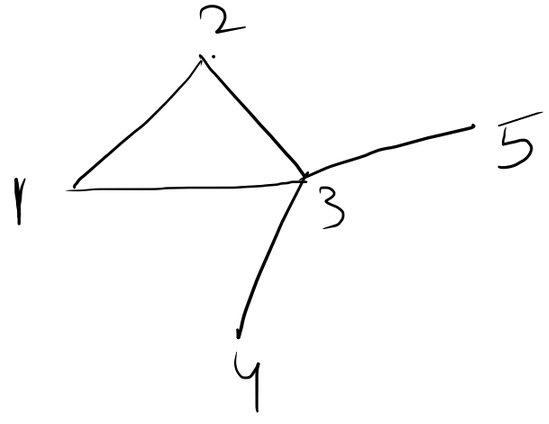
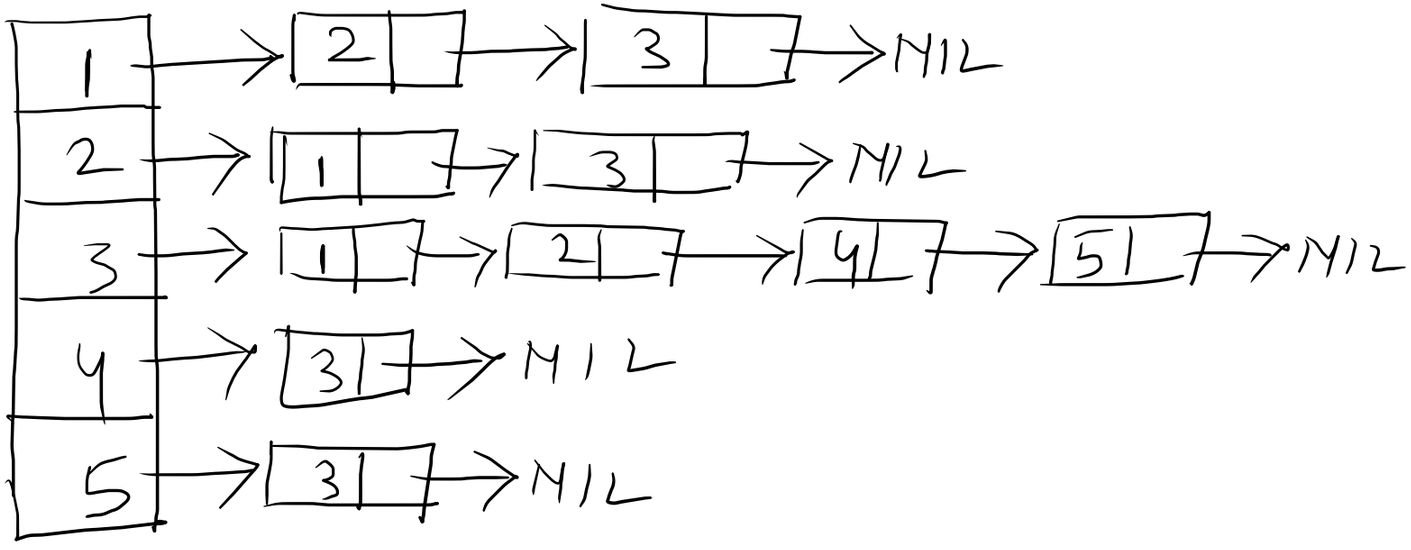
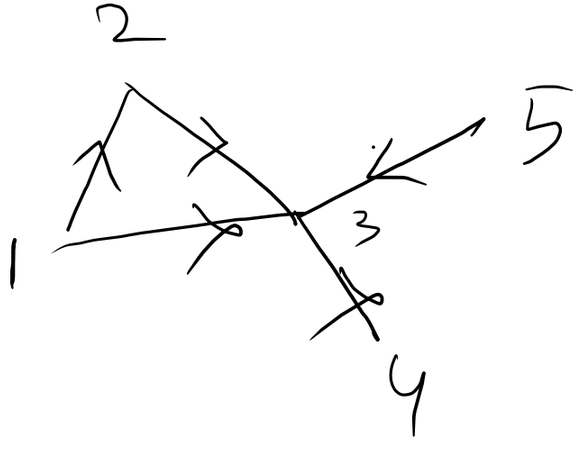
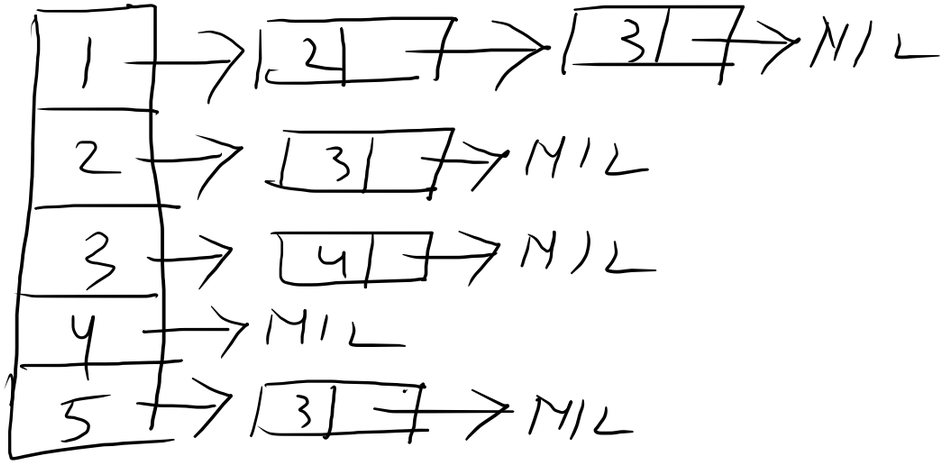
Representations of graphs

Adjacency - List









Storage

$O(|V| + |E|)$

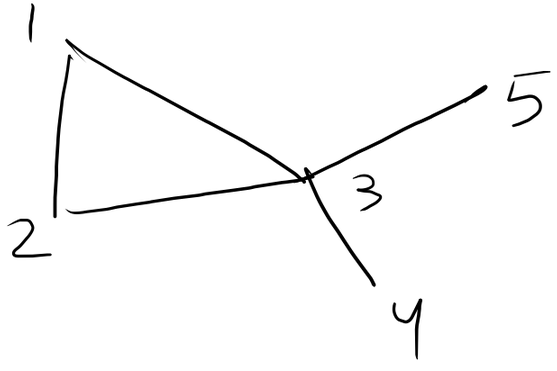
Adjacency matrix

storage: $O(|V|^2)$

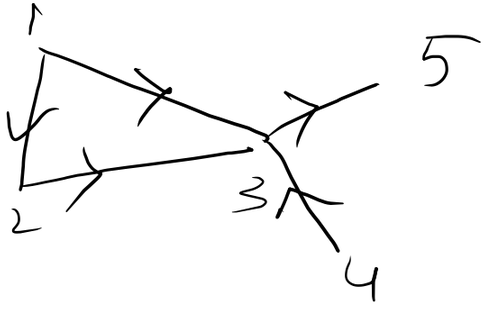
$v_1, v_2, \ldots, v_n$.

$A_{n \times n}$

$$a_{ij} = \begin{cases} 1 & \text{if } v_i \text{ is connected with } v_j \\ 0 & \end{cases}$$



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

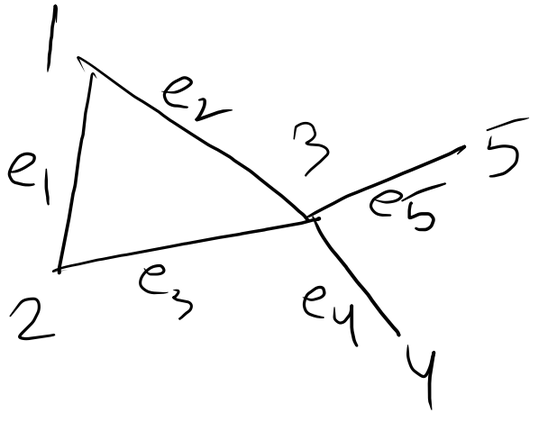


	1	2	3	4	5
1	0	1	1	0	0
2	0	0	1	0	0
3	0	0	0	0	1
4	0	0	1	0	0
5	0	0	0	0	0

Incidence matrix

$A_{n \times m}$

$$a_{ij} = \begin{cases} 1 & e_i \text{ is incident on } j\text{-th vertex} \\ 0 & \text{otherwise} \end{cases}$$



	e_1	e_2	e_3	e_4	e_5
1	1	1	0	0	0
2	1	0	1	0	0
3	0	1	1	1	1
4	0	0	0	1	0
5	0	0	0	0	1

storage:

$O(|V||E|)$

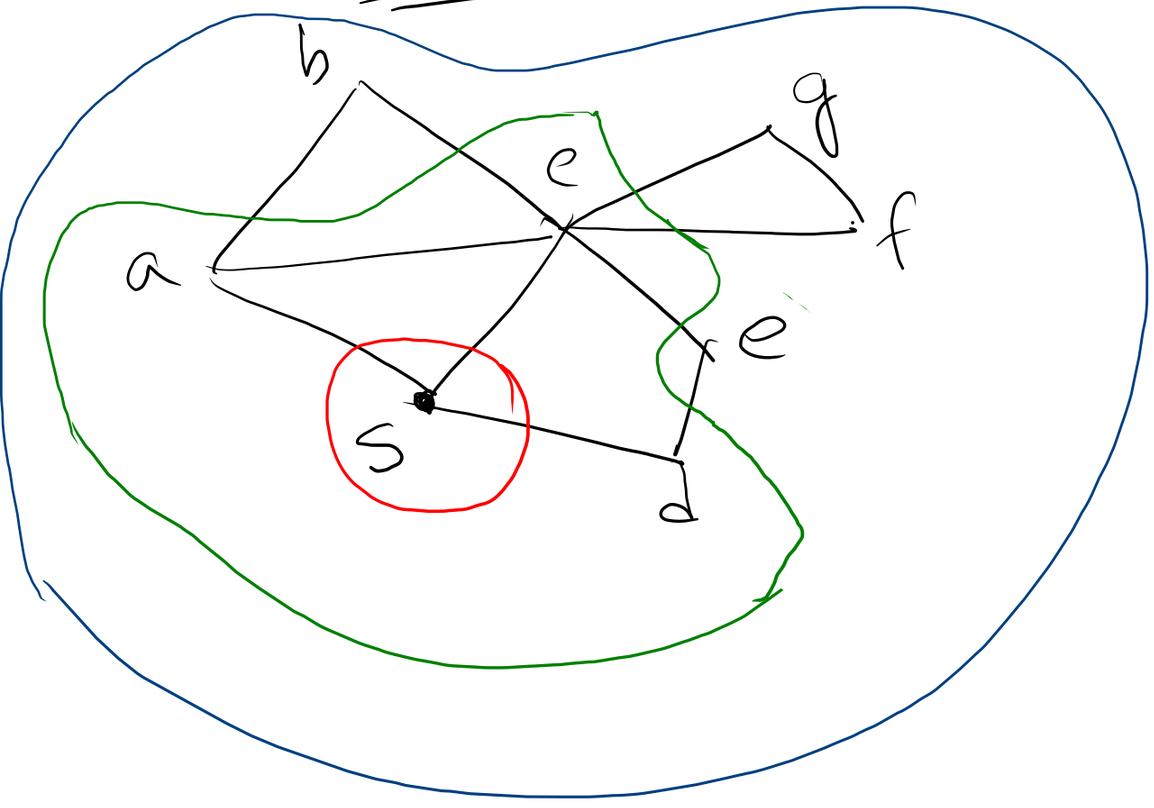
Graph searching algorithms

- systematic searching of each edge or vertices of the graph.
- For directed or undirected graph.
- Applications.
  - maze search.
  - connected compond.

Two popular algorithms.

i) BFS
ii) DFS

BFS



White: This vertex is not yet discovered.

Gray: It is discovered. but not all its neighbours are discovered.

Black: Already discovered and all its neighbours are discovered.

```
BFS (G, s)
for each u ∈ G.V
        u.color = white
        u.dist  = ∞
        u.pred  = NIL
s.color = gray
s.dist = 0
Q = new queue
Q.enqueue(s)
while Q is not empty.
        u = Q.dequeue()
        for v ∈ u.adj
                if v.color = white
                        v.color = gray
                        v.dist = u.dist + 1
                        v.pred = u
                        Q.enqueue(v)
        u.color = black.
```

Running time: $O(|V| + |E|)$