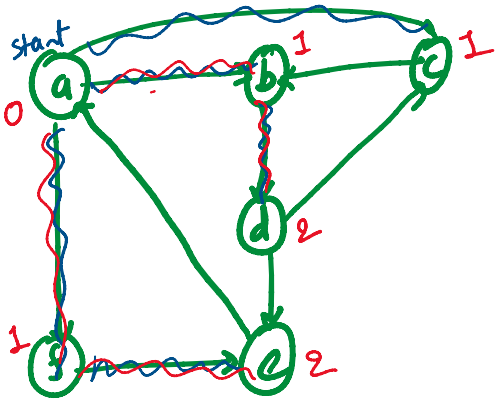
Thursday, March 21, 2024 8:44 AM

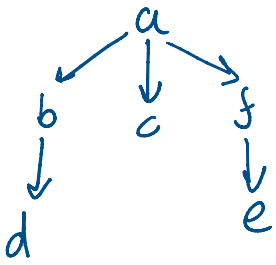
Graphs: Applications of search algorithms

Given Directed graph $G = (V, E)$

✯ BFS



BFS (G, a)



Level

0

1

2

Ex1: Find shortest path length from s to t.

shortest path distance from the root.

✯ DFS : global time = 1

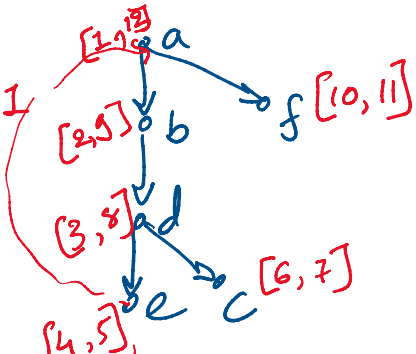
DFS (G, u) {

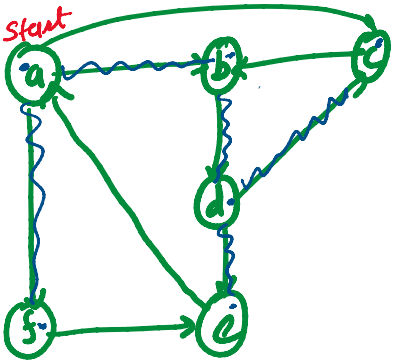
1. Mark u. D(u) = time, time += 1
2. for each v ∈ Adj(u) do
3. if v is unmarked ← parent[v] = u. DFS(G,v)
4. F(u) = time. time = time + 1

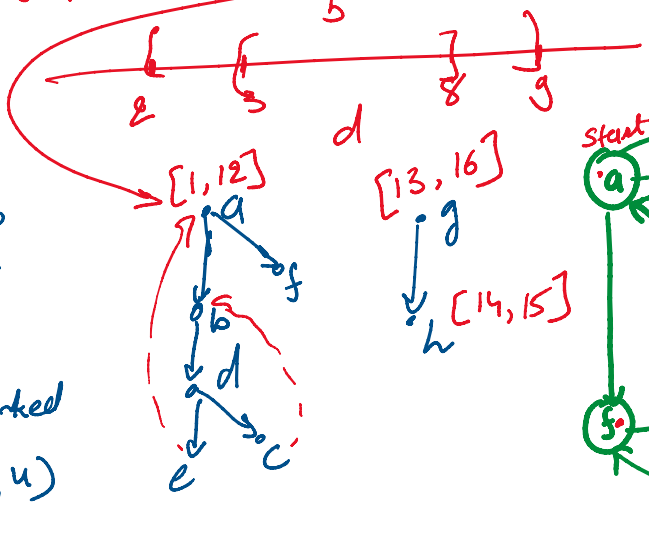
}

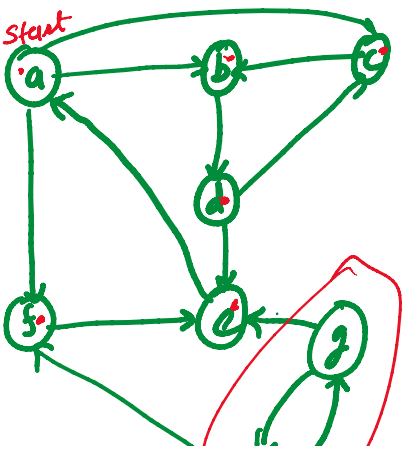
[D(u), F(u)]

DFS (G, a)







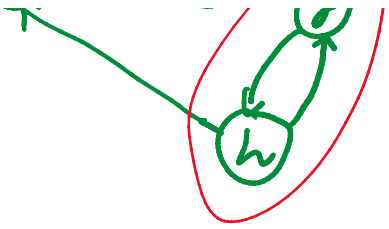


✯ DFS-All (G) {

1. for each u ∈ V do unmark u.
2. for each u ∈ V ←
3. if u unmarked
4. then DFS (G, u)

}

3.
4. Then DFS(G,u) ë c
}



Ex 2: Check if G has a cycle?

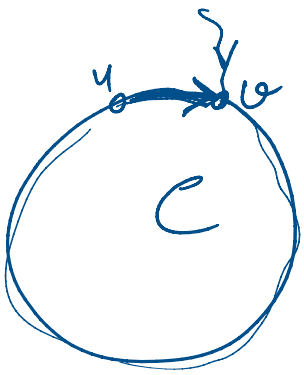
1. Run DFSAll(G)
2. if ∃ a back edge (u,v) ≡ ([D(u),F(u)] ⊂ [D(v),F(v)])
   then output YES!

Claim: back-edge ⟺ ∃ a cycle.

Pf: (⇒) back-edge ⇒ cycle.

(⇐) cycle ⇒ back-edge.

suppose DFS enters C at node v.
& let u be the vertex before v on C.
Then (u,v) is a back-edge.



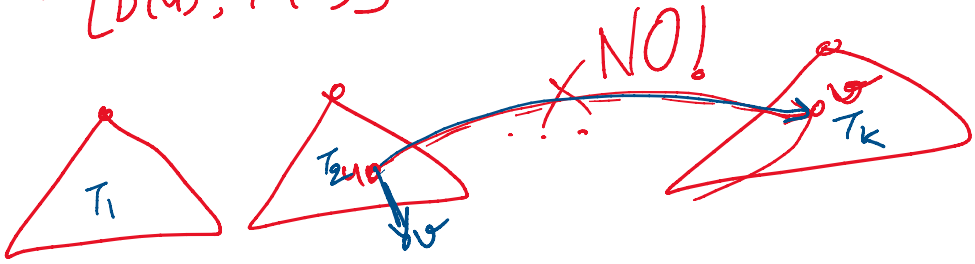
DFS Observations:

① back-edge ⟺ cycle

② v ⇝ u in DFS tree
(d[v] ≤ d[u] < f[u] ≤ f[v])

(c) $v \rightsquigarrow u$ in DFS tree
(u is a descendant of v)
Then $[D(u), F(u)] \subset [D(v), F(v)]$

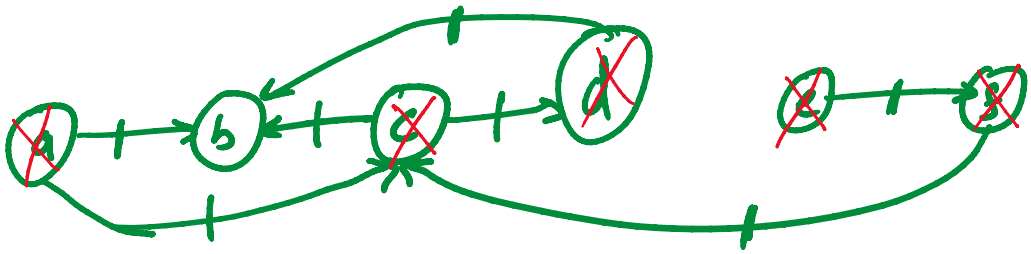
(3)



Ex 3: Topological Ordering of a DAG (acyclic graph).

Find ordering of vertices s.t. every edge goes from left to right.

eg:



a e f c d b

First idea:

1. Find a source vertex u (in-deg $(u) = 0$)
   output u
2. Remove u, repeat.
   + adjacent edges.

Q1: How to find a source u?

DFS(G)

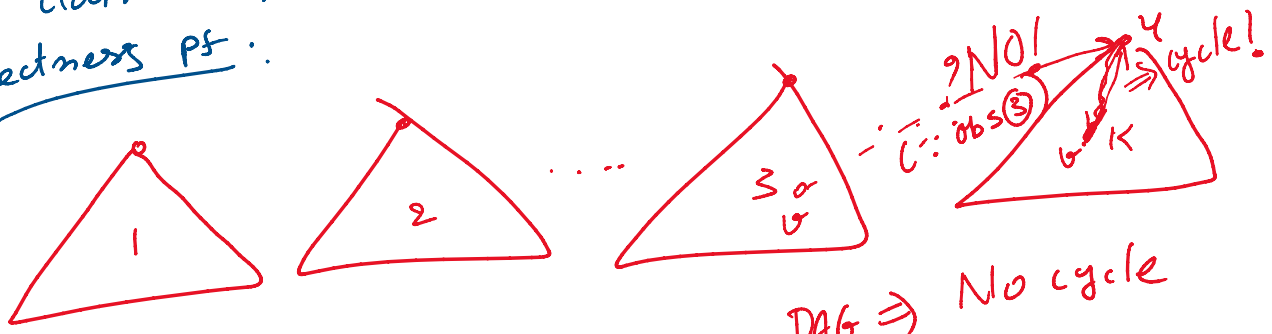
Q1: How to pick u

Run DFSALL(G).

Pick u = vertex of highest finish time (last vertex finished)

Claim: Let u be root of last tree of DFS. Then in-deg(u) = 0

Correctness Pf:



DAG ⇒ No cycle

Q2: How do we remove u & repeat?

Do nothing! just o/p nodes in decreasing order of finish time.

Final Alg'm

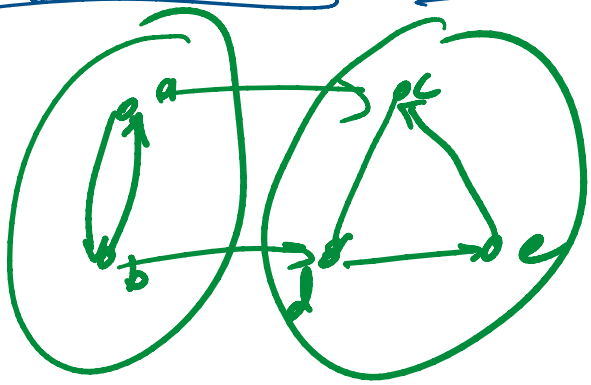
1. Run DFSALL(G)
2. output vertices in the decreasing order of finish time.

⇒ $O(m+n)$ time.

$F(u) > F(v)$

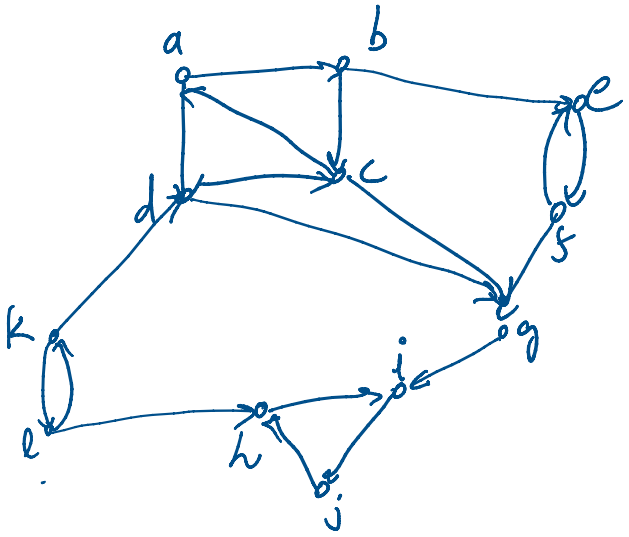
Ex 4: Strongly Connected Components: (SCC)

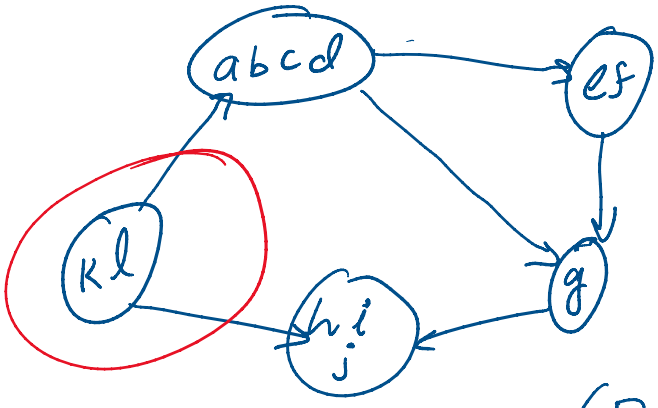
eg.



u, v are strongly connected / u, v in the same SCC $\iff$ $\exists$ $u \overset{\text{path}}{\rightsquigarrow} v$ in G & $\exists$ $v \rightsquigarrow u$ path in G.

(Equivalence Relation: symmetric, transitive...)





Meta graph (DAG)

cycles? NO!

Q: Given G, partition it's vertex set into SCC.

Naive idea:

1. For each $u, v \in V$ check if $\exists\, u \rightsquigarrow v$ path?
   For each vertex v, run DFS/BFS starting at v
2. Then form SCC

$\Downarrow$

$n(m+n) = O(mn)$.

History:

Purdom '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		

First idea:

Find u in source node of the meta graph

Find $SCC(u)$.

Remove $SCC(u)$ & Repeat.

Q1: How to find u?

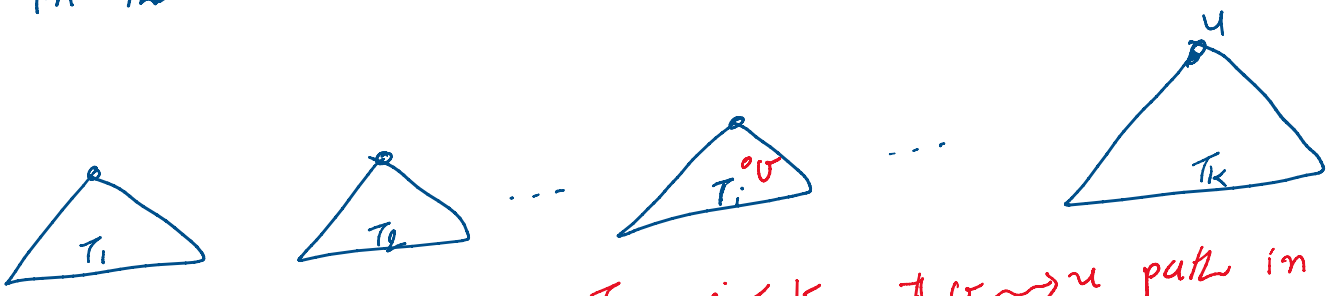
Run DFSAll (G)

Pick u = vertex w/ highest finish time
(last vertex finished)

(last vertex ...

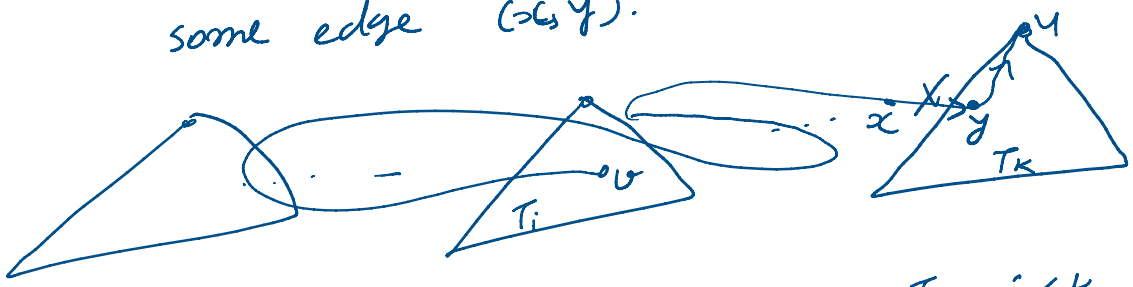
Correctness Pf Sketch:

Suppose DFSALL(G) generates trees $T_1, T_2, \ldots, T_k$ in that sequence



Claim 1: For any $v \in T_i$, $i < k$, $\nexists\ v \rightsquigarrow u$ path in G.

Pf sketch: if $\exists$ v to u path in G then this path has to enter T_k through some edge (x, y).



Then $x \notin T_k$ & $y \in T_k \Rightarrow x \in T_j$, $j < k$

This is not possible by observation ③.

Claim 2: $SCC(u) \subseteq T_k$.

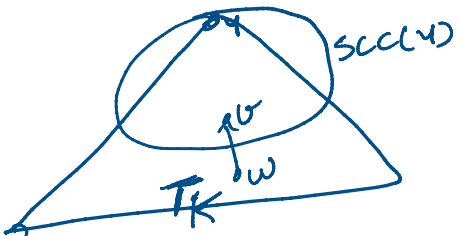
Pf: By def. $SCC(u) = \{ v \in V \mid \exists\ u \rightsquigarrow v$ path & $\exists\ v \rightarrow u$ path $\}$

Because of claim 1, if $\exists\ v \rightarrow u$ path then $v \in T_k$. Hence the proof follows.

Claim 3: $SCC(u)$ is a source node in the meta-graph. (MG)

Pf sketch:

By claim 2, $SCC(u) \subseteq T_k$



Since $\forall v \in T_k$, $\exists\ u \rightsquigarrow v$ path

$SCC(u) = \{ v \in T_k \mid \exists\ v \rightsquigarrow u \text{ path} \}$

Now, suppose $SCC(u)$ not a source in MG.

$\Rightarrow \exists (w, v) \in E$ s.t. $v \in SCC(u)$, $w \notin SCC(u)$

$\Rightarrow w \in T_k$ because of Obs ③. $\Rightarrow \exists\ u \rightsquigarrow w$ path.

But then $w \to v \rightsquigarrow u$ is a path from w to u

$\Rightarrow w$ & u are strongly connected $\Rightarrow w \in SCC(u)$!

A contradiction.

Note: The above discussion implies that we can find $SCC(u)$ as follows:

Run DFS starting at u but using "in-coming" edges. $SCC(u)$ = all the nodes this DFS reaches/marks.

(In other words let T be the tree generated by DFS (G^R, u) where $G^R = (V, E^R)$,

$E^R = \{(v, u) \mid (u, v) \in E\}$.

$SCC(u) = \{ v \mid v \in T \}$.)

Q2: How to find $SCC(u)$?

... starting at u in

Q2: How to find [illegible]

Run DFS starting at u in graph G^R where edges are reversed.

$G^R = (V, E^R)$

$E^R = \{(v, u) \mid (u, v) \in E\}$

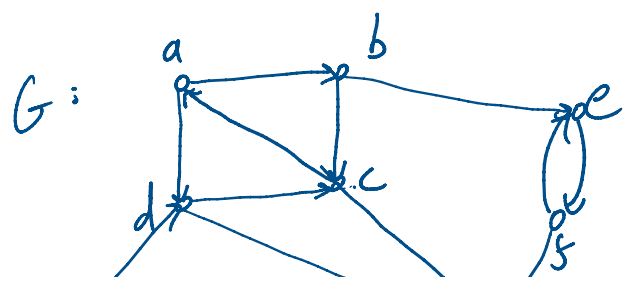
Q3: How to record & repeat?

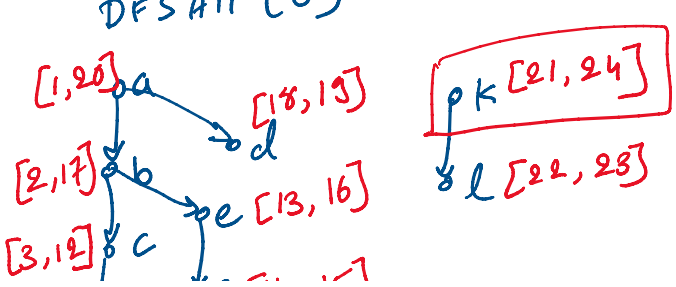
Do nothing!

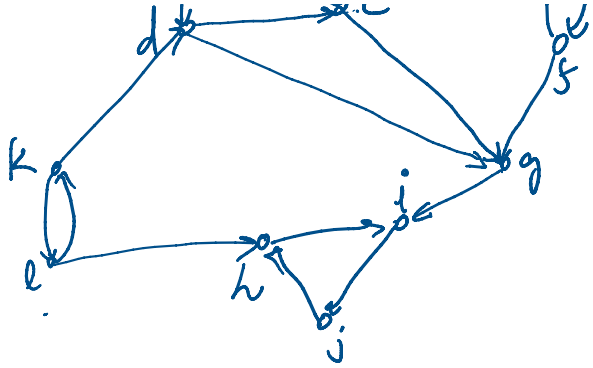
Final Alg'm

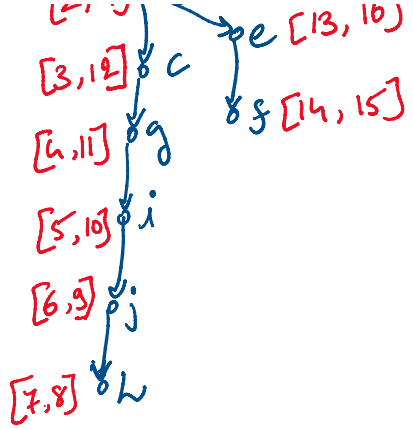
1. DFS All (G). label vertices in decreasing order of finishtime.
2. DFSAll (G^R), prefer vertices of higher label while picking root.
3. O/p DFS trees from step 2 as SCC.

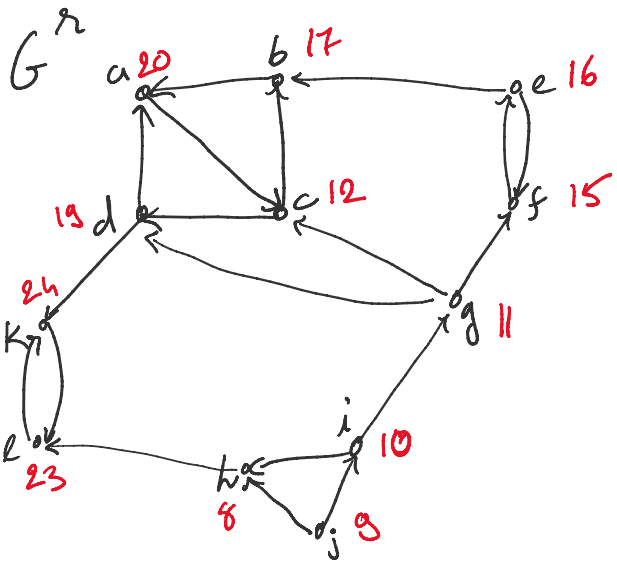
$\Rightarrow O(m+n)$ time.



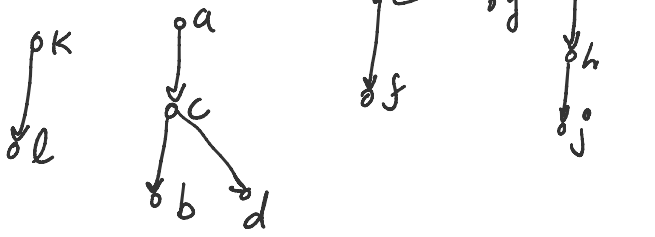








DFSALL (G^R) preferring higher label



SCCs.