Thursday, March 31, 2022 10:51 AM

Shortest Paths (cont'd)

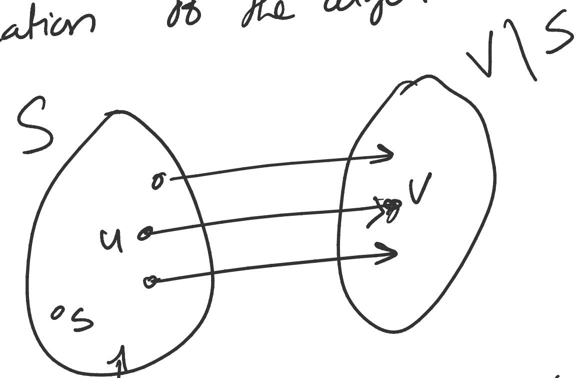
Given weighted directed graph G, $w(e) \geq 0$, $\forall e \in E$

Find $s \to t$ shortest path distance (mindist) $s, t \in V$

Dijkstra's Algo (1959): (SSSP: $mindist(s, v)$ $\forall v \in V$)

Idea: Find next nearest vertex to s

At any iteration of the algo.



Pick edge (u,v) s.t. $u \in S$, $v \in V \setminus S$ that minimizes $d[u] + w(u,v)$

$(u,v) = \underset{\substack{(u,v) \in E \\ u \in S, v \in V \setminus S}}{\text{argmin}} \left(d[u] + w(u,v) \right)$

$\forall u \in S$: $d[u] = mindist(s,u)$

Proved in last lec. → $\boxed{mindist(s,v) = d[v] = d[u] + w(u,v)}$

Update $S = S \cup \{v\}$.

☆ Efficient Implementation (using smarter data structure).

// $Q = V \setminus S$

// $\forall v \in Q$, maintain $key[v] = \min_{u \in S} d[u] + w(u,v)$

1. $Q = V$
2. $key[s] = 0$, $key[v] = \infty$, $\forall v \in V \setminus \{s\}$
3. while $Q \neq \emptyset$ {

   Pick $v \in Q$ with min key value.

3. while $Q \neq \emptyset$

→ 4. Pick $v \in Q$ with min key value.

5. $d[v] = key[v]$ // $mindist(s, v) = key[v]$

→ 6. Remove v from Q. (Delete-min op.)

7. for each out-neighbor y of $\underline{v}$.

8. if $y \in Q$ & $key[y] > d[v] + w(v, y)$

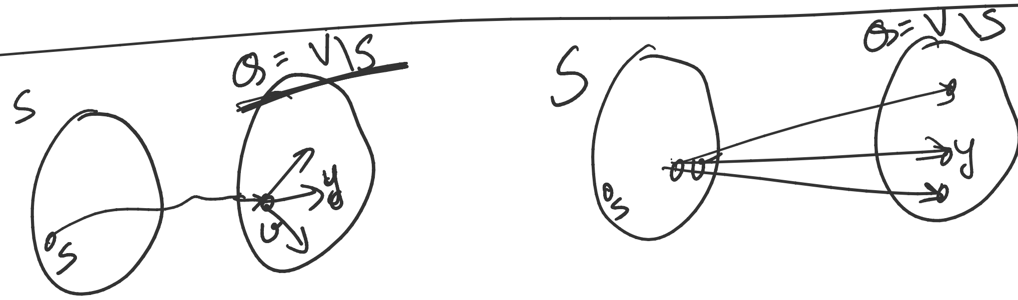
→ 9. then $key[y] = d[v] + w(v, y)$. (change-key op.) (decrease key op)

}

★ Running Time : (line 4 & line 6) $O(n)$ + (line 9) $\boxed{\sum_{v \in V} out\text{-}deg(v)}$ $= m$

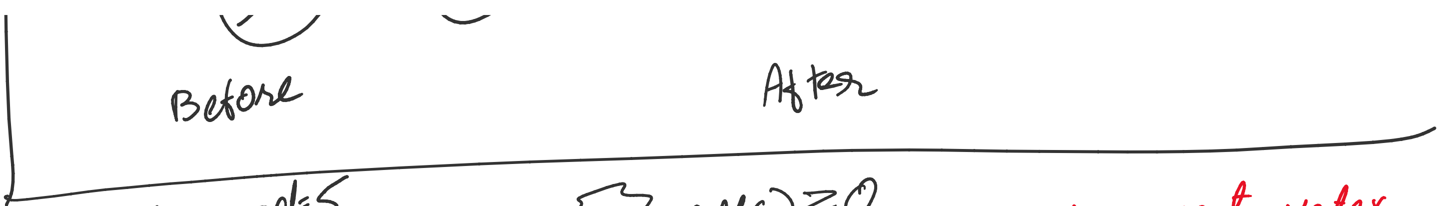
lines	No datastructure	Heap	Fibonacci Heap
4.	$O(n)$	$O(\log n)$	$O(\log n)$
6.	$O(1)$	$O(\log n)$	$O(\log n)$
9.	$O(1)$	$O(\log n)$	$O(1)$
Total:	$O(n)O(n) + O(1)O(m)$ $= O(n^2 + m)$ $= O(n^2)$	$O(n)\,O(\log n) + O(m)\,O(\log n)$ $= O(n\log n + m\log n)$ $= O((m+n)\log n)$	$O(n)\,O(\log n) + O(m)\,O(1)$ $= O(n \log n + m)$

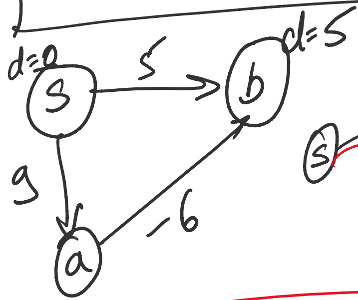
Aside:

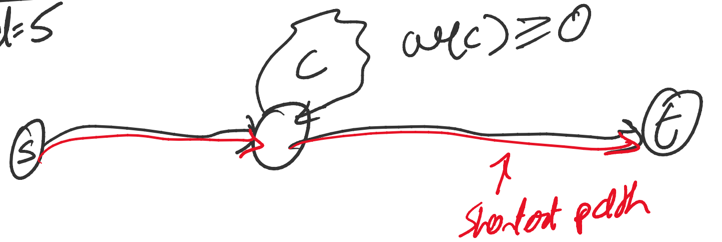


Before

After







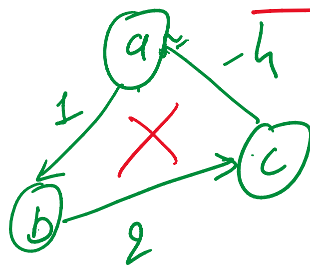
No repeat vertex on shortest path.

Bellman-Ford (1956):

- neg wts allowed
- solve SSSP.

$\boxed{\text{Assume no neg-wt cycles}}$

mindist $(a, b) = 1$
$= 0$
$= -1$
$= -2$
$\vdots$

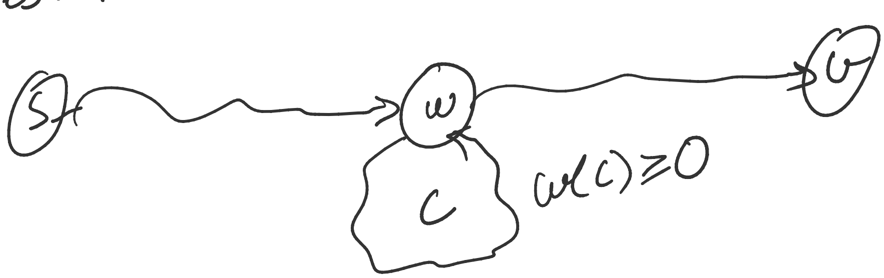


Idea :- DP!

★ Define subproblems : $v \in V$, $\ell = 0, 1, 2, \ldots, n-1$

$d(v, \ell)$ = mindist over all paths from s to v w/ #edges on the path $\leq \ell$.

Shortest path from $s \rightsquigarrow v$ then if never revisits a vertex ⇒ #edges on the path $\leq n-1$

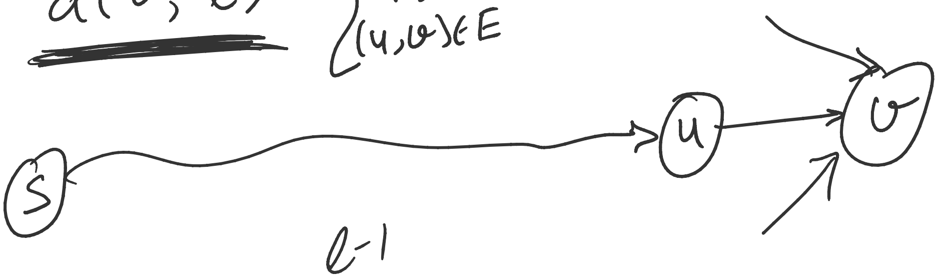


* Answer: $d(t, n-1)$

* Base case: $d(s, 0) = 0$; $d(v, 0) = \infty$, $\forall v \in V \setminus \{s\}$

* Recursive Formula:

$$d(v, \ell) = \min \begin{cases} d(v, \ell-1), \\ \min\limits_{u : (u,v) \in E} d(u, \ell-1) + w(u, v) \end{cases}$$



$\ell - 1$

* Evaluation Order: increasing order of ℓ.
$\ell = 0, 1, 2 \ldots$

* Running Time: $O\Big(n \cdot \sum\limits_{v \in V} \text{in-deg}(v)\Big)$

choices of ℓ

$= O(n \cdot m)$. Slower than Dijkstra's.

* Note: Can be used to detect -ve wt cycle.

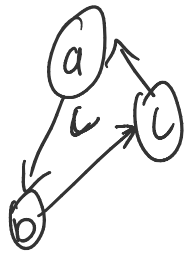
Claim: Assume all vertices are reachable from S.
Then there is no neg-wt cycle
$\iff \forall v \in V \quad d(v, n) = d(v, n-1)$

pf sketch: ($\Rightarrow$) Easy (exe).

($\Leftarrow$) $\forall (u,v) \in E$, $d(u, n-1) + w(u, v) \geq d(v, n-1)$

$\Rightarrow w(u, v) \geq d(v, n-1) - d(u, n-1)$

$\Rightarrow \quad w(u,v) \geq d(v,n-1) - d(u,n-1)$



$$
\begin{aligned}
& w(a,b) \geq d(\cancel{b,n-1}) - d(\cancel{a,n-1}) \\
+\ & w(b,c) \geq d(\cancel{c,n-1}) - d(\cancel{b,n-1}) \\
+\ & w(c,a) \geq d(\cancel{a,n-1}) - d(\cancel{c,n-1}) \\
\hline
& w(C) \geq 0 \qquad \blacksquare
\end{aligned}
$$

All Pairs Shortest Paths: (APSP)

Find shortest path distance betⁿ every pair of vertices.

* non-neg wts: Run Dijkstra starting at every vertex.

$$= O(n \cdot (n \lg n + m)) = O(n^2 \log n + mn) < O(n^3)$$

* Neg wts: Order vertices $1, 2, \dots, n$.

→ Method I: DP

Define subproblem: $i \in V,\ j \in V,\ \ell = 0, 1, \dots, n$

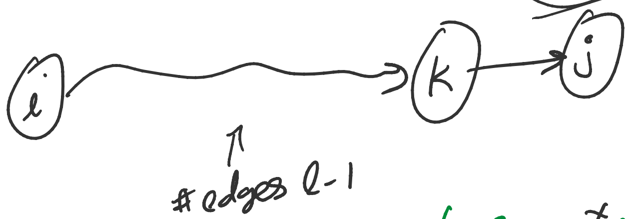
$d(i,j,\ell)$ = mindist over all paths from i to j w/ # edges $\leq \ell$.

Formula:

$$d(i,j,\ell) = \min \begin{cases} d(i,j,\ell-1) \\ \min\limits_{k:} d(i,k,\ell-1) + w(k,j) \end{cases}$$

... min over $O(n)$ many values.

$d(i, j, \ell) = \min \left\{ \min_{k : (k,j) \in E} d(i, k, \ldots \right.$ ← min over $O(n)$ many values.



Running-Time: $O(n^3 \cdot n) = O(n^4)$

↑ # subproblems

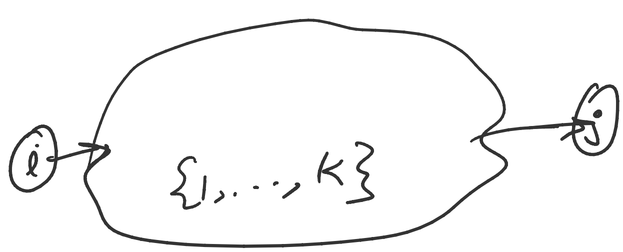
↓ improve

Aside: $O(n^3 \log n)$ by considering k at distance $\ell/2$ from i.

☆ Method 2: (Floyd-Warshall '62)

→ Define Subproblem:

$d(i, j, k)$: mindist over all paths from i to j s.t. all intermediate vertices are from $\{1, \ldots, k\}$



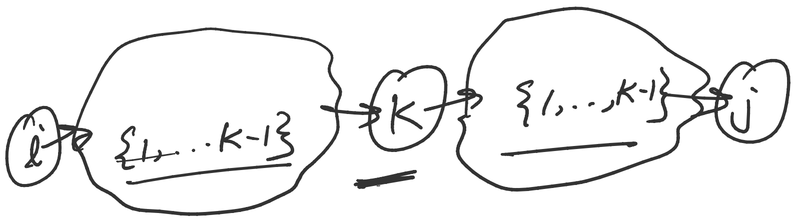
★ Answer: $d(i, j, n) \quad \forall i, j \in V$

★ Base case: exe.

★ Formula:

do not use vertex k — use k

$$d(i, j, k) = \min \left\{ d(i, j, k-1),\ d(i, k, k-1) + d(k, j, k-1) \right\}$$



★ Evaluation Order: increasing order of k.

★ Running Time: $O(n^3 \cdot O(1)) = O(n^3)$

subproblems.

How about $O(n^{2.999})$? OPEN

History:

$O(n^3 / \lg^{1/3} n)$ Fredman '76

$\vdots$

$O(n^3 / \log n)$ Chan '05

$\vdots$

$O(n^3 / c^{\sqrt{\log n}})$ Williams '14 (rand)
Chen-Williams '16 (det).