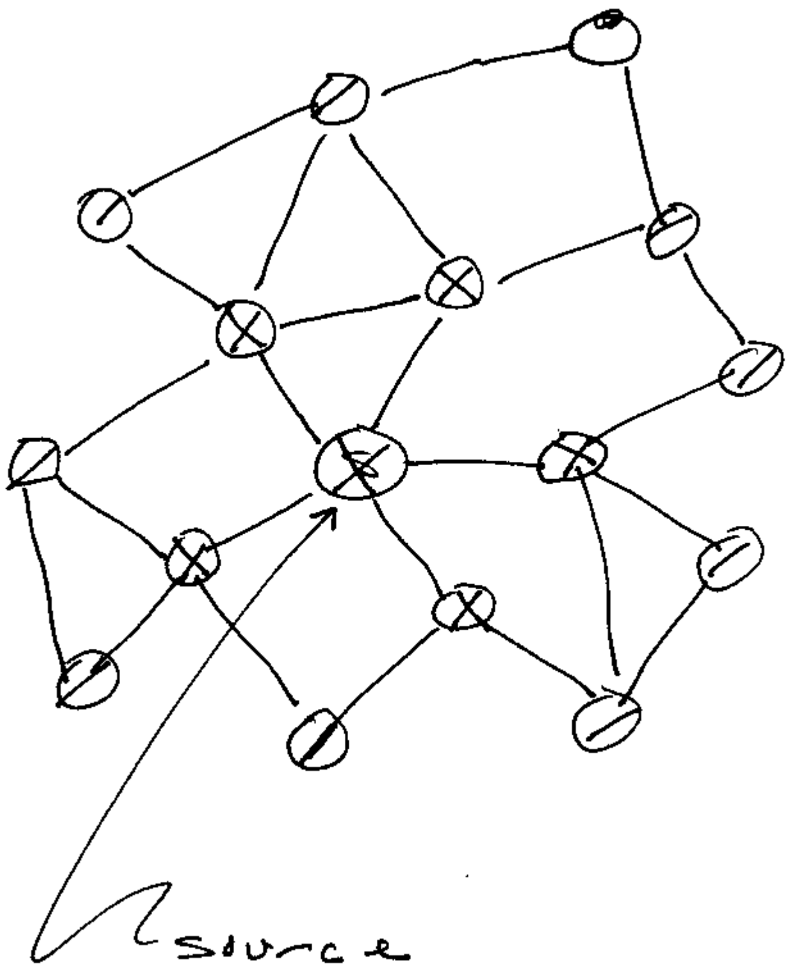
cse 101 4-16-20

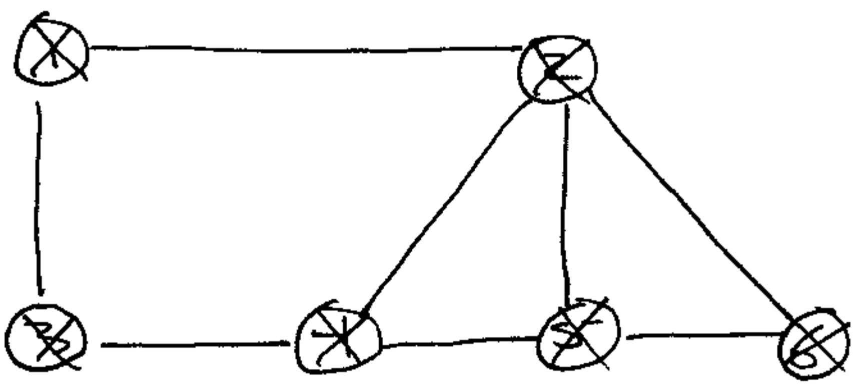
BFS



source

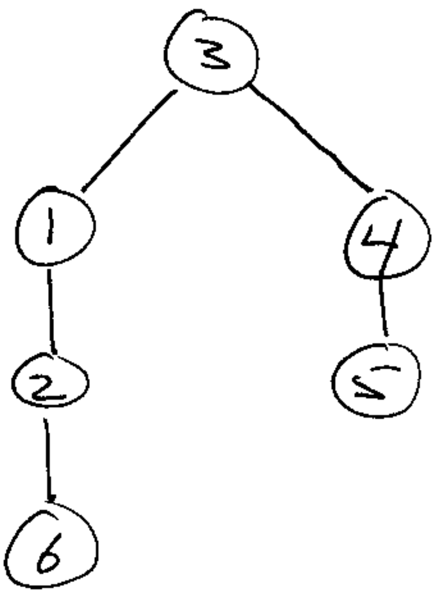
BFS uniformly expands boundary between discovered and undiscovered vertices. It determines all vertices at distance d from the source before discovering any vertices at distance (d+1).

Ex. Same : $s = 3$



Q: ~~3~~ ~~1~~ ~~4~~ ~~2~~ ~~5~~ ~~6~~

BFS Tree



dist = depth
0
1
2
3

Directed Graphs

Defn a directed graph (digraph) is a pair of sets $G=(V,E)$ where

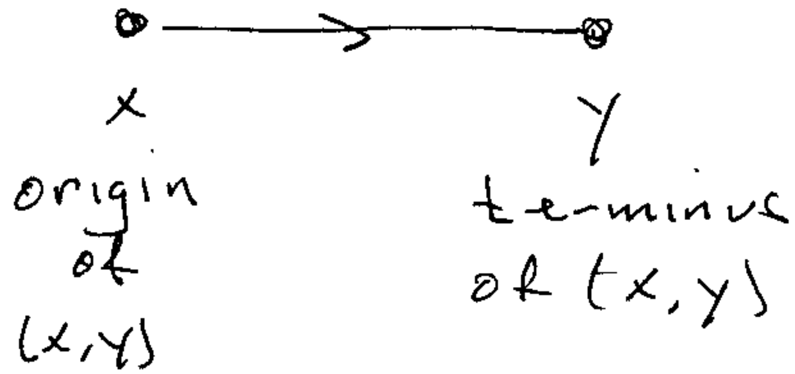
$$V \neq \emptyset$$

and $E \subseteq V \times V = V^2 = \{(x,y) \mid x \in V \text{ \& } y \in V\}$

allowed: (x,x)

allowed: $(x,y) \neq (y,x)$

Directed edge: (x, y)



<u>not</u> allowed : $(x,y) = (x,y)$

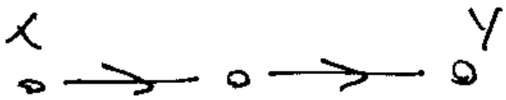
Definitions of:

- Walk (closed)
- trail (")
- Path (")
- trivial Path

- cycle

say y is <u>reachable</u> from x
iff G contains a directed x-y
path.

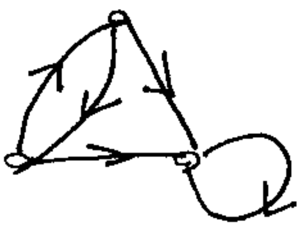
no longer necessarily a symmetric relation



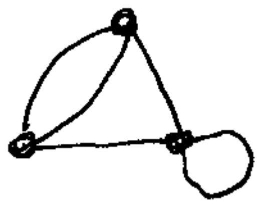
y Reachable from x
x not reachable from y

<u>Defn</u>
We say G is <u>weakly connected</u>
iff its underlying undirected multi-graph
is connected

<u>digraph</u>



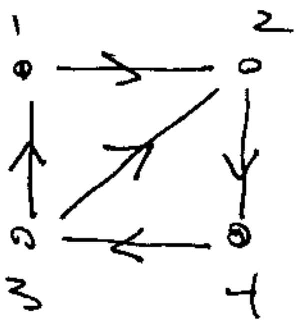
<u>underlying multi-graph</u>



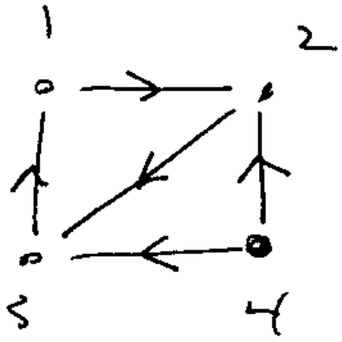
Defn

We say G is <u>strongly connected</u> iff for all $x, y \in V(G)$, both y is reachable from x, and x is reachable from y.

Ex.



strongly connected



not strongly connected

<u>Problem</u>: Given a digraph G, determine whether G is strongly connected.

Representations of digraphs

- Incidence matrix

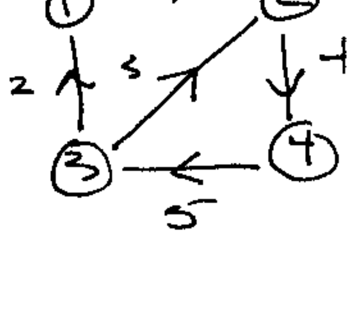
$$V = \{x_1, x_2, \ldots, x_n\}$$

$$E = \{e_1, e_2, \ldots\ldots, e_m\}$$

$I = I(G)$ is an $n \times m$ matrix

$$I_{ij} = \begin{cases} 0 & \text{if } x_i \text{ \underline{not} incident with } e_j \\ -1 & \text{if } x_i \text{ is origin of } e_j \\ +1 & \text{if } x_i \text{ is terminus of } e_j \end{cases}$$

Ex.



$$I = \begin{pmatrix} -1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & -1 & 0 \\ 0 & -1 & -1 & 0 & 1 \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix}$$

(forget about loops.)

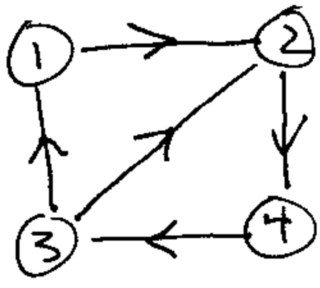
- Adjacency matrix

$V = \{x_1, x_2, \dots, x_n\}$

$A = A(G)$ is an $n \times n$ matrix

$$A_{ij} = \begin{cases} 0 & \text{if } x_i \text{ \underline{not} adjacent to } x_j \\ 1 & \text{if } x_i \longrightarrow x_j \\ 0 & \text{if } x_i \longleftarrow x_j \end{cases}$$

<u>Ex.</u>



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

- Adjacency list
  array of n lists, where
  $n = |V|$, $V = \{x_1, x_2, \dots, x_n\}$

<u>adj(G)</u>

1 : list of outgoing neighbors of x_1
2 : - - - - - - - - - - x_2
3 : - - - - - - - - - - x_3
⋮
n : - - - - - - - - - -

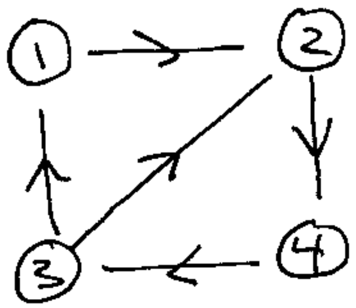
In general

x_i list - - - - x_j

$\uparrow$

if and only if $x_i \rightarrow x_j$

Ex.



adj[ ]

1 : 2

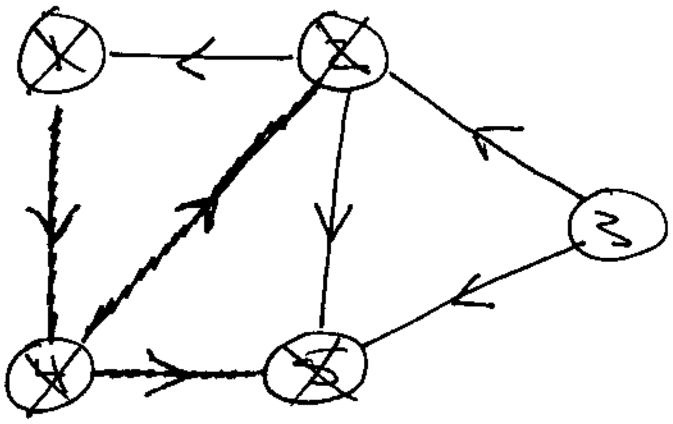
2 : 4

3 : 1 2

4 : 3

Back to BFS ....

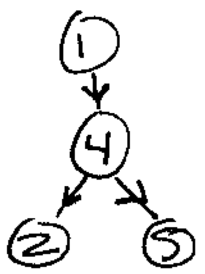
Ex. s = 1



	adj	color	d	p
1	~~4~~	~~g~~ b	0	n
2	~~1~~ ~~5~~	~~w~~ ~~g~~ b	~~∞~~ 2	~~n~~ 4
3	2 5	w	∞	n
4	~~2~~ ~~5~~	~~w~~ ~~g~~ b	~~∞~~ 1	~~n~~ 1
5		~~w~~ ~~g~~ b	~~∞~~ 2	~~n~~ 4

Q: ~~1~~ ~~4~~ ~~2~~ ~~5~~

BFS Tree:



dist

0

1

2

Predecessor Subgraph:

$$V_p = \{x \in V \mid p[x] \neq nil\} \cup \{s\}$$

$$E_p = \{ \underbrace{(p[x], x)}_{\substack{\text{ordered pair if digraph} \\ \text{un-ordered pair if graph}}} \mid p[x] \neq nil \}$$

note: $|E_p| = |V_p| - 1$

also: from algorithm, no cycles

$\therefore$ $G_p = (V_p, E_p)$ is a tree

Pred. Subgraph
or BFS Tree.