Basic Graph Algorithms

Notation:
$[n] = \{1, \dots, n\}$.
$m :=$ # of edges
$n :=$ # of vertices.

- $G = (V, E)$ $\quad$ $V = \{v_1, \dots, v_n\}$

  $E = \{e_1, \dots, e_m\}$ where $e_i = (v_{i_1}, v_{i_2})$ for some $i_1, i_2 \in [n]$.

  – Undirected graph: Edges have no orientation.
  Directed graphs: " have orientation. (direction).

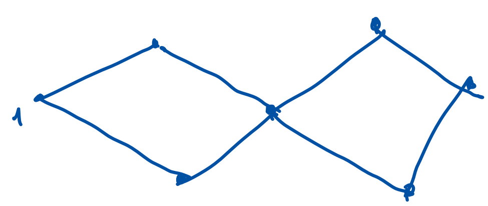
- More generally – Transportation networks
  – Social media/networks
  – Markov chains
  – Knowledge graphs
  – Networks – comm. / internet.

Running time
↳ in terms of primi-tive operations available.

- Paths, Cycles, walks → Generalize paths and cycles.
  ↳ closed walk.

  $v_{i_1}, \dots, v_{i_k}, v_{i_1}$: If all but v_{i_1} are distinct then it is a simple cycle.

  $v_{i_1}, \dots, v_{i_k}$: s.t $\forall j$ $(v_{i_j}, v_{i_{j+1}}) \in E(G)$.

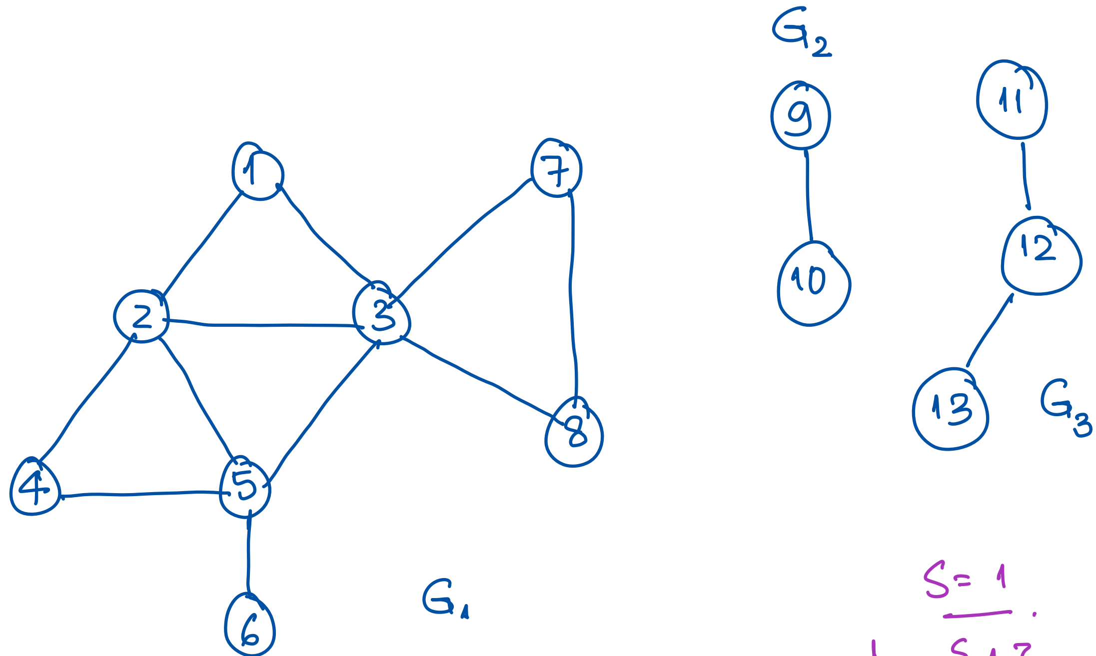


- We shall study "connectedness" in this lecture.

A (undirected) graph G is said to be connected if $\forall$ pairs u and v in $V(G)$, $\exists$ a path from u to v.

s-t connectivity: Given two vertices s and t, are s and t connected. (If s and t are in the same connected component).

Via search algorithms
→ Breadth First Search
→ Depth First Search



$\underline{S = 1}$.

$L_0 = \{1\}$

$L_1 = \{2,3\}$

$L_2 = \{4,5,7,8\}$

$L_3 = \{6\}$

Not → $\{9, 10, 11, 12, 13\}$ covered.

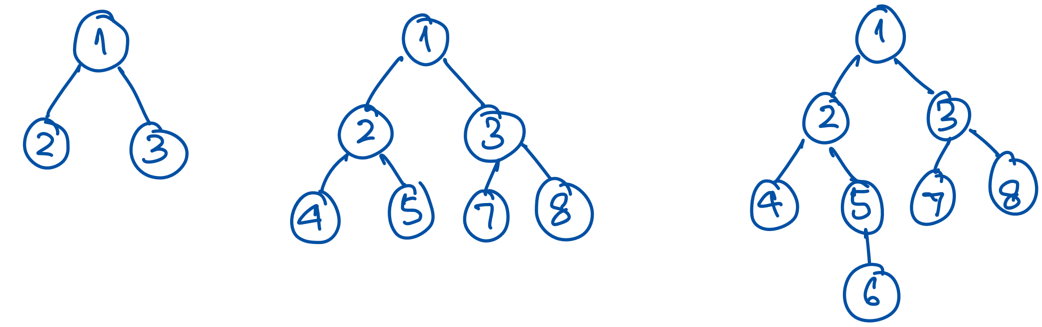
Breadth First Search (BFS):

- Layer L_1 consists of all neighbours of s
- $\forall j \geq 2$, Layer L_j contains all nodes that do not belong to $\bigcup_{i<j} L_i$ and which have an edge to vertices in L_{j-1}.

{ Implicitly there's an ordering/priority over edges

→ $O(m+n)$ – Running time.
steps.

1



<u>Obs</u>: BFS naturally gives rise to a rooted tree struct.

<u>Obs</u>: $\forall\ j \geq 1$, Layer L_j contains those nodes that are exactly distance j away from s. $\Rightarrow$ Shortest s-t path.

<u>Obs</u>: Two nodes s and t are connected if and only if t belongs to some layer in a BFS that starts from s.

Representing graphs:

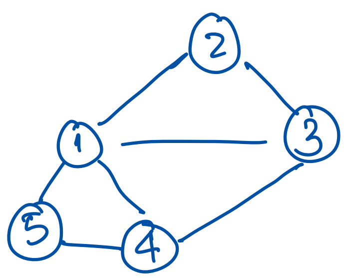
- Adjacency matrix: $|V| = n$.

Space = $O(n^2)$ $\hookrightarrow$ $\exists$ const c s.t space used $\leq c \cdot n^2$. $(n \geq n_0)$.

$$\forall\ 1 \leq i,j \leq n\ ; \quad A_{ij} = \begin{cases} 1 & \text{if } (v_i, v_j) \in E(G) \\ 0 & \text{otherwise} \end{cases}$$

Neighbours of vertex v_k is given by the k^{th} row.

$\{0,1\}^n$.



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

• Adjacency list:

$\lim_{n\to\infty} \frac{m}{n^2} \to 0$ then use adj list.

$m = n^{2-\varepsilon}$
$m < c \cdot n^2$
for all constants c, then use adjacency list

In other words, m is a magnitude / order smaller than n^2.

For each vertex v, maintain a list of its neighbours.

$N(v) = \{$list of neighbours of $v\}$.

Size of adj list $= \sum_v |N(v)| = 2m$.

If $m << n^2$ then adj. list is a more space efficient representation.

Claim: Let T be a BFS tree. Let x and y be nodes in T s.t $x \in L_i$ and $y \in L_j$. Let $(x,y) \in E(G)$. Then i and j differ by at most 1.

Pf: W.L.O.G assume $i \leq j$. BFS algorithm guarantees that x is at distance i from the root, and y is at a dist. j from the root. Suppose $i < j-1$. But from the BFS algo, after exploring x in L_i, y is added to L_{i+1} as $(x,y) \in E(G)$. This contradicts $i < j-1$. $\square$

Depth First Search.

→ Strategy is to explore through "leading edges" until you hit a "deadend" and backtrack to a node w/ unexplored neighbours.

Global knowledge.

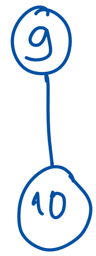
$R \leftarrow \{\}$

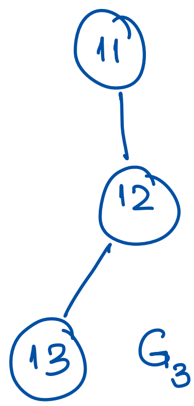
DFS(u):

- Mark u as "explored" and R ← R ∪ {u}
- For each edge (u,v) incident on u:
  - If v == t then:
    return "found t"
  - If v is not "explored" then
    DFS(v)
- Return "Not found t".

Extra lines for s-t connectivity.

G2

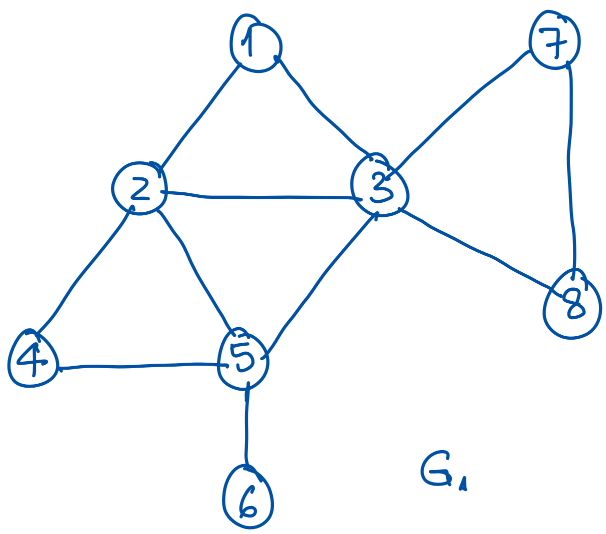




G3

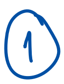
R = { }

S = 1 . DFS(s)



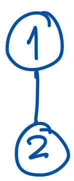
G1

DFS(1)



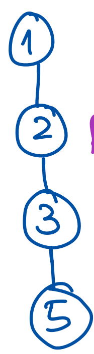
R={1}

DFS(2)



R={1,2}

DFS(3)



R={1,2,3}

DFS(5)

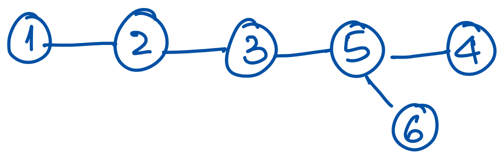
R={1,2,3,5}

DFS(4)

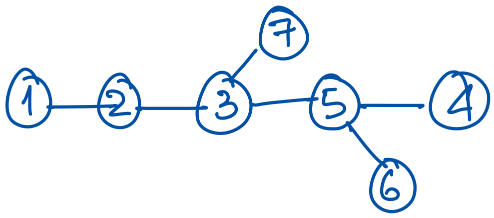
← Backtrack

DFS(4) ends here.

R={1,2,3,5,4} DFS(6)

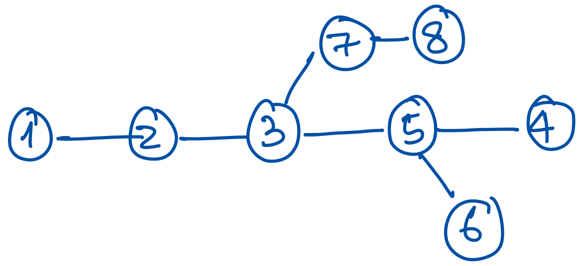


Backtracking to 3. end of DFS(6) and DFS(5).



$R = \{1, 2, 3, 4, 5, 6\}$

DFS(7)



$R = \{1, 2, 3, 4, 5, 6, 7\}$

DFS(8).

But by backtracking, we end DFS(3), DFS(2) and DFS(1).

Obs: We are building a tree - Depth first Search tree.