Differentially Private Densest Subgraph Detection

Dung Nguyen 1 2  Anil Vullikanti 1 2

Abstract
Densest subgraph detection is a fundamental graph mining problem, with a large number of applications. There has been a lot of work on efficient algorithms for finding the densest subgraph in massive networks. However, in many domains, the network is private, and returning a densest subgraph can reveal information about the network. Differential privacy is a powerful framework to handle such settings. We study the densest subgraph problem in the edge privacy model, in which the edges of the graph are private. We present the first sequential and parallel differentially private algorithms for this problem. We show that our algorithms have an additive approximation guarantee. We evaluate our algorithms on a large number of real-world networks, and observe a good privacy-accuracy tradeoff when the network has high density.

1. Introduction
Data privacy is a fundamental challenge in many real world applications, e.g., healthcare, social networks, and finance, where there is a risk of revealing private information through adversarial queries. Differential privacy (defined in Section 2), developed through the work of a number of researchers, e.g., (Dwork, 2011; Blum et al., 2005; Dwork et al., 2014; Nissim et al., 2007), has proven to be a very powerful approach to support diverse kinds of computations with rigorous privacy guarantees (see (Zhu et al., 2017; Vadhan, 2017) for extensive surveys on this topic). This framework allows database owners to support queries with a very controlled and rigorous loss of privacy. Differentially private algorithms have now been designed for a number of problems, including supervised and unsupervised machine learning, social network analysis and deep learning (Kasiviswanathan et al., 2013; Blocki et al., 2013; Abadi et al., 2016; Abowd, 2018). However, network problems have been proven to be much harder, and private algorithms for many problems (e.g., in graph mining) remain poorly understood. For some of the problems which have been studied, e.g., (Nguyen et al., 2016), good accuracy bounds are not known. Further, most private algorithms do not scale very well, especially for network problems, e.g., (Kasiviswanathan et al., 2013; Blocki et al., 2013), and there has been limited work on parallel algorithms.

Here, we study the problem of finding the densest subgraph in a graph $G(V, E)$, which is a very basic subroutine in graph mining, and has been used in diverse domains, including bioinformatics, network science, fraud detection, and social network analysis, e.g., (Cadena et al., 2018; Hooi et al., 2016; Khuller & Saha, 2009; Cadena et al., 2016; Tsourakakis et al., 2013; Rozenstein et al., 2014). There are many notions of density (Cadena et al., 2018); here, we focus on the notion of average density $\rho(S)$ of $S \subset V$, defined as $\rho(S) = |\text{edges with both end points in } S|/|S|$. The goal is to find a densest subgraph $S^* = \arg\max_{S \subseteq V} \rho(S)$; we use $\rho(G) = \rho(S^*)$ to indicate the density of the the densest subgraph in $G$. This notion of density is one of the most common studied ones, e.g., for anomaly detection in networked data (Cadena et al., 2018; Hooi et al., 2016), primarily because it can be computed very efficiently (Charikar, 2000; Asahiro et al., 2002; Khuller & Saha, 2009; Goldberg, 1984). In particular, the densest subgraph can be computed optimally using linear programming (Goldberg, 1984; Charikar, 2000), and a simple iterative greedy algorithm gives a $\frac{1}{2}$-approximation (Charikar, 2000; Asahiro et al., 2002). Highly scalable parallel algorithms have also been developed, e.g., (Bahmani et al., 2012; Ghaffari et al., 2019). However, private algorithms are not known for the dense subgraph problem, which is the focus of this paper.

There are two standard approaches for privacy in networks—edge privacy (in which the nodes are public and the edges are private) and node privacy (in which the nodes are also private). As we observe later, the maximum density value has low sensitivity can be computed easily using the Laplace mechanism. However, in graph mining settings, analysts are interested in finding the actual densest subgraph (i.e., the nodes in the subgraph), which is the focus of our paper, and is a much harder problem. It only makes sense in the edge privacy model. Our contributions are summarized below.

1Department of Computer Science, University of Virginia, Virginia, USA 2Biocomplexity Institute and Initiative, University of Virginia, Virginia, USA. Correspondence to: Dung Nguyen <dungn@virginia.edu>.

Proceedings of the 38th International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).
Differentially Private Densest Subgraph Detection

- We first present a sequential $(\epsilon, \delta)$-differentially private algorithm (SEQDENSEDP) for the densest subgraph problem in the edge privacy model. We adapt Charikar’s algorithm (Charikar, 2000), using the exponential mechanism (McSherry & Talwar, 2007; Gupta & Saha, 2009). The objective is to find a subset $S \subseteq V$, with high probability, that satisfies $\rho(S) \geq \rho(S^*)/2 - O(\log n)$, with high probability.

- We alter the sampling process in SEQDENSEDP and obtain a parallel algorithm (PARDENSEDP), which involves sampling multiple nodes in parallel, in each iteration. This works well in practice, in some regimes, but can have exponential time in the worst case. Our final algorithm (PHASEDENSEDP) groups multiple iterations into a single sampling phase (basically guessing when a node would be sampled). We prove that PHASEDENSEDP takes $O(\log n)$ phases, with high probability. Both PARDENSEDP and PHASEDENSEDP have similar accuracy as SEQDENSEDP, though the constant factors get worse. We also design a distributed version of PHASEDENSEDP in the Map-Reduce model (Dean & Ghemawat, 2008; Karloff et al., 2010; Beame et al., 2013). We show it has similar accuracy and runtime guarantees as PHASEDENSEDP.

- We evaluate our algorithms on a large number of diverse networks, in order to understand the privacy-accuracy tradeoffs. We find that in networks with high densities, our private algorithms have good accuracy, especially for larger values of $\epsilon$. Our algorithms also have reasonable recall, i.e., the private solution contains a large fraction of the optimal subgraph. SEQDENSEDP has better accuracy than PARDENSEDP, which, in turn, is better than PHASEDENSEDP. In some regimes of $\epsilon$, PARDENSEDP has significantly better accuracy than PHASEDENSEDP, and is comparable to SEQDENSEDP.

- We show an additive lower bound of $\Omega(\sqrt{\log n})$ for any differentially private algorithm for DENSEDP.

Due to the limited space, we have omitted many proofs and other discussion; these are presented in the Appendix. The full version including the Appendix is available at (Nguyen & Vullikanti, 2021). We note that our analysis is not very tight, in order to simplify the discussion; we expect the constant factors can be tightened with a more careful analysis.

We use the exponential mechanism (McSherry & Talwar, 2007) in an iterative manner, which involves choosing elements from a set with probability that is proportional to a score associated with the element. This approach has been used in private versions of other combinatorial optimization problems, e.g., vertex cover, global minimum cut and set cover, e.g., (Gupta et al.). However, this is inherently sequential, and we develop a parallel version, in which multiple elements can be chosen simultaneously. Consequently, the analysis of the performance becomes much more challenging, and is one of our contributions.

2. Preliminaries

2.1. Problem statement

Densest subgraph. Let $G = (V, E)$ denote an undirected graph. For a subset of nodes $S \subset V$, let $E[S]$ denote the set of edges with both end points in $S$. The density of a set $S$ in $G$ is defined as $\rho(S) = |E[S]|/|S|^2$ (Charikar, 2000; Khuller & Saha, 2009). The objective is to find a subset $S^*(G) = \arg\max_{S \subset V} \rho(S)$, which achieves the maximum density; we refer to it as $S^*$ when the graph is clear from the context. Let $\rho(G) = \rho(S^*)$. We say that $\rho(S)$ is an $\alpha$-approximate solution if $\rho(S) \geq \alpha \rho(S^*)$. An optimal densest subgraph (i.e., $\alpha = 1$) can be computed optimally using a linear programming based algorithm, and an iterative algorithm gives a $\alpha = 1/2$-approximation (Charikar, 2000; Khuller & Saha, 2009). However, this turns out to be difficult to achieve under privacy, and we consider a slightly relaxed version: we say $S$ is an $(\alpha, \beta)$-approximation if $\rho(S) \geq \alpha \rho(S^*(G)) - \beta$. Let $N_G(v)$ denote the set of neighbors of $v$ in $G$, and let $\deg_G(v) = |N_G(v)|$ denote the degree of $v$. For a subset $S \subset V$, let $\deg_S(v) = |N(v) \cap S|$ denote the number of neighbors of $v$ in $S$ in graph $G$; when $G$ is clear from the context, we denote it by $\deg(v)$.

Differential privacy on graphs. Let $G$ denote a set of graphs on a fixed set $V$ of nodes. For a graph $G \in G$, we use $V(G)$ and $E(G)$ to denote the set of nodes and edges of $G$, respectively. In this paper, we will focus on the notion of Edge privacy (Blocki et al., 2013), where all graphs $G \in G$ have a fixed set of nodes $V(G) = V$, and two graphs $G, G' \in G$ are considered neighbors, i.e., $G \sim G'$, if they differ in exactly one edge, i.e., $|E(G) - E(G')| = 1$. We note that another notion that has been considered is Node privacy (Kasiviswanathan et al., 2013): two graphs $G, G' \in G$ are considered neighbors, i.e., $G \sim G'$, if they differ in exactly one node, i.e., $|V(G) - V(G')| = 1$. Note that they might differ in many edges.

Definition 2.1. A (randomized) algorithm $M : G \rightarrow R$ is $(\epsilon, \delta)$-differentially private if for all subsets $S \subset R$ of its output space, and for all $G, G' \in G$, with $G \sim G'$, we have $Pr[M(G) \in S] \leq e^{\epsilon} Pr[M(G') \in S] + \delta$ (Dwork et al., 2014; Dwork, 2011; Vadhan, 2017; Blocki et al., 2013).

Problem statement: densest subgraph detection with edge differential privacy (DENSEDP). Given a family of
graphs $G$ on a set $V$ of vertices, and parameters $\epsilon, \delta$, construct an $(\epsilon, \delta)$-differentially private mechanism $M : G \rightarrow 2^V$, such that: (1) for any graphs $G \sim G'$, and $S \subset 2^V$,
\[
\Pr[M(G) \in S] \leq e^{\epsilon \epsilon} \Pr[M(G') \in S] + \delta, \text{ and } (2) \rho(M(G))
\]
maximized.

We evaluate the accuracy of the mechanism $M$ (in the second condition above) in terms of the $(\alpha, \beta)$-approximation to $\rho(S^*(G))$ for any graph $G \in G$; therefore, we would like $\alpha$ to be as large as possible, and $\beta$ to be as small as possible.

Computing density value vs finding a subgraph. The DENSEDP problem involves finding a subset $M(G) \subset V$, where $V$ is not private, but only the edges are private. It can be verified easily that the ‘sensitivity’ (Definition 2.2)
\[
\Delta_u = \max_{G \sim G'} |\rho(S^*(G)) - \rho(S^*(G'))| \leq 1, \text{ as the addition or removal of an edge can be shown to alter the density by at most 1. As a result, private computation of $\rho(S^*(G))$ can be done easily with the standard Laplacian mechanism $Lap(1)$ (Dwork et al., 2014). However, returning a subgraph with high density, and with privacy is a harder problem.

2.2. Additional background on differential privacy

Let $A_1 : D_1 \rightarrow R_1$ and $A_2 : D_2 \rightarrow R_2$ be $\epsilon_1, \delta_1$ and $(\epsilon_2, \delta_2)$-differentially private algorithms, respectively. Let $f : R_1 \rightarrow R_1'$ be an arbitrary randomized mapping. We will make extensive use of the following basic results shown in (Dwork et al., 2014), that differential privacy is preserved under:

(1) post-processing (Proposition 2.1 of (Dwork et al., 2014)), i.e., $f \circ A_1 : D \rightarrow R_1'$ is also $(\epsilon, \delta)$-differentially private, and

(2) composition (Theorem 3.16 of (Dwork et al., 2014)), i.e., $A : (D_1, D_2) \rightarrow (R_1, R_2)$ defined as $A(x) = (A_1(x_1), A_2(x_2))$ is $(\epsilon_1 + \epsilon_2, \delta_1 + \delta_2)$-differentially private.

Definition 2.2. Sensitivity of utility function (Dwork et al., 2014). Given a dataset $D$ and an arbitrary range $R$, a function $u : D \times R \rightarrow R$ has a global sensitivity $\Delta_u$ defined as:
\[
\Delta_u = \max_{r \in R} \max_{x \sim x'} |u(x, r) - u(x', r)|
\]

Definition 2.3. The Exponential Mechanism. (Definition 3.4 of (Dwork et al., 2014)) The exponential mechanism $M(x, u, R)$ that outputs an element $r \in R$ with probability proportional to $exp(\frac{u(x, r)}{\Delta_u})$ is $\epsilon$-differentially private, if the addition of an element to the data does not decrease the value of the utility function.

Theorem 2.4. Utility of the Exponential Mechanism. (Theorem 3.11 and Corollary 3.12 of (Dwork et al., 2014)) For a given dataset $x$, let $OPT = \max_{r \in R} u(x, r)$. For the exponential mechanism $M(\cdot)$, we have:
\[
\Pr[u(x, M(x, u, R))] \leq OPT - \frac{2\Delta_u}{\epsilon \epsilon} (\ln |R| + r) \leq e^{-r}
\]

Adversarial Probabilistic Process. Following (Gupta et al.), consider an adversarial probabilistic process with $n$ iterations. In each iteration $i$, the adversary tosses a coin, which gives heads with probability $p_i \in [0, 1]$, adaptively by observing previous $i-1$ iterations. We utilize this process to prove the privacy bounds of our algorithms.

Lemma 2.5. Lemma B.1 of (Gupta et al.). Let $Z_i$ be a random variable, with $Z_i = 1$ indicating that no head appears in the first $i$ iterations, and $Z_i = 0$ otherwise. Let $Y = \sum_{i=1}^{n} p_i Z_i$. For any $q$, $\Pr[Y > q] \leq e^{-q}$.

Corollary 2.5.1. Let $T$ be the first iteration in which a head appears in the process above. For any $\delta \in (0, 1]$, $\sum_{i=1}^{T-1} p_i \leq \ln \delta^{-1}$, with probability at least $1 - \delta$.

3. Related Works

Differential privacy has been a very active area of research since its introduction. We briefly summarize the main results and challenges; we focus only on works on privacy for graph that are directly relevant to our paper. Due to space constraints, we refer to (Dwork, 2011; Blum et al., 2005; Dwork et al., 2014; Nissim et al., 2007; Zhu et al., 2017; Vadhan, 2017) for surveys and details of key results in this area, and provide additional discussion in the Appendix.

Much of the initial work on graph privacy involved computing different kinds of statistics, e.g., number of edges, counts of triangles and other subgraphs, and the cost of the minimum spanning tree (Nissim et al., 2007; Karwa et al., 2014a,b; Mir & Wright, 2012; Hay et al., 2009; Kasiviswanathan et al., 2013; Blocki et al., 2013). Graph statistics are challenging due to their high sensitivity. (Nissim et al., 2007) introduce a concept of smooth sensitivity and apply it for triangle counting and minimum spanning tree problems in the edge privacy model. (Karwa et al., 2014a) generalize the method for other subgraph counting problems, such as k-star or k-triangle. Many other graph statistics in the edge-privacy model have been studied in (Karwa et al., 2014b; Mir & Wright, 2012; Hay et al., 2009). On the other hand, graph statistics in the node-privacy model are generally more difficult. (Kasiviswanathan et al., 2013; Blocki et al., 2013) are two of the first studies that utilize Lipschitz Extensions to develop node differentially private mechanisms. They transform subgraph counting problems into linear programming optimizations of which the sensitivities of the solutions are restricted. Algorithms for node differentially private graph queries such as degree distributions or random graph model estimations have been established by (Chen & Zhou, 2013; Ding et al., 2018; Day et al., 2016).

In contrast to graph statistics, releasing a subset of nodes in the input graph with differential privacy (as in the st-mincut, k-median, or the vertex cover problems), or a collection of subsets in other combinatorial optimization problems (as in the set cover problem) is much more challenging, e.g., (Mitrovic et al., 2017; Gupta et al.). One of the first
papers on this was by (Gupta et al.), who design private algorithms for different combinatorial optimization problems, including computing global mincuts, vertex cover and set cover. Our algorithms are inspired by the techniques of (Gupta et al.).

Another direction of work is on generation of differentially private synthetic graphs (Xiao et al., 2014; Nguyen et al., 2015; Chen et al., 2014; Gupta et al., 2012), which preserve some specific graph properties, such as cut, shortest path length and degree distribution queries. Such synthetic graphs are useful since post-processing does not cause any privacy violation. Finally, we note that private graph algorithms have also been considered for other notions of privacy, e.g. (Imola et al., 2020).

4. Private Densest Subgraph

Here we design and analyze SEQDENSEDP for the DENSEDP problem. This is an adaptation of the (non private) algorithm Charikar (Charikar, 2000), which gives a 1/2-approximation for the densest subgraph problem—this algorithm constructs a sequence of subgraphs $S_1, \ldots, S_n$ (by removing a minimum degree node each time), and then chooses $\arg\max_{S_i} \mu(S_i)$. SEQDENSEDP adapts this algorithm, and involves two parts. First, it generates a sequence of candidate subgraphs by removing one node at a time using an Exponential Mechanism, with probability proportional to an exponential function of its current degree times a constant. Therefore, lower-degree nodes have exponentially more chances to be removed than higher-degree nodes. Intuitively, we try to remove low-degree nodes and retain high-degree nodes to construct dense candidates. The second part of the algorithm applies the Exponential Mechanism to randomly select one of the candidates by their densities.

**Algorithm 1** SEQDENSEDP($G, \epsilon, \delta$)

**Input:** $G \in \mathcal{G}$, privacy parameters $\epsilon, \delta$.  
**Output:** $S' \subseteq V(G)$

1. Let $\epsilon' = \frac{\epsilon}{\ln(\frac{\epsilon}{\delta})}, S_0 = V(G)$
2. for $t := 1$ to $n$ do
3. Pick a node $\pi_t \in \{v : v \in S_{t-1}\}$ with probability proportional to $e^{-\epsilon' \deg_{S_{t-1}}(v)}$
4. $S_t = S_{t-1} - \pi_t$
5. end for
6. Return $S_t$ from $\{S_t : t = 0, \ldots, n - 1\}$ with probability proportional to $e^{\epsilon' \mu(S_t)}$

**Lemma 4.1.** The computation of the sequence $\pi^G = \pi_1, \ldots, \pi_n$, in the for loop in Algorithm 1 (lines 3–5) is ($\epsilon/2, \delta$)-differentially private.

**Proof.** (Short, Full proof in Theorem A.1) Let $G \sim G'$ be two graphs that differ in exactly one edge $e = (u, v)$. Let $\pi^G$ and $\pi^{G'}$ denote the permutations computed in the for loop in Algorithm 1. Let $T$ is the first iteration that one of the endpoints of $e$ (i.e., $u$ or $v$) is removed. Let $adj(e)$ define the set of nodes adjacent to edge $e$. Recall that $\deg_{G}(j)$ is the degree of node $j$ in subgraph $S_t$ of graph $G$. Fix a permutation $\pi$. Consider the ratio:

$$P = \frac{\Pr[\pi^G = \pi]}{\Pr[\pi^{G'} = \pi]} = \prod_{t=1}^{n} \frac{\exp(-\epsilon' \deg_{S_t}(\pi_t))}{\sum_{j \in adj(e)} \exp(-\epsilon' \deg_{S_t}(j))}$$

The last equality follows from the definition of $T$, because: (i) for $t = T + 1, n$, $\deg_{G}(j) = \deg_{G'}(j)$ for all nodes $j$, and, (ii) for $t = 1, T$, $\deg_{G}(\pi_t) = \deg_{G'}(\pi_t)$. We have two cases for the analysis.

**First**, suppose $G + e = G'$. In this case, $\frac{\exp(-\epsilon' \deg_{S_t}(\pi_t))}{\exp(-\epsilon' \deg_{S_t}(\pi_t))} = \exp(\epsilon')$ because $\deg_{S_t}(\pi_t) + 1 = \deg_{S_t}(\pi_t)$. Further, for each $t$, we have $\deg_{G}(j) \geq \deg_{G'}(j)$ for all $j$, which implies the product $\prod_{t=1}^{T} \frac{\sum_{j \in adj(e)} \exp(-\epsilon' \deg_{S_t}(j))}{\exp(-\epsilon' \deg_{S_t}(j))} \leq 1$. Therefore, $P \leq \exp(\epsilon')$

**Second**, suppose $G - e = G'$. Similar to the first case, $\frac{\exp(-\epsilon' \deg_{S_t}(\pi_t))}{\exp(-\epsilon' \deg_{S_t}(\pi_t))} = \exp(-\epsilon') \leq 1$. For $j \in adj(e), t \leq T$, we have $\deg_{G}(j) = \deg_{G'}(j) + 1$. For $j \notin adj(e)$, and for all $t$, $\deg_{G}(j) = \deg_{G'}(j)$. Further, for $t > T$, we have $\deg_{G'}(j) = \deg_{G'}(j)$ for all $j$. Note that whenever we use $\deg_{G}(j)$, we only consider $j \in S_{T-1}$, i.e., $j$ would not have been deleted from the graph before step $t$.

Therefore, for $t \leq T$: we can expand the term $\sum_{j \in adj(e)} \exp(-\epsilon' \deg_{S_t}(j))$ as $\sum_{j} \exp(-\epsilon' \deg_{S_t}(j)) = (\epsilon' + 1) \sum_{j} \exp(-\epsilon' \deg_{G}(j)) + \sum_{j} \exp(-\epsilon' \deg_{G'}(j))$.

Substituting it in the expression for $P$. Since $\deg_{S_t}(j) = \deg_{G'}(j)$ for all $j$ and $t > T$, we have

$$P \leq \prod_{t=1}^{T} (1 + \exp(\epsilon') - 1) \frac{\sum_{j \in adj(e)} \exp(-\epsilon' \deg_{G}(j))}{\sum_{j \in adj(e)} \exp(-\epsilon' \deg_{G'}(j))}$$

$$= \prod_{t=1}^{T} (1 + \exp(\epsilon') - 1)p_t(G)$$

$$\leq \prod_{t=1}^{T} \exp(\exp(\epsilon') - 1)p_t(G),$$

using $1 + x \leq e^x$ for $x \geq 0,$ where $p_t(G)$ is the probability that an endpoint of $e$ is picked in iteration $t$. We will show in Lemma 4.2, by using the adversarial process (Section 2.2), that $\sum_{t=1}^{T-1} p_t(G) \leq \ln \delta^{-1}$ with prob. at least $1 - \delta$ and use it to derive a bound on $P$. We say that $\pi$ is in $\delta^{-1}$-good if $\sum_{t=1}^{T-1} p_t(G) \leq \ln \delta^{-1}$ and in $\delta^{-1}$-bad otherwise (good and bad for short). Therefore,
with probability at least $1 - \delta$, $\pi$ is good and
\[
P \leq \prod_{t=1}^{T} \exp((\exp(\epsilon') - 1)p_t(G))
\leq \exp(2\epsilon' \sum_{t=1}^{T} p_t(G)), \text{ using } \epsilon' \leq 1 + 2x \text{ for } x \leq 1
\leq \exp(2\epsilon'(\ln\epsilon^- + pr(G))) \leq \exp(2\epsilon'(\ln\delta^{-1} + 1))
= \exp(2\epsilon'(\ln\epsilon/\delta)) = \exp(\epsilon/2), \text{ since } \epsilon' = \frac{\ln(\epsilon/\delta)}{\ln(\epsilon/\delta)}.
\]
which implies $\Pr[\pi^G = \pi] \leq \exp(\epsilon/2) Pr[\pi^{G'} = \pi]$.

Let $P$ be the set of output orderings that might be generated by the for loop in SEQDENSEDP. Then, we have
\[
\Pr[\pi^G \in P] = \sum_{\pi \in P} \Pr[\pi^G = \pi]
= \sum_{\pi \in P: \pi \text{ is good}} \Pr[\pi^G = \pi] + \sum_{\pi \in P: \pi \text{ is bad}} \Pr[\pi^G = \pi]
\leq \exp(\epsilon/2) Pr[\pi^{G'} \in P] + \delta.
\]
The lemma then follows.

Lemma 4.2. Let $p_t(G)$ is the probability that an endpoint of edge $e$ is picked in iteration $t$ and $T$ is the first iteration that an endpoint of edge $e$ is picked (as stated in Lemma 4.1), $\sum_{t=1}^{T-1} p_t(G) \leq \ln \delta^{-1}$ with probability at least $1 - \delta$.

Proof. We map our algorithm to the adversarial probabilistic process in Lemma 2.5 and Corollary 2.5.1. In each iteration $t$, the adversary chooses heads with probability equal to $p_t(G)$. Since $p_t(G)$ is calculated based on the outcomes of $t - 1$ previous iterations, it satisfies the conditions of the adversarial model. In any iteration $t$, if no head has appeared in the previous $t - 1$ iterations, we have $p_t(G) = \frac{\exp(-\epsilon' \deg_S^- (j))}{\sum \exp(-\epsilon' \deg_S^- (j))}$, since each node $j$ is picked with probability proportional to $\exp(-\epsilon' \deg_S^- (j))$. Therefore, the two processes (i.e., our algorithm and the adversarial probabilistic process) are now equivalent. By Corollary 2.5.1, $\sum_{t=1}^{T-1} p_t(G) \leq \ln \delta^{-1}$ with probability at least $1 - \delta$. The lemma then follows.

Theorem 4.3. Algorithm 1 is ($\epsilon, \delta$)-differentially private.

Proof. By Lemma 4.1, the sequence $\{S_1, S_2, \ldots\}$ is ($\epsilon/2, \delta$)-differentially private. For each $S_t$, adding or removing 1 edge to/from the graph changes $\rho(S_t)$ by at most 1, hence $\Delta_{\rho} = 1$. Applying the Exponential Mechanism, the last command releases an output with $\epsilon/2$-differential privacy. Using the Composition Theorem, Algorithm 1 is ($\epsilon, \delta$)-differentially private.

Lemma 4.4. In each iteration $t$, the node $v_t$ picked by the exponential mechanism satisfies $\deg_{S_t}(v_t) \leq 2\rho(S_t) + \frac{8}{\ln(1/\delta)} \ln(\epsilon/\delta) \ln(n)$, with probability at least $1 - 1/n^2$.

Proof. Since $2\rho(S_t) = \frac{2|\text{edges in } S_t|}{|S_t|} = \frac{\sum_{v \in S_t} \deg_{S_t}(v)}{|S_t|}$, it follows that $\min_{v \in S_t} \deg_{S_t}(v) \leq 2\rho(S_t)$. Let $\nu_{\min}$ be a node achieving the minimum degree in $S_t$. Then $\deg_{S_t}(\nu_{\min}) \leq 2\rho(S_t)$.

In the exponential mechanism run in iteration $t$, the sensitivity $\Delta = 1$, as $|\deg_{S_t}^- (v) - \deg_{S_t}^- (v')| \leq 1$. Applying Theorem 2.4 with $u(S_t, v) = -\deg_{S_t}^- (v)$ and $r = 2 \ln n$, it follows that $\Pr[\deg_{S_t}(v_t) \geq \deg_{S_t}(\nu_{\min}) + \frac{4}{\ln(1/\delta)} \ln(n)] \leq e^{-2\ln n} \leq 1/n^2$. Substituting $\epsilon' = \frac{\epsilon}{\ln(\epsilon/\delta)}$, the Lemma follows.

Theorem 4.5. Let $S^*$ be an optimal solution. Assuming $n > 3, \delta < 1/e$, the set $S$ output by Algorithm 1 satisfies $\rho(S) \geq \frac{\rho(S^*)}{2} - \frac{8}{\ln(1/\delta)} \ln(\epsilon/\delta) \ln(n)$, with probability at least $1 - 1/n$.

Proof. Applying a union bound with Lemma 4.4, it follows that for each iteration $t$, $\deg_{S_t}(v_t) \leq 2\rho(S_t) + \frac{24}{\ln(1/\delta)} \ln(\epsilon/\delta) \ln(n)$, with probability at least $1 - 1/n$. Let $T$ denote the first iteration that a node $v_T \in S^*$ is picked. As observed in (Bahmani et al., 2012), $\deg_{S_T}(v) \geq \rho(S^*)$, therefore
\[
\rho(S^*) \leq \deg_{S_T}(v) \leq \deg_{S_T}(v) \leq 2\rho(S_T) + \frac{24}{\ln(1/\delta)} \ln(\epsilon/\delta) \ln(n),
\]
with probability at least $1 - 1/n$. Therefore, $\rho(S_T) \geq \rho(S^*)/2 - \frac{12}{\ln(1/\delta)} \ln(\epsilon/\delta) \ln(n)$, with prob. at least $1 - 1/n$. Next, applying Theorem 2.4 to the exponential mechanism in the last line of the algorithm, we have $\Pr[\rho(S) \leq \rho(S_T) - \frac{2\ln n + r}{\epsilon}] \leq e^{-r}$. Choosing $r = \ln n$, we have $\rho(S) \geq \rho(S_T) - 8 \ln n/e$, with probability at least $1 - 1/n$. Combining this with the bound on $\rho(S_T)$, we have $\rho(S) \geq \rho(S^*)/2 - \frac{32}{\ln(1/\delta)} \ln(\epsilon/\delta) \ln(n) - 8 \ln n/e$.

Finally, we derive a coarse but simpler bound. For $\delta \leq 1/e$, we have $1 \leq \epsilon/\delta \leq 2 \ln(1/\delta)$. Using these in the lower bound for $\rho(S)$ gives $\rho(S) \geq \frac{\rho(S^*)}{2} - (24 + \frac{8}{\ln(1/\delta)}) \frac{1}{2} \ln(1/\delta) \ln(n) \geq \frac{\rho(S^*)}{2} - \frac{32}{\ln(1/\delta)} \ln(\epsilon/\delta) \ln(n)$.

5. Parallel Private Densest Subgraph

5.1. Algorithm PARDENSEDP

Instead of picking a single node in each iteration, as in Algorithm SEQDENSEDP, the main idea in PARDENSEDP is to independently sample each node with the right probability. As a result, multiple nodes might be selected simultaneously, especially when the degrees are low. This change makes the privacy analysis more challenging because (1) we have to consider different scenarios of picking the endpoints of the extra edge $e$ and (2) without the normalization terms, the probability $P$ in the proof of Lemma 4.1 loses its symmetry.

Theorem 5.1. Algorithm 2 is ($\epsilon, \delta$)-differentially private.
Algorithm 2 ParDenseDP($G, \epsilon, \delta$)

Input: $G \in \mathcal{G}$, privacy parameters $\epsilon, \delta$.

Output: A subset $S \subseteq V(G)$.

1: Let $\epsilon' = \frac{1 - 1/e}{8 \ln(e/\delta)} \epsilon$, and $c = 1/\epsilon' + 1$
2: $S_0 = V(G), t = 0$
3: while $|S_t| > 0$ do
4: $\forall j \in S_t$, assign a random variable $p_{ij} \in \{0, 1\}$ such that $Pr[p_{ij} = 1] = P_{ij} = \exp(-\epsilon'(\deg_{S_t}(j) + c))$
5: $t := t + 1$
6: $\pi_t = \{j \in S_t : p_{ij} = 1\}$
7: $S_t = S_{t-1} \setminus \pi_t$
8: end while
9: Return $S_t$ from distinct set $\{S_t\}$ with probability proportional to $e^{\epsilon|S_t|}/2$

Proof. (Sketch, full proof in Theorem A.4) Using the same notations in Lemma 4.1, we prove the sequence $\{S_1, S_2, \ldots\}$ output by the while loop is $(\epsilon/2, \delta)$-differentially private. The remaining part is the same to Theorem 4.3. We construct $P = \frac{P_{ij}(\pi \cap \pi')}{P_{ij}(\pi)}$, and define:

$A_t = \prod_{j \neq v, w} P_{ij} G - B_t = \prod_{j \neq w} 1 - P_{ij}^{G - 1} P_{ij}^{G'}$, $C = \prod_{j \neq v, w} P_{ij} D = \prod_{j \neq w} 1 - P_{ij}^{G'}$,

then expand $P = \prod_{i=1}^{n} (A_t \times B_t) \times C \times D$.

We consider 2 cases (1) $G + e = G'$ and (2) $G - e = G'$ and 2 subcases in each case: (a) both $u, v$ are picked at step $T$ and (b) exact one of them is picked at $T$. In Case 1, it is straightforward to prove $P \leq \exp(2\epsilon')$, where Case 2 is more difficult to analyze.

In Case 2, we prove $A_t = 1 \forall t < T$. We reduce $C$ and $D$, and depends on the subcases, they have slightly different forms. We then reduce $P$ to $P = \prod_{i=1}^{n} (B_i \times B_i)$ and let $P \leq \prod_{i=1}^{n} (B_i \times B_i)$.

Theorem 5.2. (Proof in Theorem A.5) Let $S^*$ be an optimal solution. If $\delta \leq 1/e$, the set $S$ output by Algorithm 2 satisfies $\rho(S) \geq \frac{\rho(S^*)}{2} - \frac{\sqrt{n}}{e} \ln(1/\delta) \ln n$, with probability at least $1 - 2/n$.

Running time of ParDenseDP. Each node is removed in an iteration with probability $e^{-c \deg_{S_{t-1}}(e)}$. This could be as small as $e^{-\Theta(n)}$, e.g., if $S_{t-1}$ has a dense subgraph in which each node has very high degree. Therefore, the number of iterations could be $e^{O(n)}$ in the worst case (this is borne out of our experiments on some networks).

5.2. Algorithm PhaseDenseDP: Logarithmic Bounded Runtime Algorithm

Algorithm 2 can execute in parallel but it has exponential runtime. This section presents a modified version of ParDenseDP that has logarithmic bounded runtime, named PhaseDenseDP. We group iterations into phases and aim to remove at least a constant fraction of nodes in each phase (Lemma 5.3). Also, we only update node’s new degrees at the end of each phase, allowing the sampling process per each node to run independently as its degree remains unchanged in the current phase. The privacy analysis of PhaseDenseDP differs from ParDenseDP in two key ways: (1) the terms corresponding to the endpoints of the extra edge $e$ are not updated until the end of the current phase and (2) it requires two instances of the Adversarial Probabilistic Process to mimic the sampling process.

Algorithm 3 PhaseDenseDP($G, \epsilon, \delta$)

Input: $G \in \mathcal{G}$, privacy parameters $\epsilon, \delta$.

Output: A subset $S \subseteq V(G)$.

1: Let $\epsilon' = \frac{1 - 1/e}{24 \ln(4/\delta)} \epsilon$, and $c = 1/\epsilon' + 1$
2: Let $\pi_0 = V, S_0 = V, T_0 = 0$, $i = 1$
3: while $S_i \neq \emptyset$ do
4: if $|S_i| \leq \ln n$ then
5: $T_i = T_{i-1} + 1$, $\pi_i = S_i$, $S_{i+1} = \emptyset$, $i = i + 1$
6: else
7: For each $v \in S_i$, let $d(v) = deg_{S_i}(v)$
8: Let $P_v = \exp(-\epsilon'(d(v) + c))$
9: Let $T_v$ be chosen from a geometric random process with probability $P_v$
10: Let $\rho(S_i) = \rho(S_i) + \frac{16}{n} \ln n + \text{Lap}(\frac{4 \ln n}{\sqrt{T_i}})$
11: Let $T_i = \exp(4\rho(S_i) + c) \cdot 4 \ln n$
12: For $t \leq T_i$, let $\pi_t = \{v : T_v = t\}$
13: $S_{i+1} = S_i \setminus \cup_{t=T_{i-1}+1}^{T_i} \pi_t$
14: $i = i + 1$
15: end if
16: end while
17: Return $S \subseteq V(G)$ with probability proportional to $e^{\epsilon|S|}/2$

Lemma 5.3. The number of phases is at most $\log n$, with probability at least $1 - \frac{1}{n^2}$.

Proof. Consider any phase $i$, except the last one, and let
In Case 2b, similar to Theorem 5.1, we prove $B \in S$ follows.

Theorem A.6. Let $A(S_i) = \bigcup_{t=t_i-1}^{t_i} \pi_i$ denote the set of nodes removed in this phase.

Therefore, with probability at least $1 - 1/n^3$, every node $v \in S_{i+1} = S_i - A(S_i)$ has $d_{S_i}(v) > 4\rho(S_i)$. We have $2|E(S_i)| = \sum_{v \in A(S_i)} d_{S_i}(v) + \sum_{v \in S_{i+1}} d_{S_i}(v) \geq |S_{i+1}|4\rho(S_i) = 4|S_{i+1}|\frac{|E(S_i)|}{|S_i|}$, which implies $|S_{i+1}| \leq |S_i|/2$. Thus, with probability at least $1 - 1/n^3$, the number of nodes in each phase reduces by factor of 2. Therefore, with probability at least $1 - \frac{\log n}{n} \geq 1 - \frac{1}{n^7}$, the Lemma follows. ■

Theorem 5.4. Algorithm 3 preserves $(\epsilon, \delta)$-differential privacy, with $\epsilon \in (0, 1], \delta \in (2n^{-2}, 1]$.

Proof. (Sketch; full proof in Theorem A.10) Similar to Theorem 5.1, we prove the sequence $\{S_1, S_2, \ldots\}$ output by the while loop is $(\epsilon/2, \delta)$-differentially private.

First, prove the sequence $\{\pi_0, \pi_1, \ldots\}$ is $(\epsilon/4, \delta/2)$-differentially private. Using the same notations and expansion, we re-analyze the same cases in Theorem 5.1. Analysis in Case 1a and 2a (where $u$ and $v$ are removed at the same iteration) remains unchanged. We re-evaluate Case 1b and 2b. Assume $u$ is removed first at $T_u$ and $v$ is removed later at $T = T_u > T_v$.

In Case 1b, we prove that $A_t = 1 \forall t \neq T_u$ and $A_{T_u} = \exp(\epsilon'), C$ and $D$ are the same as in Theorem 5.1. We split the product of $B_t$ into two ranges $B_t = \frac{1}{1-P_{t_i}^{\rho(S_i)}} \frac{1}{1-P_{t_i-1}^{\rho(S_i)}}$ for $t \in [1, T_u - 1]$ and $B_t = 1 - P_{t_i}^{\rho(S_i)}$ for $t \in [T_u, T_v - 1]$. In either case, $B_t \leq 1$. We have $P = \prod_{t=1}^{T_u-1} (A_t \times B_t) \times C \times D \leq \exp(2\epsilon')$.

In Case 2b, similar to Theorem 5.1, we prove $A_t \leq 1$, $C \leq 1$, $D = 1$, and split the product of $B_t$ into two parts, where the latter arises since we do not update the degree of $v$ after removing $u$.

Theorem 5.5. (Proof in Theorem A.11) Let $S^*$ be an optimal solution. With probability at least $1 - 2/n$, the set $S$ output by Algorithm 3 satisfies $\rho(S) \geq \frac{\rho(S^*)}{4} - \frac{3\rho(S^*)}{2\sqrt{n\log 1/e}}$.

6. Map-Reduce Implementation

We briefly describe MRDENSEDP (Algorithm 4)—a Map-Reduce implementation of PHASEDENSEDP. We prove in Lemma A.13 that MRDENSEDP preserves the same privacy guarantee and runtime as of PHASEDENSEDP.

Each iteration $i$ of MRDENSEDP contains 3 Reduce sub-phases and 1 Map sub-phase and is equivalent to phase $i$ of PHASEDENSEDP. The first Reduce subphase takes input as pairs of nodes that constitute an edge and group them by endpoints. It creates a list of neighbors of each node to keep track of current nodes and calculate their degrees. The second Reduce subphase groups the current nodes together to form a candidate set $S_i$ while the third one gathers the nodes’ degrees to calculate the density $S_i$. The Map subphase performs the sampling process. It samples the removal time of each node and compares to the cut-off time of the current phase to decide if the node is picked or not. If the node is picked, it emits the node’s self-edge to signal the next iteration’s first Reduce subphase that the node is removed. Otherwise, it sends the edges adjacent to the node for the next iteration to rebuild the graph.

7. Lower bound

Theorem 7.1. Any $(\epsilon, \delta)$-differentially private algorithm for DENSEDP must incur an additive lower bound of $\Omega(\sqrt{\log n/\epsilon})$ for $\delta \leq 1/n$.

Proof. Let $G = (V, E)$ denote a graph with $|V| = n$ and $E = \emptyset$. Let $M(G)$ be an $(\epsilon, \delta)$-differentially private mechanism. Let $V$ be partitioned into sets $V_1, \ldots, V_N$, for $N = n/a$, $\binom{N}{2} = \frac{1}{2} \log n$, with each $|V_i| = a$. For $i \in \{1, \ldots, N\}$, define $S_i = \{S \subseteq V : S \cap V_i \neq \emptyset, |S| \leq a^2\}$. We prove in Lemma A.12 that there exists an $i \in \{1, \ldots, N\}$ such that $Pr[M(G) \in S_i] \leq 2a^n/n$.

Next, let $G' = (V, E')$ be a graph such that $V_i$ is a clique in $E'$, and there are no additional edges. Then, $|E'| = \binom{N}{2} = \frac{1}{2} \log n$. By the group privacy property of $(\epsilon, \delta)$-differentially private (Lemma 2.2 of Vadhan, 2017), it follows that $Pr[M(G') \in S_i] \leq Pr[M(G) \in S_i] \exp\left(\frac{1}{3} \log n\right) + \frac{1}{3} \log n \exp\left(\frac{1}{3} \log n\right) \delta \leq \frac{2a^n}{n^{1/3}} + \frac{\log n}{3^{1/3}} \frac{1}{n^{1/3}} \leq \frac{2a^n}{n^{1/3}}$.
For any \( S \not\subseteq S_i \), we have \( \rho(S) \leq 1/2 \), since either (1) 
\[ |S| > a^2, \] which implies \( \rho(S) \leq \frac{|S|}{2} \leq \frac{a}{2}, \] since \( S \) may have up to \( \binom{n}{a} \) edges, or (2) \( |S| \leq a^2, S \cap V_i = \emptyset \), which implies \( \rho(S) = 0 \). If \( M(G') \in S_i \), we have \( \rho(M(G')) \leq a/2 \). Hence, 
\[ \mathbb{E}[ho(M(G'))] \leq \frac{a}{2} \cdot \Pr[M(G') \in S_i] + \frac{1}{2} \cdot (1 - \Pr[M(G') \in S_i]) \leq \frac{3a^3}{2a^{2n}} + \frac{1}{2} \leq 1, \] and the theorem follows since \( \rho(G') = \frac{a^3}{2^n} \).

\[ \square \]

8. Experiments

Our experiments address the following questions.

Accuracy of our proposed methods. How does the accuracy of the solution computed using our private algorithms vary with \( \epsilon \) and \( \delta \) in different graphs?

Recall. To what extent can our private algorithms find the densest subgraph, as quantified by the recall?

Efficiency of the parallel algorithms. How does the number of iterations vary, relative to \textsc{SeqDenseDP}?

8.1. Baselines and measurements

We use the algorithm of [Charikar, 2000] as a baseline to compute a non-private densest solution for performance evaluation in our experiments. Let \( S_b \) denote the subgraph output by the baseline algorithm, and let \( M(G) \) denote the solution computed by our private algorithm \( M \) on graph \( G \) (for fixed \( \epsilon, \delta \)). We consider the following metrics for evaluating our algorithms. For all these metrics, the closer they are to 1, the better our algorithms perform, in comparison to the baseline.

- **Relative density**: this is defined as \( \rho(M(G))/\rho(S_b) \).
- **Jaccard index**: \( |M(G) \cap S_b|/|M(G) \cup S_b| \), which captures similarity between \( M(G) \) and \( S_b \).
- **Recall**: \( |M(G) \cap S_b|/|S_b| \), which quantifies what fraction of nodes in \( S_b \) are selected by \( M(G) \).

Table 1 lists 6 different networks from SNAP database we use to evaluate our results—these are chosen to be of various sizes, in order to understand the impact of network structure on the results [Leskovec & Krevl, 2014]. Due to the space limit, we present the setup and experiments on an extended list of 20 networks (Table 2) in Section B.1. Also, we only show results for \textsc{SeqDenseDP} and \textsc{ParDenseDP} here; see the Appendix for a detailed comparison with \textsc{PhaseDenseDP}.

8.2. Experimental Results

Utility analysis of private Algorithms. Figure 1 shows the densities of private subgraphs in relative to the densities of the baselines. In most networks which have average degree larger than 4, we observe that \textsc{SeqDenseDP} has relative density \( \geq 75\% \) for \( \epsilon = 2 \) or 4. For \( \epsilon = 2 \), \textsc{SeqDenseDP} has the same density as the non-private baseline in four of six networks and has at least 50\% of the density of the baseline in all except one. Figure 5 shows the relationship between the relative density of the solution computed by our private mechanism and the network density (i.e., the average degree) for the twenty networks we study. It confirms that, in general, the accuracy is higher in higher density graphs. In general, we observe the trade-off between accuracy and privacy in all measurements. Higher \( \epsilon \), which means less privacy guarantee, yields better accuracy. In contrast, Figures 1, 2, 3 show that \( \delta \) does not have significant impacts on the quality of solutions. We show in the Appendix that, in general, \textsc{ParDenseDP} outperforms \textsc{PhaseDenseDP} at \( \epsilon \geq 2 \). For small \( \epsilon \), neither algorithm has good accuracy, though \textsc{PhaseDenseDP} is slightly better.

Jaccard index and recall. Figure 2 shows the Jaccard similarity between the private subgraphs and \( S_b \). The coefficients vary greatly across networks. Four out of the six networks have Jaccard similarity coefficients at least 0.5 for \( \epsilon \geq 2 \) with \textsc{SeqDenseDP}. As with the relative density, networks with higher density tend to have better Jaccard similarity coefficients. Figure 3 shows the recall of our algorithms, i.e., the fraction of nodes in \( S_b \) which are selected by our private algorithms. The recall for \textsc{SeqDenseDP} is at least 75\% in all networks for \( \epsilon \geq 1 \), which indicates that the algorithm successfully finds most nodes in \( S_b \).

---

**Table 1. Summary of 6 of the 20 networks used for the experiments (Leskovec & Krevl, 2014); the remaining networks are summarized in Table 2 of the Appendix.**

<table>
<thead>
<tr>
<th>Network</th>
<th>Nodes</th>
<th>Edges</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ca-GrQc</td>
<td>5242</td>
<td>14496</td>
<td>Collab. net. Arxiv General Rel.</td>
</tr>
<tr>
<td>muse_DE</td>
<td>9498</td>
<td>153138</td>
<td>Social net. of Twitch (DE)</td>
</tr>
<tr>
<td>musae_ENGB</td>
<td>7126</td>
<td>35324</td>
<td>Social net. of Twitch (ENGB)</td>
</tr>
<tr>
<td>ca-AstroPh</td>
<td>18772</td>
<td>198110</td>
<td>Collab. net. Arxiv Astro Phy.</td>
</tr>
<tr>
<td>muse_squirrel</td>
<td>5201</td>
<td>198493</td>
<td>Wiki page-page net. (squirrel)</td>
</tr>
<tr>
<td>facebook</td>
<td>4039</td>
<td>88234</td>
<td>Social circles from Facebook</td>
</tr>
</tbody>
</table>

**Figure 1. Accuracy in term of relative density of our private algorithms.** The number in each graph indicates the density of the graph as whole (which equals the average degree).
Differentially Private Densest Subgraph Detection

Figure 2. Jaccard similarity coefficient of private subgraphs and non-private baselines.

Figure 3. Recall: Fraction of nodes of the baseline’s subgraphs are included in our algorithm’s outputs.

Performance of parallel algorithms. Figure 4 shows the ratio of the number of iterations taken by PARDENSEDP and SEQDENSEDP (recall that the number of iterations for SEQDENSEDP is $n$). Due to the nature of the sampling probabilities in PARDENSEDP, the probability of removing a node reduces as $\epsilon$ increases. Consequently, PARDENSEDP takes more iterations to remove all nodes with larger $\epsilon$ and smaller $\delta$. In all but the musae_squirrel network (Figure 4), the number of iterations for PARDENSEDP is about 1% of that for SEQDENSEDP, when $\epsilon \leq 8$. Hence in most cases, PARDENSEDP is much more efficient.

9. Conclusions

In this paper, we design the first sequential and parallel differentially private algorithms for the densest subgraph problem in the edge-privacy model. All of them give $(1/2, O(\log n))$-approximate solutions, with high probability. In other words, they match the $1/2$ approximation of (Charikar, 2000), with an additive approximation of $O(\log n)$; we also prove a lower bound on the additive term of $\Omega(\sqrt{\log n})$. Our main technical contributions include adaptation of the exponential mechanism to be applied in parallel, and the analysis of privacy and accuracy. Our experiments on 20 networks show that our algorithms have good accuracy in high-density networks and reasonable recall, overall, when $\epsilon \geq 2$. SEQDENSEDP has better accuracy while the parallel variants reduce the number of iterations significantly in most settings. Our experiments suggest that PARDENSEDP is good enough for practical uses among two parallel variants. Our paper leads to many open questions. Can we improve the runtime, e.g., by a combination of PARDENSEDP and PHASEDENSEDP? The most intriguing problem is to close the gap between the upper and lower bounds in accuracy. In particular, is an $O(\sqrt{\log n})$ additive approximation to (Charikar, 2000) possible to achieve? Is it possible to obtain a $(1, O(\log n))$-approximate solution, i.e., a purely additive approximation?

Acknowledgement. We thank anonymous ICML reviewers for helpful suggestions. A. Vullikanti is supported by NSF grants IIS-1931628, CCF-1918656, NSF IIS-1955797, NIH grant R01GM109718.
References


Differentially Private Densest Subgraph Detection


Differentially Private Densest Subgraph Detection


