Shortest Paths: Lecture #4

**APSP**: all pairs shortest paths

**SSSP**: single source shortest path

For **APSP**: we consider **usually directed graphs**. These may be **negative edge weights** and we often assume weights are real numbers on which we can do addition, subtraction, but not multiplication, **unless otherwise stated**.

**SSSP**: We may assume the graph is **directed** and have **non-negative** edge weights.

**Algorithm**: Dijkstra's Algorithm

**SSSP, directed, non-negative**.

Maintain "estimates" of shortest distance from **S**. Initially, \( d(S) = 0 \), for all other nodes, \( d(x) = \infty \) if \( x \neq S \).

Then pick a vertex with smallest estimate, fix it, and relax all its out-edges: \( d(u) \rightleftharpoons d(u) + \min \{ d(u) + e(u \rightarrow v) \} \).

**Complexity**: \( O(n^2) \) dumb, \( O(m \log n) \) binary heap, \( O(m + n \log n) \) Fibonacci Heap.

**Bellman Ford (Moore)**

For \( i = 1 \) to \( n-1 \)

for each edge \((u, \ v)\)

\[ d(\ v) \leftarrow \min \{ d(\ v), d(u) + e(u \rightarrow v) \} \]

**Theorem**: Computes **SSSP** correctly if no negative cycle. In \( O(mn) \) time.

**Fact**: **Bellman Ford** works even if some edge lengths change \( \leq \) a negative cycle.
Better: \( [\text{Goldberg}] \) if edge weights \( \leq C \) then (even with negative cycles) \( O(\sqrt{m \ln C}) \) time

\[ \text{APSP: non-negative weights: } n \text{Dijkstra's} \quad O(mn + n^2 \lg n). \]
\[ O(mn^2) \quad \leftrightarrow \quad \text{bad.} \]

Here's a single fix: \([\text{Johnson's algorithm}]\).

Suppose \( I \) a vertex \( s \in V \) s.t. can reach every vertex \( \text{from} \ s \). Let \( D_x \) = shortest path distance of \( s \to x \) (using say Bellman-Ford). \( \leftarrow \) no neg cycles of course.

\[ \text{Fact Def: Reduced cost of an edge } xy \]
\[ \Rightarrow e'_x = D_x + l_{xy} - D_y. \quad (\Rightarrow \text{by fact that shortest path set} \]
\[ \quad \Rightarrow D_y \leq D_x + l_{xy} \quad (\forall x, y \in E) \]

Simple fact: \( \forall a, b \) paths \( P \)

\[ \ell' = \text{length}(P) + D_a - D_b. \]

\[ \Rightarrow \text{shortest } ab \text{ paths the same for both length functions} \]
\[ \Rightarrow \text{can compute shortest paths in our non-neg wt graph.} \]
\[ \Rightarrow \text{APSP in time } 1 \times \text{BF} + (n-1) \text{Dijkstra} = O(mn + n^2 \lg n). \quad \left[\text{Pettie} O(mn + n^2 \lg n)\right] \]

In fact could have used any function \( \Phi_x \) \( \forall x \in V \) s.t.

\[ \Phi_x + e_{xy} - \Phi_y \geq 0. \quad \leftarrow \text{this is called a "feasible potential"} \]
\[ \text{aka "valid labeling scheme"}. \]

\[ \text{If } l_{xy} > 0 \text{ then } \Phi = 0 \text{ is feasible potential "Adding a constant local pic does not change feasibility"} \]

\[ \text{Moreover suppose we set } \Phi_s = 0 \text{ for some node } s \in V \]

\[ \text{then } \Phi_x \text{ is an upper bound on distance from } s \to x. \]
\[ \Rightarrow \text{feasible potentials are underestimates of } s-x \text{ distance.} \]

so suppose we say \[ \max \sum_{x} \bar{f}_x \]
\[ s.t. \bar{f}_x = 0, \]
\[ \bar{g}_y \leq \bar{f}_x + d_{xy} \quad \forall x,y \in E \]

this is an LP.

and it is the dual of the standard shortest path LP. (Exercise).

---

**APSP in earnest now:** \[ \text{init } d_{ij} = \bar{g}_i \text{ if } i \neq j, \text{ 0 otherwise} \]

Floyd-Warshall:

\[ \sum_{k=1}^{n} \]
\[ \forall i = 1 \ldots n \]
\[ \forall j = 1 \ldots n \]

\[ d_{ij}^{(k)} = \min \{ d_{ij}, d_{ik} + d_{kj} \} \]

\[ O(n^3) \text{ time.} \]

---

Could use "matrix mult".

Define the min-sum product:

\[ (A \cdot B)_{ij} = \min_k \{ A_{ik} + B_{kj} \} \]

Then \[ A \cdot A = \text{shortest path using 2 hops.} \]

\[ A^n = A \cdot A \cdot A \ldots \cdot A \]

This matrix product is associative so can repeatedly square; set \( A^n \) in long mults.

\[ \Rightarrow \text{APSP}(n) \leq O(\log n \times \text{Min-Sum Prod}(n)) . \]

---

2) How fast can we do \( M^P? \)

\[ \frac{\text{[Maybe]}}{[\text{HW}]} \]

3) Can we get \( \text{APSP}(n) = \Theta(\text{Min-Sum Prod}(n)) \) \[ \Rightarrow \text{[Yes!]} \]

\[ \Rightarrow \text{bij open question.} \]

The problem is that \( (\min, +) \) is not a ring so cannot use fast mat-mult.

It is a semiring \[ \Rightarrow \text{does not have inverses under min.} \]

so currently, \[ M^P = O(n^{3-\varepsilon}) \text{ for } \varepsilon > 0 \text{ is open.} \]

But next lecture we'll see some progress.
A different tack: what if edges have small lengths, can we do something?


Directed: $G^{0.58}$ [-Zwick, AGM, Takacska, et al.]

Today: Seidel's Algorithm for undirected, unweighted graphs in time $O(n^{10/7+\epsilon})$.

[Recall: we don't know how to do minimum product even in this case]

[use undirected crucially!]

Suppose $A$ is the adjacency matrix $A_{ij}=1 \iff ij$ is an edge.

$B$ is the $2,3$-hop matrix: $B_{32} = 1$ if $A_{ij} = 1$ or $(A_{ik} A_{kj})$ is a path.

*Note*: $B = A + A^2$ (Boolean matrix mult).

Let $D$ be distances in $B$.

Let $d$ be distances in $A$.

\[
\begin{array}{cccccc}
& 0 & 1 & 2 & 3 & 4 & 5 \\
D_{(5,0)} & 1 & 2 & 3 & 2 & 2 & 1 \\
D_{(5,1)} & 1 & 1 & 2 & 2 & 3 & 1 \\
\end{array}
\]

**Fact 1**: $D(x, y) = \left\lfloor \frac{d(x, y)}{2} \right\rfloor$.

**Pf:** $d(x, y) \leq 2 D(x, y) \Rightarrow D(x, y) \geq \left\lfloor \frac{d(x, y)}{2} \right\rfloor$. And $3$-approximation $\left\lfloor \frac{d(x, y)}{2} \right\rfloor \in B$.

Good: so having computed $D$ recursively, need to figure out whether $d(x, y) = 2 D(x, y) \text{ or } 2 D(x, y) - 1$.

How do this fast?
Fact 2: if $d_{xy} = 2D_{xy}$ then $D_{xz} \geq D_{xy}$ \forall \text{ neighbourhoods in } \mathbb{A}.

**Proof:**

if not, then $D_{xz} < D_{xy}$

\[ d_{xz} \leq 2D_{xz} \leq 2D_{xy} \leq 2D_{xy} - 2 \]

\[ d_{xy} \leq d_{xz} + 1 \leq 2D_{xy} - 1. \]

contradicts $d_{xy} = 2D_{xy}$.

\[ \Rightarrow \text{ for all } x,y, D_{xz} \geq D_{xy}. \Rightarrow \sum_{2 \in \text{nbry}} D_{xz} \geq D_{xy}. \text{deg}(y). \]

Fact 3: if $d_{xy} = 2D_{xy} - 1$

then $D_{xz} \leq D_{xy}$ \forall \text{ nbhds } y \in \mathbb{A}$

and $3 \leq x \Rightarrow D_{xz} \leq D_{xy}$.

**Proof:**

\[ d_{xz} \leq 2D_{xy} - 1 + 1 \Rightarrow 2D_{xy} \Rightarrow \frac{D_{xz}}{2} = \frac{D_{xy}}{2} = D_{xy}. \]

also look at the shortest path $x,y$ and $z$ be predecessor.

then $d_{xz} = 2D_{xy} - 2 \Rightarrow D_{xz} = \left\lceil \frac{2D_{xy} - 2}{2} \right\rceil = D_{xy} - 1.

\[ \Rightarrow \text{ for } x,y: d_{xy} = \begin{cases} 2D_{xy} \text{ if } \sum_{2 \in N(y)} D_{xz} \geq D_{xy}. \text{deg}(y) \\ 2D_{xy} - 1 \text{ if } \sum_{2 \in N(y)} D_{xz} < D_{xy}. \text{deg}(y) \end{cases} \]

\[ \text{(D:A)} \]

\[ \text{D}_{xy} \]

\[ \text{shortest path matrix} \]

\[ \text{can compute } \frac{D_{xy}}{2} \text{ in } O(n^2) \text{ time}. \]
\textbf{Algorithm: Seidel} \quad (A^t + I) \\
\text{if } A' = I \text{ then return } (I - I).
\text{else}
\quad B = A^2 \leftarrow \text{Boolean Matrix Mult.} \quad (\text{OR, OR, AND})
\quad D \leftarrow \text{Seidel} (B).
\quad d_{ij} \leftarrow 2 d_{ij} - 1 \left( \text{DA}_{ij} < d_{ij} \cdot \deg(j) \right) \quad \text{[regular matrix mul]}
\quad \text{return } (d)
\}

\textbf{Runtime: } \mathcal{O}(n^3 \cdot \log n).

\textbf{Q: Can we get the paths?} \quad \text{related: doesn't it possibly take } n^3 \text{ time to untraverse paths down?}

\text{Want a "successor" matrix } S: \quad S_{ij} = k \quad \text{if } \quad k \text{ is the } ij \text{ shortest-path.}
\text{Can represent in } \mathcal{O}(n^2 \cdot \log n) \text{ bits. So no information theory lower bound.}

\textbf{How to compute successor matrix?}

\begin{align*}
\text{Product} & \text{ Boolean Matrix witness Problem: given } A, A' \text{ boolean matrices} \\
\text{out: } W_{ij} = & k \quad \text{if } A_{ik}, A'_{kj} = 1 \\
& 0 \text{ otherwise}
\end{align*}

\textbf{Q1: How to solve this problem?} \quad [\text{HW2}] \quad \text{in } \mathcal{O}(n^2 \text{ polylog } n) \text{ time w.h.p.}

\textbf{Q2: And if we can solve } \mathcal{BPMW} ?

\textbf{Thm:} Can solve successor problem with constant \# of BPMW calls, + \mathcal{O}(n^3) \text{ time}

\textbf{Proof:} next page.
Fix distance $d^*$. Compute $d_{ij}$ - distances in $G$
Sps $ij$ are distance $d^*$. Want to find $k$ st.

(1) $A_{ik} = 1$ and
(2) $d_{kj} = d^*-1$.

So define $A'_{ij} = \begin{cases} 1 & i\neq j \text{ and } d_{kj} = d^*-1. \\ 0 & \text{otherwise} \end{cases}$

and find MPWitness for $A, A'$.

will give successors for all pairs at distance $d^*$.

OK: Naively need to do $n^2$ of these calls, one for each $d^* \in [1..n-1]$.

But! Consider things modulo 3. Since unweighted, distances

so they don't interfere with each other.

$A'_{ik} = 1$ iif $ik \in E$

$A'_{ij} = 1$ iif $(d_{kj}^{(c)}) \equiv c \mod 3$ for $c \in \{0, 1, 3\}$.

find the MPWitnesses. Give the answer!

[i.e. for $ij$, look up the witness in matrix $W(d_{ij} \mod 3)$.]