Graph-based Clustering under Differential Privacy

Rafael Pinot, Anne Morvan, Florian Yger, Cedric Gouy-Pailler, Jamal Atif

Graph-based Clustering under Differential Privacy

Rafael Pinot[*1,2], **Anne Morvan**[*†1,2], **Florian Yger**[1], **Cédric Gouy-Pailler**[2], and **Jamal Atif**[1]
[1]Université Paris-Dauphine, PSL Research University, CNRS, 75016 Paris, France
[2]CEA, LIST, 91191 Gif-sur-Yvette, France

Abstract

In this paper, we present the first differentially private clustering method for arbitrary-shaped node clusters in a graph. This algorithm takes as input only an approximate Minimum Spanning Tree (MST) $\mathcal{T}$ released under weight differential privacy constraints from the graph. Then, the underlying nonconvex clustering partition is successfully recovered from cutting optimal cuts on $\mathcal{T}$. As opposed to existing methods, our algorithm is theoretically well-motivated. Experiments support our theoretical findings.

1 INTRODUCTION

Weighted graph data is known to be a useful representation data type in many fields, such as bioinformatics or analysis of social, computer and information networks. More generally, a graph can always be built based on the data dissimilarity where points of the dataset are the vertices and weighted edges express "distances" between those objects. For both cases, graph clustering is one of the key tools for understanding the underlying structure in the graph (Schaeffer, 2007). These clusters can be seen as groups of nodes close in terms of some specific similarity.

Nevertheless, it is critical that the data representation used in machine learning applications protects the private characteristics contained into it. Let us consider an application where one wants to identify groups of similar web pages in the sense of traffic volume *i.e.* web pages with similar audience. In that case, the nodes stand for the websites. The link between two vertices represents the fact that some people consult them both. In such a framework, the web browsing history of an individual is used to set the edge weights. We consider that this history can be a very sensitive information for the user, since he/she could have visited sensible content web pages. Treating such datasets as non-private could lead to leaking information such as his/her political, sexual, or religious preferences. As a standard for data privacy preservation, differential privacy (Dwork et al., 2006b) has been designed: an algorithm is differentially private if, given two close databases, it produces statistically indistinguishable outputs. Since then, its definition has been extended to weighted graphs. Though, machine learning applications ensuring data privacy remain rare, in particular for clustering which encounters severe theoretical and practical limitations. Indeed, some clustering methods lack of theoretical support and most of them restrict the data distribution to convex-shaped clusters (Nissim et al., 2007; Blum et al., 2008; McSherry, 2009; Dwork, 2011) or unstructured data (Ho and Ruan, 2013; Chen et al., 2015). Hence, the aim of this paper is to offer a theoretically motivated private graph clustering. Moreover, to the best of our knowledge, this is the first weight differentially-private clustering algorithm able to detect clusters with an arbitrary shape for weighted graph data.

Our method belongs to the family of Minimum Spanning Tree (MST)-based approaches. An MST represents a useful summary of the graph, and appears to be a natural object to describe it at a lower cost. For clustering purposes, it has the appealing property to help retrieving non-convex shapes (Zahn, 1971; Asano et al., 1988; Grygorash et al., 2006; Morvan et al., 2017). Moreover, they appear to be well-suited for incorporating privacy constraints as will be formally proved in this work.

Contributions: Our contributions are threefold: 1) we provide the first theoretical justifications of MST-based clustering algorithms. 2) We endow DBMSTCLU algorithm (Morvan et al., 2017), an MST-based clustering algorithm from the literature, with theoretical guarantees.

[*]Rafael Pinot and Anne Morvan contributed equally.

[†]Partly supported by the *Direction Générale de l'Armement* (French Ministry of Defense).

3) We introduce a differentially-private version of DBM-STCLU and give several results on its privacy/utility tradeoff.

2 PRELIMINARIES

2.1 NOTATIONS

Let $\mathcal{G} = (V, E, w)$ be a simple undirected weighted graph with a vertex set V, an edge set E, and a weight function $w := E \to \mathbb{R}$. One will respectively call the edge set and the node set of a graph $\mathcal{G}$ using the applications $E(\mathcal{G})$ and $V(\mathcal{G})$. Given a node set $S \subset V$, one denotes by $\mathcal{G}_{|S}$ the subgraph induced by S. We call $G = (V, E)$ the topology of the graph, and $\mathcal{W}_E$ denotes the set of all possible weight functions mapping E to weights in $\mathbb{R}$. For the remaining of this work, cursive letter are used to represent weighted graphs and straight letters refer to topological arguments. Since graphs are simple, the path $\mathcal{P}_{u-v}$ between two vertices u and v is characterized either as the ordered sequence of vertices $\{u, \dots, v\}$ or corresponding binding edges depending on the context. We also denote $V_{\mathcal{P}_{u-v}}$ the unordered set of such vertices. Besides, edges e_{ij} denote an edge between nodes i and j. Finally, for all positive integer K, $[K] := \{1, \dots, K\}$.

2.2 DIFFERENTIAL PRIVACY IN GRAPHS

As opposed to node-differential privacy (Kasiviswanathan et al., 2013) and edge-differential privacy (Hay et al., 2009), both based on the graph topology, the privacy framework considered here is weight-differential privacy where the graph topology $G = (V, E)$ is assumed to be public and the private information to protect is the weight function $w := E \to \mathbb{R}$. Under this model introduced by Sealfon (2016), two graphs are said to be neighbors if they have the same topology, and *close* weight functions. This framework allows one to release an almost minimum spanning tree with weight-approximation error of $O(|V| \log |E|)$ for fixed privacy parameters. Differential privacy is ensured in that case by using the Laplace mechanism on every edges weight to release a spanning tree based on a perturbed version of the weight function. The privacy of the spanning tree construction is thus provided by post-processing (cf. Th. 2.5). However, under a similar privacy setting, Pinot (2018) recently manages to produce the topology of a tree under differential privacy without relying on the post-processing of a more general mechanism such as the "Laplace mechanism". Their algorithm, called PAMST, privately releases the topology of an almost minimum spanning tree thanks to an iterative use of the "Exponential mechanism" instead. For fixed privacy parameters, the weight approximation error is $O\left(\frac{|V|^2}{|E|} \log |V|\right)$, which outperforms the former method from Sealfon (2016) on arbitrary weighted graphs under weak assumptions on the graph sparseness. Thus, we keep here privacy setting from Pinot (2018).

Definition 2.1 (Pinot (2018))**.** *For any edge set E, two weight functions $w, w' \in \mathcal{W}_E$ are neighboring, denoted $w \sim w'$, if $||w - w'||_\infty := \max_{e \in E} |w(e) - w'(e)| \leq \mu$.*

μ represents the sensitivity of the weight function and should be chosen according to the application and the range of this function. The neighborhood between such graphs is clarified in the following definition.

Definition 2.2. *Let $\mathcal{G} = (V, E, w)$ and $\mathcal{G}' = (V', E', w')$, two weighted graphs, $\mathcal{G}$ and $\mathcal{G}'$ are said to be neighbors if $V = V'$, $E = E'$ and $w \sim w'$.*

The so-called weight-differential privacy for graph algorithms is now formally defined.

Definition 2.3 (Sealfon (2016))**.** *For any graph topology $G = (V, E)$, let $\mathcal{A}$ be a randomized algorithm that takes as input a weight function $w \in \mathcal{W}_E$. $\mathcal{A}$ is called (ϵ, δ)-differentially private on $G = (V, E)$ if for all pairs of neighboring weight functions $w, w' \in \mathcal{W}_E$, and for all set of possible outputs S, one has*

$$\mathbb{P}[\mathcal{A}(w) \in S] \leq e^\epsilon \mathbb{P}[\mathcal{A}(w') \in S] + \delta.$$

If $\mathcal{A}$ is (ϵ, δ)-differentially private on every graph topology in a class $\mathcal{C}$, it is said to be (ϵ, δ)-differentially private on $\mathcal{C}$.

One of the first, and most used differentially private mechanisms is the Laplace mechanism. It is based on the process of releasing a numerical query perturbed by a noise drawn from a centered Laplace distribution scaled to the sensitivity of the query. We present here its graph-based reformulation.

Definition 2.4 (reformulation Dwork et al. (2006b))**.** *Given some graph topology $G = (V, E)$, for any $f_G : \mathcal{W}_E \to \mathbb{R}^k$, the sensitivity of the function is defined as $\Delta f_G = \max_{w \sim w' \in \mathcal{W}_E} ||f_G(w) - f_G(w')||_1$.*

Definition 2.5 (reformulation Dwork et al. (2006b))**.** *Given some graph topology $G = (V, E)$, any function $f_G : \mathcal{W}_E \to \mathbb{R}^k$, any $\epsilon > 0$, and $w \in \mathcal{W}_E$, the graph-based Laplace mechanism is $\mathcal{M}_L(w, f_G, \epsilon) = f_G(w) + (Y_1, \dots, Y_k)$ where Y_i are i.i.d. random variables drawn from Lap$(\Delta f_G/\epsilon)$, and Lap(b) denotes the Laplace distribution with scale b $\left(\text{i.e } \textit{probability density } \frac{1}{2b} \exp\left(-\frac{|x|}{b}\right)\right)$.*

Theorem 2.1 (Dwork et al. (2006b))**.** *The Laplace mechanism is ϵ-differentially private.*

We define hereafter the graph-based Exponential mechanism. In the sequel we refer to it simply as Exponential mechanism. The Exponential mechanism represents a way of privately answering arbitrary range queries. Given some range of possible responses to the query $\mathcal{R}$, it is defined according to a utility function $u_G := \mathcal{W}_E \times \mathcal{R} \to \mathbb{R}$, which aims at providing some total preorder on the range $\mathcal{R}$ according to the total order in $\mathbb{R}$. The sensitivity of this function is denoted $\Delta u_G := \max_{r \in \mathcal{R}} \max_{w \sim w' \in \mathcal{W}_E} |u_G(w, r) - u_G(w', r)|$.

Definition 2.6. *Given some graph topology $G = (V, E)$, some output range $\mathcal{R} \subset E$, some privacy parameter $\epsilon > 0$, some utility function $u_G := \mathcal{W}_E \times \mathcal{R} \to \mathbb{R}$, and some $w \in \mathcal{W}_E$ the graph-based Exponential mechanism $\mathcal{M}_{Exp}(G, w, u_G, \mathcal{R}, \epsilon)$ selects and outputs an element $r \in \mathcal{R}$ with probability proportional to* $\exp\left(\frac{\epsilon u_G(w,r)}{2\Delta u_G}\right)$.

The Exponential mechanism defines a distribution on a potentially complex and large range $\mathcal{R}$. As the following theorem states, sampling from such a distribution preserves ϵ-differential privacy.

Theorem 2.2 (reformulation McSherry and Talwar (2007))**.** *For any non-empty range $\mathcal{R}$, given some graph topology $G = (V, E)$, the graph-based Exponential mechanism preserves ϵ-differential privacy,* i.e *if $w \sim w' \in \mathcal{W}_E$,*

$$\mathbb{P}\left[\mathcal{M}_{Exp}(G, w, u_G, \mathcal{R}, \epsilon) = r\right] \leq e^{\epsilon} \mathbb{P}\left[\mathcal{M}_{Exp}(G, w', u_G, \mathcal{R}, \epsilon) = r\right].$$

Further, Th 2.3 highlights the trade-off between privacy and accuracy for the Exponential mechanism when $0 < |\mathcal{R}| < +\infty$. Th 2.4 presents the ability of differential privacy to comply with composition while Th 2.5 introduces its post-processing property.

Theorem 2.3 (reformulation Dwork and Roth (2013))**.** *Given some graph topology $G = (V, E)$, some $w \in \mathcal{W}_E$, some output range $\mathcal{R}$, some privacy parameter $\epsilon > 0$, some utility function $u_G := \mathcal{W}_E \times \mathcal{R} \to \mathbb{R}$, and denoting $OPT_{u_G}(w) = \max_{r \in \mathcal{R}} u_G(w, r)$, one has $\forall\, t \in \mathbb{R}$,*

$$u_G\left(G, w, \mathcal{M}_{Exp}(w, u_G, \mathcal{R}, \epsilon)\right) \leq OPT_{u_G}(w) - \frac{2\Delta u_G}{\epsilon}\left(t + \ln|\mathcal{R}|\right)$$

with probability at most $\exp(-t)$.

Theorem 2.4 (Dwork et al. (2006a))**.** *For any $\epsilon > 0$, $\delta \geq 0$ the adaptive composition of k (ϵ, δ)-differentially private mechanisms is $(k\epsilon, k\delta)$-differentially private.*

Theorem 2.5 (Post-Processing Dwork and Roth (2013))**.** *Let $\mathcal{A} : \mathcal{W}_E \to B$ be a randomized algorithm that is (ϵ, δ)-differentially private, and $h : B \to B'$ a deterministic mapping. Then $h \circ \mathcal{A}$ is (ϵ, δ)-differentially private.*

2.3 DIFFERENTIALLY-PRIVATE CLUSTERING

Differentially private clustering for unstructured datasets has been first discussed in Nissim et al. (2007). This work introduced the first method for differentially private clustering based on the k-means algorithm. Since then most of the work in the field focused on adaptation of this method (Blum et al., 2008; McSherry, 2009; Dwork, 2011). The main drawback of this work is that it is not able to deal with arbitrary shaped clusters. This issue has been recently investigated in Ho and Ruan (2013) and Chen et al. (2015). They proposed two new methods to find arbitrary shaped clusters in unstructured datasets respectively based on density clustering and wavelet decomposition. Even though both of them allow one to produce non-convex clusters, they only deal with unstructured datasets and thus are not applicable to node clustering in a graph. Our work focuses on node clustering in a graph under weight-differential privacy. Graph clustering has already been investigated in a topology-based privacy framework (Mülle et al., 2015; Nguyen et al., 2016), however, these works do not consider weight-differential privacy. Our work is, to the best of our knowledge, the first attempt to define node clustering in a graph under weight differential privacy.

3 DIFFERENTIALLY-PRIVATE TREE-BASED CLUSTERING

We aim at producing a private clustering method while providing bounds on the accuracy loss. Our method is an adaptation of an existing clustering algorithm DBMSTCLU. However, to provide theoretical guarantees under differential privacy, one needs to rely on the same kind of guarantees in the non-private setting. Morvan et al. (2017) did not bring them in their initial work. Hence, our second contribution is to demonstrate the accuracy of this method, first in the non-private context.

In the following we present 1) the theoretical framework motivating MST-based clustering methods, 2) accuracy guarantees of DBMSTCLU in the non-private setting, 3) PTCLUST our private clustering algorithm, 4) its accuracy under differential privacy constraints.

3.1 THEORETICAL FRAMEWORK FOR MST-BASED CLUSTERING METHODS

MST-based clustering methods, however efficient, lack proper motivation. This Section closes this gap by providing a theoretical framework for MST-based clustering. In the sequel, notations from Section 2.1 are kept. The minimum path distance between two nodes in the graph is defined which enables to explicit our notion of

Cluster.

Definition 3.1 (Minimum path distance)**.** *Let be $\mathcal{G} = (V, E, w)$ and u, $v \in V$. The minimum path distance between u and v is*

$$d(u,v) = \min_{\mathcal{P}_{u-v}} \sum_{e \in \mathcal{P}_{u-v}} w(e)$$

with $\mathcal{P}_{u-v}$ a path (edge version) from u to v in $\mathcal{G}$.

Definition 3.2 (Cluster)**.** *Let be $\mathcal{G} = (V, E, w)$, $0 < w(e) \leq 1\ \forall e \in E$ a graph, (V, d) a metric space based on the minimum path distance d defined on $\mathcal{G}$ and $D \subset V$ a node set. $C \subset D$ is a cluster iff. $|C| > 2$ and $\forall C_1, C_2$ s.t. $C = C_1 \cup C_2$ and $C_1 \cap C_2 = \emptyset$, one has:*

$$\underset{z \in D \setminus C_1}{\operatorname{argmin}} \{ \min_{v \in C_1} d(z,v) \} \subset C_2$$

Assuming that a cluster is built of at least 3 points makes sense since singletons or groups of 2 nodes can be legitimately considered as noise. For simplicity of the proofs, the following theorems hold in the case where noise is neglected. However, they are still valid in the setting where noise is considered as singletons (with each singleton representing a generalized notion of cluster).

Theorem 3.1. *Let be $\mathcal{G} = (V, E, w)$ a graph and $\mathcal{T}$ a minimum spanning tree of $\mathcal{G}$. Let also be C a cluster in the sense of Def. 3.2 and two vertices $v_1, v_2 \in C$. Then, $V_{\mathcal{P}_{v_1-v_2}} \subset C$ with $\mathcal{P}_{v_1-v_2}$ a path from v_1 to v_2 in $\mathcal{G}$, and $V_{\mathcal{P}_{v_1-v_2}}$ the set of vertices contained in $\mathcal{P}_{v_1-v_2}$.*

Proof. Let be $v_1, v_2 \in C$. If v_1 and v_2 are neighbors, the result is trivial. Otherwise, as $\mathcal{T}$ is a tree, there exist a unique path within $\mathcal{T}$ between v_1 and v_2 denoted by $\mathcal{P}_{v_1-v_2} = \{v_1, \ldots, v_2\}$. Let now prove by *reductio ad absurdum* that $V_{\mathcal{P}_{v_1-v_2}} \subset C$. Suppose there is $h \in V_{\mathcal{P}_{v_1-v_2}}$ s.t. $h \notin C$. We will see that it leads to a contradiction. We set C_1 to be the largest connected component (regarding the number of vertices) of $\mathcal{T}$ s.t. $v_1 \in C_1$, and every nodes from C_1 are in C. Because of h's definition, $v_2 \notin C_1$. Let be $C_2 = C \setminus C_1$. $C_2 \neq \emptyset$ since $v_2 \in C_2$. Let be $z^* \in \underset{z \in V \setminus C_1}{\operatorname{argmin}} \{ \min_{v \in C_1} d(z,v) \}$ and $e^* = (z^*, v^*)$ an edge that reaches this minimum. Let us show that $z^* \notin C$. If $z^* \in C$, then two possibilities hold:

1. There is an edge $e_{z^*} \in \mathcal{T}$, s.t. $e_{z^*} = (z^*, z')$ with $z' \in C_1$. This is impossible, otherwise by definition of a connected component, $z^* \in C_1$. Contradiction.
2. For all $e_{z^*} = (z^*, z')$ s.t $z' \in C_1$, one has $e_{z^*} \notin \mathcal{T}$. In particular $e^* \notin \mathcal{T}$. Since h is the neighbor of C_1 in $\mathcal{G}$ there is also $e_h \in \mathcal{T}$, s.t. $e_h = (h, h')$ with $h' \in C_1$. Once again two possibilities hold:
   (a) $w(e_{z^*}) = \min_{z \in V \setminus C_1} \{ \min_{v \in C_1} d(z,v) \} < w(e_h)$. Then, if we replace e_h by e_{z^*} in $\mathcal{T}$, its total weight decreases. So $\mathcal{T}$ is not a minimum spanning tree. Contradiction.
   (b) $w(e_{z^*}) = w(e_h)$, therefore $h \in \underset{z \in V \setminus C_1}{\operatorname{argmin}} \{ \min_{v \in C_1} d(z,v) \}$. Since $h \notin C$, one gets that $\underset{z \in V \setminus C_1}{\operatorname{argmin}} \{ \min_{v \in C_1} d(z,v) \} \not\subset C_2$. Thus, C is not a cluster. Contradiction.

We proved that $z^* \notin C$. In particular, $z^* \notin C_2$. Then, $\underset{z \in V \setminus C_1}{\operatorname{argmin}} \{ \min_{v \in C_1} d(z,v) \} \not\subset C_2$. Thus, C is not a cluster. Contradiction. Finally $h \in C$ and $V_{\mathcal{P}_{v_1-v_2}} \subset C$. □

This theorem states that, given a graph $\mathcal{G}$, an MST $\mathcal{T}$, and any two nodes of C, every node in the path between them is in C. This means that a cluster can be characterized by a subtree of $\mathcal{T}$. It justifies the use of all MST-based methods for data clustering or node clustering in a graph. All the clustering algorithms based on successively cutting edges in an MST to obtain a subtree forest are meaningful in the sense of Th.3.1. In particular, this theorem holds for the use of DBMSTCLU (Morvan et al., 2017) presented in Section 3.2.1.

3.2 DETERMINISTIC MST-BASED CLUSTERING

This Section introduces DBMSTCLU (Morvan et al., 2017) that will be adapted to be differentially-private, and provide accuracy results on the recovery of the ground-truth clustering partition.

3.2.1 DBMSTCLU algorithm

Let us consider $\mathcal{T}$ an MST of $\mathcal{G}$, as the unique input of the clustering algorithm DBMSTCLU. The clustering partition results then from successive cuts on $\mathcal{T}$ so that a new cut in $\mathcal{T}$ splits a connected component into two new ones. Each final connected component, a subtree of $\mathcal{T}$, represents a cluster. Initially, $\mathcal{T}$ is one cluster containing all nodes. Then, at each iteration, an edge is cut if some criterion, called *Validity Index of a Clustering Partition* (DBCVI) is improved. This edge is greedily chosen to locally maximize the DBCVI at each step. When no improvement on DBCVI can be further made, the algorithm stops. The DBCVI is defined as the weighted average of all *cluster validity indices* which are based on two positive quantities, the *Dispersion* and the *Separation* of a cluster:

Definition 3.3 (Cluster Dispersion)**.** *The Dispersion of a cluster C_i (*DISP*) is defined as the maximum edge*

weight of C_i. If the cluster is a singleton (i.e. contains only one node), the associated Dispersion is set to 0*. More formally:*

$$\forall i \in [K],\ \mathrm{DISP}(C_i) = \begin{cases} \max\limits_{j,\ e_j \in C_i} w_j & \text{if } |E(C_i)| \neq 0 \\ 0 & \text{otherwise.} \end{cases}$$

Definition 3.4 (Cluster Separation)**.** *The Separation of a cluster C_i (*SEP*) is defined as the minimum distance between the nodes of C_i and the ones of all other clusters $C_j, j \neq i, 1 \leq i, j \leq K, K \neq 1$ where K is the total number of clusters. In practice, it corresponds to the minimum weight among all already cut edges from $\mathcal{T}$ comprising a node from C_i. If $K = 1$, the Separation is set to* 1*. More formally, with $incCuts(C_i)$ denoting cut edges incident to C_i,*

$$\forall i \in [K],\ \mathrm{SEP}(C_i) = \begin{cases} \min\limits_{j,\ e_j \in incCuts(C_i)} w_j & \text{if } K \neq 1 \\ 1 & \text{otherwise.} \end{cases}$$

Definition 3.5 (Validity Index of a Cluster)**.** *The Validity Index of a cluster C_i is defined as:*

$$V_C(C_i) = \frac{\mathrm{SEP}(C_i) - \mathrm{DISP}(C_i)}{\max(\mathrm{SEP}(C_i), \mathrm{DISP}(C_i))} \in [-1; 1]$$

Definition 3.6 (Validity Index of a Clustering Partition)**.** *The Density-Based Validity Index of a Clustering partition $\Pi = \{C_i\}, 1 \leq i \leq K,$* DBCVI(Π) *is defined as the weighted average of the Validity Indices of all clusters in the partition where N is the number of vertices.*

$$\mathrm{DBCVI}(\Pi) = \sum_{i=1}^{K} \frac{|C_i|}{N} V_C(C_i) \in [-1, 1]$$

DBMSTCLU is summarized in Algorithm 1: $\mathrm{evaluateCut}(.)$ computes the DBCVI when the cut in parameter is applied to $\mathcal{T}$. Initial DBCVI is set -1. Interested reader could refer to (Morvan et al., 2017) Section 4. for a complete insight on this notions.

Algorithm 1 DBMSTCLU($\mathcal{T}$)

Input: $\mathcal{T}$, the MST
$dbcvi \leftarrow -1.0$
$clusters \leftarrow \emptyset$
$cut\_list \leftarrow \{E(\mathcal{T})\}$
while $dbcvi < 1.0$ **do**
  $cut\_tp \leftarrow \emptyset$
  $dbcvi\_tp \leftarrow dbcvi$
  for each cut in cut_list **do**
    $newDbcvi = \mathrm{evaluateCut}(\mathcal{T}, cut)$
    if $newDbcvi \geq dbcvi\_tp$ **then**
      $cut\_tp \leftarrow cut$
      $dbcvi\_tp \leftarrow newDbcvi$
  if $cut\_tp \neq \emptyset$ **then**
    $clusters = \mathrm{cut}(clusters, cut\_tp)$
    $dbcvi \leftarrow dbcvi\_tp$
    $cut\_list \leftarrow cut\_list \backslash \{cut\_tp\}$
  else
    break
return $clusters$

3.2.2 DBMSTClu exact clustering recovery proof

In this section, we provide theoretical guarantees for the cluster recovery accuracy of DBMSTClu. Let us first begin by introducing some definitions.

Definition 3.7 (Cut)**.** *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K clusters, $\mathcal{T}$ an MST of $\mathcal{G}$. Let denote $(C_i^*)_{i \in [K]}$ the set of the clusters. Then, $Cut_{\mathcal{G}}(\mathcal{T}) := \{e_{kl} \in \mathcal{T} \mid k \in C_i^*,\ l \in C_j^*,\ i, j \in [K]^2,\ i \neq j\}$. In the sequel, for simplicity, we denote $e^{(ij)} \in Cut_{\mathcal{G}}(\mathcal{T})$ the edge between cluster C_i^* and C_j^*.*

$Cut_{\mathcal{G}}(\mathcal{T})$ is basically the set of effective cuts to perform on $\mathcal{T}$ in order to ensure the exact recovery of the clustering partition. More generally, trees on which $Cut_{\mathcal{G}}(.)$ enables to find the right partition are said to be a partitioning topology.

Definition 3.8 (Partitionning topology)**.** *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K clusters $C_1^*, \ldots, C_K^*$. A spanning tree $\mathcal{T}$ of $\mathcal{G}$ is said to have a partitioning topology if $\forall i, j \in [K]$, $i \neq j$, $|\{e = (u, v) \in Cut_{\mathcal{G}}(\mathcal{T}) \mid u \in C_i^*, v \in C_j^*\}| = 1$.*

Def. 3.7 and 3.8 introduce a topological condition on the tree as input of the algorithm. Nevertheless, conditions on weights are necessary too. Hence, we define homogeneous separability which expresses the fact that within a cluster the edge weights are spread in a controlled manner.

Definition 3.9 (Homogeneous separability condition)**.** *Let us consider a graph $\mathcal{G} = (V, E, w)$, $s \in E$ and $\mathcal{T}$ a tree of $\mathcal{G}$. $\mathcal{T}$ is said to be homogeneously separable by s, if*

$$\alpha_{\mathcal{T}} \max_{e \in E(\mathcal{T})} w(e) < w(s) \text{ with } \alpha_{\mathcal{T}} = \frac{\max\limits_{e \in E(\mathcal{T})} w(e)}{\min\limits_{e \in E(\mathcal{T})} w(e)} \geq 1.$$

One will write for simplicity that $H_{\mathcal{T}}(s)$ is verified.

Definition 3.10 (Weak homogeneity condition of a Cluster)**.** *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K clusters $C_1^*, \ldots, C_K^*$. A given cluster C_i^*, $i \in [K]$, C_i^* is weakly homogeneous if: for all $\mathcal{T}$ an MST of $\mathcal{G}$, and $\forall j \in [K]$, $j \neq i$, s.t. $e^{(ij)} \in Cut_{\mathcal{G}}(\mathcal{T})$, $H_{\mathcal{T}_{|C_i^*}}(e^{(ij)})$ is verified. For simplicity, one denote $\underline{\alpha}_i = \max\limits_{\mathcal{T} \text{ MST of } \mathcal{G}} \alpha_{\mathcal{T}_{|C_i^*}}$*

Definition 3.11 (Strong homogeneity condition of a Cluster)**.** *Let us consider a graph $\mathcal{G} = (V, E, w)$ with*

K clusters $C_1^*, \ldots, C_K^*$. A given cluster C_i^*, $i \in [K]$, C_i^* is strongly homogeneous if: for all $\mathcal{T}$ a spanning tree (ST) of $\mathcal{G}$, and $\forall j \in [K]$, $j \neq i$, s.t. $e^{(ij)} \in Cut_{\mathcal{G}}(\mathcal{T})$, $H_{\mathcal{T}_{|C_i^*}}(e^{(ij)})$ is verified. For simplicity, one denote $\bar{\alpha}_i = \max_{\mathcal{T} \text{ ST of } \mathcal{G}} \alpha_{\mathcal{T}_{|C_i^*}}$

Weak homogeneity is indeed really natural considering the non-private cases using an MST. Strong homogeneity is more demanding, but still a reachable condition. Section 4 presents an experiment where the graph respects strong homogeneity as well as being organised in arbitrary shaped clusters. We show that the weak homogeneity condition is implied by the strong homogeneity condition.

Proposition 3.1. *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K clusters $C_1^*, \ldots, C_K^*$. If a given cluster C_i^*, $i \in [K]$ is strongly homogeneous, then, it is weakly homogeneous.*

Proof. If $\mathcal{T}$ a spanning tree of $\mathcal{G}$, and $\forall j \in [K]$, $j \neq i$, s.t. $e^{(ij)} \in Cut_{\mathcal{G}}(\mathcal{T})$, $H_{\mathcal{T}_{|C_i^*}}(e^{(ij)})$ is verified, then in particular, it is true for any MST. □

Strong homogeneity condition appears to be naturally more constraining on the edge weights than the weak one. The accuracy of DBMSTCLU is proved under the weak homogeneity condition, while the accuracy of its differentially-private version is only given under the strong homogeneity condition.

Theorem 3.2. *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K homogeneous clusters $C_1^*, \ldots, C_K^*$ and $\mathcal{T}$ an MST of $\mathcal{G}$. Let now assume that at step $k < K - 1$, DBMSTClu built $k + 1$ subtrees $\mathcal{C}_1, \ldots, \mathcal{C}_{k+1}$ by cutting $e_1, e_2, \ldots, e_k \in E$.*

Then, $Cut_k := Cut_{\mathcal{G}}(\mathcal{T}) \setminus \{e_1, e_2, \ldots, e_k\} \neq \emptyset \implies \text{DBCVI}_{k+1} \geq DBCVI_k$, i.e. if there are still edges in Cut_k, the algorithm will continue to perform some cut.

Proof. See supplementary material. □

Theorem 3.3. *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K homogeneous clusters $C_1^*, \ldots, C_K^*$ and $\mathcal{T}$ an MST of $\mathcal{G}$.*

Assume now that at step $k < K-1$, DBMSTClu built $k+1$ subtrees $\mathcal{C}_1, \ldots, \mathcal{C}_{k+1}$ by cutting $e_1, e_2, \ldots, e_k \in E$. We still denote $Cut_k := Cut_{\mathcal{G}}(\mathcal{T}) \backslash \{e_1, e_2, \ldots, e_k\}$.

If $Cut_k \neq \emptyset$ then $\operatorname{argmax}_{e \in \mathcal{T} \backslash \{e_1, e_2, \ldots, e_k\}} DBCVI_{k+1}(e) \subset Cut_k$ i.e. the cut edge at step $k + 1$ is in Cut_k.

Proof. See supplementary material. □

Theorem 3.4. *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K weakly homogeneous clusters $C_1^*, \ldots, C_K^*$ and $\mathcal{T}$ an MST of $\mathcal{G}$. Let now assume that at step $K - 1$, DBMSTClu built K subtrees $\mathcal{C}_1, \ldots, \mathcal{C}_K$ by cutting $e_1, e_2, \ldots, e_{K-1} \in E$. We still denote $Cut_{K-1} := Cut_{\mathcal{G}}(\mathcal{T}) \backslash \{e_1, e_2, \ldots, e_{K-1}\}$.*

Then, for all $e \in \mathcal{T} \backslash \{e_1, e_2, \ldots, e_{K-1}\}$, $DBCVI_K(e) < DBCVI_{K-1}$ i.e. the algorithm stops: no edge gets cut during step K.

Proof. See supplementary material. □

Corollary 3.1. *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K weakly homogeneous clusters $C_1^*, \ldots, C_K^*$ and $\mathcal{T}$ an MST of $\mathcal{G}$. $DBMSTClu(\mathcal{T})$ stops after $K - 1$ iterations and the K subtrees produced match exactly the clusters i.e. under homogeneity condition, the algorithm finds automatically the underlying clustering partition.*

Proof. Th. 3.2 and 3.4 ensure that under homogeneity condition on all clusters, the algorithm performs the $K-1$ distinct cuts within $Cut_{\mathcal{G}}(\mathcal{T})$ and stops afterwards. By definition of $Cut_{\mathcal{G}}(\mathcal{T})$, it means the DBMSTClu correctly builds the K clusters. □

3.3 PRIVATE MST-BASED CLUSTERING

This section presents our new node clustering algorithm PTCLUST for weight differential privacy. It relies on a mixed adaptation of PAMST algorithm (Pinot, 2018) for recovering a differentially-private MST of a graph and DBMSTCLU.

3.3.1 PAMST algorithm

Given a simple-undirected-weighted graph $\mathcal{G} = (V, E, w)$, PAMST outputs an almost minimal weight spanning tree topology under differential privacy constraints. It relies on a Prim-like MST algorithm, and an iterative use of the graph-based Exponential mechanism. PAMST takes as an input a weighted graph, and a utility function. It outputs the topology of a spanning tree which weight is almost minimal. Algorithm 2 presents this new method, using the following utility function:

$$\begin{array}{rccc} u_G & : & \mathcal{W}_E \times \mathcal{R} & \to & \mathbb{R} \\ & & (w, r) & \mapsto & -|w(r) - \min_{r' \in \mathcal{R}} w(r')|. \end{array}$$

PAMST starts by choosing an arbitrary node to construct iteratively the tree topology. At every iteration, it uses the Exponential mechanism to find the next edge to be added to the current tree topology while keeping the weights private. This algorithm is the state of the art to find a spanning tree topology under differential privacy. For

readability, let us introduce some additional notations. Let S be a set of nodes from G, and $\mathcal{R}_S$ the set of edges that are incident to one and only one node in S (also denoted xor-incident). For any edge r in such a set, the incident node to r that is not in S is denoted $r_{\rightarrow}$. Finally, the restriction of the weight function to an edge set $\mathcal{R}$ is denoted $w_{|\mathcal{R}}$.

Algorithm 2 PAMST(G, u_G, w, ϵ)

Input: $\mathcal{G} = (V, E, w)$ a weighted graph (separately the topology G and the weight function w), ϵ a degree of privacy and u_G utility function.
Pick $v \in V$ at random
$S_V \leftarrow \{v\}$
$S_E \leftarrow \emptyset$
while $S_V \neq V$ **do**
  $r = \mathcal{M}_{Exp}(\mathcal{G}, w, u_G, \mathcal{R}_{S_V}, \frac{\epsilon}{|V|-1})$
  $S_V \leftarrow S_V \cup \{r_{\rightarrow}\}$
  $S_E \leftarrow S_E \cup \{r\}$
return S_E

Theorem 3.5 states that using PAMST to get an almost minimal spanning tree topology preserves weight-differential privacy.

Theorem 3.5. *Let $G = (V, E)$ be the topology of a simple-undirected graph, then $\forall \epsilon > 0$,* PAMST$(G, u_G, \bullet, \epsilon)$ *is ϵ- differentially private on G.*

3.3.2 Differentially-private clustering

The overall goal of this Section is to show that one can obtain a differentially-private clustering algorithm by combining PAMST and DBMSTCLU algorithms. However, PAMST does not output a weighted tree which is inappropriate for clustering purposes. To overcome this, one could rely on a sanitizing mechanism such as the Laplace mechanism. Moreover, since DBMSTCLU only takes weights from (0,1], two normalizing parameters τ and p are introduced, respectively to ensure lower and upper bounds to the weights that fit within DBMSTCLU needs. This sanitizing mechanism is called the Weight-Release mechanism. Coupled with PAMST, it will allows us to produce a weighted spanning tree with differential privacy, that will be exploited in our private graph clustering.

Definition 3.12 (Weight-Release mechanism)**.** *Let $\mathcal{G} = (G, w)$ be a weighted graph, $\epsilon > 0$ a privacy parameter, s a scaling parameter, $\tau \geq 0$, and $p \geq 1$ two normalization parameters. The Weight-Release mechanism is defined as*

$$\mathcal{M}_{w.r}(G, w, s, \tau, p) = \left(G, w' = \frac{w + (Y_1, ..., Y_{|E|}) + \tau}{p}\right)$$

where Y_i are i.i.d. random variables drawn from $Lap(0, s)$. With $w + (Y_1, ..., Y_{|E|})$ meaning that if one gives an arbitrary order to the edges $E = (e_i)_{i \in [|E|]}$, one has $\forall i \in [|E|]$, $w'(e_i) = w(e_i) + Y_i$.

The following theorem presents the privacy guarantees of the Weight-Release mechanism.

Theorem 3.6. *Let $G = (V, E)$ be the topology of a simple-undirected graph, $\tau \geq 0$, $p \geq 1$, then $\forall \epsilon > 0$, $\mathcal{M}_{w.r}\left(G, \bullet, \frac{\mu}{\epsilon}, \tau, p\right)$ is ϵ- differentially private on G.*

Proof. Given $\tau \geq 0$, $p \geq 1$, and $\epsilon > 0$, the Weight release mechanism scaled to $\frac{\mu}{\epsilon}$ can be break down into a Laplace mechanism and a post-processing consisting in adding τ to every edge and dividing them by p. Using Theorems 2.1 and 2.5, one gets the expected result. □

So far we have presented DBMSTCLU and PAMST algorithms, and the Weight-Release mechanism. Let us now introduce how to compose those blocks to obtain a Private node clustering in a graph, called PTCLUST. The

Algorithm 3 PTCLUST$(G, w, u_G, \epsilon, \tau, p)$

Input: $\mathcal{G} = (V, E, w)$ a weighted graph (separately the topology G and the weight function w), ϵ a degree of privacy and u_G utility function.
$T =$ PAMST$(G, w, u_G, \epsilon/2)$
$\mathcal{T}' = \mathcal{M}_{w.r}(T, w_{|E(T)}, \frac{2\mu}{\epsilon}, \tau, p)$
return DBMSTCLU$(\mathcal{T}')$

algorithm 3 takes as an input a weighted graph (dissociated topology and weight function), a utility function, a privacy degree and two normalization parameters. It outputs a clustering partition. To do so, a spanning tree topology is produced using PAMST with time and space complexities respectively equal to $O(|V|^2)$ and $O(|E|)$. Afterward a randomized and normalized version of the associated weight function is released using the Weight-release mechanism. Finally the obtained weighted tree is given as an input to DBMSTCLU that performs a clustering partition with $O(|V|)$ time and space complexities. The following theorem ensures that our method preserves ϵ-differential privacy.

Theorem 3.7. *Let $G = (V, E)$ be the topology of a simple-undirected graph, $\tau \geq 0$, and $p \geq 1$, then $\forall \epsilon > 0$,* PTCLUST$(G, \bullet, u_G, \epsilon, \tau, p)$ *is ϵ-differentially private on G.*

Proof. Using Theorem 3.5 one has that T is produced with $\epsilon/2$-differential privacy, and using Theorem 3.6 one has that w' is obtained with $\epsilon/2$-differential privacy as

well. Therefore using Theorem 2.4, $\mathcal{T}'$ is released with ϵ-differential privacy. Using the post-processing property (Theorem 2.5) one gets the expected result. □

3.4 DIFFERENTIAL PRIVACY TRADE-OFF OF CLUSTERING

The results stated in this section present the security/accuracy trade-off of our new method in the differentially-private framework. PTCLUST relies on two differentially -private mechanisms, namely PAMST and the Weight-Release mechanism. Evaluating the accuracy of this method amounts to check whether using these methods for ensuring privacy does not deteriorate the final clustering partition. The accuracy is preserved if PAMST outputs the same topology as the MST-based clustering and if the Weight-Release mechanism preserves enough the weight function. According to Def. 3.8, if a tree has a partitioning topology, then it fits the tree-based clustering. The following theorem states that with high probability PAMST outputs a tree with a partitioning topology.

Theorem 3.8. *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K strongly homogeneous clusters $C_1^*, \ldots, C_K^*$ and $T = \text{PAMST}(\mathcal{G}, u_{\mathcal{G}}, w, \epsilon)$, $\epsilon > 0$. T has a partitioning topology with probability at least*

$$1 - \sum_{i=1}^{K} (|C_i^*| - 1) \exp\left(-\frac{A}{2\Delta u_{\mathcal{G}}(|V| - 1)}\right)$$

with $A = \epsilon \left(\bar{\alpha}_i \max_{e \in E\left(\mathcal{G}_{|C_i^*}\right)} (w(e)) - \min_{e \in E\left(\mathcal{G}_{|C_i^*}\right)} (w(e)) \right) + \ln |E|.$

Proof. See supplementary material. □

The following theorem states that given a tree $\mathcal{T}$ under the strong homogeneity condition, if the subtree associated to a cluster respects Def. 3.9, then it still holds after applying the Weight-Release mechanism to this tree.

Theorem 3.9. *Let us consider a graph $\mathcal{G} = (V, E, w)$ with K strongly homogeneous clusters $C_1^*, \ldots, C_K^*$ and $T = PAMST(\mathcal{G}, u_{\mathcal{G}}, w, \epsilon)$, $\mathcal{T} = (T, w_{|T})$ and $\mathcal{T}' = \mathcal{M}_{w.r}(T, w_{|T}, s, \tau, p)$ with $s \ll p, \tau$. Given some cluster C_i^*, and $j \neq i$ s.t $e^{(ij)} \in Cut_{\mathcal{G}}(\mathcal{T})$, if $H_{\mathcal{T}_{|C_i^*}}(e^{(ij)})$ is verified, then $H_{\mathcal{T}'_{|C_i^*}}(e^{(ij)})$ is verified with probability at least*

$$1 - \frac{\mathbb{V}(\varphi)}{\mathbb{V}(\varphi) + \mathbb{E}(\varphi)^2}$$

with the following notations :

- $\varphi = (\max_{j \in [|C_i^*| - 1]} Y_j)^2 - \min_{j \in [|C_i^*| - 1]} Z_j \times X^{out}$
- $Y_j \underset{iid}{\sim} Lap\left(\frac{\max_{e \in E(\mathcal{T})} w(e) + \tau}{p}, \frac{s}{p}\right)$
- $Z_j \underset{iid}{\sim} Lap\left(\frac{\min_{e \in E(\mathcal{T})} w(e) + \tau}{p}, \frac{s}{p}\right)$
- $X^{out} \sim Lap\left(\frac{w(e^{(ij)}) + \tau}{p}, \frac{s}{p}\right),$

Proof. See supplementary material. □

Note that Theorem 3.9 is stated in a simplified version. A more complete version (specifying an analytic version of $\mathbb{V}(\varphi)$ and $\mathbb{E}(\varphi)$) is given in the supplementary material.

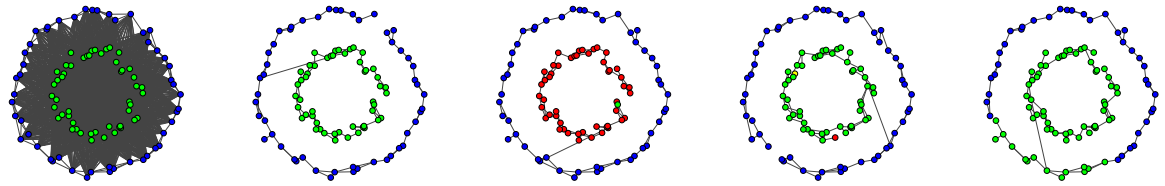
4 EXPERIMENTS

So far we have exhibited the trade-off between clustering accuracy and privacy and we experimentally illustrate it with some qualitative results. We have performed experiments on two classical synthetic graph datasets for clustering with nonconvex shapes: two concentric circles and two moons, both in their noisy versions. For the sake of readability and for visualization purposes, both graph datasets are embedded into a two dimensional Euclidean space. Each dataset contains 100 data nodes that are represented by a point of two coordinates. Both graphs have been built with respect to the strong homogeneity condition: edge weights within clusters are between $w_{min} = 0.1$ and $w_{max} = 0.3$ while edges between clusters have a weight strictly above $w_{max}^2 / w_{min} = 0.9$. In practice, the complete graph has trimmed from its irrelevant edges (*i.e.* not respecting the strong homogeneity condition). Hence, those graphs are not necessarily Euclidean since close nodes in the visual representation may not be connected in the graph. Finally, weights are normalized between 0 and 1.

Figures 1 and 2 (best viewed in color) show for each dataset (a) the original homogeneous graph $\mathcal{G}$ built by respecting the homogeneity condition, (b) the clustering partition[1] of DBMSTCLU with the used underlying MST, the clustering partitions for PTCLUST with $\mu = 0.1$ obtained respectively with different privacy degrees[2] : (c) $\epsilon = 0.5$, (d) $\epsilon = 0.7$ and (e) $\epsilon = 1.0$. The utility function $u_{\mathcal{G}}$ corresponds to the graph weight. Each experiment is carried out independently and the tree topology obtained by PAMST will eventually be different. This explains why the edge between clusters may not be the same when the experiment is repeated with a

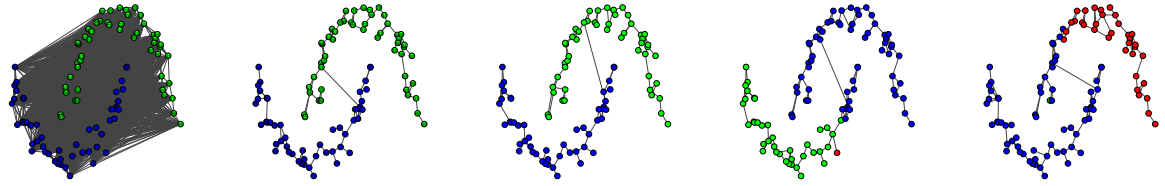
[1] For the sake of clarity, the edges in those Figures are represented based on the original weights and not on the privately released weights.

[2] Note that, although the range of ϵ is in $\mathbb{R}_+^\star$, it is usually chosen in practice in $(0, 1]$ (Dwork and Roth, 2013, Chap 1&2).



(a) Homogeneous graph (b) DBMSTCLU (c) PTCLUST, $\epsilon = 1.0$ (d) PTCLUST, $\epsilon = 0.7$ (e) PTCLUST, $\epsilon = 0.5$

Figure 1: Circles experiments for $n = 100$. PTCLUST parameters: $w_{min} = 0.1$, $w_{max} = 0.3$, $\mu = 0.1$.



(a) Homogeneous graph (b) DBMSTCLU (c) PTCLUST, $\epsilon = 1.0$ (d) PTCLUST, $\epsilon = 0.7$ (e) PTCLUST, $\epsilon = 0.5$

Figure 2: Moons experiments for $n = 100$. PTCLUST parameters: $w_{min} = 0.1$, $w_{max} = 0.3$, $\mu = 0.1$.

different level of privacy. However, this will marginally affect the overall quality of the clustering.

As expected, DBMSTCLU recovers automatically the right partition and the results are shown here for comparison with PTCLUST. For PTCLUST, the true MST is replaced with a private approximate MST obtained for suitable τ and p ensuring final weights between 0 and 1.

When the privacy degree is moderate ($\epsilon \in \{1.0, 0.7\}$), it appears that the clustering result is slightly affected. More precisely, in Figures 1c and 1d the two main clusters are recovered while one point is isolated as a singleton. This is due to the randomization involved in determining the edge weights for the topology returned by PAMST. In Figure 2c, the clustering is identical to the one from DBMSTCLU in Figure 2b. In Figure 1d, the clustering is very similar to the DBMSTClu one, with the exception of an isolated singleton. However, as expected from our theoretical results, when ϵ is decreasing (even below 0.5), the clustering quality deteriorates, as DBMSTCLU is sensitive to severe changes in the MST (cf. Figure 1e, 2e).

5 CONCLUSION

In this paper, we introduced PTCLUST, a novel graph clustering algorithm able to recover arbitrarily-shaped clusters while preserving differential privacy on the weights of the graph. It is based on the release of a private approximate minimum spanning tree of the graph of the dataset, by performing suitable cuts to reveal the clusters. To the best of our knowledge, this is the first differentially private graph-based clustering algorithm adapted to nonconvex clusters. The theoretical analysis exhibited a trade-off between the degree of privacy and the accuracy of the clustering result. Differential privacy is investigated in the framework of strong homogeneity but this is quite restrictive. A smoother result would be very interesting but it is more challenging. This will be a focus of our future work. Our work suits to applications where privacy is a critical issue and it could pave the way to metagenomics and genes classification using individual gene maps while protecting patient privacy. Future work will also be devoted to deeply investigate these applications.

References

T. Asano, B. Bhattacharya, M. Keil, and F. Yao. Clustering algorithms based on minimum and maximum spanning trees. In *Proceedings of the Fourth Annual Symposium on Computational Geometry*, SCG '88, pages 252–257, New York, NY, USA, 1988. ACM.

A. Blum, K. Ligett, and A. Roth. A learning theory approach to non-interactive database privacy. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pages 609–618, New York, NY, USA, 2008. ACM.

L. Chen, T. Yu, and R. Chirkova. Wavecluster with differential privacy. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pages 1011–1020, New York, NY, USA, 2015. ACM.

C. Dwork. A firm foundation for private data analysis. *Commun. ACM*, 54(1):86–95, January 2011.

C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3-4):211–407, 2013.

C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *Eurocrypt*, volume 4004, pages 486–503. Springer, 2006a.

C. Dwork, F. McSherry, K. Nissim, and A.Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography*, pages 265–284. Springer Berlin Heidelberg, 2006b.

O. Grygorash, Y. Zhou, and Z. Jorgensen. Minimum spanning tree based clustering algorithms. In *2006 18th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'06)*, pages 73–81, Nov 2006.

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, Dec 2009.

S.-S. Ho and S. Ruan. Preserving privacy for interesting location pattern mining from trajectory data. *Trans. Data Privacy*, 6(1):87–106, April 2013.

S. P. Kasiviswanathan, K. Nissim, S. Raskhodnikova, and A. Smith. Analyzing graphs with node differential privacy. In *Proceedings of the 10th Theory of Cryptography Conference on Theory of Cryptography*, TCC'13, pages 457–476, Berlin, Heidelberg, 2013. Springer-Verlag.

F. McSherry. Privacy integrated queries. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data (SIGMOD)*. Association for Computing Machinery, Inc., June 2009.

F. McSherry and K. Talwar. Mechanism design via differential privacy. In *Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, Providence, RI, October 2007. IEEE.

A. Morvan, K. Choromanski, C. Gouy-Pailler, and J. Atif. Graph sketching-based massive data clustering. *SIAM Data Mining 2018 (to appear)*, 2017.

Y. Mülle, C. Clifton, and K. Böhm. Privacy-integrated graph clustering through differential privacy. In *EDBT/ICDT Workshops*, 2015.

H. H. Nguyen, A. Imine, and M. Rusinowitch. Detecting communities under differential privacy. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*, WPES '16, pages 83–93, New York, NY, USA, 2016. ACM.

K. Nissim, S. Raskhodnikova, and A. Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing - STOC*. ACM Press, 2007.

R. Pinot. Minimum spanning tree release under differential privacy constraints. *ArXiv e-prints*, January 2018.

S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27 – 64, 2007.

A. Sealfon. Shortest paths and distances with differential privacy. In *Proceedings of the 35th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems - PODS*. ACM Press, 2016.

C. T. Zahn. Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Trans. Comput.*, 20 (1):68–86, January 1971.