Differentially Private Algorithms for Graphs Under Continual Observation

Hendrik Fichtenberger ✉ 🄳
University of Vienna, Austria

Monika Henzinger ✉
University of Vienna, Austria

Wolfgang Ost ✉
University of Vienna, Austria

──── **Abstract** ────

Differentially private algorithms protect individuals in data analysis scenarios by ensuring that there is only a weak correlation between the existence of the user in the data and the result of the analysis. Dynamic graph algorithms maintain the solution to a problem (e.g., a matching) on an evolving input, i.e., a graph where nodes or edges are inserted or deleted over time. They output the value of the solution after each update operation, i.e., continuously. We study (event-level and user-level) differentially private algorithms for graph problems under continual observation, i.e., differentially private dynamic graph algorithms. We present event-level private algorithms for partially dynamic counting-based problems such as triangle count that improve the additive error by a polynomial factor (in the length T of the update sequence) on the state of the art, resulting in the first algorithms with additive error polylogarithmic in T.

We also give ε-differentially private and partially dynamic algorithms for minimum spanning tree, minimum cut, densest subgraph, and maximum matching. The additive error of our improved MST algorithm is $O(W \log^{3/2} T/\varepsilon)$, where W is the maximum weight of any edge, which, as we show, is tight up to a $(\sqrt{\log T}/\varepsilon)$-factor. For the other problems, we present a partially-dynamic algorithm with multiplicative error $(1+\beta)$ for any constant $\beta > 0$ and additive error $O(W \log(nW) \log(T)/(\varepsilon\beta))$. Finally, we show that the additive error for a broad class of dynamic graph algorithms with user-level privacy must be linear in the value of the output solution's range.

2012 ACM Subject Classification Theory of computation → Dynamic graph algorithms; Security and privacy → Pseudonymity, anonymity and untraceability

Keywords and phrases differential privacy, continual observation, dynamic graph algorithms

1 Introduction

Differential privacy aims to protect individuals whose data becomes part of an increasing number of data sets and is subject to analysis. A differentially private algorithm guarantees that its output depends only very little on an individual's contribution to the input data. Roughly speaking, an algorithm is ϵ-*differentially private* if the probability that it outputs O on data set D is at most an e^ϵ-factor of the probability that it outputs O on any adjacent data set D'. Two data sets are *adjacent* if they differ only in the data of a single user. Differential privacy was introduced in the setting of databases [9, 11], where users (entities) are typically represented by rows and data is recorded in columns. An important notion that allowed for the development of generic techniques and tools (like the Laplace and the exponential mechanism) is the *sensitivity* of a function f: the static sensitivity ρ of f is the maximum $|f(D) - f(D')|$ over all adjacent pairs D, D'. Differential privacy was later

generalized to a more challenging setting, where data evolves over time [12, 6]: a differentially private algorithm under *continual observation* must provide the same privacy guarantees as before, but for a sequence (or stream) of data sets instead of just a single data set. Often, this sequence results from updates to the original data set that arrive over time. In this setting, the presence or absence of a single user in one update can affect the algorithm's output on all future data sets, i.e., two adjacent databases can differ on all future outputs and, thus, have infinite sensitivity.

In this paper, we study differentially private graph algorithms under continual observation, i.e., for *dynamic* graph problems. The input is a sequence of graphs that results from node or edge updates, i.e., insertions or deletions. *Partially dynamic* algorithms only allow either insertions or deletions, *fully dynamic* algorithms allow both. After each update, the algorithm has to output a solution for the current input, i.e., the algorithm outputs a sequence of answers that is equally long as the input sequence. For differentially private graph algorithms two notions of *adjacency of graph sequences* exist: node-adjacency and edge-adjacency. Two graph sequences are *edge-adjacent* if they only differ in a single insertion or deletion of an edge. Similarly, two graph sequences are *node-adjacent* if they only differ in an insertion or deletion of a node.[1]

We initiate the study of differentially private algorithms for *non-local* partially dynamic graph problems. We consider a problem *non-local* if its (optimum) value cannot be derived from the graph's frequency histogram of constant-size subgraphs and call it *local* otherwise. Non-local problems include the cost of the minimum spanning tree, the weight of the global and *s-t* minimum cut, and the density of the densest subgraph. We also give improved algorithms for local graph problems and show various lower bounds on the additive error for differentially private dynamic graph algorithms.

Local problems. The only prior work on differentially private dynamic algorithms is an algorithm by Song et al. [29] for various *local* graph problems such as counting high-degree nodes, triangles and other constant-size subgraphs. We present an algorithm for these local problems that improves the additive error by a factor of $\sqrt{T}/\log^{3/2} T$, where T is the length of the update sequence. We also give the first differentially private partially-dynamic algorithm for the value of the minimum spanning tree. Table 1 lists upper and lower bounds for these results, where n is the number of nodes in the graph, W is the maximum edge weight (if applicable), D is the maximum node degree, ϵ is an arbitrarily small positive constant, and δ is the failure probability of the algorithm. We state below our main contributions in more detail. The update time of all our algorithms is linear in $\log T$ plus the time needed to solve the corresponding non-differentially private dynamic graph problem.

▶ **Theorem 1** (see Section 3). *Let $\varepsilon, \delta > 0$. There exist an ε-edge-differentially as well as an ε-node-differentially private algorithm for partially-dynamic minimum spanning tree, edge count, the number of high-degree nodes, the degree histogram, triangle count and k-star count that with probability at least $1 - \delta$ give an answer with additive error as shown in Table 1.*

Non-local problems. For non-local problems we present an algorithm that, by allowing a small multiplicative error, can obtain differentially private partially dynamic algorithms for a broad class of problems that includes the aforementioned problems. Table 2 lists our results for some common graph problems. The algorithm achieves the following performance.

---

[1] Of course, a graph can also be represented by a database, where, e.g., every row corresponds to an edge, but as we present algorithms that solve graph algorithmic problems we use the graph-based terminology through the paper.

■ **Table 1** Additive errors for partially-dynamic ε-differentially private algorithms with failure probability δ. We use D for the maximum degree and W for the maximum edge weight, n for the maximum number of nodes of any graph in the input sequence, and $\Lambda = \log(1/\delta)/\varepsilon$. The upper bounds follow from Corollary 14 and Table 3. See Section 5 for results on event-level lower bounds and Section 6 for user-level lower bounds.

| Graph function | partially dynamic | | fully dynamic | |
	edge-adj. event-level	node-adj. event-level	edge-adj. event-level	edge-adj. user-level
min. spanning tree	$\Omega(W \log T)$, $O(W \log^{3/2} T \cdot \Lambda)$	$\Omega(W \log T)$, $O(DW \log^{3/2} T \cdot \Lambda)$	$\Omega(W \log T)$	$\Omega(nW)$
min. cut, max. matching	$\Omega(W \log T)$	$\Omega(W \log T)$	$\Omega(W \log T)$	$\Omega(nW)$
edge count	$\Omega(\log T)$, $O(\log^{3/2} T \cdot \Lambda)$	$\Omega(\log T)$, $O(D \log^{3/2} T \cdot \Lambda)$	$\Omega(\log T)$, $O(\log^{3/2} T \cdot \Lambda)$	$\Omega(n^2)$
high-degree nodes	$\Omega(\log T)$, $O(\log^{3/2} T \cdot \Lambda))$	$\Omega(\log T)$, $O(D \log^{3/2} T \cdot \Lambda)$	$\Omega(\log T)$	$\Omega(n)$
degree histogram	$\Omega(\log T)$, $O(D \log^{3/2} T \cdot \Lambda))$	$\Omega(\log T)$, $O(D^2 \log^{3/2} T \cdot \Lambda)$	$\Omega(\log T)$	$\Omega(n)$
triangle count	$\Omega(\log T)$, $O(D \log^{3/2} T \cdot \Lambda))$	$\Omega(\log T)$, $O(D^2 \log^{3/2} T \cdot \Lambda)$	$\Omega(\log T)$	$\Omega(n^3)$
k-star count	$\Omega(\log T)$, $O(D^k \log^{3/2} T \cdot \Lambda)$	$\Omega(\log T)$, $O(D^k \log^{3/2} T \cdot \Lambda)$	$\Omega(\log T)$	$\Omega(n^{k+1})$

■ **Table 2** Private algorithms with failure probability δ with additional multiplicative error of $(1 + \beta)$ for arbitrary $\beta > 0$. We use $\Lambda = 1/(\varepsilon\delta\log(1 + \beta))$, D for the maximum degree, W for the maximum edge weight and n for the maximum number of nodes of any graph in the input sequence.

Graph function	partially dynamic, event-level	
	edge-adjacency	node-adjacency
minimum cut	$O(W\log(nW)\log(T)\cdot\Lambda)$	$O(DW\log(nW)\log(T)\cdot\Lambda)$
densest subgraph	$O(\log(n)\log(T)\cdot\Lambda)$	$O(\log(n)\log(T)\cdot\Lambda)$
minimum s,t-cut	$O(W\log(nW)\log(T)\cdot\Lambda)$	$O(DW\log(nW)\log(T)\cdot\Lambda)$
maximum matching	$O(W\log(nW)\log(T)\cdot\Lambda)$	$O(W\log(nW)\log(T)\cdot\Lambda)$

▶ **Theorem 2** (see Theorem 25). *Let $\varepsilon, \beta, \delta, r > 0$ and let f be a function with range $[1, r]$ that is monotone on all input sequences and has sensitivity ρ. There exists an ε-differentially private dynamic algorithm with multiplicative error $(1 + \beta)$, additive error $O(\rho\log(r)\log(T)/\log(1 + \delta))$ and failure probability δ that computes f.*

Note that for partially dynamic graph algorithms it holds that $T = O(n^2)$. Thus for local problems the bounds presented in Table 1 are superior to the bounds in Table 2.

Lower bounds. We complement these upper bounds by also giving some lower bounds on the additive error of any differentially private dynamic graph algorithm. For the problems in Table 1 we show lower bounds of $\Omega(W\log T)$, resp. $\Omega(\log T)$. Note that these lower bounds apply to the partially dynamic as well as to the fully dynamic setting.

The above notion of differential privacy is also known as *event-level* differential privacy, where two graph sequences differ in at most one "event", i.e., one update operation. A more challenging notion is *user-level* differential privacy. Two graph sequences are edge-adjacent *on user-level* if they differ in *any* number of updates for a single edge (as opposed to *one* update for a single edge in the case of the former *event-level* adjacency). Note that requiring user-level edge-differential privacy is a more stringent requirement on the algorithm than event-level edge-differential privacy.[2] We show strong lower bounds for edge-differentially private algorithms on user-level for a broad class of dynamic graph problems.

▶ **Theorem 3** (informal, see Theorem 45). *Let f be a function on graphs, and let G_1, G_2 be arbitrary graphs. There exists a $T \geq 1$ so that any ε-edge-differentially private dynamic algorithm on user-level that computes f must have additive error $\Omega(|f(G_1) - f(G_2)|)$ on input sequences of length at least T.*

This theorem leads to the lower bounds for fully dynamic algorithms stated in Table 1.

Technical contribution. Local problems. Our algorithms for local problems (Theorem 1) incorporate a counting scheme by Chan et al. [6] and the *difference sequence technique* by Song et al. [29]. The difference sequence technique addresses the problem that two adjacent graphs might differ on all outputs starting from the point in the update sequence where their inputs differ. More formally, let $f_{\mathcal{G}}(t)$ be the output of the algorithm after operation t in the graph sequence $\mathcal{G}$. Then the *continuous global sensitivity* $\sum_{t=1}^{T}|f_{\mathcal{G}}(t) - f_{\mathcal{G}'}((t)|$ might be $\Theta(\rho T)$. Using the "standard" Laplacian mechanism for such a large sensitivity would, thus, lead to an additive error linear in T. The idea of [29] is to use instead the *difference sequence of f* defined as $\Delta f(t) = f(t) - f(t - 1)$, as they observed that for various local graph properties

---

[2] Node-adjacency on user-level is defined accordingly but not studied in this paper.

the continuous global sensitivity of the difference sequence, i.e., $\sum_{t=1}^{T} |\Delta f_{\mathcal{G}}(t) - \Delta f_{\mathcal{G}'}((t)|$ can be bounded by a function independent of T. However, their resulting partially-dynamic algorithms still have an additive error linear in $\sqrt{T}$. We show how to combine the continuous global sensitivity of the difference sequence with the binary counting scheme of Chan et al. [6] to achieve partially-dynamic algorithms with additive error linear in $\log^{3/2} T$.

Furthermore we show that the approach based on the continuous global sensitivity of the difference sequence fails, if the presence or absence of a node or edge can significantly change the target function's value for all of the subsequent graphs. In particular, we show that for several graph problems like minimum cut and maximum matching changes in the function value between adjacent graph sequences can occur at every time step even for partially dynamic sequences, resulting in a continuous global sensitivity of the difference sequence that is linear in T. This implies that this technique cannot be used to achieve differentially private dynamic algorithms for these problems.

Non-local problems. We leverage the fact that the sparse vector technique [13] provides negative answers to threshold queries with little effect on the additive error to approximate monotone functions f on graphs under continual observation (e.g., the minimum cut value in an incremental graph) with multiplicative error $(1 + \beta)$: If r is the maximum value of f, we choose thresholds $(1 + \beta), \ldots, (1 + \beta)^{\log_{1+\beta}(r)}$ for the queries. This results in at most $\log_{1+\beta}(r)$ positive answers, which affect the additive error linearly, while the at most T negative answers affect the additive error only logarithmically instead of linearly.

Lower bounds. Dwork et al. [12] had given a lower bound for counting in binary streams. We reduce this problem to partially dynamic graph problems on the event-level to achieve the event-level lower bounds.

For the user-level lower bounds we assume by contradiction that an ϵ-differentially private dynamic algorithm $\mathcal{A}$ with "small" additive error exists and construct an exponential number of graph sequences that are all user-level "edge-close" to a simple graph sequence $\mathcal{G}'$. Furthermore any two such graph sequences have at least one position with two very different graphs such that $\mathcal{A}$ (due to its small additive error) must return two different outputs at this position, which leads to two different output sequences if $\mathcal{A}$ answers within its error bound. Let O_i be the set of accurate output sequences of $\mathcal{A}$ on one of the graph sequences G_i. By the previous condition $O_i \cap O_j = \emptyset$ if $i \neq j$. As G_i is "edge-close" to $\mathcal{G}'$, there is a relatively large probability (depending on the degree of "closeness") that O_i is output when $\mathcal{A}$ runs on $\mathcal{G}'$. This holds for all i. However, since $O_i \cap O_j = \emptyset$ if $i \neq j$ and we have constructed exponentially many graph sequences G_i, the sum of these probabilities over all i adds up to a value larger than 1, which gives a contradiction. The proof is based on ideas of a lower bound proof for databases in [12].

Related Work. Differential privacy, developed in [9, 11], is the de facto gold standard of privacy definitions and several lines of research have since been investigated [2, 25, 20, 10, 16, 4]. In particular, differentially private algorithms for the release of various graph statistics such as subgraph counts [19, 3, 7, 21, 32], degree distributions [18, 8, 33], minimum spanning tree [26], spectral properties [31, 1], cut problems [16, 1, 14], and parameter estimation for special classes of graphs [23] have been proposed. Dwork et al. [12] and Chan et al. [6] extended the analysis of differentially private algorithms to the regime of continual observation, i.e., to input that evolves over time. Since many data sets in applications are evolving data sets, this has lead to results for several problems motivated by practice [5, 22, 27, 15, 30]. Only one prior work analyzes evolving graphs: Song et al. [29] study problems in incremental bounded-degree graphs that are functions of local neighborhoods. Our results improve all bounds for undirected graphs initially established in [29] by a factor of $\sqrt{T}/\log^{3/2} T$ in the

additive error.

2   Preliminaries

2.1   Graphs and Graph Sequences

We consider undirected graphs $G = (V, E)$, which change dynamically. Graphs may be edge-weighted, in which case $G = (V, E, w)$, where $w : E \to \mathbb{N}$. The evolution of a graph is described by a *graph sequence* $\mathcal{G} = (G_1, G_2, \dots)$, where $G_t = (V_t, E_t)$ is derived from G_{t-1} by applying updates, i.e., inserting or deleting nodes or edges. We denote by $|\mathcal{G}|$ the length of $\mathcal{G}$, i.e., the number of graphs in the sequence. At time t we delete a set of nodes ∂V_t^- along with the corresponding edges ∂E_t^- and insert a set of nodes ∂V_t^+ and edges ∂E_t^+. More formally, $V_t = (V_{t-1} \setminus \partial V_t^-) \cup \partial V_t^+$ and $E_t = (E_{t-1} \setminus \partial E_t^-) \cup \partial E_t^+$, with initial node and edge sets V_0, E_0, which may be non-empty. If a node v is deleted, then all incident edges are deleted at the same time, i.e., if $v \in \partial V_t^-$, then $(u, v) \in \partial E_t^-$ for all $(u, v) \in E_{t-1}$. Both endpoints of an edge inserted at time t need to be in the graph at time t, i.e., $\partial E_t^+ \subseteq ((V_{t-1} \setminus \partial V_t^-) \cup \partial V_t^+) \times ((V_{t-1} \setminus \partial V_t^-) \cup \partial V_t^+)$. The tuple $(\partial V_t^+, \partial V_t^-, \partial E_t^+, \partial E_t^-)$ is the *update* at time t. For any graph G and any update u, let $G \oplus u$ be the graph that results from applying u on G.

A graph sequence is *incremental* if $\partial E_t^- = \partial V_t^- = \emptyset$ at all time steps t. A graph sequence is *decremental* if $\partial E_t^+ = \partial V_t^+ = \emptyset$ at all time steps t. Incremental and decremental graph sequences are called *partially dynamic*. Graph sequences that are neither incremental nor decremental are *fully dynamic*.

Our goal is to continually release the value of a *graph function f* which takes a graph as input and outputs a real number. In other words, given a graph sequence $\mathcal{G} = (G_1, G_2, \dots)$ we want to compute the sequence $f(\mathcal{G}) = (f(G_1), f(G_2), \dots)$. We write $f(t)$ for $f(G_t)$. Our algorithms will compute an update to the value of f at each time step, i.e., we compute $\Delta f(t) = f(t) - f(t-1)$. We call the sequence Δf the *difference sequence of f*.

Given a graph function g the *continuous global sensitivity* $\mathrm{GS}(g)$ of g is defined as the maximum value of $||g(S) - g(S')||_1$ over all adjacent graph sequences S, S'. We will define adjacency of graph sequences below. In our case, we are often interested in the continuous global sensitivity of the difference sequence of a graph function f, which is given by the maximum value of $\sum_{t=1}^{T} |\Delta f_{\mathcal{G}}(t) - \Delta f_{\mathcal{G}'}(t)|$, where $\Delta f_{\mathcal{G}}$ and $\Delta f_{\mathcal{G}'}$ are the difference sequences of f corresponding to adjacent graph sequences $\mathcal{G}$ and $\mathcal{G}'$.

Two graphs are *edge-adjacent* if they differ in one edge. We also define global sensitivity of functions applied to a single graph. Let g be a graph function. Its *static global sensitivity* $\mathrm{GS}_{\mathrm{static}}(g)$ is the maximum value of $|g(G) - g(G')|$ over all edge-adjacent graphs G, G'.

2.2   Differential Privacy

The range of an algorithm $\mathcal{A}$, $\mathrm{Range}(\mathcal{A})$, is the set of all possible output values of $\mathcal{A}$. We denote the Laplace distribution with mean μ and scale b by $\mathrm{Lap}(\mu, b)$. If $\mu = 0$, we write $\mathrm{Lap}(b)$.

▶ **Definition 4** (ε-differential privacy). *A randomized algorithm $\mathcal{A}$ is ε-differentially private if for any two* adjacent *databases B, B' and any $S \subseteq \mathrm{Range}(\mathcal{A})$ we have $\Pr[\mathcal{A}(B) \in S] \leq e^{\varepsilon} \cdot \Pr[\mathcal{A}(B') \in S]$. The parameter ε is called the* privacy loss *of $\mathcal{A}$.*

To apply Definition 4 to graph sequences we now define adjacency for graph sequences. First, we define edge-adjacency, which is useful if the data to be protected is associated with

the edges in the graph sequence. Then, we define node-adjacency, which provides stronger privacy guarantees.

▶ **Definition 5** (Edge-adjacency). *Let $\mathcal{G}$, $\mathcal{G}'$ be graph sequences as defined above with associated sequences of updates (∂V_t^-), (∂V_t^+), (∂E_t^-), (∂E_t^+) and $(\partial V_t^{-'})$, $(\partial V_t^{+'})$, $(\partial E_t^{-'})$, $(\partial E_t^{+'})$. Let $\partial V_t^- = \partial V_t^{-'}$ and $\partial V_t^+ = \partial V_t^{+'}$ for all t. Let the initial node and edge sets for $\mathcal{G}$ and $\mathcal{G}'$ be $V_0 = V_0'$ and $E_0 = E_0'$. Assume w.l.o.g. that $\partial E_t^{-'} \subseteq \partial E_t^-$ and $\partial E_t^{+'} \subseteq \partial E_t^+$ for all t. The graph sequences $\mathcal{G}$ and $\mathcal{G}'$ are* adjacent on e^* *if $|\mathcal{G}| = |\mathcal{G}'|$, there exists an edge e^* and one of the following statements holds:*
1. $\partial E_t^- = \partial E_t^{-'} \, \forall t$ *and* $\exists t^*$ *such that* $\partial E_{t^*}^+ \setminus \partial E_{t^*}^{+'} = \{e^*\}$ *and* $\partial E_t^+ = \partial E_t^{+'} \, \forall t \neq t^*$;
2. $\partial E_t^+ = \partial E_t^{+'} \, \forall t$ *and* $\exists t^*$ *such that* $\partial E_{t^*}^- \setminus \partial E_{t^*}^{-'} = \{e^*\}$ *and* $\partial E_t^- = \partial E_t^{-'} \, \forall t \neq t^*$;

Remark. If $\mathcal{G}$ and $\mathcal{G}'$ are edge-adjacent, then for any index i the graphs at index i in the two sequences are edge-adjacent.

Two edge-adjacent graph sequences differ in either the insertion or the deletion of a single edge e^*. There are several special cases that fit into this definition. For example, we may have $\partial V_t^- = \partial V_t^{-'} = \partial V_t^+ = \partial V_{t'}^+ = \emptyset$, so only edge updates would be allowed. Similarly, we can use the definition in the incremental setting by assuming $\partial V_t^- = \partial V_t^{-'} = \partial E_t^- = \partial E_t^{-'} = \emptyset$.

The definition of node-adjacency is similar to that of edge-adjacency, but poses additional constraints on the edge update sets.

▶ **Definition 6** (Node-adjacency). *Let $\mathcal{G}$, $\mathcal{G}'$ be graph sequences as defined above with associated sequences of updates (∂V_t^-), (∂V_t^+), (∂E_t^-), (∂E_t^+) and $(\partial V_t^{-'})$, $(\partial V_t^{+'})$, $(\partial E_t^{-'})$, $(\partial E_t^{+'})$. Assume w.l.o.g. that $\partial V_t^{-'} \subseteq \partial V_t^-$ and $\partial V_t^{+'} \subseteq \partial V_t^+$ for all t. The graph sequences $\mathcal{G}$ and $\mathcal{G}'$ are* adjacent on v^* *if $|\mathcal{G}| = |\mathcal{G}'|$, there exists a node v^* and one of the following statements holds:*
1. $\partial V_t^- = \partial V_t^{-'} \, \forall t$ *and* $\exists t^*$ *such that* $\partial V_t^+ \setminus \partial V_t^{+'} = \{v^*\}$ *and* $\partial V_t^+ = \partial V_t^{+'} \, \forall t \neq t^*$;
2. $\partial V_t^+ = \partial V_t^{+'} \, \forall t$ *and* $\exists t^*$ *such that* $\partial V_t^- \setminus \partial V_t^{-'} = \{v^*\}$ *and* $\partial V_t^- = \partial V_t^{-'} \, \forall t \neq t^*$;
Additionally, all edges in ∂E_t^+ and ∂E_t^- are incident to at least one node in ∂V_t^+ and ∂V_t^-, respectively. Lastly, we require that $\partial E_t^{+'}$ ($\partial E_t^{-'}$) is the maximal subset of ∂E_t^+ (∂E_t^-) that does not contain edges incident to v^*.

We define the following notions of differential privacy based on these definitions of adjacency.

▶ **Definition 7.** *An algorithm is ε-edge-differentially private (on event-level) if it is ε-differentially private when considering edge-adjacency. An algorithm is ε-node-differentially private (on event-level) if it is ε-differentially private when considering node-adjacency.*

When explicitly stated, we consider a stronger version of ε-differential privacy, which provides adjacency on user-level. While adjacency on event-level only allows two graph sequences to differ in a single update, user-level adjacency allows any number of updates to differ as long as they affect the same edge (for edge-adjacency) or node (for node-adjacency), respectively.

▶ **Definition 8.** *Let $\mathcal{G} = (G_1, \ldots), \mathcal{G}' = (G_1', \ldots)$ be graph sequences. The two sequences are* edge-adjacent *on user-level if there exists an edge e^* and a sequence of graph sequences $S = (\mathcal{G}_1, \ldots, \mathcal{G}_\ell)$ so that $\mathcal{G}_1 = \mathcal{G}, \mathcal{G}_\ell = \mathcal{G}'$ and, for any $i \in [\ell-1]$, $\mathcal{G}_i$ and $\mathcal{G}_{i+1}$ are edge-adjacent on e^*. An algorithm is ε-edge-differentially private on user-level if it is ε-differentially private when considering edge-adjacency on user-level.*

2.3 Counting Mechanisms

Some of our algorithms for releasing differentially private estimates of functions on graph sequences rely on algorithms for counting in streams.

A *stream* $\sigma = \sigma(1)\sigma(2)\cdots$ is a string of *items* $\sigma(i) \in \{L_1, \ldots, L_2\} \subseteq \mathbb{Z}$, where the i-th item is associated with the i-th *time step*. A *binary stream* has $L_1 = 0$ and $L_2 = 1$. We denote the length of a stream, i.e., the number of time steps in the stream, by $|\sigma|$. Stream σ and σ' are adjacent if $|\sigma| = |\sigma'|$ and if there exists one and only one t^* such that $\sigma(t^*) \neq \sigma'(t^*)$ and $\sigma(t) = \sigma'(t)$ for all $t \neq t^*$.

A *counting mechanism* $\mathcal{A}(\sigma)$ takes a stream σ and outputs a real number for every time step. For all time steps t, $\mathcal{A}$'s output at time t is independent of all $\sigma(i)$ for $i > t$. At each time t a counting mechanism should estimate the count $c(t) = \sum_{i=1}^{t} \sigma(i)$.

Following Chan et al. [6] we describe our mechanisms in terms of *p-sums*, which are partial sums of the stream over a time interval. For a p-sum p we denote the beginning and end of the time interval by $\mathrm{start}(p)$ and $\mathrm{end}(p)$, respectively. With this notation the value of p is $\sum_{t=\mathrm{start}(p)}^{\mathrm{end}(p)} \sigma(t)$. To preserve privacy we add noise to p-sums and obtain *noisy p-sums*: given a p-sum p, a noisy p-sum is $\hat{p} = p + \gamma$, where γ is drawn from a Laplace-distribution.

2.4 Sparse Vector Technique

The *sparse vector technique* (SVT) was introduced by Dwork et al. [13] and was subsequently improved [17, 28]. SVT can be used to save privacy budget whenever a sequence of threshold queries $f_1, \ldots, f_T$ is evaluated on a database, but only $c \ll T$ queries are expected to exceed the threshold. Here, a threshold query asks whether a function f_i evaluates to a value above some threshold t_i on the input database. Using SVT, only queries that are answered positively reduce the privacy budget. We use the following variant of SVT, which is due to Lyu et al.

▶ **Lemma 9** ([24]). *Let D be a database, $\epsilon, \rho, c > 0$ and let $(f_1, t_1), \ldots$ be a sequence of mappings f_i from input databases to $\mathbb{R}$ and thresholds $t_i \in \mathbb{R}$, which may be generated adaptively one after another so that $\rho \geq \max_i \mathrm{GS}_{\mathrm{static}}(f_i)$. Algorithm 1 is ϵ-private.*

▮ **Algorithm 1** SVT algorithm [24]

---

1 Function InitializeSvt(D, ρ, ϵ, c)
2 $\quad \epsilon_1 \leftarrow \epsilon/2, \zeta \leftarrow \mathrm{Lap}(\rho/\epsilon_1), \epsilon_2 \leftarrow \epsilon - \epsilon_1, count \leftarrow 0$
3 Function ProcessSvtQuery(f_i, t_i)
4 $\quad \nu_i \leftarrow \mathrm{Lap}(2c\rho/\epsilon_2)$
5 $\quad$ **if** $count \geq c$ **then**
6 $\quad\quad$ **return** *abort*
7 $\quad$ **if** $f_i(D) + \nu_i \geq t_i + \zeta$ **then**
8 $\quad\quad count \leftarrow count + 1$, **return** $\top$
9 $\quad$ **else**
10 $\quad\quad$ **return** $\bot$

---

3 Mechanisms Based on Continuous Global Sensitivity

Some of our mechanisms for privately estimating graph functions are based on mechanisms for counting in streams. In both settings, we compute the sum of a sequence of numbers

and we will show that the mechanisms for counting can be transferred to the graph setting. However, there are differences in the analysis. In counting, the input streams differ at only one time step. This allows us to bound the difference in the true value between adjacent inputs and leads to low error. In the graph setting, the sequence of numbers can vary at many time steps. Here however, we use properties of the counting mechanisms to show that the total difference for this sequence can still be bounded, which results in the same error as in the counting setting.

We first generalize the counting mechanisms by Chan et al. [6] to streams of integers with bounded absolute value, and then transfer them to estimating graph functions.

3.1 Non-Binary Counting

We generalize the counting mechanisms of Chan et al. [6] to streams of numbers in $\{-L, \ldots, L\}$, for some constant L. We view these algorithms as releasing noisy p-sums from which the count can be estimated. The generic algorithm is outlined in Algorithm 2 on page 9.

The algorithm releases a vector of noisy p-sums over T time steps, such that at every time step the noisy p-sums needed to estimate the count up to this time are available. Each of the noisy p-sums is computed exactly once. See the proof of Corollary 14 for an example on how to use p-sums.

In order to achieve the desired privacy loss the mechanisms need to meet the following requirements. Let $\mathcal{A}$ be a counting mechanism. We define $\mathrm{Range}(\mathcal{A}) = \mathbb{R}^k$, where k is the total number of p-sums used by $\mathcal{A}$ and every item of the vector output by $\mathcal{A}$ is a p-sum. We assume that the time intervals represented by the p-sums in the output of $\mathcal{A}$ are deterministic and only depend on the length T of the input stream. For example, consider any two streams σ and σ' of length T. The ℓ-th element of $\mathcal{A}(\sigma)$ and $\mathcal{A}(\sigma')$ will be p-sums of the same time interval $[\mathrm{start}(\ell), \mathrm{end}(\ell)]$. We further assume that the p-sums are computed independently from each other in the following way: $\mathcal{A}$ computes the true p-sum and then adds noise from $\mathrm{Lap}(z \cdot \varepsilon^{-1})$, where z is a sensitivity parameter. To analyze the error of the algorithms we

---

■ **Algorithm 2** Generic counting mechanism

---

1 **Input:** privacy loss ε, stream σ of items $\{L_1, \ldots, L_2\}$ with $|\sigma| = T$
2 **Output:** vector of noisy p-sums $a \in \mathbb{R}^k$, released over T time steps
3 **Initialization:** Determine which p-sums to compute based on T
4 At each time step $t \in \{1, \ldots, T\}$:
5   Compute new p-sums $p_i, \ldots, p_j$ for t
6   **For** $\ell = i, \ldots, j$:
7 $\hat{p}_\ell = p_\ell + \gamma_\ell, \quad \gamma_\ell \sim \mathrm{Lap}((L_2 - L_1)\varepsilon^{-1})$
8   Release new noisy p-sums $\hat{p}_i, \ldots, \hat{p}_j$

---

need the following Corollary from Chan et al. [6]:

▶ **Corollary 10** (Corollary 2.9 from [6])**.** *Suppose γ_i are independent random variables, where each γ_i has Laplace distribution $\mathrm{Lap}(b_i)$. Let $Y := \sum_i \gamma_i$ and $b_M = \max_i b_i$. Let $\nu \geq \sqrt{\sum_i b_i^2}$. Suppose $0 < \delta < 1$ and $\nu > \max\left\{\sqrt{\sum_i b_i^2}, b_M \sqrt{\ln \frac{2}{\delta}}\right\}$. Then, $\Pr\left[|Y| > \nu \sqrt{8 \ln \frac{2}{\delta}}\right] \leq \delta$.*

To simplify our presentation and improve readability, we choose $\nu := \sqrt{\sum_i b_i^2} \cdot \sqrt{\ln \frac{2}{\delta}}$ and use the following slightly weaker result: with probability at least $1 - \delta$, the quantity $|Y|$ is at most $O(\sqrt{\sum_i b_i^2} \log \frac{1}{\delta})$.

The next lemma is stated informally in [6].

▶ **Lemma 11** (Observation 1 from [6])**.** *Let $\mathcal{A}$ be a counting mechanism as described in Algorithm 2 with $L_1 = 0$ and $L_2 = 1$ that releases k noisy p-sums, such that the count at any time step can be computed as the sum of at most y p-sums and every item is part of at most x p-sums. Then, $\mathcal{A}$ is $(x \cdot \varepsilon)$-differentially private, and the error is $O(\varepsilon^{-1}\sqrt{y}\log\frac{1}{\delta})$ with probability at least $1 - \delta$ at each time step.*

Proof. The error follows from Corollary 10 since the noise at each time step is the sum of at most y independent random variables with distribution $\mathrm{Lap}(1/\varepsilon)$.

We show that the vector of all noisy p-sums output by $\mathcal{A}$ preserves $(x \cdot \varepsilon)$-differential privacy. Then, since the count is estimated by summing over $(x \cdot \varepsilon)$-differentially private noisy p-sums the count itself is $(x \cdot \varepsilon)$-differentially private. As described above, $\mathrm{Range}(\mathcal{A}) = \mathbb{R}^k$, where k is the number of p-sums that the algorithm computes in total.

Let σ, σ' be two adjacent streams. We consider the joint distribution of the random variables $Z_1, \ldots, Z_k$ that correspond to the noisy p-sums output by $\mathcal{A}$. Let $p_i(z_i)$, $p_i'(z_i)$ for $z_i \in \mathrm{Range}(Z_i)$ be the probability density function of Z_i when the input to $\mathcal{A}$ is σ and σ', respectively. Note that each of these densities is the probability density function of a Laplace distribution. Let $p(z_1, \ldots, z_k)$, $p'(z_1, \ldots, z_k)$ be the joint probability density when the input to $\mathcal{A}$ is σ and σ', respectively. Note that the random variables Z_i are continuous, since the noise added to the p-sums is drawn from a continuous distribution over $\mathbb{R}$. Since the Z_i are independent, we have

$$p(z_1, \ldots, z_k) = \prod_{i=1}^{k} p_i(z_i)$$

and

$$p'(z_1, \ldots, z_k) = \prod_{i=1}^{k} p_i'(z_i).$$

We now show that these joint densities differ by at most an $\exp(x \cdot \varepsilon)$-factor for all possible outputs of $\mathcal{A}$. Let $s = (s_1, \ldots, s_k)^\top \in \mathrm{Range}(\mathcal{A})$. Note that $\mathcal{A}$ outputs a vector of noisy p-sums from which the count can be computed, so each item in s is a noisy p-sum. Let $c = (c_1, \ldots, c_k)^\top$ be the noiseless p-sums of stream σ corresponding to the noisy p-sums computed by $\mathcal{A}$. Let I be the set of indices to (noisy) p-sums involving time t'. By assumption, $|I| \le x$, since any item participates in at most x (noisy) p-sums.

$$
\begin{aligned}
\frac{p(s)}{p'(s)} &= \prod_{i=1}^{k} \frac{p_i(s_i)}{p_i'(s_i)} = \prod_{i \in I} \frac{p_i(s_i)}{p_i'(s_i)} \\
&= \prod_{i \in I} \frac{\exp(-\varepsilon|c_i - s_i|)}{\exp(-\varepsilon|c_i \pm 1 - s_i|)} \\
&= \prod_{i \in I} \exp(\varepsilon(|c_i \pm 1 - s_i| - |c_i - s_i|)) \\
&\le \prod_{i \in I} \exp(\varepsilon|c_i \pm 1 - s_i - c_i + s_i|) \\
&= \exp(\varepsilon)^{|I|} \le \exp(x \cdot \varepsilon).
\end{aligned}
\tag{1}
$$

The first equality follows from the assumption that the noisy p-sums are independent, as discussed above. Since all noiseless p-sums that do not involve the item at t' are equal, the

densities $p_j(s_j)$ and $p'_j(s_j)$ cancel out for $j \notin I$, so we can take the product over just the noisy p-sums corresponding to indices in I. Plugging in the density function of the Laplace distribution and applying the reverse triangle inequality yields the first inequality. The second inequality follows from the assumption $|I| \leq x$.

Now, for any $S \subseteq \text{Range}(\mathcal{A})$ we have

$$\Pr[\mathcal{A}(\sigma) \in S] = \int_S p(s)ds \leq \int_S e^{x \cdot \varepsilon} p'(s)ds = e^{x \cdot \varepsilon} \Pr[A(\sigma') \in S],$$

where the inequality follows from (1). Thus, $\mathcal{A}$ is indeed $(x \cdot \varepsilon)$-differentially private. ◄

To extend the counting mechanisms by Chan et al. to non-binary streams of values in $\{-L, \ldots, L\}$, we only need to account for the increased sensitivity in the scale of the Laplace distribution. By Lemma 12, we can use the mechanisms of Chan et al. [6] to compute the sum of a stream of numbers in $\{-L, \ldots, L\}$, but gain a factor $2L$ in the error.

▶ **Lemma 12** (Extension of Lemma 11). *Let $\mathcal{A}$ be a mechanism as in Algorithm 2 (see page 9) with $L_1 = -L$ and $L_2 = L$ that releases k noisy p-sums, such that the count at any time step can be computed as the sum of at most y p-sums, and every item is part of at most x p-sums. Furthermore, $\mathcal{A}$ adds noise $\text{Lap}(2L/\varepsilon)$ to every p-sum. Then, $\mathcal{A}$ is $(x \cdot \varepsilon)$-differentially private, and the error is $O(L\varepsilon^{-1}\sqrt{y}\log\frac{1}{\delta})$ with probability at least $1 - \delta$ at each time step.*

Proof. The error follows from Corollary 10.

We proceed as in the proof of Lemma 11. Let σ, σ' be two adjacent streams that differ only at time t', i.e., $\sigma(t) = \sigma'(t)$ for all $t \neq t'$ and $\sigma(t') \neq \sigma'(t')$. We denote by $\delta' = \sigma'(t') - \sigma(t')$ the difference in the items at time t'. Note that $|\delta'| \leq 2L$, since $\sigma(t), \sigma'(t) \in \{-L, \ldots, L\}$ for all t. Let $c = (c_1, \ldots, c_k)^\top$ be the noiseless p-sums of stream σ corresponding to the noisy p-sums computed by $\mathcal{A}$. Let I be the set of indices to (noisy) p-sums involving time t'. By assumption, $|I| \leq x$, since any item participates in at most x (noisy) p-sums.

As in the proof of Lemma 11 we consider the joint probability density of the output of $\mathcal{A}$ when the input is σ and σ'. Note that the noisy p-sums output by $\mathcal{A}$ are independent continuous random variables with range $\mathbb{R}$. We have

$$p(z) = \prod_{i=1}^k p_i(z_i)$$

and

$$p'(z) = \prod_{i=1}^k p'_i(z_i),$$

where the p_i and p'_i are the probability density function of the individual noisy p-sums with input stream σ and σ', respectively.

Let $s = (s_1, \ldots, s_k)^\top \in \text{Range}(\mathcal{A})$.

$$\frac{p(s)}{p'(s)} = \prod_{i=1}^k \frac{p_i(s_i)}{p'_i(s_i)}$$

$$= \prod_{i \in I} \frac{p_i(s_i)}{p'_i(s_i)}$$

$$= \prod_{i \in I} \frac{\exp\left(-\frac{\varepsilon|c_i - s_i|}{2L}\right)}{\exp\left(-\frac{\varepsilon|c_i + \delta' - s_i|}{2L}\right)}$$

$$= \prod_{i \in I} \exp\left(\frac{\varepsilon(|c_i + \delta' - s_i| - |c_i - s_i|)}{2L}\right)$$

$$\leq \prod_{i \in I} \exp\left(\frac{\varepsilon|c_i + \delta' - s_i - c_i + s_i|}{2L}\right)$$

$$= \exp\left(\frac{\varepsilon|\delta'|}{2L}\right)^{|I|} \leq \exp(x \cdot \varepsilon).$$

Thus, $\mathcal{A}$ is $x \cdot \varepsilon$-differentially private, since

$$\Pr[\mathcal{A}(\sigma) \in S] = \int_S p(s)ds \leq \int_S e^{x \cdot \varepsilon} p'(s)ds = e^{x \cdot \varepsilon} \Pr[A(\sigma') \in S]$$

for any $S \subseteq \text{Range}(\mathcal{A})$.   ◀

3.2 Graph Functions via Counting Mechanisms

■ Algorithm 3 Generic graph sequence mechanism

Input: privacy loss ε, contin. global sensitivity Γ, graph sequence $\mathcal{G} = (G_1, \ldots, G_T)$
Output: vector of noisy p-sums $a \in \mathbb{R}^k$, released over T time steps
1 Initialization: Determine which p-sums to compute based on T
2 At each time step $t \in \{1, \ldots, T\}$:
3   Compute $f(t)$ and $\Delta f(t) = f(t) - f(t-1)$, $f(0) := 0$
4   Compute new p-sums $p_i, \ldots, p_j$ for the sequence Δf
5   **For** $\ell = i, \ldots, j$:
6 $\hat{p}_\ell = p_\ell + \gamma_\ell, \quad \gamma_\ell \sim \text{Lap}(\Gamma\varepsilon^{-1})$
7   Release new noisy p-sums $\hat{p}_i, \ldots, \hat{p}_j$

We adapt the counting mechanisms to continually release graph functions by following the approach by Song et al. [29]. Algorithm 3 outlines the generic algorithm. It is similar to Algorithm 2, with the difference that the stream of numbers to be summed is computed from a graph sequence $\mathcal{G}$. The algorithm is independent of the notion of adjacency of graph sequences, however the additive error is linear in the continuous global sensitivity of the difference sequence Δf.

In counting binary streams we considered adjacent inputs that differ at exactly one time step. In the graph setting however, the stream of numbers that we sum, i.e., the difference sequence Δf, can differ in multiple time steps between two adjacent graph sequences. We illustrate this with a simple example. Let f be the function that counts the number of edges in a graph and consider two node-adjacent incremental graph sequences $\mathcal{G}, \mathcal{G}'$. $\mathcal{G}$ contains an additional node v^*, that is not present in $\mathcal{G}'$. Whenever a neighbor is added to v^* in $\mathcal{G}$, the number of edges in $\mathcal{G}$ increases by more than the number of edges in $\mathcal{G}'$. Thus, every time a neighbor to v^* is inserted, the difference sequence of f will differ between $\mathcal{G}$ and $\mathcal{G}'$.

To generalize this, consider two adjacent graph sequences $\mathcal{G}, \mathcal{G}'$ that differ in an update at time t'. Let $\Delta f_{\mathcal{G}}$ and $\Delta f_{\mathcal{G}'}$ be the difference sequences used to compute the graph function f on $\mathcal{G}$ and $\mathcal{G}'$. As discussed above we may have $\Delta f_{\mathcal{G}}(t) \neq \Delta f_{\mathcal{G}'}(t)$ for all $t \geq t'$. Thus, more than x p-sums can be different, which complicates the proof of the privacy loss. However, we observe that the set P of p-sums with differing values can be partitioned into x sets $P_1, \ldots, P_x$, where the p-sums in each P_i cover disjoint time intervals. By using a bound on the continuous global sensitivity of the difference sequence Δf, this will lead to a privacy loss

of $x \cdot \varepsilon$. We now prove that the results on privacy loss and error transfer from the counting mechanisms to the graph mechanisms.

▶ **Lemma 13** (Lemma 12 for graph sequences). *Let f be a graph function whose difference sequence has continuous global sensitivity Γ. Let $0 < \delta < 1$ and $\varepsilon > 0$. Let $\mathcal{A}$ be a mechanism to estimate f as in Algorithm 3 that releases k noisy p-sums and satisfies the following conditions:*

1. *at any time step the value of a graph function f can be estimated as the sum of at most y noisy p-sums,*
2. *$\mathcal{A}$ adds independent noise from $\mathrm{Lap}(\Gamma/\varepsilon)$ to every p-sum,*
3. *the set P of p-sums computed by the algorithm can be partitioned into at most x subsets $P_1, \ldots, P_x$, such that in each partition P_x all p-sums cover disjoint time intervals. That is, for all $P_i \in \{P_1, \ldots, P_x\}$ and all $j, k \in P_i$, $j \neq k$, it holds that (1) $\mathrm{start}(j) \neq \mathrm{start}(k)$ and (2) $\mathrm{start}(j) < \mathrm{start}(k) \implies \mathrm{end}(j) < \mathrm{start}(k)$.*

Then, $\mathcal{A}$ is $(x \cdot \varepsilon)$-differentially private, and the error is $O(\Gamma \varepsilon^{-1} \sqrt{y} \log \frac{1}{\delta})$ with probability at least $1 - \delta$ at each time step.

Proof. As before the error follows from Corollary 10 and condition 1.

Let $\mathcal{G} = (G_1, \ldots, G_T)$, $\mathcal{G}' = (G_1', \ldots, G_T')$ be two adjacent graph sequences and let $f(t) = f(G_t)$, $f'(t) = f(G_t')$. By definition of adjacent graph sequences (see Definitions 5 and 6) we have $f(0) = f'(0) = f((V_0, E_0))$. At every time step the algorithm computes $\Delta f_{\mathcal{G}}(t) = f(t) - f(t-1)$ and $\Delta f_{\mathcal{G}'}(t) = f'(t) - f'(t-1)$.

Note that the noisy p-sums computed by $\mathcal{A}$ are independent continuous random variables with joint distribution

$$ p(z) = \prod_{i=1}^{k} p_i(z_i) $$

and

$$ p'(z) = \prod_{i=1}^{k} p_i'(z_i) $$

when the input stream is $\mathcal{G}$ and $\mathcal{G}'$, respectively. As in the proofs of Lemmas 11 and 12, the p_i and p_i' are the marginal probability density functions. The random variables are continuous, since the noise added to the p-sums is drawn from a continuous distribution over $\mathbb{R}$.

Furthermore, let $c = (c_1, \ldots, c_k)^\top$ and $c' = (c_1', \ldots, c_k')^\top$ be the noiseless p-sums corresponding to the noisy p-sums output by $\mathcal{A}$ on inputs $\mathcal{G}$ and $\mathcal{G}'$, respectively. For each time step $t \in \{1, \ldots, T\}$ we define $\delta(t) = \Delta f_{\mathcal{G}'}(t) - \Delta f_{\mathcal{G}}(t)$. Note that $\sum_{t=1}^{T} |\delta(t)| \leq \Gamma$, by definition of the continuous global sensitivity Γ of the difference sequence. In this proof we use $\mathrm{start}(i)$ and $\mathrm{end}(i)$ to denote the beginning and end of the time interval corresponding to the p-sums with index i. For each $i \in \{1, \ldots, k\}$ we define $\delta_i = c_i' - c_i = \sum_{t=\mathrm{start}(i)}^{\mathrm{end}(i)} \delta(t)$. Let $I = \{i \mid \delta_i \neq 0\}$ be the indices of p-sums where the values for the two graph sequences are different. Lastly, let $s = (s_1, \ldots, s_k)^\top \in \mathrm{Range}(\mathcal{A})$.

$$ \frac{p(s)}{p'(s)} = \prod_{i=1}^{k} \frac{p_i(s_i)}{p_i'(s_i)} $$
$$ = \prod_{i \in I} \frac{p_i(s_i)}{p_i'(s_i)} $$

$$= \prod_{i \in I} \frac{\exp\left(-\frac{\varepsilon |c_i - s_i|}{\Gamma}\right)}{\exp\left(-\frac{\varepsilon |c_i + \delta_i - s_i|}{\Gamma}\right)}$$

$$= \prod_{i \in I} \exp\left(\frac{\varepsilon}{\Gamma} \cdot \left(|c_i + \delta_i - s_i| - |c_i - s_i|\right)\right)$$

$$\leq \prod_{i \in I} \exp\left(\frac{\varepsilon}{\Gamma} \cdot |\delta_i|\right) \tag{2}$$

We use condition 3 and partition I into sets of indices $I_1, \ldots, I_x$ such that for all $j \in \{1, \ldots, x\}$ the p-sums corresponding to indices in I_j cover disjoint time intervals. For each set I_j we then have

$$\sum_{i \in I_j} |\delta_i| \leq \sum_{i \in I_j} \sum_{t=\text{start}(i)}^{\text{end}(i)} |\delta(t)| \leq \sum_{t=1}^{T} |\delta(t)| \leq \Gamma. \tag{3}$$

Combining (2) and (3) yields

$$\frac{p(s)}{p'(s)} \leq \prod_{i \in I} \exp\left(\frac{\varepsilon}{\Gamma} \cdot |\delta_i|\right)$$

$$= \prod_{j=1}^{x} \prod_{i \in I_j} \exp\left(\frac{\varepsilon}{\Gamma} \cdot |\delta_i|\right)$$

$$= \prod_{j=1}^{x} \exp\left(\frac{\varepsilon}{\Gamma} \cdot \sum_{j \in I_j} |\delta_i|\right)$$

$$\leq \prod_{j=1}^{x} \exp\left(\frac{\varepsilon}{\Gamma} \cdot \Gamma\right) = \exp(x \cdot \varepsilon).$$

Thus, $\mathcal{A}$ is $x \cdot \varepsilon$-differentially private, since

$$\Pr[\mathcal{A}(\sigma) \in S] = \int_S p(s) ds \leq \int_S e^{x \cdot \varepsilon} p'(s) ds = e^{x \cdot \varepsilon} \Pr[A(\sigma') \in S]$$

for any $S \subseteq \text{Range}(\mathcal{A})$. This concludes the proof of Lemma 13. ◀

We can compute the p-sums in Algorithm 3 as in the binary mechanism [6] to release ε-differentially private estimates of graph functions.

▶ **Corollary 14** (Binary mechanism). *Let f be a graph function whose difference sequence has continuous global sensitivity Γ. Let $0 < \delta < 1$ and $\varepsilon > 0$. For each $T \in \mathbb{N}$ there exists an ε-differentially private algorithm to estimate f on a graph sequence which has error $O(\Gamma \varepsilon^{-1} \cdot \log^{3/2} T \cdot \log \delta^{-1})$ with probability at least $1 - \delta$.*

Proof. We show that the binary mechanism by Chan et al. [6] can be employed to construct an algorithm as in Algorithm 3.

The binary mechanism divides the T time steps of the input sequence into $\lfloor T/2^i \rfloor$ intervals of length 2^i for each $i = 0, \ldots, \log T$. For each interval the binary mechanism computes a p-sum. The p-sum of length 2^i used to compute the output at time step t is indexed as

$$q_i(t) = \sum_{j=0}^{i-1} \left\lfloor \frac{T}{2^i} \right\rfloor + \left\lfloor \frac{t}{2^i} \right\rfloor.$$
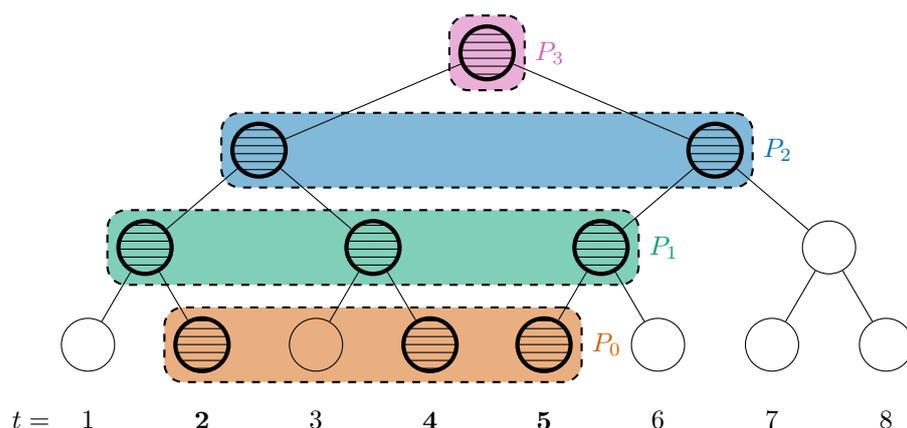
Figure 1 For a stream of length 8 the binary mechanism computes p-sums as shown. The leaves of the tree correspond to p-sums over the individual time steps; the root corresponds to a p-sum over the full stream. Now assume that in Algorithm 3 Δf differs at time steps 2, 4 and 5 (marked in bold) for adjacent graph sequences. The p-sums involving these time steps (shaded circles) can be partitioned into 4 sets P_0, P_1, P_2, P_3, indicated by dashed rectangles. P_i contains the p-sums of length 2^i.

Let $a = (a_1, \ldots, a_k)$ be the noisy p-sums of the difference sequence of a graph function f for any graph sequence of length T. The value $f(t)$ at time t is computed as the sum

$$\sum_{i=0}^{\log T} \text{bin}_i(t) \cdot a_{q_i(t)},$$

where $\text{bin}_i(t)$ is the i-th bit in the binary representation of t, with $i = 0$ for the least significant bit.

We have $y = \log T$, since the sum at any time t is computed from at most $\log T$ noisy p-sums corresponding to the at most $\log T$ bits of the binary representation of t. Every time step t participates in p-sums of length 2^i for all $i = 0, \ldots, \log T$. Thus, $x = \log T + 1$. The set P of all p-sums can be partitioned into x subsets of disjoint p-sums as illustrated in Figure 1: for each $i = 0, \ldots, \log T$ the set P_i contains the p-sums of length exactly 2^i. Thus, using the binary mechanism to sum the noisy p-sums we obtain an algorithm that satisfies the conditions of Lemma 13.

To obtain an ε-differentially private algorithm we set the privacy parameter in Lemma 13 to ε/x. The error is then $O(\Gamma \varepsilon^{-1} \cdot x \cdot \sqrt{y} \cdot \log \delta^{-1})$ with probability at least $1 - \delta$. The claim follows from $x, y = O(\log T)$. ◀

3.3 Bounds on Continuous Global Sensitivity

Song et al. [29] give bounds on the continuous global sensitivity of the difference sequence for several graph functions in the incremental setting in terms of the maximum degree D. Table 3 summarizes the results on the continuous global sensitivity of difference sequences for a variety of problems in the partially dynamic and fully dynamic setting, both for edge- and node-adjacency. Note that bounds on the continuous global sensitivity of the difference sequence for incremental graph sequences hold equally for decremental graph sequences as for every incremental graph sequence there exists an equivalent decremental graph sequence which deletes nodes and edges in the reverse order.

■ **Table 3** Global sensitivity of difference sequences

Graph Function f	Continuous Global Sensitivity of Δf			
	Partially Dynamic		Fully Dynamic	
	node-adjacency	edge-adjacency	node-adj.	edge-adj.
edge count[a]	D	1	$\geq T$	2
high-degree nodes[a]	$2D+1$	4	$\geq T$	$\geq T$
degree histogram[a]	$4D^2 + 2D + 1$	$8D$	$\geq 2T$	$\geq 2T$
triangle count[a]	$\binom{D}{2}$	D	$\geq T$	$\geq T$
k-star count[a]	$D\binom{D-1}{k-1} + \binom{D}{k}$	$2 \cdot \left(\binom{D}{k} - \binom{D-1}{k}\right)$	$\geq T$	$\geq T$
minimum spanning tree	$2DW$	$2W - 2$	$\geq T$	$\geq T$
minimum cut	$\geq T$	$\geq T$	$\geq T$	$\geq T$
maximum matching	$\geq T$	$\geq T$	$\geq T$	$\geq T$

[a]Bounds for partially dynamic node-adjacency from Song et al. [29]

In the partially dynamic setting, the continuous global sensitivity based approach works well for graph functions that can be expressed as the sum of local functions on the neighborhood of nodes. For non-local problems the approach is less successful. For the weight of a minimum spanning tree the continuous global sensitivity of the difference sequence is independent of the length of the graph sequence. However, for minimum cut and maximum matching this is not the case. In the fully-dynamic setting the approach seems not to be useful. Here, we can show that even for estimating the number of edges the continuous global sensitivity of the difference sequence scales linearly with T under node-adjacency. When considering edge-adjacency we only have low sensitivity for the edge count.

Using Corollary 14 we obtain ε-differentially private mechanisms with additive error that scales with $\log^{3/2} T$, compared to the factor $\sqrt{T}$ in [29]. Note that we recover their algorithm when using the Simple Mechanism II by Chan et al. [6] to sum the difference sequence.

For edge-adjacency we obtain the following bounds on the sensitivity of the difference sequence.

▶ **Lemma 15.** *When considering edge-adjacency in the incremental setting, the continuous global sensitivity of the difference sequence of the*
1. *number of high-degree nodes is 4;*
2. *degree histogram is $8D$;*
3. *edge count is 1;*
4. *triangle count is D;*
5. *k-star count is $2 \cdot \left(\binom{D}{k} - \binom{D-1}{k}\right)$.*

Proof. In the following, $\mathcal{G}$ and $\mathcal{G}'$ are edge-adjacent incremental graph sequences that differ in the insertion of an edge $e^* = \{u^*, v^*\}$.

1. In $\mathcal{G}$, the endpoints of e^* may cross the degree-threshold τ as soon as e^* is inserted, increasing the number of high-degree nodes by 2. In $\mathcal{G}'$, the same nodes may become high-degree nodes at a later update, increasing the number of high-degree nodes by 1 each. Thus, the difference sequences differ by at most 4.
2. The degrees of u^* and v^* are one less in $\mathcal{G}'$ compared to $\mathcal{G}$, once e^* is inserted. There are up to D insertions of edges incident to u^*, v^*. For each of these updates, the increase in the degree of u^* affects 2 histogram-bins in $\mathcal{G}$ and $\mathcal{G}'$, each. The same holds for the

increase in the degree of v^*. Thus, the continuous global sensitivity of the difference sequence of the degree histogram is $8D$.

3. The edge-update sets ∂E_t^+ and $\partial E_t^{+\prime}$ have exactly the same size at all times, except when e^* is inserted. Thus, the global sensitivity of the difference sequence the edge count is the number of edges that are inserted into $\mathcal{G}$, but not into $\mathcal{G}'$, which is 1.

4. In $\mathcal{G}'$, the u^* and v^* are not part of shared triangles. In $\mathcal{G}$ however, they may become part of up to D triangles. Thus, the continuous global sensitivity of the difference sequence of the triangle count is D.

5. In $\mathcal{G}$, after inserting e^*, u^* and v^* may be the center of up to $\binom{D}{k}$-many k-stars. Then, they were the center of $\binom{D-1}{k}$-many k-stars before the insertion of e^*. Any k-star in $\mathcal{G}'$ is also present in $\mathcal{G}$. Thus, the continuous global sensitivity of the difference sequence of the k-star count is the number of k-stars that the insertion of e^* can contribute, which is $2\left(\binom{D}{k} - \binom{D-1}{k}\right)$. ◄

The following lemmata provide evidence that the difference sequence based approach is not useful for fully dynamic graphs.

▶ **Lemma 16.** *Let $f_\tau((V,E)) = |\{v \in V \mid \deg(v) \geq \tau\}|$. The difference sequence of f_τ has continuous global sensitivity at least T when considering edge-adjacent or node-adjacent fully dynamic graph sequences.*

Proof. For the edge-adjacency case consider two graph sequences $\mathcal{G}$, $\mathcal{G}'$, where the initial graph is a τ-star with two edges e_1, e_2 removed. At time $t = 1$ insert e_1 into $\mathcal{G}$, but not into $\mathcal{G}'$. At odd times, insert e_2 into both sequences; at even times delete e_2 from both sequences. The sequence $\mathcal{G}'$ never has a node of degree at least τ, whereas $\mathcal{G}$ has one such node at odd, but not at even time steps. Thus, $\sum_{t=1}^{T} |\Delta f_{\tau,\mathcal{G}}(t) - \Delta f_{\tau,\mathcal{G}'}(t)| = T$.

For the node-adjacency case we again consider two graph sequences $\tilde{\mathcal{G}}$, $\tilde{\mathcal{G}}'$, where the initial graph is a τ-star with two nodes a, b and their incident edges removed. At time $t = 1$ insert a and its incident edge into $\tilde{\mathcal{G}}$, but not into $\tilde{\mathcal{G}}'$. At odd times, insert b and its incident edge into both sequence; at even times delete b from both sequences. As in the edge-adjacency case it follows that $\sum_{t=1}^{T} |\Delta f_{\tau,\tilde{\mathcal{G}}}(t) - \Delta f_{\tau,\tilde{\mathcal{G}}'}(t)| = T$. ◄

▶ **Lemma 17.** *The difference sequence of the degree histogram has continuous global sensitivity at least $2T$ when considering edge-adjacent or node-adjacent fully dynamic graph sequences.*

Proof. We refer to the adjacent graph sequences introduced in the proof of Lemma 16. In one of the graph sequences, the bins in the degree histogram corresponding to degrees $\tau - 1$ and τ are alternatingly zero and one. In the other graph sequence the same is true for the bins corresponding to degrees $\tau - 2$ and τ. Let $h_\mathcal{G}(t)$, $h_{\mathcal{G}'}(t)$ be the degree histograms of the former and latter sequence at time t, respectively. The corresponding difference sequences are $\Delta h_\mathcal{G}(t) = (d_{t,0}, \ldots, d_{t,n-1})$ and $\Delta h_{\mathcal{G}'}(t) = (d'_{t,0}, \ldots, d'_{t,n-1})$, where $d_{t,i}$ and $d'_{t,i}$ is the change in the number of nodes of degree i from time $t-1$ to t. For odd t, we have $d_{t,\tau} = 1$, $d_{t,\tau-1} = -1$, $d'_{t,\tau-1} = 1$, $d'_{t,\tau-2} = -1$; for even t, we have $d_{t,\tau} = -1$, $d_{t,\tau-1} = 1$, $d'_{t,\tau-1} = -1$, $d'_{t,\tau-2} = 1$. Note that $d_{t,1} = d'_{t,1}$ for $t \geq 2$ and $d_{1,1} = 2 = d_{1,1'} + 1$. All $d_{t,i}$ not explicitly given are 0. It follows that $\sum_{t=1}^{T} ||\Delta h_\mathcal{G}(t) - \Delta h_{\mathcal{G}'}(t)||_1 \geq 2T$, which concludes the proof. ◄

▶ **Lemma 18.** *Let H be a connected graph with at least two edges (nodes) and $f_H(G)$ be the function counting occurrences of H in G. When considering edge-adjacent (node-adjacent) fully dynamic graph sequences of length T, $\mathrm{GS}(\Delta f_H) \geq T$.*

Proof. To show the edge-adjacency case consider adjacent graph sequences $\mathcal{G}$, $\mathcal{G}'$ of length T, where the initial graph is a copy of H with two edges, say $\{a, b\}$, $\{c, d\}$, removed. In the first time step ($t = 1$), insert $\{a, b\}$ into $\mathcal{G}$, but not into $\mathcal{G}'$ and insert $\{c, d\}$ into both sequences. Thus, we have $f_H(G_1) = 1$ and $f_H(G_1') = 0$. In time steps where t is even, delete $\{c, d\}$ from both sequences; if t is odd, insert $\{c, d\}$ into both sequences. At all time steps t we have $f_H(G_t') = 0$ and $|\Delta f_{H,\mathcal{G}}(t)| = 1$, which implies $\sum_{t=1}^{T} |\Delta f_{H,\mathcal{G}}(t) - \Delta f_{H,\mathcal{G}'}(t)| = T$.

For the node-adjacency case consider again adjacent graph sequences $\tilde{\mathcal{G}}$, $\tilde{\mathcal{G}}'$ of length T, where the initial graph is a copy of H with two nodes, say a, b, removed. At time step $t = 1$ insert a along with the incident edges into $\tilde{\mathcal{G}}$, but not $\tilde{\mathcal{G}}'$. At every odd time step, insert b along with the incident edges into both sequences; at every even time step delete b from both sequences. Again, $\tilde{\mathcal{G}}'$ never contains a copy of H, whereas $\tilde{\mathcal{G}}$ contains a copy of H at every odd time step, but no on the even time steps. Thus, $\sum_{t=1}^{T} |\Delta f_{H,\tilde{\mathcal{G}}}(t) - \Delta f_{H,\tilde{\mathcal{G}}'}(t)| = T$. ◄

▶ **Corollary 19.** *The difference sequence of triangle count and k-star count has continuous global sensitivity $\geq T$ when considering edge-adjacent or node-adjacent graph sequences. The difference sequence of the edge count has continuous global sensitivity $\geq T$ when considering node-adjacent graph sequences.*

Proof. The claim follows from Lemma 18, by observing that triangles and k-stars have at least two edges and nodes, and noting that for the edge count $H = (\{a, b\}, \{\{a, b\}\})$. ◄

▶ **Lemma 20.** *The difference sequence of the weight of a minimum spanning tree has continuous global sensitivity at least T when considering edge-adjacent or node-adjacent fully dynamic graph sequences.*

Proof. Let $A = (V_A, E_A)$ and $B = (V_B, E_B)$ be connected graphs with $|V_A|, |V_B| \geq 3$ and $w(e) = 1$ for all $e \in E_A \cup E_B$. We denote nodes in V_A by a_i, nodes in V_B by b_i with index $i = 0, 1, \ldots$. Let $G_0 = (V_A \cup V_B, E_A \cup E_B \cup \{\{a_0, b_0\}\})$ with $w(\{a_0, b_0\}) = W \geq 3$. Let $\mathcal{G}$, $\mathcal{G}'$ be graph sequences of length T with initial graph G_0.

At time $t = 1$ we insert the edge $e^* = \{a_1, b_1\}$ with weight $w(e^*) = 1$ into $\mathcal{G}$. Then, if t is even, we insert the edges $e' = \{a_2, b_2\}$ with weight $w(e') = W - 1$ into $\mathcal{G}$ and $\mathcal{G}'$. If t is odd, we delete e' from both sequences.

The sequences $\mathcal{G}$ and $\mathcal{G}'$ are edge-adjacent. The weight of the minimum spanning tree in G_0 is $w_0 = |V_A| - 1 + |V_B| - 1 + W = |V_A| + |V_B| + W - 2$. For all $t \geq 1$, the weight of the minimum spanning tree in $\mathcal{G}$ is $w_{\mathcal{G}} = 1 + |V_A| - 1 + |V_B| - 1 = |V_A| + |V_B| - 1$. If t is even and $t \geq 2$, then the weight of the minimum spanning tree in $\mathcal{G}'$ is $w_0 - 1$; if t is odd it is w_0. Thus, we have

$$|\Delta w_{\mathrm{MST}\mathcal{G}}(t) - \Delta w_{\mathrm{MST}\mathcal{G}'}(t)| \geq 1$$

for all $t = 1, \ldots, T$, where $w_{\mathrm{MST}\mathcal{G}}$ and $w_{\mathrm{MST}\mathcal{G}'}$ denote the weight of the minimum spanning tree in $\mathcal{G}$ and $\mathcal{G}'$, respectively. Summing over all t proves the claim for edge-adjacent sequences.

To show that the bound holds for node-adjacent sequences, we replace the edge-insertions by node-insertions as follows: instead of e^* we insert a node v^* along with edges $e_1^* = \{a_1, v^*\}$ and $e_2^* = \{v^*, b_1\}$, where $w(e_1^*) = w(e_2^*) = 1$; instead of e' we insert a node v' along with edges $e_1' = \{a_2, v'\}$ and $e_2' = \{v', b_1\}$, where $w(e_1') = 1$ and $w(e_2') = W - 1$; where we delete e' in the edge-adjacency case we delete v' and the incident edges. We use the same initial graph G_0 and obtain node-adjacent graph sequences where the weights of the minimum spanning tree are increased by 1 compared to the weights in the corresponding node-adjacent graph sequences. The claim for node-adjacent graph sequences follows as for edge-adjacent graph sequences. ◄

For maximum cardinality matching and minimum cut the continuous global sensitivity of the difference sequence scales linearly with the length of the input even in the partially dynamic setting.

▶ **Lemma 21.** *Let $M(G)$ be the size of a maximum cardinality matching of G. The difference sequence of M has continuous global sensitivity at least T when considering edge-adjacent or node-adjacent partially dynamic graph sequences.*

Proof. First, we construct edge-adjacent graph sequences $\mathcal{G}, \mathcal{G}'$, where $|\Delta M_{\mathcal{G}}(t) - \Delta M_{\mathcal{G}'}(t)| = 1$ for all t. Let $V = \{0, \ldots, T\}$. At time $t = 1$ insert the edge $\{0, 1\}$ into $\mathcal{G}$ and insert no edge into $\mathcal{G}'$. $\Delta M_{\mathcal{G}}(1) = 1$, since G_1 has a maximum cardinality matching of size 1. $\Delta M_{\mathcal{G}'}(1) = 0$, since G_1' has no edges. At every time $t > 1$ insert the edge $\{t - 1, t\}$ into both $\mathcal{G}$ and $\mathcal{G}'$. If t is even, then $\Delta M_{\mathcal{G}}(t) = 0$ and $\Delta M_{\mathcal{G}'}(t) = 1$. If t is odd, then $\Delta M_{\mathcal{G}}(t) = 1$ and $\Delta M_{\mathcal{G}'}(t) = 0$. Thus, we have $\sum_{t=1}^{T} |\Delta M_{\mathcal{G}}(t) - \Delta M_{\mathcal{G}'}(t)| = T$.

For node-adjacency, both graph sequences start with the node 1. In $\mathcal{G}$, we insert node 0 along with the edge $\{0, 1\}$ at time 1. Then, at every time step $t > 1$ we insert node t and the edge $\{t, t - 1\}$ into both graph sequences. The difference sequence is the same as in the edge-adjacency case. Thus, the continuous global sensitivity of the difference sequence is also at least T.   ◀

▶ **Lemma 22.** *Let $w_{\mathrm{CUT}}(G)$ be the weight of a minimum cut of G. The difference sequence of w_{CUT} has continuous global sensitivity at least T when considering edge-adjacent or node-adjacent partially dynamic graph sequences.*

Proof. We show that for any $T \in \mathbb{N}$ there exist edge-adjacent graph sequences $\mathcal{G}, \mathcal{G}'$, such that $\sum_{t=1}^{T} |\Delta w_{\mathrm{CUT}\mathcal{G}}(t) - \Delta w_{\mathrm{CUT}\mathcal{G}'}(t)| \geq T$.

Let $A = (V_A, E_A)$, $B = (V_B, E_B)$, $C = (V_C, E_C)$ be graphs on at least $T + 2$ nodes with minimum cut of weight at least $W \cdot (T+1)$. We denote nodes in V_A by a_i, nodes in V_B by b_i and nodes in V_C by c_i, with index $i = 0, 1, \ldots$. Let $G_0 = (V_A \cup V_B \cup V_C, E_A \cup E_B \cup E_C \cup \{e_{ab}, e_{bc}\}$, where $e_{ab} = \{a_{T+1}, b_{T+1}\}$ and $e_{bc} = \{b_{T+1}, c_{T+1}\}$ with $w(e_{ab}) = 1$ and $w(e_{bc}) = 2$. Let $\mathcal{G}$, $\mathcal{G}'$ be graph sequences with initial graph G_0.

At time $t = 1$ we insert the edge $e^* = \{a_0, b_0\}$ with $w(e^*) = W$ into $\mathcal{G}$. Then, if t is even, we insert the edge $e_t = \{a_t, b_t\}$ with $w(e_t) = W$ into both $\mathcal{G}$ and $\mathcal{G}'$; If t is odd, we insert the edge $e_t = \{b_t, c_t\}$ with $w(e_t) = W$ into both $\mathcal{G}$ and $\mathcal{G}'$.

These graph sequences are edge-adjacent since they differ only in the insertion of e^*. At any time t the minimum cut in G_t is $(V_A \cup V_B, V_C)$. In $\mathcal{G}'$, the minimum cut is $(V_A, V_B \cup V_C)$ for odd t and $(V_A \cup V_B, V_C)$ for even t.

In $\mathcal{G}$, the value of the minimum cut increases by 1 at time $t = 1$ due to insertion of e^* and by W for odd $t \geq 3$, since an edge of weight W across the minimum cut is inserted. That is,

$$\Delta w_{\mathrm{CUT}\mathcal{G}}(t) = \begin{cases} 1 & \text{if } t = 1, \\ 0 & \text{if } t \text{ is even}, \\ W & \text{if } t \text{ is odd and } t \geq 3. \end{cases} \tag{4}$$

In $\mathcal{G}'$, the value of the minimum cut does not change at time $t = 1$, since no edge is inserted. At even times, the cut switches to the edges between V_B and V_C and the weight of the minimum cut increases by 1. At odd times, the cut switches to the edges between V_A and V_B, and the weight of the minimum cut increases by $W - 1$. The difference sequence for

$\mathcal{G}'$ is thus,

$$\Delta w_{\mathrm{CUT}\mathcal{G}'}(t) = \begin{cases} 0 & \text{if } t = 1, \\ 1 & \text{if } t \text{ is even,} \\ (W-1) & \text{if } t \text{ is odd and } t \geq 3. \end{cases} \tag{5}$$

From (4) and (5) we get $|\Delta w_{\mathrm{CUT}\mathcal{G}}(t) - \Delta w_{\mathrm{CUT}\mathcal{G}'}(t)| = 1$ for all t. Summing over all t we get the proposed bound on the continuous global sensitivity.

We obtain the same bound for node-adjacent partially dynamic graph sequences by modifying the above construction as follows: we start with the same initial graph G_0 as above. We replace an insertion of the edge $\{a_i, b_i\}$ by an insertion of node a_i' along with edges $\{a_i', b_i\}$ and edges $\{a_i', a_j\}$ for all $a_j \in V_A$; all these edges have weight W. Similarly, we replace an insertion of the edge $\{b_i, c_i\}$ by an insertion of node c_i' along with edges $\{c_i', b_i\}$ and edges $\{c_i', c_j\}$ for all $c_j \in V_C$; all these edges have weight W. By counting the nodes a_i' and c_i' to V_A and V_C, respectively, we obtain the same sequence of minimum cuts as in the edge-adjacency case. Thus, we get the same lower bound on global sensitivity for the node-adjacency case. ◀

The difference sequence approach can be employed to privately estimate the weight of a minimum spanning tree in partially dynamic graph sequences. If the edge weight is bounded by W, then the continuous global sensitivity of the difference sequence is $O(W)$ and $O(DW)$ under edge-adjacency and node-adjacency, respectively. In Theorems 23 and 24 we prove these bounds and use Corollary 14 to derive edge- and node-differentially private algorithms.

▶ **Theorem 23.** *There exists an ε-edge-differentially private algorithm that outputs the weight of a minimum spanning tree on an incremental graph sequence $\mathcal{G}$ with edge-weights from the set $\{1, \ldots, W\}$. At every time step, the algorithm has error $O(W\varepsilon^{-1} \cdot \log^{3/2} T \cdot \log \delta^{-1})$ with probability at least $1 - \delta$, where T is the length of the graph sequence.*

Proof. We will show that the difference sequence for the weight of a minimum spanning tree has global edge-sensitivity $2W - 2$ in the incremental setting. The claim then follows by Corollary 14.

Let $\mathcal{G} = (G_1, \ldots, G_T)$, $\mathcal{G}' = (G_1', \ldots, G_T')$ be two adjacent graph sequences with initial graphs G_0 and G_0'. We are considering edge-differential privacy and thus edge-adjacency (Definition 5). This means that $\mathcal{G}$ and $\mathcal{G}'$ differ in the insertion of an edge e^*. Without loss of generality, we assume that e^* is inserted into $\mathcal{G}$. For estimating $\mathrm{GS}(\Delta w_{\mathrm{MST}})$ we can further assume that e^* is inserted into G_0 to obtain G_1, since $\Delta w_{\mathrm{MST}\mathcal{G}}(t) = \Delta w_{\mathrm{MST}\mathcal{G}'}(t)$ for all times t before the insertion of e^*. For all $t = 0, \ldots, T$ the edge set of G_t is a superset of the edge set of G_t'. This implies $w_{\mathrm{MST}}(G_t) \leq w_{\mathrm{MST}}(G_t')$ for all t.

Note that $w_{\mathrm{MST}}(G_0) = w_{\mathrm{MST}}(G_0') = w_{\mathrm{MST}}(G_1')$, which implies $\Delta w_{\mathrm{MST}}\mathcal{G}'(1) = 0$.

As e^* might either not replace any edge in a minimum spanning tree or might replace an edge of weight W, it holds that $0 \leq \Delta w_{\mathrm{MST}}\mathcal{G}'(1) \leq W - 1$. Thus, it follows that $0 \leq \Delta w_{\mathrm{MST}\mathcal{G}}(1) - \Delta w_{\mathrm{MST}\mathcal{G}'}(1) \leq W - 1$.

It remains to bound $\sum_{t=2}^{T} |\Delta w_{\mathrm{MST}\mathcal{G}}(t) - \Delta w_{\mathrm{MST}\mathcal{G}'}(t)|$. Any edge inserted into $\mathcal{G}$ is also inserted into $\mathcal{G}'$. If inserting an edge into $\mathcal{G}$ at time t reduces the weight of the minimum spanning tree of $\mathcal{G}$ at time t, then the weight of the minimum spanning tree in $\mathcal{G}'$ at time t is reduced by at least the same amount. Thus there are two cases: (1) the weight of the minimum spanning tree in $\mathcal{G}$ and $\mathcal{G}'$ change by the same amount; (2) the weight of the minimum spanning tree in $\mathcal{G}'$ is reduced more than the weight of a minimum spanning tree in $\mathcal{G}$. It follows that for all $t > 0$, $\Delta w_{\mathrm{MST}\mathcal{G}}(t) - \Delta w_{\mathrm{MST}\mathcal{G}'}(t) \geq 0$. Thus, $\sum_{t=2}^{T} |\Delta w_{\mathrm{MST}\mathcal{G}}(t) - \Delta w_{\mathrm{MST}\mathcal{G}'}(t)|$

$= \sum_{t=2}^{T} \Delta w_{\mathrm{MST}\mathcal{G}}(t) - \Delta w_{\mathrm{MST}\mathcal{G}'}(t) = \sum_{t=2}^{T} \Delta w_{\mathrm{MST}\mathcal{G}}(t) - \sum_{t=2}^{T} \Delta w_{\mathrm{MST}\mathcal{G}'}(t)$. But then it follows that

$$\sum_{t=2}^{T} \Delta w_{\mathrm{MST}\mathcal{G}}(t) - \sum_{t=2}^{T} \Delta w_{\mathrm{MST}\mathcal{G}'}(t) = w_{\mathrm{MST}}(G_T) - w_{\mathrm{MST}}(G_1) - w_{\mathrm{MST}}(G_T') + w_{\mathrm{MST}}(G_1') \leq W - 1,$$

where the last inequality follows from $w_{\mathrm{MST}}(G_1') - w_{\mathrm{MST}}(G_1) \leq W - 1$ and $w_{\mathrm{MST}}G_T \leq w_{\mathrm{MST}}G_T'$.

Combining the results for $t = 1$ and $t > 1$, we get

$$\sum_{t=1}^{T} |\Delta w_{\mathrm{MST}\mathcal{G}}(t) - \Delta w_{\mathrm{MST}\mathcal{G}'}(t)| \leq 2W - 2,$$

which implies $\mathrm{GS}(\Delta w_{\mathrm{MST}}) \leq 2W - 2$.

We now show that this bound on the sensitivity is tight. Let $A = (V_A = \{a_1, \dots\}, E_A)$, $B = (V_B = \{b_1, \dots\}, E_B)$ be arbitrary connected graphs with $|A|, |B| > W + 1$ and $w(e) = 1$ for all $e \in E_A \cup E_B$. Let $G = (V_A \cup V_B, E_A \cup E_B \cup \{a_1, b_1\})$ and $w(\{a_1, b_1\}) = W$. Note that the edge $\{a_1, b_1\}$ is guaranteed to be in any minimum spanning tree.

We construct adjacent graph sequences $\mathcal{G}_a = (G_{a1}, G_{a2}, G_{a3})$, $\mathcal{G}_b = (G_{b1}, G_{b2}, G_{b3})$ with $G_{a1} = G_{b1} = G$, such that $\sum_{t=1}^{|\mathcal{G}_a|} |\Delta w_{\mathrm{MST}\mathcal{G}_a}(t) - \Delta w_{\mathrm{MST}\mathcal{G}_b}(t)| = 2W - 2$.

Starting with $G_{a1} = G$ we insert the edge $\{a_2, b_2\}$ with $w(\{a_2, b_2\}) = 1$ into $\mathcal{G}_a$ to obtain G_{a2}. In $\mathcal{G}_b$ we insert no edges, i.e., $G_{b2} = G_{b1} = G$. Inserting the unit-weight edge into $\mathcal{G}_a$ reduces the weight of a minimum spanning tree by $W - 1$. In $\mathcal{G}_b$, the weight of a minimum spanning tree does not change. Thus, we have $|\Delta w_{\mathrm{MST}\mathcal{G}_a}(2) - \Delta w_{\mathrm{MST}\mathcal{G}_b}(2)| = W - 1$.

In the subsequent time step we insert the edge $\{a_3, b_3\}$ with $w(\{a_3, b_3\}) = 1$ into $\mathcal{G}_a$ and $\mathcal{G}_b$, which yields the graphs G_{a3} and G_{b3}, respectively. This does not change the weight of the minimum spanning tree in $\mathcal{G}_a$, but reduces the weight of the minimum spanning tree in $\mathcal{G}_b$ by $W - 1$, since $\{a_3, b_3\}$ replaces $\{a_1, b_1\}$ in the minimum spanning tree. Thus, we have $|\Delta w_{\mathrm{MST}\mathcal{G}_a}(3) - \Delta w_{\mathrm{MST}\mathcal{G}_b}(3)| = W - 1$.

This concludes the proof. ◄

▶ **Theorem 24.** *There exists an ε-node-differentially private algorithm that outputs the weight of a minimum spanning tree on an incremental graph sequence $\mathcal{G}$ with edge-weights from the set $\{1, \dots, W\}$. At every time step, the algorithm has error $O(DW\varepsilon^{-1} \cdot \log^{3/2} T \cdot \log \delta^{-1})$ with probability at least $1 - \delta$, where T is the length of the graph sequence and D is the maximum degree.*

Proof. We will show that the difference sequence for the weight of a minimum spanning tree has continuous global sensitivity $2DW$ under node-adjacency in the incremental setting. The claim then follows by Corollary 14.

Let $\mathcal{G} = (G_1, \dots, G_T)$, $\mathcal{G}' = (G_1', \dots, G_T')$ be two adjacent graph sequences with initial graphs G_0 and G_0'. We are considering node-differential privacy and thus node-adjacency (Definition 6). This means that $\mathcal{G}$ and $\mathcal{G}'$ differ in the insertion of a node v^*. Without loss of generality, we assume that v^* is inserted into $\mathcal{G}$. For estimating $\mathrm{GS}(\Delta w_{\mathrm{MST}})$ we can further assume that v^* is inserted into G_0 to obtain G_1, since $\Delta w_{\mathrm{MST}\mathcal{G}}(t) = \Delta w_{\mathrm{MST}\mathcal{G}'}(t)$ for all times t before the insertion of v^*.

The node v^* has degree at most D, so we have $|\Delta w_{\mathrm{MST}\mathcal{G}}(1) - \Delta w_{\mathrm{MST}\mathcal{G}'}(1)| \leq DW$.

We now bound $\sum_{t=2}^{T} |\Delta w_{\mathrm{MST}\mathcal{G}}(t) - \Delta w_{\mathrm{MST}\mathcal{G}'}(t)| =: \Gamma_2$. Let E^* be the set of edges incident to v^*. Note that $|E^*| \leq D$. For each edge $e \in E^*$ there are three cases: (1) e is

not in the minimum spanning tree when it is inserted; (2) e replaces a heavier edge; (3) e increases the weight of the minimum spanning tree.

In case (1), there always exists a minimum spanning tree that does not contain e, since no edges are ever deleted. Thus, e does not contribute to the continuous global sensitivity of the difference sequence. To analyze case (2), let e_H be the edge replaced by e. This edge may be replaced in $\mathcal{G}'$ at times different from the time at which e was inserted. However, this contributes at most $w(e_H) \leq W$ to Γ_2. In case (3), the weight that e contributes to the minimum spanning tree in $\mathcal{G}$ can be reduced by at most $W - 1$. However, the edge used to replace e may increase the weight of a minimum spanning tree in $\mathcal{G}'$ by 1. Thus, e contributes up to W to Γ_2.

With $|E^*| \leq D$, we thus have $\Gamma_2 \leq DW$. This implies

$$
\begin{aligned}
\mathrm{GS}(\Delta w_{\mathrm{MST}}) &= \sum_{t=1}^{T} |\Delta w_{\mathrm{MST}\mathcal{G}}(t) - \Delta w_{\mathrm{MST}\mathcal{G}'}(t)| \\
&= |\Delta w_{\mathrm{MST}\mathcal{G}}(1) - \Delta w_{\mathrm{MST}\mathcal{G}'}(1)| + \Gamma_2 \\
&\leq 2DW.
\end{aligned}
$$
◀

4   Upper Bound for Monotone Functions

In Section 3.3, we show that privately releasing the difference sequence of a graph sequence does not lead to good error guarantees for partially dynamic problems like minimum cut. Intuitively, the reason is that even for neighboring graph sequences $\mathcal{G}, \mathcal{G}'$, the differences of the difference sequence can be non-zero for all graphs G_i, G_i'. In other words, the difference of objective values for the graphs G_i and G_i' can constantly fluctuate during continual updates. However, the difference of objective values, regardless of fluctuations, is *always small*. We show that, by allowing an arbitrarily small *multiplicative* error, we can leverage this fact for a broad class of partially dynamic problems. In particular, we prove that there exist ε-differentially private algorithms for all dynamic problems that are non-decreasing (or non-increasing) on all valid input sequences. This includes, e.g., minimum cut, maximum matching and densest subgraph on partially dynamic inputs. See Algorithm 4 for the details and Table 2 on page 4 for explicit upper bounds for applications. We state the result for monotonically increasing functions, but it is straightforward to adapt the algorithm to monotonically decreasing functions.

■ **Algorithm 4** Multiplicative error algorithm for monotone functions

---

1 Function Initialize$(D, \rho, \epsilon, r, \beta)$
2 $k_0 \leftarrow 0$
3   InitializeSvt $(D, \rho, \epsilon, \log_{1+\beta}(r))$ // see Algorithm 1
4 Function Process(f_i)
5 $k_i \leftarrow k_{i-1}$
6   **while** ProcessSvtQuery$(f_i, (1+\beta)^{k_i}) = \top$ **do**   // see Algorithm 1
7 $\big|$ $k_i \leftarrow k_i + 1$
8   **return** $(1+\beta)^{k_i}$

▶ **Theorem 25.** *Let $r > 0$ and let f be any monotonically increasing function on dynamic inputs (e.g., graphs) with range $[1, r]$ and static global sensitivity $\rho := \mathrm{GS}_{\mathrm{static}}(f)$. Let*

$\beta \in (0,1), \delta > 0$ *and let* $\alpha = 16 \log_{1+\beta}(r)\rho \cdot \ln(2T/\delta)/\varepsilon$. *There exists an* ε-*differentially private algorithm for computing* f *with multiplicative error* $(1 + \beta)$, *additive error* α *and failure probability* δ.

Proof. We reduce the dynamic problem to a problem that can be solved using the sparse vector technique: Assume that we are given the whole sequence of dynamic updates as a database D in advance. We will remove this assumption later.

Formally, let $D := \mathcal{G} = (G_1, \ldots, G_T)$ be the input to the dynamic problem and let f_i be the function that evaluates f on the i-th entry G_i (e.g., graph) of such a database. To initialize our algorithm, we run $\texttt{Initialize}(D, \rho, \epsilon, r, \beta)$. Then we call $\texttt{Process}(f_i)$ for all $f_i, i \in [T]$. We claim that the output of $\texttt{Process}(f_i)$ is an approximation to $f_i(D) = f(G_i)$ with multiplicative error $(1 + \beta)$, additive error α and failure probability δ over the coins of the whole algorithm, and that the algorithm is ϵ-differentially private.

Error. We condition on the event that all noises in Algorithm 1 are small, i.e., $|\zeta| < \alpha/4$ and, for all i, $|\nu_i| < \alpha/4$. It follows from the definition of the Laplace distribution and a union bound over all queries that this event occurs with probability at least $1 - e^{-\alpha\varepsilon_1/(4\rho)} - T \cdot e^{-\alpha\varepsilon_2/(8 \log_{1+\beta}(r)\rho)} \geq 1 - \delta$. Consider query $f_i(D)$ and let $f'_i := f_i(D) + \nu_i - \zeta$. After the while-loop of $\texttt{Process}(f_i)$, we know that $f_i(D) - \alpha/2 \leq (1 + \beta)^{k_i}$ and $f_i(D) + \alpha/2 \geq f_{i-1}(D) + \alpha/2 \geq (1 + \beta)^{k_i-1}$ due to the monotonicity of $f_1(D), \ldots, f_T(D)$. We analyze the return value of $\texttt{Process}(f_i)$ and show that it is a $(1+\beta)$ approximation with additive error α:

$$f_i(D) - \alpha \leq \left((1 + \beta)^{k_i} + \frac{\alpha}{2} \right) - \alpha \leq (1 + \beta)^{k_i}$$

$$(1 + \beta)f_i(D) + \alpha \geq (1 + \beta)\left((1 + \beta)^{k_i-1} - \frac{\alpha}{2} \right) + \alpha \geq (1 + \beta)^{k_i}.$$

Since $f(G_T) \leq r = (1 + \beta)^{\log_{1+\beta}(r)}$, it follows that the algorithm has a multiplicative error $(1 + \beta)$ and an additive error of at most α with probability at least $1 - \delta$.

Privacy. The privacy follows directly from Lemma 9 because the return value $(1 + \beta)^{k_i}$ is derived from the answers of Algorithm 1, which is ϵ-private, and public knowledge (i.e., the fact that $k_0 = 0$).

Removing the assumption. Observe that to answer query f_i on D, Algorithm 1 only needs access to G_i or, equivalently, access to the first i updates of the dynamic update sequence. In other words, G_i is read for the first time after answering query f_{i-1}. Therefore, we may ask all previous queries $f_j, j < i$, even if G_i is not available yet, e.g., when processing update j. For dynamic graph sequences and edge-adjacency or node-adjacency, any pair of same-length prefixes of two graph sequences $\mathcal{G}_1, \mathcal{G}_2$ is neighboring if $\mathcal{G}_1$ and $\mathcal{G}_2$ are neighboring (see Definitions 5 and 6). Therefore, privacy is not affected by making G_i only available to Algorithm 4 just before the i-th query, as required by the dynamic setting. ◄

5 Lower Bounds for Event-Level Privacy

In this section we show lower bounds for the error of edge-differentially private algorithms in the partially dynamic setting. We consider the problem of releasing the weight of a minimum spanning tree, the minimum cut and maximum weight matching. To derive the bounds we reduce differentially private counting in binary streams to these problems and apply a lower bound of Dwork et al. [12], which we restate here.

▶ **Theorem 26** (Lower bound for counting in binary streams [12]). *Any differentially private event-level algorithm for counting over T rounds must have error $\Omega(\log T)$ (even with $\varepsilon = 1$).*

Note that any lower bound for the incremental setting can be transferred to the decremental setting, using the same reductions but proceeding in reverse.

5.1 Minimum Spanning Tree

We show a lower bound for the error in ε-edge-differentially private estimation of the weight of a minimum spanning tree when the weights are bounded by W.

▶ **Lemma 27.** *For any pair of adjacent binary streams σ, σ' of length T there exists a pair of incremental edge-weighted graph sequences $\mathcal{G}$, $\mathcal{G}'$ that are edge-adjacent, such that the weights of minimum spanning trees in $\mathcal{G}$ and $\mathcal{G}'$ are*

$$w_{\mathrm{MST}}(G_t) = W \sum_{i=1}^{t} \sigma(i) + T$$

and

$$w_{\mathrm{MST}}(G_t') = W \sum_{i=1}^{t} \sigma'(i) + T,$$

respectively, at all time steps $t \in \{1, \ldots, T\}$, where edge weights are in $\{1, \ldots, W\}$.

Proof. We construct graph sequences on the node set

$$V = \{0, \ldots, T\} \cup \{T+1, \ldots, 2T+1\}. \tag{6}$$

The nodes $1, \ldots, T$ will form a path in these graph sequences. To achieve this we use the edge set $E_0 = \{\{i-1, i\} \mid i = 1, \ldots, T\}$ with weight 1 for each edge and initial the graph sequences with the graph $G_0 = (V, E_0)$. The edges in E form a minimum spanning tree of G_0 of weight T.

For any two adjacent binary streams σ, σ' we construct corresponding graph sequences $\mathcal{G}$, $\mathcal{G}'$ with initial graph G_0. At all times the node set of the graphs in $\mathcal{G}$ and $\mathcal{G}'$ is V, as defined in (6). We now define for each time t the sets of edge-insertions for $\mathcal{G}$ and $\mathcal{G}'$. Let $e_0^t = \{t, (t+2) \bmod T\}$ with $w(e_0^t) = W$ and $e_1^t = \{T+t, T+t+1\}$ with $w(e_1^t) = W$. At time t we insert

$$\partial E_t^+ = \begin{cases} \{e_0^t\} & \text{if } \sigma(t) = 0, \\ \{e_0^t, e_1^t\} & \text{if } \sigma(t) = 1, \end{cases}$$

$$\partial E_t^{+\prime} = \begin{cases} \{e_0^t\} & \text{if } \sigma'(t) = 0, \\ \{e_0^t, e_1^t\} & \text{if } \sigma'(t) = 1, \end{cases}$$

into $\mathcal{G}$ and $\mathcal{G}'$, respectively. Note that $\partial E_t^+ \neq \partial E_t^{+\prime}$ only if $\sigma(t) \neq \sigma'(t)$. Thus, $\mathcal{G}$ and $\mathcal{G}'$ only differ in the insertion of a single edge and are adjacent.

Inserting an edge e_0^t never changes the weight of a minimum spanning tree, since the edges e^i defined above make up a spanning tree of the nodes $\{1, \ldots, T\}$ and have lower weight. Inserting an edge e_1^t increases the weight of the minimum spanning tree by W. Thus, at any time t, the weights of the minimum spanning trees in $\mathcal{G}$ and $\mathcal{G}'$ are

$$w_{\mathrm{MST}}(G_t) = W \sum_{i=1}^{t} \sigma(i) + T$$

and

$$w_{\mathrm{MST}}(G_t') = W \sum_{i=1}^{t} \sigma'(i) + T,$$

respectively, which concludes the proof. ◀

▶ **Lemma 28.** *For any pair of adjacent binary streams σ, σ' of length T there exists a pair of incremental edge-weighted graph sequences $\mathcal{G}$, $\mathcal{G}'$ that are node-adjacent, such that the weights of minimum spanning trees in $\mathcal{G}$ and $\mathcal{G}'$ are*

$$w_{\mathrm{MST}}(G_t) = W \sum_{i=1}^{t} \sigma(i)$$

and

$$w_{\mathrm{MST}}(G'_t) = W \sum_{i=1}^{t} \sigma'(i),$$

respectively, at all time steps $t \in \{1, \ldots, T\}$, where edge weights are in $\{1, \ldots, W\}$.

Proof. Let $G_0 = (v_0, \emptyset)$. Let $\mathcal{G}$, $\mathcal{G}'$ be graph sequences with initial graph G_0.

At time t we insert the node sets

$$
\begin{aligned}
\partial V_t^+ &= \begin{cases} \{u_t\} & \text{if } \sigma(t) = 0, \\ \{u_t, v_t\} & \text{if } \sigma(t) = 1, \end{cases} \\
\partial V_t^{+\prime} &= \begin{cases} \{u_t\} & \text{if } \sigma'(t) = 0, \\ \{u_t, v_t\} & \text{if } \sigma'(t) = 1, \end{cases}
\end{aligned}
\tag{7}
$$

into $\mathcal{G}$ and $\mathcal{G}'$, respectively. When inserting v_t we also insert the edge $\{v_0, v_t\}$ with weight W. The nodes u_t stay isolated throughout the graph sequences. The weight of the minimum spanning tree is the number of nodes v_t that were inserted into the graph sequence, multiplied by the maximum edge weight W. That is, at any time t, the weights of the minimum spanning trees in $\mathcal{G}$ and $\mathcal{G}'$ are

$$w_{\mathrm{MST}}(G_t) = W \sum_{i=1}^{t} \sigma(i)$$

and

$$w_{\mathrm{MST}}(G'_t) = W \sum_{i=1}^{t} \sigma'(i)$$

respectively, which concludes the proof. ◀

▶ **Theorem 29.** *Any ε-edge-differentially private algorithm to compute the weight of a minimum spanning tree in an incremental graph sequence of length T must have additive error $\Omega(W \log T)$ when the edge weights are in $\{1, \ldots, W\}$, even with $\varepsilon = 1$.*

Proof. Suppose there exists an ε-edge-differentially private algorithm to compute the weight of a minimum spanning tree in an incremental graph sequence with error $o(W \log T)$ at all time steps. Using this algorithm and the reduction from Lemma 27, we can compute the sum of any binary stream with error $o(\log T)$ while preserving ε-differential privacy. This contradicts Theorem 26. ◀

▶ **Theorem 30.** *Any ε-node-differentially private algorithm to compute the weight of a minimum spanning tree in an incremental graph sequence of length T must have additive error $\Omega(W \log T)$ when the edge weights are in $\{1, \ldots, W\}$, even with $\varepsilon = 1$.*

Proof. Suppose there exists an ε-node-differentially private algorithm to compute the weight of a minimum spanning tree in an incremental graph sequence with error $o(W \log T)$ at all time steps. Using this algorithm and the reduction from Lemma 28, we can compute the sum of any binary stream with error $o(\log T)$ while preserving ε-differential privacy. This contradicts Theorem 26. ◀

5.2 Weighted Minimum Cut

We can prove a similar lower bound for edge-differentially private weighted minimum cut in incremental graph sequences. We again reduce differentially private counting in binary streams to differentially private minimum cut in incremental graph sequences.

▶ **Lemma 31.** *For any pair of adjacent binary streams σ, σ' of length T there exists a pair of incremental edge-weighted graph sequences $\mathcal{G}$, $\mathcal{G}'$ that are edge-adjacent, such that the weights of the minimum cut in $\mathcal{G}$ and $\mathcal{G}'$ are*

$$w_{\mathrm{CUT}}(G_t) = W \sum_{i=1}^{t} \sigma(i)$$

and

$$w_{\mathrm{CUT}}(G_t') = W \sum_{i=1}^{t} \sigma'(i)$$

respectively, at all time steps $t \in \{1, \dots, T\}$, where edge weights are in $\{1, \dots, W\}$.

Proof. At the core of our reduction is a $(T+1)$-edge-connected graph with sufficiently many missing edges. We use the $(T+1)$-dimensional hypercube graph $Q = (V_Q, E_Q)$. Each node $v \in V_Q$ is associated with a vector $\mathbf{v} \in \{0,1\}^{T+1}$. We associate every time step $t \in \{1, \dots, T\}$ with two nodes in Q, which correspond to the following vectors:

$$b_t = (b_1, \dots, b_T, 0)$$
$$\text{and} \quad \widehat{b}_t = (1 - b_1, \dots, 1 - b_T, 1),$$

where $b_1, \dots, b_T \in \{0,1\}$ such that $t = \sum_{i=1}^{T} b_i \cdot 2^{i-1}$. Since b_t and $\widehat{b}_t$ differ in at least 2 elements, they are not connected by an edge in Q.

In the following, every edge has weight W, including the edges in Q. Let $G_0 = (\{v^*\}, \emptyset) \cup Q$. G_0 has a minimum cut of weight 0, since v^* is isolated. Let $\mathcal{G}$, $\mathcal{G}'$ be incremental graph sequences with initial graph G_0. For every time step we define the edges $e_0^t = \{b_t, \widehat{b}_t\}$ and $e_1^t = \{v^*, b_t\}$. At time t we insert

$$\partial E_t^+ = \begin{cases} \{e_0^t\} & \text{if } \sigma(t) = 0, \\ \{e_0^t, e_1^t\} & \text{if } \sigma(t) = 1, \end{cases}$$

$$\partial E_t^{+\prime} = \begin{cases} \{e_0^t\} & \text{if } \sigma'(t) = 0, \\ \{e_0^t, e_1^t\} & \text{if } \sigma'(t) = 1, \end{cases}$$

into $\mathcal{G}$ and $\mathcal{G}'$, respectively. Note that these graph sequences differ in exactly one edge exactly when $\sigma(t) \neq \sigma'(t)$.

Denote by $\deg_{\mathcal{G}}(v^*, t)$ the degree of v^* in $\mathcal{G}$ at time t. We observe that the degree of v^* increases by 1 in the graph sequence if and only if there is a 1 in the corresponding binary

stream. Otherwise, its degree does not change. The minimum cut is always the trivial cut around v^*, since the degree of v^* never exceeds T and Q is $(T+1)$-edge-connected. Thus, the weight of the minimum cut at time t is

$$w_{\text{CUT}}(G_t) = W \sum_{i=1}^{t} \sigma(i)$$

and

$$w_{\text{CUT}}(G_t') = W \sum_{i=1}^{t} \sigma'(i),$$

which concludes the proof.   ◄

▶ **Lemma 32.** *For any pair of adjacent binary streams σ, σ' of length T there exists a pair of incremental edge-weighted graph sequences $\mathcal{G}$, $\mathcal{G}'$ that are node-adjacent, such that the weights of the minimum cut in $\mathcal{G}$ and $\mathcal{G}'$ are*

$$w_{\text{CUT}}(G_t) = W \sum_{i=1}^{t} \sigma(i)$$

and

$$w_{\text{CUT}}(G_t') = W \sum_{i=1}^{t} \sigma'(i)$$

respectively, at all time steps $t \in \{1, \ldots, T\}$, where edge weights are in $\{1, \ldots, W\}$.

Proof. Let $G_0 = (\{u_0, v_0\}, \emptyset)$. Let $\mathcal{G}$, $\mathcal{G}'$ be graph sequences with initial graph G_0. In the following all edges have weight W.

At time t we insert the node sets

$$\partial V_t^+ = \begin{cases} \{u_t\} & \text{if } \sigma(t) = 0, \\ \{u_t, v_t\} & \text{if } \sigma(t) = 1, \end{cases}$$

$$\partial V_t^{+'} = \begin{cases} \{u_t\} & \text{if } \sigma'(t) = 0, \\ \{u_t, v_t\} & \text{if } \sigma'(t) = 1, \end{cases}$$

into $\mathcal{G}$ and $\mathcal{G}'$, respectively. Along with every node u_t, we insert the edges $\{u_t, u_j\}$ for all $0 \le j < t$. If we insert v_t, we also insert the edges $\{v_t, v_j\}$ for all $0 \le j < t$ and the edge $\{v_t, u_0\}$.

Let $U_t = \{u_0, \ldots, u_t\}$ and $V_t = \{v_0, \ldots, v_t\}$. The subgraphs induced by U_t and V_t are always complete. The cut (U_t, V_t) always has minimum weight. The weight of this cut in $\mathcal{G}$ increases by W only if $\sigma(t) = 1$; otherwise, it does not change. The same holds for $\mathcal{G}'$ and σ'. Thus, the weight of the minimum cut at time t is

$$w_{\text{CUT}}(G_t) = W \sum_{i=1}^{t} \sigma(i)$$

and

$$w_{\text{CUT}}(G_t') = W \sum_{i=1}^{t} \sigma'(i),$$

which concludes the proof.   ◄

▶ **Theorem 33.** *Any ε-edge-differentially private algorithm to compute the weight of a minimum cut in an incremental graph sequence of length T must have additive error $\Omega(W \log T)$ when the edge weights are in $\{1, \ldots, W\}$, even with $\varepsilon = 1$.*

Proof. Suppose there exists an ε-edge-differentially private algorithm to compute the weight of a minimum cut in an incremental graph sequence with error $o(W \log T)$ at all time steps. Using this algorithm and the reduction from Lemma 31, we can compute the sum of any binary stream with error $o(\log T)$ while preserving ε-differential privacy. This contradicts Theorem 26. ◀

▶ **Theorem 34.** *Any ε-node-differentially private algorithm to compute the weight of a minimum cut in an incremental graph sequence of length T must have additive error $\Omega(W \log T)$ when the edge weights are in $\{1, \ldots, W\}$, even with $\varepsilon = 1$.*

Proof. Suppose there exists an ε-node-differentially private algorithm to compute the weight of a minimum cut in an incremental graph sequence with error $o(W \log T)$ at all time steps. Using this algorithm and the reduction from Lemma 32, we can compute the sum of any binary stream with error $o(\log T)$ while preserving ε-differential privacy. This contradicts Theorem 26. ◀

5.3   Maximum Weighted Matching

▶ **Lemma 35.** *For any pair of adjacent binary streams σ, σ' of length T there exists a pair of incremental edge-weighted graph sequences $\mathcal{G}$, $\mathcal{G}'$ that are edge-adjacent, such that the weights of the maximum weighted matching in $\mathcal{G}$ and $\mathcal{G}'$ are*

$$w_{\mathrm{M}}(G_t) = W \sum_{i=1}^{t} \sigma(i) + W \cdot T$$

and

$$w_{\mathrm{M}}(G_t') = W \sum_{i=1}^{t} \sigma'(i) + W \cdot T,$$

respectively, at all time steps $t \in \{1, \ldots, T\}$, where edge weights are in $\{1, \ldots, W\}$.

Proof. Let $G_0 = (U \cup V, E)$, where $U = \{u_1, \ldots, u_{2 \cdot T}\}$, $V = \{v_1, \ldots, v_{2 \cdot T}\}$ and $E = \bigcup_{i=1}^{T} \{u_i, u_{(i+1) \mod T}\}$. All edges have weight W. Let $\mathcal{G}$ and $\mathcal{G}'$ be graph sequences with initial graph G_0.

For all times $t \in \{1, \ldots, T\}$ we define the edges $e_0^t = \{u_t, v_t\}$ with $w(e_0^t) = 1$ and $e_1^t = \{u_{T+t}, v_{t+T}\}$ with $w(e_1^t) = W$. At time t we insert

$$\partial E_t^+ = \begin{cases} \{e_0^t\} & \text{if } \sigma(t) = 0, \\ \{e_0^t, e_1^t\} & \text{if } \sigma(t) = 1, \end{cases}$$

$$\partial E_t^{+\prime} = \begin{cases} \{e_0^t\} & \text{if } \sigma'(t) = 0, \\ \{e_0^t, e_1^t\} & \text{if } \sigma'(t) = 1, \end{cases}$$

into $\mathcal{G}$ and $\mathcal{G}'$, respectively. Note that these graph sequences differ in exactly one edge exactly when $\sigma(t) \neq \sigma'(t)$.

G_0 has a matching of weight $W \cdot T$, which consists of the edges in E. Inserting an edge e_0^t for any t does not change the weight of the maximum matching. Inserting an edge e_1^t

however increases the weight of the matching by W, since two previously isolated nodes are now connected by this new edge. Thus, the weight of the maximum matching at time t is

$$w_{\mathrm{M}}(G_t) = W \sum_{i=1}^{t} \sigma(i) + W \cdot T$$

and

$$w_{\mathrm{M}}(G'_t) = W \sum_{i=1}^{t} \sigma'(i) + W \cdot T,$$

which concludes the proof. ◄

We can also reduce differentially private binary counting to node-differentially private maximum weight matching in incremental graph sequences.

▶ **Lemma 36.** *For any pair of adjacent binary streams σ, σ' of length T there exists a pair of incremental edge-weighted graph sequences $\mathcal{G}$, $\mathcal{G}'$ that are node-adjacent, such that the weights of the maximum weighted matching in $\mathcal{G}$ and $\mathcal{G}'$ are*

$$w_{\mathrm{M}}(G_t) = W \sum_{i=1}^{t} \sigma(i)$$

and

$$w_{\mathrm{M}}(G'_t) = W \sum_{i=1}^{t} \sigma'(i)$$

respectively, at all time steps $t \in \{1, \ldots, T\}$, where edge weights are in $\{1, \ldots, W\}$.

Proof. Let $G_0 = (\{v_1, \ldots, v_T\}, \emptyset)$. Let $\mathcal{G}$, $\mathcal{G}'$ be graph sequences with initial graph G_0.

At time t, we insert a node u_t into $\mathcal{G}$ and $\mathcal{G}'$. If $\sigma(t) = 1$ we additionally insert node v'_t along with the edge $\{v'_t, v_t\}$ with weight W into $\mathcal{G}$. Similarly, if $\sigma'(t) = 1$ we insert node v'_t along with the edge $\{v'_t, v_t\}$ with weight W into $\mathcal{G}'$.

Then, the weight of the maximum matching in $\mathcal{G}$ at time t increases by W if $\sigma(t) = 1$; otherwise, it does not change. The same holds for σ' and $\mathcal{G}'$. Thus, the weight of the maximum matching at time t is

$$w_{\mathrm{M}}(G_t) = W \sum_{i=1}^{t} \sigma(i)$$

and

$$w_{\mathrm{M}}(G'_t) = W \sum_{i=1}^{t} \sigma'(i),$$

which concludes the proof. ◄

▶ **Theorem 37.** *Any ε-edge-differentially private algorithm to compute the weight of a maximum weight matching in an incremental graph sequence of length T must have additive error $\Omega(W \log T)$ when the edge weights are in $\{1, \ldots, W\}$, even with $\varepsilon = 1$.*

Proof. Suppose there exists an ε-edge-differentially private algorithm to compute the weight of a maximum weight matching in an incremental graph sequence with error $o(W \log T)$ at all time steps. Using this algorithm and the reduction from Lemma 35, we can compute the sum of any binary stream with error $o(\log T)$ while preserving ε-differential privacy. This contradicts Theorem 26. ◀

▶ **Theorem 38.** *Any ε-node-differentially private algorithm to compute the weight of a maximum weight matching in an incremental graph sequence of length T must have additive error $\Omega(W \log T)$ when the edge weights are in $\{1, \ldots, W\}$, even with $\varepsilon = 1$.*

Proof. Suppose there exists an ε-node-differentially private algorithm to compute the weight of a maximum weight matching in an incremental graph sequence with error $o(W \log T)$ at all time steps. Using this algorithm and the reduction from Lemma 36, we can compute the sum of any binary stream with error $o(\log T)$ while preserving ε-differential privacy. This contradicts Theorem 26. ◀

5.4 Subgraph Counting, High-Degree Nodes and Degree Histogram

We show that, for the number of high-degree nodes, the degree histogram and the subgraph counting problems adjacent binary streams can be reduced to adjacent incremental graph sequences. This implies a lower bound of $\Omega(\log T)$ on the error for these problems.

▶ **Lemma 39.** *There exist functions $\mathcal{G}_\tau$, $\mathcal{G}_h$, $\mathcal{G}_\Delta$, $\mathcal{G}_k$, that map a binary stream to an incremental graph sequence of the same length, such that the following holds: At time t*
1. *the number of nodes of degree at least τ in $\mathcal{G}_\tau(\sigma)$,*
2. *the number of nodes of degree 2 in $\mathcal{G}_h(\sigma)$,*
3. *the number of triangles in $\mathcal{G}_\Delta(\sigma)$ and*
4. *the number of k-stars in $\mathcal{G}_k(\sigma)$*
is equal to $\sum_{i=1}^{t} \sigma(i)$. Furthermore, all functions map adjacent binary streams to edge-adjacent graph sequences.

Proof. Let σ be a binary stream.
1. $\mathcal{G}_\tau$ outputs a graph sequence with initial graph $G_\tau = \cup_{i=1}^{T} S_i$. Each S_i is a τ-star with one missing edge. At time t, if $\sigma(t) = 1$, then the missing edge is inserted into S_t.
2. $\mathcal{G}_h$ outputs a graph sequence on the nodes $V_h = \cup_{i=1}^{T} \{a_i, b_i, c_i\}$. Initially, there are edges $\{b_i, c_i\}$ for each $i \in \{1, \ldots, T\}$. At time t, if $\sigma(t) = 1$, then the edge $\{a_t, b_t\}$ is inserted. Note that $\deg(b_t) = \sigma(t) + 1$ and $\deg(a_i), \deg(c_i) \leq 1$ for all t.
3. $\mathcal{G}_\Delta$ outputs a graph sequence on the nodes $V_\Delta = \cup_{i=1}^{T} \{a_i, b_i, c_i\}$. Initially, there are edges $\{a_i, b_i\}$ and $\{b_i, c_i\}$ for each $i \in \{1, \ldots, T\}$. At time t, if $\sigma(t) = 1$, then the edge $\{a_t, c_t\}$ is inserted.
4. $\mathcal{G}_k$ outputs a graph sequence with initial graph $G_k = \cup_{i=1}^{T} S_i$. Each S_i is a k-star with one missing edge. At time t, if $\sigma(t) = 1$, then the missing edge is inserted into S_t.
In all graph sequences no edge is inserted if $\sigma(t) = 0$; if $\sigma(t) = 1$, then a single edge is inserted. Thus, the functions map adjacent binary streams to edge-adjacent graph sequences. ◀

▶ **Lemma 40.** *There exist functions $\mathcal{G}'_\tau$, $\mathcal{G}'_h$, $\mathcal{G}'_e$, $\mathcal{G}'_\Delta$, $\mathcal{G}'_k$, that map a binary stream to an incremental graph sequence of the same length with maximum degree $D > 3$, $\tau, k \leq D$, such that the following holds: Let σ be a binary stream. At time t*
1. *the number of nodes of degree at least τ in $\mathcal{G}'_\tau(\sigma)$,*
2. *the number of nodes of degree 2 in $\mathcal{G}'_h(\sigma)$,*
3. *the number of edges in $\mathcal{G}'_e(\sigma)$,*

4. *the number of triangles in $\mathcal{G}'_\Delta(\sigma)$,*
5. *the number of k-stars in $\mathcal{G}'_k(\sigma)$*

is equal to $D \sum_{i=1}^{t} \sigma(i)$. Furthermore, all functions map adjacent binary streams to node-adjacent graph sequences.

Proof. Let σ be a binary stream.

1. $\mathcal{G}'_\tau$ outputs a graph sequence with initial graph $G_\tau = \cup_{i=1}^{T} S_i$. Each S_i is the union of $(D-1)$-many $(\tau-1)$-stars. At time t, if $\sigma(t) = 1$, then a node v_t is inserted into S_t, along with edges from v_t to the central node of each $(\tau-1)$-star in S_t. Since $D \geq \tau$, the number of τ-stars in S_t is D if $\sigma(t) = 1$ and zero otherwise. In S_t the centers of the τ-stars have degree $\geq \tau$, all other nodes have degree $< \tau$.

2. $\mathcal{G}'_h$ outputs a graph sequence with initial graph $G_h = \cup_{i=1}^{T} M_i$, where M_i consists of D pairwise connected nodes, i.e., $M_i = (\bigcup_{j=1}^{D} \{a_j, b_j\}, \bigcup_{j=1}^{D} \{\{a_j, b_j\}\})$. At time t, if $\sigma(t) = 1$, then a node v_t is inserted into M_t along with edges $\{v_t, a_j\}$ for all $j = 1, \ldots, D$. The number of nodes of degree 2 is D in M_t if $\sigma(t) = 1$ and zero otherwise.

3. $\mathcal{G}'_e$ outputs a graph sequence with initial graph $G_e = \cup_{i=1}^{T} V_i$, where $V_i = (\{1, \ldots, D\}, \emptyset)$. At time t, if $\sigma(t) = 1$, then a node v_t is inserted into V_t along with edges from v_t to all nodes in V_t. The number of edges is D in V_t if $\sigma(t) = 1$ and zero otherwise.

4. $\mathcal{G}'_\Delta$ outputs a graph sequence with initial $G_\Delta = \cup_{i=1}^{T} C_i$, where C_i is a cycle of D nodes. At time t, if $\sigma(t) = 1$, then a node v_t is inserted into C_t along with edges from v_t to every node in C_t. Thus, if $\sigma(t) = 1$ the number of triangles in C_t is D; otherwise, it is zero.

5. $\mathcal{G}'_k$ outputs a graph sequence with initial graph $G_\tau = \cup_{i=1}^{T} S_i$. Each S_i is the union of $(D-1)$-many $(\tau-1)$-stars. At time t, if $\sigma(t) = 1$, then a node v_t is inserted into S_t, along with edges from v_t to the central node of each $(\tau-1)$-star in S_t. Since $D \geq \tau$, the number of τ-stars in S_t is D if $\sigma(t) = 1$ and zero otherwise.

In all graph sequences no node is inserted if $\sigma(t) = 0$; if $\sigma(t) = 1$, then a single node is inserted. Thus, the functions map adjacent binary streams to node-adjacent graph sequences. ◀

▶ **Theorem 41.** *Any ε-edge-differentially private algorithm to compute*
1. *the number of high-degree nodes,*
2. *the degree histogram,*
3. *the number of edges,*
4. *the number of triangles,*
5. *the number of k-stars*

in an incremental graph sequence of length T must have additive error $\Omega(\log T)$, even with $\varepsilon = 1$.

Proof. By Lemma 39, adjacent binary streams can be encoded in edge-adjacent incremental graph sequences of the same length. Counting the number of edges is equivalent to binary counting. The claim follows by Theorem 26. ◀

▶ **Theorem 42.** *Any ε-node-differentially private algorithm to compute*
1. *the number of high-degree nodes,*
2. *the degree histogram,*
3. *the number of edges,*
4. *the number of triangles,*
5. *the number of k-stars*

in an incremental graph sequence of length T must have additive error $\Omega(D \log T)$, where D is the maximum degree, even with $\varepsilon = 1$.

Proof. By Lemma 40, adjacent binary streams can be encoded in node-adjacent incremental graph sequences of the same length. The claim follows by Theorem 26. ◀

6 Lower Bound for User-Level Privacy

We show that for several fundamental problems on dynamic graphs like minimum spanning tree and minimum cut, a differentially private algorithm with edge-adjacency on user-level must have an additive error that is linear in the maximum function value. Technically, we define the *spread* of a graph function as the maximum difference of the function's value on any two graphs. Then, we show that any algorithm must have an error that is linear in the graph function's spread. We write this section in terms of edge-adjacency but the corresponding result for node-adjacency carries over.

▶ **Definition 43** (adjacency transformation). *Let G_1 and G_2 be a pair of graphs. We define $\tau(G_1, G_2)$ to be an update sequence $u_1, \ldots, u_\ell$ of minimum length that transforms G_1 into G_2. We denote the graph sequence that results from applying $\tau(G_1, G_2)$ to G_1 by $\mathrm{T}(G_1, G_2)$, i.e., $\mathrm{T}(G_1, G_2) = (G_1, G_1 \oplus u_1, (G_1 \oplus u_1) \oplus u_2, \ldots, G_2)$.*

▶ **Definition 44** (spread). *Let $s, \ell : \mathbb{N} \to \{2i \mid i \in \mathbb{N}\}$ be functions. A graph function f has spread $(s(n), \ell(n))$ on inputs of size n if, for every n, there exist two graphs G_1, G_2 of size n so that $|f(G_1) - f(G_2)| \geq s(n)$ and $|\tau(G_1, G_2)| = \ell(n)$. Furthermore, f spares an edge e if $e \in E(G_1) \cap E(G_2)$.*

See Table 1 on page 3 for the resulting lower bounds.

▶ **Theorem 45.** *Let $\epsilon, \delta > 0$ and let f be a graph function with spread (s, ℓ) that spares an edge e. For streams of length T on graphs of size n, where $T > \log(e^{4\epsilon\ell}/(1 - \delta)) \in O(\epsilon\ell + \log(1/(1 - \delta)))$, every ϵ-differentially private dynamic algorithm with user-level edge-adjacency that computes f with probability at least $1 - \delta$ must have error $\Omega(s(n))$.*

Proof. For any update u, let u^{-1} be the update that reverses u. For the sake of contradiction, assume that there exists an ϵ-differentially private graph algorithm $\mathcal{A}$ with edge-adjacency on user-level that with probability at least $1 - \delta$ computes f with error less than $s(n)/2$. Let $n > 0$, $s := s(n)$, $\ell = \ell(n)$ and let G_1, G_2 be graphs of size n with spread (s, ℓ) on f that spares an edge e. We assume below wlog that ℓ is even, if it is odd the proof can be easily adapted. Let u be the update operation that inserts e into the current graph (see below). Let $(o_0, \ldots, o_{\ell-1}) := \tau(G_1, G_2)$ and let $(o_0', \ldots, o_{\ell-1}') := \tau(G_2, G_1)$. Without loss of generality, assume that T is a multiple of 2ℓ. Let $\mathcal{B}$ be the set of bit strings of length $T/(2\ell)$. For every bit string $b = (b_0, \ldots, b_{T/(2\ell)-1}) \in \mathcal{B}$, we construct a unique graph sequence $\mathcal{G}_b = (H_0, \ldots, H_T)$ as follows, resulting in $2^{T/(2\ell)}$ different graph sequences.

The sequence is partitioned into phases of length 2ℓ. Each phase has a *type*: It is either a *forward* phase or a *backward* phase. The sequence starts with a forward phase and alternates between forward and backward phases. Intuitively, forward phases transform G_1 into G_2, while backward phases transform G_2 into G_1. This takes ℓ updates. To generate an exponential number of different but "close" graph sequences each phase also contains ℓ *placeholder updates*. The purpose of a placeholder update is to basically not modify the graph in any of the graph sequences. Thus the first placeholder update updates e and the next update is the inverse operation. This is repeated $\ell/2$ times. For every phase, a corresponding bit in b decides whether the transformation happens *before* the placeholder updates or *after* placeholder updates. Thus, $2^{T/(2\ell)}$ different graph sequences are generated, where the graphs at position $2i\ell$ for any integer $i \geq 0$ in all these sequences are identical.

Formally, for integer $i = 0, 1, ..., T/(2\ell) - 1$, the bit b_i corresponds to the phase $H_{2i\ell}, \ldots, H_{2(i+1)\ell-1}$ in the sequence $\mathcal{G}_b$. Bit b_i corresponds to graphs in a forward phase if and only if i is even. We define $\mathcal{G}_b$ as follows:

forward phase, $b_i = 0$: For any j, $0 \leq j < \ell$ (i.e., the first half of the phase), we define $H_{2i\ell+j+1} := H_{2i\ell+j} \oplus o_j$. The second half of the phase, i.e., $H_{(2i+1)\ell}, \ldots, H_{2(i+1)\ell-1}$, is defined by alternating between $H_{(2i+1)\ell}$ and $H_{(2i+1)\ell} \oplus u$, which are placeholder updates.

forward phase, $b_i = 1$: This phase results from swapping the first and the second half of the forward phase with $b_i = 0$. In particular, the first half of the phase, i.e., $H_{2i\ell}, \ldots, H_{(2i+1)\ell-1}$, is defined by alternating between $H_{2i\ell}$ and $H_{2i\ell} \oplus u$, which are placeholder updates. For any j, $\ell \leq j < 2\ell$, $H_{2i\ell+j+1} := H_{2i\ell+j} \oplus o_{j-\ell}$.

backward phase, $b_i \in \{0, 1\}$: The graphs $H_{2i\ell}, \ldots, H_{2(i+1)\ell-1}$ are defined analogously to a forward phase with $b_i = 0$ or $b_i = 1$, respectively. The only difference is that instead of $o_0, \ldots, o_{\ell-1}$, the updates $o'_0, \ldots, o'_{\ell-1}$ are used.

It follows from the construction that for any bit string b and its corresponding graph sequence $(H_0, \ldots, H_T)$, it holds that $H_i = G_1$ if $i \mod 4\ell = 0$ and $H_i = G_2$ if $i \mod 4\ell = 2\ell$. However, for every pair of bit strings b, b', $b \neq b'$, and their corresponding graph sequences $(H_0, \ldots, H_{T-1})$ and $(H'_0, \ldots, H'_{T-1})$, there exists an i with $i \mod 2\ell = \ell$, so that $|f(H_i) - f(H'_i)| \geq s$. This stems from the fact that f has spread (s, ℓ) and that b and b' must differ in at least one bit, say $b_i = 0$ and $b'_i = 1$. Thus, by construction, $H_{(2i+1)\ell} = G_2$ (as $\tau(G_1, G_2)$ has already been executed in this phase) and $H'_{(2i+1)\ell} = G_1$ (as $\tau(G_1, G_2)$ has not yet been executed in this phase).

Next we show that any graph sequence $\mathcal{G}_b$ is on user-level "edge-close" to a very generic graph sequence. More specifically, let $\mathcal{G}'$ be the sequence of graphs that alternates between G_1 and $G_1 \oplus u$, i.e., $(G_1, G_1 \oplus u, G_1, G_1 \oplus u, \ldots)$. We argue that there is a (short) sequence $\Gamma_1, \Gamma_2, \ldots \Gamma_{3\ell/2+1}$ of graph sequences such that Γ_i and Γ_{i+1} are *user-level edge-adjacent* graph sequences such that $\Gamma_1 = \mathcal{G}_b$ and $\Gamma_{\ell+1} = \mathcal{G}'$. Said differently the sequence of graph sequences transforms $\mathcal{G}_b$ into $\mathcal{G}'$. Instead of talking about a transforming graph sequence we use below the notation of *user-level edge-adjacency operation*: It takes one operation to transform a graph sequence Γ_i into a user-level edge-adjacent graph sequence Γ_{i+1}.

We now give the details: For any b, each phase of $\mathcal{G}_b$ can be divided into two halves $(H_{2i\ell}, \ldots, H_{(2i+1)\ell-1})$ and $(H_{(2i+1)\ell}, \ldots H_{2(i+1)\ell-1})$: in one half (the first if $b_i = 0$, the latter if $b_i = 1$), the updates are already alternating between u, i.e., inserting e, and u^{-1}, i.e., deleting e. Therefore, we only need to transform the updates on the other half into a sequence that alternates between u and u^{-1}. The other half is either $\mathrm{T}(G_1, G_2)$ (in forward phases) or $\mathrm{T}(G_2, G_1)$ (in backward phases). By Lemma 46, there is a sequence of graph sequences of length $3\ell/2 + 1$ that starts with $\mathrm{T}(G_1, G_2)$ and ends with $\mathcal{G}'$ and a sequence of edges $(e_1, \ldots, e_{3\ell/2+1})$ so that the i-th and the $(i+1)$-th graph sequence in $\mathcal{G}'$ are *event-level edge-adjacent* on e_i. Similarly, such sequence and edges $(e'_1, \ldots, e'_{3\ell/2+1})$ exists for $\mathrm{T}(G_2, G_1)$. Recall that *user-level edge-adjacency* allows to modify a graph sequence by updating an arbitrary number of graphs in the sequence *using the same edge e, inserting e into some graphs that do not contain e and removing it from some of the graphs that contain e*. Also recall that $\mathcal{G}_b$ consists of many repetitions of $\mathrm{T}(G_1, G_2)$. Thus one user-level edge-adjacency operation using the edge e_1 allows to modify *all* occurrences of $\mathrm{T}(G_1, G_2)$ to reflect the update of edge e_1. Afterwards a second user-level edge-adjacency operation using e_2 modifies the resulting sequence etc. After $3\ell/2 + 1$ user-level edge-adjacency operations all occurrences of $\mathrm{T}(G_1, G_2)$ have been transformed into sequences that alternate G_1 and $G_1 \oplus u$. The same can be done for the repetitions of $\mathrm{T}(G_2, G_1)$. This shows that at most $3(\ell + 1)$ user-level edge-adjacency operations suffice to transform $\mathcal{G}_b$ into $\mathcal{G}'$.

Recall that we assumed the existence of an ϵ-differentially private graph algorithm $\mathcal{A}$ on user-level that, with probability at least $1-\delta$, computes f with error less than $s/2$. After each update $\mathcal{A}$ has to output f, i.e., the sequence of outputs has the same length as the sequence of updates. Let O_b be the set of output sequences of $\mathcal{A}$ with additive error at most $s/2$ when processing $\mathcal{G}_b$ with G_1 as initial graph. Now we consider the family of output sets O_b for all $b \in \mathcal{B}$. For any two $b, b' \in \mathcal{B}$ with $b \neq b'$ consider the value of f applied to the two graph sequences $\mathcal{G}_b$ and $\mathcal{G}_{b'}$ and recall that there is at least one index in these two graph sequences where the f-value applied to the two graphs at this index differs by at least s. Thus, the event that $\mathcal{A}$ gives an answer with additive error less than $s/2$ on $\mathcal{G}_b$ (i.e., returns a sequence from O_b) and the event hat $\mathcal{A}$ gives an answer with additive error less than $s/2$ on $\mathcal{G}_{b'}$ (i.e., returns a sequence from $O_{b'}$) are pairwise disjoint. However, for any $b \in \mathcal{B}$ by ϵ-differential privacy, the probability that $\mathcal{A}$ outputs an sequence from O_b when run on $\mathcal{G}'$, i.e. $\Pr[\mathcal{A}(\mathcal{G}') \in O_b]$, is at least $e^{-4\epsilon\ell} \cdot \Pr[\mathcal{A}(\mathcal{G}_b) \in O_b] \geq e^{-4\epsilon\ell}(1-\delta)$. Since $T > \log(e^{4\epsilon\ell}/(1-\delta))$, we have that $|\mathcal{B}| = 2^T > e^{4\epsilon\ell}/(1-\delta)$; it follows that $\Pr[\mathcal{A}(\mathcal{G}') \in \cup_{b \in \mathcal{B}} O_b] = \sum_{b \in \mathcal{B}} e^{-4\epsilon\ell} \Pr[\mathcal{A}(\mathcal{G}_b) \in O_b] > 1$, which is a contradiction. ◀

▶ **Lemma 46.** *Let $n > 0$ and let f be a graph function with spread (s, ℓ) that spares an edge e on G_1, G_2 according to Definition 43. Let u be the update that inserts e, and let $\mathcal{G}' := (G_1, G_1 \oplus u, G_1, G_1 \oplus u, \dots,)$ of length $\ell + 1 := |\tau(G_1, G_2)| + 1$. Then, there exists a sequence of graph sequences $(\mathcal{G}_0 := \mathrm{T}(G_1, G_2), \dots, \mathcal{G}_{3\ell/2+1} := \mathcal{G}')$ and a sequence of edges $(e_0, e_1, \dots, e_{3\ell/2+1})$ so that, for any $0 \leq i \leq 3\ell/2$, $\mathcal{G}_i$ and $\mathcal{G}_{i+1}$ are event-level edge-adjacent on e_i.*

Proof. For $i \in \{0\} \cup [2\ell]$, we construct sequences $\mathcal{G}_i = (G_{i,1}, \dots, G_{i,2\ell+1})$. Define $\mathcal{G}_0 := \mathrm{T}(G_1, G_2)$. We transform $\mathcal{G}_0$ by alternatingly applying u (i.e. we insert e) and u^{-1} (i.e. we delete e) to its graphs (in addition to the existing updates). For $i \in \{1, \dots, \ell/2\}$, we define

$$\mathcal{G}_i := (G_{i-1,1}, \dots, G_{i-1,2i-1}, G_{i-1,2i} \oplus u_i, G_{i-1,2i+1} \oplus u_i^{-1}, G_{i-1,2i+2}, \dots, G_{i-1,\ell}).$$

For example, for $i = 2$, $\mathcal{G}_1$ is the graph sequence where u and u^{-1} are applied for the graphs $G_{0,2}$ to $G_{0,4}$. Let $(u_1, \dots) := \tau(G_1, G_2)$. For $i \in \{\ell/2 + 1, \dots, 3\ell/2\}$, we define

$$\mathcal{G}_i := (G_{i-1,1}, \dots, G_{i-1,i}, G_{i-1,i+1} \oplus u_i'^{-1}, G_{i-1,i+2} \oplus u_i'^{-1}, \dots, G_{i-1,\ell} \oplus u_i'^{-1}).$$

In other words, in $\mathcal{G}_i$, the first i updates of $\mathrm{T}(G_1, G_2)$ are not present compared to $\mathcal{G}_{\ell/2}$. Since $\mathrm{T}(G_1, G_2)$ is a shortest sequence that transforms G_1 into G_2, an edge is either inserted or deleted at most once. Thus, reverting u_i' by executing $\oplus u_i'^{-1}$ for all graphs $G_{i-1,j}$ with $j \geq i + 1$ cannot invalidate the graph sequence. The claim follows because u_i' only affects a single edge. ◀

▶ **Fact 47.** *Minimum spanning tree has spread $(\Theta(nW), \Theta(n))$. Minimum cut has spread $(\Theta(nW), \Theta(n^2))$. Maximal matching has spread $(\Theta(n), \Theta(n))$. Maximum cardinality matching has spread $(\Theta(n), \Theta(n))$. Maximum weight matching has spread $(\Theta(nW), \Theta(n))$.*

───── **References** ─────

1   Raman Arora and Jalaj Upadhyay. On differentially private graph sparsification and applications. In H. Wallach, H. Larochelle, A. Beygelzimer, F. dAlché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 13399–13410. Curran Associates, Inc., 2019.

2   Boaz Barak, Kamalika Chaudhuri, Cynthia Dwork, Satyen Kale, Frank McSherry, and Kunal Talwar. Privacy, accuracy, and consistency too: A holistic solution to contingency table release. In *Proceedings of the Twenty-Sixth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '07, pages 273–282, New York, NY, USA, June 2007. Association for Computing Machinery. `doi:10.1145/1265530.1265569`.

3   Jeremiah Blocki, Avrim Blum, Anupam Datta, and Or Sheffet. Differentially private data analysis of social networks via restricted sensitivity. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*, ITCS '13, pages 87–96, New York, NY, USA, January 2013. Association for Computing Machinery. `doi:10.1145/2422436.2422449`.

4   BlumAvrim, LigettKatrina, and RothAaron. A learning theory approach to noninteractive database privacy. *Journal of the ACM (JACM)*, May 2013. `doi:10.1145/2450142.2450148`.

5   T. H. Hubert Chan, Mingfei Li, Elaine Shi, and Wenchang Xu. Differentially Private Continual Monitoring of Heavy Hitters from Distributed Streams. In Simone Fischer-Hübner and Matthew Wright, editors, *Privacy Enhancing Technologies*, Lecture Notes in Computer Science, pages 140–159, Berlin, Heidelberg, 2012. Springer. `doi:10.1007/978-3-642-31680-7_8`.

6   T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3), November 2011. `doi:10.1145/2043621.2043626`.

7   Shixi Chen and Shuigeng Zhou. Recursive mechanism: Towards node differential privacy and unrestricted joins. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 653–664, New York, NY, USA, June 2013. Association for Computing Machinery. `doi:10.1145/2463676.2465304`.

8   Wei-Yen Day, Ninghui Li, and Min Lyu. Publishing Graph Degree Distribution with Node Differential Privacy. In *Proceedings of the 2016 International Conference on Management of Data*, SIGMOD '16, pages 123–138, New York, NY, USA, June 2016. Association for Computing Machinery. `doi:10.1145/2882903.2926745`.

9   Cynthia Dwork. Differential Privacy. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *Automata, Languages and Programming*, Lecture Notes in Computer Science, pages 1–12, Berlin, Heidelberg, 2006. Springer. `doi:10.1007/11787006_1`.

10   Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, pages 371–380, New York, NY, USA, May 2009. Association for Computing Machinery. `doi:10.1145/1536414.1536466`.

11   Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating Noise to Sensitivity in Private Data Analysis. In Shai Halevi and Tal Rabin, editors, *Theory of Cryptography*, Lecture Notes in Computer Science, pages 265–284, Berlin, Heidelberg, 2006. Springer. `doi:10.1007/11681878_14`.

12   Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, page 715–724, New York, NY, USA, 2010. Association for Computing Machinery. `doi:10.1145/1806689.1806787`.

13   Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil Vadhan. On the complexity of differentially private data release: Efficient algorithms and hardness results. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, pages 381–390, New York, NY, USA, May 2009. Association for Computing Machinery. `doi:10.1145/1536414.1536467`.

14   Marek Eliáš, Michael Kapralov, Janardhan Kulkarni, and Yin Tat Lee. Differentially Private Release of Synthetic Graphs. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, Proceedings, pages 560–578. Society for Industrial and Applied Mathematics, December 2019. `doi:10.1137/1.9781611975994.34`.

15   M. A. Erdogdu and N. Fawaz. Privacy-utility trade-off under continual observation. In *2015 IEEE International Symposium on Information Theory (ISIT)*, pages 1801–1805, June 2015. `doi:10.1109/ISIT.2015.7282766`.

16   Anupam Gupta, Katrina Ligett, Frank McSherry, Aaron Roth, and Kunal Talwar. Differentially Private Combinatorial Optimization. In *Proceedings of the 2010 Annual ACM-SIAM Symposium on Discrete Algorithms*, Proceedings, pages 1106–1125. Society for Industrial and Applied Mathematics, January 2010. `doi:10.1137/1.9781611973075.90`.

17   Moritz Hardt and Guy N. Rothblum. A Multiplicative Weights Mechanism for Privacy-Preserving Data Analysis. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 61–70, October 2010. `doi:10.1109/FOCS.2010.85`.

18   M. Hay, C. Li, G. Miklau, and D. Jensen. Accurate Estimation of the Degree Distribution of Private Networks. In *2009 Ninth IEEE International Conference on Data Mining*, pages 169–178, December 2009. `doi:10.1109/ICDM.2009.11`.

19   Vishesh Karwa, Sofya Raskhodnikova, Adam Smith, and Grigory Yaroslavtsev. Private analysis of graph structure. *Proceedings of the VLDB Endowment*, 4(11):1146–1157, August 2011. `doi:10.14778/3402707.3402749`.

20   S. P. Kasiviswanathan, H. K. Lee, K. Nissim, S. Raskhodnikova, and A. Smith. What Can We Learn Privately? In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 531–540, October 2008. `doi:10.1109/FOCS.2008.27`.

21   Shiva Prasad Kasiviswanathan, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Analyzing Graphs with Node Differential Privacy. In Amit Sahai, editor, *Theory of Cryptography*, Lecture Notes in Computer Science, pages 457–476, Berlin, Heidelberg, 2013. Springer. `doi:10.1007/978-3-642-36594-2_26`.

22   Georgios Kellaris, Stavros Papadopoulos, Xiaokui Xiao, and Dimitris Papadias. Differentially private event sequences over infinite streams. *Proceedings of the VLDB Endowment*, 7(12):1155–1166, August 2014. `doi:10.14778/2732977.2732989`.

23   Wentian Lu and Gerome Miklau. Exponential random graph estimation under differential privacy. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, pages 921–930, New York, NY, USA, August 2014. Association for Computing Machinery. `doi:10.1145/2623330.2623683`.

24   Min Lyu, Dong Su, and Ninghui Li. Understanding the sparse vector technique for differential privacy. *Proceedings of the VLDB Endowment*, 10(6), 2017.

25   F. McSherry and K. Talwar. Mechanism Design via Differential Privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103, October 2007. `doi:10.1109/FOCS.2007.66`.

26   Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing*, STOC '07, pages 75–84, New York, NY, USA, June 2007. Association for Computing Machinery. `doi:10.1145/1250790.1250803`.

27   J. Le Ny and G. J. Pappas. Differentially Private Filtering. *IEEE Transactions on Automatic Control*, 59(2):341–354, February 2014. `doi:10.1109/TAC.2013.2283096`.

28   Aaron Roth and Tim Roughgarden. Interactive privacy via the median mechanism. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, pages 765–774, New York, NY, USA, June 2010. Association for Computing Machinery. `doi:10.1145/1806689.1806794`.

29   Shuang Song, Susan Little, Sanjay Mehta, Staal Vinterbo, and Kamalika Chaudhuri. Differentially private continual release of graph statistics, 2018. `arXiv:1809.02575`.

30   Q. Wang, Y. Zhang, X. Lu, Z. Wang, Z. Qin, and K. Ren. Real-Time and Spatio-Temporal Crowd-Sourced Social Network Data Publishing with Differential Privacy. *IEEE Transactions on Dependable and Secure Computing*, 15(4):591–606, July 2018. `doi:10.1109/TDSC.2016.2599873`.

31   Yue Wang, Xintao Wu, and Leting Wu. Differential Privacy Preserving Spectral Graph Analysis. In Jian Pei, Vincent S. Tseng, Longbing Cao, Hiroshi Motoda, and Guandong Xu, editors, *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 329–340, Berlin, Heidelberg, 2013. Springer. `doi:10.1007/978-3-642-37456-2_28`.

32   Jun Zhang, Graham Cormode, Cecilia M. Procopiuc, Divesh Srivastava, and Xiaokui Xiao. Private Release of Graph Statistics using Ladder Functions. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, SIGMOD '15, pages 731–745, New York, NY, USA, May 2015. Association for Computing Machinery. `doi: 10.1145/2723372.2737785`.

33   Sen Zhang, Weiwei Ni, and Nan Fu. Differentially Private Graph Publishing with Degree Distribution Preservation. *Computers & Security*, page 102285, April 2021. `doi:10.1016/j. cose.2021.102285`.