Depth-First-Search

check cycle tree

1. for (each vertex v)

   color[v] = white

1' CC# = 0

2. for (each vertex v)

   if (color[v] == white) DFS(v)

   CC# = CC# + 1

DFS(1)

for each vertex v

if (color[v] == white)

stop(not connect)

return (connect)

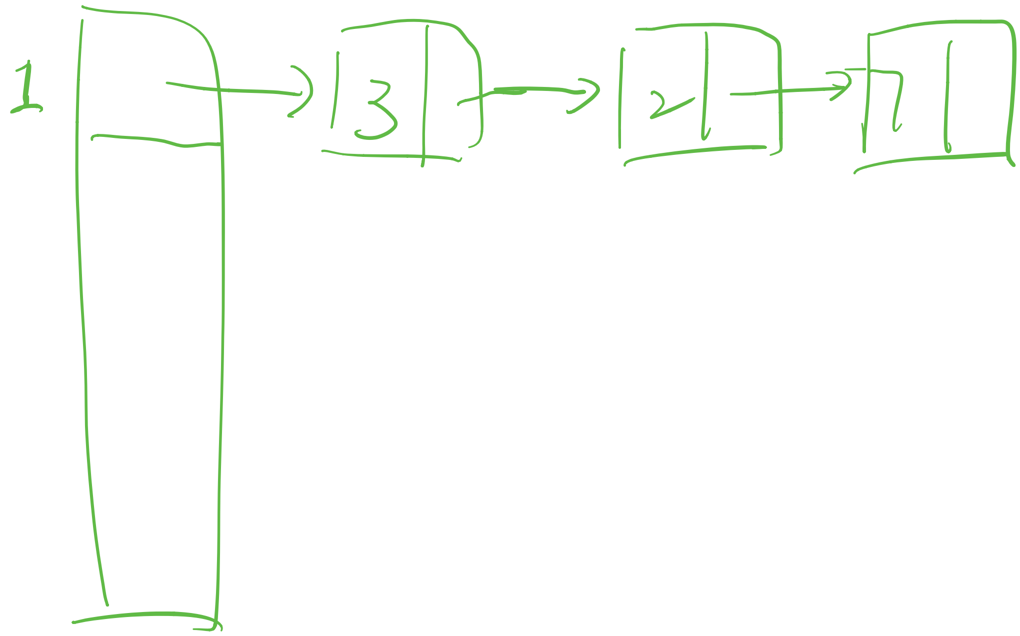
DFS(v)

1. color[v] = gray   CC[v] = CC#

2. for (each edge [v,w])

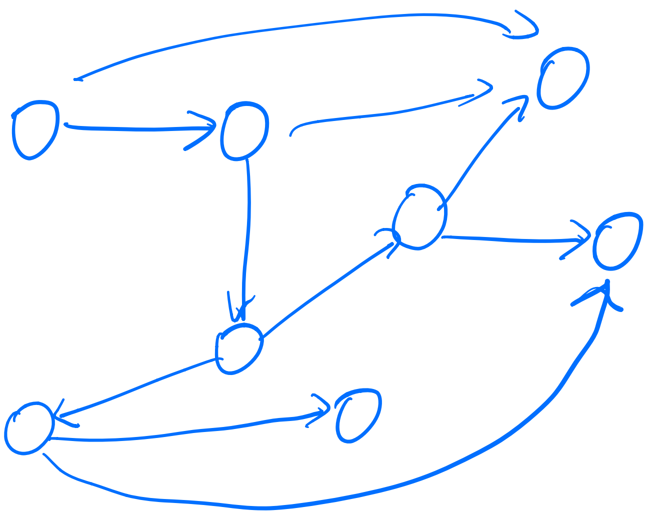
   if (color[w] == white) DFS(w)

3. color[v] = black.



D---F---S on directed graphs

input



time = O(m+n)

1. for (each vertex v)

   color[v] = white

1' h=n

2. for (each vertex v)

   if (color[v]==white) $\boxed{\text{DFS}(v)}$

for each vertex, call DFS once.

DFS(v)

1. color[v] = gray

2. for (each edge [v,w])

   if (color[w]==white) $\boxed{\text{DFS}(w)}$

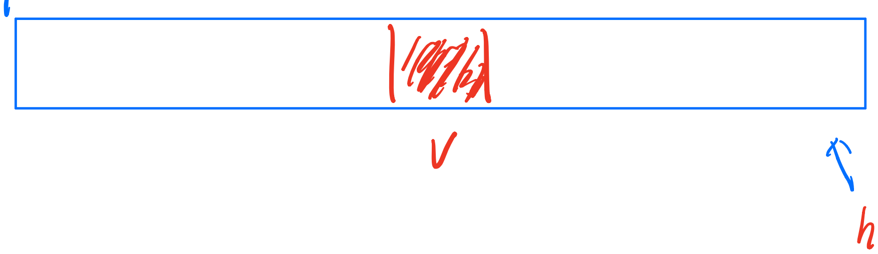
   else if (color[w]==gray) stop('deadlock')

3. color[v] = black; Tp[h--] = v

time for all DFS's $= \sum_{i=1}^{n} deg(v_i)$

$= m$

Output Tp



Claim if the graph has no cycle, then Tp gives a right order

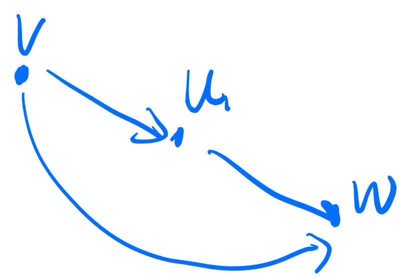
For each edge [v,w] $v \longrightarrow w$

v must appear before w in the array

Consider DFS(v)

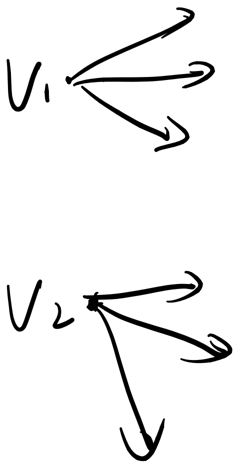
at this point

$$\text{color}[w] = ?? \begin{cases} \text{white} \\ \text{black} \\ \text{gray} \end{cases}$$



$$\mathrm{DFS}(v) \downarrow \mathrm{DFS}(u_1) \downarrow \mathrm{DFS}(w)$$

$$\begin{tikzcd} \mathrm{Dfs}(w) \arrow[d] \\ \mathrm{Dfs}(u_1) \arrow[d] \\ \mathrm{Dfs}(u_2) \arrow[d] \\ \mathrm{DFS}(v) \arrow[uuu, bend left] \end{tikzcd}$$



1. for (each vertex v)

    color[v] = white

2. for (each vertex v)

    if (color[v] == white) DFS(v)

DFS(v)

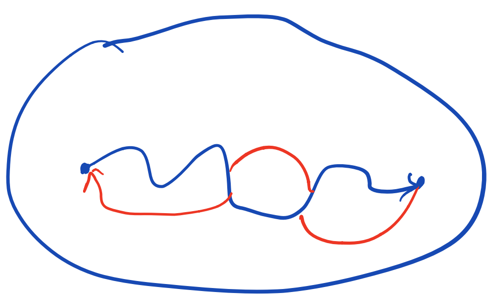
1. color[v] = gray

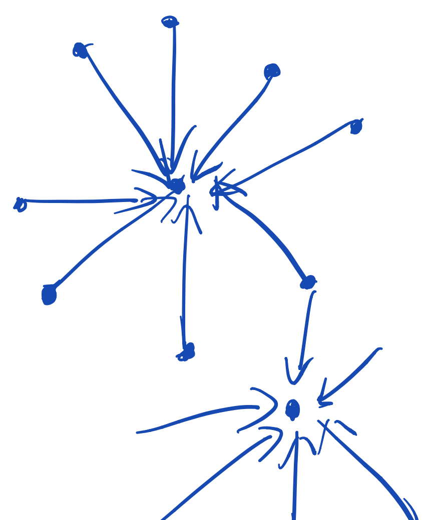
2. for (each edge [v,w])

    if (color[w] == white) DFS(w)

3. color[v] = black.

directed





a set C of vertices in a directed graph G is a strongly connected component (SCC) if

1. for each two vertices in C, there are paths going from one to another.

2. C is the maximal