APSP and Min-Sum Products

Recall Matrix Multiplication:

$$(AB)_{ij} = \sum_{k=0}^{n} (A_{ik} \cdot B_{kj})$$

Consider Min-Sum Product:

$$(A \odot B)_{ij} = \min_k (A_{ik} + B_{kj})$$

$\odot$ matrix mult. over semiring $(\mathbb{R}, \min, +)$

Let D be the 1-hop distance matrix $D_{ij} = \begin{cases} w_{ij} & (i,j) \in E \\ \infty & (i,j) \notin E, i \neq j \\ 0 & i = j \end{cases}$

Claim: $\underbrace{D \odot D \odot \cdots \odot D}_{k \text{ times}}$ is the k-hop distance matrix:

$D^{\odot k}(i,j)$ = shortest path with $\leq k$ edges

Proof by induction on k.

Remark: similar proof to Floyd-Warshall.

How to compute $D^{\odot n-1}$? $n-1$ iterations, n^3 each $\rightarrow n^4$

repeated square: $n^3 \log n$

APSP Conjecture: There is no APSP algo. in time $n^{2.99}$ $(n^{3-\epsilon})$

Current best: $n^3 / 2^{\Theta(\sqrt{\log n})}$ [Williams '14]

Seidel's Algorithm: $n^\omega \log n$ time on unweighted, undirected graphs

Seidel's Algorithm: $n^\omega \log n$ time on unweighted, undirected graphs

Def: Adjacency matrix $A_{ij} = 1$ if $ij \in E$
0 if $ij \notin E$

Consider graph G^2, the square of G,
edge in $G^2 \iff$ 2-hop path in G.
Adj. matrix of G^2: $A_{G^2} = A_G * A_G + A_G$.

Claim: If d_{xy}, D_{xy} are shortest path distances between x, y in G, G^2,
then $D_{xy} = \lceil \frac{d_{xy}}{2} \rceil$.

Proof: $D_{xy} \le \lceil \frac{d_{xy}}{2} \rceil$: take path in G, show path in G^2
$D_{xy} \ge \lceil \frac{d_{xy}}{2} \rceil$: if not, get path length $\le 2D_{xy} < d_{xy}$, $\not\in$

So, $d_{uv} \in \{2D_{uv}, 2D_{uv} - 1\}$

Claim: If $d_{uv} = 2D_{uv}$, then $D_{uw} \ge D_{uv}$ $\forall$ neighbors w of v.
Proof: If not, $\exists w$: $D_{uw} \le D_{uv} - 1$

$$2D_{uw} \le 2D_{uv} - 2$$

take path with edge (w,v): $\le 2D_{uv} - 1 < 2D_{uv} = d_{uv}$ $\not\in$
u-v in G

Claim: If $d_{uv} = 2D_{uv} - 1$, then $D_{uw} \le D_{uv}$ $\forall$ neighbors w of v, and
$\exists$ neighbor z: $D_{uz} < D_{uv}$.

Proof: u-v path in $G + (v,w)$: $d_{uw} \le d_{uv} + 1 = (2D_{uv} - 1) + 1 = 2D_{uv}$

$$\Rightarrow D_{uw} = \lceil \frac{d_{uw}}{2} \rceil \le D_{uv}$$

take shortest path $u \xrightarrow{z} v$

take shortest path $u \xrightarrow{z} v$

$d_{uz} = d_{uv} - 1 = 2D_{uv} - 2$, so $D_{uz} = \left\lceil \frac{2D_{uv}-2}{2} \right\rceil = D_{uv} - 1$.

Hence, $d_{uv} = 2D_{uv} \iff$ average $D_{uw} \geq D_{uv}$. $w \sim v$

How to compute ? $\Leftrightarrow$

$$(DA)_{uv} = \sum_w D_{uw} A_{wv} = \sum_{w \sim v} D_{uw}$$

Def: an algorithm is subcubic-time if it runs in $O(n^{3-\varepsilon})$ time for some constant $\varepsilon > 0$.

APSP Conjecture: There is no subcubic-time algo for APSP on weighted digraphs with integer weights in $\pm \text{poly}(n)$

Current fastest APSP: $n^3 / 2^{\Omega(\sqrt{\log n})}$ time.

Fine-Grained Reductions: Either all of the following problems admit subcubic-time algos, or none of them do:

- APSP
- APSP on non-negative weights (Johnson's edge re-weighting)
- Negative-weight triangle
- Minimum-weight cycle on graph with non-negative weights
- Min-Sum Product

Last lecture: APSP in $O(\log n)$ Min-Sum Products.
Subcubic MSP $\Rightarrow$ subcubic APSP.

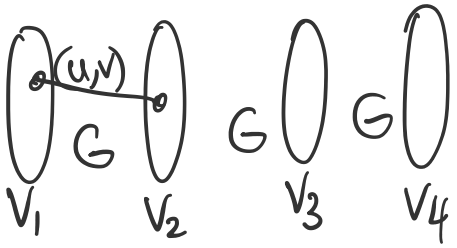
Claim: Subcubic APSP $\Rightarrow$ Subcubic MSP.

Claim: Subcubic APSP $\Rightarrow$ Subcubic MSP.

Proof: Given MSP instance, construct 

, call APSP

Claim: Subcubic APSP $\Rightarrow$ Subcubic NWT.

Proof: Given NWT instance, construct 

, call APSP

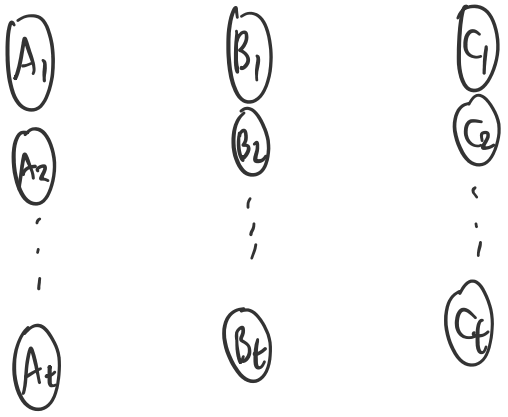
Claim: Subcubic NWT $\Rightarrow$ Subcubic APSP.

Proof: Subcubic APNWT $\Rightarrow$ Subcubic APSP: Parallel Binary search

All-pairs Neg-Weight Triangle:
Given $A, B, C \subseteq V$, for all $u \in A$, $v \in B$,
find a vertex $w \in C$ s.t. (u,v,w) is NWT or
determine none exist.

Subcubic NWT $\Rightarrow$ Subcubic APNWT:

Reduction: $t = n^{2/3}$



Algorithm:

- Initialize outputs $D(u,v) = \phi$
- For $i = 1..t$, $j = 1..t$, $k = 1...t$
  - while NWT($G[A_i \cup B_j \cup C_k]$) finds a NWT (u,v,w):
    - mark $D(u,v) = w$
    - permanently remove arcs (u,v), (v,w) from G

Total: $t^3 + n^2$ calls to NWT on n/t vertices

If NWT in $(n/t)^{3-\varepsilon}$ time then

$(t^3 + n^2)(n/t)^{3-\varepsilon} = n^2 \cdot (n^{1/3})^{3-\varepsilon} = n^{3-\varepsilon/3}$ time.