Triangle Counting with Local Edge Differential Privacy

Talya Eden ✉ 🏠 📵
Bar Ilan University, Ramat Gan, IL

Quanquan C. Liu ✉ 🏠 📵
Northwestern University, Evanston, IL, US

Sofya Raskhodnikova ✉ 🏠 📵
Boston University, MA, US

Adam Smith ✉ 🏠 📵
Boston University, MA, US

──── **Abstract** ────

Many deployments of differential privacy in industry are in the local model, where each party releases its private information via a differentially private randomizer. We study triangle counting in the noninteractive and interactive local model with edge differential privacy (that, intuitively, requires that the outputs of the algorithm on graphs that differ in one edge be indistinguishable). In this model, each party's local view consists of the adjacency list of one vertex.

In the noninteractive model, we prove that additive $\Omega(n^2)$ error is necessary, where n is the number of nodes. This lower bound is our main technical contribution. It uses a reconstruction attack with a new class of linear queries and a novel mix-and-match strategy of running the local randomizers with different completions of their adjacency lists. It matches the additive error of the algorithm based on Randomized Response, proposed by Imola, Murakami and Chaudhuri (USENIX2021) and analyzed by Imola, Murakami and Chaudhuri (CCS2022) for constant ε. We use a different postprocessing of Randomized Response and provide tight bounds on the variance of the resulting algorithm.

In the interactive setting, we prove a lower bound of $\Omega(n^{3/2})$ on the additive error. Previously, no hardness results were known for interactive, edge-private algorithms in the local model, except for those that follow trivially from the results for the central model. Our work significantly improves on the state of the art in differentially private graph analysis in the local model.

1 Introduction

Triangle counting is a fundamental primitive in graph analysis, used in numerous applications and widely studied in different computational models [3, 12, 26, 37, 7, 44, 45, 48, 50]. Statistics based on triangle counts reveal important structural information about networks (as discussed, e.g., in [30, 46, 51]). They are used to perform many computational tasks on social networks, including community detection [49], link prediction [25], and spam filtering [5]. See [1] for a survey on algorithms for and applications of triangle counting.

In applications where a graph (e.g., a social network) holds sensitive information, the algorithm that computes on the graph has to protect personal information, such as friendships between specific individuals. Differential privacy [22] has emerged as the standard of rigorous privacy guarantees. See [53] for a survey of differentially private graph analysis. The most investigated setting of differential privacy is called the *central model*. It implicitly assumes a curator that collects all the data, computes on it, and provides data releases. In some situations, however, it might be undesirable to collect all information in one place, for instance, because of trust or liability issues. To address this, the *local model* of differential privacy was proposed [29, 22, 40] and is now used in many industry deployments [28, 8, 14, 2, 17].

In this model, each party releases its private information via a differentially private randomizer. Then the algorithm processes the information and, in the case of the local *noninteractive* model, outputs the answer. In the case of the local *interactive* model, the algorithm may have multiple rounds where it asks all parties to run different randomizers on their private data. These randomizers can have arbitrary dependencies on previous messages. Differential privacy in the local model is defined with respect to the whole transcript of interactions between the parties and the algorithm. In the local model applied to graph data, each vertex represents a party. It receives the list of its neighbors as input and applies local randomizers to it. In contrast to the typical datasets, where information belongs to individual parties, in the graph setting, each pair of parties (vertices) share the information of whether there is an edge between them.

Differential privacy, intuitively, guarantees that, for any two neighboring datasets, the output distributions of the algorithm are roughly the same. There are two natural notions of neighboring graphs: *edge-neighboring* and *node-neighboring*. Two graphs are edge-neighboring if they differ in one edge; they are node-neighboring if they differ in one node and its adjacent edges. Edge differential privacy is, in general, easier to attain, but node differential privacy provides stronger guarantees. Edge differential privacy was introduced and first applied to triangle counting in [47]. The edge-differentially private algorithm from [47] was generalized and implemented in [38]. The first node-differentially private algorithms appeared in [9, 41, 13], and all three of these articles considered the problem of triangle counting. Edge differential privacy in the local model has been studied in [52, 31, 54, 56, 33, 34, 16] with most of the listed articles focusing on triangle counting.

In this work, we investigate edge differentially private algorithms for estimating the number of triangles in a graph in the local model. Our goal is to understand the additive error achievable by such algorithms both in the noninteractive and in the interactive model. For the noninteractive model, we provide upper and lower bounds on additive error. Our bounds are tight in terms of n, the number of nodes in the input graph. For the interactive model, we provide the first lower bound specific to local, edge differentially private (LEDP) algorithms. There are trivial lower bounds for the central model (based on global sensitivity) which apply to the local model, but no lower bounds specific to the local model were previously known for any graph problem, even for 2-round algorithms. Together, our results improve our understanding of both noninteractive and interactive LEDP algorithms.

1.1 Results

Our results and comparison to previous work are summarized in Table 1.

■ **Table 1** Summary of lower and upper bounds on the additive error for triangle counting in the noninteractive and interactive models. Note that the largest value of $C_4(G)$ is $\binom{n}{4} = \Theta(n^4)$. For ease of comparison, the results of [33] and [34] are stated for graphs with $d_{max} = \Theta(n)$.

Model		Previous Results		Our Results	
Non-interactive	Lower Bound	$\Omega(n^{3/2})$	[33]	$\Omega(n^2)$	Thm. 1.1
	Upper Bound	$O(n^2)$ (constant ε)	[35]	$O\left(\frac{\sqrt{C_4(G)}}{\varepsilon} + \frac{n^{3/2}}{\varepsilon^3}\right)$	Thm. 1.2
Interactive	Lower Bound	$\Omega(n)$	easy	$\Omega(\frac{n^{3/2}}{\varepsilon})$	Thm. 1.3
	Upper Bound	$O\left(\frac{\sqrt{C_4(G)}}{\varepsilon} + \frac{n^{3/2}}{\varepsilon^2}\right)$	[34]		

1.1.1 Lower Bound for the Noninteractive Local Model

Our main technical contribution is a lower bound in the noninteractive setting. It uses a reconstruction attack (for the central model) with a new class of linear queries and a novel mix-and-match strategy of running local randomizers with different completions of their adjacency lists. While reconstruction attacks are a powerful tool in proving lower bounds in the central model of differential privacy, they have not been used to obtain bounds in the local model. Previous lower bounds in the local model are based on quite different techniques – typically, information-theoretic arguments (see, for example, [40, 6, 19] and many subsequent works).

▶ **Theorem 1.1** (Noninteractive Lower Bound, informal version). *Let $\varepsilon \in (0, 1/20)$ and $\delta \geq 0$ be a sufficiently small constant. There exists a family of graphs such that every noninteractive (ε, δ)-local edge differentially private algorithm that gets an n-node graph from the family as input and approximates the number of triangles in the graph within additive error at most α (with sufficiently high constant probability) must have $\alpha = \Omega(n^2)$.*

Our lower bound holds for all small $\delta \geq 0$ (the case referred to as "approximate" differential privacy). Observe that such lower bounds are stronger than those for $\delta = 0$ (the case referred to as "pure" differential privacy), because they include $\delta = 0$ as a special case. The only previously known lower bound, due to Imola et al. [33], showed that noninteractive algorithms must have error $\Omega(\sqrt{n} \cdot d_{max})$.

To prove the lower bound in Theorem 1.1, we develop a novel mix-and-match technique for noninteractive local model. For a technical overview of the proof of Theorem 1.1, see Section 1.2.

Our lower bound matches the upper bound of $O(n^2)$ proved by [35, Theorem G.3] (for constant ε) for an algorithm based on randomized response. In this work, we give a simpler variant of the algorithm and a more refined analysis, which works for all ε.

1.1.2   Tight Analysis of Randomized Response

The most natural algorithm for the noninteractive model is Randomized Response, which dates back to Warner [55]. In this algorithm, each bit is flipped with probability $\frac{1}{e^\varepsilon+1}$, where ε is the privacy parameter. In the case of graphs, each bit represents a presence or absence of an edge. An algorithm based on Randomized Response for triangle counting was first analyzed by [33] for the special case of Erdős-Rényi graphs, and then [35] proved that this algorithm has $O(n^2)$ additive error for constant ε for general graphs. These works first compute the number of triangles and other induced subgraphs with three vertices as though the noisy edges are real edges and then appropriately adjust the estimate using these counts to make it unbiased.

We use a different postprocessing of Randomized Response. We rescale the noisy edges right away, so we need not compute counts for graphs other than triangles, which makes the analysis much simpler. We obtain tight upper and lower bounds on the variance of the resulting algorithm that hold for all ε. Our bounds are more refined, as they are stated in terms of $C_4(G)$, the number of four cycles in the graph.

▶ **Theorem 1.2** (Analysis of Randomized Response). *For all $\varepsilon > 0$, there exists a noninteractive ε-LEDP algorithm based on Randomized Response that gets an n-node graph as input and returns an unbiased estimate $\hat{T}$ of the number of triangles in a graph that has variance $\Theta\left(\frac{C_4(G)}{\varepsilon^2} + \frac{n^3}{\varepsilon^6}\right)$.*

In particular, with high constant probability, $\hat{T}$ has additive error $\alpha = O\left(\frac{\sqrt{C_4(G)}}{\varepsilon} + \frac{n^{3/2}}{\varepsilon^3}\right)$.

Note that for constant ε, Theorem 1.2 implies an upper bound of $O(n^2)$ on the additive error of the algorithm's estimate. Thus, Randomized Response is optimal for graphs that have $C_4 = \Theta(n^4)$ by our lower bound in Theorem 1.1. Also, observe that Randomized Response achieves pure differential privacy (with $\delta = 0$), whereas the lower bound in Theorem 1.1 holds even for approximate differential privacy. Even though allowing $\delta > 0$ results in better accuracy for many problems, it does not give any additional utility for noninteractive triangle counting. The proof of Theorem 1.2 is deferred to the full version.

1.1.3   Lower Bound for the Interactive Local Model

Next, we investigate triangle counting in the interactive setting. Imola et al. [33] present an ε-LEDP for triangle counting in the interactive model with additive error of $O\left(\sqrt{C_4(G)}/\varepsilon + \sqrt{n} \cdot d_{max}/\varepsilon^2\right)$, where d_{max} is an upper bound on the maximum degree.

We give a lower bound on the additive error of LEDP algorithms for triangle counting in the interactive model. Note that $\Omega(n)$ additive error is unavoidable for triangle counting even in the central model, because the (edge) global sensitivity of the number of triangles is $n - 2$ (and this lower bound is tight in the central model). There were no previously known lower bounds for this problem (or any other problem on graphs) specific to the interactive LEDP model that applied to even 2-round algorithms. Our lower bound applies to interactive algorithms with *any* number of rounds.

▶ **Theorem 1.3** (Interactive Lower Bound). *There exist a family of graphs and a constant $c > 0$ such that for every $\varepsilon \in (0,1)$, $n \in \mathbb{N}, \alpha \in (0, n^2]$ and $\delta \in \left[0, \frac{1}{10^5} \cdot \frac{\varepsilon^3 \alpha_0^2}{n^5 \ln(n^3/\varepsilon\alpha_0)}\right]$, every (potentially interactive) (ε, δ)-local edge differentially private algorithm that gets an n-node graph from the family as input and approximates the number of triangles in the graph with additive error at most α (with probability at least $2/3$) must have $\alpha \geq c \cdot \frac{n^{3/2}}{\varepsilon}$.*

Our lower bound is obtained via a reduction from the problem of computing the summation of n randomly sampled bits in $\{0,1\}$ in the LDP model, studied in a series of works [6, 10, 20, 36]. Our lower bound matches the upper bound of [33] for constant ε and for graphs where $d_{max} = \Theta(n)$ and $C_4(G) = O(n^3)$. It is open whether additive error of $o(n^2)$ can be achieved for general graphs.

1.2 Technical Overview of the Noninteractive Lower Bound

Typical techniques for proving lower bounds in the local model heavily rely on two facts that hold for simpler datasets: first, each party's information is not seen by other parties; second, arbitrary changes to the information of one party have to be protected. Both of these conditions fail for graphs in the LEDP model: each edge is shared between two parties, and only changes to one edge are protected in the strong sense of neighboring datasets, imposed by differential privacy.

To overcome these difficulties, we develop a new lower bound method, based on reconstruction attacks in the central model. Such attacks use accurate answers to many queries to reconstruct nearly all the entries of a secret data set [18, 23, 24, 42, 43, 15]. They are usually applied to algorithms that release many different values. However, a triangle-estimation algorithm returns a single number. Consider a naïve attempt to mount an attack using the algorithm as a black box, that is, by simulating every query using a separate invocation of the triangle counting algorithm. This would require us to run the local randomizers many times, degrading their privacy parameters and making a privacy breach vacuous.

To overcome this difficulty, in our attack, we use the noninteractive triangle-estimation algorithm as a *gray box*. Since the algorithm is noninteractive, it is specified by local randomizers for all vertices and a postprocessing algorithm that runs on the outputs of the randomizers. We use a secret dataset X to create a secret subgraph, run the randomizers for the vertices in the secret subgraph only twice, and publish the results. By properties of the randomizers and by composition, the resulting procedure is differentially private. In the next phase, we postprocess the published information to complete the secret subgraph to different graphs corresponding to the queries needed for our attack. Then we feed these graphs to the triangle approximation algorithm, except that for the vertices in the secret subgraph, we rely only on the published outputs. If the triangle counting algorithm is accurate, we get accurate answers to our queries. Even though the randomness used to answer different queries is correlated, we show that a good approximation algorithm for triangle counts allows us to get most of the queries answered correctly. Finally, we use a novel anti-concentration bound (Lemma 1.4, below) to demonstrate that our attack succeeds in reconstructing most of the secret dataset with high probability. This shows that the overall algorithm we run in this process is not differentially private, leading to the conclusion that a very accurate triangle counting algorithm cannot exist in the noninteractive LEDP setting.

We call the queries used in our attack *outer-product queries*. The queries are linear, but their entries are dependent. To define this class of queries, we represent the secret dataset X with n^2 bits as an $n \times n$ matrix. An outer-product query to X specifies two vectors A and B of length n with entries in $\{-1,1\}$ and returns $A^T X B$, that is, $\sum_{i,j \in [n]} A_i X_{ij} B_j$.

To analyze our reconstruction attack, we prove the following anti-concentration bound for random outer-product queries, which might be of independent interest.

▶ **Lemma 1.4** (Anti-concentration for random outer-product queries). *Let M be an $n \times n$ matrix with entries $M_{ij} \in \{-1,0,1\}$ for all $i,j \in [n]$ and m be the number of nonzero entries in M. Let A and B be drawn uniformly and independently from $\{-1,1\}^n$. If $m \geq \gamma n^2$ for some constant γ, then*

$$\Pr\left[|A^T M B| > \frac{\sqrt{m}}{2}\right] \geq \frac{\gamma^2}{16}.$$

The literature on reconstruction attacks describes other classes of dependent queries [42]; the outer-product queries arising here required a new and qualitatively different analysis.

1.3   Additional Related Work

One of the difficulties with proving lower bounds in the local model is that Randomized Response, despite providing strong privacy guarantees, supplies enough information to compute fairly sophisticated statistics. For example, Gupta, Roth and Ullman [32] show how the output of Randomized Response can be used to estimate the density of all cuts in a graph. Karwa et al. [39] show how to fit exponential random graph models based on randomized response output. For certain model families, this would entail estimation of the number of triangles; however, they provide no theoretical error analysis, only experimental evidence for convergence. Randomized Response has also been studied in the statistics literature with a focus on small probabilities of flipping an edge. Balachandran et al. [4] analyze the distribution of the naive estimator that counts the number of triangles in the randomized responses (when flip probabilities are very low). Chang et al. [11] give estimation strategies for settings where the flip rate is unknown but multiple replicates with independent noise are available. To the best of our understanding, these works do not shed light on the regime most relevant to privacy, where edge-flip probabilities are close to $1/2$.

A number of works have looked at triangle counting and other graph problems in the empirical setting [54, 52, 31, 56] in "decentralized" privacy models. In all but [54], the local view consists of the adjacency list. The local views in Sun et al. [54] consist of two-hop neighborhoods. Such a model results in less error since nodes can see all of their adjacent triangles and can report their adjacent triangles using the geometric mechanism.

1.4   Organization

Various models of differential privacy, including LEDP, are defined in Section 2. Our proof of the lower bound for the noninteractive model, Theorem 1.1, appears in Section 3. The anti-concentration lemma for out-product queries (Lemma 1.4) is proved in Section 3.2. Our analysis of Randomized Response and the proof of Theorem 1.2 appears in the full version [27]. The proof of Theorem 1.3 for the interactive LEDP model appears in Section 4.

2   Background on Differential Privacy

We begin with the definition of differential privacy that applies to datasets represented as vectors as well as to graph datasets.

▶ **Definition 2.1** (Differential Privacy [22, 21])**.** *Let $\varepsilon > 0$ and $\delta \in [0, 1)$. A randomized algorithm $\mathcal{A}$ is (ε, δ)-differentially private (DP) (with respect to the neighbor relation on the universe of the datasets) if for all events S in the output space of $\mathcal{A}$ and all neighboring datasets X and X',*

$$\Pr[\mathcal{A}(x) \in S] \leq \exp(\varepsilon) \cdot \Pr[\mathcal{A}(X') \in S] + \delta.$$

When $\delta = 0$, the algorithm is ε-differentially private (sometimes also called "purely differentially private").

Differential privacy can be defined with respect to any notion of neighboring datasets. When datasets are represented as vectors, datasets X and Y are consider neighbors if they differ in one entry. In the context of graphs, there are two natural notions of neighboring graphs that can be used in the definition: edge-neighboring and node-neighboring. We use predominantly the former, but define both to make discussion of previous work clear.

▶ **Definition 2.2.** *Two graphs $G = (V, E)$ and $G' = (V', E')$ are **edge-neighboring** if G and G' differ in exactly one edge, that is, if $V = V'$ and E and E' differ in exactly one element. Two graphs are **node-neighboring** if one can be obtained from the other by removing a node and its adjacent edges.*

If the datasets are graphs with edge (respectively, node) neighbor relationship, we call a differentially private algorithm simply *edge-private* (respectively, *node-private*).

2.1 The local model

The definition of differential privacy implicitly assumes a trusted curator that has access to the data, runs a private algorithm on it, and releases the result. This setup is called the *central model* of differential privacy. In contrast, in the *local model* of differential privacy, each party participating in the computation holds its own data. The interaction between the parties is coordinated by an algorithm $\mathcal{A}$ that accesses data via *local randomizers*. A local randomizer is a differentially private algorithm that runs on the data of one party. In the context of graph datasets, the input graph is distributed among the parties as follows: each party corresponds to a node of the graph and its data is the corresponding row in the adjacency matrix of the graph. In each round of interaction, the algorithm $\mathcal{A}$ assigns each party a local randomizer (or randomizers) that can depend on the information obtained in previous rounds.

We adapt the definition of local differential privacy from [36, 40] to the graph setting. Consider an undirected graph $G = ([n], E)$ represented by an $n \times n$ adjacency matrix A. Each party $i \in [n]$ holds the i-th row of A, denoted $\mathbf{a}_{i*}$. We sometimes refer to $\mathbf{a}_{i*}$ as the *adjacency vector* of party i. Entries of A are denoted a_{ij} for $i, j \in [n]$.

▶ **Definition 2.3** (Local Randomizer). *Let $\varepsilon > 0$ and $\delta \in [0, 1)$. An (ε, δ)-**local randomizer** $R : \{0, 1\}^n \to \mathcal{Y}$ is an (ε, δ)-edge DP algorithm that takes as input the set of neighbors of one node, represented by an adjacency vector $\mathbf{a} \in \{0, 1\}^n$. In other words, $\Pr[R(\mathbf{a}) \in Y] \leq e^\varepsilon \cdot \Pr[R(\mathbf{a}') \in Y] + \delta$ for all $\mathbf{a}$ and $\mathbf{a}'$ that differ in one bit and all sets of outputs $Y \subseteq \mathcal{Y}$. The probability is taken over the random coins of R (but* not *over the choice of the input). When $\delta = 0$, we say that R is an ε-local randomizer.*

A randomized algorithm $\mathcal{A}$ on a distributed graph is (ε, δ)-LEDP if it satisfies Definition 2.4.

▶ **Definition 2.4** (Local Edge Differential Privacy). *A **transcript** π is a vector consisting of 5-tuples $(S_U^t, S_R^t, S_\varepsilon^t, S_\delta^t, S_Y^t)$ – encoding the set of parties chosen, set of randomizers assigned, set of randomizer privacy parameters, and set of randomized outputs produced – for each round t. Let S_π be the collection of all transcripts and S_R be the collection of all randomizers. Let $\perp$ denote a special character indicating that the computation halts. An **algorithm** in this model is a function $\mathcal{A} : S_\pi \to (2^{[n]} \times 2^{S_R} \times 2^{\mathbb{R}^{\geq 0}} \times 2^{\mathbb{R}^{\geq 0}}) \cup \{\perp\}$ mapping transcripts to sets of parties, randomizers, and randomizer privacy parameters. The length of the transcript, as indexed by t, is its round complexity.*

Given $\varepsilon \geq 0$ and $\delta \in [0, 1)$, a randomized algorithm $\mathcal{A}$ on a (distributed) graph G is (ε, δ)**-locally edge differentially private (LEDP)** *if the algorithm that outputs the entire transcript generated by $\mathcal{A}$ is (ε, δ)-edge differentially private on graph G. When $\delta = 0$, we say that $\mathcal{A}$ is an ε-LEDP.*

If $t = 1$, that is, if there is only one round, then $\mathcal{A}$ is called **noninteractive**. *Otherwise, $\mathcal{A}$ is called* **interactive***.*

Observe that a noninteractive LEDP algorithm is specified by a local randomizer for each node and a postprocessing algorithm $\mathcal{P}$ that takes the outputs of the local randomizers as input.

We use a local algorithm known as *randomized response*, initially due to [55], but since adapted to differential privacy [40].

▶ **Definition 2.5** (Randomized Response)**.** *Given a privacy parameter $\varepsilon > 0$ and a k-bit vector* **a***, the algorithm* $\mathsf{RandomizedResponse}_\varepsilon(\mathbf{a})$ *outputs a k-bit vector, where for each $i \in [k]$, bit i is a_i with probability $\frac{e^\varepsilon}{e^\varepsilon + 1}$ and $1 - a_i$ otherwise.*

▶ **Theorem 2.6** (Randomized Response is ε-LR)**.** *Randomized response is an ε-local randomizer.*

Additional privacy tools are described in the full version of this paper.

3   The Noninteractive Lower Bound

In this section, we prove Theorem 1.1, which we restate formally here.

▶ **Theorem 3.1.** *There exists a family of graphs, such that every noninteractive (ε, δ)-LEDP algorithm with $\varepsilon \in (0, \frac{1}{20})$ and $\delta \in [0, \frac{1}{100})$ that gets an n-node graph from the family as an input and approximates the number of triangles in the graph within additive error α with probability at least $1 - \frac{1}{3^6 \cdot 2^7}$, must have $\alpha = \Omega(n^2)$.*

At a high level, the lower bound is proved by showing that a noninteractive local algorithm for counting triangles can be used to mount a reconstruction attack on a secret dataset X in the central model of differential privacy. A groundbreaking result of Dinur and Nissim [18] – generalized in subsequent works [23, 24, 42, 43, 15] – shows that if an algorithm answers too many random linear queries on a sensitive dataset of N bits too accurately then a large constant fraction of the dataset can be reconstructed. This is referred to as a "reconstruction attack". Specifically, Dinur and Nissim show that N random linear queries answered to within $\pm O(\sqrt{N})$ are sufficient for reconstruction. It is well known that if the output of an algorithm on a secret dataset can be used for reconstruction, then this algorithm is not differentially private. This line of reasoning leads to a lower bound of $\Omega(\sqrt{N})$ on the additive error of any differentially private algorithm answering N random linear queries.

Suppose we could show that an LEDP triangle counting algorithm with $O(n^2)$ additive error can be used to construct a DP algorithm for answering n linear queries with $O(\sqrt{n})$ additive error on some data set of size n – then by the above, we reach a contradiction to the privacy of the algorithm. While indeed a triangle counting algorithm can be used to answer a *single* linear query, the main challenge is that the Dinur-Nissim reconstruction attack requires answering not one, but rather n, linear queries on the same dataset. Let $\mathcal{A}$ be an (ε, δ)-LEDP triangle counting algorithm. If we naively try to answer each linear query to X using a new invocation of the triangle counting algorithm in a black-box manner, this would result in n invocations of $\mathcal{A}$. This in turn would cause the privacy parameters to grow linearly with n, making the privacy breach vacuous. That is, the result would be of

the following sort. An (ε, δ)-LEDP algorithm for triangle counting with low additive error implies an $(O(\varepsilon n), O(n\delta))$-DP algorithm for answering linear queries with low additive error. Since the latter statement is too weak to be used with the results of Dinur and Nissim, we take a different approach.

In order to avoid making n invocations of a triangle counting algorithm, we develop a new type of reconstruction attack on a secret dataset X, where the set of allowed linear queries has a special combinatorial structure. We call the new type of queries *outer-product queries*. We show that, given access to an (ε, δ)-LEDP algorithm $\mathcal{A}$ that approximates the number of triangles up to $O(n^2)$ additive error, we can design a $(2\varepsilon, 2\delta)$-DP algorithm $\mathcal{B}$ for answering $\Theta(n^2)$ outer-product queries on dataset X of size $N = n^2$, so that a constant fraction of them is answered with $O(n)$ additive error. (The dataset size is n^2, so asymptotically the number of random queries and the required accuracy are the same as in the Dinur-Nissim attack.) The main insight is that instead of using $\mathcal{A}$ as a black-box, we use it in a "gray-box" manner. This allows us to answer all $\Theta(n^2)$ queries without degrading the privacy parameters of $\mathcal{B}$. This in turn allows us to reconstruct X, which is a contradiction to the privacy of algorithm $\mathcal{B}$, and thus also to the privacy of algorithm $\mathcal{A}$. Hence, we conclude that any LEDP triangle-counting algorithm must have $\Omega(n^2)$ additive error.

The rest of Section 3.1 is organized as follows. In Section 3.1, we define outer-product queries, and show that an (ε, δ)-DP algorithm $\mathcal{A}$ for triangle-counting with low additive error can be used to construct a $(2\varepsilon, 2\delta)$-DP algorithm $\mathcal{B}$ for answering outer-product queries with low additive error. In Section 3.2, we prove an anti-concentration result for random outer-product queries. In Section 3.3, we use the anti-concentration result to show that an algorithm $\mathcal{B}$ that accurately answers $\Theta(n^2)$ outer-product queries on a sensitive data set $X \in \{0,1\}^{n \times n}$ can be used to reconstruct most of X and complete the proof of Theorem 3.1.

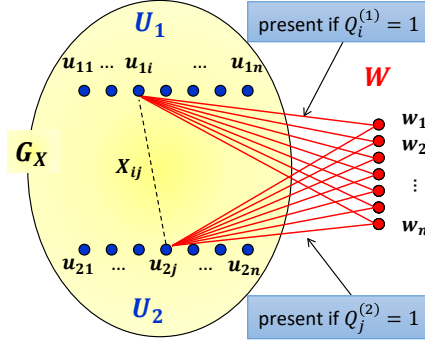## 3.1   Reduction from Outer-product Queries to Triangle Counting

In this section, we prove Lemma 3.3, which is at the heart of our reduction. It shows that, given access to an (ε, δ)-LEDP algorithm $\mathcal{A}$ for approximating the number of triangles with low additive error, we can construct an $(2\varepsilon, 2\delta)$-DP algorithm $\mathcal{B}$ (in the central model) that accurately answers $\Theta(n^2)$ outer-product queries on a sensitive data set X. We start by formally defining this new class of queries.

▶ **Definition 3.2** (Outer-product queries). *Let $X \in \{0,1\}^{n \times n}$. An* outer-product query *to X specifies two vectors A and B of length n with entries in $\{-1, 1\}$ and returns $A^T X B$, that is, $\sum_{i,j \in [n]} A_i X_{ij} B_j$.*

Let γ be the desired reconstruction parameter that indicates that the attack has been successful if we reconstruct at least $(1 - \gamma)n^2$ bits of X correctly. (Later, in Section 3.3, γ will be set to $\frac{1}{9}$ and the number of queries, k, will be set to $\Theta(n^2)$.)

▶ **Lemma 3.3** (Answering Outer-product Queries via Triangle Counting). *Let $\varepsilon, \delta > 0$ and $\gamma \in (0, 1/2)$. Assume that there is a noninteractive (ε, δ)-LEDP algorithm $\mathcal{A}$ that, for every $3n$-node graph, approximates the number of triangles with probability at least $1 - \frac{\gamma^2}{9 \cdot 128}$ and has additive error at most $\frac{\sqrt{\gamma} n^2}{20}$. Then there is an $(2\varepsilon, 2\delta)$-DP algorithm $\mathcal{B}$ in the central model that, for every secret dataset $X \in \{0,1\}^{n \times n}$ and every set of k outer-product queries $(A^{(1)}, B^{(1)}), \ldots, (A^{(k)}, B^{(k)})$, gives answers $a_1, \ldots, a_k$ satisfying*

$$\Pr\left[\left|\left\{\ell \in [k] : \left|(A^{(\ell)})^T X B^{(\ell)} - a_\ell\right| > \frac{\sqrt{\gamma} n}{4}\right\}\right| > \frac{\gamma^2 k}{64}\right] \le \frac{1}{6}.$$

■ **Figure 1** The construction of the query graph $G_{X,Q}$. Each of the parts U_1, U_2, W consists of n nodes. The dashed line is an edge iff $X_{ij} = 1$. Only the subgraph G_X (induced by $U_1 \cup U_2$) holds secret information.

That is, with probability at most $5/6$, for every dataset X and a set of k outer-product queries, Algorithm $\mathcal{B}$ answers inaccurately at most $\frac{\gamma^2 k}{64}$ of the k queries, where by inaccurately we mean with additive error more than $\frac{\sqrt{\gamma} n}{4}$.

Proof. Consider an algorithm $\mathcal{A}$ described in the premise of the lemma. Since $\mathcal{A}$ is local noninteractive, it is specified by a local randomizer $R_v(\mathbf{a})$ for each vertex v, as well as a postprocessing algorithm $\mathcal{P}$. Each randomizer takes an adjacency vector $\mathbf{a} \in \{0,1\}^n$ as input and passes its output to $\mathcal{P}$. Next, we define algorithm $\mathcal{B}$ that, given a sensitive dataset X and a set of k outer-product queries, uses the randomizers and the postprocessing algorithm as subroutines to obtain accurate answers to the outer-product queries.

Fix a dataset $X \in \{0,1\}^{n \times n}$. For each outer-product query (A, B), algorithm $\mathcal{B}$ constructs several corresponding *query* graphs. All query graphs are on the same vertex set V of size $3n$, partitioned into three sets U_1, U_2, and W of size n. The vertices in U_t for $t \in \{1, 2\}$ are denoted $u_{t1}, \ldots, u_{tn}$. The vertices of W are denoted $w_1, \ldots, w_n$. See Figure 1 for an illustration.

Algorithm $\mathcal{B}$ first forms a bipartite graph G_X with parts U_1 and U_2 with X as the adjacency matrix; that is, it adds an edge (u_{1i}, u_{2j}) for each $i, j \in [n]$ with $X_{ij} = 1$. We call G_X the *secret subgraph*, because it will be included as a subgraph in every query graph and it will be the only part of that graph that contains any information about the original sensitive dataset X. Note that G_X does not depend on the outer-product query. The remaining edges of each query subgraph are between $U_1 \cup U_2$ and W and are specific to each query graph, so that overall the resulting graph is tripartite. For each $v \in U_1 \cup U_2$, let $\Gamma_X(v)$ denote the neighbors of v in the secret subgraph G_X. A key idea in the construction is that every node in the secret subgraph G_X will have one of only two possible neighborhoods in each query graph. This allows algorithm $\mathcal{B}$ to simulate triangle-counting computations on all query graphs by invoking a local randomizer on each vertex in $U_1 \cup U_2$ only twice. For each vertex $v \in U_1 \cup U_2$, algorithm $\mathcal{B}$ runs its local randomizer $R_v(\cdot)$ twice: once with the adjacency list specified by $\Gamma_X(v)$ and once with the adjacency list specified by $\Gamma_X(v) \cup W$. Algorithm $\mathcal{B}$ then records the output of the former invocation as as $r_0(v)$, and the latter as $r_1(v)$.

By the composition property of differential privacy, the algorithm that simply outputs the vector of all $4n$ responses of the local randomizers is $(2\varepsilon, 2\delta)$-DP by composition, because each bit of X is encoded as a potential edge and used in two executions of the randomizers for its endpoints, where each execution (of all randomizers) is (ε, δ)-LEDP. In the remaining steps, algorithm $\mathcal{B}$ only postprocesses the vector of responses, and thus it is $(2\varepsilon, 2\delta)$-DP.

Next, we describe how to postprocess the vector of responses to obtain an answer to an outer-product query (A, B). To answer each outer-product query, algorithm $\mathcal{B}$ will first obtain answers to three linear queries that we call *submatrix queries*. Submatrix queries are defined the same way as outer-product queries, except that vectors A and B have entries in $\{0, 1\}$ instead of $\{-1, 1\}$. Next, we explain how to answer submatrix queries on X, deferring to Claim 3.5 the description of the simulation of each outer-product query with submatrix queries.

To answer a submatrix query $Q = (Q^{(1)}, Q^{(2)})$ on dataset X, algorithm $\mathcal{B}$ completes the secret subgraph G_X to a query graph $G_{X,Q}$ as follows. For each vertex $u_{ti} \in U_1 \cup U_2$, where $t \in \{1, 2\}$ and $i \in [n]$, it adds edges determined by $Q^{(t)}$: specifically, if $Q_i^{(t)} = 1$, it adds edges from u_{ti} to all vertices in W. Next claim states the relationship between the number of triangles in $G_{X,Q}$ and the answer to the submatrix query Q.

▷ **Claim 3.4.** The number of triangles in graph $G_{X,Q}$ is equal to $n \cdot (Q^{(1)})^T X Q^{(2)}$.

Proof. Observe that $G_{X,Q}$ is tripartite with parts (U_1, U_2, W), so all triangles must have one vertex in each part. The answer to the submatrix query $Q = (Q^{(1)}, Q^{(2)})$ is

$$(Q^{(1)})^T X Q^{(2)} = \sum_{i,j \in [n]} Q_i^{(1)} Q_j^{(2)} X_{ij}.$$

For each term in the sum, both u_{1i} and u_{2j} are adjacent to all nodes in W iff $Q_i^{(1)} = Q_j^{(2)} = 1$. If the edge (u_{1i}, u_{2j}) is present in the graph, then this results in n triangles. Thus, each term where $Q_i^{(1)} = Q_j^{(2)} = X_{ij} = 1$ corresponds to n triangles of the form (u_{1i}, u_{2j}, w_ℓ), where $\ell \in [n]$. All other terms create no triangles, since either $X_{ij} = 0$, in which case the edge (u_{1i}, u_{2j}) is not present in the graph, or either $Q_i^{(1)} = 0$ or $Q_j^{(2)} = 0$, in which case u_{1i} and u_{2j} do not have common neighbors. ◁

To answer a submatrix query Q, algorithm $\mathcal{B}$ simulates a call to the triangle-counting algorithm $\mathcal{A}$ on the corresponding query graph $G_{X,Q}$. First, $\mathcal{B}$ runs the local randomizers for the vertices in W with their adjacency vectors specified by the graph $G_{X,Q}$. Note that these vertices do not have access to any private information, so this operation does not affect privacy. For each vertex $u_{ti} \in U_1 \cup U_2$, where $t \in \{1, 2\}$ and $i \in [n]$, algorithm $\mathcal{B}$ uses the result $r_b(u_{ti})$ from the previously run randomizer, where $b = Q_i^{(t)}$. E.g., if $Q_i^{(1)} = 0$, then $\mathcal{B}$ uses the result $r_0(u_{1i})$, and if $Q_i^{(1)} = 1$, it uses the result $r_1(u_{1i})$. Now algorithm $\mathcal{B}$ has results from all vertex randomizers on the graph $G_{X,Q}$ and it simply runs the postprocessing algorithm $\mathcal{P}$ on these results. To obtain the answer to the submatrix query, $\mathcal{B}$ divides the output of $\mathcal{P}$ by n.

Finally, algorithm $\mathcal{B}$ answers each outer-product query as specified in the following claim, by getting answers to three submatrix queries.

▷ **Claim 3.5.** An outer-product query to X can be simulated with three submatrix queries to X. Moreover, if all three submatrix queries are answered with additive error at most α, then the outer product query can be answered with additive error at most 5α.

Proof. Consider an outer-product query to an $n \times n$ matrix X specified by $A, B \in \{-1, 1\}^n$. Define n-bit vectors $A' = \frac{1}{2}(A + \vec{1})$ and $A'' = \frac{1}{2}(-A + \vec{1})$, where $\vec{1}$ denotes a vector of 1s of length n. Define B' and B'' analogously. Then, as illustrated in Figure 2,

$$A^T X B = 2((A')^T X B' + (A'')^T X B'') - \vec{1}^T X \vec{1}.$$

Figure 2 Every outer-product query can be simulated using three submatrix queries. For the illustration, the entries of all vectors are rearranged to group the same values together. The outer product is denoted $\otimes$.

That is, the answer to the outer-product query (A, B) can be computed from the answers to the submatrix queries $(A', B'), (A'', B'')$, and $(\vec{1}, \vec{1})$, and the additive error increases from α to 5α, as stated. $\triangleleft$

It remains to prove the following claim.

$\triangleright$ **Claim 3.6.** Let $\mathcal{A}$ be as in the premise of Lemma 3.3. For every secret dataset $X \in \{0, 1\}^{n \times n}$ and every set of k outer-product queries $\{(A^{(\ell)}, B^{(\ell)})\}_{\ell \in [k]}$, algorithm $\mathcal{B}$ gives answers $a_1, \ldots, a_k$ satisfying

$$\Pr\left[\left|\left\{\ell \in [k] : \left|(A^{(\ell)})^T X B^{(\ell)} - a_\ell\right| > \frac{\sqrt{\gamma} n}{4}\right\}\right| > \frac{\gamma^2 k}{64}\right] \le \frac{1}{6}.$$

That is, the number of "incorrectly" answered outer-product queries exceeds $\frac{\gamma^2 k}{64}$ with probability at most $1/6$.

Proof. By the assumption on $\mathcal{A}$, for every graph G, algorithm $\mathcal{A}$ returns the number of triangles in G within an additive error at most $\frac{\sqrt{\gamma} n^2}{20}$ with probability at least $1 - \frac{\gamma^2}{9 \cdot 128}$.

Given a secret dataset X and k outer-product queries, algorithm $\mathcal{B}$ first creates k triples of submatrix queries corresponding to the outer-product queries. Then $\mathcal{B}$ uses $\mathcal{A}$ as a gray box, to answer all $3k$ submatrix queries simultaneously. Recall that this is achieved by invoking the local randomizers on vertices holding private information (that is, vertices in parts U_1, U_2) twice, once for each potential value of the bit that corresponds to this vertex in a specific query. Then for each individual submatrix query, one local randomizer is invoked on each of the n vertices in W with the adjacency list that corresponds to that specific query graph. Then, to answer each specific submatrix query, algorithm $\mathcal{B}$ combines the new outputs of the vertices from W with the stored outputs from running randomizers on $U_1 \cup U_2$ that correspond to that specific query, and invokes the postprocessing algorithm $\mathcal{P}$ on this vector of $3n$ outputs. Finally, $\mathcal{B}$ divides $\mathcal{P}$'s answer by n to obtain the answer to the submatrix query.

Each invocation of $\mathcal{P}$ by $\mathcal{B}$ simulates one triangle-counting computation. Overall, we have $3k$ (*dependent*) simulated triangle-counting computations. By the assumption on $\mathcal{A}$, stated in the premise of Lemma 3.3, the postprocessing algorithm $\mathcal{P}$ answers each simulated triangle-counting computation inaccurately (i.e., with additive error exceeding $\frac{\sqrt{\gamma} n^2}{20}$) with probability at most $\frac{\gamma^2}{9 \cdot 128}$ (where this probability is taken over the random coins of the individual $3n$ local randomizers, as well as the random coins of $\mathcal{P}$). Overall, there are $3k$ (dependent) simulations, and so the expected number of simulated triangle-counting computations for which $\mathcal{A}$ returns additive error greater than $\frac{\sqrt{\gamma} n^2}{20}$ is at most $\frac{\gamma^2 \cdot (3k)}{9 \cdot 128} = \frac{\gamma^2 \cdot k}{6 \cdot 64}$. Hence, by Markov's inequality, the probability that the number of inaccurate simulated triangles queries exceeds $\frac{\gamma^2 \cdot k}{64}$ is at most $\frac{1}{6}$.

Condition on the event that at most $\frac{\gamma^2 \cdot k}{64}$ of the triangle-counting computations are answered inaccurately, so that the remaining computations are answered with error at most $\alpha = \frac{\sqrt{\gamma} n^2}{20}$, and denote this even by E. Recall that each triangle-counting computation is used to answer a single submatrix query, and that by Claim 3.4, if a triangle-counting computation is answered with additive error α, then the corresponding submatrix query is answered with additive error α/n. Hence, by the above conditioning, at most $\frac{\gamma^2 \cdot k}{64}$ of the submatrix queries are answered with additive error greater than α/n. Each inaccurate answer to a triangle-counting computation can spoil the answer to at most one outer-product query. Furthermore, by Claim 3.5, if all three submatrix queries used to compute a single outer-product query are answered to within additive error α/n, then the outer-product query is answered to within additive error $5\alpha/n$. Hence, by the above conditioning, at most $\frac{\gamma^2 \cdot k}{64}$ of the outer-product queries are answered with additive error greater than $5\alpha/n = \frac{\sqrt{\gamma} n}{4}$. Since event E occurs with probability at least $5/6$, we get that with probability at least $5/6$, the fraction of outer-product queries that is answered with additive error greater than $\frac{\sqrt{\gamma} n}{4}$ is at most $\frac{\gamma^2 \cdot k}{64}$, so that Claim 3.6 holds. ◁

This completes the proof of Lemma 3.3. ◀

3.2 Anti-Concentration for Random Outer-Product Queries

In this section, we prove Lemma 1.4. To analyze our reconstruction attack, we will consider the differences between the true dataset X and a potential reconstructed dataset Y. Let M denote $X - Y$. Then, for an outer-product query (A, B), the difference between the answers to this query on dataset X and on dataset Y is $A^T X B - A^T Y B = A^T M B$. The main result of this section shows that if X and Y differ on many entries (that is, M has lots of nonzero entries) then a random outer-product query is likely to produce significantly different answers on X and Y.

Proof of Lemma 1.4. Let $Z_{ij} = A_i B_j$ for all $i, j \in [n]$, and $U = A^T M B$. We prove the lemma by computing the expectation and the second and the fourth moments of U, and then apply the Paley-Zigmund inequality to U^2.

By independence of A_i and B_j for all $i, j \in [n]$, we have $\mathbb{E}[Z_{ij}] = \mathbb{E}[A_i] \cdot \mathbb{E}[B_j] = 0$ and $\mathrm{Var}[Z_{ij}] = \mathbb{E}[Z_{ij}^2] = \mathbb{E}[A_i^2 B_j^2] = 1$. By definition of U and the linearity of expectation,

$$\mathbb{E}[U] = \mathbb{E}[A^T M B] = \mathbb{E}\Big[\sum_{i,j \in [n]} M_{i,j} Z_{i,j} \Big] = \sum_{i,j \in [n]} M_{i,j} \mathbb{E}[Z_{i,j}] = 0.$$

Note that random variables Z_{ij} are pairwise independent. This is an important feature of random outer-product queries and the main reason to use them instead of the submatrix queries. This feature greatly simplifies the analysis. Since U is unbiased, $\mathbb{E}[U^2] = \mathrm{Var}[U]$. By pairwise independence of Z_{ij},

$$\mathrm{Var}[U] = \mathrm{Var}\Big[\sum_{i,j \in [n]} M_{ij} Z_{ij} \Big] = \sum_{i,j \in [n]} M_{ij}^2 \, \mathrm{Var}[Z_{ij}] = \sum_{i,j \in [n]} M_{ij}^2 = m.$$

Next, we give an upper bound on the 4th moment of U.

▷ **Claim 3.7.** $\mathbb{E}[U^4] \leq 9n^4$.

Proof. We use the definition of U, write it out as a sum, and multiply out the terms of the product:

$$
\mathbb{E}[U^4] = \mathbb{E}[(A^T M B)^4] = \mathbb{E}\Big[\Big(\sum_{i,j\in[n]} M_{ij} Z_{ij}\Big)^4\Big]
$$

$$
= \sum_{(i_1,j_1),\ldots,(i_4,j_4)\in[d]\times[d]} M_{i_1 j_1} M_{i_2 j_2} M_{i_3 j_3} M_{i_4 j_4}\, \mathbb{E}[Z_{i_1 j_1} Z_{i_2 j_2} Z_{i_3 j_3} Z_{i_4 j_4}], \tag{1}
$$

where Equation (1) is obtained by using the linearity of expectation. Next, we evaluate the expectation of the product in Equation (1):

$$
\mathbb{E}[Z_{i_1 j_1} Z_{i_2 j_2} Z_{i_3 j_3} Z_{i_4 j_4}] = \mathbb{E}[A_{i_1} B_{j_1} A_{i_2} B_{j_2} A_{i_3} B_{j_3} A_{i_4} B_{j_4}]
$$

$$
= \mathbb{E}[A_{i_1} A_{i_2} A_{i_3} A_{i_4}]\, \mathbb{E}[B_{j_1} B_{j_2} B_{j_3} B_{j_4}],
$$

where the last equality follows by independence of A_i and B_j for all $i, j \in [n]$. The expression $\mathbb{E}[A_{i_1} A_{i_2} A_{i_3} A_{i_4}]$ is 0 if at least one of the indices appears only once in the tuple (i_1, i_2, i_3, i_4), since, in this case, we can use the independence of the corresponding factor A_i from the remaining factors to represent this expression as $\mathbb{E}[A_i]$ multiplied by the expectation of the product of the remaining factors. Since $\mathbb{E}[A_i] = 0$ for all $i \in [n]$, the overall expression evaluates to 0.

Note that if one of the factors appears exactly three times, then another factor appears exactly once. Therefore, the remaining case is when each factor appears an even number of times. If there are two factors, say A_i and A_j that appear twice, then the expression evaluates to $\mathbb{E}[A_i^2 A_j^2] = 1$. It also evaluates to 1 when $i = j$.

Thus, each term in Equation (1) is either 0 or 1. By symmetry, it can potentially be 1 only if each index in the tuple (i_1, i_2, i_3, i_4) and each index in the tuple (j_1, j_2, j_3, j_4) appears an even number of times. It remains to give an upper bound on the number of such terms. There are $\binom{n}{2}$ ways to choose two distinct i-indices and $\binom{4}{2} = 6$ possible positions for them in the 4-tuple. In addition, there are n ways to choose an index that appears 4 times in the 4-tuple. So, the number of possibilities for nonzero $\mathbb{E}[A_{i_1} A_{i_2} A_{i_3} A_{i_4}]$ is at most $3n^2$. The same bounds holds for $\mathbb{E}[B_{j_1} B_{j_2} B_{j_3} B_{j_4}]$. Consequently, the number of terms equal to 1 in Equation (1) is at most $9n^4$. Thus, the sum evaluates to at most $9n^4$. This completes the proof of Claim 3.7. ◁

Since U^2 is a nonnegative random variable with finite variance, the Paley-Zygmund inequality gives that, for all $\theta \in [0, 1]$,

$$
\Pr\left[U^2 > \theta\, \mathbb{E}[U^2]\right] \geq (1-\theta)^2 \frac{(\mathbb{E}[U^2])^2}{\mathbb{E}[U^4]} \geq (1-\theta)^2 \frac{m^2}{9n^4} \geq (1-\theta)^2 \frac{(\gamma n^2)^2}{9n^4} = (1-\theta)^2 \frac{\gamma^2}{9},
$$

where the last inequality uses the bound $m \geq \gamma n^2$ stated in the lemma. Finally, we set $\theta = 1/4$ and get:

$$
\Pr\left[|A^T M B| > \frac{\sqrt{m}}{2}\right] = \Pr\left[|U| > \frac{\sqrt{m}}{2}\right] = \Pr\left[U^2 > \frac{m}{4}\right] \geq \frac{3^2}{4^2} \frac{\gamma^2}{9} = \frac{\gamma^2}{16},
$$

completing the proof of Lemma 1.4. ◀

3.3   Reconstruction Attack Using Outer-Product Queries

To simplify notation in this section, we represent our datasets and outer-product queries as vectors. Formally, X here denotes the vectorization of the original sensitive dataset, i.e., a vector in $\{0,1\}^{n^2}$. For an outer-product query (A, B), we let $Q \in \{0,1\}^{n^2}$ represent the vectorization of $A \otimes B$, the outer product of A and B. (In other words, Q is the Kronecker product of A and B.) Then the answer to the query is the dot product $Q \cdot X$.

In this section, we define and analyze the attacker's algorithm $\mathcal{C}$ and complete the proof of Theorem 3.1. The attacker $\mathcal{C}$ runs algorithm $\mathcal{B}$ from Section 3.1 on the sensitive dataset X and a set of k random outer-product queries $Q_1, \ldots, Q_k$ to obtain answers $a_1, \ldots, a_k$. For all $\ell \in [k]$, we call the answer a_ℓ *accurate for a dataset* Y if $|Q_\ell \cdot Y - a_\ell| \le \frac{\sqrt{\gamma}n}{4}$; otherwise, we call a_ℓ *inaccurate for* Y. The attacker $\mathcal{C}$ outputs any dataset $Y^* \in \{0,1\}^{n^2}$ for which at most $\frac{\gamma^2 k}{64}$ answers among $a_1, \ldots, a_k$ are inaccurate for Y^*. By Lemma 3.3, the probability that X satisfies this requirement is at least $\frac{5}{6}$. If this event occurs, algorithm $\mathcal{C}$ will be able to output some Y^*. (Otherwise, the attack fails.)

Next, we analyze the attack. Let $\|X - Y\|_1$ denote the Hamming distance between datasets X and Y. Call a dataset Y *bad* if $\|X - Y\|_1 > \gamma n^2$, i.e., if it differs from X on more than γn^2 entries. We will show that $\mathcal{C}$ is unlikely to choose a bad data set as Y^*. Fix a bad dataset Y. Let $M = X - Y$, and observe that M has $m > \gamma n^2$ nonzero entries. We say that a set of queries $\{Q_1, \ldots, Q_k\}$ *catches* the dataset Y if more than $\frac{\gamma^2 k}{32}$ entries in $(|Q_1 \cdot M|, \ldots, |Q_k \cdot M|)$ exceed $\frac{\sqrt{\gamma}n}{2}$.

▶ **Lemma 3.8.** *Suppose the attacker $\mathcal{C}$ makes $k = \frac{128n^2}{\gamma^2}$ uniformly random outer-product queries. Then the probability that there exists a bad dataset not caught by the attacker's set of queries is at most $\frac{1}{6}$.*

Proof. Consider a set of k uniformly random outer-product queries $\{Q_\ell\}_{\ell \in [k]}$. Fix a bad dataset Y. Then $\|X - Y\|_1 > \gamma n^2$. Let $M = X - Y$.

For every $\ell \in [k]$, let $\chi_\ell = 1$ if $|Q_\ell \cdot M| > \frac{\sqrt{\gamma}n}{2}$, and otherwise let $\chi_\ell = 0$. Also, let $\chi = \sum_{\ell=1}^k \chi_\ell$. By definition, the difference vector $M = X - Y$ has more than γn^2 nonzero entries. By the anti-concentration bound in Lemma 1.4, $\Pr\left[|Q_\ell \cdot M| > \frac{\sqrt{\gamma}n}{2}\right] \ge \frac{\gamma^2}{16}$. Therefore, $\mathbb{E}[\chi_\ell] \ge \frac{\gamma^2}{16}$. By the Chernoff bound, we have that for $k = \frac{128n^2}{\gamma^2}$ and for $n \ge 3$,

$$\Pr\left[\chi \le \frac{\gamma^2 \cdot k}{32}\right] \le \exp\left(-\frac{\gamma^2 k}{128}\right) = \exp\left(-n^2\right) < \frac{1}{6 \cdot 2^{n^2}}.$$

Hence, the set $\{Q_\ell\}_{\ell \in [k]}$ fails to catch each specific bad dataset with probability at most $\frac{1}{6 \cdot 2^{n^2}}$. By a union bound over at most 2^{n^2} bad datasets, the probability that there exists a bad dataset not caught by the attacker's queries is at most $1/6$.   ◀

▶ **Lemma 3.9** (Reconstruction Lemma). *If algorithm $\mathcal{B}$ has additive error at most $\frac{\sqrt{\gamma}n}{4}$ on all but at most $\frac{\gamma^2 k}{64}$ answers, and the set of queries it uses catches all bad datasets Y, then the reconstruction attack is successful, that is, the attacker $\mathcal{C}$ outputs Y^* that differs from X on at most γn^2 entries, i.e., $\|X - Y^*\|_1 \le \gamma n^2$.*

Proof. By the first premise of the lemma, the dataset X "disagrees" with at most $\frac{\gamma^2 k}{64}$ of the answers a_ℓ. Hence, necessarily, the attacker $\mathcal{C}$ outputs some dataset Y^*. Assume towards a contradiction that Y^* is a bad dataset. Let $\{Q_\ell\}_{\ell \in [k]}$ be the set of queries chosen by $\mathcal{B}$. Let $M^* = X - Y^*$ be the difference vector. By the triangle inequality, $|Q_\ell M^*| = |Q_\ell X - Q_\ell Y^*| \le |Q_\ell X - a_\ell| + |Q_\ell Y^* - a_\ell|$. From the first assumption in the

lemma, $|Q_\ell X - a_\ell| \leq \frac{\sqrt{\gamma} n}{4}$ for all but at most $\frac{\gamma^2 k}{64}$ of the queries. By the description of the attack $\mathcal{C}$, the output Y^* is such that for all but at most $\frac{\gamma^2 k}{64}$ of the queries, $|Q_\ell Y^* - a_\ell| \leq \frac{\sqrt{\gamma} n}{4}$. Therefore, for all but at most $\frac{\gamma^2 k}{32}$ of the queries, $|Q_\ell M^*| \leq |Q_\ell X - a_\ell| + |Q_\ell Y^* - a_\ell| \leq \frac{\sqrt{\gamma} n}{2}$. Since $\{Q_\ell\}_{\ell \in [k]}$ catches all bad datasets, it in particular catches Y^*, because Y^* is bad. By definition of catching, $|Q_\ell M^*| > \frac{\sqrt{\gamma} n}{2}$ for more than $\frac{\gamma^2 k}{32}$ of the values $Q_\ell M^*$. Hence, we have reached a contradiction, implying that Y^* is a good dataset. ◀

The final ingredient for proving Theorem 3.1 is the following lemma, which is based on an argument of [15]. Any algorithm that outputs a large fraction of its secret dataset is definitely not private, for any reasonable notion of privacy. Lemma 3.10 states that such an algorithm is not differentially private.

▶ **Lemma 3.10.** *Let $\mathcal{C}$ be an algorithm that takes as input a secret data set X in $\{0,1\}^N$ and outputs a vector in the same set, $\{0,1\}^N$. If $\mathcal{C}$ is (ε, δ)-differentially private and X is uniformly distributed in $\{0,1\}^N$, then*

$$\mathbb{E}\left[\|\mathcal{C}(X) - X\|_1\right] \geq e^{-\varepsilon}\left(\tfrac{1}{2} - \delta\right)N.$$

Lemma 3.10 above only bounds the expectation of $\|\mathcal{C}(X) - X\|_1$. The more sophisticated argument in [15] yields much tighter concentration results. We use the simpler version here since it allows for a self-contained presentation.

Proof. Fix an index $i \in [N]$ and a bit $r \in \{0,1\}$. Let $X_{i \to r}$ denote the vector obtained by replacing the i-th entry of X with the bit r.

Consider the pair of random variables $(X, \mathcal{C}(X))$. Because $\mathcal{C}$ is (ε, δ)-differentially private, this is distributed similarly to the pair $(X_{i \to R}, \mathcal{C}(X))$, where R is a uniformly random bit independent of the other values. Specifically, for any event $E \subseteq \{0,1\}^N \times \{0,1\}^N$,

$$\Pr[(X_{i \to R}, \mathcal{C}(X)) \in E] \leq e^\varepsilon \Pr[(X, \mathcal{C}(X)) \in E] + \delta.$$

Applying this inequality to the event $E_i = \{(x, y) : x_i \neq y_i\}$ shows that

$$\tfrac{1}{2} = \Pr[\mathcal{C}(X)_i \neq R] \leq e^\varepsilon \Pr[\mathcal{C}(X)_i \neq X_i] + \delta \quad \text{and thus} \quad \Pr[\mathcal{C}(X)_i \neq X_i] \geq e^{-\varepsilon}\left(\tfrac{1}{2} - \delta\right).$$

The Hamming distance $\|\mathcal{C}(X) - X\|_1$ is the sum of the indicator random variables for the events $\mathcal{C}(X)_i \neq X_i$. By linearity of expectation, the expected Hamming distance is at least $e^{-\varepsilon}\left(\tfrac{1}{2} - \delta\right)N$. ◀

Finally, we use Lemmas 3.3 and 3.8–3.10 to compete the proof of the main theorem.

Proof of Theorem 3.1. We set $\gamma = \frac{1}{9}$. Assume towards a contradiction that for some ε and δ as in the statement of the theorem, there exists an (ε, δ)-LEDP algorithm $\mathcal{A}$ that for every $3n$-node graph approximates the number of triangles in the graph up to additive error $\alpha = \frac{\sqrt{\gamma} n^2}{20}$ with probability at least $1 - \frac{\gamma^2}{9 \cdot 128} = 1 - \frac{1}{3^6 \cdot 2^7}$. Then by Lemma 3.3, there exists a $(2\varepsilon, 2\delta)$-DP algorithm $\mathcal{B}$ that, for every secret dataset X and every set of k outer-product queries, answers inaccurately (i.e., with additive error more than $\frac{\sqrt{\gamma} n}{4}$) on at most $\frac{\gamma^2 k}{64}$ of the k queries with probability at least $\frac{5}{6}$. By Lemma 3.8, the probability that a set of $k = \frac{128 n^2}{\gamma^2}$ random outer-product queries chosen by the attacker $\mathcal{C}$ does not catch all bad datasets is at most $\frac{1}{6}$. By a union bound, with probability at least $\frac{2}{3}$, the attacker $\mathcal{C}$ satisfies the premise of Lemma 3.9 and the set of chosen queries catches all bad data sets. Hence, with probability at least $\frac{2}{3}$, the attacker $\mathcal{C}$ outputs a dataset Y^* which coincides with X on at least $(1 - \gamma)n^2$ entries. The expected Hamming distance $\mathbb{E}[\|X - Y^*\|_1]$ is therefore at most $\frac{2}{3} \cdot \gamma n^2 + \frac{1}{3} n^2 = \frac{1 + 2\gamma}{3} n^2$. When $\gamma = \frac{1}{9}$, the expected distance is less than $0.41 n^2$.

Recall that the attacker $\mathcal{C}$ runs $(2\varepsilon, 2\delta)$-DP algorithm $\mathcal{B}$ on a secret dataset X and then post processes the output of $\mathcal{B}$. Thus, $\mathcal{C}$ is $(2\varepsilon, 2\delta)$-DP, and we can apply Lemma 3.10 to conclude that the expected Hamming distance $\mathbb{E}\|\mathcal{C}(X) - X\|_1$ is at most $e^{-2\varepsilon}(\frac{1}{2} - 2\delta)n^2$. Since, by assumption, $\varepsilon \leq 1/20$ and $\delta \leq 1/100$, we have $\mathbb{E}\|\mathcal{C}(X) - X\|_1 \geq 0.43n^2$. This contradicts the upper bound of $0.41n^2$ above.   ◄

4   The Interactive Lower Bound

In this section, we present an $\Omega\left(\frac{n^{3/2}}{\varepsilon}\right)$ lower bound on the additive error of every ε-LEDP algorithm for estimating the number of triangles in a graph, stated formally in Theorem 1.3. We reduce from the problem of computing the summation in the LDP model.

▶ **Definition 4.1** (Summation function). *Let SUM_n be the following function. For all $x_1, \ldots, x_n \in \{0, 1\}$, $SUM_n(x_1, \ldots, x_n) = \sum_{i=1}^n x_i$.*

This problem was shown to have an additive error lower bound of $\Omega(\sqrt{n}/\varepsilon)$ [36, Theorem 5.3 of arxiv v2]. We substitute $\alpha = \alpha_0/n$ and $\beta = \varepsilon\alpha_0/n$ to obtain the following lemma.

▶ **Lemma 4.2** ([10, 6, 36]). *There exists a constant $c > 0$ such that for every $\varepsilon \in (0, 1)$, $n \in \mathbb{N}, \alpha_0 \in (0, n]$ and $\delta \in \left[0, \frac{1}{10^5} \cdot \frac{\varepsilon^3 \alpha_0^2}{n^3 \ln(n^2/\varepsilon\alpha_0)}\right]$, if $\mathcal{B}$ is an (ε, δ)-LDP algorithm where each party i receives input $x_i \in \{0, 1\}$ and $\mathcal{B}$ estimates SUM_n up to additive error α_0 with probability at least $2/3$, then $\alpha_0 \geq c \cdot \sqrt{n}/\varepsilon$.*

Proof of Theorem 1.3. We reduce from SUM_n in the local model. By Lemma 4.2, every (potentially interactive) algorithm that approximates SUM_n with additive error at most α_0 (with sufficiently high constant probability) must have $\alpha_0 = \Omega(\frac{\sqrt{n}}{\varepsilon})$. In our reduction, we will set the additive error of the triangle-counting algorithm, $\alpha = \alpha_0 n$.

Our reduction is black-box. Given an instance of SUM_n, where each local party holds one bit X_i of the vector $(X_1, \ldots, X_n)$, the parties implicitly create the following graph G. The vertex set consists of two sets of nodes, V_1 and V_2, where V_1 has size n and V_2 has size $2n$. The nodes in V_1 will not have any secret information and can be simulated by any local party. The nodes in V_2 are $[2n]$, and each party $i \in [n]$ is responsible for simulating nodes $2i - 1$ and $2i$ in V_2. To create the edges of G, we add edges of the complete bipartite graph between V_1 and V_2. In addition, each pair of nodes $(2i - 1, 2i)$ in V_2 has an edge between them if and only if $x_i = 1$. See Figure 3 for an illustration.

Let $S = x_1 + \ldots x_n$. Observe that any triangle in G must have two vertices in V_2 and an edge between a pair of matched nodes. Any such edge contributes exactly n triangles. So, the total number of triangles in G is $T = Sn$.

For the sake of contradiction, suppose there is an (ε, δ)-LEDP algorithm $\mathcal{A}$ that estimates the number of edges with error $o(\frac{n^{3/2}}{\varepsilon})$. We run it on G. By construction, party i can simulate the two nodes assigned to it, and anybody can simulate nodes in V_1. When the algorithm gets an estimate $\widehat{T}$ for the number of triangles, it outputs $\hat{S} = \widehat{T}/n$. If $\widehat{T} = T \pm o(\frac{n^{3/2}}{\varepsilon})$, then $\hat{S} = \widehat{T}/n = T/n \pm o(\frac{\sqrt{n}}{\varepsilon}) = S \pm o(\frac{\sqrt{n}}{\varepsilon})$, which is a contradiction to Lemma 4.2.

Moreover, if $\mathcal{A}$ is (ε, δ)-LEDP, then the reduction algorithm is (ε, δ)-LDP with respect to the secret dataset X. However, the latter contradicts Lemma 4.2. Thus, $\mathcal{A}$ cannot exist.   ◄
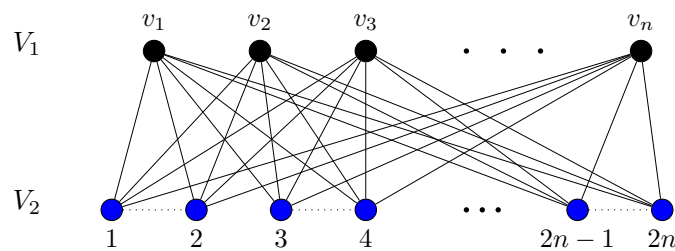
Figure 3 An instance of the interactive $\Omega(n^{3/2})$ lower bound consists of a complete bipartite graph with parts V_1, V_2 of sizes n and $2n$, respectively; in addition, there is an edge between each pair $\{2i-1, 2i\}$ iff the secret input bit $X_i = 1$.

References

1 Mohammad Al Hasan and Vachik S Dave. Triangle counting in large networks: a review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(2):e1226, 2018.

2 Differential Privacy Team Apple. Learning with privacy at scale differential, 2017.

3 Sepehr Assadi, Michael Kapralov, and Sanjeev Khanna. A simple sublinear-time algorithm for counting arbitrary subgraphs via edge sampling. In *ITCS*, volume 124 of *LIPIcs*, pages 6:1–6:20. Schloss Dagstuhl – Leibniz-Zentrum fuer Informatik, 2019.

4 Prakash Balachandran, Eric D. Kolaczyk, and Weston D. Viles. On the propagation of low-rate measurement error to subgraph counts in large networks. *JMLR*, 18(61):1–33, 2017.

5 Luca Becchetti, Paolo Boldi, Carlos Castillo, and Aristides Gionis. Efficient semi-streaming algorithms for local triangle counting in massive graphs. In *Proceedings of the 14th ACM SIGKDD*, pages 16–24, 2008.

6 Amos Beimel, Kobbi Nissim, and Eran Omri. Distributed private data analysis: Simultaneously solving how and what. In *CRYPTO*, pages 451–468. Springer, 2008.

7 Amartya Shankha Biswas, Talya Eden, Quanquan C. Liu, Ronitt Rubinfeld, and Slobodan Mitrovic. Massively parallel algorithms for small subgraph counting. In Amit Chakrabarti and Chaitanya Swamy, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX/RANDOM 2022, September 19-21, 2022, University of Illinois, Urbana-Champaign, USA (Virtual Conference)*, volume 245 of *LIPIcs*, pages 39:1–39:28. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. `doi:10.4230/LIPIcs.APPROX/RANDOM.2022.39`.

8 Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th SOSP*, SOSP '17, pages 441–459, New York, NY, USA, 2017.

9 Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. Differentially private data analysis of social networks via restricted sensitivity. In Robert D. Kleinberg, editor, *ITCS '13, Berkeley, CA, USA, January 9-12, 2013*, pages 87–96. ACM, 2013. `doi:10.1145/2422436.2422449`.

10 TH Hubert Chan, Elaine Shi, and Dawn Song. Optimal lower bound for differentially private multi-party aggregation. In *ESA*, pages 277–288. Springer, 2012.

11 Jinyuan Chang, Eric D. Kolaczyk, and Qiwei Yao. Estimation of subgraph densities in noisy networks. *Journal of the American Statistical Association*, 117(537):361–374, 2022.

12 Justin Y Chen, Talya Eden, Piotr Indyk, Honghao Lin, Shyam Narayanan, Ronitt Rubinfeld, Sandeep Silwal, Tal Wagner, David Woodruff, and Michael Zhang. Triangle and four cycle counting with predictions in graph streams. In *ICLR*, 2021.

13   Shixi Chen and Shuigeng Zhou. Recursive mechanism: towards node differential privacy and unrestricted joins. In Kenneth A. Ross, Divesh Srivastava, and Dimitris Papadias, editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, June 22-27, 2013*, pages 653–664. ACM, 2013. `doi:10.1145/2463676.2465304`.

14   Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. Privacy at scale: Local differential privacy in practice. In *Proceedings of the 2018 SIGMOD*, SIGMOD '18, pages 1655–1658, New York, NY, USA, 2018.

15   Anindya De. Lower bounds in differential privacy. In *TCC*, pages 321–338. Springer, 2012.

16   Laxman Dhulipala, Quanquan C. Liu, Sofya Raskhodnikova, Jessica Shi, Julian Shun, and Shangdi Yu. Differential privacy from locally adjustable graph algorithms: k-core decomposition, low out-degree ordering, and densest subgraphs. In *FOCS 2022, Denver, CO, USA, October 31 – November 3, 2022*, pages 754–765. IEEE, 2022. `doi:10.1109/FOCS54457.2022.00077`.

17   Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. Collecting telemetry data privately. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 3574–3583, Red Hook, NY, USA, 2017. Curran Associates Inc.

18   Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In Frank Neven, Catriel Beeri, and Tova Milo, editors, *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART PODS, June 9-12, 2003, San Diego, CA, USA*, pages 202–210. ACM, 2003. `doi:10.1145/773153.773173`.

19   John Duchi, Michael Jordan, and Martin Wainwright. Local privacy and statistical minimax rates. In *IEEE Symposium on Foundations of Computer Science*, FOCS '13, pages 429–438, Berkeley, CA, USA, 2013. `arXiv:1302.3203`.

20   John C Duchi, Michael I Jordan, Martin J Wainwright, et al. Minimax optimal procedures for locally private estimation. *Journal of the American Statistical Association*, 113(521):182–201, 2018.

21   Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In Serge Vaudenay, editor, *EUROCRYPT 2006, St. Petersburg, Russia, May 28 – June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 486–503. Springer, 2006. `doi:10.1007/11761679_29`.

22   Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, pages 265–284, Berlin, Heidelberg, 2006. Springer-Verlag.

23   Cynthia Dwork, Frank McSherry, and Kunal Talwar. The price of privacy and the limits of LP decoding. In *Proceedings of the Thirty-Ninth ACM STOC*, pages 85–94. ACM, 2007.

24   Cynthia Dwork and Sergey Yekhanin. New efficient attacks on statistical disclosure control mechanisms. In *CRYPTO*, pages 469–480. Springer, 2008.

25   Jean-Pierre Eckmann and Elisha Moses. Curvature of co-links uncovers hidden thematic layers in the world wide web. *Proceedings of the national academy of sciences*, 99(9):5825–5829, 2002.

26   Talya Eden, Amit Levi, Dana Ron, and C Seshadhri. Approximately counting triangles in sublinear time. *SIAM Journal on Computing*, 46(5):1603–1646, 2017.

27   Talya Eden, Quanquan C. Liu, Sofya Raskhodnikova, and Adam D. Smith. Triangle counting with local edge differential privacy. *CoRR*, abs/2305.02263, 2023. `doi:10.48550/arXiv.2305.02263`.

28   Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC CCS*, CCS '14, pages 1054–1067, New York, NY, USA, 2014.

29   Alexandre V. Evfimievski, Johannes Gehrke, and Ramakrishnan Srikant. Limiting privacy breaches in privacy preserving data mining. In Frank Neven, Catriel Beeri, and Tova Milo, editors, *Proceedings of the Twenty-Second ACM SIGACT-SIGMOD-SIGART PODS, June 9-12, 2003, San Diego, CA, USA*, pages 211–222. ACM, 2003. `doi:10.1145/773153.773174`.

30   Illés J Farkas, Imre Derényi, Albert-László Barabási, and Tamas Vicsek. Spectra of "real-world" graphs: Beyond the semicircle law. *Physical Review E*, 64(2):026704, 2001.

31   Tianchong Gao, Feng Li, Yu Chen, and XuKai Zou. Local differential privately anonymizing online social networks under hrg-based model. *IEEE Transactions on Computational Social Systems*, 5(4):1009–1020, 2018.

32   Anupam Gupta, Aaron Roth, and Jonathan R. Ullman. Iterative constructions and private data release. In Ronald Cramer, editor, *Theory of Cryptography – 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings*, volume 7194 of *Lecture Notes in Computer Science*, pages 339–356. Springer, 2012. `doi:10.1007/978-3-642-28914-9_19`.

33   Jacob Imola, Takao Murakami, and Kamalika Chaudhuri. Locally differentially private analysis of graph statistics. In *30th USENIX Security Symposium, USENIX Security 2021, August 11-13, 2021*, pages 983–1000, 2021. URL: `https://www.usenix.org/conference/usenixsecurity21/presentation/imola`.

34   Jacob Imola, Takao Murakami, and Kamalika Chaudhuri. Communication-efficient triangle counting under local differential privacy. In *31st USENIX Security Symposium, USENIX Security 2022, August 10-12, 2022*, 2022. `arXiv:2110.06485`.

35   Jacob Imola, Takao Murakami, and Kamalika Chaudhuri. Differentially private subgraph counting in the shuffle model. *CoRR*, abs/2205.01429, 2022. `arXiv:2205.01429`, `doi:10.48550/arXiv.2205.01429`.

36   Matthew Joseph, Jieming Mao, Seth Neel, and Aaron Roth. The role of interactivity in local differential privacy. In David Zuckerman, editor, *60th IEEE Annual FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 94–105. IEEE Computer Society, 2019. `doi:10.1109/FOCS.2019.00015`.

37   John Kallaugher and Eric Price. A hybrid sampling scheme for triangle counting. In *Proceedings of the Annual ACM-SIAM SODA*, pages 1778–1797, 2017.

38   Vishesh Karwa, Sofya Raskhodnikova, Adam D. Smith, and Grigory Yaroslavtsev. Private analysis of graph structure. *ACM Trans. Database Syst.*, 39(3):22:1–22:33, 2014. `doi:10.1145/2611523`.

39   Vishesh Karwa, Aleksandra B. Slavković, and Pavel Krivitsky. Differentially private exponential random graphs. In *Privacy in Statistical Databases*, pages 143–155. Springer International Publishing, 2014.

40   Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

41   Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. Analyzing graphs with node differential privacy. In Amit Sahai, editor, *10th Theory of Cryptography Conference, TCC 2013, Tokyo, Japan, March 3-6, 2013. Proceedings*, volume 7785 of *Lecture Notes in Computer Science*, pages 457–476. Springer, 2013. `doi:10.1007/978-3-642-36594-2_26`.

42   Shiva Prasad Kasiviswanathan, Mark Rudelson, Adam Smith, and Jonathan Ullman. The price of privately releasing contingency tables and the spectra of random matrices with correlated rows. In *Proceedings of the 42nd ACM STOC*, STOC '10, pages 775–784. ACM, 2010.

43   Shiva Prasad Kasiviswanathan, Mark Rudelson, and Adam D. Smith. The power of linear reconstruction attacks. In Sanjeev Khanna, editor, *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1415–1433. SIAM, 2013. `doi:10.1137/1.9781611973105.102`.

44   Andrew McGregor and Sofya Vorotnikova. Triangle and four cycle counting in the data stream model. In *ACM SIGMOD-SIGACT-SIGART PODS*, pages 445–456, 2020.

45   Andrew McGregor, Sofya Vorotnikova, and Hoa T Vu. Better algorithms for counting triangles in data streams. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI PODS*, pages 401–411, 2016.

46   Ron Milo, Shai Shen-Orr, Shalev Itzkovitz, Nadav Kashtan, Dmitri Chklovskii, and Uri Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

47   Kobbi Nissim, Sofya Raskhodnikova, and Adam D. Smith. Smooth sensitivity and sampling in private data analysis. In David S. Johnson and Uriel Feige, editors, *Proceedings of the 39th Annual ACM Symposium on Theory of Computing, San Diego, California, USA, June 11-13, 2007*, pages 75–84. ACM, 2007. Full paper: `http://www.cse.psu.edu/~asmith/pubs/NRS07.` `doi:10.1145/1250790.1250803`.

48   Rasmus Pagh and Charalampos E Tsourakakis. Colorful triangle counting and a mapreduce implementation. *Information Processing Letters*, 112(7):277–281, 2012.

49   Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, 2005.

50   Ha-Myung Park, Francesco Silvestri, U Kang, and Rasmus Pagh. Mapreduce triangle enumeration with guarantees. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 1739–1748, 2014.

51   Arnau Prat-Pérez, David Dominguez-Sal, Josep M. Brunat, and Josep Lluís Larriba-Pey. Put three and three together: Triangle-driven community detection. *ACM Trans. Knowl. Discov. Data*, 10(3):22:1–22:42, 2016. `doi:10.1145/2775108`.

52   Zhan Qin, Ting Yu, Yin Yang, Issa Khalil, Xiaokui Xiao, and Kui Ren. Generating synthetic decentralized social graphs with local differential privacy. In *Proceedings of the 2017 ACM SIGSAC CCS*, CCS '17, pages 425–438, New York, NY, USA, 2017.

53   Sofya Raskhodnikova and Adam D. Smith. Differentially private analysis of graphs. In *Encyclopedia of Algorithms*, pages 543–547. Springer, 2016. `doi:10.1007/978-1-4939-2864-4_549`.

54   Haipei Sun, Xiaokui Xiao, Issa Khalil, Yin Yang, Zhan Qin, Hui (Wendy) Wang, and Ting Yu. Analyzing subgraph statistics from extended local views with decentralized differential privacy. In *Proceedings of the 2019 ACM CCS*, CCS '19, pages 703–717, New York, NY, USA, 2019.

55   Stanley L Warner. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

56   Qingqing Ye, Haibo Hu, Man Ho Au, Xiaofeng Meng, and Xiaokui Xiao. Towards locally differentially private generic graph metric estimation. In *2020 IEEE 36th ICDE*, pages 1922–1925, 2020.