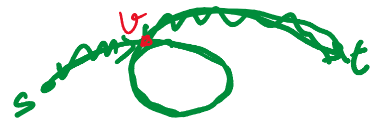
Tuesday, March 26, 2024 11:05 AM

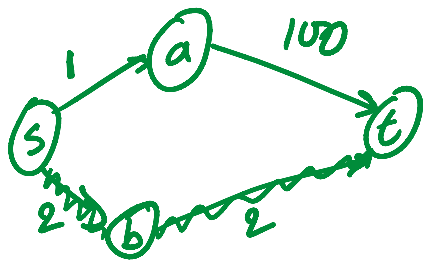
Shortest Paths:

Given a directed graph $G = (V, E)$, $w : E \to \mathbb{R}^+$, $s, t \in V$

Find a path from s to t that minimizes $\sum_{e \in P} w(e)$



e.g.



Never repeats any vertex / edge.

Special case: $w(e) = 1$, $\forall e \in E$ (unit capacity)

BFS(G, s). $\boxed{\text{level}(u) = s\text{-}u \text{ shortest path length.}}$

$O(m+n)$ time.

Single source shortest path (SSSP)

Find $\underline{s\text{-}u \text{ shortest path length}}$ $\forall u \in V$

mindist(s, u)

Special case 2: DAG (no cycles).

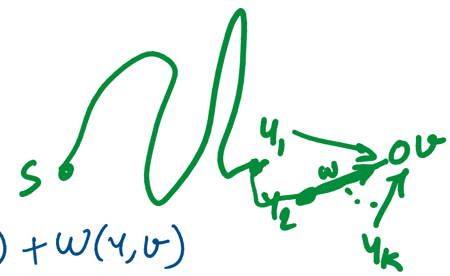
Dynamic Programming.

Subproblem: $\forall u \in V$

$d(u) = s\text{-}u$ shortest path length.

Ans : $d(t)$

Base case : $d(s) = 0$

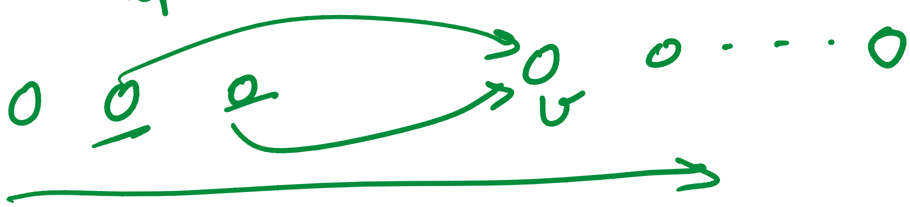


R.F. :

$$\underline{d(v)} = \min_{u : (u,v) \in E} d(u) + w(u,v)$$

Evaluation Order:

Topological Sort



Runtime: $\underline{\text{Topological Sort} - O(m+n)}$

DP $\left\{ \begin{array}{l} \#\text{ subproblems} = O(n) \\ \boxed{\begin{array}{l}\text{time subproblem} = O(|\text{in-neighbors}(v)|) \\ \quad d(v) \qquad\qquad = O(n)\end{array}} \end{array} \right. \Rightarrow O(n^2)$

- Better runtime analysis of DP:

$$\text{total time} : O\left(\sum_{v \in V} |\text{in-neighbors}(v)|\right)$$

$$= O(m) < O(n^2)$$

Final Runtime : $O(m+n)$

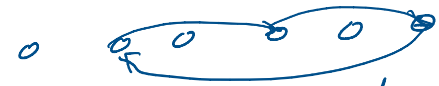
Remarks : (DAG)

- Topological Sort $\Longleftrightarrow$ DAG

Remarks: (DAG)

① Topological Sort $\iff$ DAG

($\Leftarrow$) Algo. above

($\Rightarrow$) If $\exists$ cycle then 

↳ Not allowed in a topological sort!

② works w/ $-ve$ weights as well.

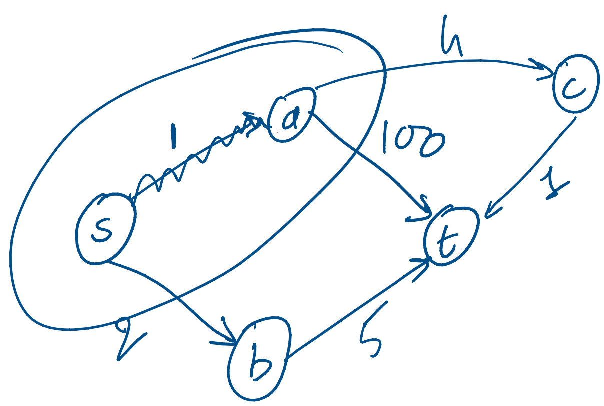
$w(e) = (-1)\ \forall e \in E \Rightarrow$ Finds "longest path" in DAGS!

General Case:

Dijkstra's Alg'm (1959):

SSSP: Find, $\forall v \in V$ $\text{mindist}(s, v)$

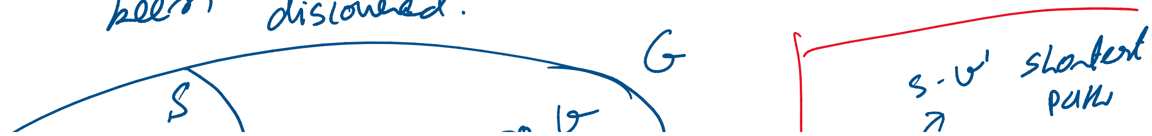
Smart Greedy!

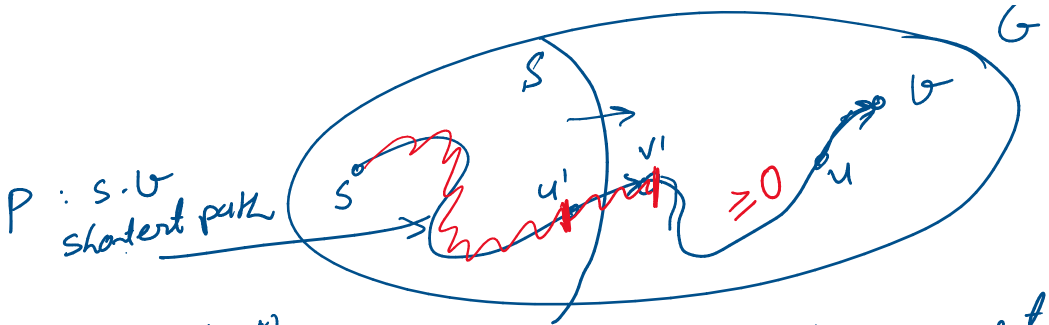


Goal: Compute shortest paths in increasing order of their length.

Observation(s):

S: All nodes for which shortest path length from s has been discovered.





$|S| = k$ then S : k first nearest vertices from s.

Does the next nearest vertex, say v, has "no" in-coming edge from set S? **NO!**

$$\text{mindist}(s,v) = \text{length}(P) \geq \text{length}(s\text{-}v' \text{ seg of } P)$$

$$\geq \text{mindist}(s, v')$$

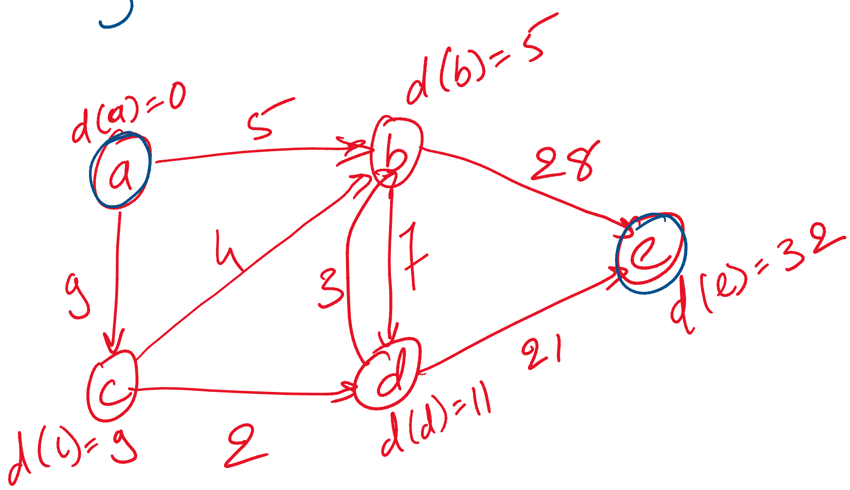
Then v' is next nearest. Contradiction!

High level Alg'm:

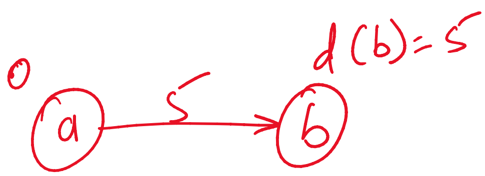
1. $S = \{s\}$, $d[s] = 0$
2. while $S \neq V$ do {
3. $O(m)$ time: pick $(u^*, v^*) = \arg\min_{\substack{(u,v) \in E \\ u \in S,\ v \notin S}} d[u] + w(u,v)$
4. set $d[v^*] = d[u^*] + w(u,v)$. Parent$(v^*) = u^*$
5. $S = S \cup \{v^*\}$

}

$O(mn)$ time

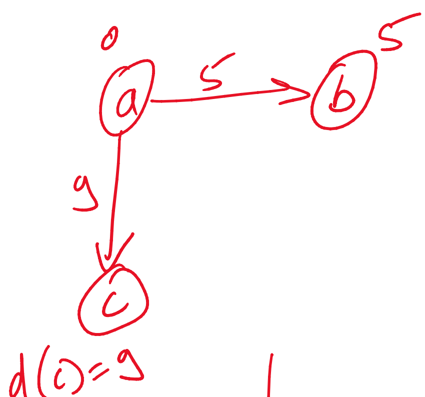


s = a
t = e.



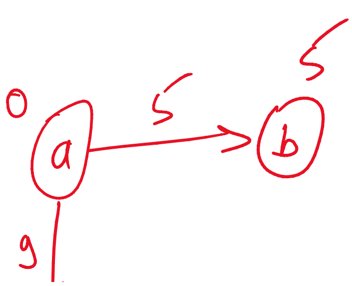
S' = {a}

(a,b) : d(a)+w(a,b) = 0+5
(a,c) : d(a)+w(a,c) = 0+9



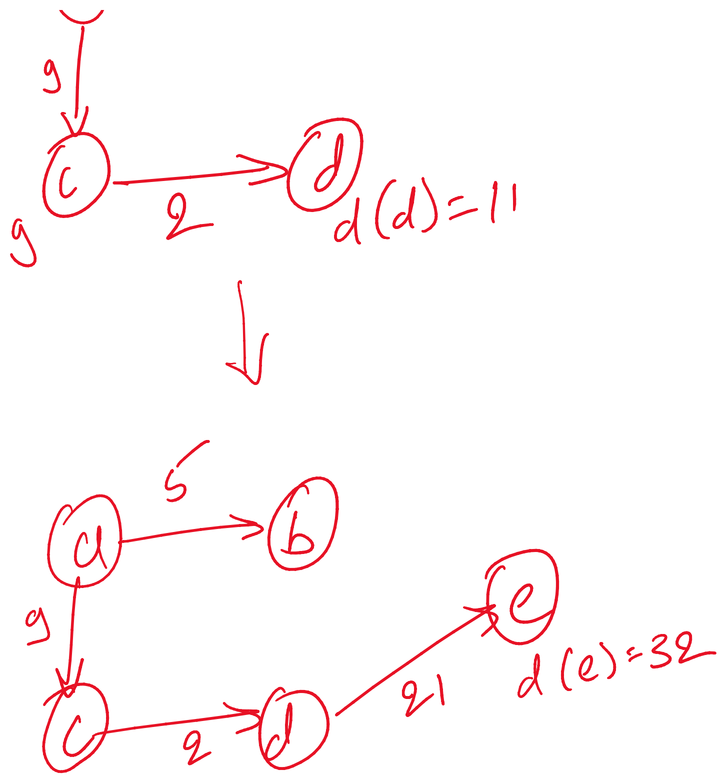
S = {a,b}

(a,c): 0+9
(b,e): 5+28
(b,d): 5+7



S = {a,b,c}

(b,e): 5+28
(b,d): 5+7
(c,d): 9+2



$S = \{a, b, c, d\}$

$(b, e) : 5 + 28 = 33$

$(d, e) : 11 + 21 = 32$

Proof of Correctness :

Let $d(v)$ denote s-v shortest path length

claim: suppose, $\forall u \in S$, $d(u)$ is correctly computed.

& let $(u^*, v^*) = \underset{\substack{(u,v) \in E \\ u \in S,\ v \notin S}}{\arg\min}\ d(u) + w(u, v)$

Then $d(v^*) = d(u^*) + w(u^*, v^*)$

s-v^* shortest path dist.

$(\leq)$

$(\geq)$

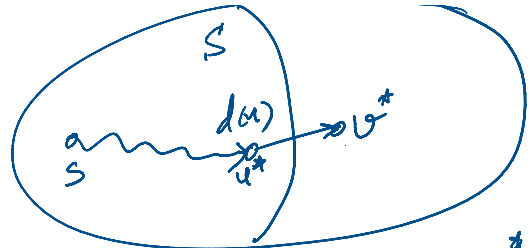
$\geq d(u) +$

Pf: Let P^* be the s-v^* Shortest path

$(\leq)$ T.P.T. $d(v^*) \leq d(u^*) + w(u^*, v^*)$

path P : $s \rightsquigarrow u^* \to v^*$

$(\;\;)$

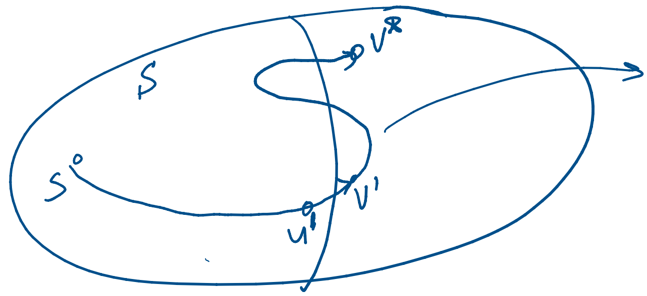


path $P : s \rightsquigarrow u^* \to v$

$d(v^*) = \text{length}(p^*) \le \text{length}(P) = s\text{-}u^* \text{ shortest path length} + w(u^*, v^*)$

$= d(u^*) + w(u^*, v^*)$.

$(\ge)$. TPT $\quad d(v^*) \ge d(u^*) + w(u^*, v^*)$



shortest s-v^* path, say p^*

$$
\begin{aligned}
d(v^*) = \text{length}(p^*) &= \text{length}(\underbrace{s\text{-}u' \text{ seg of } p^*}_{\ge d(u')}) \\
&\quad + w(u', v') \\
&\quad + \text{length}(\underbrace{v'\text{-}v^* \text{ seg of } p^*}_{\ge 0}) \\
&\ge d(u') + w(u', v') + 0 \\
&\ge d(u^*) + w(u^*, v^*) \\
&\quad (\because \text{choice of } (u^*, v^*))
\end{aligned}
$$

Implementation via Priority queue.

1. $Q = V$ // Q: nodes whose shortest path is yet to be computed.
2. $key(v) = \text{infinity}$, $\forall v \ne s$; $key(s) = 0$.
3. While $Q \ne \emptyset$ do {

$\quad$, Q with minimum key value.

3. While $Q \neq \phi$

4. Pick $u \in Q$ with minimum key.value.

5. Remove u. $d[u] = key[u]$

6. for each $(u,v) \in E$ do

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

8. then $key[v] = d[u] + w(u,v)$. $Parent[v] = u$.

}

Runtime : No data structure.

line 4: $O(n)$
line 5: $O(1)$
line 7, 8: $O(1)$
lines 6-8: $O(|Adj(u)|)$
Total time: $O\left(n.n + \sum_{u \in V} |Adj(u)|\right) = O(n^2+m)$
$= O(n^2)$

Runtime : Priority queue/heap.

line 4: $O(\log n)$
line 5: $O(\log n)$
line 8: $O(\log n)$

Total time: $O\left(n.\log n + \left(\sum_{u \in V} |Adj(u)|\right) \cdot \log n\right)$

$= O(n.\log n + m.\log n)$

$$= O(n \cdot \log n + m \cdot \log n)$$

$$= O((n+m) \cdot \log n).$$