Shortest paths

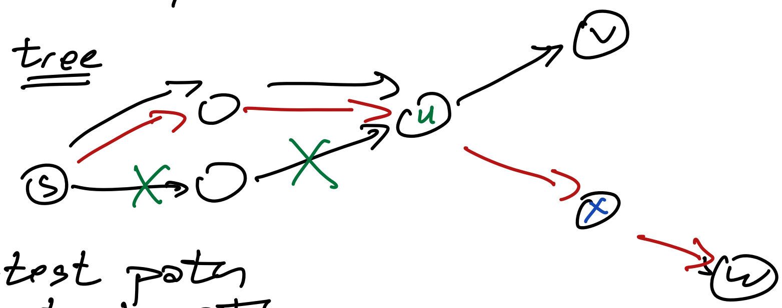
Given directed graph $G=(V,E)$

edge e has length $\ell(e)$

source vertex s

Find lengths of shortest paths from s to every other vertex in G

Shortest-path <u>tree</u>



Any ~~prefix~~ subpath of shortest path is a shortest path

Final output: dist(v) = shortest-path distance from s to v

pred(v) = second to last vertex on that path

Ford (1956)

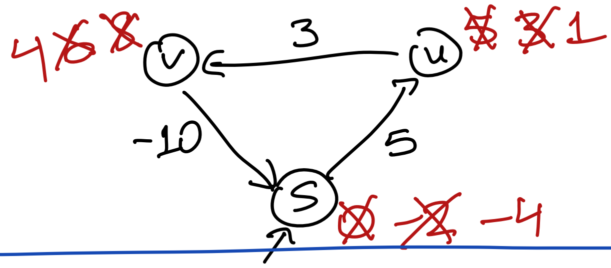
Init:
- dist(s) $\leftarrow 0$
- Pred(s) $\leftarrow$ NULL
- for all $v \neq s$
  - dist(v) $\leftarrow \infty$
  - Pred(v) $\leftarrow$ NULL

$u \to v$ is <u>tense</u> if

$\text{dist}(u) + \ell(u\to v) < \text{dist}(v)$

Relax($u \to v$):
- dist(v) $\leftarrow$ dist(u) + $\ell(u\to v)$
- pred(v) $\leftarrow u$

<u>SSSP:</u>
- Init
- while any edge is tense
  - relax any tense edge



Negative cycles are bad.

Unweighted — BFS	Non-neg weights — Dijkstra
DAG — DFS/DP	Arbitrary weights — Bellman Ford

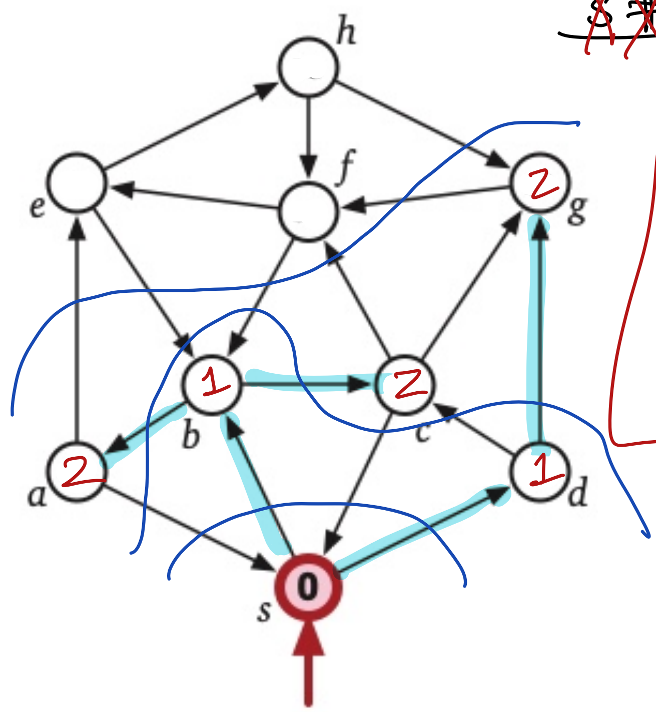
```
BFSWithToken(s):
  InitSSSP(s)
  Push(s)
  Push(✠)                        ⟨⟨start the first phase⟩⟩
  while the queue contains at least one vertex
      u ← Pull()
      if u = ✠
          Push(✠)                ⟨⟨start the next phase⟩⟩
      else
          for all edges u→v
              if dist(v) > dist(u) + 1        ⟨⟨if u→v is tense⟩⟩
                  dist(v) ← dist(u) + 1       ⟨⟨relax u→v⟩⟩
                  pred(v) ← u
                  Push(v)
```

O(E)

O(V)

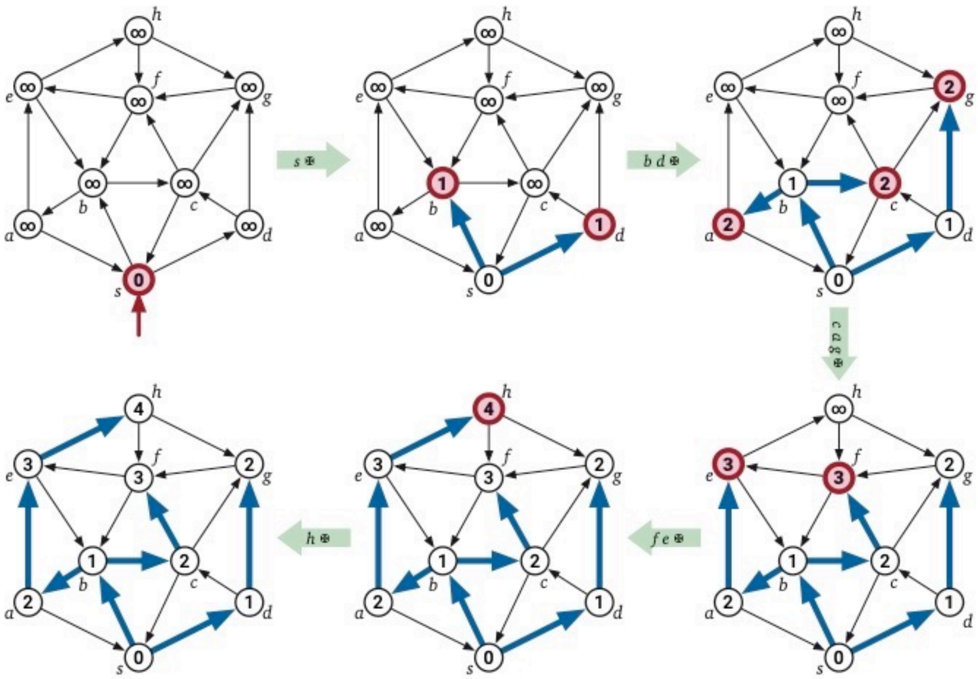
Queue ←

~~s#bd#~~acg#



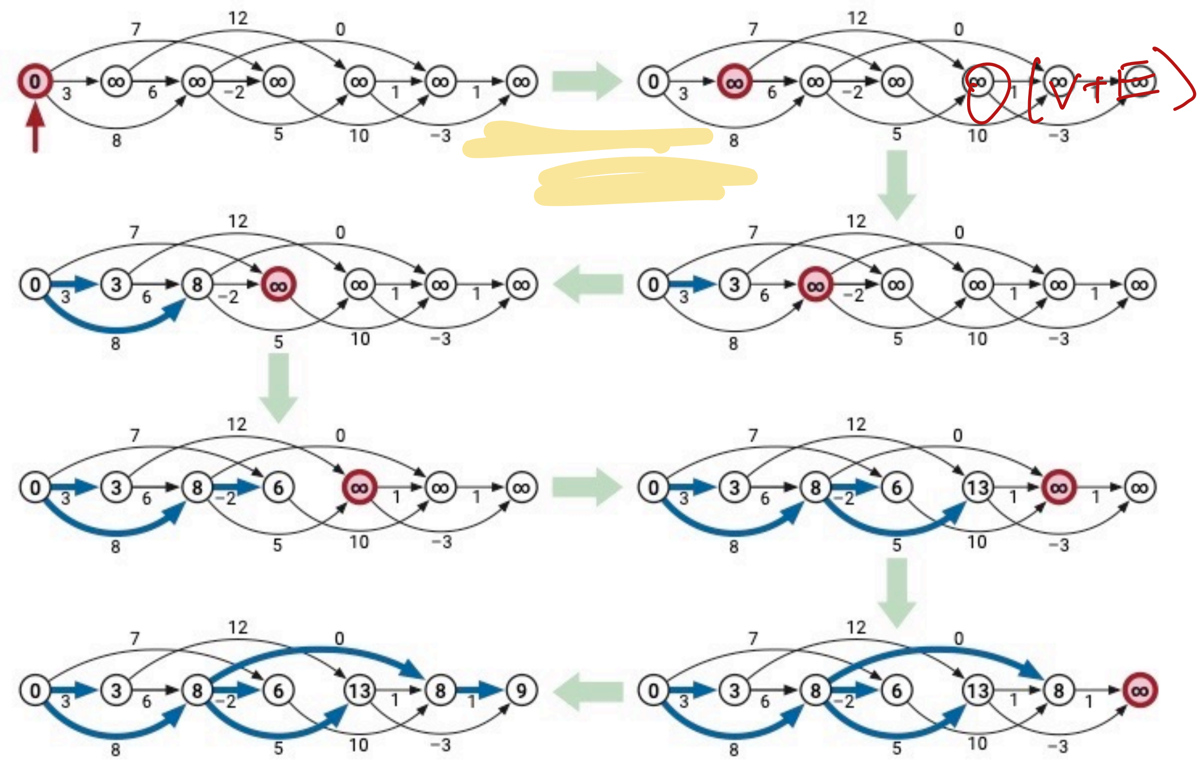
At the end of the ith phase, we have found all verts with dist ≤ i

O(V+E) time



s ⊕
b d ⊕
c a g ⊕
f e ⊕
h ⊕

DAGSSSP(s):



$O(V+E)$

$$dist(v) = \begin{cases} 0 & \text{if } v = s \\ \min\limits_{u \to v} \left(dist(u) + w(u \to v) \right) & \text{otherwise} \end{cases}$$

$\min \emptyset = \infty$

length of shortest path from s to v.

worst case: $\Theta(2^V)$ Non-neg: $O(E \log V)$

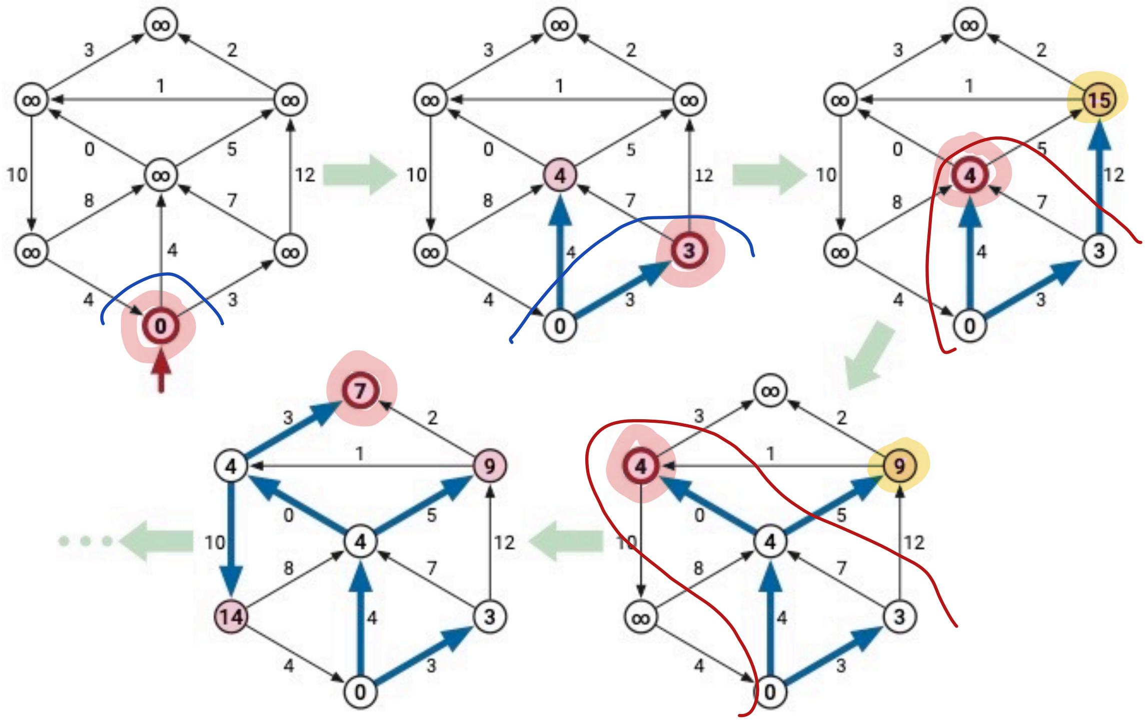
Always correct, usually fast in practice

```
DIJKSTRA(s):
  INITSSSP(s)
  INSERT(s, 0)
  while the priority queue is not empty
      u ← EXTRACTMIN()
      for all edges u→v
          if u→v is tense
              RELAX(u→v)
              if v is in the priority queue
                  DECREASEKEY(v, dist(v))
              else
                  INSERT(v, dist(v))
```

Priority queue:
contains vertices
priority = dist

only if some
outgoing edge
might be tense

$3 \xrightarrow{4} \cancel{12}\ 7 \xrightarrow{3} 15$



If all edge weights $\geq 0 \implies$ dist(v) is correct when v is Extracted

```
NONNEGATIVEDIJKSTRA(s):
  INITSSSP(s)
  for all vertices v
      INSERT(v, dist(v))
  while the priority queue is not empty
      u ← EXTRACTMIN()
      for all edges u→v
          if u→v is tense
              RELAX(u→v)
              DECREASEKEY(v, dist(v))
```

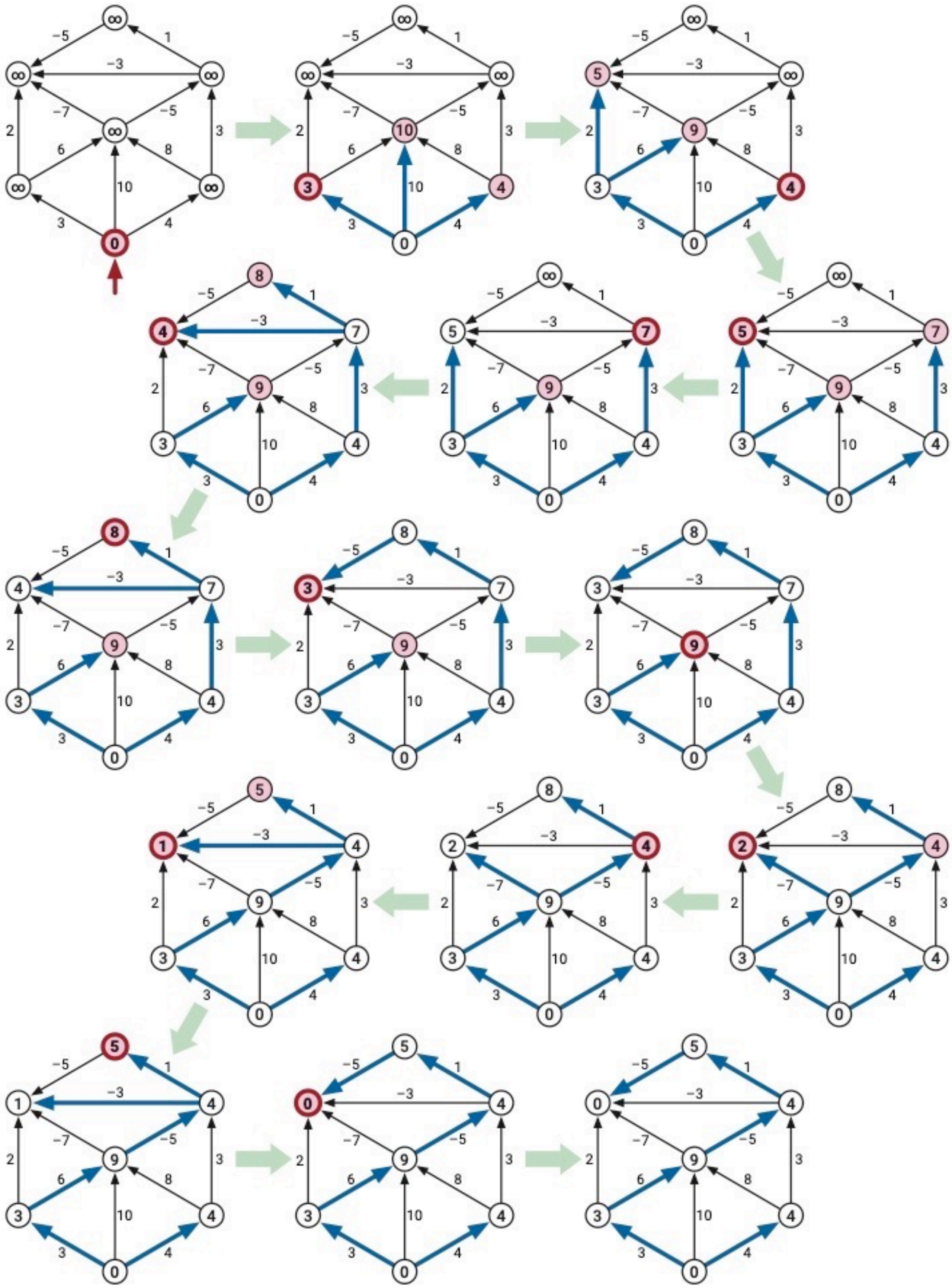
at most V times

O(log V) each

Wikipedia

at most E times

O(E log V) time



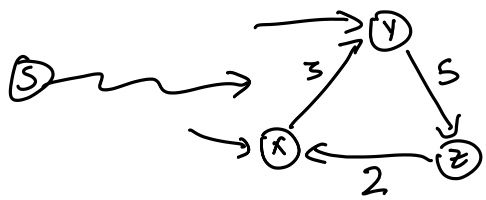
```
BellmanFord(s)
    InitSSSP(s)
    repeat V times:
        for every edge u→v
            if u→v is tense
                Relax(u→v)
```

```
BellmanFord(s)
    InitSSSP(s)
    repeat V − 1 times
        for every edge u→v
            if u→v is tense
                Relax(u→v)
    for every edge u→v
        if u→v is tense
            return "Negative cycle!"
```

O(VE) time

dist(v) = distance from s to v:

$$dist(v) = \begin{cases} 0 & \text{if } v = s \\ \min\limits_{u \to v} \left(dist(u) + w(u \to v) \right) & \text{otherwise} \end{cases}$$

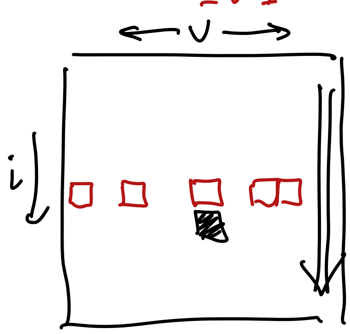


dist(z)
calls dist(y)
calls dist(x)
calls dist(z)
⋮
Bus error!
Seg fault!

$dist_{\le i}(v)$ = length of shortest path from s to v with $\le i$ edges

$$dist_{\le i}(v) = \begin{cases} 0 & \text{if } \cancel{i = 0 \text{ and}}\ v = s \\ \infty & \text{if } i = 0 \text{ and } v \ne s \\ \min \left\{ \begin{array}{c} dist_{\le i-1}(v) \\ \min\limits_{u \to v} \left(dist_{\le i-1}(u) + w(u \to v) \right) \end{array} \right\} & \text{otherwise} \end{cases}$$

We want $dist_{\le V-1}(v)$ for all v.



BellmanFordDP(s)

$dist[0, s] \leftarrow 0$
for every vertex $v \ne s$
  $dist[0, v] \leftarrow \infty$
for $i \leftarrow 1$ to $V - 1$
  for every vertex v
    $dist[i, v] \leftarrow dist[i-1, v]$
    for every edge $u \to v$
      if $dist[i, v] > dist[i-1, u] + w(u \to v)$
        $dist[i, v] \leftarrow dist[i-1, u] + w(u \to v)$

```
ObviousAPSP(V, E, w):
    for every vertex s
        dist[s, ·] ← SSSP(V, E, w, s)
```

$\Theta(V^3)$ time

```
FloydWarshall(V, E, w):
    for all vertices u
        for all vertices v
            dist[u, v] ← w(u→v)

    for all vertices r
        for all vertices u
            for all vertices v
                if dist[u, v] > dist[u, r] + dist[r, v]
                    dist[u, v] ← dist[u, r] + dist[r, v]
```