Privacy and Anonymity in Graph Data

Michael Hay, Siddharth Srivastava, Philipp Weis

May 2006

Outline

1. Introduction
2. Emipirical Analysis of Data Disclosure
3. Modelling Privacy and Disclosure for Graph Data
4. Graph Anonymization Techniques

Single-table anonymization

What anonymization is about:

- Want to publish data about invidivuals without revealing any private information
- Examples: census data, medical records, network traces, ...
- High level idea: separate sensitive from non-sensitive information, and remove all (or most) sensitive information

Anonymization of single-table data is studied widely and used in practice.

k-Anonymity

- Introduced in [?].
- Ensures that any individual cannot be distinguished within a group of at least k individuals.
- This is achieved by generalizing attribute values to ranges.

k-Anonymity

- Introduced in [?].
- Ensures that any individual cannot be distinguished within a group of at least k individuals.
- This is achieved by generalizing attribute values to ranges.

[FL, GU]	[96932, 99401]	PAXSON COMMUNICATIONS CORP	REP	2000
[FL, GU]	[96932, 99401]	PAXSON COMMUNICATIONS CORP	DEM	300
[FL, GU]	[96932, 99401]	PAXSON COMMUNICATIONS CORP	DEM	300
[FL, GU]	[96932, 99401]	PAXSON COMMUNICATIONS CORP	DEM	1000
[FL, GU]	[96932, 99401]	PAXSON COMMUNICATIONS CORP	REP	300
[FL, GU]	[96932, 99401]	PAXSON COMMUNICATIONS CORP	DEM	500
[FL, GU]	[96932, 99401]	PAXSON COMMUNICATIONS CORP	DEM	500
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	250
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	250
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	250
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	250
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	250
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	500
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	250
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	250
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	2000
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	250
MA	01002	[AMHERST COLLEGE, BULKELY RICHARDSON]	DEM	250

Goals of the Project

- Obtain examples of graph data, get a feeling for private and non-sensitive properties of these graphs, experiment with re-identification
- Develop a theoretical framework for graph data publication, privacy, anonymization and information disclosure
- Investigate conventional anonymization techniques on graph data. Where do they fail?
- Develop new techniques that can be used to anonymize graph data

Outline

Adversary's Perspective on Graph Anonymization

- What properties about the real-world can the adversary infer from published data?
- We investigate the following **re-identification** task:
  input:
  - a set of real-world objects (Enron employees)
  - some background knowledge about the objects
  - a published graph (email communications), 'anonymized' by removing object identifiers (e.g. *joe@enron.com* becomes v_{10})

  output:
  - map each real-world object to a vertex (or a subset of vertices) in the published graph (e.g. *joe@enron.com* $\rightarrow \{v_4, v_{10}, v_{17}, v_{65}\}$)
- Turns out re-identification can be succinctly described as a constraint satisfaction problem (CSP), except enumerate all assignments rather than find a single assignment

What is a Constraint Satisfaction Problem?

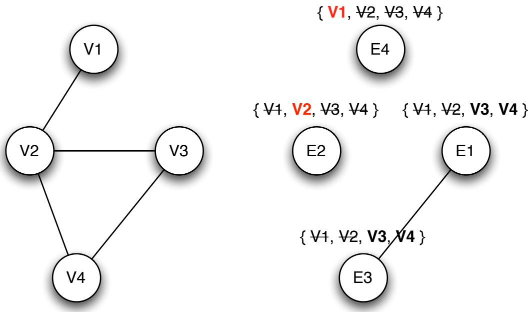
- A CSP is defined by:
  - a set of **variables** $X_1, \ldots, X_n$
  - each variable X_i has a **domain** D_i of possible **values**
  - a set of **constraints** $C_1, \ldots, C_m$ which constrain the possible values that a variables can take on
  - A **solution** is an assignment of variables to values such that constraints are satisfied.
  - Any CSP can be represented as a **constraint graph**: one vertex per variable and an edge for each binary constraint.

Re-identification as a CSP

- **variables**: one per real-world object
- **domains**: the set of vertices in published graph $\{v_1, \ldots, v_n\}$
- **constraints**: background knowledge
  - unary constraints: $degree(o_i)$, $connected\_component\_size(o_i)$
  - binary constraint: $edge(o_i, o_j)$, $path_k(o_i, o_j)$
  - n-ary constraint: $all\_different(o_1, \ldots, o_n)$
- **solution**: for each object o, the set of plausible vertices. I.e. a subset of vertices $V' \subseteq \{v_1, \ldots, v_n\}$ such that when o was mapped to $v \in V'$ a valid solution was found
- **constraint graph**: surprisingly sparse, so CSP solver runs fast!

Toy Example

PUBLISHED GRAPH CONSTRAINT GRAPH



Background Knowledge:
degree(E2) = 3
edge(E1,E3)

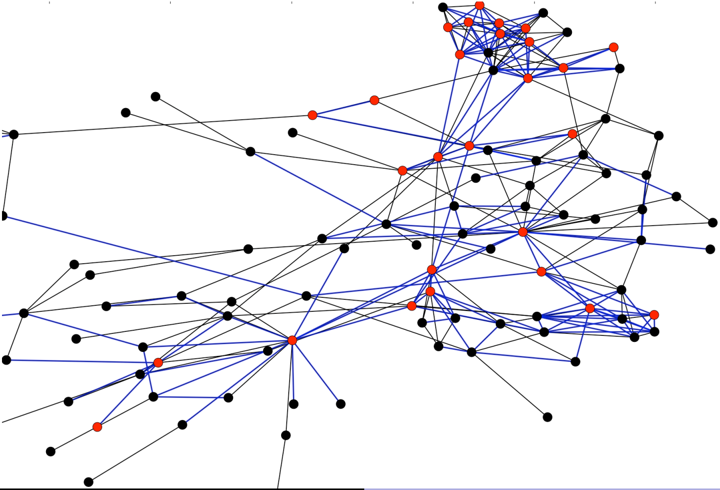
Empirical Analysis: How does background knowledge help?

Email communications of 117 Enron employees, private data that is now part of public record (following subpoena). Task: re-identify Enron employees in graph of email communication (edge means ≥ 5 emails both directions).

Background Knowledge	Ave. Domain Size	No. Reidentified
None	117	0 (out of 117)
Centrality Quartile	29.2	0
Degree Only	13.2	4
Degree And Centrality Quartile	5.4	12
25% edges	-	-
Degree And 25% edges	8.2	28
Degree And 50% edges	2.40	63

Re-identifying Enron Employees from Emails

Background knowledge was node degree and a sample of 25% of the edges (shown in blue), weighted by frequency of communication. Red nodes have been re-identified.



Outline

Node properties and types

Goals of the anonymization:

- We consider information about specific individuals private.
- We want to publish a modified version of the original data that does not reveal any private information, but is still useful.

Node properties and types

Goals of the anonymization:

- We consider information about specific individuals private.
- We want to publish a modified version of the original data that does not reveal any private information, but is still useful.

Classify nodes in the graph with respect to their properties. The type of a node is a summary of all relevant properties of a node. Types contain information like

- Node attributes (just as in the tabular case)
- Degree
- Centrality
- Neighborhood information

Anonymization with node types

How we anonymize our data

- Remove identifiers (names) from some or all nodes
- Anonymize node and edge attributes (as with classical anonymization)
- Modify the graph

Anonymization with node types

How we anonymize our data

- Remove identifiers (names) from some or all nodes
- Anonymize node and edge attributes (as with classical anonymization)
- Modify the graph

Let N be the set of individuals represented in the graph, and let V be the set of (unnamed) nodes in the graph.

Anonymization with node types

How we anonymize our data

- Remove identifiers (names) from some or all nodes
- Anonymize node and edge attributes (as with classical anonymization)
- Modify the graph

Let N be the set of individuals represented in the graph, and let V be the set of (unnamed) nodes in the graph.

The adversary tries to use his background knowledge to re-identify certain individuals, i.e. he tries to learn their type.

Anonymization with node types

How we anonymize our data

- Remove identifiers (names) from some or all nodes
- Anonymize node and edge attributes (as with classical anonymization)
- Modify the graph

Let N be the set of individuals represented in the graph, and let V be the set of (unnamed) nodes in the graph.

The adversary tries to use his background knowledge to re-identify certain individuals, i.e. he tries to learn their type.

Background knowledge $\kappa : N \to \mathcal{P}(V)$ or $\kappa : N \to \mathcal{P}(T)$.
Compare this to knowledge after the publication κ'.

Background Knowledge and Disclosure

Background knowledge $\kappa : N \rightarrow \mathcal{P}(V)$ or $\kappa : N \rightarrow \mathcal{P}(T)$ (or probability distributions).

To guarantee k-anonymity: for all individuals $n \in N$: $\kappa'(n) \geq k$. Ignore adversary's background knowledge here.

Background Knowledge and Disclosure

Background knowledge $\kappa : N \rightarrow \mathcal{P}(V)$ or $\kappa : N \rightarrow \mathcal{P}(T)$ (or probability distributions).

To guarantee k-anonymity: for all individuals $n \in N$: $\kappa'(n) \geq k$. Ignore adversary's background knowledge here.

Consider different distance measures $d(\kappa, \kappa')$ to measure the amound of disclosure.

$$d_{\max}(\kappa, \kappa') = \max \left\{ \frac{\kappa(n) - \kappa'(n)}{\kappa(n)} \,\middle|\, n \in N \right\}$$

Include distance measure between types.

Outline

1. Introduction
2. Emipirical Analysis of Data Disclosure
3. Modelling Privacy and Disclosure for Graph Data
4. Graph Anonymization Techniques

Hurdles in Guaranteeing Privacy in Graphs

From	To	Count
Samson	Delilah	50
Arthur	Merlin	65
Alice	Bob	0
Delilah	Alice	50

Anonymizing Graphs is Difficult

- Tuples are interdependent: cannot merge tuples on any single attribute without possibly disturbing the others or making the graph inconsistent. Renders most anonymization algorithms infeasible.
- Each individual could occur in several tuples but still need not by anonymized.

Degree-Types: A Simple Case

Node-Degree is a simple yet interesting node type to consider.

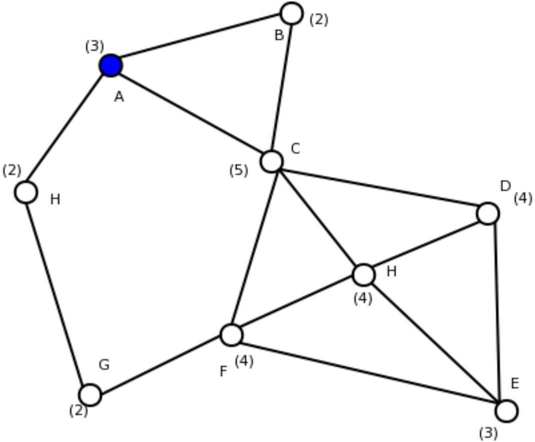
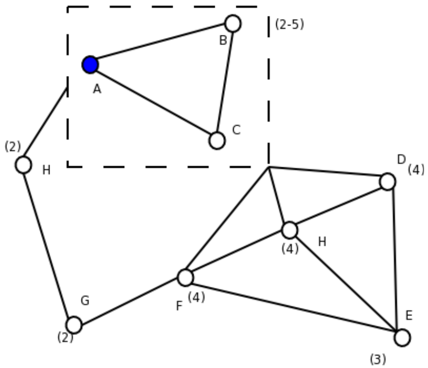


Figure: A Component of Enron Email Communication Graph (only senior/known employees)

Providing Privacy for Degree-Types

- We wish to anonymize without creating false information. Adding/deleting edges to manipulate degrees is ruled out.
- Can add vagueness to the graph.
- Only way to manipulate degrees is to generalize nodes or edges.
- Generalizing nodes is easier. Edge generalization causes more side effects.

A Connectivity Respecting 3-Anonymization on Degree



We can keep the edges of triangle A, B, C because there is one edge between every pair.

How did we do that?

- Basic Idea: merge nodes until all degree-types have at least k nodes.
- Any such grouping will work - but some groupings are better at preserving graph properties than others.

Naive Degree-Based Anonymization

While (! k-anonymized)

1. Find lowest degree with fewer than k nodes.
2. Merge its nodes with nodes of next largest degree with fewer than k nodes.

Naive Degree-Based Anonymization

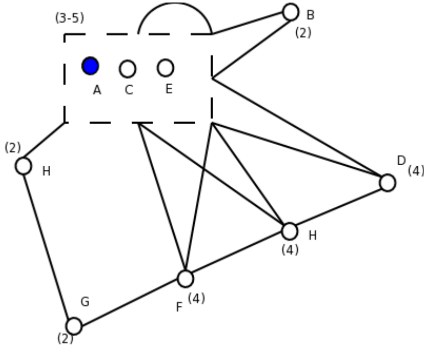
While (! k-anonymized)

1. Find lowest degree with fewer than k nodes.
2. Merge its nodes with nodes of next largest degree with fewer than k nodes.

Comparison of The Two Approaches

- Problem: Does not care about graph structure!
- However, it keeps the Type-ranges small.
- Two counter-acting aspects of utility: Graph Structure and Type-Range.

Result of Naive Degree-Based Anonymization



Advantage: smaller degree ranges.

Degree-Types and k-anonymization

It turns out that achieving privacy for Degree-Types can be done through k-anonymization:
QuasiID = Degree

Employee	Degree
Samson	4
Delilah	2
Arthur	8
Alice	5
Bob	9
Merlin	1

2-anonymization $\Longrightarrow$

Employee	Degree
Merlin	[1-2]
Delilah	[1-2]
Samson	[4-5]
Alice	[4-5]
Bob	[8-9]
Arthur	[8-9]

If adversary has degree information about any individual it will match at least two individuals in published data.

Utilizing k-Anonymization Algorithms

- Anonymization with degree as an attribute will treat nodes with similar degrees as "close" to each other for merging.
- We might want "close"-ness to be defined in terms of graph connectivity.
- Create another attribute, which captures closeness in the graph.
- k-anonymize using this new attribute and the degree.
- Post-process results of k-anonymization to merge nodes whose degrees were merged, into supernodes in the graph.

General Class of Type-Anonymization Algorithms

While (! Anonymized)

1. Use Type-Histogram to determine the Type with lowest frequency.
2. Choose nodes N_h, N_l of highest and lowest degree of this type.
3. Perform one of the following in a suitable ratio:
   - Choice 1: Merge N_h with closest node/supernode of a Type with lesser than k nodes.
   - Choice 2: Merge N_l with node/supernode of the most similar Type with lesser than k nodes.
4. Label the merged node.
5. Update the histogram.

References