Chương 7: LÝ THUYẾT ĐỒ THỊ

Vấn đề lý thuyết đồ thị

Ngày nay lý thuyết đồ thị được ứng dụng khá rộng rãi trong nhiều lĩnh vực khác nhau. Các hệ thống tính toán an toàn giao thông cho vận tải, các bài toán tối ưu về lập lịch, phân bố tần số cho các trạm phát sóng, truyền tải dữ liệu trong hệ thống mạng…đều dựa trên lý thuyết đồ thị.

Riêng trong lĩnh vực tin học, khái niệm cây trong lý thuyết đồ thị còn được sử dụng để tổ chức thư mục, lưu trữ dữ liệu, biễu diễn tính toán…

Nội dung chính

- Các khái niệm cơ bản.
- Biểu diễn đồ thị.
- Đồ thị liên thông, đường đi và chu trình.
- Chu trình Euler, Hamilton.

Các khái niệm cơ bản

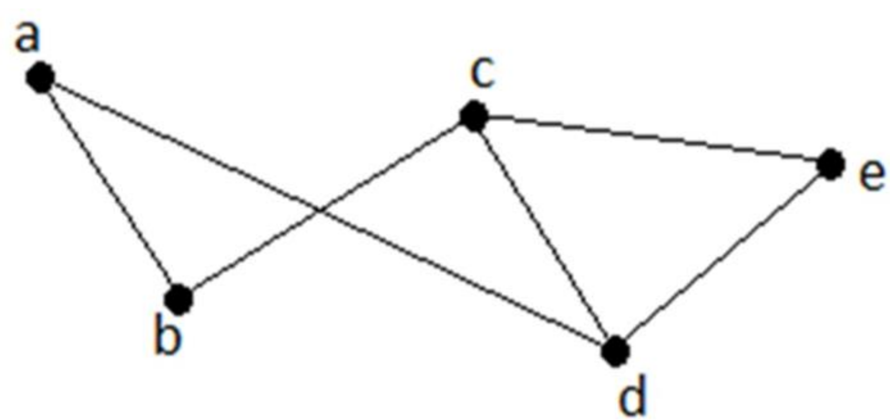
1. Đồ thị vô hướng: Tập hợp *G* = *(V, E)*, với *V* là tập hợp các đỉnh và *E* là tập hợp các cạnh. Một tập hợp các cạnh không được sắp xếp theo thứ tự tạo nên một *đồ thị vô hướng*.

   Hai đỉnh kề: hai đỉnh x, y gọi kề nhau nếu thuộc một cạnh của G.

Ví dụ: cho đồ thị vô hướng G =(V, E)

Trong đó: V = {a, b, c, d, e}

E = { ab, bc, cd, ce, de, ad}

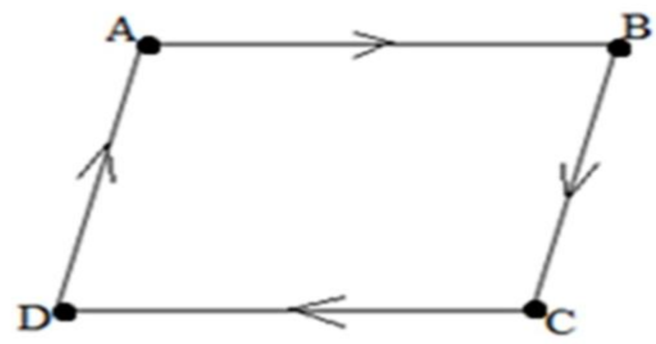


Đồ thị có hướng

Cho đồ thị G = (V ,E), nếu một cạnh (hay cung) nối liền hai đỉnh (*x*, *y*) của đồ thị và có hướng đi từ *x* đến *y* thì đồ thị *G* là *đồ thị có hướng.*

Ví dụ: $G_2 = (V, E)$; $V = \{A, B, C, D\}$

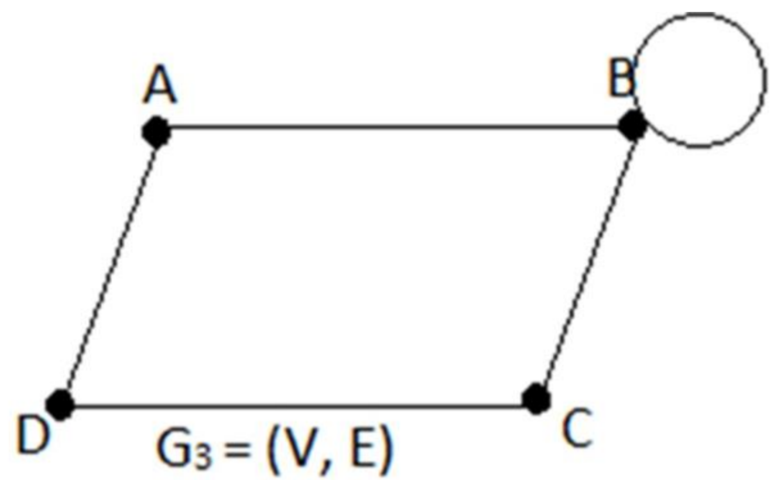
$E = \{AB, BC, CD, DA\}$



Khuyên

Khuyên của đồ thị chính là một cạnh có đỉnh đầu và đỉnh cuối trùng nhau.

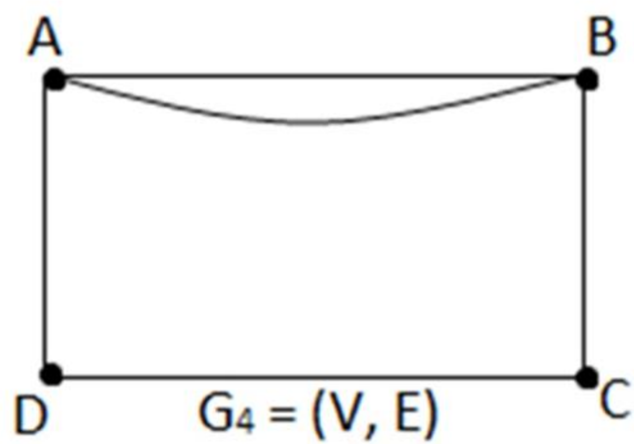
Ví dụ: Đồ thị vô hướng G_3 có khuyên tại đỉnh B



Cạnh song song

Hai cạnh của đồ thị được gọi là song song (hay cạnh bội) nếu hai cạnh có chung một cặp đỉnh.

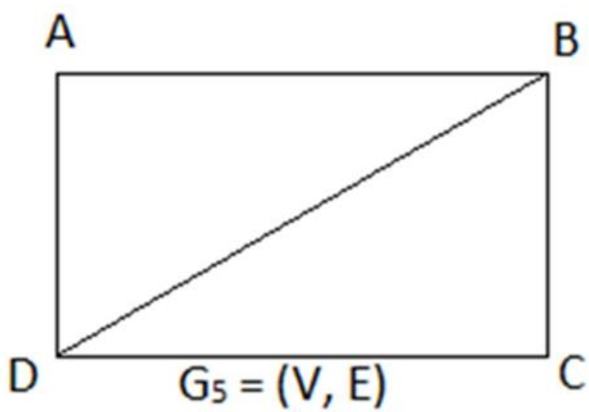
Ví dụ: đồ thị vô hướng G_4 có cạnh AB là song song

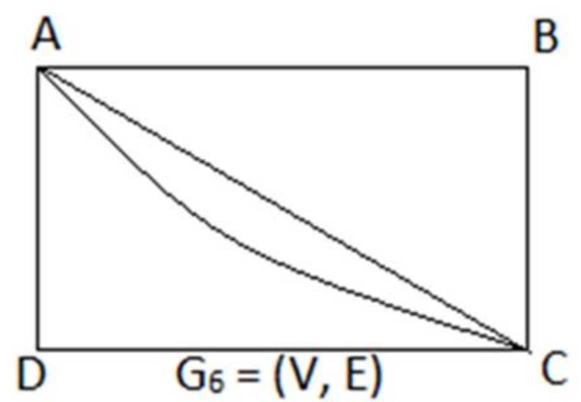


Đơn đồ thị và đa đồ thị

- *Đơn đồ thị:* *G* không tồn tại cạnh song song và khuyên thì G được gọi là đơn đồ thị.
- *Đa đồ thị:* *G* có cạnh song song hoặc khuyên thì G được gọi là đa đồ thị.

Ví dụ: G_5 là đơn đồ thị; G_6 là đa đồ thị





Bậc của đỉnh

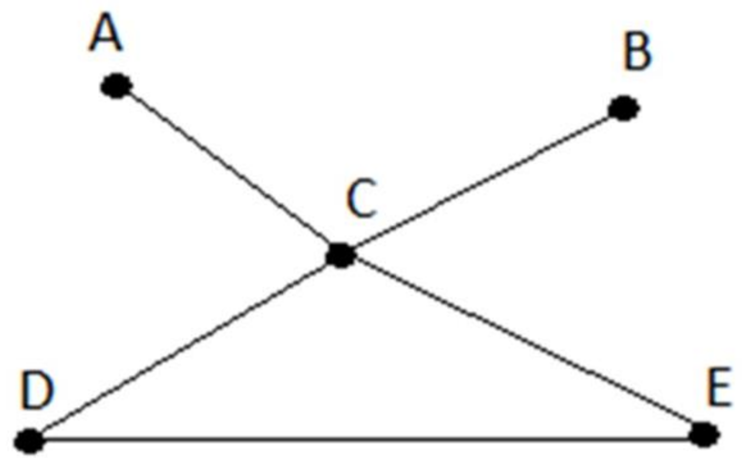
Cho đồ thị vô hướng G = (V, E).

- *Cạnh kề:* ta gọi cạnh của đỉnh x kề với đỉnh y là cạnh kề của đỉnh x và đỉnh y .
- *Bậc của đỉnh:* x là số cạnh kề với đỉnh x. Ký hiệu là d(x)
- Trong trường hợp đồ thị có hướng:
  - Bậc ngoài của đỉnh *x* là số cạnh xuất phát từ đỉnh *x*, ký hiệu là $d^+(x)$.
  - Bậc trong của đỉnh x là số cạnh đi đến *x*, ký hiệu là $d^-(x)$. Ta luôn có:

$$d(x) = d^+(x) + d^-(x)$$

- *Đỉnh treo*: nếu đỉnh x có d(x) = 0 thì đỉnh x được gọi là đỉnh treo hay đỉnh cô lập.

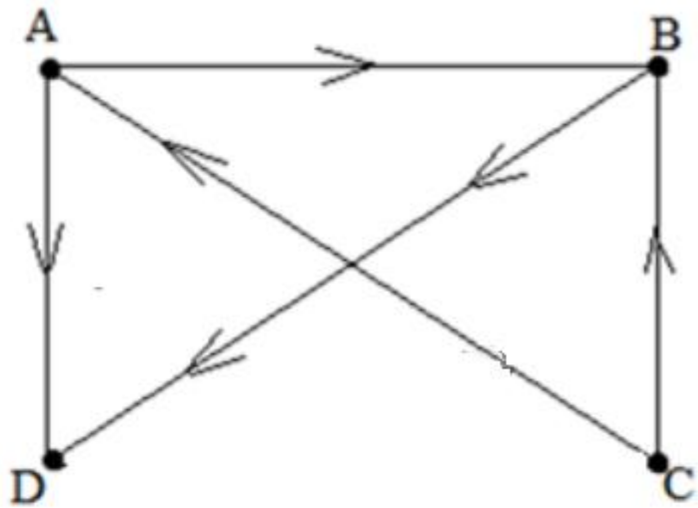
Ví dụ: tính bậc của đỉnh đồ thị vô hướng



$G_7 = (V, E)$

Đồ thị G_7 có bậc của đỉnh d(A) = 1, *d(B)* = *1*, *d(C)* = *4*, *d(D)* = *2*, *d(E)* = 2.

Ví dụ: Tính bậc đỉnh A của đồ thị có hướng

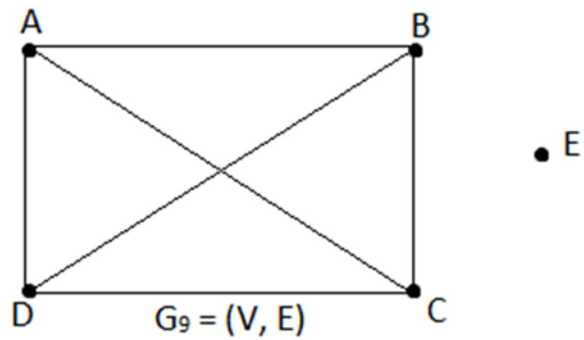


Ta có: $d^+(A) = 2$, $d^-(A) = 1$

Nên $d(A) = d^+(A) + d^-(A) = 2 + 1 = 3$

Ví dụ:

- Đồ thị G_9 có bậc của đỉnh E là đỉnh treo (cô lập). Do d(E) = 0



Đồ thị con

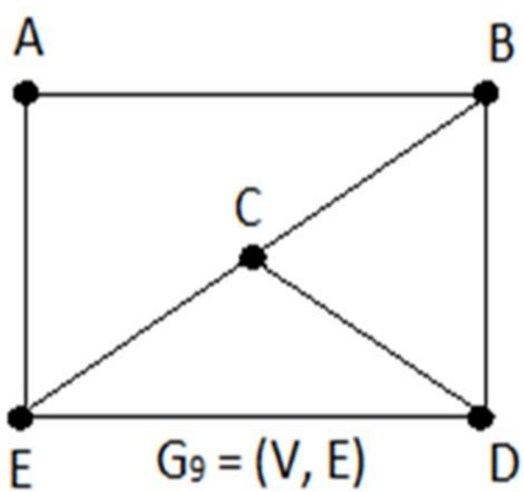
- Cho đồ thị $G = (V, E)$, đồ thị $H = (V', E')$ được gọi là *đồ thị con* của G, ký hiệu $H \subseteq G$ nếu $V' \subseteq V$ và $E' \subseteq E$.
- Trong trường hợp *H* có *V' = V* thì ta nói *H* là *đồ thị con khung* của *V*.

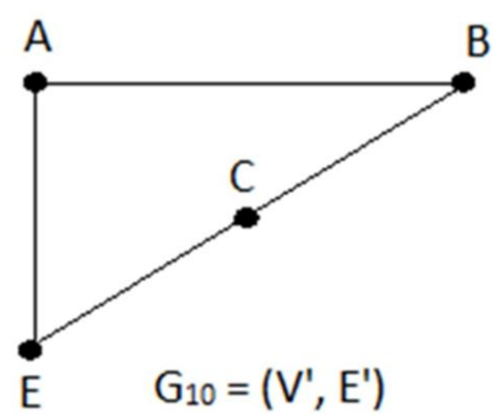
Ví dụ:

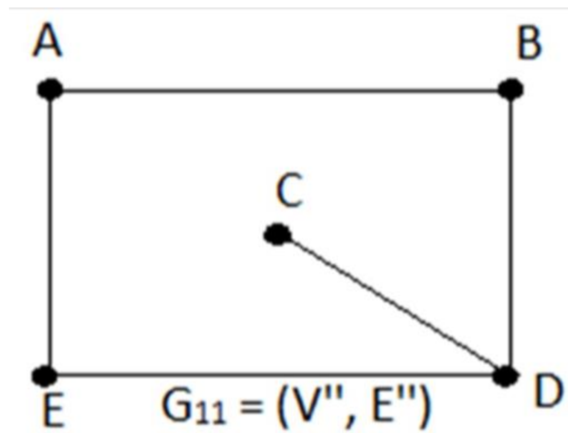
Cho đồ thị vô hướng G_9

Đồ thị vô hướng G_{10} là đồ thị con G_9

Đồ thị vô hướng G_{11} là đồ thị con khung G_9







Một số đồ thị vô hướng đặc biệt

- *Đồ thị vô hướng đầy đủ*

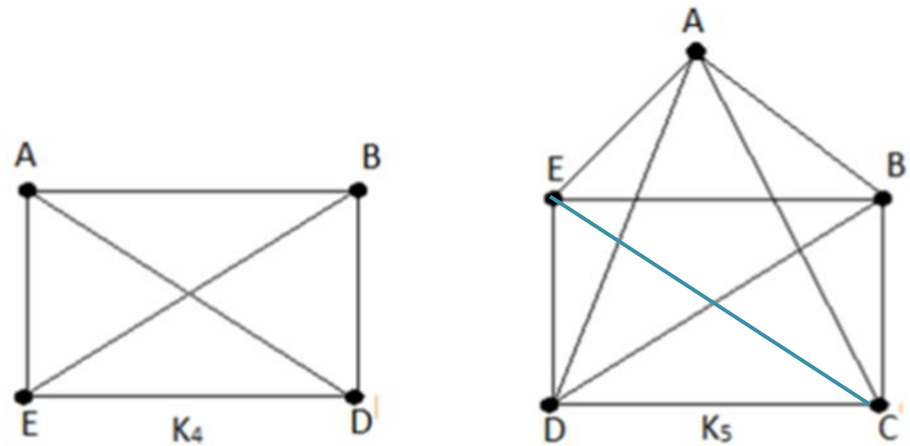
  Một đồ thị vô hướng G được gọi là đầy đủ khi tất cả các đỉnh của đồ thị đều kề với nhau từng cặp đôi một.

  Khi đó: đồ thị vô hướng đầy đủ *n* đỉnh;

  sẽ có n(n-1)/2 cạnh và mỗi đỉnh đều có n-1 bậc. ký kiệu K_n

Ví dụ

Cho đồ thị cho đồ thị vô hướng đầy đủ K_4 và K_5

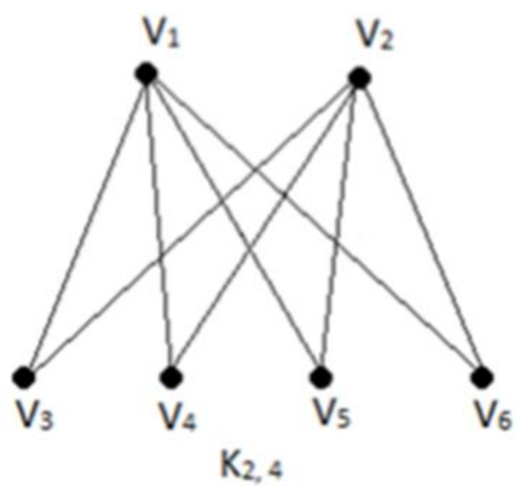


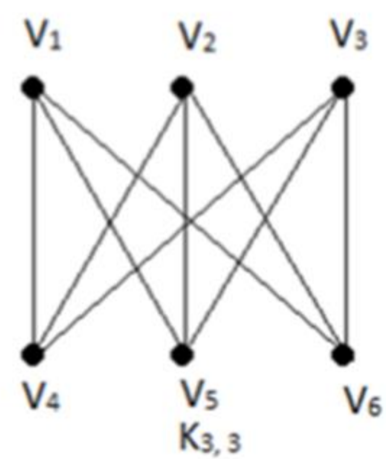
Đồ thị phân đôi

Đinh nghĩa: Một đồ thị đơn G = (V, E) là đồ thị phân đôi, nếu $V = V_1 \cup V_2$ là các tập con không rỗng và $V_1 \cap V_2 = \emptyset$, sao cho mỗi cạnh của G nối một đỉnh của V_1 với một đỉnh của V_2

Ký hiệu: $K_{m,n}$

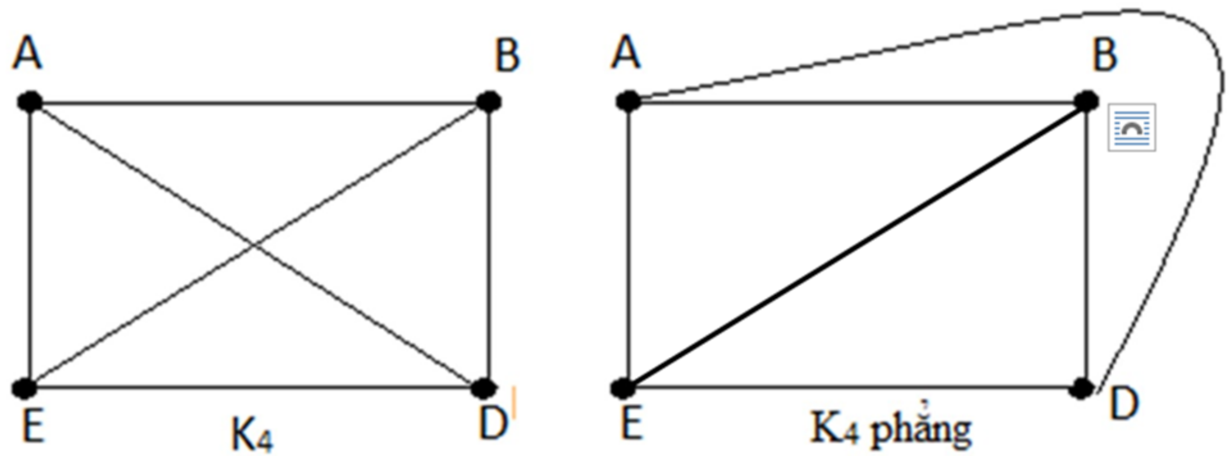
Ví dụ:





Đồ thị phẳng

Một đồ thị được gọi là phẳng nếu nó có thể được vẽ trên một mặt phẳng mà không có cạnh nào cắt nhau.

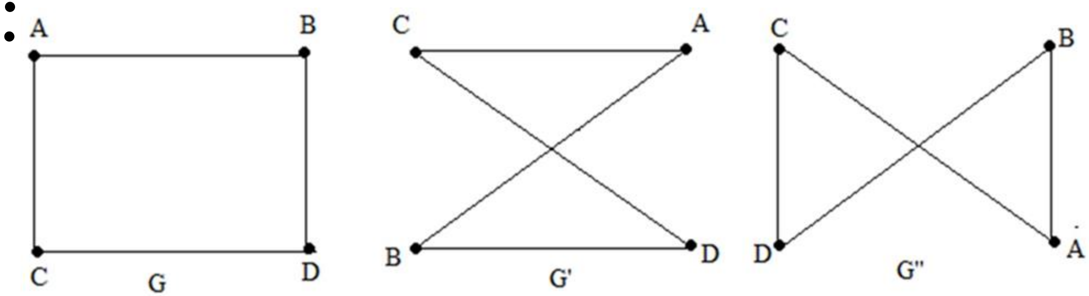


Sự đẳng cấu của đồ thị

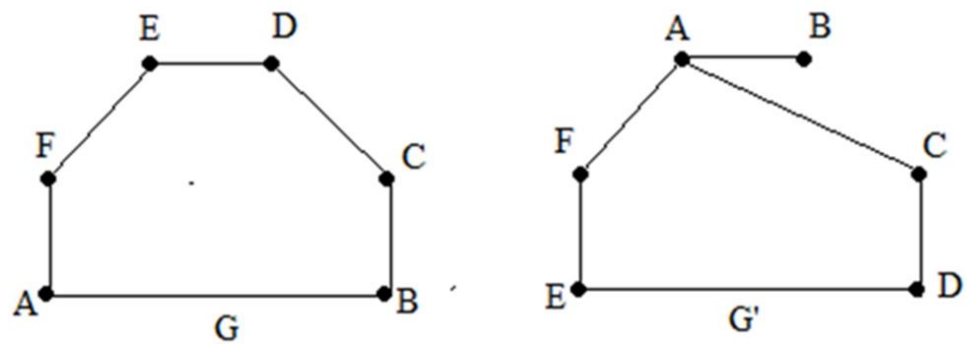
- Các đồ thị $G_1 = (V_1, E_1)$ và $G_2 = (V_2, E_2)$ được gọi là đẳng cấu với nhau nếu có một song ánh $f: V_1 \rightarrow V_2$ sao cho nếu *a* và *b* là liền kề trong V_1 thì $f(a)$ và $f(b)$ liền kề trong V_2; $\forall a, b \in V_1$. Khi đó song ánh *f* được gọi là một đẳng cấu.
- Hay nói theo cách khác hai đồ thị được gọi là đẳng cấu với nhau nếu chúng có cùng số đỉnh, cùng số cạnh và cùng số đỉnh với bậc cho sẵn.

Ví dụ

- Cho ba đồ thị vô hướng *G*, *G'* và *G"* là đẳng cấu với nhau:



- Cho hai đồ thị *G* và *G'*. *G* và *G'* không đẳng cấu với nhau vì trên *G'* có một đỉnh bậc 3.



Biểu diễn đồ thị

Việc biểu diễn đồ thị sẽ phụ thuộc các thuật toán khác nhau vào từng bài toán cụ thể, ta sử dụng những cấu trúc dữ liệu thích hợp để mô tả đồ thị trên máy tính..

Một đồ thị có thể được biểu diễn bằng *hình học, một ma trận hay một bảng*.

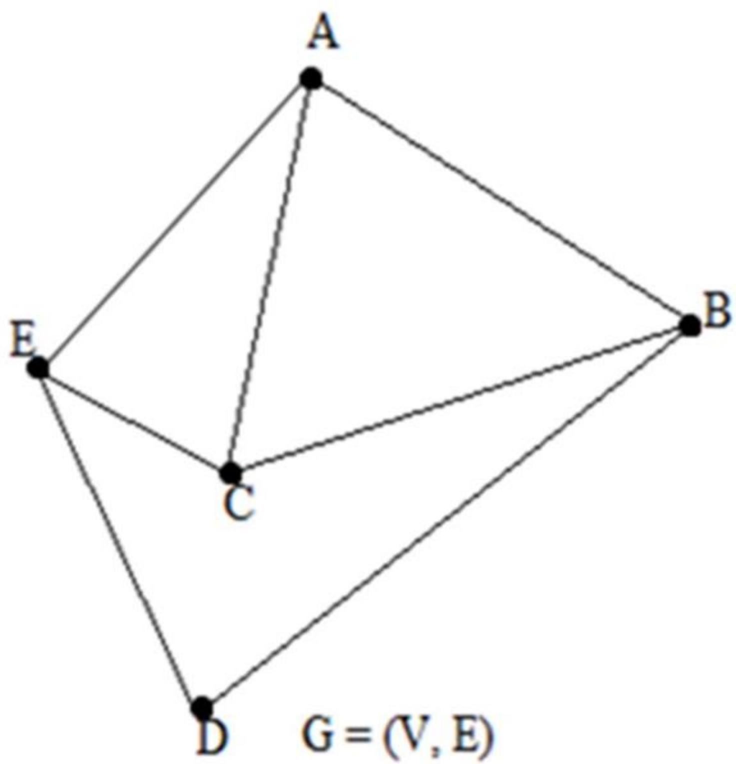
Biểu diễn đồ thị danh sách liền kề

- Một đồ thị G = (V, E) có thể được biểu diễn dưới dạng danh sách liền kề.

  Lập bảng liệt kê tất cả các đỉnh của đồ thị. Bắt đầu tại một đỉnh bất kỳ, ta liệt kê tất cả các đỉnh kề với nó.

Ví dụ:

Cho đồ thị G =(V, E)



Đỉnh	Đỉnh liền kề
A	B, C, E
B	A, C, D
C	A, B, E
D	B, E
E	A, C, D

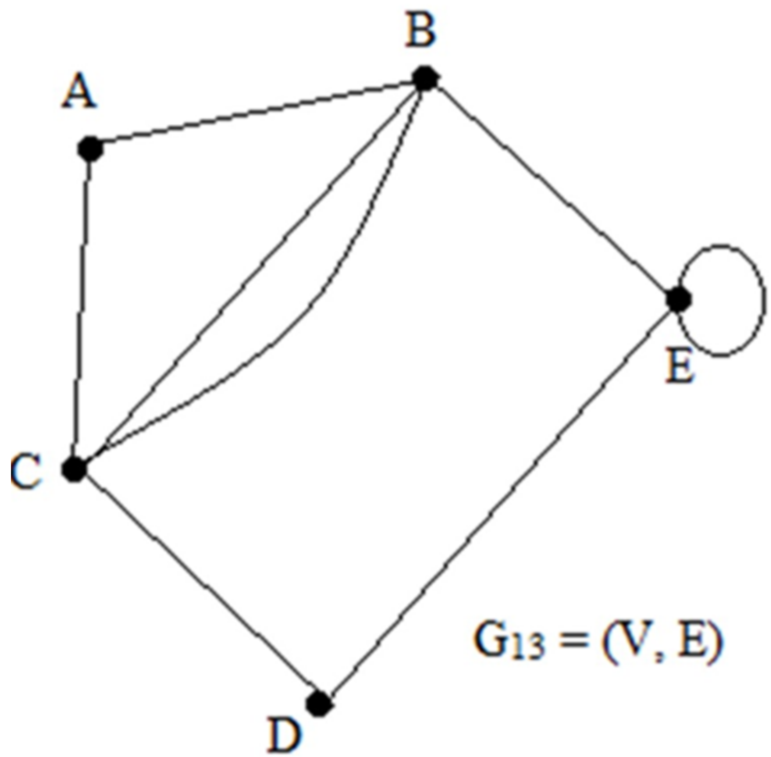
Biểu diễn đồ thị bằng ma trận kề

Cho đồ thị vô hướng $G = (V, E)$ với tập đỉnh $V = \{v_1, v_2, \ldots, v_n\}$. Ta biểu diễn đồ thị G bằng ma trận kề là *ma trận vuông cấp n* là:

$A_{nxn} = [a_{ij}]$ như sau:

$$a_{ij} = \begin{cases} 1 \text{ nếu } \{v_i, v_j\} \text{ là một cạnh của } G \\ 0 \text{ nếu không có cạnh nối đỉnh } v_i \text{ đến } v_j \end{cases}$$

Ví dụ: dùng ma trận liền kề biểu diễn đồ thị

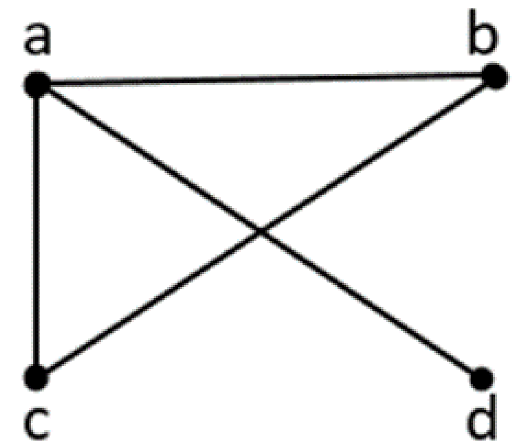


$$
\begin{array}{c c}
 & \begin{array}{ccccc} A & B & C & D & E \end{array} \\
\begin{array}{c} A \\ B \\ C \\ D \\ E \end{array} &
\begin{pmatrix}
0 & 1 & 1 & 0 & 0 \\
1 & 0 & 2 & 0 & 1 \\
1 & 2 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 1 \\
0 & 1 & 0 & 1 & 1
\end{pmatrix}
\end{array}
$$

Ví dụ:

Hãy vẽ đồ thị có ma trận liền kề có các đỉnh a, b, c, d.

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



Liên thông, đường đi và chu trình

Đường đi: đường đi độ dài n từ u đến v là số nguyên dương trong một đồ thị vô hướng là một dãy n cạnh $e_1, e_2, \ldots, e_n$ liên tiếp của đồ thị.

- Đường đi đơn: đường đi không có cạnh nào xuất hiện quá một lần.
- Đường đi sơ cấp: đường đi không có đỉnh nào xuất hiện quá một lần.
- Chu trình: đường đi bắt đầu và kết thúc tại cùng một đỉnh và có độ dài $n>=3$ được gọi là một chu trình.

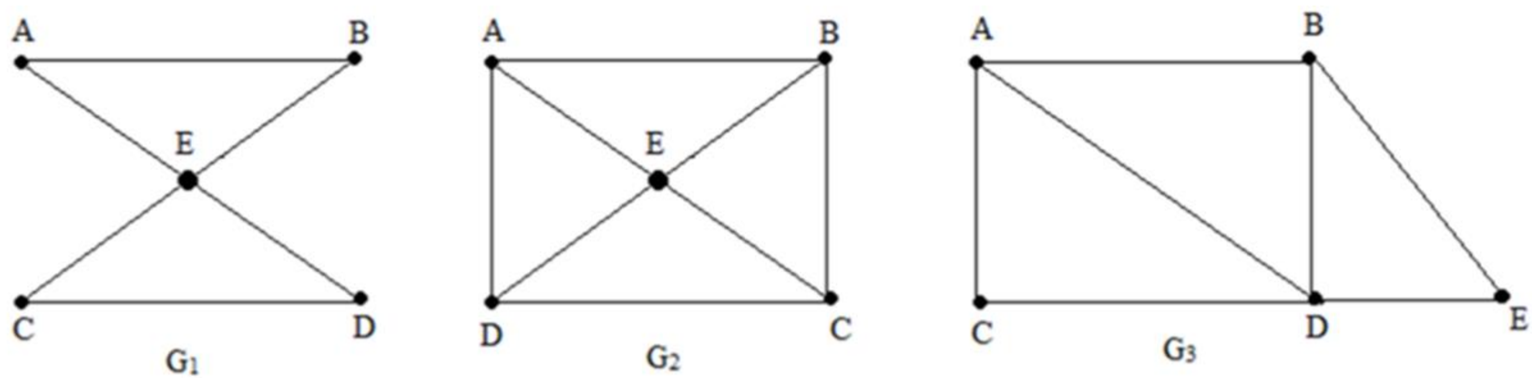
Liên thông

- Đồ thị vô hướng *G = (V, E)* được gọi là liên thông nếu với mọi cặp đỉnh phân biệt *u, v* luôn có đường đi đến chúng. Ngược lại, đồ thị *G* không là liên thông.

Chu trình Euler (đồ thị Euler)

- Cho G = (V, E) là một đa đồ thị liên thông
- *Chu trình Euler*: một chu trình đơn chứa *tất cả các cạnh* của đồ thị G.
- *Đồ thị Euler*: Một đồ thị liên thông (hay liên thông yếu đối với đồ thị có hướng) có chứa một chu trình Euler.

Ví dụ: cho các đồ thị G1, G2, G3



- Đồ thị G_1 là đồ thị Euler.
- Đồ thị G_2 , G_3 không có chu trình Euler

Định lý

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

Chu trình Hamilton (đồ thị Hamilton)

Cho đồ thị G = (V, E)

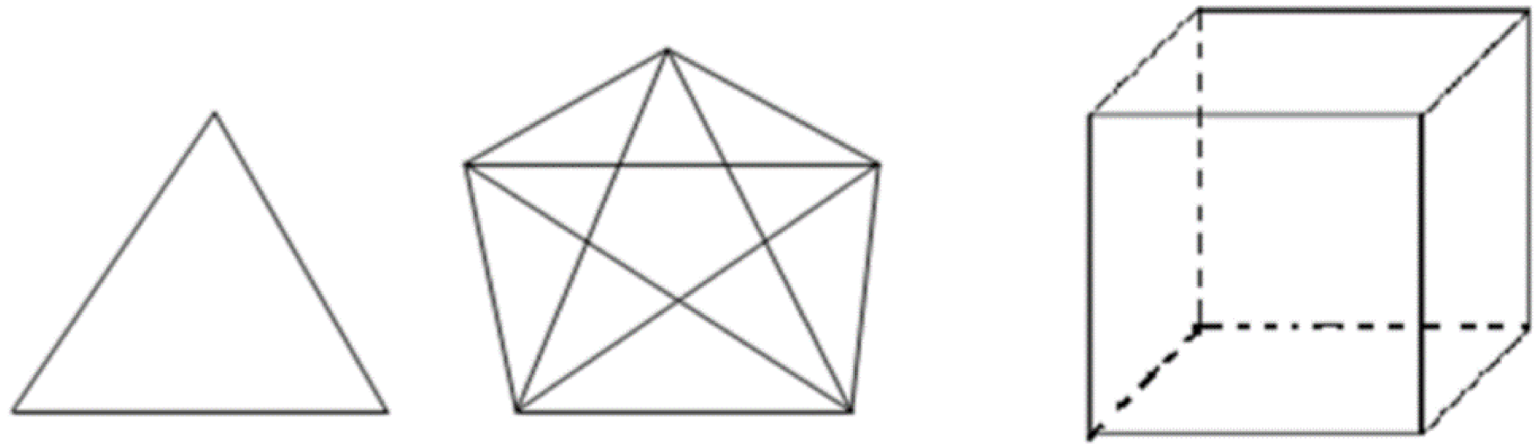
- *Chu trình Hamlton*: một chu trình sơ cấp đi qua tất cả các đỉnh của đồ thị G mỗi đỉnh chỉ đi qua một lần.
- *Đồ thị Hamilton*: đồ thị G có chứa chu trình Hamilton.

Các định lý

- *Định lý Ore*: Cho G = (V, E) là đồ thị liên thông có n đỉnh (n ≥ 3), *Nếu ∀a, b∈ V, với a, b không kề nhau trong G mà deg(a) + deg(b) ≥ n* thì G có chu trình Hamilton.
- *Định lý Dirac*: đơn đồ thị G = (V, E) là đồ thị liên thông có n đỉnh (n ≥ 3), *Nếu ∀a∈ V mà deg(a)>n/2* thì G có chu trình Hamilton.

Ví dụ: Minh họa đồ thị Hamilton

- Minh họa các đồ thị Hamilton



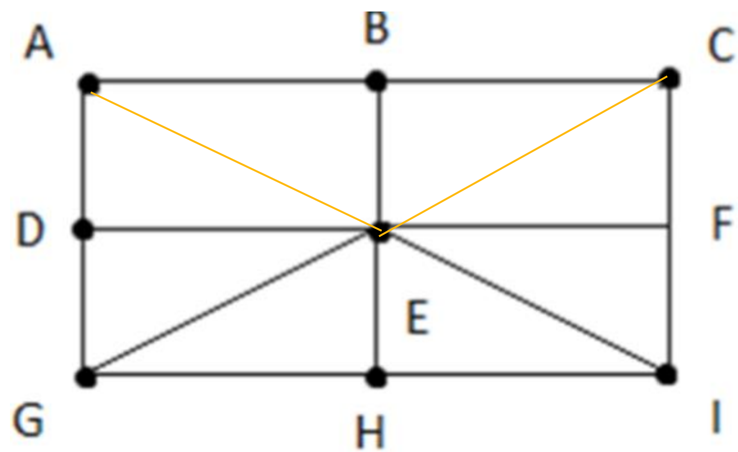
Phương pháp tìm chu trình Hamilton

Cho đồ thị G = (V, E). Để tìm một chu trình Hamilton trong đồ thị G, Theo 4 qui tắc sau:

- Nếu tồn tại một đỉnh v của G có d(v)≤1 thì đồ thị G không có chu trình Hamilton.
- Nếu đỉnh v có bậc là 2 d(v)=2 thì cả 2 cạnh tới v đều phải thuộc chu trình Hamilton.
- Chu trình Hamilton không chứa bất kỳ chu trình con thực sự nào.
- Trong quá trình xây dựng chu trình Hamilton, sau khi đã lấy 2 cạnh tới một đỉnh v đặt vào chu trình Hamilton rồi thì không thể lấy thêm cạnh nào tới v nữa, do đó có thể xóa mọi cạnh còn lại tới v.

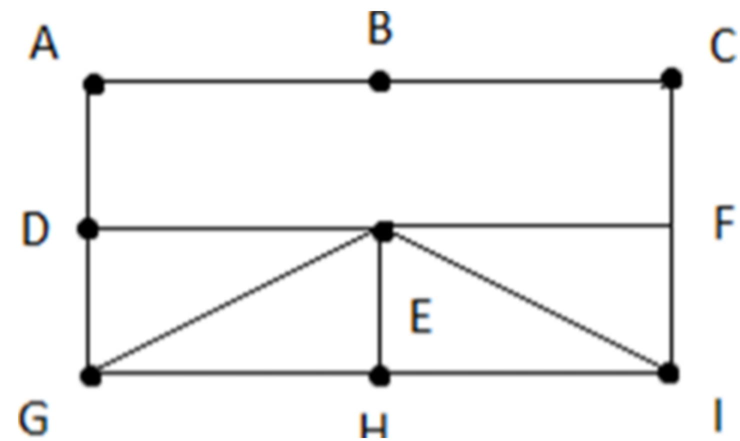
Ví dụ :

Tìm một chu trình Hamilton của đồ thị



Xuất phát từ đỉnh A . Ta có deg(A) = 3, ta giữ lại 2 cạnh liên thuộc với A: ta chọn AB và AD, ta bỏ cạnh AE . Tương tự tại B, ta bỏ cạnh BE, tại C ta bỏ cạnh CE .

Khi đó ta có đồ thị:



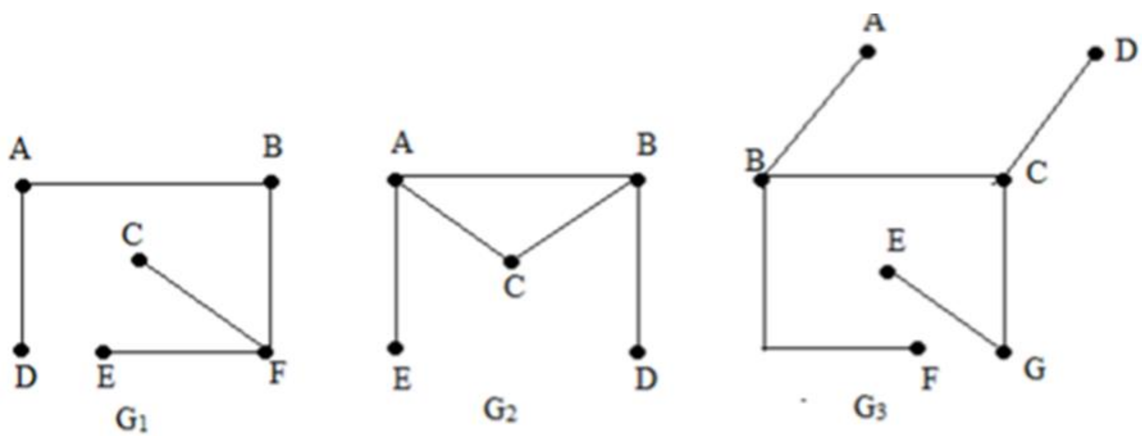
Tại đỉnh F, ta bỏ cạnh FE. Tại đỉnh I ta bỏ cạnh IE, tại đỉnh H ta bỏ cạnh HG , tại đỉnh E ta bắt buộc đi theo EG, GD . Tại đỉnh A ta phải chọn cạnh DA. Vậy, ta có chu trình Hamilton: A, B, C, F, I, H, E, G, D, A.

Cây và rừng

Định nghĩa

- *Cây*: là Đồ thị G = (V, E) vô hướng và *không có chu trình sơ cấp*. Do cây không có chu trình sơ cấp nên một *cây không có cạnh song song và khuyên*.

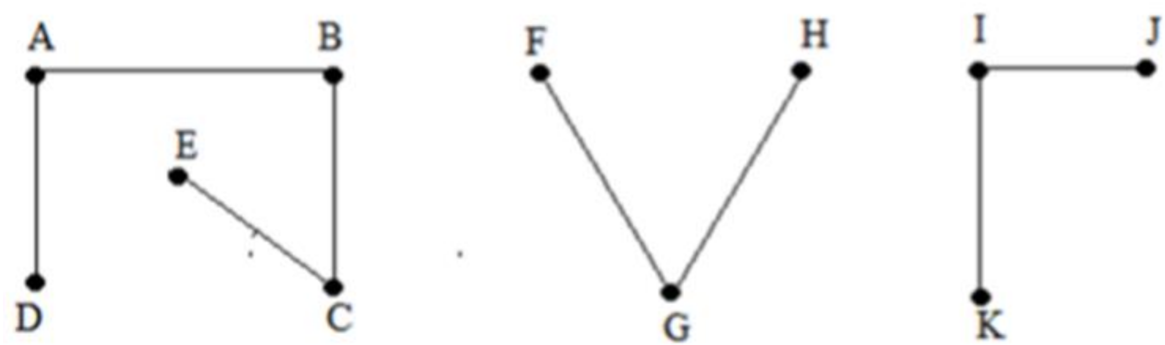
Ví dụ:



G1, G3 là cây nhưng G2 không là cây vì có chu trình ABC

- *Rừng*: là đồ thị vô hướng và không có chu trình; hay rừng là một đồ thị có nhiều thành phần liên thông mà mỗi thành phần liên thông là một cây.

Ví dụ:



G_4

G4 là một rừng và có 3 thành phần liên thông

Định lý Daisy-chain

Giả sử T là một đồ thị có n đỉnh. Khi đó, 6 mệnh đề sau là tương đương:

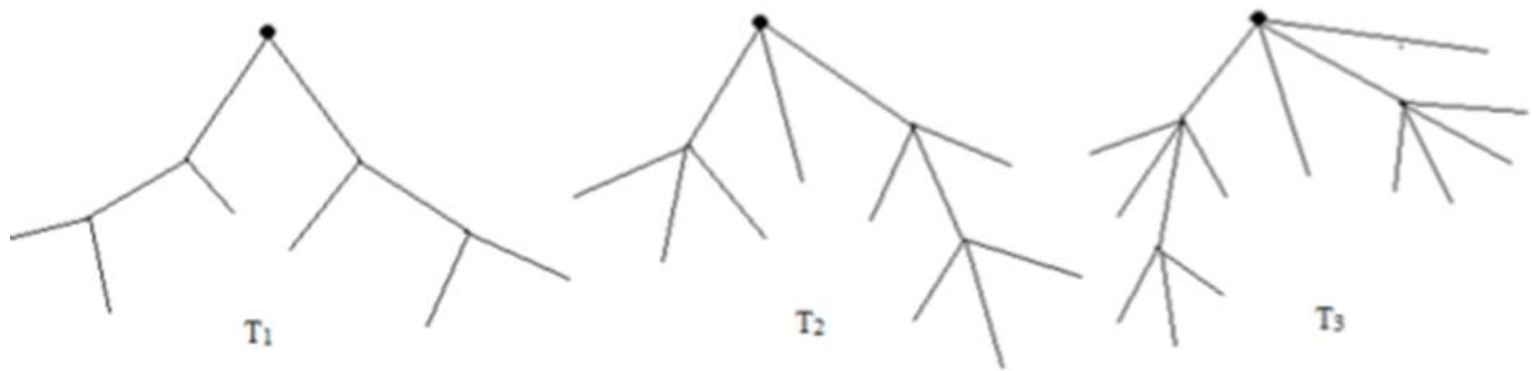
- i. T là một cây,
- ii. T không có chu trình và có n - 1 cạnh.
- iii. T là một đồ thị liên thông và nếu hủy bất kỳ một cạnh nào của nó cũng làm mất tính liên thông.
- iv. Giữa 2 đỉnh bất kỳ của T, luôn tồn tại một đường đi sơ cấp duy nhất nối 2 đỉnh này.
- v. T không có chu trình và nếu thêm một cạnh mới nối 2 đỉnh bất kỳ của T thì sẽ tạo ra một chu trình.
- vi. T liên thông và có n - 1 cạnh

Tính chất của cây

- *Cây có gốc*: Trong một cây, ta chọn một đỉnh đặc biệt gọi là gốc của cây và *chọn hướng các cạnh trên cây từ gốc đi ra* thì *ta được một đồ thị có hướng gọi là cây có gố*c. Tùy theo các gốc của đỉnh mà ta chọn ban đầu sẽ thu được các cây có gốc khác nhau.
- *Cây m–phân*: 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 của nó có đúng m – con*.

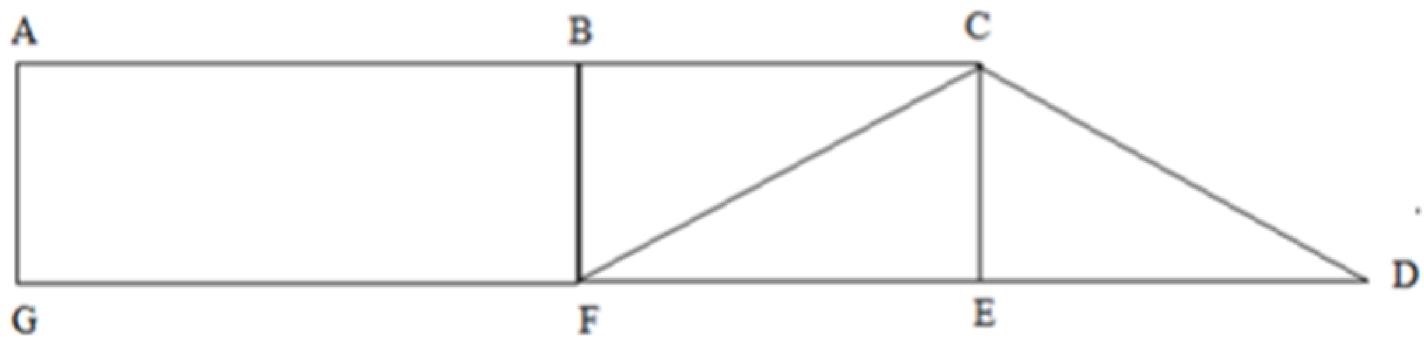
Ví dụ

- Cho các cây với T1 là cây nhị phân đầy đủ, T2 là cây tam phân đầy đủ, T3 là cây tứ phân không đầy đủ.

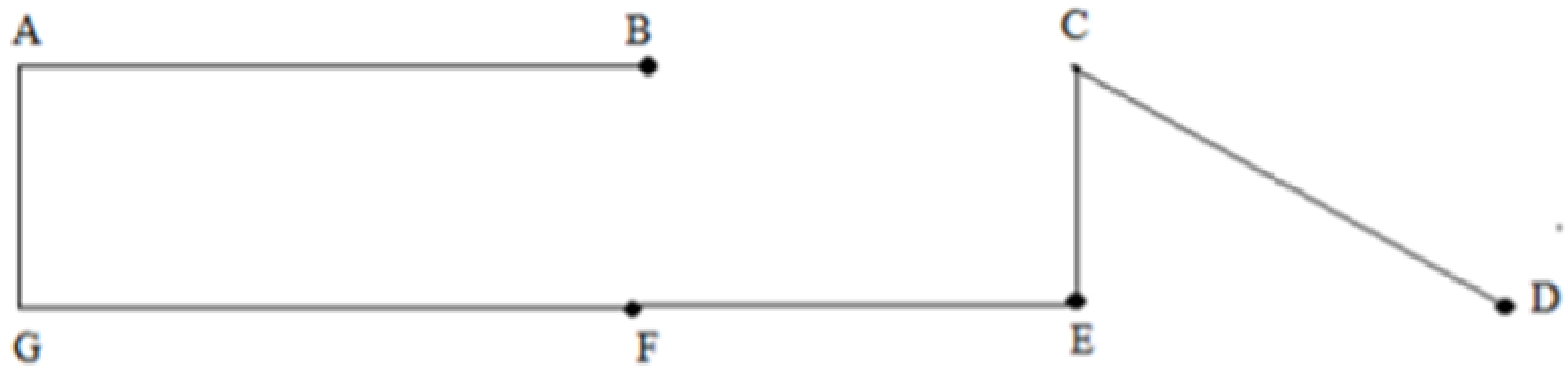


Cây khung tối thiểu

- *Một cây khung*: *là cây* của một đơn đồ thị G nếu nó là *đồ thị con của G và chứa tất cả các đỉnh của G*.
- Ví dụ: cho đồ thị



- Cây khung T chứa tất cả các đỉnh của G. Cây T là đồ thị con của G. Ta thấy T được tạo ra bằng cách xóa các cạnh của G tạo nên các chu trình đơn

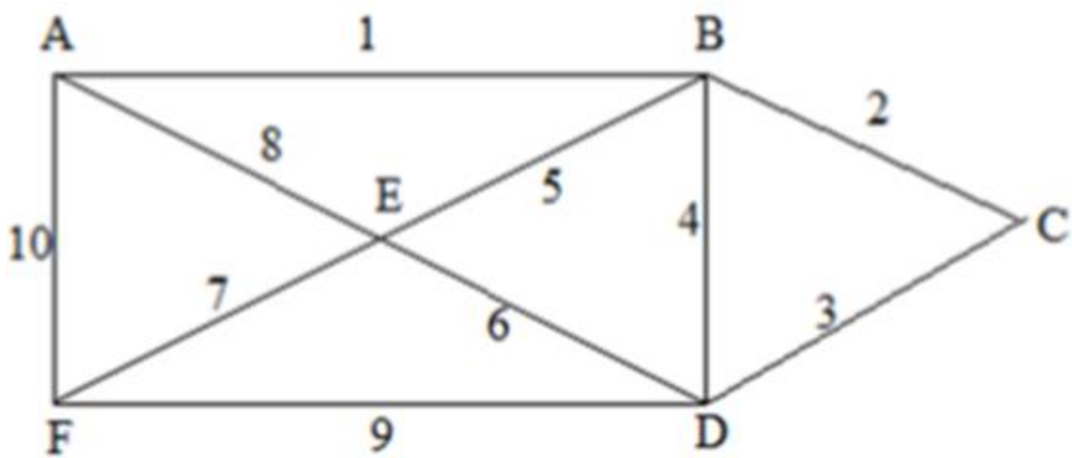


Thuật toán tìm cây khung tối thiểu

- Cho đồ thị có trọng số trên các cạnh, một cây khung tối thiểu là *cây khung có tổng trọng số nhỏ nhất*.
- *Thuật toán Prim:* tìm cây khung tối thiểu của đồ thị có n đỉnh như sau:
  - Chọn một đỉnh tùy ý và xét các đỉnh kề với đỉnh này
  - Chọn cạnh có *trọng số nhỏ nhất kề với đỉnh* tùy ý ban đầu đặt vào cây khung.
  - Tiếp tục ghép các cạnh có trọng số tăng dần, nếu một cạnh đặt vào *cây khung mà tạo ra chu trình thì bỏ đi cạnh đó*. Thuật toán *dừng lại khi (n – 1) cạnh được chọn*.

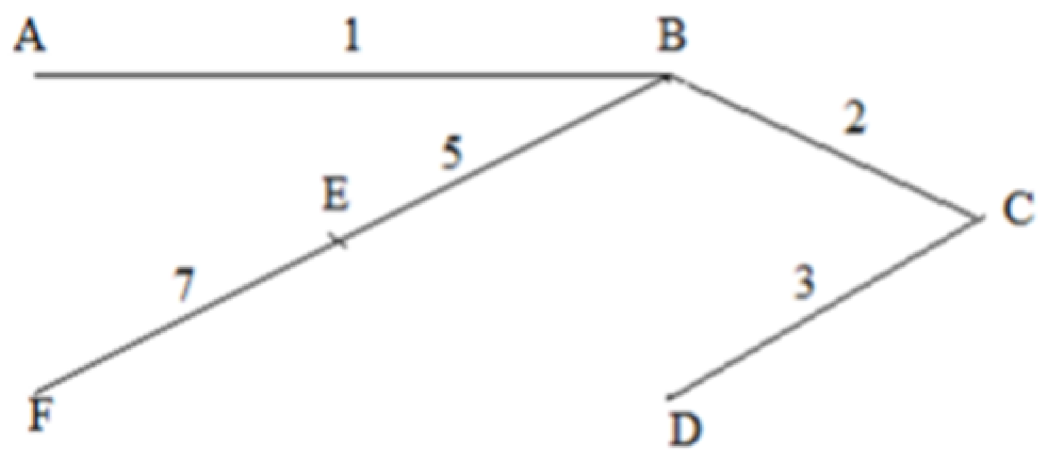
Ví dụ

Tìm cây khung tối thiểu của đồ thị G với trọng số như sau:



Chọn đỉnh tùy ý ban đầu là A. Đỉnh A có các đỉnh kề là B, E, F. Chọn cạnh AB đưa vào cây khung vì có trọng số nhỏ nhất

- Xét hai đỉnh A và B. Chọn cạnh kề với hai đỉnh này với trọng số nhỏ nhất là đỉnh C.
- Tiếp theo xét 3 đỉnh A, B và C. Chọn đỉnh C với cạnh CD vì có trọng số nhỏ nhất.
- Tiếp tục như vậy ta được cây khung AB, BC, CD, BE, EF, ta được cây khung tối thiểu T như sau:

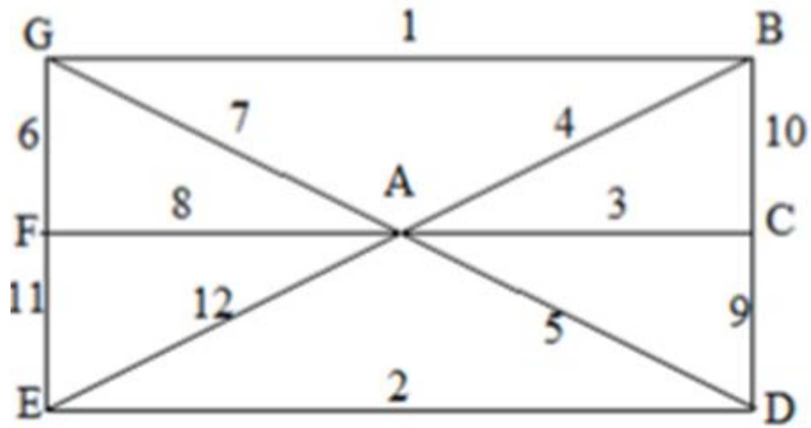


- Tổng trọng số của cây khung tối thiểu T là: 1 + 2 + 3 + 5 + 7 = 18

Thuật toán Kruskal

- Dùng xác định cây khung tối thiểu của đồ thị có n đỉnh
  - Lần lượt chọn các cạnh có trọng số từ nhỏ đến lớn để ghép vào cây khung. Khi cạnh được ghép mà tạo ra chu trình thì ta loại bỏ nó. Thuật toán kết thúc khi (n – 1) cạnh được chọn.

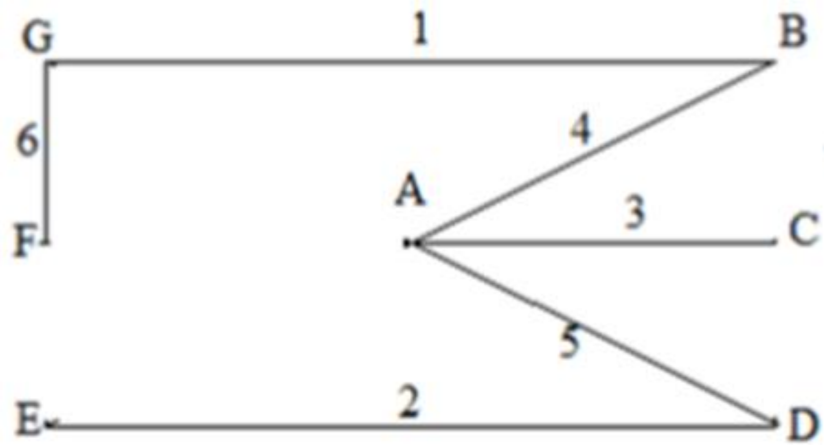
- Ví dụ: Tìm cây khung tối thiểu của đồ thị sau



Sắp xếp các cạnh theo trọng số tăng dần: BG, ED, AC, AB, AD, FG, AG, AF, CD, BC, EF, AE.

Chọn cạnh có trọng số nhỏ nhất đưa vào cây khung: BG, ED, AC, AB, AD, GF.

- Ta được cây khung tối thiểu như sau



- Tổng trọng số: 1 + 2 + 3 + 4 + 5 + 6 = 21.
- So sánh sự khác nhau giữa thuật toán Prim và Kruskal. Với Prim ta nối với những đỉnh của cạnh đã chọn để cây đang được tạo ra có liên thông. Trong trường hợp Kruskal, ta không nhất thiết phải chọn cạnh tiếp theo phải nối với các đỉnh của cạnh đã chọn.

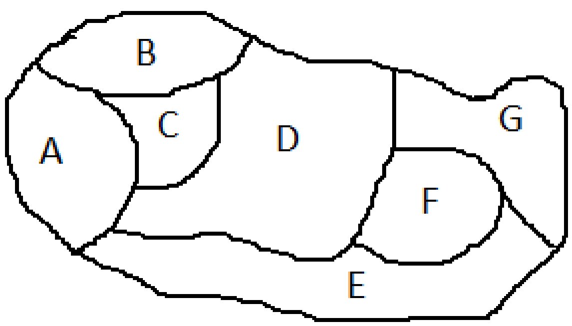
Các bài toán về đồ thị

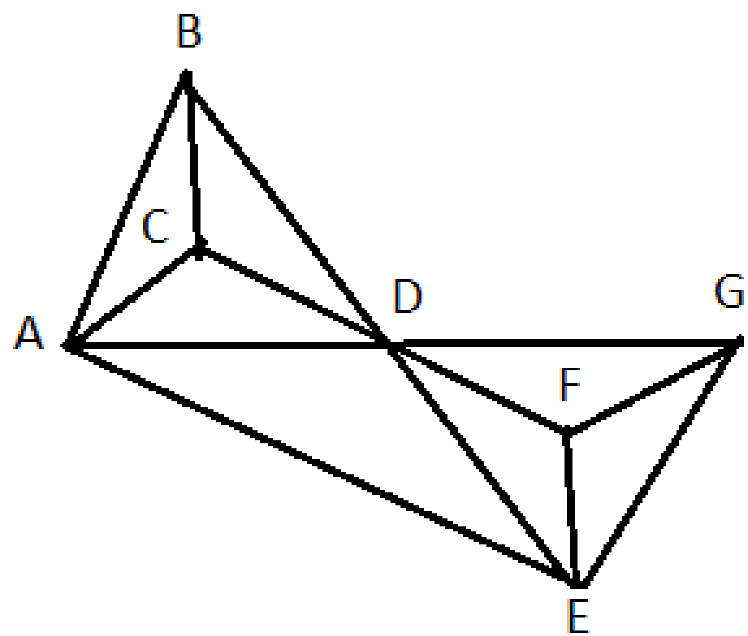
Một số bài toán phổ biến và được ứng dụng rộng rãi của đồ thị là bài toán tô màu và bài toán tìm đường đi.

- *Bài toán tô màu đồ thị:* tô màu một bản đồ của hai nước có chung một biên giới phải tô bằng hai màu khác nhau. Sao cho số màu cần dùng là ít nhất.

Các bài toán về đồ thị (tt)

- *Bài toán đối ngẫu:* Lập sự tương ứng mỗi miền của bản đồ là một đỉnh của đồ thị, mỗi cạnh nối hai đỉnh là có chung biên giới. Khi đó bản đồ trên mặt phẳng biểu diễn bằng đồ thị.
- Ví dụ:





Định nghĩa

- Tô màu đồ thị đơn là sự gán màu cho các đỉnh của nó sao cho không có hai đỉnh liền kề được gán cùng một màu.
- Số màu của đồ thị số tối thiểu các màu cần thiết để tô màu đồ thị G. Ký hiệu: $\chi(G)$.

Các định lý

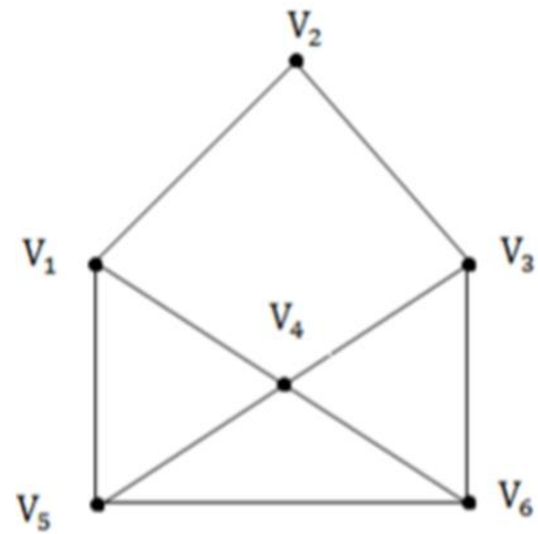
- Định lý 1: mọi *đồ thị đầy đủ phẳng* K_n *ta luôn* có $\chi(k_n) = n$.
- Định lý 2: (Appel – Haken) *đồ thị phẳng đều có số màu <= 4*.
- Định lý 3: Mọi *chu trình độ dài lẻ* đều có số *màu tô là 3*.
- Định lý 4: Một đơn đồ thị G = (V, E) có thể được tô *bằng hai màu khi nó không có chu trình độ dài lẻ*.

Thuật toán Welch-Powell:

Dùng để tô màu đồ thị G = (V, E):

- Sắp xếp *bậc các đỉnh V của G theo thứ tự giảm dần*.
- Dùng *một màu đầu tiên để tô đỉnh đầu tiên* và *cũng dùng màu này để tô màu các đỉnh tiếp theo trong danh sách mà không kề với nó*.
- Tiếp theo trở lại danh sách, *tô màu thứ hai cho đỉnh chưa được tô và lặp lạ*i các bước trên cho đến khi tất cả các đỉnh đều được tô màu.

- Ví dụ: Dùng thuật toán Welch-Powell để tô màu đồ thị sau:

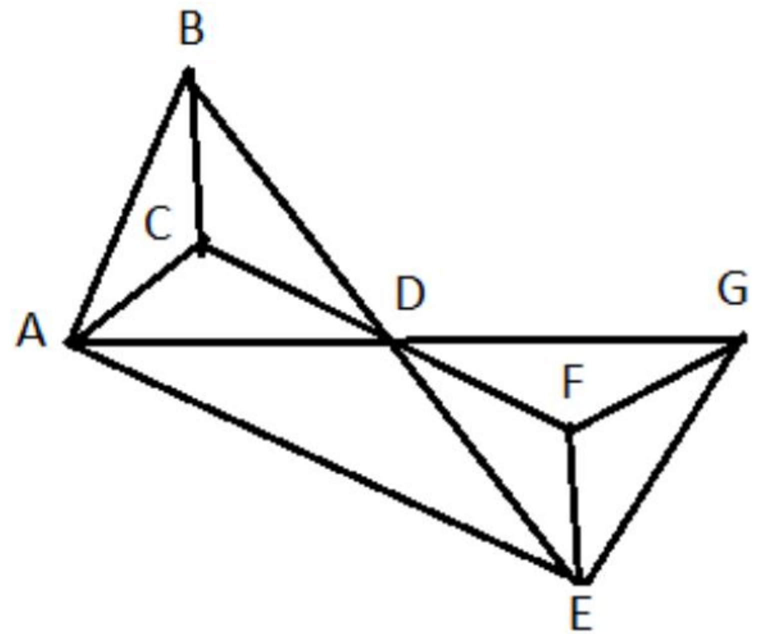


Theo thuật toán Welch-Powell, ta có bảng sau:

Bậc của đỉnh đồ thị	V_4	V_6	V_5	V_1	V_3	V_2
Số bậc	4	3	3	3	3	2
Màu dùng để tô	a	b	c	b	c	a

Ví dụ:

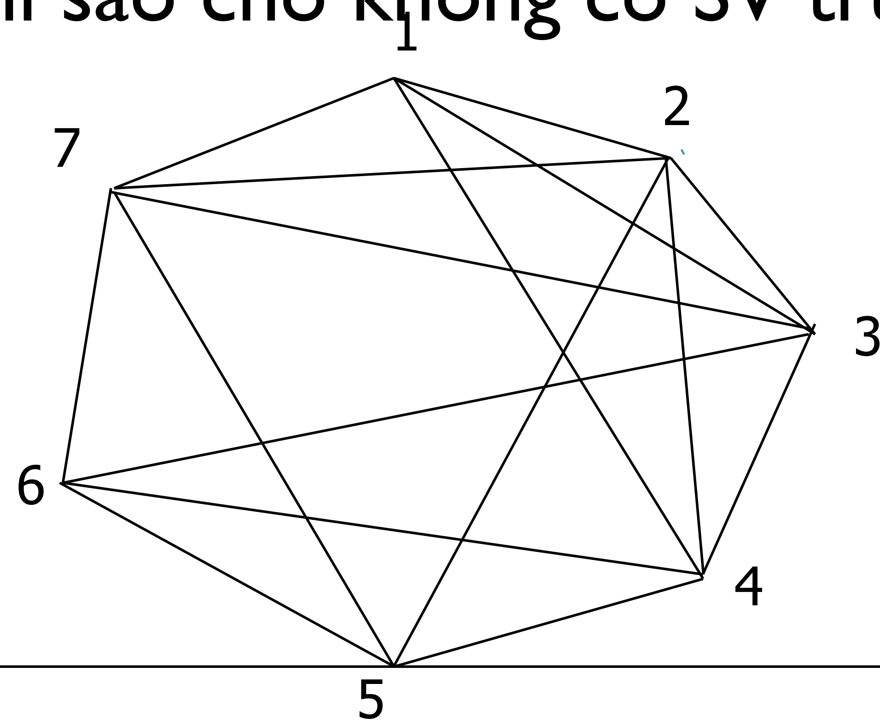
- Dùng thuật toán Welch-Powell để tô màu đồ thị sau :



Các đỉnhBF	D	A	E	C	B	F	G
Số bậc	6	4	4	3	3	3	3
Dùng màu tô	a	b	c	d	c	d	b

Ví dụ: có 7 môn cần xếp lịch thi, được đánh số từ 1 đến 7 môn thi, trong đó các cặp môn thi có chung SV: 1 và 2, 1 và 3, 1 và 4, 1 và 7, 2 và 3, 2 và 4, 2 và 5, 2 và 7, 3 và 4, 3 và 6, 3 và 7, 4 và 5, 4 và 6, 5 và 6, 5 và 7, 6 và 7.

Ta lập lịch thi sao cho không có SV trùng lịch thi



- Dùng thuật toán Welch-Powell để tô màu đồ thị

Đỉnh	2	3	4	7	1	5	6
Bậc	5	5	5	5	4	4	4
Màu tô	I	II	III	III	IV	II	I

Như vậy: dùng 4 màu tô đồ thị tương ứng tổ chức 4 đợt thi cho 7 môn thi để SV không bị trùng lịch thi.

Đợt thi	Môn thi
I	2, 6
II	3, 5
III	4, 7
IV	1

Bài toán tim đường đi ngắn nhất

- Bài toán tìm đường đi ngắn nhất giữa 2 đỉnh của đồ thị liên thông có trọng số. Trong thực tế tìm đường đi ngắn nhất giữa 2 địa điểm trong thành phố, giữa hai đỉnh của mạng máy tính, trong ngành hàng không.

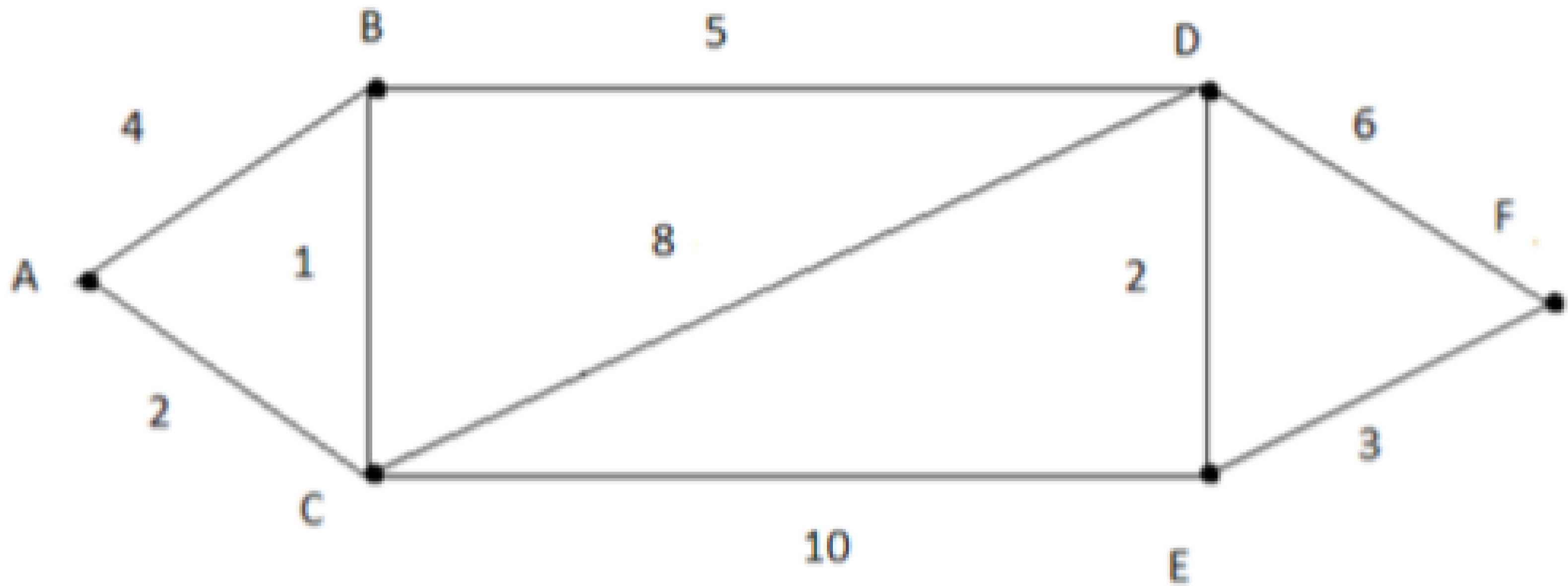
Thuật toán Dijkstra tìm đường đi ngắn nhất

- Do nhà toán học E.Dijkstra người Hà Lan đề xuất năm 1959.
- Cho đồ thị vô hướng G = (V, E) liên thông có trọng số là dương. Ta áp dụng thuật toán Dijkstra để tìm đường đi ngắn nhất từ đỉnh a đến đỉnh z là d(a, z). Thuật toán này dựa trên một dãy các bước lặp.

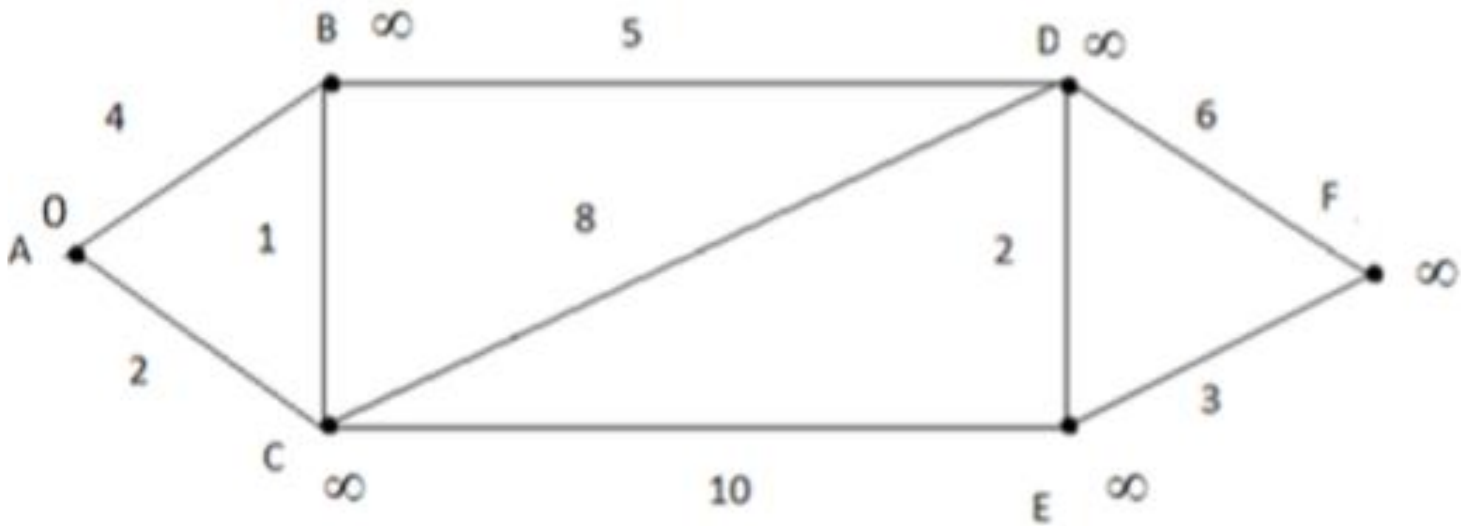
Thuật toán bao gồm các bước sau :

- i. Khởi tạo: Đặt $L(a):=0; S:=\emptyset$. Với mỗi đỉnh $v \neq a$, đặt $L(v) = \infty$
- ii. Kết thúc: Nếu $z \in S$, thuật toán dừng khi đó $L(z)$ là đường đi ngắn nhất từ a đến z
- iii. Tìm đỉnh tiếp theo: Chọn $v \in V \setminus S$ sao cho $L(v)$ có giá trị nhỏ nhất . Đặt $S: = S \cup \{v\}$
- iv. Gán nhãn cho các đỉnh: Với đỉnh x kề với y nhưng không thuộc S đặt: $L(x)=\min \{L(x), L(v)+(v, x)\}$. Quay lại bước ii.

- Ví dụ: Dùng thuật toán Dijkstra, tìm đường đi ngắn nhất từ A đến F trong đồ thị sau



- Ta có : Ở các bước lặp
  - Bước 0: V = {A,B,C,D,E,F}; S=Ø
  - Bước 1: Gán 0 cho đỉnh A → L(A) = 0 và gán ∞ cho các đỉnh còn lại.



- Trong các đỉnh không thuộc S = {A} và kề với A có 2 đỉnh B và C. Ta có:
  - $L(B) = \min\{\infty,\ L(A) + w(AB)\}$
    $= \min\{\infty, 0 + 4\} = 4.$
  - $L(C) = \min\{\infty, L(A) + w(AC)\}$
    $= \min\{\infty, 0 + 2\} = 2.$

  Ta có L(C) nhỏ nhất nên $C \in S$. $\Longrightarrow$ S = {A, C}
- Trong các đỉnh không thuộc S mà kề với C có 3 đỉnh là B, D, E.

$L(B) = \min\{4, L(C) + w(CB)\} = \min\{4, 2+1\} = 3.$
$L(E) = \min\{\infty, L(C) + w(CE)\} = \min\{\infty, 12\} = 12.$
$L(D) = \min\{\infty, L(C) + w(CD)\} = \min\{\infty, 2+8\}= 10.$
Ta có L(B) nhỏ nhất nên $B \in S$, $\Longrightarrow$ S = {A,C, B}.

- Trong các đỉnh không thuộc S mà kề với B là D.
  - $L(D) = \min\{10, L(B) + w(BD)\}$
    $= \min\{10, 3 + 5\} = 8$

  $\Longrightarrow D \in S$, vậy $S = \{A, C, B, D\}$
- Trong các đỉnh kề với D mà không thuộc S, có: E, F.
  - $L(E) = \min\{12, L(D) + w(DE)\}$
    $= \min\{12, 8+2\} = 10$
  - $L(F) = \min\{\infty, L(D) + w(DF)\}$
    $= \min\{\infty, 8+6\} = 14$

  $\Longrightarrow E \in S$, vậy $S = \{A, C, B, D, E\}$.
- Trong các đỉnh kề với E mà không thuộc S: F.
  - $L(F) = \min\{14, L(E) + w(EF)\}$
    $= \min\{14, 10+3\} = 13$.
- Vậy, đường đi ngắn nhất từ A đến F là: A, C, B, D, E, F với độ dài 13.