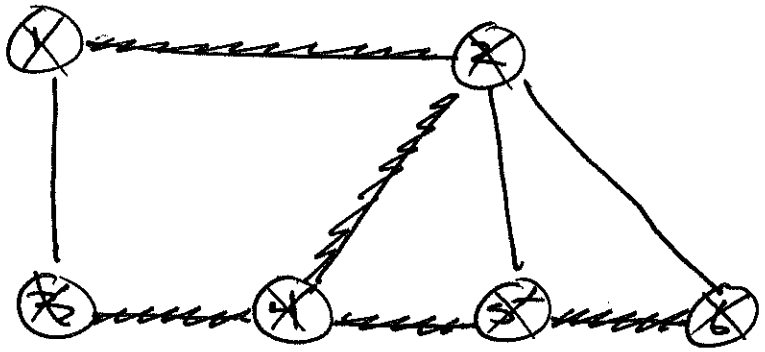
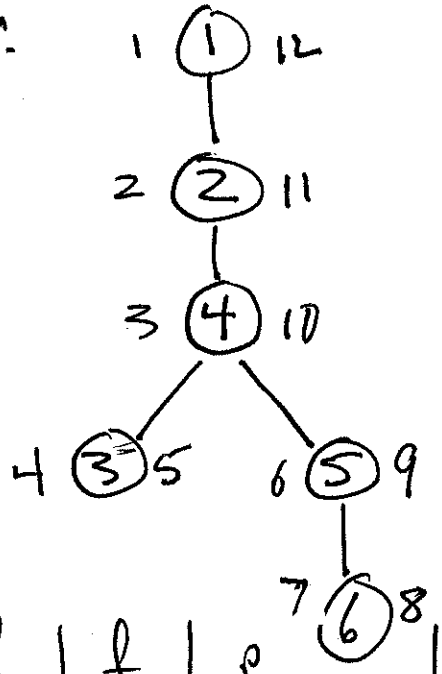
CSE 101 4-20-23

Ex.

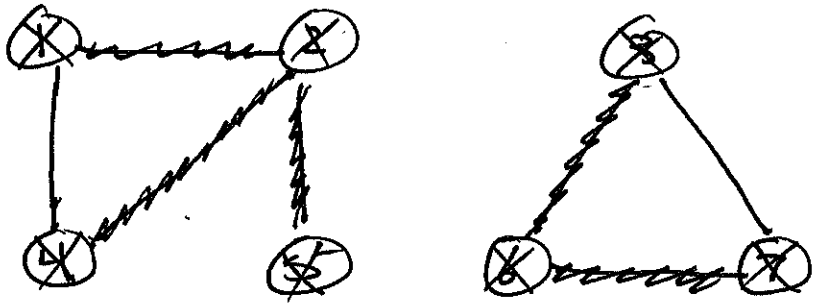


DFS Forest:

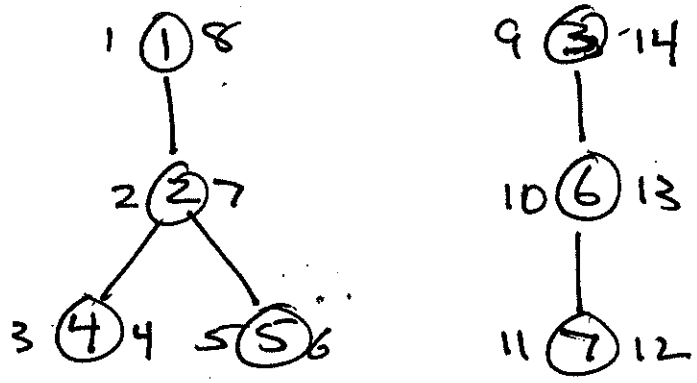


	adj	d	f	p
1	2 3	1	2	N
2	1 4 5 6	2	11	1
3	1 4	4	5	4
4	2 3 5	3	10	2
5	2 4 6	6	9	4
6	2 5	7	8	5

Ex.



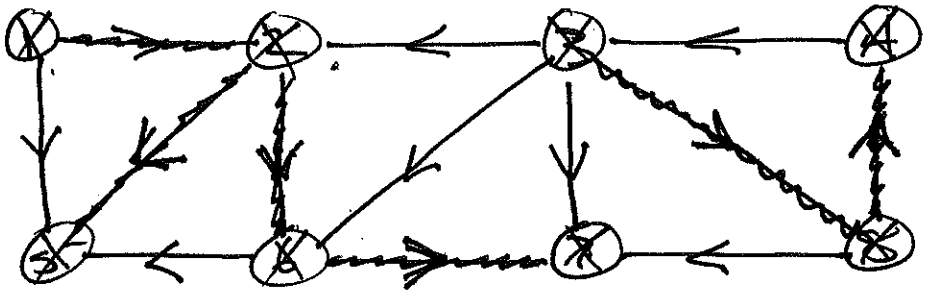
Forest :



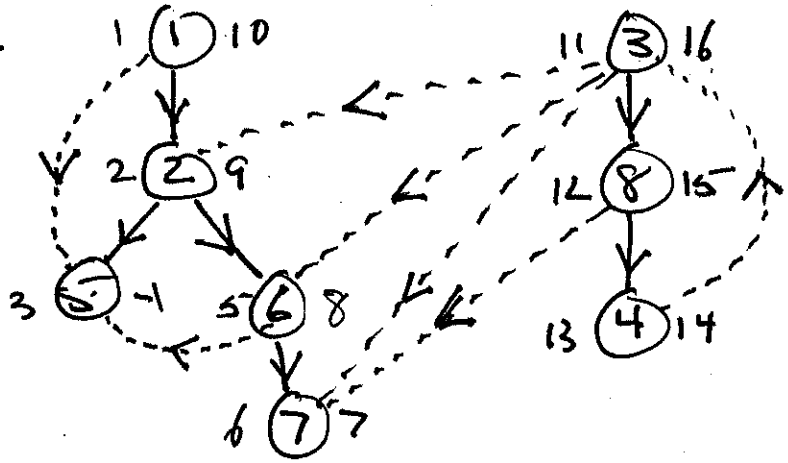
Theorem : (Undirected)

When DFS is run on a graph G, the trees of the forest span the conn. components of G.

Ex.



Forest:



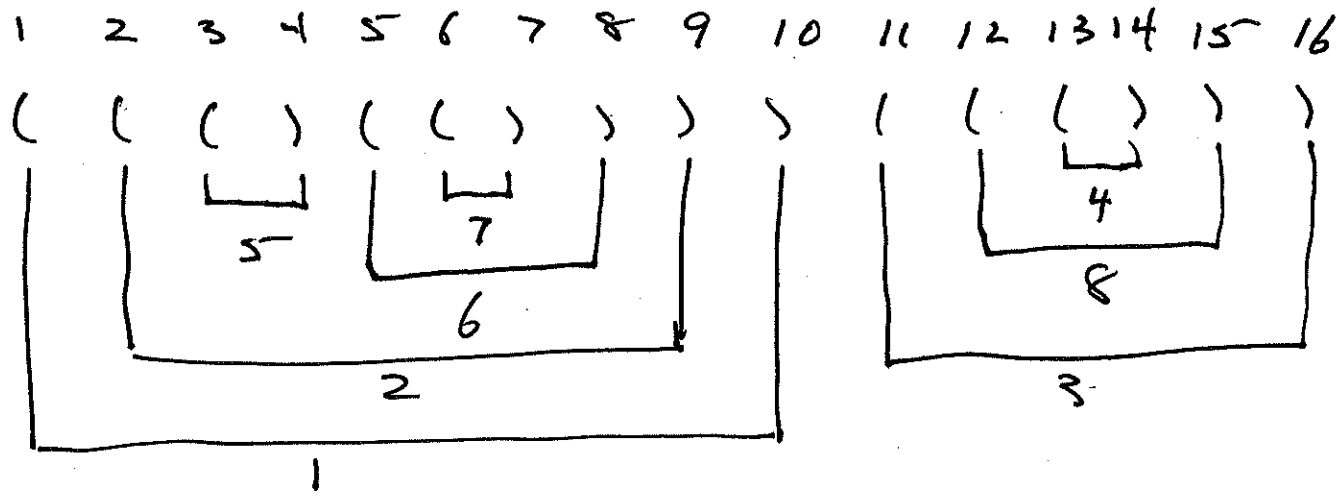
tree (1,2), (2,5) (2,6), (6,7), (3,8) (8,4)

back: (4,3)

forward: (1,5)

cross:
- (3,2), (3,6), (3,7) (8,7)
- (6,5)

Parenthesis string:



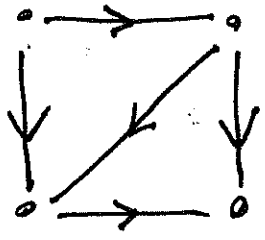
Defn

A dig-aph $G = (V, E)$ is said to be <u>acyclic</u> (or a DAG) iff G contains no directed cycles.

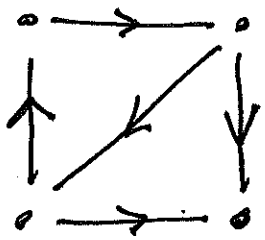
Dir. cycles

allowed

Ex.



acyclic



not acyclic

Edge classification

let G be a dig-aph, run DFS(G).

Tree: belong to DFS forest

Back: Join a vertex to an ancestor

Forward: Join a vertex to a descendant (other than a direct child)

Cross: all others

- tree-to-tree
- cousin-to-cousin

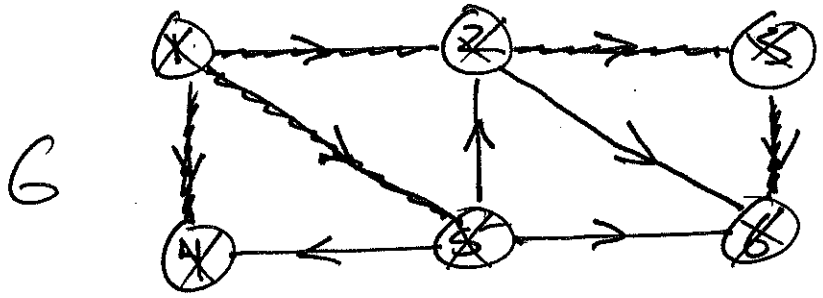
Thm
Dig-aph G is acyclic iff DFS(G) yields no back edges. Equivalently, G contains a back edge iff it contains a dir. cycle.

Def

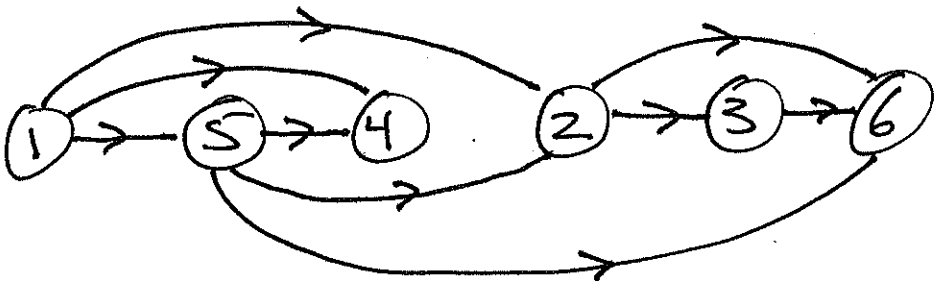
Let $G = (V, E)$ be a DAG.

A Topological sort of G is a linear ordering of V such that: if $(x, y) \in E$, then x is before y in the ordering.

Ex



Topological sort of $V(G)$:



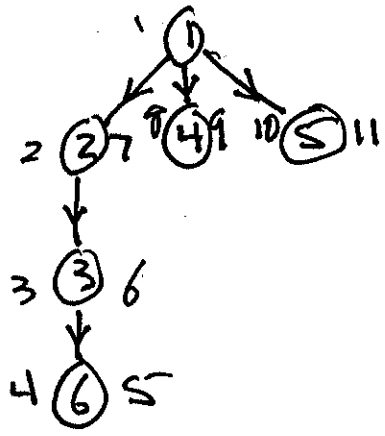
To find a Topological sort:

- Run DFS
- as vertices finish, Push them onto a stack.

Thm when this is done, the stack is a Topological sort of $V(G)$.

Ex. (same)

Forest:



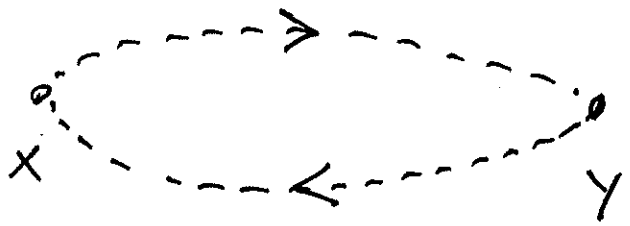
stack

1
5
4
2
3
6

Strongly Connected Components (SCC)

Defn

Let $G=(V,E)$ be a digraph. We say G is <u>strongly connected</u> iff for all $x,y \in V$: x is reachable from y, and y from x.



more generally: $S \subseteq V$ is <u>strongly connected</u> iff for all $x,y \in G$:

x is reachable from y and y from x.

Defn

Let $S \subseteq V(G)$, G is a dig-aph.

We call S a <u>strongly connected component</u> of G if

(1) S is strongly connected

and

(2) S is maximal W.r.t. (1)

Ex.