CS 388R: Randomized Algorithms, Fall 2019 October 8th, 2019

Lecture 12: All Pairs Shortest Path

Prof. Eric Price *Scribe: Joshua A. Cook, Supawit Chockchowwat*

NOTE: THESE NOTES HAVE NOT BEEN EDITED OR CHECKED FOR CORRECTNESS

1 The Hand Raising Game

First, we will play a game. Everyone will close their eyes, and each person will either raise their hands or not. We get a prize if exactly one person raises their hands.

If the number of people n is known, each person randomly raise with probability $\frac{1}{n}$. The probability that exactly one person acts is

$$n\left[\frac{1}{n}(1-\frac{1}{n})^{n-1}\right] \simeq n\left[\frac{(e^{-1/n})^{n-1}}{n}\right] \simeq e^{-1+1/n} \simeq \frac{1}{e}$$

This works fine, but what if the exact number of people n is unknown? What if we only have an upperbound $N \geq n$ on the number of people? Can we still get a reasonable probability of success for any sampling size of $n \leq N$?

We could make a guess k on the number of people, and raise our hands with probability $\frac{1}{k}$. Let $k = \frac{N}{2}$, we then individually act with probability $\frac{1}{k} = \frac{2}{N}$, so the success probability is approximately $\frac{2n}{N}e^{-\frac{2n}{N}}$. This is bad when n is small. If our guess is on a different order of magnitude, we are unlikely to succeed. But if our guess within a small factor of the correct answer, we have a good chance of success.

We first guess an order of magnitude, $k \in \{2^0, 2^1, ..., 2^{\log n}\}$. Then, we raise our hands with probability $\frac{1}{k}$. The probability of success then is at least the probability we chose the correct order of magnitude, $\frac{1}{\log(n)}$, times the probability of success if we guessed right, $\frac{1}{2e}$ The expected probability of success w.r.t. the choice of k becomes $\frac{1}{2e\log(n)}$.

We will utilize this to retrieve shortest paths in the later section.

2 Deterministic All Pair Shortest Path (APSP)

Given a dense graph $G = (V, E)$ with $|V| = n$ vertices, $|E| \simeq \binom{n}{2} = O(n^2)$ edges with integer weights, find $D_{u,v}$ the shortest distance from u to v, for all $u, v \in V$.

Deterministic algorithms are:

- Floyd-Warshall: $O(|V|^3) = O(n^3)$

- Bellman-Ford: $O(|V| \cdot |V||E|) = O(n^4)$
- Johnson (Bellman-Ford + Dijkstra): $O(|V||E| + |V| \log |V|) = O(n^3)$
- BFS: $O\Big(|V| \cdot (|V| + |E|)\Big) = O(n^3)$

3 Approximation Algorithm

We want to approximate all pair shortest distances up to factor of 2, i.e. $\frac{\hat{D}_{u,v}}{D_{u,v}} \in \left[\frac{1}{2}, 2\right]$.

Let A be adjacency matrix with edges between all pairs and self-loop,

$$A_{i,j} = \begin{cases} 1 & ; (i,j) \in E \text{ or } i = j \\ 0 & ; \text{ otherwise} \end{cases}$$

Consider A^k, the entry $A^k_{u,v} \neq 0$ if and only if there is a path of length less than or equal to k between u and v. Since the longest possible path is of length $n-1$, we need to consider no more than A^n. The solution to approximate APSP distances is then to compute $A^1, A^2, A^4, A^8, \ldots, A^{n'}$ where $n' = n^{\lceil \log n \rceil}$ and find the earliest non-zero entry for each pair (u, v).

The time complexity for matrix multiplications and searching will be $O(n^\omega \log n + n^2 \log n) = O(n^\omega \log n)$ for smallest known exponent ω for matrix multiplication algorithm (currently, $\omega = 2.373$). Note the second term, the term for finding the first entry, can be reduced to $O(n^2 \log \log n)$ through binary search. In any case, the first term for the matrix multiplications will dominate the search time.

4 All Pair Shortest Path Distance

Suppose $A' = A^2$ and we know $D'_{u,v}$ the APSP distances using A' as the adjacency matrix, we can infer $D_{u,v}$ efficiently as below.

Certainly, $D'_{u,v} = \lceil \frac{D_{u,v}}{2} \rceil$ because all paths of length 2 on A are contained in A^2 and any shortest path can be shortened by a factor of two plus the parity. Therefore, if we can determine the parity (odd or even) of $D_{u,v}$, we can infer it from $D'_{u,v}$.

One way to do this is to look at the $D'_{u,w}$ for w where $(w, v) \in E$. If $D_{u,v} = k$ exactly if the minimum of its neighbors are distance k - 1. Thus for this minimum neighbor w, if k is even, then $D'_{u,w} = \lceil \frac{D_{u,w}}{2} \rceil = \frac{k}{2}$, and if k is odd $D'_{u,w} = \frac{k-1}{2}$.

Thus $D_{u,v}$ is even if and only if $D'_{u,w} \geq D'_{u,v}$ for *all* neighbors w (in other words, $D_{u,v}$. Similarly, $D_{u,v}$ is odd if and only if $D'_{u,w} < D'_{u,v}$ for *some* neighbors w). Unfortunately, trying all neighbors would take $O(n)$ time.

To efficiently find such w for each, use matrix multiplication. Note that if $D_{u,v}$ is even, then $\forall w, (w, v) \in E : D'_{u,w} \geq D'_{u,v}$. If $D_{u,v}$ is odd, then $\forall w, (w, v) \in E : D'_{u,w} \leq D'_{u,v}$ and $\exists w, (w, v) \in$

$E : D'_{u,w} = D'_{u,w} - 1$. The product of A and D' can result in two outcomes:

$$(AD')_{uv} = \sum_{w:A_{w,v}=1} D'_{u,w} = \begin{cases} \geq D'_{u,w}|N(u)| & D_{u,v} \equiv 0 \mod 2 \\ \leq D'_{u,w}|N(u)| - 1 & D_{u,v} \equiv 1 \mod 2 \end{cases}$$

Therefore, given D', we can infer D in $O(n^\omega + n^2) = O(n^\omega)$. Call this routine recursively, dividing n be half each times. The overall time complexity becomes $O(n^\omega \log n)$.

5 APSP from APSP Distances

Now we have a deterministic algorithm that gives us APSP distance, but not the paths themselves. To retrieve the path, utilize the strategy to solve the hand raising game.

Tripartite Graph: Consider a graph with three disjoint sets of vertices X, Y, Z where $X \cup Y \cup Z = V$ and $E \subseteq \{(x, y)|x \in X, y \in Y\} \cup \{(y, z)|y \in Y, z \in Z\}$. Given $x \in X$ and $z \in Z$, we want to find $y \in Y$ such that there is a path (x, y, z).

- The easiest case is when there is exactly one unique $y^{(*)}$ or none. Let $E_1 \subseteq \{(x, y)|x \in X, y \in Y\}$ and $E_2 = \{(y, z)|y \in Y, z \in Z\}$, construct corresponding adjacency matrices A_1 and A_2. We can find such $y^{(*)}$ of a pair (x, z) because $A_1(x, y^{(*)}) \neq 0$ and $A_2(y^{(*)}, z) \neq 0$.

  We want to do matrix multiplication to look for the path, but we will encode the intermediate y in the value of the result. To do this, after the first matrix multiplication when we get y, we will multiply by a diagonal matrix that will set the value of the ys appropriately. Since only one of these intermediate ys map to z, the magnitude of the result will indicate the y that was taken.

$$(A_1 \ diag(1, 2, \ldots, |Y|) \ A_2)_{x,z} = \sum_{y=1}^{|Y|} yA_1(x, y)A_2(y, z) = \begin{cases} y^{(*)} & ; \text{ if } y^{(*)} \text{ exists} \\ 0 & ; \text{ otherwise} \end{cases}$$

- Suppose there are exactly n of such y, we can randomly choose a subset of Y, each with probability of inclusion $1/n$, and use the method above. The probability that we select exactly one such y is equal to the probability proved earlier in the hand raising problem with known number of people.

- Similarly, if the number of such y is unknown, we first guess the number of y with exponential orders and follow the procedure in the two cases above. The probability of success is then

$$\mathbb{P}[\text{unique } y \text{ is selected}] \geq \frac{1}{2e} \cdot \frac{1}{\log n}$$

  Repeat this $O(\log^2 n)$ times and we will succeed w.h.p. Note that verifying a y is correct takes constant time, so we only need high probability we will guess correctly once.

Back to our dense graph, for all pairs (u, v), we'd like to find $w^{(*)}$ s.t. $(w^{(*)}, v) \in E$ and is contained in the shortest path of length $D_{u,v}$; in other words, $D_{u,v} = D_{u,w^{(*)}} + A_{w^{(*)},v}$ by property of shortest path.

In specific case where $D_{u,v} = L$, we know that $D_{u,w^{(*)}} = L - 1$ and $A_{w^{(*)},v} = 1$. We can define a matrix $F^{(L)} \in \{0,1\}^{n \times n}$ where $F^{(L)}_{i,j} = 1$ iff $D_{i,j} = L - 1$. So, if L is fixed, we now reduce the problem to the tripartite problem, where $A_1 = F^{(L)}$ and $A_2 = A$. To solve for all possible L, we need to compute $F^{(1)}, \ldots, F^{(n-1)}$ which solely takes $O(n^3)$ time.

However, recall that $D_{u,w} \in \{L-1, L, L+1\}$ for all w, $(w,v) \in E$, so we can split all pairs (u,v) into 3 groups: $\{(u,v) | D_{u,v} \equiv l \mod 3\}$ for $l \in \{0,1,2\}$. Instead of $F^{(l)}$, we compute

$$G^{(l)}_{i,j} = \begin{cases} 1 & ;\ \text{if } D_{i,j} \equiv l \mod 3 \\ 0 & ;\ \text{otherwise} \end{cases}$$

Solve each one separately, the total time complexity is $O(n^\omega \log^2 n)$ with high probability of success.