CS 223 Lec 23

- note about built-in Java Priority Queues ...
  - doesn't have decreaseKey(x)
  - instead:
    - remove(x)
    - decrease its key
    - add(x)

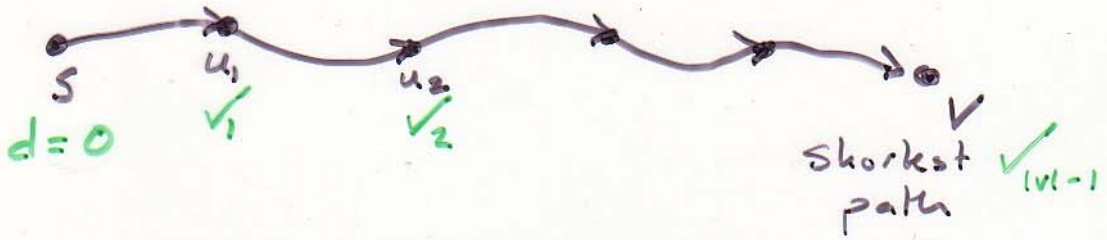
Proof that B-F is correct

Case 1: G doesn't contain a neg. weight cycle reachable from s.

So $\delta(s, v) \neq -\infty$ for any $v \in V$

Suppose $\delta(s, v) < \infty$

length $\leq |V| - 1$



Why B-F finds $d[v] = \delta(s, v)$?

after 1 iteration

$$d[u_1] = \delta(s, u_1)$$

after i iterations

$$d[u_i] = \delta(s, u_i)$$

after $|V| - 1$ iterations

$$d[v] = \delta(s, v)$$

Case 2 G contains a neg. weight cycle.

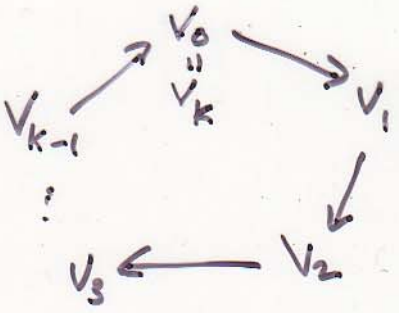
~~⇐⇒~~ B-F alg is correct provided it detects

an edge

$$d[v] > d[u] + w(u,v)$$

Assume B-F doesn't detect such an edge...

So $d[v] \leq d[u] + w(u,v)$ for all edges



neg. wt. cycle.

$$\text{So } \sum_{i=1}^{k} d[v_i] \leq \sum_{i=1}^{k} \left(d[v_{i-1}] + w(v_{i-1}, v_i) \right)$$

$\sum_{i=1}^{k} d[v_i]$ = sum all d values in the cycle = S

$\sum_{i=1}^{k} d[v_{i-1}]$ = S

... subtract S from both sides.

$$0 \leq \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

~~$\times$~~ with that $\langle v_0, \ldots v_{k-1} \rangle$ was supposed to be a neg. wt. cycle.

So B-F algorithm also works correctly in this case.

All-pairs Shortest Path Problem

Find $\delta(u, v)$ for all vertices $u, v \in V$

Floyd-Warshall Algorithm

- uses adjacency matrix to represent edges.

$$W = (w_{ij})$$

$$w_{ij} = \begin{cases} 0 & \text{if } i = j \\ \text{weight of edge } (i,j) & \text{if } i \neq j \text{ and } (i,j) \in E \\ \infty & \text{otherwise} \end{cases}$$

- no negative weight cycles.

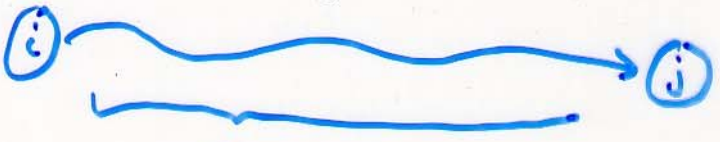
Problem

Let $D = (d_{ij})$

$d_{ij} = \delta(i,j)$

Find D ?

Let $d_{ij}^{(k)}$ = the shortest path distance from i to j that only uses vertices $\{1, \ldots, k\}$ as intermediate vertices.

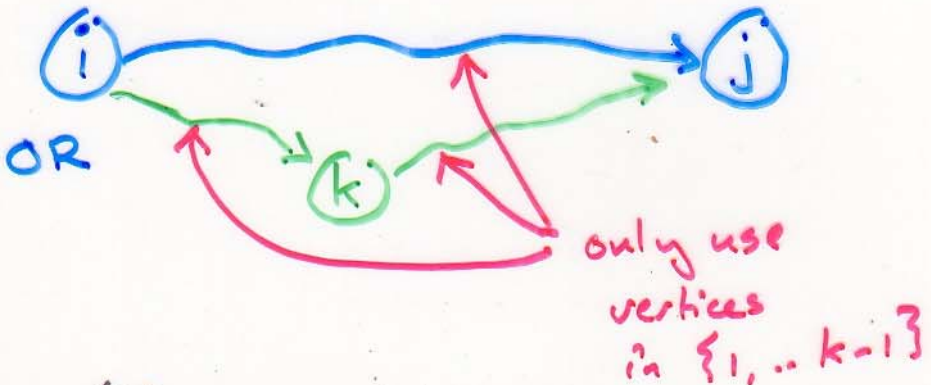


only use vertices from $\{1, .., k\}$

if $k = 0$

$$d_{ij}^{(0)} = w_{ij}$$

if $k > 0$,



$$d_{ij}^{(k)} = \min\left(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)} \right)$$

(recurrence relation)

let $D^{(k)} = (d_{ij}^{(k)})$

matrix

So $D^{(0)} = W$

from $D^{(0)}$ we can find $D^{(1)}$.

— $D^{(1)}$ ——— $D^{(2)}$

Want to find:

$$D^{(|V|)} = \left(d_{ij}^{(|V|)} \right)$$

and $d_{ij}^{|V|} = \delta(i,j)$

$D^{(0)} = W \quad (n = |V|)$

for $k = 1$ to n

  for $i = 1$ to n

    for $j = 1$ to n

      $D^{(k)}(i,j) =$

$$\min \left(\begin{matrix} D^{(k-1)}(i,j), \\ D^{(k-1)}(i,k) + D^{(k-1)}(k,j) \end{matrix} \right)$$

$O(n^3)$ time

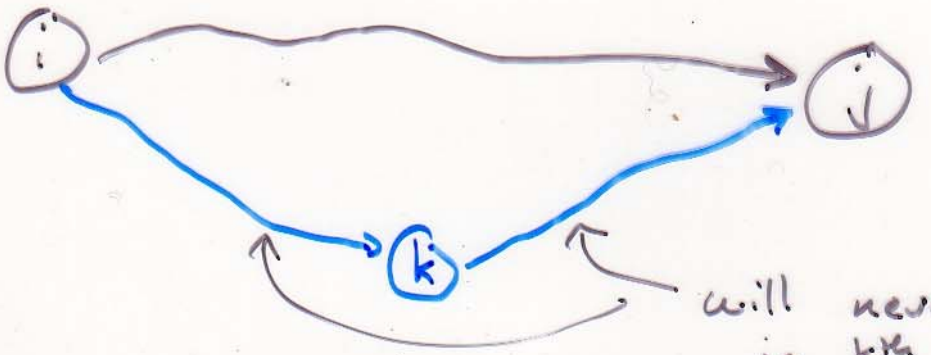
$O(n^3)$ space

$D^{(n)}$ is what we want.

to reduce space,

just keep one matrix

D and overwrite it



So only

need one copy of a

D matrix...