Hanoi University of Science and Technology
School of Information and Communications Technology

Discrete Mathematics

Nguyễn Khánh Phương

Department of Computer Science
School of Information and Communication Technology
E-mail: phuongnk@soict.hust.edu.vn

PART 1
COMBINATORIAL THEORY
(Lý thuyết tổ hợp)

PART 2
GRAPH THEORY
(Lý thuyết đồ thị)

Content of Part 2

Hanoi University of Science and Technology
School of Information and Communications Technology

Chapter 2: Graph representation

Graph Representation

1. Incidence matrix
2. Adjacency matrix
3. Weight matrix
4. Adjacency list

Graph Representation

1. **Incidence matrix**
2. Adjacency matrix
3. Weight matrix
4. Adjacency list

1.Incidence Matrix

G = (V, E) is an undirected graph:

- $V = \{v_1, v_2, v_3, \ldots, v_n\}$
- $E = \{e_1, e_2, \ldots, e_m\}$

Then the incidence matrix with respect to this ordering of V and E is the n x m matrix $M = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i \\ 0 & \text{otherwise} \end{cases}$$

Can also be used to represent :

- **Multiple edges:** by using columns with identical entries, since these edges are incident with the same pair of vertices
- **Loops:** by using a column with exactly one entry equal to 1, corresponding to the vertex that is incident with the loop

1.Incidence Matrix

Matrix $M_{|V| \times |E|} = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i \\ 0 & \text{otherwise} \end{cases}$$

Can also be used to represent :

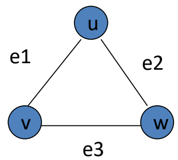
- **Multiple edges:** by using columns with identical entries, since these edges are incident with the same pair of vertices
- **Loops:** by using a column with exactly one entry equal to 1, corresponding to the vertex that is incident with the loop

1.Incidence Matrix

Matrix $M_{|V| \times |E|} = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i \\ 0 & \text{otherwise} \end{cases}$$

Example: G = (V, E)



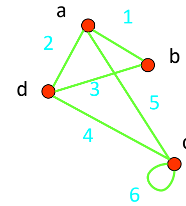
	e_1	e_2	e_3
v	1	0	1
u	1	1	0
w	0	1	1

1.Incidence Matrix

Matrix $M_{|V| \times |E|} = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1 & \text{when edge } e_j \text{ is incident with } v_i \\ 0 & \text{otherwise} \end{cases}$$

Example: What is the incidence matrix M for the following graph G based on the order of vertices a, b, c, d and edges 1, 2, 3, 4, 5, 6?



Solution:

$$M = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

Note: Incidence matrices of undirected graphs contain two 1s per column for edges connecting two vertices and one 1 per column for loops.

1.Incidence Matrix

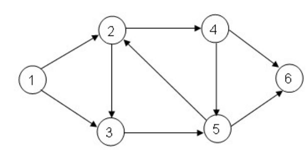
G = (V, E) is a directed graph:

- $V = \{v_1, v_2, v_3, \ldots, v_n\}$
- $E = \{e_1, e_2, \ldots, e_m\}$

Then the incidence matrix with respect to this ordering of V and E is the n x m matrix $M = [m_{ij}]$, where

$$m_{ij} = \begin{cases} 1 & \text{when } v_i \text{ is initial vertex of } e_j \\ -1 & \text{when } v_i \text{ is terminal vertex of } e_j \\ 0 & \text{otherwise} \end{cases}$$

1.Incidence Matrix



$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{ccccccccc} (1,2) & (1,3) & (2,3) & (2,4) & (3,5) & (4,5) & (4,6) & (5,2) & (5,6) \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & -1 & -1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 \end{bmatrix} \end{array}$$

Graph Representation

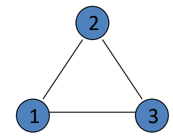
1. Incidence matrix
2. **Adjacency matrix**
3. Weight matrix
4. Adjacency list

2. Adjacency Matrix

The Adjacency Matrix (NxN) $A = [a_{ij}]$ where $|V| = N$

For undirected graph

$$a_{ij} = \begin{cases} 1 & \text{if } \{v_i, v_j\} \text{ is an edge of G} \\ 0 & \text{otherwise} \end{cases}$$

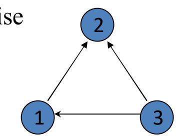


$$A=\begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$$

diagonally symmetric matrix

For directed graph

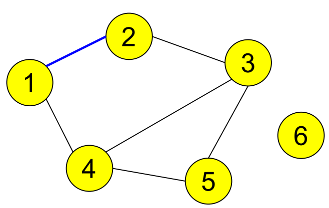
$$a_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \text{ is an edge of G} \\ 0 & \text{otherwise} \end{cases}$$



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

This makes it easier to find subgraphs, and to reverse graphs if needed.

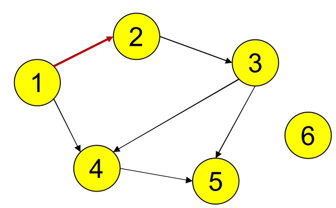
2. Adjacency Matrix



$$A[u,v] = \begin{cases} 1 \text{ if } \{u,v\} \in E \\ 0 \text{ otherwise} \end{cases}$$

$$\begin{array}{c|c} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Representation- Adjacency Matrix

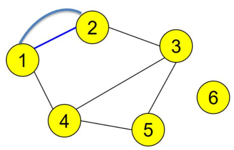


$$A[u,v] = \begin{cases} 1 \text{ if } (u,v) \in E \\ 0 \text{ otherwise} \end{cases}$$

$$\begin{array}{c|c} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \begin{pmatrix} 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Representation- Adjacency Matrix

- The adjacency matrix of simple undirected graphs are symmetric ($a_{ij} = a_{ji}$) (why?)
- When there are relatively few edges in the graph the adjacency matrix is a **sparse matrix**
- Directed Multigraphs can be represented by using a_{ij} = number of edges from v_i to v_j

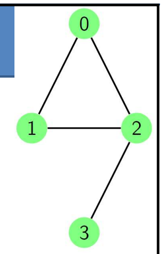


$$
\begin{array}{c|cccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & 0 & 2 & 0 & 1 & 0 & 0 \\
2 & 2 & 0 & 1 & 0 & 0 & 0 \\
3 & 0 & 1 & 0 & 1 & 1 & 0 \\
4 & 1 & 0 & 1 & 0 & 1 & 0 \\
5 & 0 & 0 & 1 & 1 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}
$$

Representation- Adjacency List

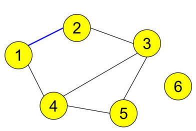
```
int A[MAX][MAX];
int V, E;

void input() {
    cin >> V;
    for (int i = 0; i < V; i++)
       for (int j = 0; j < V; j++)
          cin>>A[i][j];
}
```

0	1	1	0
1	0	1	0
1	1	0	1
0	0	1	0



Analyze the cost

- Memory Space
  - $|V|^2$ bits
- Time to answer the query
  - Two vertices *i* and *j are adjacent*? $O(1)$
  - Add or delete one edge $O(1)$
  - Add one vertice increase the size of matrix
  - Enumerate the adjacent vertices of *u* $O(|V|)$ (even when *u* is an isolated vertice).



$$
\begin{array}{c|cccccc}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & 0 & 1 & 0 & 1 & 0 & 0 \\
2 & 1 & 0 & 1 & 0 & 0 & 0 \\
3 & 0 & 1 & 0 & 1 & 1 & 0 \\
4 & 1 & 0 & 1 & 0 & 1 & 0 \\
5 & 0 & 0 & 1 & 1 & 0 & 0 \\
6 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}
$$

Graph Representation

1. Incidence matrix
2. Adjacency matrix
3. **Weight matrix**
4. Adjacency list

3. Weight matrix

- **Weighted** graphs have values associated with edges.
- In the case weighted graphs, instead of adjacency matrix, we use weight matrix to represent the graph

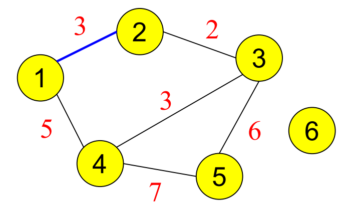
$$C = c[i, j], \; i, j = 1, 2, \ldots, n,$$

where

$$c[i,j] = \begin{cases} c(i,j), & \text{if } (i,j) \in E \\ \theta, & \text{if } (i,j) \notin E, \end{cases}$$

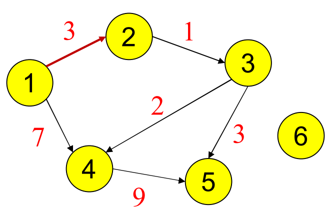
- θ: special value to identify (i, j) is not an edge; depends on the case, the value of θ could be: 0, $+\infty$, $-\infty$.

Weight matrix of undirected graph



$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 0 & 3 & 0 & 5 & 0 & 0 \\ 2 & 3 & 0 & 2 & 0 & 0 & 0 \\ 3 & 0 & 2 & 0 & 3 & 6 & 0 \\ 4 & 5 & 0 & 3 & 0 & 7 & 0 \\ 5 & 0 & 0 & 6 & 7 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Weight matrix of directed graph



$$\begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline 1 & 0 & 3 & 0 & 7 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 2 & 3 & 0 \\ 4 & 0 & 0 & 0 & 0 & 9 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}$$

Graph Representation

1. Incidence matrix
2. Adjacency matrix
3. Weight matrix
4. **Adjacency list**

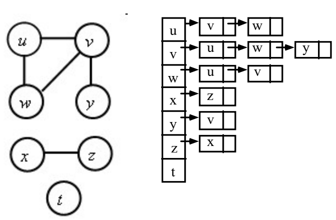
3. Adjacency List

Adjacency list: each vertex has a list of which vertices it is adjacent

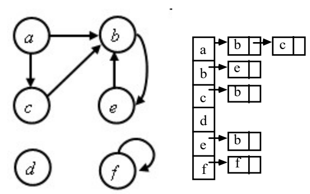
- Is an array Adjacency consiststing of $|V|$ list
- Each vertex has 1 list
- Each vertex $u \in V$: Adjacency[u] consists of nodes that are adjacent to u.

Example:

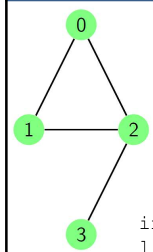
Undirected graph

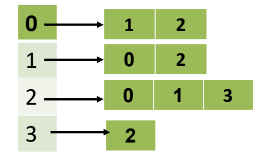


Directed graph



Representation- Adjacency List





```
List of edges:
0 -> 1
0 -> 2
1 -> 0
1 -> 2
2 -> 0
2 -> 1
2 -> 3
3 -> 2
```

```
int V, E;
list <int> Adj[MAX];

void input() {
    cin >> V >>E;
    for(int k = 1; k <= E; k++){
        int i,j;
        cin >> i >> j;
        Adj[i].push_back(j);
        Adj[j].push_back(i);//for undirected graph
    }
}
```


Analyze the cost

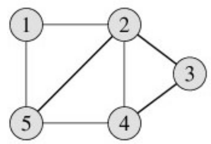
Memory Space

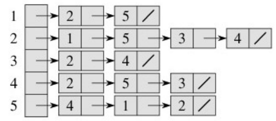
- $\Theta(|V|+|E|)$
- Is often much smaller copmpared to $|V|^2$, especially for sparse graph
- Sparse graph: $|E| \leq k\,|V|$ where $k < 10$.
- Note: Most of the graph in real-world application is sparse graph! ➔ Adjacency list representation is usually preferred since it is more efficient in representing sparse graphs.
- Time to answer the query
  - Add an edge $O(1)$
  - Delete an edge go through the Adjacency lists of initial vertex and terminal vertex
  - Enumerate all adjacent vertex of v: O(<#adjacent vertices>) (better than adjacency matrix)
  - Two vertices i and j *are adjacent*?
    - Search on the Adjacency[i]: Θ(degree(i)). In the worst case $O(|V|)$ => worse than adjacency matrix

Graph Representation

- **Incidence Matrix:** Most useful when information about edges is more desirable than information about vertices.
- **Adjacency (Matrix/List):** Most useful when information about the vertices is more desirable than information about the edges. This representation is also more popular since information about the vertices is often more desirable than edges in most applications

Graph representation



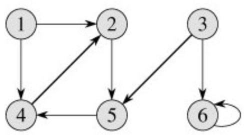


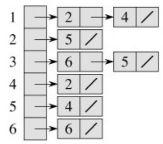
	1	2	3	4	5
1	0	1	0	0	1
2	1	0	1	1	1
3	0	1	0	1	0
4	0	1	1	0	1
5	1	1	0	1	0

graph

Adjacency list

Adjacency matrix





	1	2	3	4	5	6
1	0	1	0	1	0	0
2	0	0	0	0	1	0
3	0	0	0	0	1	1
4	0	1	0	0	0	0
5	0	0	0	1	0	0
6	0	0	0	0	0	1