Midterm 2 Next Mon 7-9pm

recursion — divide and conquer, backtracking, DP, ~~greedy~~

graphs — traversal, top. sort, shortest paths, ~~MST~~

Conflict Tue 10-1 / Review Thu Fri

DRES **today** / Hopefully Sat (stay tuned)

Allpairs shortest path

Input: Directed graph $G=(V,E)$

edge weights $w(e)$ for each edge e

Output: $dist[1..V, 1..V]$ — shortest path lengths

$pred[1..V, 1..V]$ — predecessors

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

Best possible (?)

G is unweighted — SSSP = BFS — $O(V(V+E)) = O(V^3)$

G is a dag — SSSP = DFS — $O(V(V+E)) = O(V^3)$

G has no neg edges — SSSP = Dijkstra — $O(VE\log V) = O(V^3 \log V)$

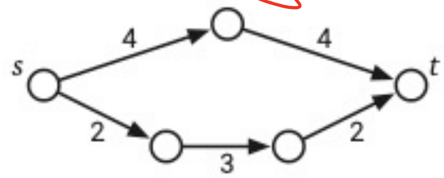
o/w — SSSP = Bellman-Ford — $O(V^2E) = O(V^4)$

Better [Chan et al.] — $O(V^3/\log^2 V)$ time

Better?? MAJOR OPEN PROBLEM!

Negative weights are bad.

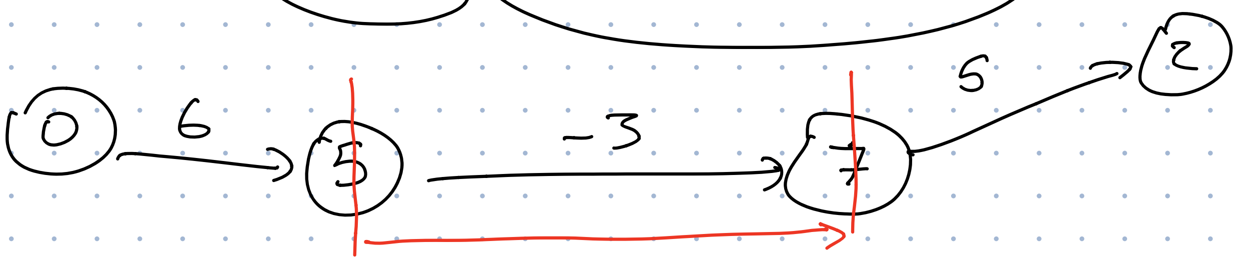
Adding same to every edge changes shortest paths.



Reweighting / Prices

Price

$$w'(u\to v) = \pi(u) + w(u\to v) - \pi(v)$$



$$w'(s \rightsquigarrow t) = \pi(s) + w(s \rightsquigarrow t) - \pi(t)$$

Fix a vertex s, compute dist(s,v) for all v via BF.

$$\boxed{\text{Set } \pi(v) = dist(s,v)}$$

$$w'(u\to v) = \boxed{dist(s,u) + w(u\to v) - dist(s,v)} \geq 0$$

```
JOHNSONAPSP(V, E, w):
  ⟨⟨Add an artificial source⟩⟩
  add a new vertex s
  for every vertex v
      add a new edge s→v
      w(s→v) ← 0
  ⟨⟨Compute vertex prices⟩⟩
  dist[s,·] ← BELLMANFORD(V, E, w, s)
  if BELLMANFORD found a negative cycle
      fail gracefully
  ⟨⟨Reweight the edges⟩⟩
  for every edge u→v ∈ E
      w′(u→v) ← dist[s,u] + w(u→v) − dist[s,v]
  ⟨⟨Compute reweighted shortest path distances⟩⟩
  for every vertex u
      dist′[u,·] ← DIJKSTRA(V, E, w′, u)
  ⟨⟨Compute original shortest-path distances⟩⟩
  for every vertex u
      for every vertex v
          dist[u,v] ← dist′[u,v] − dist[s,u] + dist[s,v]
```

$O(VE\log V) = O(V^3 \log V)$

Dynamic Programming

$$dist(u,v) = \begin{cases} 0 & \text{if } u = v \\ \min\limits_{x \to v} \big(dist(u,x) + w(x \to v)\big) & \text{otherwise} \end{cases}$$

Additional parameter = max #edges.

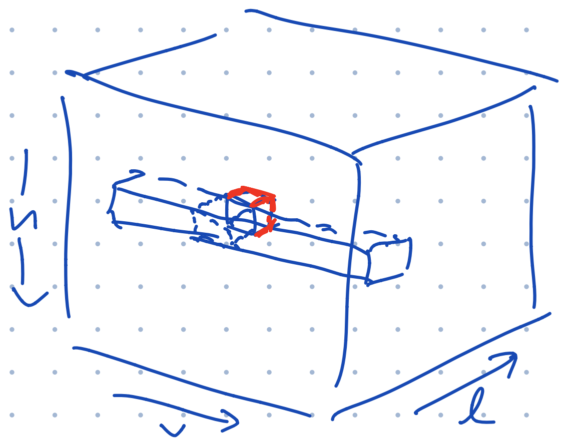
$O(V^4)$

$$dist(u,v,\ell) = \begin{cases} 0 & \text{if } \ell = 0 \text{ and } u = v \\ \infty & \text{if } \ell = 0 \text{ and } u \neq v \\ \min \left\{ \begin{array}{c} dist(u,v,\ell-1) \\ \min\limits_{x \to v} \big(dist(u,x,\ell-1) + w(x \to v)\big) \end{array} \right\} & \text{otherwise} \end{cases}$$

length of shortest path from u to v with $\leq \ell$ edges

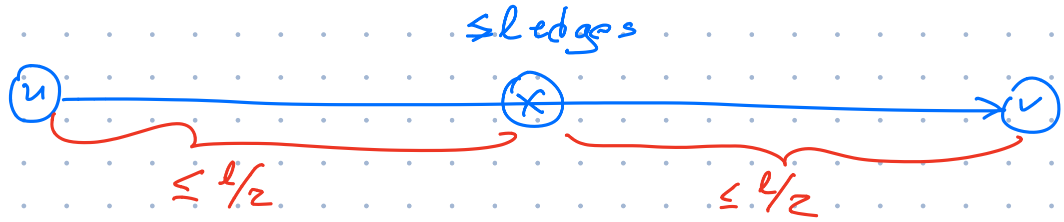
Memoize into 3D table

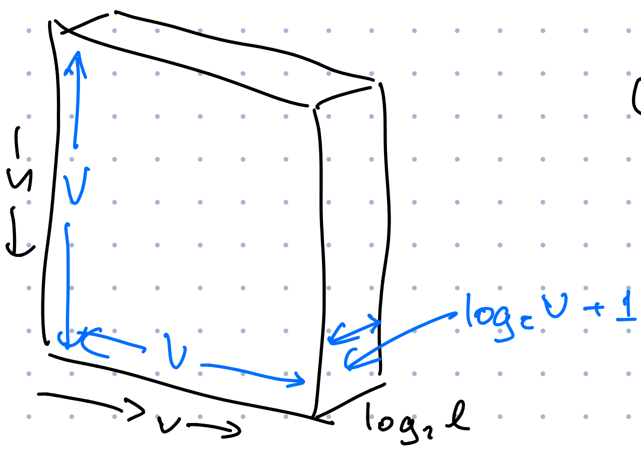
= V × Bellman Ford



$O(\log V)$ values

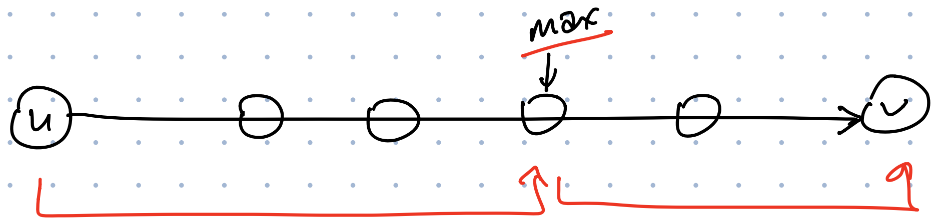
$$dist(u,v,\ell) = \begin{cases} w(u \to v) & \text{if } i = 1 \\ \min_x \big(dist(u,x,\ell/2) + dist(x,v,\ell/2)\big) & \text{otherwise} \end{cases}$$



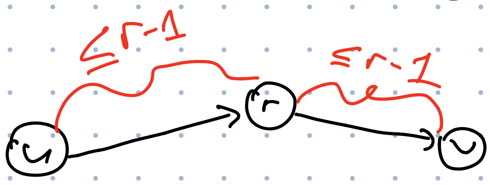


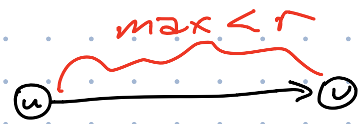
$O(V^3 \log V)$ time

```
LEYZOREKAPSP(V, E, w):
  for all vertices u
      for all vertices v
          dist[u, v] ← w(u→v)
  for i ← 1 to ⌈lg V⌉          ⟨⟨ℓ = 2^i⟩⟩
      for all vertices u
          for all vertices v
              for all vertices x
                  if dist[u, v] > dist[u, x] + dist[x, v]
                      dist[u, v] ← dist[u, x] + dist[x, v]
```



dist(u, v, r) = length of the shortest path from u to v where all interior vertices have index $\leq r$





$$dist(u,v,r) = \begin{cases} w(u \to v) & \text{if } r = 0 \\ \min \left\{ \begin{array}{c} dist(u,v,r-1) \\ dist(u,r,r-1) + dist(r,v,r-1) \end{array} \right\} & \text{otherwise} \end{cases}$$

$O(V^3)$ 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]
```

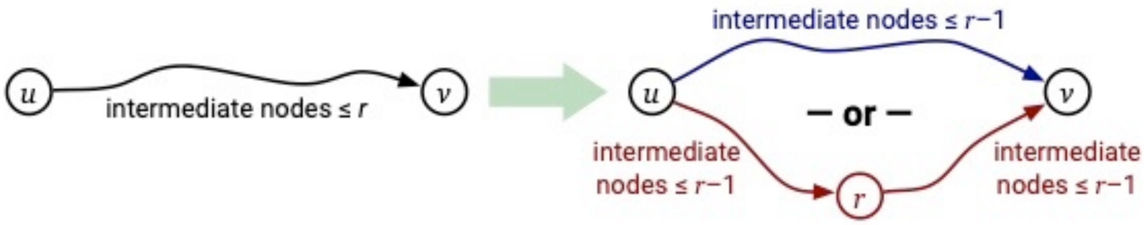
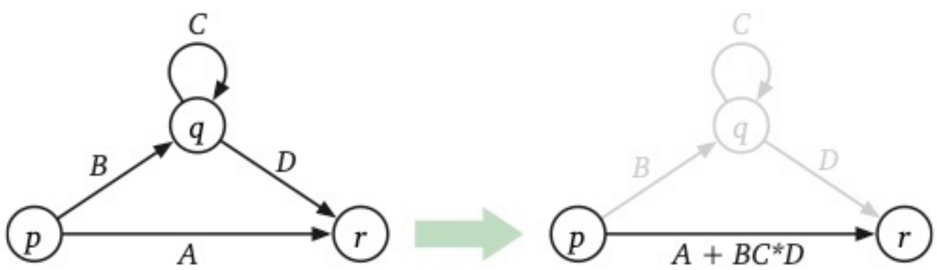


Figure 9.3. Recursive structure of the restricted shortest path $\pi(u, v, r)$.



One step in Kleene's/Han and Wood's reduction algorithm.