TOÁN RỜI RẠC
Đồ thị Euler – Đồ thị Hamilton

NGUYỄN HẢI TRIỀU[1]

[1]Bộ môn Kỹ thuật phần mềm,

Khoa Công nghệ thông tin, Trường ĐH Nha Trang

NhaTrang, February 2022

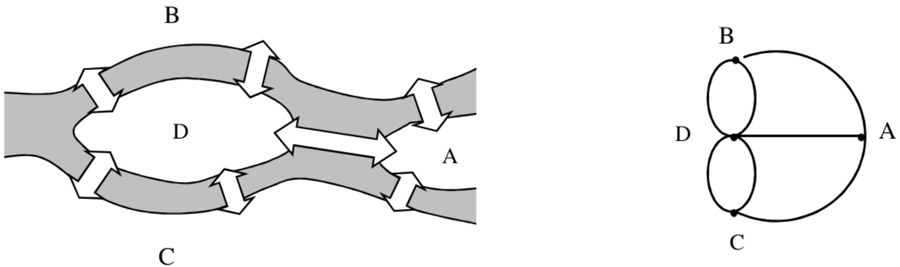
Tổng quan

Đồ thị Euler – Đồ thị Hamilton

- Đồ thị Euler – Đồ thị Hamilton có vị trí quan trọng trong lý thuyết đồ thị và trong ngành Topo của toán học hiện đại và có nhiều ứng dụng trong đời sống và trong Tin học.
- ứng dụng điển hình: bài toán người đưa thư Trung Hoa (CPP), bài toán người du lịch (TSP), bài toán lập lịch, kỹ thuật mã hóa thông tin...

Đồ thị Euler

Đồ thị Euler được Euler nghiên cứu giải quyết bài toán "Bài toán bảy chiếc cầu ở thành phố Kónigsberg": Có thể đi dạo qua tất cả các cầu sao cho mỗi cầu chỉ qua một lần rồi trở về nơi xuất phát được không?"



Euler đã biểu diễn bài toán bằng đồ thị như hình trên và khẳng định, đồ thị như vậy không thể vẽ được bởi một đường liền nét đi qua tất cả các cạnh.

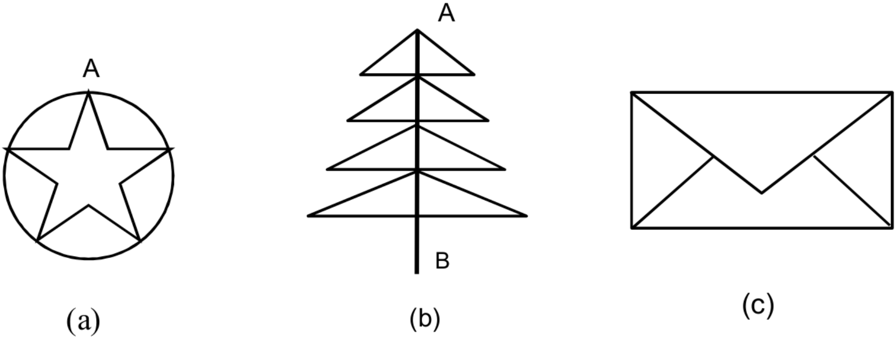
Hành trình Euler, hành trình Euler khép kín

Định nghĩa 1.1

Trong một đồ thị:

- *một hành trình đi qua tất cả các cạnh của đồ thị, mỗi cạnh chỉ qua một lần, được gọi là hành trình Euler. Đồ thị có hành trình Euler được gọi là đồ thị nửa Euler.*
- *một hành trình Euler có đỉnh đầu và cuối trùng nhau được gọi là hành trình Euler khép kín. Đồ thị có hành trình Euler khép kín được gọi là đồ thị Euler.*

Hành trình Euler, hành trình Euler khép kín



Hình 1: Kiểm tra xem hình nào có chu trình Euler

Điều kiện cần và đủ của đồ thị nửa Euler và đồ thị Euler

Định lý 1.1

Đồ thị liên thông G là đồ thị Euler khi và chỉ khi mọi đỉnh của G đều có bậc chẵn.

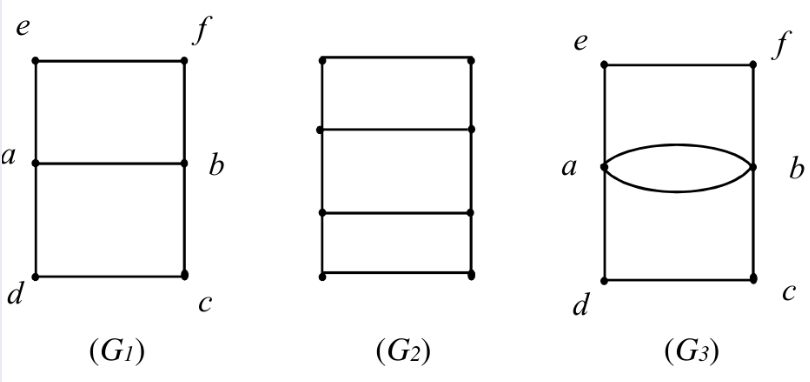
Hệ quả

Đồ thị là nửa Euler khi và chỉ khi có đúng hai đỉnh bậc lẻ. Trường hợp bài toán "Bảy chiếc cầu ở thành phố Kónigsberg" gồm bốn đỉnh bậc lẻ, nên đồ thị này không có hành trình Euler khép kín.

Điều kiện cần và đủ của đồ thị nửa Euler và đồ thị Euler

Ví dụ 1.1

Chỉ ra đồ thị Euler và nửa Euler trong các đồ thị bên dưới.



Thuật toán xác định hành trình Euler khép kín

Thuật toán xác định hành trình Euler khép kín W

- xuất phát từ một đỉnh bất kỳ của đồ thị Euler.
- xác định cạnh tiếp theo của thuật toán là việc thăm-duyệt đỉnh theo thuật toán DFS và một ngăn xếp Stack được sử dụng.
- mỗi cạnh được chọn sẽ bị loại khỏi đồ thị, đỉnh cô lập sẽ bị loại khỏi Stack và được kết nạp vào hành trình W
- kết thúc khi mọi đỉnh của đồ thị trở thành cô lập.

Algorithm 1: Thuật toán xác định hành trình Euler khép kín

Input : Đồ thị G
Output: W - hành trình Euler khép kín

```
Function Euler_ Walk
    Initial : stack = ∅; W = ∅ ; // W - hành trình Euler khép kín
    stack ← u ; // Đưa đỉnh xuất phát u vào stack
    while stack! = ∅ do
        x ← top(stack) ; // Phần tử đầu stack
        if ke(x)! = ∅ then
            y ← top(ke(x))
            stack ← y ; // Nạp y vào stack
            ke(x) = ke(x)\y
            ke(y) = ke(y)\x; // Loại bỏ cạnh xy khỏi G
        else                                        /* x là đỉnh cô lập */
            x ← stack ; // Loại x khỏi stack
            W ← x ; // Nạp x vao W
        end
    end
end
```

Thuật toán này cũng được áp dụng để xác định hành trình Euler trên đồ thị nửa Euler nhưng đỉnh xuất phát phải là một đỉnh lẻ.

Thuật toán Fleury

Thuật toán Fleury cho phép xác định một hành trình Euler trên G khi làm bằng tay. Xuất phát từ một đỉnh nào đó của G ta đi theo các cạnh của nó một cánh tùy ý chỉ cần tuân thủ hai quy tắc sau đây:

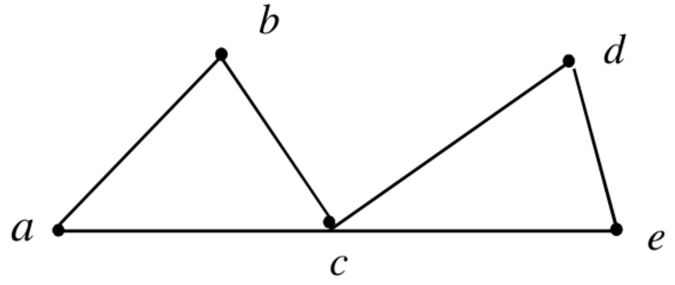
- Xóa bỏ cạnh đi qua và đồng thời xóa cả những đỉnh cô lập tạo thành.
- Ở mỗi bước ta chỉ đi qua cầu khi không còn cách lựa chọn nào khác.

Dễ dàng thấy rằng cạnh tiếp theo của thuật toán Fleury là việc thăm-duyệt đỉnh theo thuật toán DFS sao cho cạnh được chọn không phải là cầu (không làm mất tính liên thông của đồ thị đang xét).

Ví dụ 1.2

Sử dụng thuật toán Fluery cho đồ thị bên dưới với đỉnh xuất phát là a, ta thu được một hành trình Euler khép kín

$$W = (a, c, d, e, c, b, a)$$



(G)

CPP (Chinese Postman Problem)

Mỗi ngày, người đưa thư đến Bưu điện nhận những bức thư và đi phân phát theo địa chỉ ghi trên bì thư, sau đó quay trở về bưu điện. *Hãy xác định một hành trình cho người đưa thư sao cho hoàn thành công việc sớm nhất.*

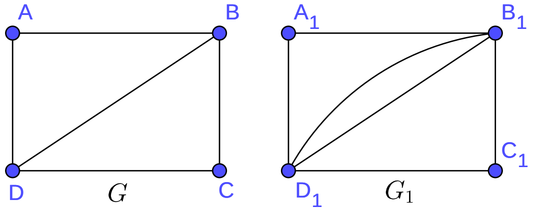
Gọi G là đồ thị của bài toán người đưa thư:

1. Với G là đồ thị không có trọng số cạnh: bài toán yêu cầu tìm hành trình khép kín đi qua tất cả các cạnh sao cho số cạnh của hành trình đi qua là bé nhất.
2. Với G là đồ thị có trọng số các cạnh: yêu cầu tìm hành trình khép kín đi qua tất cả các cạnh sao cho tổng trọng số của các cạnh trên hành trình là bé nhất.

G không có trọng số

Nếu G là đồ thị Euler thì mọi hành trình Euler khép kín của đồ thị đều là hành trình tối ưu cho người đưa thư.

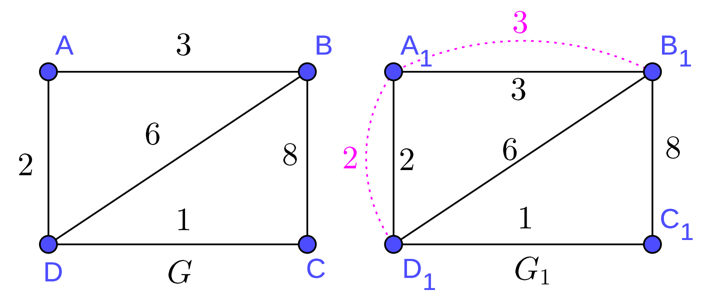
Nếu G không phải là đồ thị Euler, một hành trình của người đưa thư sẽ đi qua một số cạnh nào đó nhiều hơn một lần. Ví dụ về đồ thị nửa Euler G bên dưới:



Bằng cách nhân đôi cạnh (B, D) của đồ thị G -> G_1 là đồ thị Euler: giả sử bưu điện đặt tại A_1. Hành trình Euler khép kín có thể thu được là $W = (A_1, B_1, C_1, D_1, B_1, D_1, A_1)$, $l(W) = 7$(cạnh)

G có trọng số

Hành trình tối ưu cho người đưa thư chính là một hành trình khép kín có tổng trọng số của các cạnh trên hành trình là bé nhất. Thuật toán (Ford-Fulkerson-Meigu Guan) xây dựng đồ thị Euler G_1 bằng cách nhân đôi các cạnh tương ứng với trọng số trên đường đi ngắn nhất giữa các cặp đỉnh lẻ của đồ thị G.



Hình 2: Đồ thị G và đồ thị nhân đôi G_1

Một hành trình Euler khép kín của G_1 là hành trình tối ưu tương ứng của người đưa thư trên G. Đường đi Euler khép kín trong hình 2 xuất phát từ đỉnh A_1 đi dọc theo tất cả các cạnh của G_1 , chẳng hạn như: $W = (A_1, B_1, C_1, D_1, A_1, D_1, B_1, A_1), l(W) = 2(3+2)+6+8+1 = 25$.

G có trọng số

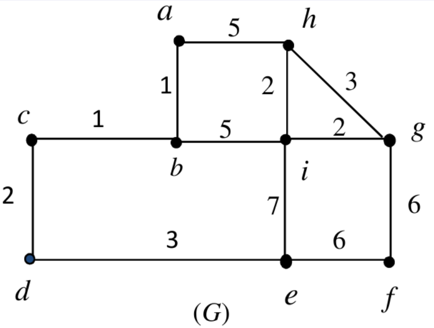
Tổng quát với đồ thị G có $2k$ đỉnh lẻ.

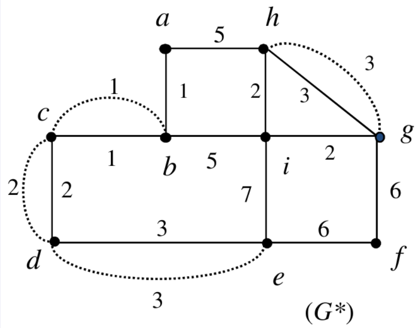
1. đầu tiên chúng ta cần xác định tất cả C_{2k}^2 đường đi ngắn nhất giữa các cặp đỉnh lẻ của G.
2. chọn một phân hoạch của tập các đỉnh lẻ gồm hai tập con V_1 và V_2, $|V_1| = |V_2| = k$ sao cho k đường đi ngắn nhất (có các đỉnh đầu mút thuộc V_1 và V_2)
3. xây dựng đa đồ thị Euler G^* từ đồ thị G bằng cách nhân đôi các cạnh của k đường đi ngắn nhất đã chọn này.

G có trọng số

Ví dụ 1.3

Xét bài toán CPP với đồ thị trọng số G: từ bốn đỉnh bậc lẻ là b, e, h, g*, ta chọn được một phân hoạch* $V_1 = b, h, V_2 = e, g$ *với hai đường đi ngắn nhất là* (b, c, d, e) *và* (h, g) *có tổng chi phí nhỏ nhất, bằng 9. Nhân đôi các cạnh trên đường đi ngắn nhất -> G**

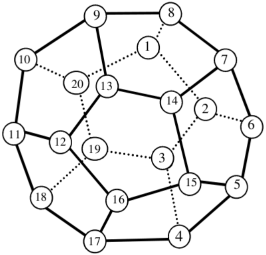




Trò chơi vòng quanh thế giới

Trò chơi đố vui của nhà toán học R. Hamilton

Hãy tìm một đường đi từ một đỉnh đi dọc theo các cạnh của khối thập nhị diện đều (mỗi mặt là một hình ngũ giác đều như trên hình 3) ghé thăm các đỉnh còn lại, mỗi đỉnh đúng một lần rồi trở về nơi đã xuất phát.



Hình 3: Trò chơi vòng quanh thế giới của Hamilton.

Đường Hamilton và chu trình Hamilton

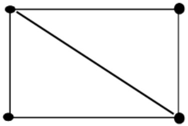
Định nghĩa 2.1

Trong một đồ thị, một đường đi qua tất cả các đỉnh của đồ thị được gọi là đường Hamilton. Một chu trình đi qua tất cả các đỉnh của đồ thị được gọi là chu trình Hamilton.

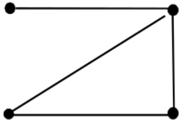
Trò chơi vòng quanh thế giới của Hamilton chính là bài toán yêu cầu xác định một chu trình Hamilton trên đồ thị hình 3.

Định nghĩa 2.2

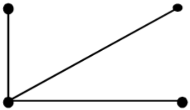
Một đồ thị được gọi là nửa Hamilton nếu nó chứa đường Hamilton. Một đồ thị được gọi là Hamilton nếu nó chứa chu trình Hamilton.



(G1)



(G2)



(G3)

Ví dụ 2.1

Hãy kiểm tra xem các đồ thị ở hình trên có phải là đồ thị Hamilton, nửa Hamilton hay không?

Điều kiện đủ để một đồ thị trở thành Hamilton và nửa Hamilton: sử dụng các định lý sau với yêu cầu các định lý đều có một điểm chung là yêu cầu đồ thị phải dày, nghĩa là số cạnh không bé hơn $n(n-1)/4$ với n là số đỉnh của đồ thị.

Định lý 2.1 (Dirac, 1952)

Giả sử G là một đơn đồ thị liên thông n đỉnh ($n \geq 3$) và mọi đỉnh có bậc ít nhất bằng $n/2$. Khi đó G là đồ thị Hamilton.

Định lý 2.2

- *Giả sử đồ thị G gồm $n \geq 3$ đỉnh và mọi cặp đỉnh của nó đều có tổng bậc không bé hơn n. Khi đó G là đồ thị Hamilton.*
- *Giả sử đồ thị G gồm $n \geq 3$ đỉnh và mọi cặp đỉnh của nó đều có tổng bậc không bé hơn $n-1$. Khi đó G là đồ thị nửa Hamilton.*

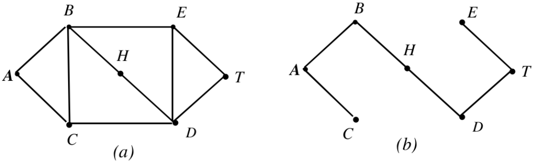
Định lý 2.3 (Ore, 1960)

Giả sử đồ thị G gồm $n \geq 3$ đỉnh và mọi cặp đỉnh không kề nhau đều có tổng bậc không bé hơn n. Khi đó G là đồ thị Hamilton.

Định lý 2.4

Giả sử G là đồ thị Hamilton. Khi đó,

1. *mọi đỉnh của nó phải có bậc lớn hơn hoặc bằng 2.*
2. *một đỉnh bậc 2 thì hai cạnh của nó phải nằm trên một chu trình Hamilton.*
3. *một đỉnh có bậc lớn hơn 2 và có hai cạnh của nó nằm trên một chu trình Hamilton thì các cạnh còn lại của nó không nằm trên chu trình Hamilton đó.*



Hình 4

Ví dụ 2.2

Xét đồ thị G ở hình 4(a):

- *Theo định lý 2.4(2). G có ba đỉnh bậc hai là A, H và T. Vì vậy, nếu G có chu trình Hamilton C chứa các cạnh AB, AC, BH, DH, TD, TE;*
- *Theo định lý 2.4(3), hai cạnh BC và BE của đỉnh B sẽ không nằm trên C. Loại bỏ hai cạnh này khỏi G. Tương tự với đỉnh D, bỏ CD, DE. Đồ thị còn lại là một đường Hamilton (hình 4(b)) -> G không có chu trình Hamilton C mà chỉ có đường Hamilton.*

Thuật toán liệt kê các chu trình Hamilton

Dựa trên cơ sở của thuật toán quay lui

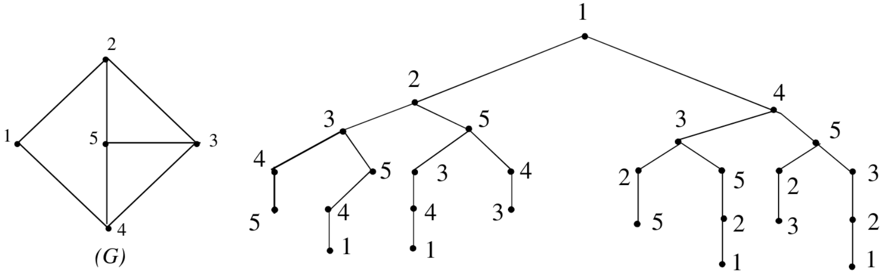
1. xuất phát từ một đỉnh tùy ý của đồ thị
2. xây dựng đường đi từ đỉnh này bằng cách lần lượt ghép các cạnh của đồ thị sao cho mỗi cạnh mới ghép sẽ nối với đỉnh cuối cùng trên đường đi với một đỉnh còn chưa thuộc đường đi
3. thực hiện như vậy cho đến khi đường đi qua tất cả các đỉnh của đồ thị để thu được một chu trình Hamilton.
4. quay lui lại đỉnh trước đó để tìm chu trình Hamilton tiếp theo

Code tham khảo xem tại đây

Thuật toán liệt kê các chu trình Hamilton

Ví dụ 2.3 (Bài tập thực hành)

Áp dụng thuật toán liệt kê các chu trình Hamilton dựa trên quay lui cho đồ thị dưới đây

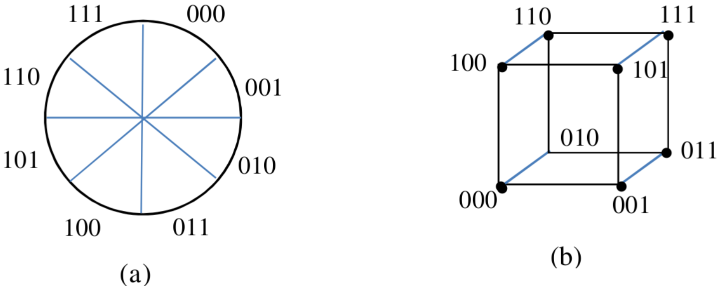


Hình 5: Đồ thị G và cây liệt kê các đường đi và chu trình Hamilton

Ứng dụng của đường đi và chu trình Hamilton

Mã Gray

Bài toán lập mã Gray yêu cầu đặt các chuỗi nhị phân n-bit lên trên một vòng tròn được phân chia thành 2^n cung tròn bằng nhau sao cho hai chuỗi kề nhau chỉ sai khác nhau một bit.

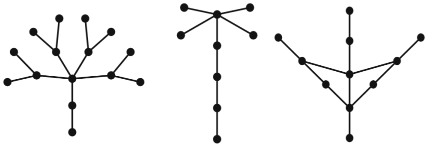


Hình 6: Đồ thị minh họa mã Gray cho các chuỗi nhi phân 3-bit

Định nghĩa và tính chất

Định nghĩa 3.1

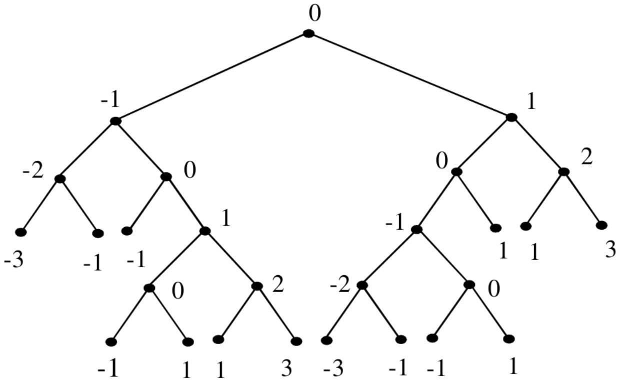
Cây là đồ thị vô hướng, liên thông, không có chu trình. Trong một cây, có thể chọn một đỉnh nào đó làm gốc. Đỉnh có bậc bằng 1 được gọi là lá (trừ đỉnh gốc). Chiều cao của cây là khoảng cách lớn nhất (theo số cạnh) từ gốc đến lá. Đồ thị không có chu trình được gọi là rừng.



Hình 7: Một rừng gồm các cây T_1, T_2, T_3 theo thứ tự

Ví dụ 3.1

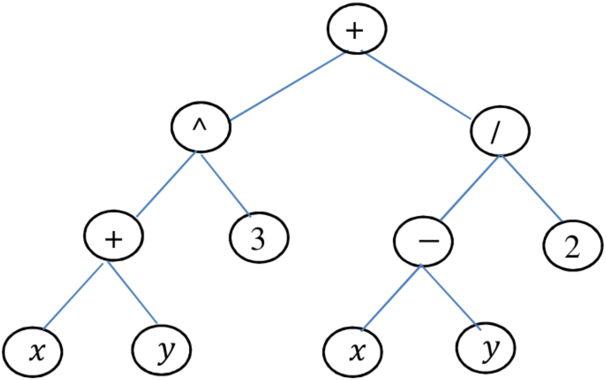
Liệt kê các đường đi khác nhau của robot biết rằng ban đầu robot đứng tại gốc của trục Ox và có thể di chuyển sang trái hoặc phải một đơn vị, robot sẽ dừng lại nếu đi đến vị trí ± 3 hoặc tại vị trí mà nó đã đi qua (không phải gốc).



Hình 8: Cây biểu diễn 14 đường đi khác nhau của robot.

Ví dụ 3.2

Mỗi biểu thức toán học sẽ được biểu diễn bởi một cây và được gọi là cây có thứ tự, trong đó các đỉnh trong (đỉnh có bậc lớn hơn 1) biểu thị các toán tử, còn các lá (đỉnh bậc 1) biểu thị các toán hạng.



Hình 9: Cây biểu diễn biểu thức $(x+y)^3+(x-y)/2$

Tính chất của cây

Tính chất của cây

- Một cây có n đỉnh thì có $n-1$ cạnh
- Nếu T là một rừng, có n đỉnh, k thành phần liên thông thì T có $n-k$ cạnh
- Một cây có nhiều hơn 1 đỉnh thì có ít nhất 2 lá
- T là một cây có k cạnh. Nếu G là đồ thị có bậc của tất cả các đỉnh đều lớn hơn hoặc bằng k thì T là đồ thị con của G. Hay G chứa tất cả các cây có tối đa $k+1$ đỉnh.

Tính chất của cây

Tính chất của cây

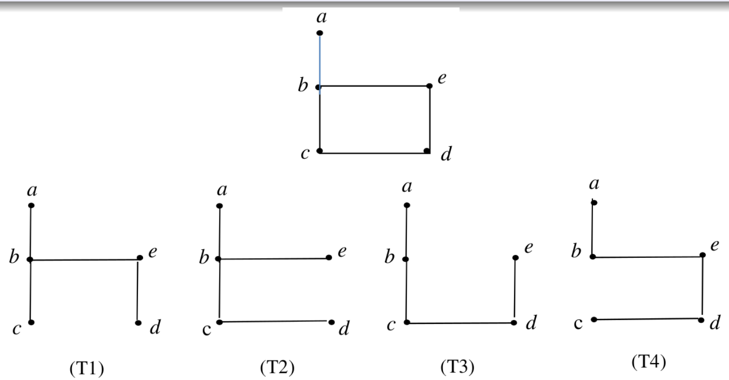
Đồ thị G có n đỉnh là một cây khi và chỉ khi thỏa mãn một trong các điều kiện:

- Không có chu trình và có n-1 cạnh
- Liên thông và có n-1 cạnh
- Giữa 2 đỉnh bất kỳ có đúng một đường đi đơn
- G không có chu trình nhưng nếu thêm bất kỳ 1 cạnh mới nào sẽ tạo thành chu trình
- Liên thông, tất cả các cạnh đều là cầu
- Liên thông, tất cả các đỉnh không phải lá của cây đều là đỉnh rẽ nhánh

Cây khung của đồ thị

Định nghĩa 3.2

Cho trước đồ thị $G = (V, E)$; cây T là cây khung của G nếu T là đồ thị con của G và chứa tất cả các đỉnh của G.

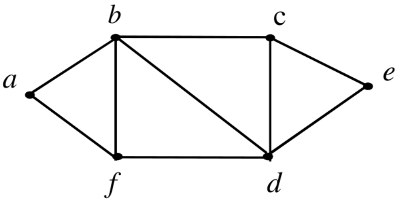


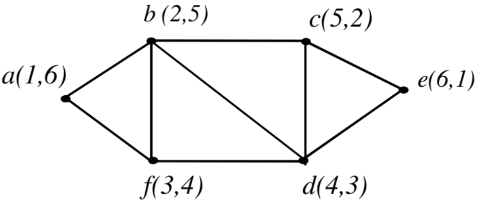
Hình 10: đồ thị G và bốn cây khung của nó.

Cách xác định cây khung của một đồ thị là tìm cách loại bỏ các cạnh để đồ thị không còn bất kỳ chu trình nào. Tuy nhiên, việc làm này là không hiệu quả trong lập trình vì đòi hỏi phải nhận biết các đỉnh đang xét có thể thuộc một chu trình trong đồ thị đã cho hay không.

Sử dụng thuật toán DFS để xây dựng cây khung của đồ thị.

- Chọn một đỉnh tùy ý $v \in V$ làm gốc và sử dụng thuật toán $DFS(v)$
- Xây dựng đường đi từ đỉnh này: lần lượt ghép các cạnh liên thuộc vào T sao cho mỗi cạnh mới ghép sẽ nối với đỉnh cuối cùng trên đường đi với một đỉnh còn chưa thuộc đường đi (đỉnh này xem như được thăm và đưa vào Stack).
- Tiếp tục như vậy cho đến khi không còn thêm được nữa. Nếu T chứa tất cả các đỉnh của đồ thị thì nó là một cây khung cần tìm.





Ví dụ 3.3

Xác định một cây khung trong đồ thị G sử dụng thuật toán DFS.
Giải: Giả sử đỉnh xuất phát được chọn là a. Một kết quả thăm-duyệt đồ thị G bởi thuật toán $DFS(a)$ *và cây khung tương ứng thu được là* $T = \{ab, bf, fd, dc, ce\}$.

Tương tự, áp dụng thuật toán BFS cho việc xây dựng cây khung của đồ thị.

Tài liệu tham khảo

- Đ.N. An
  Giáo Trình Toán Rời Rạc. *Trường ĐH Nha Trang, (2021).*
- Giáo trình Toán rời rạc
  Giáo trình Toán Rời Rạc. *Trường ĐHSP Huế. (2003), 22-35.*
- N.T. Nhựt
  Bài giảng Toán Rời Rạc. *Trường ĐH KHTN Tp.HCM. (2011).*