1 Seidel's algorithm for unweighted undirected APSP

Theorem 1. *APSP in unweighted undirected n-node graphs can be computed in $\mathcal{O}(n^\omega \log n)$ time.*

Proof. Given a graph $G = (V, E)$ we first use fast matrix multiplication to compute graph $G^2 = (V, E^2)$, where

$$E^2 = \{(u, v) | (u, v) \in E \vee \exists_w (u, w) \in E \wedge (w, v) \in E\}.$$

We compute APSP recursively in G^2.

Since the diameter of each connected component halves with each recursive call, after $\log n$ calls we are left with a collection of cliques – each shortest path is of length 0, 1, or ∞, and we can easily compute all of them in $\mathcal{O}(n^2)$ time.

Now, given shortest paths $D(u, v)$ in G^2 we want to compute shortest paths $d(u, v)$ in G. Note that either $d(u, v) = 2D(u, v)$ or $d(u, v) = 2D(u, v) - 1$. We only need to distinguish between these two cases.

If $d(u, v) = 2D(u, v)$, then for all neighbors w of v there must $D(u, w) \geqslant D(u, v)$ (do you see why?). On the other hand, if $d(u, v) = 2D(u, v) - 1$, then there must exists a neighbor w of v with $D(u, w) = D(u, v) - 1$, and for all neighbors w there must $D(u, w) \leqslant D(u, v)$ (do you see why?).

Hence, we use fast matrix multiplication to compute the matrix Z such that

$$Z(u, v) = \sum_w D(u, w) \cdot \mathbb{1}[(w, v) \in E].$$

Now

$$d(u, v) = \begin{cases} 2D(u, v) & \text{if } Z(u, v) \geqslant \deg_v \cdot D(u, v), \\ 2D(u, v) - 1 & \text{if } Z(u, v) < \deg_v \cdot D(u, v). \end{cases}$$

□

Exercise 1. Which parts of this algorithm do not generalize to directed graphs?

2 Faster $(\min, +)$-product for small integers

Theorem 2. *The $(\min, +)$-product of two $n \times n$ matrices with entries in $\{0, \ldots, M\}$ can be computed in $\widetilde{\mathcal{O}}(Mn^\omega)$ time.*

Proof. Replace each input entry x with $(n + 1)^x$; compute the standard $(+, \cdot)$-product; look at the least significant non-zero digits in $(n+1)$-ary encodings of the output entries.

Transformed entries need $M \log n$ bits to represent. Arithmetic operations (addition, subtraction, multiplication, division) on such numbers take $\widetilde{\mathcal{O}}(M \log n)$ time. Hence, the total running time is $\widetilde{\mathcal{O}}(Mn^\omega)$. □

Exercise 2. Show that the above requirement that the entries are nonnegative can be easily circumvented.

Exercise 3. Does the above algorithm, together with the reduction from APSP to $(\min, +)$-product, yield a $\widetilde{\mathcal{O}}(Mn^\omega)$ time algorithm for APSP in directed graphs with weights in $\{-M, \ldots, M\}$?