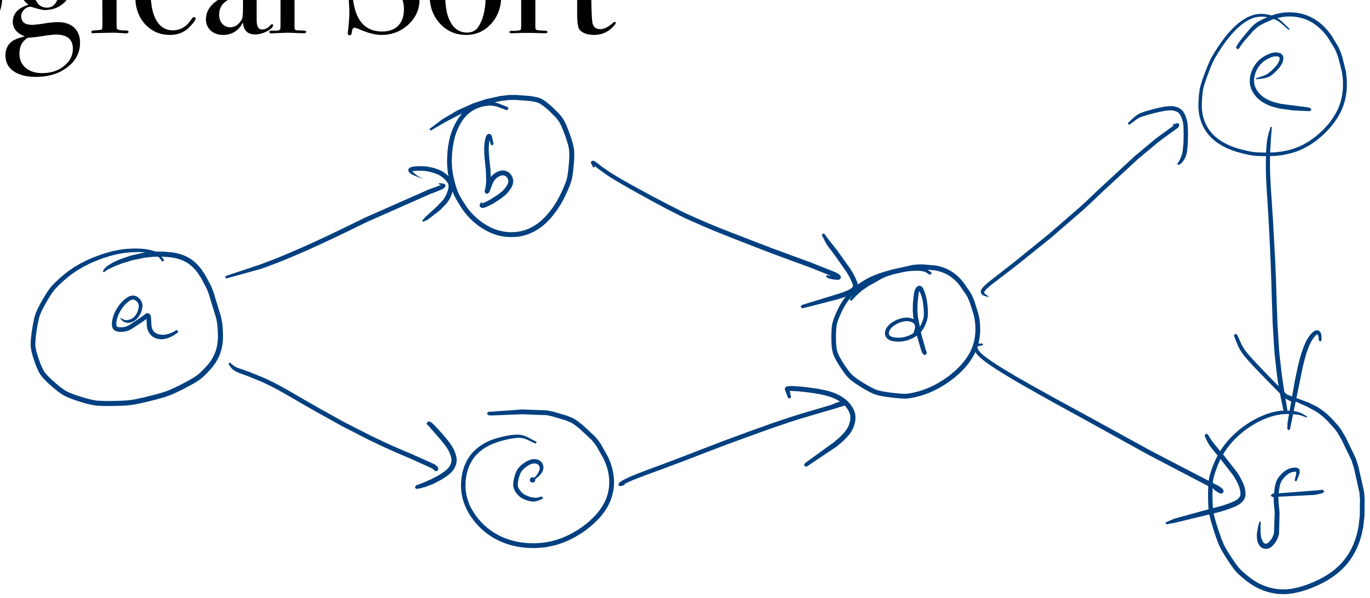
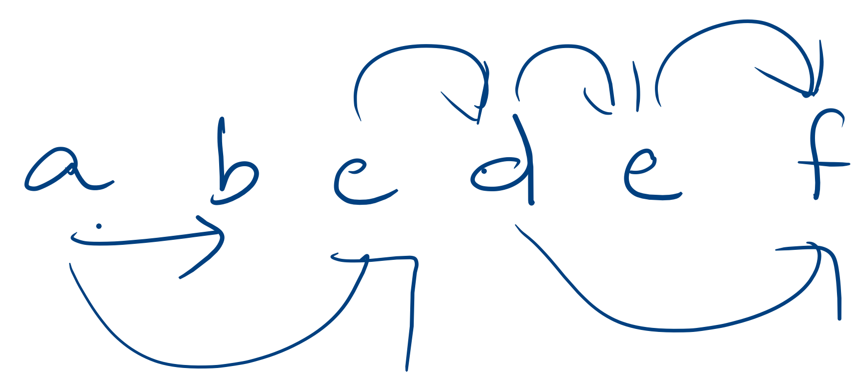
Applications of BFS and DFS

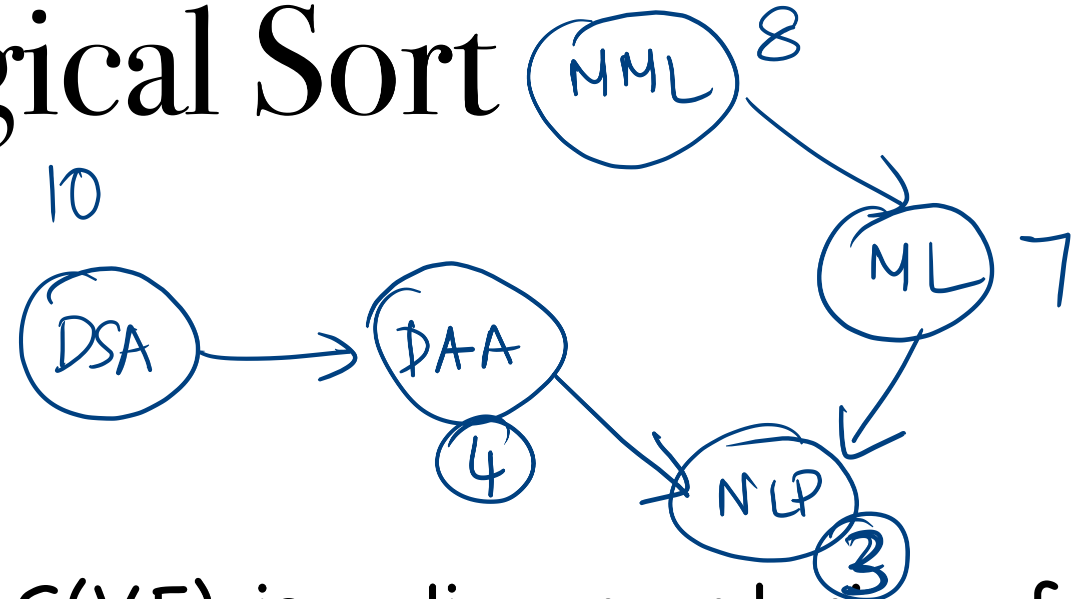
Topological Sort



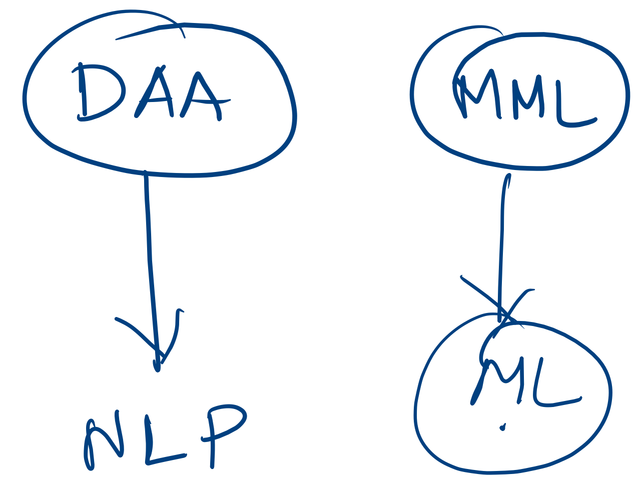
A topological sort of a DAG G(V,E) is a linear ordering of vertices of G such that if G contains an edge (u,v) then u appears before v in the ordering.



Topological Sort



A topological sort of a DAG G(V,E) is a linear ordering of vertices of G such that if G contains an edge (u,v) then u appears before v in the ordering.



- used to show precedence among events

DSA – MML – ML – DAA – NLP

Topological Sort

$u \rightarrow v$

$\Longleftrightarrow f[u] > f[v]$

- call DFS(G) to compute f[v] for each v
- As each vertex is finished, insert it to the front of a linked list
- return the linked list

Topological Sort

A directed graph G is acyclic if an only if a DFS of G yields no back edges

$\Leftarrow$ Let (u,v) be a back edge. Path from ~~u~~ to ~~v~~ in DFS tree with (u,v) forms a cycle.

$\Rightarrow \exists\, C\ (v_1 - v_2 - \cdots v_k - v_1)$. Let v_1 be the first vertex in C to be discovered. At $d[v_1]$, $v_2 \dots v_k$ are all white

$\exists\ v_1 \rightsquigarrow v_k$ white path $\Rightarrow v_k$ is a descendant of v_1.

$\Rightarrow (v_k, v_1)$ is a back edge.

Topological Sort

Algorithm is correct.

If $(u,v) \in E$, then $f(u) > f(v)$.

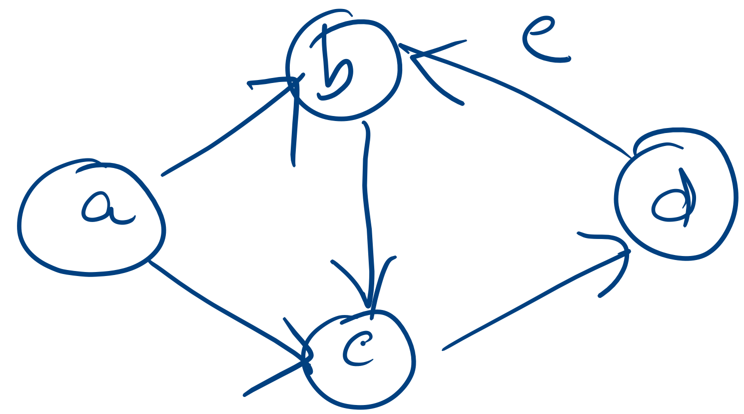
when the edge (u,v) is explored,

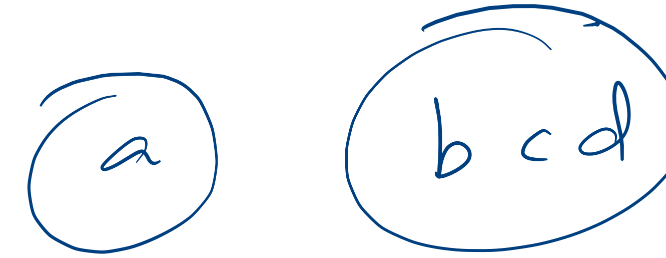
Case 1: v is white $\Rightarrow$ v is a descendant of $u \Rightarrow f(u) > f(v)$

Case 2: " gray $\Rightarrow (u,v)$ is a back edge $\rightarrow$ not possible

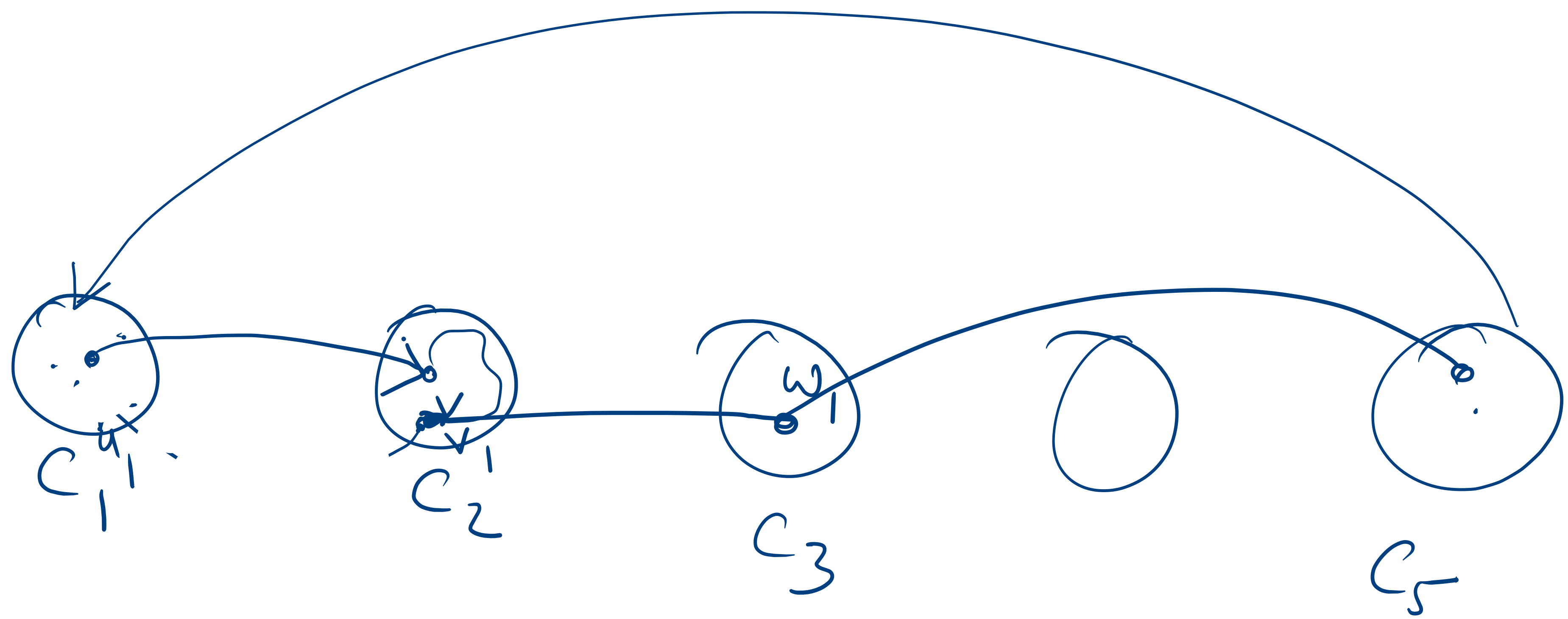
3: " black $\Rightarrow f(v) < f(u)$

Strongly Connected Components

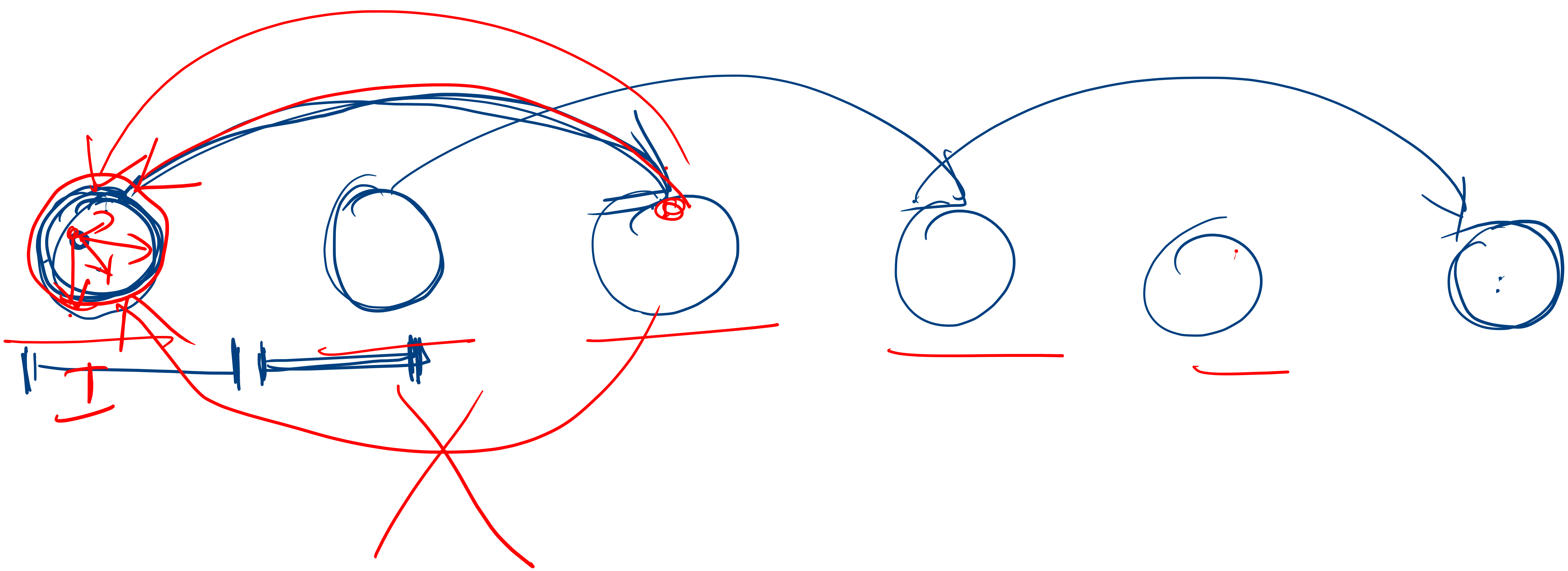




- Strongly connected component of a directed graph $G(V, E)$ is a maximal set of vertices $C \subseteq V$ such that for every pair of vertices u and v in C, u and v are reachable from each other.



u1
C1
v1
C2
w1
C3
C5

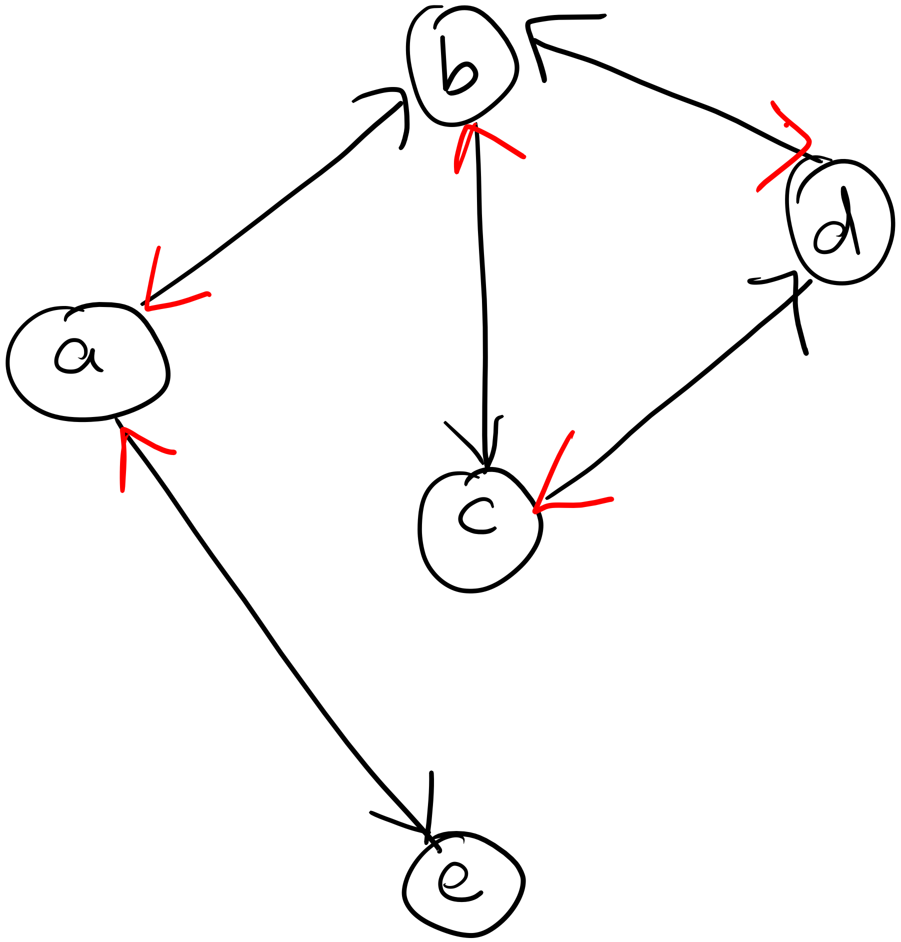


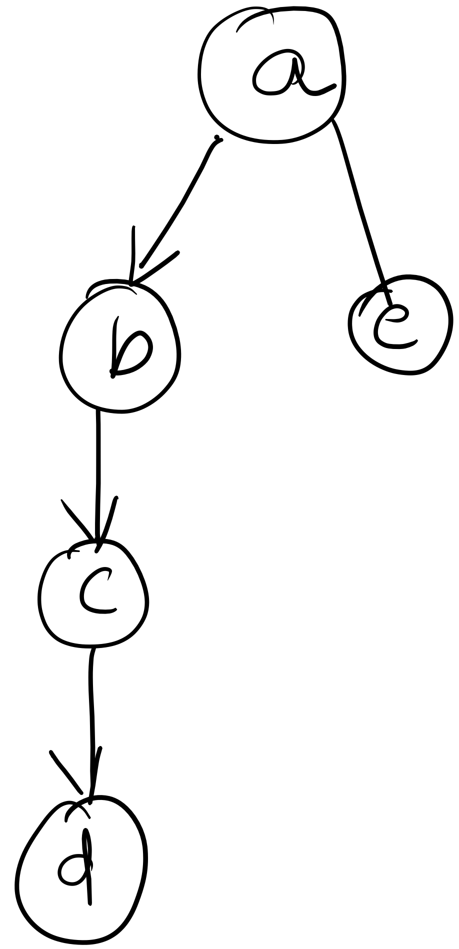
Call DFS(G) to compute $f[u]$ for each vertex u

Compute $G^T = (V, E^T)$ $E^T = \{(u,v) \mid (v,u) \in E\}$

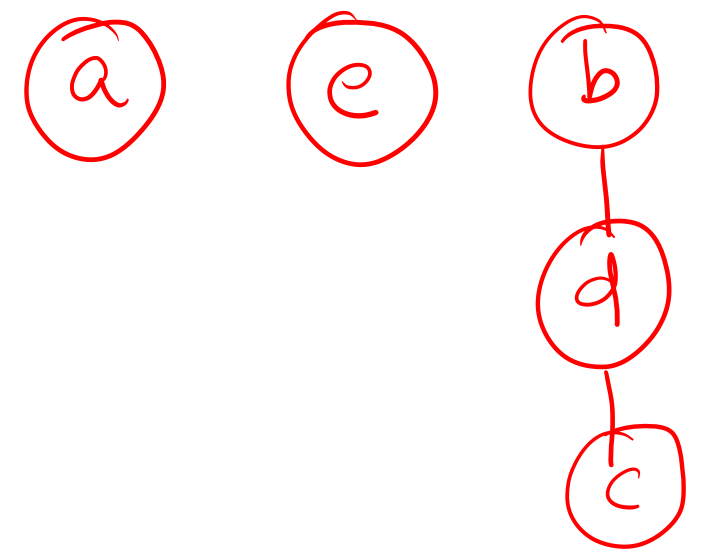
call DFS(G^T), consider the vertices in order of decreasing $f[u]$

Output the vertices in each tree in the DFS forest (formed in the previous step) as a separate strongly connected component





d.
c
b
e
a



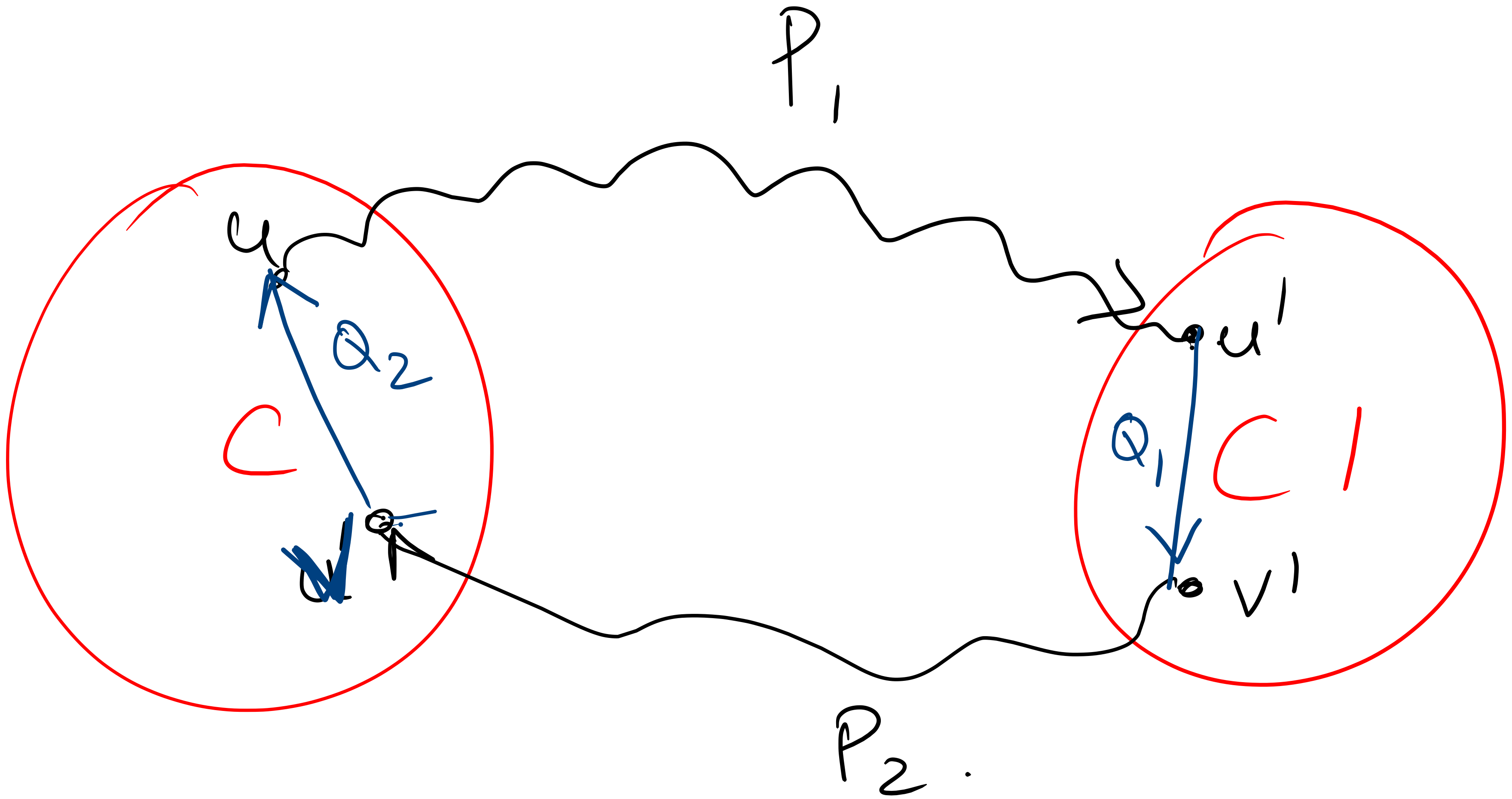
$\{a\}$

$\{e\}$

$\{b, d, c\}$

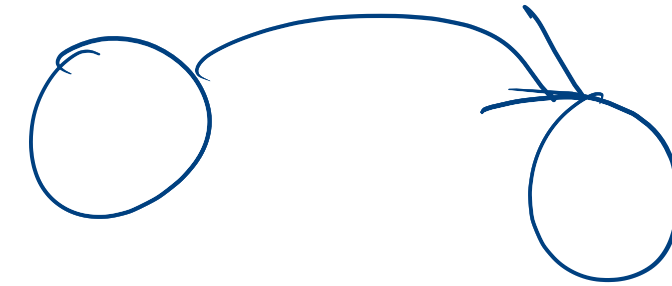
Let C, C' be distinct strongly connected components in directed graph $G(V, E)$.

Let $u, v \in C, u', v' \in C'$. Suppose there is a path from u to u'. Then there cannot also be a path from ~~v to v'.~~ v' to v.



$Q_1 P_2 Q_2$ is a path from u' to u

P_1 " u to u'

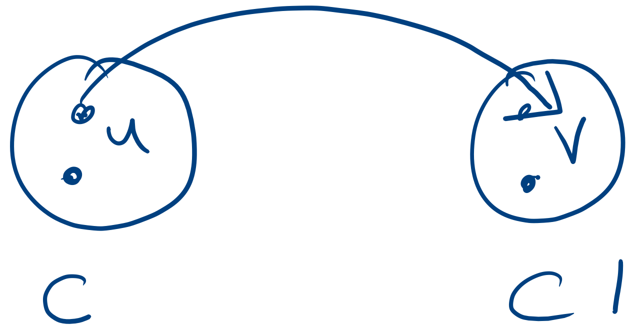


Let C, C' be distinct strongly connected components in directed graph $G(V, E)$.

- Suppose $(u, v) \in E$ such that $u \in C$ and $v \in C'$. Then $f(C) > f(C')$.

$$f(C) = \max_{v \in C} f(v)$$

$$d(C) = \min_{v \in C} d(v)$$



$f(C) > f(C')$

Case 1: $d(C) < d(C')$

$d(C) = d[x]$

$\Rightarrow$ Every vertex in C & C' are white at $d[x]$.

- $\exists$ white paths from x to all vertices in C & C'

$\Rightarrow$ They are descendants of x.

$f(C) = f(x) > f(C')$

$\Rightarrow f(C) > f(C')$

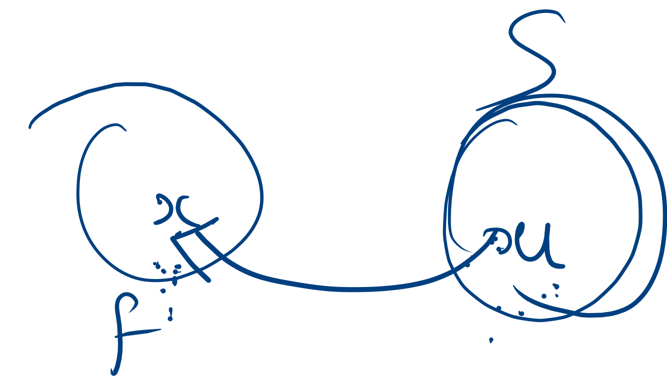
Case 2: $d(C) > d(C')$

$d(C') = d[y]$

$\exists$ white paths from y to C'

$\cancel{\exists}$ " "

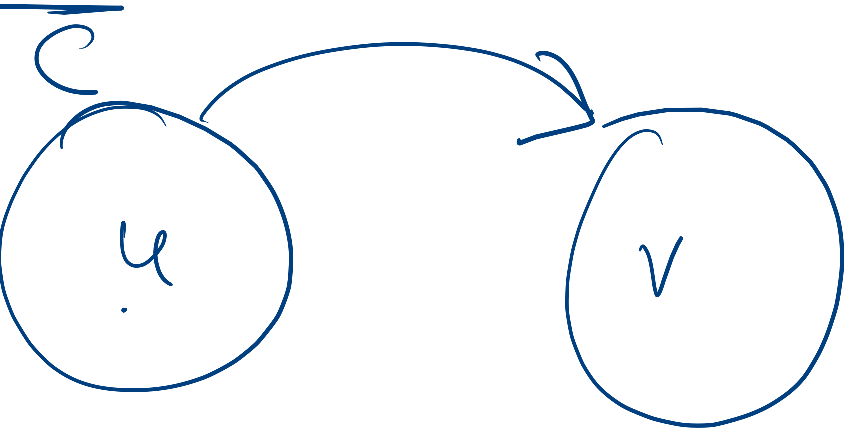
$f[C'] < d(C) < f(C)$



Let C, C' be distinct strongly connected components in directed graph $G(V, E)$.

- Suppose $(u, v) \in E$ such that $u \in C$ and $v \in C'$. Then $f(C) > f(C')$.

- Suppose there exists $(u, v) \in E^T, u \in C, v \in C'$. Then $f(C) < f(C')$.



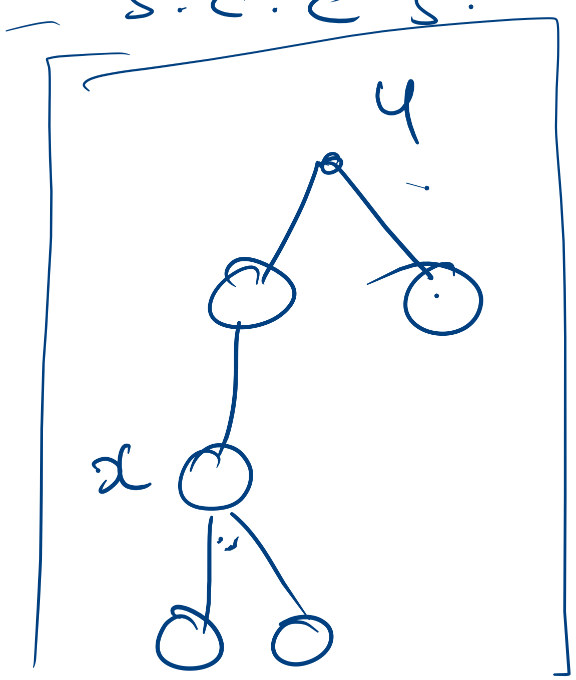
Algo. is correct

Proof:

By indn of DFS trees, k.

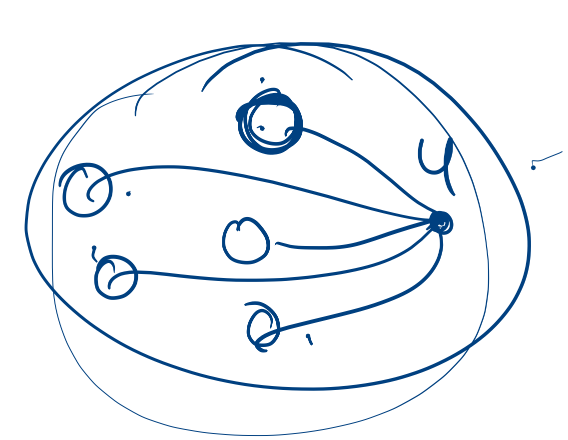
$k = 0$ Base case, trivially true.

k trees are returned.
Each of them correspond to s.c.c's.



Let u be the root of $(k+1)^{th}$ DFS tree.

Let S be the s.c.c. that contain u.



G^T.

$V(S) = V(T)$

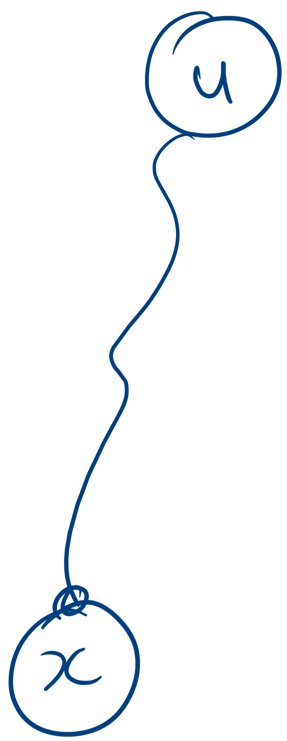
At $d'[u]$, all vertices in S are white.
By white path theorem, they are all in T.

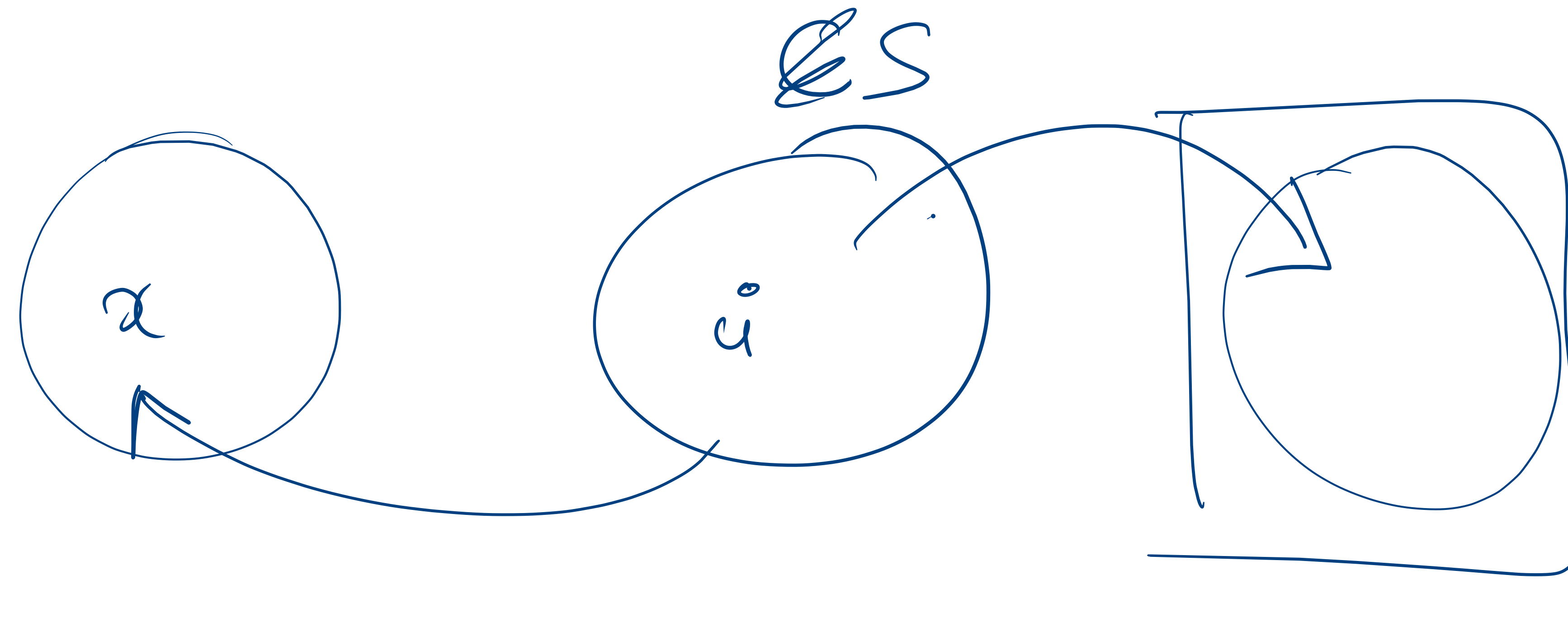
$\Rightarrow V(S) \subseteq V(T)$

To prove $V(T) \subseteq V(S)$.
Assume not.
$\exists\, x \in T$, but $x \notin S$.
x cannot belong to a s.c.c. that is yet to be seen.

x ~~is~~ $\in$ s.cc. that is already seen.

Contradicts Induction hypothesis.


u
x


x
u
∉ S