COL 702 Lecture 22 Oct 25

Bellman Ford algorithm for SSSP

$\delta(u,v)$ is the shortest path distance between vertices u & v

$u \rightsquigarrow v$ is the notation for a path from u to v

$\delta(s,v)$ is the final output of the algorithm for all v

$D(v)$: is an upper bound on $\delta(s,v)$

Initially $D(v) = \infty$, $D(s) = 0 = \delta(s,s)$

Finally $D(v) = \delta(s,v)$

Relax (u,v) $(u,v) \in E$

If $D(v) > D(u) + w(u,v)$
then $D(v) \leftarrow D(u) + w(u,v)$

Pred$(v) \leftarrow u$
for path construction

Repeat $|V|-1$ times
for all edges $(u,v) \in E$
Relax (u,v)

Output the final $D(v)$ for all $v \in V$

<u>Claim</u> At the end of $|V|-1$ iterations

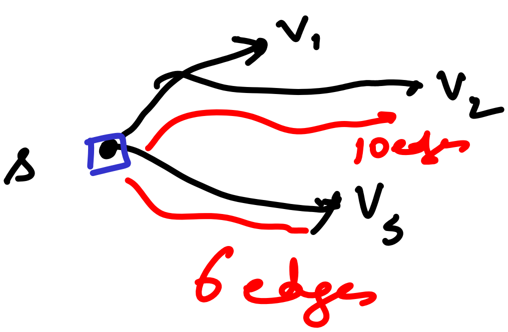
$$D(v) = \delta(s,v)$$

Note: Running time for BF is $O(|V| \cdot |E|)$

\# iteration

cost of relaxation for all edges

Proof by induction on the # of edges ℓ in the shortest path from s to v



For $\ell = 0$

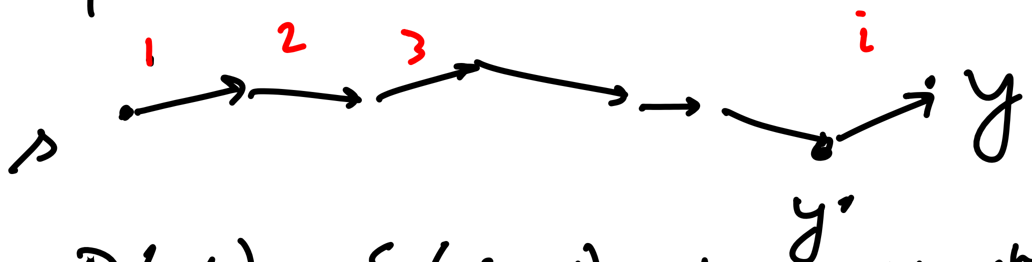
$D(s) = 0 = \delta(s,s)$

So correctly initialised

Suppose it is correct for $< i$ iterations, i.e. all vertices x whose short path from s consists of $< i$ edges have $\delta(s,x) = D(x)$ after $i-1$ iterations

During the i^{th} iteration all edges will undergo relax operation

Consider a vertex y s.t the shortest path has i edges. . . .



$D(y') = \delta(s,y')$ before the i^{th} iteration

So when we relax edge (y', y)

$$\delta(s,y) \leq D(y) = D(y') + w(y',y)$$
$$= \delta(s,y') + w(y',y)$$
$$= \delta(s,y)$$

$\Rightarrow$ All vertices have their correct distances

<u>Remark</u>: If there is a -ve cycle in the graph, then BF algorithm can detect it if we let it run for $|V|$ iterations

If there is any change in the D value of a vertex then there must be a negative cycle (shortest cycle must have length $\leq n$)

For non-negative weights we can do better by using Dijkstra

In Dijkstra's algorithm, the relax operation are carefully scheduled so that all edges is relaxed at most once

Partition V

$$\begin{tikzcd} & V \arrow[dl] \arrow[dr] & \\ S & & U \end{tikzcd}$$

S: the set of vertices v : $D(v) = \delta(s,v)$

U: $D(v) \geq \delta(s,v)$ the correct distances are yet to be determined

Initially $S = \{s\}$ $\quad D(s) = \delta(s,s) = 0$

Relax all edges (s,v)

Repeat until $U = \phi$

iterations $V-1$

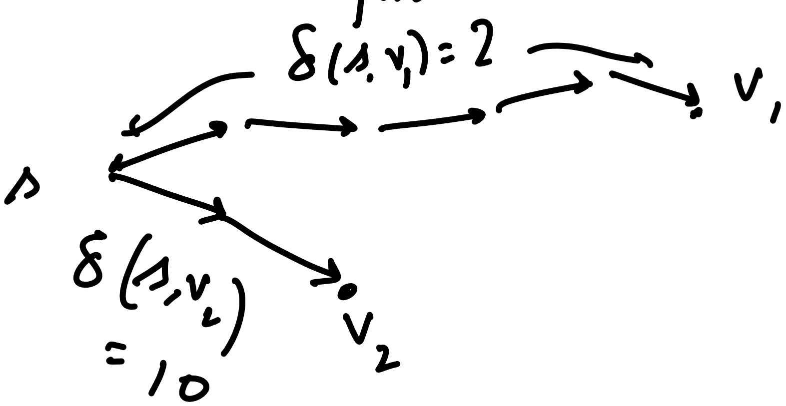
1. Choose the vertex with smallest label in U, say x $\quad O(\log V)$ priority queue
2. Relax all outgoing edges from x $\quad$ outdegree (x)
   Move x from U to S

Total time: $\sum_{v} \log|V| + \text{outdegree}(v)$

$\leq O(|V| \log V + |E|)$

Why does Dijkstra algorithm output the correct distances?

Claim: The algorithm discovers (moved fm U to S) the shortest path distances in order of their actual distance fm s

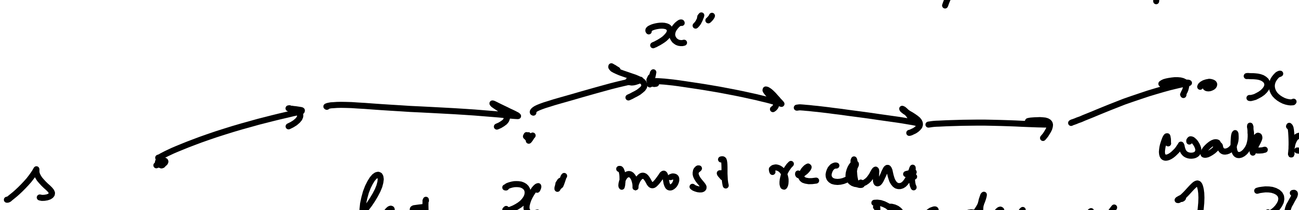


v_1 will move into S before v_2 in Dijkstra (and the reverse for B.F.)

When a vertex x has the smallest $D(x)$ in U then $D(x) = \delta(s,x)$

Proof by contradiction: Suppose not, i.e. $D(x) > \delta(s,x)$

Consider the shortest path from $s \leadsto x$



walk backwards from x

let x' most recent predecessor of x in the path that has $D(x') = \delta(s, x')$, and $x' \in S$

Then (x', x'') must have been relaxed when x' moved to S

$$D(x'') \overset{?}{:} D(x)$$

$<$ because of non-negative weights

So x'' should have been picked