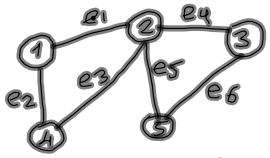
Graphs

A graph G is a set V of vertices and a set E of edges that connect vertices.

$G = (V, E)$ $T = (V, F)$

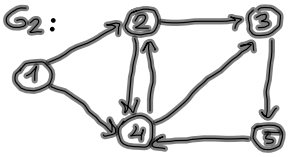
2 types directed & undirected (digraph)

G1:



undirected

G2:



digraph

Path: sequences of vertices with each adjacent pair connected by an edge.

in G_1, $1-2-3-5$ in G_2, $1 \to 2 \to 4 \to 3 \to 5$

Length of path: # edges on path.

$\langle 1,2,3,5 \rangle$: path of length 3

Degree of a vertex: # edges incident on vertex.

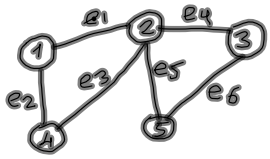
Digraph:

indegree: # edges directed toward vertex

outdegree: # edges directed away

Graph Representations

1) **Adjacency matrix:** $|V|$ by $|V|$ array of booleans. $|V| \times |V|$



$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array} \begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$

symmetric

$|V| \times |V|$

Maximum possible edges:

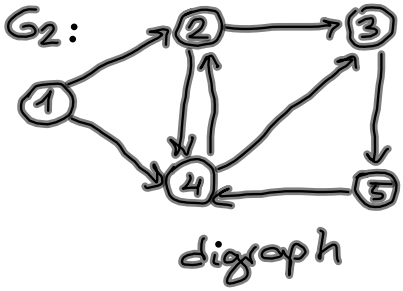
undirected: $\frac{|V| \cdot |V-1|}{2} = O(V^2)$

digraph: $|V| \cdot |V-1| = O(V^2)$

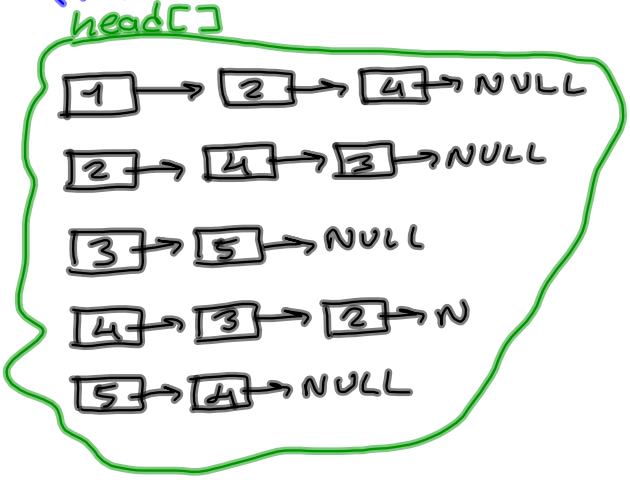
Memory used: $O(|V|^2)$

dense graphs

2. Adjacency List: Each vertex v has a linked list of edges out (incident)

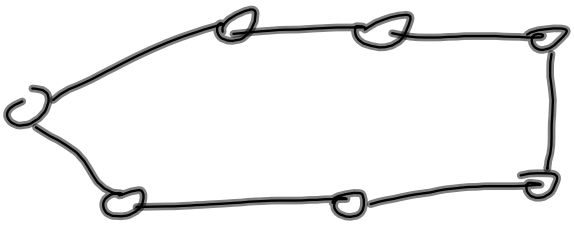


digraph

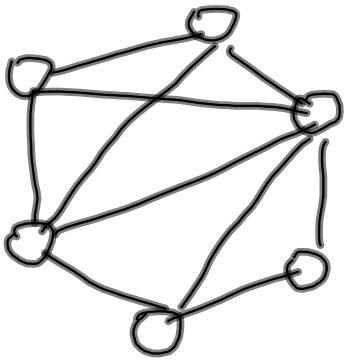


Memory used: $O(|V|+|E|)$

Sparse graphs



sparse

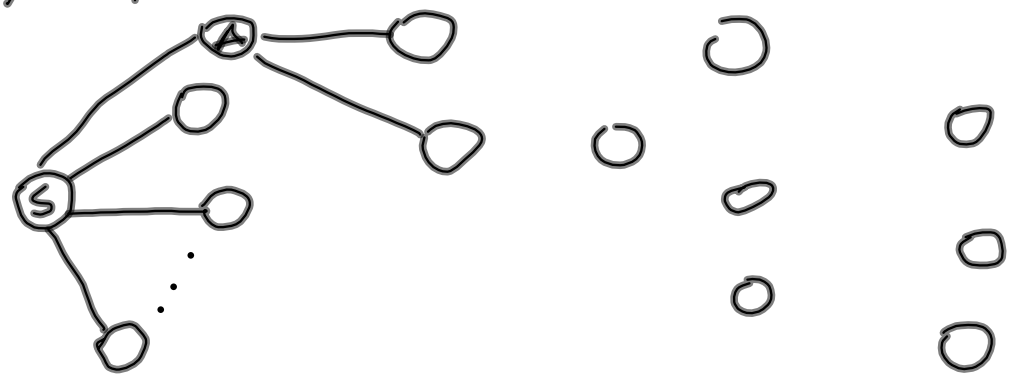


dense

Facebook friendship network (sparse graph)

users = 10^9

friends/users = 10^3



for Adjacency matrix, $10^{18} = 10^{18}$

for Adjacency list, $10^9 + 10^9 \cdot 10^3 \cong 10^{12}$

10^6 = 1 million x

Adjacency list is time efficent for a sparse graph but less efficient for a complete graph.

$$A=\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{array}\begin{array}{c} \begin{array}{ccccc} 1 & 2 & 3 & 4 & 5 \end{array} \\ \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{bmatrix} \end{array}$$ undirected

1. Write a function that returns the degree of a given vertex in an undirected graph which is represented by adjacency matrix.
2. Write a function that returns the outdegree of a given vertex in digraph which is represented by adjacency list.

1.
```
int degree(int A[][], int u, int n)
    int result=0,i;
    for (i=0; i<n; i++)
        result+= A[u][i];
    return result;
}
```

2.
```
int outdegree(struct node *head[], int u){
    int result=0;
    while( head[u]!=NULL){
        result++;
        head[u]=head[u]->next;
    }
    return result;
}
```