Distributed Algorithms and Lower Bounds – Exercise 3

In this exercise we will develop an algorithm in the CONGEST model that computes a $(1+\epsilon)$ −approximation to all-pairs shortest paths in nearly-linear time.

Suppose each edge $\{u,v\} \in E$ has a weight $w(\{u,v\}) \in \{1,\dots,N\}$, where $N = n^c$ for some constant $c \geq 1$. We assume that the weight $w(\{u,v\})$ is initially known to both endpoints, u and v.

For convenience we will write $w(u,v)$ instead of $w(\{u,v\})$. For a path $\pi = u_1,\dots,u_k$ we denote

$$w(\pi) = \sum_{i=1}^{k-1} w(u_i,u_{i+1}).$$

Consider the following approach:

- Fix some parameter C. We divide each edge $\{w,v\}$ into "virtual edges", each of weight C. The number of "virtual edges" we need to break $\{u,v\}$ into is
$$\left\lceil \frac{w(u,v)}{C} \right\rceil$$
- Note that because $w(u,v)$ doesn't have to be an integer multiple of C, we might have increased the weight of the edge a little. Let
$$w'(u,v) = C \cdot \left\lceil \frac{w(u,v)}{C} \right\rceil$$
be the total new weight of the edge, after we divide it into $\left\lceil \frac{w(u,v)}{C} \right\rceil$ "virtual edge" of weight C each.
- Now we run the *unweighted all-pairs shortest path* algorithm we saw in the lecture today on the resulting "virtual graph" that we got by breaking up the edges. This lets us compute all-pairs shortest paths with respect to w'. But w' is not exactly the same as w… so our answer is not exact, it is approximate.

Now let's figure out the details:

(1) Let $\pi = u_1,\dots,u_k$ be a path of total weight $w(\pi) = \sum_{i=1}^{k-1} w(u_i,u_{i+1})$. What is the "new virtual weight" of π? That is, what is $w'(\pi) = \sum_{i=1}^{k-1} w'(u_i,u_{i+1})$ as a function of C, k and $w(u)$?

(2) Taking into account the fact that $k \leq n$, how should we choose C to ensure that $w(\pi) \leq w'(\pi) \leq (1+\epsilon)w(\pi)$ for this specific path π?

(3) Since we don't know $w(\pi)$ (this is what we are trying to compute…), we will *guess* various values. Specifically, we will consider exponentially increasing values $d_0 = 1, d_1 = 2, d_2 = 4, \dots, d_{\lceil \log(n \cdot N) \rceil} = 2^{\lceil \log(n \cdot N) \rceil} \geq n \cdot N$.
What is the right way to choose a value C_i corresponding to a guess $d_i = 2^i$, such that *for all paths* π with $2^i \leq w(\pi) \leq 2^{i+1}$ we have $w(\pi) \leq w'(\pi) \leq (1+\epsilon)w(\pi)$?

Notice that for some guesses, we get a virtual graph with large diameter, so the unweighted all-pairs shortest path will take long. So, we need to know when to stop it.

(4) For a path π of weight $w(\pi)$ where $2^i \leq w(\pi) \leq 2^{i+1}$, how many virtual edges do we get when we divide each edge of π into $\left\lceil \frac{w(u,v)}{C_i} \right\rceil$ virtual edges of weight C_i each?

Now combine all of these ingredients into a distributed algorithm that computes $(1+\epsilon)$-approximate APSP in $O\left(n\left(1+\frac{1}{\epsilon}\right)\right)$ rounds!