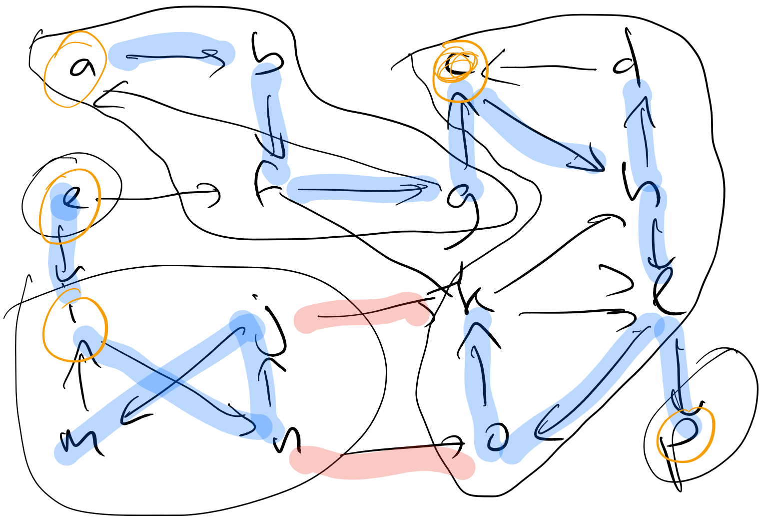
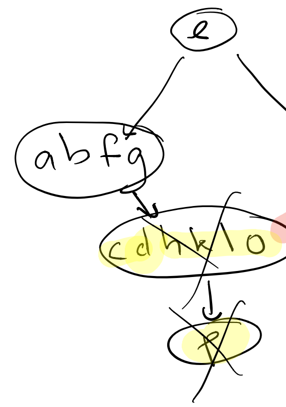
TODAY
- SCC decomposition
- shortest paths
  - BFS, DAG, Dijkstra
  - Bellman Ford





DFS(h) = {cdhklo}
DFS(p) = {p}

$$scc(v) = \underset{DFS}{reach(v)} \cap \underset{DFS}{reach^{-1}(v)}$$

Given v can find scc(v) in O(V+E) time

for all $v \in V$ $\quad O(V \cdot (V+E))$
DFS(v)
$DFS^{-1}(v)$ ⟵ DFS on reverse graph
compute intersection

① Every SCC has exactly one node w/ parent not inside SCC in a DFS tree

② DFS from any node in sink component returns all nodes in that SCC and no others

③ Last vertex in post order G is in source component

③b Last vertex in post-order rev(G) is in the sink component

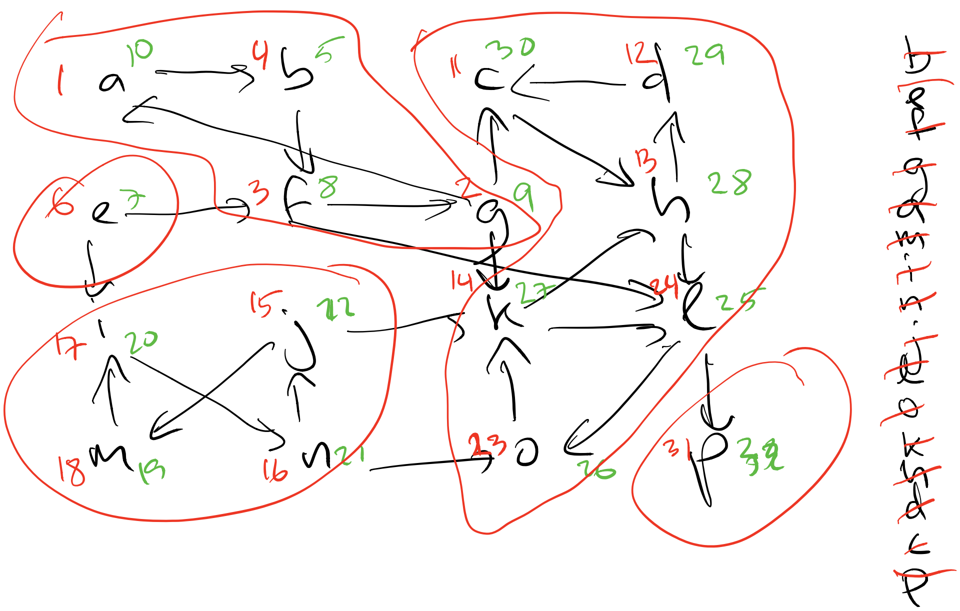
- do post order of rev(G)
- DFS$_G$ on node w/ highest post-order
- repeat for any unvisited nodes, taking highest node as DFS

```
KosarajuSharir(G):
    S ← new empty stack
    for all vertices v
        unmark v
        v.root ← None
    ⟨⟨Phase 1: Push in postorder in rev(G)⟩⟩
    for all vertices v
        if v is unmarked
            PushPostRevDFS(v, S)
    ⟨⟨Phase 2: DFS again in stack order⟩⟩
    while S is non-empty
        v ← Pop(S)
        if v.root = None
            LabelOneDFS(v, v)
```

```
PushPostRevDFS(v, S):
    mark v
    for each edge u→v    ⟨⟨Reversed!⟩⟩
        if u is unmarked
            PushPostRevDFS(u, S)
    Push(v, S)
```

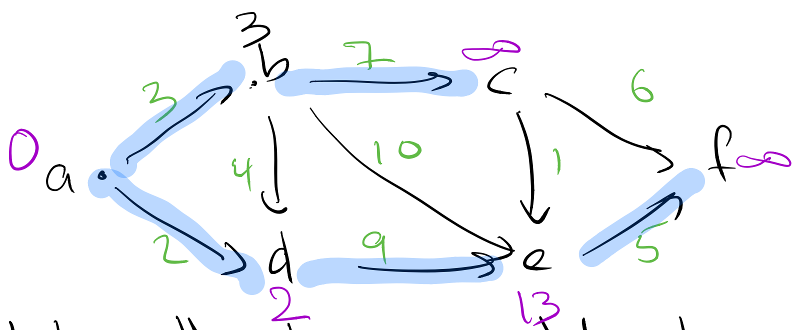
```
LabelOneDFS(v, r):
    v.root ← r
    for each edge v→w
        if w.root = None
            LabelOneDFS(w, r)
```

Figure 6.16. The Kosaraju-Sharir strong components algorithm



Shortest paths

- Given source s target t find shortest path from s to t
- Given source s find shortest path to all targets t
  Single source shortest path (SSSP)
- Given source s, find length of shortest path to all targets t



Shortest path tree rooted at source
where tree path to any t is shortest path

$$SP(s,u) \leq SP(s,v) + \ell(v,u)$$

DIST[u] → tentative SP length from s to u

DIST[s] = 0

DIST[u] = ∞ for all $u \neq s$

tense edge (v,u)

$$DIST[u] > DIST[v] + \ell(v,u)$$

relax $DIST[u] = DIST[v] + \ell(v,u)$

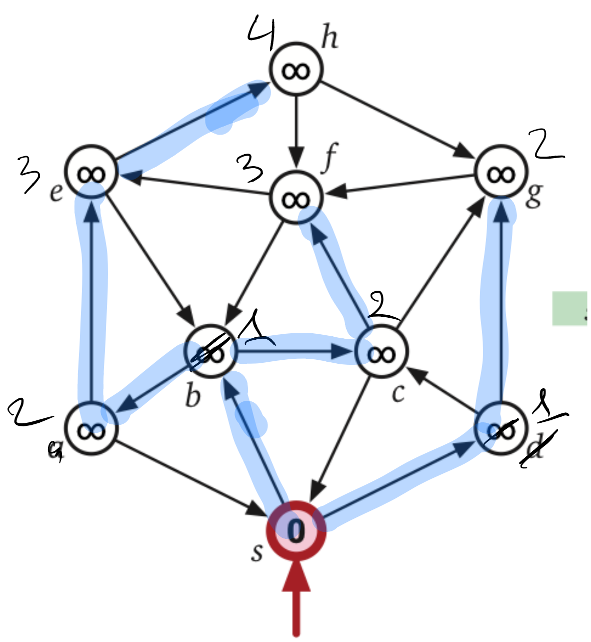
b→e	DIST[e]	DIST[b]	+	ℓ(b→e)
	∞	∞	+	7
a→b	∞	0	+	3

If edges are unweighted

BFS

BFS(s):
  INITSSSP(s)
  PUSH(s)
  while the queue is not empty
    $u \leftarrow$ PULL()
    for all edges $u \to v$
      if $dist(v) > dist(u) + 1$ ⟨⟨if $u \to v$ is tense⟩⟩
        $dist(v) \leftarrow dist(u) + 1$ ⟨⟨relax $u \to v$⟩⟩
        $pred(v) \leftarrow u$
        PUSH(v)

INITSSSP(s):
  $dist(s) \leftarrow 0$
  $pred(s) \leftarrow$ NULL
  for all vertices $v \neq s$
    $dist(v) \leftarrow \infty$
    $pred(v) \leftarrow$ NULL



~~s~~ ~~b~~ ~~d~~ ~~a~~ ~~c~~ ~~g~~ ~~e~~ ~~f~~ ~~h~~
0 | 1 1 | 2 2 2 | 3 3 | 4

$O(V+E)$

SSSP on DAGs

$$SP(s,u) \leq SP(s,v) + \ell(v,u)$$

$$\boxed{\begin{aligned} SP(s,u) &= \min_{v \to u} SP(s,v) + \ell(v,u) \\ SP(s,s) &= 0 \end{aligned}}$$

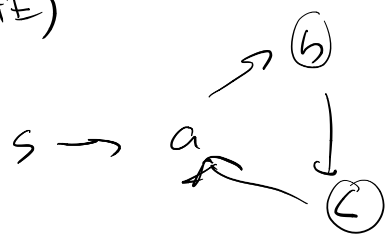
$SP(s,s)=0$ $SP(s,t)=\infty$ for all $t \neq s$

For u in topo order of G

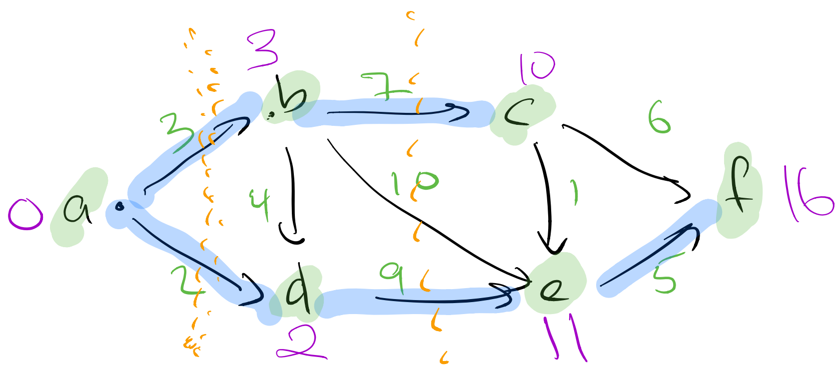
$$SP(s,u) = \min_{v \to u} \left(SP(s,v) + \ell(v,u) \right)$$

$$\text{parent}[u] = \arg\min_{v} SP(s,v) + \ell(v,u)$$

$O(V+E)$



Dijkstra



Finished nodes → have exact SP length

Unfinished nodes → don't " " "

Relax edges from finished nodes to unfinished

Closest unfinished node can be marked finished

Dijkstra

mark all nodes unfinished

Init DIST[s] = 0 DIST[u] = ∞ ← $O(V)$

while there are unfinished nodes $O(V)$

- take smallest unfinished node v (DIST)
- mark as finished
- relax all edges from v

once per vertex V

Ex relax

$O(V^2)$ $O(V^2 + E)$ E is $O(V^2)$