Almost Tight Bounds for Differentially Private Densest Subgraph[*]

Michael Dinitz[†] Satyen Kale[‡] Silvio Lattanzi[§] Sergei Vassilvitskii[¶]

Abstract

We study the Densest Subgraph (DSG) problem under the additional constraint of differential privacy. DSG is a fundamental theoretical question that plays a central role in graph analytics, and so privacy is a natural requirement. All known private algorithms for Densest Subgraph lose constant multiplicative factors, despite the existence of non-private exact algorithms. We show that, perhaps surprisingly, this loss is not necessary: in both the classic differential privacy model and the LEDP model (local edge differential privacy, introduced recently by Dhulipala et al. [FOCS 2022]), we give (ϵ, δ)-differentially private algorithms with no multiplicative loss whatsoever. In other words, the loss is *purely additive*. Moreover, our additive losses match or improve the previous state of the art additive loss (in any version of differential privacy) when $1/\delta$ is polynomial in n, and are almost tight: in the centralized setting, our additive loss is $O(\log n/\epsilon)$ while there is a known lower bound of $\Omega(\sqrt{\log n/\epsilon})$.

We also give a number of extensions. First, we show how to extend our techniques to both the node-weighted and the directed versions of the problem. Second, we give a separate algorithm with pure differential privacy (as opposed to approximate DP) but with worse approximation bounds. And third, we give a new algorithm for privately computing the optimal density which implies a separation between the structural problem of privately computing the densest subgraph and the numeric problem of privately computing the density of the densest subgraph.

1 Introduction

Differential privacy has become the *de facto* standard for private data analysis. It is often applied to data which is inherently numeric, but there is growing interest in providing solutions on more structural inputs, particularly on graphs. In this setting, much work has focused on private computation of numerical functions of a graph, e.g., the number of triangles. But there is nothing in the definition of differential privacy which requires numeric output, and indeed, it is often possible to privately compute *objects* which are approximately optimal solutions (sometimes represented explicitly, sometimes implicitly) to combinatorial optimization problems on graphs. This dates back to [38,57], and there is now a relatively developed literature on private graph algorithms [3,17,21,27,30,31,39].

In this work we study the *densest subgraph problem* (DSG), one of the most fundamental algorithmic problems in data mining and graph analytics [1,2,16,22,36,49,61], in classic edge differential privacy and in the *local edge differential privacy* (LEDP) [21] models. In the DSG problem we are asked to find a subset S of nodes which maximizes the number of edges in the subgraph induced by S divided by the size of S, i.e., which maximizes half of the average induced degree. This value is known as the *density* of S, and is denoted by $\rho(S) = |E(S)|/|S|$. Densest subgraph has been studied extensively in the non-private setting, and is well-known to admit an exact polynomial-time algorithm based on flow [37], as well as a fast greedy 2-approximation [13] and fast streaming, parallel and dynamic algorithms [2,5,7,28,29,34,52,62].

Due to its importance in data mining and its multiple applications in diverse domains, including bioinformatics, network science, fraud detection, and social network analysis [11,12,36,44,60], differentially private versions of DSG have recently been developed [20,21,32,56]. This line of work focuses on *edge*-differential privacy, where the private information is the set of edges (not the set of nodes). This setting is of practical importance for DSG as

[*]The full version of the paper can be accessed at https://arxiv.org/abs/2308.10316

[†]Department of Computer Science, Johns Hopkins University, Baltimore, MD. mdinitz@cs.jhu.edu. Supported in part by NSF grants CCF-1909111 and CCF-2228995. Work partially done while a Visiting Researcher at Google Research New York.

[‡]Google Research. Currently at Apple. satyen@satyenkale.com.

[§]Google Research. silviol@google.com.

[¶]Google Research. sergeiv@google.com.

Privacy Model	Approximation Ratio	Reference / Notes
ϵ-DP	$\left(2, O\left(\frac{1}{\epsilon}\log^{3.5} n\right)\right)$	High probability version in [32]
	$\left(1+\eta, O\left(\frac{1}{\epsilon}\log^{4} n\right)\right)$	[21]
ϵ-LEDP	$\left(4+\eta, O\left(\frac{1}{\epsilon}\log^{3} n\right)\right)$	[21]
	$\left(2, O\left(\frac{1}{\epsilon}\log n\right)\right)$	[20] (Concurrent with this paper)
	$\left(2+\eta, O\left(\frac{1}{\epsilon}\frac{1}{\eta}\log^{2} n\right)\right)$	This work (Corollary 6.2)
(ϵ, δ)-DP	$\left(2, O\left(\frac{1}{\epsilon}\log\frac{1}{\delta}\log n\right)\right)$	[56]
	$\left(1, O\left(\frac{1}{\epsilon}\sqrt{\log n \log \frac{n}{\delta}}\right)\right)$	This work (Theorem 3.6)
	$\left(\alpha, \Omega\left(\frac{1}{\alpha}\sqrt{\frac{1}{\epsilon}\log n}\right)\right)$ for any $\alpha \geq 1$	Lower bound: [32] (high probability, $\delta \leq 1/n$)
	$\left(1, \Omega\left(\sqrt{\frac{1}{\epsilon}\log n}\right)\right)$	Lower bound: [56] (expectation, $\delta \leq 1/n$)
(ϵ, δ)-LEDP	$\left(1, O\left(\frac{1}{\epsilon}\log n\sqrt{\log\frac{1}{\delta}}\right)\right)$	This work (Corollary 3.2)

Table 1: A summary of results on differentially private DSG. Different privacy models represent different assumptions on the attacker. Recall that ϵ-LEDP is a stronger privacy guarantee than ϵ-DP, which is, in turn stronger than (ϵ, δ)-DP. On the other hand, (ϵ, δ)-LEDP is stronger than (ϵ, δ)-DP, but is incomparable with ϵ-DP.

well as for other clustering problems. In fact, in many practical scenarios one is interested in analyzing the structure of the network without leveraging private information contained in the edges. For example, in social network analysis one is interested in network statistics without revealing information about the connection between two specific nodes, or in messaging networks where one wants to analyze the networks without revealing the frequency of messages between two nodes. This is critical in settings where the messages are confidential and should not be leaked(for example consider messages between recruiters and employed candidates or, even more importantly, between political dissidents). In many of these settings we might also want *local* edge-differential privacy (LEDP), where rather than trusting a central curator, the individual nodes communicate with an untrusted curator in a way that protects their private edge information [21] (see Section 1.2 for definitions of these models). For these reasons, even beyond DSG, several other problems have been studied in the local edge differential privacy setting or the edge differential privacy setting in the data mining and machine learning literature [9, 15, 17, 26, 50, 55].

Letting S^* denote the optimal solution, we say that an algorithm is an (α, β)-approximation if it outputs a set S with $\rho(S) \geq \frac{1}{\alpha}\rho(S^*) - \beta$. There has been significant recent and concurrent work on private DSG with nontrivial approximation guarantees in most model variants: ϵ-DP [21, 32], ϵ-LEDP [20, 21], and (ϵ, δ)-DP [56]. See Table 1 for a summary of these results. We note that the most interesting regime for δ is when it is inverse polynomial in n, so we will usually think of $\log(1/\delta)$ as being on the same order as $\log n$.

The most notable feature of all of the known results is that *they all incur multiplicative loss*. This is the case even though exact algorithms with no loss exist in the non-private setting [37]. The smallest multiplicative loss is $1 + \eta$ (from [21]), which incurs an additional $O(\frac{1}{\epsilon}\log^4 n)$ additive loss and is not in the local model. But while it is known that additive loss is necessary even if there is also multiplicative loss [32], there is no known lower bound which indicates that multiplicative loss is actually necessary. This is the fundamental question that we attack: is it possible to design a private DSG algorithm with no multiplicative loss whatsoever, and still only small (polylogarithmic) additive loss? If so, how small can we make the additive loss?

1.1 Our Results and Techniques

1.1.1 Main result: (ϵ, δ)-(L)EDP Our main set of results gives a surprising answer to the above question: in both the (ϵ, δ)-DP model and the (ϵ, δ)-LEDP model, it is possible to design a private algorithm with no multiplicative loss whatsoever! Slightly more carefully, we design a polynomial time algorithm in the LEDP model and prove that it is (ϵ, δ)-differentially private and that it returns a $\left(1, O\left(\frac{1}{\epsilon}\log n\sqrt{\log(1/\delta)}\right)\right)$-approximation with high probability. Note that this is the first purely additive approximation in *any* version of differential privacy.

We can then improve this LEDP algorithm in the classic (ϵ, δ)-DP setting to achieve nearly optimal (and state of the art) results in term of additive and multiplicative bound at the same time. In particular, we give a $\left(1, O\left(\frac{1}{\epsilon}\sqrt{\log n \log(n/\delta)}\right)\right)$-approximation with high probability. We emphasize that this is almost tight: in the regime where δ is an inverse polynomial we are only a $\sqrt{\log n/\epsilon}$ factor away from the known lower bounds.

While our focus is on achieving no multiplicative loss, we note that our algorithms also have better additive loss than any previous result. And compared to the independent concurrent result of [20], we achieve the same additive loss in the centralized setting and only $\sqrt{\log(1/\delta)}$ more additive loss in the local setting, while not losing anything multiplicatively (compared to their multiplicative factor 2 loss).

Finally, we observe that our algorithms are simple to run and implement: all of the complexity lies in the analysis bounding the exact amount of noise we have to add, and in the analysis of the resulting techniques.

Techniques. At a very high level, our algorithm uses the classical multiplicative weights method to privately solve an LP formulation of DSG. However, standard settings and analyses of differentially private multiplicative weights do not suit our needs, and standard methods of privately solving LPs using multiplicative weights are also too weak for us. So we need to design a new private multiplicative weights algorithm, and show via a complex and delicate analysis that it can be used in a noisy version of the classical Plotkin, Shmoys, Tardos framework [59] to privately solve the LP (at least approximately). While differentially private algorithms have been introduced in the past to solve linear programs [45], we are able to leverage the problem structure to obtain substantially stronger bounds on the quality of the solution.

We first show (in Section 2) that the classical Multiplicative Weights Update/Hedge algorithm can be extended to handle *noisy* losses (adversarial losses plus random noise) while still having low regret compared to the adversarial losses. This allows us to give a differentially private version of Hedge. To the best of our knowledge this is the first analysis of the multiplicative update algorithm with noisy updates for differential privacy and we expect it to be a useful technique of independent interest.[1]

Then in Section 3 we use Hedge with noisy losses in combination with an idea from a recent result of Chekuri, Quanrud, and Torres [14] in the non-private setting to get our main result. Their goals were quite different: they wanted to give provable bounds on a heuristic called Greedy++ due to Boob et al. [8] which seems to work extremely well in practice despite a lack of theoretical guarantees, and they wanted to extend beyond densest subgraph to the more general setting of "Densest Supermodular Subset" (DSS). To do this, they showed that Greedy++ (and an extension to DSS which they call "Super-Greedy++") can be interpreted as actually being a multiplicative-weights algorithm which approximately solves a highly non-obvious LP formulation of DSG in the Plotkin, Shmoys, Tardos (PST) [59] LP-solving framework, resulting in a $(1 + \epsilon)$-multiplicative approximation.

Taking this algorithm as our starting point, we want to add privacy. It turns out that we can again show that a simple combinatorial algorithm based on counting (noisy) degrees can be "interpreted" as running multiplicative-weights, and in particular as running Hedge with noisy losses in the PST framework. Interestingly, there is a well-known differentially private version of multiplicative weights [42], so one might hope that we can simply plug it into the algorithm and analysis of [14]. Unfortunately this fails, since it turns out that the versions of privacy and noise that we want are essentially orthogonal to those achieved by [42]. In particular, in order to preserve privacy we need to add (carefully chosen) noise to the updates in multiplicative weights, and be able to precisely figure out the true optimum of the Lagrangean relaxation without any noise. In [42], on the other hand, the authors update the weights *exactly* but choose the query (i.e., solve the Lagrangean relaxation) only approximately. Moreover,

[1]Note that our paper is not the first paper analyzing multiplicative updates in the field of differential privacy [39–42], although previous work focused on answering linear queries and the noise is added to determine whether to "update" in current round. Importantly, updates in this setting are always exact and so their analysis is orthogonal to our result and cannot be used in our context.

the algorithm of [42] requires knowledge of the "true" answer, and so cannot be implemented in the local model. Similarly, there are well-known ways of privately solving LPs [45], but these algorithms were developed for generic LPs, rather the specific LP arising from DSG, they do not give strong enough bounds for our purposes (and are also not in the local model).

The limitations of [45] show the difficulty of "simply" applying Hedge with noisy losses to the PST framework: for generic LPs, this approach does not work (see the discussion at the beginning of Section 3). Fortunately, we can leverage the special structure of the DSG LP. Even with that structure, since we have noisy updates in multiplicative weights, we end up with a natural tension. The more iterations that multiplicative weights runs the more accurate its answer (it converges at a rate of roughly $1/\sqrt{T}$, where T is the number of iterations). But the more iterations it runs the more noise we need to add to maintain privacy due to sequential composition. It is not clear which of these "wins out". For ϵ-DP, unfortunately the noise wins: running for T iterations requires adding noise on the order of T, which dominates the $1/\sqrt{T}$ convergence. But by moving to (ϵ, δ)-DP, we are able to add significantly less noise, on the order of $\sqrt{T}$. So now we need to add $\sqrt{T}$ noise but get convergence at a rate of $1/\sqrt{T}$. These almost exactly balance out, but we can show through a sufficiently delicate analysis that the noise required grows *slower* than the accuracy obtained, giving us the claimed bounds.

Comparison to Previous Techniques. Most previous methods of privately solving DSG are not based on LP methods, and usually are based on the classical peeling algorithm of [13] with appropriate noise added. In particular, [32] uses clever techniques to give a very fast (centralized) version of that algorithm, while [56] and the LEDP algorithm of [21] take inspiration from parallel algorithms (in different ways). The exception is the centralized DP algorithm of [21], which is related to the basic LP relaxation of Charikar [13]. As discussed, we obtain our stronger bounds by using the stronger LP relaxation of [14] and carefully analyzing Noisy Hedge combined with [59] in that context.

1.1.2 Extensions and Other Results

We give a number of extensions and other results; since they are not the main contribution of this paper, we will only discuss them here briefly. See the appropriate sections for details.

Weights. In Section 4 we show that an extension of our techniques allows us to give similar bounds in the presence of node weights, i.e., where there is a weight c_v for every node v and our goal is to find the set $S \subseteq V$ which maximizes $|E(S)|/\sum_{v \in V} c_v$. This is, to the best of our knowledge, the first known bound for differentially private node-weighted DSG. However, we must assume that $c_v \geq 1$ for all $v \in V$. We note that this is a relatively standard assumption in the literature, see, e.g., [63][2].

Unfortunately, in the LEDP setting, we lose both an arbitrarily small multiplicative factor as well as an additional additive loss; informally, our algorithm is (ϵ, δ)-LEDP and with high probability is a $\left(1+\eta, O\left(\frac{\log^{3/2} n\sqrt{\log(1/\delta)}}{\eta^{1.5}\epsilon}\right)\right)$-approximation (see Corollary 4.2 for the precise statement). This arises from the fact that in the weighted setting, we have to *actually run* multiplicative weights, unlike in the unweighted setting where we could show that a combinatorial algorithm can be "interpreted" as running multiplicative weights. Doing this requires binary search over an unknown parameter, causing the extra loss. In the centralized setting, on the other hand, we do not suffer any multiplicative error and get the same additive error as in the centralized unweighted setting (see Theorem 4.4).

Directed. In Section 5 we show that a further extension of our main techniques allows us to design an (ϵ, δ)-LEDP algorithm for *directed* densest subgraph [13, 47, 63], where we are given a directed graph and our goal is to find sets $S, T \subseteq V$ to maximize $\frac{|E(S,T)|}{\sqrt{|S||T|}}$. As in the weighted setting, this is the first known bound for differentially-private directed DSG. It was recently shown that in the non-private setting there is a reduction from the directed version to the undirected node-weighted version [63], so we can use our extension to vertex-weighted graphs. Unfortunately, this reduction requires node weights that are less than 1, violating our assumption. So we only get a weaker bound, with additional additive loss that depends on how "balanced" the optimal solution is. In particular, if (S^*, T^*) is the optimal solution, let $b = \max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)$. Then (informally) in the centralized setting we give an (ϵ, δ)-DP algorithm which with high probability is a $\left(1, O\left(\frac{1}{\epsilon}\sqrt{b \log n \log(n/\delta)}\right)\right)$. See Corollary 5.2 for the precise statement in the LEDP setting and Theorem 5.3 for the centralized setting.

[2] We remark that it is obviously impossible to handle edge weights under edge differential privacy, because with arbitrary edge weights two neighboring databases could have arbitrarily different optimal densities.

ϵ-LEDP. As a secondary result, in Section 6 we also give an algorithm with improved bounds in the ϵ-LEDP model. We show a very simple algorithm which is ϵ-LEDP and is a $(2+\eta, O(\frac{1}{\epsilon\eta}\log^2 n))$-approximation, and which moreover can be parallelized to run in $O(\frac{1}{\eta}\log n)$ rounds. This is an improvement to the ϵ-LEDP algorithm of [21], and as a side benefit is significantly simpler than the previous ϵ-DP algorithms. Its bounds are dominated by the concurrent work of [20], but on the other hand our algorithm runs in $O(\frac{1}{\eta}\log n)$ rounds, while [20] requires $O(n)$ rounds.

Our main technical idea is to start with Charikar's sequential 2-approximation [13], as was done in [56] and [32]. Both of the previous papers show that through very clever analysis, it is possible to prove privacy guarantees much stronger than would be obtained by simply using sequential composition for the n iterations of Charikar's algorithm. Instead of carefully reasoning about composition of noise as in those papers, we take a different and far simpler approach: we use the parallel version of Charikar's algorithm from [5] to get an algorithm that only requires $O(\log n)$ rounds, and then use parallel composition [25]. Since there is a small number of rounds we do not need to add much noise. A few more relatively straightforward modifications allow us to do this in the ϵ-LEDP model.

Numeric approximation. A natural question in many settings is whether it is possible to privately output a *structure* with the same quality guarantees that we would get if we only cared about outputting the *value* of the structure. For example, it was recently shown that while the *value* of the min $s-t$ cut can be computed privately with small noise (constant in expectation, logarithmic with high probability), actually outputting the cut itself requires a *linear* additive loss [19]. One interpretation of our main result is that DSG does not exhibit this phenomenon: outputting the structure requires loss that is "close" to the noise required to output the value (no multiplicative loss, polylogarithmic additive loss).

But how close is close? That is, from a more fine-grained perspective, how much loss is necessary when outputting the density of the densest subgraph, rather than the node set? It is not hard to see that the density has global sensitivity of at most 1, and so the Laplacian or Gaussian mechanism can be used to output a value with additive loss that is at most $1/\epsilon$ in expectation and at most $O(\log n/\epsilon)$ with high probability (for the Gaussian mechanism, assuming that δ is at most inverse polynomial in n).

We show that it is actually possible to do better. Intuitively, if the density is small then we are free to give an inaccurate answer, while if it is large then this implies (by the definition of density) that the optimal subgraph actually has many nodes and edges, and hence the sensitivity of the density is significantly less than 1. We formalize this intuition, showing that a simple variant of the propose-test-release framework [23] can be used to give algorithms that lose only $\sqrt{1/\epsilon}$ in expectation or $\sqrt{\log n/\epsilon}$ with high probability. Note that the lower bound of [56] of $\Omega(\sqrt{\log n/\epsilon})$ for computing the densest subgraph actually holds even in expectation, and hence our upper bound of $O(\sqrt{1/\epsilon})$ provides a separation between the value and the structure. So while the cost of computing the structure and computing the value are "close" insofar as neither requires multiplicative loss and the both require small additive loss, there is actually a separation between them if we take a more fine-grained look at the additive loss necessary.

1.1.3 Followup and Concurrent Work

Since the initial posting of this paper, there have been two pieces of concurrent or followup work, both of which have focused on the related *k-core problem* but have also given results on DSG. Due to their focus on k-core, their techniques are quite different than ours, and in particular cannot be used to obtain a private algorithm for DSG with purely additive loss (there is a required multiplicative loss of 2 when using k-core for DSG, and unlike us they do not use LP-based methods).

First, as discussed, [20] gave a $(2, O(\log n/\epsilon))$-approximation in the ϵ-LEDP model. This improves on the accuracy of our ϵ-LEDP algorithm, but compared to to our main (ϵ,δ)-LEDP algorithm has a multiplicative 2 loss rather than being purely additive (and has essentially the same additive loss). Moreover, their algorithm takes a linear number of rounds, while our ϵ-LEDP algorithm is arguably simpler and takes only $O(\log n)$ rounds.

Second, [43] gave similar but slightly weaker bounds. They first obtain a $(2, O(\log^2 n))$-approximation in the ϵ-LEDP model. This does not match [20] or our main (ϵ,δ)-(LE)DP result), but it slightly beats our $2+\eta$ multiplicative loss in the ϵ-LEDP model. However, it requires a linear number of rounds. They also give a $(4+\eta, O(\log n \log\log n))$-approximation in $O(\log^2 n)$ rounds. This is a factor two multiplicatively worse than our ϵ-LEDP algorithm while also being slower, but improves the additive loss to $O(\log n \log\log n)$ rather than our $O(\log^2 n)$.

We note that neither our techniques nor any of the techniques from previous or concurrent work seem to be able to get purely additive error with *pure* DP. Designing such an algorithm is an intriguing open question.

1.2 Preliminaries and Notation

We use $\log(\cdot)$ to denote the natural logarithm. Binary logarithm is specified as $\log_2(\cdot)$. For a natural number n, we let $[n] := \{1, 2, \ldots, n\}$.

Given a graph $G = (V, E)$, and a subset $S \subseteq V$, let $\rho(S) = |E(S)|/|S|$, where $E(S) \subseteq E$ is the set of edges induced by S (i.e., with both endpoints in S). Let $\rho(G) = \max_{S \subseteq V} \rho(S)$.

Definition 1.1. *The* Densest Subgraph Problem *(DSG) is the optimization problem in which we attempt to find an S maximizing $\rho(S)$, i.e., find an S with $\rho(S) = \rho(G)$. In the presence of node weights $\{c_v\}_{v \in V}$, we redefine the density to be $\rho(S) = |E(S)|/\sum_{v \in S} c_v$. If the edges are directed, then we let $E(S, T) = \{(u, v) \in E : u \in S, v \in T\}$ and define the density with reference to two sets: $\rho(S, T) = |E(S, T)|/\sqrt{|S| \cdot |T|}$.*

For a vertex $v \in V$ and a set $S \subseteq V$, let $d_S(v)$ denote the degree of v into S, i.e., the number of edges incident on v with other endpoint in S. Let $d_{avg}(S) = \sum_{v \in S} d_S(v)/|S|$ denote the average degree in the subgraph induced by S. Note that $d_{avg}(S) = 2 \cdot \rho(S)$.

Given a randomized algorithm $\mathcal{A}$ for the DSG problem, differential privacy captures the impact of small changes in the input (captured by the neighboring relation) on the distribution of outputs. See the excellent book by Dwork and Roth [25] for a thorough introduction to the topic.

There are two natural notions of differential privacy on graphs: node differential privacy, where two graphs are considered neighboring if they differ in one node, and edge differential privacy, where two graphs are considered neighbors if they differ in one edge. We will be concerned with edge differential privacy in this paper.

Definition 1.2. (Edge-Neighboring [57]) *Graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are* edge-neighboring *if they differ in one edge, i.e., if $V_1 = V_2$ and the size of the symmetric difference of E_1 and E_2 is 1.*

Definition 1.3. (Edge Differential Privacy [57]) *Algorithm $\mathcal{A}(G)$ that takes as input a graph G and outputs an object in $R(\mathcal{A})$ is* (ϵ, δ)-edge differentially private *((ϵ, δ)-edge DP) if for all $S \subseteq R(\mathcal{A})$ and all edge-neighboring graphs G and G',*

$$\Pr[\mathcal{A}(G) \in S] \leq e^{\epsilon} \Pr[\mathcal{A}(G') \in S] + \delta$$

If an algorithm is $(\epsilon, 0)$-edge DP then we say that it is ϵ-edge DP (or ϵ-DP).

In the case of (ϵ, δ)-DP, most algorithms give a trade-off between the ϵ and δ they achieve. To make use of this trade-off in our analysis, we will use the concept of zCDP:

Definition 1.4. (Zero-Concentrated Differential Privacy(zCDP) [10]) *Algorithm $\mathcal{A}(G)$ that takes as input a graph G and outputs something in $R(\mathcal{A})$ is* ρ-zCDP *if for all $\alpha \in (1, \infty)$ and all edge-neighboring graphs G and G',*

$$D_\alpha(\mathcal{A}(G) || \mathcal{A}(G')) \leq \alpha\rho,$$

where D_α is the Rényi divergence of order α.

Importantly, zCDP and (ϵ, δ)-edge DP are connected and a result in one setting can be translated in the other. In essentially all of our analysis we will use zCDP, and then we will translate into (ϵ, δ)-edge DP using standard results from [10, 54].

Local DP. Finally, all of the definitions above assume the existence of a trusted central curator who can execute the algorithm $\mathcal{A}$. In *local* differential privacy, instead of assuming a trusted curator we assume that each agent controls its own data and there is an *untrusted* curator who does not initially know the database, but can communicate with the agents. We require that the entire *transcript* of communication and outputs satisfies differential privacy, so even the curator cannot learn private information. This model was originally introduced by [48] in the context of learning, and has since become an important model of privacy (including in practice; see, for example, the discussion in [18]). In the context of graphs and edge-DP, this corresponds to the *local* edge-differential privacy model from [21] (suitably modified to handle (ϵ, δ)-DP rather than just pure ϵ-DP). In this model there is a curator which initially knows only the vertex set (and thus the number of vertices n), and each node initially knows its incident edges, but nothing else (except possibly n, which can be sent by the curator initially). During each round, the curator first queries a set of nodes for information. Individual nodes, which have access only to their own (private) adjacency lists (and whatever information was sent by the curator), then send information back to the curator. However, since we require that the entire transcript satisfy (ϵ, δ)-DP, without loss of generality all transmissions are really *broadcasts* rather than point-to-point. The actual formalization of this model is somewhat involved, so we defer it to Appendix A (or to [21]).

We will use a number of standard DP mechanisms (the Laplacian, geometric, and Gaussian mechanisms in particular) and composition theorems (parallel composition, adaptive sequential composition, and advanced composition), as well as the standard post-processing theorem. We give all of these formally in Appendix A, but they are all well-known and can be found in standard textbooks [25,64]. It was observed by [21] that they all hold in the LEDP model as well (see Appendix A for more discussion). Since we use the Gaussian mechanism frequently, we will use $N(\mu, \tau^2)$ to denote the normal (Gaussian) distribution with mean μ and variance τ^2 (so with standard deviation τ).

2 Hedge with Noisy Losses

In this section we analyze the classic Hedge/Multiplicative Weights Update (MWU) algorithm for the fundamental online learning setting of prediction with expert advice. The main twist here is that the expert losses can be noisy. This will be useful in developing our algorithms for DSG which add noise for preserving privacy.

The setting of online learning with expert advice is as follows. In each of T rounds, indexed by $t = 1, 2, \ldots, T$, we have access to n "experts", and are required to choose a distribution over the experts, after which losses of all experts are revealed. The losses are given by a random vector $\hat{m}^{(t)} \in \mathbb{R}^n$. Let $m^{(t)} = \mathbb{E}[\hat{m}^{(t)} \mid \hat{m}^{(1)}, \hat{m}^{(2)}, \ldots, \hat{m}^{(t-1)}]$. We assume that $m^{(t)} \in [-1, 1]^n$, and that the distribution $\hat{m}^{(t)} - m^{(t)}$, conditioned on all $\hat{m}^{(s)}$ for $s < t$, is sub-Gaussian with variance proxy $\nu^2 \leq 1$. In each round t, we are required to output a distribution on the experts $p^{(t)}$, and suffer the expected loss $\langle \hat{m}^{(t)}, p^{(t)} \rangle$. The regret of an online algorithm for this problem is the difference between the expected cumulative loss of the algorithm and the (expected) cumulative loss of the best fixed expert in hindsight. The goal is to develop an online algorithm with regret growing sublinearly in T.

Consider the classic Hedge algorithm [33] given as Algorithm 1. While it is usually used in the setting where losses are adversarial but bounded, we can provide a new regret bound for it in the above setting with unbounded but noisy losses. See Appendix B for the proof of the following theorem. The proof is essentially along the lines of the standard proof; the main change needed is the use log-sum-of-weights as a potential function and an application of Jensen's inequality.

Algorithm 1 Hedge(T)

```
1: Set $\eta = \sqrt{\frac{\log n}{T}}$.
2: Set $w_i^{(1)} = 1$ for all $i \in [n]$.
3: for $t = 1$ to $T$ do
4:     Output the distribution $p^{(t)}$ such that $p_i^{(t)} \propto w_i^{(t)}$ for all $i \in [n]$.
5:     Observe $\hat{m}^{(t)}$.
6:     For all $i \in [n]$: set
               $w_i^{(t+1)} = w_i^{(t)} \cdot \exp(-\eta \hat{m}_i^{(t)})$.
7: end for
```

THEOREM 2.1. *Suppose $\nu \leq 1$ and $T \geq \log n$. The Hedge algorithm guarantees that after T rounds, for any expert $i \in [n]$, we have*

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle m^{(t)}, p^{(t)} \rangle\right] \leq \mathbb{E}\left[\sum_{t=1}^{T} m_i^{(t)}\right] + 4\sqrt{T \log(n)}.$$

We now discuss the application of this algorithm in a privacy-preserving setting. Rather than thinking of the distributions of $\hat{m}^{(t)}$ as the fundamental object and the $m^{(t)}$ values as their expectations, we can equivalently think of the $m^{(t)}$ values as being the fundamental objects and can add noise to them to get distributions with the correct expectations. So suppose that the loss vectors $m^{(t)}$ depend upon some underlying private dataset, and the past losses $m^{(s)}$ for $s < t$. Furthermore, assume that the ℓ_2 sensitivity (w.r.t. changes of a single item in the underlying dataset) of the loss vectors is bounded by Δ (see [25] for a formal definition of ℓ_2-sensitivity). The goal is to design an online learning algorithm with sublinear regret such that sequence of distributions over the experts output by the algorithm are collectively differentially private.

Consider the **DP-Hedge** algorithm, which simply uses Hedge with the noisy losses $\hat{m}^{(t)}$ constructed by adding independent Gaussian noise to the true losses:

$$\hat{m}_i^{(t)} \sim N(m_i^{(t)}, \nu^2), \quad \forall i \in [n].$$

Clearly, $\hat{m}^{(t)}$ satisfies the assumptions of Theorem 2.1, so the regret bound holds. The privacy guarantee of DP-Hedge is given below:

THEOREM 2.2. *The DP-Hedge algorithm is $\frac{\Delta^2 T}{2\nu^2}$-zCDP.*

Proof. This is a direct consequence of the fact that the Gaussian mechanism (Lemma A.3) ensures that each $\hat{m}^{(t)}$ is $\frac{\Delta^2}{2\nu^2}$-zCDP, and then appealing to the adaptive sequential composition theorem for zCDP (Theorem A.1). ∎

Note that the regret bound given in Theorem 2.1 is identical, up to constant factors, to that of the standard Hedge/Multiplicative Weights Update algorithm (see, e.g., [4]) despite potentially unbounded losses. This allows us to leverage Hedge with noisy losses in applications where losses can be unbounded but have bounded expectation, including specifically the Lagrangian relaxation approach to solving packing/covering LPs due to Plotkin, Shmoys and Tardos [59], which is relevant to the DSG problem.

3 Main Result: Purely Additive Private DSG

We now show our main result: it is possible to get purely additive logarithmic loss, even in the strong (ϵ, δ)-LEDP model. As discussed in Section 1.1, at a very high level, our approach follows the non-private algorithm of [14], by replacing the version of multiplicative weights that they use with our DP-Hedge algorithm from Section 2. In particular, they use the Plotkin, Shmoys, and Tardos (PST) framework [59] to solve an LP formulation of the problem via multiplicative weights. We similarly use PST on the same LP formulation, but with noise added to preserve privacy (i.e., with DP-Hedge). While this may sound straightforward, the interaction between the noise needed for privacy and the ability of the PST framework to solve the LP is technical, and requires significantly extending the standard analysis of PST. In fact, this is the main reason why we cannot give guarantees for pure ϵ-LEDP using this method: the noise we would have to add to preserve pure differential privacy would overwhelm the ability of PST to find a good solution.

We are certainly not the first to use multiplicative weights to solve LPs privately: most notably, see [45], which explored the limits of this approach. Unfortunately, the results of [45] are not applicable in our setting. Since they were attempting to solve an extremely general problem (privately solving LPs), they were not able to take advantage of any problem-specific structure, and so in many settings they were only able to give impossibility results. In particular, for the type of LP that we will be considering, one edge change will result in two constraints having many coefficients change by one. This is what they call a "high-sensitivity constraint-private LP", for which they were able to prove only lower bounds.

But we of course are not concerned with general LPs, but rather the specific LP corresponding to DSG. We show that this LP has some very nice features which, if we are careful enough in the analysis, allow us to solve it privately. Most notably, the PST framework [59] requires an "oracle" to solve the Lagrangian relaxation of the problem. This would normally require extra noise in order to compute it privately, and this extra noise could propagate throughout the computation to cause essentially a complete loss of utility. But for the specific LP we use, it turns out that this Lagrangian problem can be solved *exactly* even in the private setting!

We begin in Section 3.1 with some background from [14]; most importantly, the definition of the LP formulation. We then show in Section 3.2 that we can use DP-Hedge inside of the PST framework to solve this LP privately with purely additive loss, but with the limitation that we have to assume knowledge of the optimal value λ^* and we only succeed with constant probability. In Section 3.3 we give a private version of the useful "peeling" primitive. This allows us in Section 3.4 to finally give our full (ϵ, δ)-LEDP algorithm, and its full analysis in Section 3.5. Finally, in Section 3.6 we show how to give improved bounds in the centralized (rather than local) model.

3.1 Background From Chekuri, Quanrud, Torres [14]

In order to prove the approximation quality of our algorithm, we first need to give some background from [14]. They showed that a particularly simple non-private algorithm (essentially a variation of the Greedy++ algorithm of [8]) can be interpreted as running multiplicative-weights in the dual of a particular LP formulation for DSG.

We define some notation (mostly taken from [14]). Given an ordering σ of the vertices V, let $q(\sigma) \in \mathbb{R}^n$ be the vector where the coordinate for $v \in V$ is $q(\sigma)_v = |\{\{u, v\} \in E : u \prec_\sigma v\}|$. In other words, $q(\sigma)_v$ is the number of edges from v to nodes that precede v in σ. Another way of thinking about this is that if we "peeled" (removed) the vertices in the *reverse* order of σ, then $q(\sigma)_v$ would be the degree of v when it is removed (see Section 3.3). Given an ordering σ and a vertex v, let $S_v^\sigma = \{u \in V : u \preceq_\sigma v\}$ be the vertices in the prefix of σ defined by v, and let $E_v^\sigma = \{\{x, y\} \in E : x, y \in S_v^\sigma\}$ be the edges induced by S_v^σ. Given σ, let S_σ^* be the prefix of σ with maximum density, i.e., S_σ^* is the S_v^σ which maximizes $\rho(S_v^\sigma) = |E_v^\sigma|/|S_v^\sigma|$.

A particularly useful lemma from [14] is the following.

Lemma 3.1. (Lemma 4.2 of [14]) *For any $x \in [0, 1]^V$, we have that $\sum_{\{u,v\}\in E} \min\{x_u, x_v\} = \min_\sigma \langle x, q(\sigma)\rangle = \sum_{v\in V} x_v q(\sigma)_v$. Moreover, for every x we have that* $\operatorname{argmin}_\sigma \langle x, q(\sigma)\rangle$ *is the permutation σ which orders the vertices in nonincreasing order of values x_v.*

This allowed them to write the following covering LP formulation of DSG, where λ^* is the optimal density and S_V is the symmetric group of V (i.e., the set of all permutations of V).

$$\begin{aligned} \min \ & \sum_{v\in V} x_v & \\ \text{s.t. } & \langle x, q(\sigma)\rangle \geq \lambda^* & \forall \sigma \in S_V \\ & x_v \geq 0 & \forall v \in V \end{aligned}$$

It was shown in [14] (and it is not too hard to see) that for the correct value of λ^*, this LP is feasible with objective 1 (we refer the interested reader to [14] for more discussion about this LP, or to Section 4.2 for a discussion of a related generalized LP for the weighted setting). The dual of this LP is the following packing LP, which also (by strong duality) has objective value 1 for the correct value of λ^*:

$$\begin{aligned} \max \ & \lambda^* \sum_{\sigma\in S_V} y_\sigma & \\ \text{s.t. } & \sum_{\sigma\in S_V} q(\sigma)_v y_\sigma \leq 1 & \forall v \in V \\ & y_\sigma \geq 0 & \forall \sigma \in S_V \end{aligned}$$

3.2 Noisy PST to Solve LP Privately

3.2.1 The Algorithm We can now give the first algorithm that we analyze, which (as discussed) is what we would get if we run Hedge with noisy updates in the Plotkin, Shmoys, Tardos (PST) LP solving framework [59] applied to the dual packing LP. In particular, there will be an "expert" for each constraint (node), and we will keep track of a weight for each expert. We get Algorithm 2, which we call "Noisy-Order-Packing-MWU". It takes two parameters: T (the number of iterations to run) and τ (a parameter which controls the added noise and thus the privacy guarantee). We note that we are assuming knowledge of λ^*, which we do not actually know. We could do binary search over guesses of λ^* (which is what we do in the weighted setting, see Section 4), but this would incur an additional loss. Fortunately, we will be able to get around this by eventually using a simpler algorithm with the same behavior that does not need λ^* (see Section 3.4), so for now we will assume knowledge of λ^* for convenience.

We note that since the algorithm runs Hedge but passes it noisy losses, we will be able to use our previous analysis from Section 2.

Algorithm 2 Noisy-Order-Packing-MWU$(G=(V,E),T,\tau)$

```
1: Set $\rho = \frac{n+\tau}{\lambda^*}$ and $\nu = \frac{\tau}{\rho\lambda^*}$.
2: Instantiate Hedge$(T)$ with $n$ experts corresponding to nodes in $V$.
3: for $t = 1$ to $T$ do
4:     Obtain the distribution $p^{(t)}$ on $V$ from Hedge.
5:     Let $\sigma^{(t)}$ be the ordering of $V$ in nonincreasing order of $p^{(t)}$, breaking ties consistently, e.g., by node IDs.
       Set $y^{(t)}_{\sigma^{(t)}} = 1/\lambda^*$, and set $y^{(t)}_\sigma = 0$ for all $\sigma \neq \sigma^{(t)}$ (this is the Lagrangean oracle from PST [59]).
6:     For each $v \in V$, set the loss to be
           $$\hat{m}^{(t)}_v \sim N(m^{(t)}_v, \nu^2), \text{ where } m^{(t)}_v = \frac{1}{\rho}\left(1 - \sum_{\sigma\in S_V} q(\sigma)_v y^{(t)}_\sigma\right) = \frac{1}{\rho}\left(1 - \frac{q(\sigma^{(t)})_v}{\lambda^*}\right).$$
7:     Supply $\hat{m}^{(t)}$ as the loss vector to Hedge.
8: end for
9: Let $t$ be chosen uniformly at random from $[T]$.
10: return $(p^{(t)}, \sigma^{(t)})$.
```

Oracle. The PST framework requires an "oracle" that solves the Lagrangian relaxation. In our setting, a crucial feature is that we can implement this oracle *exactly* despite the noise added for privacy. Using the notation of [4], we let $\mathcal{P}$ be the polytope defined by the "easy" constraints: $\mathcal{P} = \{y : y_\sigma \geq 0 \text{ for all } \sigma \in S_V \text{ and } \sum_{\sigma\in S_V} y_\sigma = 1/\lambda^*\}$. Let w denote the current weights in Hedge, and p the distribution obtained by dividing each weight by the sum of the weights. Then we need to find a feasible solution for the Lagrangean relaxation, which in our case is the problem of finding a $y \in \mathcal{P}$ such that $\sum_{v\in V} w_v \sum_{\sigma\in S_V} q(\sigma)_v y_\sigma \leq \sum_{v\in V} w_v$. Equivalently, in terms of p, we need to find a $y \in \mathcal{P}$ such that $\sum_{v\in V} p_v \sum_{\sigma\in S_V} q(\sigma)_v y_\sigma \leq 1$). To do this, it is sufficient to find a y minimizing the left hand side, which (after changing the order of summations) is the same as finding a y minimizing $\sum_{\sigma\in S_V} y_\sigma \left(\sum_{v\in V} p_v q(\sigma)_v\right)$. Since by the definition of $\mathcal{P}$ we must have that $\sum_{\sigma\in S_V} y_\sigma = 1/\lambda^*$, finding such a y is the same thing as finding the σ which minimizes $\sum_{v\in V} p_v q(\sigma)_v$ and setting that y_σ to $1/\lambda^*$, and setting all others to 0. But we know from Lemma 3.1 that this σ is precisely the permutation which orders the vertices in nonincreasing order of p. So we can actually compute this ordering even without knowing the values of $q(\sigma)_v$ (as these are degrees and so are private). So our oracle (line 5 in the algorithm) computes such a σ and sets y appropriately.

3.2.2 Analysis We now analyze this algorithm, showing both privacy and utility bounds. We first show that Noisy-Order-Packing-MWU is edge-differentially-private, under the assumption that we know λ^*. As with our earlier analysis of DP-Hedge, we will use Rényi differential privacy.

Lemma 3.2. *Noisy-Order-Packing-MWU satisfies $\frac{T}{\tau^2}$-zCDP.*

Proof. This is essentially straightforward from our analysis of DP-Hedge. Consider one iteration t of Noisy-Order-Packing-MWU. Since changing an edge can affect at most one value of $q(\sigma^{(t)})$, the ℓ_2 sensitivity of the loss vector in round t is at most $\Delta = \frac{1}{\rho\lambda^*}$. Furthermore, note that $\nu = \frac{\tau}{\rho\lambda^*}$. Thus, by Theorem 2.2, Noisy-Order-Packing-MWU satisfies $\frac{\Delta^2 T}{2\nu^2} = \frac{T(\rho\lambda^*)^2}{2(\rho\lambda^*)^2\tau^2} = \frac{T}{2\tau^2}$ zCDP. ∎

We now move on to our utility bounds. We define a key quantity that will be needed in the analysis:

$$\alpha = 8\rho\sqrt{\frac{\log n}{T}} \tag{3.1}$$

Theorem 3.1. *Suppose $\alpha \leq \frac{1}{2}$. Then with probability at least $\frac{1}{2}$ over the choice of an index $t \in [T]$ chosen uniformly at random, and over the randomness in the noise, the point $(1+2\alpha)p^{(t)}$ is a feasible solution to the primal covering LP with objective value $1+2\alpha$.*

Proof. We aim to apply Theorem 2.1, so we first verify that it assumptions are met. Note that the setting $\rho = \frac{n+\tau}{\lambda^*}$

ensures that $\nu = \frac{\tau}{\rho\lambda^*} \leq 1$, and for any $v \in V$, we have $m_v^{(t)} = \frac{1}{\rho}\left(1 - \frac{q(\sigma^{(t)})_v}{\lambda^*}\right) \in [-1, 1]$ since $q(\sigma^{(t)})_v \leq n$, as required.

The objective value is equal to $1 + 2\alpha$ by construction (for every t). So let r be the probability that for a randomly chosen $t \in [T]$, the point $(1 + 2\alpha)p^{(t)}$ is a feasible solution to the primal covering LP.

For each round t, we have the following:

$$\text{(definition of } m_v^{(t)}) \qquad \langle m^{(t)}, p^{(t)} \rangle = \frac{1}{\rho} \sum_{v \in V} \left(\left(1 - \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)} \right) p_v^{(t)} \right) = \frac{1}{\rho}\left(1 - \sum_{v \in V} p_v^{(t)} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)}\right).$$

For all t, we know that $\frac{1}{\rho}\left(1 - \sum_{v \in V} p_v^{(t)} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma\right) \geq 0$, since the feasibility of the LP implies that the oracle always returns a y such that the Lagrangean relaxation is satisfied, i.e., $1 \geq \sum_{v \in V} p_v^{(t)} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)}$. So for all t, we have that $\langle m^{(t)}, p^{(t)} \rangle \geq 0$.

Now if $(1+2\alpha)p^{(t)}$ is an *infeasible* solution to the primal covering LP, then we can prove a stronger statement. In particular, there exists a permutation σ' that $\sum_{v \in V} p_v^{(t)} q(\sigma')_v < \lambda^*/(1 + 2\alpha)$. Since $\sigma^{(t)}$ is the permutation which minimizes $\sum_{v \in V} p^{(t)} q(\sigma)_v$ (by the definition of the oracle and Lemma 3.1), we have that

$$\sum_{v \in V} p_v^{(t)} q(\sigma^{(t)})_v \leq \sum_{v \in V} p_v^{(t)} q(\sigma')_v < \frac{\lambda^*}{1 + 2\alpha}.$$

Now the definition of $y^{(t)}$ (from the Oracle) implies that

$$\sum_{v \in V} p_v^{(t)} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)} = \sum_{v \in V} p_v^{(t)} q(\sigma^{(t)})_v / \lambda^* < \frac{1}{1 + 2\alpha}. \tag{3.2}$$

So we have that

$$\text{(Eq. (3.2))} \qquad \langle m^{(t)}, p^{(t)} \rangle = \frac{1}{\rho}\left(1 - \sum_{v \in V} p_v^{(t)} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)}\right) \geq \frac{1}{\rho}\left(1 - \frac{1}{1 + 2\alpha}\right) = \frac{1}{\rho} \cdot \frac{2\alpha}{1 + 2\alpha} \geq \frac{\alpha}{\rho},$$

since $\alpha \leq \frac{1}{2}$. Thus, for all t, we have:

$$\langle m^{(t)}, p^{(t)} \rangle \geq \frac{\alpha}{\rho} \cdot \mathbf{1}[(1 + 2\alpha)p^{(t)} \text{ is infeasible}].$$

Hence, recalling that r is the probability that $(1 + 2\alpha)p^{(t)}$ is feasible for a randomly chosen index t, we get

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle m^{(t)}, p^{(t)} \rangle\right] \geq \frac{\alpha}{\rho} \cdot \sum_{t=1}^{T} \Pr[(1 + 2\alpha)p^{(t)} \text{ is infeasible}] = \frac{\alpha}{\rho} \cdot (1 - r)T. \tag{3.3}$$

This gives us a bound on the LHS of the inequality in Theorem 2.1.

To bound the RHS, we proceed as follows. Let $\bar{y} = \mathbb{E}[\frac{1}{T}\sum_{t=1}^{T} y^{(t)}]$, and fix any $v \in V$. By the structure of the oracle, we know that

$$\sum_{t=1}^{T} \frac{q(\sigma^{(t)})_v}{\lambda^*} = \sum_{t=1}^{T} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)} = \sum_{\sigma \in S_V} q(\sigma)_v \cdot \left(\sum_{t=1}^{T} y_\sigma^{(t)}\right).$$

Thus we have

$$\mathbb{E}\left[\sum_{t=1}^{T} m_v^{(t)}\right] = \mathbb{E}\left[\sum_{t=1}^{T} \frac{1}{\rho}\left(1 - \frac{q(\sigma^{(t)})_v}{\lambda^*}\right)\right] = \mathbb{E}\left[\frac{1}{\rho}\left(T - \sum_{\sigma \in S_V} q(\sigma)_v \cdot \left(\sum_{t=1}^{T} y_\sigma^{(t)}\right)\right)\right] = \frac{T}{\rho}\left(1 - \sum_{\sigma \in S_V} q(\sigma)_v \bar{y}_\sigma\right). \tag{3.4}$$

Using (3.3) and (3.4) in Theorem 2.1, we get

$$\frac{\alpha}{\rho}\cdot(1-r)T \leq \frac{T}{\rho}\left(1-\sum_{\sigma\in S_V} q(\sigma)_v \bar{y}_\sigma\right)+4\sqrt{\log(n)T}.$$

Simplifying, and using the value $\alpha = 8\rho\sqrt{\frac{\log n}{T}}$, we get

$$\sum_{\sigma\in S_V} q(\sigma)_v \bar{y}_\sigma \leq 1+\frac{\alpha}{2}-\alpha\cdot(1-r).$$

Note that the above inequality holds for *all* $v \in V$. We claim that this implies that $r \geq \frac{1}{2}$. Otherwise, the RHS above is strictly less than 1. This implies that $\bar{y}$ is strictly feasible: none of the constraints are tight. Thus we can scale it up by a small amount, to get a new solution (to the dual) which is also feasible but has strictly larger objective value than $\bar{y}$. Since $\bar{y}$ has objective value $\lambda^* \cdot \langle \vec{1}, \bar{y}\rangle = \frac{\lambda^*}{T}\mathbb{E}[\langle \vec{1}, y^{(t)}\rangle] = 1$, by construction, this means that there is a feasible solution with objective value strictly larger than 1. But this is a contradiction, since the primal and dual both have objective value equal to 1. □

We will use the following result from [14].

Lemma 3.3. (Lemma 4.3 of [14]) *Given a feasible solution x to the primal LP, there exists a $\tau \in [0,1]$ such that the set $S_\tau = \{v \in V : x_v \geq \tau\}$ has density at least $\lambda^*/\sum_{v\in V} x_v$.*

We can now prove our main theorem about Noisy-Order-Packing-MWU. The proof is a relatively straightforward application of Theorem 3.1. Recall that for a permutation σ, we defined S^*_σ to be the prefix of σ with maximum density.

Theorem 3.2. *Noisy-Order-Packing-MWU returns (p,σ) such that σ is just V in nonincreasing order of p, and $\rho(S^*_\sigma) \geq (1-2\alpha)\lambda^*$ with probability at least $1/2$.*

Proof. It is easy to see from the definition of Noisy-Order-Packing-MWU that it returns (p,σ) such that σ is just V in nonincreasing order of p.

If $\alpha \geq 1/2$ then $(1-2\alpha)\lambda^* \leq 0$, and so the theorem is trivially true; any permutation σ will work. On the other hand, if $\alpha < 1/2$ then we can combine Theorem 3.1 and Lemma 3.3 to get that with probability at least 1/2, Noisy-Order-Packing-MWU returns weights p such that there is a $\tau \in [0,1]$ where the set $S_\tau = \{v \in V : (1+2\alpha)p_v \geq \tau\}$ has density at least

$$\frac{\lambda^*}{1+2\alpha} \geq \lambda^*(1-2\alpha).$$

Since σ is just non-increasing order of p, this implies that there is some prefix of σ with the same density, i.e., $\rho(S^*_\sigma) \geq (1-2\alpha)\lambda^*$ as claimed. □

3.3 Subroutine: Peeling

We now give a useful subroutine that will allow us to find the (approximate) best subset when we do peeling according to a given permutation σ, as well as an estimate of its density, in the LEDP model. Consider Algorithm 3, Peeling(σ, ς).

Algorithm 3 Peeling$(G=(V,E), \sigma, \varsigma)$

```
1: The curator sends $\sigma$ to all nodes.
2: Each node $v$ computes $\hat{q}(\sigma)_v = q(\sigma)_v + N_v$, where $N_v \sim N(0,\varsigma^2)$, and sends $\hat{q}(\sigma)_v$ to the curator.
3: For each $x \in V$, the curator computes $\widehat{\rho}(S^\sigma_x) = \frac{\sum_{v\in S^\sigma_x}\hat{q}(\sigma)_v}{|S^\sigma_x|}$.
4: Let $u = \operatorname{argmax}_{x\in V}\widehat{\rho}(S^\sigma_x)$. The curator returns $(S^\sigma_u, \widehat{\rho}(S^\sigma_u))$.
```

Lemma 3.4. *Peeling(G, σ, ς) is $\frac{1}{2\varsigma^2}$-zCDP.*

Proof. Since every edge can contribute to $q(\sigma)_v$ for exactly one v, the Gaussian mechanism (Lemma A.3) and parallel composition (Theorem A.3) imply that step 2 is $\frac{1}{2\varsigma^2}$-zCDP. Steps 3 and 4 are post-processing, hence the entire algorithm is $\frac{1}{2\varsigma^2}$-zCDP. Note that this algorithm is implementable in the local model, so is $\frac{1}{2\varsigma^2}$-zCDP. In fact, step 2 is local and the other steps are only aggregation and post-processing steps. ∎

Next, we analyze the output of Peeling:

Lemma 3.5. *Let $S \subseteq V$ be the vertices returned by Peeling(G, σ, ς). For any constant $c > 0$, with probability at least $1 - 2n^{-c}$, we have $|\widehat{\rho}(S) - \rho(S)| \leq 2\varsigma\sqrt{(1+c)\log n}$ and $\rho(S) \geq \rho(S^*_\sigma) - 2\varsigma\sqrt{(1+c)\log n}$.*

Proof. Consider some $x \in V$. Let $N = \sum_{v \in S^\sigma_x} N_v$ be the total noise added to nodes in S^σ_x. Then N is distributed as $N(0, |S^\sigma_x|\varsigma^2)$. So Lemma A.4 (the standard Gaussian concentration bound) implies that

$$\Pr[|N| \geq \sqrt{(1+c)\log n}\sqrt{|S^\sigma_x|}\varsigma] \leq 2 \cdot \exp\left(-\frac{\left(\sqrt{(1+c)\log n}\sqrt{|S^\sigma_x|}\varsigma\right)^2}{|S^\sigma_x|\varsigma^2}\right)$$
$$= 2 \cdot \exp(-(1+c)\log n) = 2n^{-(1+c)}.$$

Note that

$$\widehat{\rho}(S^\sigma_x) = \frac{\sum_{v \preceq_\sigma x} \hat{q}(\sigma)_v}{|S^\sigma_x|} = \frac{\sum_{v \preceq_\sigma x}(q(\sigma)_x + N_x)}{|S^\sigma_x|} = \frac{|E^\sigma_x| + N}{|S^\sigma_x|} = \rho(S^\sigma_x) + \frac{N}{|S^\sigma_x|}.$$

Hence we have that

$$\Pr\left[|\widehat{\rho}(S^\sigma_x) - \rho(S^\sigma_x)| \geq \frac{2\sqrt{(1+c)\log n}\varsigma}{\sqrt{|S^\sigma_x|}}\right] \leq 2n^{-(1+c)}$$

Taking a union bound over all $x \in V$ implies that with probability at least $1 - 2n^{-c}$, we have $|\widehat{\rho}(S^\sigma_x) - \rho(S^\sigma_x)| \leq 2\sqrt{(1+c)\log n}\varsigma$ for all $x \in V$. This clearly implies the lemma, since every prefix has estimated density within $2\sqrt{(1+c)\log n}\varsigma$ of its true density and the algorithm returns the prefix with the highest estimated density. ∎

3.4 The Final Algorithm

We can now give our true algorithm. We first give a "simpler" version of Noisy-Order-Packing-MWU which does not need to know λ^*, and which can easily be implemented in the local model. We call this algorithm DSG-LEDP-core (Algorithm 4), and it is essentially a "noisy" version of the Greedy++ algorithms of [8, 14]. For T iterations, we will repeatedly update loads on all of the nodes as a function of each node's (noisy) degree. These judiciously chosen updates are, in fact, simulating Noisy-Order-Packing-MWU.

Since the method only succeeds with constant probability we need to repeat the process $O(\log n)$ times. When combined with Peeling, this gives our final algorithm, DSG-LEDP (Algorithm 5). The c parameter in DSG-LEDP is used to specify success probability of at least $1 - n^{-c}$ in the utility bound (see Theorem 3.4).

We now show that there is a tight relationship between Noisy-Order-Packing-MWU and DSG-LEDP-core: they are essentially the same algorithm! Slightly more carefully, we show that if they are provided with the same random string to use for their sampling, then they will construct the exact same sequence of orderings. As a corollary, the distributions of their outputs are identical.

Theorem 3.3. *If Noisy-Order-Packing-MWU and DSG-LEDP-core are provided with the same random string to use for sampling, then for every $t \in [T]$, the ordering $\sigma^{(t)}$ computed in Noisy-Order-Packing-MWU(T, τ) and the ordering $\pi^{(t)}$ computed in DSG-LEDP-core(T, τ) are the same.*

Proof. We use induction on t. This is obviously true for $t = 1$, since for all v we have that $w^{(1)}_v = \ell^{(1)}_v = 1$ and so $p^{(1)}_v = 1/n$ and both algorithms use the same consistent tiebreaking.

Algorithm 4 DSG-LEDP-core$(G = (V, E), T, \tau)$

```
1: The curator initializes $\ell_v^{(1)} = 0$ for all $v \in V$
2: for $t = 1$ to $T$ do
3:     The curator computes the permutation $\pi^{(t)}$ of $V$ defined by ordering the nodes in non-increasing order of $\{\ell_v^{(t)}\}_{v \in V}$ (breaking ties in some consistent way, e.g., by node ID). The curator then sends $\pi^{(t)}$ to each node.
4:     Each node $v$ computes $\hat{q}(\pi^{(t)})_v = q(\pi^{(t)})_v + N_v^{(t)}$, where $N_v^{(t)} \sim N\left(0, \tau^2\right)$
5:     Each node $v$ then sends $\hat{q}(\pi^{(t)})_v$ to the curator.
6:     The curator updates all loads by setting $\ell_v^{(t+1)} = \ell_v^{(t)} + \hat{q}(\pi^{(t)})_v$
7: end for
8: The curator chooses $t$ uniformly at random from $[T]$
9: return $\pi^{(t)}$
```

Algorithm 5 DSG-LEDP$(G = (V, E), T, \varsigma, c)$

```
1: Set $\tau = \sqrt{T}\varsigma$.
2: for $i = 1$ to $c \log_2 n$ do
3:     Let $\pi^{(i)} \leftarrow$ DSG-LEDP-core$(G, T, \tau)$.
4:     Let $(S^{(i)}, \widehat{\rho}(S^{(i)})) \leftarrow$ Peeling $\left(G, \pi^{(i)}, \varsigma\right)$
5: end for
6: The curator computes $i^* = \operatorname{argmax}_{i \in [c \log n]} \widehat{\rho}(S^{(i)})$
7: return $S^{(i^*)}$
```

Now consider some $t > 1$. The loss that Noisy-Order-Packing-MWU supplies to Hedge for node v is, by construction,

$$\hat{m}_v^{(t)} \sim N\left(\frac{1}{\rho}\left(1 - \frac{q(\sigma^{(t)})_v}{\lambda^*}\right), \left(\frac{\tau}{\rho\lambda^*}\right)^2\right).$$

By standard properties of the normal distribution, we know that this distribution is identical to

$$\frac{1}{\rho}\left(1 - \frac{q(\sigma^{(t)})_v}{\lambda^*}\right) - N\left(0, \left(\frac{\tau}{\rho\lambda^*}\right)^2\right) = \frac{1}{\rho}\left(1 - \frac{q(\sigma^{(t)})_v}{\lambda^*}\right) - \frac{N(0, \tau^2)}{\rho\lambda^*}.$$

Hence we may assume without loss of generality that Noisy-Order-Packing-MWU compute $\hat{m}_v^{(t)}$ by sampling a value $J_v^{(t)} \sim N(0, \tau^2)$ and setting $\hat{m}_v^{(t)} = \frac{1}{\rho}\left(1 - \frac{q(\sigma^{(t)})_v}{\lambda^*}\right) - \frac{J_v^{(t)}}{\rho\lambda^*}$. Since $N_v^{(t)} \sim N(0, \tau^2)$ in DSG-LEDP-core, this implies that if the two algorithms are given the same random bits then $N_v^{(t)} = J_v^{(t)}$. Thus $\hat{m}_v^{(t)} = \frac{1}{\rho}\left(1 - \frac{q(\sigma^{(t)})_v + N_v^{(t)}}{\lambda^*}\right)$.

Now by definition of the weight updates in Hedge used in Noisy-Order-Packing-MWU, we know that

$$\begin{aligned}
w_v^{(t)} = \prod_{i=1}^{t-1} e^{-\eta \hat{m}_v^{(i)}} = \exp\left(-\eta \sum_{i=1}^{t} \hat{m}_v^{(i)}\right) &= \exp\left(-\eta \sum_{i=1}^{t-1} \frac{1}{\rho}\left(1 - \frac{q(\sigma^{(i)})_v + N_v^{(i)}}{\lambda^*}\right)\right) \\
&= \exp\left(-\frac{\eta}{\rho}(t-1) + \eta \sum_{i=1}^{t-1} \frac{q(\sigma^{(i)})_v + N_v^{(i)}}{\rho\lambda^*}\right) \\
\text{(induction)} \qquad &= \exp\left(-\frac{\eta}{\rho}(t-1) + \eta \sum_{i=1}^{t-1} \frac{q(\pi^{(i)})_v + N_v^{(i)}}{\rho\lambda^*}\right) \\
\text{(def of } \hat{q}(\pi^{(i)})_v\text{)} \qquad &= \exp\left(-\frac{\eta}{\rho}(t-1) + \eta \sum_{i=1}^{t-1} \frac{\hat{q}(\pi^{(i)})_v}{\rho\lambda^*}\right)
\end{aligned}$$

(def of $\ell_v^{(t)}$) $\qquad = \exp\left(-\frac{\eta}{\rho}(t-1) + \frac{\eta}{\rho\lambda^*}\ell_v^{(t)}\right)$.

Since η, ρ, and λ^* are independent of v, this means that $w_v^{(t)} < w_{v'}^{(t)}$ if and only if $\ell_v^{(t)} < \ell_{v'}^{(t)}$, and hence $p_v^{(t)} < p_{v'}^{(t)}$ if and only if $\ell_v^{(t)} < \ell_{v'}^{(t)}$. Since $\sigma^{(t)}$ is by definition the ordering of V in non-increasing order of $p_v^{(t)}$, and $\pi^{(t)}$ is the ordering of V in non-increasing order of $\ell_v^{(t)}$, and we break ties in the same consistent way in both algorithms, this implies that $\sigma^{(t)} = \pi^{(t)}$. ◻

3.5 Final Analysis

Now that we have shown that DSG-LEDP-core and Noisy-Order-Packing-MWU are essentially the same algorithm, we can finally analyze our complete algorithm DSG-LEDP.

3.5.1 Privacy We first discuss the privacy of the algorithm. This is not quite direct from the fact that Noisy-Order-Packing-MWU is private, since Theorem 3.3 only implies that the *outputs* are the same, not the computation itself, and the LEDP model requires that the full transcript be private (not just the output). But the intuition and analysis is essentially identical.

Lemma 3.6. *DSG-LEDP-core(G, T, τ) is $\frac{T}{2\tau^2}$-zCDP.*

Proof. Fix an iteration t. Computing the permutation $\pi^{(t)}$ does not require using any private information, so it is private by post-processing (Theorem A.4). For each v, the Gaussian Mechanism (Lemma A.3) implies that $\hat{q}(\pi^{(t)})$ is $1/2\tau^2$-zCDP. Since every edge contributes to $\hat{q}(\pi^{(t)}))_v$ for a single v, parallel composition (Theorem A.3) then implies that the full vector $q(\pi^{(t)})$ is $1/2\tau^2$-zCDP, and then post-processing implies that iteration t as a whole is $1/2\tau^2$-zCDP.

Finally, sequential composition (Theorem A.1) implies that all T iterations combined is $T/2\tau^2$-zCDP, and then post-processing implies that DSG-LEDP-core is $T/2\tau^2$-zCDP. ◻

Lemma 3.7. *DSG-LEDP(G, T, ς, c) is $\frac{c\log_2(n)}{\varsigma^2}$-zCDP.*

Proof. Note that DSG-LEDP runs $c\log_2 n$ copies of DSG-LEDP-core followed by a run of Peeling. Lemma 3.6 implies that each run of DSG-LEDP-core is $\frac{T}{2\tau^2}$-zCDP. Since $\tau = \sqrt{T}\varsigma$, this can be rewritten as $\frac{1}{2\varsigma^2}$-zCDP. Lemma 3.4 shows that each call to Peeling is $\frac{1}{2\varsigma^2}$-zCDP. Thus, by the sequential composition for zCDP (Theorem A.1), we get that DSG-LEDP is $\frac{c\log_2(n)}{\varsigma^2}$-zCDP, as required. ◻

Corollary 3.1. *Let $\delta \in (0,1)$ and $\epsilon \in (0, 8\log(1/\delta))$ be given privacy parameters. Set $\varsigma = \frac{4\sqrt{c\log_2(n)\log(1/\delta)}}{\epsilon}$ in DSG-LEDP. Then DSG-LEDP is (ϵ,δ)-LEDP.*

Proof. The zCDP guarantee for DSG-LEDP translates to $(\alpha, \frac{\alpha c\log_2(n)}{\varsigma^2})$-RDP guarantee for any $\alpha \geq 1$ using Lemma A.5, which in turn translates to $(\frac{\alpha c\log_2(n)}{\varsigma^2} + \frac{\log(1/\delta)}{\alpha-1}, \delta)$-LEDP for any $\alpha \geq 1$ (see Lemma A.6, i.e., Proposition 3 in [54]). Choosing $\alpha = \varsigma\sqrt{\frac{\log(1/\delta)}{c\log_2(n)}} + 1$ for ς as specified above in the lemma statement, we get that DSG-LEDP is (ϵ,δ)-DP.

Furthermore it is easy to note that DSG-LEDP-core only does local operations. In fact, in step 5 each node computes a noisy estimate of its $q(\pi^{(t)})$ and all the remaining steps are post-processing done by a central coordinator. Thus DSG-LEDP is (ϵ,δ)-LEDP. ◻

3.5.2 Utility We can now analyze the utility of our full algorithm, DSG-LEDP (Algorithm 5).

Theorem 3.4. *Suppose T and ς in DSG-LEDP are set so that $\tau = \sqrt{T}\varsigma \geq n$. Then with probability at least $1 - 3n^{-c}$, DSG-LEDP returns a set S with $\rho(S) \geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right)$.*

Proof. Since DSG-LEDP-core chooses a random $t \in [T]$ and returns the permutation from it, Theorem 3.3 and Theorem 3.2 imply that with probability at least 1/2, DSG-LEDP-core returns a permutation σ with

$$\rho(S_\sigma^*) \geq (1-2\alpha)\lambda^* \geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right),$$

where the second inequality follows since

$$\alpha = 8\rho\sqrt{\frac{\log n}{T}} = O\left(\frac{n+\tau}{\lambda^*}\sqrt{\frac{\log n}{T}}\right) = O\left(\frac{\sqrt{T}\varsigma}{\lambda^*}\sqrt{\frac{\log n}{T}}\right) = O\left(\frac{\sqrt{\log n}\varsigma}{\lambda^*}\right).$$

Since DSG-LEDP runs $c\log_2 n$ independent copies of DSG-LEDP-core, we conclude that with probability at least $1-n^{-c}$, there is at least one index $i \in [c\log_2 n]$ in which $\rho(S^*_{\pi^{(i)}}) \geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right)$. We do not know which iteration this is, but DSG-LEDP then runs Peeling (Algorithm 3) on each of these permutations. Using Lemma 3.5, we conclude that with probability at least $1-2n^{-c}$, for all $i \in [c\log_2 n]$, the set $S^{(i)}$ that we get from calling Peeling has both true and estimated density within $O\left(\sqrt{\log n}\varsigma\right)$. of $S^*_{\pi^{(i)}}$. Thus with probability at least $1-3n^{-c}$, we have

$$\begin{aligned}
\rho(S) &\geq \max_{i\in[c\log n]} \rho(S^*_{\pi^{(i)}}) - O\left(\sqrt{\log n}\varsigma\right)\\
&\geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right) - O\left(\sqrt{\log n}\varsigma\right)\\
&\geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right)
\end{aligned}$$

as claimed. ∎

Corollary 3.2. *Let $\delta \in (0,1)$ and $\epsilon \in (0, 8\log(1/\delta)$ be given privacy parameters. Set $\varsigma = \frac{4\sqrt{c\log_2(n)\log(1/\delta)}}{\epsilon}$ and $T = \lceil \frac{n^2}{\varsigma^2} \rceil$ in DSG-LEDP. Then DSG-LEDP is (ϵ,δ)-LEDP, and with probability at least $1-3n^{-c}$, DSG-LEDP returns a set S with $\rho(S) \geq \lambda^* - O\left(\frac{\log(n)\sqrt{\log(1/\delta)}}{\epsilon}\right)$.*

Proof. The privacy guarantee follows from Corollary 3.1, and the output guarantee from Theorem 3.4. ∎

3.6 Improvement in the Centralized Setting

In this section we provide improved approximation guarantees in the centralized (not local) edge differential-privacy setting via the techniques of Papernot and Steinke [58], which builds upon the work of Liu and Talwar [51]. At a very high level, Liu-Talwar and Papernot-Steinke both show that if we run many copies of the same algorithm (but with different parameters) and only output the best one, then we lose far less privacy than if we output all of them. By interpreting our random seed as a parameter, this will allow us to improve our additive loss in our main result. However, it requires the centralized model rather than LEDP since it depends on not requiring privacy for all of the parameter settings that we *do not* output, violating the LEDP requirement that the entire transcript be DP.

The setup of [58], adapted to our setting, is as follows. Suppose Q is a randomized mechanism operating on datasets D with output $Q(D)$ of the form (s,q) where s is the actual desired output (e.g., a subset of nodes in the DSG problem) and $q \in \mathbb{R}$ is a measure of its quality (e.g., the density of the output subset of nodes in the DSG problem) – higher quality is more desirable. Then, the results of Papernot and Steinke [58] imply the following result:

Theorem 3.5. *Suppose Q is ρ-zCDP. Given $\gamma \in (0,1)$, consider the algorithm $\mathcal{A}_Q$ that samples J from the standard geometric distribution with success probability γ, runs J copies of Q with independent random seeds and return the output (s,q) of Q with the highest value of q among all outputs. Then for any $\delta \in (0,1)$, the algorithm is $(6\sqrt{\rho\log(\frac{1}{\gamma\delta})},\delta)$-DP. Furthermore, the quality of the output of $\mathcal{A}_Q$ is at least q^* with probability at least $1-\frac{\gamma}{\Pr_{(s,q)\sim Q(D)}[q\geq q^*]}$.*

Proof. In the terminology of Papernot and Steinke [58], the mechanism $\mathcal{A}_Q$ is obtained by using the distribution $\mathcal{D}_{1,\gamma}$, which is the geometric distribution with success probability γ. Then using Corollary 4 of their paper, we get that for any $\lambda \geq 1+\sqrt{\frac{1}{\rho}\log(1/\gamma)}$, the mechanism $\mathcal{A}_Q$ satisfies (λ,ϵ')-RDP (see Definition A.4) where

$$\epsilon' = \rho\cdot(\lambda-1) + \tfrac{1}{\lambda-1}\log(1/\gamma) + 4\sqrt{\rho\log(1/\gamma)} - \rho.$$

For any $\delta \in (0,1)$, this translates (see Proposition 3 in [54], or Lemma A.6) to

$$\left(\rho \cdot (\lambda - 1) + \tfrac{1}{\lambda-1}\log(1/\gamma) + 4\sqrt{\rho\log(1/\gamma)} - \rho + \tfrac{1}{\lambda-1}\log(1/\delta), \delta\right)\text{-DP}.$$

Setting $\lambda = 1 + \sqrt{\frac{1}{\rho}\log(\frac{1}{\gamma\delta})}$, this simplifies to

$$\left(2\sqrt{\rho\log(\tfrac{1}{\gamma\delta})} + 4\sqrt{\rho\log(1/\gamma)} - \rho, \delta\right)\text{-DP}.$$

The privacy guarantee stated in the theorem statement is a weaker, but simpler, form of the above guarantee.

The utility guarantee follows directly from Theorem 3.3 of [51]. ∎

Clearly any mechanism in the LEDP model can also be run in the centralized model, so consider the following centralized mechanism which first runs DSG-LEDP-core (centralized) and then Peeling:

Algorithm 6 Centralized-DSG-core(G, T, ς)

```
1: Set $\tau = \sqrt{T}\varsigma$.
2: Run DSG-LEDP-core$(G, T, \tau)$ and obtain a permutation $\sigma$.
3: Output $(S, \tilde{\rho}(S)) \leftarrow$ Peeling$(G, \sigma, \varsigma)$.
```

LEMMA 3.8. *Centralized-DSG-core(G, T, ς) is $\frac{1}{\varsigma^2}$-zCDP.*

Proof. Since DSG-LEDP-core is equivalent to Noisy-Order-Packing-MWU, Lemma 3.6 implies that DSG-LEDP-core is $\frac{T}{2\tau^2}$-zCDP. Since $\tau = \sqrt{T}\varsigma$, this can be rewritten as $\frac{1}{2\varsigma^2}$-zCDP. Lemma 3.4 shows that the call to Peeling is $\frac{1}{2\varsigma^2}$-zCDP. Thus, by the composition results for zCDP [10], we get that DSG-LEDP is $\frac{1}{\varsigma^2}$-zCDP, as required. ∎

LEMMA 3.9. *Suppose T and ς in Centralized-DSG-core are set so that $\tau = \sqrt{T}\varsigma \geq n$. Then with probability at least $1/4$, Centralized-DSG-core(G, T, ς) returns a set S with $\tilde{\rho}(S) \geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right)$.*

Proof. Exactly as argued in Theorem 3.4, with probability at least $1/2$, DSG-LEDP-core returns a permutation σ with

$$\rho(S^*_\sigma) \geq (1 - 2\alpha)\lambda^* \geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right).$$

Using Lemma 3.5 with $c = \frac{2}{\log n}$, we conclude that with probability at least $1 - 2e^{-2} \geq 1/2$, the set S that we get from calling Peeling has both true and estimated density within $O\left(\sqrt{\log n}\varsigma\right)$ of S^*_σ. Thus with probability at least $1/4$, we have

$$\begin{aligned}\tilde{\rho}(S) &\geq \rho(S^*_\sigma) - O\left(\sqrt{\log n}\varsigma\right)\\ &\geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right) - O\left(\sqrt{\log n}\varsigma\right)\\ &\geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right)\end{aligned}$$

as claimed. ∎

Now, let $c > 0$ be a given failure probability parameter, and define $\gamma = n^{-c}$ for notational convenience. Let Centralized-DSG be the mechanism $\mathcal{A}_Q$ obtained by applying the mechanism of Theorem 3.5 with $\gamma = n^{-c}$ as specified above, $Q =$ Centralized-DSG-core, $s = S$, the set it outputs, and $q = \tilde{\rho}(S)$, the estimated density it outputs, and q^* set to $\lambda^* - O\left(\sqrt{\log n}\varsigma\right)$ as specified in Lemma 3.9.

THEOREM 3.6. *Let $\delta \in (0,1)$ and $\epsilon > 0$ be given privacy parameters. Set $\varsigma = \frac{6\sqrt{\log(n^c/\delta)}}{\epsilon}$ and $T = \lceil\frac{n^2}{\varsigma^2}\rceil$ in Centralized-DSG-core. Then Centralized-DSG is (ϵ, δ)-DP, and with probability at least $1 - 6n^{-c}$, the density of the set it outputs is at least $\lambda^* - O\left(\frac{\sqrt{\log(n)\log(n/\delta)}}{\epsilon}\right)$.*

Proof. With the notation as defined in the paragraph right before this theorem, Lemma 3.9 implies that

$$\Pr_{(s,q)\sim Q(D)}[q \geq q^*] \geq \frac{1}{4}.$$

Since Centralized-DSG-core(T, ς) is $\frac{1}{\varsigma^2}$-zCDP, applying Theorem 3.5, we conclude that Centralized-DSG is

$$\left(\frac{6\sqrt{\log(1/(\gamma\delta))}}{\varsigma}, \delta\right)\text{-DP} = (\epsilon, \delta)\text{-DP}$$

for the specified value of ς, and the estimated density $\tilde{\rho}(S)$ of the set S it outputs is at least q^* with probability at least $1-4\gamma$. Now, recall the random variable J used in Centralized-DSG which is drawn from the geometric distribution with success probability γ. Since $\Pr[J > k] = (1-\gamma)^k \leq \exp(-\gamma k)$, we conclude that $\Pr[J \leq \frac{\log(1/\gamma)}{\gamma}] \geq 1-\gamma$. Conditioned on $J \leq \frac{\log(1/\gamma)}{\gamma}$, using Lemma 3.5 and a union bound over the J calls to Q in $\mathcal{A}_Q$, we conclude that with probability at least $1-\gamma$, in each call to Q, we have

$$|\rho(S) - \tilde{\rho}(S)| \leq O\left(\sqrt{\log n}\varsigma\right)$$

for an appropriately chosen constant in the $O(\cdot)$ notation. Thus, overall, using the union bound, with probability at least $1 - 6\gamma = 1 - 6n^{-c}$, the true density of the set output by Centralized-DSG is at least

$$\lambda^* - O\left(\sqrt{\log n}\varsigma\right) = \lambda^* - O\left(\frac{\sqrt{\log(n)\log(n/\delta)}}{\epsilon}\right)$$

as claimed. □

4 Node-Weighted Densest Subgraph

Recall that in the node-weighted setting the edges are unweighted, but vertices can have weights. To fix notation, we will say that every node v has cost $c_v \geq 1$ and $C_{\max} = \max_{v\in V} c_v$. We note that the assumption that every $c_v \geq 1$ is not without loss of generality: rescaling weights to all be at least 1 will incur a cost due to the fact that we have an additive part to our approximation (if our approximation were purely multiplicative then this would be WLOG). However, this is a relatively standard assumption; see, e.g., [63]. We let $c(S) = \sum_{v\in S} c_v$ for any $S \subseteq V$, and define the density as $\rho(S) = |E(S)|/c(S)$. Our goal is to find the densest subgraph.

This is in some sense a relatively minor change: as we will show, it is not too hard to adapt our version of noisy PST, Noisy-Order-Packing-MWU (Algorithm 2), to the node-weighted setting. However, recall that in the unweighted setting, we didn't actually *run* Noisy-Order-Packing-MWU. We just showed that every iteration of it was identical to an iteration of our *real* algorithm, DSG-LEDP-core (Algorithm 4). The presence of node weights, unfortunately, destroys this approach: we cannot simply keep track of "loads" (i.e., degrees) and build a combinatorial algorithm using them which simulates the Hedge algorithm. Intuitively, this is because in the proof of Theorem 3.3, the expression for $w_v^{(t)}$ has in the exponent a term that is independent of v and a term that depends on the degrees. But in the presence of node weights, it turns out that the first term actually becomes a function of the node weight. So to give the same ordering, we would have to trade these off in the same way as in the Hedge algorithm. But that tradeoff depends on λ^*, which we do not actually know. So we cannot simulate the Hedge algorithm via a combinatorial, degree-based algorithm.

Instead, we will directly run a node-weighted version of Noisy-Order-Packing-MWU. Unfortunately, doing so requires the curator to know λ^*. If we were in the centralized edge-DP model we could of course just compute λ^* and add noise to preserve privacy, but in the local model this is significantly more difficult. Instead, we "guess" λ^* and run our algorithm assuming our guess is correct. Then, by running a grid search for exponentially increasing guesses of λ^*, we are able to assume that we have an approximation of λ^*. Compared to the unweighted case, this incurs a small multiplicative loss and an additional logarithmic additive loss.

4.1 Peeling

The Peeling algorithm can be easily modified to work for the node-weighted setting:

Algorithm 7 Weighted-Peeling(G, σ, ς)

```
1: The curator sends $\sigma$ to all nodes.
2: Each node $v$ computes $\hat{q}(\sigma)_v = q(\sigma)_v + N_v$, where $N_v \sim N(0, \varsigma^2)$, and sends $\hat{q}(\sigma)_v$ to the curator.
3: For each $x \in V$, the curator computes $\widehat{\rho}(S_x^\sigma) = \frac{\sum_{v \in S_x^\sigma} \hat{q}(\sigma)_v}{c(S_x^\sigma)}$.
4: Let $u = \text{argmax}_{x \in V}\, \widehat{\rho}(S_x^\sigma)$. The curator returns $(S_u^\sigma, \widehat{\rho}(S_u^\sigma))$.
```

Lemmas 3.4 and 3.5 hold without change for Weighted-Peeling: the proof of the former is unchanged, and the proof of the latter follows using the fact that $c(S_x^\sigma) \geq |S_x^\sigma|$ since all node capacities c_v are at least 1. For concreteness, we provide the following lemma statements but omit proofs:

LEMMA 4.1. *Weighted-Peeling(σ, ς) is $\frac{1}{2\varsigma^2}$-zCDP for any $\alpha \geq 1$.*

LEMMA 4.2. *Let $S \subseteq V$ be the vertices returned by Weighted-Peeling(σ, ς). For any constant $c > 0$, with probability at least $1 - 2n^{-c}$, we have $|\widehat{\rho}(S) - \rho(S)| \leq 2\sqrt{(1+c)\log n}\varsigma$ and $\rho(S) \geq \rho(S_\sigma^*) - 2\sqrt{(1+c)\log n}\varsigma$.*

4.2 Weighted versions of LPs from [14]

In the weighted context, the equivalent of Charikar's LP [13] is the following:

$$\begin{aligned} \max \ & \sum_{e \in E} y_e \\ \text{s.t.} \ & \sum_{v \in V} c_v x_v \leq 1 \\ & y_{u,v} \leq x_u && \forall \{u, v\} \in E \\ & y_{u,v} \leq x_v && \forall \{u, v\} \in E \end{aligned}$$

Note that since we are assuming that $c_v \geq 1$ for all $e \in E$, the first constraint implies that $x_v \leq 1$ for all $v \in V$. It is not hard to see that this precisely characterizes the node-weighted DSG problem; we prove this for completeness.

THEOREM 4.1. *This LP is an exact formulation: for every set $S \subseteq V$ there is a feasible LP solution with objective at least $\rho(S)$, and for every feasible solution (x, y) there is a set S with $\rho(S) \geq \sum_{e \in E} y_e$.*

Proof. We begin with the first part. Let $S \subseteq V$. Set $x_v = \frac{1}{c(S)}$ for each $v \in S$, and $x_v = 0$ otherwise. Set $y_{u,v} = \min(x_u, x_v)$ for each $\{u, v\} \in E$. Then $\sum_{v \in V} c_v x_v = \sum_{v \in S} c_v / c(S) = 1$, so the solution is feasible, and $\sum_{e \in E} y_e = \sum_{e \in E(S)} 1/c(S) = |E(S)|/c(S) = \rho(S)$ as claimed.

For the other direction, let (x, y) be a feasible solution. For every $\tau \in [0, 1]$, let $S_\tau = \{v \in V : x_v \geq \tau\}$. We claim that there exists some τ such that $\rho(S_\tau) = |E(S_\tau)|/c(S_\tau) \geq \sum_{e \in E} y_e$, which is enough to prove the theorem. To see this, suppose for contradiction that it is false. Then $\rho(S_\tau) < \sum_{e \in E} y_e$ for all τ. Consider choosing τ uniformly at random from $[0, 1]$. Then we have that

$$\begin{aligned} &\text{(assumption for contradiction)} && \mathbb{E}[|E(S_\tau)|] < \mathbb{E}\left[c(S_\tau) \sum_{e \in E} y_e\right] = \left(\sum_{e \in E} y_e\right) \mathbb{E}[c(S_\tau)] \\ &\text{(linearity of expectations)} && = \left(\sum_{e \in E} y_e\right) \sum_{v \in V} c_v x_v \\ &\text{(feasible)} && \leq \sum_{e \in E} y_e \leq \sum_{\{u,v\} \in E} \min(x_u, x_v) \\ & && = \sum_{\{u,v\} \in E} \Pr[\{u, v\} \in E(S_\tau)] \\ &\text{(linearity of expectations)} && = \mathbb{E}[|E(S_\tau)|] \end{aligned}$$

Since there is a strict inequality, this is a contradiction. Hence some such τ exists (and we can find it efficiently since there are at most n distinct values for τ). □

To put this into the language used by [14], we can define $\hat{f}(x) = \mathbb{E}[|E(S_\tau)|]$. Then the proof of the previous theorem implies that Charikar's LP is identical to the following:

$$\begin{aligned} \max\ & \hat{f}(x) \\ \text{s.t.}\ & \sum_{v \in V} c_v x_v \le 1 \\ & x_v \ge 0 && \forall v \in V \end{aligned}$$

Since $\hat{f}$ is independent of weights, Lemma 3.1 is still true. Hence we have yet another equivalent formulation:

$$\begin{aligned} \max\ & \min_{\sigma \in S_V} \langle x, q(\sigma) \rangle \\ \text{s.t.}\ & \sum_{v \in V} c_v x_v \le 1 \\ & x_v \ge 0 && \forall v \in V \end{aligned}$$

Let λ^* be the value of the optimal solution. Then we can switch the objective and the constraint to get the equivalent of the LP from [14], i.e., the equivalent of the covering LP discussed in Section 3.1:

$$\begin{aligned} \min\ & \sum_{v \in V} c_v x_v \\ \text{s.t.}\ & \langle x, q(\sigma) \rangle \ge \lambda^* && \forall \sigma \in S_V \\ & x_v \ge 0 && \forall v \in V \end{aligned}$$

By the above discussion and the definition of λ^*, we know that the optimal value of this LP is equal to 1. When we take its dual, we get the following packing LP, which by strong duality also has optimal solution equal to 1.

$$\begin{aligned} \max\ & \lambda^* \sum_{\sigma \in S_V} y_\sigma \\ \text{s.t.}\ & \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma \le c_v && \forall v \in V \\ & y_\sigma \ge 0 && \forall \sigma \in S_V \end{aligned}$$

This is the main LP which we will be arguing about. Note that it's the exact same as in the unweighted case, except for the right hand side being the costs rather than just 1.

4.3 Noisy PST in the Presence of Weights

Now let's apply the PST framework [59], as described by [4], to the above LP in the presence of noise. This is quite similar to Section 3.2.1; we just have to be careful how the costs on the right hand side affect the definitions and the algorithm. The big difference here is that unlike in the unweighted case we were able to get away without knowing the exact value of λ^* in the algorithm due to the special structure of the problem, here we do not have a way to use the same strategy and we must resort to a search to guess the value of λ^*. To simplify the discussion, in the rest of this section, we discuss the analysis where we have a particular guess λ for λ^* and the consequences of it being too high or too low. Then in the next section we discuss the grid search algorithm to get the correct λ.

Feasibility LPs. We use the following pair of feasibility LPs to decide whether the guess λ is too high or too low. The primal LP is the following:

$$\sum_{v \in V} c_v x_v = 1$$

$$\begin{aligned} &\text{s.t. } \langle x, q(\sigma)\rangle \geq \lambda && \forall \sigma \in S_V \\ (\text{PLP}(\lambda)) \qquad & x_v \geq 0 && \forall v \in V \end{aligned}$$

It is not hard to see based on the discussion in Section 4.2 that (PLP(λ)) is feasible if and only if $\lambda \leq \lambda^*$.

To produce feasible solutions for the primal LP, we will actually use the following dual LP:

$$\begin{aligned} &\lambda \sum_{\sigma \in S_V} y_\sigma = 1 && \\ &\frac{1}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma \leq 1 && \forall v \in V \\ (\text{DLP}(\lambda)) \qquad & y_\sigma \geq 0 && \forall \sigma \in S_V \end{aligned}$$

It is easy to see that (DLP(λ)) is feasible if and only if $\lambda \geq \lambda^*$. Note that this is the exact opposite criterion as for the feasibility of (PLP(λ)).

Oracle. To check for feasibility for (DLP(λ)), we proceed as in the unweighted case. Define $\mathcal{P} = \{y : y_\sigma \geq 0 \text{ for all } \sigma \in S_V \text{ and } \sum_{\sigma \in S_V} y_\sigma = 1/\lambda\}$. In each iteration of Hedge with noisy losses, if p is the current distribution on V, then we need to find a feasible solution for the Lagrangean relaxation i.e., we need to find a $y \in \mathcal{P}$ such that $\sum_{v \in V} \frac{p_v}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma \leq 1$. So it is sufficient to find a y minimizing the left hand side. Exactly as in unweighted case, this y can be found by choosing the permutation σ that orders nodes in nonincreasing order of $\langle \frac{p_v}{c_v} \rangle_v$, and setting $y_\sigma = \frac{1}{\lambda}$, and $y'_\sigma = 0$ for all permutations $\sigma' \neq \sigma$. This defines our Oracle.

Algorithm. Plugging these costs into the Plotkin-Shmoys-Tardos framework leads to a version of Noisy-Order-Packing-MWU for weighted graphs; see Algorithm 8. Note that this algorithm can be directly implemented in the LEDP model just as in the unweighted setting.

Algorithm 8 Weighted-Noisy-Order-Packing-MWU(G, λ, T, τ)

```
1: Set $\rho = \frac{n+\tau}{\lambda}$ and $\nu = \frac{\tau}{\rho\lambda}$.
2: Instantiate Hedge($T, \nu$) with $n$ experts corresponding to nodes in $V$.
3: for $t = 1$ to $T$ do
4:     Obtain the distribution $p^{(t)}$ on $V$ from Hedge.
5:     Use the Oracle, as discussed: let $\sigma^{(t)}$ be the ordering of $V$ in nonincreasing order of $\langle \frac{p_v^{(t)}}{c_v} \rangle_v$ (breaking ties
       consistently, e.g., by node IDs)
6:     For each $v \in V$, set the costs:
           $$\hat{m}_v^{(t)} \sim N(m_v^{(t)}, \nu^2), \text{ where } m_v^{(t)} = \frac{1}{\rho}\left(1 - \frac{1}{c_v}\sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)}\right) = \frac{1}{\rho}\left(1 - \frac{q(\sigma^{(t)})_v}{c_v \lambda}\right).$$
7:     Supply $\hat{m}^{(t)}$ as the loss vector to Hedge.
8: end for
9: Let $t$ be chosen uniformly at random from $[T]$.
10: Define $x_v^{(t)} := \frac{p_v^{(t)}}{c_v}$ for all $v \in V$.
11: return $(x^{(t)}, \sigma^{(t)})$.
```

Analysis. We're going to follow the analysis for the unweighted case from Section 3.5 and prove the following theorem:

THEOREM 4.2. *Let $L \geq 4(n+\tau)\sqrt{\frac{\log n}{T}}$. Suppose $\lambda^* \leq \lambda < \lambda^* + L$. Then with probability at least $\frac{1}{2}$ over the choice of an index $t \in [T]$ chosen uniformly at random, and over the randomness in the noise, the output point x of Weighted-Noisy-Order-Packing-MWU(G, λ, T, τ) is a feasible solution to PLP($\lambda - 4L$).*

Proof. First, note that if $\lambda < 4L$, then the statement is true with probability 1 since any solution x returned by Weighted-Noisy-Order-Packing-MWU is feasible. So in the following, we assume that $\lambda > 4L$.

Since $\lambda^* \le \lambda < \lambda^* + L$, (DLP($\lambda$)) is feasible. Now we follow the analysis in the proof of Theorem 3.1. Most calculations are identical, so we skip most details. Let r be the probability that the output $x^{(t)}$ is infeasible for PLP$(\lambda - 4L)$. Now, for any round t, we have the following:

$$\begin{aligned}\langle m^{(t)}, p^{(t)} \rangle &= \sum_{v \in V} m_v^{(t)} p_v^{(t)} = \frac{1}{\rho} \sum_{v \in V} \left(\left(1 - \frac{1}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)} \right) p_v^{(t)} \right) \\ &= \frac{1}{\rho} \sum_{v \in V} \left(1 - \frac{p_v^{(t)}}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)} \right) \\ &= \frac{1}{\rho} \left(1 - \sum_{v \in V} \frac{p_v^{(t)}}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma^{(t)} \right).\end{aligned}$$

Since (DLP(λ)) is feasible, the oracle (by definition) always returns a y such that

$$\sum_{v \in V} \frac{p_v^{(t)}}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v y_\sigma \le 1.$$

Hence, we conclude that $\langle m^{(t)}, p^{(t)} \rangle \ge 0$ for all t.

Next, suppose t is a round where $x^{(t)}$ is infeasible for PLP$(\lambda - 4L)$. Note that by definition, $\sum_v x_v^{(t)} c_v = \sum_v p_v^{(t)} = 1$. Thus, the infeasibility must arise because there exists a $\sigma' \in S_V$ such that $\langle x^{(t)}, q(\sigma') \rangle < \lambda - 4L$, i.e., $\sum_v \frac{p_v^{(t)}}{c_v} q(\sigma'')_v < \lambda - 4L$. Since $\sigma^{(t)}$ is the permutation that minimizes $\sum_v \frac{p_v^{(t)}}{c_v} q(\sigma'')_v$ over all $\sigma'' \in S_V$, we must have $\sum_v \frac{p_v^{(t)}}{c_v} q(\sigma^{(t)})_v < (\lambda - 4L)$, which is equivalent to

$$\sum_{v \in V} \frac{p_v^{(t)}}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v y_{\sigma^{(t)}} < 1 - \frac{4L}{\lambda}.$$

This implies that

$$\langle m^{(t)}, p^{(t)} \rangle = \frac{1}{\rho} \left(1 - \sum_{v \in V} \frac{p_v^{(t)}}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v y_{\sigma^{(t)}} \right) \ge \frac{4L}{\lambda \rho}.$$

Thus, for all t, we have, exactly as in the proof of Theorem 3.1 that

$$\langle m^{(t)}, p^{(t)} \rangle \ge \frac{4L}{\lambda \rho} \cdot \mathbf{1}[x^{(t)} \text{ is infeasible}],$$

and hence,

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle m^{(t)}, p^{(t)} \rangle \right] \ge \frac{4L}{\lambda \rho} \cdot \sum_{t=1}^{T} \Pr[x^{(t)} \text{ is infeasible}] = \frac{4L}{\lambda \rho} \cdot (1 - r) T. \tag{4.5}$$

Then following the proof of Theorem 3.1 from that point, we end up with the following version of (3.4):

$$\mathbb{E}\left[\sum_{t=1}^{T} m_v^{(t)} \right] = \mathbb{E}\left[\sum_{t=1}^{T} \frac{1}{\rho} \left(1 - \frac{q(\sigma^{(t)})_v}{c_v \lambda} \right) \right] = \mathbb{E}\left[\frac{T}{\rho} \left(1 - \frac{1}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v \bar{y}_\sigma \right) \right] = \frac{T}{\rho} \left(1 - \frac{1}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v \bar{y}_\sigma \right). \tag{4.6}$$

Continuing with the analysis, we get the following bound via Theorem 2.1 for $\bar{y} = \mathbb{E}[\frac{1}{T} \sum_{t=1}^{T} y^{(t)}]$:

$$\frac{4L}{\lambda \rho} \cdot (1 - r) T \le \frac{T}{\rho} \left(1 - \frac{1}{c_v} \sum_{\sigma \in S_V} q(\sigma)_v \bar{y}_\sigma \right) + 4\sqrt{\log(n) T}.$$

Simplifying, we get

$$\frac{1}{c_v}\sum_{\sigma\in S_V} q(\sigma)_v \bar{y}_\sigma \le 1 + 4\rho\sqrt{\frac{\log n}{T}} - \frac{4L}{\lambda}\cdot(1-r) \le 1 + \frac{L}{\lambda} - \frac{4L}{\lambda}\cdot(1-r),$$

where the last inequality follows from the assumption that $L \ge 4(n+\tau)\sqrt{\frac{\log n}{T}}$ and the setting $\rho = \frac{n+\tau}{\lambda}$ Note that the inequality displayed above holds for *all* $v \in V$. We claim that this implies that $r \ge \frac{1}{2}$. Otherwise, the RHS above is less than $1 - \frac{L}{\lambda}$. Now consider $\tilde{y} = \frac{\lambda}{\lambda - L}\bar{y}$ and define $\lambda' = \lambda - L < \lambda^*$. Note that DLP($\lambda'$) is infeasible. But $\tilde{y}$ satisfies all the inequality constraints of DLP(λ'), and furthermore we have

$$\lambda' \sum_{\sigma\in S_V} \tilde{y}_\sigma = (\lambda - L)\cdot\frac{\lambda}{\lambda - L}\sum_{\sigma\in S_V}\bar{y}_\sigma = 1,$$

since $\lambda\sum_{\sigma\in S_v}\bar{y}_\sigma = 1$ by construction. But this is a contradiction to the infeasibility of DLP(λ'), and hence we conclude that $r \ge \frac{1}{2}$ as claimed. ∎

4.4 Full Algorithm and Analysis

We now can give our full algorithm, Weighted-DSG-LEDP (Algorithm 9), which essentially wraps Weighted-Noisy-Order-Packing-MWU in a grid search for λ^*, and repeats each guess a number of times in order to amplify the probability of getting a good solution from $\frac{1}{2}$ to $1 - n^{-c}$. The grid points in the search for λ^* are arranged in a geometric progression with factor $1+\beta$ covering the entire possible range for λ^*. The c parameter in Weighted-DSG-LEDP is used to specify success probability of at least $1 - 3n^{-c}$ in the utility bound (see Theorem 4.3).

Algorithm 9 Weighted-DSG-LEDP$(G, T, \varsigma, c, \beta)$

```
1: Let $\lambda_0 = 1/(2C_{\max})$, $K = c\log_2(n)\cdot(\log_{1+\beta}(2C_{\max}n)+1)$, and $\tau = \sqrt{T}\varsigma$
2: for $i = 1$ to $\log_{1+\beta}(2C_{\max}n)+1$ do
3:     Let $\lambda_i = (1+\beta)^{i-1}\lambda_0$
4:     for $j = 1$ to $c\log_2 n$ do
5:         Let $(\pi^{(i,j)}, x^{(i,j)}) \leftarrow$ Weighted-Noisy-Order-Packing-MWU$(G, \lambda_i, T, \tau)$.
6:         Let $(S^{(i,j)}, \widehat{\rho}(S^{(i,j)})) \leftarrow$ Weighted-Peeling$\big(G, \pi^{(i,j)}, \varsigma\big)$
7:     end for
8: end for
9: The curator computes $(i^*, j^*) = \mathrm{argmax}_{i,j}\,\widehat{\rho}(S^{(i,j)})$ and returns $S^{(i^*,j^*)}$.
```

4.4.1 Privacy The following lemma has the exact same proof as Lemma 3.2:

LEMMA 4.3. *Weighted-Noisy-Order-Packing-MWU is* $\frac{T}{\tau^2}$*-zCDP.*

LEMMA 4.4. *Weighted-DSG-LEDP$(G, T, \varsigma, c, \beta)$ is* $\frac{K}{\varsigma^2}$*-zCDP.*

Proof. Note that Weighted-DSG-LEDP runs K copies of Weighted-Noisy-Order-Packing followed by a run of Weighted-Peeling. By Lemma 3.6 each run of Weighted-Noisy-Order-Packing is $\frac{T}{2\tau^2}$-zCDP. Since $\tau = \sqrt{T}\varsigma$, this can be rewritten as $\frac{1}{2\varsigma^2}$-zCDP. Lemma 3.4 shows that each call to Weighted-Peeling is $\frac{1}{2\varsigma^2}$-zCDP. Thus, by the composition results for zCDP [10], we get that Weighted-DSG-LEDP is $\frac{K}{\varsigma^2}$-zCDP, as required. ∎

COROLLARY 4.1. *Let $\delta \in (0,1)$ and $\epsilon \in (0, 8\log(1/\delta))$ be given privacy parameters. Set $\varsigma = \frac{4\sqrt{K\log(1/\delta)}}{\epsilon}$ in Weighted-DSG-LEDP. Then Weighted-DSG-LEDP is (ϵ,δ)-LEDP.*

Proof. The zCDP guarantee for Weighted-DSG-LEDP translates to $(\alpha, \frac{\alpha K}{\varsigma^2})$-RDP guarantee for any $\alpha \ge 1$ (Lemma A.5), which in turn translates to $(\frac{\alpha K}{\varsigma^2} + \frac{\log(1/\delta)}{\alpha-1}, \delta)$-LEDP for any $\alpha \ge 1$ (see Proposition 3 in [54] or Lemma A.6). Choosing $\alpha = \varsigma\sqrt{\frac{\log(1/\delta)}{K}} + 1$ for ς as specified above in the corollary statement, we get that Weighted-DSG-LEDP is (ϵ, δ)-DP.

Furthermore it is easy to note that Weighted-DSG-LEDP only does local operations. Thus Weighted-DSG-LEDP is (ϵ, δ)-LEDP. ∎

4.4.2 Utility In order to prove the utility of Weighted-DSG-LEDP, we first need to prove the weighted equivalent of Lemma 3.3: we need to show that a good solution to (PLP(λ)) not only implies a good subgraph, but that it has a particular structure.

LEMMA 4.5. *Given a feasible solution x to (PLP(λ)), there exists a $\tau \in [0,1]$ such that the set $S_\tau = \{v \in V : x_v \geq \tau\}$ has density at least λ.*

Proof. For each $\tau \in [0,1]$, let $E_\tau = \{\{u,v\} \in E : \min(x_u, x_v) \geq \tau\}$ be the edges induced by S_τ. Suppose for contradiction that the lemma is false: $\rho(S_\tau) = |E_\tau|/c(S_\tau) < \lambda$ for every $\tau \in [0,1]$. Consider picking τ uniformly at random from $[0,1]$. Since x is feasible for (PLP(λ)), we know that $\langle x, q(\sigma)\rangle \geq \lambda$ for all $\sigma \in S_V$. By Lemma 3.1 (which still applies since it is independent of the weights), this means that $\sum_{\{u,v\}\in E} \min(x_u, x_v) \geq \lambda$. Since $\Pr[\{u,v\} \in E_\tau] = \min(x_u, x_v)$, we know from linearity of expectations that $\mathbb{E}[|E_\tau|] \geq \lambda$.

Now we have that

$$\begin{aligned}
&\text{(assumption for contradiction)} && \lambda \leq \mathbb{E}[|E_\tau|] = \mathbb{E}\left[c(S_\tau)\rho(S_\tau)\right] < \mathbb{E}\left[c(S_\tau)\lambda\right] \\
&\text{(linearity of expectations)} && = \lambda\,\mathbb{E}[c(S_\tau)] = \lambda \sum_{v\in V} c_v \Pr[v \in S_\tau] \\
&\left(\textstyle\sum_{v\in V} c_v x_v = 1 \text{ since } x \text{ is feasible for (PLP}(\lambda))\right) && = \lambda \sum_{v\in V} c_v x_v = \lambda
\end{aligned}$$

This is a contradiction, since obviously it is not true that $\lambda < \lambda$. Thus the lemma is true. ∎

We can now analyze the utility of Weighted-DSG-LEDP (Algorithm 9).

THEOREM 4.3. *Suppose T and ς in Weighted-DSG-LEDP are set so that $\tau = \sqrt{T}\varsigma \geq n$. Then with probability at least $1 - (2K+1)n^{-c}$, Weighted-DSG-LEDP returns a set S with $\rho(S) \geq (1-4\beta)\lambda^* - O\left(\frac{1}{\beta}\sqrt{\log n}\varsigma\right)$.*

Proof. We may assume without loss of generality that $\lambda^* \geq \frac{8\sqrt{\log n}\varsigma}{\beta}$; otherwise, the guarantee in the theorem holds trivially. The geometric grid search for λ^* ensures that there is at least one grid point, λ_i, which satisfies $\lambda^* \leq \lambda_i < (1+\beta)\lambda^*$. Note that $\beta\lambda^* \geq 8\sqrt{\log n}\varsigma \geq 4(n+\tau)\sqrt{\frac{\log n}{T}}$, since $\sqrt{T}\varsigma \geq n$. Thus, we can apply Theorem 4.2 with $L = \beta\lambda^*$ and $\lambda = \lambda_i$ to conclude that with probability at least 1/2, Weighted-Noisy-Order-Packing-MWU(G, λ_i, T, τ) outputs a feasible solution x to PLP$(\lambda_i - 4\beta\lambda^*)$, and its corresponding permutation σ. Recall that S^*_σ is the prefix of σ with largest density. Since σ is just nondecreasing order of x, Lemma 4.5 implies that

$$\rho(S^*_\sigma) \geq \lambda_i - 4\beta\lambda^* \geq (1-4\beta)\lambda^*,$$

since $\lambda_i \geq \lambda^*$. Since Weighted-DSG-LEDP runs $c\log_2 n$ independent copies of Weighted-Noisy-Order-Packing-MWU(G, λ_i, T, τ), we conclude that with probability at least $1 - n^{-c}$, there is at least one index $j \in [c\log_2 n]$ in which $\rho(S^*_{\pi^{(i,j)}}) \geq (1-4\beta)\lambda^*$.

We do not know which specific pair of indices (i,j) this is, but Weighted-DSG-LEDP then runs Weighted-Peeling (Algorithm 7) on each of the permutations obtained from Weighted-Noisy-Order-Packing-MWU. Using Lemma 4.2, we conclude that via a union bound that with probability at least $1-2Kn^{-c}$, for all $i \in [c\log_2 n]$ and $j \in [\log_{1+\beta}(2C_{\max}n)+1]$, the set $S^{(i,j)}$ that we get from calling Weighted-Peeling has both true and estimated density within $O\left(\sqrt{\log n}\varsigma\right)$ of $S^*_{\pi^{(i,j)}}$. Thus, with probability at least $1-(2K+1)n^{-c}$, we have that

$$\begin{aligned}
\rho(S) &\geq \max_{i\in[c\log n], j\in[\log_{1+\beta}(2C_{\max}n)+1]} \rho(S^*_{\pi^{(i,j)}}) - O\left(\sqrt{\log n}\varsigma\right) \\
&\geq (1-4\beta)\lambda^* - O\left(\tfrac{1}{\beta}\sqrt{\log n}\varsigma\right),
\end{aligned}$$

as required. ∎

COROLLARY 4.2. *Let $\delta \in (0,1)$ and $\epsilon \in (0, 8\log(1/\delta)$ be given privacy parameters. Set $\varsigma = \frac{4\sqrt{K\log(1/\delta)}}{\epsilon}$ and $T = \lceil \frac{n^2}{\varsigma^2} \rceil$ in Weighted-DSG-LEDP. Then Weighted-DSG-LEDP is (ϵ,δ)-LEDP, and with probability at least $1-(2K+1)n^{-c}$, DSG-LEDP returns a set S with $\rho(S) \geq (1-4\beta)\lambda^* - O\left(\frac{\log(n)\sqrt{\log(C_{\max}n)\log(1/\delta)}}{\beta^{1.5}\epsilon}\right)$.*

Proof. The privacy guarantee follows from Corollary 4.1, and the output guarantee from Theorem 4.3, using the bound $\varsigma = \frac{4\sqrt{K\log(1/\delta)}}{\epsilon} = O\left(\frac{\sqrt{\log(n)\log(C_{\max}n)\log(1/\delta)}}{\sqrt{\beta}\epsilon}\right)$. ∎

4.5 Improvement in the Centralized Setting

Similar to the unweighted case, the Centralized-Weighted-DSG algorithm (Algorithm 10) runs Weighted-Noisy-Order-Packing-MWU (centralized) followed by Centralized-Peeling. The main twist here is that Weighted-Noisy-Order-Packing-MWU requires a guess λ for λ^*. This is chosen randomly from an arithmetic grid with spacing $12\tau\sqrt{\frac{\log n}{T}}$ covering the entire range of λ^* so as to ensure Theorem 4.2 can be applied.

Algorithm 10 Centralized-Weighted-DSG-core(G, T, ς)

```
1: Set $\tau = \sqrt{T}\varsigma$ and $N = \left\lceil \frac{2C_{\max}n}{4(n+\tau)\sqrt{\log(n)/T}} \right\rceil$.
2: Sample $k \sim [N]$ uniformly at random and set $\lambda = k \cdot 4(n+\tau)\sqrt{\frac{\log n}{T}}$.
3: Run Weighted-Noisy-Order-Packing-MWU$(G, \lambda, T, \tau)$ and obtain a permutation $\sigma$.
4: Output $(S, \tilde{\rho}(S)) \leftarrow$ Weighted-Peeling$(G, \sigma, \varsigma)$.
```

The following lemma is proved exactly on the lines of Lemma 3.8, together with the observation that the sampling of the index k doesn't depend on any private information, i.e., the edges of G:

LEMMA 4.6. *Centralized-Weighted-DSG-core(G, T, ς) is $\frac{1}{\varsigma^2}$-zCDP.*

LEMMA 4.7. *Suppose T and ς in Centralized-Weighted-DSG-core are set so that $\tau = \sqrt{T}\varsigma \geq n$. Then with probability at least $\frac{1}{4N}$, Centralized-Weighted-DSG-core(G, T, ς) returns a set S with $\tilde{\rho}(S) \geq \rho(G) - O(\sqrt{\log n}\varsigma)$.*

Proof. Let $\lambda^* = \rho(G)$. Note that with probability $\frac{1}{N}$, the guess λ chosen in Centralized-Weighted-DSG-core satisfies $\lambda^* \leq \lambda < \lambda^* + 4(n+\tau)\sqrt{\frac{\log n}{T}}$. Conditioned on this happening, Theorem 4.2 and Lemma 4.5 imply that with probability at least 1/2, Weighted-Noisy-Order-Packing-MWU outputs a permutation σ such that

$$\rho(S^*_\sigma) \geq \lambda - 16(n+\tau)\sqrt{\frac{\log n}{T}} \geq \lambda^* - 16(n+\tau)\sqrt{\frac{\log n}{T}}$$

since $\lambda \geq \lambda^*$. Thus, using Lemma 4.2, we conclude that with probability at least $\frac{1}{4N}$, the following holds for the set S and estimated density $\tilde{\rho}(S)$ output by Centralized-Weighted-DSG-core:

$$\rho(S) \geq \rho(S^*_\sigma) - O(\sqrt{\log n}\varsigma) \geq \lambda^* - O\left((n+\tau)\sqrt{\frac{\log n}{T}} + \sqrt{\log n}\varsigma\right) \tag{4.7}$$

and

$$|\rho(S) - \tilde{\rho}(S)| \leq O(\sqrt{\log n}\varsigma). \tag{4.8}$$

For the specified value of T and τ, we have

$$(n+\tau)\sqrt{\frac{\log n}{T}} + \sqrt{\log n}\varsigma = O(\sqrt{\log n}\varsigma).$$

Hence Eq. (4.7) and (4.8) imply that

$$\tilde{\rho}(S) \geq \lambda^* - O\left(\sqrt{\log(n)}\varsigma\right),$$

as required. ∎

Now, let Centralized-Weighted-DSG be the mechanism $\mathcal{A}_Q$ obtained by applying the mechanism of Theorem 3.5 with $Q =$ Centralized-Weighted-DSG-core, $s = S$, the set it outputs, and $q = \tilde{\rho}(S)$, the estimated density it outputs, $q^* = \rho(G) - O\left(\sqrt{\log(n)}\varsigma\right)$ as specified in Lemma 4.7, and $\gamma = n^{-c}$, for any given constant c.

THEOREM 4.4. *Let $\delta \in (0,1)$ and $\epsilon > 0$ be given privacy parameters. Set $\varsigma = \frac{6\sqrt{\log(n^c/\delta)}}{\epsilon}$ and $T = \lceil \frac{n^2}{\varsigma^2} \rceil$ in Centralized-Weighted-DSG-core. Then Centralized-Weighted-DSG is (ϵ,δ)-DP, and with probability at least $1-(4N+2)n^{-c}$, the density of the set it outputs is at least $\rho(G) - O\left(\frac{\sqrt{\log(n)\log(n/\delta)}}{\epsilon}\right)$.*

Proof. Using Lemma 4.7, we have

$$\Pr_{(s,q)\sim Q(D)}[q \geq q^*] \geq \frac{1}{4N}.$$

Since Centralized-Weighted-DSG-core(G,ς) is $\frac{1}{\varsigma^2}$-zCDP, applying Theorem 3.5, we conclude that Centralized-Weighted-DSG is

$$\left(\frac{6\sqrt{\log(1/(\gamma\delta))}}{\varsigma}, \delta\right)\text{-DP} = (\epsilon,\delta)\text{-DP}$$

for the specified value of ς, and the estimated density $\tilde{\rho}(S)$ of the set S it outputs is at least q^* with probability at least $1-4N\gamma$. Now, recall the random variable J used in Centralized-Weighted-DSG which is drawn from the geometric distribution with success probability γ. Since $\Pr[J > k] = (1-\gamma)^k \leq \exp(-\gamma k)$, we conclude that $\Pr[J \leq \frac{\log(1/\gamma)}{\gamma}] \geq 1-\gamma$. Conditioned on $J \leq \frac{\log(1/\gamma)}{\gamma}$, using Lemma 3.5 and a union bound over the J calls to Q in $\mathcal{A}_Q$, we conclude that with probability at least $1-\gamma$, in each call to Q, we have

$$|\rho(S) - \tilde{\rho}(S)| \leq O\left(\sqrt{\log n}\varsigma\right)$$

for an appropriately chosen constant in the $O(\cdot)$ notation. Thus, overall, using the union bound, with probability at least $1-(4N+2)\gamma$, the true density of the set output by Centralized-Weighted-DSG is at least

$$\rho(G) - O\left(\sqrt{\log n}\varsigma\right) = \rho(G) - O\left(\frac{\sqrt{\log(n)\log(n/\delta)}}{\epsilon}\right)$$

as claimed. ∎

5 Directed Densest Subgraph

In the directed version we are given a *directed* graph $G = (V,E)$. Given two subsets $S,T \subseteq V$, we let $E(S,T)$ denote the edges from S to T. The density of S,T is defined as,

$$\rho_G(S,T) = \frac{|E(S,T)|}{\sqrt{|S||T|}},$$

and the maximum subgraph density is $\rho(G) = \max_{S,T\subseteq V} \rho_G(S,T)$. This problem was introduced by [47], and has been received significant study since. Charikar [13] showed how to solve this problem exactly using $O(n^2)$ separate linear programs, and also showed that it was possible to give a $1-\epsilon$ approximation while solving only $O(\log n/\epsilon)$ linear programs. More recently, Sawlani and Wang [63] gave a *black-box* reduction to the node-weighted undirected setting. Since we designed an (ϵ,δ)-LEDP algorithm for the node-weighted undirected setting in Section 4, this reduction will form our starting point. As one piece of notation, for any node v and subset $T \subseteq V$, we let $\deg_T(v) = |\{(v,u) : u \in T\}|$ be the number of edges from v to nodes in T.

5.1 The reduction of Sawlani and Wang [63]

We now give a quick overview of the reduction from the directed case to the node-weighted undirected case given by [63]. Suppose we are given a directed graph $G = (V, E)$. Given a parameter t, we build a new node-weighted, undirected bipartite graph $G_t = (V_t, E_t, c_t)$. We do this by setting $V_t = V_t^L \cup V_t^R$, where V_t^L and V_t^R are both copies of V. For every edge (u, v) of E, we create an edge $\{u, v\} \in E_t$ with $u \in V_t^L$ and $v \in V_t^R$ (i.e., we create an undirected edge where the left endpoint is the tail of the original directed edge and the right endpoint is the head of the original directed edge). Our weight function c_t sets $c_t(u) = 1/(2t)$ for $u \in V_t^L$, and $c_t(u) = t/2$ for $u \in V_t^R$.

The first lemma says that no matter what the value of t, the density of sets in the directed graph is no smaller than the density of sets in the weighted undirected graph (and hence the optimal density of the weighted graph is at most the optimal density of the directed graph). This is primarily a consequence of the AM-GM inequality.

Lemma 5.1. (Lemma 5.1 of [63], slightly rephrased) *Let $S, T \subseteq V$, let t be arbitrary, let S^L denote the copies of S in V_t^L, and let T^R denote the copies of T in V_t^R. Then $\rho_G(S, T) \geq \rho_{G_t}(S^L \cup T^R)$.*

It turns out that for the "correct" guess of t, we have equality.

Lemma 5.2. (Lemma 5.2 of [63], slightly rephrased) *Let (S, T) be the optimal densest subgraph in G, i.e., $\rho(G) = \rho_G(S, T)$. Let $t = \sqrt{|S|/|T|}$. Let S^L be the copies of S in V_t^L, and let T^R be the copies of T in V_t^R. Then $\rho_G(S, T) = \rho_{G_t}(S^L \cup T^R)$.*

Since there are at most n possible choices for $|S|$ and n possible choices for $|T|$, there are at most n^2 different possible values of t for us to try. So the previous two lemmas imply that we can just try all n^2 of these possibilities and choose the best. Not surprisingly, it was also shown in [63] that it is possible to lose a $(1 - \epsilon)$-approximation factor and reduce this to only trying $O(\frac{1}{\epsilon} \log n)$ different values of t. We will use this fact, but not as a black box.

The main difficulty in combining this reduction with our node-weighted algorithm from Section 4 to get an LEDP algorithm for directed DSG is that for any value of t other than 1, either V_t^L or V_t^R will have node weights that are less than 1, and for the extreme value of t, the weights could be as small as $1/\sqrt{n}$. But our algorithm from Section 4 requires all node weights to be at least 1.

5.2 An LEDP algorithm for Directed DSG

Our algorithm is relatively simple: the curator can try different values of t, and then we can run Weighted-DSG-LEDP on each of the node-weighted undirected graphs resulting from the reduction of [63]. For each value of t the curator tries, it can announce that value to the nodes, and every node v will know the weight of its two copies. Node v can then act as if it is each of its copies when responding. The only subtlety is that we need to rescale so all node weights are at least 1, but this can clearly be done locally by the nodes since they will know t. This algorithm is presented as Directed-DSG-LEDP in Algorithm 11.

We begin by showing that this algorithm is private. To simplify notation, we will set $M = 2c \log_2(n) \cdot (\log_{1+\beta}(2n^{3/2}) + 1) \cdot \log_{1+\beta}(n)$.

Theorem 5.1. *Directed-DSG-LEDP$(G, T', \varsigma, c, \beta)$ satisfies $\frac{M}{\varsigma^2}$-zCDP.*

Proof. Fix an iteration i. We know from Lemma 4.4 that U_i is $\frac{K}{\varsigma^2}$-zCDP, and we know from the Gaussian mechanism (Lemma A.3) and parallel composition that the collection of all $\widehat{\deg}_{T_i}(v)$ values is $\frac{1}{\varsigma^2}$-zCDP. Everything else in iteration i is postprocessing, and hence each iteration is $\frac{2K}{\varsigma^2}$-zCDP. Note that K is a function of $C_{\max}$, but in our setting we know that $C_{\max} \leq \sqrt{n}$. Hence each iteration is $\frac{2c \log_2(n) \cdot (\log_{1+\beta}(2n^{3/2}) + 1)}{\varsigma^2}$-zCDP. Since there are $\log_{1+\beta}(n)$ iterations in total, sequential composition (Theorem A.1) implies the theorem. □

Corollary 5.1. *Let $\delta \in (0, 1)$ and $\epsilon \in (0, 8 \log(1/\delta))$ be given privacy parameters. Set $\varsigma = \frac{4\sqrt{M \log(1/\delta)}}{\epsilon}$ in Directed-DSG-LEDP. Then Directed-DSG-LEDP is (ϵ, δ)-LEDP.*

Proof. We first show that the algorithm can be implemented in the local model. This is straightforward, as all computations are local, except the fact that it calls Weighted-DSG-LEDP in step 6 on G'_{t_i}, rather than on G.

Algorithm 11 Directed-DSG-LEDP$(G, T', \varsigma, c, \beta)$

```
1: Let $t_0 = 1/\sqrt{n}$
2: for $i = 0$ to $\log_{1+\beta} n$ do
3:     Let $t_i = (1+\beta)^i t_0$
4:     The curator announces $t_i$
5:     For every node $v$, the curator and $v$ both compute $\alpha_i = \min(1/(2t_i), t_i/2)$, and compute the weight $c^L_{v,i} = 1/(2t_i\alpha_i)$ of its copy in $V^L_{t_i}$ and the weight $c^R_{v,i} = t_i/(2\alpha_i)$ of its copy in $V^R_{t_i}$. Let $G'_{t_i}$ denote $G_{t_i}$ but with these weights (which are scaled up by exactly $1/\alpha_i$ from the weights in $G_{t_i}$)
6:     Let $U_i \leftarrow$ Weighted-DSG-LEDP$(G'_{t_i}, T', \varsigma, c, \beta)$
7:     Let $S_i = U_i \cap V^L_{t_i}$ and $T_i = U_i \cap V^R_{t_i}$.
8:     The curator announces $S_i$ and $T_i$
9:     Every node $v \in S_i$ draws a random value $N_v^{(i)} \sim N(0, \varsigma^2)$ and sends $\widehat{\deg}_{T_i}(v) = \deg_{T_i}(v) + N_v^{(i)}$ to the curator.
10:    The curator computes $\widehat{\rho}(S_i, T_i) = \frac{\sum_{v \in S_i} \widehat{\deg}_{T_i}(v)}{\sqrt{|S_i||T_i|}}$
11: end for
12: The curator computes $i^* = \operatorname{argmax}_i \widehat{\rho}(S_i, T_i)$ and returns $(S_{i^*}, T_{i^*})$.
```

But note that every node knows the weight and incident edges of both of its copies in G'_{t_i} and so can simulate both of its copies, and the curator also knows the weights and the nodes in G'_{t_i} so can simulate the curator of Weighted-DSG-LEDP. So Directed-DSG-LEDP can be implemented in the local model, and hence we only need to show differential privacy.

Now we just need to transfer the zCDP guarantee of Theorem 5.1 to (ϵ, δ)-DP. By definition, the $\frac{M}{\varsigma^2}$-zCDP guarantee of Theorem 5.1 translates to a $\left(\alpha, \frac{\alpha M}{\varsigma^2}\right)$-RDP guarantee for any $\alpha \geq 1$ (Lemma A.5). This in turn translates to a $\left(\frac{\alpha M}{\varsigma^2} + \frac{\log(1/\delta)}{\alpha - 1}, \delta\right)$-DP guarantee for any $\alpha \geq 1$ (see Propostion 3 in [54], or Lemma A.6). Choosing $\alpha = \varsigma\sqrt{\frac{\log(1/\delta)}{M}} + 1$ for ς as specified above in the corollary statement, we get that Directed-DSG-LEDP is (ϵ, δ)-LEDP. ∎

We can now analyze the utility of Directed-DSG-LEDP.

THEOREM 5.2. *Suppose T' and ς are set so $\sqrt{T'}\varsigma \geq n$. Then with probability at least $1 - O(n^{-c})$, Directed-DSG-LEDP returns (S, T) such that*

$$\rho_G(S, T) \geq (1 - O(\beta))\rho(G) - O\left(\frac{1}{\beta}\varsigma\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)}\sqrt{\log n}\right),$$

where (S^*, T^*) is the optimal solution.

Proof. We will argue that for the "correct" i the algorithm will return a good solution, and that for any of the other $\log_{1+\beta} n$ values of i where the solution is significantly worse, our estimate of their density is sufficiently accurate so that we will not be fooled.

Let (S^*, T^*) be the optimal solution, and let $t = \sqrt{|S^*|/|T^*|}$. Clearly $1/\sqrt{n} \leq t \leq \sqrt{n}$, so by the definition of Directed-DSG-LEDP there is some i such that $t \leq t_i \leq (1+\beta)t$. Fix this i.

Note that all weights in G_t and in G_{t_i} are within $1+\beta$ of each other, so every set has density that is only different by at most a $1+\beta$ factor between the two. Let S^*_i denote the copies of S^* in $V^L_{t_i}$, and let T^*_i denote the copies of T^* in $V^R_{t_i}$. As in the algorithm, let $U_i = S_i \cup T_i$ be the set returned by Weighted-DSG-LEDP when called on G'_{t_i}, and let U'_i be the optimal solution for G'_{t_i}. We know from the Theorem 4.3 that with probability at least $1 - O(Kn^{-c})$,

$$\rho_{G'_{t_i}}(U_i) \geq (1 - 4\beta)\rho_{G'_{t_i}}(U'_i) - O\left(\frac{1}{\beta}\varsigma\sqrt{\log n}\right). \tag{5.9}$$

So we have:

$$\begin{aligned}
&\text{(Lemma 5.1)} & \rho_G(S_i,T_i) &\geq \rho_{G_t}(U_i) \\
&& &\geq \frac{1}{1+\beta}\rho_{G_{t_i}}(U_i) \\
&& &= \frac{1}{1+\beta}\cdot\frac{|E(U_i)|}{|S_i|\alpha_i c^L_{v,i}+|T_i|\alpha_i c^R_{v,i}} \\
&& &= \frac{1}{1+\beta}\cdot\frac{1}{\alpha_i}\cdot\rho_{G'_{t_i}}(U_i) \\
&\text{(Eq. (5.9))} & &\geq \frac{1}{1+\beta}\cdot\frac{1}{\alpha_i}\cdot\left((1-4\beta)\rho_{G'_{t_i}}(U'_i)-O\left(\frac{1}{\beta}\varsigma\sqrt{\log n}\right)\right) \\
&\text{(U'_i optimal for G'_{t_i})} & &\geq \frac{1}{1+\beta}\cdot\frac{1}{\alpha_i}\cdot\left((1-4\beta)\rho_{G'_{t_i}}(S^*_i\cup T^*_i)-O\left(\frac{1}{\beta}\varsigma\sqrt{\log n}\right)\right) \\
&& &= \frac{1}{1+\beta}\cdot\frac{1}{\alpha_i}\cdot\left(\alpha_i(1-4\beta)\rho_{G_{t_i}}(S^*_i\cup T^*_i)-O\left(\frac{1}{\beta}\varsigma\sqrt{\log n}\right)\right) \\
&& &\geq \frac{1}{(1+\beta)^2}(1-4\beta)\rho_{G_t}(S^*_i\cup T^*_i)-O\left(\frac{1}{\alpha_i\beta}\varsigma\sqrt{\log n}\right) \\
&& &\geq (1-O(\beta))\rho_{G_t}(S^*_i\cup T^*_i)-O\left(\frac{1}{\alpha_i\beta}\varsigma\sqrt{\log n}\right) \\
&\text{(Lemma 5.2)} & &= (1-O(\beta))\rho_G(S^*,T^*)-O\left(\frac{1}{\alpha_i\beta}\varsigma\sqrt{\log n}\right) \\
&& &\geq (1-O(\beta))\rho_G(S^*,T^*)-O\left(\frac{\varsigma\sqrt{\max\left(\frac{|S^*|}{|T^*|},\frac{|T^*|}{|S^*|}\right)}\sqrt{\log n}}{\beta}\right).
\end{aligned}$$

Now consider an *arbitrary* i (not necessarily the "right" i as above). Using essentially the argument as in Lemma 3.5, we know that with probability at least $1-O(n^{-c})$, we have

$$|\widehat{\rho}(S_i,T_i)-\rho(S_i,T_i)| \leq O\left(\varsigma\sqrt{\log n}\right).$$

Since this holds for all i, including the correct value of i, we get that with probability at least $1-O(Kn^{-c})$,

$$\begin{aligned}
& & \rho_G(S,T) &\geq \rho_G(S_i,T_i)-O\left(\varsigma\sqrt{\log n}\right) \\
&& &\geq (1-O(\beta))\rho_G(S^*,T^*)-O\left(\frac{\varsigma\sqrt{\max\left(\frac{|S^*|}{|T^*|},\frac{|T^*|}{|S^*|}\right)}\sqrt{\log n}}{\beta}\right)-O\left(\varsigma\sqrt{\log n}\right) \\
&(\because \beta\leq 1) & &\geq (1-O(\beta))\rho_G(S^*,T^*)-O\left(\frac{\varsigma\sqrt{\max\left(\frac{|S^*|}{|T^*|},\frac{|T^*|}{|S^*|}\right)}\sqrt{\log n}}{\beta}\right) \\
&& &= (1-O(\beta))\rho(G)-O\left(\frac{\varsigma\sqrt{\max\left(\frac{|S^*|}{|T^*|},\frac{|T^*|}{|S^*|}\right)}\sqrt{\log n}}{\beta}\right)
\end{aligned}$$

as claimed. □

Corollary 5.2. *Let $\delta \in (0,1)$ and $\epsilon \in (0, 8\log(1/\delta)$ be given privacy parameters. Set $\varsigma = \frac{4\sqrt{M\log(1/\delta)}}{\epsilon}$ and $T' = \lceil \frac{n^2}{\varsigma^2} \rceil$ in Directed-DSG-LEDP. Then Directed-DSG-LEDP is (ϵ,δ)-LEDP, and with probability at least $1 - O(Kn^{-c})$, DSG-LEDP returns a set (S,T) with*

$$\rho(S,T) \geq (1 - O(\beta))\rho(G) - O\left(\frac{1}{\beta^2\epsilon}\log^2 n\sqrt{\log(1/\delta)}\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)}\right).$$

Proof. The privacy guarantee follows from Corollary 5.1. For the output guarantee, we have that

(Theorem 5.2)

$$\rho(S,T) \geq (1 - O(\beta))\rho(G) - O\left(\frac{1}{\beta}\varsigma\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)}\sqrt{\log n}\right)$$

(def of ς)

$$\geq (1 - O(\beta))\rho(G) - O\left(\frac{1}{\beta\epsilon}\sqrt{M\log(1/\delta)\log n}\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)}\right)$$

$$= (1 - O(\beta))\rho(G) - O\left(\frac{1}{\beta\epsilon}\sqrt{\log_2(n)\cdot(\log_{1+\beta}(2n^{3/2})+1)\cdot\log_{1+\beta}(n)\log(1/\delta)\log n}\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)}\right)$$

$$\geq (1 - O(\beta))\rho(G) - O\left(\frac{1}{\beta^2\epsilon}\log^2 n\sqrt{\log(1/\delta)}\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)}\right),$$

as claimed. ◻

5.3 Improvement in the Centralized Setting

To obtain an improvement in the directed setting similar to what we obtained in the weighted setting, we can essentially replace the call to Weighted-DSG-LEDP in Directed-DSG-LEDP to a call to Centralized-Weighted-DSG, and then apply Theorem 3.5. In order to more fully utilize the directed setting, though, we will actually call Weighted-Noisy-Order-Packing-MWU instead. We also need to compute a new estimate of the (directed) density of the returned solution, since the estimate from the weighted reduction could be quite inaccurate if we choose the wrong parameters (t in particular). This gives the following algorithm.

Algorithm 12 Centralized-Directed-DSG-core(G, T, ς)

```
1: Set $\tau = \sqrt{T}\varsigma$, and $N = \left\lceil \frac{2n^2}{4(n+\tau)\sqrt{\log(n)/T}} \right\rceil$.
2: Sample $s', t' \sim [n]$ uniformly at random and set $t = \sqrt{s'/t'}$.
3: Sample $k \sim [N]$ uniformly at random and set $\lambda = k \cdot 4(n+\tau)\sqrt{\frac{\log n}{T}}$.
4: Set $\alpha = \min(1/(2t), t/2)$
5: For each node $v$, compute the weight $c_v^L = 1/(2t\alpha)$ of its copy in $V_t^L$ and the weight $c_v^R = t/(2\alpha)$ of its copy
   in $V_t^R$. Let $G_t'$ denote $G_t$ but with these weights (which are scaled up by exactly $1/\alpha$ from the weights in $G_t$)
6: Let $\sigma \leftarrow$ Weighted-Noisy-Order-Packing-MWU($G_t', \lambda, T, \tau$).
7: Let $(U, \tilde{\rho}(U)) \leftarrow$ Weighted-Peeling($G', \sigma, \varsigma$).
8: Let $S' = U \cap V_t^L$ and $T' = U \cap V_t^R$.
9: Let $\tilde{\rho}(S', T') = \rho(S', T') + N(0, \varsigma^2)$
10: Return $(S', T', \tilde{\rho}(S, T))$.
```

The following lemma is proved exactly on the lines of Lemma 4.6, together with the observation that the sampling of the values s, t and index k doesn't depend on any private information, i.e., the edges of G, and accounting for the privacy loss in line 9 of the algorithm:

Lemma 5.3. *Centralized-Weighted-DSG-core(G, T, ς) is $\frac{3}{2\varsigma^2}$-zCDP.*

Lemma 5.4. *Suppose T and ς in Centralized-Directed-DSG-core are set so that $\tau = \sqrt{T}\varsigma \geq n$. Then with probability at least $\frac{1}{8Nn^2}$, Centralized-Weighted-DSG-core(G, T, ς) returns a pair of sets (S', T') with $\tilde{\rho}(S', T') - O\left(\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)\log n \cdot \varsigma}\right)$.*

Proof. We can now proceed with the utility analysis essentially as we did in the weighted case. Let S^*, T^* be the optimal solution, and let $t^* = \sqrt{|S^*|/|T^*|}$. Then $t = t^*$ with probability at least $1/n^2$. Let λ^* denote the optimal density of G'_{t^*}, and note that by Lemma 5.1, Lemma 5.2, and the definition of G'_{t^*} we have $\lambda^* = \alpha\rho_G(S^*, T^*) = \alpha\rho(G)$. With probability $1/N$, the value λ picked by Centralized-Directed-DSG-core satisfies $\lambda^* \leq \lambda < \lambda^* + 4(n+\tau)\sqrt{\frac{\log n}{T}}$.

So the probability that both of these events occur is at least $1/(Nn^2)$. Conditioned on this, Theorem 4.2 and Lemma 4.5 imply that with probability at least $1/2$, Weighted-Noisy-Order-Packing-MWU outputs a permutation σ such that

$$\rho(S^*_\sigma) \geq \lambda - 16(n+\tau)\sqrt{\frac{\log n}{T}} \geq \lambda^* - 16(n+\tau)\sqrt{\frac{\log n}{T}}$$

since $\lambda \geq \lambda^*$.

Thus, using Lemma 4.2, we get that with probability at least $\frac{1}{4Nn^2}$,

$$\rho(U) \geq \rho(S^*_\sigma) - O(\sqrt{\log n}\varsigma) \geq \lambda^* - O\left((n+\tau)\sqrt{\frac{\log n}{T}} + \sqrt{\log n}\varsigma\right) \geq \lambda^* - O\left(\sqrt{\log n}\varsigma\right) \tag{5.10}$$

where the last inequality follows due to the specified value of T and τ.

Now we can reason about the density of the solution (S', T') returned by the algorithm. We have that with probability at least $\frac{1}{4Nn^2}$,

$$\begin{aligned}
&\text{(Lemma 5.1, def of } G'_t\text{)} && \rho(S', T') \geq \frac{1}{\alpha}\rho(U) \\
&\text{(Eq. (5.10))} && \geq \frac{1}{\alpha}\left(\lambda^* - O\left(\sqrt{\log n}\varsigma\right)\right) \\
& && \geq \rho(G) - O\left(\tfrac{1}{\alpha}\sqrt{\log n}\varsigma\right) \\
&\text{(def of } \alpha\text{)} && \geq \rho(G) - O\left(\sqrt{\max\left(\tfrac{|S^*|}{|T^*|}, \tfrac{|T^*|}{|S^*|}\right)\log n \cdot \varsigma}\right).
\end{aligned}$$

Now the standard concentration bound for the Gaussian mechanism (Lemma A.4) implies that with with probability at least $\frac{1}{8Nn^2}$, we not only have the above bound on $\rho(S', T')$, but also have that

$$\tilde{\rho}(S', T') \geq \rho(G) - O\left(\sqrt{\max\left(\tfrac{|S^*|}{|T^*|}, \tfrac{|T^*|}{|S^*|}\right)\log n \cdot \varsigma}\right).$$

as claimed. ∎

Now, let Centralized-Directed-DSG be the mechanism $\mathcal{A}_Q$ obtained by applying the mechanism of Theorem 3.5 with $Q =$ Centralized-Directed-DSG-core, $s = (S', T')$ (the pair of sets it outputs), $q = \tilde{\rho}(S', T')$ (the estimated density it outputs), $q^* = \rho(G) - O\left(\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)\log n \cdot \varsigma}\right)$ as specified in Lemma 5.4, and $\gamma = n^{-c}$ for any given constant c.

Theorem 5.3. *Let $\delta \in (0,1)$ and $\epsilon > 0$ be given privacy parameters. Set $\varsigma = \frac{8\sqrt{\log(n^c/\delta)}}{\epsilon}$ and $T = \lceil\frac{n^2}{\varsigma^2}\rceil$ in Centralized-Weighted-DSG-core. Then Centralized-Weighted-DSG is (ϵ, δ)-DP, and with probability at least $1 - (8Nn^2 + 2)n^{-c}$, the density of the set it outputs is at least $\rho(G) - O\left(\frac{\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)\log(n)\log(n/\delta)}}{\epsilon}\right)$.*

Proof. Using Lemma 5.4, we have

$$\Pr_{(s,q)\sim Q(D)}[q \geq q^*] \geq \frac{1}{8Nn^2}.$$

Since Centralized-Directed-DSG-core(G, ς) is $\frac{3}{2\varsigma^2}$-zCDP, applying Theorem 3.5, we conclude that Centralized-Weighted-DSG is

$$\left(\frac{8\sqrt{\log(1/(\gamma\delta))}}{\varsigma}, \delta\right)\text{-DP} = (\epsilon, \delta)\text{-DP}$$

for the specified value of ς, and the estimated density $\tilde{\rho}(S)$ of the pair of sets (S', T') it outputs is at least q^* with probability at least $1 - 8Nn^2\gamma$. Now, recall the random variable J used in Centralized-Weighted-DSG which is drawn from the geometric distribution with success probability γ. Since $\Pr[J > k] = (1-\gamma)^k \leq \exp(-\gamma k)$, we conclude that $\Pr[J \leq \frac{\log(1/\gamma)}{\gamma}] \geq 1 - \gamma$. Conditioned on $J \leq \frac{\log(1/\gamma)}{\gamma}$, using the Gaussian concentration bound (Lemma A.4) and a union bound over the J calls to Q in $\mathcal{A}_Q$, we conclude that with probability at least $1-\gamma$, in each call to Q, we have

$$|\rho(S', T') - \tilde{\rho}(S', T')| \leq O\left(\sqrt{\log n}\varsigma\right)$$

for an appropriately chosen constant in the $O(\cdot)$ notation. Thus, overall, using the union bound, with probability at least $1 - (8Nn^2 + 2)\gamma$, the true density of the pair of sets (S', T') output by Centralized-Weighted-DSG is at least

$$\rho(G) - O\left(\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)\log n} \cdot \varsigma\right) = \rho(G) - O\left(\frac{\sqrt{\max\left(\frac{|S^*|}{|T^*|}, \frac{|T^*|}{|S^*|}\right)\log(n)\log(n/\delta)}}{\epsilon}\right).$$

∎

6 A Simple ϵ-LEDP Algorithm

We now show that a simple modification of the parallel version [5] of Charikar's algorithm [13] can be implemented in the ϵ-LEDP model with only minor accuracy loss. We will simply add noise at each iteration in order to keep things private, and then appeal to parallel composition. We note that the the algorithm of [5] requires computing the average degree in the remaining graph in order to set the appropriate threshold, and this can easily be done in the ϵ-DP model (with noise added to keep it private), but cannot easily be done in the LEDP model since the curator cannot compute the average degree. Instead, we directly use the noisy individual degrees to compute a noisy average degree. This actually makes the running time and privacy analysis *simpler*, but makes it slightly more difficult to show the approximation bound.

Algorithm 13 Simple ϵ-LEDP

```
1: $S_1 := V$, $i := 1$ (this is public, or equivalently in the zero'th round the curator makes $V$ public).
2: while $S_i \neq \emptyset$ do
3:     For every vertex $v \in S_i$, let $D_i(v)$ be a noisy version of its remaining degree: $D_i(v) := d_{S_i}(v) + \mathsf{Geom}(e^{\epsilon})$
4:     Every vertex $v \in S_i$ sends $D_i(v)$ to the curator.
5:     The curator computes an estimate of $\rho(S_i)$ by setting $\hat{\rho}(S_i) = \frac{1}{|S_i|}\sum_{v \in S_i} D_i(v)/2$.
6:     The curator computes a noise threshold $T_i$:
           $T_i := (1+\eta) \cdot \frac{1}{|S_i|}\sum_{v \in S_i} D_i(v)$
7:     The curator computes $L_i := \{v \in S_i : D_i(v) \leq T_i\}$
8:     The curator computes and makes public $S_{i+1} := S_i \setminus L_i$; $i = i+1$
9: end while
10: The curator selects whichever $S_i$ has largest $\hat{\rho}(S_i)$.
```

We give our algorithm as Algorithm 13. It is easy to see that this algorithm can be implemented in the LEDP model, so we need to determine the running time (number of rounds), the privacy guarantee, and the approximation bound. We begin with the running time, since it is simple and will be useful when arguing about privacy.

LEMMA 6.1. *The number of iterations is at most* $O(\frac{1}{\eta}\log n)$.

Proof. Consider iteration i. By definition, T_i is $(1+\eta)$ times the average of the $D_i(v)$ values. So a simple averaging argument (i.e., Markov's inequality) implies that $|S_i \setminus L_i| \leq \frac{1}{1+\eta}|S_i|$. Thus the total number of iterations is at most $\log_{1+\eta} n = O(\frac{1}{\eta}\log n)$, as claimed. $\square$

6.1 Privacy Analysis

We now show that our algorithm satisfies ϵ-LEDP.

THEOREM 6.1. *The algorithm satisfies* $O\left(\frac{\epsilon}{\eta}\log n\right)$*-LEDP.*

Proof. Let k be the number of iterations in the algorithm. We know from Lemma 6.1 that $k = O\left(\frac{1}{\eta}\log n\right)$. Consider the entire sequence of values and sets chosen by the algorithm; we'll show that this sequence satisfies the claimed differential privacy bound. Slightly more formally, let $\Gamma_0 = (S_1)$, and let $\Gamma_i = (\{D_i(v)\}_{v\in S_i}, \hat{\rho}(S_i), T_i, L_i, S_{i+1})$ be the sets and values constructed in iteration i. We claim that the sequence $(\Gamma_i)_{i\in[k]}$ is $\frac{2\epsilon}{\eta}\log n$-DP. This implies that the entire algorithm is $\frac{2\epsilon}{\eta}\log n$-LEDP, since the final step is post-processing so by Theorem A.4 does not affect the privacy.

We claim by induction that the prefix $(\Gamma_j)_{0\leq j\leq i}$ is $i\epsilon$-DP for all $0 \leq i \leq k$. Plugging in $i = k$ then implies the claim. For the base case, note that (Γ_0) is clearly 0-DP, since for neighboring graphs G, G' we know that G and G' have the same vertex set $V = S_1$.

Now for the inductive step, consider some $1 \leq i \leq \ell$ and assume that the prefix $(\Gamma_j)_{0\leq j\leq i-1}$ satisfies $(i-1)\epsilon$-DP. The parallel composition theorem (Theorem A.3) and the Geometric Mechanism (Lemma A.2) imply that $\{D_i(v)\}_{v\in S_i}$ is 2ϵ-DP: The sensitivity of the degree is 1, but, as discussed earlier, every edge is in two parts of the partition. Everything in iteration i after computing the $\{D_i(v)\}$ values is post-processing, so Theorem A.4 implies that Γ_i is 2ϵ-DP. Finally, sequential composition (Theorem A.1) and the inductive hypothesis imply that $(\Gamma_j)_{0\leq j\leq i}$ is 2ϵ-DP as required. $\square$

6.2 Approximation

We begin by arguing that the noisy average we use $\hat{d}_i = \frac{1}{|S_i|}\sum_{v\in S_i} D_i(v)$ is concentrated around $d_{avg}(S_i)$. Before we can do this, though, we will need an additional probabilistic tool to give concentration bounds for averages of independent sub-exponential random variables: Bernstein's inequality (Theorem 2.8.3 of [65]).

DEFINITION 6.1. (SUB-EXPONENTIAL RANDOM VARIABLE: DEFINITION 2.7.5 OF [65]) *A random variable X is* sub-exponential *if the moment-generating function of X is bounded at some point, i.e., if* $\mathbb{E}[\exp(|X|/K)] \leq 2$ *for some constant K. The sub-exponential norm of X is the smallest K for which this is true: more formally, it is* $\inf\{t > 0 : \mathbb{E}[\exp(|X|/t)] \leq 2\}$.

LEMMA 6.2. (BERNSTEIN'S INEQUALITY, COROLLARY 2.8.3 OF [65]) *Let $X_1, \ldots, X_N$ be independent, mean zero, sub-exponential random variables. Then, for every $t \geq 0$, we have*

$$\Pr\left[\left|\frac{1}{N}\sum_{i=1}^{N} X_i \geq t\right|\right] \leq 2\cdot\exp\left(-cN\min\left(\frac{t^2}{K^2}, \frac{t}{K}\right)\right)$$

for some absolute constant $c > 0$, where K is the maximum sub-exponential norm of any X_i.

With this tool, we will be able to prove concentration for $\hat{d}_i$.

LEMMA 6.3. *Consider some iteration i. Then* $\Pr[|\hat{d}_i - d_{avg}(S_i)| > \Omega(\frac{1}{\epsilon}\log n)] \leq 1/n^4$

Proof. Clearly $\mathbb{E}[\hat{d}_i] = d_{avg}(S_i)$ by the symmetry of the added noise. We also claim that the symmetric geometric distribution with parameter e^ϵ (i.e., the noise added to each degree) is sub-exponential with sub-exponential norm $\Theta(1/\epsilon)$. This can easily be seen by the fact that the sub-exponential norm of a random variable X is equivalent, up to a universal constant, to the value of K such that $\Pr[|X| \geq t] \leq 2 \cdot \exp(-t/K)$ (see [65, Proposition 2.7.1]. So if we set $t = \frac{1}{\epsilon} \log \frac{1}{\sigma}$, Lemma A.2 implies that $\Pr[|\mathsf{Geom}(e^\epsilon)| \geq t] \leq \sigma = e^{-\epsilon t}$. Thus the sub-exponential norm of $\mathsf{Geom}(e^\epsilon)$ is $\Theta(1/\epsilon)$. So Bernstein's inequality implies that

$$\Pr\left[|\hat{d}_i - d_{avg}(S_i)| \geq t\right] \leq 2 \cdot \exp\left(-c|S_i| \min\left(\epsilon^2 t^2, \epsilon t\right)\right).$$

So if we set $t = \Theta(\frac{1}{\epsilon} \log n)$, we get that $\Pr[|\hat{d}_i - d_{avg}(S_i)| \geq \Omega(\frac{1}{\epsilon} \log n)] \leq 1/n^4$ as desired. ∎

We now show that in each iteration, nodes with large degree are likely to survive to the next iteration.

Lemma 6.4. *Consider some iteration i. Then there is some constant $c > 0$ such that for every $v \in S_i$, if*

$$d_{S_i}(v) > (1+\eta) d_{avg}(S_i) + c\frac{1+\eta}{\epsilon} \log n$$

then $\Pr[v \in L_i] \leq 1/n^3$.

Proof. Let $v \in S_i$ be a node with $d_{S_i}(v) > (1+\eta) d_{avg}(S_i) + \Omega\left(\frac{1+\eta}{\epsilon} \log n\right)$. Lemma 6.3 implies that $\hat{d}_i \leq d_{avg}(S_i) + O(\frac{1}{\epsilon} \log n)$ with probability at least $1 - 1/n^4$. So we may assume that this event occurs by adding $1/n^4$ to our failure probability.

By definition, $v \in L_i$ only if

$$\begin{aligned} d_{S_i}(v) + \mathsf{Geom}(e^\epsilon) \leq T_i &= (1+\eta) \cdot \frac{1}{|S_i|} \sum_{v \in S_i} D_i(v) = (1+\eta)\hat{d}_i \\ &\leq (1+\eta) \cdot \left(d_{avg}(S_i) + O\left(\frac{1}{\epsilon} \log n\right)\right). \end{aligned}$$

By our assumption on $d_{S_i}(v)$, this occurs only if

$$\begin{aligned} \mathsf{Geom}(e^\epsilon) &\leq (1+\eta) \cdot \left(d_{avg}(S_i) + O\left(\frac{1}{\epsilon} \log n\right)\right) - d_{S_i}(v) \\ &\leq (1+\eta) \cdot O\left(\frac{1}{\epsilon} \log n\right) - c\frac{1+\eta}{\epsilon} \log n. \end{aligned}$$

By setting c to be a large enough constant (say, a large constant factor larger than the constant hidden in the $O(\cdot)$ notation in the above inequality), we get that $v \in L_i$ only if $\mathsf{Geom}(e^{\epsilon_1}) \leq -c' \frac{1+\eta}{\epsilon} \log n$ for any constant c' that we want. Lemma A.2 now implies that this happens with probability at most $1/n^4$. Hence our total probability of v being in L_i is at most $1/n^4 + 1/n^4 \leq 1/n^3$, as claimed. ∎

This lemma immediately gives the following corollary.

Corollary 6.1. *With probability at least $1 - 1/n$, every L_i consists entirely of nodes with $d_{S_i}(v) \leq (1 + \eta) d_{avg}(S_i) + O\left(\frac{1+\eta}{\epsilon} \log n\right)$.*

Proof. Use Lemma 6.4 on every vertex in every iteration, using Lemma 6.1 to bound the number of iterations, and take a union bound. ∎

We can now prove the main approximation bound.

Theorem 6.2. *With probability at least $1 - 2/n$, the density of the subset returned by the algorithm is at least*

$$\frac{OPT}{2(1+\eta)} - O\left(\frac{1}{\epsilon} \log n\right).$$

Proof. We first argue that, like in the non-DP case [5], the best of the iterations is a good approximation. Then we argue that we return a solution that is essentially as good.

Let $\Lambda = O\left(\frac{1+\eta}{\epsilon}\log n\right)$ be the additive loss from Corollary 6.1. We know from Corollary 6.1 that with probability at least $1-1/n$ when every node is removed it has degree at most $d_{S_i}(v) \le (1+\eta)d_{avg}(S_i) + \Lambda$ (where i is the iteration in which v is removed). Now orient every edge towards whichever of the endpoints is removed earlier, making an arbitrary choice if both endpoints are removed in the same iteration. So the in-degree in this orientation of every node removed in iteration i is at most $(1+\eta)d_{avg}(S_i)+\Lambda$. Lemma 3 of [13] implies that the density of G is upper bounded by the maximum in-degree in any orientation (in particular the above orientation), i.e., $\rho(G) \le \max_{v\in V} d_{in}(v)$. Hence we have that

$$\rho(G) \le \max_{v\in V} d_{in}(v) \le \max_i((1+\eta)d_{avg}(S_i)+\Lambda) \le 2(1+\eta)\max_i \rho(S_i) + \Lambda. \tag{6.11}$$

So we know the best of the $k = O(\frac{1}{\eta}\log n)$ subgraphs is a good approximation (where we are using Lemma 6.1 to bound the number of iterations). However, we do not return the S_i with maximum $\rho(S_i)$, but rather the S_i with maximum estimated density $\hat{\rho}(S_i)$. But Lemma 6.3 implies that $\Pr[|\hat{\rho}(S_i)-\rho(S_i)| \ge \Omega\left(\frac{1}{\epsilon}\log n\right)] \le 1/n^3$ for each iteration i. So a trivial union bound implies that $|\hat{\rho}(S_i)-\rho(S_i)| \le O\left(\frac{1}{\epsilon}\log n\right)$ for all i with probability at least $1-1/n^2$. If this happens, then combined with (6.11) we get that

$$\rho(G) \le 2(1+\eta)\max_i \rho(S_i) + \Lambda \le 2(1+\eta)\max_i \hat{\rho}(S_i) + 2\Lambda.$$

This clearly implies the theorem. □

Putting it all together. We can now combine all of this into one easy-to-use corollary.

Corollary 6.2. *Let $\epsilon, \eta > 0$. There is an ϵ-LEDP algorithm for Densest Subgraph which runs in $O\left(\frac{1}{\eta}\log n\right)$ rounds and returns a set $S \subseteq V$ such that*

$$\rho(G) \le 2(1+\eta)\rho(S) + O\left(\frac{1}{\epsilon\eta}\log^2 n\right)$$

with probability at least $1-2/n$.

Proof. Use our algorithm but use DP parameter setting $\epsilon' = \Theta(\epsilon\eta/\log n)$. Then the number of rounds is implied by Lemma 6.1, Theorem 6.1 implies that the algorithm is ϵ-LEDP, and Theorem 6.2 implies the density bound. □

7 Private Density Approximation

In this section we show that if we want to return the *density* of the densest subgraph $\rho(G)$, rather than the set of nodes itself, then it is straightforward to have accuracy $O(\sqrt{1/\epsilon})$ in expectation or $O\left(\sqrt{\frac{\log n}{\epsilon}}\right)$ with high probability in the centralized edge-DP model. Notably, the lower bound of [56] implies that if we want to output a *set* $S \subseteq V$ of maximum density, then our expected additive loss must be at least $\Omega(\sqrt{\log n/\epsilon})$. Hence our upper bound implies that the expected additive loss must be strictly larger for outputting the set than for outputting its density.

To do this, we can use a variant of the propose-test-release [23] mechanism which takes advantage of the fact that a graph can only have large density if its sensitivity is low. This allows us to actually simplify the framework by just using a function with smaller global sensitivity to approximate the density.

More formally, for $x \in \mathbb{R}^+$, let $\rho_x(G) = \max(\rho(G), x)$ (where recall that $\rho(S) = |E(S)|/|S|$ and $\rho(G) = \max_{S\subset V}\rho(S)$). We will compute a differentially private approximation to $\rho_x(G)$ for appropriate x, and then claim that this is a good approximation to $\rho(G)$. Let us first analyze the sensitivity of the ρ_x function.

Lemma 7.1. *The sensitivity of ρ_x is at most $\frac{1}{2x-1}$.*

Proof. Let G and G' be two graphs that differ in exactly one edge $e = \{u, v\}$, which without loss of generality is contained in G' and not contained in G. We break into two cases depending on $\rho(G)$.

First, suppose that $\rho(G) \leq x - 1$. Then clearly $\rho(G') \leq x$, and hence $\rho_x(G) = \rho_x(G') = x$. Thus the sensitivity is 0.

For the more interesting case, suppose that $\rho(G) > x - 1$. Let $S \subseteq V$ be the densest subgraph of G'. If $\{u, v\} \not\subseteq S$ then S has the same density in both G and G' and no other set has larger density in G than in G'. Hence $\rho(G) = \rho(G')$ and so $\rho_x(G) = \rho_x(G')$ and we are done. So assume without loss of generality that $u, v \in S$. Then $\rho(G) \geq \frac{E_{G'}(S)-1}{|S|} = \rho(G') - \frac{1}{|S|}$. Moreover, since $|E(S)| \leq \binom{|S|}{2}$ for any $S \subseteq V$, we have that $\rho(G') = \rho(S) \leq \binom{|S|}{2}/|S| = \frac{|S|-1}{2}$, and so $|S| \geq 2\rho(G') + 1 \geq 2\rho(G) + 1 > 2x - 1$. Hence

$$\rho(G) \geq \rho(G') - \frac{1}{2x-1},$$

and thus $\rho_x(G') - \rho_x(G) \leq \frac{1}{2x-1}$ as claimed. ◻

Now we can define our algorithm: we use the Laplace mechanism on $\rho_x(G)$ (see Lemma A.1 and [24]). Slightly more formally, we compute $\rho_x(G)$, draw noise $N \sim \text{Lap}\left(\frac{1}{(2x-1)\epsilon}\right)$, and return $\widehat{\rho}_x(G) = \rho_x(G) + N$.

Lemma 7.2. *This algorithm is ϵ-edge DP.*

Proof. This follows directly from Lemma 7.1 and the standard analysis of the Laplace mechanism (Lemma A.1). ◻

Lemma 7.3. *The expectation of $|\widehat{\rho}_x(G) - \rho(G)|$ is at most $O\left(\frac{1}{(2x-1)\epsilon}\right) + x$. And with high probability, $|\widehat{\rho}_x(G) - \rho(G)| \leq O\left(\frac{\log n}{(2x-1)\epsilon}\right) + x$.*

Proof. Clearly $\mathbb{E}[|\widehat{\rho}_x(G) - \rho(G)|] \leq x + \mathbb{E}[|N|] = O\left(\frac{1}{(2x-1)\epsilon}\right) + x$ (see Lemma A.1). Moreover, standard tail bounds for the Laplace distribution imply that $|N| \leq O\left(\frac{\log n}{(2x-1)\epsilon}\right)$ with high probability. If this event occurs, then we have that

$$|\widehat{\rho}_x(G) - \rho(G)| \leq N + \rho_x(G) - \rho(G) \leq O\left(\frac{\log n}{(2x-1)\epsilon}\right) + x$$

as claimed. ◻

It immediately follows that if we set $x = \Theta\left(\sqrt{\frac{\log n}{\epsilon}}\right)$, then with high probability $|\widehat{\rho}_x(G) - \rho(G)| \leq O\left(\sqrt{\frac{\log n}{\epsilon}}\right)$, so we have the following corollary:

Corollary 7.1. *There is an ϵ-edge DP algorithm which outputs a value $\widehat{\rho}$ such that $|\widehat{\rho} - \rho(G)| \leq O\left(\sqrt{\frac{\log n}{\epsilon}}\right)$ with high probability.*

On the other hand, if we set $x = \Theta(\sqrt{1/\epsilon})$, we get the following corollary.

Corollary 7.2. *There is an ϵ-edge DP algorithm which outputs a value $\widehat{\rho}$ such that $\mathbb{E}[|\widehat{\rho} - \rho(G)|] \leq O\left(\sqrt{\frac{1}{\epsilon}}\right)$.*

Acknowledgments

The authors wish to thank Abhradeep Guha Thakurta and Kunal Talwar for helpful discussions.

References

[1] T. Akiba, Y. Iwata, and Y. Yoshida. Fast exact shortest-path distance queries on large networks by pruned landmark labeling. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pages 349–360, 2013.

[2] A. Angel, N. Koudas, N. Sarkas, D. Srivastava, M. Svendsen, and S. Tirthapura. Dense subgraph maintenance under streaming edge weight updates for real-time story identification. *The VLDB journal*, 23:175–199, 2014.

[3] R. Arora and J. Upadhyay. On differentially private graph sparsification and applications. *Advances in neural information processing systems*, 32, 2019.

[4] S. Arora, E. Hazan, and S. Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012.

[5] B. Bahmani, R. Kumar, and S. Vassilvitskii. Densest subgraph in streaming and mapreduce. *Proc. VLDB Endow.*, 5(5):454–465, jan 2012.

[6] V. Balcer and S. Vadhan. Differential Privacy on Finite Computers. In A. R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 43:1–43:21, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[7] S. Bhattacharya, M. Henzinger, D. Nanongkai, and C. Tsourakakis. Space-and time-efficient algorithm for maintaining dense subgraphs on one-pass dynamic streams. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 173–182, 2015.

[8] D. Boob, Y. Gao, R. Peng, S. Sawlani, C. Tsourakakis, D. Wang, and J. Wang. Flowless: Extracting densest subgraphs without flow computations. In *Proceedings of The Web Conference 2020*, WWW '20, page 573–583, New York, NY, USA, 2020. Association for Computing Machinery.

[9] M. Bun, M. Elias, and J. Kulkarni. Differentially private correlation clustering. In *International Conference on Machine Learning*, pages 1136–1146. PMLR, 2021.

[10] M. Bun and T. Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

[11] J. Cadena, F. Chen, and A. Vullikanti. Graph anomaly detection based on steiner connectivity and density. *Proceedings of the IEEE*, 106(5):829–845, 2018.

[12] J. Cadena, A. K. Vullikanti, and C. C. Aggarwal. On dense subgraphs in signed network streams. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 51–60. IEEE, 2016.

[13] M. Charikar. Greedy approximation algorithms for finding dense components in a graph. In K. Jansen and S. Khuller, editors, *Approximation Algorithms for Combinatorial Optimization, Third International Workshop, APPROX 2000, Saarbrücken, Germany, September 5-8, 2000, Proceedings*, volume 1913 of *Lecture Notes in Computer Science*, pages 84–95. Springer, 2000.

[14] C. Chekuri, K. Quanrud, and M. R. Torres. Densest subgraph: Supermodularity, iterative peeling, and flow. In J. S. Naor and N. Buchbinder, editors, *Proceedings of the 2022 ACM-SIAM Symposium on Discrete Algorithms, SODA 2022, Virtual Conference / Alexandria, VA, USA, January 9 - 12, 2022*, pages 1531–1555. SIAM, 2022.

[15] H. Chen, V. Cohen-Addad, T. d'Orsi, A. Epasto, J. Imola, D. Steurer, and S. Tiegel. Private estimation algorithms for stochastic block models and mixture models. *Advances in Neural Information Processing Systems*, 36:68134–68183, 2023.

[16] J. Chen and Y. Saad. Dense subgraph extraction with application to community detection. *IEEE Transactions on knowledge and data engineering*, 24(7):1216–1230, 2010.

[17] V. Cohen-Addad, C. Fan, S. Lattanzi, S. Mitrovic, A. Norouzi-Fard, N. Parotsidis, and J. M. Tarnawski. Near-optimal correlation clustering with privacy. *Advances in Neural Information Processing Systems*, 35:33702–33715, 2022.

[18] G. Cormode, S. Jha, T. Kulkarni, N. Li, D. Srivastava, and T. Wang. Privacy at scale: Local differential privacy in practice. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, page 1655–1658, New York, NY, USA, 2018. Association for Computing Machinery.

[19] M. Dalirrooyfard, S. Mitrovic, and Y. Nevmyvaka. Nearly tight bounds for differentially private multiway cut. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

[20] L. Dhulipala, G. Z. Li, and Q. C. Liu. Near-optimal differentially private k-core decomposition. *CoRR*, abs/2312.07706, 2023.

[21] L. Dhulipala, Q. C. Liu, S. Raskhodnikova, J. Shi, J. Shun, and S. Yu. Differential privacy from locally adjustable graph algorithms: k-core decomposition, low out-degree ordering, and densest subgraphs. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 754–765. IEEE, 2022.

[22] Y. Dourisboure, F. Geraci, and M. Pellegrini. Extraction and classification of dense communities in the web. In *Proceedings of the 16th international conference on World Wide Web*, pages 461–470, 2007.

[23] C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 371–380, New York, NY, USA, 2009. Association for Computing Machinery.

[24] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith. Calibrating noise to sensitivity in private data analysis. In *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006.

[25] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, aug 2014.

[26] T. Eden, Q. C. Liu, S. Raskhodnikova, and A. D. Smith. Triangle counting with local edge differential privacy. In K. Etessami, U. Feige, and G. Puppis, editors, *50th International Colloquium on Automata, Languages, and Programming, ICALP 2023, July 10-14, 2023, Paderborn, Germany*, volume 261 of *LIPIcs*, pages 52:1–52:21. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023.

[27] M. Eliáš, M. Kapralov, J. Kulkarni, and Y. T. Lee. Differentially private release of synthetic graphs. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 560–578. SIAM, 2020.

[28] A. Epasto, S. Lattanzi, and M. Sozio. Efficient densest subgraph computation in evolving graphs. In *Proceedings of the 24th international conference on world wide web*, pages 300–310, 2015.

[29] H. Esfandiari, M. Hajiaghayi, and D. P. Woodruff. Brief announcement: Applications of uniform sampling: Densest subgraph and beyond. In *Proceedings of the 28th ACM Symposium on Parallelism in Algorithms and Architectures*, pages 397–399, 2016.

[30] C. Fan, P. Li, and X. Li. Private graph all-pairwise-shortest-path distance release with improved error rate. *Advances in Neural Information Processing Systems*, 35:17844–17856, 2022.

[31] A. Farhadi, M. Hajiaghayi, and E. Shi. Differentially private densest subgraph. In *International Conference on Artificial Intelligence and Statistics*, pages 11581–11597. PMLR, 2022.

[32] A. Farhadi, M. Hajiaghayi, and E. Shi. Differentially private densest subgraph. In G. Camps-Valls, F. J. R. Ruiz, and I. Valera, editors, *International Conference on Artificial Intelligence and Statistics, AISTATS 2022, 28-30 March 2022, Virtual Event*, volume 151 of *Proceedings of Machine Learning Research*, pages 11581–11597. PMLR, 2022.

[33] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

[34] M. Ghaffari, S. Lattanzi, and S. Mitrović. Improved parallel algorithms for density-based network clustering. In *International Conference on Machine Learning*, pages 2201–2210. PMLR, 2019.

[35] A. Ghosh, T. Roughgarden, and M. Sundararajan. Universally utility-maximizing privacy mechanisms. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 351–360, New York, NY, USA, 2009. Association for Computing Machinery.

[36] D. Gibson, R. Kumar, and A. Tomkins. Discovering large dense subgraphs in massive graphs. In *Proceedings of the 31st international conference on Very large data bases*, pages 721–732, 2005.

[37] A. V. Goldberg. Finding a maximum density subgraph. 1984.

[38] A. Gupta, K. Ligett, F. McSherry, A. Roth, and K. Talwar. Differentially private combinatorial optimization. In M. Charikar, editor, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1106–1125. SIAM, 2010.

[39] A. Gupta, K. Ligett, F. McSherry, A. Roth, and K. Talwar. Differentially private combinatorial optimization. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, pages 1106–1125. SIAM, 2010.

[40] A. Gupta, A. Roth, and J. Ullman. Iterative constructions and private data release. In *Theory of Cryptography: 9th Theory of Cryptography Conference, TCC 2012, Taormina, Sicily, Italy, March 19-21, 2012. Proceedings 9*, pages 339–356. Springer, 2012.

[41] M. Hardt, K. Ligett, and F. McSherry. A simple and practical algorithm for differentially private data release. *Advances in neural information processing systems*, 25, 2012.

[42] M. Hardt and G. N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, page 61–70, USA, 2010. IEEE Computer Society.

[43] M. Henzinger, A. R. Sricharan, and L. Zhu. Tighter bounds for local differentially private core decomposition and densest subgraph, 2024.

[44] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos. Fraudar: Bounding graph fraud in the face of camouflage. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 895–904, 2016.

[45] J. Hsu, A. Roth, T. Roughgarden, and J. Ullman. Privately solving linear programs. In J. Esparza, P. Fraigniaud, T. Husfeldt, and E. Koutsoupias, editors, *Automata, Languages, and Programming*, pages 612–624, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[46] G. Kamath. Lecture 4 — Intro to Differential Privacy, Part 2. CS 860: Algorithms for Private Data Analysis. Available at `http://www.gautamkamath.com/CS860notes/lec4.pdf`, 2020.

[47] R. Kannan and V. Vinay. Analyzing the structure of large graphs. Unpublished manuscript, 1999.

[48] S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[49] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 611–617, 2006.

[50] D. Liu. Better private algorithms for correlation clustering. In *Conference on Learning Theory*, pages 5391–5412. PMLR, 2022.

[51] J. Liu and K. Talwar. Private selection from private candidates. In *51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 298–309, 2019.

[52] A. McGregor, D. Tench, S. Vorotnikova, and H. T. Vu. Densest subgraph in dynamic graph streams. In *International Symposium on Mathematical Foundations of Computer Science*, pages 472–482. Springer, 2015.

[53] F. McSherry. Privacy integrated queries. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD)*. Association for Computing Machinery, Inc., June 2009. For more information, visit the project page: http://research.microsoft.com/PINQ.

[54] I. Mironov. Rényi differential privacy. In *30th IEEE Computer Security Foundations Symposium, CSF 2017, Santa Barbara, CA, USA, August 21-25, 2017*, pages 263–275. IEEE Computer Society, 2017.

[55] M. S. Mohamed, D. Nguyen, A. Vullikanti, and R. Tandon. Differentially private community detection for stochastic block models. In *International Conference on Machine Learning*, pages 15858–15894. PMLR, 2022.

[56] D. Nguyen and A. Vullikanti. Differentially private densest subgraph detection. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8140–8151. PMLR, 2021.

[57] K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, page 75–84, New York, NY, USA, 2007. Association for Computing Machinery.

[58] N. Papernot and T. Steinke. Hyperparameter tuning with renyi differential privacy. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

[59] S. A. Plotkin, D. B. Shmoys, and Éva Tardos. Fast approximation algorithms for fractional packing and covering problems. *Mathematics of Operations Research*, 20(2):257–301, 1995.

[60] P. Rozenshtein, A. Anagnostopoulos, A. Gionis, and N. Tatti. Event detection in activity networks. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1176–1185, 2014.

[61] B. Saha, A. Hoch, S. Khuller, L. Raschid, and X.-N. Zhang. Dense subgraphs with restrictions and applications to gene annotation graphs. In *Research in Computational Molecular Biology: 14th Annual International Conference, RECOMB 2010, Lisbon, Portugal, April 25-28, 2010. Proceedings 14*, pages 456–472. Springer, 2010.

[62] S. Sawlani and J. Wang. Near-optimal fully dynamic densest subgraph. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 181–193, 2020.

[63] S. Sawlani and J. Wang. Near-optimal fully dynamic densest subgraph. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 181–193, New York, NY, USA, 2020. Association for Computing Machinery.

[64] S. P. Vadhan. The complexity of differential privacy. In Y. Lindell, editor, *Tutorials on the Foundations of Cryptography*, pages 347–450. Springer International Publishing, 2017.

[65] R. Vershynin. *High-Dimensional Probability: An Introduction with Applications in Data Science*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2018.

A Differential Privacy Basics

In this appendix we provide details about the basic DP definitions and primitives that we use. We begin with a formal discussion of the LEDP model.

A.1 Local Edge Differential Privacy

We use the formalization of *local* edge-differential privacy from [21], suitably modified to handle (ϵ, δ)-DP rather than just pure ϵ-DP. In this formalization, LEDP algorithms are described in terms of an (untrusted) curator, who does not have access to the graph's edges, and individual nodes. During each round, the curator first queries a set of nodes for information. Individual nodes, which have access only to their own (private) adjacency lists (and whatever information was sent by the curator), then release information via local randomizers, defined next.

Definition A.1. (Local Randomizer [21]) *An* (ϵ, δ)-local randomizer $R : a \to \mathcal{Y}$ *for node* v *is an* (ϵ, δ)*-edge DP algorithm that takes as input the set of its neighbors* $N(v)$*, represented by an adjacency list* $a = (b_1, \ldots, b_{|N(v)|})$.

In other words, a local randomizer is just an edge-DP algorithm where the private input is the adjacency list of v. Note that such an algorithm can use other (public) information, such as whatever has been broadcast by the curator. The information released via local randomizers is public to all nodes and the curator. The curator performs some computation on the released information and makes the result public. The overall computation is formalized via the notion of the transcript.

Definition A.2. (LEDP [21]) *A* transcript π *is a vector consisting of* 4*-tuples* $(S_U^t, S_R^t, S_\epsilon^t, S_Y^t)$ *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. A* protocol *is an algorithm* $\mathcal{A} : S_\pi \to (2^{[n]} \times 2^{S_R} \times 2^{\mathbb{R}^{\geq 0}} \times 2^{\mathbb{R}^{\geq 0}}) \cup \{\perp\}$ *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 $\epsilon, \delta \geq 0$*, a randomized protocol* $\mathcal{A}$ *on a (distributed) graph* G *is* (ϵ, δ)*-locally edge differentially private (*(ϵ, δ)*-LEDP) if the algorithm that outputs the entire transcript generated by* $\mathcal{A}$ *is* (ϵ, δ)*-edge differentially private on graph* G.

As in [21], we assume each user can see the public information for each round on a public "bulletin board". If an algorithm is $(\epsilon, 0)$-LEDP then we say that it is ϵ-LEDP.

This definition is somewhat unwieldy, and moreover does not neatly align with the intuitive explanation given in terms of an untrusted curator (the curator does not appear at all in the above definition of a protocol). Fortunately, it is not hard to see the correspondence. This was implicit in [21], but we make it explicit here.

Suppose that we have an untrusted curator which initially knows only V, and each node initially knows only V and its incident edges. Suppose that every node runs an (ϵ, δ)-edge DP algorithm in every round, and broadcasts the output of this algorithm. Then in each round, the algorithm run by a node is aware of all of the public information and the (private) local edge information. If we think of this as an algorithm which has the public information hard-coded in and which takes the local edge information as input, then this is an (ϵ, δ)-local randomizer. So the local randomizer in every round is exactly a function of the previous public information, as required. And the curator's choice of who to query and with what question is a function from the previous transcript to a set of parties and randomizers, and hence this satisfies the definition of a "protocol" from Definition A.2. Hence we will give our algorithms in terms of a multi-round algorithm with an untrusted curator and (ϵ, δ)-edge DP algorithms at each node, rather than through local randomizers and protocols.

A.2 Useful DP Primitives

We first give the simple and well-known differential private mechanisms that form the building blocks of our algorithms, and then the various composition theorems that we will use to combine them.

A.2.1 Basic Mechanisms For a function f from datasets to $\mathbb{R}$, let Δ_f denote the global sensitivity of f, i.e., the maximum over neighboring datasets X, X' of $f(X) - f(X')$. All of the basic mechanisms will depend on Δ_f.

The first mechanism is the standard Laplace mechanism.

Lemma A.1. (Laplace Mechanism [24]) *Adding noise drawn from the Laplace distribution with parameter Δ_f/ϵ satisfies ϵ-differential privacy. Moreover, the expected additive loss of the Laplace mechanism is $O(\Delta_f/\epsilon)$, and with high probability the loss is at most $O((\Delta_f/\epsilon)\log n)$.*

The next basic mechanism is essentially the discrete analog of the standard Laplace mechanism.

Definition A.3. (Symmetric geometric distribution [6,35]) *Let $\gamma > 1$. The symmetric geometric distribution $\mathsf{Geom}(\gamma)$ takes integer values such that the probability mass function at k is $\frac{\gamma-1}{\gamma+1} \cdot \gamma^{-|k|}$.*

Lemma A.2. (Geometric Mechanism (see [6,32,35])) *The Geometric Mechanism for query f is the function $f'(x) + \mathsf{Geom}(\exp(\epsilon/\Delta_f))$. The geometric mechanism satisfies ϵ-DP. Moreover, for every input x and $\sigma \in (0,1)$, the error $|f(x) - f'(x)|$ of the geometric mechanism is at most $O(\frac{\Delta_f}{\epsilon} \cdot \log \frac{1}{\sigma})$ with probability at least $1-\sigma$.*

The following mechanism allows us to add noise to achieve zCDP rather than pure DP.

Lemma A.3. (Gaussian Mechanism for zCDP [10]) *Let $\rho \in (0,1)$ be arbitrary. Adding noise drawn from $N(0,\sigma^2)$ is $(\Delta_f^2/2\sigma^2)$-zCDP.*

The following standard concentration bound will be useful when analyzing the Gaussian Mechanism.

Lemma A.4. *Let $X \sim N(0,\sigma^2)$. Then* $\Pr[|X - \mathbb{E}[X]| \geq t] \leq 2\exp(-t^2/\sigma^2)$

A.2.2 zCDP to Approximate DP Most of our intermediate results in this paper are stated in terms of zCDP, which we then translate to (ϵ,δ)-DP for our final results. This translation generally goes through *Rényi Differential Privacy* (RDP), introduced by [54], whichis defined as follows.

Definition A.4. *Algorithm $\mathcal{A}(G)$ that takes as input a graph G and outputs something in $R(\mathcal{A})$ is* (α,ϵ)-RDP *if for all $\alpha \in (0,1)$ and all edge-neighboring graphs G and G',*

$$D_\alpha(\mathcal{A}(G)||\mathcal{A}(G')) \leq \epsilon,$$

where D_α is the Rényi divergence of order α.

The following is immediate from Definition 1.4.

Lemma A.5. *If an algorithm satisfies ρ-zCDP, then it satisfies $(\alpha,\rho\alpha)$-RDP for all $\alpha > 1$.*

An RDP guarantee can then be translated into an (ϵ,δ)-DP guarantee using Proposition 3 of [54]:

Lemma A.6. (Proposition 3 of [54]) *If an algorithm satisfies (α,ϵ)-RDP, then it satisfies $(\epsilon + \frac{\log(1/\delta)}{\alpha-1}, \delta)$-DP for any $0 < \delta < 1$.*

A.2.3 Composition Theorems We will need a few different composition theorems. All of them are standard and well-known, and can be found in standard textbooks [25,64].

Theorem A.1. (Adaptive Sequential Composition [10,46,53]) *Suppose $M = (M_1,\ldots,M_k)$ is a sequence of (ϵ,δ)-differentially private algorithms, potentially chosen sequentially and adaptively. Then M is $(k\epsilon,k\delta)$-differentially private. If each M_i is ρ-zCDP, then M is $k\rho$-zCDP.*

Theorem A.2. (Advanced Composition [25, Corollary 3.21]) *Given target privacy parameters $0 < \epsilon' < 1$ and $\delta' > 0$, to ensure $(\epsilon', k\delta + \delta')$ cumulative privacy loss over k mechanisms, it suffices that each mechanism is $\left(\frac{\epsilon'}{2\sqrt{2k\log(1/\delta')}}, \delta\right)$-differentially private.*

Theorem A.3. (Parallel Composition [53]) *Let M_i each provide (ϵ,δ)-DP. Let D_i be arbitrary disjoint subsets of the input domain D, and let $X \subseteq D$ be a database. Then the sequence of $M_i(X \cap D_i)$ is (ϵ,δ)-differentially private. If each M_i is ρ-zCDP, then the sequence of $M_i(X \cap D_i)$ is ρ-zCDP.*

In a few places we will use a slight extension of parallel composition: if every element of D is in at most *two* of the D_i sets (rather than at most 1, in the disjoint setting), then the sequence of $(M_i(X \cap D_i))$ is $(2\epsilon, 2\delta)$-DP. This is a straightforward consequence of Theorem A.1 and Theorem A.3.

Theorem A.4. (Post-Processing) *Let M be an (ϵ, δ)-differentially private mechanism, and let f be a randomized or deterministic function whose domain is the range of M. Then the randomized function $g(x) = f(M(x))$ is (ϵ, δ)-differentially private. If M is ρ-zCDP, then g is ρ-zCDP.*

A.2.4 Connection to LEDP

All of the previous composition theorems are phrased in terms of centralized differential privacy. It is not hard, however, to see that they continue to hold in the LEDP model.

To see this, consider an algorithm in the LEDP model: an untrusted curator knowing only public information (V, and whatever is output by the nodes and the curator itself along the way), and an algorithm at each node that has access to the public information and its incident edges. Suppose that each node runs an (ϵ, δ)-edge DP algorithm on its incident edges. Then when the computation for this round is viewed as a whole, every edge is in the private information of exactly two nodes, and hence parallel composition and sequential composition imply that the combined output of a round (the collection of all outputs from all nodes) is $(2\epsilon, 2\delta)$-edge DP. The computation done at the curator is simply post-processing (since it does not access any private information directly), so by Theorem A.4 we can view each round as a $(2\epsilon, 2\delta)$-edge DP algorithm in the traditional centralized DP model, and then sequential composition and advanced composition apply as usual. Since we will not care about constant factors in the privacy parameters, we will simply treat each round as being (ϵ, δ)-edge DP (we can always go back and set our true privacy parameters to half of whatever we would set them to).

B Proofs from Section 2

Proof. [Proof of Theorem 2.1] As in the standard analysis of the Multiplicative Weights method, we use $\Phi^{(t)} = \log(\sum_{i=1}^n w_i^{(t)})$ as a potential function and track its evolution over time. We have

$$\Phi^{(t+1)} = \log\left(\langle \exp(-\eta \hat{m}^{(t)}), w^{(t)}\rangle\right) = \Phi^{(t)} + \log\left(\langle \exp(-\eta \hat{m}^{(t)}), p^{(t)}\rangle\right)$$

In the following, let $\vec{1}$ denote the all 1's vector, and for a vector v, let v^2 denote the vector obtained by squaring v coordinate-wise. Let $\mathbb{E}^{(t)}[\cdot]$ denote the expectation conditioned on all the randomness up to and including round t. Then we have

$$\begin{aligned}
\mathbb{E}^{(t)}[\Phi^{(t+1)}] &= \mathbb{E}^{(t)}\left[\Phi^{(t)} + \log\left(\langle \exp(-\eta \hat{m}^{(t)}), p^{(t)}\rangle\right)\right] \\
&\le \Phi^{(t)} + \log \mathbb{E}^{(t)}\left[\left(\langle \exp(-\eta \hat{m}^{(t)}), p^{(t)}\rangle\right)\right] \qquad \text{(Jensen's inequality: } \log \mathbb{E}[X] \ge \mathbb{E}[\log X]) \\
&= \Phi^{(t)} + \log\left(\left\langle \mathbb{E}^{(t)}\left[\exp(-\eta \hat{m}^{(t)})\right], p^{(t)}\right\rangle\right) \qquad (p^{(t)} \text{ is deterministic conditioned on the past}) \\
&= \Phi^{(t)} + \log\left(\langle \exp\left(-\eta m^{(t)} + \tfrac{\eta^2\nu^2}{2}\vec{1}\right), p^{(t)}\rangle\right) \qquad \text{(using the formula for the mgf of a normal distribution)} \\
&\le \Phi^{(t)} + \log\left(\left\langle \vec{1} - \eta m^{(t)} + \tfrac{\eta^2\nu^2}{2}\vec{1} + (-\eta m^{(t)} + \tfrac{\eta^2\nu^2}{2}\vec{1})^2, p^{(t)}\right\rangle\right) \\
&\qquad \text{(since } |m_i^{(t)}|, \nu, \eta \in [0,1], \text{ and using the fact that } \exp(x) \le 1 + x + x^2 \text{ for } |x| \le 1.5) \\
&\le \Phi^{(t)} + \log(1 - \eta\langle m^{(t)}, p^{(t)}\rangle + 3\eta^2) \qquad (\text{since } |m_i^{(t)}|, \nu, \eta \in [0,1]) \\
&\le \Phi^{(t)} - \eta\langle m^{(t)}, p^{(t)}\rangle + 3\eta^2,
\end{aligned}$$

where the final inequality follows from the fact that $\log(1 + x) \le x$ for all $x > -1$. Taking expectations on both sides of the above inequality over the randomness up to and including round t, we get

$$\mathbb{E}[\Phi^{(t+1)}] \le \mathbb{E}[\Phi^{(t)}] - \eta\,\mathbb{E}[\langle m^{(t)}, p^{(t)}\rangle] + 3\eta^2.$$

Thus, by induction, using the fact that $\Phi^{(1)} = \log(n)$, we have

$$\mathbb{E}[\Phi^{(T+1)}] \le \log(n) - \eta \sum_{t=1}^T \mathbb{E}[\langle m^{(t)}, p^{(t)}\rangle] + 3\eta^2 T.$$

On the other hand, for any given index i, we have

$$\Phi^{(T+1)} = \log\left(\textstyle\sum_{i'} \exp\left(\sum_{t=1}^{T} -\eta \hat{m}_{i'}^{(t)}\right)\right) \geq -\eta \sum_{t=1}^{T} \hat{m}_i^{(t)},$$

and hence

$$\mathbb{E}[\Phi^{(T+1)}] \geq -\eta\, \mathbb{E}[\textstyle\sum_{t=1}^{T} \hat{m}_i^{(t)}] = -\eta\, \mathbb{E}[\sum_{t=1}^{T} m_i^{(t)}].$$

Putting the above inequalities together, and simplifying, we get

$$\mathbb{E}\left[\sum_{t=1}^{T} \langle m^{(t)}, p^{(t)} \rangle\right] \leq \mathbb{E}\left[\sum_{t=1}^{T} m_i^{(t)}\right] + 3\eta T + \frac{\log n}{\eta}.$$

Using the value $\eta = \sqrt{\frac{\log n}{T}}$, we get the stated regret bound. $\square$