Oded (August 29, 2022): On the All Pairs Distances Algorithm of Seidel (1995).

Seidel's algorithm reduces the problem of finding all pairwise distances in an unweighted and undirected graph to a logarithmic number of matrix multiplications (for matrices with small integer entries).

The starting observation is that the distances between pairs of vertices in a graph G' that represents paths of length at most two deterimines the pairwise distances in the original graph G up to one unit. That is, let $A = (a_{i,j})$ be the adjacency matrix of the graph G, and $B = (b_{i,j})$ be the adjacency matrix of the graph G' that represents the existence of paths of length at most two in G (i.e., $b_{i,j} = 1$ iff either $a_{i,j} = 1$ or $\sum_k a_{i,k}a_{k,j} > 0$). Then, $\mathrm{dist}_G(u,v) \in \{2 \cdot \mathrm{dist}_{G'}(u,v), 2 \cdot \mathrm{dist}_{G'}(u,v) - 1\}$, where $\mathrm{dist}_H(u,v)$ denotes the distance between u and v in H.

The foregoing yields a recursive procedure for computing all pairwise distances, provided we can distinguish between the case of $\mathrm{dist}_G(u,v) = 2 \cdot \mathrm{dist}_{G'}(u,v)$ and the case of $\mathrm{dist}_G(u,v) = 2 \cdot \mathrm{dist}_{G'}(u,v) - 1$. Performing the latter task relies on the following dichotomy:

- If $\mathrm{dist}_G(u,v) = 2 \cdot \mathrm{dist}_{G'}(u,v)$, then for every neighbour w of v it holds that $\mathrm{dist}_{G'}(u,w) \geq \mathrm{dist}_{G'}(u,v)$. Hence, in this case
$$\sum_{w\in\Gamma(v)} \mathrm{dist}_{G'}(u,w) \geq |\Gamma(v)| \cdot \mathrm{dist}_{G'}(u,v)$$
where $\Gamma(v)$ denotes the set of neighbors of v in G.

- If $\mathrm{dist}_G(u,v) = 2\cdot\mathrm{dist}_{G'}(u,v) - 1$, then for every neighbour w of v it holds that $\mathrm{dist}_{G'}(u,w) \leq \mathrm{dist}_{G'}(u,v)$ and strict inequality holds for at least one w. Hence, in this case
$$\sum_{w\in\Gamma(v)} \mathrm{dist}_{G'}(u,w) < |\Gamma(v)| \cdot \mathrm{dist}_{G'}(u,v).$$

Lastly, observe that $\sum_{w\in\Gamma(v)} \mathrm{dist}_{G'}(u,w)$ equals the $(u,v)^{\text{th}}$ entry of TA, where $T = (\mathrm{dist}_{G'}(i,j))$ is the matrix representing distances in G'. Hence, the recursive algorithm, denoted `APD`, proceeds as follows, when given an adjacency matrix A of a (connected) graph.

1. Compute $Z = AA$ by using matrix multication.

   Let $B = (b_{i,j})$ be a Boolean matrix such that, for every $i \neq j$, it holds that $b_{i,j} = 1$ if and only if either $a_{i,j} = 1$ or $z_{i,j} > 0$.

2. If $b_{i,j} = 1$ for every $i \neq j$, then return $2B - A$.

   (Indeed, in this case, $2 \cdot b_{i,j} - a_{i,j}$ equals 1 if $a_{i,j} = 1$ and equals 2 otherwise.)

3. Compute $T = (t_{u,v}) \leftarrow \mathtt{APD}(B)$ by a recursive call.

   (Recall that $\mathrm{dist}_G(u,v) \in \{2 \cdot t_{u,v}\ ,\ 2 \cdot t_{u,v} - 1\}$.)

4. Compute $X = (x_{i,j}) \leftarrow TA$ by a matrix multiplication.

5. For every $u \neq v$, let $d_{u,v} = 2 \cdot t_{u,v}$ if $x_{u,v} \geq |\Gamma(v)| \cdot t_{u,v}$ and $d_{u,v} = 2 \cdot t_{u,v} - 1$ otherwise.

   Return $D = (d_{u,v})$.

Note that (integer) matrix multiplication was used twice: Once with 0-1 matrices, and once with n-by-n matrices with entries in $\{0, 1, ..., n - 1\}$.

Finding shortests paths. While an explicit description of all shortest paths may have length that is cubic in the number of vertices, it is reasonable to seek faster computation for the next vertex of some shortest path. That is, given the distance matrix $D = (d_{u,v})$ of a graph, we seek a matrix $W = (w_{u,v})$ such that, for every u, v that satisfy $d_{u,v} \geq 2$, it holds that $w_{u,v}$ is the first vertex on a shortest path that connects u and v.

We first consider a simpler problem in which for Boolean n-by-n matrices $A = (a_{i,j})$ and $B = (b_{i,j})$ we wish to find $w_{i,j}$'s such that if $\sum_k a_{i,k}b_{k,j} > 0$ then $a_{i,w_{i,j}} = b_{w_{i,j},j} = 1$. The key observation is that such $w_{i,j}$ can be found by applying several *random sieves* to the matrix A. Specifically, if $\sum_k a_{i,k}b_{k,j} \in [2^{t-1}, 2^t]$, then letting $s_{i,k} = 1$ with probability 2^{-t} and setting $s_{i,k} = 0$ otherwise, implies that $\Pr[\sum_k s_{i,k}a_{i,k}b_{k,j} = 1] = \Omega(1)$. (Indeed, we shall set $a'_{i,k} = s_{i,k}a_{i,k}$.) In this case, there exists a unique $w \in \{k : s_{i,k}=1\}$ such that $a_{i,k}b_{k,j} = 1$. Now, letting $a''_{i,k} = k$ is $s_{i,k} = 1$ and $a''_{i,k} = 0$ otherwise, we observe that this unique w equals $\sum_k a''_{i,k}b_{k,j}$. This implies the following randomized procedure.

1. Select t uniformly in $[\lceil \log_2 n \rceil]$.
2. For each $i \neq k$, set $s_{i,k} = 1$ with probability 2^{-t} (and let $s_{i,k} = 0$ otherwise).[1]
3. Compute $C = (c_{i,j}) \leftarrow A'B$, where $A' = (a'_{i,k}) = (s_{i,k}a_{i,k})$.

   Recall that $c_{i,j} = 1$ indicates that there exists a unique $w \in \{k : s_{i,k}=1\}$ such that $a_{i,k}b_{k,j} = 1$.
4. Compute $T = (t_{i,j}) \leftarrow A''B$, where $A'' = (a''_{i,k}) = (a'_{i,k}k)$.
5. For every $i \neq j$, if $c_{i,j} = 1$, then set $w_{i,j} = t_{i,j}$.

Note that, for every $i \neq j$ such that the $(i, j)^{\text{th}}$ entry of AB is non-zero, $w_{i,j}$ is set with probability $\Omega(1/\log n)$. Hence, repeating the procedure for $O(\log^2 n)$ times, yields the desired matrix $W = (w_{i,j})$; that is, all relevant $w_{i,j}$'s are set. (We comment that the foregoing description is different from the one presented by Seidel.)

Getting back to the original problem (i.e., computing the next vertex on each shortest path), we observe that we can solve this problem for pairs at distance δ by letting A be the adjacency matrix of the graph and B be a Boolean matrix that represents all pairs at distance $\delta - 1$ (i.e., $B = B^{(\delta-1)}$ represents all vertex pairs that are at distance exactly $\delta - 1$). Unfortuntely, this will require us to solve $n-2$ instances of the simple (two Boolean matrix) problem. Using the fact that every $w \in \Gamma(u)$ satisfies $\text{dist}(w, v) \in [\text{dist}(u, v) - 1, \text{dist}(u, v) + 1]$, we infer that $\text{dist}(w, v) = \text{dist}(u, v) - 1$ holds if and only if $\text{dist}(w, v) \equiv \text{dist}(u, v) - 1 \pmod 3$ holds. Hence the problem for pairs at distances that are congruent to τ modulo 3 can be solved considering a Boolean matrix that reprsents all pairs that are at a distance that is congrient to $\tau - 1$ modulo 3. Thus, we may reduce the original problem regarding the distance matrix D to three instances of the simple problem; specifically, the instances $(A, B^{(\tau')})$ for $\tau' \in \{0, 1, 2\}$, where $A = (a_{i,j})$ is the adjacency matrix of the graph (i.e., $a_{i,j} = 1$ iff $d_{i,j} = 1$) and $B^{(\tau')} = (b^{(\tau')}_{i,j})$ such that $b^{(\tau')}_{i,j} = 1$ if $d_{i,j} \equiv \tau' \pmod 3$ and $b^{(\tau')}_{i,j} = 0$ otherwise.

[1]Actually, for each $k \in [n]$, we can set all $s_{i,k}$'s to equal r_k, where the (Boolean) r_k's are selected independently such that $\Pr[r_k = 1] = 2^{-t}$. Furthermore, multiplying an n-by-n matrix that has m non-zero columns by an n-by-n matrix reduces to multiplying an n-by-m matrix by an m-by-n matrix, which reduced to $\lceil n/m \rceil^2$ multiplications of m-by-m matrices. This yields a complexity improvement if the "exponent of matrix multiplication" is larger than 2.