọn một khai triển tổng các tích có bốn biến.

Ví dụ 7. Dùng phương pháp Quine – McCluskey để rút gọn tổng các tích $\omega xy\bar{z} + \omega\bar{x}yz + \omega\bar{x}y\bar{z} + \bar{\omega}xyz + \bar{\omega}x\bar{y}z + \bar{\omega}\,\bar{x}yz + \bar{\omega}\,\bar{x}\,\bar{y}z$

Giải. Trước hết chúng ta sẽ biểu diễn các tiểu hạng bằng các xâu bit, rồi nhóm các số hạng theo số các số 1 trong các xâu bit đó. Điều này được thể hiện trong Bảng 5. Tất cả các tích Boole được tạo thành bằng cách lấy tổng Boole của các tích đó được cho trong Bảng 6.

BẢNG 5

Số hạng	Xâu bit	Số các số 1
$\omega xy\bar{z}$	1110	3
$\omega\bar{x}yz$	1011	3
$\bar{\omega}xyz$	0111	3
$\omega\bar{x}y\bar{z}$	1010	2
$\bar{\omega}x\bar{y}z$	0101	2
$\bar{\omega}\,\bar{x}yz$	0011	2
$\bar{\omega}\,\bar{x}\,\bar{y}z$	0001	1

BẢNG 6								
			Bước 1			Bước 2		
	Số hạng	Xâu bit	Số hạng		Xâu	Số hạng		Xâu
1	$\omega xy\overline{z}$	1110	(1,4)	$\omega y\overline{z}$	1 - 10	(3,5,6,7)	$\overline{\omega}z$	0 - - 1
2	$\omega\overline{x}yz$	1011	(2,4)	$\omega\overline{x}y$	101 -			
3	$\overline{\omega}xyz$	0111	(2,6)	$\overline{x}yz$	- 011			
4	$\omega\overline{x}y\overline{z}$	1010	(3,5)	$\overline{\omega}xz$	01 - 1			
5	$\overline{\omega}x\overline{y}z$	0101	(3,6)	$\overline{\omega}yz$	0 - 11			
6	$\overline{\omega}\,\overline{x}yz$	0011	(5,7)	$\overline{\omega}\,\overline{y}z$	0 - 01			
7	$\overline{\omega}\,\overline{x}\,\overline{y}z$	0001	(6,7)	$\overline{\omega}\,\overline{x}z$	00 - 1			

Các tích duy nhất không được dùng để tạo các tích có ít biến hơn là $\overline{\omega}z$, $\omega y\overline{z}$, $\omega\overline{x}y$ và $\overline{x}yz$. Bảng 7 cho thấy các tiểu hạng được phủ bởi các tích đó. Để phủ các tiểu hạng này, trước hết cần phải đưa vào $\overline{\omega}z$ và $\omega y\overline{z}$ vì các tích này là các tích duy nhất phủ $\overline{\omega}xyz$ và $\omega xy\overline{z}$, tương ứng. Một khi các tích này đã được đưa vào, chúng ta thấy rằng chỉ cần phải đưa vào một trong hai tích còn lại. Do đó ta có thể lấy $\overline{\omega}z + \omega y\overline{z} + \omega\overline{x}y$ hoặc $\overline{\omega}z + \omega y\overline{z} + \overline{x}yz$ như đáp số cuối cùng

■

BẢNG 7							
	$\omega xy\overline{z}$	$\omega\overline{x}yz$	$\overline{\omega}xyz$	$\omega\overline{x}y\overline{z}$	$\overline{\omega}x\overline{y}z$	ωxyz	$\overline{\omega}\,\overline{x}\,\overline{y}z$
$\overline{\omega}z$			x		x	x	x
$\omega y\overline{z}$	x			x			
$\omega\overline{x}y$		x		x			
$\overline{x}yz$		x				x	

BÀI TẬP

1. a) Vẽ bản đồ Karnaugh đối với một hàm hai biến và ghi số 1 vào ô biểu diễn $\overline{x}y$.

   b) Các tiểu hạng nào được biểu diễn bởi các ô kề với ô nói trên.

2. Tìm khai triển tổng các tích được biểu diễn bởi các bản đồ Karnaugh sau :

	y	$\bar{y}$
x	1	
$\bar{x}$	1	1

a)

	y	$\bar{y}$
x	1	1
$\bar{x}$		

b)

	y	$\bar{y}$
x	1	1
$\bar{x}$	1	1

c)

3. Vẽ các bản đồ Karnaugh của những khai triển tổng các tích hai biến sau :

a) $x\bar{y}$ b) $xy + \bar{x}\,\bar{y}$ c) $xy + x\bar{y} + \bar{x}y + \bar{x}\,\bar{y}$

4. Dùng bản đồ Karnaugh để tìm khai triển cực tiểu của các hàm hai biến sau :

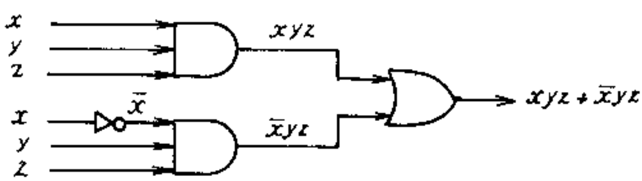
a) $\bar{x}y + \bar{x}\,\bar{y}$ b) $xy + x\bar{y}$ c) $xy + x\bar{y} + \bar{x}y + \bar{x}\,\bar{y}$.

5. a) Vẽ bản đồ Karnaugh cho một hàm ba biến. Ghi số 1 vào ô biểu diễn $\bar{x}y\bar{z}$.

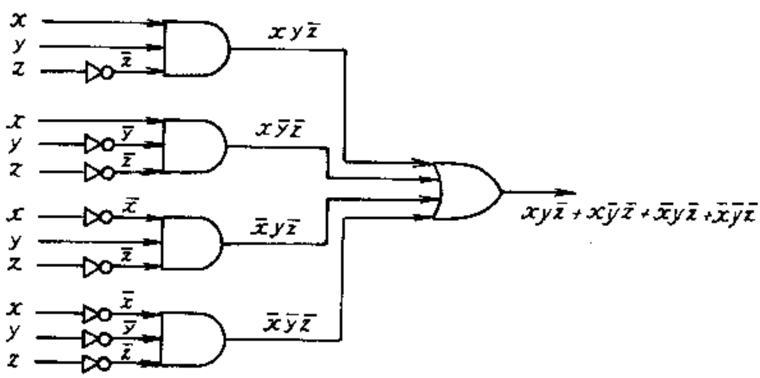
b) Các tiểu hạng nào được biểu diễn bởi các ô kề với ô nói trên.

6. Dùng các bản đồ Karnaugh để tìm các mạch đơn giản hơn có cùng đầu ra đối với các mạch sau :

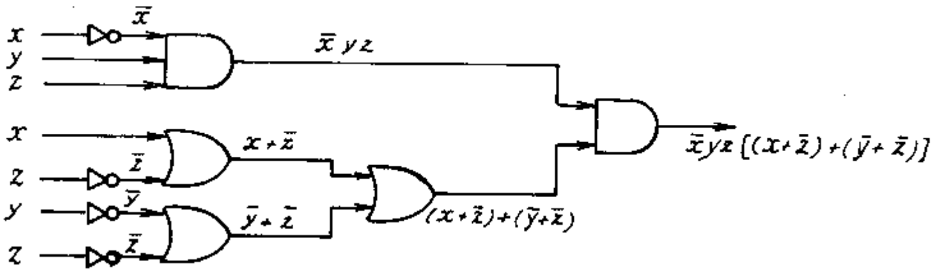
a)



b)



c)



7. Vẽ các bản đồ Karnaugh của những khai triển tổng các tích Boole ba biến sau

a) $x\bar{y}\bar{z}$

b) $\bar{x}yz + \bar{x}\,\bar{y}\,\bar{z}$

c) $xyz + xy\bar{z} + \bar{x}\,y\,\bar{z} + \bar{x}\,\bar{y}\,z$.

8. Dùng các bản đồ Karnaugh tìm khai triển cực tiểu của các hàm ba biến sau :

a) $\bar{x}\,yz + \bar{x}\,\bar{y}\,z$

b) $xyz + xy\bar{z} + \bar{x}yz + \bar{x}y\bar{z}$

c) $xy\bar{z} + x\,\bar{y}\,z + x\,\bar{y}\,\bar{z} + \bar{x}\,yz + \bar{x}\,\bar{y}\,z$

d) $xyz + x\bar{y}\,z + x\bar{y}\,\bar{z} + \bar{x}\,yz + \bar{x}\,y\,\bar{z} + \bar{x}\,\bar{y}\,\bar{z}$.

9. a) Vẽ bản đồ Karnaugh cho một hàm bốn biến. Ghi số 1 ở ô biểu diễn số hạng $\bar{\omega}\,xy\bar{z}$.

b) Các tiểu hạng nào được biểu diễn bởi các ô kề với ô nói trên.

10. Dùng bản đồ Karnaugh tìm khai triển cực tiểu của các hàm bốn biến sau :

a) $\omega xyz + \omega x\bar{y}z + \omega x\bar{y}\,\bar{z} + \omega\bar{x}y\bar{z} + \omega\bar{x}\,\bar{y}\,z$

b) $\omega xy\bar{z} + \omega x\bar{y}z + \omega\bar{x}yz + \bar{\omega}x\bar{y}z + \bar{\omega}\,\bar{x}y\bar{z} + \bar{\omega}\,\bar{x}\,\bar{y}z$

c) $\omega xyz + \omega xy\bar{z} + \omega x\bar{y}z + \omega\bar{x}\,\bar{y}z + \omega\bar{x}\,\bar{y}\,\bar{z} + \bar{\omega}x\bar{y}z + \bar{\omega}\,\bar{x}y\bar{z} + \bar{\omega}\,\bar{x}\,\bar{y}z$

d) $\omega xyz + \omega xy\bar{z} + \omega x\bar{y}z + \omega\bar{x}yz + \omega\bar{x}y\bar{z} + \bar{\omega}xyz + \bar{\omega}\,\bar{x}yz + \bar{\omega}\bar{x}y\bar{z} + \bar{\omega}\,\bar{x}\,\bar{y}\,z$

11. a) Một bản đồ Karnaugh năm biến có bao nhiêu ô ?

b) Trong một bản đồ Karnaugh năm biến có bao nhiêu ô kề với một ô đã cho ?

12*. Dùng các bản đồ Karnaugh tìm khai triển cực tiểu của các hàm Boole có đầu vào là mã nhị phân của các chữ số thập phân và tạo một đầu ra là 1 nếu và chỉ nếu chỉ số tương ứng với đầu vào là :

a) một số lẻ

b) là số không chia hết cho 3

c) không phải là 4, 5 hoặc 6.

13*. Giả sử một ủy ban có năm thành viên, nhưng Smith và Jones luôn bỏ phiếu ngược với Marcus. Hãy thiết kế một mạch thực hiện việc bỏ phiếu theo đa số của ủy ban đó, có dùng đến quan hệ nói trên giữa các lá phiếu.

14. Dùng phương pháp Quine – McCluskey để rút gọn các khai triển tổng các tích Boole cho trong Ví dụ 3.

15. Cũng hỏi như trên đối với các khai triển cho trong Bài tập 8.

16. Cũng hỏi như trên đối với các khai triển cho trong Ví dụ 4.

17. Cũng hỏi như trên đối với các khai triển cho trong Bài tập 10.

18*. Hãy giải thích làm thế nào có thể dùng các bản đồ Karnaugh để rút gọn khai triển tích các tổng ba biến. (*Gợi ý* : Đánh dấu bằng số 0 tất cả các đại hạng trong khai triển và tổ hợp các khối của các đại hạng).

19. Dùng phương pháp ở Bài tập 18, hãy rút gọn khai triển tích các tổng $(x + y + z)(x + y + \overline{z})(x + \overline{y} + z)(x + \overline{y} + z)(\overline{x} + y + z)$.

20*. Vẽ bản đồ Karnaugh cho 16 tiểu hạng bốn biến trên bề mặt của hình xuyến.

21. Dùng các cổng *OR, AND* và các bộ đảo để dựng một mạch cho đầu ra bằng 1 nếu chữ số thập phân được mã hóa nhị phân chia hết cho 3 và bằng 0 trong các trường hợp còn lại.

Trong các Bài tập từ 22 đến 24, hãy tìm khai triển cực tiểu của tổng các tích Boole tương ứng với bản đồ Karnaugh đã cho, có chỉ rõ các điều kiện không cần quan tâm bằng các chữ d.

22.

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
ωx	d	1	d	1
$\omega\bar{x}$		d	d	
$\bar{\omega}\bar{x}$		d	1	
$\bar{\omega}x$		1	d	

23.

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
ωx	1			1
$\omega\bar{x}$		d	1	
$\bar{\omega}\bar{x}$		1	d	
$\bar{\omega}x$	d			d

24.

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
ωx		d	d	1
$\omega\bar{x}$	d	d	1	d
$\bar{\omega}\bar{x}$				
$\bar{\omega}x$	1	1	1	d

CÂU HỎI ÔN TẬP

1. Định nghĩa hàm Boole bậc n

2. Có bao nhiêu hàm Boole bậc 2 ?

3. Cho một định nghĩa đệ qui về tập các biểu thức Boole

4. a) Đối ngẫu của biểu thức Boole là gì ?

b) Nguyên lý đối ngẫu là gì ? Dùng nó như thế nào để tìm các hằng đẳng thức mới từ các biểu thức Boole.

5. Giải thích cách xây dựng khai triển tổng các tích của một hàm Boole.

6. a) Thế nào là tập đầy đủ các phép toán ?

b) Tập {+, .} có là đầy đủ không ?

c) Có các tập gồm chỉ một phép toán là đầy đủ không ?

7. Hãy giải thích làm thế nào có thể dùng các cổng *OR, AND* và các bộ đảo để dựng một mạch cho hệ đèn được điều khiển bởi hai công tắc.

8. Dựng bộ nửa cộng bằng cách dùng các cổng *OR, AND* và các bộ đảo.

9. Có một loại cổng duy nhất nào có thể được dùng để dựng tất cả các mạch đã được dựng từ các cổng *OR, AND* và các bộ đảo không ?

10. a) Các bản đồ Karnaugh được dùng như thế nào để rút gọn các khai triển tổng các tích có ba biến Boole ?

b) Dùng bản đồ Karnaugh để rút gọn khai triển tổng các tích $xyz + x\bar{y}z + x\bar{y}\bar{z} + \bar{x}yz + \bar{x}\bar{y}\bar{z}$.

11. a) Các bản đồ Karnaugh được dùng như thế nào để rút gọn các khai triển tổng các tích có bốn biến Boole ?

b) Dùng bản đồ Karnaugh để rút gọn khai triển tổng các tích $\omega xyz + \omega xy\bar{z} + \omega x\bar{y}z + \omega x\bar{y}\bar{z} + \omega\bar{x}yz + \omega\bar{x}\bar{y}z + \bar{\omega}xyz + \bar{\omega}\bar{x}yz + \bar{\omega}\bar{x}y\bar{z}$

12. a) Điều kiện *không cần quan tâm* là gì ?

b) Hãy cho biết điều kiện *không cần quan tâm* được dùng như thế nào để dựng một mạch bằng các cổng *OR*, *AND* và các bộ đảo cho đầu ra là 1 nếu một chữ số thập phân là 6 hoặc lớn hơn và là 0 nếu chữ số đó nhỏ hơn 6.

13. a) Hỏi phải sử dụng phương pháp Quine – McCluskey như thế nào để rút gọn các khai triển tổng các tích ?

b) Dùng phương pháp đó để rút gọn $xy\bar{z} + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z + \bar{x}\bar{y}\bar{z}$.

BÀI TẬP BỔ SUNG

1. Với các giá trị nào của các biến Boole x, y, z ta có :

a) $x + y + z = xyz$? b) $x(y + z) = x + yz$?

c) $\bar{x}\bar{y}\bar{z} = x + y + z$?

2. Cho x và y thuộc $\{0,1\}$. Hỏi có nhất thiết phải suy ra $x = y$ không nếu tồn tại một giá trị của $z \in \{0, 1\}$ sao cho

a) $xz = yz$? b) $x + z = y + z$?

c) $x \oplus z = y \oplus z$? d) $x \downarrow z = y \downarrow z$?

e) $x \mid z = y \mid z$?

Một hàm Boole F được gọi là **tự đối ngẫu** *nếu và chỉ nếu*

$$F(x_1, ..., x_n) = \overline{F(\bar{x}_1, ..., \bar{x}_n)}$$

3. Hãy cho biết các hàm nào dưới đây là tự đối ngẫu ?

a) $F(x, y) = x$ b) $F(x, y) = xy + \bar{x}\bar{y}$

c) $F(x, y) = x + y$ c) $F(x,y) = xy + \bar{x}y$

4. Hãy cho một ví dụ về một hàm Boole ba biến tự đối ngẫu.

5*. Có bao nhiêu hàm Boole bậc n tự đối ngẫu ?

Ta định nghĩa quan hệ $\leq$ trên tập các hàm Boole bậc n sao cho $F \leq G$ nếu và chỉ nếu $G(x_1, x_2, ..., x_n) = 1$ với mọi $F(x_1, x_2, ..., x_n) = 1$.

6. Hãy xác định xem $F \leqslant G$ hay $G \leqslant F$ đối với các cặp hàm sau :

a) $F(x,y) = x$, $\quad G(x,y) = x + y$

b) $F(x,y) = x + y$, $\quad G(x,y) = xy$

c) $F(x,y) = \overline{x}$, $\quad G(x,y) = x + y$

7. Chứng minh rằng nếu F và G là các hàm Boole bậc n, thì

a) $F \leqslant F + G$ $\quad$ b) $FG \leqslant F$

8. Chứng minh rằng nếu F, G và H là các hàm Boole bậc n, thì $F + G \leqslant H$ nếu và chỉ nếu $F \leqslant H$ và $G \leqslant H$.

9*. Đối với mỗi đẳng thức sau, hãy chứng minh nó là một hằng đẳng thức hoặc tìm được một tập giá trị của các biến mà đẳng thức đó không đúng :

a) $x \mid (y \mid z) = (x \mid y) \mid z$

b) $x \downarrow (y \downarrow z) = (x \downarrow y) \downarrow (x \downarrow z)$

c) $x \downarrow (y \mid z) = (x \downarrow y) \mid (x \downarrow z)$.

Người ta định nghĩa phép toán Boole $\odot$ *như sau :* $1 \odot 1 = 1$, $1 \odot 0 = 0$, $0 \odot 1 = 0$ *và* $0 \odot 0 = 1$.

10. Chứng minh rằng $x \odot y = xy + \overline{x}\,\overline{y}$.

11. Cbứng minh rằng $x \odot y = \overline{x \oplus y}$

12. Chứng minh các hằng đẳng tbức sau :

a) $x \odot x = 1$ $\quad$ b) $x \odot \overline{x} = 0$

c) $x \odot y = y \odot x$

13. Đẳng thức sau có phải là một hằng đẳng thức không

$(x \odot y) \odot z = x \odot (y \odot z)$?

14*. Hãy xác định xem tập $\{\odot\}$ có là đầy đủ không ?

15*. Trong số 16 hàm Boole với hai biến x và y có bao nhiêu hàm được biểu diễn bởi tập các phép toán sau cùng với các biến x, y và các giá trị 0, 1 ?

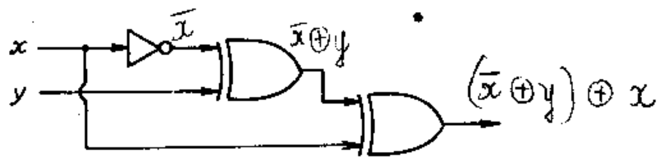
a) $\{\overline{}\}$ $\quad$ b) $\{.\}$

c) $\{+\}$ $\quad$ d) $\{., +\}$

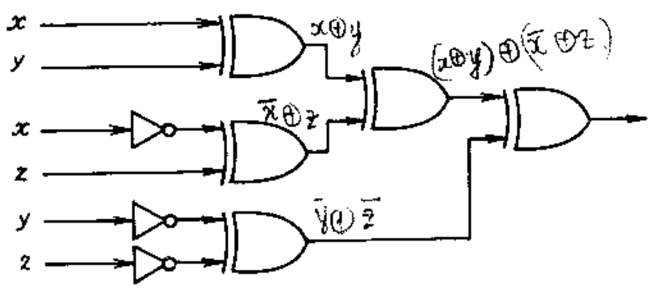
Ký hiệu của cổng XOR, cổng tạo đầu ra là $x \oplus y$ từ đầu vào là x và y, là như sau :

16. Xác định đầu ra của các mạch sau :

a)



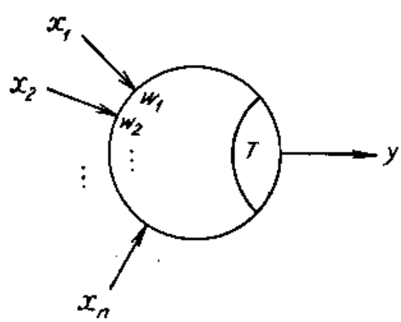
b)



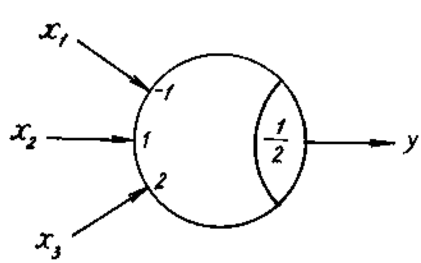
17. Hãy cho biết làm thế nào có thể xây dựng bộ nửa cộng với số cổng ít hơn so với cách dựng trên Hình 8 của Tiết 9.3 bằng cách dùng thêm các cổng XOR cùng với các cổng *OR, AND* và các bộ đảo ?

18. Hãy thiết kế một mạch để xác định xem ba hoặc nhiều hơn thành viên trong số bốn thành viên của một ủy ban có bỏ phiếu tán thành cho một đề nghị hay không, biết rằng mỗi thành viên dùng một công tắc để bỏ phiếu.

Cổng ngưỡng *là cổng tạo đầu ra y là 0 hoặc 1 với tập các giá trị đầu vào đã cho đối với các biến Boole* $x_1, x_2, ..., x_n$. *Cổng ngưỡng có* **giá trị ngưỡng** T *– đó là một số thực và các* **trọng số** $\omega_1, \omega_2, ..., \omega_n$ *cũng là các số thực. Đầu ra y của cổng ngưỡng có giá trị 1 khi và chỉ khi* $\omega_1 x_1 + \omega_2 x_2 + ... \omega_n x_n \geq T$. *Cổng ngưỡng với giá trị ngưỡng T và các trọng số* $\omega_1, \omega_2, ..., \omega_n$ *được biểu diễn bởi giản đồ sau. Các cổng ngưỡng rất tiện ích đối với việc mô hình hóa trong sinh lý học nơron và trí tuệ nhân tạo.*



19. Cổng ngưỡng biểu diễn một hàm Boole. Tìm biểu thức Boole cho hàm Boole được biểu diễn bởi cổng ngưỡng sau.



20. Hàm Boole có thể được biểu diễn bởi một cổng ngưỡng được gọi là **hàm ngưỡng**. Chứng minh rằng các hàm sau đều là hàm ngưỡng

a) $F(x) = \overline{x}$ b) $F(x,y) = x + y$

c) $F(x,y) = xy$ d) $F(x,y) = x \mid y$

e) $F(x,y) = x \downarrow y$

f) $F(x,y,z) = x + yz$

g) $F(\omega,x,y,z) = \omega + xy + z$

h) $F(\omega,x,y,z) = \omega xz + x\bar{y}z$

21*. Chứng minh rằng $F(x,y) = x \oplus y$ không phải là hàm ngưỡng.

22*. Chứng minh rằng $F(\omega,x,y,z) = \omega x + yz$ không phải là hàm ngưỡng.

BÀI TẬP TRÊN MÁY TÍNH

Viết chương trình với đầu vào và đầu ra sau :

1. Cho giá trị của hai biến Boole x và y, tìm các giá trị của $x + y$, $x \oplus y$, xy, xy và $x \downarrow y$.

2. Lập bảng liệt kê tập các giá trị của tất cả 256 hàm Boole bậc 3.

3. Cho các giá trị của hàm Boole n biến, với n là số nguyên dương, tìm khai triển tổng các tích của hàm này.

4. Cho bảng các giá trị của một hàm Boole, hãy biểu diễn hàm này bằng cách chỉ dùng các phép toán . và $\bar{}$.

5. Cho bảng các giá trị của một hàm Boole, hãy biểu diễn hàm này mà chỉ dùng các phép toán + và $\bar{}$.

6*. Cho bảng các giá trị của một hàm Boole, hãy biểu diễn hàm đó mà chỉ dùng phép toán $|$.

7*. Cũng hỏi như trên với phép toán $\downarrow$.

8. Cho bảng các giá trị của một hàm Boole bậc 3, hãy dựng bản đồ Karnaugh của nó.

9. Cũng hỏi như trên đối với hàm Boole bậc 4.

10**. Cho bảng các giá trị của một hàm Boole, hãy dùng phương pháp Quine – McCluskey để tìm biểu diễn tổng cực tiểu các tích của hàm đó.

11. Cho giá trị ngưỡng và tập các trọng số đối với một cổng ngưỡng và các giá trị của n biến Boole ở đầu vào, hãy xác định đầu ra của cổng đó.

12. Cho số nguyên dương n, hãy dựng một biểu thức Boole ngẫu nhiên n biến dưới dạng tuyển chuẩn tắc.

TÍNH TOÁN VÀ KHÁM PHÁ

Dùng các chương trình mà bạn đã viết làm các bài tập sau

1. Tính số hàm Boole bậc 7, 8, 9 và 10.

2. Lập bảng các hàm Boole bậc 3. Lập bảng các hàm Boole bậc 4.

4. Biểu diễn các biểu thức Boole ba biến dưới dạng tuyển chuẩn tắc mà chỉ dùng một phép toán *NAND* với số lượng ít nhất có thể được. Số lớn nhất các phép toán *NAND* khi đó cần phải dùng là bao nhiêu ?

5. Hãy biểu diễn các biểu thức Boole bốn biến dưới dạng tuyển chuẩn tắc mà chỉ dùng một phép toán *NOR* với số lượng ít nhất có thể được. Số lớn nhất các phép toán *NOR* khi đó cần phải dùng là bao nhiêu.

6. Tạo ngẫu nhiên 10 biểu thức Boole bốn biến khác nhau và xác định số trung bình các bước cần thiết để cực tiểu hóa chúng bằng cách dùng phương pháp Quine - Mc Cluskey.

7. Cũng hỏi như trên đối với các biểu thức năm biến.

VIẾT TIỂU LUẬN

Dùng các tư liệu ở ngoài cuốn sách này viết các tiểu luận trả lời những câu hỏi sau đây:

1. Mô tả một số máy tính ở giai đoạn đầu được chế tạo để giải một số bài toán về logic, chẳng hạn như Máy chứng minh Stanhope, Máy logic của Jevon, và Máy Marquand

2. Giải thích sự khác nhau giữa mạch tổ hợp và mạch tuần tự. Sau đó hãy giải thích *mạch lật* được dùng như thế nào để dựng các mạch tuần tự.

3. Hãy định nghĩa *thanh ghi chuyển dịch* và thảo luận cách dùng thanh ghi này. Hãy cho biết cách dùng các mạch lật và các cổng logic để tạo các thanh ghi chuyển dịch ?

4. Hãy cho biết cách dùng các cổng logic để dựng các *bộ nhân*.

5. Hãy tìm hiểu cấu trúc vật lý của các cổng logic. Hãy thảo luận xem các cổng *NAND* và *NOR* có được dùng để xây dựng các mạch không.

6. Hãy giải thích *khái niệm phụ thuộc* có thể được dùng như thế nào để mô tả các mạch chuyển phức tạp.

7. Hãy mô tả việc dùng các bộ dồn tín hiệu để dựng các mạch chuyển.

8. Hãy giải thích những ưu điểm của việc dùng các cổng ngưỡng để xây dựng các mạch chuyển. Hãy minh họa điều này hằng cách dùng các cổng ngưỡng để xây dựng các bộ nửa cộng và bộ cộng đầy đủ.

9. Hãy mô tả khái niệm *mạch chuyển không rủi ro* và nêu một số nguyên lý được dùng để thiết kế các mạch đó.

10. Hãy giải thích cách dùng các bản đồ Karnaugh để cực tiểu hóa các hàm năm hoặc sáu biến.

11. Mô tả **sự phân tích hàm** của một hàm Boole n biến có nghĩa là gì ? Hãy thảo luận các thủ tục phân tích các hàm Boole thành hợp thành của các hàm Boole có số hiến ít hơn.

CHƯƠNG 10

MÔ HÌNH TÍNH TOÁN

Các máy tính có thể thực hiện được nhiều nhiệm vụ. Với một nhiệm vụ đã cho, có hai vấn đề được đặt ra. Thứ nhất là: hiện máy tính có thể thực hiện được nhiệm vụ đó không? Một khi đã biết câu trả lời cho câu hỏi này là khẳng định, ta có thể đặt tiếp câu hỏi thứ hai là: Nhiệm vụ này có thể được thực hiện như thế nào? Các mô hình tính toán được dùng sẽ giúp ta trả lời các câu hỏi trên.

Chúng ta sẽ nghiên cứu ba loại cấu trúc được dùng trong các mô hình tính toán, cụ thể đó là các văn phạm, các máy hữu hạn trạng thái và các máy Turing. Các văn phạm được dùng để tạo các từ của một ngôn ngữ và để xác định một từ có ở trong một ngôn ngữ nào đó hay không. Những ngôn ngữ hình thức được sinh ra bởi các văn phạm sẽ cung cấp những mô hình cho cả các ngôn ngữ tự nhiên, như tiếng Anh, lẫn các ngôn ngữ lập trình như Pascal, Fortran, Prolog và C. Đặc biệt, các văn phạm là cực kỳ quan trọng trong việc xây dựng cũng như trong lý thuyết các chương trình dịch. Các văn phạm mà chúng ta sẽ xét ở đây, lần đầu tiên đã được nhà ngôn ngữ Mỹ Noam Chomsky sử dụng vào những năm 1950.

Các loại máy hữu hạn trạng thái được dùng trong việc mô hình hóa. Tất cả các máy hữu hạn trạng thái đều có một tập các trạng thái, kể cả trạng thái xuất phát, một bảng chữ cái các đầu vào và một hàm chuyển có nhiệm vụ gán một trạng thái mới cho mỗi cặp gồm một trạng thái và một đầu vào. Các trạng thái của máy hữu hạn trạng thái cho nó những khả năng nhớ hạn chế. Một số máy hữu hạn trạng thái còn tạo được một ký hiệu đầu ra cho mỗi chuyển tiếp; các máy này có thể được dùng để mô hình hóa nhiều loại máy, như các máy bán hàng, máy trễ, các bộ cộng nhị phân, và các bộ nhận dạng ngôn ngữ. Chúng ta cũng sẽ nghiên cứu cả các máy hữu hạn trạng thái không có đầu ra nhưng có các trạng thái cuối cùng. Các máy này được dùng rất rộng rãi trong việc nhận dạng ngôn ngữ. Các xâu được nhận dạng là những xâu đưa trạng thái xuất phát tới trạng thái kết thúc. Các khái niệm về văn phạm và các máy hữu hạn trạng thái có quan hệ chặt chẽ với nhau. Chúng ta sẽ đưa ra những đặc trưng cho các tập hợp được chấp nhận bởi một máy hữu hạn trạng thái và chứng tỏ rằng đó chính là các tập hợp được sinh ra bởi một loại văn phạm nào đó.

Cuối cùng, chúng ta sẽ đưa vào khái niệm máy Turing và sẽ chỉ ra các máy Turing được dùng để nhận dạng các tập hợp như thế nào. Chúng ta cũng sẽ cho thấy cách mà các máy Turing được dùng để tính các hàm của lý thuyết số. Cuối cùng, chúng ta sẽ thảo luận luận đề Church – Turing phát biểu rằng mọi tính toán hiệu quả đều có thể được thực hiện bằng cách dùng máy Turing.

10.1 NGÔN NGỮ VÀ VĂN PHẠM

MỞ ĐẦU

Các từ trong tiếng Anh có thể được tổ hợp theo nhiều cách khác nhau. Văn phạm của tiếng Anh cho chúng ta biết một tổ hợp của các từ có phải là một câu đúng hay không. Ví dụ, *the frog writes neatly* là một câu đúng, vì nó được tạo bởi một danh ngữ – the frog (con ếch) – được tạo bởi quán từ the và danh từ frog và tiếp sau là động ngữ – writes neatly (viết rõ ràng) – được tạo bởi động từ writes và trạng từ neatly. Chúng ta không quan tâm tới việc đây là một câu vô nghĩa, vì chúng ta chỉ quan tâm tới **cú pháp** của câu chứ không phải tới **ngữ nghĩa** của nó. Cũng cần lưu ý rằng tổ hợp các từ *swims* (bơi) *quickly* (nhanh) *mathematics* (toán) không phải là một câu đúng vì nó không theo đúng các quy tắc của tiếng Anh.

Cú pháp của một **ngôn ngữ tự nhiên**, tức là ngôn ngữ nói, như tiếng Anh, tiếng Pháp, tiếng Đức hoặc tiếng Tây Ban Nha, đều cực kỳ phức tạp. Và thực tế dường như không thể chỉ ra hết các quy tắc cú pháp đối với một ngôn ngữ tự nhiên. Việc nghiên cứu sự dịch tự động một ngôn ngữ này sang một ngôn ngữ khác đã dẫn tới khái niệm **ngôn ngữ hình thức**, một ngôn ngữ không giống như các ngôn ngữ tự nhiên, nó có một tập hoàn toàn xác định các quy tắc cú pháp. Các quy tắc cú pháp là quan trọng không chỉ trong ngôn ngữ học, trong nghiên cứu các ngôn ngữ tự nhiên, mà cả trong sự nghiên cứu các ngôn ngữ lập trình.

Chúng ta sẽ mô tả các câu của một ngôn ngữ hình thức bằng cách dùng một văn phạm. Việc dùng văn phạm sẽ giúp ích rất nhiều khi ta xét hai lớp các vấn đề được đặt ra rất thường xuyên trong ứng dụng của các ngôn ngữ lập trình: (1) Làm thế nào xác định được một tổ hợp các từ có là một câu đúng trong một ngôn ngữ hình thức hay không? (2) Làm thế nào có thể tạo ra các câu đúng trong một ngôn ngữ hình thức?

Trước khi cho một định nghĩa có tính kỹ thuật về văn phạm, chúng ta sẽ mô tả một ví dụ về một văn phạm tạo ra một tập con của tiếng Anh. Tập con này được định nghĩa bằng cách dùng một bảng liệt kê các quy tắc cho biết một câu đúng có thể được tạo ra như thế nào. Cụ thể là:

1. Một câu được tạo bởi **danh ngữ** và tiếp sau là **động ngữ**
2. Danh ngữ được tạo bởi một **quán từ**, tiếp sau là một **tính từ** và sau đó là một **danh từ** hoặc
3. Một **danh ngữ** tạo bởi một **quán từ** và tiếp sau là một **danh từ**
4. **Động ngữ** được tạo bởi một **động từ** và tiếp theo là một **trạng từ** hoặc
5. **Động ngữ** được tạo bởi một **động từ**
6. Một quán từ là *a* hoặc

7. Một quán từ là *the*

8. Một tính từ là *large* (lớn) hoặc

9. Một tính từ là *hungry* (đói)

10. Một danh từ là *rabbit* (thỏ) hoặc

11. Một danh từ là *mathematician* (nhà toán học)

12. Một động từ là *eats* (ăn) hoặc

13. Một động từ là *hops* (nhảy)

14. Một trạng từ là *quickly* (nhanh), hoặc

15. Một trạng từ là *wildty* (như điên)

Từ những quy tắc trên chúng ta có thể lập các câu đúng bằng cách dùng một loạt các thay thế cho tới khi không còn một quy tắc nào có thể được dùng nữa. Chẳng hạn, chúng ta có thể theo dãy các thay thế sau để nhận được một câu đúng:

câu

danh ngữ động ngữ

quán từ tính từ danh từ động ngữ

quán từ tính từ danh từ động từ trạng từ

the **tính từ danh từ động từ trạng từ**

the large **danh từ động từ trạng từ**

the large rabbit **động từ trạng từ**

the large rabbit hops **trạng từ**

the large rabbit hops quickly

Cũng dễ dàng thấy rằng, một số câu đúng khác là: *a hungry mathematician eats wildly, a large mathematician hops, the rabbit eats quickly* v.v... Chúng ta cũng thấy ngay rằng *the quickly eats mathematician* không phải là một câu đúng.

VĂN PHẠM CẤU TRÚC CÂU

Trước khi cho định nghĩa hình thức của một văn phạm, chúng ta sẽ đưa vào một ít thuật ngữ

ĐỊNH NGHĨA 1. Một *từ vựng* (hay một *bộ chữ cái*) V là một tập không rỗng, hữu hạn; các phần tử của tập này được gọi là các *ký hiệu*. Một *từ* (hoặc *một câu*) trên V là một xâu các phần tử của V có chiều dài hữu hạn. *Xâu rỗng*, được ký hiệu là λ, là xâu không chứa một ký hiệu nào. Tập tất cả các từ trên V được ký hiệu là V^*. Một *ngôn ngữ* trên V là một tập con của V^*.

Chú ý rằng xâu rỗng λ là xâu không chứa một ký hiệu nào. Nó khác với tập rỗng $\emptyset$. Từ đây suy ra rằng $\{\lambda\}$ là tập chỉ chứa đúng một xâu, đó là xâu rỗng.

Các ngôn ngữ có thể được chỉ rõ bằng nhiều cách khác nhau. Một trong những cách đó là liệt kê tất cả các từ trong ngôn ngữ đó. Một cách khác là cho một số tiêu chuẩn mà các từ thuộc ngôn ngữ đó cần phải thỏa mãn. Trong tiết này chúng ta sẽ mô tả một cách quan trọng khác để chỉ rõ một ngôn ngữ, đó là cách thông qua việc dùng một văn phạm, giống như tập các quy tắc mà chúng ta đã cho ở phần Mở đầu của tiết này. Một văn phạm cung cấp một tập các ký hiệu thuộc các loại khác nhau, và một tập các quy tắc để sản sinh ra các từ. Nói một cách chính xác hơn, một văn phạm có một **từ vựng** V - đó là tập hợp các ký hiệu được dùng để dẫn xuất ra các thành phần của một ngôn ngữ. Một số phần tử của từ vựng không thể được thay thế bởi các ký hiệu khác, các phần tử này được gọi là ký hiệu **kết thúc**. Trong khi đó, các phần tử khác của từ vựng có thể được thay thế bởi các ký hiệu khác, chúng được gọi là ký hiệu **không kết thúc**. Tập các từ kết thúc và không kết thúc thường được ký hiệu tương ứng là T và N. Trong ví dụ được cho ở cuối phần Mở đầu của tiết này, tập các từ kết thúc là {*a, the, rabbit, mathematician, hops, eats, quickly, wildly*} và tập các từ không kết thúc là {**câu, danh ngữ, động ngữ, tính từ, quán từ, danh từ, động từ, trạng từ**}. Trong từ vựng có một phần tử đặc biệt được gọi là **ký hiệu xuất phát** và được ký hiệu là S. Đây là phần tử của từ vựng mà ta luôn bắt đầu từ nó. Trong ví dụ cho ở phần Mở đầu, ký hiệu xuất phát là **câu**. Các quy tắc chỉ rõ khi nào ta có thể thay thế một xâu trong V^* – tập tất cả các xâu gồm các phần tử trong từ vựng – bởi một xâu khác được gọi là **sản xuất** của văn phạm đó. Sản xuất chỉ rõ rằng w_0 có thể được thay thế bởi w_1 được ký hiệu là $w_0 \rightarrow w_1$. Các sản xuất trong văn phạm cho ở phần mở đầu của tiết này đã được liệt kê ra. Sản xuất đầu tiên được viết theo ký hiệu nói trên là **câu** $\rightarrow$ **danh ngữ động ngữ**. Chúng ta tổng kết những điều nói trên trong định nghĩa sau.

ĐỊNH NGHĨA 2. Một *văn phạm cấu trúc câu* $G = (V, T, S, P)$ gồm một từ vựng V, một tập con T của V gồm các phần tử kết thúc, một ký hiệu xuất phát S và tập các sản xuất P. Tập $V - T$ được ký hiệu là N. Các phần tử thuộc N được gọi là các ký hiệu *không kết thúc*. Mỗi sản xuất trong P cần phải chứa ít nhất một ký hiệu không kết thúc ở vế trái của nó.

Ví dụ 1. Cho $G = \{V, T, S, P\}$, trong đó $V = \{a, b, A, B, S\}$, $T = \{a, b\}$, S là ký hiệu xuất phát và $P = \{S \rightarrow ABa, A \rightarrow BB, B \rightarrow ab, AB \rightarrow b\}$. G là ví dụ về một văn phạm cấu trúc câu.

■

Chúng ta sẽ quan tâm tới các từ có thể được sinh bởi các sản xuất của một văn phạm cấu trúc câu

ĐỊNH NGHĨA 3. Cho $G = \{V, T, S, P\}$ là một văn phạm cấu trúc câu. Cho $w_0 = lz_0r$ (tức là phép ghép của l, z_0 và r) và $w_1 = lz_1r$ là các xâu trên V. Nếu $z_0 \rightarrow z_1$ là một sản xuất của G, thì ta nói rằng w_1 được dẫn xuất trực tiếp từ w_0 và viết $w_0 \Rightarrow w_1$. Nếu $w_0, w_1, \ldots w_n$ với $n \geqslant 0$ là các xâu trên V sao cho $w_0 \Rightarrow w_1$, $w_1 \Rightarrow w_2, \ldots, w_{n-1} \Rightarrow w_n$, thì ta nói rằng w_n được dẫn xuất từ w_0 và viết $w_0 \Rightarrow w_n$. Dãy các bước được dùng để nhận được w_n từ w_0 được gọi là một *dẫn xuất*.

Ví dụ 2. Xâu $Aaba$ được dẫn xuất trực tiếp từ ABa trong văn phạm cho trong Ví dụ 1, vì $B \rightarrow ab$ là một sản xuất trong văn phạm đó. Xâu $abababa$ được dẫn xuất từ ABa, vì $ABa \Rightarrow Aaba \Rightarrow BBaba \Rightarrow Bababa \Rightarrow abababa$ bằng cách dùng lần lượt $B \rightarrow ab$, $A \rightarrow BB$, $B \rightarrow ab$ và $B \rightarrow ab$.

■

ĐỊNH NGHĨA 4. Cho $G = \{V, T, S, P\}$ là một văn phạm cấu trúc câu. *Ngôn ngữ được sinh bởi văn phạm G* (hay *ngôn ngữ của G*), được ký hiệu là $L(G)$, là tập của tất cả các xâu ký hiệu kết thúc được dẫn xuất từ ký hiệu xuất phát S. Nói cách khác,

$$L(G) = \{w \in T^* \mid S \stackrel{*}{\Rightarrow} w\}$$

Trong hai ví dụ sau chúng ta sẽ tìm ngôn ngữ được sinh bởi một văn phạm cấu trúc câu.

Ví dụ 3. Cho G là văn phạm với từ vựng $V = \{S, A, a, b\}$, tập các ký hiệu kết thúc $T = \{a, b\}$, ký hiệu xuất phát S và các sản xuất $P = \{S \to aA, S \to b, A \to aa\}$. Xác định ngôn ngữ $L(G)$ của văn phạm đó.

Giải: Từ ký hiệu xuất phát S ta có thể dẫn ra aA bằng cách dùng sản xuất $S \to aA$. Ta cũng có thể dùng sản xuất $S \to b$ để dẫn ra b. Từ aA, sản xuất $A \to aa$ có thể được dùng để dẫn xuất ra aaa. Ngoài ra, không thể dẫn thêm một từ nào nữa. Do đó $L(G) = \{b, aaa\}$

■

Ví dụ 4. Cho G là văn phạm với từ vựng $V = \{S, 0, 1\}$, tập các ký hiệu kết thúc $T = \{0, 1\}$, ký hiệu xuất phát S và các dẫn xuất $P = \{S \to 11S, S \to 0\}$. Xác định ngôn ngữ $L(G)$ của văn phạm đó.

Giải: Từ S có thể dẫn xuất ra 0 bằng cách dùng sản xuất $S \to 0$; hoặc dẫn xuất ra $11S$ bằng cách dùng sản xuất $S \to 11S$. Từ $11S$ ta có thể dẫn xuất ra 110 hoặc $1111S$. Từ $1111S$ lại có thể dẫn xuất ra 11110, hoặc $111111S$. Ở mỗi giai đoạn bất kỳ của một dẫn xuất, ta có thể hoặc thêm hai số 1 ở cuối xâu hoặc kết thúc dẫn xuất bằng cách thêm một số 0 vào cuối xâu đó. Chúng ta phỏng đoán rằng $L(G) = \{0, 110, 11110, 1111110, \ldots\}$ – đó là tập tất cả các xâu bắt đầu bằng một số chẵn các số 1 và kết thúc bằng một số 0. Điều này có thể chứng minh bằng phương pháp quy nạp với lập luận rằng sau khi n sản xuất đã được sử dụng, các xâu duy nhất chứa các ký hiệu kết thúc được sinh ra bởi các sản xuất đó gồm $n-1$ hoặc ít hơn các cặp 11 tiếp theo bởi một số 0. (Điều này dành cho các bạn như một bài tập).

■

Người ta cũng thường gặp bài toán xây dựng một văn phạm sinh ra một ngôn ngữ đã cho. Ba ví dụ sau đây mô tả các bài toán thuộc loại đó.

Ví dụ 5. Cho văn phạm sinh ra tập $\{0^n1^n \mid n = 0, 1, 2,...\}$

Giải: Có thể dùng hai sản xuất để sinh ra tất cả các xâu gồm một xâu các số 0 và tiếp sau là một số giống như thế các số 1, kể cả xâu rỗng. Sản xuất thứ nhất xây dựng các xâu dài tuần tự trong ngôn ngữ đó bằng cách thêm một số 0 vào đầu của xâu và một số 1 vào cuối của xâu đó. Sản xuất thứ hai là thay S bằng một xâu rỗng. Lời giải là văn phạm $G = \{V, T, S, P\}$ trong đó $V = \{0, 1, S\}$, $T = \{0, 1\}$, S là ký hiệu xuất phát, và các sản xuất là:

$$S \to 0S1$$

$$S \to \lambda$$

Việc chứng minh văn phạm này sinh ra một tập đúng như đầu bài yêu cầu xin dành lại cho bạn đọc như một bài tập.

■

Ví dụ trên liên quan đến tập các xâu tạo bởi các số 0 tiếp sau bởi các số 1 với số lượng các số 0 và các số 1 là như nhau. Ví dụ tiếp sau cũng xét tập các xâu gồm các số 0 tiếp sau bởi các số 1 nhưng số các số 0 và các số 1 có thể khác nhau.

Ví dụ 6. Tìm văn phạm cấu trúc câu sinh ra tập $\{0^m1^n \mid m$ và n là các số nguyên không âm$\}$.

Giải: Chúng ta sẽ cho hai văn phạm G_1 và G_2 sinh ra tập này. Điều này sẽ minh họa cho điều là hai văn phạm có thể sinh ra cùng một ngôn ngữ.

Văn phạm G_1 có từ vựng $V = \{S, 0, 1\}$, các ký hiệu kết thúc $T = \{0, 1\}$ và các sản xuất $S \to 0S$, $S \to S1$ và $S \to \lambda$. Việc chứng minh chi tiết văn phạm này sinh ra tập đúng như yêu cầu xin dành lại cho độc giả như một bài tập.

■

Đôi khi một tập có thể mô tả dễ dàng lại được sinh ra bởi một văn phạm phức tạp. Điều này được minh họa trong ví dụ sau:

Ví dụ 7. Một văn phạm sinh tập $\{0^n1^n2^n \mid n = 0, 1, 2, 3,...\}$ là $G =$

$\{V, T, S, P\}$ với $V = \{0, 1, 2, S, A, B\}$, $T = \{0, 1, 2\}$, ký hiệu xuất phát là S và các sản xuất $S \to 0SAB$, $S \to \lambda$, $BA \to AB$, $0A \to 01$, $1A \to 11$, $1B \to 12$, $2B \to 22$. Chúng tôi xin dành cho độc giả việc chứng minh khẳng định trên như một bài tập. Văn phạm được cho ở trên là loại văn phạm đơn giản nhất sinh ra tập đó. Đơn giản nhất ở đây được hiểu theo nghĩa sẽ được làm sáng tỏ ở cuối tiết này. Bạn đọc có thể ngạc nhiên không hiểu lấy đâu ra văn phạm này, vì dường như rất khó mò ra văn phạm đó. Bạn có thể an lòng biết rằng văn phạm này có thể được xây dựng một cách bài bản bằng cách dùng những kỹ thuật từ lý thuyết tính toán vượt ra ngoài phạm vi của cuốn sách này.

■

CÁC LOẠI VĂN PHẠM CẤU TRÚC CÂU

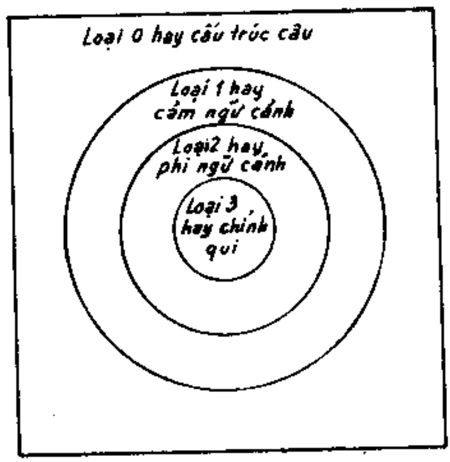
Các văn phạm cấu trúc câu được phân loại theo các loại sản xuất cho phép. Chúng ta sẽ mô tả sơ đồ phân loại do Noam Chomsky đưa ra. Trong tiết 10.4 ta sẽ thấy rằng các loại ngôn ngữ khác nhau được định nghĩa trong sơ đồ này tương ứng với các lớp ngôn ngữ có thể chấp nhận được khi dùng các mô hình máy tính khác nhau.

Một văn phạm **loại 0** không có các hạn chế đối với các sản xuất của chúng. Một văn phạm **loại 1** có thể chỉ có các sản xuất dạng $w_1 \to w_2$ trong đó chiều dài của w_2 lớn hơn hoặc bằng chiều dài của w_1, hoặc có dạng $w_1 \to \lambda$. Một văn phạm **loại 2** có thể chỉ có các sản xuất dạng $w_1 \to w_2$ với w_1 là ký hiệu đơn và không phải là ký hiệu kết thúc. Một văn phạm **loại 3** có thể chỉ có các sản xuất dạng $w_1 \to w_2$ với $w_1 = A$ và $w_2 = aB$ hoặc $w_2 = a$ trong đó A và B là các ký hiệu không kết thúc và a là ký hiệu kết thúc, hoặc với $w_1 = S$ và $w_2 = \lambda$.

Từ các định nghĩa trên, ta thấy rằng mọi văn phạm loại 3 đều là loại 2, mọi văn phạm loại 2 đều là loại 1 và mọi văn phạm loại 1 đều là loại 0. Các văn phạm loại 2 được gọi là **văn phạm phi ngữ cảnh**, vì một ký hiệu không kết thúc ở vế trái của sản xuất có thể được thay bằng một xâu bất cứ khi nào nó xuất hiện, bất chấp còn có gì ở xâu đó. Ngôn ngữ được sinh bởi một văn phạm phi ngữ cảnh được gọi là **ngôn ngữ phi**

ngữ cảnh. Khi có một sản xuất dạng $lw_1r \rightarrow lw_2r$ (chứ không phải dạng $w_1 \rightarrow w_2$) văn phạm được gọi là loại 1 hoặc **cảm ngữ cảnh** vì w_1 có thể được thay bằng w_2 chỉ khi nó được bao quanh bởi các xâu l và r. Các văn phạm loại 3 còn được gọi là các **văn bản chính quy**. Ngôn ngữ được sinh ra bởi một văn phạm chính quy được gọi là **ngôn ngữ chính quy**. Tiết 10.4 sẽ xem xét mối quan hệ giữa các ngôn ngữ chính quy và các máy hữu hạn trạng thái. Giản đồ Venn trong Hình 1 cho mối quan hệ giữa các văn phạm khác nhau.

Ví dụ 8. Từ Ví dụ 6 ta biết rằng $\{0^m1^n \mid m, n = 0, 1, 2,...\}$ là một ngôn ngữ chính quy, vì nó được sinh ra bởi một văn phạm chính quy, đó là văn phạm G_2 trong Ví dụ 6. ■



Hình 1. Các loại văn phạm

Ví dụ 9. Từ Ví dụ 5 suy ra rằng $\{0^n1^n \mid n = 0, 1, 2,...\}$ là một ngôn ngữ phi ngữ cảnh, vì các sản xuất trong văn phạm này là $S \rightarrow 0S1$ và $S \rightarrow \lambda$. Tuy nhiên, nó không phải là ngôn ngữ chính quy. Điều này sẽ được chứng tỏ trong tiết 10.4. ■

Tập $\{0^n1^n2^n \mid n = 0, 1, 2,...\}$ là một ngôn ngữ cảm ngữ cảnh, vì nó có thể được sinh bởi một văn phạm loại 1 như ví dụ 7 cho thấy, nhưng không bởi bất kỳ văn phạm loại 2 nào. (Điều này được chứng minh trong Bài tập 28 thuộc phần các bài tập bổ sung ở cuối chương này).

Bảng 1 tổng kết các thuật ngữ đã được dùng để phân loại các văn phạm cấu trúc câu.

CÂY DẪN XUẤT

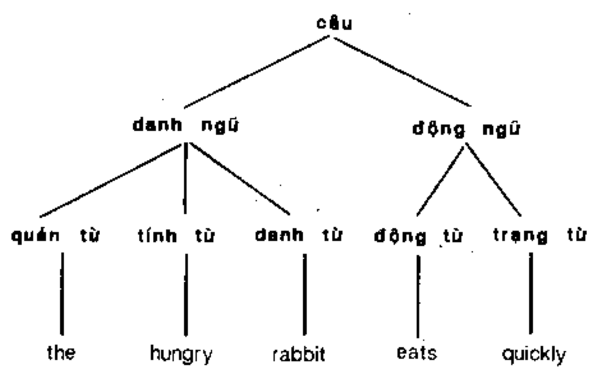
Một dẫn xuất trong ngôn ngữ được sinh bởi một văn phạm phi ngữ cảnh có thể được biểu diễn bằng đồ thị nhờ dùng một cây được gọi là **cây dẫn xuất** hoặc cây **phân tích cú pháp**. Gốc của cây này biểu diễn ký hiệu xuất phát. Các đỉnh trong của cây biểu diễn các ký hiệu không kết thúc xuất hiện trong dẫn xuất đó. Các lá của cây biểu diễn các ký hiệu kết thúc xuất hiện ở cây đó. Nếu sản xuất $A \to w$ xuất hiện trong dẫn xuất đã cho với w là một từ, thì đỉnh biểu diễn A có các đỉnh con biểu diễn mỗi ký hiệu trong w theo thứ tự từ trái sang phải.

BẢNG 1. Các loại văn phạm

Loại	Những hạn chế đối với các sản xuất $w_1 \to w_2$
0	Không có hạn chế nào
1	$l(w_1) \leq l(w_2)$ hoặc $w_2 = \lambda$
2	$w_1 = A$ với A là ký hiệu không kết thúc
3	$w_1 = A$ và $w_2 = aB$ hay $w_2 = a$ với $A \in N$, $B \in N$, và $a \in T$, hay $S \to \lambda$

Ví dụ 11. Dựng cây dẫn xuất cho dẫn xuất *the hungry rabbit eats quickly* (con thỏ đói ăn nhanh) được cho trong phần mở đầu của tiết này.

Giải: Cây dẫn xuất cần tìm được cho trên Hình 2 ■



Hình 2. Cây dẫn xuất.

Bài toán xác định một xâu có ở trong ngôn ngữ được sinh bởi một văn phạm phi ngữ cảnh hay không là bài toán thường xuất hiện trong nhiều ứng dụng, chẳng hạn như trong việc xây dựng các chương trình dịch. Có hai cách tiếp cận vấn đề này được chỉ ra trong ví dụ sau.

Ví dụ 12. Hãy xác định xem từ *cbab* có thuộc ngôn ngữ được sinh ra bởi văn phạm $G = (V, T, S, P)$ trong đó $V = \{a, b, c, A, B, C, S\}$, $T = \{a, b, c\}$, S là ký hiệu xuất phát và các sản xuất

$S \to AB$

$A \to Ca$

$B \to Ba$

$B \to Cb$

$B \to b$

$C \to cb$

$C \to b$

Giải: Một cách tiếp cận bài toán này là bắt đầu với S và tìm cách dẫn ra *cbab* bằng cách dùng một loạt các sản xuất. Vì chỉ có một sản xuất với S ở vế trái, ta cần phải bắt đầu với $S \Rightarrow AB$. Sau đó ta dùng sản xuất duy nhất có A ở vế trái, cụ thể là $A \to Ca$ để nhận được $S \Rightarrow AB \Rightarrow CaB$. Vì *cbab* bắt đầu với các ký hiệu *cb*, nên ta dùng sản xuất $C \to cb$. Kết quả cho $S \Rightarrow AB \Rightarrow CaB \Rightarrow cbaB$. Cuối cùng, bằng cách dùng sản xuất $B \to b$, ta nhận được $S \Rightarrow AB \Rightarrow CaB \Rightarrow cbaB \Rightarrow cbab$. Cách tiếp cận vừa được dùng ở trên được gọi là **sự phân giải từ trên xuống dưới**, vì nó bắt đầu với ký hiệu xuất phát và phát triển bằng cách áp dụng tuần tự các sản xuất.

Phương pháp thứ hai tiếp cận bài toán này được gọi là **sự phân giải từ dưới lên**. Trong cách tiếp cận này, ta tiến hành theo cách giật lùi. Vì *cbab* là xâu cần được dẫn xuất ra, nên ta có thể dùng sản xuất $C \to cb$, sao cho $Cab \to cbab$. Sau đó ta có thể dùng sản xuất $A \to Ca$, sao cho $Ab \Rightarrow Cab \Rightarrow cbab$. Bằng cách dùng sản xuất $B \to b$, ta có $AB \Rightarrow Ab \Rightarrow Cab \Rightarrow cbab$. Cuối cùng, dùng $S \to AB$, ta có một dẫn xuất đầy đủ đối với *cbab*, đó là $S \Rightarrow AB \Rightarrow Ab \Rightarrow Cab \Rightarrow cbab$.

DẠNG BACKUS - NAUR

Có một cách ký hiệu khác đôi khi được dùng để chỉ một văn phạm loại 2, được gọi là *dạng Backus – Naur* theo tên của John Backus – người phát minh ra nó – và Peter Naur người đã sửa sang nó để dùng đặc tả ngôn ngữ lập trình ALGOL. Các sản xuất trong văn phạm loại 2 có một ký hiệu đơn không kết thúc ở vế trái. Thay vì liệt kê tất cả các sản xuất tách biệt nhau, ta có thể gộp tất cả các sản xuất có cùng một ký hiệu không kết thúc ở vế trái thành một mệnh đề. Thay cho ký hiệu → trong một sản xuất, ta dùng ký hiệu :: =. Ta sẽ đưa tất cả các ký hiệu không kết thúc vào trong các dấu ngoặc < >, và liệt kê tất cả vế phải của các sản xuất vào cùng một mệnh đề, ngăn cách giữa chúng là một vạch đứng. Ví dụ, các sản xuất $A \rightarrow Aa$, $A \rightarrow a$ và $A \rightarrow AB$ được gộp thành $<A> ::= <A>a \mid a \mid <A><B>$.

Ví dụ 13. Viết dạng Backus – Naur của văn phạm cho tập con của tiếng Anh được mô tả ở phần mở đầu của tiết này.

Giải: Dạng Backus – Naur của văn phạm đó là:

<câu> :: = *<danh ngữ><động ngữ>*

<danh ngữ> :: = *<quán từ><tính từ><danh từ>* | *<quán từ><danh từ>*

<động ngữ> :: = *<động từ><trạng từ>* | *<động từ>*

<quán từ> :: = *a* | *the*

<tính từ> :: = *large* | *hungry*

<danh từ> :: = *rabbit* | *mathematician*

<động từ> :: = *eats* | *hops*

<trạng từ> :: = *quickly* | *wildly*

■

Ví dụ 14. Viết dạng Backus – Naur cho sản xuất các số nguyên có dấu trong biểu diễn thập phân. (Một **số nguyên có dấu** là một số nguyên không âm được đặt ở trước một dấu cộng hoặc một dấu trừ).

Giải: Dạng Backus – Naur cho văn phạm sản xuất các số nguyên có dấu là:

<số nguyên có dấu> :: = *<dấu><số nguyên>*

<dấu> :: = + | −

<số nguyên> :: = *<chữ số>* | *<chữ số><số nguyên>*

<chữ số> :: = 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9

■

BÀI TẬP

Các bài tập từ 1 – 3 liên quan với văn phạm có ký hiệu xuất phát là **câu**, tập các ký hiệu kết thúc T = {*the, sleepy, happy, tortoise, hare, passes, runs, quickly, slowly*}, tập các ký hiệu không kết thúc N = {**danh ngữ, động ngữ ngoại, động ngữ nội, quán từ, tính từ, danh từ, động từ, trạng từ**} và các sản xuất

câu → danh ngữ động ngữ ngoại danh ngữ

câu → danh ngữ động ngữ nội

danh ngữ → quán từ tính từ danh từ

danh ngữ → quán từ danh từ

động ngữ ngoại → ngoại động từ

động ngữ nội → nội động từ trạng từ

động ngữ nội → nội động từ

quán từ → *the*

tính từ → *sleepy*

tính từ → *happy*

danh từ → *tortoise*

danh từ → *hare*

ngoại động từ → *passes*

nội động từ → *runs*

trạng từ → *quickly*

trạng từ → *slowly*

1. Dùng tập các sản xuất chứng minh rằng các câu cho dưới đây đều là đúng

   a) *the happy hare runs*

   b) *the sleepy tortoise runs quickly*

   c) *the tortoise passes the hare*

   d) *the sleepy hare passes the happy tortoise*

2. Tìm 5 câu đúng khác với những câu cho trong Bài tập 1.

3. Chứng minh rằng câu *the hare runs the sleepy tortoise* không phải là một câu đúng.

*4. Cho $V = \{S, A, B, a, b\}$ và $T = \{a, b\}$. Tìm ngôn ngữ được sinh bởi văn phạm $\{V, T, S, P\}$ với tập P các sản xuất bao gồm:

   a) $S \to AB$, $A \to ab$, $B = bb$

   b) $S \to AB$, $S \to aA$, $A \to a$, $B \to ba$

   c) $S \to AB$; $S \to AA$, $A \to aB$, $A \to ab$, $B \to b$

   d) $S \to AA$, $S \to B$, $A \to aaA$, $A \to aa$, $B \to bB$, $B \to b$

   e) $S \to AB$, $A \to aAb$, $B \to bBa$, $A \to \lambda$, $B \to \lambda$

5. Dựng một dẫn xuất của $0^3 1^3$ bằng cách dùng văn phạm cho trong Ví dụ 5.

6. Chứng minh rằng văn phạm cho trong Ví dụ 5 sinh ra tập $\{0^n 1^n \mid n = 0, 1, 2, \ldots\}$

7. a) Dựng một dẫn xuất của $0^2 1^4$ bằng cách dùng văn phạm G_1 cho trong Ví dụ 6.

   b) Dựng một dẫn xuất của $0^2 1^4$ bằng cách dùng văn phạm G_2 cho trong Ví dụ 6.

8. a) Chứng tỏ rằng văn phạm G_1 cho trong Ví dụ 6 sinh ra tập $\{0^m1^n \mid m,n = 0, 1, 2,...\}$

b) Chứng minh rằng văn phạm G_2 cho trong Ví dụ 6 cũng sinh ra tập trong câu (a).

9. Dựng một dẫn xuất của $0^21^22^2$ trong văn phạm được cho trong Ví dụ 7.

***10.** Chứng minh rằng văn phạm cho trong Ví dụ 7 sinh ra tập $\{0^n1^n2^n \mid n = 0, 1, 2,...\}$.

***11.** Tìm văn phạm cấu trúc câu cho từng ngôn ngữ sau:

a) Tập tất cả các xâu nhị phân chứa một số chẵn các số 0 và không chứa số 1 nào.

b) Tập tất cả các xâu nhị phân tạo bởi một số 1 và tiếp sau là một số lẻ các số 0.

c) Tập tất cả các xâu nhị phân chứa một số chẵn các số 0 và một số chẵn các số 1.

d) Tập tất cả các xâu chứa 10 hoặc nhiều hơn các số 0 và không chứa số 1 nào.

e) Tập tất cả các xâu chứa số các số 0 nhiều hơn số các số 1

f) Tập tất cả các xâu chứa số các số 0 và số các số 1 bằng nhau.

g) Tập tất cả các xâu chứa số các số 0 và số các số 1 không bằng nhau.

12. Xây dựng các văn phạm cấu trúc câu sinh ra các tập sau:

a) $\{01^{2n} \mid n \geq 0\}$

b) $\{0^n1^{2n} \mid n \geq 0\}$

c) $\{0^n1^m0^n \mid m \geq 0,\ n \geq 0\}$

13. Cho $V = \{S, A, B, a, b\}$ và $T = \{a, b\}$. Hãy xác định xem $G = \{V, T, S, P\}$ có phải là văn phạm loại 0 nhưng không phải là văn phạm loại 1, là văn phạm loại 1 nhưng không phải là văn phạm loại 2 hay là văn phạm loại 2 nhưng không là văn phạm loại 3 hay không, nếu tập P của các sản xuất là:

a) $S \to aAB,\ A \to Bb,\ B \to \lambda$

b) $S \to aA,\ A \to a,\ A \to b$

c) $S \to ABa,\ AB \to a$

d) $S \to ABA,\ A \to aB,\ B \to ab$

e) $S \to bA,\ A \to B,\ B \to a$

f) $S \to aA,\ aA \to B,\ B \to aA,\ A \to b$

g) $S \to bA,\ A \to b,\ S \to \lambda$

h) $S \to AB,\ B \to aAb,\ aAb \to b$

i) $S \to aA,\ A \to bB,\ B \to b,\ B \to \lambda$

j) $S \to A,\ A \to B,\ B \to \lambda$

14. Một xâu được gọi là **thuận nghịch độc** nếu xâu đó đọc xuôi và ngược đều như nhau, nghĩa là xâu w với $w = w^R$, trong đó w^R là xâu lộn ngược của w. Hãy tìm một văn phạm phi ngữ cảnh sinh ra tập các xâu thuận nghịch độc trên bộ chữ cái $\{0, 1\}$.

*15. Cho G_1 và G_2 là hai văn phạm phi ngữ cảnh sinh ra hai ngôn ngữ $L(G_1)$ và $L(G_2)$ tương ứng. Chứng minh rằng tồn tại một văn phạm phi ngữ cảnh sinh ra các tập sau:

a) $L(G_1) \cup L(G_2)$

b) $L(G_1)\ L(G_2)$

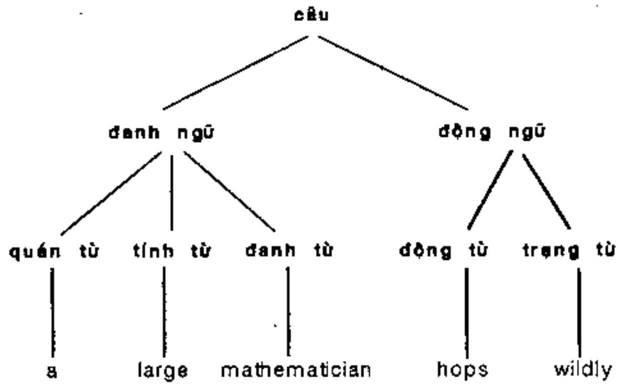
c) $L(G_1)^*$

16. Tìm các xâu được dựng bằng cách dùng các cây dẫn xuất ở trang sau.

17. Dựng các cây dẫn xuất cho các câu trong Ví dụ 1.

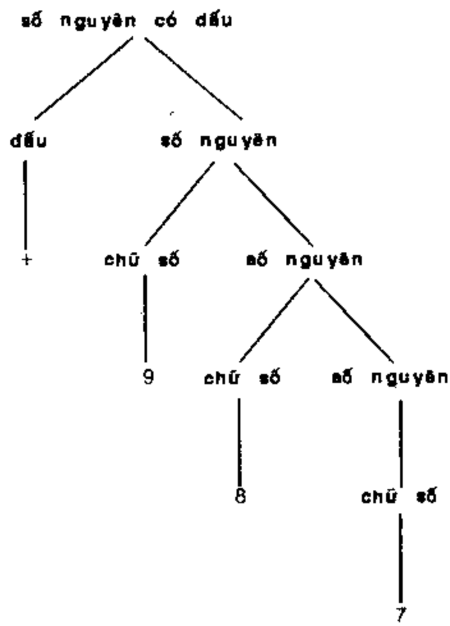
18. Cho G là văn phạm với $V = \{a, b, c, S\}$, $T = \{a, b, c\}$, ký hiệu xuất phát S và các sản xuất $S \to abS$, $S \to bcS$, $S \to bbS$, $S \to a$, $S \to cb$. Dựng các cây dẫn xuất cho

a) $bcbba$



b) *bbbcbba*

c) *bcabbbbbcb*



***19.** Dùng phép phân tích cú pháp từ trên xuống để xác định xem các xâu sau có thuộc ngôn ngữ được sinh bởi văn phạm cho trong Ví dụ 12 không

a) *baba*

b) *abab*

c) *cbaba*

d) *bbbcba*

***20.** Dùng phép tính cú pháp từ dưới lên để xác định các xâu cho trong Bài tập 19 có thuộc ngôn ngữ được sinh bởi văn phạm cho trong Ví dụ 12 không.

21. Dựng cây dẫn xuất cho -109 bằng cách dùng văn phạm cho trong Ví dụ 14.

22. a) Xác định các sản xuất trong một văn phạm, nếu dạng Backus - Naur đối với các sản xuất đó là

$$\langle expression\rangle ::= (\langle expression\rangle) \mid \langle expression\rangle + \langle expression\rangle \mid \langle expression\rangle * \langle expression\rangle \mid \langle variable\rangle$$

<*variable*> :: = $x \mid y$

b) Tìm cây dẫn xuất cho $(x * y) + y$ trong văn phạm đó.

23. a) Xây dựng một văn phạm cấu trúc câu sinh ra tất cả các số thập phân có dấu, gồm một dấu hoặc + hoặc −; một phần nguyên không âm và phần thập phân hoặc là một xâu rỗng hoặc là một dấu phảy thấp phân tiếp theo là một số nguyên trong đó các số không ban đầu trong số nguyên đó là được phép.

b) Viết dạng Backus – Naur của văn phạm đó

c) Xây dựng cây dẫn xuất cho −31,4 trong văn phạm đó.

24. a) Xây dựng một văn phạm cấu trúc câu cho tập tất cả các phân số có dạng a/b trong đó a là một số nguyên có dấu trong ký hiệu thập phân và b là một số nguyên dương.

b) Xác định dạng Backus – Naur cho văn phạm đó.

c) Xây dựng cây dẫn xuất của +311/17 trong văn phạm đó.

25. Cho G là một văn phạm và R là một quan hệ chứa cặp sắp thứ tự (w_0, w_1) nếu và chỉ nếu w_1 được dẫn xuất trực tiếp từ w_0 trong G. Xác định bao đóng phản xạ, bắc cầu của R.

10.2. CÁC MÁY HỮU HẠN TRẠNG THÁI CÓ ĐẦU RA

MỞ ĐẦU

Nhiều loại máy, kể cả các linh kiện trong các máy tính, đều có thể được mô hình hóa nhờ dùng một cấu trúc được gọi là các máy hữu hạn trạng thái. Một số loại máy hữu hạn trạng thái được dùng rất rộng rãi trong các mô hình. Tất cả các phiên bản này của các máy hữu hạn trạng thái đều bao gồm một tập hữu hạn các trạng thái, với một trạng thái xuất phát đã được chỉ rõ, một bộ chữ cái đầu vào và một hàm chuyển gán cho mỗi cặp gồm một trạng thái và một đầu vào một trạng thái mới. Trong tiết này ta sẽ nghiên cứu các máy hữu hạn trạng thái có tạo ra đầu ra. Ta sẽ cho thấy các máy hữu hạn trạng thái có thể được dùng để mô hình hóa một máy bán hàng, một máy làm trễ đầu vào, một máy cộng các số nguyên và máy xác định một xâu nhị phân có chứa một cấu hình đặc biệt nào đó hay không.

Trước khi cho một định nghĩa hình thức, ta sẽ xem một máy bán hàng được mô hình như thế nào.

Một máy bán hàng chấp nhận các đồng 5 xu, 10 xu, và 25 xu. Khi ta thả vào máy tổng cộng 30 xu hoặc nhiều hơn, máy sẽ lập tức thối lại số tiền vượt quá 30 xu. Khi 30 xu đã được nằm trong máy và số tiền dư đã được thối lại, người mua có thể ấn núm màu da cam và nhận được cốc nước cam, hoặc ấn núm màu đỏ và nhận được cốc nước táo. Chúng ta có thể mô tả sự hoạt động của máy bằng cách chỉ rõ các trạng thái của nó cùng với sự thay đổi các trạng thái đó khi máy tiếp nhận một đầu vào và đầu ra được tạo ra đối với mỗi tổ hợp của đầu vào và trạng thái hiện thời.

Máy có thể ở trạng thái bất kỳ trong số bảy trạng thái khác nhau s_i với $i = 0, 1, 2, \ldots 6$, ở đó máy đã nhận được $5i$ xu. Các đầu vào khả dĩ là 5 xu, 10 xu, 25 xu, núm màu da cam (O) và núm màu đỏ (R). Đầu ra có thể là không có gì (n), 5 xu, 10 xu, 15 xu, 20 xu, 25 xu, cốc nước cam và cốc nước táo.

Ta minh họa sự hoạt động của mô hình máy bán hàng này bằng ví dụ sau. Giả sử một sinh viên thả vào máy một đồng 10 xu, rồi tiếp theo một đồng 25 xu. Sau khi nhận được 5 xu thối lại, anh ta ấn nút màu da cam để nhận được cốc nước cam.

Máy bắt đầu ở trạng thái s_0. Đầu vào đầu tiên là 10 xu làm cho máy chuyển sang trạng thái s_2 và không cho đầu ra nào. Đầu vào thứ hai là 25 xu, nó làm thay đổi trạng thái từ s_2 đến s_6 và cho đầu ra là 5 xu thối lại. Đầu vào tiếp theo là nút màu da cam, nó làm thay đổi trạng thái từ

s_6 về s_0 (vì máy trở về trạng thái xuất phát) và cho đầu ra là một cốc nước cam.

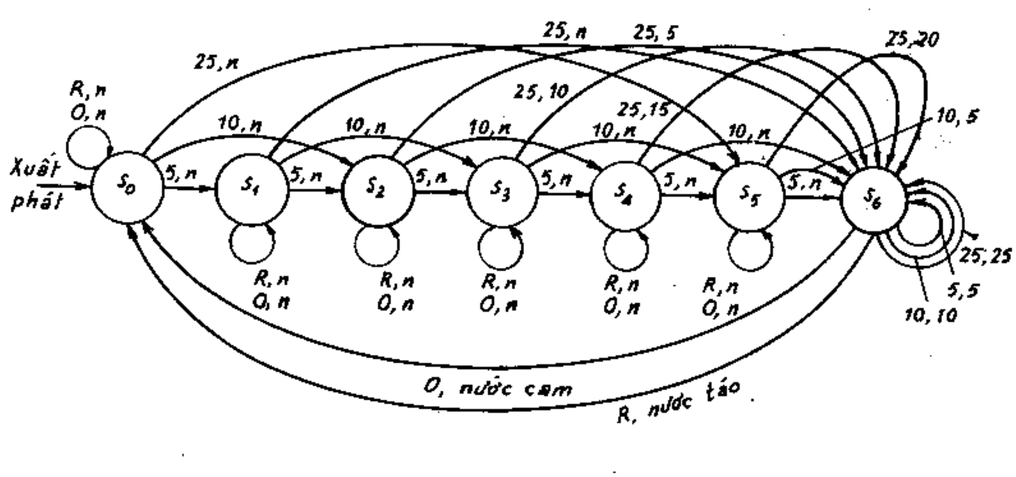
Chúng ta có thể biểu diễn tất cả những thay đổi trạng thái nói trên và đầu ra của máy đó trong một bảng. Để làm điều đó, ta cần phải chỉ rõ trạng thái tiếp theo và đầu ra nhận được đối với mỗi một tổ hợp của trạng thái và đầu vào. Bảng 1 cho thấy các dịch chuyển trạng thái và các đầu ra đối với mỗi cặp gồm một trạng thái và một đầu vào.

BẢNG 1. Bảng trạng thái của một máy bán hàng

Trạng thái	Trạng thái tiếp theo					Đầu ra				
	Đầu vào					Đầu vào				
	5	10	25	O	R	5	10	25	O	R
s_0	s_1	s_2	s_5	s_0	s_0	n	n	n	n	n
s_1	s_2	s_3	s_6	s_1	s_1	n	n	n	n	n
s_2	s_3	s_4	s_6	s_2	s_2	n	n	5	n	n
s_3	s_4	s_5	s_6	s_3	s_3	n	n	10	n	n
s_4	s_5	s_6	s_6	s_4	s_4	n	n	15	n	n
s_5	s_6	s_6	s_6	s_5	s_5	n	5	20	n	n
s_6	s_6	s_6	s_6	s_0	s_0	5	10	25	OJ*	AJ*

*OJ – cốc nước cam

**AJ – cốc nước táo.



Hình 1. Máy bán hàng.

Một cách khác để cho thấy hoạt động của máy là dùng các đồ thị có hướng với các cạnh được đánh dấu, trong đó mỗi trạng thái được biểu diễn bởi một vòng tròn, các cạnh biểu diễn sự chuyển dịch trạng thái và được đánh dấu bằng đầu vào và đầu ra ứng với chuyển dịch đó. Hình 1 cho đồ thị có hướng biểu diễn máy bán hàng.

MÁY HỮU HẠN TRẠNG THÁI CÓ ĐẦU RA

Bây giờ ta sẽ cho định nghĩa hình thức của một máy hữu hạn trạng thái có đầu ra.

ĐỊNH NGHĨA 1. *Một máy hữu hạn trạng thái* $M = (S, I, O, f, g, s_0)$ gồm một tập hữu hạn S các trạng thái, một *bộ chữ cái hữu hạn đầu vào* I, *một bộ chữ cái hữu hạn đầu ra* O, một *hàm chuyển* f gán cho mỗi cặp gồm một trạng thái và một đầu vào một trạng thái mới, một *hàm đầu ra* g gán cho mỗi cặp gồm một trạng thái và một đầu vào một đầu ra, và một trạng thái ban đầu s_0.

Giả sử $M = (S, I, O, f, g, s_0)$ là một máy hữu hạn trạng thái. Ta sẽ dùng một **bảng trạng thái** để biểu diễn các giá trị của hàm chuyển f và hàm đầu ra g cho tất cả các cặp gồm một trạng thái và một đầu vào. Ở trên ta đã từng xây dựng một bảng như vậy cho máy bán hàng được xét trong phần mở đầu của tiết này.

Ví dụ 1. Bảng 2 là bảng trạng thái mô tả một máy hữu hạn trạng thái với $S = \{s_0, s_1, s_2, s_3\}$, $I = \{0, 1\}$ và $O = \{0, 1\}$. Các giá trị của hàm f được cho trong hai cột đầu tiên và các giá trị của hàm g được cho trong hai cột cuối cùng.

■

Một cách khác để biểu diễn một máy hữu hạn trạng thái là dùng giản đồ trạng thái. Đó là một đồ thị có hướng với các cạnh được đánh dấu. Trong giản đồ này mỗi trạng thái được biểu diễn bởi một vòng tròn, các mũi tên được đánh dấu bởi cặp đầu vào và đầu ra cho mỗi chuyển dịch trạng thái.

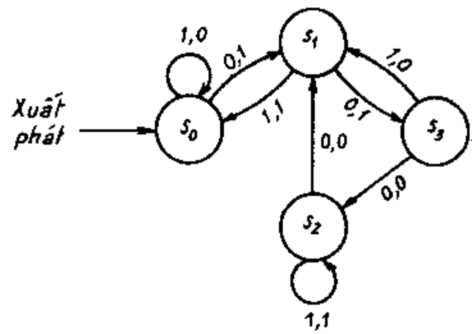
Ví dụ 2. Dựng giản đồ trạng thái cho một máy hữu hạn trạng thái với bảng trạng thái là Bảng 2.

Giải: Giản đồ trạng thái của máy này được cho trên Hình 2.

■

BẢNG 2

	f		g	
Trạng thái	Đầu vào		Đầu vào	
	0	1	0	1
s_0	s_1	s_0	1	0
s_1	s_3	s_0	1	1
s_2	s_1	s_2	0	1
s_3	s_2	s_1	0	0



Hình 2. Giản đồ trạng thái biểu diễn máy hữu hạn trạng thái cho trong Bảng 2.

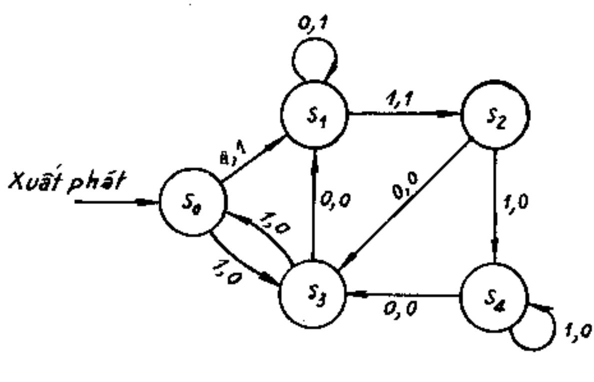
Ví dụ 3. Lập bảng trạng thái cho một máy hữu hạn trạng thái với giản đồ trạng thái cho trên Hình 3.

Giải: Bảng trạng thái cho máy đó là Bảng 3.

■

BẢNG 3

	f		g	
Trạng thái	Đầu vào		Đầu vào	
	0	1	0	1
s_0	s_1	s_3	1	0
s_1	s_1	s_2	1	1
s_2	s_3	s_4	0	0
s_3	s_1	s_0	0	0
s_4	s_3	s_4	0	0



Hình 3. Máy hữu hạn trạng thái.

Một xâu đầu vào đưa trạng thái xuất phát qua một dãy các trạng thái được xác định bởi hàm chuyển. Vì chúng ta đọc xâu đầu vào theo từng ký hiệu một (từ trái sang phải), nên mỗi ký hiệu đầu vào đưa máy từ một trạng thái này sang trạng thái khác. Vì mỗi một chuyển dịch trạng thái tạo ra một đầu ra, nên xâu đầu vào cũng tạo ra một xâu đầu ra.

Giả sử xâu đầu vào là $x = x_1x_2...x_k$. Khi đó việc đọc đầu vào này sẽ đưa máy từ trạng thái s_0 đến trạng thái s_1 với $s_1 = f(s_0, x_1)$, rồi tới trạng thái s_2 với $s_2 = f(s_1, x_2)$, v.v... cho tới khi kết thúc ở trạng thái s_k với $s_k = f(s_{k-1}, x_k)$. Dãy các dịch chuyển trạng thái này tạo ra một xâu đầu ra $y = y_1y_2...y_k$, với $y_1 = g(s_0, x_1)$ là đầu ra tương ứng với chuyển dịch từ s_0 đến s_1, $y_2 = g(s_1, x_2)$ là đầu ra tương ứng với chuyển dịch từ s_1 đến s_2, v.v... Nói một cách tổng quát $y_j = g(s_{j-1}, x_j)$ với $j = 1, 2,..., k$. Do đó, ta có thể mở rộng định nghĩa của hàm đầu ra g cho các xâu đầu vào sao cho $g(x) = y$ ở đây y là xâu đầu ra tương ứng với xâu đầu vào x. Khái niệm này rất hữu ích trong nhiều ứng dụng.

Ví dụ 4. Tìm xâu đầu ra được sinh bởi một máy hữu hạn trạng thái cho trên Hình 3 nếu xâu đầu vào là xâu 101011

Giải: Xâu đầu ra nhận được là 001000. Dãy các trạng thái và đầu ra tuần tự được cho trong Bảng 4.

■

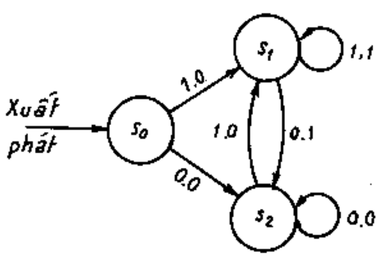
Bây giờ chúng ta có thể cho một số ví dụ về các máy hữu hạn trạng thái hữu ích. Các ví dụ này minh họa cho điều là các trạng thái của máy làm cho nó có một khả năng nhớ hạn chế. Các trạng thái có thể được dùng để nhớ những tính chất của các ký hiệu được đọc bởi máy. Tuy nhiên, vì máy chỉ có một số hữu hạn các trạng thái khác nhau, nên các máy hữu hạn trạng thái không thể được dùng cho một số mục đích quan trọng. Điều này sẽ được minh họa trong Tiết 10.4.

BẢNG 4

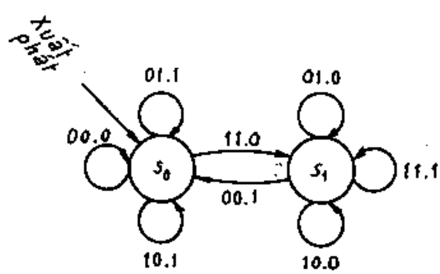
Đầu vào	1	0	1	0	1	1	–
Trạng thái	s_0	s_3	s_1	s_2	s_3	s_0	s_3
Đầu ra	0	0	1	0	0	0	–

Ví dụ 5. Một phần tử quan trọng trong nhiều dụng cụ điện tử là *máy trễ – đơn vị*, đây là máy tạo nên đầu ra chính là xâu đầu vào nhưng bị trễ một lượng thời gian cho trước. Vậy một máy hữu hạn trạng thái có thể được xây dựng như thế nào để nó làm trễ xâu đầu vào một đơn vị thời gian, tức là tạo nên đầu ra là xâu nhị phân $0x_1x_2...x_{k-1}$ từ xâu đầu vào đã cho là $x_1x_2...x_k$?

Giải: Một máy trễ có thể được xây dựng sao cho nó có hai đầu vào khả dĩ, cụ thể là 0 và 1. Máy này cần có một trạng thái xuất phát s_0. Vì máy cần phải nhớ đầu vào trước đó là 0 hay 1 nên máy này cần có hai trạng thái s_1 và s_2 trong đó máy sẽ ở trạng thái s_1 nếu đầu vào trước đó là 1 và ở trạng thái s_2 nếu đầu vào trước đó là 0. Đầu ra 0 sẽ được tạo ra đối với chuyển dịch ban đầu từ s_0. Mỗi một dịch chuyển từ s_1 sẽ cho đầu ra là 1, và mỗi một dịch chuyển từ s_2 sẽ cho đầu ra là 0. Như vậy, đầu ra tương ứng với xâu đầu vào $x_1...x_k$ sẽ là xâu bắt đầu bằng số 0, tiếp theo bởi x_1, rồi x_2 và kết thúc ở x_{k-1}. Giản đồ trạng thái của máy này được cho trên Hình 4. ■



Hình 4. Máy trễ-đơn vị



Hình 5. Máy cộng

Ví dụ 6. Tạo một máy hữu hạn trạng thái để cộng hai số nguyên ở dạng nhị phân

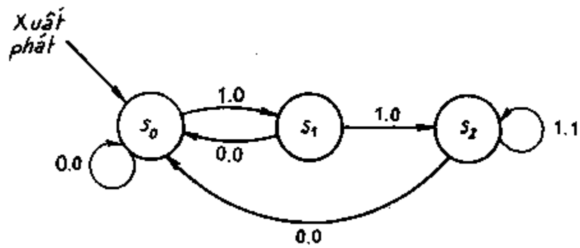
Giải: Khi $(x_n...x_1x_0)_2$ và $(y_n...y_1y_0)_2$ được cộng với nhau, thủ tục (như được mô tả ở tiết 2–4) sẽ diễn ra như nhau. Trước hết, x_0 và y_0 được cộng với nhau tạo thành bit tổng z_0 và bit nhớ c_0. Bit nhớ này hoặc là 0 hoặc là 1. Sau đó, các bit x_1 và y_1 được cộng với nhau cùng với bit nhớ c_0. Kết quả được bit tổng z_1 và bit nhớ c_1. Thủ tục này cứ tiếp tục mãi cho tới giai đoạn thứ n, ở đó x_n, y_n và số nhớ trước c_{n-1} được cộng với nhau cho bit tổng z_n và số nhớ c_n, số nhớ này đúng bằng bit tổng z_{n+1}.

Một máy hữu hạn trạng thái thực hiện phép cộng này có thể được xây dựng bằng cách chỉ dùng hai trạng thái. Để đơn giản ta giả sử rằng cả hai bit khởi đầu x_n và y_n đều là 0 (vì nếu không ta cần phải có sự bố trí đặc biệt liên quan đến bit tổng z_{n+1}). Trạng thái xuất phát s_0 được dùng để nhớ rằng bit nhớ trước đó là 0 (hay đối với phép cộng các bit ở bên phải cùng). Một trạng thái khác là s_1 được dùng để nhớ rằng bit nhớ trước đó là 1. Vì đầu vào của máy là cặp các bit, nên có bốn đầu vào khả dĩ. Chúng ta biểu diễn các khả năng này là 00 (khi cả hai bit đều là 0), 01 (khi bit thứ nhất là 0, bit thứ hai là 1), 10 (khi bit thứ nhất là 1, bit thứ hai là 0) và 11 (khi cả hai bit đều là 1). Các chuyển dịch trạng thái và đầu ra được xây dựng từ tổng hai bit được biểu diễn bởi đầu vào và bit nhớ được biểu diễn bởi trạng thái. Ví dụ, khi máy ở trạng thái s_1 và nhận đầu vào là 01, thì trạng thái tiếp sau vẫn là s_1 và đầu ra là 0 vì tổng xuất hiện khi này là $0 + 1 + 1 = (10)_2$. Giản đồ trạng thái của máy này cho trên Hình 5.

■

Ví dụ 7. Trong một sơ đồ mã hóa nào đó, khi có ba số 1 liên tiếp xuất hiện trong một thông báo, thì máy thu thông báo biết rằng đã có một sai sót truyền tin. Hãy xây dựng một máy hữu hạn trạng thái cho bit đầu ra là 1 nếu và chỉ nếu ba bit cuối cùng nhận được đều là số 1.

Giải: Máy này cần có ba trạng thái. Trạng thái xuất phát s_0 nhớ rằng giá trị đầu vào trước đó, nếu có, không phải là 1. Trạng thái s_1 nhớ rằng đầu vào trước đó là 1, nhưng đầu vào trước đầu vào trước đó, nếu có, lại không phải là 1. Trạng thái s_2 nhớ rằng hai đầu vào trước đó đều là 1. Đầu vào 1 đưa s_0 sang s_1 vì bây giờ một số 1 chứ không phải hai số 1 liên tiếp đã được đọc. Đầu vào 1 đưa s_1 sang s_2 vì bây giờ hai số 1 liên tiếp đã được đọc và đưa s_2 tới chính nó vì ít nhất có hai số 1 liên tiếp đã được đọc. Đầu vào 0 đưa tất cả các trạng thái về s_0 vì nó phá vỡ mọi xâu gồm các số 1 liên tiếp. Đầu ra đối với chuyển dịch từ s_2 đến chính nó là 1 khi 1 được đọc, vì tổ hợp gồm đầu vào và trạng thái này cho thấy ba số 1 liên tiếp đã được đọc. Tất cả các đầu ra khác đều là 0. Giản đồ trạng thái của máy này cho trên Hình 6.



Hình 6. Máy hữu hạn trạng thái cho đầu ra là 1 nếu và chỉ nếu xâu đầu vào được đọc tới lúc đó kết thúc bởi 111.

Máy trên Hình 6 là một ví dụ của **bộ nhận ngôn ngữ**, bởi vì nó tạo một đầu ra là 1 nếu và chỉ nếu xâu đầu vào được đọc tới lúc này có một tính chất đã được chỉ rõ trước. Sự chấp nhận ngôn ngữ là một ứng dụng quan trọng của các máy hữu hạn trạng thái.

Các loại máy hữu hạn trạng thái. Có nhiều loại máy hữu hạn trạng thái khác nhau đã được phát triển để mô hình hóa các máy tính toán. Trong tiết này ta đã cho định nghĩa của một loại máy hữu hạn trạng thái. Trong loại máy này các đầu ra tương ứng với sự chuyển dịch giữa các trạng thái. Những máy thuộc loại này được gọi là các **máy Mealy** vì chúng được nghiên cứu đầu tiên bởi G. H. Mealy vào năm 1955. Còn một loại máy hữu hạn trạng thái có đầu ra quan trọng khác, trong đó đầu ra chỉ được xác định bởi trạng thái. Loại máy này được gọi là các **máy Moore** vì E. F. Moore đã đưa ra loại máy này vào năm 1956. Các máy Moore được xét trong chuỗi các bài tập ở cuối tiết này.

Trong Ví dụ 7 chúng ta đã cho thấy một máy Mealy có thể được dùng để chấp nhận ngôn ngữ như thế nào. Tuy nhiên, một loại máy hữu hạn trạng thái khác, không cho đầu ra, thường được dùng cho mục đích đó. Các máy này cũng còn được gọi là các ôtômat hữu hạn, chúng có một tập các trạng thái kết thúc và chấp nhận một xâu nếu và chỉ nếu xâu đó đưa được trạng thái xuất phát đến một trạng thái kết thúc. Chúng ta sẽ nghiên cứu loại máy hữu hạn trạng thái này trong Tiết 10.3.

BÀI TẬP

1. Vẽ giản đồ trạng thái đối với các mãy hữu hạn trạng thái có bảng trạng thái sau:

a)

Trạng thái	f		g	
	Đầu vàa		Đầu vàa	
	0	1	0	1
s_0	s_1	s_0	0	1
s_1	s_0	s_2	0	1
s_2	s_1	s_1	0	0

b)

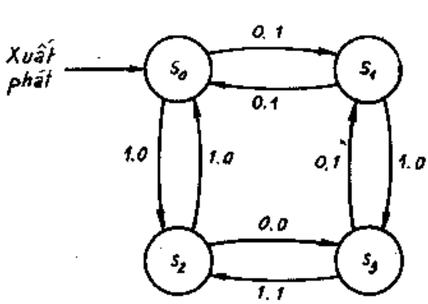
Trạng thái	f		g	
	Đầu vào		Đầu vào	
	0	1	0	1
s_0	s_1	s_0	0	0
s_1	s_2	s_0	1	1
s_2	s_0	s_3	0	1
s_3	s_1	s_2	1	0

c)

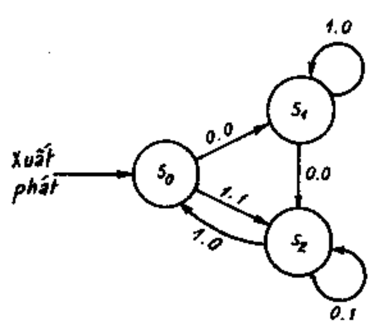
Trạng thái	f		g	
	Đầu vào		Đầu vàa	
	0	1	0	1
s_0	s_0	s_4	1	1
s_1	s_0	s_3	0	1
s_2	s_0	s_2	0	0
s_3	s_1	s_1	1	1
s_4	s_1	s_3	1	0

2. Lập bảng trạng thái đối với các máy hữu hạn trạng thái có giản đồ trạng thái sau:

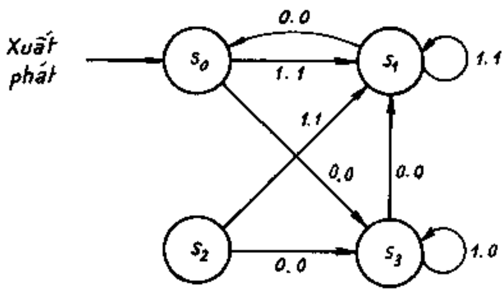
a)



b)



c)



3. Cho máy hữu hạn trạng thái nói trong Ví dụ 2, hãy xác định đầu ra ứng với mỗi xâu đầu vào sau:

a) 0111 b) 11011011 c) 01010101010

4. Cho máy hữu hạn trạng thái nói trong Ví dụ 3, hãy xác định đầu ra ứng với mỗi xâu đầu vào sau:

a) 0000 b) 101010 c) 11011100010

5. Dựng một máy hữu hạn trạng thái là mô hình của một máy bán đồ uống. Máy này nhận các đồng 5 xu, 10 xu và 25 xu. Máy chấp nhận thay đổi chừng nào có 35 xu được thả vào máy. Nó sẽ thối lại số tiền vượt quá 35 xu. Sau đó, khách hàng ấn các nút có thể nhận được một cốc coca, một cốc bia hoặc một li rượu gừng.

6. Dựng một máy hữu hạn trạng thái là mô hình của máy bán báo. Máy này có một cửa chỉ mở được sau khi đã thả vào máy ba đồng 10 xu (và một số bất kỳ các đồng xu khác) hoặc một đồng 25 xu và một đồng 5 xu (và một số bất kỳ các đồng xu khác). Một khi cửa có thể mở được, khách hàng sẽ mở nó, lấy báo rồi lại đóng cửa lại. Tiền thừa sẽ không được thối lại bất kể số tiền thừa là bao nhiêu. Khách hàng tiếp sau bắt đầu không có nợ nần gì đối với người trước.

7. Dựng một máy hữu hạn trạng thái làm trễ hai bit một xâu đầu vào bằng cách cho hai bit đầu tiên của đầu ra là 00.

8. Dựng một máy hữu hạn trạng thái làm thay đổi mỗi một bit khác, bắt đầu từ bit thứ hai, của một xâu đầu vào để các bit khác không thay đổi.

9. Dựng một máy hữu hạn trạng thái để tăng nhập vào thủ tục đối với một máy tính, trong đó người dùng nối với hệ thống bằng cách đưa vào số định danh (ID) của người dùng – được xem như một đầu vào đơn – và sau đó, đưa vào mật khẩu – cũng được xem như một đầu vào đơn. Nếu mật khẩu không đúng, thì người dùng lại bị hỏi lại số định danh người dùng.

10. Dựng một máy hữu hạn trạng thái cho một khóa tổ hợp chứa các số từ 1 đến 40. Khóa chỉ mở khi đưa vào tổ hợp đúng: 10 bên phải, 8 thứ hai bên trái, 37 bên phải. Mỗi đầu vào là một bộ ba bao gồm một con số, hướng quay và số lần khóa được quay theo hướng đó.

11. Dựng một máy hữu hạn trạng thái cho một máy thu phí cầu đường. Máy sẽ mở cổng sau khi 25 xu (có thể gồm các đồng 5 xu, 10 xu và 25 xu) được thả vào máy. Không có sự trao đổi nào nếu người lái xe trả quá 25 xu và khi đó cũng không có sự gán nợ cho người lái xe tiếp ngay sau.

12. Dựng một máy hữu hạn trạng thái cho đầu ra là 1 nếu số các ký hiệu đầu vào được đọc cho tới lúc này chia hết cho 3 và cho đầu ra là 0 trong các trường hợp còn lại.

13. Dựng một máy hữu hạn trạng thái để xác định xem một xâu đầu vào có một số 1 ở vị trí cuối cùng và một số 0 ở vị trí thứ ba đối với vị trí cuối cùng được đọc tới lúc đó hay không.

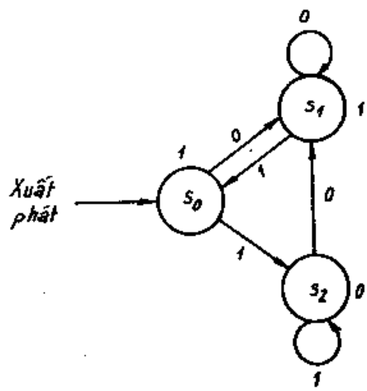
14. Dựng một máy hữu hạn trạng thái để xác định xem xâu đầu vào được đọc cho tới lúc này có kết thúc bằng năm số 1 liên tiếp hay không.

15. Dựng một máy hữu hạn trạng thái để xác định xem từ *computer* đã được đọc như 8 ký tự cuối cùng trong đầu vào được đọc tới lúc này hay không. Biết rằng đầu vào có thể là xâu bất kỳ gồm các chữ cái tiếng Anh.

Máy Moore $M = (S, I, O, f, g, s_0)$ *gồm một tập hữu hạn S các trạng thái, một bảng chữ cái đầu vào I, một bảng chữ cái đầu ra O, một hàm chuyển f gán trạng thái tiếp theo cho mỗi cặp một trạng thái và một đầu vào, một hàm đầu ra g gán một đầu ra cho mỗi trạng thái, và trạng thái* *xuất phát s_0. Một máy Moore có thể được biểu diễn bởi một bảng liệt kê các chuyển dịch đối với mỗi cặp trạng thái và đầu vào, và đầu ra đối với mỗi trạng thái. Máy Moore cũng có thể được biểu diễn bởi giản đồ trạng thái trong đó thể hiện các trạng thái, những chuyển dịch giữa các trạng thái và đầu ra đối với mỗi trạng thái. Trong giản đồ trạng thái, các chuyển dịch được chỉ bằng các mũi tên có đánh dấu bằng đầu vào, còn các đầu ra được ghi cạnh các trạng thái tương ứng.*

16. Dựng giản đồ trạng thái cho máy Moore có bảng trạng thái sau:

Trạng thái	f		g
	Đầu vào		
	0	1	
s_0	s_0	s_2	0
s_1	s_3	s_0	1
s_2	s_2	s_1	1
s_3	s_2	s_0	1



17. Dựng bảng trạng thái cho một máy Moore có giản đồ trạng thái cho ở hình trên.

Mỗi xâu đầu vào đối với một máy Moore M sẽ tạo ra một xâu đầu ra. Đặc biệt, đầu ra tương ứng với xâu đầu vào $a_1a_2...a_k$ là xâu $g(s_0)g(s_1)...g(s_k)$, ở đây $s_i = f(s_{i-1}, a_i)$ với $i = 1, 2, ..., k$.

18. Tìm xâu đầu ra tạo bởi máy Moore trong Bài tập 16 với các xâu đầu vào tương ứng là:

a) 0101 b) 111111 c) 11101110111

19. Cũng hỏi như trên cho máy Moore trong Bài tập 17.

20. Dựng một máy Moore cho đầu ra là 1 bất kỳ khi nào số các ký hiệu trong xâu đầu vào được đọc tới lúc này chia hết cho 4.

21. Dựng một máy Moore để xác định xem một xâu đầu vào chứa một số chẵn hay số lẻ các số 1. Máy sẽ cho đầu ra là 1 nếu trong xâu có một số chẵn các số 1 và đầu ra 0 nếu trong xâu có một số lẻ các số 1.

10.3. MÁY HỮU HẠN TRẠNG THÁI KHÔNG CÓ ĐẦU RA

MỞ ĐẦU

Một trong những ứng dụng quan trọng của các máy hữu hạn trạng thái là sự chấp nhận ngôn ngữ. Ứng dụng này đóng vai trò cơ bản trong việc thiết kế và xây dựng các chương trình dịch cho các ngôn ngữ lập trình. Trong Tiết 10.2, chúng ta đã chứng tỏ rằng một máy hữu hạn trạng thái có đầu ra có thể được dùng để chấp nhận các ngôn ngữ bằng cách cho đầu ra 1 khi một xâu của ngôn ngữ đã được đọc và đầu ra 0 trong trường hợp ngược lại. Tuy nhiên, có các loại máy hữu hạn trạng thái khác được thiết kế chuyên để chấp nhận các ngôn ngữ. Thay vì tạu ra đầu ra, các máy này có những trạng thái kết thúc. Một xâu được chấp nhận nếu và chỉ nếu nó đưa trạng thái xuất phát tới một trạng thái kết thúc.

TẬP CÁC XÂU

Trước khi thảo luận về các máy hữu hạn trạng thái không có đầu ra, chúng tôi sẽ giới thiệu một số kiểu thức cơ sở quan trọng về tập các xâu. Các phép toán được định nghĩa ở đây sẽ được dùng rộng rãi trong thảo luận của chúng ta về sự chấp nhận ngôn ngữ bởi các máy có hữu hạn trạng thái.

ĐỊNH NGHĨA 1. Cho A và B là hai tập con của V^*, với V là một từ vựng. **Phép ghép** của A và B, được ký hiệu bởi AB, là tập tất cả các xâu có dạng xy trong đó x là xâu thuộc A và y là xâu thuộc B.

Ví dụ 1. Cho $A = \{0, 11\}$ và $B = \{1, 10, 110\}$. Tìm AB và BA.

Giải: Tập AB chứa tất cả các phép ghép của một xâu trong A và một xâu trong B. Do đó, $AB = \{01, 010, 0110, 111, 1110, 11110\}$. Tập BA chứa tất cả các phép ghép của một xâu trong B và một xâu trong A. Do đó, $BA = \{10, 111, 100, 1011, 1100, 11011\}$.

■

Chú ý rằng không nhất thiết phải có $AB = BA$, khi A và B là các tập con của V^*, như Ví dụ 1 đã cho thấy.

Từ định nghĩa phép ghép của hai tập các xâu, ta có thể định nghĩa A^n với $n = 0, 1, 2, \ldots$ Điều này được làm một cách đệ quy như sau:

$$A^0 = \{\lambda\}$$

$$A^{n+1} = A^nA \text{ với } n = 0, 1, 2, \ldots$$

Ví dụ 2. Cho $A = \{1, 00\}$. Tìm A^n với $n = 0, 1, 2$ và 3

Giải: Ta có $A^0 = \{\lambda\}$ và $A^1 = A^0A = \{\lambda\}A = \{1, 00\}$. Để tìm A^2 ta ghép các cặp phần tử của A. Kết quả được $A^2 = \{11, 100, 001, 0000\}$. Để tìm A^3, ta ghép các phần tử trong A^2 và A, kết quả cho $A^3 = \{111, 1100, 1001, 10000, 0011, 00100, 00001, 000000\}$.

■

ĐỊNH NGHĨA 2. Cho A là một tập con của V^*. Khi đó *bao đóng Kleen* của A – được ký hiệu là A^* – là tập gồm các phép ghép một số tùy ý các xâu thuộc A. Điều này có nghĩa là $A^* = \cup_{k=0}^{\infty} A^k$

Ví dụ 3. Tìm bao đóng Kleen của các tập sau: $A = \{0\}$, $B = \{0, 1\}$ và $C = \{1, 1\}$.

Giải: Bao đóng Kleen của A là phép ghép của xâu 0 với chính nó một số hữu hạn lần tùy ý. Do đó, $A^* = \{0^n \mid n = 0, 1, 2, \ldots\}$. Bao đóng Kleen của B là phép ghép một số tùy ý các xâu, trong đó mỗi xâu là 0 hoặc 1. Đây chẳng qua là tập các xâu trên bộ chữ cái $\{0, 1\}$. Tức $B^* = V^*$. Cuối cùng, bao đóng Kleen của C là phép ghép xâu 11 với chính nó một số lần tùy ý. Do đó, C^* là tập các xâu gồm một số chẵn các số 1. Tức là, $C^* = \{1^{2n} \mid n = 0, 1, 2, \ldots\}$.

■

ÔTÔMAT HỮU HẠN

Bây giờ chúng ta sẽ cho định nghĩa của máy hữu hạn trạng thái không có đầu ra. Những máy này cũng được gọi là các **ôtômat hữu hạn** và từ đây trở đi ta sẽ dùng thuật ngữ này. Các ôtômat hữu hạn khác với các máy hữu hạn trạng thái mà ta đã xét trong Tiết 10.2 ở chỗ chúng không tạo ra đầu ra mà có một tập các trạng thái kết thúc. Như chúng ta sẽ thấy, các ôtômat hữu hạn chấp nhận các xâu đưa trạng thái xuất phát tới một trạng thái kết thúc.

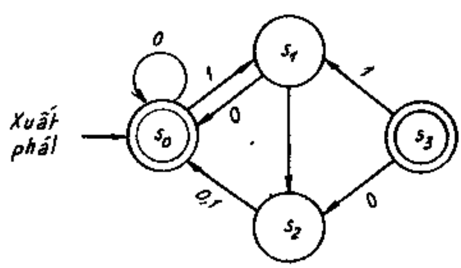
ĐỊNH NGHĨA 3. *Một ôtômat hữu hạn* $M = (S, I, f, s_0, F)$ gồm một tập hữu hạn S *các trạng thái*, một *bộ chữ cái đầu vào* I, một hàm chuyển f gán trạng thái tiếp theo cho mỗi cặp trạng thái và đầu vào, trạng thái xuất phát s_0 và một tập con F của S gồm các trạng thái kết thúc.

Chúng ta có thể biểu diễn một ôtômat hữu hạn bằng cách dùng các bảng trạng thái hoặc các giản đồ trạng thái. Các trạng thái kết thúc được thể hiện bằng các vòng tròn kép trong giản đồ trạng thái.

Ví dụ 4. Dựng giản đồ trạng thái của ôtômat hữu hạn $M = (S, I, f, s_0, F)$ với $S = \{s_0, s_1, s_2, s_3\}$, $I = \{0, 1\}$, $F = \{s_0, s_3\}$ và hàm chuyển f được cho trong Bảng 1.

BẢNG 1

Trạng thái	f: Đầu vào 0	f: Đầu vào 1
s_0	s_0	s_1
s_1	s_0	s_2
s_2	s_0	s_0
s_3	s_2	s_1



Hình 1. Giản đồ trạng thái của một ôtômat hữu hạn

Giải: Giản đồ trạng thái được cho trên Hình 1. Chú ý rằng vì cả hai đầu vào 0 và 1 đều đưa s_2 tới s_0, nên ta viết 0, 1 trên cạnh từ s_2 đến s_0.

Hàm chuyển f có thể được mở rộng sao cho nó được định nghĩa cho mọi cặp gồm một trạng thái và một xâu. Giả sử $x = x_1x_2...x_k$ là xâu trong I^*. Khi đó $f(s_1, x)$ là trạng thái nhận được bằng cách dùng tuần tự các ký hiệu trong x, từ trái sang phải, làm đầu vào, bắt đầu với trạng thái s_1. Từ s_1 ta có thể đi tới trạng thái $s_2 = f(s_1, x_1)$, sau đó tới trạng thái $s_3 = f(s_2, x_2)$ v.v... với $f(s_1, x) = f(s_k, x_k)$.

Một xâu x được gọi là **được chấp nhận** bởi máy $M = (S, I, f, s_0, F)$ nếu nó đưa trạng thái xuất phát tới một trạng thái kết thúc, tức là $f(s_0, x)$ là một trạng thái thuộc F. **Ngôn ngữ được chấp nhận** bởi máy M, được ký hiệu là $L(M)$, là tập tất cả các xâu được chấp nhận bởi M. Hai ôtômat hữu hạn được gọi là **tương dương**, nếu chúng cùng chấp nhận một ngôn ngữ.

Ví dụ 5. Xác định ngôn ngữ được chấp nhận bởi các ôtômat hữu hạn M_1, M_2 và M_3 trên Hình 2.

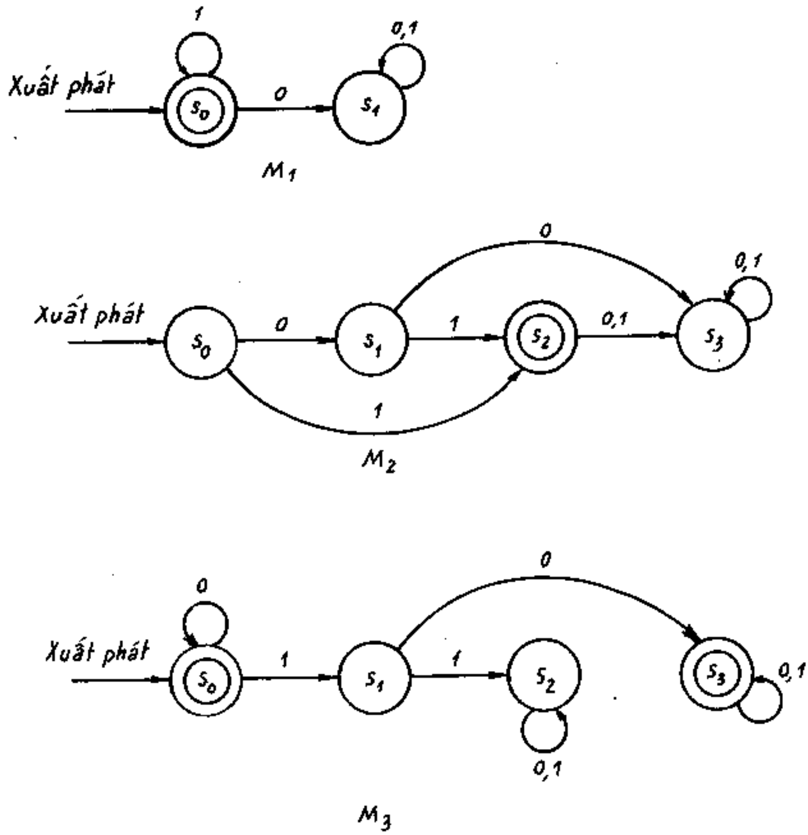
Giải: Trạng thái kết thúc duy nhất của M_1 là s_0. Các xâu đưa s_0 tới chính nó là xâu rỗng và các xâu chứa toàn các số 1. Do đó, $L(M_1) = \{1^n \mid n = 0, 1, 2,...\}$

Trạng thái kết thúc duy nhất của M_2 là s_2. Các xâu duy nhất đưa s_0 tới s_2 là 1 và 01. Do đó, $L(M_2) = \{1, 01\}$.

Các trạng thái cuối cùng của M_3 là s_0 và s_3. Các xâu duy nhất đưa s_0 đến chính nó là λ, 0, 00, 000,..., tức là xâu rỗng và các xâu gồm toàn số 0. Còn các xâu duy nhất đưa s_0 tới s_3 **là xâu** rỗng hoặc các xâu gồm các số 0 liên tiếp, tiếp sau bởi 10 rồi tiếp sau nữa bởi một xâu bất kỳ. Do đó, $L(M_3) = \{0^n, 0^n10x \mid n = 0, 1, 2,...$ và x là xâu bất kỳ$\}$.

■

Các ôtômat được xét cho tới đây đều là các **ôtômat tất định**, vì đối với mỗi cặp trạng thái và giá trị đầu vào có một trạng thái kế tiếp duy nhất được cho bởi hàm chuyển. Tuy nhiên, còn có một loại ôtômat hữu hạn quan trọng khác trong đó có thể có một số trạng thái kế tiếp khả dĩ ứng với mỗi cặp giá trị đầu vào và trạng thái. Những máy này được gọi là **không tất định**. Các ôtômat hữu hạn không tất định đóng vai trò quan trọng trong việc xác định các ngôn ngữ nào là được chấp nhận bởi một ôtômat hữu hạn.



Hình 2. Một số ôtômat hữu hạn

ĐỊNH NGHĨA 4. *Ôtômat hữu hạn không tất định* $M = (S, I, f, s_0, F)$ gồm tập S các trạng thái, một bộ chữ cái đầu vào I, một hàm chuyển f gán cho mỗi cặp gồm trạng thái và đầu vào một tập các trạng thái, trạng thái xuất phát s_0 và tập con F của S gồm các trạng thái kết thúc.

Ta cũng có thể biểu diễn các ôtômat hữu hạn không tất định bằng một bảng các trạng thái hoặc giản đồ trạng thái. Khi chúng ta dùng bảng trạng thái, đối với mỗi cặp gồm trạng thái và giá trị đầu vào, ta cho một liệt kê các trạng thái kế tiếp khả dĩ. Còn trong giản đồ trạng thái, chúng ta đưa vào các cạnh từ mỗi trạng thái tới tất cả các trạng thái kế tiếp khả dĩ, và đánh dấu các cạnh đó bằng đầu vào hoặc các đầu vào dẫn tới các chuyển dịch đó.

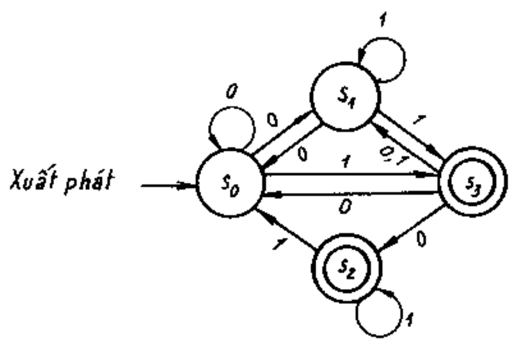
Ví dụ 6. Tìm giản đồ trạng thái cho ôtômat hữu hạn không tất định với bảng trạng thái là Bảng 2. Biết các trạng thái kết thúc là s_2 và s_3.

Giải: Giản đồ trạng thái của ôtômat hữu hạn không tất định này được cho trên Hình 3.

■

BẢNG 2

Trạng thái	f Đầu vào 0	 1
s_0	s_0, s_1	s_3
s_1	s_0	s_1, s_3
s_2		s_0, s_2
s_3	s_2, s_1, s_2	s_1



Hình 3. Ôtômat hữu hạn không tất định với bảng trạng thái là Bảng 2

Ví dụ 7. Lập bảng trạng thái đối với ôtômat hữu hạn không tất định có giản đồ trạng thái cho trên Hình 4.

Giải: Bảng trạng thái cần tìm được cho trên Bảng 3.

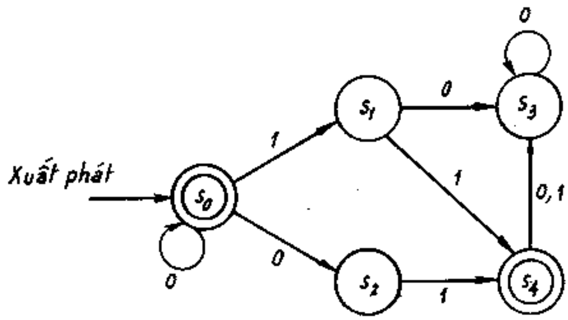
■

Đối với một ôtômat hữu hạn không tất định, việc chấp nhận một xâu $x = x_1x_2..x_k$ có nghĩa là như thế nào? Ký hiệu đầu tiên của đầu vào x_1 đưa trạng thái xuất phát s_0 tới tập S_1 các trạng thái. Ký hiệu thứ hai của đầu vào x_2 đưa mỗi một trạng thái của tập S_1 tới một tập các trạng thái. Gọi S_2 là hợp của các tập này. Chúng ta tiếp tục quá trình này bằng cách ở mỗi giai đoạn gộp vào tất cả các trạng thái nhận được bằng cách dùng một trạng thái nhận được ở giai đoạn trước và ký hiệu đầu vào hiện thời. Chúng ta **chấp nhận** một xâu x nếu có một trạng thái kết thúc trong tập tất cả các trạng thái có thể nhận được từ s_0 khi dùng x. **Ngôn ngữ**

được chấp nhận bởi một ôtômat hữu hạn không tất định là tập tất cả các xâu được chấp nhận bởi ôtômat đó.

BẢNG 3

Trạng thái	f: Đầu vào 0	1
s_0	s_0, s_2	s_1
s_1	s_3	s_4
s_2		s_4
s_3	s_3	
s_4	s_3	s_3



Hình 4. Một ôtômat hữu hạn không tất định.

Ví dụ 8. Tìm ngôn ngữ được chấp nhận bởi ôtômat hữu hạn không tất định được cho trên Hình 4.

Giải: Vì s_0 là trạng thái kết thúc và có một chuyển dịch s_0 đến chính nó khi đầu vào là 0, nên máy này chấp nhận tất cả các xâu là xâu rỗng hoặc xâu chứa toàn các số 0. Hơn nữa, vì s_4 cũng là trạng thái kết thúc, nên mọi xâu có s_4 ở trong tập các trạng thái có thể đưa tới từ s_0 nhờ xâu đầu vào đó, đều được chấp nhận. Những xâu duy nhất có tính chất này là xâu rỗng hoặc các xâu gồm toàn số không tiếp sau bởi 01 hoặc 11. Vì s_0 và s_4 là các trạng thái kết thúc duy nhất, nên ngôn ngữ được chấp nhận bởi máy là $\{0^n, 0^n01, 0^n11 \mid n \geqslant 0\}$.

■

Có một tính chất quan trọng là: một ngôn ngữ được chấp nhận bởi một ôtômat hữu hạn không tất định cũng sẽ được chấp nhận bởi một ôtômat hữu hạn tất định. Chúng ta sẽ sử dụng kết quả này trong tiết sau khi chúng ta xác định các ngôn ngữ nào được chấp nhận bởi một ôtômat hữu hạn.

ĐỊNH LÝ 1. Nếu ngôn ngữ L được chấp nhận bởi một ôtômat hữu hạn không tất định M_0 thì L cũng được chấp nhận bởi một ôtômat tất định M_1.

Chứng minh. Chúng ta sẽ mô tả cách làm thế nào dựng được một ôtômat hữu hạn tất định M_1 chấp nhận L từ M_0 là ôtômat hữu hạn không tất định chấp nhận ngôn ngữ này. Mỗi một trạng thái trong M_1 sẽ được tạo bởi một tập các trạng thái trong M_0. Ký hiệu xuất phát của M_1 là $\{s_0\}$ – là tập có chứa trạng thái xuất phát của M_0. Tập đầu vào của M_1 cũng là tập đầu vào của M_0. Cho trạng thái $\{s_{i1}, s_{i2}, \ldots, s_{ik}\}$ của M_1, ký hiệu đầu vào x đưa trạng thái này tới hợp các tập của các trạng thái kế tiếp đối với mỗi phần tử của tập đó, tức là hợp của các tập hợp $f(s_{i1}), f(s_{i2}), \ldots, f(s_{ik})$. Các trạng thái của M_1 tất thảy đều là các tập con của S – tập các trạng thái của M_0 – nhận được bằng cách đó xuất phát từ s_0. (Có cả thảy 2^n trạng thái trong ôtômat hữu hạn tất định, với n là số trạng thái trong máy không tất định, vì tất cả các tập con đều có thể xuất hiện như các trạng thái, kể cả tập rỗng, mặc dù thông thường số các trạng thái thường gặp ít hơn nhiều). Các trạng thái kết thúc của M_1 là các tập nói trên có chứa một trạng thái kết thúc của M_0.

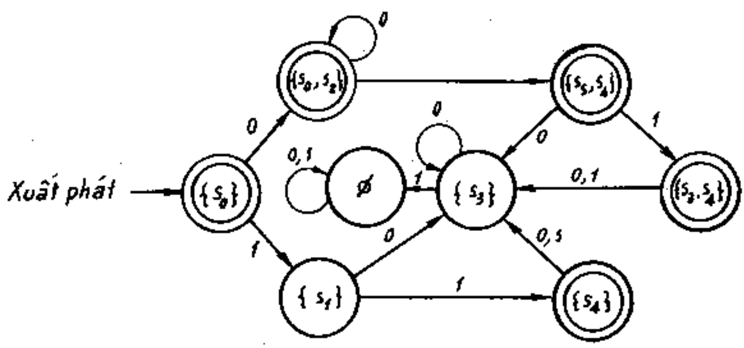
Giả sử một xâu đầu vào được chấp nhận bởi M_0. Khi đó một trong số các trạng thái có thể tới được từ s_0 nhờ xâu đầu vào này là một trạng thái kết thúc (độc giả có thể chứng minh điều này bằng quy nạp). Điều này có nghĩa là trong M_1, xâu đầu vào này dẫn từ $\{s_0\}$ tới một tập các trạng thái của M_0 có chứa một trạng thái kết thúc. Tập con này là một trạng thái cuối cùng của M_1, vậy xâu này được chấp nhận bởi M_1. Cũng như vậy, một xâu đầu vào không được chấp nhận bởi M_0 cũng không dẫn tới trạng thái kết thúc nào của M_0 (Độc giả nên chứng minh chi tiết khẳng định này). Do đó, xâu đầu vào này không dẫn từ $\{s_0\}$ tới một trạng thái kết thúc nào trong M_1.

□

Ví dụ 9. Tìm một ôtômat hữu hạn tất định chấp nhận cùng một ngôn ngữ như ôtômat hữu hạn không tất định cho trong Ví dụ 7.

Giải: Ôtômat tất định cho trên Hình 5 được xây dựng từ ôtômat không tất định cho trong Ví dụ 7. Các trạng thái của ôtômat tất định này là các tập con của tập tất cả các trạng thái của ôtômat không tất định. Trạng thái kế tiếp của một tập con dưới tác động của một ký hiệu đầu vào là một tập con chứa các trạng thái kế tiếp trong ôtômat không tất định của tất cả các phần tử trong tập con nói lúc đầu. Ví dụ, với đầu vào 0, $\{s_0\}$ sẽ chuyển tới $\{s_0, s_2\}$, vì s_0 có những chuyển dịch tới chính nó và tới s_2 trong ôtômat không tất định, tập con $\{s_0, s_2\}$ với đầu vào 1 sẽ chuyển tới $\{s_1, s_4\}$, vì trong máy không tất định với đầu vào 1 s_0 chỉ chuyển tới s_1 và s_2 chỉ chuyển tới s_4; với đầu vào 0 tập $\{s_1, s_4\}$ sẽ chuyển tới $\{s_3\}$ vì trong máy không tất định, với đầu vào 0, s_1 và s_4 đều chỉ chuyển tới s_3. Tất cả các tập con nhận được bằng cách đó đều được bao hàm trong máy tất định. Chú ý rằng tập rỗng cũng là một trong những trạng thái của máy này, vì nó là tập con chứa tất cả các trạng thái kế tiếp của $\{s_3\}$ với đầu vào 1. Trạng thái xuất phát là $\{s_0\}$ và các trạng thái kết thúc là các tập con có chứa s_0 hoặc s_4.

■



Hình 5. Ôtômat tất định tương ứng với ôtômat không tất định cho trong Ví dụ 7

BÀI TẬP

1. Cho $A = \{0, 11\}$ và $B = \{00, 01\}$. Hãy tìm các tập sau:

   a) AB b) BA c) A^2 d) B^3

2. Chứng minh rằng nếu A là tập các xâu, thì $A\varnothing = \varnothing A = \varnothing$

3. Tìm tất cả các tập A và B của các xâu sao cho $AB = \{10, 111, 1010, 1000, 10111, 101000\}$

4. Chứng minh các đẳng thức sau:

a) $\{\lambda\}^* = \{\lambda\}$

b) $(A^*)^* = A^*$ với mọi tập các xâu A

5. Mô tả các phần tử của tập A^* với các giá trị sau của A

a) $\{10\}$ b) $\{111\}$ c) $\{0, 01\}$ d) $\{1, 101\}$

6. Cho V là bộ chữ cái (từ vựng) và A và B là các tập con của V^*. Chứng minh rằng $|AB| \leq |A||B|$

7. Cho V là bộ chữ cái, A và B là các tập con của V^* với $A \subseteq B$. Chứng minh rằng $A^* \subseteq B^*$.

8. Cho A là một tập con của V^* với V là một bộ chữ cái. Hãy chứng minh hoặc bác bỏ các mệnh đề sau:

a) $A \subseteq A^2$ b) nếu $A = A^2$ thì $\lambda \in A$

c) $A\{\lambda\} = A$ d) $(A^*)^* = A^*$

e) $A^*A = A^*$ f) $|A^n| = |A|^n$

9. Xác định xem xâu 11101 có nằm trong các tập sau không?

a) $\{0, 1\}^*$ b) $\{1\}^*\{0\}^*\{1\}^*$ c) $\{11\}\{1\}^*\{01\}$

d) $\{11\}^*\{10\}^*$ e) $\{111\}^*\{0\}^*\{1\}$ f) $\{111, 000\}\{00, 01\}$

10. Hãy xác định xem các xâu dưới đây có được chấp nhận bởi ôtômat hữu hạn tất định trong Hình 1 không?

a) 010 b) 1101 c) 1111110 d) 010101010

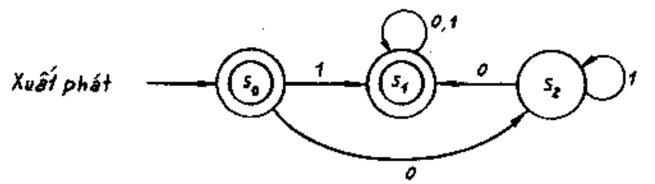
11. Hãy xác định xem tất cả các xâu trong các tập sau có được chấp nhận bởi ôtômat hữu hạn tất định trong Hình 1 không?

a) $\{0\}^*$ b) $\{0\}\{0\}^*$ c) $\{1\}\{0\}^*$

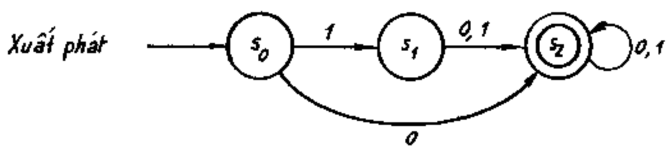
d) $\{01\}^*$ e) $\{0\}^*\{1\}^*$ f) $\{1\}\{0,1\}^*$

Trong các Bài tập từ 12 đến 16, hãy tìm ngôn ngữ được chấp nhận bởi một ôtômat hữu hạn tất định được cho dưới đây

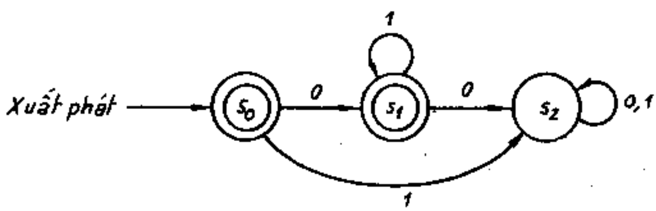
12.


Xuất phát
s_0
s_1
s_2
1
0
0
0,1
1

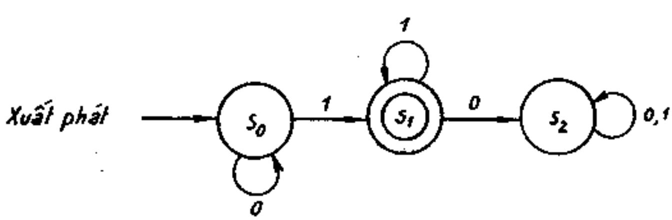
13.


Xuất phát
s_0
s_1
s_2
1
0,1
0
0,1

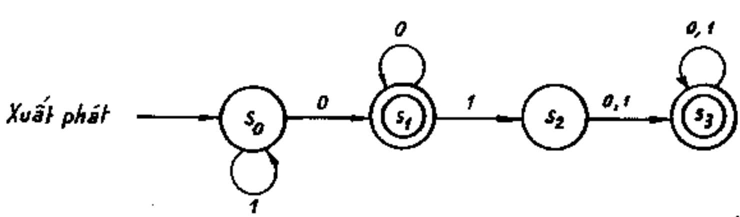
14.


Xuất phát
s_0
s_1
s_2
0
0
1
1
0,1

15.

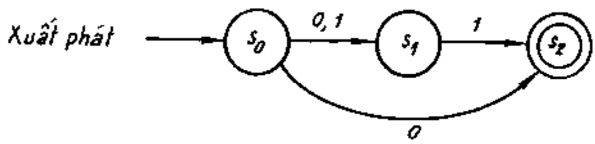

Xuất phát
s_0
s_1
s_2
1
0
1
0
0,1

16.

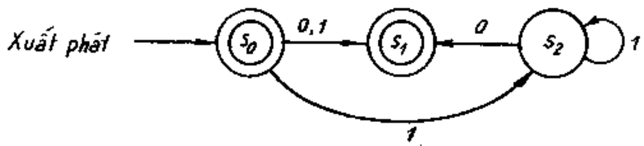

Xuất phát
s_0
s_1
s_2
s_3
0
1
0,1
0
1
0,1

Trong các Bài tập từ 17 đến 21, hãy tìm ngôn ngữ được chấp nhận bởi ôtômat hữu hạn không tất định được cho dưới đây

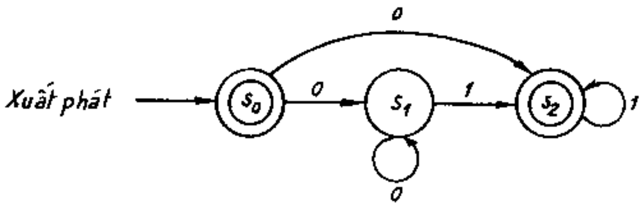
17.



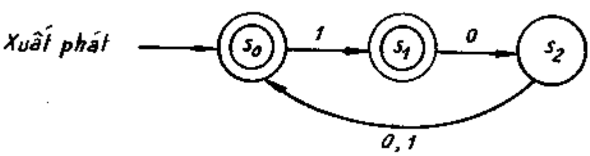
18.



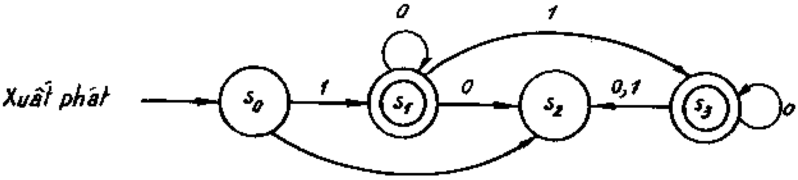
19.



20.



21.



22. Tìm ôtômat hữu hạn tất định chấp nhận cùng một ngôn ngữ như ôtômat hữu hạn không tất định trong Bài tập 17

23. Cũng hỏi như trên đối với ôtômat hữu hạn không tất định trong Bài tập 18

24. Cũng hỏi như trên đối với ôtômat hữu hạn không tất định trong Bài tập 19

25. Cũng hỏi như trên đối với ôtômat hữu hạn không tất định trong Bài tập 20

26. Cũng hỏi như trên đối với ôtômat hữu hạn không tất định trong Bài tập 21

27. Tìm các ôtômat hữu hạn tất định chấp nhận các tập sau:

a) $\{0\}$ b) $\{1, 00\}$ c) $\{1^n \mid n = 2, 3, 4...\}$

28. Tìm ôtômat hữu hạn không tất định chấp nhận các ngôn ngữ trong Bài tập 27 và có ít trạng thái hơn, nếu có thể, so với ôtômat hữu hạn tất định mà bạn đã tìm được trong bài tập đó.

***29.** Chứng minh rằng không có một ôtômat hữu hạn chấp nhận tập các xâu nhị phân chứa số các số 0 và số các số 1 bằng nhau.

10.4. SỰ CHẤP NHẬN NGÔN NGỮ

MỞ ĐẦU

Như chúng ta đã thấy ở trên, các ôtômat hữu hạn có thể được dùng như các bộ tiếp nhận ngôn ngữ. Vậy các máy này có thể chấp nhận các tập nào? Mặc dù điều này dường như là một bài toán cực kỳ khó, nhưng các tập có thể được chấp nhận bởi các ôtômat hữu hạn lại có một đặc trưng khá đơn giản. Bài toán này lần đầu tiên đã được nhà toán học Mỹ Stephen Kleene giải quyết vào năm 1956. Ông đã chứng minh được rằng tồn tại một ôtômat hữu hạn chấp nhận một tập hợp nếu và chỉ nếu tập đó có thể được xây dựng từ tập rỗng, xâu rỗng và các xâu chỉ chứa một ký hiệu bằng cách ghép, lấy hợp và lấy các bao đóng Kleene theo một trật tự tuỳ ý. Những tập có thể được xây dựng bằng cách như vậy được gọi là các **tập chính quy**.

Các văn phạm chính quy đã được định nghĩa trong Tiết 10.1. Theo thuật ngữ được sử dụng, ta chắc sẽ không có gì ngạc nhiên rằng có một mối quan hệ giữa các tập chính quy được chấp nhận bởi các ôtômat hữu hạn và các văn phạm chính quy. Đặc biệt, một tập là chính quy nếu và chỉ nếu nó được sinh bởi một văn phạm chính quy.

Cuối cùng, có những tập không thể được chấp nhận bởi các ôtômat hữu hạn. Ta sẽ cho một ví dụ về các tập đó. Ta cũng sẽ xem xét một cách ngắn gọn các mô hình tính toán mạnh hơn, như các ôtômat đẩy xuống (push down) và các máy Turing, ở cuối tiết này.

CÁC TẬP CHÍNH QUY

Các tập chính quy là các tập có thể được tạo bằng cách dùng các phép toán ghép, hợp và bao đóng Kleene theo một trật tự tùy ý xuất phát từ tập rỗng, xâu rỗng và các tập của các xâu chỉ chứa một ký hiệu. Chúng ta sẽ thấy rằng các tập chính quy là các tập có thể được chấp nhận bởi một ôtômat hữu hạn. Để định nghĩa các tập chính quy, trước hết ta cần phải định nghĩa các biểu thức chính quy.

ĐỊNH NGHĨA 1. *Các biểu thức chính quy* trên một tập I được định nghĩa một cách đệ quy như sau:

ký hiệu $\varnothing$ là một biểu thức chính quy

ký hiệu λ là một biểu thức chính quy

ký hiệu x là một biểu thức chính quy, với mọi $x \in I$

các ký hiệu $(\mathbf{AB})$, $(\mathbf{A} \cup \mathbf{B})$ và $\mathbf{A}^*$ là các biểu thức chính quy với mọi $\mathbf{A}$ và $\mathbf{B}$ là các biểu thức chính quy.

Mỗi biểu thức chính quy biểu diễn một tập được đặc tả bởi các quy tắc sau:

$\emptyset$ biểu diễn tập rỗng, tức là tập không chứa xâu nào;

λ biểu diễn tập $\{\lambda\}$, là tập chứa xâu rỗng;

x biểu diễn tập $\{x\}$ chứa xâu chỉ có một ký hiệu x;

$(\mathbf{AB})$ biểu diễn sự ghép của các tập được biểu diễn bởi $\mathbf{A}$ và $\mathbf{B}$

$\mathbf{A}^*$ biểu diễn bao đóng Kleene của tập được biểu diễn bởi $\mathbf{A}$

Các tập được biểu diễn bởi các biểu thức chính quy được gọi là các **tập chính quy**. Từ đây trở đi, các biểu thức chính quy sẽ được dùng để mô tả các tập chính quy, vì vậy khi ta nói tới tập chính quy A là muốn nói tới tập chính quy được biểu diễn bởi biểu thức chính quy A. Ví dụ sau cho thấy các biểu thức chính quy được dùng để đặc tả các tập chính quy như thế nào.

Ví dụ 1. Xác định các xâu trong các tập chính quy được đặc tả bởi các biểu thức chính quy sau: $\mathbf{10}^*$, $(\mathbf{10})^*$, $\mathbf{0} \cup \mathbf{01}$, $\mathbf{0}(\mathbf{0} \cup \mathbf{1})^*$ và $(\mathbf{0}^*\mathbf{1})^*$

Giải: Các tập chính quy được biểu diễn bởi các biểu thức đó được cho trong Bảng 1. (Độc giả nên kiểm tra lại).

■

BẢNG 1

Biểu thức	Xâu
10^*	Một số 1 được tiếp theo bởi một số bất kỳ số 0 (kể cả không có số không nào)
$(10)^*$	Một số bất kỳ các cặp 10 (kể cả xâu rỗng)
$0 \cup 01$	Xâu 0 hoặc xâu 01
$0(0 \cup 1)^*$	Xâu bất kỳ bắt đầu bằng 0
$(0^*1)^*$	Xâu bất kỳ không kết thúc bằng 0

ĐỊNH LÝ KLEENE

Năm 1956 Kleene đã chứng minh được rằng các tập chính quy là các tập được chấp nhận bởi một ôtômat hữu hạn. Do đó kết quả quan trọng này được gọi là định lý Kleene.

ĐỊNH LÝ 1 – ĐỊNH LÝ KLEENE Một tập là chính quy nếu và chỉ nếu nó được chấp nhận bởi một ôtômat hữu hạn.

Định lý Kleene là một trong những kết quả trung tâm của lý thuyết ôtômat. Ta sẽ chỉ chứng minh phần *chỉ nếu* của định lý này, cụ thể ta sẽ chứng minh rằng mọi tập chính quy đều được chấp nhận bởi một ôtômat hữu hạn. Sự chứng minh phần *nếu*, tức là chứng minh một tập được chấp nhận bởi một ôtômat hữu hạn là chính quy, xin dành cho bạn đọc như một bài tập.

Chứng minh. Cần nhớ lại rằng một tập chính quy được định nghĩa qua các biểu thức chính quy, mà các biểu thức chính quy lại được định nghĩa một cách đệ quy. Vì vậy, chúng ta có thể chứng minh được rằng mọi tập chính quy đều được chấp nhận bởi một ôtômat hữu hạn nếu chúng ta làm được các việc sau:

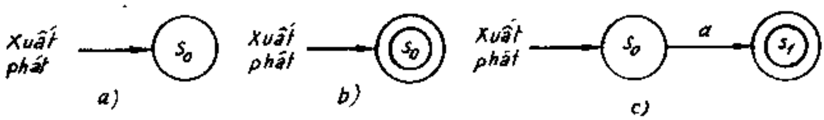
1. Chứng minh được rằng $\emptyset$ được chấp nhận bởi một ôtômat hữu hạn.
2. Chứng minh được rằng λ được chấp nhận bởi một ôtômat hữu hạn.
3. Chứng minh được rằng $\{a\}$ được chấp nhận bởi một ôtômat hữu hạn với mọi a là một ký hiệu trong I.
4. Chứng minh được rằng AB được chấp nhận bởi một ôtômat hữu hạn với mọi A và B đều được chấp nhận.
5. Chứng minh được rằng $A \cup B$ được chấp nhận bởi một ôtômat hữu hạn với mọi A và B đều đã được chấp nhận.
6. Chứng minh được rằng A^* được chấp nhận bởi một ôtômat hữu hạn với mọi A đã được chấp nhận.

□

Bây giờ chúng ta sẽ xét từng nhiệm vụ trên một. Trước hết, ta sẽ chứng tỏ rằng $\emptyset$ được chấp nhận bởi một ôtômat hữu hạn không tất định. Để làm điều này, ta chỉ cần một ôtômat không có các trạng thái kết thúc. Một ôtômat như vậy được cho trên Hình 1(a).

Thứ hai, ta sẽ chứng tỏ rằng $\{\lambda\}$ được chấp nhận bởi một ôtômat hữu hạn. Để làm điều đó, ta chỉ cần một ôtômat chấp nhận λ – xâu rỗng, nhưng không chấp nhận một xâu nào khác. Điều này có thể làm được bằng cách làm cho trạng thái xuất phát s_0 là trạng thái kết thúc và không có chuyển dịch nào, vì vậy không có xâu nào khác đưa s_0 đến trạng thái kết thúc. Ôtômat không tất định cho trên Hình 1(b) là một máy như vậy.

Thứ ba, ta sẽ chứng tỏ rằng $\{a\}$ được chấp nhận bởi một ôtômat hữu hạn không tất định. Để làm điều đó, ta có thể dùng một máy có trạng thái xuất phát s_0 và trạng thái kết thúc s_1. Ta có một chuyển dịch từ s_0 đến s_1 khi đầu vào là 1 và không có một chuyển dịch nào khác. Xâu duy nhất được chấp nhận bởi máy này là a. Máy này được cho trên Hình 1(c)



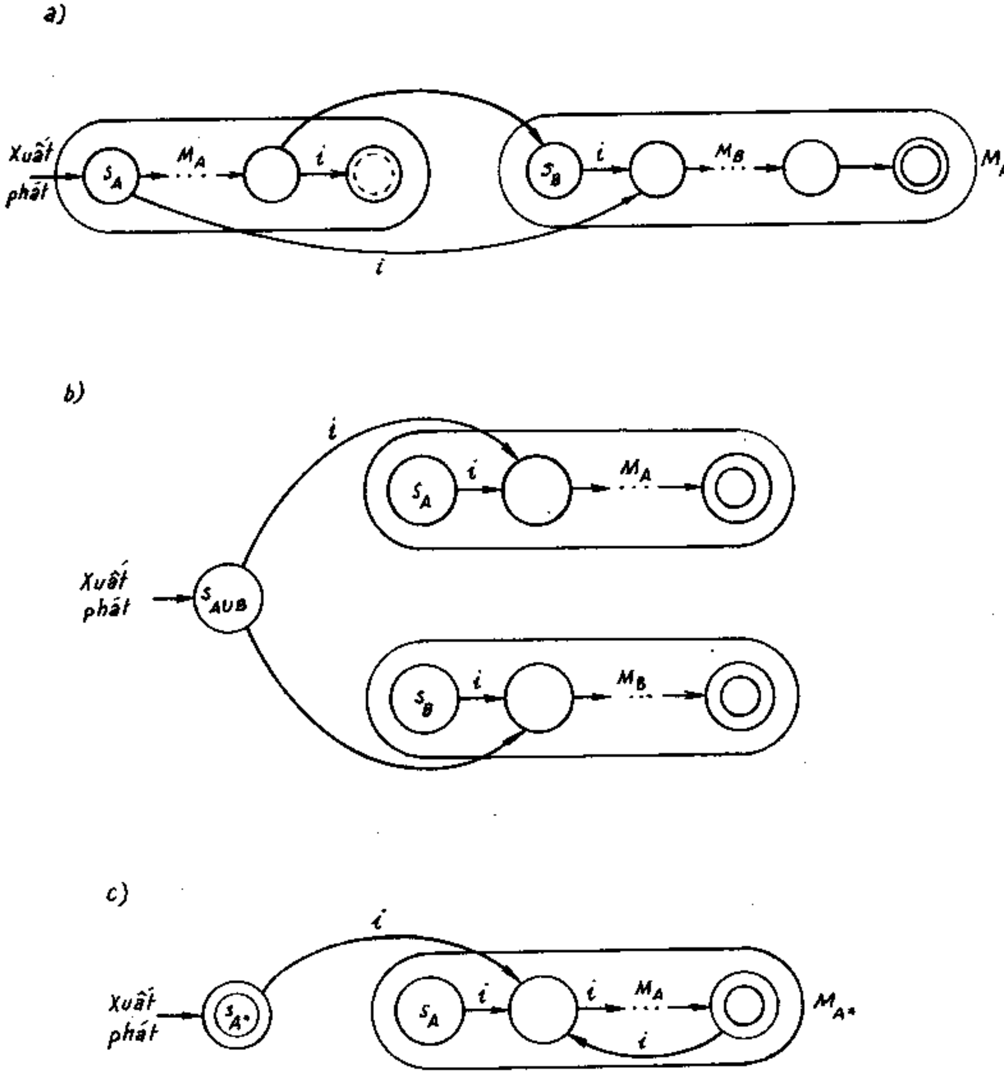
Hình 1. Ôtômat hữu hạn không tất định chấp nhận một số tập cơ sở.

Tiếp theo, ta sẽ chứng minh rằng AB và $A \cup B$ đều có thể được chấp nhận bởi các ôtômat hữu hạn nếu A và B là các ngôn ngữ đã được chấp nhận bởi các ôtômat hữu hạn. Giả sử A được chấp nhận bởi $M_A = (S_A, I, f_A, s_A, F_A)$ và B được chấp nhận bởi $M_B = (S_B, I, f_B, s_B, F_B)$

Chúng ta bắt đầu bằng việc dựng máy $M_{AB} = (S_{AB}, I, f_{AB}, s_{AB}, F_{AB})$ chấp nhận AB – ghép của A và B. Ta dựng máy này bằng tổ hợp các máy cho A và B nối tiếp với nhau, sao cho một xâu trong A đưa máy tổ hợp từ s_A – trạng thái xuất phát của M_A đến s_B – trạng thái xuất phát của M_B. Xâu trong B phải đưa máy tổ hợp này từ s_B đến một trạng thái kết thúc của máy tổ hợp. Do đó, ta phải tiến hành chế tạo như sau: cho $S_{AB} = S_A \cup S_B$. Trạng thái xuất phát s_{AB} như s_A. Tập các trạng thái kết thúc F_{AB} là tập các trạng thái kết thúc của M_B, có chứa cả s_{AB} nếu và chỉ nếu $\lambda \in A \cap B$. Các chuyển dịch trong M_{AB} bao gồm tất cả các chuyển dịch trong M_A và trong M_B, cũng như một số chuyển dịch mới. Đối với mỗi chuyển dịch trong M_A dẫn tới một trạng thái kết thúc, ta tạo được một chuyển dịch trong M_{AB} từ cùng trạng thái đó tới s_B với cùng một đầu vào. Theo cách đó, một xâu trong A đưa M_{AB} từ s_{AB} đến s_B, rồi sau đó một xâu trong B đưa s_B đến một trạng thái kết thúc của M_{AB}. Hơn nữa, đối với mỗi chuyển dịch từ s_B, chúng ta tạo được một dịch chuyển trong M_{AB} từ s_{AB} tới chính trạng thái đó. Hình 2(a) minh họa cho sự xây dựng vừa nói ở trên.

Bây giờ chúng ta sẽ dựng một máy $M_{A\cup B} = (S_{A\cup B}, I, f_{A\cup B}, s_{A\cup B}, F_{A\cup B})$ chấp nhận $A \cup B$. Ôtômat này có thể được xây dựng bằng cách tổ hợp M_A và M_B theo kiểu song song, có dùng một trạng thái xuất phát mới với những chuyển dịch mà cả s_A và s_B đều có. Cho $S_{A\cup B} = S_A \cup S_B \cup \{s_{A\cup B}\}$, ở đây $s_{A\cup B}$ là trạng thái mới và cũng là trạng thái xuất phát của $M_{A\cup B}$. Cho tập các trạng thái kết thúc $F_{A\cup B}$ bằng $F_A \cup F_B \cup \{s_{A\cup B}\}$ nếu $\lambda \in A \cup B$, và bằng $F_A \cup F_B$ trong trường hợp ngược lại. Các chuyển dịch trong $M_{A\cup B}$ bao gồm tất cả các chuyển dịch trong M_A và trong M_B. Đối với mỗi chuyển dịch từ s_A đến một trạng thái s dưới tác dụng của đầu vào i, ta cũng kể là một chuyển dịch từ $s_{A\cup B}$ đến s cũng dưới tác dụng của đầu vào i và đối với mỗi chuyển dịch từ s_B đến s với đầu vào i ta cũng kể là một chuyển dịch từ $s_{A\cup B}$ đến s với đầu vào i. Theo cách đó, một xâu trong A dẫn từ $s_{A\cup B}$ tới một trạng thái kết thúc trong máy mới và một xâu trong B cũng dẫn từ $s_{A\cup B}$ đến một trạng thái kết thúc trong máy mới. Hình 2(h) minh họa sự xây dựng máy $M_{A\cup B}$ vừa nói ở trên.

Cuối cùng, chúng ta sẽ xây dựng $M_{A^*} = (S_{A^*}, I, f_{A^*}, s_{A^*}, F_{A^*})$ – *máy chấp nhận* A^*, tức bao đóng Kleene của A. Cho S_{A^*} bao gồm tất cả các trạng thái của A và một trạng thái thêm s_{A^*} là trạng thái xuất phát của máy mới. Tập các trạng thái kết thúc F_{A^*} bao gồm tất cả các trạng thái trong F_A cũng như trạng thái xuất phát s_{A^*}, vì λ cần phải được chấp nhận. Để chấp nhận ghép của một số tùy ý các xâu thuộc A, chúng ta sẽ đưa vào tất cả các chuyển dịch trong M_A cũng như các chuyển dịch từ s_{A^*} khớp với các chuyển dịch từ s_A và các chuyển dịch từ mỗi trạng thái kết thúc khớp với các chuyển dịch từ s_A. Với tập hợp các chuyển dịch đó, một xâu được tạo bởi phép ghép các xâu trong A sẽ đưa s_{A*} tới một trạng thái kết thúc khi xâu thứ hai trong A đã được đọc xong và v.v... Hình 2(c) minh họa việc xây dựng mà ta nói ở trên.



Hình 2. Xây dựng các ôtômat chấp nhận các phép ghép, hợp và bao đóng Kleene.

Có thể xây dựng được một ôtômat hữu hạn không tất định cho một tập chính quy bất kỳ bằng cách dùng thủ tục mô tả trong chứng minh trên. Chúng ta sẽ minh họa điều này bằng ví dụ sau:

Ví dụ 2. Dựng một ôtômat hữu hạn không tất định chấp nhận tập chính quy $\mathbf{1}^*\cup\ \mathbf{01}$.

Giải: Ta bắt đầu bằng việc xây dựng một máy chấp nhận $\mathbf{1}^*$. Điều này được làm bằng cách xây dựng máy chấp nhận **1** rồi sau đó dùng cách dựng cho M_{A^*} đã được mô tả trong chứng minh Định lý 1. Tiếp theo ta dựng máy chấp nhận **01** bằng cách dùng các máy chấp nhận **0** và **1** và cách dựng M_{AB} trong chứng minh Định lý 1. Cuối cùng, dùng cách dựng $M_{A\cup B}$ được dùng trong chứng minh đó, chúng ta sẽ dựng được máy chấp nhận $\mathbf{1}^*\cup\ \mathbf{01}$. Các ôtômat hữu hạn được dùng để dựng máy này được cho trên Hình 3. Những trạng thái trong các máy kế tiếp được đánh dấu bằng cách dùng các chỉ số dưới khác nhau, thậm chí khi một trạng thái được tạo thành từ một trạng thái đã được dùng trước đó trong một máy khác. Chú ý rằng cách xây dựng nói trên không tạo ra máy đơn giản nhất chấp nhận $\mathbf{1}^*\cup\ \mathbf{01}$. Máy đơn giản hơn chấp nhận tập này được cho trên Hình 3(b).

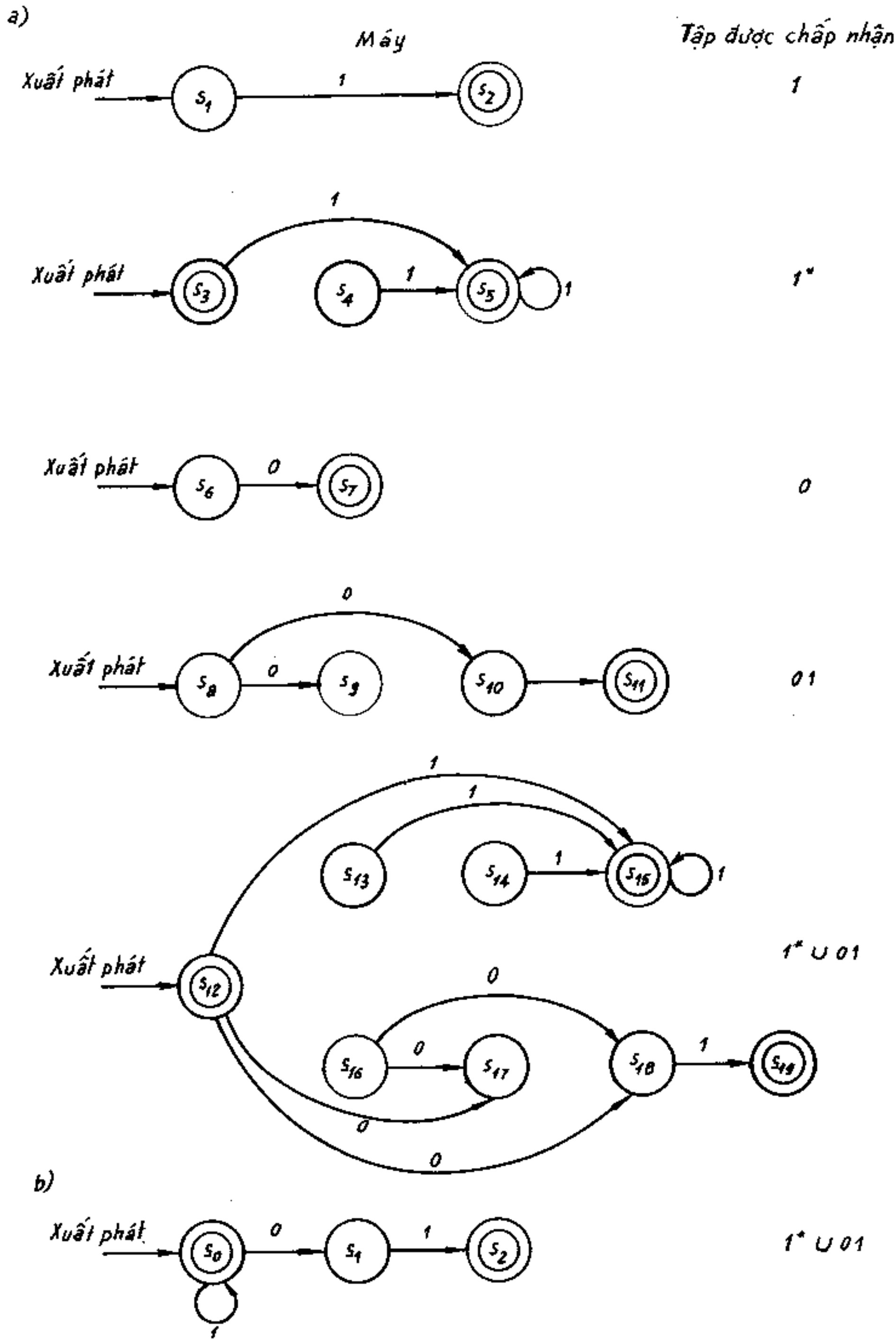
■

TẬP HỢP CHÍNH QUY VÀ VĂN PHẠM CHÍNH QUY

Trong Tiết 10.1 ta đã giới thiệu các văn phạm cấu trúc câu và định nghĩa các loại văn phạm khác nhau. Đặc biệt, ta đã định nghĩa văn phạm chính quy hay văn phạm loại 3, là văn phạm có dạng $G = (V, T, S, P)$ với mỗi sản xuất có dạng $S \to \lambda$, $A \to a$ hay $A \to aB$, trong đó a là ký hiệu kết thúc, A và B là các ký hiệu không kết thúc. Như tên gọi đã gợi ý, giữa các văn phạm chính quy và các tập (hợp) chính quy có một mối quan hệ khăng khít.

ĐỊNH LÝ 2. Một tập sinh bởi một văn phạm chính quy nếu và chỉ nếu nó là một tập chính quy.

Chứng minh. Trước hết, ta chứng minh rằng một tập được sinh bởi một



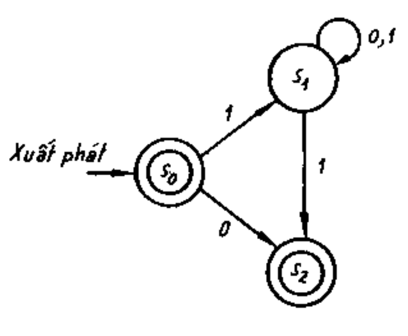
Hình 3. Các ôtômat hữu hạn không tất định chấp nhận $1^* \cup 01$.

văn phạm chính quy là một tập chính quy. Giả sử rằng $G = (V, T, S, P)$ là một văn phạm chính quy sinh ra tập $L(G)$. Để chứng minh $L(G)$ là chính quy ta sẽ xây dựng một máy hữu hạn trạng thái không tất định $M = (S, I, f, s_0, F)$ chấp nhận $L(G)$. Giả sử S, tập hợp các trạng thái, có chứa trạng thái s_A đối với mỗi ký hiệu không kết thúc A của G và trạng thái phụ s_F là một trạng thái kết thúc. Trạng thái xuất phát s_0 được lập từ ký hiệu xuất phát S. Các chuyển dịch trong M được tạo từ các sản xuất của G theo cách sau. Chuyển dịch từ s_A đến s_F ứng với đầu vào a sẽ được đưa vào nếu $A \to a$ là một sản xuất và chuyển dịch từ s_A đến s_B ứng với đầu vào a sẽ được đưa vào nếu $A \to aB$ là một sản xuất. Tập các trạng thái kết thúc bao gồm s_F và cũng bao gồm cả s_0 nếu $S \to \lambda$ là một sản xuất trong G. Không mấy khó khăn có thể chứng tỏ được rằng ngôn ngữ được chấp nhận bởi M bằng ngôn ngữ được sinh bởi văn phạm G, tức là $L(M) = L(G)$. Điều này có thể được làm bằng cách xác định các từ dẫn tới một trạng thái kết thúc. Các chi tiết xin dành cho độc giả, xem như một bài tập.

□

Trước khi chứng minh phần đảo lại, ta sẽ minh họa cách xây dựng một máy không tất định chấp nhận cùng một tập như một văn phạm chính quy.

Ví dụ 3. Dựng một ôtômat hữu hạn không tất định chấp nhận ngôn ngữ được sinh bởi văn phạm chính quy $G = (V, T, S, P)$ trong đó $V = \{0, 1, A, S\}$, $T = \{0, 1\}$ và các sản xuất trong P là $S \to 1A$, $S \to 0$, $S \to \lambda$, $A \to 0A$, $A \to 1A$ và $A \to 1$.



Hình 4. Ôtômat hữu hạn không tất định chấp nhận L(G).

Giải: Giản đồ trạng thái của ôtômat không tất định chấp nhận $L(G)$ được cho trên Hình 4. Ôtômat này được xây dựng theo thủ tục được mô tả trong phần chứng minh ở trên. Trong ôtômat này s_0 là trạng thái tương ứng với S, s_1 là trạng thái tương ứng với A và s_2 là trạng thái kết thúc.

■

Bây giờ ta sẽ hoàn tất việc chứng minh Định lý 2.

Chứng minh Ta sẽ chứng minh rằng nếu một tập là chính quy, thì tồn tại một văn phạm chính quy sinh ra tập chính quy đó. Giả sử M là một máy hữu hạn trạng thái chấp nhận tập đó với tính chất là s_0 – trạng thái xuất phát của M – không bao giờ là trạng thái kế tiếp đối với một chuyển dịch. (Ta sẽ tìm được máy có tính chất này ở Bài tập 14). Văn phạm $G = (V, T, S, P)$ được xác định như sau: Tập V các ký hiệu của G được lập bằng cách gán một ký hiệu cho mỗi trạng thái của S và mỗi ký hiệu đầu vào trong I. Ký hiệu xuất phát S là ký hiệu được lập từ trạng thái xuất phát s_0. Tập P các sản xuất trong G được lập từ các chuyển dịch trong M. Đặc biệt, nếu trạng thái s chuyển tới một trạng thái kết thúc khi đầu vào là s thì sản xuất $A_s \rightarrow a$ sẽ được bao hàm trong P, ở đây A_s là ký hiệu không kết thúc được tạo từ trạng thái s. Nếu trạng thái s chuyển tới trạng thái t khi đầu vào là a thì sản xuất $A_s \rightarrow aA_t$ sẽ được bao hàm trong P. Sản xuất $S \rightarrow \lambda$ được bao hàm trong P nếu và chỉ nếu $\lambda \in L(M)$. Vì các sản xuất của G tương ứng với các chuyển dịch trong M và các sản xuất dẫn tới các ký hiệu kết thúc tương ứng với các chuyển dịch đến trạng thái kết thúc, nên không khó khăn gì ta có thể chứng minh được rằng $L(G) = L(M)$. Các chi tiết xin dành cho bạn đọc như một bài tập.

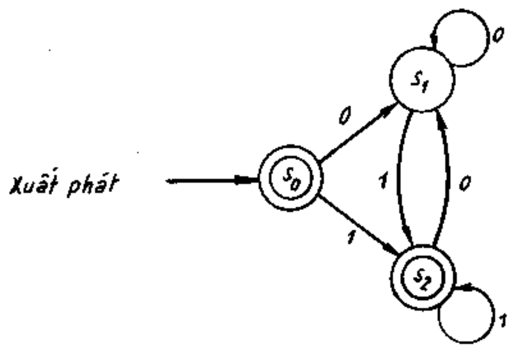
□

Ví dụ sau minh họa cách xây dựng một văn phạm từ một ôtômat chấp nhận ngôn ngữ sinh bởi văn phạm đó.

Ví dụ 4. Tìm một văn phạm chính quy sinh tập chính quy được chấp nhận bởi ôtômat cho trên Hình 5.

Giải: Văn phạm $G = (V, T, S, P)$ sinh ra tập được chấp nhận bởi ôtômat đó, với $V = \{S, A, B, 0, 1\}$ trong đó S, A, B, tương ứng với các trạng thái s_0, s_1 và s_2, $T = \{0, 1\}$, S là ký hiệu xuất phát và các sản xuất là $S \rightarrow 0A$, $S \rightarrow 1B$, $S \rightarrow 1$, $S \rightarrow \lambda$, $A \rightarrow 0A$, $A \rightarrow 1B$, $A \rightarrow 1$, $B \rightarrow 0A$, $B \rightarrow 1B$ và $B \rightarrow 1$.

■



Hình 5. Một ôtômat hữu hạn.

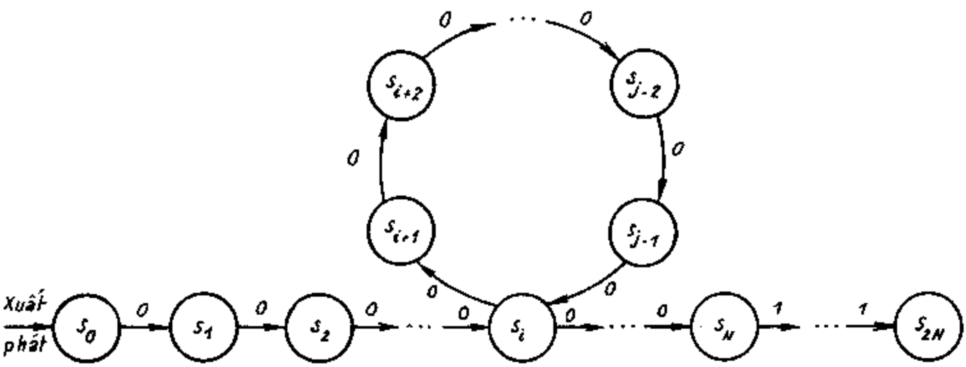
TẬP KHÔNG ĐƯỢC CHẤP NHẬN BỞI MỘT ÔTÔMAT HỮU HẠN

Chúng ta đã thấy rằng một tập được chấp nhận bởi một ôtômat hữu hạn nếu và chỉ nếu nó là một tập chính quy. Bây giờ ta sẽ thấy rằng có những tập không phải là chính quy và ta sẽ cho một ví dụ về tập như vậy. Kỹ thuật được dùng dưới đây để mô tả một tập không phải là chính quy là một phương pháp quan trọng để chỉ ra một số tập không phải là chính quy.

Ví dụ 5. Chứng tỏ rằng tập $\{0^n1^n \mid n = 0, 1, 2...\}$ tạo bởi tất cả các xâu gồm một khối các số 0 tiếp sau bởi một khối các số 1 với số lượng như nhau là một tập không chính quy.

Giải: Giả sử tập này là chính quy. Khi đó sẽ có một ôtômat hữu hạn tất định $M = (S, I, f, s_0, F)$ chấp nhận tập đó. Giả sử N là số trạng thái trong M, tức là $N = |S|$. Vì M chấp nhận tất cả các xâu tạo bởi một số các số 0 và tiếp theo bởi một số như thế các số 1, nên M phải chấp nhận 0^N1^N. Giả sử $s_0, s_1, \ldots, s_{2N}$ là dãy các trạng thái nhận được bắt đầu từ s_0 và dùng các ký hiệu 0^N1^N như một đầu vào, sao cho $s_1 = f(s_0, 0)$, $s_2 = f(s_1, 0), \ldots, s_N = f(s_{N-1}, 0)$, $s_{N+1} = f(s_N, 1), \ldots, s_{2N} = f(s_{2N-1}, 1)$. Chú ý rằng s_{2N} là trạng thái kết thúc.

Vì chỉ có N trạng thái, nên theo nguyên lý Dirichlet, ít nhất có hai trạng thái trong số $N+1$ trạng thái đầu tiên $s_0, \ldots, s_N$ phải như nhau. Ví dụ, s_i và s_j là hai trạng thái đồng nhất đó, với $0 \leq i < j \leq N$. Điều này có nghĩa là $f(s_i, 0^t) = s_j$ với $t = j - i$. Từ đây suy ra rằng vòng dẫn từ s_i quay lại chính nó nhận được bằng cách dùng đầu vào 0 tổng cộng t lần, như được biểu diễn bởi giản đồ trạng thái trên Hình 6.



Hình 6. Đường đi tạo bởi $0^N 1^N$.

Bây giờ ta xét xâu đầu vào $0^N 0^t 1^N = 0^{N+t} 1^N$. Đoạn đầu của xâu này có t số 0 liên tiếp nhiều hơn số các số 1 tiếp sau. Vì xâu này không có dạng $0^n 1^n$ (do có số số 0 nhiều hơn số số 1) nên nó không được chấp nhận bởi M. Do đó, $f(s_0, 0^{N+t} 1^N)$ không phải là trạng thái kết thúc. Tuy nhiên, khi ta dùng xâu $0^{N+t} 1^N$ như một đầu vào ta sẽ kết thúc ở chính trạng thái s_{2N} như trước. Đó là bởi vì t số 0 thừa trong xâu đó sẽ đưa chúng ta đi quanh vòng từ s_i trở lại chính nó thêm một lần nữa như cho thấy trên Hình 6. Sau đó phần còn lại của xâu sẽ đưa chúng ta tới đúng trạng thái kết thúc như trước. Mâu thuẫn này chứng tỏ $\{0^n 1^n \mid n = 0, 1, 2\}$ không phải là tập chính quy.

■

CÁC LOẠI MÁY MẠNH HƠN

Các ôtômat hữu hạn không thể thực hiện được nhiều tính toán. Hạn chế chủ yếu của các máy này là lượng nhớ hữu hạn của chúng. Chính điều này đã ngăn trở nó chấp nhận các ngôn ngữ không chính quy, như $\{0^n1^n \mid n = 0, 1, 2...\}$. Tuy nhiên, có một văn phạm phi ngữ cảnh chấp nhận tập này. Một văn phạm như vậy đã được cho trong Ví dụ 5 ở Tiết 10.1.

Vì những hạn chế của các ôtômat hữu hạn, nên cần phải dùng các mô hình tính toán khác mạnh hơn. Một mô hình như vậy là **ôtômat đẩy xuống** (push down). Một ôtômat đẩy xuống là một ôtômat hữu hạn kèm theo một băng đẩy xuống tạo một bộ nhớ không hạn chế. Các ký hiệu có thể được đặt vào hoặc lấy ra ở đỉnh của băng đó. Một tập sẽ được một ôtômat đẩy xuống chấp nhận theo hai cách. Thứ nhất, tập sẽ được chấp nhận nếu nó gồm tất cả các xâu tạo ra một băng trống khi các xâu đó được dùng làm đầu vào. Thứ hai, tập sẽ được chấp nhận nếu nó gồm tất cả các xâu dẫn tới một trạng thái kết thúc khi các xâu đó được dùng làm đầu vào. Người ta có thể chứng minh được rằng một tập được chấp nhận bởi một ôtômat đẩy xuống nếu và chỉ nếu nó là ngôn ngữ được sinh bởi một văn phạm phi ngữ cảnh.

Tuy nhiên, có những tập không thể biểu diễn như một ngôn ngữ được sinh bởi một văn phạm phi ngữ cảnh. Một trong những tập như vậy là $\{0^n1^n2^n \mid n = 0, 1, 2...\}$. Ta sẽ chỉ ra tại sao tập này không thể được chấp nhận bởi một ôtômat đẩy xuống, nhưng sẽ không chứng minh vì chúng ta không có đủ các công cụ cần thiết. (Tuy nhiên, một phương pháp chứng minh sẽ được cho trong Bài tập 28 của phần Bài tập bổ sung ở cuối chương này). Băng có thể được dùng để cho thấy rằng xâu bắt đầu với dãy các số 0 được tiếp theo bởi số các số 1 đúng bằng thế bằng cách đặt một ký hiệu lên băng đối với mỗi số 0 (chừng nào chỉ các số 0 còn đang được đọc) và lấy đi một trong số các ký hiệu đó đối với mỗi số 1 (chừng nào chỉ những số 1 tiếp theo các số không còn đang được đọc). Nhưng một khi điều đó đã được làm, băng sẽ là trống và không còn cách nào để xác định xem trong xâu có số các số 2 đúng bằng số các số 0 hay không.

Có những máy khác được gọi là **ôtômat tuyến tính giới nội**. Các máy này mạnh hơn các ôtômat đẩy xuống, chúng có thể chấp nhận các tập như $\{0^n1^n2^n \mid n = 0, 1, 2...\}$. Đặc biệt, các ôtômat tuyến tính giới nội có thể chấp nhận được cả các ngôn ngữ cảm ngữ cảnh. Tuy nhiên, các máy này lại không thể chấp nhận tất cả các ngôn ngữ được sinh bởi các văn phạm cấu trúc câu. Để tránh những hạn chế của các loại máy trên, người ta sử dụng một mô hình được gọi là **máy Turing**, theo tên nhà toán học Anh Alan Turing. Một máy Turing là một ôtômat hữu hạn có kèm theo một băng vô hạn về cả hai phía. Một máy Turing đọc và viết trên băng và nó có thể chuyển động tới lui dọc theo băng đó. Các máy Turing có thể chấp nhận tất cả các ngôn ngữ được sinh bởi các văn phạm cấu trúc câu. Thêm vào đó, các máy Turing có thể mô hình tất cả các tính toán có thể được thực hiện trên một máy tính. Do sức mạnh của mình, các máy Turing đã được nghiên cứu rộng rãi trong tin học lý thuyết. Chúng ta sẽ nghiên cứu một cách vắn tắt các máy này ở tiết sau.

BÀI TẬP

1. Mô tả bằng lời các xâu trong các tập chính quy sau:

a) $\mathbf{1^*0}$ b) $\mathbf{1^*00^*}$

c) $\mathbf{111 \cup 001}$ d) $(1 \cup 00)^*$

e) $(\mathbf{00^*1})^*$ f) $(0\cup1)(0\cup1)^*\mathbf{00}$

2. Xâu 1011 có thuộc các tập chính quy cho dưới đây không?

a) $\mathbf{10^*1^*}$ b) $\mathbf{0^*}(\mathbf{10}\cup\mathbf{11})^*$

c) $\mathbf{1(01)^*1^*}$ d) $\mathbf{1^*01}(0\cup1)$

e) $(\mathbf{10^*}(\mathbf{11})^*$ f) $\mathbf{1(00)^*(11)^*}$

g) $(\mathbf{10})^*\mathbf{1011}$ h) $(\mathbf{1}\cup\mathbf{00})(\mathbf{01}\cup\mathbf{0})\mathbf{1^*}$

3. Dùng các biểu thức chính quy biểu diễn các tập sau:

a) Tập các xâu có một hoặc nhiều hơn số 0 được tiếp sau bởi một số 1.

b) Tập các xâu có hai hoặc nhiều hơn ký hiệu được tiếp sau bởi ba hoặc nhiều hơn số 0.

c) Tập các xâu hoặc không có số 1 nào đứng trước một số 0 hoặc không có số 0 nào đứng trước một số 1.

d) Tập các xâu chứa một xâu các số 1 sao cho số các số 1 bằng 2 mođun 3, được tiếp theo bởi một số chẵn các số 0.

4. Dựng một ôtômat hữu hạn tất định chấp nhận các tập sau từ I^* với I là một bộ chữ cái.

a) $\varnothing$ b) $\{\lambda\}$ c) $\{a\}$ với $a \in I$.

*5. Chứng minh rằng nếu A là một tập chính quy, thì tập tất cả các xâu lộn ngược A^R của các xâu trong A cũng sẽ là chính quy.

6. Tìm một ôtômat hữu hạn chấp nhận

a) $\{\lambda, 0\}$ b) $\{0, 11\}$ c) $\{0, 11, 000\}$

7. Dùng các cách dựng được cho trong phần chứng minh của Định lý Kleene, tìm một ôtômat hữu hạn không tất định chấp nhận các tập sau:

a) $\mathbf{0^*1^*}$ b) $(\mathbf{0} \cup \mathbf{11})^*$ c) $\mathbf{01^*} \cup \mathbf{00^*1}$

8. Dựng một ôtômat hữu hạn tất định chấp nhận các ngôn ngữ được sinh bởi văn phạm chính quy $G = (V, T, S, P)$ trong đó $V = \{0, 1, S, A, B\}$, $T = \{0, 1\}$, S là ký hiệu xuất phát, và tập P các sản xuất là:

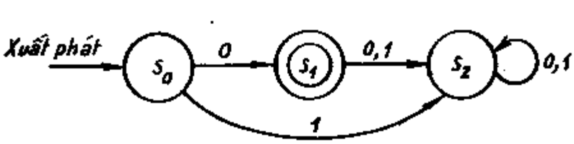
a) $S \to 0A$, $S \to 1B$, $A \to 0$, $B \to 0$

b) $S \to 1A$, $S \to 0$, $S \to \lambda$, $A \to 0B$, $B \to 1B$, $B \to 1$

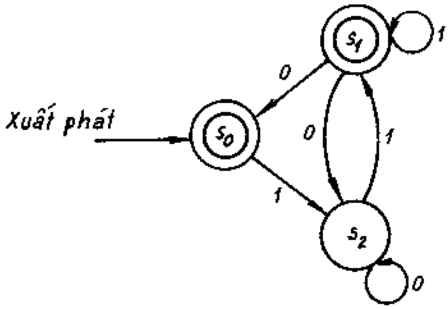
c) $S \to 1B$, $S \to 0$, $A \to 1A$, $A \to 0B$, $A \to 1$, $A \to 0$, $B \to 1$

Trong các Bài tập từ 9-11, hãy xây dựng một văn phạm $G = (V, T, S, P)$ sinh ra ngôn ngữ được chấp nhận bởi máy hữu hạn trạng thái đã cho.

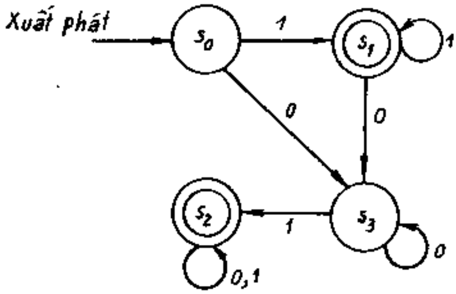
9.



10.



11.



12. Chứng tỏ rằng một ôtômat hữu hạn được xây dựng từ một văn phạm chính quy trong phần chứng minh Định lý 2 chấp nhận tập được sinh bởi văn phạm đó.

13. Chứng minh rằng văn phạm chính quy được xây dựng từ một ôtômat hữu hạn trong phần chứng minh của Định lý 2 sẽ sinh ra tập được chấp nhận bởi ôtômat đó.

14. Chứng tỏ rằng mỗi một ôtômat hữu hạn tất định đều tương đương với một ôtômat khác có tính chất là trạng thái xuất phát của nó không bao giờ được tái ngộ.

***15.** Cho $M = (S, I, f, s_0, F)$ là một ôtômat hữu hạn tất định. Chứng tỏ rằng ngôn ngữ được chấp nhận bởi M, tức $L(M)$, là vô hạn nếu và chỉ nếu có một từ x được chấp nhận bởi M có $l(x) \geq |S|$

***16.** Một kỹ thuật quan trọng để chứng minh một tập nào đó là không chính quy được gọi là **Bổ đề bơm**. Bổ đề bơm phát biểu rằng nếu $M = (S, I, f, s_0, F)$ là một ôtômat hữu hạn tất định và nếu x là một xâu thuộc $L(M)$ – ngôn ngữ được chấp nhận bởi M – với $l(x) \geq |S|$, thì có các xâu u, v, và w trong I^* sao cho $x = uvw$, $l(uv) \leq |S|$ và $l(v) \geq 1$ và $uv^iw \in L(M)$ với $i = 0, 1, 2...$ Hãy chứng minh Bổ đề bơm. (*Gợi ý*: dùng chính ý tưởng được dùng trong Ví dụ 5).

***17.** Chứng tỏ rằng tập $\{0^{2n}1^n\}$ là không chính quy. Bạn có thể dùng Bổ đề bơm cho trong Bài tập 16.

***18.** Chứng tỏ rằng tập $\{1^{n^2} \mid n = 0, 1, 2...\}$ là không chính quy. Bạn có thể dùng Bổ đề bơm cho trong Bài tập 16.

***19.** Chứng tỏ rằng tập các xâu thuận nghịch đọc trên tập $\{0, 1\}$ là không chính quy. Bạn có thể dùng Bổ đề bơm cho trong Bài tập 16. (*Gợi ý:* Xét các xâu có dạng 0^N10^N).

****20.** Chứng tỏ rằng tập được chấp nhận bởi một ôtômat hữu hạn là một tập chính quy. (Đây là phần *nếu* của Định lý Kleene).

10.5. MÁY TURING

MỞ ĐẦU

Các ôtômat hữu hạn được nghiên cứu ở trên không thể được dùng như các mô hình tính toán tổng quát. Những điều mà các máy đó làm được còn hạn chế. Chẳng hạn, các ôtômat hữu hạn có thể chấp nhận được các tập chính quy, nhưng lại không thể chấp nhận được nhiều tập dễ mô tả, kể cả tập $\{0^n1^n \mid n \geq 0\}$ mà các máy tính chấp nhận khi dùng bộ nhớ. Ta có thể dùng các ôtômat hữu hạn để tính các hàm tương đối đơn giản như tổng của hai số, nhưng ta không thể dùng chúng để tính các hàm mà các máy tính có thể, chẳng hạn như tích của hàm số. Để khắc phục những thiếu sót đó, chúng ta có thể dùng một loại máy mạnh hơn được biết là máy Turing, gọi theo tên nhà toán học và nhà khoa học về máy tính nổi tiếng Alan Turing, người đã phát minh ra nó vào những năm 1930.

Về cơ bản, máy Turing gồm một đơn vị điều khiển mà tại một bước bất kỳ đơn vị này ở một trong một số hữu hạn các trạng thái kết thúc khác nhau, và kèm theo một băng vô hạn ở hai phía và được chia thành các ô. Các máy Turing đọc và viết các ký hiệu trên băng khi đơn vị điều khiển chạy tới chạy lui dọc theo băng, làm thay đổi các trạng thái tùy thuộc vào ký hiệu được đọc trên băng. Các máy Turing mạnh hơn các máy hữu hạn trạng thái vì chúng có các năng lực nhớ mà các máy hữu hạn trạng thái không có. Chúng ta sẽ chứng tỏ có thể dùng các máy Turing để chấp nhận các tập như thế nào, kể cả các tập không được chấp nhận bởi các máy hữu hạn trạng thái. Chúng ta cũng sẽ cho thấy dùng máy Turing có thể tính được các hàm như thế nào. Các máy Turing là những mô hình tính toán tổng quát nhất, về căn bản nó có thể làm được tất cả những gì mà một máy tính có thể làm được.

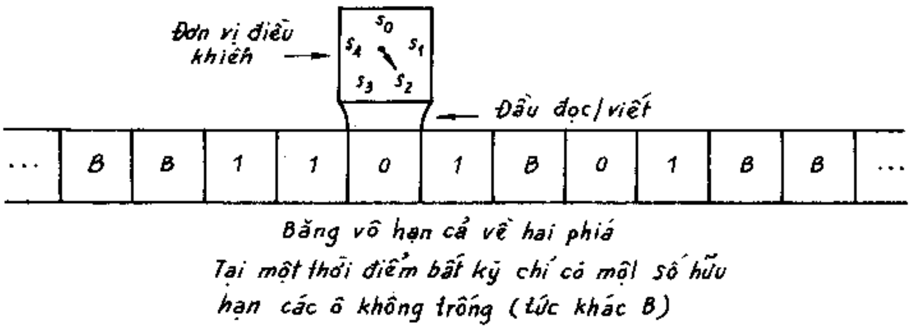
ĐỊNH NGHĨA MÁY TURING

Bây giờ ta có thể cho một định nghĩa hình thức của máy Turing. Sau đó, ta sẽ giải thích định nghĩa hình thức này có thể được diễn giải như thế nào qua đầu điều khiển, đầu có thể đọc và viết các ký hiệu trên băng và chuyển động hoặc sang phải hoặc sang trái dọc theo băng đó.

ĐỊNH NGHĨA 1. *Máy Turing* $T = (S, I, f, s_0)$ gồm một tập hữu hạn S các trạng thái, bộ chữ cái I chứa ký hiệu khoảng trống B, một hàm bộ phận f từ $S \times I$ đến $S \times I \times \{R, L\}$ (R là hướng phải, L là hướng trái – ND) và trạng thái xuất phát s_0.

Cần nhớ lại phần chú thích cho Bài tập 39 ở Tiết 1.6 trong đó có nói rằng một hàm bộ phận chỉ được xác định đối với các phần tử thuộc miền xác định của nó. Điều này có nghĩa là đối với một số cặp (trạng thái, ký hiệu) hàm f có thể không được xác định; nhưng đối với cặp mà f xác định thì có một bộ ba duy nhất (trạng thái, ký hiệu, hướng) liên kết với cặp đó.

Để giải thích định nghĩa này qua một máy, ta hãy xét đơn vị điều khiển và băng được chia thành các ô, vô hạn ở hai đầu, chỉ có một số hữu hạn các ký hiệu khác B (tức không trống) trên đó ở bất kỳ thời điểm đã cho nào, như được thấy trên Hình 1. Hành động của máy Turing ở mỗi bước hoạt động của nó tùy thuộc vào giá trị của hàm bộ phận f đối với trạng thái và ký hiệu trên băng hiện thời.



Hình 1. Biểu diễn một máy Turing.

Ở mỗi một bước, đơn vị điều khiển đọc ký hiệu x hiện thời trên băng. Nếu đơn vị điều khiển ở trạng thái s và nếu hàm bộ phận xác định đối với cặp (s, x) với $f(s, x) = (s', x', d)$ thì đơn vị điều khiển:

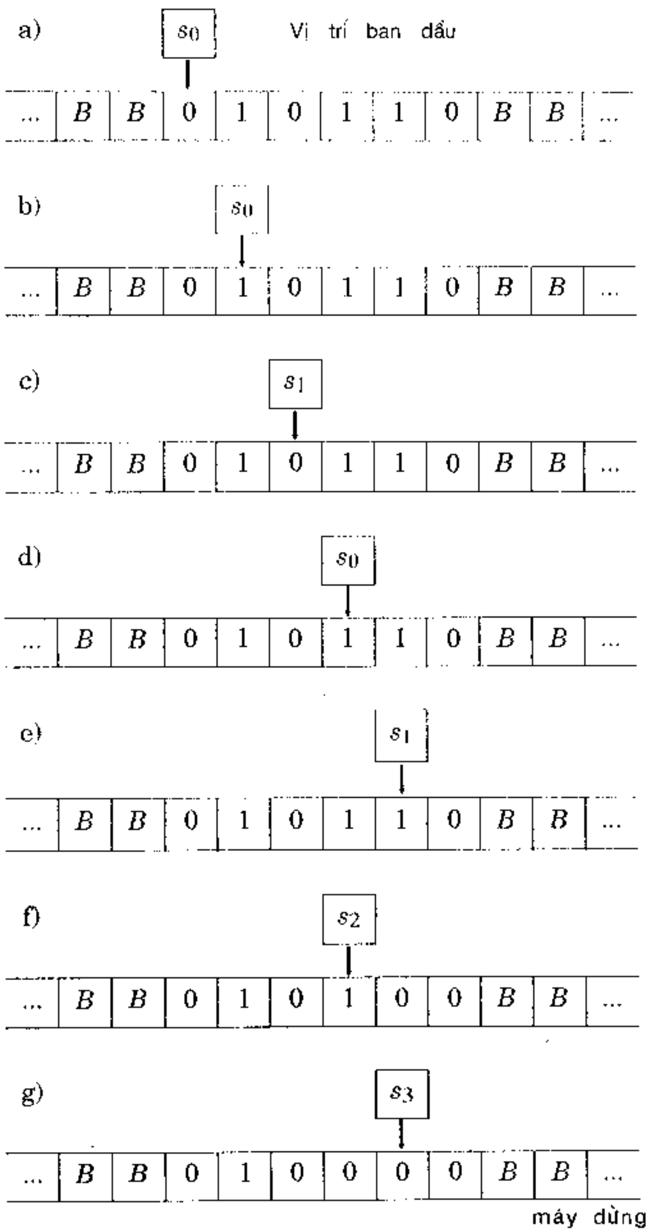
1. chuyển vào trạng thái s'
2. viết ký hiệu x' vào ô hiện thời sau khi đã xóa x, rồi
3. chuyển sang phải một ô nếu $d = R$ hoặc chuyển trạng trái một ô nếu $d = L$

Ta viết bước này như một bộ năm phần tử (s, x, s', x', d). Nếu hàm bộ phận f không xác định đối với cặp (s, x) thì máy Turing sẽ *dừng lại.*

Một cách thông thường để định nghĩa một máy Turing là đặc tả tập các bộ năm phần tử dạng (s, x, s', x', d). Khi đó tập các trạng thái và bộ chữ cái đầu vào đã ngầm được xác định.

Khi bắt đầu hoạt động, máy Turing được giả thiết là ở trạng thái ban đầu s_0 và được đặt hên trên ký hiệu khác B ở bên trái cùng của băng. Nếu băng là khoảng trống hoàn toàn, đầu điều khiển sẽ định vị ở trên một ô bất kỳ. Ta sẽ gọi vị trí của đầu điều khiển ở trên ký hiệu khác B ở bên trái cùng của băng là *vị trí ban đầu* của máy.

Ví dụ sau minh họa sự hoạt động của một máy Turing.



Hình 2. Các bước tạo bởi T chạy trên băng trong Hình 1.

Ví dụ 1. Xác định băng kết thúc khi máy Turing T được xác định bởi bảy bộ năm phân tử sau $(s_0, 0, s_0, 0, R)$, $(s_0, 1, s_1, 1, R)$, (s_0, B, s_3, B, R), $(s_1, 0, s_0, 0, R)$, $(s_1, 0, s_0, 0, R)$, $(s_1, 1, s_2, 0, L)$, (s_1, B, s_3, B, R), $(s_2, 1, s_3, 0, R)$ chạy trên băng như được cho trên Hình 2(a).

Giải: Ta bắt đầu hoạt động với T ở trạng thái s_0 và định vị ở trên ký hiệu khác B ở phía trái cùng của băng. Bước đầu tiên là dùng bộ năm phần tử $(s_0, 0, s_0, 0, R)$ đọc số 0 ở ô không trống bên trái cùng, dừng lại ở trạng thái s_0, viết một số 0 vào ô đó và dịch sang phải một ô. Bước thứ hai là dùng bộ năm $(s_0, 1, s_1, 1, R)$ đọc số 1 ở ô hiện thời, chuyển sang trạng thái s_1, viết số 1 vào ô đó và dịch sang phải một ô. Bước thứ ba là dùng bộ năm $(s_1, 0, s_0, 0, R)$ đọc số 0 ở ô hiện thời, chuyển sang trạng thái s_0, viết số 0 vào ô đó và dịch sang phải một ô. Bước thứ tư là dùng bộ năm $(s_0, 1, s_1, 1, R)$ đọc số 1 ở ô hiện thời, chuyển sang trạng thái s_1, viết số 1 vào ô đó và dịch sang phải một ô. Bước thứ năm là dùng bộ năm $(s_1, 1, s_2, 0, L)$ đọc số 1 ở ô hiện thời, chuyển sang trạng thái s_2, viết số 0 vào ô đó và dịch sang trái một ô. Bước thứ sáu là dùng bộ năm $(s_2, 1, s_3, 0, R)$ đọc số 1 ở ô hiện thời, chuyển sang trạng thái s_3, viết số 0 vào ô đó và dịch sang phải một ô. Cuối cùng, ở bước thứ bảy máy dừng lại vì không có bộ năm nào bắt đầu bằng cặp $(s_3, 0)$ trong đặc tả của máy. Các bước trên được minh họa trên Hình 2.

Chú ý rằng T đã làm thay đổi cặp số 1 liên tiếp đầu tiên trên băng thành các số 0 và sau đó dừng lại. ■

DÙNG MÁY TURING ĐỂ CHẤP NHẬN CÁC TẬP

Các máy Turing có thể được dùng để chấp nhận các tập. Để làm như vậy đòi hỏi ta phải định nghĩa khái niệm trạng thái kết thúc như sau. *Trạng thái kết thúc* của một máy Turing T là trạng thái không phải là trạng thái đầu tiên trong bất kỳ bộ năm phần tử nào trong đặc tả của T. (Ví dụ, trạng thái s_3 trong Ví dụ 1).

Bây giờ chúng ta có thể định nghĩa sự chấp nhận một xâu bởi một máy Turing. Cho một xâu, ta viết các ký hiệu liên tiếp trong xâu đó vào các ô liên tiếp.

ĐỊNH NGHĨA 2. Cho V là tập con của bộ chữ cái I. Một máy Turing $T = (S, I, f, s_0)$ **chấp nhận** xâu x trong V^* nếu và chỉ nếu T xuất phát từ vị trí ban đầu khi x đã được viết trên băng sẽ dừng lại ở một trạng thái kết thúc. T được nói là chấp nhận một tập con A của V^* nếu và chỉ nếu với mọi x thuộc A, x được chấp nhận bởi T.

Chú ý rằng để chấp nhận một tập con A của V^*, ta có thể dùng các ký hiệu không chứa trong V. Điều này có nghĩa là bảng chữ cái đầu vào I có thể chứa các ký hiệu không chứa trong V. Các ký hiệu dư này thường được dùng làm phần tử đánh dấu (xem Ví dụ 3).

Khi nào một máy Turing T không chấp nhận một xâu x trong V^*? Câu trả lời ở đây là: x không được chấp nhận nếu T không dừng lại hoặc dừng lại không phải ở trạng thái kết thúc khi máy hoạt động trên băng có chứa các ký hiệu của x trong các ô liên tiếp và xuất phát từ vị trí ban đầu. (Độc giả nên hiểu rằng đây chỉ là một trong nhiều cách khả dĩ để định nghĩa sự chấp nhận các tập bởi các máy Turing).

Ta sẽ minh họa khái niệm này bằng ví dụ dưới đây.

Ví dụ 2. Tìm một máy Turing chấp nhận tập các xâu nhị phân có số 1 là bit thứ hai của chúng. (Tức là tập chính quy $(\mathbf{0} \vee \mathbf{1})1(\mathbf{0} \vee \mathbf{1})^*$).

Giải: Ta muốn có một máy Turing xuất phát ở ô không trống bên trái cùng của băng, dịch chuyển sang phải và xác định ký hiệu thứ hai của xâu có phải là 1 hay không. Nếu ký hiệu thứ hai là 1, máy sẽ phải chuyển tới một trạng thái kết thúc. Nếu ký hiệu thứ hai không phải là 1, máy sẽ không dừng lại hoặc dừng lại không phải ở trạng thái kết thúc.

Để xây dựng một máy như vậy, ta đưa vào hai bộ năm phần tử $(s_0, \mathbf{0}, s_1, \mathbf{0}, R)$ và $(s_0, 1, s_1, 1, R)$ để đọc vào ký hiệu đầu tiên và đưa máy Turing vào trạng thái s_1. Tiếp theo, ta đưa vào hai bộ năm $(s_1, \mathbf{0}, s_2, 0, R)$ và $(s_1, 1, s_3, 1, R)$ để đọc vào ký hiệu thứ hai và chuyển tới trạng thái s_2 nếu ký hiệu đó là 0 hoặc chuyển tới trạng thái s_3 nếu ký hiệu đó là 1. Chúng ta không muốn chấp nhận xâu có số **0** là bit thứ hai, nên s_2 sẽ không được là trạng thái kết thúc. Trạng thái kết thúc mà ta muốn là s_3. Vì vậy, ta có thể đưa vào bộ năm $(s_2, 1, s_2, 0, R)$. Vì ta cũng không muốn chấp nhận xâu rỗng cũng như xâu chỉ có một bit, ta cũng sẽ đưa vào các bộ năm (s_0, B, s_2, 0, R) và (s_1, B, s_2, 0, R).

Máy Turing gồm bảy bộ năm cho ở trên sẽ kết thúc ở trạng thái kết thúc s_3 nếu và chỉ nếu xâu nhị phân đầu vào có ít nhất hai bit và bit thứ hai của nó là số 1. Nếu xâu nhị phân chứa ít hơn hai bit hoặc bit thứ hai không phải là 1, thì máy sẽ kết thúc ở trạng thái không kết thúc s_2.

■

Cho một tập chính quy, có thể xây dựng một máy Turing luôn dịch chuyển sang phải và chấp nhận tập này (như trong Ví dụ 2). Để xây dựng máy Turing như thế, trước hết phải tìm một ôtômat hữu hạn chấp nhận tập đó, rồi sau đó xây dựng máy Turing có hàm chuyển của ôtômat hữu hạn và luôn dịch chuyển sang phải.

Bây giờ chúng ta sẽ minh họa cách xây dựng một máy Turing chấp nhận một tập không chính quy.

Ví dụ 3. Tìm máy Turing chấp nhận tập $\{0^n1^n \mid n \geq 1\}$

Giải: Để xây dựng một máy như vậy, ta sẽ dùng một ký hiệu phụ M trên băng như một phần tử đánh dấu. Ta có $V = \{0, 1\}$ và $I = \{0, 1, M\}$. Ta chỉ muốn chấp nhận các xâu trong V^*. Ta sẽ có một trạng thái kết thúc là s_6. Máy Turing sẽ tuần tự thay số 0 ở vị trí trái cùng của xâu bằng M và thay số 1 ở vị trí phải cùng của xâu bằng M, khi quét tới quét lui và sẽ kết thúc ở một trạng thái cuối cùng nếu và chỉ nếu xâu gồm một khối các số 0 tiếp theo bởi một khối các số 1 với số lượng như nhau.

Mặc dù điều này dễ mô tả và cũng dễ thực hiện bởi một máy Turing, nhưng máy chúng ta cần dùng lại hơi phức tạp. Chúng ta dùng ký hiệu đánh dấu M để bám sát các ký hiệu trái cùng và phải cùng mà chúng ta đã xem xét. Các bộ năm mà chúng ta dùng ở đây là: (s_0, 0, s_1, M, R), (s_1, 0, s_1, 0, R), (s_1, 1, s_1, 1, R), (s_1, M, s_2, M, L), (s_1, B, s_2, B, L), (s_2, 1, s_3, M, L), (s_3, 1, s_3, 1, L), (s_3, 0, s_4, 0, L), (s_3, M, s_5, M, R), (s_4, 0, s_4, 0, L), (s_4, M, s_0, M, R), (s_5, M, s_6, M, R). Ví dụ, xâu 000111 sẽ liên tiếp trở thành: M00111, M0011M, MM011M, MM01MM, MMM1MM, $MMMMMM$ khi máy hoạt động cho tới khi dừng lại. (Chú ý rằng xâu này không thay đổi ở tất cả các bước hoạt động của máy Turing).

Chúng tôi sẽ dành cho độc giả việc giải thích các hành động của máy Turing và tại sao nó lại chấp nhận tập $\{0^n1^n \mid n \geq 1\}$ (Bài tập 13 ở cuối tiết này).

■

Người ta có thể chứng minh rằng một tập có thể được chấp nhận bởi một máy Turing nếu và chỉ nếu nó được sinh bởi một văn phạm loại 0 hay nói một cách khác nếu tập được sinh bởi một văn phạm cấu trúc câu. Chúng tôi sẽ không trình bày chứng minh của khẳng định đó.

TÍNH CÁC HÀM BẰNG MÁY TURING

Máy Turing có thể được xem như một máy tính có khả năng tìm được giá trị của các hàm bộ phận. Để thấy điều đó, ta giả sử rằng máy Turing T khi đã cho xâu đầu vào x sẽ dừng lại với xâu y ở trên băng của nó. Khi đó ta có thể định nghĩa $T(x) = y$. Miền xác định của T là tập các xâu làm cho T sẽ dừng lại. $T(x)$ sẽ không xác định nếu T không dừng lại khi x đã cho như một đầu vào. Việc xem một máy Turing như một máy tính được các giá trị của một hàm trên các xâu là rất hữu ích, nhưng làm thế nào dùng máy Turing để tính các hàm được xác định trên các số nguyên, trên các cặp số nguyên, trên bộ ba các số nguyên v.v...?

Để xem một máy Turing như một máy tính các hàm từ tệp các bộ k số nguyên không âm đến tập các số nguyên không âm (những hàm này được gọi là các **hàm lý thuyết số**) chúng ta cần phải có một cách để biểu diễn các bộ k số nguyên trên băng của máy. Muốn vậy, ta sẽ dùng **biểu diễn nhất phân của các số nguyên**. Cụ thể, ta sẽ biểu diễn số nguyên n bằng một xâu gồm $n + 1$ số 1, sao cho, ví dụ, 0 được biểu diễn bằng xâu 1, 5 được biểu diễn bởi xâu 111111. Để biểu diễn bộ **k** số nguyên (n_1, n_2,..., n_k), ta dùng một xâu gồm $n_1 + 1$ số 1, tiếp sau là một dấu sao (*), tiếp sau nữa là xâu gồm $n_2 + 1$ số 1, rồi lại đến một dấu sao, v.v... và cuối cùng kết thúc bằng xâu gồm $n_k + 1$ số 1. Ví dụ, để biểu diễn bộ bốn số nguyên (2, 0, 1, 3) ta dùng xâu 111*1*11*1111.

Bây giờ ta có thể xem một máy Turing T như một máy tính dãy các hàm lý thuyết số T, T^1,...T^k,... Các hàm T^k được định nghĩa như tác dụng của T lên các bộ k số nguyên dưới dạng biểu diễn nhất phân và được ngăn cách với nhau bởi các dấu sao.

Ví dụ 4. Xây dựng một máy Turing để cộng hai số nguyên

Giải: Ta cần phải dựng một máy Turing tính được hàm $f(n_1, n_2) = n_1 + n_2$. Cặp (n_1, n_2) được biểu diễn bằng xâu gồm n_1+1 số 1 tiếp sau là một dấu sao, rồi tiếp sau nữa là xâu gồm $n_2 + 1$ số 1. Máy T cần phải nhận xâu này như đầu vào và tạo đầu ra trên băng là xâu gồm $n_1 + n_2 + 1$ con số 1. Có một cách để làm điều này như sau. Máy xuất phát ở số 1 phía trái cùng của xâu đầu vào và thực hiện các bước để xóa số 1 này (máy sẽ dừng nếu $n_1 = 0$ sao cho không còn số 1 nào trước dấu sao nữa) rồi thay dấu sao bằng số 1 trái cùng còn lại và sau đó dừng lại. Ta có thể dùng các bộ năm phần tử sau để làm điều đó: $(s_0, 1, s_1, B, R)$, $(s_1, *, s_3, B, R)$, $(s_1, 1, s_2, B, R)$, $(s_2, 1, s_2, B, R)$, $(s_2, *, s_3, 1, R)$

■

Không may, việc xây dựng các máy Turing để tính các hàm tương đối đơn giản lại có thể đòi hỏi quá đáng. Ví dụ, một máy Turing để nhân hai số nguyên không âm như được trình bày trong nhiều cuốn sách đòi hỏi tới 31 bộ năm phần tử và 11 trạng thái. Nếu việc xây dựng các hàm Turing để tính các hàm tương đối đơn giản mà đã đầy thách thức như vậy, thì hỏi liệu ta có hy vọng xây dựng các máy Turing để tính các hàm phức tạp hơn không? Một cách để đơn giản hóa bài toán là dùng một máy Turing đa băng, tức là máy dùng đồng thời hơn một băng và dựng các máy Turing đa băng cho hợp thành của các hàm. Người ta chứng minh được rằng đối với một máy Turing đa băng bất kỳ, luôn tồn tại một máy Turing một băng có thể làm được hệt như máy đa băng đó.

Một hàm có thể được tính bởi một máy Turing được gọi là hàm tính được. Không mấy khó khăn có thể chứng tỏ rằng có những hàm lý thuyết số là không tính được. Tuy nhiên, việc tạo ra một hàm như vậy lại không dễ dàng chút nào. Hàm được định nghĩa trong phần chú thích trước Bài tập 23 ở cuối tiết này là một ví dụ của hàm không tính được. Một cách để chứng minh hàm đó là không tính được là chứng tỏ rằng nó tăng nhanh hơn một hàm tính được bất kỳ (xem Bài tập 24).

CÁC LOẠI MÁY TURING KHÁC

Có nhiều biến tướng đối với định nghĩa của máy Turing. Ta có thể mở rộng các khả năng của máy đó bằng rất nhiều cách. Ví dụ, ta có thể cho nó ở mỗi bước chuyển dịch sang phải, sang trái hoặc không chuyển dịch hoàn toàn. Ta cũng có thể cho phép máy Turing hoạt động đồng thời với nhiều băng. Khi n băng được sử dụng, ta sẽ cần các bộ $(2 + 3n)$ phần tử để đặc tả máy Turing. Ta cũng lại có thể cho băng là hai chiều, trong đó ở mỗi bước ta có thể dịch lên, dịch xuống, sang phải hoặc sang trái chứ không phải chỉ dịch sang phải hoặc sang phải như đối với băng một chiều. Ta cũng có thể cho phép các đầu đọc đa băng đọc các ô khác nhau đồng thời. Hơn thế nữa, ta có thể cho phép máy Turing là không tất định, bằng cách cho phép cặp (trạng thái-ký hiệu trên băng) có thể xuất hiện như các phần tử trong hơn một bộ năm phần tử đặc tả máy Turing. Ta cũng có thể thu bớt các khả năng của máy Turing bằng các cách khác nhau. Chẳng hạn, ta có thể hạn chế băng là vô hạn chỉ theo một hướng hoặc hạn chế bộ chữ cái của băng chỉ có hai ký hiệu. Tất cả những biến tướng đó của máy Turing đều đã được nghiên cứu một cách chi tiết.

Một điểm quan trọng là ở chỗ dù ta có dùng biến tướng hay tổ hợp các biến tướng nào của máy Turing đi nữa, ta cũng không bao giờ làm tăng hoặc giảm sức mạnh của nó. Bất kỳ điều gì mà một biến tướng của máy Turing là được, thì chính máy Turing được định nghĩa trong tiết này cũng làm được và ngược lại. Sở dĩ một số biến tướng này đôi khi tỏ ra hữu ích chỉ bởi vì chúng làm cho việc thực hiện một công việc đặc biệt nào đó trở nên dễ dàng hơn so với máy Turing được định nghĩa theo Định nghĩa 1, chứ chúng không bao giờ làm tăng được sức mạnh của máy đó.

LUẬN ĐỀ CHURCH - TURING

Các máy Turing tương đối đơn giản. Chúng có thể chỉ có một số hữu hạn trạng thái và có thể chỉ đọc và viết một lần một ký hiệu trên băng một chiều. Tuy thế, nhưng hóa ra các máy Turing lại cực kỳ mạnh. Chúng ta đã thấy rằng các máy Turing có thể được tạo dựng để cộng và nhân các số. Mặc dù có thể là khó dựng được thực sự một máy Turing để tính một hàm đặc biệt nào đó có thể tính được nhờ một thuật toán, nhưng một máy Turing như vậy luôn luôn tìm được. Đây chính là mục đích ban đầu của Turing khi ông phát minh ra các máy này.

Hơn thế nữa có rất nhiều bằng chứng ủng hộ cho **luận đề Church - Turing**, luận đề phát biểu rằng: đối với một bài toán đã cho bất kỳ có thể giải được bằng một thuật toán hiệu quả, thì sẽ tồn tại một máy Turing có thể giải được bài toán đó. Lý do để phát biểu trên được gọi là một *luận đề* chứ không phải một định lý là ở chỗ khái niệm giải được bằng một thuật toán là một khái niệm không hình thức và không chính xác, đối lập với khái niệm giải được bằng máy Turing, một khái niệm hình thức và chính xác. Mặc dù một bài toán được giải trên máy tính bằng một chương trình được viết trong một ngôn ngữ nào đó, có thể sử dụng một lượng bộ nhớ không hạn chế, có lẽ cũng có thể được xem là một bài toán giải được một cách hiệu quả.

Nhiều lý thuyết khác nhau đã được phát triển để nắm bắt khái niệm tính được một cách hiệu quả. Đó là các lý thuyết của Turing, của Church, cũng như các lý thuyết được đề xuất bởi Kleene và Post. Các lý thuyết đó nhìn bề ngoài dường như rất khác nhau, nhưng một điều đáng ngạc nhiên là chúng lại được chứng tỏ là tương đương với nhau theo nghĩa chúng xác định chính xác cùng một lớp các hàm. Với bằng chứng đó, dường như những ý tưởng cội nguồn của Turing, được phát biểu trước khi phát minh ra các máy tính hiện đại, đã mô tả được những năng lực tối hậu của máy này.

BÀI TẬP

1. Cho T là máy Turing được xác định bởi các bộ năm phần tử $(s_0, 0, s_1, 1, R)$, $(s_0, 1, s_1, 0, R)$, $(s_0, B, s_1, 0, R)$, $(s_1, 0, s_2, 1, L)$, $(s_1, 1, s_1, 0, R)$ và $(s_1, B, s_2, 0, L)$. Đối với các băng ban đầu cho dưới đây, hãy xác định băng kết thúc khi T dừng lại. Giả sử rằng T xuất phát từ vị trí ban đầu.

a)

...	B	B	0	0	1	1	B	B	...

b)

...	B	B	1	0	1	B	B	B	...

c)

...	B	B	1	1	B	0	1	B	...

d)

...	B	B	B	B	B	B	B	B	...

2. Cũng hỏi như trên đối với máy Turing được xác định bởi các bộ năm phần tử $(s_0, 0, s_1, 0, R)$, $(s_0, 1, s_1, 0, L)$, $(s_0, B, s_1, 1, R)$, $(s_1, 0, s_2, 1, R)$, $(s_1, 1, s_1, 1, R)$, $(s_1, B, s_2, 0, R)$ và $(s_2, B, s_3, 0, R)$.

a)

...	B	B	0	1	0	1	B	B	...

b)

...	B	B	1	1	1	B	B	B	...

b)

...	B	B	0	0	B	0	0	B	...

d)

...	B	B	B	B	B	B	B	B	...

3. Máy Turing được đặc tả bởi các bộ năm phần tử $(s_0, 0, s_0, 0, R)$, $(s_0, 1, s_1, 0, R)$, (s_0, B, s_2, B, R), $(s_1, 0, s_1, 0, R)$, $(s_1, 1, s_0, 1, R)$, (s_1, B, s_2, B, R) sẽ làm gì khi cho đầu vào là một xâu nhị phân?

4. Cũng hỏi như trên đối với máy Turing được đặc tả bởi các bộ năm phần tử $(s_0, 0, s_1, B, R)$, $(s_0, 1, s_1, 1, R)$, $(s_1, 0, s_1, 0, R)$, $(s_1, 1, s_2, 1, R)$, $(s_2, 0, s_1, 0, R)$, $(s_2, 1, s_3, 0, L)$, $(s_3, 0, s_4, 0, R)$ và $(s_3, 1, s_4, 0, R)$?

5. Xây dựng một máy Turing với các ký hiệu trên băng là 0, 1 và B, biết rằng máy thay số 0 đầu tiên bằng số 1 và không làm thay đổi bất cứ một ký hiệu nào khác trên băng.

6. Xây dựng một máy Turing với các ký hiệu trên băng là 0, 1, và B. Biết rằng với đầu vào là một xâu nhị phân máy sẽ thay tất cả các số 0 trên băng bằng các số 1 và không làm thay đổi bất kỳ một số 1 nào trên băng.

7. Cũng cho và hỏi như trên nhưng bây giờ máy thay tất cả các số 1 trừ số 1 bên trái cùng trên bằng bằng các số 0 và không thay đổi các ký hiệu khác trên bằng.

8. Cũng hỏi và cho như trên nhưng bây giờ máy thay hai số 1 liên tiếp đầu tiên trên băng bằng các số 0 và không thay đổi các ký hiệu nào khác trên băng.

9. Dựng một máy Turing chấp nhận tập tất cả các xâu nhị phân kết thúc bằng một số 0.

10. Dựng một máy Turing chấp nhận tập tất cả các xâu nhị phân chứa ít nhất hai số 1.

11. Dựng một máy Turing chấp nhận tập tất cả các xâu nhị phân chứa một số chẵn con số 1.

12. Cho biết nội dung của băng ở mỗi bước của máy Turing cho trong Ví dụ 3 xuất phát từ các xâu sau:

a) 0011 b) 00011 c) 101100 d) 000111

13. Giải thích tại sao máy Turing trong Ví dụ 3 chấp nhận một xâu nhị phân nếu và chỉ nếu xâu đó có dạng 0^n1^n với n là một số nguyên dương.

*14. Dựng một máy Turing chấp nhận tập $\{0^{2n}1^n \mid n \geq 0\}$

*15. Dựng một máy Turing chấp nhận tập $\{0^n1^n2^n \mid n \geq 0\}$

16. Dựng một máy Turing tính hàm $f(n) = n + 2$ với mọi số nguyên không âm n.

17. Dựng một máy Turing tính hàm $f(n) = n - 3$ nếu $n \geq 3$ và $f(n) = 0$ nếu $n = 0, 1, 2$ đối với mọi số nguyên không âm n.

18. Dựng một máy Turing tính hàm $f(n) = n \textbf{ mod } 3$

19. Dựng máy Turing tính hàm $f(n) = 3$ nếu $n \geq 5$ và $f(n) = 0$ nếu $n = 0, 1, 2, 3$, hoặc 4.

20. Dựng máy Turing tính hàm $f(n_1, n_2) = n_2 + 2$ đối với mọi cặp số nguyên không âm n_1 và n_2.

***21.** Dựng máy Turing tính hàm $f(n_1, n_2) = \min(n_1, n_2)$ với mọi cặp số nguyên không âm n_1 và n_2.

22. Dựng một máy Turing tính hàm $f(n_1, n_2) = n_1 + n_2 + 1$ đối với mọi cặp số nguyên không âm n_1 và n_2.

Cho $B(n)$ là số cực đại các số 1 mà một máy Turing có n trạng thái và bộ chữ cái $\{1, B\}$ có thể in ra trên băng với băng ban đầu là trống hoàn toàn. Bài toán xác định $B(n)$ đối với các giá trị cụ thể của n đã được Tibor Rada nghiên cứu lần đầu tiên vào năm 1962. Hiện nay người ta đã biết rằng $B(2) = 4$, $B(3) = 6$ và $B(4) = 13$ nhưng $B(n)$ với $n \geqslant 5$ thì chưa ai biết.

***23.** Chứng tỏ rằng B(2) ít nhất là 4 bằng cách tìm một máy Turing có hai trạng thái và bộ chữ cái $\{1, B\}$, đồng thời máy sẽ dừng khi có bốn số 1 liên tiếp ở trên băng.

***24.** Chứng tỏ rằng hàm $B(n)$ không thể tính được bằng bất cứ máy Turing nào.

(*Gợi ý*: Giả sử có một máy Turing có thể tính được $B(n)$ dưới dạng nhị phân. Hãy xây dựng một máy Turing T, xuất phát với băng trống hoàn toàn, viết ra số n dưới dạng nhị phân, rồi tính $B(n)$ dưới dạng nhị phân, sau đó đổi $B(n)$ từ biểu diễn nhị phân sang biểu diễn nhất phân. Chứng tỏ rằng với n đủ lớn, số các trạng thái trong T sẽ nhỏ hơn $B(n)$ và dẫn tới mâu thuẫn).

CÂU HỎI ÔN TẬP

1. a) Nêu định nghĩa của văn phạm cấu trúc câu.

b) Một xâu được dẫn xuất từ xâu w bằng một văn phạm cấu trúc câu G là nghĩa thế nào?

2. a) Ngôn ngữ được sinh bởi một văn phạm cấu trúc câu G là gì?

b) Xác định ngôn ngữ được sinh bởi văn phạm G với từ vựng $\{S, 0, 1\}$, tập các ký hiệu kết thúc $T = \{0, 1\}$, ký hiệu xuất phát S và các sản xuất $S \to 000S$, $S \to 1$?

c) Xác định văn phạm cấu trúc câu sinh ra tập $\{01^n \mid n = 0, 1, 2...\}$

3. a) Nêu định nghĩa của văn phạm loại 1

b) Cho một ví dụ về văn phạm không phải loại 1

c) Nêu định nghĩa của văn phạm loại 2

d) Cho một ví dụ về một văn phạm không phải loại 2 nhưng là loại 1

e) Nêu định nghĩa của văn phạm loại 3 hay là văn phạm phi ngữ cảnh.

f) Cho một ví dụ về một văn phạm không phải loại 3 nhưng là loại 2.

4. a) Nêu định nghĩa của văn phạm chính quy

b) Nêu định nghĩa của ngôn ngữ chính quy

c) Cho một ví dụ về văn phạm không chính quy nhưng là văn phạm loại 3

d) Chứng minh rằng tập $\{0^m1^n \mid m, n = 0, 1, 2...\}$ là một ngôn ngữ chính quy

5. a) Dạng Backus – Naur là gì?

b) Cho một ví dụ về dạng Backus – Naur của văn phạm đối với một tập con tiếng Anh mà bạn tùy chọn.

6. a) Máy hữu hạn trạng thái là gì?

b) Cho biết làm thế nào dùng một máy hữu hạn trạng thái có thể lập mô hình một máy bán hàng chỉ nhận các đồng 25 xu và trả cho khách hàng một cốc nước giải khát sau khi đã bỏ 75 xu vào máy.

7. a) Bao đóng Kleene của tập các xâu là gì?

b) Tìm bao đóng Kleene của tập {11, 0}

8. a) Nêu định nghĩa của một ôtômat hữu hạn

b) Một xâu được chấp nhận bởi một ôtômat hữu hạn có nghĩa là gì?

9. a) Nêu định nghĩa của ôtômat hữu hạn không tất định

b) Chứng tỏ rằng với một ôtômat hữu hạn không tất định đã cho, tồn tại một ôtômat hữu hạn tất định chấp nhận cùng một ngôn ngữ.

10. a) Nêu định nghĩa của tập các biểu thức chính quy trên một tập I.

b) Hãy cho biết các biểu thức chính quy được dùng để biểu diễn các tập chính quy như thế nào?

11. Hãy phát biểu định lý Kleene

12. Chứng minh rằng một tập được sinh bởi một văn phạm chính quy nếu và chỉ nếu nó là một tập chính quy

13. Cho một ví dụ về một tập không được chấp nhận bởi một ôtômat hữu hạn. Chứng tỏ rằng không có một ôtômat hữu hạn nào chấp nhận tập đó.

14. Nêu định nghĩa của máy Turing

15. Hãy mô tả một máy Turing được dùng để chấp nhận các tập như thế nào

16. Hãy mô tả một máy Turing được dùng để tính các hàm lý thuyết số như thế nào.

BÀI TẬP BỔ SUNG

***1.** Tìm văn phạm cấu trúc câu sinh ra các ngôn ngữ sau

a) Tập các xâu nhị phân có dạng $0^{2n}1^{3n}$ với n là một số nguyên không âm.

b) Tập các xâu nhị phân có số các số 0 nhiều gấp hai lần số các số 1.

c) Tập các xâu nhị phân có dạng w^2 với w là một xâu nhị phân.

***2.** Tìm văn phạm cấu trúc câu sinh ra tập $\{0^{2^n} \mid n \geq 0\}$

Đối với các Bài tập 3 và 4, cho $G = (V, T, S, P)$ là một văn phạm phi *ngữ cảnh với* $V = \{(,), S, A, B\}$, $T = \{(,)\}$, *ký hiệu xuất phát là* S *và các sản xuất là* $S \to A$, $A \to AB$, $A \to B$, $B \to (A)$ *và* $B \to (\ )$, $S \to \lambda$

3. Dựng cây dẫn xuất của

a) (()) b) ()(()) c) ((()()))

***4.** Chứng minh rằng L(G) là tập tất cả các xâu gồm các dấu ngoặc đúng đã được định nghĩa trong Chương 3.

Một văn phạm phi ngữ cảnh là ***không rõ nghĩa*** *nếu có một từ trong L(G) có hai dẫn xuất tạo ra các cây dẫn xuất khác nhau.*

5. Chứng tỏ rằng văn phạm $G = (V, T, S, P)$ với $V = \{0, S\}$, $T = \{0\}$, ký hiệu xuất phát là S và các sản xuất là $S \to 0S$, $S \to S0$ và $S \to 0$ là không rõ nghĩa bằng cách xây dựng hai cây dẫn xuất khác nhau cho 0^3.

6. Chứng tỏ rằng văn phạm $G = (V, T, S, P)$ với $V = \{0, S\}$, $T = \{0\}$, S là ký hiệu xuất phát và các sản xuất là $S \to 0S$, $S \to 0$ là rõ nghĩa.

7. Giả sử A và B là hai tập con hữu hạn của V^* với V là một bộ chữ cái. Hỏi có nhất thiết phải đúng đẳng thức $|AB| = |BA|$?

8. Chứng minh hoặc bác bỏ các khẳng định sau đối với các tập con A, B và C của V^*

a) $A(B \cup C) = AB \cup AC$ b) $A(B \cap C) = AB \cap AC$

c) $(AB)C = A(BC)$ d) $(A \cup B)^* = A^* \cup B^*$

9. Cho A và B là các tập con của V^*. Hỏi có thể suy ra $A \subseteq B$ nếu $A^* \subseteq B^*$?

10. Xác định tập các xâu với các ký hiệu trong tập {0, 1, 2} được biểu diễn bởi biểu thức chính quy $(\mathbf{2}^*)(0 \cup (\mathbf{12}^*))^*$

Độ cao tính tú *h*(**E**) *của một biểu thức chính quy trên tập I được định nghĩa một cách đệ quy như sau:*

$h(\varnothing) = \mathbf{0}$

$h(\mathbf{x}) = 0$ nếu $\mathbf{x} \in I$

$h(\mathbf{E}_1 \cup \mathbf{E}_2) = h(\mathbf{E}_1\mathbf{E}_2) = \max(h(\mathbf{E}_1), h(\mathbf{E}_2))$ nếu $\mathbf{E}_1$ và $\mathbf{E}_2$ là các biểu thức chính quy

$h(\mathbf{E}^*) = h(\mathbf{E}) + 1$ nếu $\mathbf{E}$ là biểu thức chính quy

11. Tìm độ cao tính tú của các biểu thức chính quy sau:

a) $\mathbf{0}^*\mathbf{1}$ b) $\mathbf{0}^*\mathbf{1}^*$ c) $(\mathbf{0}^*\mathbf{01})^*$ d) $((\mathbf{0}^*\mathbf{1})^*)^*$

e) $(\mathbf{010}^*)(\mathbf{1}^*\mathbf{01}^*)^*((\mathbf{01})^*(\mathbf{10})^*)^*$ f) $((((\mathbf{0}^*)\mathbf{1})^*\mathbf{0})^*)\mathbf{1})^*$

***12.** Đối với mỗi biểu thức chính quy sau đây, tìm một biểu thức chính quy biểu diễn cùng một ngôn ngữ nhưng có độ cao tính tú cực tiểu

a) $(\mathbf{0}^*\mathbf{1}^*)^*$ b) $(\mathbf{0}(\mathbf{01}^*\mathbf{0}^*))^*$ c) $(\mathbf{0}^*\cup(\mathbf{01})^*\cup\mathbf{1}^*)^*$

13. Dựng một máy hữu hạn trạng thái có đầu ra tạo một đầu ra là 1, nếu xâu nhị phân được đọc tới lúc này như một đầu vào chứa bốn hoặc nhiều hơn các số 1. Sau đó, dựng một ôtômat hữu hạn tất định chấp nhận tập đó.

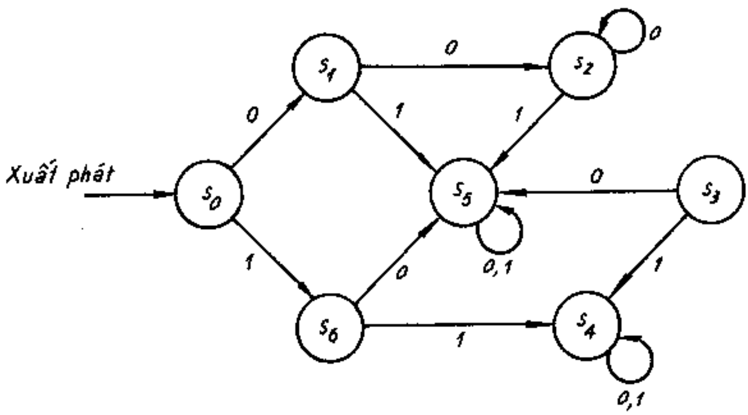
14. Dựng một máy hữu hạn trạng thái có đầu ra tạo một đầu ra là 1, nếu xâu nhị phân được đọc tới lúc này như một đầu vào gồm bốn hoặc nhiều hơn các số 1 liên tiếp. Sau đó, dựng một ôtômat hữu hạn tất định chấp nhận tập đó.

15. Cũng cho và hỏi như trên nhưng với xâu đầu vào kết thúc bởi bốn hoặc nhiều hơn các số 1 liên tiếp.

16. Trạng thái s' trong một máy hữu hạn trạng thái được nói là **đạt tới được từ** trạng thái s, nếu có một xâu đầu vào x sao cho $f(s, x) = s'$. Trạng thái s được gọi là **đệm** nếu có một **xâu** đầu vào không rỗng x sao cho $f(s, x) = s$. Một trạng thái s được gọi là **chìm** nếu $f(s, x) = s$ với mọi xâu đầu vào x.

Hãy trả lời các câu hỏi sau về máy hữu hạn trạng thái có giản đồ trạng thái cho dưới đây.

a) Các trạng thái nào đạt tới được **từ** s_0?



b) Các trạng thái nào đạt tới được từ s_2?

c) Các trạng thái nào là đệm?

d) Các trạng thái nào là chìm?

***17.** Cho S, I và O là các tập hữu hạn sao cho $|S| = n$, $|I| = k$, và $|O| = m$.

a) Hỏi có thể dựng được bao nhiêu máy hữu hạn trạng thái (máy Mealy) khác nhau $M = (S, I, O, f, g, s_0)$ với trạng thái xuất phát s_0 có thể chọn tùy ý?

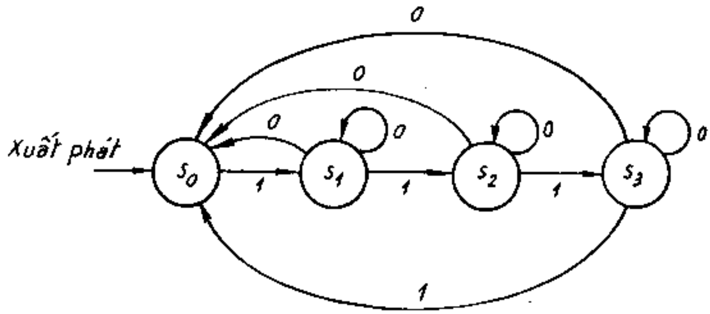
b) Hỏi có thể dựng được bao nhiêu máy Moore khác nhau $M = (S, I, O, f, g, s_0)$ với trạng thái xuất phát s_0 có thể chọn tùy ý?

***18.** Cho S và I là các tập hữu hạn sao cho $|S| = n$ và $|I| = k$. Hỏi có bao nhiêu ôtômat hữu hạn khác nhau $M = (S, I, f, s_0, F)$ với trạng thái xuất phát s_0 và tập con F của S chứa các trạng thái kết thúc có thể chọn tùy ý

a) nếu ôtômat là tất định

b) nếu ôtômat có thể là không tất định (*chú ý:* điều này có thể bao hàm cả các ôtômat tất định)

19. Dựng một ôtômat hữu hạn tất định tương đương với ôtômat không tất định có giản đồ trạng thái cho dưới đây



20. Xác định ngôn ngữ được chấp nhận bởi ôtômat cho trong Bài tập 19.

21. Dựng các ôtômat hữu hạn chấp nhận các tập sau:

a) $\mathbf{0}^*(\mathbf{10})^*$ b) $(\mathbf{01} \cup \mathbf{111})^*\mathbf{10}^*(\mathbf{0} \cup \mathbf{1})$ c) $(\mathbf{001} \cup (\mathbf{11})^*)^*$

***22.** Tìm các biểu thức chính quy biểu diễn tập tất cả các xâu nhị phân

a) được tạo bởi các khối gồm một số chẵn các con số 1 "rắc" xen kẽ với các khối gồm một số lẻ các con số 0.

b) có ít nhất hai số **0** hoặc ba số 1 liên tiếp

c) không có ba số không hoặc hai số 1 liên tiếp

***23.** Chứng minh rằng nếu A là một tập chính quy thì $\overline{A}$ cũng là một tập chính quy

***24.** Chứng minh rằng nếu A và B là hai tập chính quy, thì $A \cap B$ cũng là chính quy.

***25.** Tìm các ôtômat hữu hạn chấp nhận tập các xâu nhị phân sau:

a) tập hợp các xâu xuất phát với không hơn ba số **0** liên tiếp và chứa ít nhất hai số 1 liên tiếp.

b) tập tất cả các xâu có một số chẵn các ký hiệu và không chứa khối **101**.

c) tập tất cả các xâu có ít nhất ba khối gồm hai hoặc nhiều hơn các số 1 và ít nhất hai số 0.

***26.** Chứng minh rằng $\{0^{2^n} \mid n \in Z\}$ là không chính quy. Bạn có thể dùng Bổ đề bơm được cho trong Bài tập 16 của Tiết 10.4.

***28.** Đối với các ngôn ngữ phi ngữ cảnh có một kết quả tương tự với Bổ đề bơm cho các tập chính quy. Giả sử $L(G)$ là ngôn ngữ phi ngữ cảnh được sinh bởi văn phạm phi ngữ cảnh G. Kết quả nói trên phát biểu rằng: tồn tại một hằng số N sao cho nếu z là một từ trong $L(G)$ với $l(z) \geqslant N$, thì z có thể được viết dưới dạng $uvwxy$ trong đó $l(vwx) \leqslant N$, $l(v, x) \geqslant 1$ và uv^iwx^iy thuộc $L(G)$ với $i = 0, 1, 2, 3...$ Dùng kết quả trên chứng minh rằng không có một văn phạm phi ngữ cảnh G nào với $L(G) = \{0^n1^n2^n \mid n = 0, 1, 2...\}$.

BÀI TẬP TRÊN MÁY TÍNH

Viết các chương trình với Input và Output sau:

1. Cho các sản xuất trong một văn phạm cấu trúc câu, hãy xác định đó là loại văn phạm nào trong sơ đồ phân loại của Chomsky.

***2.** Cho các sản xuất trong một văn phạm phi ngữ cảnh và một xâu, hãy tạo cây dẫn xuất của xâu đó, nếu nó nằm trong ngôn ngữ được sinh bởi văn phạm đó.

3. Cho bảng trạng thái của máy Moore và một xâu đầu vào, hãy tạo xâu đầu ra được sinh bởi máy đó.

4. Cho bảng trạng thái của máy Mealy và một xâu đầu vào, hãy tạo xâu đầu ra sinh bởi máy đó.

5. Cho bảng trạng thái của một ôtômat hữu hạn tất định và một xâu, hãy quyết định xem xâu này có được chấp nhận bởi ôtômat đó hay không.

8. Cho bảng trạng thái của một ôtômat hữu hạn không tất định và một xâu, hãy quyết định xem xâu này có được chấp nhận bởi ôtômat đó hay không.

***7.** Cho bảng trạng thái của một ôtômat hữu hạn không tất định, hãy lập bảng trạng thái của một ôtômat hữu hạn tất định chấp nhận cùng một ngôn ngữ.

****8.** Cho một biểu thức chính quy, hãy dựng một ôtômat hữu hạn không tất định chấp nhận tập mà biểu thức đó biểu diễn.

9. Cho một văn phạm chính quy, hãy dựng một ôtômat hữu hạn chấp nhận ngôn ngữ được sinh bởi văn phạm đó.

10. Cho một ôtômat hữu hạn, hãy xây dựng một văn phạm chính quy sinh ra ngôn ngữ được chấp nhận bởi ôtômat đó.

***11.** Cho một máy Turing, hãy tìm xâu đầu ra tạo bởi xâu đầu vào đã cho.

TÍNH TOÁN VÀ KHÁM PHÁ

Dùng các chương trình mà bạn đã viết để làm các bài tập sau

1. Tìm $B(n)$ (xem Bài tập 23 Tiết 10.5) đối với trường hợp hai trạng thái bằng cách thử tất cả các máy Turing khả dĩ có hai trạng thái và bộ chữ cái $\{1, B\}$.

***2.** Cũng hỏi như trên cho trường hợp ba trạng thái.

****3.** Cũng hỏi như trên cho trường hợp bốn trạng thái.

****4.** Sẽ tiến bộ rất nhiều, nếu bạn có thể tiến tới giải bài toán trên với 5 trạng thái.

****5.** Sẽ tiến bộ rất nhiều, nếu bạn có thể tiến tới giải bài toán trên với 6 trạng thái.

VIẾT TIỂU LUẬN

Dùng các tư liệu ngoài cuốn sách này, viết các tiểu luận trả lời những câu hỏi sau

1. Hãy mô tả cách lập mô hình sự tăng trưởng của một số loại cây bằng cách dùng hệ Lidenmeyer. Một hệ như thế dùng văn phạm với các sản xuất mô hình hóa các cách tăng trưởng khả dĩ khác nhau của các cây.

2. Hãy giải thích khái niệm cực tiểu hóa các ôtômat hữu hạn. Cho một thuật toán thực hiện sự cực tiểu hóa đó.

3. Nêu định nghĩa của ôtômat tế bào. Giải thích các ứng dụng của nó.

4. Nêu định nghĩa của ôtômat đẩy xuống. Hãy giải thích các ôtômat đẩy xuống được dùng để chấp nhận các tập như thế nào? Hãy phác thảo chứng minh khẳng định câu trả lời của bạn.

5. Nêu định nghĩa của ôtômat tuyến tính – giới nội. Hãy giải thích các ôtômat tuyến tính giới nội được dùng để chấp nhận các tập như thế nào? Các tập nào được chấp nhận bởi một ôtômat tuyến tính giới nội? Hãy phác thảo một chứng minh khẳng định câu trả lời của bạn.

6. Tìm lại định nghĩa gốc của Turing về cái mà chúng ta gọi là máy Turing. Động cơ nào của ông để đưa ra định nghĩa đó?

7. Mô tả khái niệm máy Turing phổ dụng. Hãy giải thích các máy này được xây dựng như thế nào?

8. Hãy giải thích các loại ứng dụng trong đó các máy Turing không tất định được dùng thay cho các máy Turing tất định.

9. Chứng minh rằng một máy Turing có thể mô phỏng bất kỳ hành động nào của một máy Turing không tất định.

10. Chứng tỏ rằng một tập được chấp nhận bởi một máy Turing nếu và chỉ nếu nó được sinh bởi một văn phạm cấu trúc câu.

11. Mô tả những khái niệm cơ bản của giải thích – lamđa và hãy giải thích nó được dùng như thế nào để nghiên cứu tính tính được của hàm.

12. Chứng tỏ rằng một máy Turing như được định nghĩa trong chương này có thể làm được tất cả những gì mà một máy Turing n băng làm được.

13. Chứng tỏ rằng một máy Turing với băng vô hạn theo một hướng có thể làm được tất cả những gì mà một máy Turing với băng vô hạn theo cả hai hướng làm được.

14. Chứng tỏ rằng với một máy Turing và một xâu x đã cho, không tồn tại một thuật toán nào xác định được T sẽ dừng lại với x đã cho như một đầu vào hay không. Bài toán này được biết là *bài toán dừng máy*.

$\cap$ $\cup$

$\wedge$