Strongly Connected Components

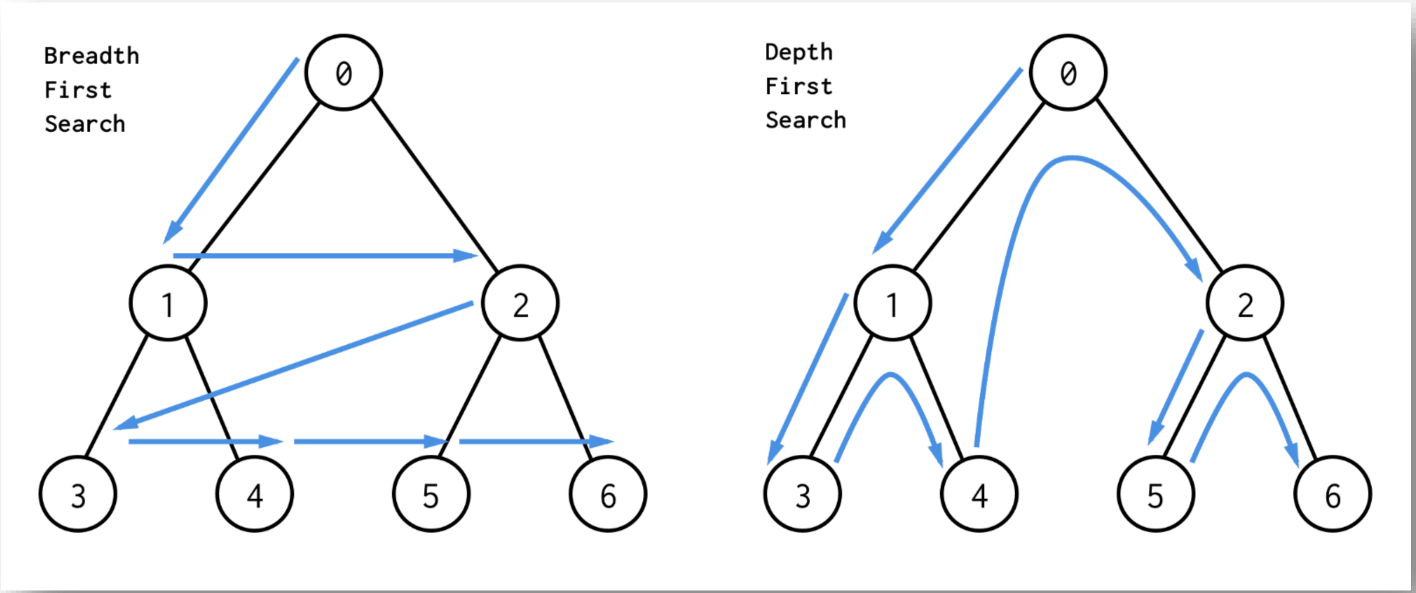
Kuan-Yu Chen (陳冠宇)

2019/05/22 @ TR-310-1, NTUST

Review

- Breadth-first search
  - BFS uses a **queue** as an auxiliary data structure to store nodes for further processing

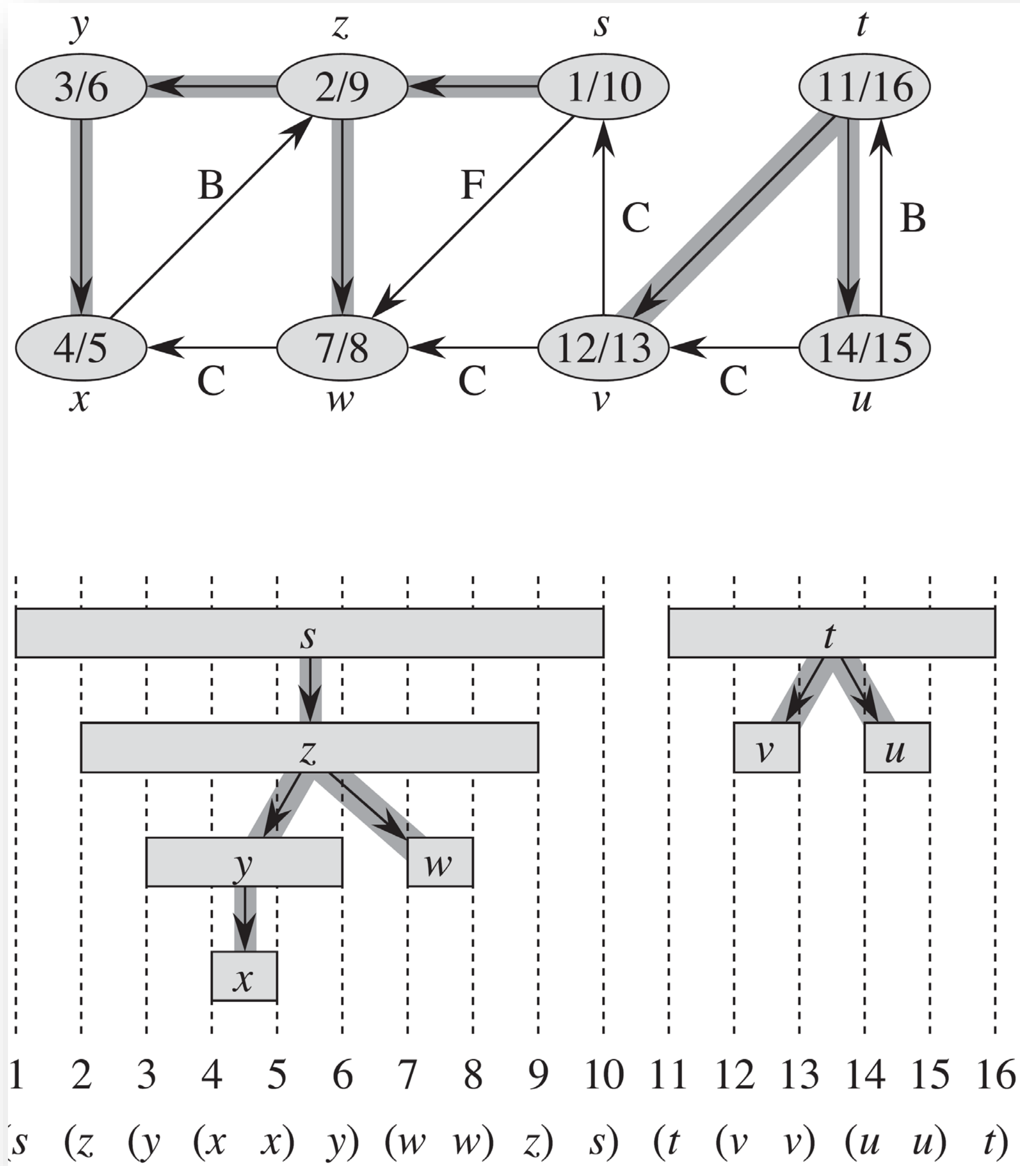
- Depth-first search
  - DFS uses a **stack** to store nodes for further processing



https://www.quora.com/What-are-the-differences-between-DFS-and-BFS

DFS and Parenthesis Structure.

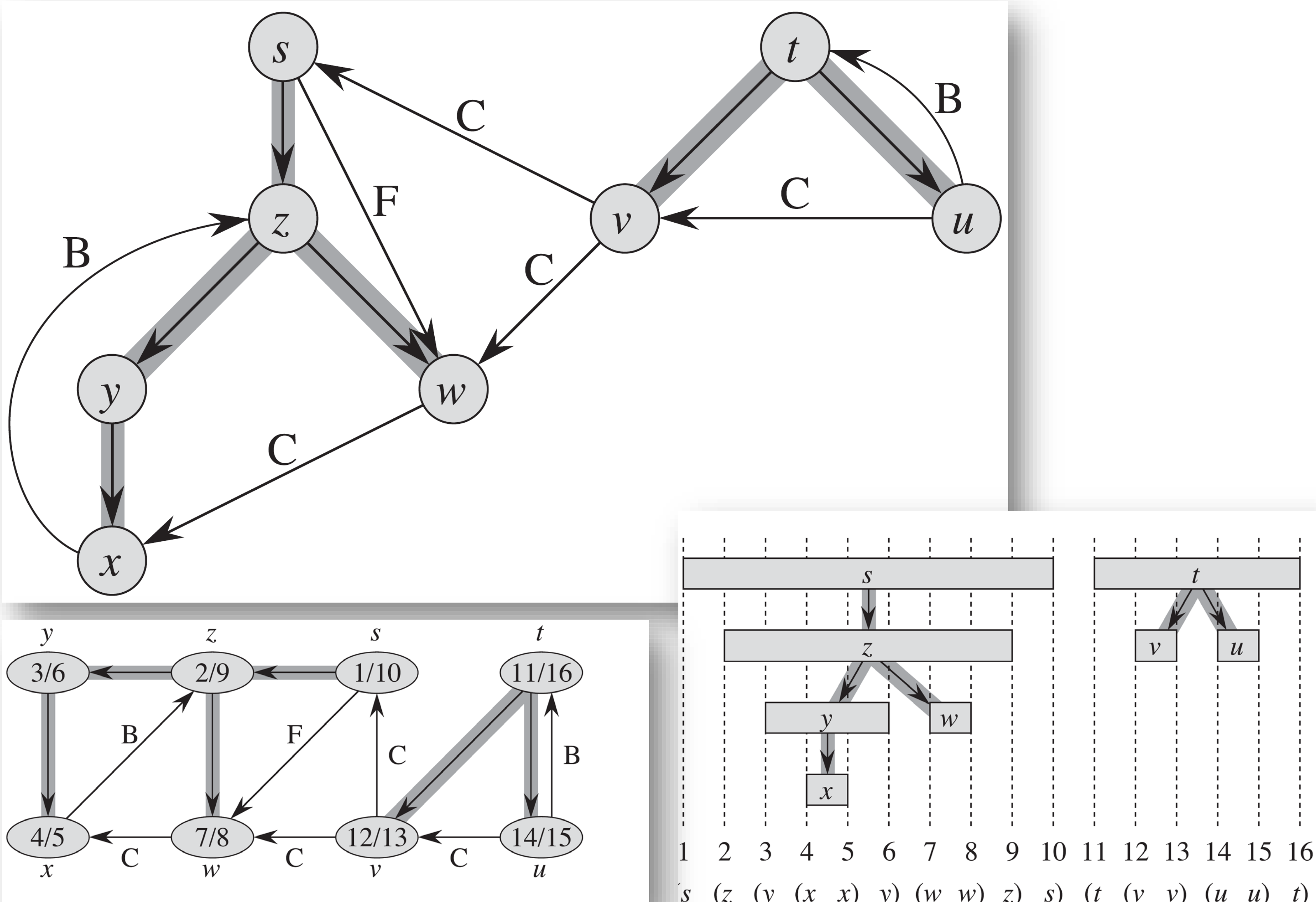
- An important property of depth-first search is that discovery and finishing times have ***parenthesis structure***



DFS and Parenthesis Structure..

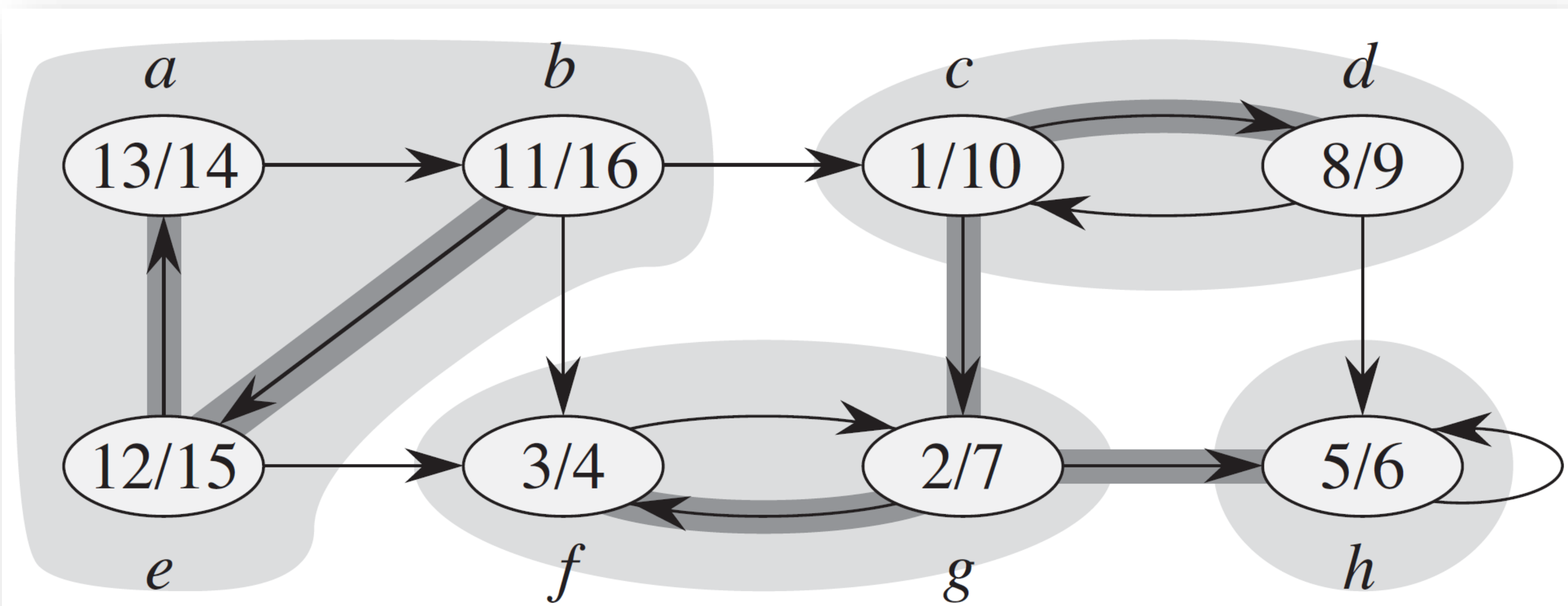
- Parenthesis theorem
  - In any depth-first search of a (directed or undirected) graph $G = (V, E)$, for any two vertices u and v, exactly one of the following three conditions holds
    - The intervals $[u.d, u.f]$ and $[v.d, v.f]$ are entirely disjoint, and neither u nor v is a descendant of the other in the depth-first forest
    - The interval $[u.d, u.f]$ is contained entirely within the interval $[v.d, v.f]$, and u is a descendant of v in a depth-first tree
    - The interval $[v.d, v.f]$ is contained entirely within the interval $[u.d, u.f]$, and v is a descendant of u in a depth-first tree

DFS and Parenthesis Structure…



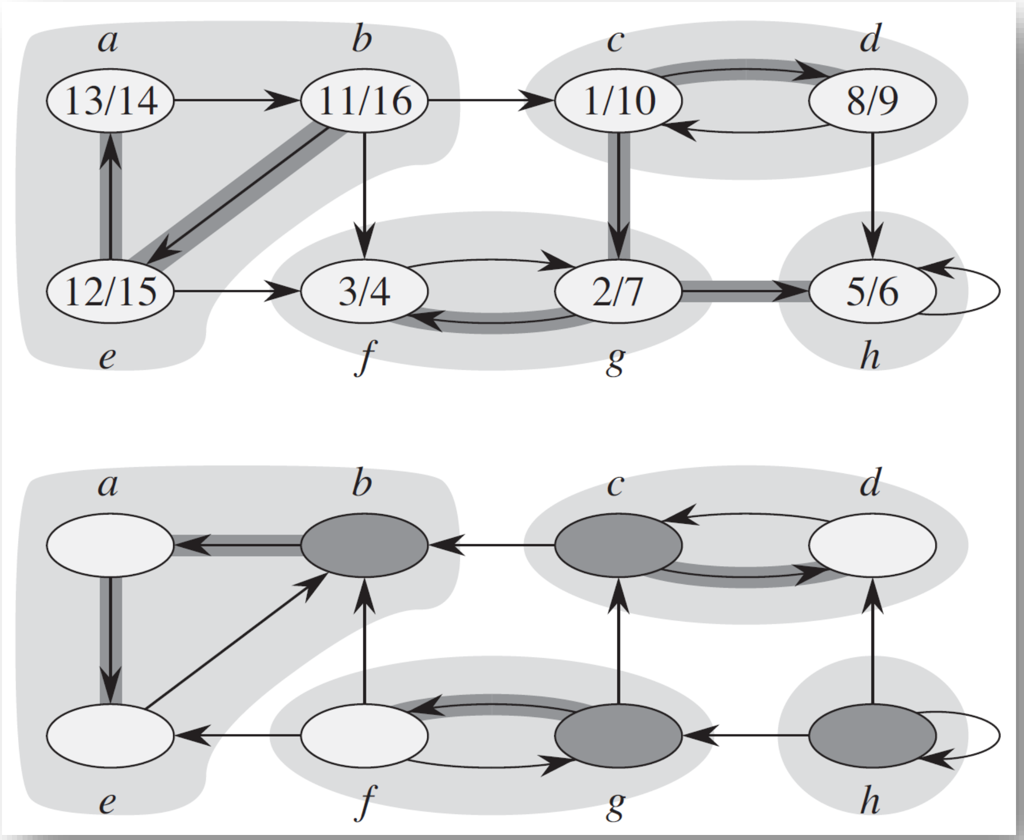
Strongly Connected Components.

- A 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, we have both $u \rightsquigarrow v$ and $v \rightsquigarrow u$
  - Vertices u and v are reachable from each other



Strongly Connected Components..

- If we create a graph $G^T = (V, E^T)$, which is transpose of G
  - It is interesting to observe that G and G^T have exactly the same strongly connected components
    - E^T consists of the edges of G with their directions reversed
    - u and v are reachable from each other in G if and only if they are reachable from each other in G^T



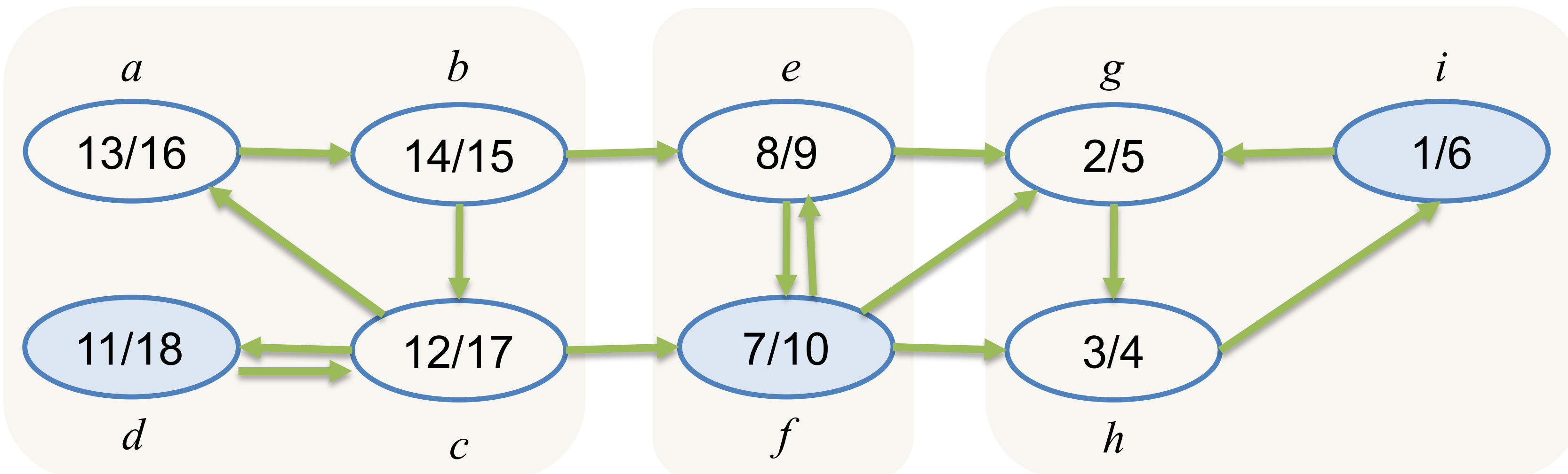
Strongly Connected Components…

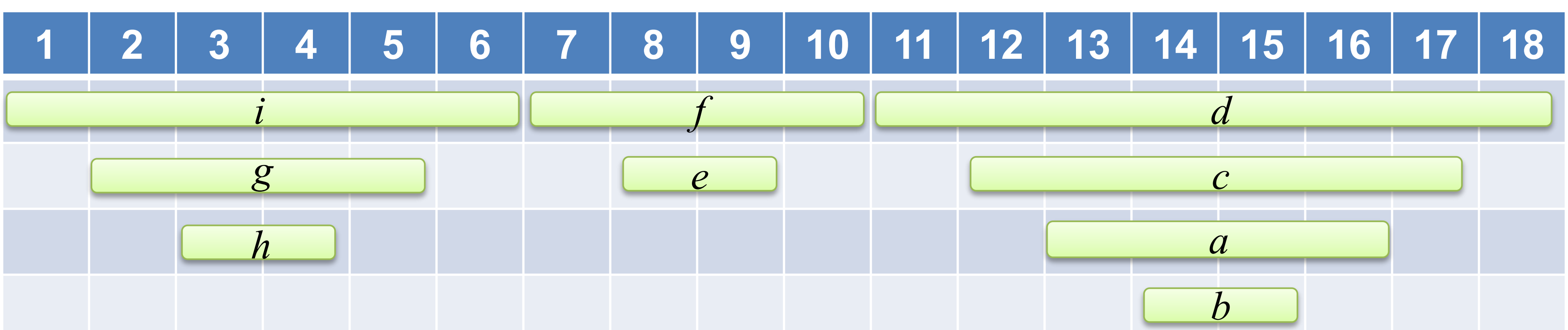
- Lemma:
  - Let C and C' be distinct strongly connected components in directed graph $G = (V, E)$, let $u, v \in C$ and $u', v' \in C'$, and suppose that G contains a path $u \leadsto u'$. Then G cannot also contain a path $v' \leadsto v$

- Prove:
  - If G contains a path $v' \leadsto v$, then it contains paths $v' \leadsto v \leadsto u$ and $u \leadsto u' \leadsto v'$
  - Thus, u and v' are reachable from each other, thereby contradicting the assumption that C and C' are distinct strongly connected components

SCC by DFS.

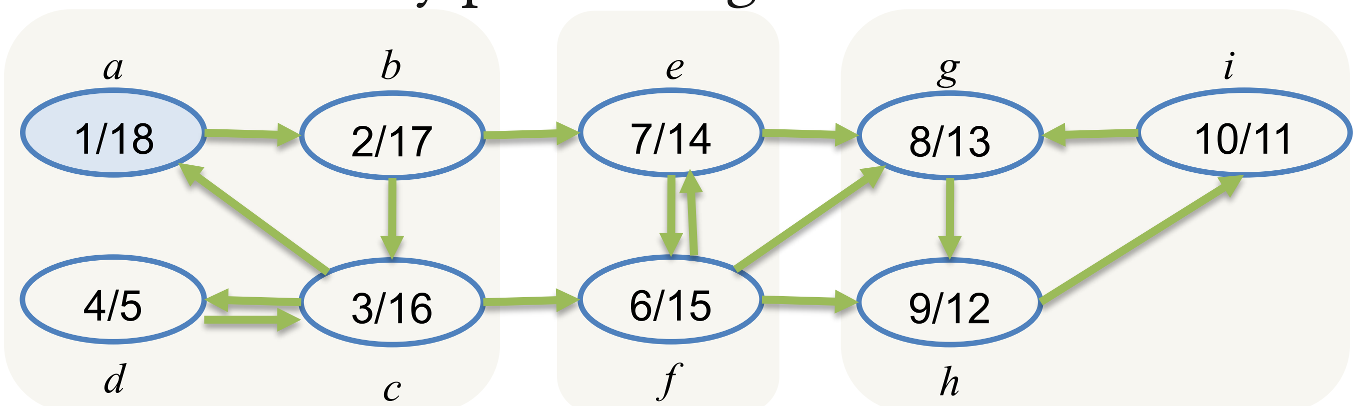
- SCC can be found by performing a DFS?

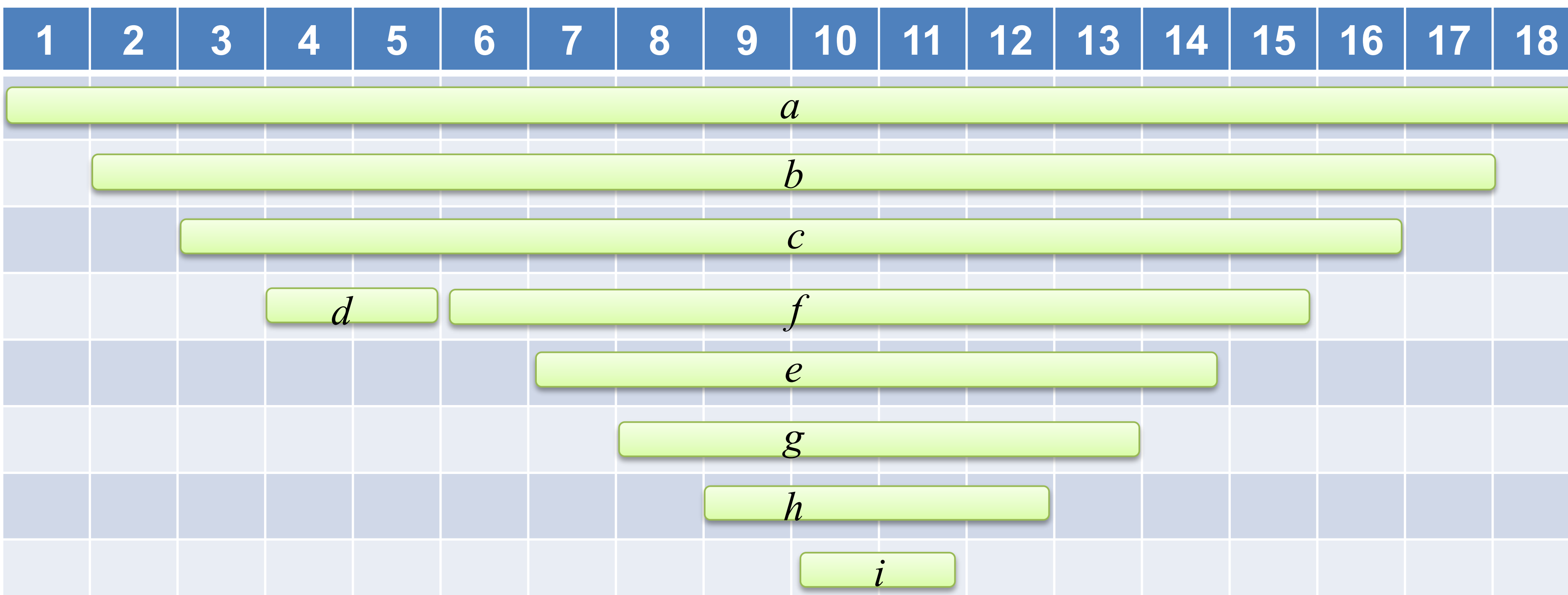




SCC by DFS..

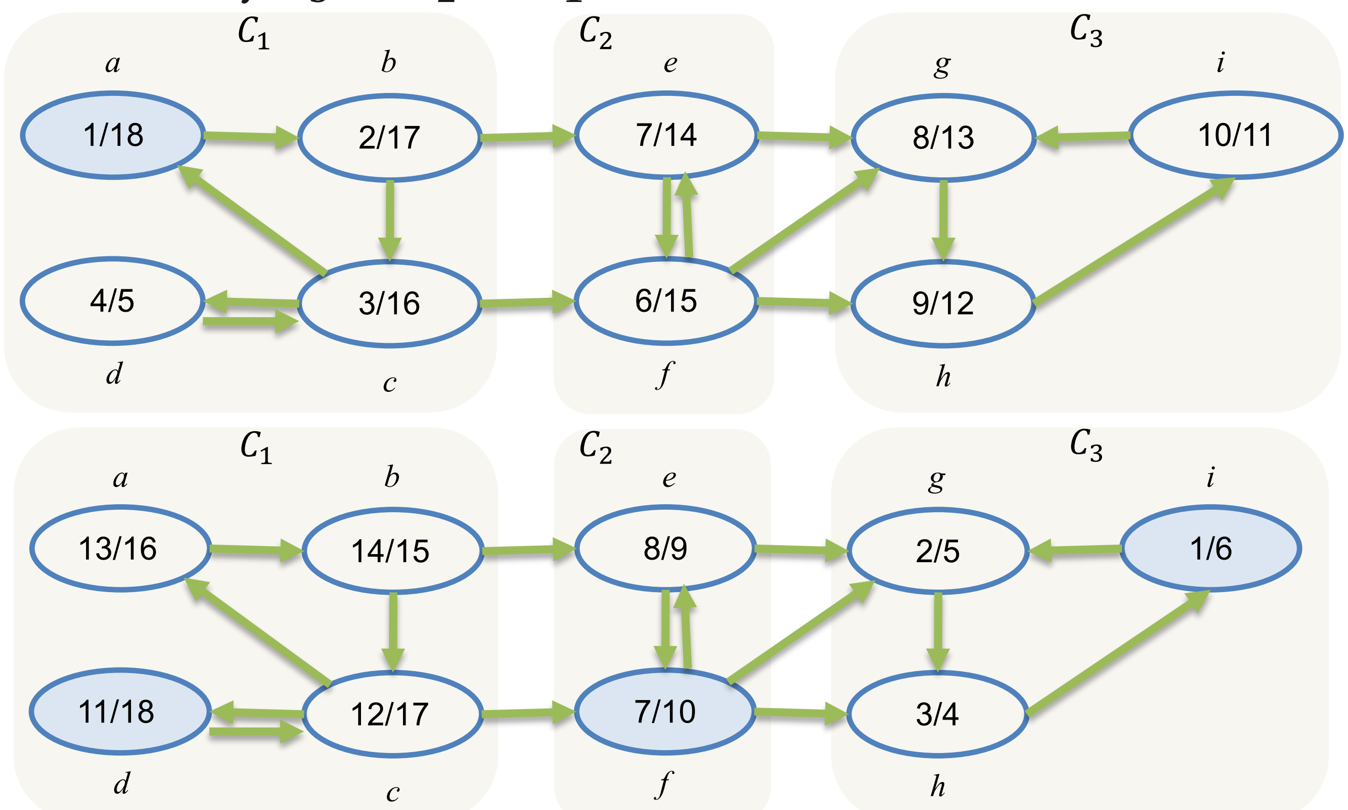
- SCC can be found by performing a DFS?





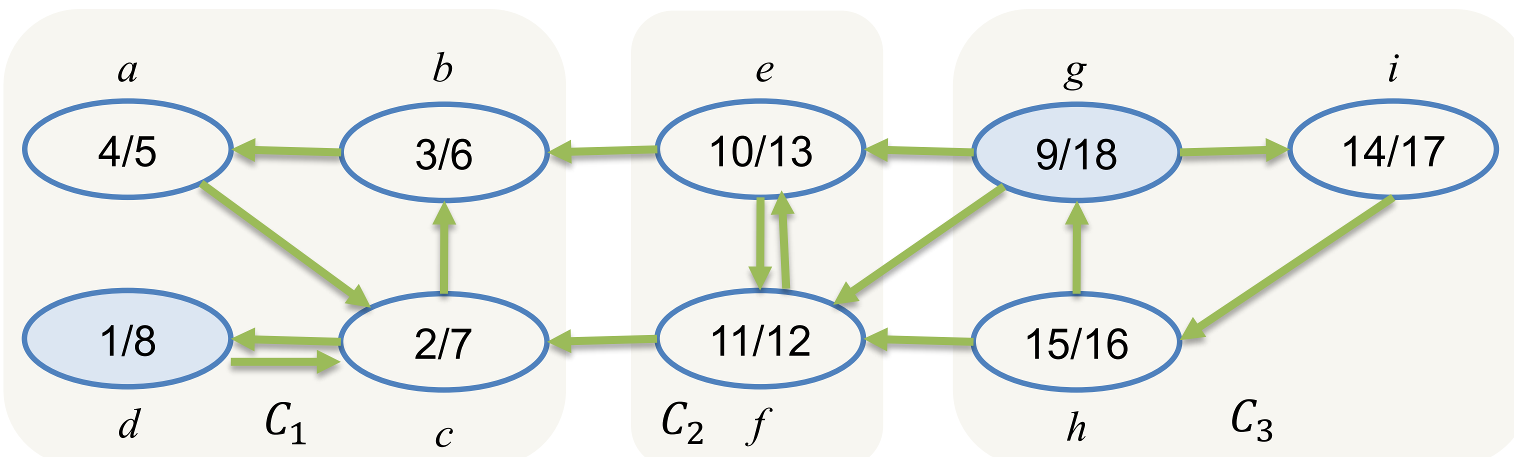
SCC by DFS...

- Observation1
  - The maximum finish time of C_1, C_2 and C_3 is always $C_1 > C_2 > C_3$
    - If we want to construct SCC by using DFS, we should choose vertex by $C_3 \rightarrow C_2 \rightarrow C_1$



SCC by DFS...

- Observation2
  - If the maximum finish time of a DAG G, which contains three components C_1, C_2 and C_3, is $C_1 > C_2 > C_3$
  - The maximum finish time of G^T is always $C_1 < C_2 < C_3$



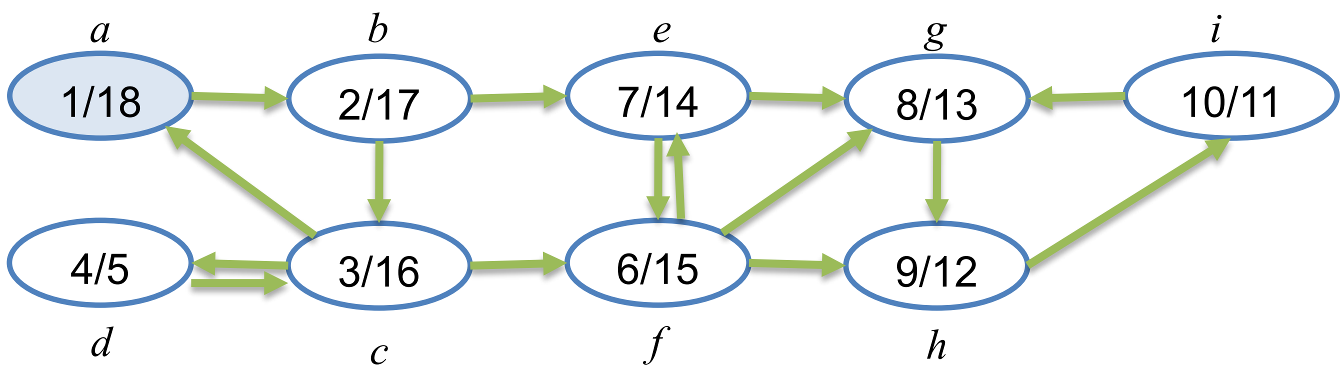
SCC by DFS….

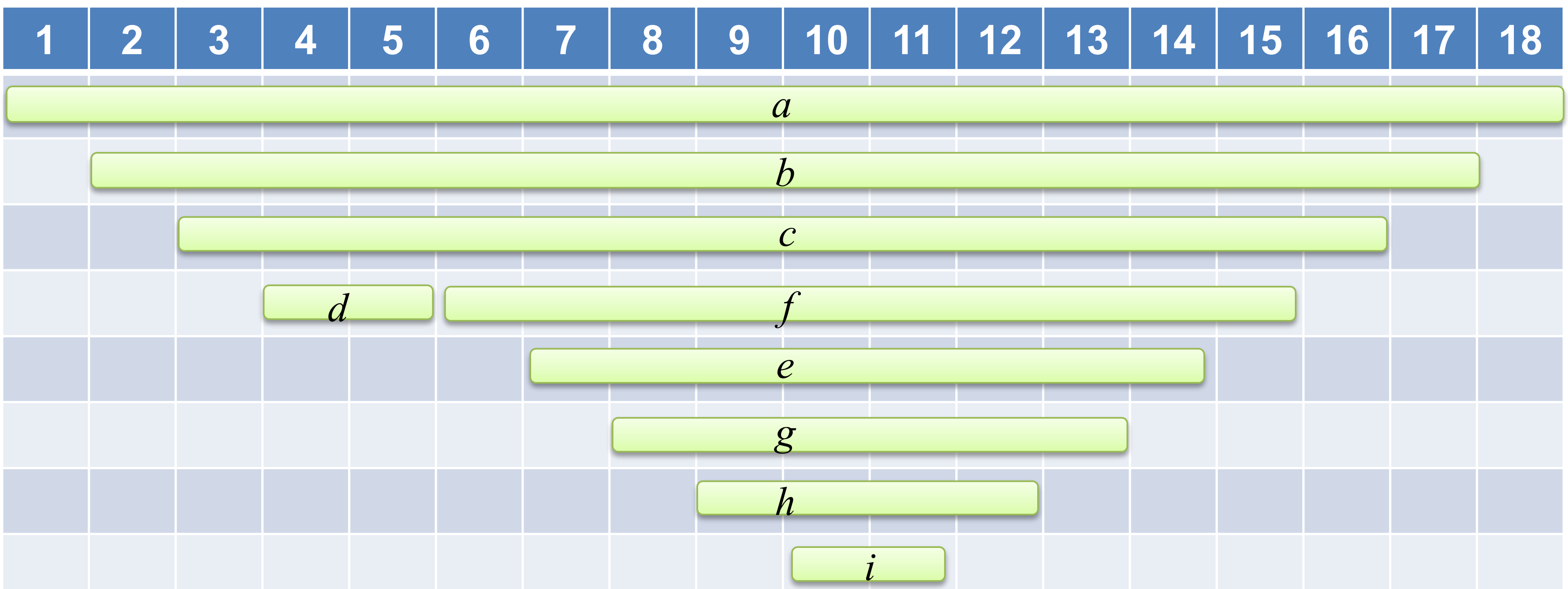
- By considering observations 1 and 2

```
STRONGLY-CONNECTED-COMPONENTS(G)
1  call DFS(G) to compute finishing times u.f for each vertex u
2  compute G^T
3  call DFS(G^T), but in the main loop of DFS, consider the vertices
       in order of decreasing u.f (as computed in line 1)
4  output the vertices of each tree in the depth-first forest formed in line 3 as a
       separate strongly connected component
```

Example.

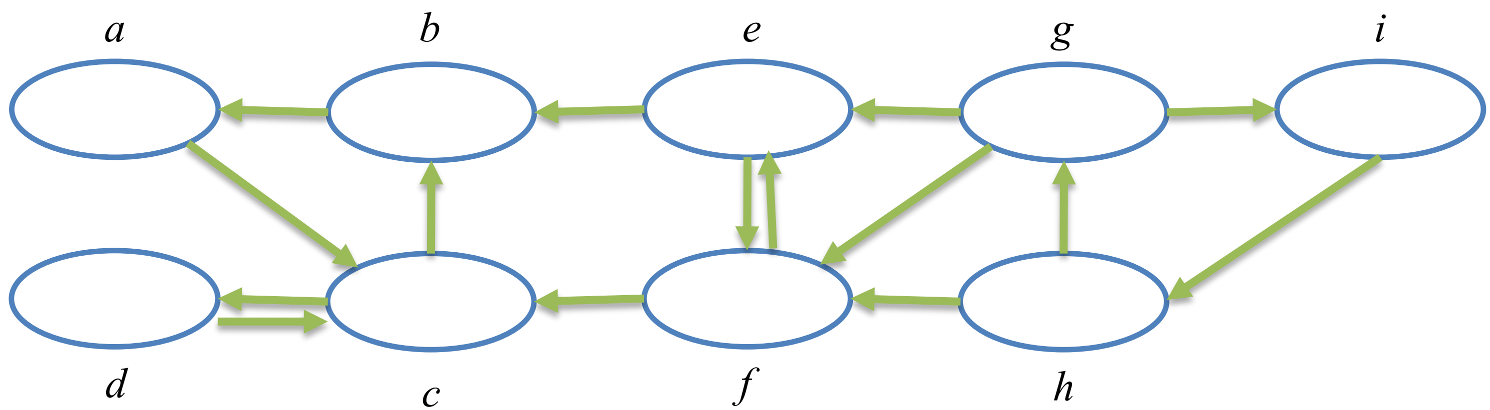
- Step1: $DFS(G)$





Example..

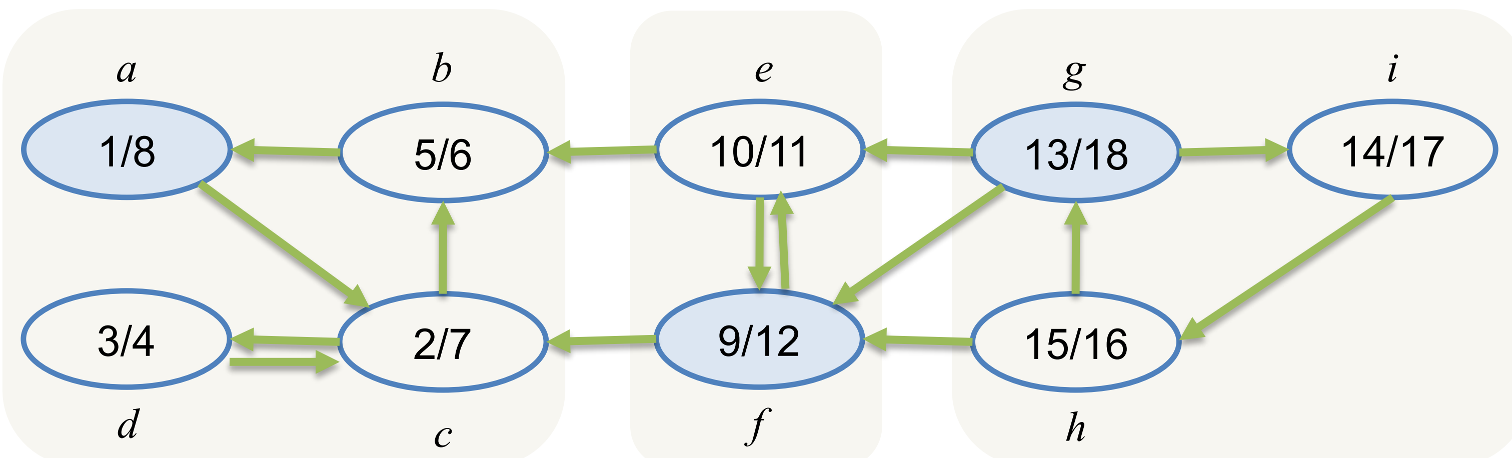
- Step2: Construct G^T



Example...

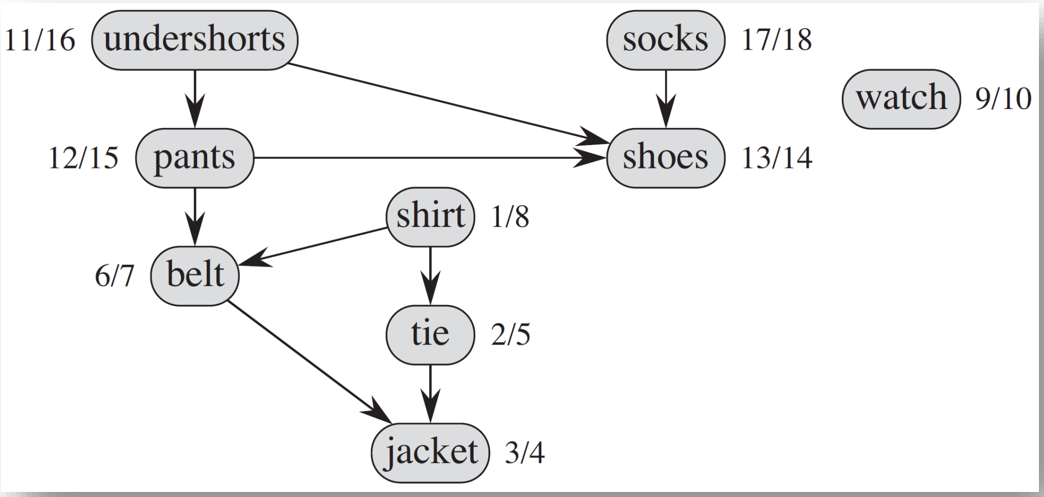
- Step3: $DFS(G^T)$, but in the main loop of DFS, consider the vertices in order of decreasing $u.f$

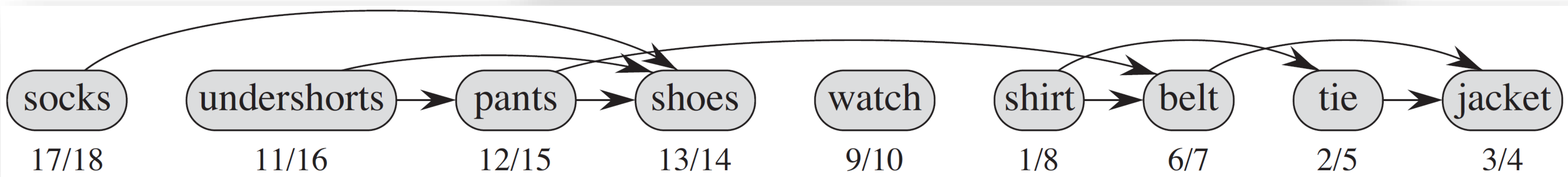
$$a \to b \to c \to f \to e \to g \to h \to i \to d$$



Topologocal Sort.

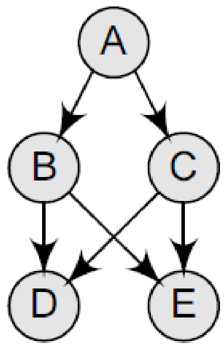
- Depth-first search can be used to perform a topological sort of a directed acyclic graph (DAG)
  - A ***topological sort*** of a DAG $G = (V, E)$ is a linear ordering of all its vertices
    - If G contains an edge (u, v), then u appears before v in the ordering





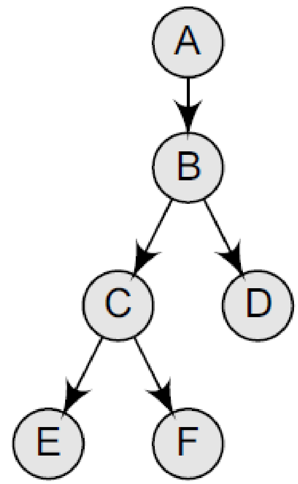
Topological Sorting..

- Every DAG has one or more number of topological sorts



Topological sort can be given as:

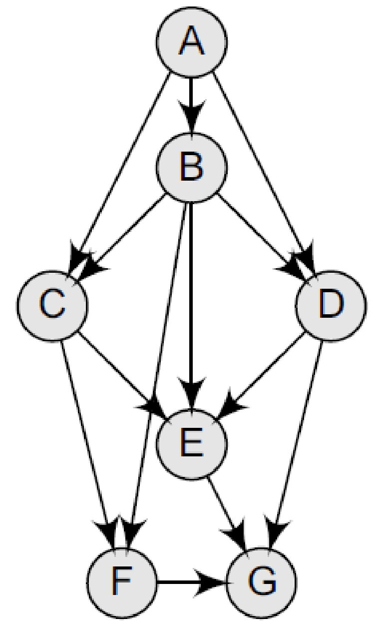
- A, B, C, D, E
- A, B, C, E, D
- A, C, B, D, E
- A, C, B, E, D



Topological sort can be given as:

- A, B, D, C, E, F
- A, B, D, C, F, E
- A, B, C, D, E, F
- A, B, C, D, F, E

....



Topological sort can be given as:

- A, B, C, F, D, E, C
- A, B, C, D, E, F, G
- A, B, C, D, F, E, G
- A, B, D, C, E, F, G

....

Questions?

kychen@mail.ntust.edu.tw