

ECS 122A Lecture 9

Shortest Path Algorithms

Jasper Lee

Setting: Given ^{directed} weighted graph (each edge e has weight w_e)
compute shortest paths b/w vertices.

Variations:

- Single source: given vertex u ,
compute shortest path from u to all other vertices v
may be negative

- All pairs: for all pairs u, v ,
compute shortest u to v path.

Single - source shortest path

Goal: Derive the Bellman - Ford alg.

Different intuition from basic "either or"

DP table?

D for distance $\rightarrow D[v] \doteq$ shortest path len from u to v
Just rephrased original problem...

Q: DP needs subproblems. What are the subproblems here?

Any path is a seq of hops through graph.
⇒ Limit the # of hops for subproblems

DP table

$D[v, k] \doteq$ shortest path from u to v
using $\leq k$ hops.

Equivalently, insert self loops of weight 0 into graph

$D[v, k] \doteq$ shortest path from u to v
using exactly k hops.

Recurrence

$$D[v, 0] \doteq \begin{cases} \infty & \text{if } v \neq u \\ 0 & \text{if } v = u \end{cases}$$

$$D[v, k] = \min \left(D[v, k-1] + \overset{\nearrow 0}{w_{v \rightarrow v}}, \min_{x \neq v} D[x, k-1] + w_{x \rightarrow v} \right)$$

How large can k be? $n-1$

non-existent edges have weight ∞

Implementation:

Initialize $D \leftarrow \infty$, $D[u, 0] = 0$.

for $k = 1$ to $n-1$

for each edge $a \rightarrow v$

$$D[v, k] \leftarrow \min(D[v, k], w_{a \rightarrow v} + D[a, k-1])$$

$O(|V||E|)$
time

only
enumerate

edges instead of all vertex pairs (v, x)

(without constraint on #hops)

Issue: Shortest path^v undefined if there is negative cycle reachable from u ...

Claim: There is a negative cycle reachable from u if and only if

$$\exists v \text{ s.t. } D[v, n] < D[v, n-1].$$

Proof

$\exists v \Rightarrow \text{neg cycle}$

Let $u \rightarrow a_1 \rightarrow a_2 \rightarrow \dots \rightarrow a_{n-1} \rightarrow v$ be a path of min weight for $D[v, n]$. $\hat{= a_n$

Since there are only n vertices, by pigeonhole, $\exists a_i = a_j$ for $i < j$

Now, the $n-1$ hop path pad to $n-1$ hops
 $u \rightarrow a_1 \rightarrow \dots \rightarrow a_i \rightarrow a_{j+1} \rightarrow v (\rightarrow \dots \rightarrow v)$

has weight $\geq D[v, n-1] > D[v, n]$.

So $a_i \rightarrow \dots \rightarrow a_j = a_i$ is a negative cycle.

Neg cycle $\Rightarrow \exists v$

($\forall v, D[v, n] \geq D[v, n-1]$) \Rightarrow No neg cycle

Consider any cycle $a_1 \rightarrow \dots \rightarrow a_i$

$$D[a_{i+1}, n-1] \leq D[a_{i+1}, n] \leq D[a_i, n-1] + w_{a_i \rightarrow a_{i+1}}$$

Summing over cycle $\Rightarrow \sum w_{a_i \rightarrow a_{i+1}} \geq 0$.

Alg: Do one more inner loop to check
if there are any updates.
If yes \Rightarrow neg cycle
no \Rightarrow no neg cycle.

Optimizing implementation:

- Only need to keep index $k-1$ to compute index k .
- In fact, can even ignore index and keep updating

Initialize $D[v] \leftarrow \infty$ for all $v \in V$, $D[u] \leftarrow 0$

for $k = 1$ to $n-1$

for each edge $u \rightarrow v$

$D[v] \leftarrow \min(D[v], w_{u \rightarrow v} + D[u])$.

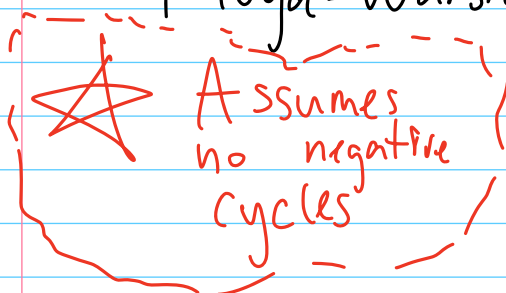
Do another
inner loop
to check
neg cycle \rightarrow

Correctness proof slightly messier.

All-pairs shortest path

If we run Bellman-Ford on all source vertices,
we can get all-pairs shortest path in $O(|V|^2|E|)$ time.

Floyd-Warshall improves to $O(|V|^3)$ time.



Assumes
no negative
cycles

$|E|$ can be
as big as $O(|V|^2)$

$|E| \geq |V|$ so no
worse than repeating
Bellman-Ford.

Subproblem structure not necessarily obv

$D[i, j, k] \doteq$ shortest path from i to j
where intermediate vertices are
all in $\{1 \dots k\}$.

Recurrence

$$D[i, j, k] = \min(D[i, j, k-1], D[i, k, k-1] + D[k, j, k-1])$$

$$D[i, j, 0] = w_{i \rightarrow j} \text{ (which may be } \infty \text{)}$$

Correctness:

Either use vertex k or not.

If using, must reach k using $\{1 \dots k-1\}$
and then reach j using $\{1 \dots k-1\}$.

for $k = 1$ to n
 for $i = 1$ to n
 for $j = 1$ to n

$$D[i, j, k] \leftarrow \min(D[i, j, k-1], D[i, k, k-1] + D[k, j, k-1])$$

Same optimization as before

$$D[i, j] \leftarrow \min(D[i, j], D[i, k] + D[k, j])$$