Differentially Private All-Pairs Shortest Path Distances: Improved Algorithms and Lower Bounds

Badih Ghazi  
Google Research  
badihghazi@gmail.com

Ravi Kumar  
Google Research  
ravi.k53@gmail.com

Pasin Manurangsi  
Google Research  
pasin@google.com

Jelani Nelson  
UC Berkeley & Google Research  
minilek@alum.mit.edu

March 31, 2022

Abstract

We study the problem of releasing the weights of all-pair shortest paths in a weighted undirected graph with differential privacy (DP). In this setting, the underlying graph is fixed and two graphs are neighbors if their edge weights differ by at most 1 in the \( \ell_1 \)-distance. We give an \( \epsilon \)-DP algorithm with additive error \( \tilde{O}(n^{2/3}/\epsilon) \) and an \((\epsilon, \delta)\)-DP algorithm with additive error \( \tilde{O}(\sqrt{n}/\epsilon) \) where \( n \) denotes the number of vertices. This positively answers a question of Sealfon [Sea16, Sea20], who asked whether a \( o(n) \)-error algorithm exists. We also show that an additive error of \( \Omega(n^{1/6}) \) is necessary for any sufficiently small \( \epsilon, \delta > 0 \).

Finally, we consider a relaxed setting where a multiplicative approximation is allowed. We show that, with a multiplicative approximation factor \( k \), the additive error can be reduced to \( \tilde{O}(n^{1/2+O(1/k)}/\epsilon) \) in the \( \epsilon \)-DP case and \( \tilde{O}(n^{1/3+O(1/k)}/\epsilon) \) in the \((\epsilon, \delta)\)-DP case, respectively.

1 Introduction

In recent years, differential privacy (DP) [DMNS06, DKM+:06] has emerged as a popular notion for quantifying the leakage of sensitive user information by algorithms for a variety of tasks, from data analytics to machine learning, and seen increasing adoption in industry [App17, DKY17, Gue21] and government agencies [Abo18]. Loosely speaking, DP requires that the output of a (randomized) algorithm remains statistically near-indistinguishable when the input dataset is replaced by another so-called “neighboring” dataset, which only differs on the contributions of a single user. DP has been applied to graph problems, including the all-pairs shortest distances problem defined next.

In the all-pairs shortest path distances (APSD) problem, we are given a weighted undirected graph \( G = (V, E, w) \). The goal is to output, for all \( u, v \in V \), an estimate \( \tilde{\text{dist}}(u, v) \) of the smallest-weight distance \( \text{dist}_G(u, v) \) among all paths from \( u \) to \( v \). APSD is among the most well-studied algorithmic graph problems, with known classic algorithms such as Floyd–Warshall that can solve the problem in \( O(n^3) \) time. More recently, (slightly) faster algorithms have been discovered [Wil18]. Additionally, the hypothesis that the problem cannot be solved in \( O(n^{3-\Omega(1)}) \) time serves as one of the main starting points in fine-grained complexity theory (see e.g., [WW18]).
Sealfon [Sea16] was the first to formally study APSD with privacy. In the DP framework, we require that our algorithm be \((\epsilon, \delta)\)-DP, where—following [Sea16]—the neighboring graphs correspond to those whose weight vector \(w\) differs by at most 1 in the \(\ell_1\)-distance. Throughout, we measure the error in terms of the maximum (i.e., \(\ell_\infty\)) error, defined as \(\max_{u,v \in V} |\text{dist}_G(u, v) - \text{dist}(u, v)|\). We say that an output is \(\alpha\)-accurate iff the error is at most \(\alpha\). We use \(n\) to denote the number of vertices. Sealfon gave an \(O(n \log n/\epsilon)\)-error “input-perturbation” algorithm for APSD, which adds Laplace noise to all edge weights and computes the shortest path in this resulting graph with noisy weights. Furthermore, he showed that, if we are required to release the shortest \emph{paths} themselves, then this error is tight up to a logarithmic factor. However, this lower bound does not apply to the APSD problem, where only the weights have to be released.

Given the above state of the problem, Sealfon [Sea16, Sea20] asked the following question:

\begin{quote}
Is it possible to privately release all-pairs distances with error sublinear in \(n\)?
\end{quote}

### 1.1 Our Results

We positively answer this question, by providing private algorithms with sublinear error:

**Theorem 1.** For any \(\epsilon > 0\), there is an \(\epsilon\)-DP algorithm that is \(\tilde{O}(n^{2/3}/\epsilon)\)-accurate w.h.p.

**Theorem 2.** For any \(\epsilon, \delta \in (0,1)\), there is an \((\epsilon, \delta)\)-DP algorithm that is \(O(\sqrt{n}/\epsilon)\)-accurate w.h.p.

We also prove a lower bound of \(\Omega(n^{1/6})\) in terms of accuracy, which not only affirms that polynomial dependency in \(n\) is necessary but—in light of \(O(\log^{2.5} n)\)-accurate algorithm on trees [Sea16]—also separates APSD on general graphs from that on trees.

**Theorem 3.** For any \(\beta \in (0,1)\) and any sufficiently small \(\epsilon, \delta > 0\) (depending only on \(\beta\)), no \((\epsilon, \delta)\)-DP algorithm for APSD can be \(o(n^{1/6})\)-accurate with probability \(1 - \beta\).

Finally, we consider a slightly more relaxed setting where a \emph{multiplicative} approximation is also allowed. More precisely, we say that an output estimate \(\tilde{\text{dist}}(\gamma, \alpha)\)-accurate iff, for all \(u, v \in V\), we have

\[\text{dist}_G(u, v) - \alpha \leq \tilde{\text{dist}}(u, v) \leq \gamma \cdot \text{dist}_G(u, v) + \alpha.\]

Multiplicative approximation has long been widely studied in the non-private setting (e.g., [ABCP98, Coh98, TZ05]). Under this relaxed setting, we give algorithms with improved additive error guarantees. Specifically, for a multiplicative error \(\gamma > 1\), the additive error for our \(\epsilon\)-DP and \((\epsilon, \delta)\)-DP algorithms are \(\tilde{O}(n^{1/2+O(1/\gamma)}/\epsilon)\) and \(\tilde{O}(n^{1/3+O(1/\gamma)}/\epsilon)\) respectively.

**Theorem 4.** For any \(\epsilon > 0\) and \(k \in \mathbb{N}\), there is an \(\epsilon\)-DP algorithm that is \((2k - 1, \tilde{O}_k(n^{(k+1)/(2k+1)}/\epsilon))\)-accurate w.h.p.

**Theorem 5.** For any \(\epsilon, \delta \in (0,1)\) and \(k \in \mathbb{N}\), there is an \((\epsilon, \delta)\)-DP algorithm that is \((2k - 1, \tilde{O}_k(n^{(k+1)/(3k+1)}/\epsilon))\)-accurate w.h.p.

---

1. Note that, similar to [Sea16], we have to keep the underlying graph \((V, E)\) fixed and we are only allowed to change the weights. This is required to get any non-trivial result: as pointed out in [Sea16], removing an edge can make the graph disconnected and therefore the distance can go from finite to infinite.

2. For readability, our upper bounds in this section suppress factors that are polylogarithmic in \(n\) and \(1/\delta\). We use \(\tilde{O}\) to hide a dependency of the form \((\log n)^{O(1)}\) in the \(\epsilon\)-DP case and \((\log n \cdot \log(1/\delta))^{O(1)}\) in the \((\epsilon, \delta)\)-DP case.
We remark that our lower bound (Theorem 3) does not apply to the case where multiplicative approximation is allowed, and it remains an interesting open question whether it is possible to achieve an $n^{o(1)}$ additive error in this case.

### 1.2 Technical Overview

**Additive-Approximation Algorithms.** We start with algorithms that only incur an additive error (Theorems 1 and 2). Our algorithms will combine two primitives: the aforementioned “input-perturbation” algorithm and an “output-perturbation” algorithm. For the latter (also noted in [Sea16, Sea20]), we simply add Laplace noise to each of the shortest path distances. Since each shortest path distance has sensitivity one, it suffices to take the Laplace noise scale to be $n^2/\epsilon$ to achieve $\epsilon$-DP. This yields an error of $\tilde{O}(n^2/\epsilon)$ for the problem.

To combine the two, we need a notion of $t$-hop path. For $t \in \mathbb{N}$, a path is said to be a $t$-hop path if it consists of at most $t$ edges. As proved in [Sea16], for any pair $u, v$ for which the shortest path is a $t$-hop path, the error of the input-perturbation algorithm is in fact just $\tilde{O}(t/\epsilon)$ instead of the worst-case bound of $\tilde{O}(n/\epsilon)$.

Our strategy is now as follows: we sample a subset $S \subseteq V$ of a given size $s$. Vertices in $S$ will serve as intermediate vertices for long paths. We compute the shortest path distances between every pair of vertices in $S$ using the output-perturbation algorithm, whose error is now only $\tilde{O}(s^2/\epsilon)$. We also use the input-perturbation algorithm to compute shortest $t$-hop path distances between all pairs of vertices in $V$; this has an error of $\tilde{O}(t/\epsilon)$. Finally, for each pair $u, v$, we compute the distance between them by taking the minimum of (i) the shortest $t$-hop path distance between $u$ and $v$ and (ii) the minimum over all $w, z \in S$ of the sum of the shortest $t$-hop path distance between $u, w$, the shortest path distance between $w, z$ and the shortest $t$-hop path distance between $z, v$.

For this algorithm to work, $t$ has to be sufficiently large so that every shortest path contains a vertex in $S$ among its first $t$ and last $t$ points. It is not hard to see that this w.h.p. holds as long as we pick $t = \tilde{O}(n/s)$. In total, the error is then $\tilde{O}(s^2/\epsilon) + \tilde{O}(t/\epsilon) = \tilde{O}(\frac{1}{3}(s^2 + n/s))$. This is minimized when we take $s = \tilde{O}(n^{1/3})$, which yields an error of $\tilde{O}(n^{2/3}/\epsilon)$ as claimed in Theorem 1.

In the case of $(\epsilon, \delta)$-DP, we may use the Gaussian mechanism instead of the Laplace mechanism. This reduces the error in the output-perturbation algorithm to $\tilde{O}(s/\epsilon)$. By taking $s = \tilde{O}(\sqrt{n})$, we get the claimed bound in Theorem 2.

We end by noting that in our actual algorithms we do not use the Laplace/Gaussian mechanisms for computing the shortest path distances between points in $S$; rather, we use recent improved results [SU16, DK20, GKM21] that allow us to remove a logarithmic bound in the error.

**Lower Bounds.** Our lower bounds follow from a reduction from the linear queries\(^3\) problem. A set of linear queries can be encoded as a matrix $A \in \{0, 1\}^{d \times n}$. The input to the algorithm is a vector $z \in \{0, 1\}^n$ (where two datasets $z, z'$ are neighbors iff they differ on a single coordinate). The goal is to output an estimate of $Az$. We write $a_i$ as the $i$th row of $A$.

To encode linear queries into a graph, we let each $z_i$ be a weight of some edge in our graph, and the goal is to identify each linear query $a_i$ with a shortest path. Our goal is that the shortest path has length $\langle a_i, z \rangle$ (perhaps plus some constant that we know beforehand). Although this seems promising, it is not simple to embed linear queries into a graph in this way. Consider, for example, two linear queries $z_1 + z_2 + z_4$ and $z_1 + z_3 + z_4$. Here, if the first one corresponds to a path that has

\(^3\)Since we are using only Boolean matrices $A$ here, this version is sometimes called counting queries.
z_1, z_2, z_4 in this order and the second corresponds to a path that has z_1, z_3, z_4 in this order, then clearly both of them cannot be the shortest path at the same time.

To avoid the issue described above, we embed the graph while ensuring that each linear query appears in a unique shortest path between a pair of endpoints. Roughly speaking, this is done by taking a graph \( H = (U, E_H, w_H) \) that has many unique shortest paths, then scaling the edge weights so that the gap between any shortest path and the second shortest path (w.r.t. the same pair) is sufficiently large. Then, we replace each vertex \( i \in U \) by an edge, where the weight is equal to \( z_i \). (Note that, strictly speaking, such a replacement is only well-defined when there is an ordering on the vertices; this is formalized in Section 4.) In this way, each linear query includes all the vertices presented in a corresponding unique shortest path.

To obtain a concrete bound, we use a graph \( H \) where each vertex is a point in the plane and there is an edge between two vertices iff the line segment between them does not contain another vertex and the weight is simply the Euclidean distance. In this case, the unique shortest paths are simply lines in the plane. These “point-line” set systems have been studied before, and we appeal to a discrepancy lower bound of \( \Omega(n^{1/6}) \) of such a set system by Chazelle and Lvov \[CL01\]; together with the known connection between discrepancy and linear queries lower bounds \[MN12\], we arrive at our lower bound for the APSD problem.

On a tangent, the original work of Chazelle and Lvov \[CL01\] proved a nearly tight upper bound of \( O(n^{1/6} \log^{2/3} n) \) for point-line systems. We observe that a sharper partial coloring lemma shown by Lovett and Meka \[LM13\] immediately implies an improved—and tight—upper bound of \( O(n^{1/6}) \), which might be of independent interest. The details of this proof can be found in Appendix A.1.

### Additive and Multiplicative Approximation Algorithms

When also allowed multiplicative approximation, we use the algorithm of Thorup and Zwick \[TZ05\], which constructs a \((2^k - 1)\)-stretch spanner with only \( O_k(n^{1+1/k}) \) edges. We privatize their algorithm by replacing each non-private primitive with a private one (e.g., selecting a closeby point is now done using the Exponential Mechanism \[MT07\]). Roughly speaking, the benefit of this algorithm is that, unlike in the “naive” output-perturbation algorithm we described above that queries the distances of all \( O(n^2) \) pairs of vertices, here we only need to query the shortest path distances of only \( O_k(n^{1+1/k}) \) pairs of vertices. Therefore, this also reduces the noise needed for each query to attain the same level of privacy. By using this algorithm to compute the shortest paths within \( S \), we can thus reduce the additive error of the overall algorithm, albeit at the price of incurring a multiplication error.

### 1.3 Related Work

As alluded to earlier, our work is most closely related to that of Sealfon \[Sea16\]. In addition to the already discussed results, \[Sea16\] also considered several special cases, including an \( \epsilon \)-DP \( O(\log^{2.5} n/\epsilon) \)-accurate algorithm for trees. It is worth noting that this is a generalization of the prefix sum (aka one-dimensional range query) problem and Sealfon’s algorithm can be considered a generalization of previous algorithms for prefix sum \[DNPR10, CSS11, DNRR15\]. Another result from \[Sea16\] concerns the case where each weight is bounded by \( M \) (i.e., \( w_e \leq M \) for all \( e \in E \)), in which case an algorithm with error \( \tilde{O}(\sqrt{nM}) \) was shown. Interestingly, Sealfon’s algorithm also employs the idea of \( t \)-hop paths. However, instead of trying to find \( S \) that covers the shortest paths

\[4\]For point-line system, the discrepancy is the minimum across all 2-colorings of all points of the maximum difference between the number of each color on any line. For a more formal definition, see Definition \[14\]
(as we do), his algorithm directly finds $S$ that covers the original graph $G$ via $t$-hop paths. Note that this can be computed with no privacy loss at all. The algorithm then proceeds to compute the all-pair shortest path distances for $S$ via the output-perturbation algorithm. The main bottleneck here is that, since we are only guaranteed that there are some $t$-hop paths from any vertex $v \in V$ to $S$ but this path may not coincide with (a subpath of) the shortest path, the error bound becomes $t \cdot M$ (because we can only guarantee a bound of $M$ on each edge); this causes a dependency on $M$ in the final bound.

2 Background

For any $m \in \mathbb{N}$, we write $[m]$ as a shorthand for $\{1, \ldots, m\}$. Let $x \sim D$ denote that the random variable $x$ is sampled according to the distribution $D$. We use the following notion of neighboring datasets throughout the paper: two input datasets $X, X'$ are said to be neighbors if $\|X - X'\|_1 \leq 1$.

We first recall the definition of differential privacy (DP).

**Definition 6** (Differential Privacy). An algorithm $M$ is $(\epsilon, \delta)$-differentially private ($(\epsilon, \delta)$-DP) iff for every neighboring input datasets $X, X'$ and every set $S$ of possible outcomes, we have $\Pr[M(X) \in S] \leq e^\epsilon \cdot \Pr[M(X') \in S] + \delta$. When $\delta = 0$, we simply write $\epsilon$-DP instead of $(\epsilon, \delta)$-DP.

**Definition 7** (Sensitivity). The $\ell_p$ sensitivity of $f : \mathcal{X} \to \mathbb{R}^D$ is defined as $\Delta_p(f) := \max_{X, X'} \|f(X) - f(X')\|_p$, where the maximum is over all neighboring datasets $X, X'$.

We will also use the Laplace mechanism [DMNS06], which we recall here. Let Lap($b$) denote the Laplace distribution with scale $b$.

**Lemma 8** (Laplace Mechanism [DMNS06]). Given any function $f : \mathcal{X} \to \mathbb{R}^D$, the Laplace mechanism, on input $X \in \mathcal{X}$, outputs $f(X) + (\eta_1, \ldots, \eta_D)$ where each $\eta_i \sim$ Lap($\Delta_1(f)/\epsilon$) is independent. The Laplace mechanism is $\epsilon$-DP, and it is $O(\Delta_1(f) \cdot \log(1/\beta)/\epsilon)$-accurate with probability $1 - \beta$ for all $\beta \in (0, 0.5)$.

Next we state the composition theorems for DP, which will be used in our analysis.

**Theorem 9** (Basic Composition [DMNS06, DL09]). Let $\epsilon_1, \ldots, \epsilon_k, \delta_1, \ldots, \delta_k > 0$. If we run $k$ (possibly adaptive) mechanisms where the $i$th mechanism is $(\epsilon_i, \delta_i)$-DP, then the entire algorithm is $(\sum_{i=1}^k \epsilon_i, \sum_{i=1}^k \delta_i)$-DP.

We will also use the advanced composition; the version stated below is a special case of [DR14, Corollary 3.21].

**Theorem 10** (Advanced Composition [DRV10, DRT14]). Let $\epsilon, \delta \in (0, 1]$ and $k \in \mathbb{N}$. Suppose that we run $k$ (possibly adaptive) mechanisms where each mechanism is $\left(\frac{\epsilon}{2\sqrt{2k\ln(2/\delta)}}, \frac{\delta}{2k}\right)$-DP. Then the entire algorithm is $(\epsilon, \delta)$-DP.

---

5 The pdf of the Laplace distribution with scale $b$ is $\propto \exp(-|x|/b)$. 

---

5
2.1 Answering Multiple Queries

We use the following results that provide stronger guarantees than the union bound when answering multiple queries. As stated earlier, these help save a logarithmic factor compared to using the Laplace or the Gaussian mechanisms directly.

**Theorem 11** ([SU16]). Let $f : \mathcal{X} \rightarrow \mathbb{R}^D$ be any function, $\epsilon > 0$ and $\beta \in (0, 0.5)$. There exists an $\epsilon$-DP algorithm $M$ such that, for any input $X$, with probability $1 - \beta$, we have $\|M(X) - f(X)\|_{\infty} \leq O\left(D + \log(1/\beta)\right) \cdot (\Delta_\infty(f)/\epsilon)$.

**Theorem 12** ([GKM21]). Let $f : \mathcal{X} \rightarrow \mathbb{R}^D$ be any function and $\epsilon, \delta > 0$. There exists an $(\epsilon, \delta)$-DP algorithm $M$ such that, for any input $X$, with probability $1 - O(1/D^{10})$, we have $\|M(X) - f(X)\|_{\infty} \leq O\left(\sqrt{D \log(1/\delta)}\right) \cdot (\Delta_\infty(f)/\epsilon)$.

2.2 Selection via the Exponential Mechanism

In the selection problem, we are given a set $C$ of candidates together with sensitivity-1 functions $f_c$ for all $c \in C$. The goal is to output a candidate $c \in C$ with the smallest $f_c(X)$ where $X$ denotes the (private) input. We say that the output $(\hat{c}, \tilde{f}_\hat{c})$ is $\alpha$-accurate iff $f_\hat{c}(X) \leq \min_{c \in C} f_c(X) + \alpha$ and $|\tilde{f}_\hat{c} - f_\hat{c}(X)| \leq \alpha$.

The exponential mechanism [MT07] can solve this problem with error $O(\log |C|/\epsilon)$, stated more precisely below.

**Theorem 13** ([MT07]). For any $\epsilon > 0$, there is an $\epsilon$-DP algorithm for selection that with probability $1 - \beta$ is $O\left(\log(|C|/\beta)/\epsilon\right)$-accurate.

2.3 Linear Queries

A set of linear queries is parametrized by a matrix $A \in \mathbb{R}^{D \times N}$. The input to the problem is a vector $v \in \{0, 1\}^n$ and the goal is to compute $Av$. As before, two inputs $v, v'$ are neighbors iff $\|v - v\|_1 \leq 1$.

We exploit the connection between linear queries and discrepancy proved in [MN12]. Before stating their result, we need a variant of discrepancy they employ.

**Definition 14** (Discrepancy). For $\gamma \in (0, 1]$, let $\text{disc}_\gamma(A) := \min_{z \in \{-1, 0, +1\}^N, \|z\|_1 \geq \gamma N} \|A^T z\|_{\infty}$. When $\gamma = 1$, we may write $\text{disc}(A)$ as a shorthand for $\text{disc}_1(A)$.

We can now state their results, which allow us to translate lower bounds on the discrepancy of $A$ to that of the corresponding linear query problem.

**Theorem 15** ([MN12]). For any $\beta \in (0, 1)$, there exist $\gamma, \epsilon, \delta > 0$ such that, for any $A$, no $(\epsilon, \delta)$-DP mechanism is $(\text{disc}_\gamma(A)/2)$-accurate with probability $1 - \beta$ for the $A$-linear query problem.

---

6 The original exponential mechanism from [MT07] does not output the estimate $\hat{f}_c$ but this can easily be done using, e.g., the Laplace mechanism.

7 The case $\gamma = 1$ is the standard definition of discrepancy; see e.g., [Cha01] for more details on discrepancy theory.
3 Algorithm

Recall from the Introduction that a path is said to be a $t$-hop path if it consists of at most $t$ edges. We write $\text{dist}^t_{S}(u, v)$ to denote the smallest-weight among all $t$-hop paths from $u$ to $v$.

3.1 Input-Perturbation Algorithm

As described earlier in the Introduction, the “input-perturbation” algorithm that adds Laplace noise already produces accurate distances for shortest $t$-hop paths when $t$ is small. This is stated below. Note that [Sea16] already proved a similar statement but with one-sided error; we reprove it here for completeness.

**Theorem 16.** Let $t \in \mathbb{N}$ be any positive integer and $\beta \in (0, 0.5)$. There is an $\epsilon$-DP algorithm for computing $\text{dist}^t(\tilde{u}, \tilde{v})$ such that with probability $1 - \beta$ we have

$$\max_{u, v \in V} |\text{dist}^t(u, v) - \text{dist}^t_G(u, v)| \leq O(t \cdot \log(n/\beta)/\epsilon).$$

**Proof.** The algorithm works as follows. First, construct the weight $\tilde{w}$ by applying the $\epsilon$-DP Laplace mechanism (Lemma 3) to $w$, i.e., let $\tilde{w}(u, v) = w(u, v) + z(u, v)$ where $z(u, v) \sim \text{Lap}(1/\epsilon)$. Finally, we let $\text{dist}^t(\tilde{u}, \tilde{v})$ be the weight of the shortest $t$-hop path from $\tilde{u}$ to $\tilde{v}$ in $\tilde{G} = (V, E, \tilde{w})$. Since $\tilde{w}$ is a result of an $\epsilon$-DP Laplace mechanism and our final output $\text{dist}^t$ is a post-processing of $\tilde{w}$, we can conclude that our algorithm is $\epsilon$-DP.

We next argue its utility. From the utility guarantee of the Laplace mechanism (Lemma 3), we can conclude that with probability $1 - \beta$, we have

$$\|\tilde{w} - w\|_{\infty} \leq O(\log(|E|/\beta)/\epsilon) \leq O(\log(n/\beta)/\epsilon).$$

(1)

Since each $t$-hop path consists of at most $t$ edges, (1) implies that the weight of any $t$-hop path in $G$ and that in $\tilde{G}$ differ by at most $O(t \cdot \log(n/\beta)/\epsilon)$. From our definitions of $\text{dist}^t$ and $\text{dist}^t$, the claimed bound immediately follows. \qed

3.2 Output-Perturbation Algorithm

For a subset $S \subseteq V$, we define the S-pair shortest path distances problem to be the problem of outputting, for all $u, v \in S$, an estimate $\text{dist}(u, v)$ of the smallest-weight $\text{dist}_G(u, v)$ among all paths from $u$ to $v$. (Note that $\text{dist}_G(u, v)$ is still the distance in the original graph $G$, i.e., the path is allowed to leave the subset $S$.) Of course, the original APSD problem is simply the case $S = V$.

Notice that each $\text{dist}_G(u, v)$ for all $u, v \in S$ has sensitivity at most one. Therefore, we may apply Theorem 11 and Theorem 12 directly to the function $f(w) = (\text{dist}_G(u, v))_{u, v \in S}$ to achieve the following guarantees for the S-pair shortest path distances problem.

**Corollary 17.** For any $S \subseteq V$ and $\epsilon > 0$, there exists an $\epsilon$-DP algorithm for S-pair shortest path distances problem that is $O(|S|^2/\epsilon)$-accurate with probability $1 - O(|S|^{-10})$.

**Corollary 18.** For any $S \subseteq V$ and $\epsilon, \delta > 0$, there exists an $(\epsilon, \delta)$-DP algorithm for S-pair shortest path distances problem that is $O(|S|\sqrt{\log(1/\delta)}/\epsilon)$-accurate with probability $1 - O(|S|^{-10})$. \footnote{Note that this can be computed in polynomial-time using a simple dynamic programming procedure: let $\text{dist}^0(u, u) = 0$ for all $u \in V$ and $\text{dist}^0(u, v) = \infty$ for all $u \neq v \in V$. Then, for $i \in [t], u, v \in V$, let $\text{dist}^i(u, v) = \min_{(u, w) \in E}(\text{dist}^i(u, w) + \text{dist}^{i-1}(w, v))$.}
3.3 Combining the Two Algorithms: Beating the Linear Error

In this section, we show how to combine the input-perturbation and output-perturbation algorithms to get our final error bound. We start with the following generic lemma for combining the two algorithms. (We state this lemma in such a way that includes the multiplicative error, since we will use it in the latter sections as well.)

Lemma 19. Suppose that for any set $S$, there exists an $(\epsilon, \delta)$-DP algorithm that is $(\gamma_{\epsilon,\delta}(|S|), \alpha_{\epsilon,\delta}(|S|))$-accurate with probability at least $1 - \beta_{\epsilon,\delta}(|S|)$. Then, for any parameter $s \in [|V|]$, there exists an $(\epsilon, \delta)$-DP algorithm that is $(\gamma_{\epsilon/2,\delta}(s), O(\alpha_{\epsilon/2,\delta}(s) + n/s \cdot \log^2 n/\epsilon))$-accurate with probability $1 - \beta_{\epsilon/2,\delta}(s) - O(1/n)$.

Proof. The algorithm works as follows:

- Randomly sample a subset $S \subseteq V$ of size $s$.
- Run the $(\epsilon/2, \delta)$-DP algorithm to compute estimates $\hat{\text{dist}}^S(u, v)$ for all $u, v \in S$.
- Run the $(\epsilon/2)$-DP algorithm from Theorem 16 to compute $\hat{\text{dist}}^{|t|}(u, v)$ for all $u, v \in V$ where $t := \lceil 10 \cdot (n/s) \log n \rceil$.
- Finally, for all $u, v \in V$, let
  \[ \hat{\text{dist}}(u, v) = \min \left\{ \hat{\text{dist}}^{|t|}(u, v), \min_{w,z \in S} \hat{\text{dist}}^S(u, w) + \hat{\text{dist}}^S(w, z) + \hat{\text{dist}}^S(z, v) \right\}, \]
  and output $\hat{\text{dist}}$.

Basic composition (Theorem 9) immediately implies that our algorithm is $(\epsilon, \delta)$-DP. As for the utility, the guarantee of the $S$-pair shortest path distances algorithm implies that, with probability $1 - \beta_{\epsilon/2,\delta}(s)$, we have

\[ \text{dist}_G(w, z) - \alpha_{\epsilon/2,\delta}(s) \leq \hat{\text{dist}}^S(w, z) \leq \gamma_{\epsilon/2,\delta}(s) \cdot \text{dist}_G(w, z) + \alpha_{\epsilon/2,\delta}(s) \quad \forall w, z \in S. \tag{2} \]

Furthermore, Theorem 16 ensures that, with probability $1 - 1/n$, we have

\[ |\hat{\text{dist}}^{|t|}(u, v) - \text{dist}_G^{|t|}(u, v)| \leq O(t \cdot \log n/\epsilon) \quad \forall u, v \in V. \tag{3} \]

Next, consider any $u, v \in V$. Let $u = p_0, p_1, \ldots, p_\ell = v$ denote a shortest path from $u$ to $v$ (ties broken arbitrarily). If $\ell > t$, let $i^{u,v,\text{min}}, i^{u,v,\text{max}}$ denote the smallest and largest indices $i$ such that $p_i \in S$. Since $S$ is a u.a.r. $s$-size subset of $V$, a simple probability argument implies that $i^{u,v,\text{min}} \leq t$ and $i^{u,v,\text{max}} \geq \ell - t$ with probability $O(1/n^2)$.

Henceforth, we will condition on these three events to happen; this occurs with probability at least $1 - \beta_{\epsilon/2,\delta}(s) - O(1/n)$. Under this assumption, we have

\[ \hat{\text{dist}}(u, v) = \min \left\{ \hat{\text{dist}}^{|t|}(u, v), \min_{w,z \in S} \hat{\text{dist}}_G^{|t|}(u, w) + \hat{\text{dist}}_G^{|t|}(w, z) + \hat{\text{dist}}_G^{|t|}(z, v) \right\} \]

\[ > \geq \min \left\{ \text{dist}^{|t|}(u, v), \min_{w,z \in S} \text{dist}_G^{|t|}(u, w) + \text{dist}_G^{|t|}(w, z) + \text{dist}_G^{|t|}(z, v) \right\} \]
− 2 \cdot \alpha_{\epsilon/2, \delta}(s) − O(t \cdot \log n/\epsilon) \\
\geq \text{dist}_G(u, v) − O(\alpha_{\epsilon/2, \delta}(s) + n/s \cdot \log^2 n/\epsilon).

As for the upper bound on \( \tilde{\text{dist}}(u, v) \), consider two cases, based on whether the shortest path between \( u, v \) has at least \( t \) edges. If not, then we have
\[
\tilde{\text{dist}}(u, v) \leq \tilde{\text{dist}}^{\leq t}(u, v) \leq \text{dist}^{\leq t}(u, v) + O(t \cdot \log n/\epsilon) = \text{dist}(u, v) + O(n/s \cdot \log^2 n/\epsilon).
\]
Otherwise, we also have
\[
\begin{align*}
\tilde{\text{dist}}(u, v) & \leq \min_{u, z \in S} \tilde{\text{dist}}^{\leq t}(u, w) + \tilde{\text{dist}}^{S}(w, z) + \tilde{\text{dist}}^{\leq t}(z, v) \\
& \leq \text{dist}^{\leq t}(u, p^u_{v, \text{min}}) + \text{dist}^{S}(p^u_{v, \text{min}}, p^u_{v, \text{max}}) + \text{dist}^{\leq t}(p^u_{v, \text{max}}, v) \\
& \leq \gamma \cdot \text{dist}^{\leq t}(u, p^u_{v, \text{min}}) + \text{dist}^{S}(p^u_{v, \text{min}}, p^u_{v, \text{max}}) + \text{dist}^{\leq t}(p^u_{v, \text{max}}, v) \\
& \quad + 2 \cdot \alpha_{\epsilon/2, \delta}(s) + O(t \cdot \log n/\epsilon) \\
& \quad + 2 \cdot \alpha_{\epsilon/2, \delta}(s) + O(n/s \cdot \log^2 n/\epsilon) \\
& = \gamma \cdot \text{dist}_G(u, v) + O(\alpha_{\epsilon/2, \delta}(s) + O(n/s \cdot \log^2 n/\epsilon)) .
\end{align*}
\]

Therefore, our algorithm is \((\gamma, O(\alpha_{\epsilon/2, \delta}(s) + n/s \cdot \log^2 n/\epsilon)) \) with probability \( 1 - \beta_{\epsilon/2, \delta}(s) - O(1/n) \) as desired. \( \square \)

By plugging Corollary 17 into Lemma 19 with \( s = 3\sqrt{n \log^2 n} \), we immediately arrive at the following, which is a more detailed version of Theorem 4.

**Theorem 20.** For any \( \epsilon > 0 \), there exists an \( \epsilon\)-DP algorithm for APSD that is \( O\left(\frac{n^{2/3} \log^{4/3} n}{\epsilon}\right) \)-accurate w.h.p.

Similarly, by plugging Corollary 18 into Lemma 19 with \( s = \sqrt{n \log n / \log(1/\delta)} \), we immediately arrive at the following, which is a more detailed version of Theorem 2.

**Theorem 21.** For any \( \epsilon, \delta > 0 \), there exists an \((\epsilon, \delta)\)-DP algorithm for APSD that is \( O\left(\frac{\sqrt{n \log n \log(1/\delta)}}{\epsilon}\right) \)-accurate w.h.p.

### 4 Lower Bound

In this section, we prove our lower bound (Theorem 3).

#### 4.1 Reducing Linear Queries to Shortest Path

We start by presenting our reduction from linear queries to APSD. We follow the terminology of [Bod19] for the notion of strongly metrizable path systems.
**Definition 22** (Strongly Metrizable Path System [Bod19]). A path system is a pair $(U, \Pi)$ where $U$ is the set of vertices and $\Pi$ is a set of paths. A path system $(U, \Pi)$ is strongly metrizable if there is a weighted graph $H = (U, E_H, w_H)$ such that each path $\pi \in \Pi$ is the unique shortest path between its endpoints in $H$.

Additionally, an ordered path system $(U, \Pi, \prec)$ is a path system $(U, \Pi)$ together with a (total) ordering $\prec$ on $U$ such that every path $\pi \in \Pi$ respects such an ordering, i.e., $\pi = (p_1, \ldots, p_\ell)$ satisfies $p_i \prec p_j$ for all $1 \leq i < j \leq \ell$. We say that an ordered path system $(U, \Pi, \prec)$ is strongly metrizable iff the underlying path system $(U, \Pi)$ is strongly metrizable.

Finally, we define the incidence matrix of the path system $\Pi$ to be the matrix $A^{\Pi}$ whose rows are indexed by $\pi \in \Pi$ and columns are indexed by $u \in U$ such that $A^{\Pi}_{\pi, u} = 1$ iff $u \in \pi$.

Below we provide our reduction from linear queries to APSD. This reduction works by replacing each vertex in $U$ by an edge with a weight being the input of the linear query problem, as outlined in the Introduction.

**Lemma 23.** Let $(U, \Pi, \prec)$ be any strongly metrizable ordered path system. If there exists an $(\epsilon, \delta)$-DP algorithm that is $\alpha$-accurate with probability $1 - \beta$ for APSD on all $2|U|$-vertex graphs, then there exists an $(\epsilon, \delta)$-DP algorithm that is $\alpha$-accurate with probability $1 - \beta$ for $A^{\Pi}$-linear query.

**Proof.** Since $(U, \Pi, \prec)$ is strongly metrizable, there exists a weighted graph $H = (U, E_H, w_H)$ such that every path $\pi \in \Pi$ is the unique shortest path between its endpoints in $H$. We may assume w.l.o.g. that the weight of each $\pi \in \Pi$ is smaller than the second-shortest path by an additive factor of $|U| + 1$; otherwise, we can simply scale the weights by an appropriate factor.

We construct the graph $G = (V, E)$ for APSD as follows:

- For every $u \in U$, construct two vertices $u^{\text{in}}$ and $u^{\text{out}}$ in $V$, and add an edge $(u^{\text{in}}, u^{\text{out}})$ to $E$.
- For every edge $(u, v) \in E_H$ where $u \prec v$, add an edge $(u^{\text{out}}, v^{\text{in}})$ in $E$.

Given an input $z \in \{0, 1\}^n$ to the $A^{\Pi}$-linear query, we construct the weights $w$ for $G$ as follows:

- For every $u \in U$, let $w((u^{\text{in}}, u^{\text{out}})) = z_u$.
- For every $(u^{\text{out}}, v^{\text{in}}) \in E$, let $w((u^{\text{out}}, v^{\text{in}})) = w_H(u, v)$. (Note that this is independent of $z$.)

It is simple to verify that two neighboring inputs $z, z'$ give rises to neighboring weights $w, w'$. Furthermore, it is also easy to see that, for every path $\pi = (u = p_1, p_2, \ldots, p_\ell = v) \in \Pi$, the unique shortest path between $u^{\text{in}}$ and $v^{\text{out}}$ in $G$ is exactly the path $(u^{\text{in}}, u^{\text{out}}, p_2^{\text{out}}, \ldots, v^{\text{in}}, v^{\text{out}})$. This means that

$$\text{dist}_G(u^{\text{in}}, v^{\text{in}}) = w_H(\pi) + \sum_{p \in \pi} z_p = w_H(\pi) + (A^{\Pi}z)_\pi.$$ 

Therefore, we may run the $(\epsilon, \delta)$-DP algorithm for APSD to get an estimate $\widetilde{w}$ and then output $\text{dist}(u^{\text{in}}, v^{\text{in}}) - \tilde{w}_H(\pi)$ for all $\pi \in \Pi$. This is an $(\epsilon, \delta)$-DP algorithm for $A^{\Pi}$-linear query. Furthermore, if the APSD algorithm is $\alpha$-accurate with probability $1 - \beta$, then the linear query algorithm is also $\alpha$-accurate with probability $1 - \beta$.

We remark that, if we were working in the directed graph case, then we may start with the graph $H = (U, E_H, w_H)$ that is directed and there would be no need for the ordering $\prec$. However, the ordering is needed for undirected graphs in the proof above to avoid the situation where some edge $(p_1^{\text{in}}, p_1^{\text{out}})$ could be skipped in some shortest path in $G$. 

---

10
4.2 Lower Bound via Discrepancy of Point-Line System

An \((N, D)\)-planar point-line system \((N, D)\)-PPLS is given by \((\mathcal{P}, \mathcal{L})\) where \(\mathcal{P} = \{p_1, \ldots, p_N\} \subseteq \mathbb{R}^2\) is a set of points in the plane and \(\mathcal{L} = \{L_1, \ldots, L_D\}\) is a set of lines; when \(D = N\), we abbreviate it as \(N\)-PPLS. The adjacency matrix \(A^{\mathcal{P}, \mathcal{L}} \in \{0, 1\}^{D \times N}\) is such that \(A^{\mathcal{P}, \mathcal{L}}_{p, p'}\) is one iff \(p, p' \in \mathcal{L}\). The discrepancy of \((\mathcal{P}, \mathcal{L})\) is defined as the discrepancy of \(A^{\mathcal{P}, \mathcal{L}}\).

Chazelle and Lvov [CL01] proved the following lower bound for the point-line system.

**Theorem 24 ([CL01]).** There is an \(N\)-PPLS with discrepancy \(\Omega(N^{1/6})\).

Using the same technique as in [MN12] it is possible to also show a lower bound for the discrepancy variant where \(\gamma \neq 1\) (Definition [14]).

**Corollary 25.** For any \(\gamma > 0\), there is a \(\mathcal{P}\)LS \((\mathcal{P}, \mathcal{L})\), \(|\mathcal{P}| \leq N\) such that \(\text{disc}_\gamma(A^{\mathcal{P}, \mathcal{L}}) \geq \Omega(\gamma N^{1/6})\).

Before we continue with our lower bound proof, let us note that Chazelle and Lvov [CL01] also gave an upper bound of \(O(N^{1/6} \log^{2/3} N)\) for the discrepancy of \(N\)-PPLS. This leaves a gap of \(O(\log^{2/3} N)\) compared to the lower bound. Thanks to the recent advancements in discrepancy theory, we observe that a tight upper bound of \(O(N^{1/6})\) can be shown—thereby closing this gap. We provide a proof in Appendix [A.1].

We are now ready to prove our lower bound.

**Proof of Theorem 3.** Let \(N = \lfloor n/2 \rfloor\), and let \((\mathcal{P}, \mathcal{L})\) be the PPLS as in Corollary [25] with \(|\mathcal{P}| \leq N\). Let \(U = \mathcal{P}\) and let \(\prec\) denote the lexicographic ordering of the points, and for every \(L \in \mathcal{L}\), add a single path \(\pi_L\) to \(\Pi\) corresponding to an enumeration of points on \(L\) in the order defined by \(\prec\).

To see that \((U, \Pi, \prec)\) is strongly metrizable, consider the graph \(H = (U, E_H, w_H)\) where there is an edge between all points \(p, p'\) such that there is no other point on the line segment \(pp'\) and the weight of such an edge is \(w_H(p, p') = \|p - p'\|_2^2\). In the graph \(H\), the path \(\pi_L\) for each line \(L\) with starting point \(u\) and end point \(v\) is the unique shortest path of length \(\|u - v\|_2\) between \(u\) and \(v\).

Therefore, by Lemma [23] and Theorem [15] for any \(\beta \in (0, 1)\) and any sufficiently small \(\gamma, \epsilon, \delta > 0\), no \((\epsilon, \delta)\)-DP algorithm for APSD can be \(\alpha\)-accurate w.p. at least \(1 - \beta\) for

\[
\alpha = \Omega \left( \text{disc}_\gamma(A^U) \right) = \Omega \left( \text{disc}_\gamma(A^{\mathcal{P}, \mathcal{L}}) \right) \geq \Omega \left( N^{1/6} \right). \quad \square
\]

5 Algorithm with Both Additive and Multiplicative Error

In this section, we show how to reduce the additive error when we allow multiplicative errors.

5.1 Private Distance Oracles

We start by considering the \(S\)-pair shortest path distances problem and give an improved additive error compared to the algorithms in Section [3.2]. As stated in the Introduction, our algorithm uses an approximate distance oracle due to Thorup and Zwick [TZ05], except that we use the private selection algorithm (Theorem [13]) in each step. We formalize this below for the \(\epsilon\)-DP case.

**Theorem 26.** Let \(k \in \mathbb{N}\) and \(\epsilon > 0\). There is an \(\epsilon\)-DP algorithm for the \(S\)-pair shortest path distances problem that is \(\left(2k - 1, O \left( \frac{k^2 |S|^{1+1/k} \log^2 n}{\epsilon} \right) \right)\)-accurate w.h.p.
Proof. For a subset $T \subseteq V$, we use $\text{dist}_G(v, T)$ to denote $\min_{u \in T} \{\text{dist}_G(v, u)\}$. Let $s = |S|$. Let $q = s^{-1/k}, r = \lceil 10s^{1/k} \log n \rceil$, and $\epsilon' = \frac{\alpha}{srk}$. The algorithm works as follows:

1. Create sets $S = A_0 \subseteq \cdots \subseteq A_{k-1}$, where for each $i \in [k-1]$, each element of $A_{i-1}$ is included in $A_i$ independently with probability $q$.

2. For each $v \in S$:
   (a) Let $B(v) = \emptyset$.
   (b) For each $i \in \{0\} \cup [k-1]$:
      i. For each $j \in [r]$:
         A. Use $\epsilon'$-DP algorithm for selection where the candidate set is $C = A_i \setminus B(v)$ and the scoring function is $f_u \equiv \text{dist}(u, v)$. Let $(\hat{u}, \text{dist}(u, v))$ denote the output. We then add $\hat{u}$ to $B(v)$.
      ii. Let $p_i(v) \equiv \text{argmin}_{u \in A_i \cap B(v)} \{\text{dist}(u, v)\}$, where ties are broken arbitrarily.

3. For each $u, v \in V$, the approximate distance $\hat{\text{dist}}(u, v)$ is calculated as follows:
   (a) Let $u_0 = w_0 = u$ and $v_0 = v$.
   (b) For each $i \in [k-1]$:
      i. If $w_{i-1} \in B(v_{i-1})$, break.
      ii. Otherwise, let $u_i = v_{i-1}, v_i = u_{i-1}$, and $w_i = p_i(u_i)$.
   (c) Let $\hat{\text{dist}}(u, v) \equiv \text{dist}(w, u) + \text{dist}(w, v)$, where $w$ denotes the last $w_i$ that got assigned in the above.

The basic composition theorem implies that our algorithm is $\epsilon$-DP as desired. We will next analyze its utility. Let $r$-$\text{nd}_d(v, A_i)$ denote the distance between $v$ to the $r$th closest vertex in $A_i$. The utility guarantees from Theorem 13 implies that, with probability $1 - O(1/n^3)$, we have

\[
\forall u \in A_i \text{ such that } \text{dist}_G(v, u) \leq r$-$\text{nd}_d(v, A_i) - \alpha, \quad u \in B(v),
\]

and

\[
\forall u \in A_i, \quad |\text{dist}_G(u, v) - \hat{\text{dist}}(u, v)| \leq \alpha,
\]

where $\alpha = O\left(\frac{\log n}{\epsilon'}\right) = O\left(\frac{k^{1+1/k} \log^2 n}{\epsilon}\right)$.

Furthermore, notice that, for all $i \in \{0\} \cup [k-2]$, since each element in $A_i$ is kept in $A_{i+1}$ with probability $q$, we have

\[
\Pr[r$-$\text{nd}_d(v, A_i) > \text{dist}_G(v, A_{i+1})] = (1 - q)^r < 1/n^3.
\]

Therefore, by a union bound, with probability $1 - O(1/n)$ we have

\[
\forall v \in V, i \in \{0\} \cup [k-2], \quad r$-$\text{nd}_d(v, A_i) \leq \text{dist}_G(v, A_{i+1}),
\]

which combined with (4) implies that, for all $i \in \{0\} \cup [k-2]$, we have

\[
\forall u \in A_i \text{ such that } \text{dist}_G(v, u) \leq \text{dist}_G(v, A_{i+1}) - \alpha, \quad u \in B(v).
\]
Similarly, notice that each element is included in $A_{k-1}$ w.p. $q^{-1} = s^{1/k}/s$ independently. Therefore, with probability $1 - O(1/n)$ we have $|A_{k-1}| \leq r$, which means that

$$A_{k-1} \subseteq B(v).$$

Note that this event means that the distance calculation is valid because if we reach the end of the loop (i.e., reaching $i = k - 1$), we will always have $w \in A_{k-1} \subseteq B(u), B(v)$.

Henceforth, we will assume that these events occur.

Now, consider any $u, v \in V$. Since $\tilde{\text{dist}}(u, v) := \hat{\text{dist}}(w, u) + \hat{\text{dist}}(w, v)$ for some $w \in V$, we may lower bound $\tilde{\text{dist}}(u, v)$ by

$$\tilde{\text{dist}}(u, v) \geq \hat{\text{dist}}(w, u) + \hat{\text{dist}}(w, v) - 2\alpha \geq \text{dist}(u, v) - 2\alpha.$$ 

As for the upper bound on $\tilde{\text{dist}}(u, v)$, let $\ell \in [k]$ denote the largest value of $i$ for which $u_i, v_i$ are assigned (before the loop breaks/ends). For any $i \in [\ell]$, since the loop does not break, we have $w_{i-1} \notin B(v_{i-1})$. From (6), this implies that

$$\text{dist}_G(w_{i-1}, v_{i-1}) \geq \text{dist}_G(v_{i-1}, A_i) - \alpha.$$ 

Furthermore, the definition of $p_i(v_{i-1})$ together with (5) implies that

$$\text{dist}_G(w_{i-1}, v_{i-1}) = \text{dist}_G(p_i(v_{i-1}), v_{i-1}) \leq \text{dist}_G(v_{i-1}, A_i) + \alpha.$$ 

Therefore, we have

$$\text{dist}_G(w_i, u_i) = \text{dist}_G(w_i, v_{i-1}) \leq \text{dist}_G(w_{i-1}, v_{i-1}) + 2\alpha \leq \text{dist}_G(w_{i-1}, u_{i-1}) + \text{dist}_G(u_{i-1}, v_{i-1}) + 2\alpha = \text{dist}_G(w_{i-1}, u_i) + \text{dist}_G(u, v) + 2\alpha,$$

where the second inequality follows from the triangle inequality.

As a result, we have

$$\tilde{\text{dist}}(u, v) = \tilde{\text{dist}}(u, w_{\ell}) + \tilde{\text{dist}}(w_{\ell}, v_{\ell})$$

$$= \hat{\text{dist}}(u_{\ell}, w_{\ell}) + \hat{\text{dist}}(v_{\ell}, w_{\ell})$$

$$\leq \text{dist}_G(u_{\ell}, w_{\ell}) + \text{dist}_G(v_{\ell}, w_{\ell}) + 2\alpha$$

(Triangle inequality) \leq 2 \text{dist}_G(u_{\ell}, w_{\ell}) + \text{dist}_G(u, v) + 2\alpha \leq 2 \text{dist}_G(u_0, w_0) + (2\ell + 1) \cdot \text{dist}_G(u, v) + 2(2\ell + 1)\alpha

(Because $w_0 = u_0$) = $(2\ell + 1) \cdot \text{dist}_G(u, v) + 2(2\ell + 1)\alpha \leq (2k - 1) \cdot \text{dist}_G(u, v) + (4k - 2)\alpha.$

Therefore, we can conclude that w.p. $1 - O(1/n)$, the output is $(2k - 1, (4k - 2)\alpha)$-accurate. \[QED\]

The $(\epsilon, \delta)$-DP case is nearly identical except we use advanced composition instead of basic composition.
Theorem 27. Let \( k \in \mathbb{N} \) and \( \epsilon, \delta \in [0, 1) \). There is an \( \epsilon \)-DP algorithm for the \( S \)-pair shortest path distances problem that is 
\[
(2k - 1, O \left( \frac{k^{3/2} \log^{3/2} n \sqrt{|S|^{1/8} \log(1/\delta)}}{\epsilon} \right) ) - \text{accurate w.h.p.}
\]

Proof. The proof is exactly the same as in that of Theorem 26 except that here we use an \( \epsilon'' \)-DP selection algorithm with \( \epsilon'' := \frac{\epsilon}{2\sqrt{2(k+1)\log(2/\delta)}} \). Advanced composition (Theorem 10) implies that the algorithm is \((\epsilon, \delta)\)-DP. As for the error, we now have 
\[
\alpha = O \left( \frac{\log n}{\epsilon''} \right) = O \left( \frac{\sqrt{k}a^{1/6} \log(1/\delta) \log^{3/2} n}{\epsilon} \right),
\]
resulting in the claimed additive error.

5.2 Combining with Input Perturbation Algorithm

By plugging Theorem 26 into Lemma 19 with \( s = (n/k^2)^{k/(2k+1)} \), we immediately arrive at the following, which is a more detailed version of Theorem 5.

Theorem 28. Let \( k \in \mathbb{N} \). For any \( \epsilon > 0 \), there exists an \( \epsilon \)-DP algorithm for APSD that is 
\[
(2k - 1, O \left( \frac{k \cdot n^{(k+1)/(2k+1)} \log^2 n}{\epsilon} \right) ) - \text{accurate w.h.p.}
\]

Similarly, by plugging Theorem 27 into Lemma 19 with \( s = \left( \frac{n \log^{1/2} n}{k^{3/2} \log^{1/2}(1/\delta)} \right)^{2k/(3k+1)} \), we immediately arrive at the following, which is a more detailed version of Theorem 5.

Theorem 29. Let \( k \in \mathbb{N} \). For any \( \epsilon, \delta \in (0, 1) \), there exists an \((\epsilon, \delta)\)-DP algorithm for APSD that is 
\[
(2k - 1, O \left( \frac{k \cdot n^{(k+1)/(3k+1)} \log n^{(5k+2)/(3k+1)} \log(1/\delta)^{k/(3k+1)}}{\epsilon} \right) ) - \text{accurate w.h.p.}
\]

6 Discussion and Open Questions

In this work, we give an algorithm for the APSD problem with error \( \tilde{O}(n^{2/3}/\epsilon) \) in the \( \epsilon \)-DP setting and \( \tilde{O}(\sqrt{n}/\epsilon) \) in the \((\epsilon, \delta)\)-DP setting, together with a lower bound of \( \Omega(n^{1/6}) \) for any sufficiently small \( \epsilon, \delta > 0 \).

An obvious open problem here is to close the gap between the lower and upper bounds. On this front, we note that our lower bound technique cannot go beyond an \( \tilde{O}(n^{1/4}) \) additive error. The reason is that, for any strongly metrizable path system \((U, \Pi)\), \( \mathbf{A}^\Pi \) has VC dimension \( d \leq 2 \), since unique shortest paths cannot “fork and rejoin”. Thus, we can immediately use an algorithm of [MN12] to get an error of \( \tilde{O}_{\epsilon, \delta}(n^{1/2 - 1/(2d)}) \leq O_{\epsilon, \delta}(n^{1/4}) \). It remains interesting whether a lower bound of \( \Omega_{\epsilon, \delta}(n^{1/4}) \) can be shown via our technique. A possibly helpful tool in this direction is the work of [Bod19], which characterizes the set of strongly metrizable path systems based on a family of (infinite) prohibited subgraphs; it may be possible to use this characterization to construct a more elaborate graph with higher discrepancy than the ones we obtain from PPLS.

When a multiplicative approximation is allowed, we give algorithms with improved, but still polynomial in \( n \), additive errors. As mentioned earlier, our lower bound does not apply in this case, and it remains a possibility that an algorithm with, e.g., \( O(1) \) multiplicative approximation and \( \text{polylog}(n) \) additive error exists. Coming up with such an algorithm or refuting its existence is also an interesting future direction.

\footnotesize{In fact, [MN12] gives a root mean square error guarantee. However, one may use the boosting framework of [DRV10] to achieve a similar (up to polylogarithmic factors) bound for the maximum error.}
References

[ABCP98] Baruch Awerbuch, Bonnie Berger, Lenore Cowen, and David Peleg. Near-linear time construction of sparse neighborhood covers. *SIAM J. Comput.*, 28(1):263–277, 1998.

[Abo18] John M Abowd. The US Census Bureau adopts differential privacy. In *KDD*, pages 2867–2867, 2018.

[App17] Apple Differential Privacy Team. Learning with privacy at scale. *Apple Machine Learning Journal*, 2017.

[BC87] Jozsef Beck and William W. L. Chen. *Irregularities of Distribution*. Cambridge University Press, 1987.

[Bod19] Greg Bodwin. On the structure of unique shortest paths in graphs. In *SODA*, pages 2071–2089, 2019.

[Cha01] Bernard Chazelle. *The Discrepancy Method - Randomness and Complexity*. Cambridge University Press, 2001.

[CL01] Bernard Chazelle and Alexey Lvov. A trace bound for the hereditary discrepancy. *Discret. Comput. Geom.*, 26(2):221–231, 2001.

[Coh98] Edith Cohen. Fast algorithms for constructing $t$-spanners and paths with stretch $t$. *SIAM J. Comput.*, 28(1):210–236, 1998.

[CSS11] T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, 2011.

[DK20] Yuval Dagan and Gil Kur. A bounded-noise mechanism for differential privacy. *CoRR*, abs/2012.03817, 2020.

[DKM+06] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *EUROCRYPT*, pages 486–503, 2006.

[DKY17] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In *NeurIPS*, pages 3571–3580, 2017.

[DL09] Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *STOC*, pages 371–380, 2009.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, pages 265–284, 2006.

[DNPR10] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *STOC*, pages 715–724, 2010.

[DNRR15] Cynthia Dwork, Moni Naor, Omer Reingold, and Guy N. Rothblum. Pure differential privacy for rectangle queries via private partitions. In *ASIACRYPT*, pages 735–751, 2015.
A Discrepancy Bounds for Planar Point-Line Systems

A.1 Tight Upper Bound

As mentioned earlier, Chazelle and Lvov [CL01] prove an upper bound of $O(N^{1/6} \log^{2/3} N)$ on the discrepancy of any $N$-PPLS. Below, we improve this upper bound to $O(N^{1/6})$, which is tight in light of the lower bound in [CL01].

**Theorem 30.** For any $(N, D)$-PPLS $(\mathcal{P}, \mathcal{L})$, we have $\text{disc}(A^{\mathcal{P}, \mathcal{L}}) \leq O(N^{1/6})$. 

The proof of Theorem 30 is similar to that of [CL01], which uses the Szemerédi–Trotter theorem [ST83] (stated below) that bounds the number of lines that pass through many points; the main difference is that we will use the partial coloring result of [LM15] (also stated below) which is stronger than that in the original proof of [CL01].

**Theorem 31** (Szemerédi–Trotter [ST83]). There exists a constant \( c \geq 1 \) such that, for any \( N, k \in \mathbb{N} \) with \( k \geq 2 \) and \( N \) points \( x_1, \ldots, x_N \in \mathbb{R}^2 \), the number of lines that pass through at least \( k \) points is at most \( c(N^2/k^3 + N/k) \).

**Theorem 32** (Partial Coloring [LM15]). Let \( v_1, \ldots, v_m \in \mathbb{R}^n \) be vectors, \( x^{\text{start}} \in [-1,+1]^n \) be the “starting” vector and \( c_1, \ldots, c_m > 0 \) be thresholds.\(^{10}\) If \( \sum_{j \in [m]} \exp(-c_j^2/16) \leq n/16 \), then there exists a point \( x^{\text{end}} \in [-1,+1]^n \) such that

\[
(i). \quad |\langle x^{\text{end}} - x^{\text{start}}, v_j \rangle| \leq c_j \|v_j\|_2 \quad \text{for all } j \in [m],
\]

\[
(ii). \quad x_i^{\text{end}} \in \{\pm 1\} \quad \text{for at least } n/2 \text{ indices } i \in [n].
\]

Compared to the partial coloring result used in [CL01] (proved in [BC87, Cha01]), Theorem 32 is sharper in that it allows for \( O(n) \) “stringent” constraints where \( c_j = o(1) \), whereas the version used in [CL01] only allows \( O(n/\log n) \) such constraints. When it comes to Szemerédi–Trotter theorem, this means that [CL01] can only let the lines with at least \( k \geq \Theta(N^{1/3} \log^{1/3} N) \) points be these “stringent” constraints; on the other hand, Theorem 32 allows us to take \( k = \Theta(N^{1/3}) \). This is indeed the reason for the improvement achieved here.

**Proof of Theorem 30** We construct the final vector \( x \) via the standard partial coloring approach as follows:

- Let \( t = 0 \) and \( x^0 = (0, \ldots, 0) \) be the all-zeros vector in \( \mathbb{R}^N \).
- While \( x^t \notin \{\pm 1\}^N \):
  - Let \( I_t = \{i \in [N] \mid x_i^t \notin \{\pm 1\}\} \) denote the set of fractional coordinates, let \( P_t = \{p_i \mid i \in I_t\} \) be the set of corresponding points, and let \( n_t = |I_t| \).
  - Let \( L_t \) denote the set of lines that pass through at least two points in \( P_t \). For each \( L_j \in L_t \), let \( v_j \) denote the \( j \)th row in \( A_{P_t} \). Furthermore, let \( c_j = \frac{12800 \cdot n_t^{1/6}}{\|v_j\|_2} \), where \( c \) is the constant from Theorem 31.
  - Apply Theorem 32 on \( \{v_j\}_{L_j \in L_t} \) with the starting vector \( x^{\text{start}} = x_j^t \) and thresholds \( \{c_j\}_{L_j \in L_t} \) to get \( x^{\text{end}} \in [-1,+1]^t \).
  - Finally, let \( x^{t+1} \) be such that
    \[
    x_j^{t+1} = \begin{cases} 
    x_j^t & \text{if } j \notin I_t, \\
    x_j^{\text{end}} & \text{otherwise}.
    \end{cases}
    \]

\(^{10}\) Note that the original formulation of [LM15] also has an “approximation error” parameter \( \delta \) and the guarantee is only that \( |x_i^{\text{end}}| \geq 1 - \delta \) for more than half of the coordinates. However, we may derive our formulation by simply halving each of \( c_j \) and picking \( \delta = \max\{c_1, \ldots, c_m\}/n \). The reason their work contains \( \delta \) is mainly due to the algorithmic aspect, i.e., the running time grows as \( (1/\delta)^{O(1)} \). Since we are not interested in computational complexity, we choose to discard \( \delta \) from the formulation.
- \( t = t + 1 \).

- Let \( x = x^t \) be the coloring.

Let us start by checking that the invocation of Theorem 32 is valid, i.e., that \( \sum_{L_j \in \mathcal{L}^t} \exp\left(-\frac{c_t^j}{16}\right) \leq n^t/16 \). Let \( q = \lceil 64cn_t^{1/3} \rceil \). We may bound the sum using the Szemerédi–Trotter theorem as follows:

\[
\sum_{L_j \in \mathcal{L}^t} \exp\left(-\frac{c_t^j}{16}\right) = \sum_{L_j \in \mathcal{L}^t} \exp\left(-\frac{800 \cdot cn_t^{1/6}}{\|v^t_j\|_2}\right)
\]

\[
= \sum_{k=2}^{n_t} \sum_{L_j \in \mathcal{L}^t, |L_j \cap P_t| = k} \exp\left(-\frac{800 \cdot cn_t^{1/6}}{\sqrt{k}}\right)
\]

\[
= \sum_{k=q}^{n_t} \sum_{L_j \in \mathcal{L}^t, |L_j \cap P_t| = k} \exp\left(-\frac{800 \cdot cn_t^{1/6}}{\sqrt{k}}\right) + \sum_{k=2}^{q-1} \sum_{L_j \in \mathcal{L}^t, |L_j \cap P_t| = k} \exp\left(-\frac{800 \cdot cn_t^{1/6}}{2(s+1)/2}\right)
\]

\[
\leq \left| \{ L_j \in \mathcal{L}^t \mid |L_j \cap P_t| \geq q \} \right| + \sum_{s=1}^{\log(q-1)} \sum_{L_j \in \mathcal{L}^t, |L_j \cap P_t| \geq 2^s} \exp\left(-\frac{800 \cdot cn_t^{1/6}}{2(s+1)/2}\right)
\]

\[
= \frac{n_t}{32} + \sum_{s=1}^{\log(q-1)} 4097 \cdot \frac{n_t^2}{2^{3s}} \cdot \exp\left(-560 \cdot \frac{n_t^{1/6}}{2^s}\right).
\]

Now, let \( a(s) = 70cn_t^{1/6}/2^s/2 \). For \( s \leq \log(q-1) \), we have \( a(s) \geq 70cn_t^{1/6}/\sqrt{q-1} \geq 1 \). From this, we have \( e^{a(s)} \geq a(s) \). In other words, \( e^{-a(s)} \leq 1/a(s) \). Plugging this back into the above, we have

\[
\sum_{L_j \in \mathcal{L}^t} \exp\left(-\frac{c_t^j}{16}\right) \leq \frac{n_t}{32} + \sum_{s=1}^{\log(q-1)} 4097 \cdot \frac{n_t^2}{2^{3s}} \cdot \left( \frac{1}{70cn_t^{1/6}/2^s/2} \right)^8.
\]

\[
\leq \frac{n_t}{32} + \frac{n_t^{2/3}}{10^6c} \cdot \sum_{s=1}^{\log(q-1)} 2^s
\]

\[
\leq \frac{n_t}{32} + \frac{n_t^{2/3}}{10^6c} \cdot 2^{\log(q-1)} + 1
\]

\[
\leq \frac{n_t}{32} + \frac{n_t^{2/3}}{10^6c} \cdot 128 \cdot cn_t^{1/3}
\]

\[
\leq \frac{n}{16},
\]

as desired.
As for the discrepancy, we have
\[
\|A^{P,L}x\|_{\infty} = \max_{L_j \in L} \langle v_j^0, x \rangle \\
\leq \max_{L_j \in L} \left( 1 + \sum_{t \geq 0} \langle v_j^t, x^{t+1} - x^t \rangle \right)
\]
(Theorem 32) \leq \max_{L_j \in L} \left( 1 + \sum_{t \geq 0} O \left( n_t^{1/6} \right) \right)
\leq O(N^{1/6}).

Note that the extra 1 on the second line comes from the fact that we only include in \(L^t\) lines that pass through at least two points in \(P_t\). 

A.2 Lower Bound for \(\gamma < 1\)

In this section we prove a lower bound for discrepancy of partial coloring, i.e., for \(\gamma < 1\).

Proof of Corollary 25 Let \(c_1 > 0\) be the constant from Theorem 24 such that for all \(N \in \mathbb{N}\), there is an \(N\)-PPLS \((P, L)\) such that \(\text{disc}(A^{P,L}) \geq c_1 N^{1/6}\). Furthermore, let \(c_2 > 0\) be the constant from Theorem 30 such that for all \(N \in \mathbb{N}\), for any \(N\)-PPLS \((P', L')\) we have \(\text{disc}(A^{P',L'}) \leq c_2 N^{1/6}\).

Let \(c = \frac{c_1}{12} \cdot \frac{\log(1/(1-\gamma))}{\log(2c_2/c_1)}\). We will show that for all \(N \in \mathbb{N}\), there exists a PPLS \((P^*, L^*)\) with \(|P^*| \leq N\) such that \(\text{disc}_\gamma(A^{P^*,L^*}) \geq c N^{1/6}\). Suppose for the sake of contradiction that there exists \(N \in \mathbb{N}\) such that for all PPLS \((P^*, L^*)\) with \(|P^*| \leq N\), we have \(\text{disc}_\gamma(A^{P^*,L^*}) < c N^{1/6}\).

Let \((P, L)\) be the \(N\)-PPLS guaranteed by Theorem 24. Let \(\tilde{N} = N(0.5c_1/c_2)^{6}\). Consider the following recursive coloring algorithm for \((P, L)\):

- Let \(t = 0\) and \(P_0 = P\).
- While \(|P_t| > \tilde{N}\):
  - Apply the hypothesis on \((P_t, L)\) to get a partial coloring \(x^t \in \{-1, 0, +1\}^{P_t}\) such that \(\|x^t\|_1 \geq \gamma |P_t|\) and \(\|A^{P_t,L}x^t\|_{\infty} \leq c N^{1/6}\).
  - Let \(P_{t+1} := \{p \in P_t \mid x^t_p = 0\}\).
  - \(t = t + 1\).
- Apply Theorem 30 on \((P_t, L)\) to get a coloring \(x^t \in \{\pm 1\}^{P_t}\) such that \(\|A^{P_t,L}x^t\|_{\infty} \leq c_2 \tilde{N}^{1/6}\).
- Let \(x\) be the coloring generated above, i.e., \(x_p = x_p^t\) for all \(p \in P_t \setminus P_{t+1}\).

Let \(T\) denote the number of iterations executed by the algorithm. Since \(|P_t|\) reduces by a factor of \((at least) 1/(1-\gamma)\) each time, we have \(T \leq \frac{\log(N/\tilde{N})}{\log(1/(1-\gamma))} \leq \frac{6\log(2c_2/c_1)}{\log(1/(1-\gamma))}\). Therefore, we have
\[
\|A^{P,L}x\|_{\infty} < T \cdot c N^{1/6} + c_2 \tilde{N}^{1/6} \leq 0.5c_1 N^{1/6} + 0.5c_1 N^{1/6} = c_1 N^{1/6},
\]
which contradicts our choice of \((P, L)\).

\[\square\]