Đồ thị và một số bài toán trên đồ thị

Nguyễn Việt Anh

Mục tiêu

- Hiểu được một số khái niệm về lý thuyết đồ thị
- Nắm được một số thuật toán giải quyết các bài toán liên quan đến đồ thị: Đường đi ngắn nhất,Cây bao trùm ngắn nhất

Nội dung

1. Một số khái niệm
2. Đường đi ngắn nhất
    1. Đường đi ngắn nhất từ 1 đỉnh đến các đỉnh còn lại của đồ thị G
    2. Đường đi ngắn nhất giữa mọi cặp đỉnh trong đồ thị
3. Cây bao trùm ngắn nhất

1. Khái niệm

- X là một tập hữu hạn và không rỗng các phần tử nào đó và $U \subseteq X \times X$. G = (X,U) gọi là đồ thị hữu hạn.
- Mỗi phần tử $x \in X$ gọi là một đỉnh của đồ thị
- Mỗi phần tử $u = (x,y) \in U$ gọi là một cạnh của đồ thị
- X gọi là tập các đỉnh, U gọi là tập các cạnh

1. Khái niệm (tiếp…)

- G = (X,U) là đồ thị vô hướng nếu với mọi cạnh u=(x,y) $\in$ U không phân biệt thứ tự các đỉnh x và y, tức là từ x đến y không kể hướng, hay (x,y) = (y,x)
- G = (X,U) là đồ thị có hướng nếu với mọi cạnh u=(x,y) $\in$ U có phân biệt thứ tự các đỉnh x và y, có hướng x đến y, hay (x,y) $\neq$ (y,x)

1. Khái niệm (tiếp…)

- Cho G = (X,U) và u =(x,y) $\in$ U là một cạnh nối đỉnh x và y. Khi đó ta nói u là cạnh chứa đỉnh x,y hoặc x,y là các đỉnh thuộc cạnh u. Khi đó x,y được gọi là hai đỉnh kề nhau
- Hai cạnh kề nhau nếu giữa chúng có đỉnh chung.
  - Ví dụ với u=(x,y) và v =(y,z) thì u,v là hai cạnh kề nhau

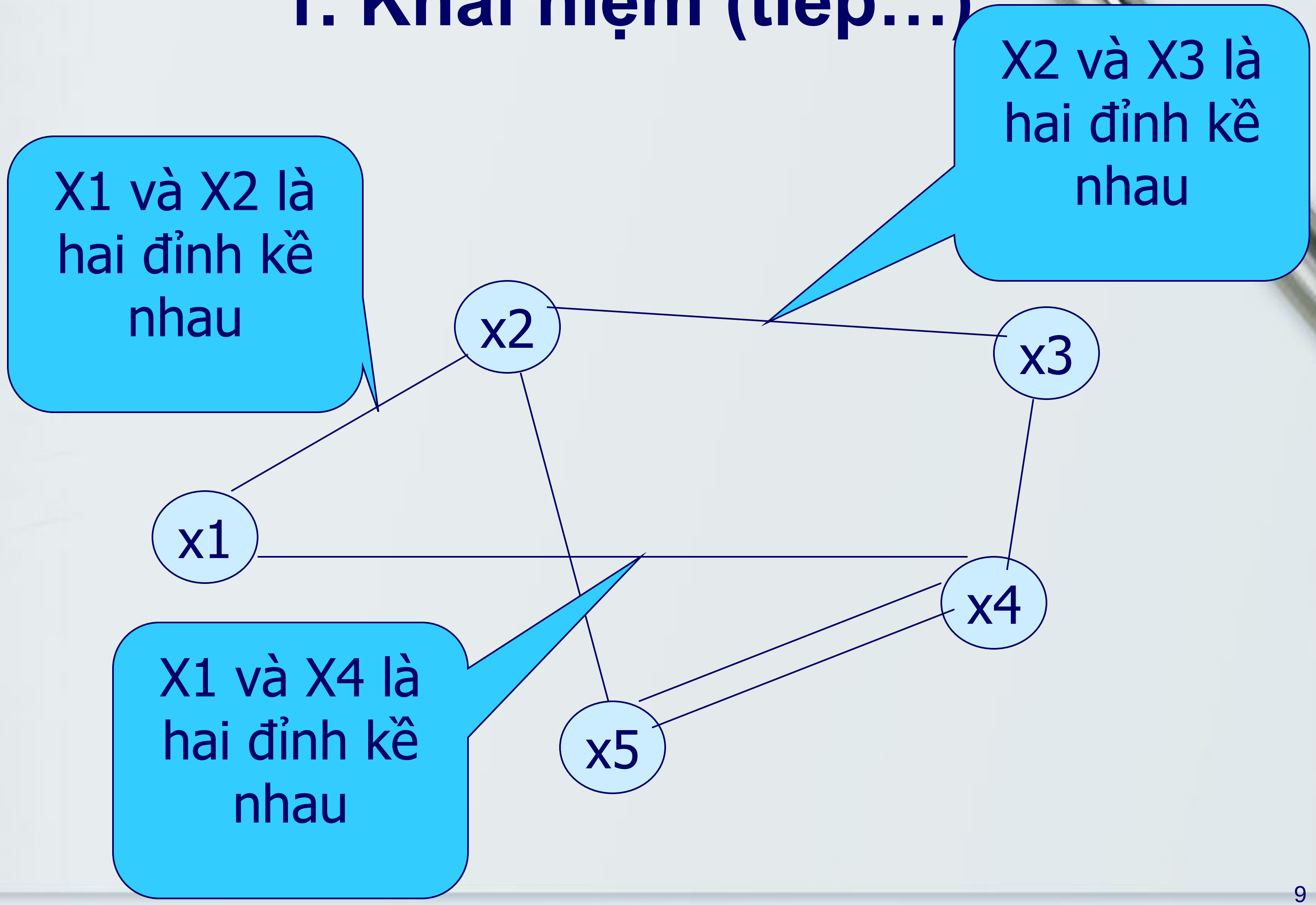
1. Khái niệm (tiếp…)

- Đồ thị G = (X,U) gọi là đồ thị đơn nếu giữa hai đỉnh bất kỳ được nối với nhau bởi không quá một cạnh
- Đồ thị G = (X,U) gọi là đa đồ thị nếu nó có ít nhất một cặp đỉnh được nối với nhau bởi hai cạnh trở lên

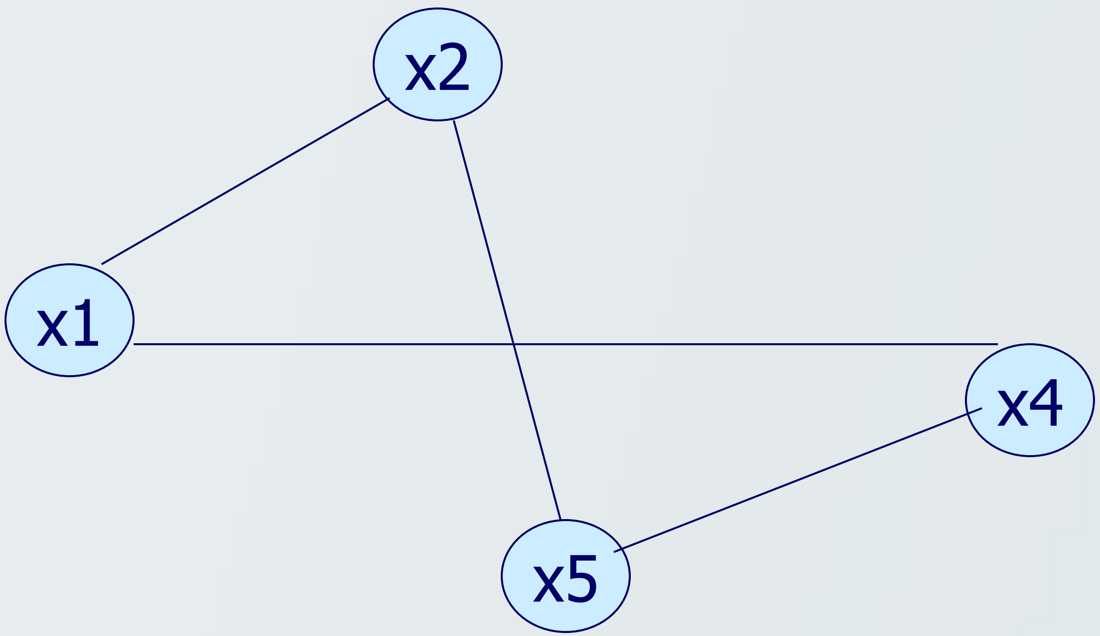
1. Khái niệm (tiếp…)

- Nếu trong đồ thị ta bỏ đi một số đỉnh nào đó và các cạnh chứa đỉnh đó thì phần còn lại của đồ thị được gọi là đồ thị con của đồ thị đã cho.
- Nếu trong đồ thị ta bỏ đi một số cạnh giữ nguyên các đỉnh thì phần còn lại của đồ thị được gọi là đồ thị bộ phận của đồ thị đã cho.

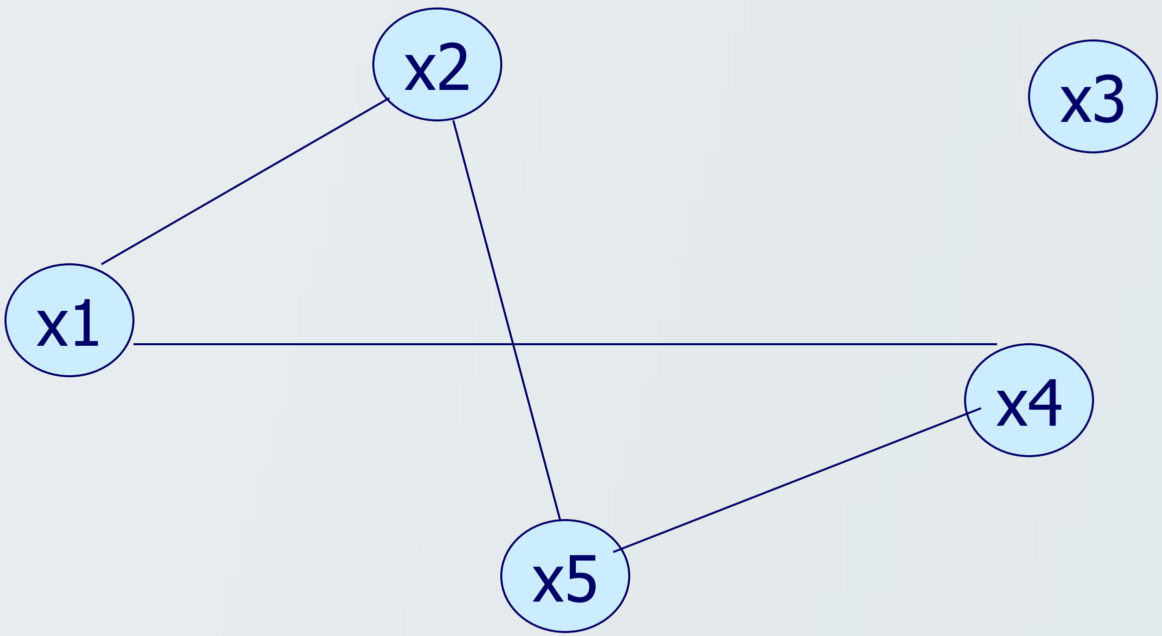
1. Khái niệm (tiếp…)



1. Khái niệm (tiếp…)



1. Khái niệm (tiếp…)



1. Khái niệm (tiếp…)

- G = (X,U) là đồ thị vô hướng (hoặc có hướng). Một đường đi trong đồ thị là một dãy $x_1 u_{i1} x_2 u_{i2} \ldots x_{ik} u_{ik} x_{ik+1}$, trong đó các x_{ij} là các đỉnh còn các u_{ij} là các cạnh sao cho với $\forall$ j $\in$ {1,2 …,k} thì đỉnh x_{ij} và đỉnh x_{ij+1} là hai đỉnh kề nhau.
- Chu trình trong đồ thị là một đường đi có đỉnh xuất phát và đỉnh kết thúc trùng nhau.

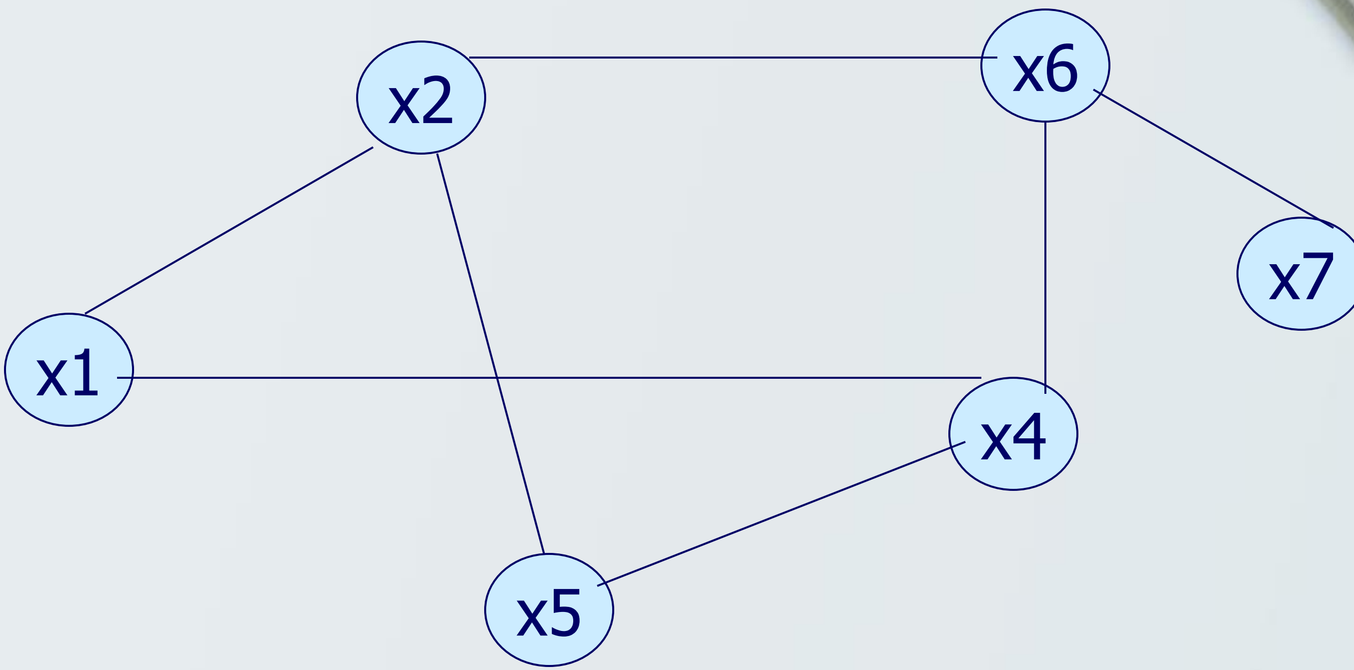
2. Biểu diễn đồ thị (tiếp…)

1. Biểu diễn hình học
2. Biểu diễn bằng ma trận kề

2.1. Biểu diễn hình học

1. Mỗi đỉnh x $\in$ X ta đặt tương ứng với mỗi điểm trên một mặt phẳng.
2. Với G =(X,U) là đồ thị vô hướng. Trong trường hợp này nếu u =(x,y) $\in$ U thì trong mặt phẳng, các đỉnh x và đỉnh y được nối với nhau bởi một cạnh không có hướng
3. Nếu (x,x) $\in$ U thì tại đỉnh x sẽ có một khuyên

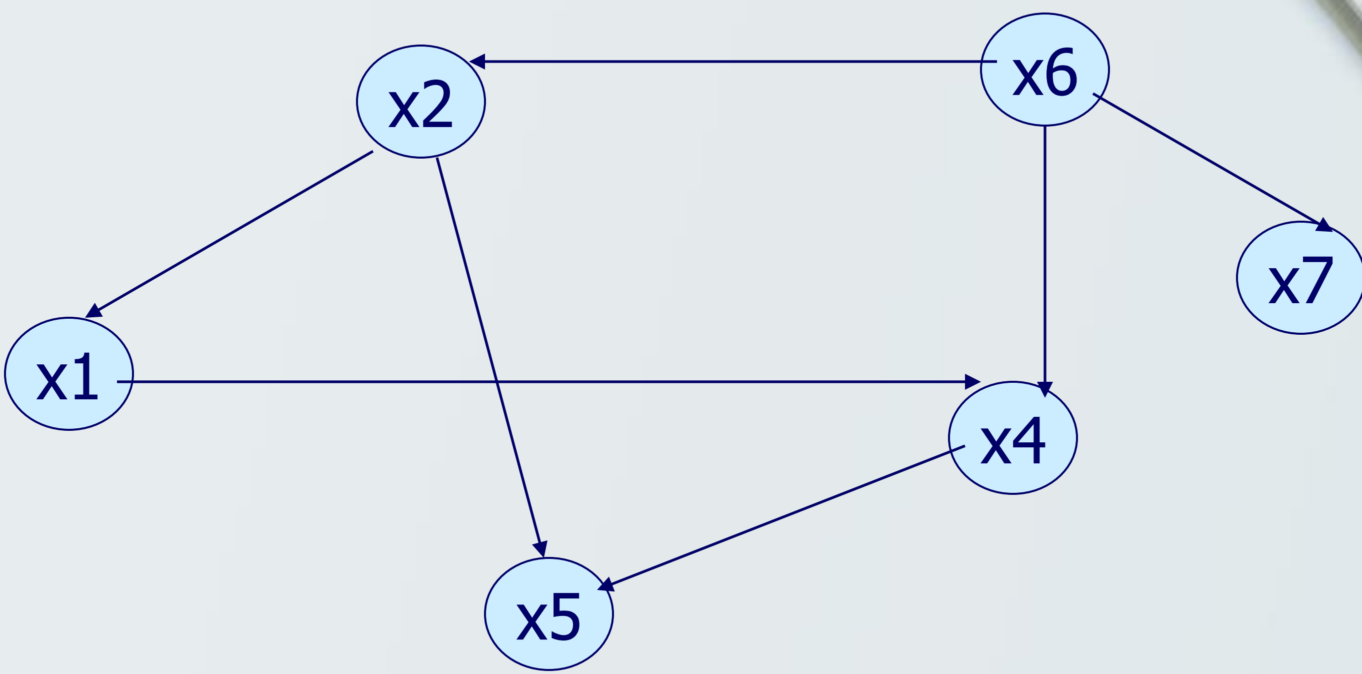
2.1. Biểu diễn hình học tiếp…)



2.1. Biểu diễn hình học (tiếp…)

1. Mỗi đỉnh $x \in X$ ta đặt tương ứng với mỗi điểm trên một mặt phẳng.
2. Với $G = (X,U)$ là đồ thị có hướng. Trong trường hợp này nếu $u = (x,y) \in U$ thì trong mặt phẳng sẽ có một cung có hướng đi từ đỉnh x đến đỉnh y
3. Nếu $(x,x) \in U$ thì tại đỉnh x sẽ có một khuyên có hướng vào chính nó

2.1. Biểu diễn hình học (tiếp…)



2.2. Biểu diễn bằng ma trận kề

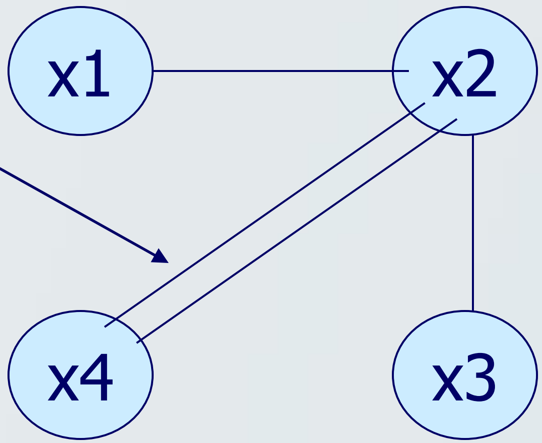
1. G =(X,U) là đồ thị vô hướng với X = {x_1,x_2, …, x_n}. Đồ thị G có thể biểu diễn bằng ma trận kề vuông cấp n mà phần tử a_{ij} ở hàng i cột j được xác đinh như sau:
   1. a_{ij} = d nếu giữa cặp đỉnh x_i và x_j có d cạnh
   2. **a_{ij} =** 0 nếu cặp đỉnh x_i và x_j không được nối với nhau

2.2 Biểu diễn bằng ma trận kề (tiếp…)

1. G =(X,U) là đồ thị có hướng với X = {x_1,x_2, …, x_n}. Đồ thị G có thể biểu diễn bằng ma trận kề vuông cấp n mà phần tử a_{ij} ở hàng i cột j được xác đinh như sau:
    1. a_{ij} = d nếu giữa cặp đỉnh x_i và x_j có d cạnh từ x_i đến x_j
    2. **a_{ij} =** 0 nếu cặp đỉnh x_i và x_j không được nối với nhau

2.2 Biểu diễn bằng ma trận kề (tiếp...)

$$
\begin{pmatrix}
0 & 1 & 0 & 0 \\
1 & 0 & 1 & 2 \\
0 & 1 & 0 & 0 \\
0 & 2 & 0 & 0
\end{pmatrix}
$$



3. Đường đi ngắn nhất

1. Đồ thị có trọng số:

   Đó là các đồ thị mà mỗi cạnh (u,v) được gắn với một số c(u,v). Số c(u,v) được gọi là trọng số của cung nó được gọi là giá hoặc độ dài của cạnh (u,v)

2. Đồ thị liên thông:

   1. Cho đồ thị G =(X,U). Hai đỉnh phân biệt x,y $\in$ X được gọi là liên thông nếu tồn tại một đường đi nối các đỉnh x,y với nhau.
   2. Đồ thị G gọi là liên thông nếu với hai đỉnh phân biệt bất kỳ trong đồ thị đều là liên thông

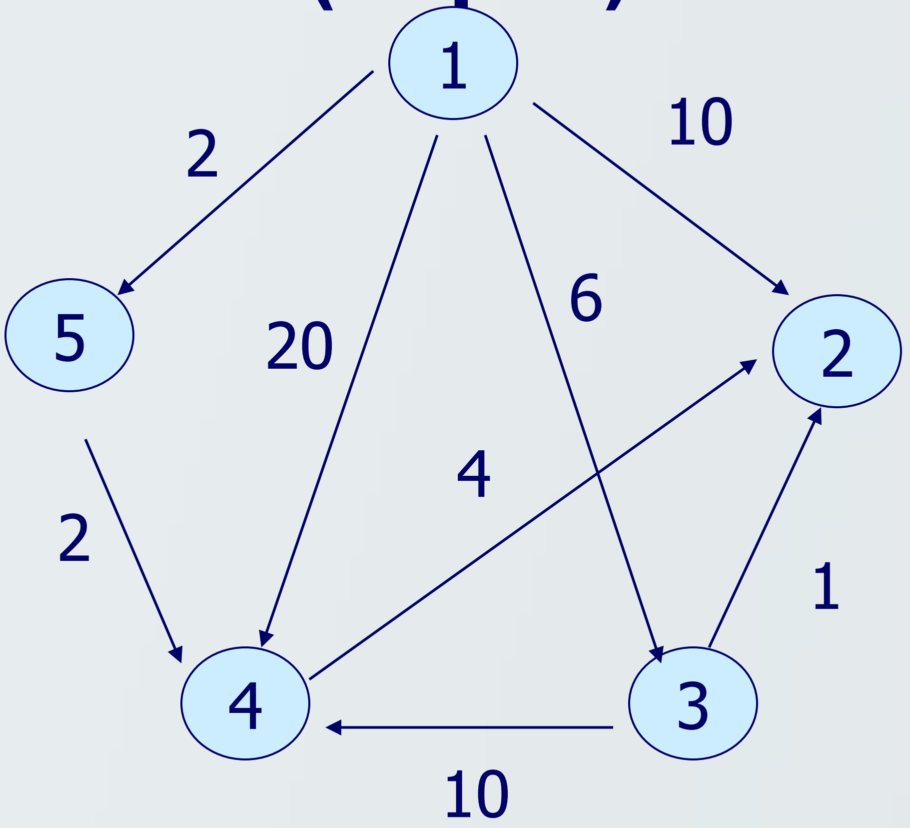
3. Đường đi ngắn nhất (tiếp…)

1. Tìm đường đi ngắn nhất từ một đỉnh nguồn tới các đỉnh các đỉnh còn lại của đồ thị
2. Tìm đường đi ngắn nhất giữa một cặp đỉnh của đồ thị

3. Đường đi ngắn nhất từ một đỉnh nguồn - Thuật toán Dijkstra (tiếp…)

1. Bài toán:
   1. Cho đồ thị G =(X,U) đơn liên thông có trọng số
   2. Tìm đường đi ngắn nhất từ đỉnh nguồn ∈X đến tất cả các đỉnh còn lại.
   3. Các đỉnh x được đánh số 1 tới n
   4. Đồ thị được biểu diễn bởi ma trận kề C[1..n,1..n], trong đó C[i,j] là độ dài cung (i,j) nếu không có cung (i,j) thì C[i,j] = ∞

3. Đường đi ngắn nhất từ một đỉnh nguồn - Thuật toán Dijkstra (tiếp…)



3. Đường đi ngắn nhất từ một đỉnh nguồn - Thuật toán Dijkstra (tiếp…)

Đường đi : 1,5,4,2

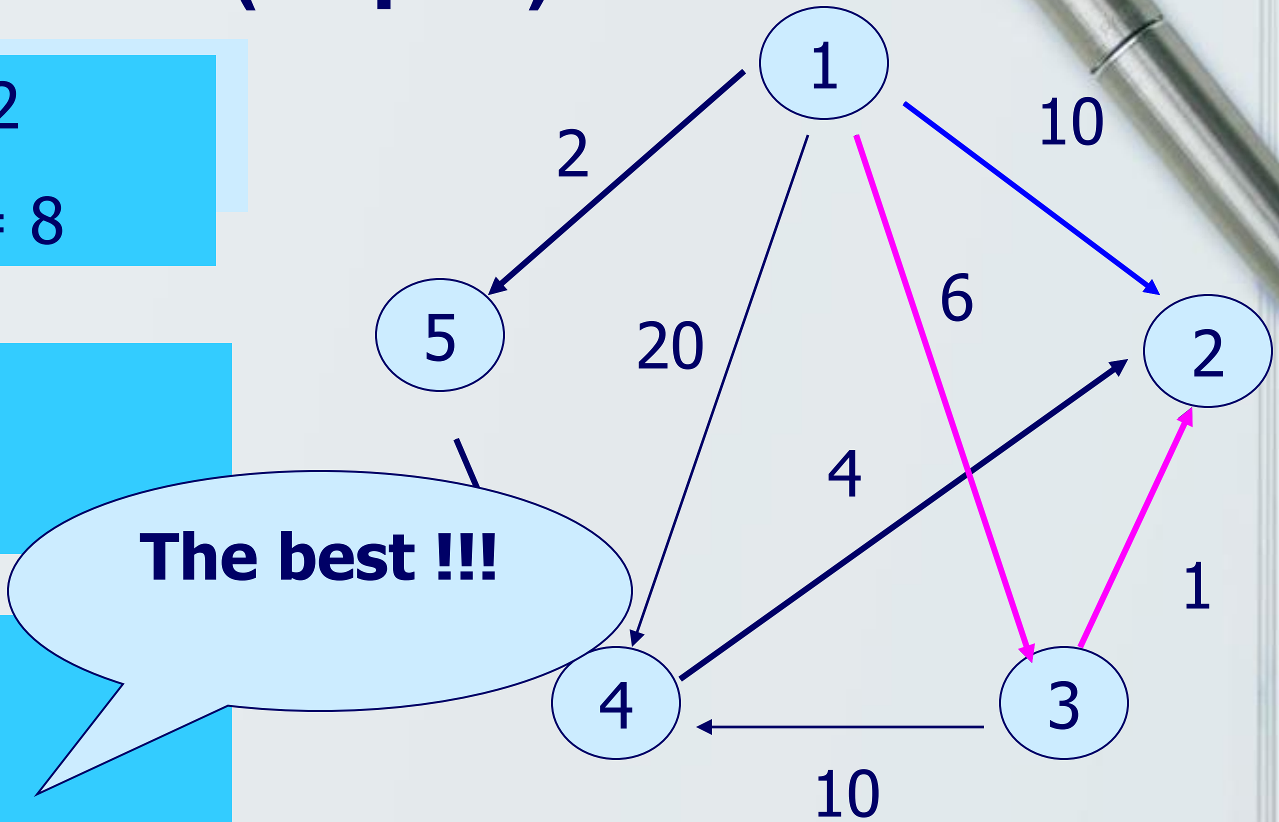
Độ dài : 2+2+4 = 8

Đường đi : 1,2

Độ dài : 10

Đường đi : 1,3,2

Độ dài : 6+1 =7

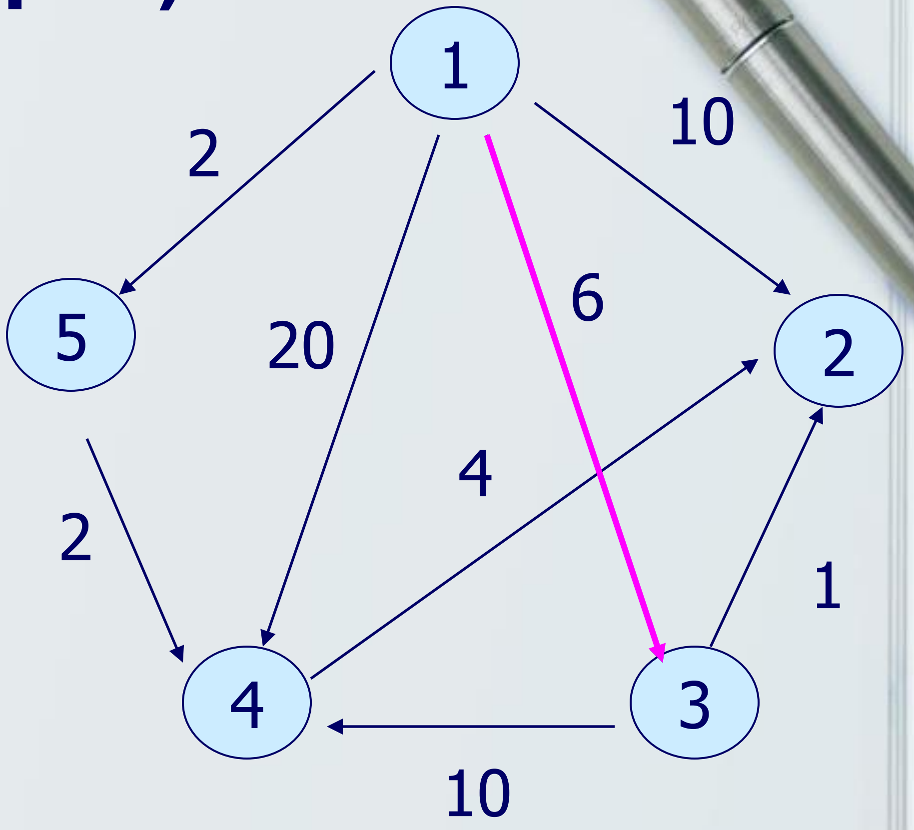


3. Đường đi ngắn nhất từ một đỉnh nguồn - Thuật toán Dijkstra (tiếp…)

Tìm đường đi từ đỉnh 1 đến 3

Đường đi : 1,3

Độ dài : 6



3. Đường đi ngắn nhất từ một đỉnh nguồn - Thuật toán Dijkstra (tiếp…)

Tìm đường đi từ đỉnh 1 đến 4

Đường đi : 1,4

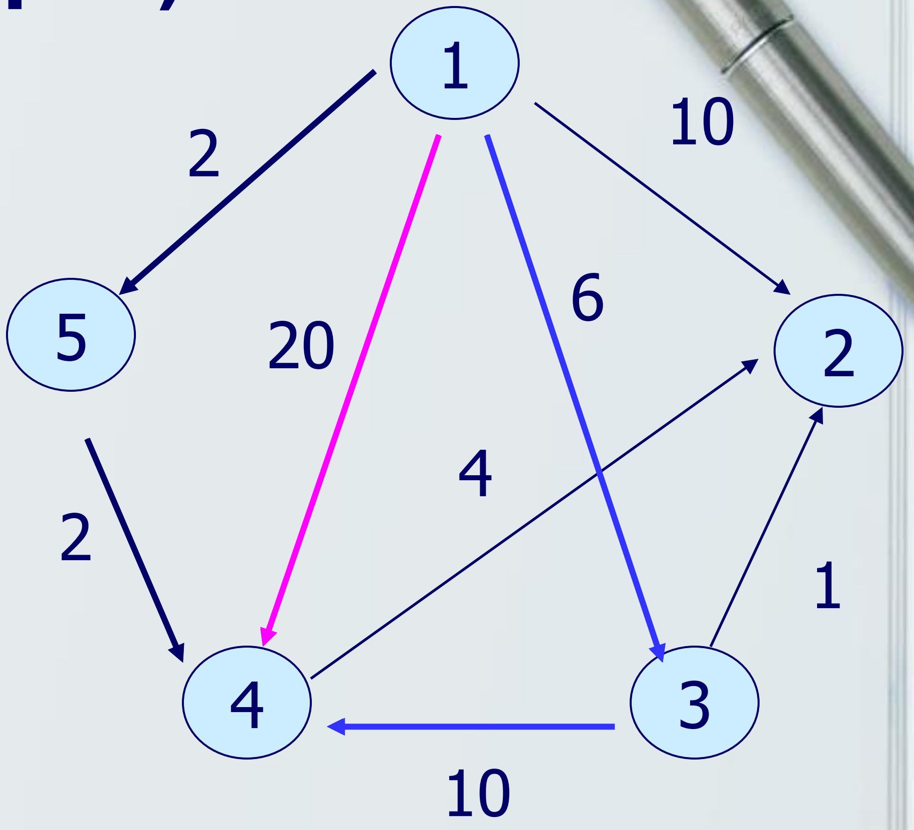
Độ dài : 20

Đường đi : 1,3,4

Độ dài : 6+10 =16

Đường đi : 1,5,4

Độ dài : 2+2 =4



3. Thuật toán Dijkstra (tiếp…) – Giải thuật

1. Gán trọng số các đỉnh
   1. Trọng số của đỉnh xuất phát là t(a) = 0
   2. Tại các đỉnh còn lại ta ghi một trọng số dương t đủ lớn sao cho giá trị t này lớn hơn trọng số của các đỉnh từ a tới.

3. Thuật toán Dijkstra (tiếp…) – Giải thuật

2. Thực hiện việc giảm trọng số các đỉnh
   1. Giả sử tại đỉnh x có trọng số t(x)
   2. Nếu tồn tại đỉnh y có trọng số t(y) từ y đến x mà **t(x) > t(y) + l(y,x)** thì ta thay trọng số t(x) bởi trọng số **t’(x) = t(y) + l(y,x)**
   3. Trong trường hợp t(x) < t(y) + l(y,x) ta giữ nguyên giá trị t(x)
   4. Quá trình này lặp lại cho đến khi trọng số của tất cả trong G=(X,U) đạt giá trị cực tiểu nghĩa là với mọi x thuộc X không tồn tại y thuộc X kề với x mà t(y) + l(y,x) < t(x)

3. Thuật toán Dijkstra (tiếp…) – Giải thuật

1. Ta xác định đường đi ngắn nhất từ đỉnh nguồn tới các đỉnh còn lại qua các bước, mỗi bước ta xác định đường đi ngắn nhất từ đỉnh nguồn tới một đỉnh trong đồ thị
2. Lưu các đỉnh mà đường đi ngắn nhất tới chúng đã được xác định vào tập S
3. Ban đầu tập S chỉ chứa đỉnh nguồn
4. Ta gọi các đường đi từ đỉnh nguồn tới các đỉnh u khác nhưng chỉ đi qua các đỉnh nằm trong S là đường đi đặc biệt

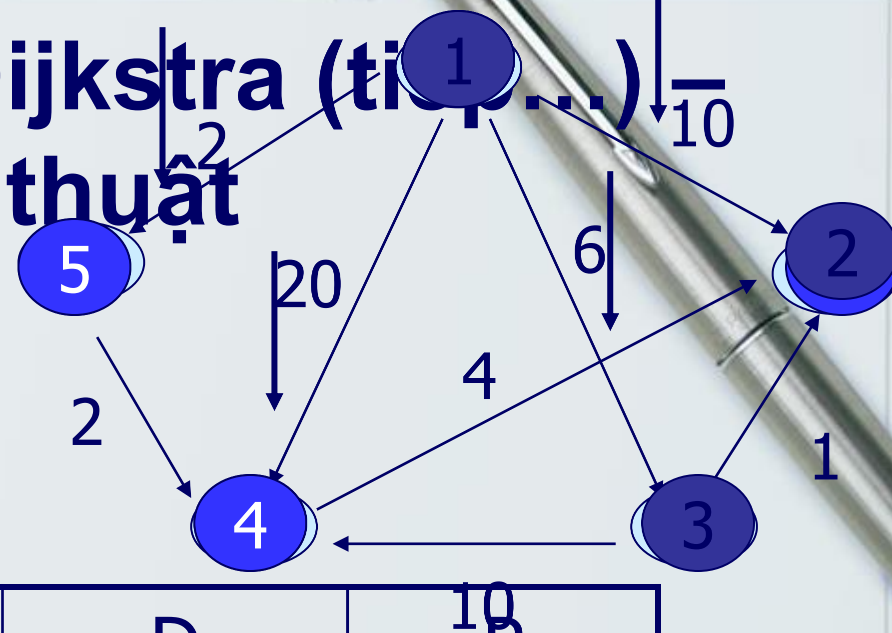
3. Thuật toán Dijkstra (tiếp…) – Giải thuật

5. Dùng mảng D[2..n], trong đó D[u] lưu độ dài đường đi đặc biệt ngắn nhất từ đỉnh nguồn tới u
6. Ban đầu S chứa đỉnh 1 (đỉnh nguồn), ta xác định D[u] = C[1,u] (u =2,…n)
7. Tại mỗi bước ta sẽ chọn một đỉnh x $\in$ X-S mà D[x] là nhỏ nhất và thêm x vào S
8. Sau khi thêm x vào S, ta xác định lại các D[y] với y $\notin$ S. Nếu độ dài đường đi đặc biệt qua đỉnh x (vừa được chọn) tới y mà nhỏ hơn D[y] thì ta lấy D[y] làm độ dài đường đi đó

3. Thuật toán Dijkstra (tiếp…) – Giải thuật

9. Khi S = X thì D[u] sẽ lưu độ dài đường đi ngắn nhất từ 1 tới u với u =2…n
10. Để lưu vết của đường đi ta sử dụng mảng P[2..n], trong đó P[y] =x nếu ta đi tới y theo cung (x,y)

3. Thuật toán Dijkstra (tiếp...) – Giải thuật



Nếu độ dài đường đi đặc biệt qua đỉnh x (vừa được chọn) tới y mà nhỏ hơn D[y] thì ta lấy D[y] làm độ dài đường đi đó

1->2 : qua đỉnh 3 Độ dài 7

1->3: Độ dài 6

1->4 : qua đỉnh 5 Độ dài 4

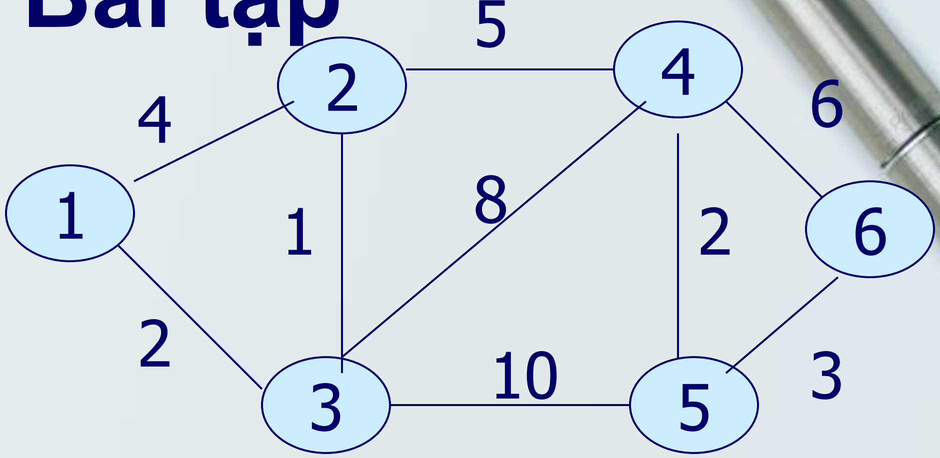
1->5: Độ dài 2

Bước	x	X-S	D	P
			[10,6,20,2]	[1,1,1,1]
			[10 ,6 ,4 ,2]	[1,1,5,1]
			[8 ,6 ,4 ,2]	[4,1,5,1]
3	3	{2}	[7 ,6 ,4 ,2]	[3,1,5,1]

3. Thuật toán Dijkstra (tiếp…) – Giải thuật

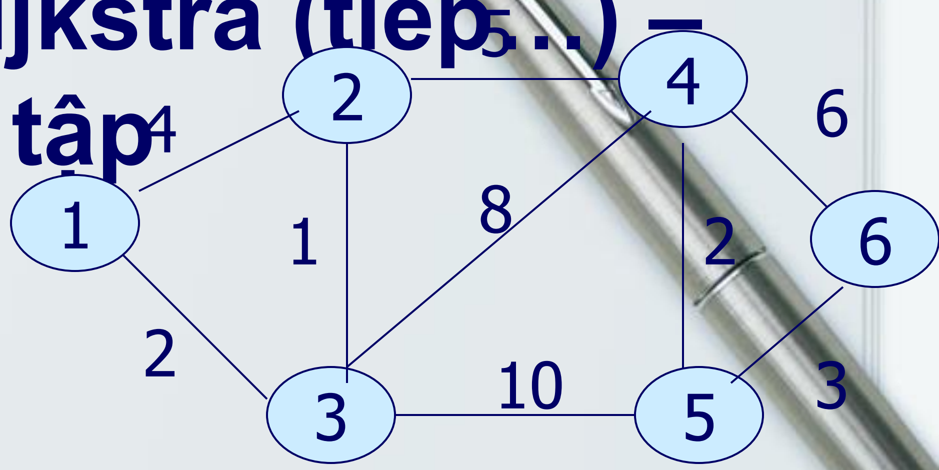
- Disjktra;
  1. S = {1};
  2. For i = 2 to n do
     1. D[i] = C[1,i];
     2. P[i]=1;
  3. While X-S $\neq \emptyset$ do
     1. Chọn x $\in$ X- S mà D[x] nhỏ nhất;
     2. S =S $\cup$ {x}
  4. For y $\in$ X- S do
     1. If D[x] +C[x,y] < D[y] then
        1. D[y] = D[x] +C[x,y];
        2. P[y]=x;

3. Thuật toán Dijkstra (tiếp…) – Bài tập



Bước	x	X-S	D	P
1				
2				
3				
4				
5				
6				

3. Thuật toán Dijkstra (tiếp…) – Bài tập



Bước	x	X-S	D	P
1	1	{2,3,4,5,6}	[4,2,∞, ∞, ∞]	[1,1, 1,1,1]
2	3	{2,4,5,6}	[3,2,10,12, ∞]	[3,1,3,3, 1]
3	2	{4,5,6}	[3,2,8,12, ∞]	[3,1,2,3, 1]
4	4	{5,6}	[3,2,8,10,15]	[3,1,2,4,4]
5	5	{6}	[3,2,8,10,13]	[3,1,2,4,5]
6	6	∅		

4. Đường đi ngắn nhất giữa mọi cặp đỉnh: Thuật toán Floyd

1. Bài toán:
   1. Cho đồ thị G =(X,U) đơn liên thông có trọng số
   2. Tìm đường đi ngắn nhất giữa mọi cặp đỉnh x,y trong đồ thị
   3. Các đỉnh x được đánh số 1 tới n
   4. Đồ thị được biểu diễn bởi ma trận kề C[1..n,1..n], trong đó C[i,j] là độ dài cung (i,j) nếu không có cung (i,j) thì C[i,j] = ∞

4. Đường đi ngắn nhất giữa mọi cặp đỉnh: Thuật toán Floyd (tiếp…)

2. Nhận xét:
    1. Chúng ta có thể giải quyết bài toán này bằng thuật toán Dijkstra với các đỉnh nguồn lần lượt là 1..n
    2. Thuật toán Floyd giải quyết trực tiếp vấn đề trên

4. Đường đi ngắn nhất giữa mọi cặp đỉnh: Thuật toán Floyd (tiếp…)

2. Nhận xét:
   1. Nếu đỉnh k nằm trên đường đi ngắn nhất từ đỉnh i tới đỉnh j thì các đường đi từ i tới k và từ k tới j phải là đường đi ngắn nhất từ i tới k và từ k tới j tương ứng

4. Đường đi ngắn nhất giữa mọi cặp đỉnh: Thuật toán Floyd (tiếp…)

3. Giải thuật:
   1. Sử dụng ma trận A để lưu độ dài đường đi ngắn nhất giữa mọi cặp đỉnh.
   2. Ban đầu ta đặt A[i,j] = C[i,j], có nghĩa là ban đầu A chứa độ dài đường đi trực tiếp các đỉnh x đến y thuộc đồ thị mà không đi qua đỉnh nào cả.

4. Đường đi ngắn nhất giữa mọi cặp đỉnh: Thuật toán Floyd (tiếp…)

3. Sau đó ta thực hiện n lần lặp. Sau lần lặp thứ k, ma trận A sẽ chứa độ dài các đường đi ngắn nhất chỉ đi qua các đỉnh thuộc {1,2,…,k}
4. Do đó sau n bước lặp ta nhận được ma trận A chứa độ dài các đường đi ngắn nhất

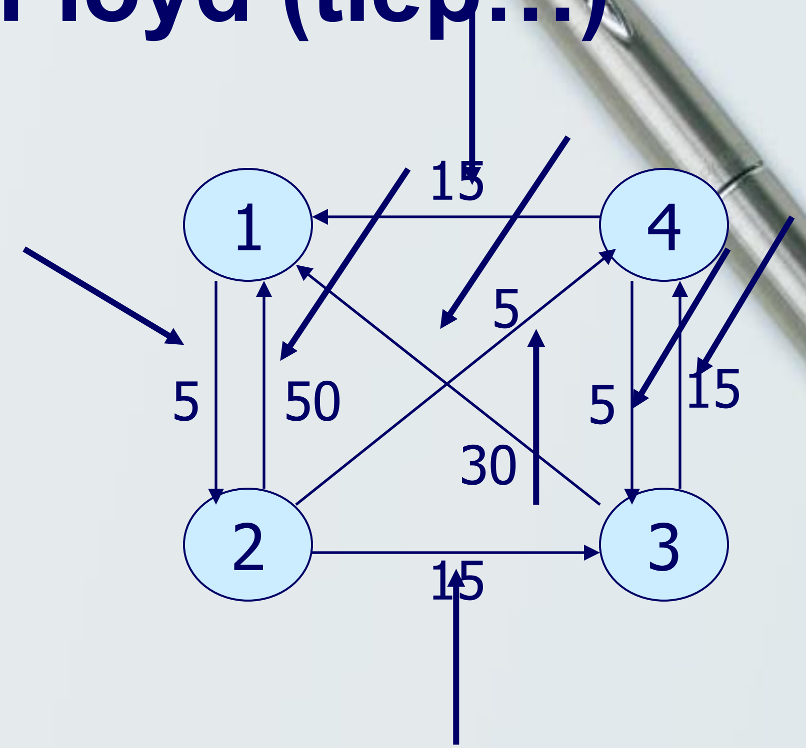
4. Đường đi ngắn nhất giữa mọi cặp đỉnh: Thuật toán Floyd (tiếp…)

5. Kí hiệu A_k là ma trận A sau lần lặp thứ k, khi đó $A_k[i,j]$ được tính theo công thức sau:

   $A_k[i,j] = \min(A_{k-1}[i,j], A_{k-1}[i,k]+A_{k-1}[k,j])$

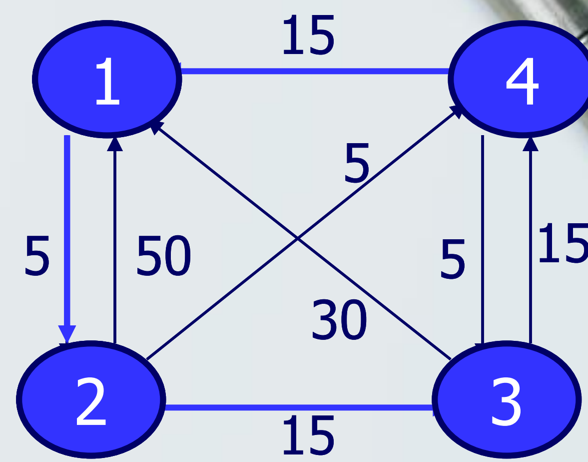
3. Thuật toán Floyd (tiếp…)

$$C = \begin{vmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 5 \\ 30 & \infty & 0 & 15 \\ 15 & \infty & 5 & 0 \end{vmatrix}$$



3. Thuật toán Floyd (tiếp…)

$$A_1 = \begin{vmatrix} 0 & 5 & \infty & \infty \\ 50 & 0 & 15 & 5 \\ 30 & [redacted] & 0 & 15 \\ 15 & 20 & 5 & 0 \end{vmatrix}$$

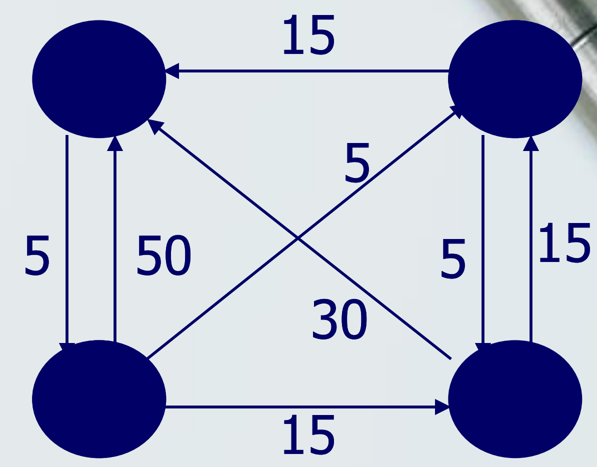


$$A_2 = \begin{vmatrix} 0 & 5 & 20 & 10 \\ 50 & 0 & 15 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{vmatrix}$$

$$A_3 = \begin{vmatrix} 0 & 5 & 20 & 10 \\ 45 & 0 & 15 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{vmatrix}$$

3. Thuật toán Floyd (tiếp…)

$$A_4 = \begin{vmatrix} 0 & 5 & 20 & 10 \\ 20 & 0 & 10 & 5 \\ 30 & 35 & 0 & 15 \\ 15 & 20 & 5 & 0 \end{vmatrix}$$

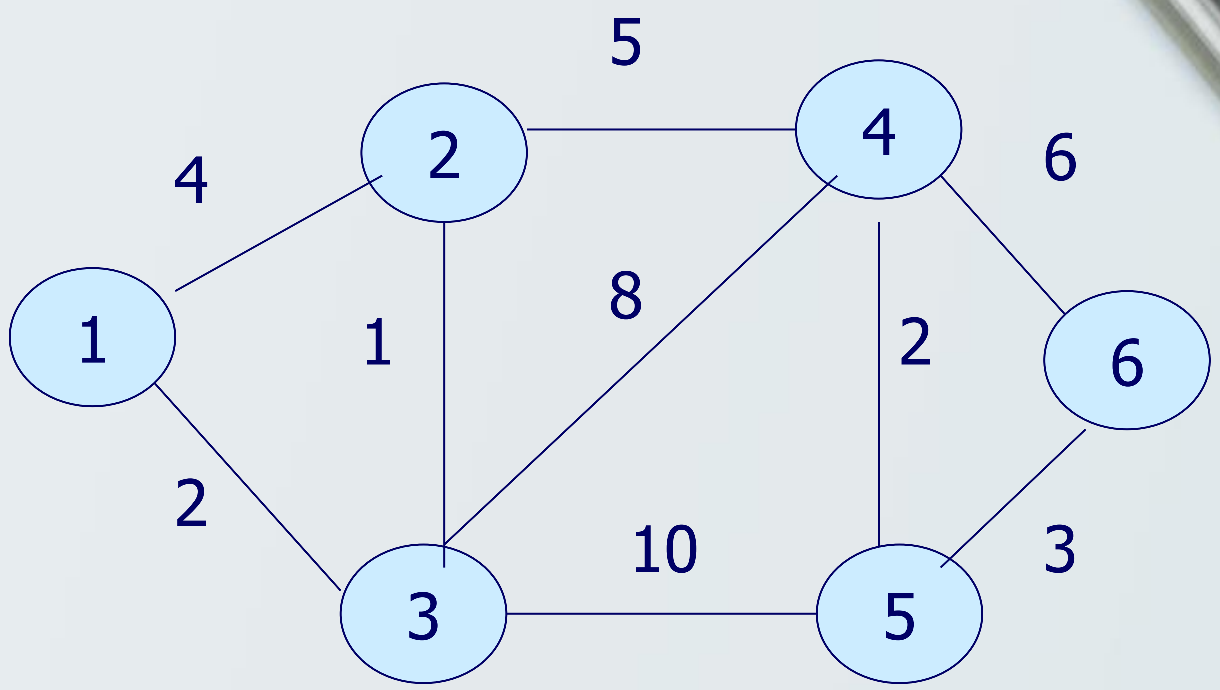


4. Đường đi ngắn nhất giữa mọi cặp đỉnh: Thuật toán Floyd (tiếp…)

Floyd

1. For i =1 to n do
    1. For j =1 to n do
        1. A[i,j] = C[i,j]
2. For k=1 to n do
    1. For i =1 to n do
        1. For j =1 to n do
            1. If A[i,k] +A[k,j] <A[i,j] then A[i,j] = A[i,k] +A[k,j]

3. Thuật toán Floyd (tiếp…) – Bài tập



Cây bao trùm ngắn nhất

1. Khái niệm cây
2. Khái niệm cây bao trùm
3. Cây bao trùm ngắn nhất
4. Thuật toán Prim tìm cây bao trùm ngắn nhất

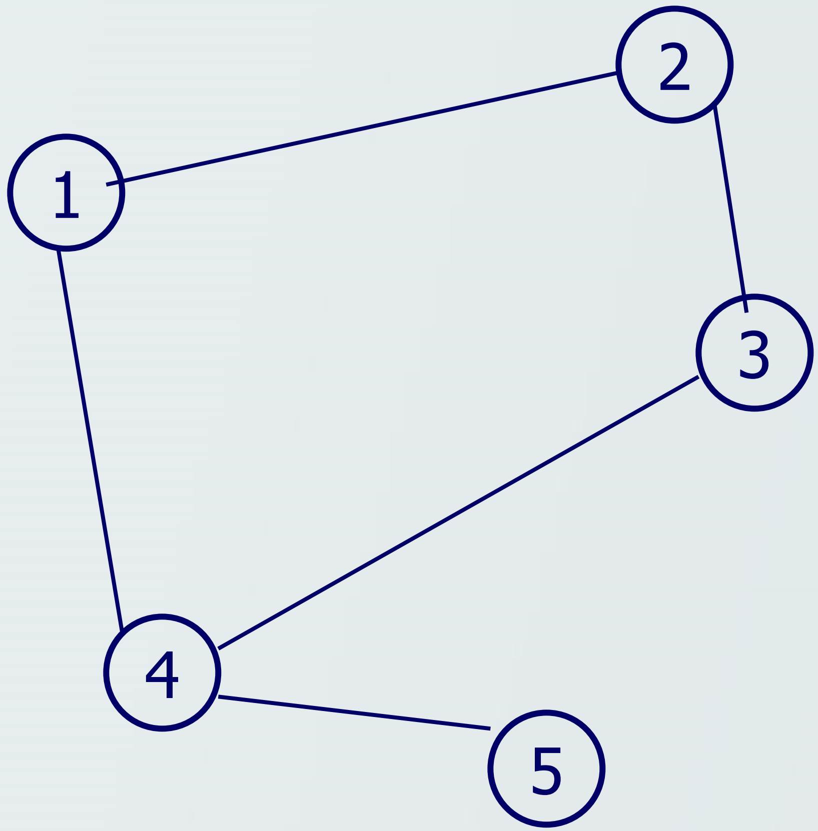
Khái niệm Cây

1. G=(X,U) gọi là cây nếu G là đồ thị liên thông và không có chu trình (X = n).
2. Khi đó 6 tính chất sau là tương đương:
   1. G là đồ thị liên thông và không có chu trình
   2. G không có chu trình và có n-1 cạnh
   3. G liên thông và có n-1 cạnh
   4. G không có chu trình và nếu thêm vào một cạnh nối hai đỉnh không kề nhau thì G xuất hiện duy nhất một chu trình
   5. G liên thông và nếu bỏ đi một cạnh tuỳ ý thì đồ thị nhận được sẽ không liên thông
   6. Mỗi cặp đỉnh trong G được nối với nhau bởi duy nhất 1 cạnh

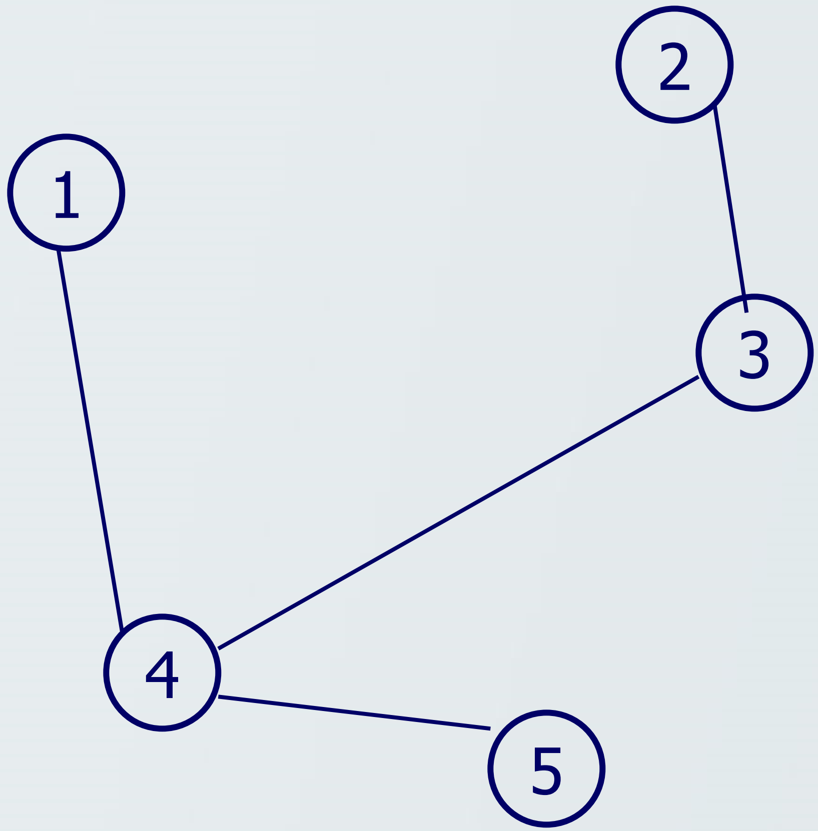
Khái niệm Cây bao trùm

1. Cho đồ thị G =(X,U). Giả sử G’ là đồ thị bộ phận của G. Nếu G’ =(X,U’) là một cây thì G’ gọi là cây bao trùm của đồ thị G.

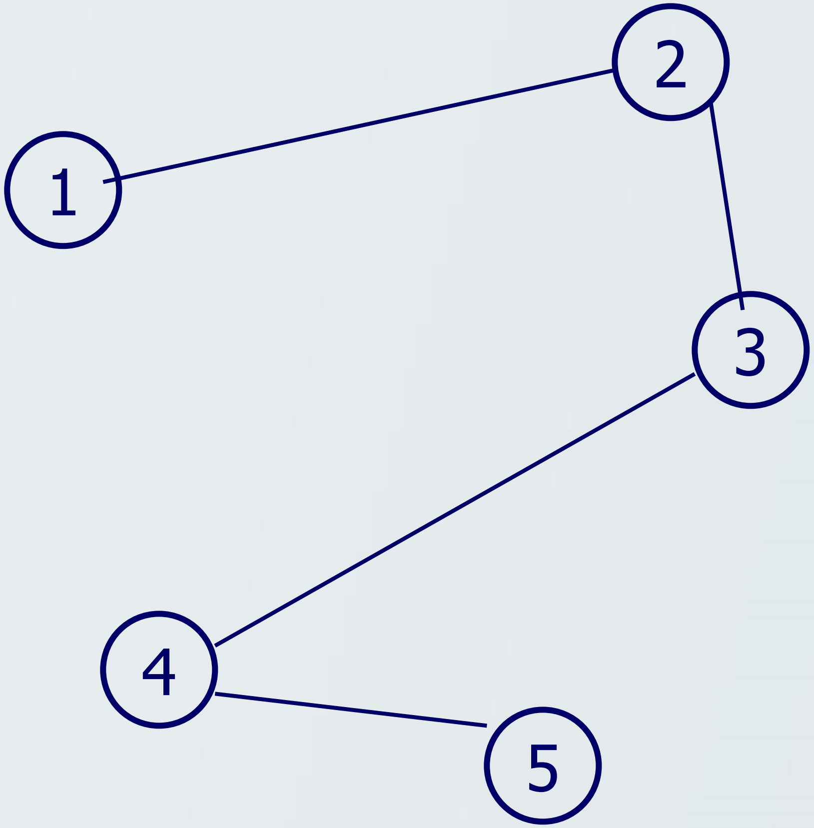
Cây bao trùm (tiếp…)



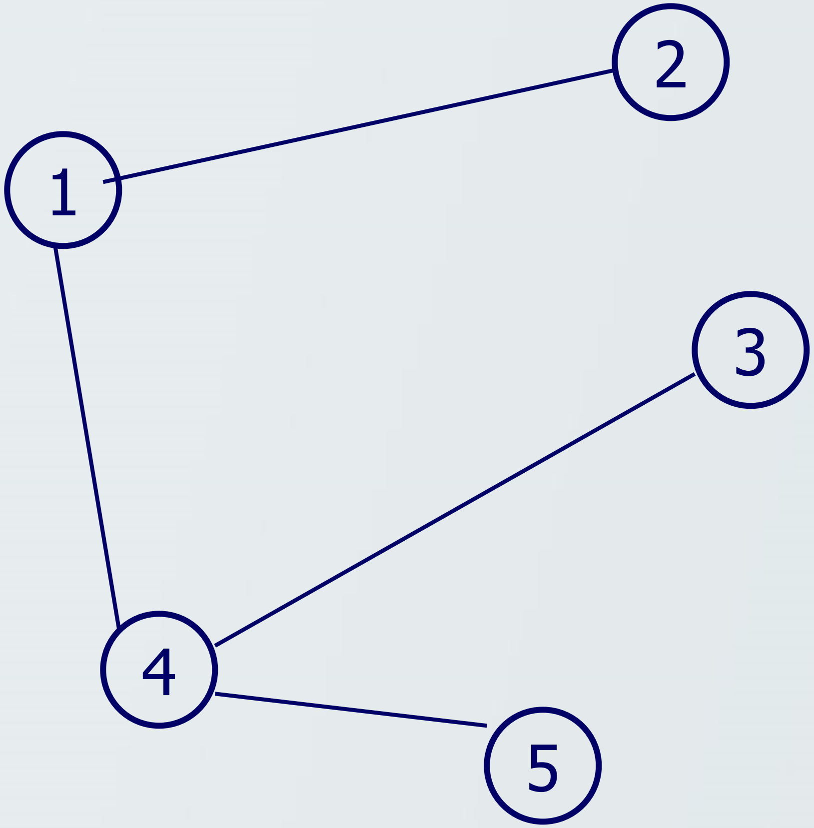
Cây bao trùm (tiếp…)



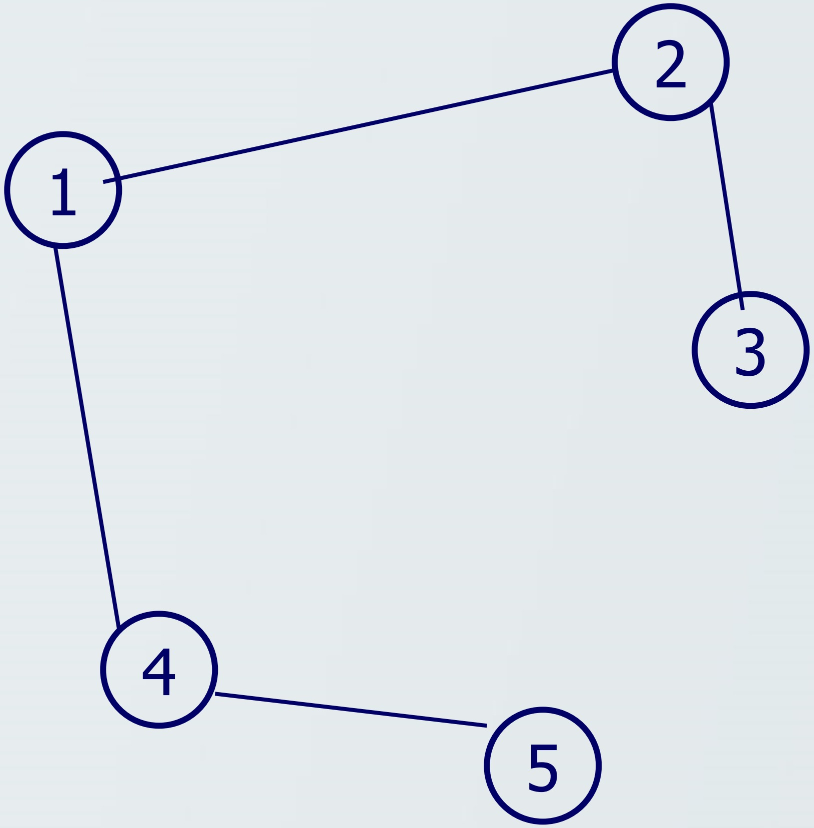
Cây bao trùm (tiếp…)



Cây bao trùm (tiếp…)



Cây bao trùm (tiếp…)



Cây bao trùm ngắn nhất

1. Cho đồ thị G =(X,U). Mỗi cạnh (x,y) thuộc U có trọng số c(u,v) >= 0
2. G’ = (X,U’) là cây bao trùm của G
3. Độ dài của cây G’ được xem là tổng trọng số của các cạnh tạo thành G’
4. Tìm G’ của G sao cho G’ có độ dài ngắn nhất

Thuật toán Prim

Cho G =(X,U) là đồ thi mà mỗi cạnh u thuộc U có trong số l(u)>=0

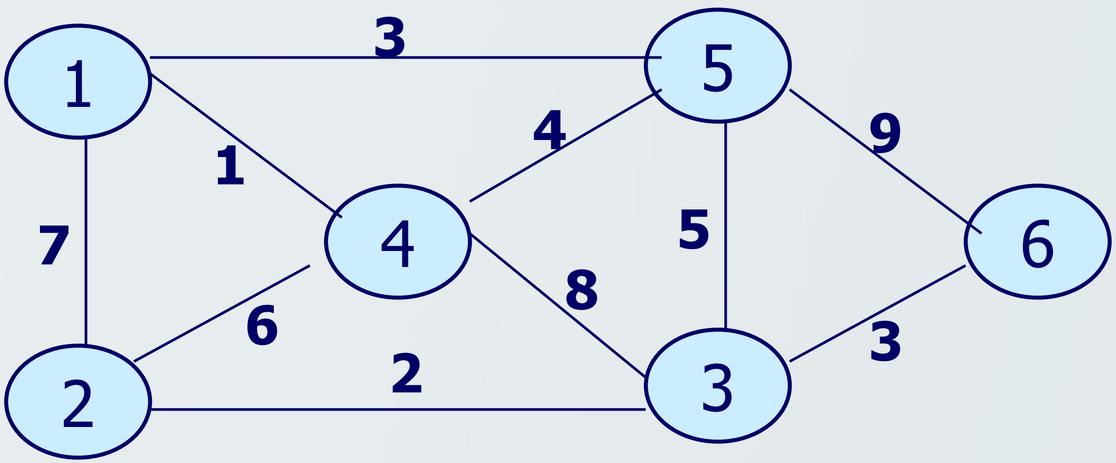
1. Xác định cạnh có trọng số bé nhất trong tất cả các cạnh trong U. Giả sử là u_1
2. Giả sử u_2 là cạnh có trọng số bé nhất trong các cạnh U \ {u_1}
3. Giả sử u_3 là cạnh có trọng số bé nhất trong các cạnh U \{u_1,u_2}. Với điều kiện {u_1,u_2,u_3} không tạo thành chu trình

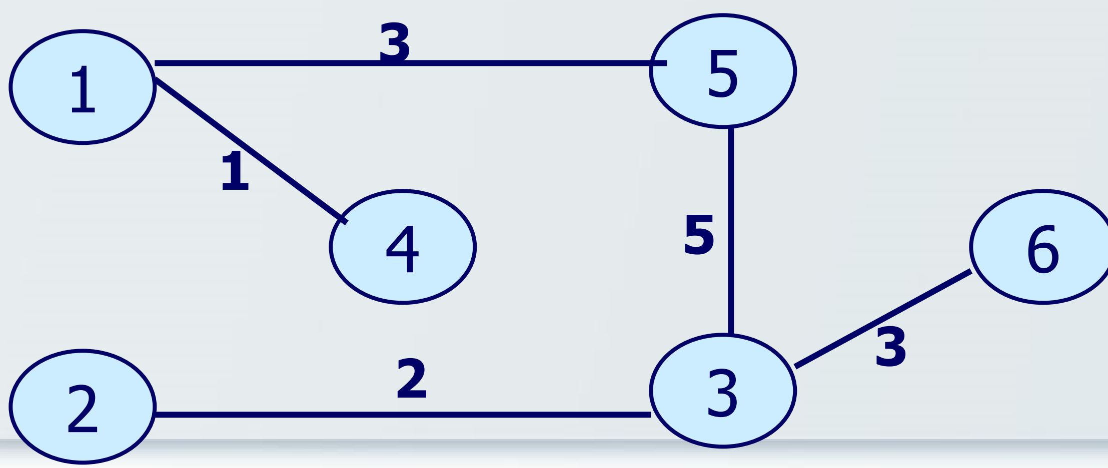
Thuật toán Prim (tiếp…)

….

- Giả sử bước k ta đã xác định được $\{u_1,u_2,u_3,\ldots,u_k\}$ có trọng số bé nhất và không tạo thành chu trình
- Thực hiện bước n-1 thì dừng lại. Khi đó ta có $\{u_1,u_2,u_3,\ldots,u_{n-1}\}$ không tạo thành chu trình và có trọng số bé nhất.
- Khi đó ta có G' =(X,U_{n-1}) là cây bao trùm bé nhất của G cần tìm

Thuật toán Prim (tiếp…)





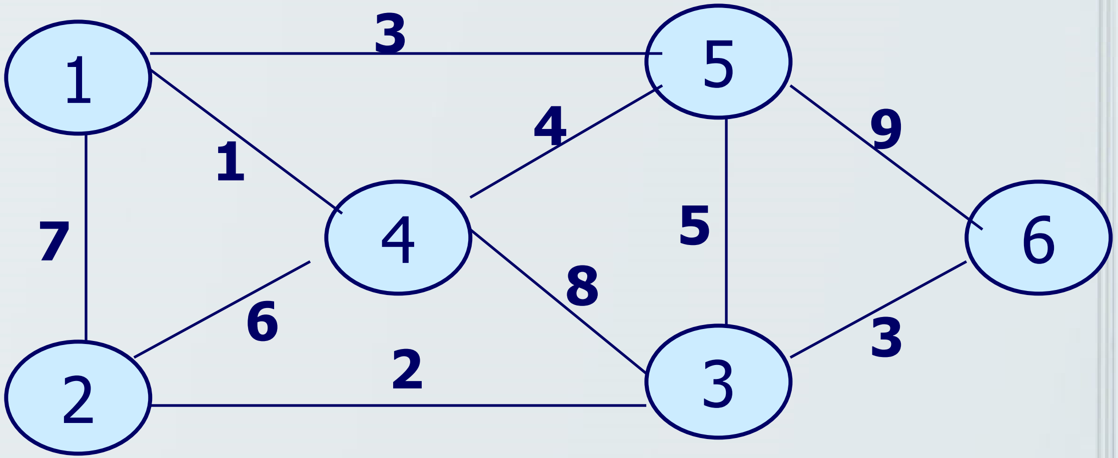
Thuật toán Prim (tiếp…)

- Cho đồ thị G = (X,U)
- Gọi X’ là tập các đỉnh kề các cạnh trong G’
- Ban đầu X chứa một đỉnh tuỳ ý trong G giả sử x, tập các cạnh G’ rỗng
- Ở mỗi bước ta chọn cạnh (x,y) ngắn nhất sao cho $x \in X'$ và $y \in X-X'$. Thêm y vào X’ và thêm (x,y) vào G’
- Tiếp tục phát triển G’ cho đến khi X’ =X. Khi đó G’ trở thành cây bao trùm ngắn nhất của G

Thuật toán Prim (tiếp…)

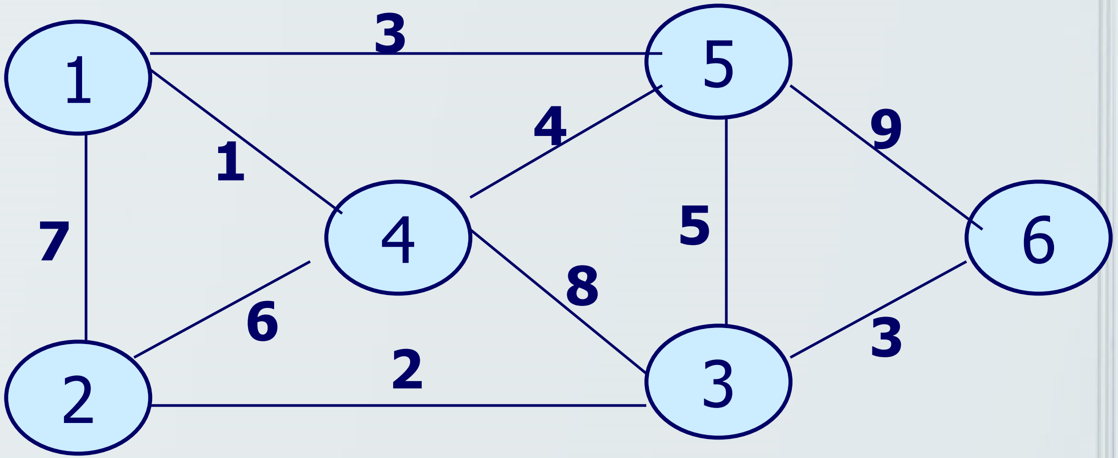
1. X ={x} (x: đỉnh tuỳ ý thuộc X)
2. G =∅
3. Thực hiện các bước sau cho đến khi X'=X
   1. Chọn cạnh (x,y) có trọng số nhỏ nhất với x ∈ X', y ∈ X-X'
   2. X' = X' ∪ {x}
   3. G' =G' ∪ {(x,y) }

Thuật toán Prim (tiếp…)



Bước	X’	(x,y)
Khởi tạo	{1}	-
1	{1,4}	(1,4)
2	{1,4, 5}	(1,5)
3	{1,4, 5, 3}	(5,3)
4	{1,4, 5, 3, 2}	(3, 2)
5	{1,4, 5, 3, 2, 6}	(3, 6)

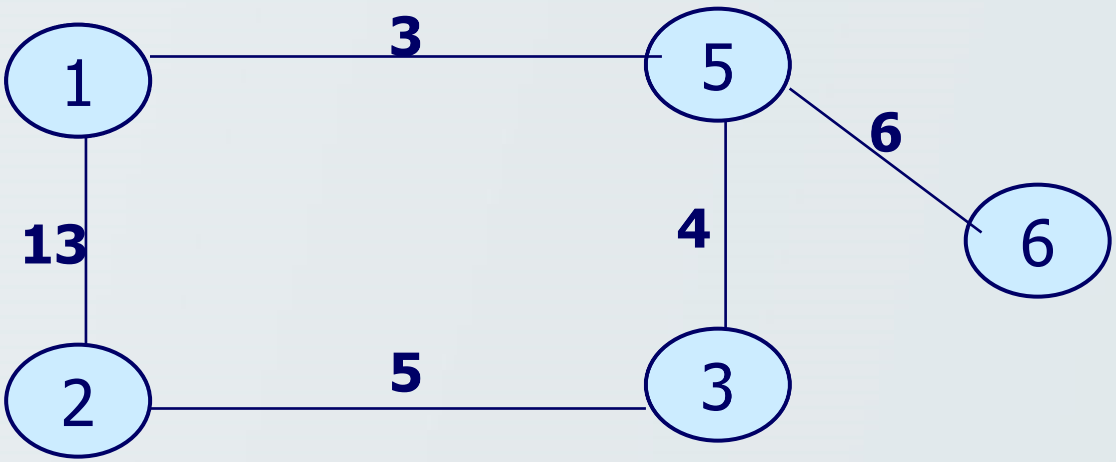
Thuật toán Prim (tiếp…)



Bước	X’	(x,y)
Khởi tạo	{3}	-
1	{3, 2}	(3, 2)
2	{3, 2, 6}	(3, 6)
3	{3, 2, 6,5}	(3, 5)
4	{3, 2, 6, 5, 1}	(5, 1)
5	{3, 2, 6, 5, 1, 4}	(1,4)

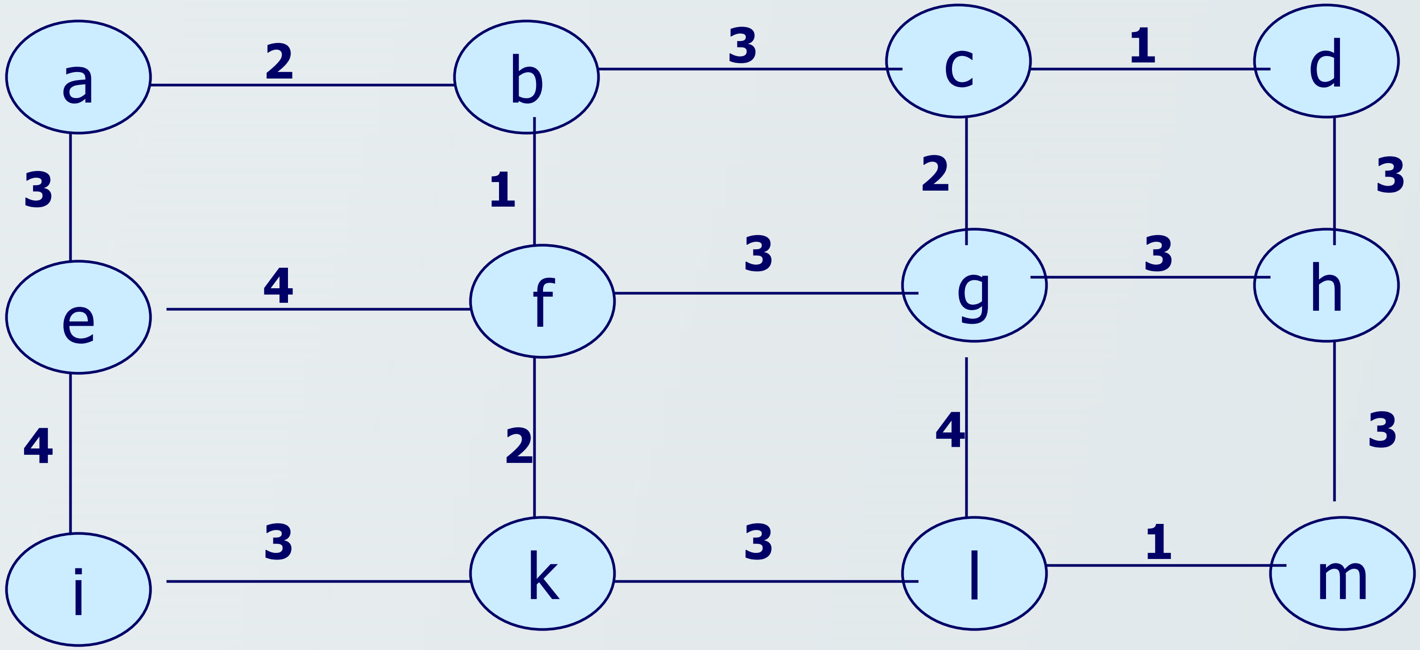
Thuật toán Prim (tiếp…)- Bài tập

Tìm cây bao trùm ngắn nhất của đồ thị G



Thuật toán Prim (tiếp…)- Bài tập

Tìm cây bao trùm ngắn nhất của đồ thị G



Duyệt đồ thị theo chiều sâu (Depth First Search)

- Bước 1:Duyệt tiếp tới đỉnh liền kề mà chưa được duyệt. Đánh dấu đỉnh mà đã được duyệt. Hiển thị đỉnh đó và đẩy vào trong một ngăn xếp (stack).

- Bước 2: Nếu không tìm thấy đỉnh liền kề, thì lấy một đỉnh từ trong ngăn xếp. (Giải thuật sẽ lấy tất cả các đỉnh từ trong ngăn xếp mà không có các đỉnh liền kề nào)

- Bước 3: Lặp lại các bước 1 và bước 2 cho tới khi ngăn xếp là trống.

Duyệt đồ thị theo chiều rộng (Breadth First Search)

- **Bước 1:** Duyệt tiếp tới đỉnh liền kề mà chưa được duyệt. Đánh dấu đỉnh mà đã được duyệt. Hiển thị đỉnh đó và đẩy vào trong một hàng đợi (queue).
- **Bước 2**: Nếu không tìm thấy đỉnh liền kề, thì xóa đỉnh đầu tiên trong hàng đợi.
- **Bước 3**: Lặp lại Bước 1 và 2 cho tới khi hàng đợi là trống.