Applications of DFS

DFS(G)

```
For each vertex u ∈ G.V
    u.color = WHITE
    u.π = NIL
time = 0
For each vertex u ∈ G.V
    if u.color == WHITE
        DFS-Visit(G,u)
```

DFS-Visit(G,u)

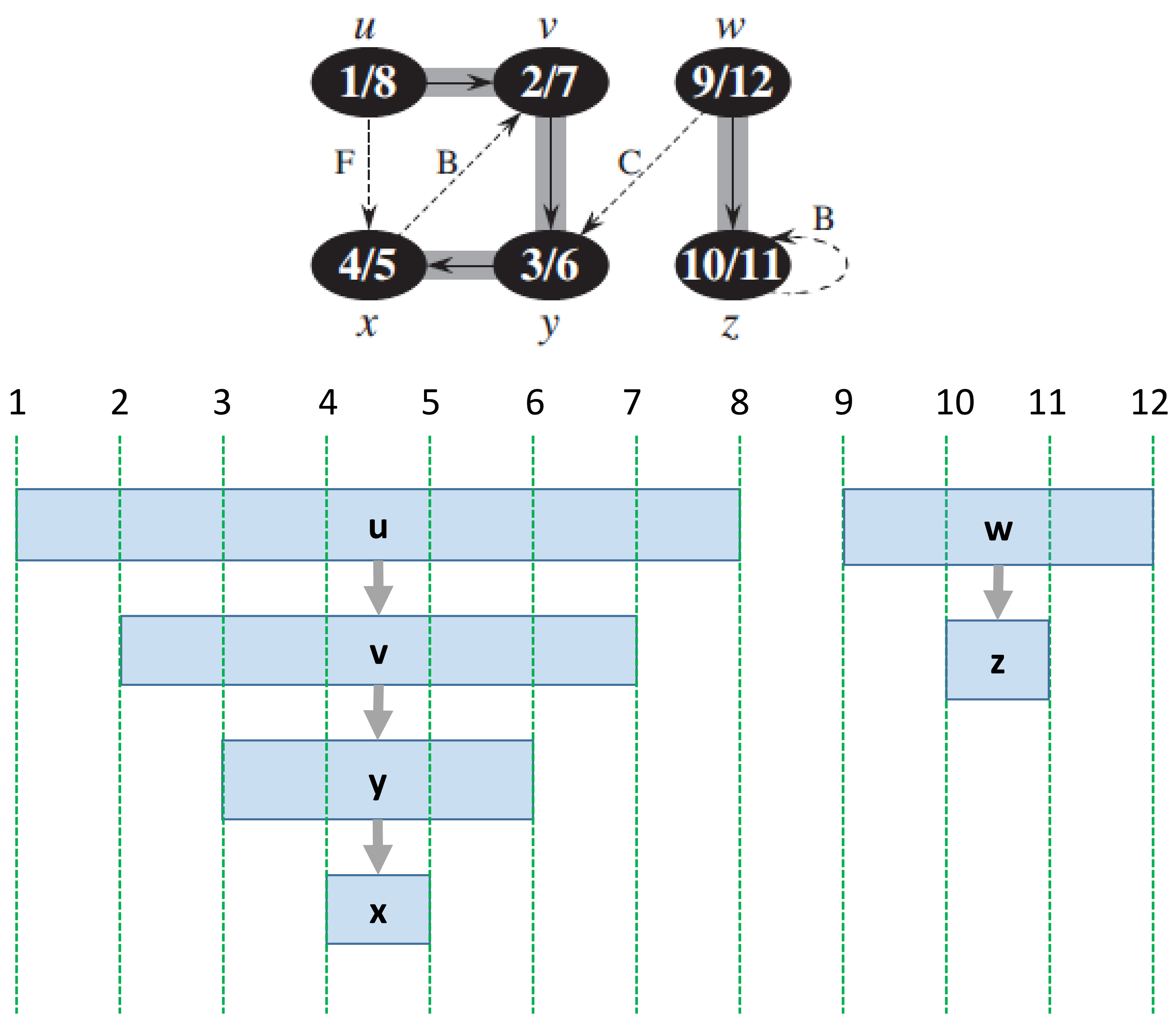
```
time = time + 1
u.d = time
u.color = GRAY
For each vertex v ∈ G.Adj[u]
    if v.color == WHITE
        v.π = u
        DFS-Visit(G,v)
u.color = BLACK
time = time + 1
u.f = time
```

Each vertex has two timestamps:

$\boldsymbol{u.d}$: First discovered and GRAYed

$\boldsymbol{u.f}$: Finished and Blackened

$\forall \boldsymbol{u}: 1 \leq \boldsymbol{u.d} < \boldsymbol{u.f} \leq 2\boldsymbol{n}$



u
v
w
1/8
2/7
9/12
F
B
C
B
4/5
3/6
10/11
x
y
z
1
2
3
4
5
6
7
8
9
10
11
12
u
w
v
z
y
x

```
DFS(G)
For each vertex u ∈ G.V
        u.color = WHITE
        u.π = NIL
time = 0
For each vertex u ∈ G.V
        if u.color == WHITE
                DFS-Visit(G,u)
```

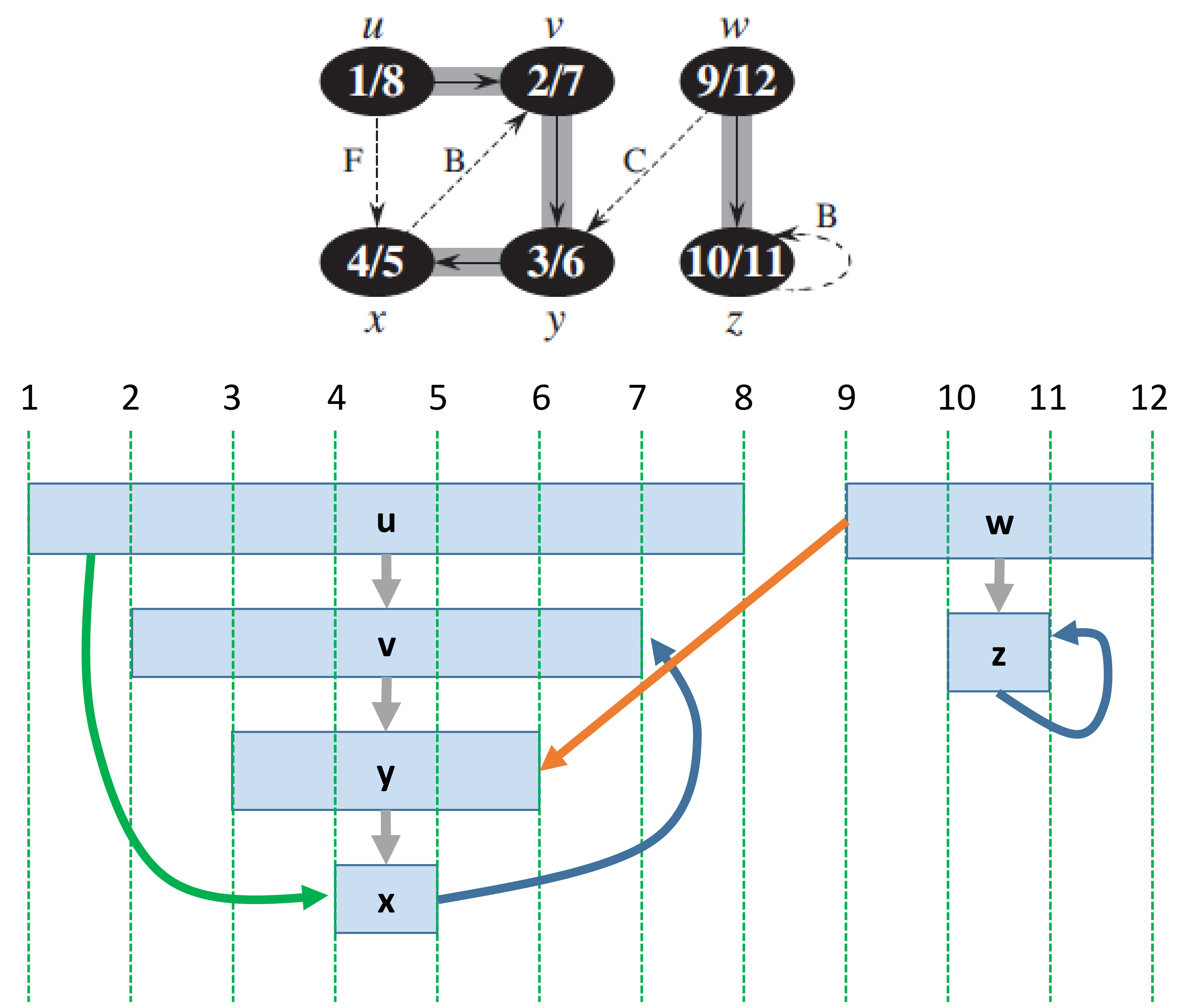
```
DFS-Visit(G,u)
time = time + 1
u.d = time
u.color = GRAY
For each vertex v ∈ G.Adj[u]
        if v.color == WHITE
                v.π = u
                DFS-Visit(G,v)
u.color = BLACK
time = time + 1
u.f = time
```

Thm 2: $[v.d, v.f] \subset [u.d, u.f]$ iff v is a descendant of u in the DFS-forest, $[v.d, v.f] \cap [u.d, u.f] = \emptyset$ iff u and v are unrelated in the DFS-forest $\square$

```
DFS(G)
For each vertex u ∈ G.V
    u.color = WHITE
    u.π = NIL
time = 0
For each vertex u ∈ G.V
    if u.color == WHITE
        DFS-Visit(G,u)
```

```
DFS-Visit(G,u)
time = time + 1
u.d = time
u.color = GRAY
For each vertex v ∈ G.Adj[u]
    if v.color == WHITE
        v.π = u
        DFS-Visit(G,v)
u.color = BLACK
time = time + 1
u.f = time
```

Thm 3: v is a descendant of u in the DFS-forest iff when we invoke DFS-Visit(u) there is a white path from u to v



u
v
w
1/8
2/7
9/12
F
B
C
B
4/5
3/6
10/11
x
y
z
1
2
3
4
5
6
7
8
9
10
11
12
u
w
v
z
y
x

Edge classification

- Tree edges: (u,v), DFS-Visit(v) was called from DFS-visit(u)
- Back edges: (u,v) such that v is an ancestor of u in the DFS-forest
- Forward edges: nontree edges (u,v) such that v is a descendant of u
- Cross edges: (u,v) such that $v.f < u.d$

Observe, among non-tree edges:

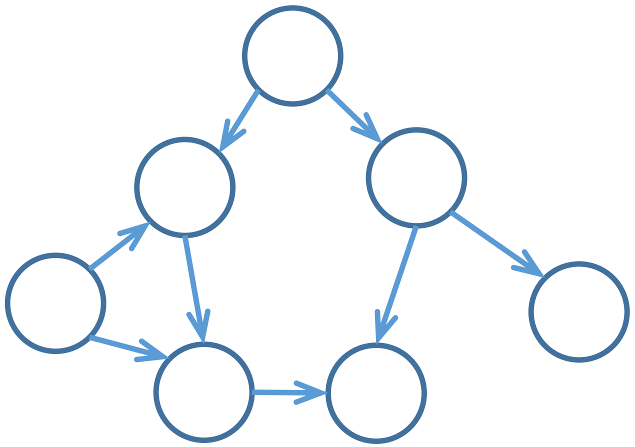
1. (Only) backward edges go to a vertex with a later finish time
2. (Only) forward edges go to a vertex with later discovery time

Is my directed graph acyclic ?

Thm 5: G is acyclic iff DFS on G does not produce back edges

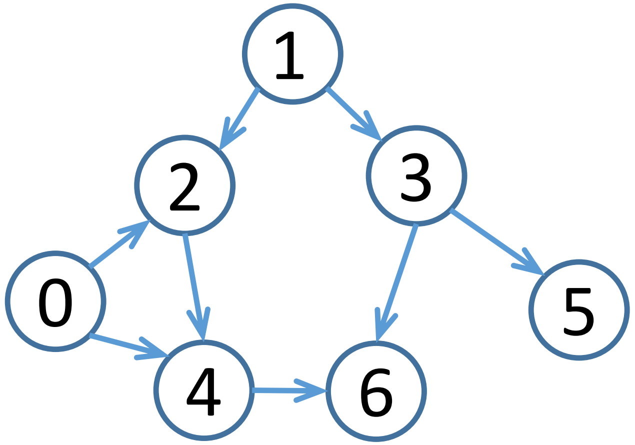
Topological sort

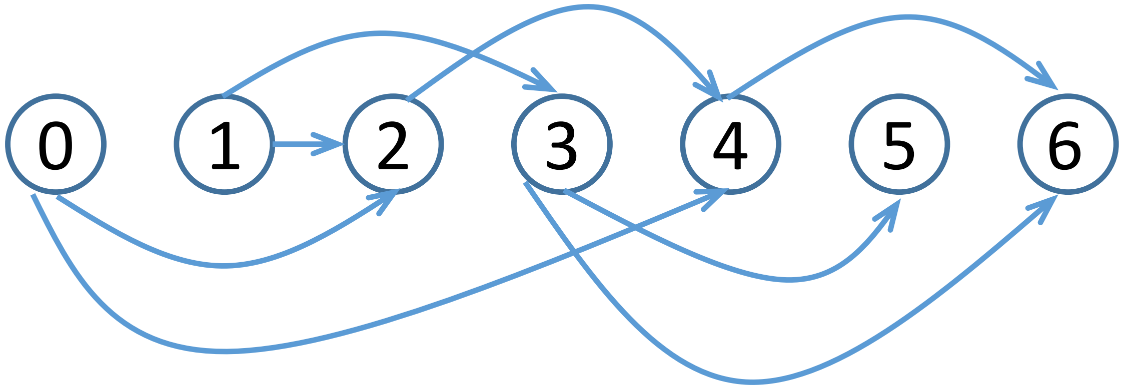
- Input: $G = (V, E)$ a directed, acyclic graph (DAG)



Topological sort

- Input: $G = (V, E)$ a directed, acyclic graph (DAG)
- Output: Ordering of the vertices such that if $(u, v) \in E$ then u precedes v





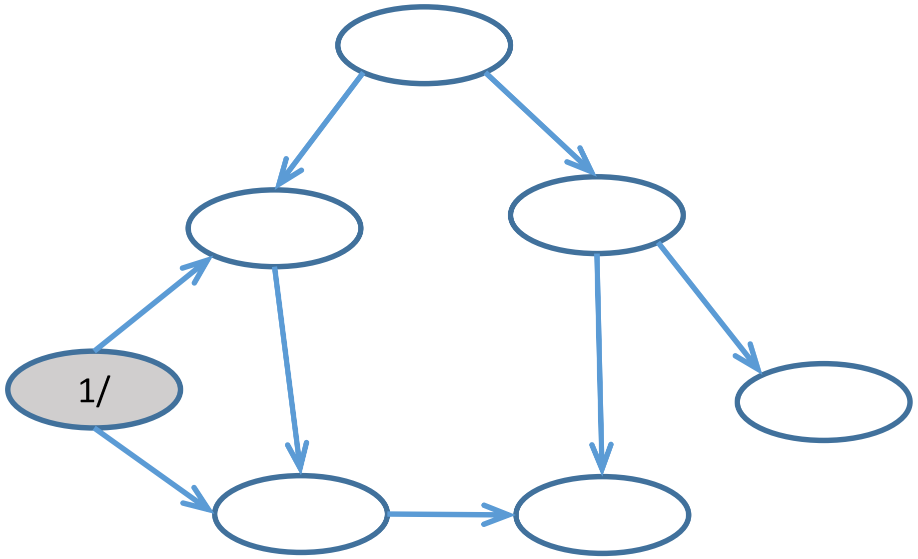
Recall:

Observe, among non-tree edges:

1. (Only) backward edges go to a vertex with a later finish time
2. (Only) forward edges go to a vertex with later discovery time

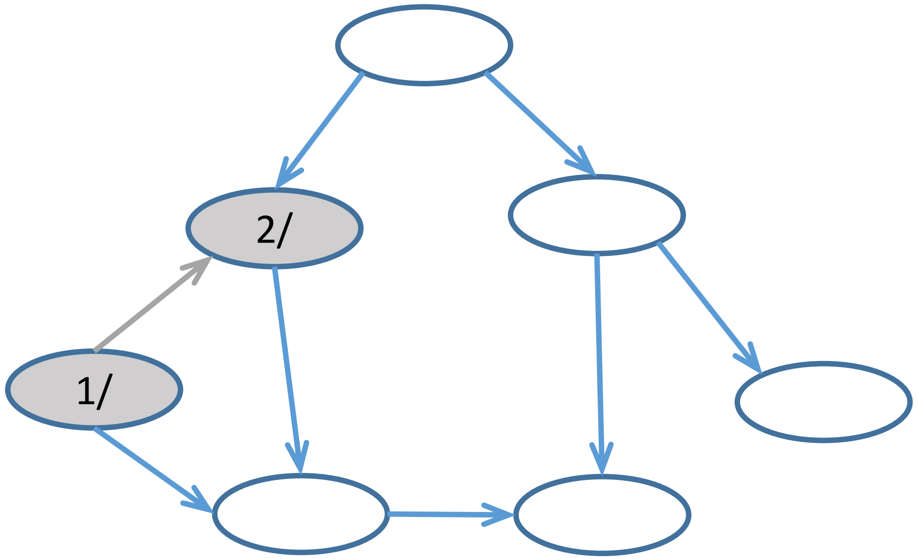
Topological sort

- Lets run DFS



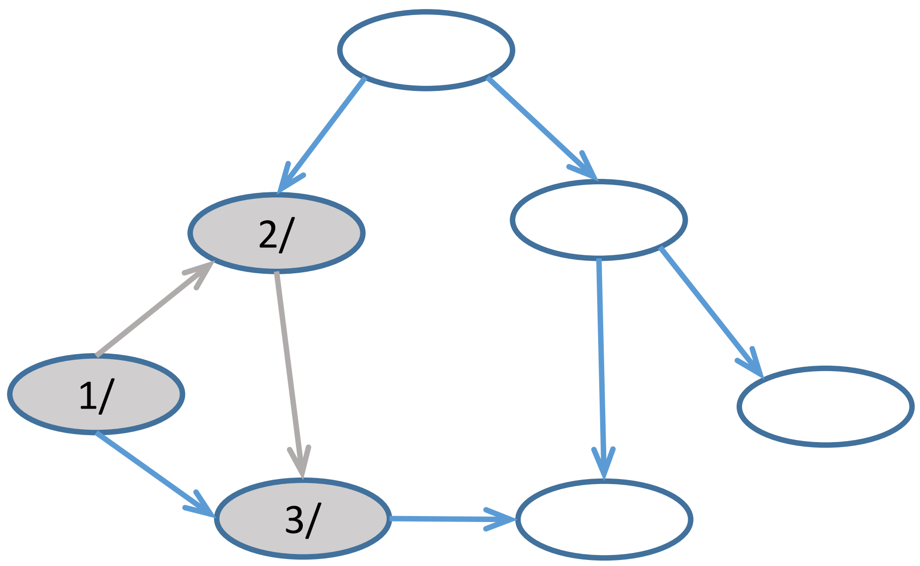
Topological sort

- Lets run DFS



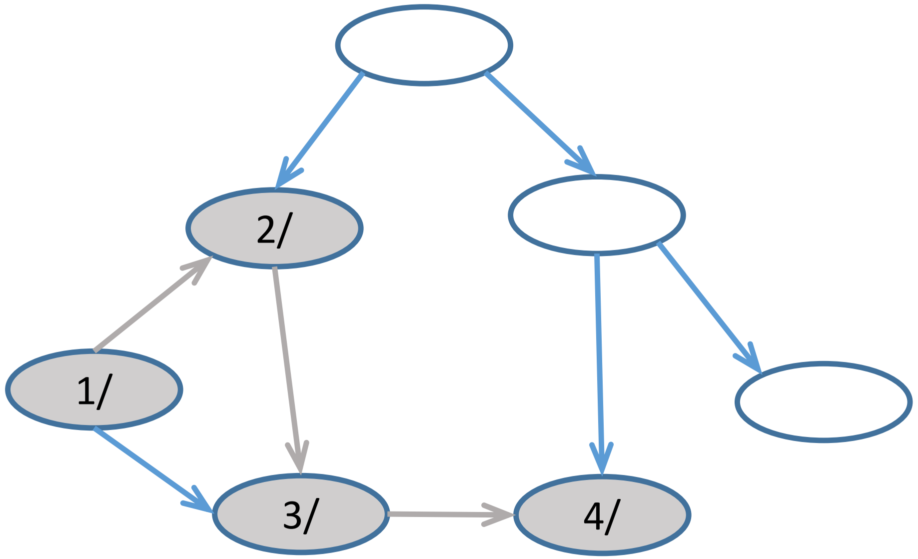
Topological sort

- Lets run DFS



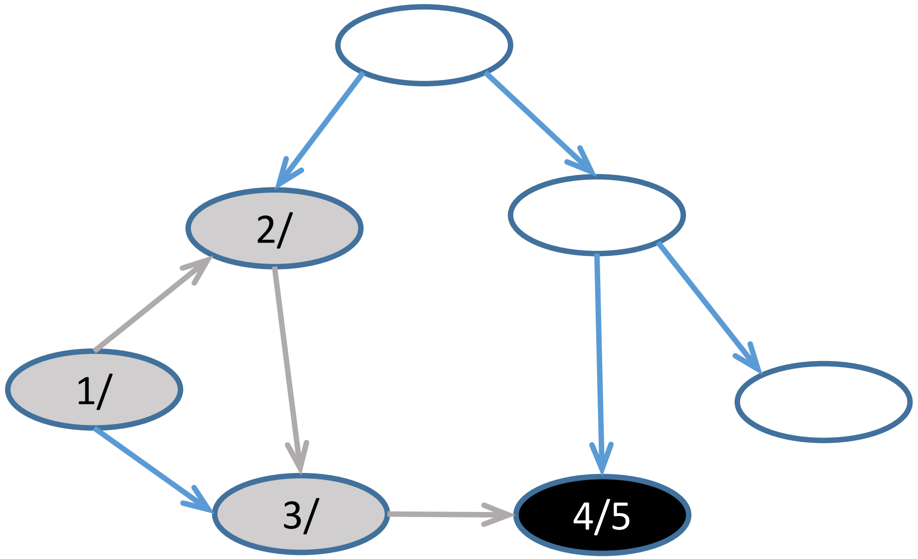
Topological sort

- Lets run DFS



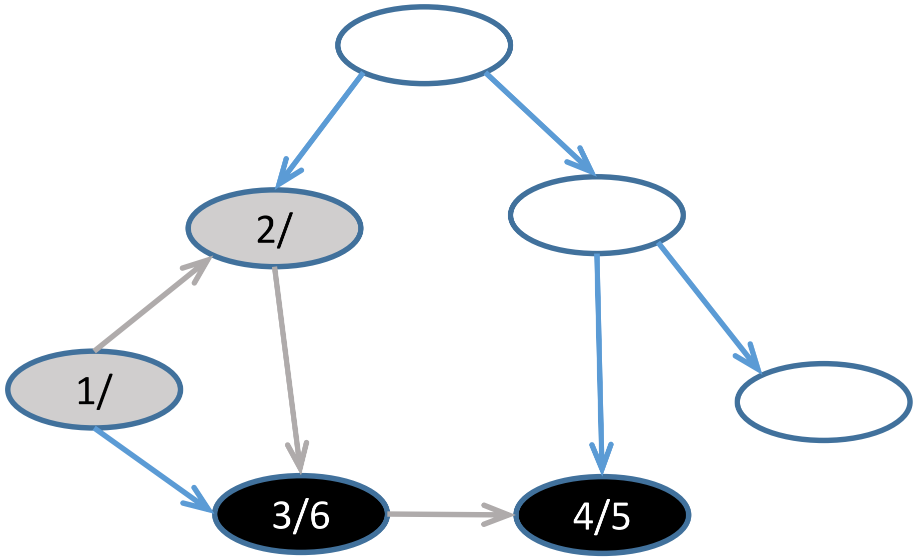
Topological sort

- Lets run DFS



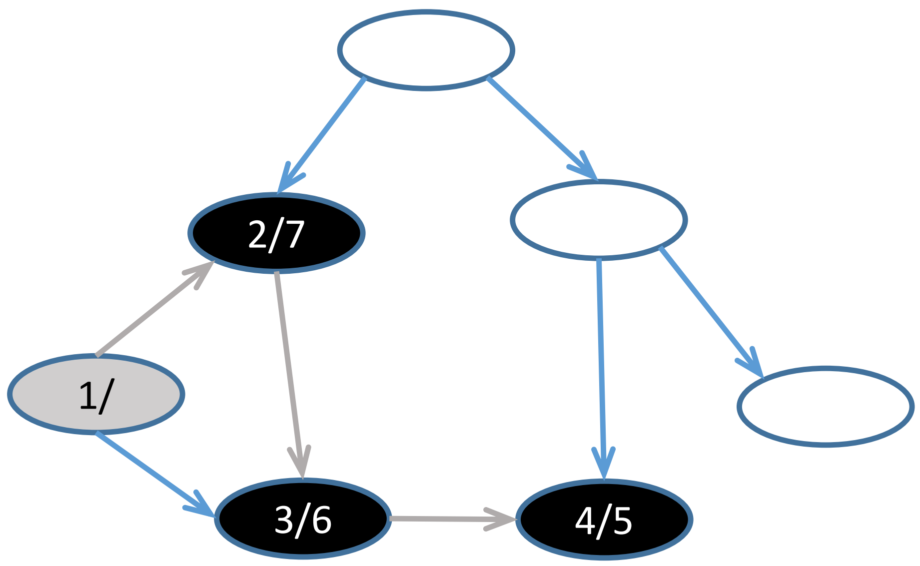
Topological sort

- Lets run DFS



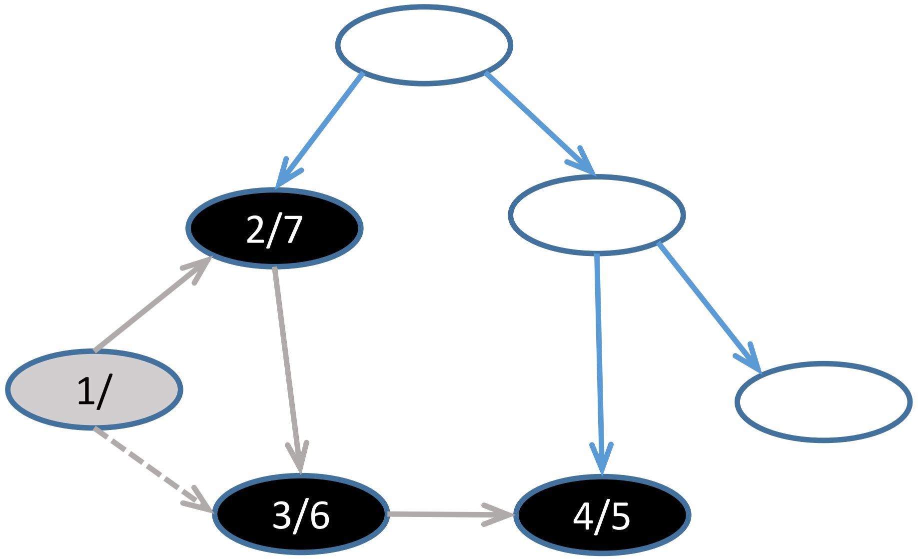
Topological sort

- Lets run DFS



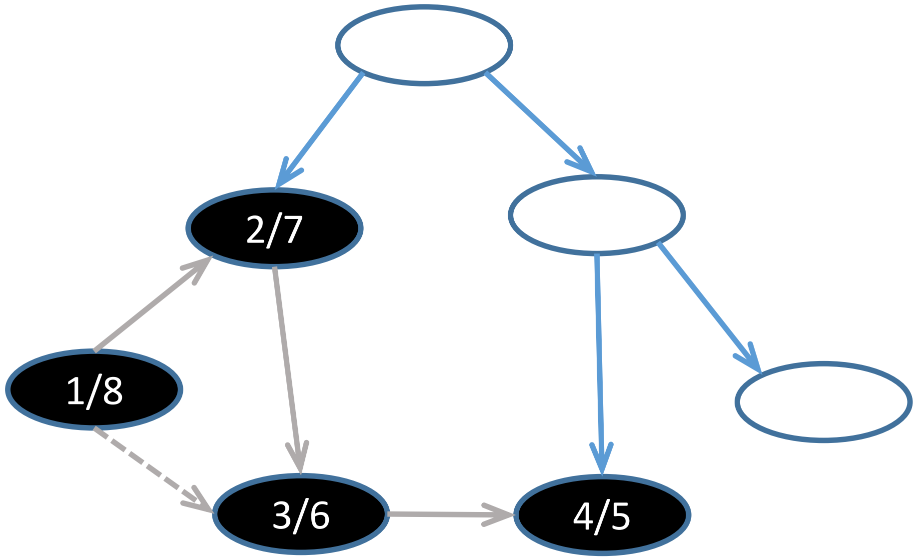
Topological sort

- Lets run DFS



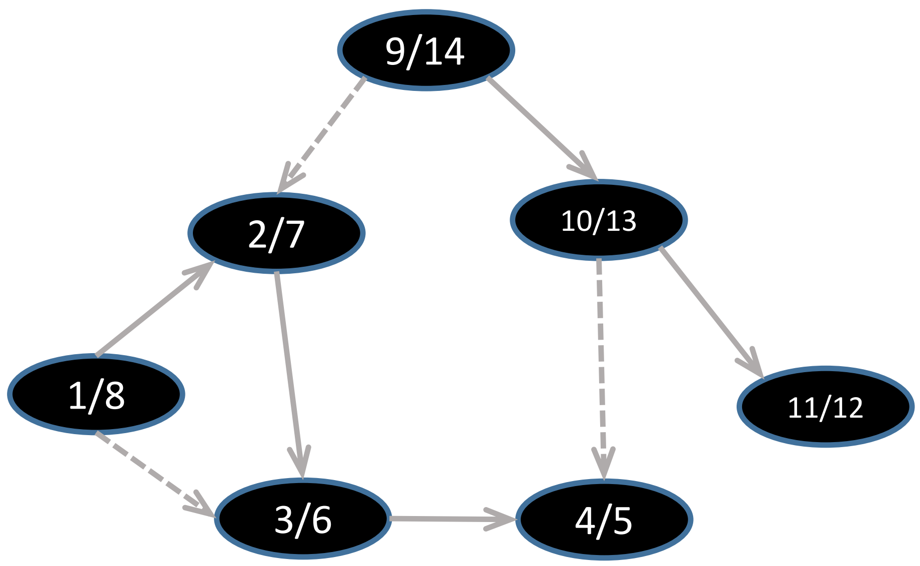
Topological sort

- Lets run DFS



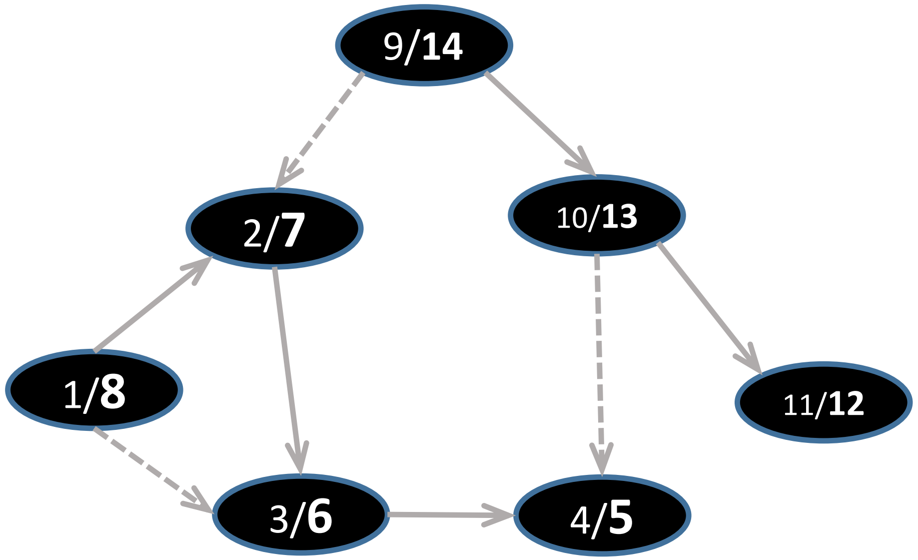
Topological sort

- Lets run DFS



Topological sort

- Order the vertices by reverse finishing times



Topological sort

- Order the vertices by reverse finishing times