APSP 2

Problem 1 (Maximum-Weight Triangle). In the *Maximum-Weight Triangle* problem, given an undirected tripartite graph $G = (X, Y, Z, E)$ with weights $w : E \to \mathbb{Z}$, the task is to find a triangle (x, y, z) maximizing $w(x, y) + w(y, z) + w(z, x)$. Show that APSP $_3 \longleftrightarrow _3$ Maximum-Weight Triangle.

Hint: For the APSP-hardness direction reduce from Negative Triangle. For the APSP-easyness direction reduce to $(\min, +)$*-Matrix Multiplication.*

Problem 2 (Graph Radius). Let $G = (V, E)$ be a directed graph with nonnegative edge weights $w : E \to \mathbb{Z}_{\geq 0}$. The *radius* of G is the smallest distance r such that some center node u can reach all other nodes v within distance r; formally the radius is

$$\min_{u \in V} \max_{v \in V} \operatorname{dist}(u, v).$$

The *Graph Radius* problem is to compute the radius of a given directed graph. Show that APSP $_3 \longleftrightarrow _3$ Graph Radius. Can you show the same statement for Graph Radius in undirected graphs?

Bonus Problem 3 (APSP with Small Weights). Recall that under the APSP hypothesis there cannot be a truly subcubic-time algorithm for $(\min, +)$-Matrix Multiplication for polynomially bounded entries. The goal of this exercise is to show that there is a truly subcubic-time algorithm in the case where the entries in all matrices are small.

1. Show that the $(\min, +)$-product of matrices $A, B \in \{0, \dots, W\}^{n \times n}$ can be computed in time $O(n^\omega \cdot W^2)$.
2. Show that the $(\min, +)$-product of matrices $A, B \in \{0, \dots, W\}^{n \times n}$ can even be computed in time $\widetilde{O}(n^\omega \cdot W)$.

   Hint: Try to combine Fast Matrix Multiplication with the Fast Fourier Transform (from sheet 1).

Bonus Problem 4 (Second Shortest Path). In the *Second Shortest Path* problem the input is a directed graph $G = (V, E)$ with edge weights $w : E \to \mathbb{Z}_{\geq 0}$ with two designated vertices $s, t \in V$ and a shortest s-t-path P. The goal is to compute the shortest s-t-path P' *different from* P. Show that APSP $_3 \longleftrightarrow _3$ Second Shortest Path.

Bonus Problem 5 (Strassen's Algorithm). In the lecture we have sketched some ideas behind Strassen's algorithm for matrix multiplication. In this exercise we will make these ideas precise. Recall that the main insight behind Strassen's algorithm is that the matrix product of 2×2 matrices can be expressed as

$$\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \cdot \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} M_1 + M_4 - M_5 + M_7 & M_3 + M_5 \\ M_2 + M_4 & M_1 - M_2 + M_3 + M_6 \end{bmatrix}$$

using only 7 multiplications

$$\begin{aligned} M_1 &= (A_{11} + A_{22}) \cdot (B_{11} + B_{22}), \\ M_2 &= (A_{21} + A_{22}) \cdot B_{11}, \\ M_3 &= A_{11} \cdot (B_{12} - B_{22}), \\ M_4 &= A_{22} \cdot (B_{21} - B_{11}), \\ M_5 &= (A_{11} + A_{12}) \cdot B_{22}, \\ M_6 &= (A_{21} - A_{11}) \cdot (B_{11} + B_{12}), \\ M_7 &= (A_{12} - A_{22}) \cdot (B_{21} + B_{22}). \end{aligned}$$

Based on this insight, design a *recursive* algorithm to multiply $n \times n$ matrices with a running time $T(n)$ that follows the recurrence $T(n) \leq 7 \cdot T(n/2) + O(n^2)$. Conclude (e.g., by induction or by the Master theorem[1]) that this algorithm runs in truly subcubic time $T(n) = O(n^{\log_2(7)}) \leq O(n^{2.81})$.

[1] https://en.wikipedia.org/wiki/Master_theorem_(analysis_of_algorithms)