Chương 5: ĐỒ THỊ

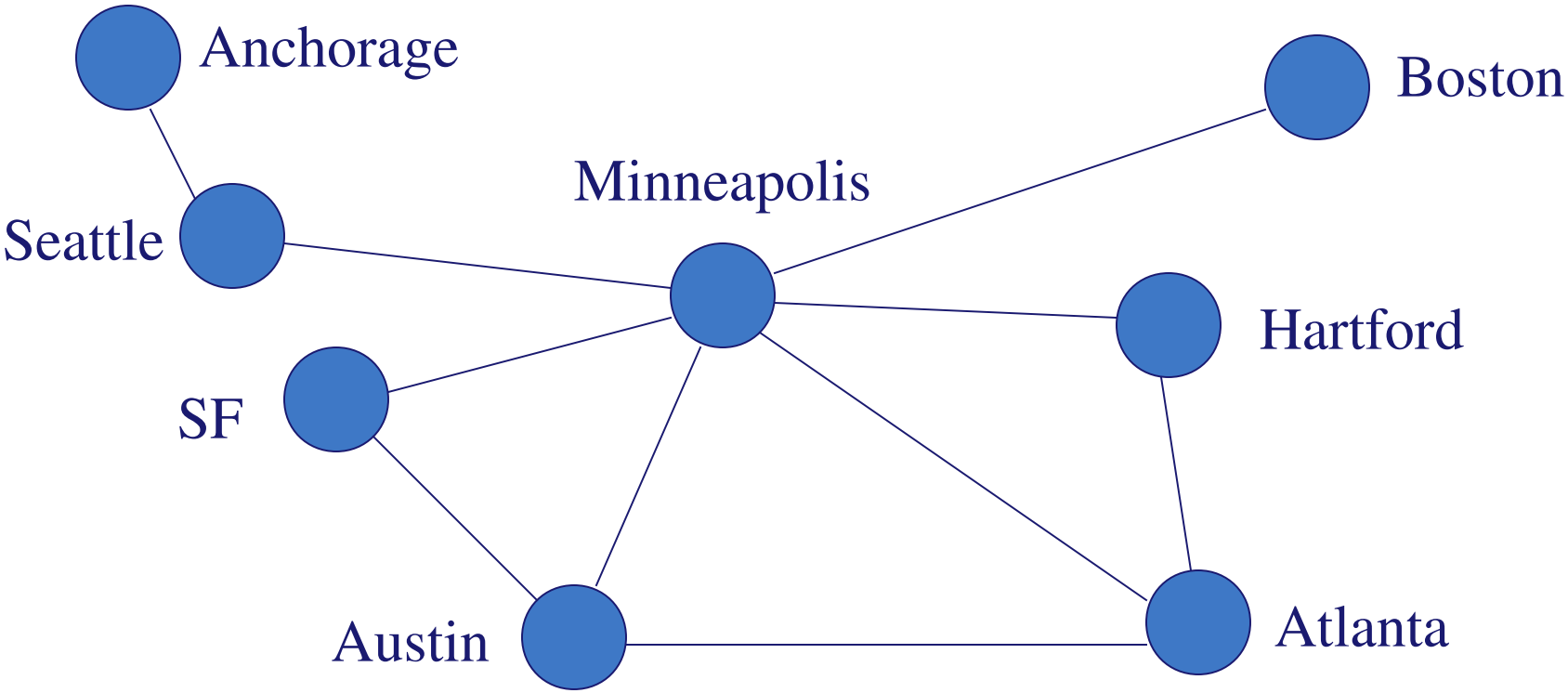
Ths. Phạm Thanh An
Bộ môn Khoa học máy tính - Khoa CNTT
Trường Đại học Ngân hàng TP.HCM

LOGO

Mục tiêu của chương

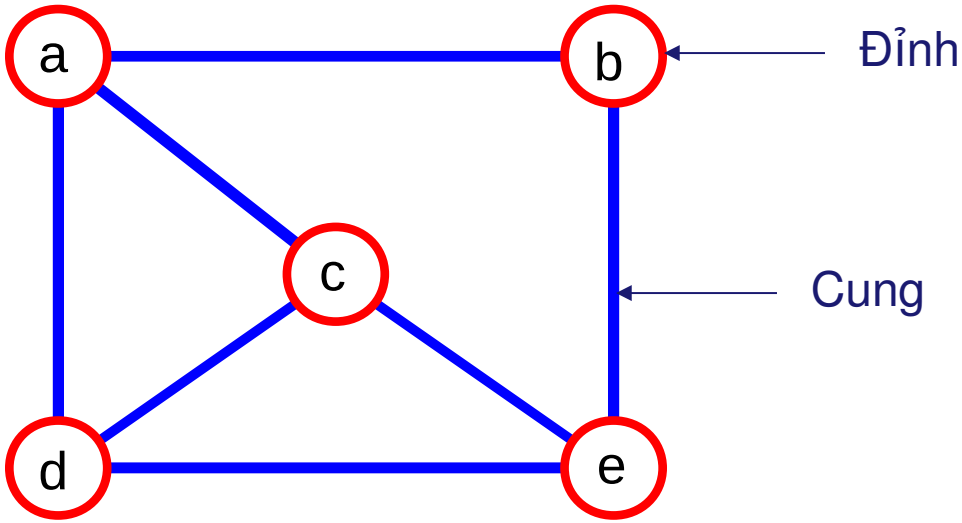
- Trình bày những kiến thức căn bản về lý thuyết đồ thị, cách biểu diễn, một số thuật toán trên đồ thị
- Đánh giá thuật toán
- Một số ứng dụng của đồ thị

Định nghĩa


Anchorage
Boston
Minneapolis
Seattle
Hartford
SF
Austin
Atlanta

Định nghĩa

- Đồ thị G = (V,E)
  - V = tập hợp hữu hạn các phần tử (đỉnh hay nút)
  - E V × V, tập hữu hạn các cạnh (cung)



Các khái niệm

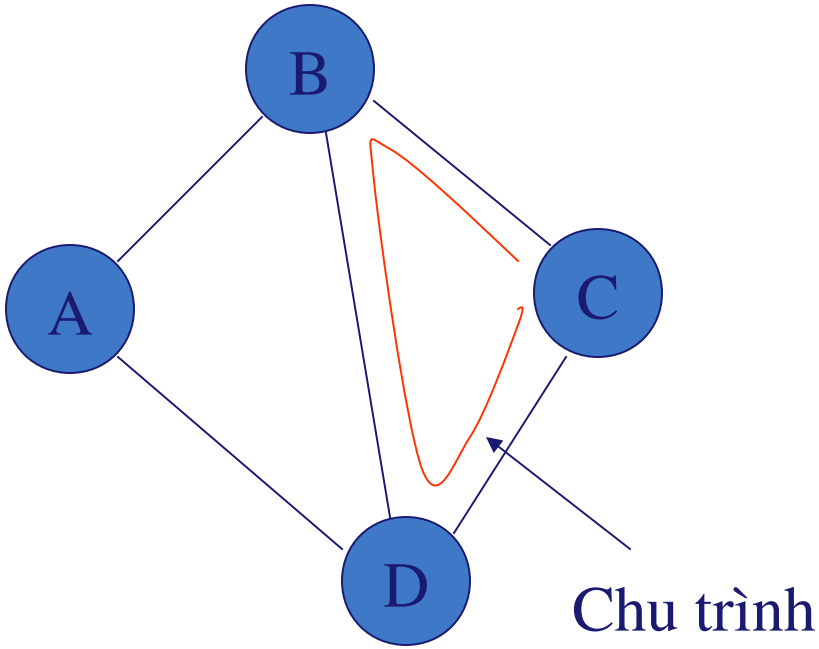
- Nếu (x,y) E
  - x gọi là đỉnh gốc, y là ngọn
  - Nếu x ≡ y, (x,y) gọi là khuyên
- Một dãy $u_1,u_2,\ldots,u_n$, u_i V (i=1,n) gọi là một đường, nếu (u_{i-1},u_i) E
- Độ dài đường: length$(u_1,u_2,\ldots,u_n)=n$
- $(u_1,u_2,\ldots,u_n)$ đường đi đơn, nếu $u_i \neq u_j$, $1<$ $i \neq j<n$ (là đường đi, mà các đỉnh phân biệt, ngoại trừ đình đầu và đỉnh cuối)

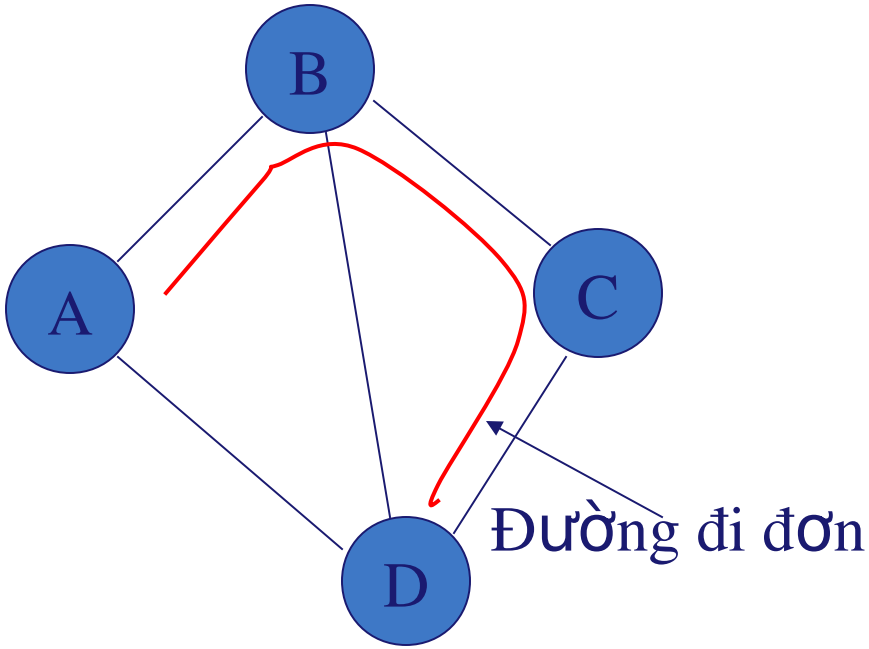
Các khái niệm (tt)

- Chu trình (cycle) = $(u_1,u_2,\ldots,u_n)$, $u_1 \equiv u_n$
- Đồ thị định hướng (directed graph)
  - (x,y) E : (x,y) ≠ (y,x)
- Đồ thị vô hướng
  - (x,y) E : (y,x) E
  - (x,y) ≡ (y,x)

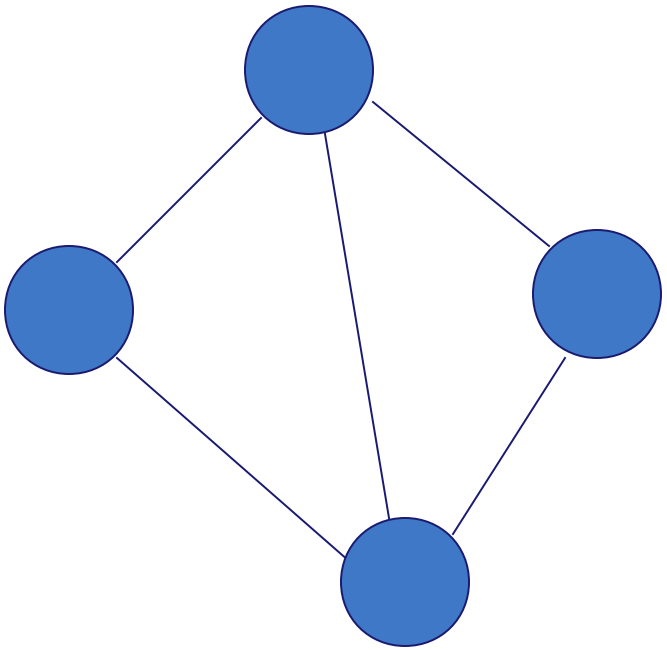
Các khái niệm (tt)

- *CBDC là một chu trình*

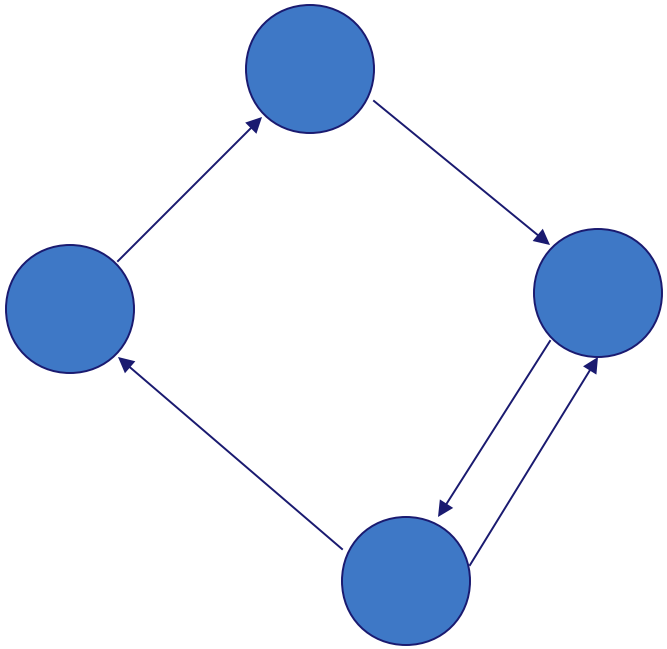




Các khái niệm (tt)



Đồ thị vô hướng

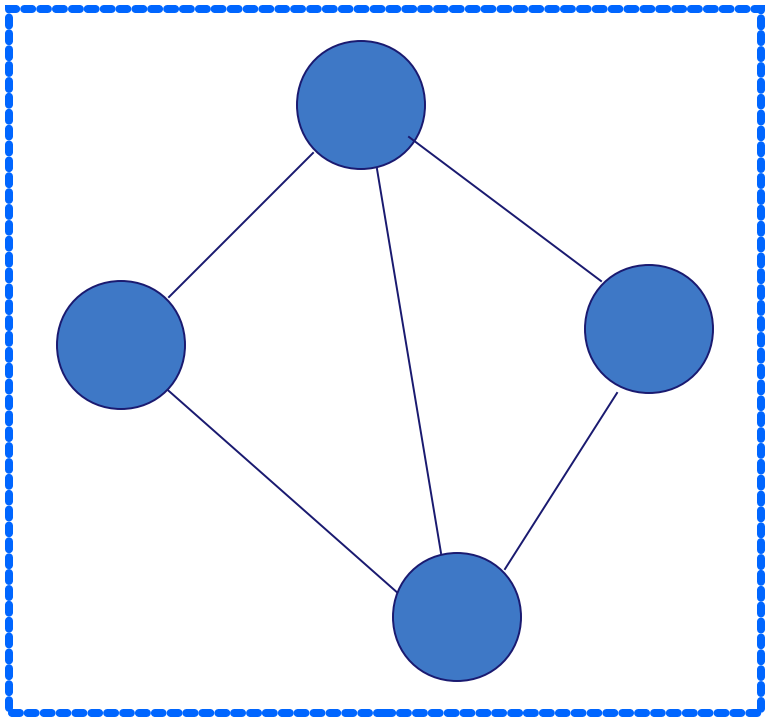


Đồ thị định hướng

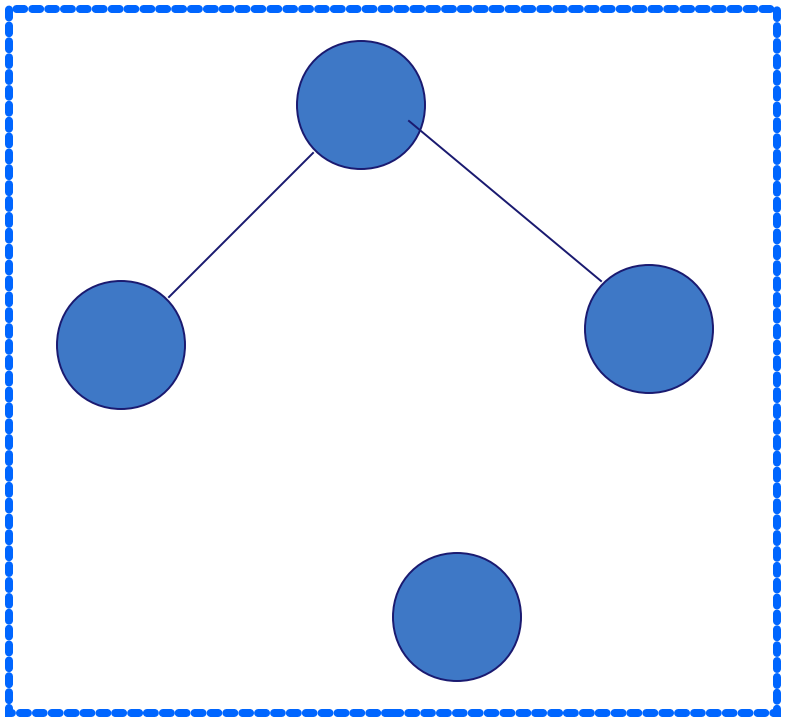
Các khái niệm (tt)

- Tính liên thông (connectivity)
  - Trong đồ thị G, hai đỉnh x,y gọi là liên thông (connected), nếu có một đường từ x đến y
  - Đồ thị G liên thông, nếu (x,y) E, đường đi từ x đến y
- Đồ thị G gọi là có trọng số, nếu mỗi cung được gán một giá trị số đặc trưng

Các khái niệm (tt)



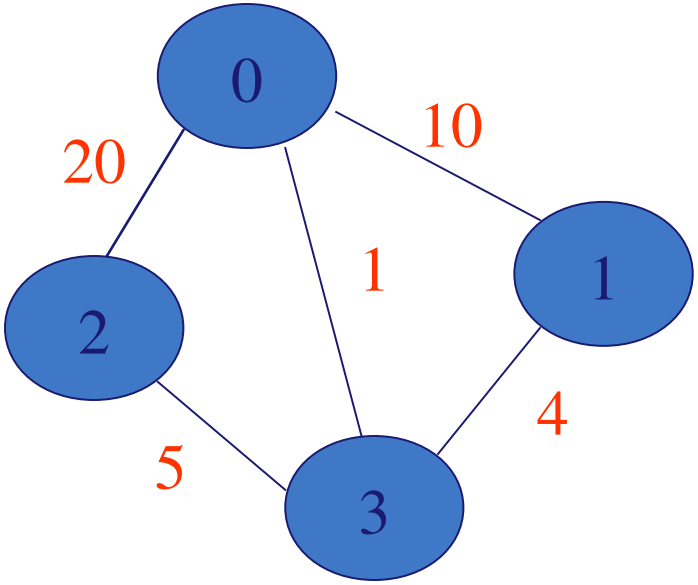
Đồ thị liên thông



Đồ thị không liên thông

Các khái niệm (tt)

- Đồ thị có trọng số



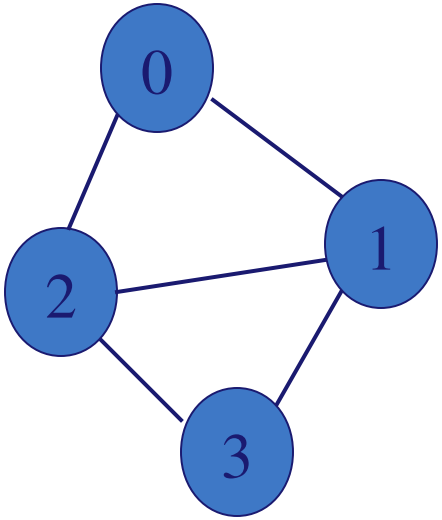
Biểu diễn đồ thị

- Biểu diễn bằng ma trận kề
  - Adjacency matrice
- Biểu diễn bằng danh sách kề
  - Adjacency list

Biểu diễn bằng ma trận kề

- Xét $G=(V,E)$ với $V=\{x_1,\ldots,x_n\}$
- Biểu diễn G bằng ma trận $A=(a_{ij})$, $i,j=1..n$
  - $a_{ij}=1$, nếu $\exists\ (x_i,x_j)\quad E$
  - $a_{ij}=0$, nếu $!\exists\ (x_i,x_j)\quad E$

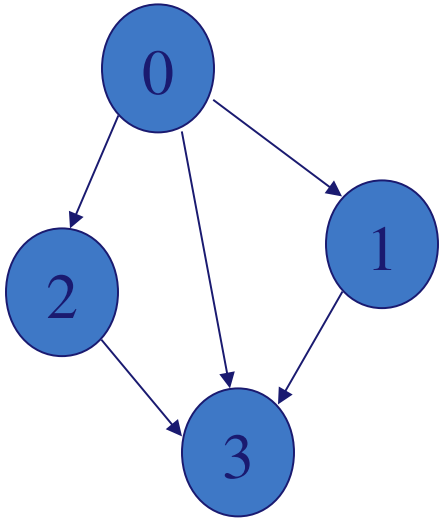
Biểu diễn bằng ma trận kề(tt)



A[i][j]	0	1	2	3
0	0	1	1	0
1	1	0	1	1
2	1	1	0	1
3	0	1	1	0

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

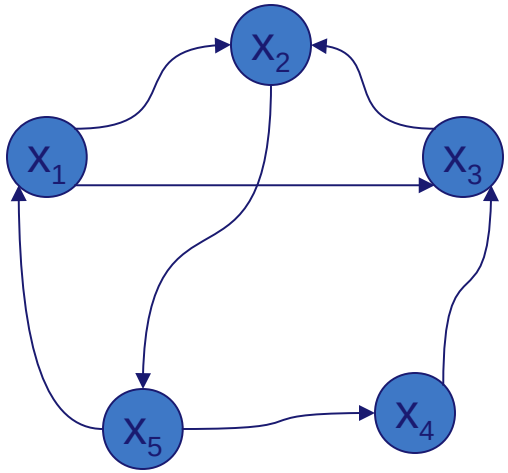
Biểu diễn bằng ma trận kề(tt)



A[i][j]	0	1	2	3
0	0	1	1	1
1	0	0	0	1
2	0	0	0	1
3	0	0	0	0

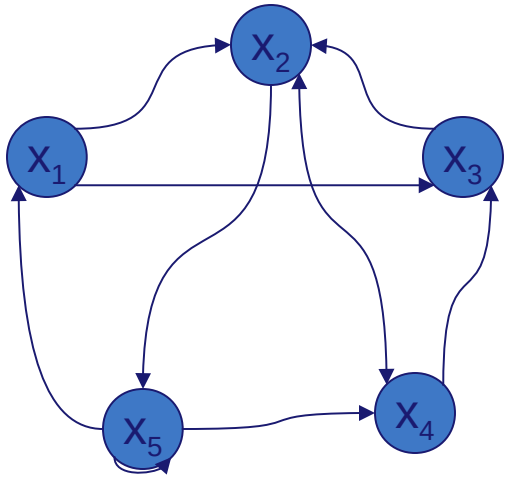
$$A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Biểu diễn bằng ma trận kề(tt)



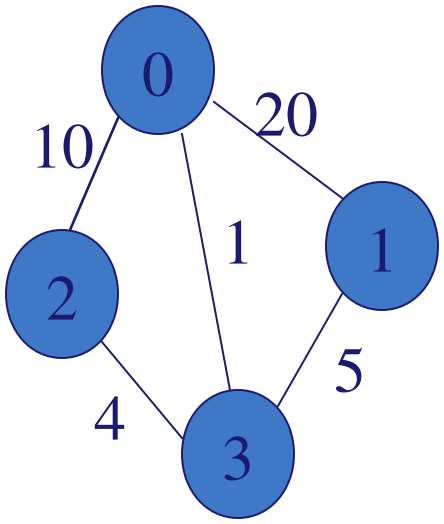
$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Biểu diễn bằng ma trận kề (tt)



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

Biểu diễn bằng ma trận kề (tt)



A[i][j]	0	1	2	3
0	0	20	10	1
1	20	0	0	5
2	10	0	0	4
3	1	5	4	0

$$A = \begin{pmatrix} 0 & 20 & 10 & 1 \\ 20 & 0 & 0 & 5 \\ 10 & 0 & 0 & 4 \\ 1 & 5 & 4 & 0 \end{pmatrix}$$

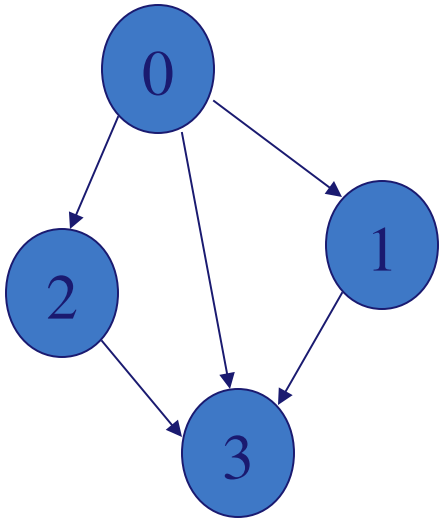
Biểu diễn bằng ma trận kề (tt)

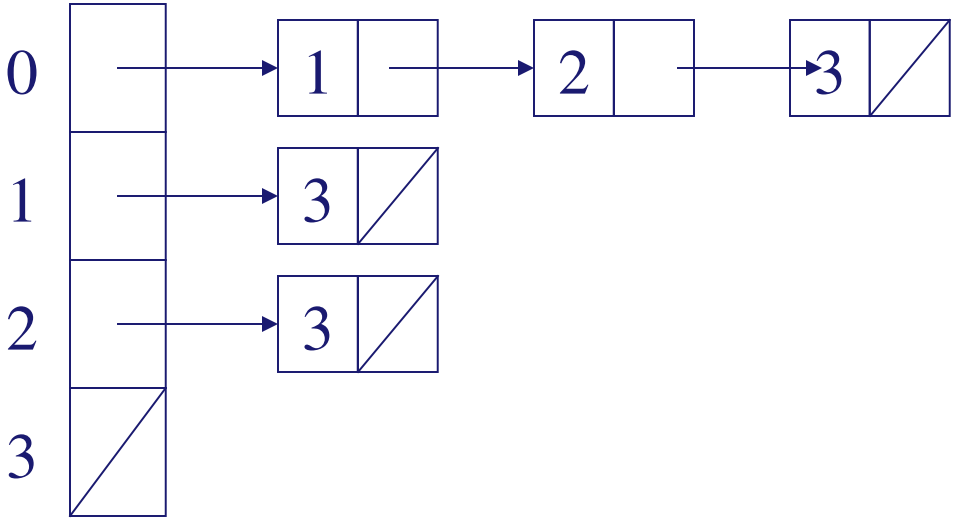
- Chú ý
  - Đối với đồ thị không định hướng, ma trận kề là ma trận đối xứng
  - Đối với đồ thị định hướng, số lượng phần tử 0 khá lớn
  - Đối với đồ thị có trọng số, thay thế giá trị 1 bằng giá trị trọng số

Biểu diễn đồ thị bằng danh sách kề

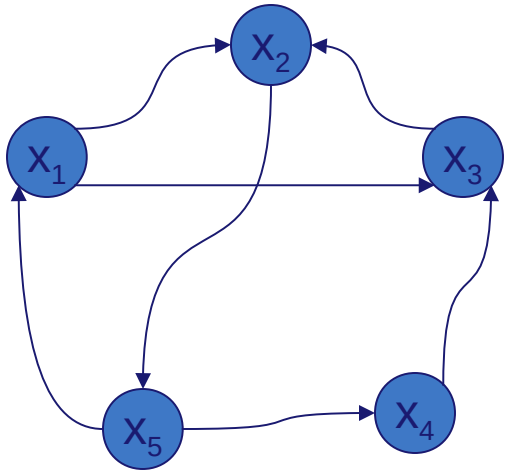
- Là một mảng các danh sách
- Ở đây, n hàng của ma trận kề thay thế bằng n danh sách liên kết động
- Mỗi đỉnh của G có một danh sách, mỗi nút trong danh sách thể hiện các đỉnh lân cận của nút này
  - Cấu trúc mỗi nút
    - id: tên đỉnh (chỉ số, danh hiệu)
    - next: con trỏ đến nút kế tiếp

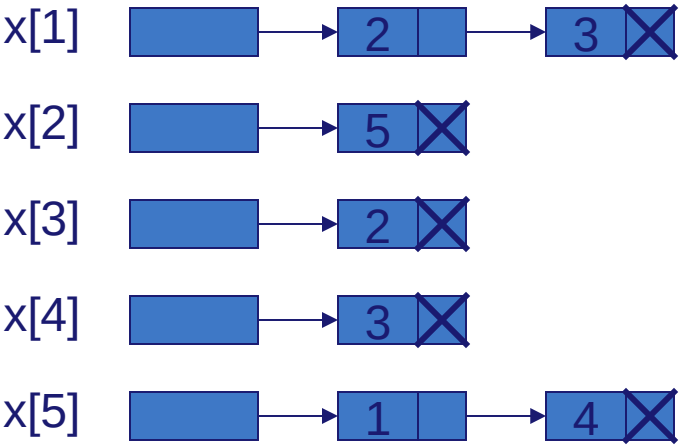
Biểu diễn đồ thị bằng danh sách kề (tt)



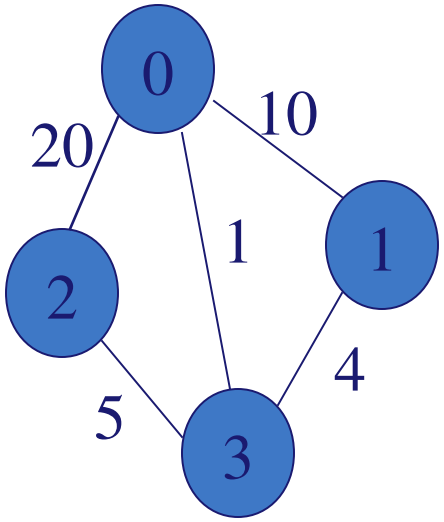


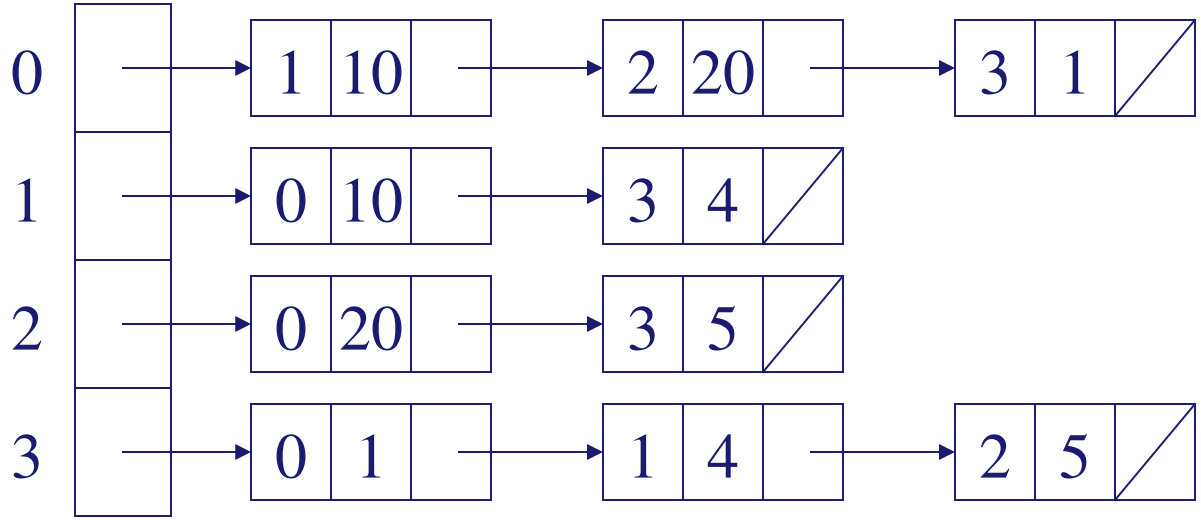
Biểu diễn đồ thị bằng danh sách kề (tt)





Biểu diễn đồ thị bằng danh sách kề (tt)





Biểu diễn đồ thị bằng danh sách kề (tt)

- Chú ý
  - Các nút đầu danh sách được lưu vào một mảng (truy cập nhanh)
  - Với đồ thị không định hướng có n đỉnh và e cạnh, thì cần n nút đầu và 2e nút ‘trong’ danh sách
  - Với đồ thị định hướng có n đỉnh và e cạnh, thì chỉ cần e nút ‘trong’ danh sách
  - Thứ tự các nút không quan trọng

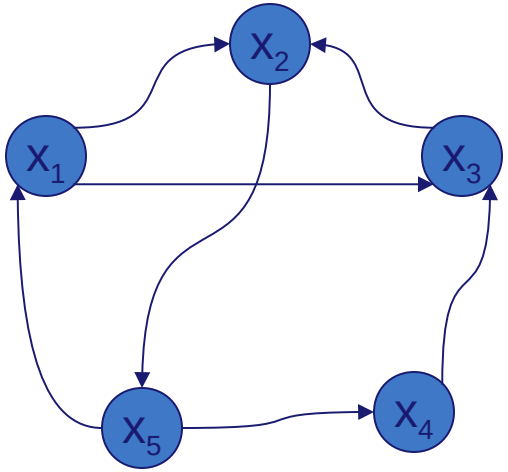
Phép duyệt đồ thị

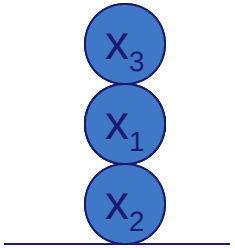
- Từ một đỉnh, liệt kê tất cả các đỉnh của đồ thị
  - Phép tìm kiếm theo chiều sâu
    - Depth first search
  - Phép tìm kiếm theo chiều rộng
    - Breadth first search

Phép tìm kiếm theo chiều sâu

- Ý tưởng
  - Tại đỉnh v bất kỳ, duyệt đỉnh v, và xét tập các đỉnh đến được từ đỉnh v
  - Lập lại thao tác trên đối với đỉnh w bất kỳ trong tập các đỉnh từ v nói trên

Phép tìm kiếm theo chiều sâu (tt)





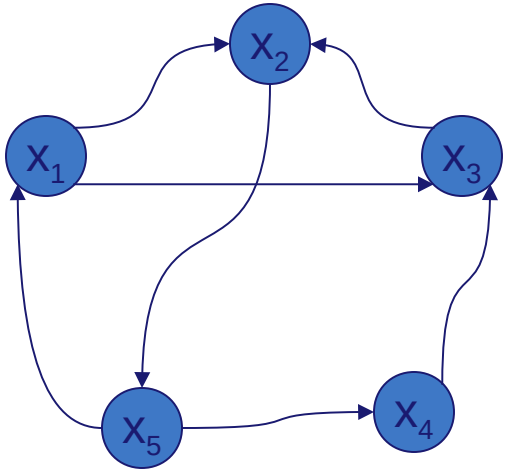
Phép tìm kiếm theo chiều sâu (tt)

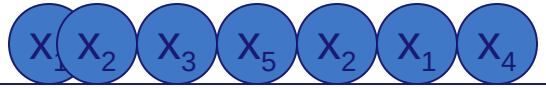
- Nhận xét
  - Thời gian thực hiện giải thuật ~ (n+e), nếu G được biểu diễn bằng danh sách kề
  - Thời gian thực hiện giải thuật ~ n^2, nếu G được biểu diễn bằng ma trận kề
  - Giải thuật này sử dụng để chứng minh một đồ thị có liên thông hay không

Phép tìm kiếm theo chiều rộng

- Tại điểm v bất kỳ, duyệt đỉnh v, thu được tập hợp W gồm các đỉnh w xuất phát từ v
- Lặp lại thao tác trên đối với tất cả các đỉnh w trong W, thu được tập hợp đỉnh Z
- Lặp lại thao tác trên đối với tất cả các đỉnh z trong Z
- Lặp lại cho đến khi tất cả mọi đỉnh đều được duyệt qua ít nhất một lần

Phép tìm kiếm theo chiều rộng(tt)





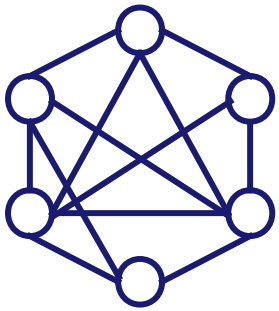
Phép tìm kiếm theo chiều rộng(tt)

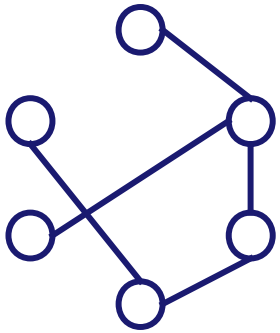
- Nhận xét
  - Thời gian thực hiện giải thuật ~ (n+e), nếu G được biểu diễn bằng danh sách kề
  - Thời gian thực hiện giải thuật ~ n^2, nếu G được biểu diễn bằng ma trận kề

Cây khung (Spanning tree)

- T=(V,E’) G=(V,E), với E E’ bao gồm các cung thuộc một phép duyệt từ một đỉnh đến các đỉnh còn lại trong V
- Giá của cây khung T = tổng trọng số của các cung thuộc E’

Đồ thị G





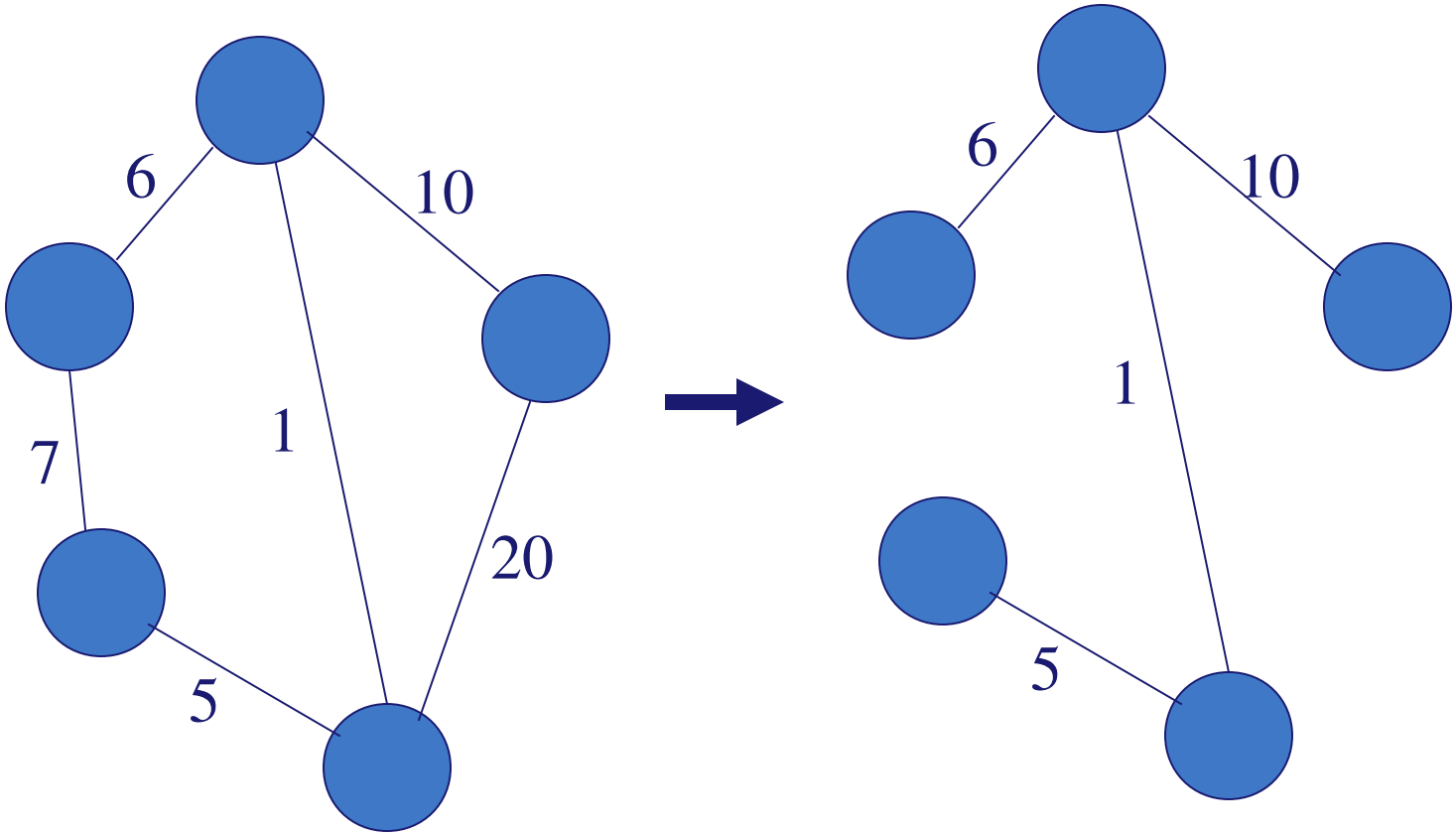
Cây khung T
của đồ thị G

Cây khung (Spanning tree)

- Chú ý
  - Một đồ thị G có thể có nhiều cây khung
  - Cây khung theo chiều rộng, theo chiều sâu
  - Các cung trong cây khung không tạo nên chu trình
    - Giữa hai đỉnh trong một cây khung chỉ tồn tại duy nhất một đường đi từ đỉnh này đến đỉnh kia
  - Nếu đồ thị có n đỉnh, thì cây khung có n-1 cạnh

Cây khung cực tiêu

- Là cây khung với tổng các trọng số là cực tiểu

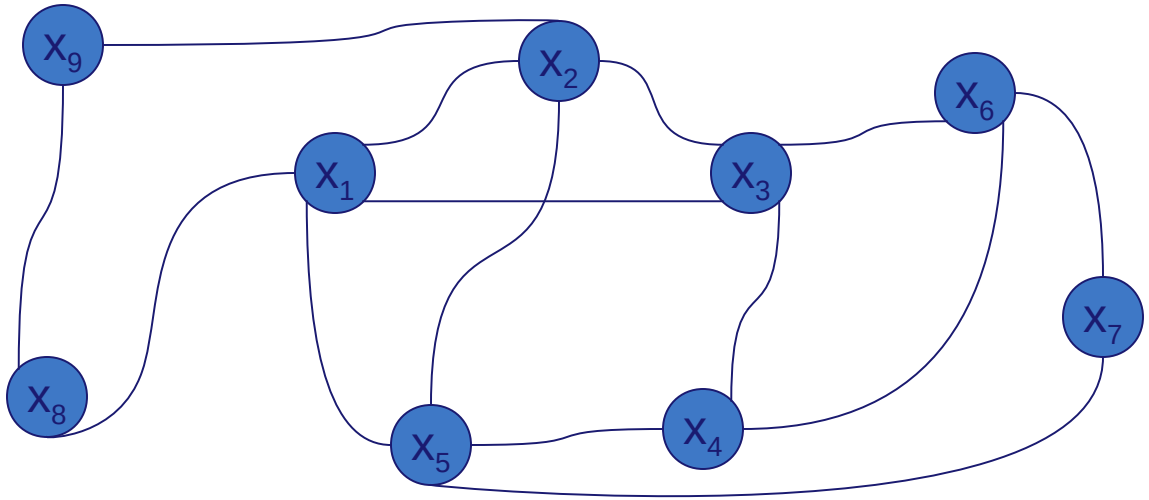


Thuật toán Kruskal

- Sắp xếp các cung theo thứ tự không giảm đối với trọng số
- Bắt đầu từ T=Ø
- Lặp cho đến khi ko còn đỉnh nào (|E'| = n-1)
  - lấy ra cung w có trọng số nhỏ nhất
  - thêm cung w vào T với điều kiện không tạo chu trình trong T

Cây khung (Spanning tree)

- Ví dụ: Cho một đồ thị G vô hướng, liên thông, có trọng số. Hãy tìm cây khung cực tiểu của G



Thuật toán Kruskal

- Để kiểm tra xem có tạo ra chu trình trong T hay không, chúng ta xem hai đỉnh của cung được thêm có thuộc tập các đỉnh hiện có trong T không, nếu có, nghĩa là sẽ tạo nên chu trình

Bài toán bao đóng truyền ứng

- Cho đồ thị G = (V,E)
  - Có tồn tại đường đi giữa hai nút x và y trong đồ thị G hay không?
  - Bài toán này có thể giải được dễ dàng bằng cách sử dụng ma trận kề của đồ thị

Bài toán bao đóng truyền ứng

- Ma trận kề A của đồ thị G=(V,E)
  - a_{ij}=true, nếu ∃ (x_i,x_j) E
  - a_{ij}=false, nếu ngược lại
- Phép cộng A=(a_{ij}), B=(b_{ij})
  - $A \vee B = C$, với $c_{ij}=a_{ij} \vee b_{ij}$
- Phép nhân A=(a_{ij}), B=(b_{ij})
  - $D = A \wedge B$, với $d_{ij}=\bigvee_{k=1}^{n}(a_{ik} \wedge b_{kj})$

Với 1<=i, j, <=n

Bài toán bao đóng truyền ứng

- Với ma trận A, nếu $a_{ij} = 1$, có nghĩa là có một cung từ i tới j.
- Xét $A^{(2)} = A \wedge A$. Rõ ràng nếu phần tử ở hàng i, cột j của $A^{(2)}$ bằng 1, thì có ít nhất có một đường đi có độ dài 2, từ đỉnh i đến đỉnh j, vì

  $a^{(2)}_{ij} = \bigvee_{k=1} (a_{ik} \wedge a_{kj})$

  - $a_{ik} \wedge a_{kj} = 1$, khi $a_{ik} = 1$ và $a_{kj} = 1$, => tức là có đường đi độ dài 1 từ i tới k và có đường đi đô dài 1 từ k tới j

Bài toán bao đóng truyền ứng

- Từ đó suy ra $A^{(r)} = A \wedge A^{(r-1)}$, R=2, 3,.. Nghĩa là $a^{(r)}_{ij} = 1$, thì có ít nhất 1 đường đi đô dài r, từ i tới j.
- Ta lập ma trân $P = A \vee A^{(2)} \vee .. \vee A^{(n)}$
- Thì P sẽ cho biết có hay không một đường đi có độ dài lớn nhất là n, từ đỉnh i tới j. P được gọi là ma trận đường đi.

Bài toán bao đóng truyền ứng

- Thuật toán WARSHALL

```
Void WARSHALL(A, P, n){
   For (int k=0;k<n;k++)
      For (int i=0;i<n;i++)
         For (int j=0;j<n;j++)
            P[i,j]=P[i,j] V (P[i,k] Λ P[k,j])
   }
```

Bài toán đường đi ngắn nhất

- Vấn đề
  - Cho một đồ thị định hướng, liên thông, có trọng số G
  - Hãy tìm đường đi ngắn nhất từ một đỉnh đến tất cả các đỉnh khác trong đồ thị

Thuật toán Dijkstra

- Xét đồ thị có hướng G=(V,E), với |V|=n
- Ma trận trọng số d[u,v]≥0, (u,v) E
- s V là điểm xuất phát
- H[v]=chiều dài cực tiểu từ s đến v (v V)

Thuật toán Dijkstra

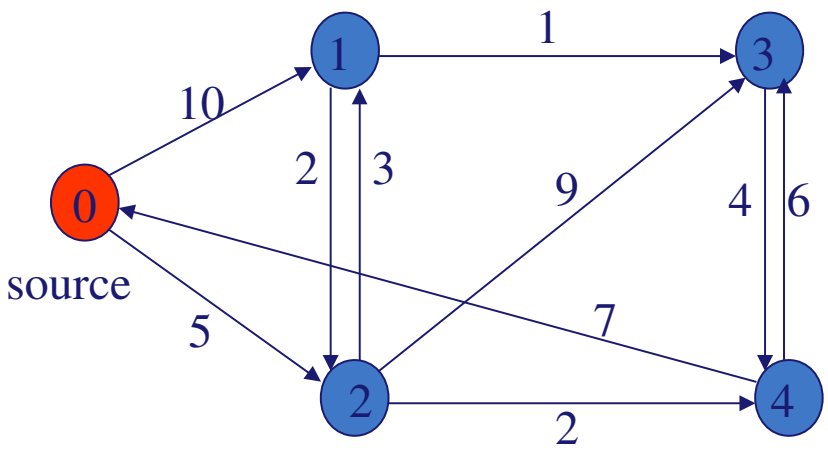
- Bắt đầu duyệt từ đỉnh s
- Gán giá trị cho H[v]
  - H[v]=d(s,v), nếu (s,v) E
  - H[v]=∞, nếu ngược lại
- Lặp lại cho đến khi duyệt hết các đỉnh
  - Chọn đỉnh w chưa duyệt có H[w] nhỏ nhất
  - Duyệt đỉnh w này
  - Với các đỉnh t chưa duyệt khác
    - H[t] = min(H[t],H[w]+d(w,t))

Thuật toán Dijkstra

- Hoạt động tốt trên đồ thị trọng số dương
- Độ phức tạp giải thuật là $O(n^2)$

Thuật toán Dijkstra

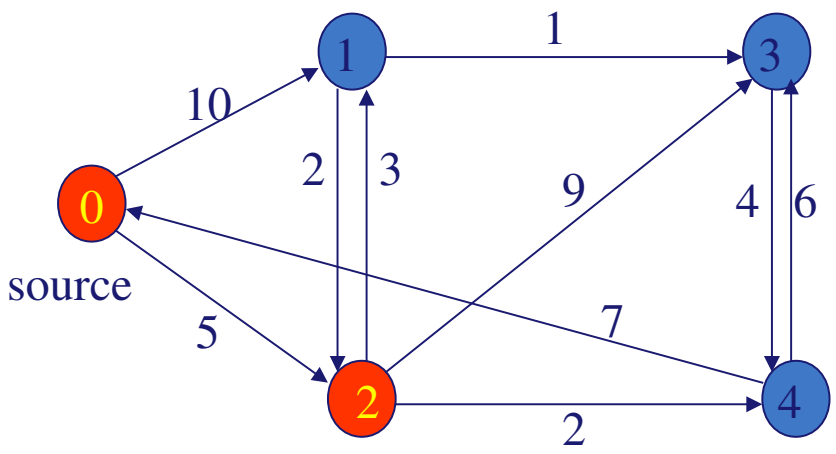
- Tìm đường đi từ v_0 tới các đỉnh còn lại



node	from node V_0 to other nodes			
V_1	10			
V_2	5			
V_3				
V_4				
best				

Thuật toán Dijkstra

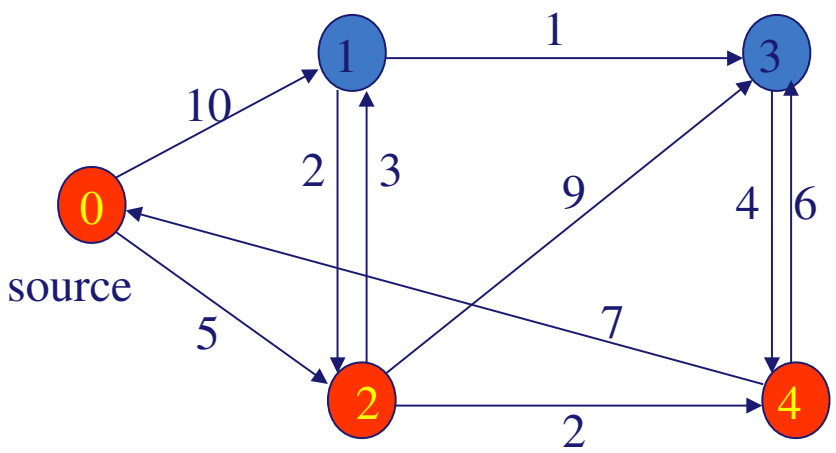
- step 1: tìm đường đi ngắn nhất từ 0
  - node 2 được chọn



node	from node V_0 to other nodes			
V_1	10			
V_2	5			
V_3				
V_4				
best	V_2			

Thuật toán Dijkstra

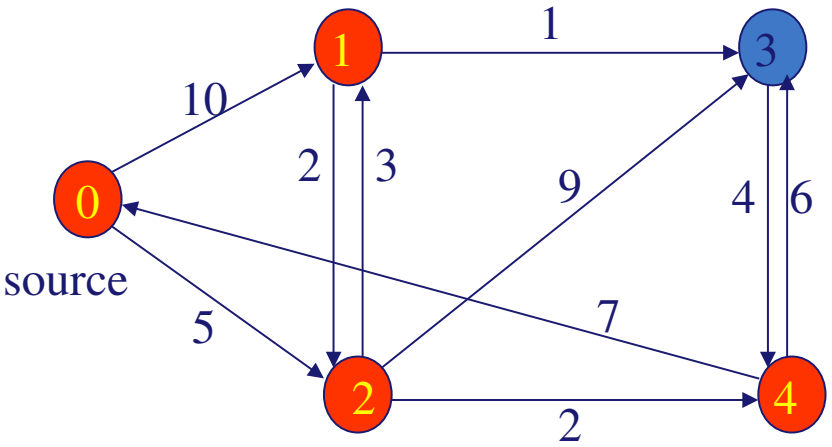
- step 2: Tính toán lại các đường đi đến tất cả các đỉnh
  - Tìm đường đi ngắn nhất đến node 0. Node 4 được chọn



node	from node V_0 to other nodes			
V_1	10	8		
V_2	5	5		
V_3		14		
V_4		7		
best	V_2	V_4		

Thuật toán Dijkstra

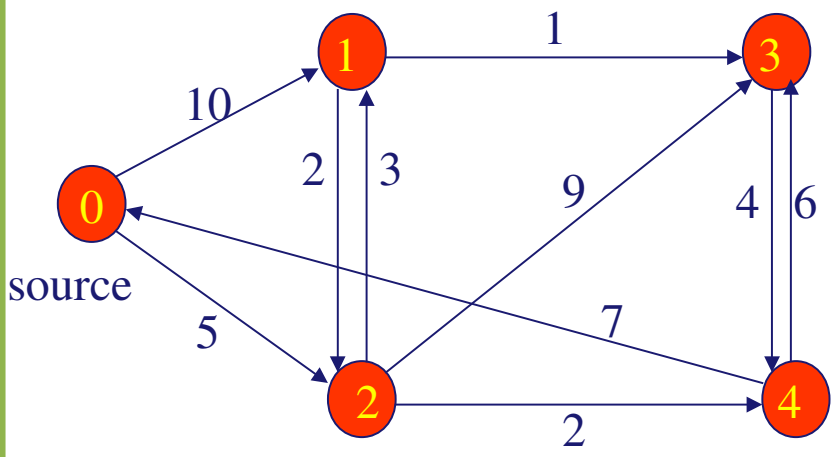
- step 2: Tính toán lại các đường đi đến tất cả các đỉnh
  - Tìm đường đi ngắn nhất từ node 0. node 1 được chọn



node	from node V_0 to other nodes			
V_1	10	8	8	
V_2	5	5	5	
V_3		14	13	
V_4		7	7	
best	V_2	V_4	V_1	

Thuật toán Dijkstra

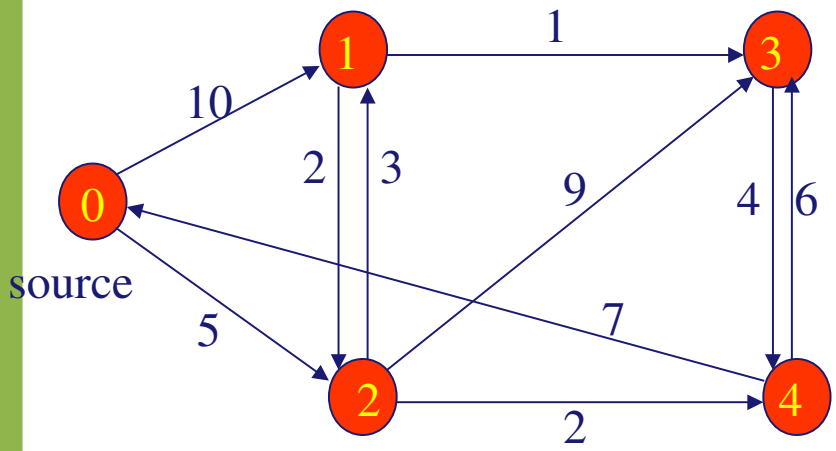
- step 2: Tính toán lại các đường đi đến tất cả các đỉnh
  - Tìm đường đi ngắn nhất từ node 0. node 3 được chọn

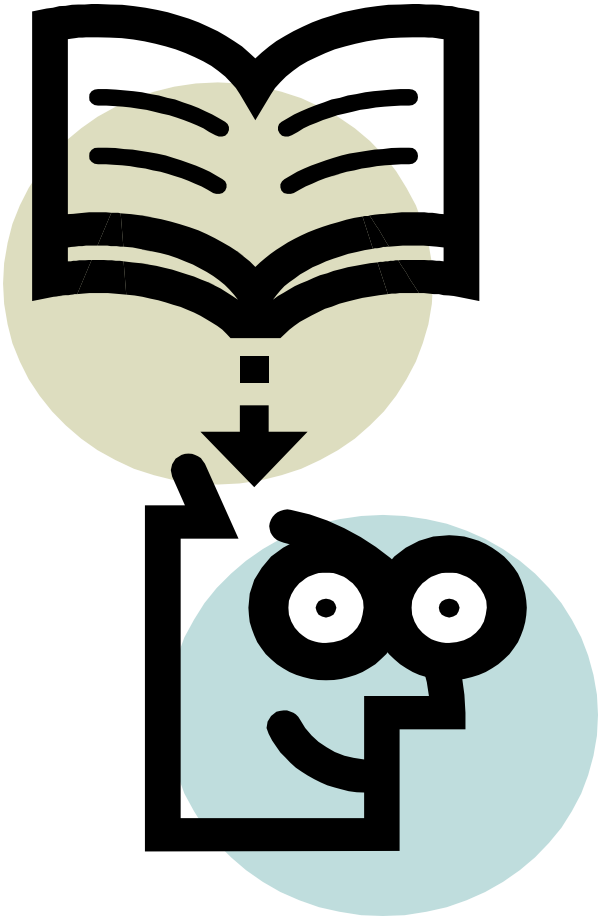


node	from node V_0 to other nodes			
V_1	10	8	8	8
V_2	5	5	5	5
V_3		14	13	9
V_4		7	7	7
best	V_2	V_4	V_1	V_3

Thuật toán Dijkstra

- Chúng ta có tất cả các đường đi từ v_0



node	from node V_0 to other nodes			
V_1	10	8 (0,2)	8 (0,2)	8
V_2	5 (0,2)	5	5	5
V_3		14 (0,2,3)	13 (0,2,4,3)	9 (0,2,1,3)
V_4		7 (0,2,4)	7	7
best	V_2	V_4	V_1	V_3

Q&A