VIP Blowfish Privacy in Communication Graphs

Mohamed Nassar[1][a], Elie Chicha[2,3], Bechara AL Bouna[2] and Richard Chbeir[3]

[1]*Computer Science Department, American University of Beirut, Lebanon*

[2]*TICKET Lab., Antonine University, Hadat-Baabda, Lebanon*

[3]*Univ. Pau & Pays Adour, UPPA - E2S, LIUPPA, Anglet, France*

Keywords: Blowfish Privacy, Social Networks.

Abstract: Communication patterns analysis is becoming crucial for global health security especially with the spread of epidemics such as COVID-19 by the means of social contact. At the same time, personal privacy is considered an essential human right. Privacy-preserving frameworks enable communication graph analysis within formal privacy guarantees. In this paper, we present a summary of Blowfish privacy and explore the possibility of applying it in the context of undirected communication graphs. Communication graphs represent social contact or call detail records databases. We define the notions of neighborhood, discriminative secrets, and policies for these graphs. We study several examples of queries and compute their sensitivity. Even though not addressed in the original Blowfish privacy paper, we explore the idea of having a discriminative secret graph per individual. This allows us to treat some persons as VIP and put their privacy on top priority, where other persons can have lower privacy constraints. This may help to offer privacy as a service and increase the utility of the anonymized communication graph to an appropriate level.

1 INTRODUCTION

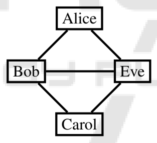
Communication patterns analysis is becoming crucial for global health security especially with the spread of epidemics such as COVID-19 by the means of social contact. In the same time, personal privacy is considered as an essential human right. Privacy-preserving frameworks enable communication graph analysis within formal privacy guarantees. Differential Privacy (DP) provides ways for trading-off the privacy of individuals in a statistical database for the utility of data analysis.

DP has a single tuning knob, namely ε, sometimes two (ε and δ). For example, increasing ε means more utility and less privacy. The idea of Blowfish privacy (BP) is to provide more tuning knobs by introducing policies He et al. (2014). In BP, a *policy* specifies:

- *secrets*: information that must be kept secret. Since not all the information has to be secret, we can increase the utility of the data by lessening the protection of certain properties.
- and *constraints*: known properties about the data. Constraints add protection against an adversary who knows these constraints.

DP can be considered as an instance of BP where:

[a] https://orcid.org/0000-0001-8857-4436



id	tuple
Bob	(0, 1, 1, 1)
Alice	(1, 0, 1, 0)
Eve	(1, 1, 0, 1)
Carol	(1, 0, 1, 0)

Figure 1: Communication graph and its database.

- every property about an individual's record is protected,
- every individual is independent of all the other individuals in the dataset. There is no correlations.

Because of its generalized framework and powerful expressiveness of adversarial knowledge, we expect that BP can solve privacy challenges in graph-based databases. In this paper, we explore the application of BP to communication graphs such as social networks and call detail records databases. We model the secrets and the auxiliary knowledge in terms of the BP model and give numerous examples.

2 BP FOR COMMUNICATION GRAPHS

A communication graph is a graph where vertices represent individuals and an edge between two in-

Nassar, M., Chicha, E., Al Bouna, B. and Chbeir, R.
VIP Blowfish Privacy in Communication Graphs.
DOI: 10.5220/0009875704590467
In *Proceedings of the 17th International Joint Conference on e-Business and Telecommunications (ICETE 2020) - SECRYPT*, pages 459-467
ISBN: 978-989-758-446-6

dividuals exists if a communication has happened in between the individuals corresponding to the vertices. Social networks and call detail records can be modeled as communication graphs. One way to anonymize the data of a communication graph is to remove the identifiers at the vertices. The goal of an adversary is therefore to discover the individual corresponding to a node in the graph.

A communication graph $G_c(V_c, E_c)$ can be represented as a database D of size $n = |V_c|$ where each tuple t of D corresponds to an individual id. The tuple dimension is $m = |V_c|$ as well. The ith attribute of t is 1 if $t.\_id$ has communicated with the individual corresponding to node i, and 0 otherwise. A row in D represents the ego network of a vertex.

An example of a communication graph and its corresponding database is shown in Figure 1. Note that we consider a binary communication event (0 or 1). Other models might be explored in future work, for instance annotating the edges with call frequency, average duration, call time or other meta-data. since the BP framework is defined over categorical data, binning might be used if the meta data is not categorical.

The BP notation is based on the DP notation as summarized in Table 1.

2.1 Secrets

In addition, BP defines secrets and discriminative pairs of secrets as shown in Table 1. We give examples of secrets and pairs of secrets over a communication graph in Table 2.

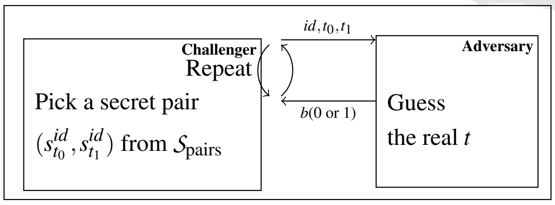


Figure 2: Discriminative pair of secrets as a game.

The *discriminative secret graph* generalizes the specification of discriminative pairs of secrets. It is a graph where vertices represent secrets and edges link only the discriminative pairs of secrets. More formally it is denoted $G = (V, E)$ where $V = \mathcal{T}$ and $E \subseteq \mathcal{T} \times \mathcal{T}$. Even though not addressed in the original BP paper, we explore the idea of having a discriminative secret graph per individual. This allows us to treat some persons as VIP and put their privacy on top priority, where other persons can have lower privacy constraints. This may help increase the utility of the communication graph to an appropriate level.

Table 1: Notation of BP, secrets, and discriminative pairs.

Symbol	Description
D	Database of n tuples
$\mathcal{T} = A_1 \times A_2 \times \ldots \times A_m$	Domain of m categorical attributes
$t \in \mathcal{T}$	A single tuple
$t.\_id$	Id of the tuple's real owner
$t.A_i$	Value of the ith attribute in tuple t
I_n	Set of all possible datasets with size n ($\lvert D \rvert = n$)
$(D_1, D_2) \in N$	In DP, D_1 and D_2 are neighbors, they differ in the value of one tuple
$\mathcal{M}$	A randomized mechanism, for example adding random noise to the result of a query
$S \subseteq$ range$(\mathcal{M})$	A set of the outputs generated by $\mathcal{M}$
ε-DP	For every S and every two neighbors (D_1, D_2): $\Pr[\mathcal{M}(D_1) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(D_2) \in S]$
$f : I_n \to \mathbb{R}^d$	A function that takes a database as input and returns a vector of real numbers as output, for example a `countIf` query
$S(f)$	The global sensitivity of f is the max Manhattan distance between the outputs for any two neighbor databases: $S(f) = \max\limits_{(D_1, D_2) \in N} \lVert f(D_1) - f(D_2) \rVert_1$
$\mathcal{M}^{\text{Lap}}$	The Laplace Mechanism adds $\eta \in \mathbb{R}^d$ to $f(D)$, where η is a vector of independent random variables. Each η_i is drawn from the Laplace distribution with parameter $S(f)/\varepsilon$: $\Pr[\eta_i = z] \propto e^{-z.\varepsilon/S(f)}$
$\mathcal{P} = (P_1, \ldots, P_k)$	A partitioning of the domain $\mathcal{T}$
$h_{\mathcal{P}} : I_n \to \mathbb{Z}^k$	A histogram query. $h_{\mathcal{P}}(D)$ outputs for each P_i the number of times values in P_i appear in D. The sensitivity of histogram queries is $S(h_{\mathcal{P}}) = 2$ since replacing a tuple by another one may decrease the count of a partition and increase the count of another partition.
$h_{\mathcal{T}}$	The complete histogram query, it outputs for each $t \in \mathcal{T}$ the number of times it appears in D
$\mathcal{E}_{\mathcal{M}}(D)$	The expected mean squared error of $\mathcal{M}$: $\mathcal{E}_{\mathcal{M}}(D) = \sum\limits_i \mathbb{E}[(f_i(D) - \bar{f}_i(D))^2]$ where $f_i(D)$ and $\bar{f}_i(D)$ are the ith components of the true answer and the noisy answer, respectively. For Laplace mechanism and histogram queries, this error is: $\mathcal{E}_{\mathcal{M}^{\text{Lap}}_{h_{\mathcal{P}}}}(D) = \lvert \mathcal{T} \rvert . \mathbb{E}(\text{Laplace}(2/\varepsilon))^2 = 8 \lvert \mathcal{T} \rvert / \varepsilon^2$. A large epsilon means less error, hence more utility.
s	An arbitrary statement over the values in the database. *Example1*: $t.\_id$ = 'Bob' $\wedge$ $t.disease$ = 'cancer'. *Example2*: $t.\_id$ = 'Bob' $\wedge$ $t_2.\_id$ = 'Alice' $\wedge$ $t_1.disease = t_2.disease$
S	A set of secrets that the data owner would like to protect, e.g. $\{$*Example1*, *Example2*$\}$

Table 1: Notation of BP, secrets, and discriminative pairs. (cont.)

Symbol	Description
$(s,s') \in S \times S$	A pair of secrets, e.g. $(Example1, Example2)$
A discriminative pair of secrets (s,s')	A mutually exclusive pair of secrets. Two statements that cannot be true at the same time. An adversary must not be able to distinguish which one is true and which one is false, e.g. $(t.\_id =$ 'Bob' $\wedge t = x$, $t.\_id =$ 'Bob' $\wedge t = y)$
s_x^i	The secret $t.\_id = i \wedge t = x$ where $x \in \mathcal{T}$, e.g. $s^{\text{'Bob'}}_{(\text{'cancer'},65)}$
$\mathcal{S}_{\text{pairs}}$	A set of discriminative pairs of secrets, e.g. $\mathcal{S}^{\text{full}}_{\text{pairs}}$, $\mathcal{S}^{\text{attr}}_{\text{pairs}}$, $\mathcal{S}^{\mathcal{P}}_{\text{pairs}}$, $\mathcal{S}^{d,\theta}_{\text{pairs}}$
$\mathcal{S}^{G}_{\text{pairs}}$	A set of discriminative pairs of secrets based on graph $G(V,E)$, i.e. $\{(s_x^i, s_y^i) \mid \forall i, \forall (x,y) \in E\}$
Full domain: $\mathcal{S}^{\text{full}}_{\text{pairs}}$	For every individual, the value is not known to be x or y, i.e. $\{(s_x^i, s_y^i) \mid \forall i, \forall (x,y) \in \mathcal{T} \times \mathcal{T}\}$
Attributes: $\mathcal{S}^{\text{attr}}_{\text{pairs}}$	For every individual and every two tuples differing in the value of only one attribute A where one of them is real, the real tuple is not known. The privacy definition is weaker than in full domain $\mathcal{S}^{\text{full}}_{\text{pairs}}$ since the real tuple is distinguishable if more than one attribute differs, i.e. $\{(s_x^i, s_y^i) \mid \forall i, \exists A, x[A] \neq y[A] \wedge x[\bar{A}] = y[\bar{A}]\}$
Partitioned: $\mathcal{S}^{\mathcal{P}}_{\text{pairs}}$	For every individual and every two tuples coming from the same partition where one of them is real, the real tuple is not known, i.e. $\{(s_x^i, s_y^i) \mid \forall i, \exists j, (x,y) \in P_j \times P_j\}$. This privacy definition is very useful for location data.
Distance threshold: $\mathcal{S}^{d,\theta}_{\text{pairs}}$	For every individual and every two tuples having their distance less than or equal to a threshold θ where one of them is real, the real tuple is not known, i.e. $\{(s_x^i, s_y^i) \mid \forall i, d(x,y) \leq \theta\}$

In this direction, the idea of a discriminative secret is very similar to what consists a game in cryptography. We prefer to call it a privacy game here and represent it as shown in Figure 2. In this game, a challenger picks an Id (e.g. Bob) and a pair of discriminative secrets at random (e.g. "Bob has called Alice" or "Bob has not called Alice"). The pair is represented by two tuples, or an edge in the discriminative secret graph of the Id. The edge vertices identify the two tuples. The challenger sends the Id and the two tuples to the adversary (e.g. which one does belong to Bob?). The adversary has to guess which of the two tuples belongs to the id and responds with only 1 bit b. $b = 0$ is chosen for t_0 and $b = 1$ for t_1.

Our goal is to make the probability of the adversary guessing the assumed right tuple not significantly different than a coin flip.

An important remark about undirected communication graphs is that not all the graphs are feasible. If Bob has talked to Alice, it means that Alice has talked to Bob. The database matrix is symmetric. Another constraint is that $t_i[i]$ must be 0, and all other entries are either 0 or 1. The BP framework allows to define constraints about the dataset, and redefines the notion of neighborhood databases by excluding intermediate, yet infeasible ones. Therefore, we suggest that BP is a more suitable framework for communication graphs than its DP predecessor.

Table 2: Examples of notions of secrets for a communication database.

Symbol	Description - Example
Secret: s	Bob has talked to Alice: $t_i.\_id =$ 'Bob' $\wedge$ $t_j.\_id =$ 'Alice' $\wedge t_i[j] = t_j[i] = 1$
A discriminative pair of secrets (s,s')	Given two communication tuples (ego networks), we cannot distinguish which one of them belongs to Bob, for example, $(t.\_id =$ 'Bob' $\wedge t = (0,1,1,1)$, $t.\_id =$ 'Bob' $\wedge t = (0,0,0,1))$
s_x^i	The secret where individual i has ego network x, for example, $s^{\text{'Bob'}}_{(0,1,1,1)}$
$\mathcal{S}^{\text{full}}_{\text{pairs}}$	For an individual, all ego networks are discriminative
$\mathcal{S}^{\text{attr}}_{\text{pairs}}$	For an individual and two vectors that differ in only one communication, we cannot tell which one is real.
$\mathcal{S}^{\mathcal{P}}_{\text{pairs}}$	For an individual and two tuples belonging to the same partition, we cannot tell which tuple is the real one.
$\mathcal{S}^{(d,\theta)}_{\text{pairs}}$	Given a distance metric and a threshold. The privacy game is to challenge the adversary with one individual and two records having their distance less than or equal to threshold. A suitable distance for communication graphs is the Hamming distance (or the number of different bits), which is equivalent to the Manhattan distance in this case.

2.2 Auxiliary Knowledge

Auxiliary knowledge is usually formalized using correlations, for example $c(R = r_1) + c(R = r_2) = a_1$ where $c(r_1)$ is the count of records having the attribute R equal to r_1, $c(r_2)$ is the count of records having the attribute R equal to r_2, and a_1 is known. BP suggests to formalize auxiliary knowledge in terms of a set of constraints Q that a database D must satisfy. It denotes $I_Q \subset I_n$ the subset of all possible database instances. In the case of undirected communication graphs, we have two inherent constraints:

- the matrix of D is symmetric: $t^i_j = t^j_i$
- the ego attributes are zero: $t^i_i = 0$

It is also possible to use directed communication graphs where a directed edge from Bob to Alice means that Bob has called Alice. In this case the first constraint above is not considered. Additional constraints which are not necessarily inherent to the graph representation can be considered, for example:

- *Count Queries*. The number of individuals that have 5 neighbors.
- *Marginal Constraints*. A marginal is the projection of the database on a given subset of columns. Rows having the same projection are grouped in one record along with their count. In our context we project on a subset of nodes. For example let's project the database in Figure 1 on Bob and Eve only (columns 1 and 3). Alice and Carol have the same projection since both have called Bob and Eve. Therefore the projection have 3 rows: (Bob,1), (Eve,1) and (Alice-Carol,2).
- *Meta-node Constraints*. A meta-node is a node representing a sub-graph or a group of individuals. Meta-node auxiliary knowledge is for example the number of people calling a group of individuals, or the number of calls in between two groups of individuals. The adversary may know that the group Bob-Carol and the group Alice-Eve have three calls linking them.
- *Clique Constraints*. A clique is a complete graph. The adversary may know that a group of nodes makes a clique. For example Bob, Alice and Eve form a clique.

2.3 Policy and Privacy Definitions

To apply BP, one must define a policy $P(\mathcal{T}, G, I_Q)$ which is composed of a set of tuples $\mathcal{T}$, a discriminative secret graph $G(V,E)$ based on sets of discriminative pairs S_{pairs}, and a set of possible database instances I_Q under the auxiliary knowledge constraints. One also has to devise a randomized mechanism $\mathcal{M}$ that satisfies (ε, P)-BP. Concretely, for every pair of neighboring databases, denoted $(D_1, D_2) \in N(P)$, and every set of outputs $S \subseteq \text{range}(\mathcal{M})$, we have:

$$\Pr[\mathcal{M}(D_1) \in S] \leq e^{\varepsilon} \Pr[\mathcal{M}(D_2) \in S]$$

To see how it differs from DP, let's consider $D_1 = D \cup \{x\}$ and $D_2 = D \cup \{y\}$, two databases that differ in one tuple, and suppose $P = (\mathcal{T}, G, I_n)$, i.e., no constraints. D_1 and D_2 are not considered neighbors unless $(s_x^i, s_y^i) \in S^G_{\text{pairs}}$. Otherwise, having $\mathcal{M}$ that satisfies (ε, P)-BP means that:

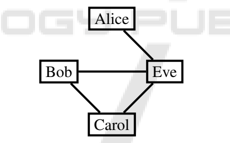
$$\Pr[\mathcal{M}(D_1) \in S] \leq e^{\varepsilon . d_G(x,y)} \Pr[\mathcal{M}(D_2) \in S]$$

since BP is shown to satisfy sequential composition. Similarly to increasing ε, the chance of an attacker to distinguish between pairs farther apart in the graph is higher. We gain overall utility by scarifying local privacy of some users.

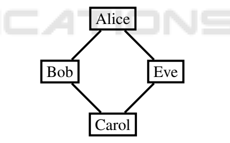
The Laplace mechanism $\mathcal{M}^{\text{Lap}}$ ensures $(\varepsilon, P(\mathcal{T}, G, I_Q))$-BP for any query function, $f : I_Q \rightarrow \mathbb{R}^d$, by outputting $f(D) + \eta$ where $\eta \in \mathbb{R}^d$ is a vector of independent random numbers drawn from $\text{Lap}(S(f,P)/\varepsilon)$. $S(f,P)$ is the policy-specific global sensitivity and is defined as $\max_{(D_1,D_2)\in N(P)} \|f(D_1) - f(D_2)\|_1$.

Following the definition of neighbors in He et al. (2014), let $T(D_1, D_2)$ the set of discriminative pairs (s_x^i, s_y^i) such as the ith tuples in D_1 and D_2 are x and y. Let $\Delta(D_1, D_2) = D_1 \backslash D_2 \cup D_2 \backslash D_1$. D_1 and D_2 are neighbors, if: (1) they both comply to the constraints, (2) $T \neq \emptyset$, and (3) T has the smallest size, there is no feasible database D_3 such that $T(D_1, D_3) \subset T(D_1, D_2)$ or $T(D_1, D_3) = T(D_1, D_2)$ & $\Delta(D_1, D_3) \subset \Delta(D_1, D_2)$. In our communication graph representation, two databases are candidate neighbors if they differ by the ego network of one individual, and this difference is represented in the security graph of that individual. Note that this means that one or several edges might be added or removed between two neighbor communication graphs.

To give an example, the two graphs in Figure 3 are different in three tuples: $|\Delta(D_1, D_2)| = 6$. If only one of the different pairs is in the security graph, for instance Bob's pairs, we have $|T| = 1$. There is no database having a non-empty subset of T, and no feasible database with same T and a subset of Δ. (To do so, we need to make Alice's ego network indifferent, or Eve's ego network indifferent, which is not possible due to symmetry constraints). We consider that these two graphs are neighbors.



id	tuple
Bob	(0, 0, 1, 1)
Alice	(0, 0, 1, 0)
Eve	(1, 1, 0, 1)
Carol	(1, 0, 1, 0)



id	tuple
Bob	(0, 1, 0, 1)
Alice	(1, 0, 1, 0)
Eve	(0, 1, 0, 1)
Carol	(1, 0, 1, 0)

Figure 3: Neighboring graphs and their databases. The ego network of Bob has changed, and Bob has a G^{full} policy.

Under $P(\mathcal{T}, G^{\text{full}}, I_Q)$, we can obtain two neighbor communication graphs by taking one vertex and changing its ego network. Any two communication graphs that differ in $n+1$ tuples where n tuples differ in one bit and one tuple differs in n bits are considered neighbors under G^{full}.

To make the concept of neighbor databases used throughout the paper more straightforward, we demonstrate the following result:

Theorem 1. *Given $G_c(V_c, E_c)$, its database/matrix representation $M(G_c)$ and the policy $P(\mathcal{T}, G, I_Q)$, where $\mathcal{T}$ represents all binary vectors of size $|V_c|$, G represents the* *overall graph of discriminative secret graphs for all the nodes, and I_Q constrains the possible databases to have: (1) $\forall i \neq j, M_{i,j} = M_{j,i} = 0$ or $M_{i,j} = M_{j,i} = 1$ and (2) $\forall i, M_{i,i} = 0$. If $G = G^{attr}$ or G is any non-empty subset of G^{attr}, we have that: Two graphs G_c^1 and G_c^2 are neighbors, i.e., $(G_c^1, G_c^2) \in N(P)$, if they differ by one and only one edge $e(i,j) = e(j,i)$ and for at least one vertex of the edge (either i or j) the discriminative secret pair (s_x^i, s_y^i) (where x and y differ at the bit j) or (s_a^j, s_b^j) (where a and b differ at the bit i) is in the security graph G.*

Proof. G^{attr} means that two tuples form a discriminative secret pair if they differ by only one attribute. This difference is reflected in the communication graph by the addition or removal of one edge.

If G_C^1 and G_C^2 differ by one or more edges that do not correspond to discriminative pairs in the security graph G, then $T(G_C^1, G_C^2) = \emptyset$ and the graphs are not neighbors.

If G_C^1 and G_C^2 differ by many edges that affect many secret pairs in G, then we can build a graph G_C^3 that takes only one of these edges that affects one (or two) secret pairs in G to form a subset of $T(G_C^1, G_C^2)$, and therefore the two graphs are not neighbors.

For the case where G_C^1 and G_C^2 differ by many edges and for only one of them $e(i,j)$ we have $(s_x^i, s_y^i) \in G$ or $(s_a^j, s_b^j) \in G$ or both, then $T(G_C^1, G_C^2)$ is the minimal possible set. But we can find a sub-graph G_C^3 of G_C^2 where $T(G_C^1, G_C^2) = T(G_C^1, G_C^3)$ by removing the extra edges which do not have any discriminative secret pairs that belong to the security graph. Then, G_C^1 and G_C^2 are not neighbors.

For the case where G_C^1 and G_C^2 differ by only one edge $e(i,j)$, and we have $(s_x^i, s_y^i) \in G$ or $(s_a^j, s_b^j) \in G$ or both, then $T(G_C^1, G_C^2)$ is minimal and there is no feasible intermediate database. Only in this case G_C^1 and G_C^2 are neighbors. $\square$

2.4 BP with Individualized Security Graphs

We consider the possibility that different individuals may have different security graphs. For example, we can divide the users into two extreme sub-groups: VIP and Standard. The discriminative secret graph for a VIP user is complete or attribute-based. The discriminative secret graph for a standard user has 0 edges.

The application of Theorem 1 to the case where standard nodes' security graph is G^{empty} and VIP nodes' security graph is G^{attr} can be explained as follows. Take two communication graphs that differ by only one edge:

- **Case I.** If the vertices of the edge are standard nodes, then $T(G_C^1, G_C^2) = \emptyset$ and the graphs are not neighbors.
- **Case II.** If one of the vertices is VIP and the other is standard, then the size of T is 1 and the graphs are neighbors.
- **Case III.** If the vertices of the edge are two VIP nodes, then the size of T is 2 and the graphs are neighbors. Any intermediate database that makes $|T| = 1$ is infeasible.

3 EXAMPLES OF QUERIES AND BP MECHANISMS

To apply BP given a query or a function f over the protected database D, one has to determine first the global sensitivity Dwork et al. (2006) of f, based on the privacy policy $P = (\mathcal{T}, G, I_Q)$:

$$S(f,P) = \max_{(D_1,D_2)\in N(P)} ||f(D_1) - f(D_2)||_1$$

Once the global sensitivity $S(f,P)$ is identified, outputting $f(D) + \eta$ ensures (ε, P)-BP if $\eta \in \mathbb{R}^d$ is a vector of independent random numbers drawn from $Lap(S(f,P)/\varepsilon)$.

3.1 Example 1: Histogram Query for Degrees of Vertices

Under P_{attr}, two graphs G_C^1 and G_C^2 are neighbors if $G_C^2 = G_C^1 \cup \{e\}$. Assume $DV = dv_1, ..., dv_{|DV|}$ is the set of all degrees for the vertices in these graphs.

If the edge e is added between node a having degree dv_i and node b with degree dv_j, $i \neq j$, then the count of dv_i and dv_j will decrease each by 1 while dv_{i+1} and dv_{j+1} will increase each by 1. Also the cumulative count of dv_i (respectively dv_j) decreases by 1, yet the cumulative count of dv_{i+1} (respectively dv_{j+1}) stays unchanged, as shown in Figure 4.

If the edge e is added between two nodes both having the same degree dv_i, then the count of dv_i will decrease by 2 and dv_{i+1} will increase by 2. Also the cumulative count of dv_i decreases by 2, yet the cumulative count of dv_{i+1} stays unchanged, as shown in Figure 5. Taking both cases into account, the global sensitivity of complete histogram query is $S(f_{\text{complete}}, P_{\text{attr}}) = 4$, and the global sensitivity of a cumulative histogram query is $S(f_{\text{cumulative}}, P_{\text{attr}}) = 2$.

Under P_{full}, two graphs G_C^1 and G_C^2 are neighbors if they differ by the ego network of one vertex. In the worst case, the vertex passes from degree 0 to degree $n-1$, where n is the number of vertices in the graph. All the other vertices have their degrees shifted by $+1$. In total, $2n$ bins are affected and the sensitivity is $2n$. In cumulative histogram query, in a worst case scenario, n vertices change their degrees and move from one bin to another, however the receptive bin does not change its count. The sensitivity is n.

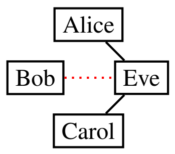
3.2 Example 2: Histogram of Degrees of Vertices for Standard Nodes

In this exercise, we divide the communication graph vertices into two groups: VIP nodes and standard nodes. The discriminative secret graph for a VIP node is built as follows: There is no edge between two tuples if they differ by more than one attribute (i.e. G^{attr}). In addition, we consider only attributes that belong to a VIP vertex. Two tuples differing by an attribute corresponding to a standard node are not connected in the discriminative secret graph. We denote this set of secret pairs: $\mathcal{S}_{\text{pairs}}^{\text{attr,VIP}}$.

Consider the following query: ”Histogram of degrees of vertices for standard nodes”. To compute their sensitivity we examine the three cases: (a) the edge we add/remove is between two VIP nodes: nothing will change in the histogram of the query; (b) the edge we add/remove is between one VIP node and one standard node: one of the bins in the histogram will decrease by 1 and its right-hand neighbor will increase by 1; (c) the edge we add/remove is between two standard nodes. This edge does not correspond to a secret pair. It means that this case will not occur for two neighbor graphs and can therefore be ignored.

It follows that the sensitivity of this query under $\mathcal{S}_{\text{pairs}}^{\text{attr,VIP}}$ is only 2. The sensitivity is reduced by 50% in comparison to the full histogram query. By limiting the privacy focus to the VIP nodes, we gain in terms of utility for queries over the standard nodes.

3.3 Example 3: Histogram of the Number of Connections between VIP Nodes and Standard Nodes



degree	0	1	2	3	0	1	2	3
count	1	2	1	0	0	3	0	1
cumulative count	1	3	4	4	0	3	3	4

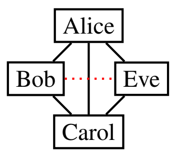
Figure 4: Counts and cumulative counts of node degree for graphs G_C^1 and G_C^2 (with dashed red edge), the added edge e connects two nodes of different degrees.

A similar query is the ”Histogram of the number of connections between a VIP node and standard nodes” or ”Histogram of the number of connections between a standard node and VIP nodes” . To compute their sensitivity we examine the three cases:

(a) the edge we add/remove is between two VIP nodes: nothing will change in the query’s result,

(b) the edge we add/remove is between one VIP node and one standard node: two of the bins in the histogram will vary by ± 1,

(c) the edge we add/remove is between two standard nodes. This edge does not correspond to a secret pair and does not change the query result in the same time.



degree	2	3	2	3
count	2	2	0	4
cumulative count	2	4	0	4

Figure 5: Counts and cumulative counts of node degree for graphs G_C^1 and G_C^2 (with dashed red edge), the added edge e connects two nodes of the same degree.

The sensitivity of these queries under $\mathcal{S}_{\text{pairs}}^{\text{attr,VIP}}$ is 2. These queries are useful in a graph where the standard nodes are the members of a company’s support team and the VIP nodes are the customers. The calls between a support team member and the customers are the target. We aim to study, for example, if a load balancing strategy works well, or how many clients a support member is serving in average. At the same time, we are protecting the privacy of the customers.

4 EXPERIMENTS AND RESULTS

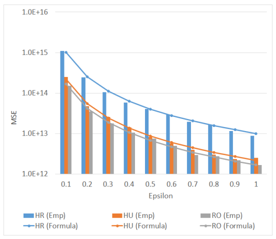
In this section, we present the results of our experiments to evaluate the BP on graphs both in terms of utility and privacy. We compare G^{attr} and G^{full} for histogram queries. In addition, we show the utility of some queries under $G^{\text{attr,VIP}}$. Our experiments use three graph datasets collected from the music streaming service Deezer Rozemberczki et al. (2019). These datasets represent the friendship Network of users from Croatia (HR) of 54,573 nodes and 498,202 edges, Hungary (HU) of 47,538 nodes and 222,887 edges and Romania (RO) of 41,773 nodes and 125,826 edges.

4.1 MSE of Complete Histogram

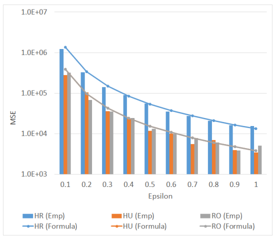
The Mean Squared Error (MSE) for complete histogram queries under G^{attr} and G^{full} are:

$$\mathcal{E}_{\mathcal{M}^{\text{Lap}}_{h,\mathcal{P}_{\text{attr}}}}(D) = b\mathbb{E}[\text{Laplace}(4/\varepsilon)^2] = 32b/\varepsilon^2$$
$$\mathcal{E}_{\mathcal{M}^{\text{Lap}}_{h,\mathcal{P}_{\text{full}}}}(D) = b\mathbb{E}[\text{Laplace}(2n/\varepsilon)^2] = 8n^2b/\varepsilon^2$$

where n is the number of vertices and b is the number of bins. We empirically sample the MSE of the complete histogram query for a given ε by generating the real histogram of each graph (HR, HU, and RO) and k noisy versions. We compute the MSE of the noisy



(a) Full Policy Complete Histogram.



(b) Attribute Policy Complete Histogram.

Figure 6: MSE of complete histograms queries under Full and Attribute Policies ($k = 10$).

versions as follows:

$$\text{MSE}(H, H_\varepsilon) = \sum_{i=1}^{b} \text{mean}_k[(H(i) - H_\varepsilon(i))^2]$$

where H is the original histogram and H_ε is its ε-noisy version. We choose ten epsilon values: 0.1, 0.2, ..., 1. The number of bins b is equal to 421, 113 and 113 for the graphs HR, HU and RO respectively. The results are shown in Figure 6a under G^{full} and in Figure 6b under G^{attr}. G^{full} has null utility whereas for G^{attr} we expect the standard deviation per bin to be around 32 to 56 (depending on ε). Using coarser bins may reduce this error by decreasing b.

4.2 MSE of Cumulative Histogram

MSE for cumulative histogram under G^{attr} and G^{full}:

$$\mathcal{E}_{\mathcal{M}^{\text{Lap}}_{h_{\mathcal{P}_{\text{attr}}}}}(D) = b\mathbb{E}(\text{Laplace}(2/\varepsilon))^2 = 8b/\varepsilon^2$$
$$\mathcal{E}_{\mathcal{M}^{\text{Lap}}_{h_{\mathcal{P}_{\text{full}}}}}(D) = b\mathbb{E}(\text{Laplace}(n/\varepsilon))^2 = 2n^2b/\varepsilon^2$$

The difference in MSE under G^{full} and G^{attr} is also clear for cumulative histogram queries as shown in Figure 7a and Figure 7b.

4.3 Simulating Sensitivity Results

In this experiment, we sample neighbors of our input graphs and compute the difference in the values of the histogram queries. We compare the obtained values with our derived sensitivity formulas. We sample the sensitivity values for our input graphs as follows:

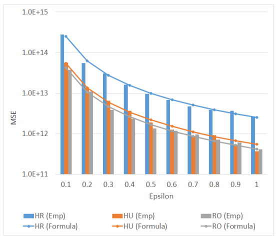
(a) take an input graph, and compute its histogram H,

(b) take a vertex at random,

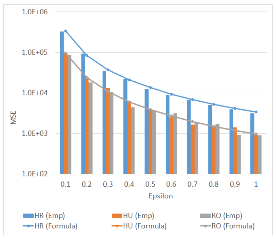
(c) randomly change the ego network of the vertex for G^{full}, change the value of only one edge for G^{attr},

(d) compute the histogram query for the obtained graph H', and compute the L1-norm $||H - H'||_1$,

(e) repeat starting at (c),



(a) Full Policy Cumulative Histogram.



(b) Attribute Policy Cumulative Histogram.

Figure 7: MSE of cumulative histograms queries under Full and Attribute Policies ($k = 10$).

(f) take max $||H - H'||_1$ and compare to the corresponding sensitivity: $2n$ for G^{full} and just 4 for G^{attr},

(g) repeat starting at (a),

Different strategies can be adopted to change the ego network under G^{full}:

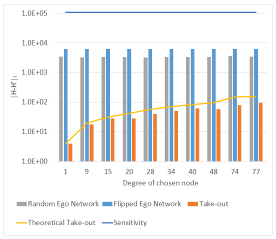
- Take-out: The chosen vertex is taken out by removing all its connections similarly to node-based DP.
- Random Ego Network: The chosen vertex samples a Bernoulli distribution with predefined probability p (e.g. 0.5) to decide on linking to each node in the graph.
- Flipped Ego Network: The chosen vertex deletes all its neighbors and connects to all non-neighbors.

We apply the above strategies to the complete histogram queries and vertices of different degrees. The results are shown in Figure 8a. In addition we show the derived sensitivity formula and the worst case take-out difference $(2 \times (\deg_v + 1))$. It is clear that G^{full} is unreasonably pessimistic about the sensitivity of complete histograms queries. We can gain more utility by deriving more specific secret graphs (based on the strategy of ego network manipulation) or adding constraints and auxiliary knowledge to the policy. For instance, it is unreasonable that a single node would be connected to all the nodes in the graph.

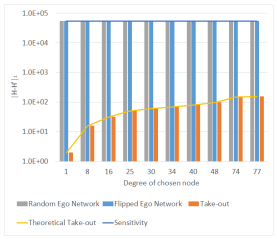
Similar experiments for the cumulative histogram queries are shown in 8b. In contrast, random and flipped ego network values are very close to the derived sensitivity formula.

4.4 Extrapolation of Queries under G^{attr}

We have shown that limiting privacy to some VIP nodes provides utility for queries such as ”Histogram of degrees of vertices for standard nodes” and ”Histogram of the number of connections between a VIP node and standard nodes”. In this experiment, we show that these queries can be exploited to estimate information about the complete graph. For instance, the histogram of degrees of vertices for stan-



(a) Complete Histogram.



(b) Cumulative Histogram.

Figure 8: Simulation of the queries under Full Policy.

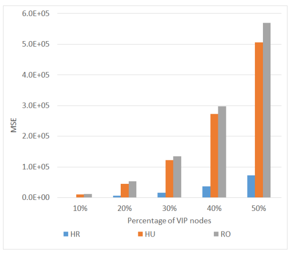


Figure 9: MSE of extrapolation for complete histogram queries under VIP/Standard partition.

dard nodes can be extrapolated to estimate the histogram of degrees of vertices for all the nodes. The histogram of the number of connections between a VIP node and standard nodes can be extrapolated to estimate the histogram of degrees of VIP nodes, and so on. The histogram of degrees of vertices for standard nodes can be extrapolated as follows:

$$H_{\text{all nodes}}(i) = H(i)_{\text{standard}} \times \frac{100}{100-\%_{VIP}}$$

The MSE of the extrapolated histogram query compared to the complete histogram query is shown in Figure 9. Naturally, the error depends on the ratio of the number of VIP nodes to the total number of vertices in the graph.

5 RELATED WORK

The authors of this paper are recently conducting research in secure outsourcing of computations Nassar et al. (2013, 2016) and privacy preserving applications Barakat et al. (2016); Chicha et al. (2018). To our knowledge, no previous work concerning BP towards graph datasets has been considered in the literature. However, many DP mechanisms were prospered for graphs. In Hay et al. (2009), Hay et al. divide these mechanisms into two types: edge-DP and node-DP.

Usually, a node in a graph represents a person while an edge represents a connection between two persons. The purpose of edge-DP Blocki et al. (2012) is to prevent the usage of these connections for revealing the identity of a person. On the other hand, node-DP achieve similar data protection by blurring node appearance in the graph. Node-DP is much more sensitivity than edge-DP, which is usually preferable.

Graph projection is a technique to apply node-DP. A parameter α is used to transform a graph to be α-degree-bounded. In Kasiviswanathan et al. (2013), all the nodes with a higher degree than α are removed, which causes a much higher number of edges to be removed than necessary.

In Raskhodnikova and Smith (2015), Raskhodnikova et al. use a Lipschitz extension tool and a generalized exponential mechanism to release an approximate histogram of degree distribution in a graph under node-DP with a sensitivity of 6α. Many works propose to generate noisy degree distributions from graphs under DP. Generative methods are then used to create output graphs fulfilling noisy input distributions Day et al. (2016). Qin et al. Qin et al. (2017) propose LDPGen for decentralized social networks. It collects neighbor lists of the nodes and reconstructs the graph in two phases under local edge-DP.

Local and smooth sensitivity achieves less noise than the global sensitivity Nissim et al. (2007). Finally, Karma et al. Karwa et al. (2011) presents efficient algorithms to provide noisy answers to sub graph counting queries under a relaxed version of edge-DP. Our work is a departure from previous works, first because it employs the framework of BP, and second because it allows a per-identity customized privacy policy for individuals in the graph.

6 CONCLUSION

In this paper, we have summarized BP, its formal model and the enhanced privacy-utility trade-off that it brings with respect to its predecessor, DP. We enrolled examples of its application to communication graph databases and their typical queries. We further studied the idea of privacy as a service with differentiation among different groups of individuals. We showed that this relaxation is formally feasible and proved its utility through the enrollment of several queries and computing their sensitivity. We work under the settings of binary differentiation (standard vs. VIP) and binary communication status (0 or 1).

ACKNOWLEDGMENTS

The authors would like to acknowledge the National Council for Scientific Research of Lebanon (CNRS-

L) and Univ. Pau & Pays Adour, UPPA - E2S, LIUPPA for granting a doctoral fellowship to Elie Chicha. This work is also partially suported by a grant from the University Research Board of the American University of Beirut (URB-AUB-2019/2020), the National Council for Scientific Research of Lebanon, and the Antonine University.

REFERENCES

S. Barakat, B. A. Bouna, M. Nassar, and C. Guyeux. On the evaluation of the privacy breach in disassociated set-valued datasets. *arXiv preprint arXiv:1611.08417*, 2016.

J. Blocki, A. Blum, A. Datta, and O. Sheffet. The johnson-lindenstrauss transform itself preserves differential privacy. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 410–419. IEEE, 2012.

E. Chicha, B. Al Bouna, M. Nassar, and R. Chbeir. Cloud-based differentially private image classification. *Wireless Networks*, pages 1–8, 2018.

W.-Y. Day, N. Li, and M. Lyu. Publishing graph degree distribution with node differential privacy. In *Proceedings of the 2016 International Conference on Management of Data*, pages 123–138, 2016.

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

M. Hay, C. Li, G. Miklau, and D. Jensen. Accurate estimation of the degree distribution of private networks. In *Data Mining, 2009. ICDM'09. Ninth IEEE International Conference on*, pages 169–178. IEEE, 2009.

X. He, A. Machanavajjhala, and B. Ding. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1447–1458. ACM, 2014.

V. Karwa, S. Raskhodnikova, A. Smith, and G. Yaroslavtsev. Private analysis of graph structure. *Proceedings of the VLDB Endowment*, 4(11):1146–1157, 2011.

S. P. Kasiviswanathan, K. Nissim, S. Raskhodnikova, and A. Smith. Analyzing graphs with node differential privacy. In *Theory of Cryptography Conference*, pages 457–476. Springer, 2013.

M. Nassar, A. Erradi, F. Sabri, and Q. M. Malluhi. Secure outsourcing of matrix operations as a service. In *2013 IEEE Sixth International Conference on Cloud Computing*, pages 918–925. IEEE, 2013.

M. Nassar, N. Wehbe, and B. Al Bouna. K-nn classification under homomorphic encryption: application on a labeled eigen faces dataset. In *2016 IEEE Intl Conference on Computational Science and Engineering (CSE) and IEEE Intl Conference on Embedded and Ubiquitous Computing (EUC) and 15th Intl Symposium on Distributed Computing and Applications for Business Engineering (DCABES)*, pages 546–552. IEEE, 2016.

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*, pages 75–84. ACM, 2007.

Z. Qin, T. Yu, Y. Yang, I. Khalil, X. Xiao, and K. Ren. Generating synthetic decentralized social graphs with local differential privacy. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 425–438, 2017.

S. Raskhodnikova and A. Smith. Efficient lipschitz extensions for high-dimensional graph statistics and node private degree distributions. *arXiv preprint arXiv:1504.07912*, 2015.

B. Rozemberczki, R. Davies, R. Sarkar, and C. Sutton. Gemsec: Graph embedding with self clustering. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2019*, pages 65–72. ACM, 2019.