Locally Differentially Private Analysis of Graph Statistics

Jacob Imola* (UCSD)  Takao Murakami* (AIST)  Kamalika Chaudhuri (UCSD)

LDP (Local Differential Privacy)

Graph Statistics

Graph Statistics
- Is important to understand a connection pattern in a social graph.

E.g., Degree distribution
- Degree = #edges connected to a node.
- Degree distribution = distribution of #friends in a social network.
E.g., Subgraph Counts
- **Triangle** is a set of 3 nodes with 3 edges.
- **$k$-star** consists of a central node connected to $k$ other nodes.

<table>
<thead>
<tr>
<th>Shape</th>
<th>Name</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Triangle" /></td>
<td>Triangle</td>
<td>2</td>
</tr>
<tr>
<td><img src="image" alt="2-star" /></td>
<td>2-star</td>
<td>15</td>
</tr>
<tr>
<td><img src="image" alt="3-star" /></td>
<td>3-star</td>
<td>6</td>
</tr>
</tbody>
</table>

E.g., Clustering Coefficient
- Probability that two friends of a user will also be a friend.
- $= 3 \times \text{#triangles} / \text{#2-stars}$ (40% in the above graph).

Will be a friend (after friend suggestion)?
Privacy Issues

Triangle/$k$-star counts can reveal (sensitive) friendship information.

E.g., Suppose that $v_2$ is an (honest-but-curios) adversary.

I know all edges between $v_3$ ... $v_7$.
Who are friends with $v_1$?

<table>
<thead>
<tr>
<th>Shape</th>
<th>Name</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2-star</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Triangle</td>
<td>5</td>
</tr>
</tbody>
</table>

Friends of $v_1$ are $v_3, v_4, v_6$.

We need to obfuscate #k-stars and #triangles to strongly protect user privacy.
Outline

- Local Differential Privacy (LDP)
  - User obfuscates her personal data by herself (i.e., no trusted third party).

![Local DP](image)

![Centralized DP](image)

Strong Privacy

1. Privacy is protected against attackers with any background knowledge.
2. Original data are not leaked from DB (unlike centralized DP).

- Our Contributions
  - We provide algorithms for $k$-stars and triangles under LDP with utility guarantees.
  - In particular, we show upper/lower-bounds on the estimation error.
Contents

LDP on Graphs
(Local Graph Model, Edge LDP)

Our Algorithms
(Our Algorithms for $k$-Stars/Triangles, Upper-Bounds)

Lower-Bounds

Experiments
LDP on Graphs

- **Graph**
  - Can be represented as an adjacency matrix $A$ (1: edge, 0: no edge).
  - User $v_i$ knows her neighbor list $a_i$ ($i$-th row of $A$).

- **Local Graph Model**
  - User $v_i$ obfuscates her neighbor list $a_i$ and sends noisy data $R_i(a_i)$ to a server.

```
graph G
v1 - v2 - v3 - v4

adjacency matrix A

v1 | 0 | 1 | 1 | 0
v2 | 1 | 0 | 1 | 1
v3 | 1 | 1 | 0 | 1
v4 | 0 | 0 | 1 | 0

R_1(a_1) = a_1
R_n(a_n)
```

randomizer
### LDP on Graphs

- **Edge LDP [Qin+, CCS17]**
  - Protects a single bit in a neighbor list \( a \in \{0,1\}^n \) with privacy budget \( \varepsilon \).

Randomizer \( R \) provides \( \varepsilon \)-edge LDP if for all \( a, a' \in \{0,1\}^n \) that differ in one bit and all \( y \in \mathcal{Y} \),

\[
\Pr[R(a) = y] \leq e^\varepsilon \Pr[R(a') = y]
\]

- 1 edge affects 2 elements of \( A \)  \( \rightarrow \) each edge is protected with at most \( 2\varepsilon \).
- Our triangle algorithm uses only \( \rightarrow \) each edge is protected with \( \varepsilon \).
Contents

LDP on Graphs
(Local Graph Model, Edge LDP)

Our Algorithms
(Our Algorithms for $k$-Stars/Triangles, Upper-Bounds)

Lower-Bounds

Experiments
Our Algorithms

- Our Algorithm for $k$-Stars (Overview)
  1. Each user $v_i$ adds the Laplacian noise to her $k$-star count $r_i$. → edge LDP.
  2. Server calculates the sum of noisy counts as an estimate.

![Diagram of 2-stars and summation]

- Upper-Bound ($n$: #users, $d_{max}$: max degree ($\ll n$))
  - For a fixed $\varepsilon$, the expected l2-loss (square error) of our estimate is: $O(nd_{max}^{2k-2})$.
  - Later, we prove that this is order optimal in the one-round LDP model.
Our Algorithms

- Triangles
  - More challenging because a user cannot see an edge between others.

- Our Algorithm for Triangles (1st Round)
  - Each user applies RR to each bit of her neighbor list. → edge LDP.
  - Each user sends **noisy edges**. Server publishes the noisy graph $G'$.

![Diagram of Triangles and RR](image)
Our Algorithms

- **Our Algorithm for Triangles (2nd Round)**
  - Each user can count *triangles including one noisy edge* using noisy graph $G'$.
  - Each user sends #noisy triangles (with post-processing) + Lap. $\rightarrow$ edge LDP.
  - Server calculates an unbiased estimate of #triangles.

- **Upper-Bound ($n$: #users, $d_{max}$: max degree ($\ll n$))**
  - All edges are noisy (1st round) $\rightarrow$ Only one edge is noisy (2nd round).
  - Expected l2-loss is reduced from $O(n^4)$ (1st round) to $O(nd_{max}^3)$ (2nd round).
Contents

LDP on Graphs
(Local Graph Model, Edge LDP)

Our Algorithms
(Our Algorithms for $k$-Stars/Triangles, Upper-Bounds)

Lower Bounds

Experiments
Overview

- Our $k$-star algorithm achieves the l2-loss of $O\left( nd_{max}^{2k-2} \right)$ ($n$: #users, $d_{max} \ll n$).
- We show that the factor of $n$ is necessary for $k$-stars and triangles in one-round LDP.

How?

- We introduce a set of graphs called "independent cube".
- We show there is a lower bound for the set of graphs.
Independent Cube (Informal)

- Consider a query $f$ (e.g. \#triangles, \#$k$-stars) on a graph $G$ with $n$ nodes.
- Prepare edges $M$ s.t. each node has one edge (i.e. perfect matching).

- We say a set of graphs $\mathcal{A}$ forms an $(n, D)$-independent cube if adding edge $e \in M$ independently increases (or decreases) $f$ by $C_e \geq D$.

\begin{align*}
 f(G_1) &= 0 \\
 f(G_2) &= 3 \\
 f(G_3) &= 2 \\
 f(G_4) &= 5
\end{align*}

\begin{align*}
 C(v_1,v_2) &= 2 \\
 C(v_3,v_4) &= 3
\end{align*}
There exist independent cubes for $k$-stars and triangles (→ our paper).

In one-round LDP, the expected $l_2$-loss for an $(n,D)$-independent cube is: $\Omega(nD^2)$. 

Lower-Bounds for Independent Cubes
Lower Bounds

- Upper/Lower-Bounds
  - In $k$-stars, our one-round local algorithm is order optimal.
  - Any one-round local algorithm is outperformed by the centralized one.
  - Yet, our algorithms achieve $O(n)$ (when we ignore $d_{max}$), which is small.

<table>
<thead>
<tr>
<th></th>
<th>Centralized</th>
<th>One-Round Local</th>
<th>Two-Rounds Local</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper-Bound</td>
<td>$O(d_{max}^{2k-2})$</td>
<td>$\Omega(d_{max}^{2k-2}n)$</td>
<td>$O(d_{max}^{2k-2}n)$</td>
</tr>
<tr>
<td>Lower-Bound</td>
<td>$\Omega(d_{max}^{2k-2}n)$</td>
<td>$O(d_{max}^{2k-2}n)$</td>
<td>-</td>
</tr>
<tr>
<td>$k$-stars</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>triangles</td>
<td>$O(d_{max}^2)$</td>
<td>$\Omega(d_{max}^2n)$</td>
<td>$O(n^4)$</td>
</tr>
</tbody>
</table>
Contents

LDP on Graphs
(Local Graph Model, Edge LDP)

Our Algorithms
(Our Algorithms for $k$-Stars/Triangles, Upper-Bounds)

Lower Bounds

Experiments
Experiments

- IMDB (Internet Movie Database)
  - Graph with 896308 nodes (actors).
  - Average degree = 63.7.

- Orkut Dataset
  - Social graph with 3072441 nodes (users).
  - Average degree = 38.1. More sparse than IMDB.

- For each dataset, we randomly selected $n$ nodes from the whole graph.
Experiments

- Result 1: l2-loss
  - In triangles, Local2R (2-rounds) outperforms Local1R (1-round).
  - Difference is larger in Orkut because it is more sparse ($d_{max}$ is smaller).
  - Local is outperformed by Central.
  - As $n$ increases, the l2-loss increases ↩ true counts (#triangles and #k-stars) increase.
Experiments

- Result 2: Relative Error
  - Relative error \(\left(= \frac{|true\ count - estimate|}{true\ count}\right)\) decreases as \(n\) increases.
  - Our algorithms achieve relative error \(\ll 1\) (high utility) when \(\varepsilon = 1\) or 2.

![Relative Error Graphs](image-url)
Conclusions

- This Work
  - For $k$-stars, we provided an order optimal algorithm.
  - For triangles, we showed an additional round significantly improves utility.
  - We provided new lower-bounds for $k$-stars and triangles.

- Future Work
  - Algorithms for other subgraph counts; e.g., $\#\text{cliques}$, $\#k$-hop paths.
Thank you for your attention!

Q&A

jimola at eng.ucsd.edu, takao-murakami at aist.go.jp