Shortest Paths Publishing With Differential Privacy

Bin Cai, *Member, IEEE*, Weihong Sheng, Jiajun Chen, Chunqiang Hu, *Member, IEEE*, and Jiguo Yu, *Fellow, IEEE*

***Abstract*—The growing prevalence of graphs representations in our society has led to a corresponding rise in the publishing of graphs by researchers and organizations. To protect the privacy, it is important to ensure that graphs including sensitive data are not disclosed. Since the weight of edges could be utilized to infer confidential information, the graph should be privately published to avoid ethical and legal issues. In this paper, we propose a novel method for privately publishing shortest paths while preserving the privacy of sensitive edge weights in graph. Specifically, we divide the edge weights into internal and external edges based on their edge betweenness centrality. Then, we give two different differentially private algorithms to perturb edge weights based on the distinction between internal and external edges, respectively. To reduce the error ratios between differentially private shortest paths and real shortest paths, we employ edge betweenness centrality to search for the shortest path, which is closest to the true one. Our experimental results show that our mechanisms can effectively reduce the error in the average shortest path distance by 1.1% for large graphs, while for the shortest path change rate, our mechanisms can reduce it by 8.3%.**

***Index Terms*—Differential privacy, graph theory, randomized response, shortest path.**

I. Introduction

THE graph is an abstract structure that provides a visual representation of entities, enabling a more comprehensive comprehension of their relationships. It is composed of vertices that represent entities (e.g., individuals or organizations), and edges that represent directed or undirected associations between vertices. In addition to the structural relationships between vertices, the graph also contains attributes that describe vertices and weights that describe edges. The graph structure can be used to represent a variety of data, including social graphs, knowledge graphs, road graphs, and more. Mining information from graphs is an extensive area of research, including path scheduling, community detection, pattern matching, and others. While this potent tool facilitates a deeper understanding of intricate relationships and supports informed decision-making, it inevitably raises concerns regarding privacy breaches.

With the growing awareness of privacy concerns, privacy-preserving is getting increasingly critical. The privacy-preserving can refer to user identity protection or untraceability [1], [2], [3], [4], [5]. However, in the context of both graph data analysis and graph data publishing, the privacy-preserving generally refers to preventing sensitive information from leaking since attributes, weights, and even structural information can be sensitive and may raise ethical or legal issues if unintentionally exposed. To safeguard the privacy of individuals within a social graph, early studies employed techniques such as de-identification [6] or using k-anonymity [7]. Nevertheless, these traditional anonymization methods are based on the assumption that potential adversaries possess limited background knowledge and are unable to effectively counter increasingly powerful attacks [8]. Given the vulnerabilities of traditional anonymization techniques that hinder data analysis and publication, researchers are actively pursuing research and development efforts to identify effective ways to protect privacy and mitigate the risk of privacy breaches.

Differential privacy is a popular privacy-preserving standard that has won the favor of researchers because of its strict mathematical definition and robust privacy measure. Its core concept revolves around minimizing the impact of input variations on output outcomes by introducing randomness. Differential privacy has many desirable properties, including: (1) maximum background knowledge assumption: it assumes that the attacker knows all the records except the target one; (2) privacy mechanism synthesis: simple differential privacy mechanisms can be combined into complex differential privacy mechanisms; and (3) privacy loss quantification: a specific quantification of the privacy-preserving capability of the mechanism. With these advantages, differential privacy has found applications in major tech companies such as Microsoft [9], Google [10], and Apple [11] for the collection of telemetry data while safeguarding individual privacy.

Differential privacy in the context of graphs can be categorized into two primary domains: node differential privacy and edge differential privacy [12]. Node differential privacy is concerned with reducing the impact on the output of modifying (adding

Manuscript received 25 May 2023; revised 30 September 2023; accepted 27 October 2023. Date of publication 3 November 2023; date of current version 3 April 2024. This work was supported in part by National Natural Science Foundation of China under Grants 62372075, 62072065, and 62272256, in part by Sichuan Science and Technology Program under Grant 2023YFQ0029, in part by Regional Innovation Cooperation Project of Sichuan Province under Grant 2023YFQ0028, in part by Major Program of Shandong Provincial Natural Science Foundation for the Fundamental Research under Grant ZR2022ZD03, in part by the Pilot Project for Integrated Innovation of Science, Education and Industry of Qilu University of Technology (Shandong Academy of Sciences) under Grant 2022XD001, in part by the Key Project of Technology Innovation and Application Development of Chongqing under Grant CSTC2019jscx-mbdxX0064, and in part by Graduate Scientific Research and Innovation Foundation of Chongqing under Grant CYB23054. Recommended for acceptance by M. Obaidat. *(Corresponding author: Chunqiang Hu.)*

Bin Cai, Weihong Sheng, Jiajun Chen, and Chunqiang Hu are with the School of Big Data and Software Engineering, Chongqing University, Chongqing 400044, China (e-mail: caibin@cqu.edu.cn; im.swh@outlook.com; jiajunchen@cqu.edu.cn; chu@cqu.edu.cn).

Jiguo Yu is with Big Data Institute, Qilu University of Technology, Jinan, Shandong 250353, China (e-mail: jiguoyu@sina.com).

Digital Object Identifier 10.1109/TSUSC.2023.3329995

or removing) a node and its associated edges. In contrast, edge differential privacy aims to reduce the impact on the output when certain edges are altered. There are numerous implementations of edge differential privacy, such as estimating the cost of a minimum spanning tree [13], counting subgraphs [14], publishing algebraic connectivity [15]. And in node differential privacy, there are estimating degree distribution [16], discovering frequent graph patterns [17], and fitting a high-dimensional statistical model to a sparse network [18].

The shortest path is commonly employed in graph analysis, with applications ranging from path scheduling in traffic graphs to friend matching and betweenness centrality calculation in social graphs. For clarity, we use the term 'shortest path' to refer to the sequence of vertices traversed by the shortest path and 'distance' to denote the sum of weights associated with the edges traversed by the shortest path. Based on edge differential privacy, Sealfon [19] first proposed the weight private graph model that the structure is public but weights are private which was applied to a single shortest path and all-pairs distances privately release. Intuitively, the weight private graph model is a sensible assumption that aligns well with real-world scenarios, like road graphs where the landmarks are visible to all visitors, but traffic conditions reported by vehicles might be sensitive due to the location information contained. Another example is social graphs, where the weights may represent the degree of trust between individuals, which should preserve privacy, but it is difficult to conceal friendships. Therefore, an adversary with either internal or external access to the graph, such as someone who can only access a published graph, can easily obtain the graph topology. Subsequent research on shortest path and distance publishing based on differential privacy has largely adopted this model [20], [21], [22].

Motivation: While the author in [19] has proposed several algorithms for privately releasing all-pairs shortest path distances and shortest paths between pairs of vertices, accompanied by rigorous theoretical analyses, certain research gaps remain. (1) The proposed algorithms belong to central differential privacy that needs a trusted center to respond to shortest path and distance queries, which lacks the desired flexibility. (2) The assumption of adjacent objects, namely neighboring weight functions which differ by at most 1 may not be enough to fully protect the privacy of the weights. For example, if the range of weights is a finite set of natural numbers, the range of candidate weights guessed by the adversary can be very small. (3) This work primarily focuses on how to reduce the approximate distance errors and ignores the change of shortest paths. For instance, the released approximate shortest path may be slightly or completely different from the true shortest path. In practical scenarios such as navigation, obtaining the approximate shortest path may not suffice; knowledge of the true shortest path may be essential.

Problem definition: Our objective is to develop an algorithm capable of privately publishing shortest paths while preserving the privacy of edge weights and minimizing the alteration of these paths. Following the weight-private graph model, we depart from the approach taken in [19] and opt to release a synthetic graph that accommodates arbitrary shortest path queries. In pursuit of robust privacy protection, we consider the worst-case scenario: adjacent graphs differ in an edge weight by at most $b - a$, where a and b are the minimum and maximum weights within the graph, respectively. However, this setting of adjacent graphs can introduce significant distortions to the true shortest paths, resulting from the injection of large-scale noises into the weights. Consequently, our challenge lies in enhancing the utility of the shortest paths while avoiding potential privacy breaches.

In this paper, we focus on accurately publishing shortest paths while preserving the privacy of edge weights. As previously mentioned, we seek to achieve this through the use of differential privacy, as traditional privacy-preserving techniques have proven less effective as deterrents. Based on edge differential privacy, we divide edges into two distinct groups: no shortest paths passed through and some shortest paths passed through. To mitigate errors, we apply different perturbation techniques to each of these groups. Furthermore, in our quest to further reduce errors, we explore the application of edge betweenness centrality as a means to filter out undesirable shortest paths, ultimately selecting optimal approximate shortest paths.

The contributions are listed as follows.

- We give a method for dividing edges based edge betweenness centrality and prove that the core edge does not exist for all-pairs shortest paths.
- We design two differentially private algorithms to protect weights in graphs which have lower errors in shortest path and distances.
- We propose an novel algorithm based on cumulative multiplication of edge between centrality to improve the utility of the outputs of differentially private algorithms.

The remainder of this paper is organized as follows. Section II provides an overview of related work, both with and without the incorporation of differential privacy. Section III introduces the theoretical foundations relevant to the algorithms developed in this study. Section IV details the specific algorithm we have designed. Section V presents the experimental results and analysis of our algorithm. Section VI discusses the limitations and potential threats to our study. Finally, Section VII offers a concise summary of this paper and outlines potential avenues for future research.

II. Related Work

In the scenario of privately publishing shortest paths, there are three basic method types: (1) traditional anonymity techniques based methods, such as using k-anonymity or noise to hide sensitive information; (2) cryptography based methods that using homomorphic encryption or others; and (3) differential privacy based methods.

Based on traditional anonymity techniques. Liu et al. [23] provided a greedy perturbation algorithm to reduce the negative effect of distorted weights on the shortest paths. But it has a huge computational overhead since the algorithm needs to travel all the edges and has to compute the all-pairs shortest paths for each edge traversal. Another related work is that Wang et al. [24] proposed using k-anonymity to ensure that there are at least k-indistinguishable shortest paths from source to target but it

just focuses on the privacy of shortest path. Wang et al. [25] also combined node degree and k-anonymity to improve their discovery of at least k-indistinguishable shortest paths between the source and target nodes with the same degree. The solution prevents the attacker from having background knowledge about the degree of nodes.

Based on cryptography. Some researchers focused on cryptography based shortest paths computing or query. Wu et al. [26] proposed an efficient privacy-preserving shortest paths computing method that focuses on sparse graph in real-time navigation by compressing the next-hop routing matrices in road map. Ramezanian et al. [27] extended Floyd Warshall algorithm to output a all-pairs shortest paths related matrix, which can be used to query shortest paths by private information retrieval (PIR) techniques. However, this solution applies homomorphic encryption based PIR, which results in heavy overheads in computation and query. The same issue exists in [28].

Based on differential privacy. There are currently only a few works on differential privacy-preserving the shortest path publishing. Sealfon [19] first formally proposed the implementation of how to privately publish all-pairs shortest path distances, demonstrating an additional error of approximate distance at most $O(n \log n/\varepsilon)$. Sealfon also left an open question: Whether can privately release all-pairs distances with error sublinear in n. Therefore, [20], [21], [22], [29] proposed their solutions based on the open question. Ghazi et al. [20] published all-pairs shortest path distances with upper error $\tilde{O}(n^{2/3}/\epsilon)$ and lower error $\Omega(n^{1/6})$ in pure differential privacy that need to bridge the gap between the two errors. Fan et al. [29] proposed two methods on the tree and grid graph with error $O(\log^{1.5} n)$ and $\tilde{O}(n^{3/4})$, respectively. And to give a solution for general graphs, Fan et al. [22] proposed a method that can answer all-pairs shortest path distances with error $\tilde{O}(n^{1/2})$ via constructing a synthetic graph in approximate differential privacy. Chen et al. [21] improved the error to about $n^{(\sqrt{17}-3)/2+o(1)}$ in pure differential privacy if weights are bounded.

However, the above differential privacy based studies almost always only concentrate on the distance and overlook the change in the shortest path. Our motivation drives us to study the change of shortest paths with large weights perturbations, which is the first work to study the change of shortest paths in differential privacy.

III. Preliminaries

We consider a simple, undricted, and bounded-weight graph $G = (V, E, W)$, where V is a set of vertices(i.e., landmarks in a traffic graph, users in a social graph, etc), E is a set of edges and W is the set of weights associated with edges. Let $n \in \mathbb{N}$ be the total number of vertices in V, then $V = \{v_1, \ldots, v_n\}$. For $i, j \leq n$, $e_{i,j}$ represents the relationship of v_i and v_j, namely, $e_{i,j} \in E$, $w_{i,j} \in W$, respectively. We may as well assume that the weights are integers and are constrained to $[a, b]$ where a and b are the minimum and maximum weights in the graph, respectively. Note that here both $a, b > 0$. For graphs with negative weights, we can simply add a constant to all weights to make them satisfy the above condition.

A. Differential Privacy

The formal mathematical definition of differential privacy was initially introduced by Dwork [30]. The main idea is to limit the impact of changes in the input data on the output results that mask the contribution of any individual in the database. Driven by our motivations, we provide a variant version of the original differential privacy model.

Definition III.1: For any two graphs G and G', where $G = (V, E, W)$ and $G' = (V, E, W')$, we say they are adjacent if there exists only at most one weight difference:

$$\|W - W'\|_1 = \sum_{e_{i,j} \in E} \left|w_{i,j} - w'_{i,j}\right| = \left|w_{i,j} - w'_{i,j}\right| \leq k \quad (1)$$

where $k = b - a$. We denote $G \sim G'$.

The concept of this 'adjacent' relationship bears similarities to the notion found in edge differential privacy, as both concentrate on the alteration of individual edges. However, in the context of the graph model, our primary concern lies in the finer-grained details of edge weights, rather than the mere structural presence of a specific edge.

Definition III.2: Assume there is a randomized algorithm that $\mathcal{M} : \mathcal{G} \to \mathcal{O}$, where $\mathcal{G}$ is the universe of our bounded-weight graphs. Consider any two graphs $G, G' \in \mathcal{G}$, $\mathcal{M}$ satisfies ϵ-differential privacy if $G \sim G'$, and for all outputs $O \in \mathcal{O}$, have:

$$\Pr[\mathcal{M}(G) \in O] \leq \exp(\varepsilon) \Pr\left[\mathcal{M}\left(G'\right) \in O\right] \quad (2)$$

The above definition is symmetric, allowing for the interchange of the positions of G and G'. This formula describes the similarity between the probability distributions of output O under e^{ϵ}. Here, ϵ is referred to as the privacy budget. A small ϵ provides a robust privacy guarantee, which diminishes as ϵ increases. Extremely, with $\epsilon = 0$, maximum randomness is enforced, effectively reducing the impact of the $G \sim G'$ approach to zero and resulting in the destruction of data utility.

Theorem III.1: Suppose $\mathcal{M}_i$ satisfies ϵ_i-differential privacy. Let domain G be divided into n disjoint subsets G_i where $i \in [n]$. Then the set $\{\mathcal{M}_1(G_1), \ldots, \mathcal{M}_n(G_n)\}$ provides $\max_{i \in [n]}\{\epsilon_i\}$-differential privacy. This is a parallel composition property of differential privacy [31].

Definition III.3: Let $f : \mathcal{G} \to \mathcal{Z}^m$ be any query function. The classic differential privacy mechanism called Laplace mechanism can be defined as:

$$\mathcal{M}(G) = f(G) + (Y_1, \ldots, Y_m) \quad (3)$$

where the Y_i is an independent Laplace random variable drawn from the Laplace distribution with mean 0 and scale parameters $b = \Delta f/\epsilon$, which is defined as:

$$Lap(x|b) = \frac{1}{2b} exp\left(-\frac{|x|}{b}\right) \quad (4)$$

And sensitivity Δf:

$$\Delta f = \max_{G \sim G'} \|f(G) - f\left(G'\right)\|_1 \quad (5)$$

B. Randomized Response

Randomized response was initially introduced by Warner et al. [32] to achieve plausible deniability when answering binary alphabets. Assume a research is being conducted involving the collection of sensitive attributes $x \in \{0, 1\}$. Volunteers are eager to participate but hesitant to reveal their personal information, as indicated by the symbol x_i. Consequently, volunteers flip a biased coin, honestly answer x_i with probability p, and deny with probability $1 - p$. The aforementioned randomized response mechanism is straightforward and optimal across all privacy levels; however, it is limited to binary attributes. Kairouz et al. in [33] extended this mechanism to a more general form suitable for handling attributes with multiple values.

Definition III.4: Suppose the attribute univese $\mathcal{X}$ is discrete and bounded. For any $x_i \in \mathcal{X}$, the response $y_i \in \mathcal{X}$ satisfies:

$$y_i = \begin{cases} x_i & \text{w.p.} \frac{e^{\varepsilon}}{|\mathcal{X}|-1+e^{\varepsilon}} \\ x_i' & \text{w.p.} \frac{|\mathcal{X}|-1}{|\mathcal{X}|-1+e^{\varepsilon}} \end{cases}. \tag{6}$$

where x_i' is a discrete uniform random variable from $\mathcal{X}\backslash\{x_i\}$ so that y_i has the probability of responsing any attribute in $\mathcal{X}$. The sum of probabilities is 1.

This definition implies that the response y_i has a higher probability of representing the true sensitive attribute x_i when $\epsilon > 0$, while having an equal probability of representing any other attribute in the set $\mathcal{X}\backslash\{x_i\}$. To eliminate any potential ambiguity, it is important to emphasize that the subsequent discussion of randomized response pertains to the multi-alphabet version.

C. Betweenness Centrality

In graph theory, the betweenness centrality is an important measure of centrality, which is given by the shortest paths. The number of shortest paths that passed through the vertex divided by the total number of shortest paths yields the node betweenness centrality. It is the degree of interaction between the current node and others. Accordingly, edge betweenness centrality has a similar definition to the above one, but the numerator is the number of shortest paths passed through the edge.

Definition III.5: The edge betweenness centrality of edge $e_{i,j}$ is defined as follow:

$$C_B(e_{i,j}) = \sum_{\substack{s,t \in V \\ s \neq t}} \frac{\sigma_{s,t|e_{i,j}}}{\sigma_{s,t}} \tag{7}$$

where $\sigma_{s,t}$ is the number of shortest paths from source s to target t, and $\sigma_{s,t|e_{i,j}}$ is the number of shortest paths from s to t and passed through edge $e_{i,j}$.

IV. Our Mechanism

In this section, we will introduce the mechanisms for privately publishing the graph while minimizing errors in approximating shortest paths and distances.

A. Publish Graph

Our goal is to perturb all the weights in the graph and make it satisfy ϵ-differential privacy. Intuitively, a solution is to add Laplace noise, which is calibrated by Δf. Liu et al. [23] employs a similar strategy with Gaussian noise. However, using noise directly will obviously change a significant portion of the shortest paths from source to target.

Lemma IV.1: Let $a = 0$, and f be a query function that returns the shortest paths between two vertices, the Laplace mechanism will distort shortest paths with approximate probability at least $1 - (1 - e^{-\frac{\epsilon}{b}})^l$ where l is the shortest path length.

For small ϵ and large l, there is a high probability that the shortest path deviates from the original state.

Inspired by the edge partition approach outlined in [23], edges can be categorized into three distinct groups based on their interaction with shortest paths: those through which all shortest paths pass, those through which no shortest paths pass, and those through which some but not all shortest paths pass. Intuitively, when we increase the weights of edges through which no shortest paths pass, the shortest path and the associated distance remain unchanged.

Consequently, we can perturb edges separately rather than introducing noise on a uniform scale. For the sake of simplicity, in the context of publishing all-pairs shortest paths, we will reclassify the edge types and provide a proof demonstrating that no edge is traversed by all shortest paths.

Definition IV.1: Given an edge $e_{i,j}$, we call it the core edge if all the shortest paths $p_{u,v}$ pass through it for any source and target vertex pair (u, v) in G.

Definition IV.2: If there is no shortest path $p_{u,v}$ that passes through an edge $e_{i,j}$ for any source and target vertex pair (u, v) in G, we call it an external edge.

Definition IV.3: If there is at least one shortest path $p_{u,v}$ that passes through an edge $e_{i,j}$ for any source and target vertex pair (u, v) in G, we call it an internal edge.

Theorem IV.2: In a connected graph with degree no less than 3, there is no core edge for all-pairs shortest paths.

Proof: Suppose G is a connected and undirected graph and has a core edge $e_{s,t}$ between s and t. Let u be a neighboring vertex of s different from t. Then the shortest path $p_{u,s}$ must be: $p_1(u \to \ldots \to t) + e_{t,s}$ and distance $d_{u,s}$ is equal to $d_{u,t} + w_{s,t}$, we have:

$$\begin{aligned} d_{u,s} &= d_{u,t} + w_{t,s} \\ &= w_{t,s} + \sum_{e_{i,j} \in p_1} w_{i,j} < w_{u,s} \end{aligned} \tag{8}$$

Because $d_{u,t} < w_{u,s} + w_{s,t}$, if there is no other paths from u to t, p_1 must be the shortest path $p_{u,t}$. But in order to pass through core edge $e_{s,t}$, there must exist other paths which contain $p_{u,t}$. Therefore, there must be a path from u to s, denoted as p_2 where $p_{u,t} = p_2(u \to \ldots \to s) + e_{s,t}$. Let v be a vertex which is in path p_2 and is a neighboring vertex of s. Then p_2 is: $u \to \ldots \to v \to s$ and we have:

$$d'_{u,t} = w_{s,t} + \sum_{e_{i,j} \in p_2} w_{i,j} \tag{9}$$

Algorithm 1: Publish Graph With Positive and Negative Noise.

Input: $G = (V, E, W), \epsilon_1, \epsilon_2, \Delta f$
Output: $\hat{G}$
1: calculate edge betweenness centrality $\mathbf{B}$
2: **for** $b_{i,j}$ in $\mathbf{B}$ **do**
3: **if** $(b_{i,j} = 0)$ **then**
4: $\hat{w}_{i,j} = w_{i,j} + Lap^{+}(\frac{\Delta f}{\epsilon_1})$
5: **else**
6: $\hat{w}_{i,j} = w_{i,j} + Lap^{-}(\frac{\Delta f}{\epsilon_2})$
7: **end if**
8: $w_{i,j} \leftarrow \hat{w}_{i,j}$
9: **end for**
10: **return** $\hat{G}, \mathbf{B}$

Similarly, for shortest path $p_{v,s}$, there must be a path p_3 where $p_{v,s} = p_3(v \to \ldots \to t) + e_{t,s}$, and we have:

$$d_{v,t} < w_{v,s} + w_{s,t} \tag{10}$$

However, this will cause $p_2 + e_{s,t}$: $u \to \ldots \to v \to s \to t$ is not the shortest path from u to t, and $p_{u,s}$ should be: $p_2 - e_{v,s} + p_3$, namely: $u \to \ldots \to v \to \ldots \to t$ which not passes $e_{s,t}$. This is a paradox. Therefore the theorem holds. ■

The external edges are irrelevant to all shortest paths. To meet the ϵ-differential privacy requirement, we would like to add positive noise to reduce the distortion of shortest paths. If the weights of internal edges are reduced, this perturbation can ensure that external edges cannot be transformed into internal edges.

Intuitively, unlike adding Laplace noises to all edges in edge differential privacy, we would like to add positive Laplace noises to external edges and negative ones to internal edges, respectively.

With the above definitions and theorems, we now have an intuitive way to publish a whole graph G. Algorithm 1 describes our first solution.

The four input parameters, namely the target graph G, two different privacy budgets ϵ_1, ϵ_2 and sensitivity Δf. The First step is to calculate edge betweenness centrality $\mathbf{B}$. For every edge, we add positive Laplace noise which is drawn from Laplace distribution to its weight by filtering the negative noise if there is no short path passed through(namely, $b_{i,j}$ is zero). Otherwise, we add negative Laplace noise. After processing all edges in G, we privately publish $\hat{G}$.

Lemma IV.3: Algorithm 1 satisfies $\max\{\epsilon_1, \epsilon_2\}$-differential privacy.

Proof: Let $f : G \to O$ where $O \in \mathcal{Z}^m$ be an arbitrary weight-dependent query function on a graph G with output that can be decomposed into a sequence of weights, namely $O = \{o_1, o_2, \ldots, o_m\}$. Since the shortest paths only pass through the internal edges, we only need to consider the weights with negative noise if f is a shortest path query function.

Genenally, we can divide O into two parts: $O_1 = \{o_1, o_2, \ldots, o_l\}$ if corresponding edgs are internal edges and $O_2 = \{o_{l+1}, o_{l+2}, \ldots, o_m\}$ if corresponding edges are external

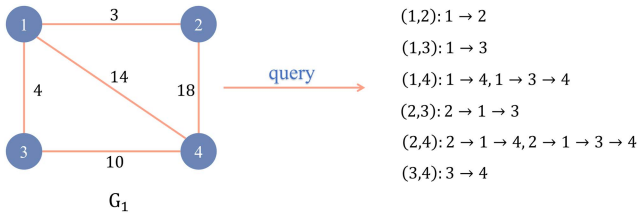

Fig. 1. An example of querying all-pairs shortest paths.

edges, respectively. For internal edges, we have:

$$\begin{aligned}
\frac{Pr_G(O_1)}{Pr_{G'}(O_1)} &= \frac{\prod_{i=1}^{l} \exp\left(\frac{\varepsilon_1(o_i - f(G)_i)}{\Delta f}\right)}{\prod_{i=1}^{l} \exp\left(\frac{\varepsilon_1(o_i - f(G')_i)}{\Delta f}\right)} \\
&= \prod_{i=1}^{l} \exp\left(\frac{\varepsilon_1\left((o_i - f(G)_i) - ((o_i - G')_i)\right)}{\Delta f}\right) \\
&= \prod_{i=1}^{l} \exp\left(\frac{\varepsilon_1\left(f(G')_i - f(G)_i\right)}{\Delta f}\right) \\
&\le \left(\frac{\varepsilon_1 \|f(G') - f(G)\|_1}{\Delta f}\right) \\
&\le \exp(\varepsilon_1)
\end{aligned} \tag{11}$$

For external edges, we have:

$$\begin{aligned}
\frac{Pr_G(O_2)}{Pr_{G'}(O_2)} &= \frac{\prod_{i=l+1}^{m} \exp\left(-\frac{\varepsilon_2(o_i - f(G)_i)}{\Delta f}\right)}{\prod_{i=l+1}^{m} \exp\left(-\frac{\varepsilon_2(o_i - f(G')_i)}{\Delta f}\right)} \\
&= \prod_{i=l+1}^{m} \exp\left(\frac{\varepsilon_2\left((o_i - f(G')_i) - (o_i - f(G)_i)\right)}{\Delta f}\right) \\
&= \prod_{i=l+1}^{m} \exp\left(\frac{\varepsilon_2\left(f(G)_i - f(G')_i\right)}{\Delta f}\right) \\
&\le \exp(\varepsilon_2)
\end{aligned} \tag{12}$$

Because O_1 and O_2 are outputs of two mutually disjoint datasets, Algorithm 1 satisfies $\max\{\epsilon_1, \epsilon_2\}$-differential privacy based on the parallel composition. ■

With the assurance of differential privacy, we can confidently publish the graph for shortest path queries while maintaining the privacy of edge weights. While knowledgeable adversaries may discern the edge types using $\mathbf{B}$ and potentially deduce the added noise type for each edge, our privacy-preserving measures remain effective, primarily due to our well-defined concept of adjacent graphs.

B. Comparison

In this subsection, we will use a concrete distance to demonstrate our first solution. As shown in Fig. 1, the left part of G_1 is a connected, undirected, and weighted graph, and the right part is query results. Note that for vertex pairs (1,4) and (2,4), they both have the two shortest paths that have the same distance.

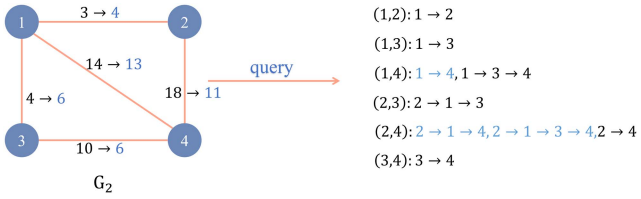

Fig. 2. Adding Laplace noise.

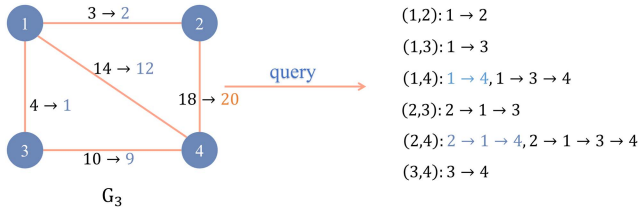

Fig. 3. Adding positive and negative Laplace noise.

Fig. 2 is the result of adding standard Laplace noises to the graph. For the sake of simplicity, we assume here that all noise values are integers (just a hypothesis). These noises are random, thus making some weights larger and some weights smaller. The shortest paths of G_2 have changed in Fig. 2, the vertex pair (1,4) has lost one shortest path $1 \rightarrow 4$, and the vertex pair (2,4) has lost two shortest paths $\{2 \rightarrow 1 \rightarrow 4, 2 \rightarrow 1 \rightarrow 3 \rightarrow 4\}$, but added a new shortest path $\{2 \rightarrow 4\}$. In Fig. 3, however, we add positive and negative Laplace noises.

As shown in Fig. 3, it is not hard to find that the edges $\{e_{1,2}, e_{1,4}, e_{1,3}, e_{3,4}\}$ are internal edges, but $e_{2,4}$ is an external edge. Therefore, only the weight of $e_{2,4}$ is increased. In this case, the shortest paths of vertex pairs (1,4) and (2,4) are still changed. (1,4) lost the shortest path as well $\{1 \rightarrow 4\}$, while (2,4) only lost $\{2 \rightarrow 1 \rightarrow 4\}$.

The shortest paths change rate is $3/8$ when compared to G_1 (we only calculate how many shortest paths there are in G_1 but not in G_2). And the shortest paths change rate of G_3 is $2/8$, which is lower than G_2. This example shows the core idea of our solution, which is to perturb the weights without affecting external edges.

C. Post-Processing

An important property of differential privacy is that its post-processing does not reveal any privacy information about the original dataset. And sometimes, proper post-processing helps to improve the utility.

In our case, if all weights in graph G are discrete and bounded, the weight $\hat{w}_{i,j}$ may deviate from the original domain after adding noise. For example, if the weights domain is [1,2,...,21], the noise-added weights may be in $[-11.1, \ldots, 32.11]$. Note that although Laplace noises are continuous, we will constrain them to a fixed precision, which is what researchers do by default in practice.

Back to our solution, there should be a post-processing step to round the noise weights to the origin domain. Otherwise, the fractional part will reveal the noise itself and give the adversary more confidence to infer the hidden weights, while the utility of the data will be reduced.

Algorithm 2: Post-Processing.

Input: $\hat{G}$, weight range$[a, b]$
Output: $\hat{G}'$
1: **for** $\hat{w}_{i,j}$ in $\hat{G}$ **do**
2: **if** $\hat{w}_{i,j} > b$ **then**
3: $\hat{w}'_{i,j} = b$
4: **else if** $\hat{w}_{i,j} < a$ **then**
5: $\hat{w}'_{i,j} = a$
6: **else**
7: $\hat{w}'_{i,j} = rr(\hat{w}_{i,j})$
8: **end if**
9: $\hat{w}'_{i,j} \leftarrow \hat{w}_{i,j}$
10: **end for**
11: **return** $\hat{G}'$

For out-of-bounds values, we reassign them to the nearest boundary. And for values within the bounds, we do not use normal rounding, but use random rounding. As shown in Algorithm 2, rr is a random rounding function that rounding our weights $\hat{w}_{i,j}$ to integer form. Random rounding means rounding up a value with the probability of its fractional part. For example, for a value 6.6, with the probability 0.6 and 0.4, $rr(6.6) = 7$ and $rr(6.6) = 6$, respectively.

Lemma IV.4: Random rounding will not introduce a additional error.

Proof: $X_i = a_i + b_i$ is a random variable, a_i and b_i are the integer and fractional part, respectively. Let $\hat{X}_i$ denotes the results of X_i after random rounding, and we have:

$$\hat{X}_i = \begin{cases} a_i + 1 & \text{w.p. } b_i \\ a_i & \text{w.p. } 1 - b_i \end{cases} \tag{13}$$

$$\mathbb{E}\left[\hat{X}_i\right] = (a_i + 1) \times b_i + a_i \times (1 - b_i) = a_i + b_i \tag{14}$$

Because the expectation of random rounding is unbiased, there is no additional error. ■

Therefore, to avoid introduce addtional error, we prefer to use random rounding as the post-processing if the graph is discrete but adding continuous noise.

D. Specific Solution

For the graph with bounded and discrete weights, the continuous noise destroys its discrete property.

In practice, it is better to avoid introducing operations that may introduce additional errors when designing a privacy-preserving mechanism.

Therefore, we apply the random response mechanism to graphs with both bounded and discrete weights. In most scenarios where this mechanism is applied, there is no central server that aggregates data. The above is called local differential privacy, which differs from our centralized model. In fact, random responses are equally applicable to our centralized model and can avoid additional errors.

Algorithm 3: Publish Graph With Randomized Response.

Input: $G = (V, E, W), \epsilon_1, \epsilon_2, w \in [a, b]$
Output: $\hat{G}$
calculate edge betweenness centrality $\mathbf{B}$
for $b_{i,j}$ in $\mathbf{B}$ **do**
  if $(b_{i,j} = 0)$ **then**
    $\hat{w}_{i,j} = w_{i,j} + Lap^{+}(\frac{\Delta f}{\epsilon_1})$
  else
    let α be a continuous uniform random variable in range $(a, b + e^{\epsilon_2}]$
    if $(\alpha \in (a + i, a + i + 1]$ where $i \in [0, 1, \ldots, b - a - 1])$ **then**
      $\hat{w}_{i,j} = a + i$
    else
      $\hat{w}_{i,j} = w_{i,j}$
    end if
  end if
end for
return $\hat{G}$

Algorithm 4: Correct Shortest Path.

Input: $\hat{G}', \mathbf{B}, t$, vertex pair (u, v)
Output: $\hat{\mathcal{P}}'_{u,v}$
$\hat{\mathcal{P}}_{u,v} \leftarrow$ shortest paths of (u, v) in $\hat{G}'$
$\hat{\mathcal{P}}^t_{u,v} \leftarrow$ top t simple shortest paths of (u, v) in $\hat{G}'$
for $\hat{p}$ in $\hat{\mathcal{P}}^t_{u,v}$ **do**
  $\beta = \prod_{e_{i,j} \in \hat{p}} b_{i,j}$
end for
$\hat{\mathcal{P}}^t_{u,v} \leftarrow \hat{\mathcal{P}}^t_{u,v}$ in descending order of β
$\hat{\mathcal{P}}'_{u,v} \leftarrow$ first $|\hat{\mathcal{P}}_{u,v}|$ paths in $\hat{\mathcal{P}}^t_{u,v}$
return $\hat{\mathcal{P}}'_{u,v}$

The main difference between Algorithms 1 and 3 is that the latter replaces adding negative noises with a randomized response mechanism. There is a random choice involved in whether weight $w_{i,j}$ remains constant with the probability $\frac{e^{\epsilon_2}}{b-a+e^{\epsilon_2}}$ or is assigned to any of the elements in $[a, b]$ with the probability $\frac{1}{b-a+e^{\epsilon_2}}$.

We use a continuous uniform random variable α to demonstrate how to implement random choice. For $i \in [0, 1, \ldots, b - a - 1]$, $\alpha \in (a + i, a + i + 1)$ with probability $\frac{1}{b-a+e^{\epsilon_2}}$ and $\alpha > b$ with probability $\frac{e^{\epsilon_2}}{b-a+e^{\epsilon_2}}$.

Lemma IV.5: Algorithm 3 satisfies $\max\{\epsilon_1, \epsilon_2\}$-differential privacy.

Proof: Similarly, the query function f on edges with $b_{i,j} = 0$ satisfy ϵ_1-differential privacy.

For query function f' on edges with $b_{i,j} > 0$, let w and w' be records in G and G', respectively. G and G' are differ in the ith record. Let $O \in \mathbb{R}^m$ be any output, i.e., $O = \{o_1, o_2, \ldots, o_m\}$, and we have:

$$\begin{aligned}
\frac{P(O \mid G)}{P(O \mid G')} &= \frac{\prod_i^m P(o_i \mid w_i)}{\prod_i^m P(o_i \mid w'_i)} \\
&= \frac{P(o_1 \mid w_1) \ldots P(o_i \mid w_i) \ldots P(o_n \mid w_n)}{P(o_1 \mid w'_1) \ldots P(o_i \mid w'_i) \ldots P(o_n \mid w'_n)} \\
&= \frac{P(o_i \mid w_i)}{P(o_i \mid w_i')} \\
&\leq \max\left(\frac{P(o_i \mid w_i)}{P(o_i \mid w_i')}\right) \\
&= \frac{\frac{e^{\epsilon_2}}{b-a+e^{\epsilon_2}}}{\frac{1}{b-a+e^{\epsilon_2}}} \\
&= e^{\epsilon_2} \qquad (15)
\end{aligned}$$

Similarly, because f and f' are applied to disjoint datasets, Algorithm 3 satisfies $\max\{\epsilon_1, \epsilon_2\}$-differential privacy based on the sequential composition. ■

As a result of the preceding Algorithm 3, we can conceal the true weights $w_{i,j}$ while always restricting them to the discrete domain without rounding.

E. Find Paths

In practical applications, we are very concerned about whether the approximate shortest path which is from differentially private graph $\hat{G}$ will change or not. For example, in vehicle navigation, even if we perturb the weights between related landmarks, the user can still achieve correct navigation based on the correct approximate shortest path with the lowest cost. Reducing the error between the approximate shortest path and the true shortest path can effectively improve the utility of the perturbed data. The work described in [23] used greedy thoughts to constantly measure the impact on the shortest paths after perturbing the weights. But it will compute the shortest paths multiple times, which causes a great overhead. To avoid wasting resources, we consider reusing the intermediate variables of our proposed algorithms.

In this section, we give a method to assist in finding the correct shortest paths. With the Δf up to $b - a$, the output $\hat{G}'$ of Algorithms 1 and 3 suffers such a great disturbance that it is hard to find the true shortest paths. As we have stated before, we prefer to protect every weight but publish the relationship (topological structure). Therefore, we develop the Algorithm 4.

There are four input parameters in Algorithm 4, the privacy-preserving graph $\hat{G}'$, edge betweenness centrality $\mathbf{B}$, path numbers t, and source and target vertex pair (u, v). Fortunately, because $\mathbf{B}$ is an intermediate result of Algorithm 1 and 3, there is no additional overhead to obtain it. The output is then $\hat{p}_{u,v}$, which is an approximation of the shortest path from source u to target v.

The first step is to obtain the shortest paths from u to v, denoted by $\hat{\mathcal{P}}_{u,v}$. The second is to identify the top t shortest paths of (u, v), denoted by $\hat{\mathcal{P}}^t_{u,v}$. In this paper, we focus on the simple shortest path and ignore the variations of loopless t shortest paths. The third step is to traverse all the candidate shortest paths (abbreviated as candidates) in $\hat{\mathcal{P}}^t_{u,v}$ and calculates associated

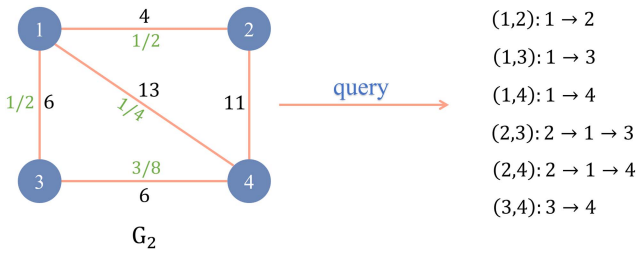

Fig. 4. Corrected shortest paths of G_2.

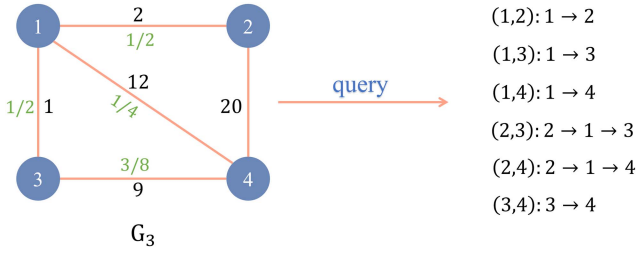

Fig. 5. Corrected shortest paths of G_3.

confidence, β. Each candidate has a confidence level, which is determined by multiplying the edge betweenness centrality of each edge along the path by a cumulative factor. This algorithm computes all the β and returns the more closer approximate shortest path collection, $\hat{\mathcal{P}}'_{u,v}$, which contains the candidates $\hat{p}$ with the top $|\hat{\mathcal{P}}_{u,v}|$ highest confidence.

As we described in Section III, edge betweenness centrality is measured by the number of shortest paths passed through. Assume $e_{i,j}$, the edge of vertex i and j, is a component of shortest path $p_{u,v}$. Intuitively, its betweenness centrality $b_{i,j}$ reflects the priority of shortest paths passed through $e_{i,j}$. The shortest paths prefer $e_{i,j}$ in the local substructure around $e_{i,j}$ with high $b_{i,j}$ rather than the edge candidates with low betweenness centrality.

We reuse previous examples as shown in Figs. 1, 2, and 3. Figs. 4 and 5 demonstrate the corrected shortest paths of G_2 and G_3, respectively. The green fractions represent the edge betweenness and centrality for each edge. The results of Algorithm 4 are shown on the right.

We suppose $t = 2$. First, let us focus on vertex pairs (1,4) and (2,4). The maximum perturbation has been applied to G_2, and its shortest paths have changed from $\{1 \to 4, 1 \to 3 \to 4\}$ to $\{1 \to 3 \to 4\}$. According to Algorithm 4, the β of the first shortest path $\{1 \to 3 \to 4\}$ is $3/16$, but the β of the second shortest path $\{1 \to 4\}$ is $1/4$. As a result, the shortest path of the vertex pair (1,4) is $\{1 \to 4\}$. Similarly, the first shortest path of (2,4) is $\{2 \to 4\}$, but the edge $e_{2,4}$ is an external edge that the edge betweenness centrality $b_{i,j}$ is 0. $\{2 \to 1 \to 3 \to 4\}$ with $\beta = 3/32$ is the second-shortest path of $(2, 4)$. As a result, the shortest path of a vertex pair of (2,4) is $\{2 \to 1 \to 3 \to 4\}$. At the same time, the shortest paths between other vertex pairs remain unchanged. The shortest path for (1,4) changes from $\{1 \to 3 \to 4\}$ to $\{1 \to 4\}$ using the same process. And the shortest path for (2,4) is $\{2 \to 1 \to 4\}$.

All the shortest paths in Fig. 4 are real shortest paths, and the change rate has been reduced to $1/4$. We can note that the main contribution of β is to avoid $1 \to 4$ to be the shortest path. But in G_3, it seems to make no sense because positive and negative Laplace noises will not be involved in forming the shortest paths. Our experiments will show that Algorithm 4 makes sense for sampling of Laplace noises.

TABLE I
PARAMETER SETTING

Parameter		Value
$Laplace$	t	$\lvert\hat{\mathcal{P}}_{u,v}\rvert + 2$
	ϵ_1	$10, 15, 20, 25, 30, 35, 40, 45, 50, 55$
	ϵ_2	$10, 15, 20, 25, 30, 35, 40, 45, 50, 55$
$Lap^+ + Lap^-$	t	$\lvert\hat{\mathcal{P}}_{u,v}\rvert + 2$
	ϵ_1	$10, 15, 20, 25, 30, 35, 40, 45, 50, 55$
	ϵ_2	$10, 15, 20, 25, 30, 35, 40, 45, 50, 55$
$Lap^+ + Lap$	t	$\lvert\hat{\mathcal{P}}_{u,v}\rvert + 2$
	ϵ_1	$10, 15, 20, 25, 30, 35, 40, 45, 50, 55$
	ϵ_2	$10, 15, 20, 25, 30, 35, 40, 45, 50, 55$
$Lap^+ + RR$	t	$\lvert\hat{\mathcal{P}}_{u,v}\rvert + 2$
	ϵ_1	$1, 2, 3, 4, 5, 6, 7, 8, 9, 10$
	ϵ_2	$1, 2, 3, 4, 5, 6, 7, 8, 9, 10$

TABLE II
DATASETS STATISTICS

Datasets	n	m	$weight$
EIES	48	830	[1,4]
Alpha	3783	24186	[-10,10]
OTC	5881	35592	[-10,10]

V. EVALUATION

In this section, we evaluate the effectiveness of our proposed algorithms using three real-world datasets. We use *average shortest path distance* (ASPD) and change rates to measure data utility and use ϵ to measure privacy. In order to demonstrate the effectiveness of our algorithms, we include two additional experimental settings. The first setting involves adding Laplace noise to the global edge weights, while the second setting involves using Laplace noise to replace the negative noise values.

A. Datasets

There are three real-world datasets: EIES, Alpha and OTC. Their relevant statistics are described in Table II. n is the number of nodes, m is the number of edges, $weight$ is the range of edge weight.

EIES [34] is an EIES network from Freeman at time 2. This is an acquaintance network, where vertices denote researchers and edges denote their relationships, which is a directed relationship. The weights on the edges indicate the degree of intimacy. To conform to our pre-defined privacy model, we convert the directed edges to undirected edges. That is, if there are two edges between two nodes, we delete these two edges and add an undirected edge, and the new weight is the average of the two weights in the deleted edges; otherwise, simply change the directed edges to undirected edges and keep the weights the same.

Alpha [35] is a directed, weighted trust network. The nodes are people who trade Bitcoin on the Bitcoin Alpha platform. Edges are the trust scores between nodes which are also directed. The original edge domain is from -10 (total distrust) to $+10$ (total trust). Similarly, we follow the same method to convert it to an undirected graph. Since it has negative weights, to simplify the shortest path calculation, we use 11 to minus the weight values of every edge so that its weight belongs to [1,21].

OTC [35] is also a directed weighted trust network, which is similar to Alpha, but from another platform: Bitcoin OTC. Therefore, the pre-processing is consistent with Alpha.

B. Parameters Setting

In this paper, we evaluate the utility of graphs by ASPD and shortest path change rate. We have already mentioned the change rate in our example. So we briefly introduce the ASPD.

Definition V.1: ASPD refers to the average shortest path distance of $G = (V, E, W)$, it is defined as follows

$$\xi = \frac{1}{n^2 - n} \sum_{\substack{0<i\le|V| \\ i<j\le|V|}} d_{i,j} \tag{16}$$

where $n = |V|$ is the number of nodes in G, and $d_{i,j}$ is the distance between node i and j.

As shown in Table I, $|\hat{\mathcal{P}}_{u,v}|$ refers to the total number of candidate shortest paths between u and v. The $Laplace$ means adding Laplace noises to all weights with the same ϵ values. Our Algorithm 1 and Algorithm 4 are $Lap^+ + Lap^-$ and $Lap^+ + RR$, respectively. The remaining $Lap^+ + Lap$ refers to adding Laplace noises to edges with non-zero edge betweenness centrality and adding positive Laplace noises to edges with zero edge betweenness centrality.

Because of their high sensitivity, the parameters in Table I are only appropriate for datasets Alpha and OTC. For EIES, ϵ_1 and ϵ_2 are all set to $\{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$.

To reduce the effect of randomness, for EIES, we implement the above four settings on the entire graph, then repeat 20 times to obtain the mean value. For Alpha and OTC, because the all-pairs shortest-path calculation has such a large overhead, we randomly select 200 nodes and calculate their shortest paths on the entire graph. Similarly, to reduce the effect of randomness, we repeat these operations 10 times. Note that since ϵ_1 and ϵ_2 take the same value in the experiment, we will not distinguish between ϵ_1 and ϵ_2 later.

C. Utility

Fig. 6 presents the relative ASPD errors for three datasets across four experimental settings: $Laplace$, $Lap^+ + Lap^-$, $Lap^+ + Lap$, and $Lap^+ + RR$. Due to significant differences in weight ranges and dataset sizes between the EIES dataset and the other two, we calculate the absolute values of the relative ASPD errors for each dataset to ensure a fair comparison. Each row of plots shows the results for a single dataset under the four experimental settings, while each column displays the results for all three datasets under the same experimental configuration. For simplicity, the pre-ASPD refers to the ASPD error before applying Algorithm 4, whereas post-ASPD represents the ASPD error after implementing Algorithm 4. To provide a concise summary of the results, Table III presents the average improvement between the pre-ASPD and post-ASPD values for each plot depicted in Fig. 6.

TABLE III
AVERAGE IMPROVEMENT OF ASPD

dataset	EIES	Alpha	OTC
$Laplace$	-0.014	0.007	0.006
$Lap^+ + Lap^-$	-0.033	0.005	0.005
$Lap^+ + Lap$	-0.025	0.007	0.006
$Lap^+ + RR$	0.020	0.011	0.011

For ASPD relative errors, it can be observed that the overall trend of all plots is decreasing with increasing ϵ because the large $epsilon$ means applying a larger magnitude of perturbation to the dataset. Except for the plots of EIES, other results all illustrate that the post-ASPD is lower than the pre-ASPD, which indicates that our Algorithm 4 helps reduce the error of ASPD. As shown in Table III, the improvement is up to 1.1% in large graph, Alpha and OTC.

Although the improvement is small, it is important to note that the main purpose of Algorithm 4 is not to reduce the ASPD error, but to find more approximate shortest paths. However, the anomalous results of EIES in Table III, i.e., $-1.4\%, -3.3\%, -2.5\%$, reveal Algorithm 4 is not applicable to small graph though there is a great improvement which is up to 10% when $\epsilon < 4$.

It is worth noting that the overall relative ASPD errors for OTC and Alpha are generally higher than those for EIES. For instance, when the lines in most plots stabilize, the pre-ASPD errors for Alpha and OTC are approximately around 53%, whereas the errors for EIES are less than 38%. This difference can be attributed to the longer average shortest path lengths in Alpha and OTC, which makes them more susceptible to the influence of noise on distances, as depicted in Fig. 7.

Fig. 7 presents the average shortest path lengths for the three datasets at different ϵ values ($\epsilon = 1, 5, 10$). The average shortest path length for each dataset remains relatively stable, approximately at 2.7, 6.2, and 7 for EIES, Alpha, and OTC, respectively. Consequently, Alpha and OTC exhibit higher errors due to their longer average shortest path lengths.

The $Laplace$ experiment serves as a comparison to $Lap^+ + Lap^-$. Notably, $Laplace$ exhibits lower relative ASPD errors compared to $Lap^+ + Lap^-$, primarily because the positive and negative noises in $Laplace$ tend to cancel each other out, while this balancing effect does not occur in $Lap^+ + Lap^-$. Similarly, $Lap^+ + Lap$ is a comparable work of $Lap^+ + RR$, and they demonstrate comparable errors. However, it is worth noting that the latter is more influenced by Algorithm 4, leading to improvements of up to 2% for EIES and 1.1% for other datasets, as shown in Table III. One possible explanation for this discrepancy could be the unbounded nature of Laplace noise, resulting in greater variance in the weights compared to the original values.

Next, we examine the change rate of shortest paths under the same settings using Fig. 8 and Table IV. The pre-change

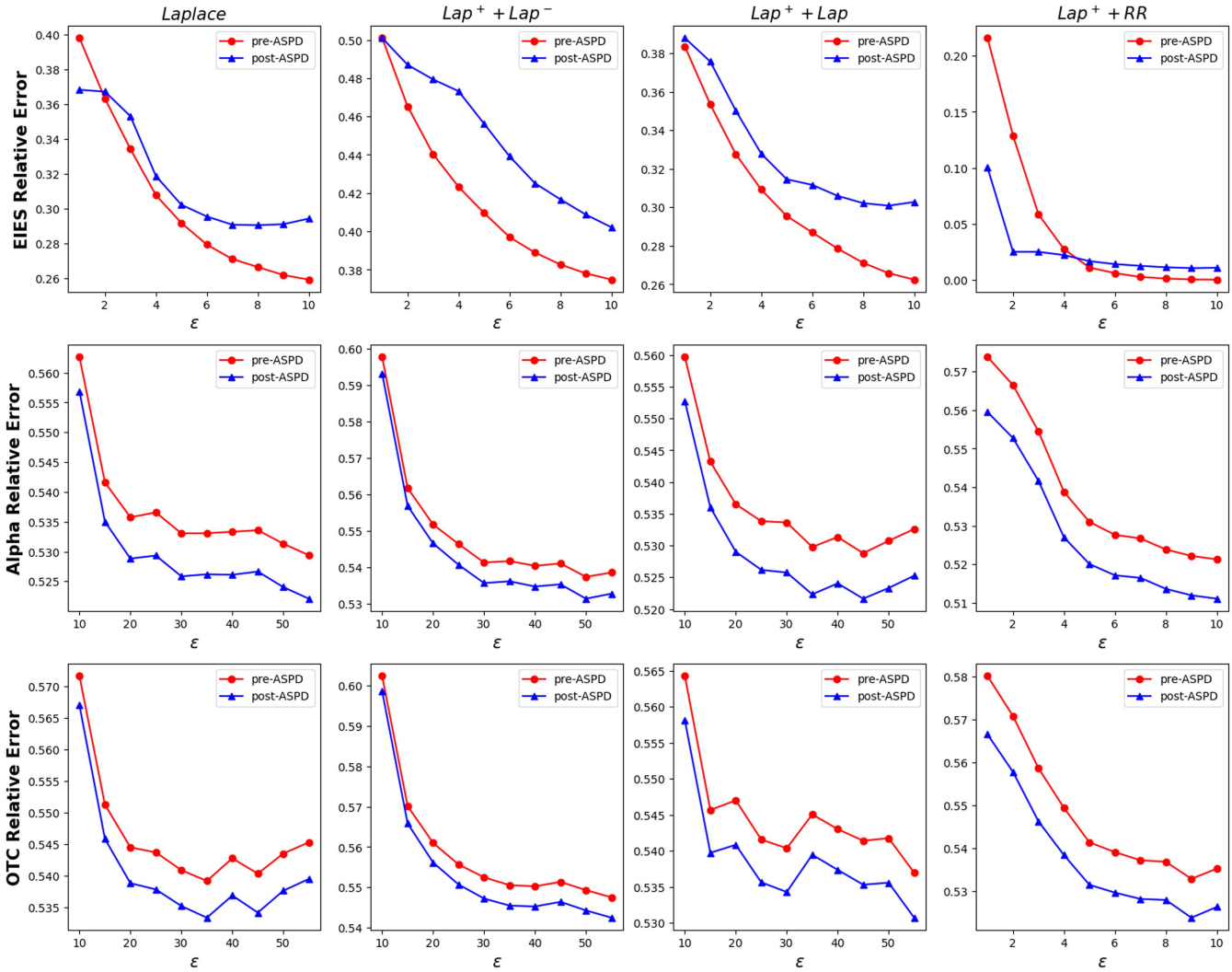

Fig. 6. Relative ASPD errors.

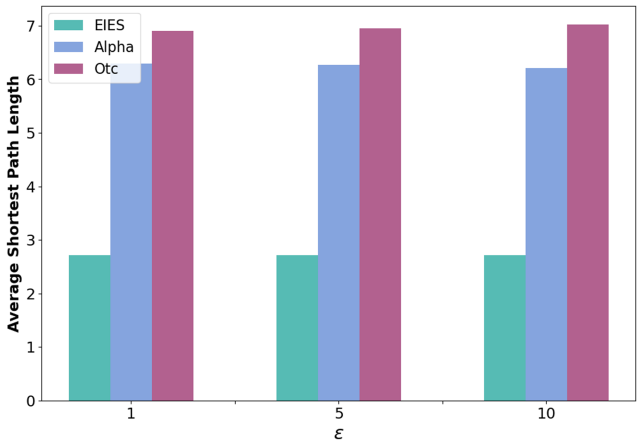

Fig. 7. Average shortest path length of selected part.

rate refers to shortest paths change rate before using Algorithm 4, while post-change rate refers to the change rate after using Algorithm 4.

TABLE IV
AVERAGE IMPROVEMENT OF CHANGE RATE

dataset	EIES	Alpha	OTC
$Laplace$	0.089	0.025	0.016
$Lap^+ + Lap^-$	0.015	0.026	0.028
$Lap^+ + Lap$	0.083	0.026	0.016
$Lap^+ + RR$	0.011	0.050	0.037

For small graph EIES, it is obvious that the pre-change rates of $Lap^+ + Lap^-$ and $Lap^+ + RR$ are less than 18% and 10% when ϵ are 10 and 5, respectively. Meanwhile, the pre-change rate of other two cases are large than 20% in the same setting which shows $Lap^+ + Lap^-$ and $Lap^+ + RR$ help to reduce the change rate for small graph. And for Alpha and OTC, the pre-change rates of $Lap^+ + Lap^-$ are less than 55% and 63%, which are obviously lower than other two cases, respectively. Furthermore, the post-change rates of the approximate shortest paths corrected by Algorithm 4 have all improved which is even

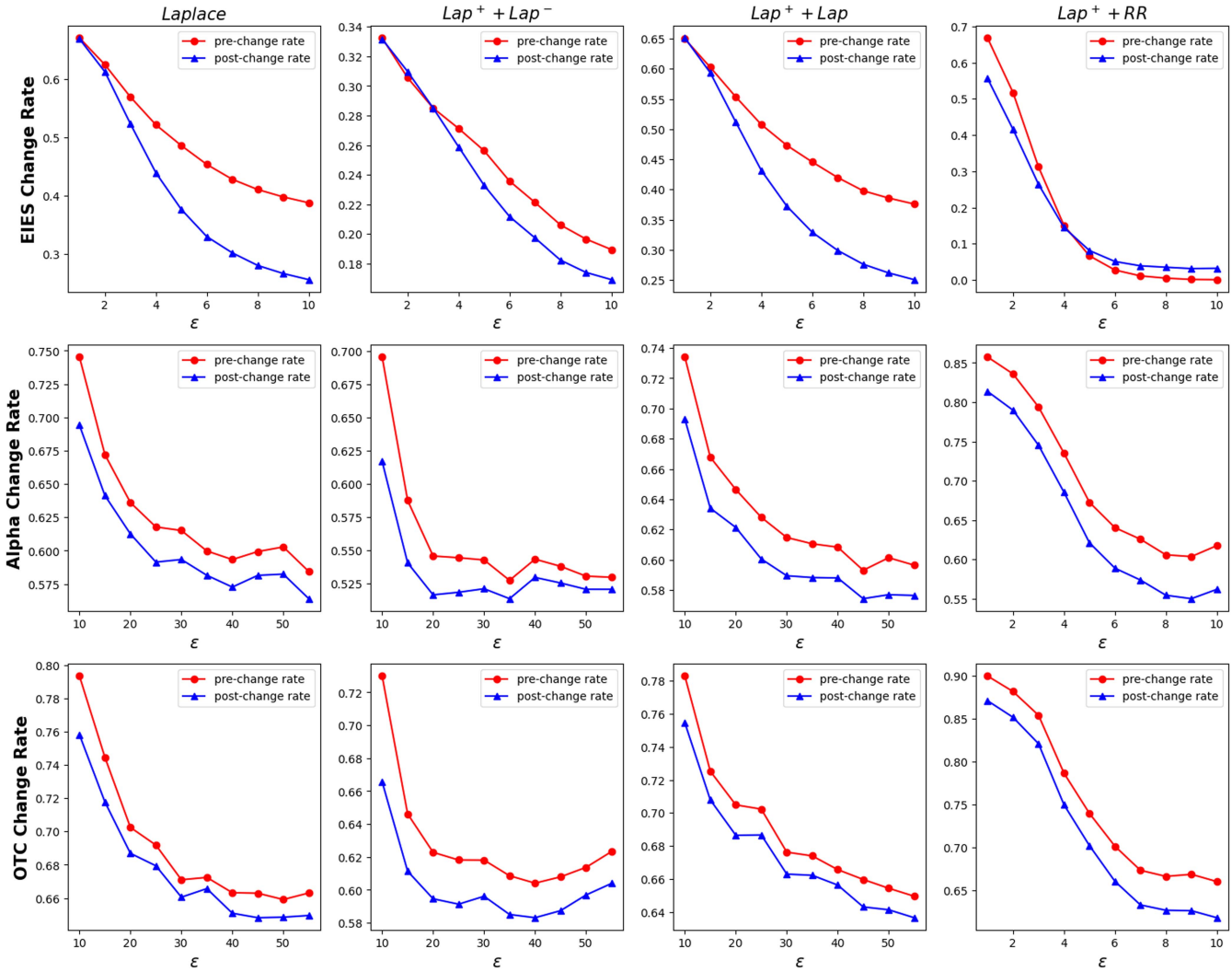

Fig. 8. Shortest paths change rate.

up to 8.3% in Table IV. Thus, Algorithm 4 helps find more approximate shortest paths for both large and small graphs.

When comparing $Lap^+ + Lap^-$ to $Laplace$, it is evident that $Lap^+ + Lap^-$ exhibits a lower change rate across all three datasets at the same ϵ setting. As presented in Table IV, the improvement offered by $Lap^+ + Lap^-$ is more pronounced in the large graph, Alpha, and OTC datasets, but not in the case of EIES. One apparent reason for this observation is that $Lap^+ + Lap^-$ already achieves a low change rate in the small EIES graph. This phenomenon is similar for both $Lap^+ + Lap^-$ and $Lap^+ + RR$, where the latter experiences an even lower change rate in the small EIES graph, resulting in suboptimal performance for Algorithm 4.

VI. Discussion

In this section, we will discuss the limitation and potential threats to internal, external, statistical, and construct validity.

A. Limitation

While our proposed mechanisms effectively preserve privacy, they do introduce changes in at least half of the shortest paths for large graphs, resulting in a significant impact on utility. For instance, as illustrated in Fig. 8, even when the pre-change rate and post-change rate tend to stabilize, their values remain above 50%. Such relatively high values may limit the applicability of our algorithms in scenarios where higher precision is required.

This limitation arises from the larger scale of noise introduced due to our stringent privacy requirements. It is crucial to recognize the inherent trade-off between privacy and utility. While reducing our privacy requirements and fine-tuning the difference between adjacent graphs to 1 can significantly decrease the magnitude of noise and enhance utility, it would ultimately compromise our primary objective of enhancing the level of privacy-preserving.

B. Threat

The potential threats to the internal validity could be the setting of t and ϵ. t directly determines the search scope of Algorithm 4 that means too small t will cause the true shortest path out of scope while large t will introduce more irrelevant paths. Intuitively, we choose to set t to $|\hat{\mathcal{P}}_{u,v}| + 2$ which will search for up to two more shortest paths in our experiments. This

is a conservative setting that there should be more thoughtful consideration to improve experimental results. And ϵ has a direct impact on privacy and errors. Therefore, we defined different ϵ depending on the sensitivity of the input graph and perturbation method. For graph EIES with sensitivity 3 and perturbation method RR, we set ϵ to a small range [1,10]. And for other two graphs and other three perturbation methods, we set ϵ to a large range [10,55] where the increase step is 5. Finally, ϵ_1 and ϵ_2 are applied to different parts of the graphs, and they are set to the same values for clearly facilitating the presentation of the experimental results. Therefore, differentiated settings for ϵ_1 and ϵ_2 may lead to worse or better results.

The main threat to external validity should be the assumption in Algorithm 4 that the shortest paths between any two vertices can be multiple. For large and complex graph such as Alpha and OTC, it is common for most vertices to have multiple shortest paths. And for graphs smaller and simpler than EIES, only a small number of multiple shortest paths may exist. Therefore, it may be challenging to generalize this assumption to all graphs. The other threat is the datasets. But because we are using real datasets and the size of the datasets are different, the good results reduce the danger of that threat to some extent.

For the threats to statistical validity, there are the randomness of Laplace noise and graph. We adopt the strategy of repeating the experiment to take the average value to reduce the effect of randomness of Laplace noise. The latter threat refers to the fact that we can only select some vertices on the Alpha and OTC to compute the all-pair shortest paths due to the limitation of computational resources. Although we reduce the contingency by repeatedly selecting randomly, this threat still exists.

Finally, the accuracy of Laplace noise poses a significant threat to the construct validity. In theory, Laplace noise should be a continuous value. However, computers can only handle discrete values, which requires us to round the noises to a certain number of decimal places. This rounding may introduce privacy risks and affect the accuracy of the noise. Therefore, the accuracy of the Laplace noise should be carefully considered to ensure the validity of the results.

VII. Conclusion and Future Work

In this paper, we propose differentially private algorithms for privately publishing all-pairs shortest paths in weight-sensitive graphs with low errors. Due to the complexity of edge structures, perturbations on edges can cause significant distortion of shortest paths. Therefore, we divide edges into internal and external edges based on edge betweenness centrality and add positive and negative noise separately. To avoid unnecessary rounding, we implement randomized response on internal edges. Furthermore, we propose Algorithm 4 to find shorter paths closer to the real ones from approximate shortest paths. This algorithm uses cumulative multiplication of edge betweenness centrality to select the better shortest paths.

Compared with adding global Laplace noise, the two perturbations we propose can effectively reduce the change rate of the shortest paths for both large and small graph. Additionally, Algorithm 4 can significantly improve the change rate, reducing it by 8.3% compared to the original value. While our proposed methods do not directly reduce the error of ASPD, Algorithm 4 can still help to reduce the change rate of large graphs by 1.1% compared to the original values.

Therefore, our proposed approach of dividing edges into internal and external based on edge betweenness centrality and adding noise separately can provide a better balance between privacy and accuracy when publishing all-pairs shortest paths and distances in weight-sensitive graphs. Furthermore, Algorithm 4 can effectively reconstruct the true shortest paths from approximate ones and reduce the error in large graphs, which highlights the importance of considering the edge structure when designing differentially private algorithms for graph data.

Given our acknowledged limitations, our future research directions encompass several key areas. One significant avenue is the development of an online differential privacy mechanism for answering shortest path queries using the exponential mechanism, which offers better control over privacy. In this context, we can circumvent the need to introduce noise to all weights, thereby minimizing the degradation of utility. Another promising direction involves the publication of shortest paths in decentralized graphs with reduced errors. In social networks, users are often hesitant to share edge weights related to sensitive data with third parties, necessitating the perturbation of sensitive weights before sharing. However, this approach may limit our ability to use edge betweenness centrality for edge classification and to identify closer approximate shortest paths. Consequently, we must explore alternative methods to enhance utility in such scenarios. Lastly, it is imperative to address the potential privacy vulnerabilities posed by the threats we have identified. Ensuring robust privacy protection remains a paramount concern in our future endeavors.

References

[1] D. Wang and P. Wang, "On the anonymity of two-factor authentication schemes for wireless sensor networks: Attacks, principle and solutions," *Comput. Netw.*, vol. 73, pp. 41–57, 2014.
[2] D. Wang, D. He, P. Wang, and C.-H. Chu, "Anonymous two-factor authentication in distributed systems: Certain goals are beyond attainment," *IEEE Trans. Dependable Secure Comput.*, vol. 12, no. 4, pp. 428–442, Jul./Aug. 2015.
[3] D. Wang and P. Wang, "Two birds with one stone: Two-factor authentication with security beyond conventional bound," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 4, pp. 708–722, Jul./Aug. 2018.
[4] Q. Jiang, N. Zhang, J. Ni, J. Ma, X. Ma, and K.-K. R. Choo, "Unified biometric privacy preserving three-factor authentication and key agreement for cloud-assisted autonomous vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 9390–9401, Sep. 2020.
[5] Z. Liu et al., "A privacy-preserving outsourcing computing scheme based on secure trusted environment," *IEEE Trans. Cloud Comput.*, vol. 11, no. 3, pp. 2325–2336, Jul.-Sep. 2023.
[6] M. Hay, G. Miklau, D. Jensen, P. Weis, and S. Srivastava, "Anonymizing social networks," *Comput. Sci. Dept. Fac. Pub. Ser.*, 2007, Art. no. 180.
[7] K. Liu and E. Terzi, "Towards identity anonymization on graphs," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2008, pp. 93–106.
[8] A. Narayanan and V. Shmatikov, "De-anonymizing social networks," in *Proc. 30th IEEE Symp. Secur. Privacy*, 2009, pp. 173–187.
[9] B. Ding, J. Kulkarni, and S. Yekhanin, "Collecting telemetry data privately," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 3574–3583.
[10] Ú. Erlingsson, V. Pihur, and A. Korolova, "Rappor: Randomized aggregatable privacy-preserving ordinal response," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2014, pp. 1054–1067.

[11] H. Asi, V. Feldman, T. Koren, and K. Talwar, "Near-optimal algorithms for private online optimization in the realizable regime," in *Proc. 40th Int. Conf. Mach. Learn.*, 2023, pp. 1107–1120.
[12] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in *Proc. IEEE 9th Int. Conf. Data Mining*, 2009, pp. 169–178.
[13] K. Nissim, S. Raskhodnikova, and A. Smith, "Smooth sensitivity and sampling in private data analysis," in *Proc. 39th Annu. ACM Symp. Theory Comput.*, 2007, pp. 75–84.
[14] J. Imola, T. Murakami, and K. Chaudhuri, "Locally differentially private analysis of graph statistics," in *Proc. 30th USENIX Secur. Symp.*, 2021, pp. 983–1000.
[15] B. Chen, C. Hawkins, K. Yazdani, and M. Hale, "Edge differential privacy for algebraic connectivity of graphs," in *Proc. IEEE 60th Conf. Decis. Control*, 2021, pp. 2764–2769.
[16] S. Raskhodnikova and A. Smith, "Lipschitz extensions for node-private graph statistics and the generalized exponential mechanism," in *Proc. IEEE 57th Annu. Symp. Found. Comput. Sci.*, 2016, pp. 495–504.
[17] E. Shen and T. Yu, "Mining frequent graph patterns with differential privacy," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2013, pp. 545–553.
[18] C. Borgs, J. Chayes, and A. Smith, "Private graphon estimation for sparse graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1369–1377.
[19] A. Sealfon, "Shortest paths and distances with differential privacy," in *Proc. 35th ACM SIGMOD-SIGACT-SIGAI Symp. Princ. Database Syst.*, 2016, pp. 29–41.
[20] B. Ghazi, R. Kumar, P. Manurangsi, and J. Nelson, "Differentially private all-pairs shortest path distances: Improved algorithms and lower bounds," 2022, *arXiv:2203.16476*.
[21] J. Y. Chen, S. Narayanan, and Y. Xu, "All-pairs shortest path distances with differential privacy: Improved algorithms for bounded and unbounded weights," 2022, *arXiv:2204.02335*.
[22] C. Fan, P. Li, and X. Li, "Breaking the linear error barrier in differentially private graph distance release," 2022, *arXiv:2204.14247*.
[23] L. Liu, J. Wang, J. Liu, and J. Zhang, "Privacy preservation in social networks with sensitive edge weights," in *Proc. Soc. Ind. Appl. Math. – SIAM Int. Conf. Data Min., Proc. Appl. Math.*, 2009, pp. 954–965.
[24] S.-L. Wang et al., "Anonymizing shortest paths on social network graphs," in *Proc. Asian Conf. Intell. Inf. Database Syst.*, 2011, pp. 129–136.
[25] S.-L. Wang, C.-C. Shih, I.-H. Ting, and T.-P. Hong, "Degree anonymization for K-shortest-path privacy," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, 2013, pp. 1093–1097.
[26] D. J. Wu, J. Zimmerman, J. Planul, and J. C. Mitchell, "Privacy-preserving shortest path computation," 2016, *arXiv:1601.02281*.
[27] S. Ramezanian, T. Meskanen, and V. Niemi, "Privacy preserving shortest path queries on directed graph," in *Proc. 22nd Conf. Open Innov. Assoc.*, 2018, pp. 217–223.
[28] M. Anagreh and P. Laud, "A parallel privacy-preserving shortest path protocol from a path algebra problem," in *Proc. Int. Workshop Data Privacy Manage. Int. Workshop Cryptocurrencies Blockchain Technol.*, 2023, pp. 120–135.
[29] C. Fan and P. Li, "Distances release with differential privacy in tree and grid graph," in *Proc. IEEE Int. Symp. Inf. Theory*, 2022, pp. 2190–2195.
[30] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in *Proc. Theory Cryptogr. Conf.*, 2006, pp. 265–284.
[31] F. D. McSherry, "Privacy integrated queries: An extensible platform for privacy-preserving data analysis," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2009, pp. 19–30.
[32] S. L. Warner, "Randomized response: A survey technique for eliminating evasive answer bias," *J. Amer. Stat. Assoc.*, vol. 60, no. 309, pp. 63–69, 1965.
[33] P. Kairouz, S. Oh, and P. Viswanath, "Extremal mechanisms for local differential privacy," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 492–542.
[34] S. C. Freeman and L. C. Freeman, *The Networkers Network: A Study of the Impact of a New Communications Medium on Sociometric Structure*. Irvine, CA, USA: School of Social Sciences University of California, 1979.
[35] S. Kumar, B. Hooi, D. Makhija, M. Kumar, C. Faloutsos, and V. Subrahmanian, "REV2: Fraudulent user prediction in rating platforms," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, 2018, pp. 333–341.

Bin Cai (Member, IEEE) received the BSc degree from Southwest Normal University, China, in 2002, the MSc and PhD degrees in mechanical engineering from Chongqing University, China, in 2005 and 2012, respectively. He is currently an associate professor with the School of Big Data and Software Engineering, Chongqing University. His research interests include software engineering, optimization method, and cryptography.

Weihong Sheng received the BS degree in software engineering from Chongqing University, Chongqing, China, in 2022. He is currently working toward the master degree with the School of Big Data and Software Engineering, Chongqing University. His research interests include differential privacy and distributed computing.

Jiajun Chen is currently working toward the PhD degree with the School of Big Data and Software Engineering, Chongqing University, Chongqing, China. His current research interests include social networks, differential privacy, Big Data, and privacy preservation.

Chunqiang Hu (Member, IEEE) received the BS degree in computer science and technology from Southwest University, Chongqing, China, in 2006, the MS and PhD degrees in computer science and technology from Chongqing University, in 2009 and 2013, respectively, and the second PhD degree in computer science from The George Washington University, Washington, DC, USA, in 2016. He was a visiting scholar with George Washington University from 2011 to 2012. He is currently a faculty member with the School of Big Data and Software Engineering, Chongqing University. His research interests include Blockchain, privacy-aware computing, Big Data security and privacy, and algorithm design and analysis. He was honored with the Hundred-Talent Program by Chongqing University. He is a senior member of CCF (China Computer Federation) and a member of the ACM.

Jiguo Yu (Fellow, IEEE) received the PhD degree in School of mathematics from Shandong University, in 2004. From 2007, he has been a professor in the School of Computer Science, Qufu Normal University, Shandong, China. He is currently a professor in the Big Data Institute, Qilu University of Technology. His main research interests include wireless networks, distributed algorithms, peer-to-peer computing, and graph theory. In particular, he is interested in designing and analyzing algorithms for many computationally hard problems in networks. He is a senior member of the CCF (China Computer Federation).