Strongly Connected Component (SCC) via DFS

Given directed $G = (V, E)$ $n = |V|$, $m = |E|$

Recall:

Global time = 1; $FV[1 \ldots n]$; $i = 1$

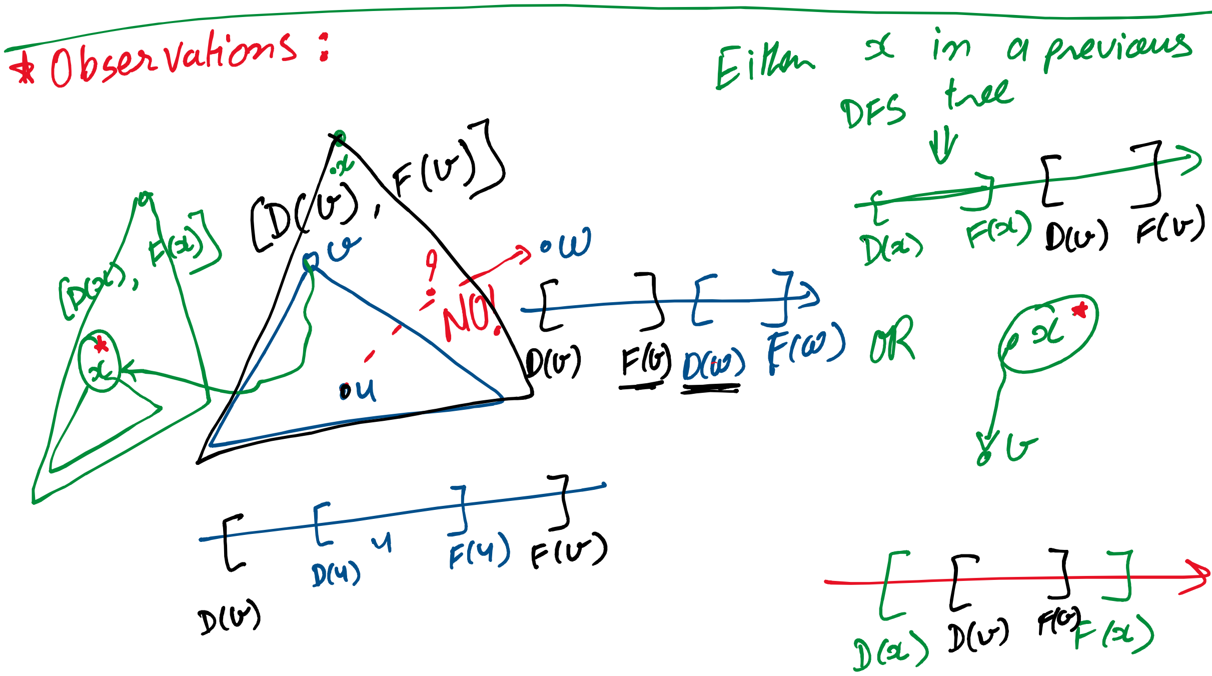
DFSAll(G){
1. Unmark all $v \in V \to O(n)$
2. for each $v \in V${
3. if v unmarked.
4. then DFS(G, v)

}

DFS(G, v) {
// Find all unmarked vertices
// reachable from v.
1. Mark v; $D(v)$ = time++;
2. for each $u \in Adj(v)${
3. if u unmarked then
Parent$(u) = v$; DFS(G, u)
4. }
5. $F(v)$ = time++; $FV[i] = v$
$i = i + 1$;

Runtime: $O(n) + \underbrace{O(m)}_{\text{each edge accessed exactly once}}$
$= O(m+n)$

*** Observations:**



★ Strongly Connected Components (SCCs)

Given a dir. graph $G = (V, E)$

nodes $u, v \in V$ are $\equiv \boxed{u \leftrightsquigarrow v}$ $\iff$ $u \overset{\text{Path}}{\rightsquigarrow} v$ in G & $v \underset{\text{Path}}{\rightsquigarrow} u$ in G
Strongly Connected (SC)

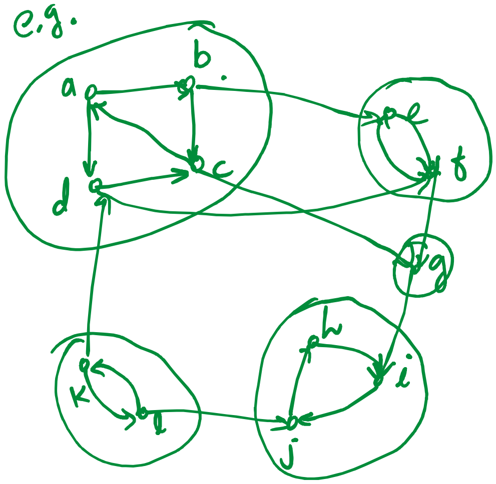
→ Symmetric : $u \leftrightsquigarrow v$ then $v \leftrightsquigarrow u$.
→ Transitive : $u \leftrightsquigarrow v \leftrightsquigarrow w$ then $u \leftrightsquigarrow w$.
⇒ SC is an Equivalence Relation.

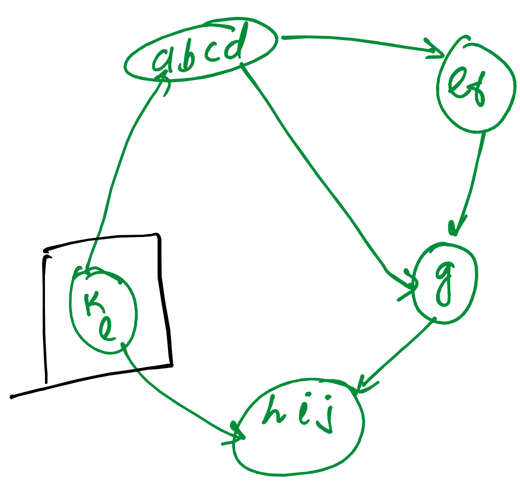
$SCC(u) = \{ v \in V \mid u \leftrightsquigarrow v \}$ is an equiv. class.

$\Rightarrow \forall v \in SCC(u),\ SCC(v) = SCC(u)$

→ every pair is s.c.

e.g.





Meta graph

★ Meta graph $= Meta(G) = (V^{scc}, E^{scc})$ where

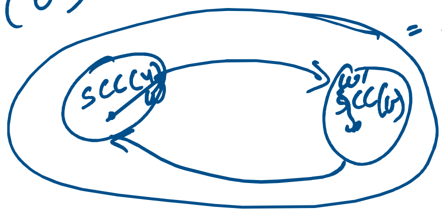
$V^{scc} = \{ scc(u) \mid u \in V \}$

$E^{scc} = \{ (scc(u), scc(v)) \mid \exists\, w \in scc(u),\ w' \in scc(v)$ s.t. $(w, w') \in G$ & $scc(u) \neq scc(v) \}$

(…) is acyclic.

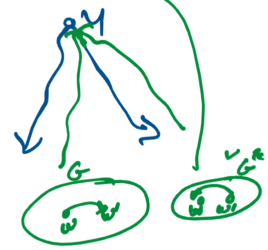
Claim: Meta(G) is acyclic.

$= scc(u) = scc(v)$



$\boxed{\text{Goal: Compute Meta}(G).}$

Naive Al'g: (1) $\forall u \in V$, compute $scc(u) = \left(\text{DFS}(G, u) \wedge \text{DFS}(G^R, u) \right)$



(2) Then compute edges of Meta(G)

$O(m)$ + $n * O(m+n)$ $\Big\}$ $\rightarrow O(m) + O(mn) = O(mn)$.

Purdam '68 $O(n^2)$

Munro '71 $O(m + n \log n)$

Tarjan '72 $O(m+n)$ complicated

Kosaraju '78 $O(m+n)$ easy

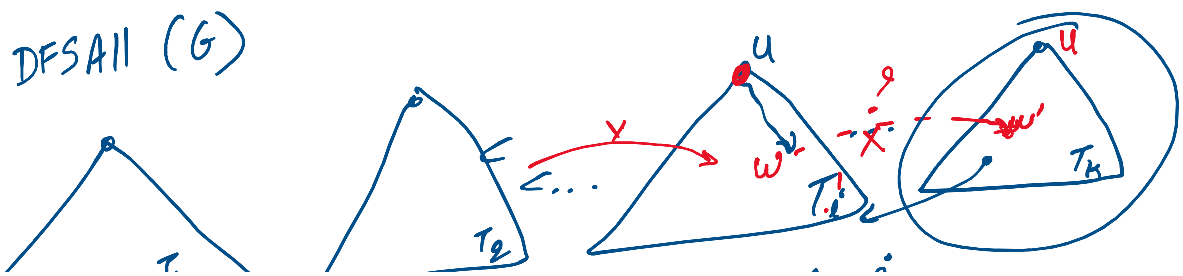
Sharir '81

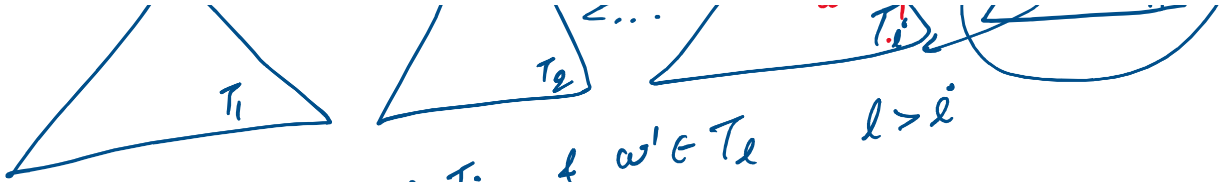
High-level Approach:

(1) Compute the source component in Meta(G)

(2) Remove it & Repeat.

DFSAll (G)

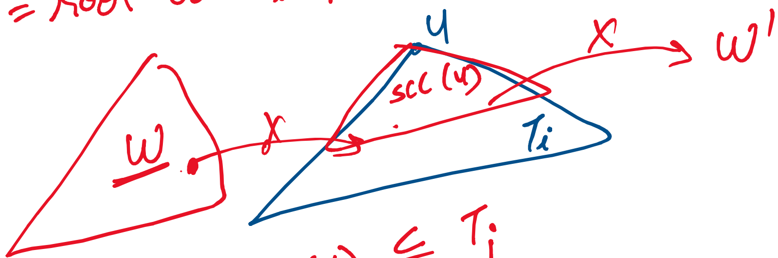




If $w \in T_i$ & $w' \in T_l$ $\quad l > i$

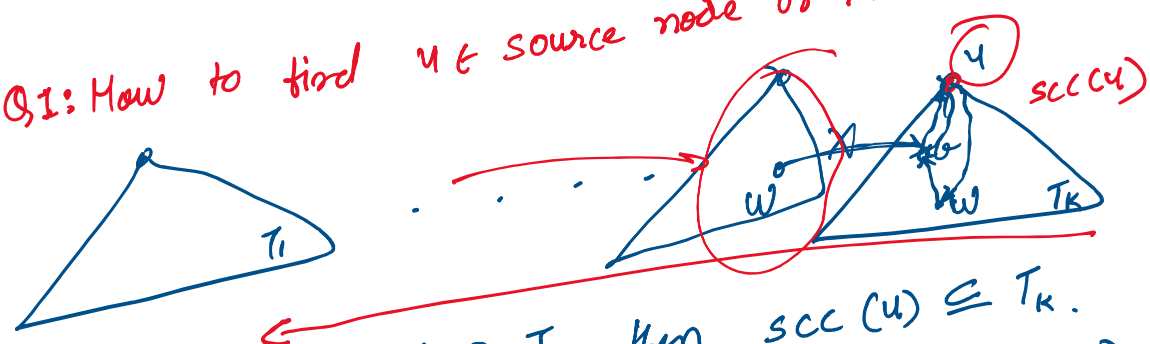
s.t. $(w, w') \notin E$

Q0: u = root of T_i, $SCC(u)$?



$SCC(u) \subseteq T_i$

Q1: How to find $u \in$ source node of Meta(G)



u = root of T_K then $SCC(u) \subseteq T_K$.

$\boxed{\exists w \notin SCC(u)}$ s.t. $w \to v$ & $v \in SCC(u)$

① $w \notin T_i$ $\quad i \neq K$.

② $w \in T_K \Rightarrow$ $w \to v \rightsquigarrow u$ ($\because$ v is sc w/ u)

& $u \rightsquigarrow w$ ($\because$ w is in T_K w/ u as the root)

$\Downarrow$

$u \leftrightsquigarrow w \Rightarrow w \in SCC(u)$

(contradiction)

$\therefore$ $SCC(u)$ Must be source node in Meta(G).

Q2: How to compute $SCC(u)$

$DFS(G^r, u)$

Q2: How to compute $SCC(u)$

$$SCC(u) = \underset{O(m+n)}{T_K} \cap \underset{O(m+n)}{DFS(G^r, u)}$$

Q3: Remove & Repeat!

Do nothing!

★ Final Al'g:

① Run DFSAll (G). Sort vertices in decreasing order of finish time. — $O(m+n)$

$v_1 \; v_2 \; v_3 \; \ldots \; v_n$

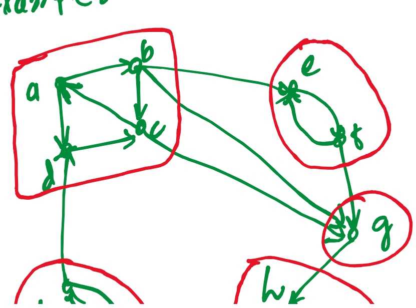
② Run DFSAll (G^r) but consider vertices in the above order. — $O(m+n)$

③ O/p all trees from step 2 as SCC of G. & construct edges of Meta (G) — $O(m)$

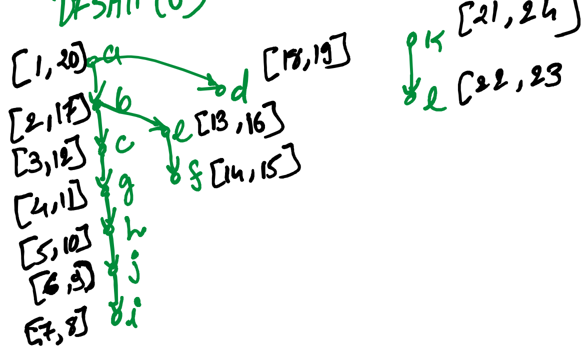
```
DFSAll (G^r) {
① Unmark all vertices
   v1 v2 ... vn
② for i = 1....n do{
③     if vi is unmarked
④         DFS (G^r, vi)
}
```

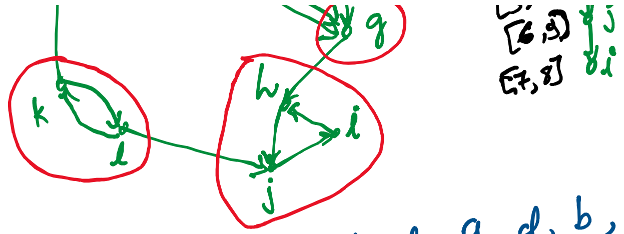
Total Run time : $O(m+n)$.

Example:



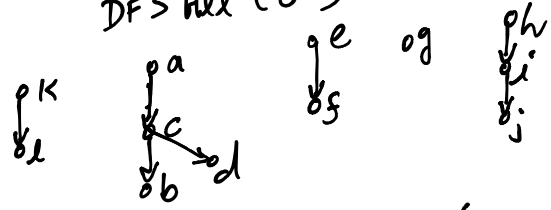
DFSAll (G)

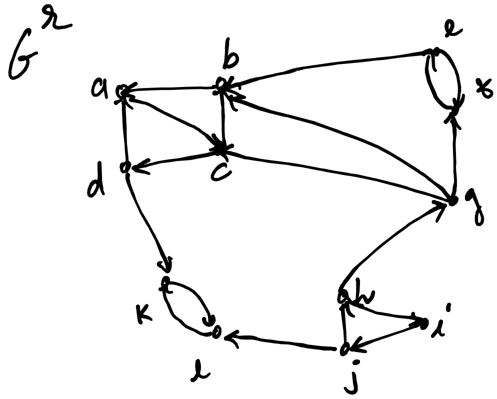




Decreasing order of F(u) : k, l, a, d, b, e, f, c, g, h, j, i

DFS All (G^r) w/ ← order





These trees = the scc of G.