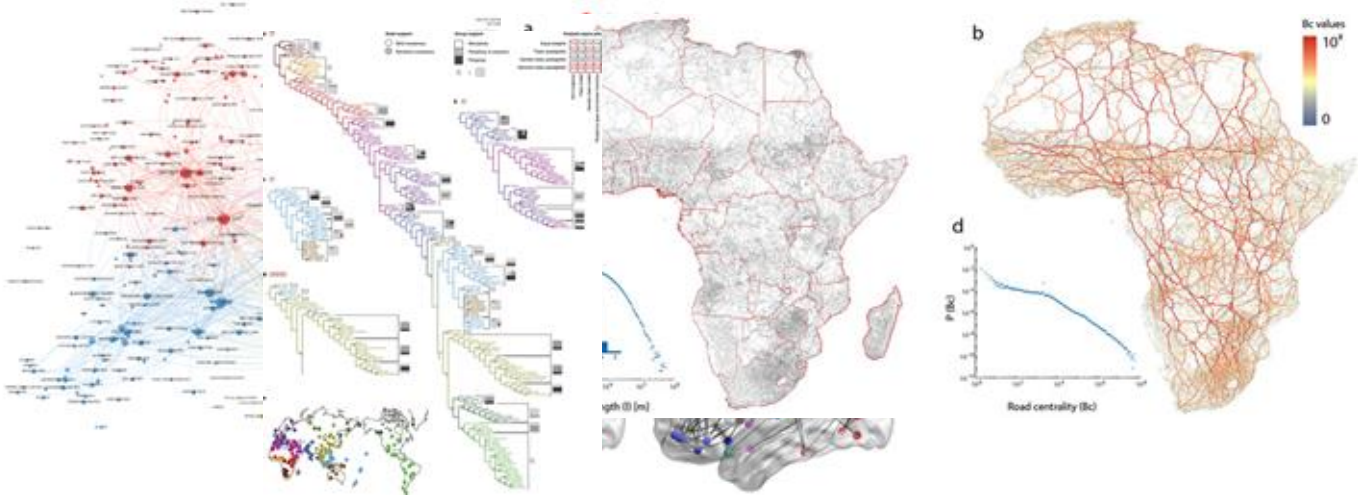
An Efficient Query Recovery Attack Against a Graph Encryption Scheme

Francesca Falzon

Joint work with Kenneth G. Paterson

Graph-Structured Data

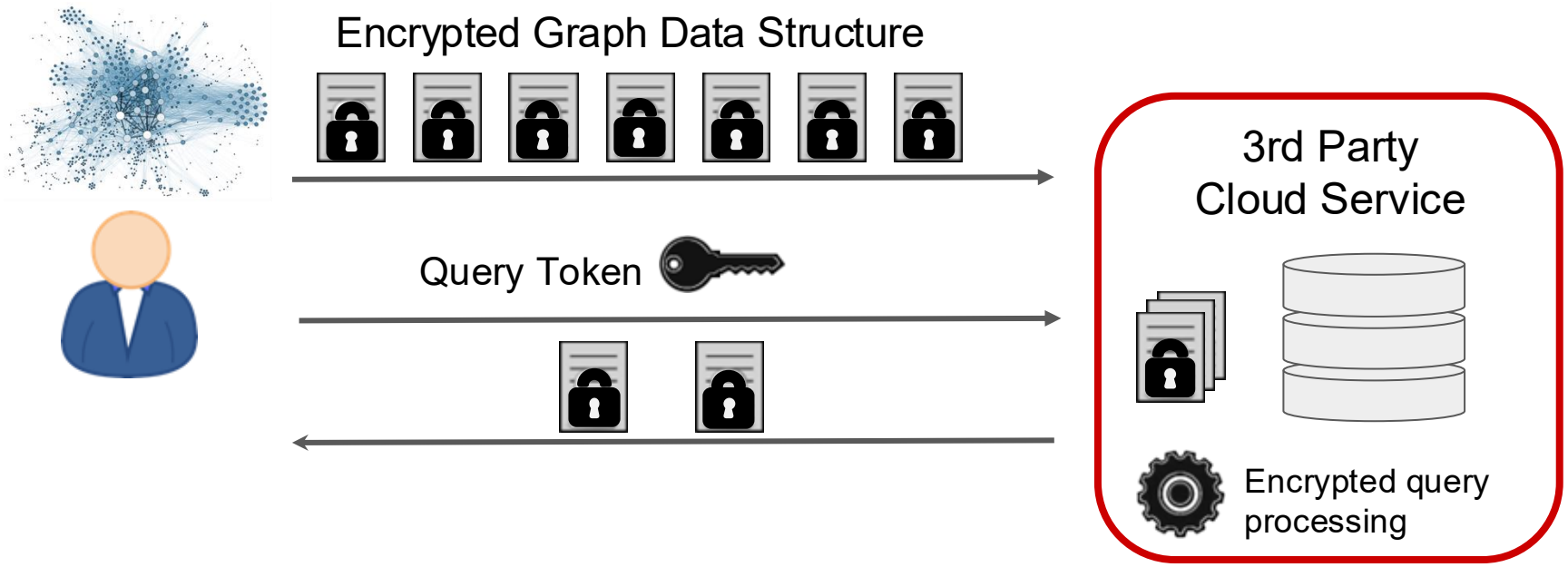
- Can be used to model many real-world phenomena
- Graph Databases in Industry e.g. Amazon Neptune, Facebook TAO, Neo4j, GraphDB



Graphs

- $G = (V, E)$
  - $n = |V|$
  - $m = |E|$
- Weighted or unweighted
- Directed or undirected
- For convenience we consider simple graphs

Setting: Outsourcing a Graph to the Cloud



Graph Encryption Scheme

- A **graph encryption scheme (GES)** is a tuple of 5 algorithms with the following syntax:
  - **KeyGen** is a probabilistic algorithm that takes as input a security parameter λ and outputs a secret key K.
  - **Encrypt** is a probabilistic algorithm that takes a secret key K and a graph G and outputs an encrypted database EDB.
  - **Token** take a key K and query q and returns a search token tk.
  - **Search** takes an encrypted database EDB and a search token tk and outputs a response resp.
  - **Reveal** takes a key K and response resp and outputs plaintext m.

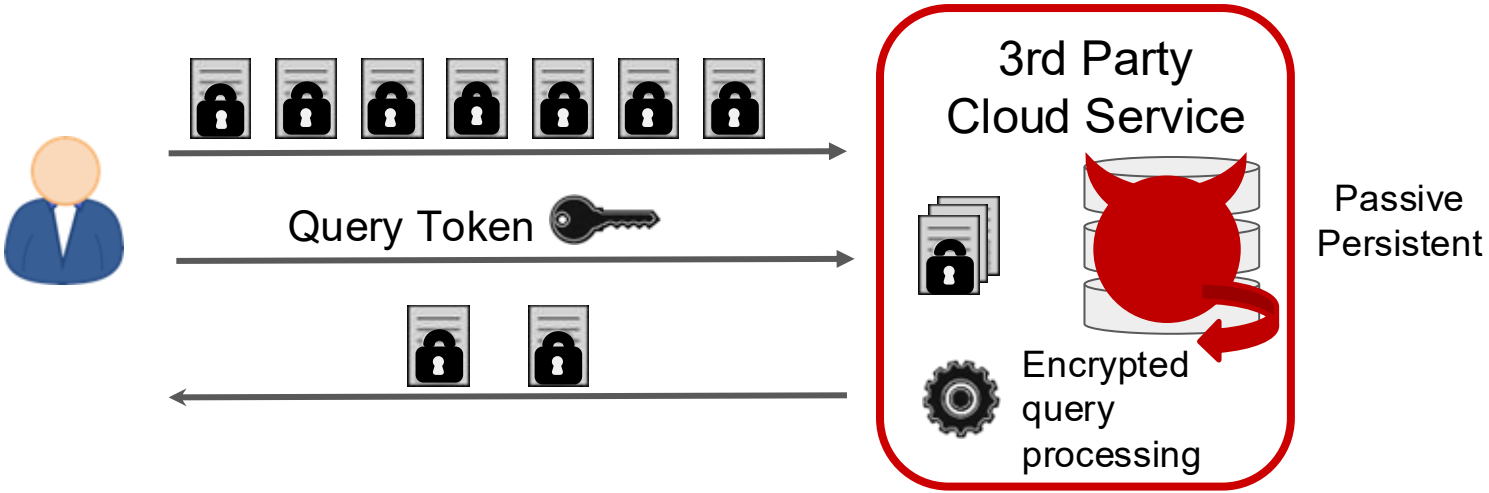
The GKT Scheme by Ghosh et al. (AsiaCCS 2021)

- Encrypts a graph $G = (V, E)$
  - V is a collection of nodes
  - E is a collection of edges connecting pairs of nodes
  - Edges may be weighted
- Supports **Single-Pair Shortest Path** (SPSP) queries on G
  - Input: $(u, v) \in V \times V$
  - Output: path from u to v in G of minimal length

Attacks Against GES

- Goetschmann 2020
  - Presents a query recovery attack against one of the schemes by Meng et al. (CCS 2015) that supports *approximate* shortest *distance* queries
- Falzon and Paterson 2022
  - Present a query recovery attack against the scheme by Ghosh et al. (AsiaCCS 2021) that supports *exact* shortest *path* queries

Server Side Threat Model



Our Contributions

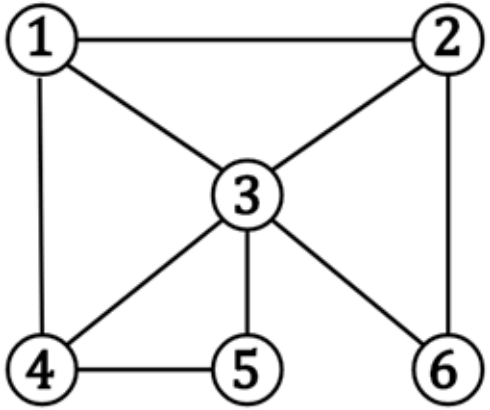
1. Present the first attack against a graph encryption scheme that supports shortest path queries (Ghosh et al. AsiaCCS 21).
2. Runs in $O(n^3)$ and matches the setup of the GKT scheme.
3. Extend a classical algorithm for the tree isomorphism problem.
4. Prove the information theoretic limitations of what a passive adversary can reconstruct from the scheme's leakage.
5. Implement and evaluate our attack against real-world datasets.

Talk Overview

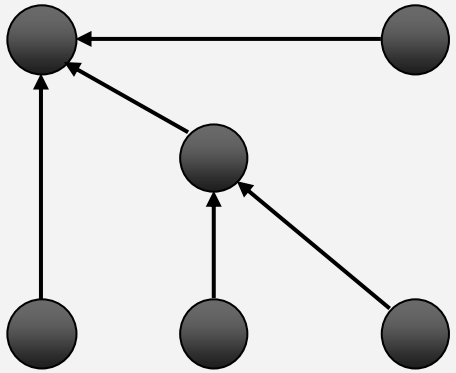
- Introduction
- **Attacking a Graph Encryption Scheme**
  - **GKT Scheme and its Leakage**
  - Extending a Tree Isomorphism Algorithm
  - Query Recovery
- Experiments

GKT Scheme (AsiaCCS 21)

Query pairs
(5,1) & (4,1)

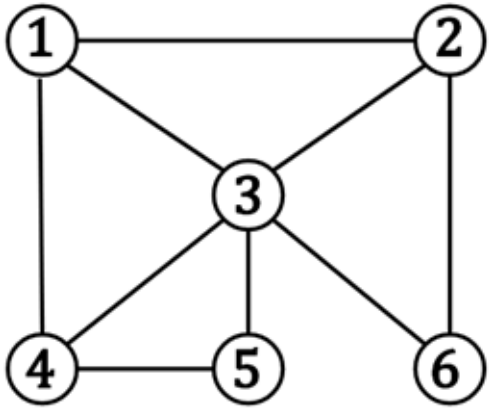


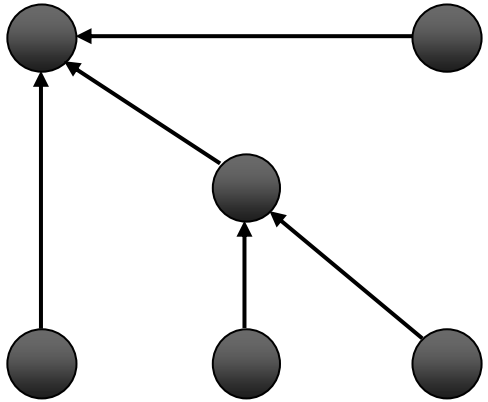
What the adversary sees.



GKT Scheme (AsiaCCS 21)

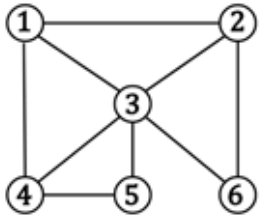
- **Setup Leakage**: # of vertices that are connected
- **Query Leakage**: search pattern + length of path + the intersection of paths that share the same destination





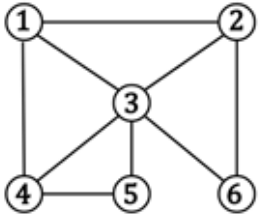
Key Observation

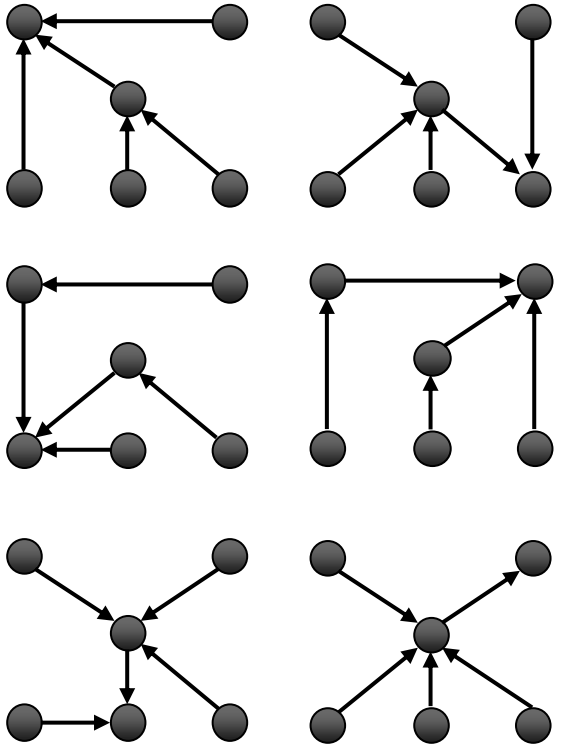
Question: But how can we map shortest paths to shortest paths in isomorphic graphs?



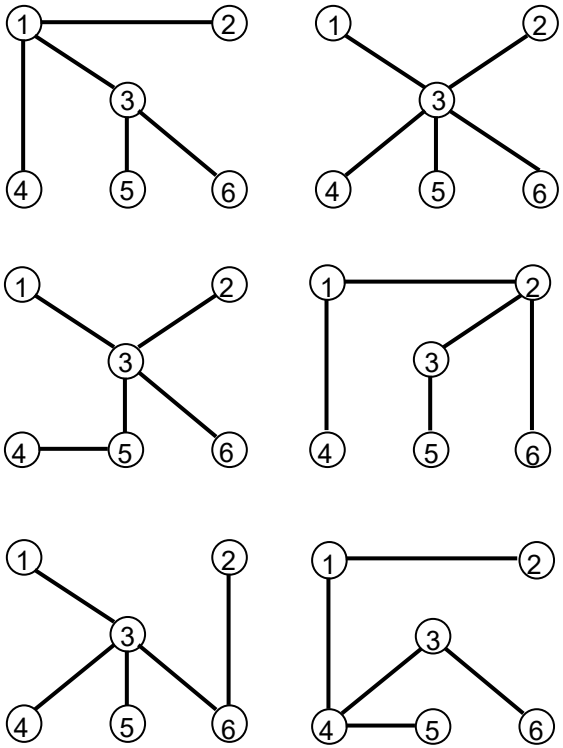
Key Observation

Question: But how can we map shortest paths to shortest paths in isomorphic graphs?

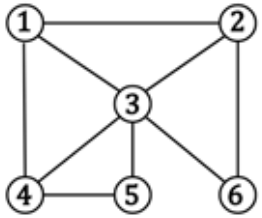


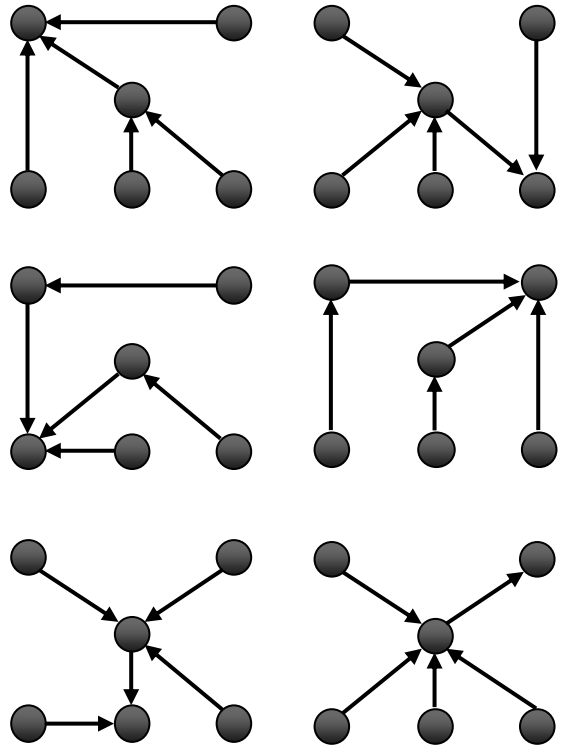


Key Observation

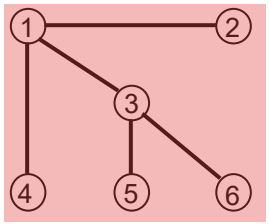


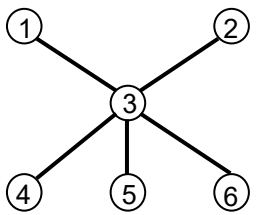
Question: But how can we map shortest paths to shortest paths in isomorphic graphs?

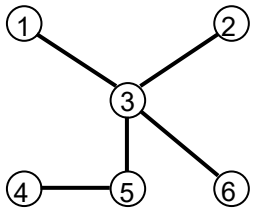


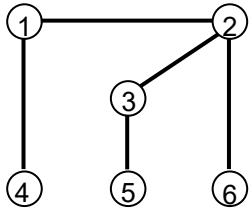


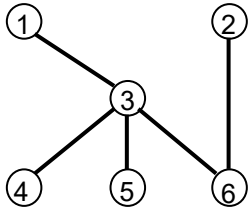
Key Observation

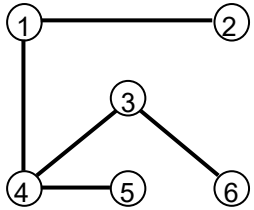




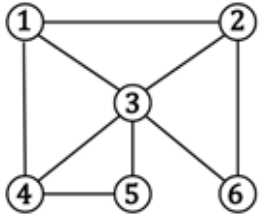


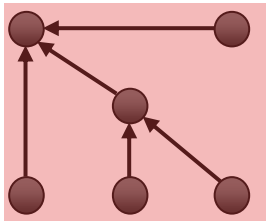


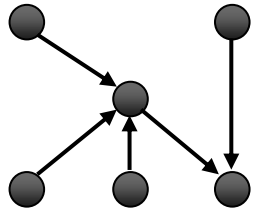


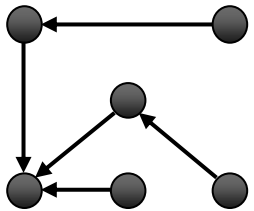


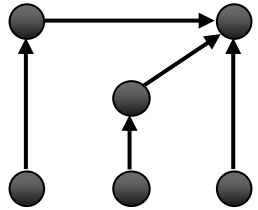
Question: But how can we map shortest paths to shortest paths in isomorphic graphs?

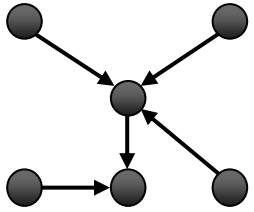


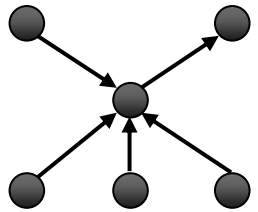




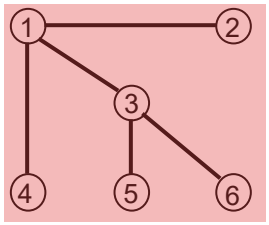


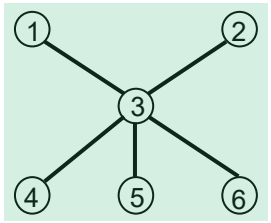


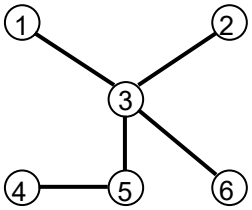


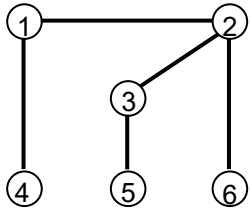


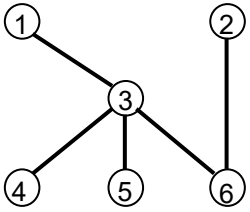
Key Observation

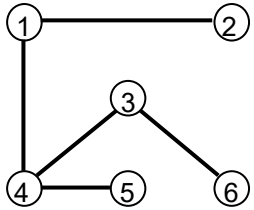




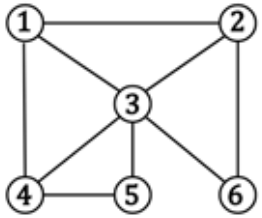


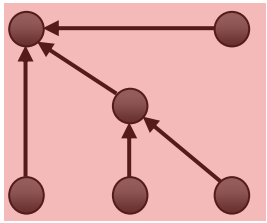


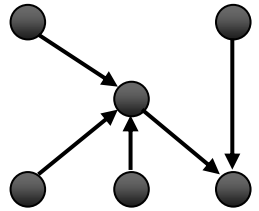


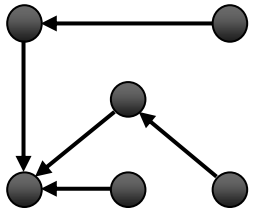


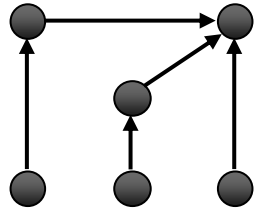
Question: But how can we map shortest paths to shortest paths in isomorphic graphs?

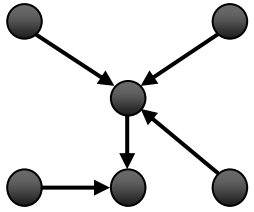


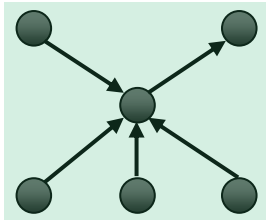




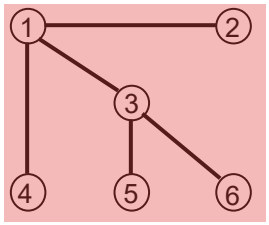


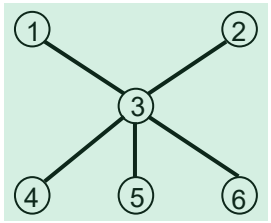


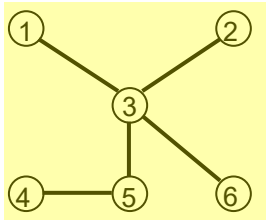


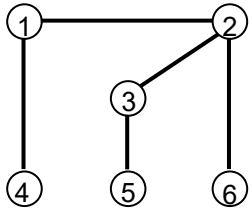


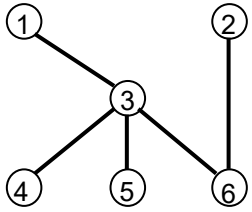
Key Observation

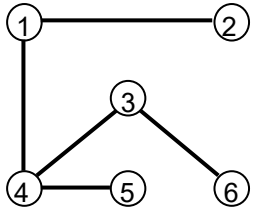




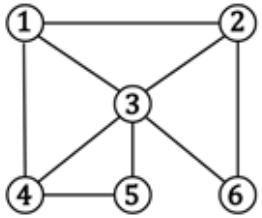


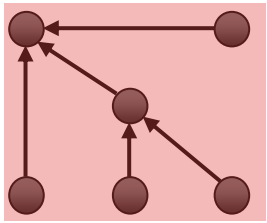


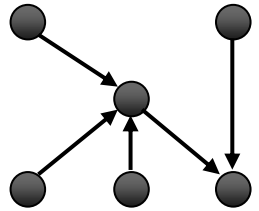


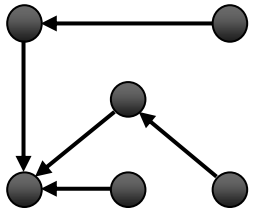


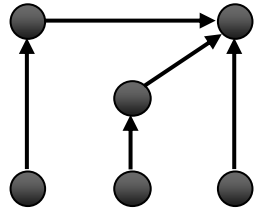
Question: But how can we map shortest paths to shortest paths in isomorphic graphs?

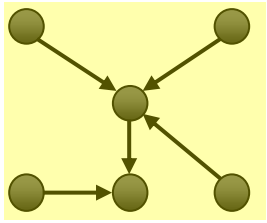


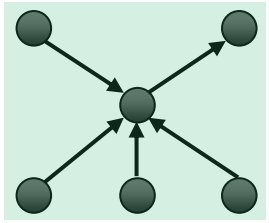




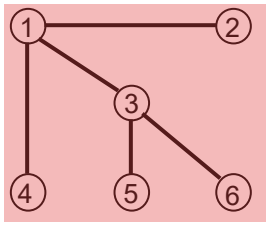


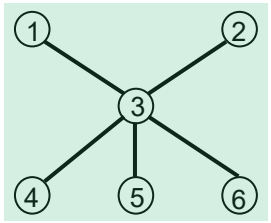


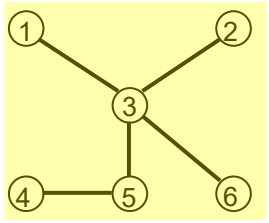


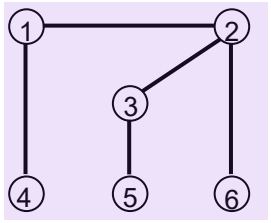


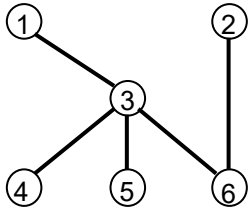
Key Observation

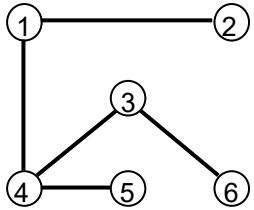




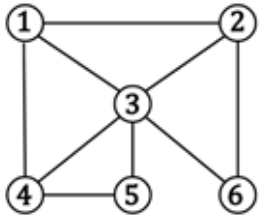


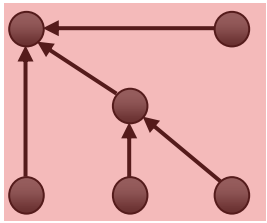


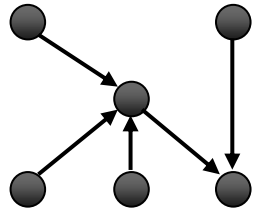


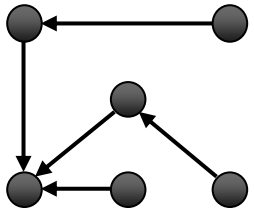


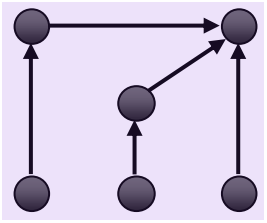
Question: But how can we map shortest paths to shortest paths in isomorphic graphs?

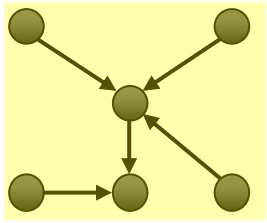


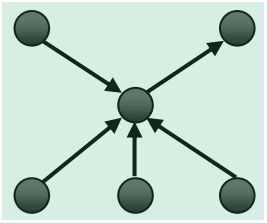




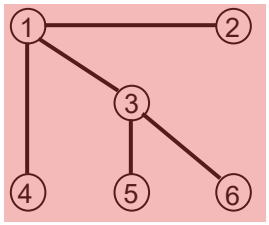


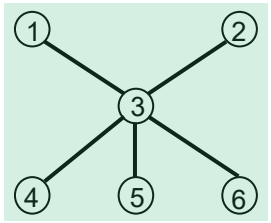


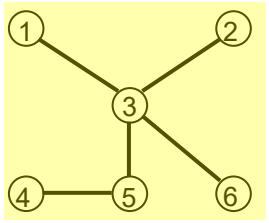


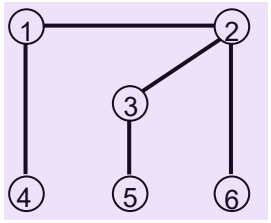


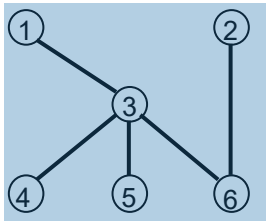
Key Observation

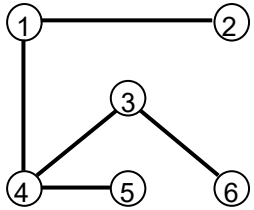




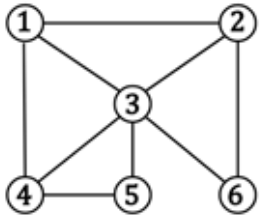


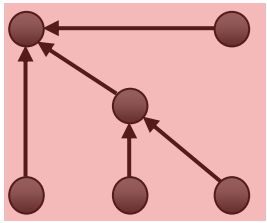


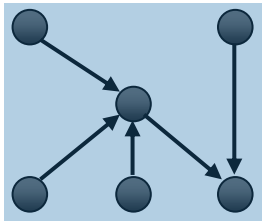


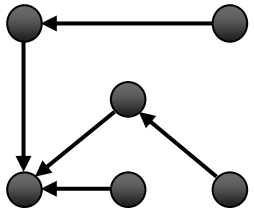


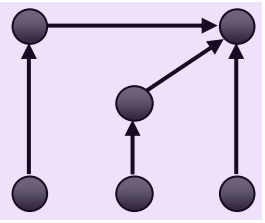
Question: But how can we map shortest paths to shortest paths in isomorphic graphs?

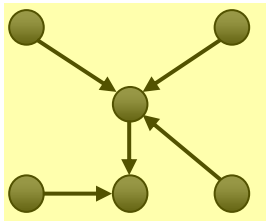


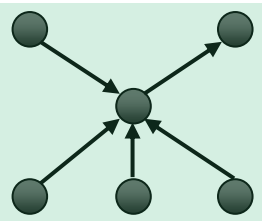




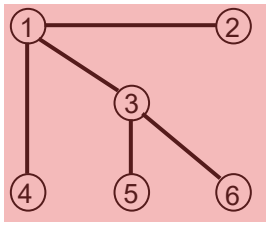


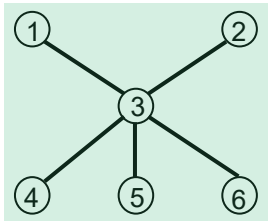


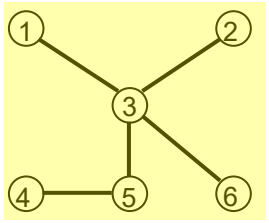


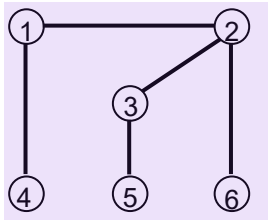


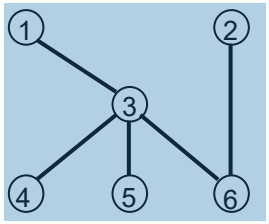
Key Observation

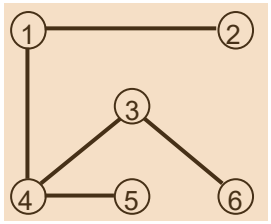




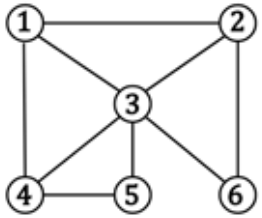


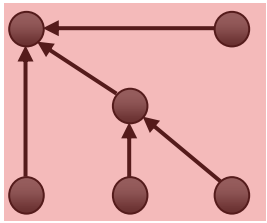


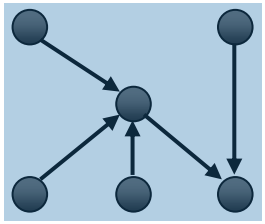


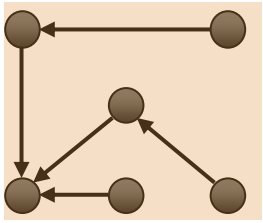


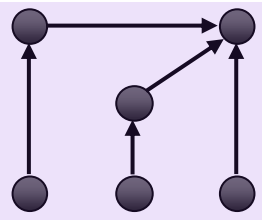
Question: But how can we map shortest paths to shortest paths in isomorphic graphs?

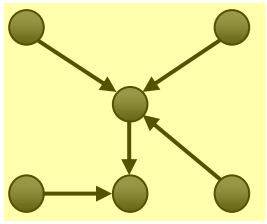


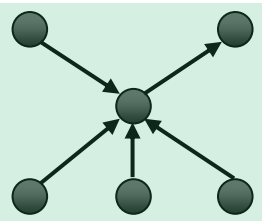












Talk Overview

- Introduction
- **Attacking a Graph Encryption Scheme**
  - GKT Scheme and its Leakage
  - **Extending a Tree Isomorphism Algorithm**
  - Query Recovery
- Experiments

Isomorphisms

- An **isomorphism of rooted trees** $T = (V, E, r)$ and $\mathrm{T}' = (V', E', r')$ is a map φ from T to T' (as graphs) such that $\varphi(r) = r'$ and which preserves edge relationships between the two trees.
    - Denote this as $T \cong T'$

The AHU Algorithm

- Gives us an efficient way to tell if two trees are isomorphic.

Question: But how can we map shortest paths to shortest paths in isomorphic graphs?

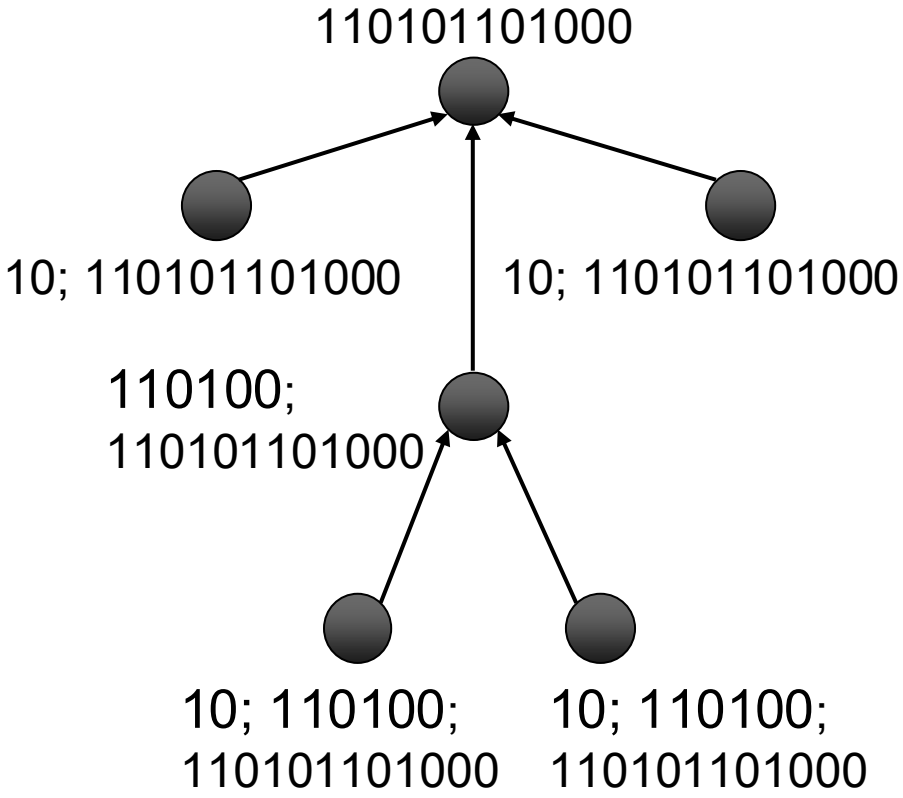
Query Recovery Attack

- **Query Recovery (QR)**: Given graph G and the leakage of a set of SPSP queries on G, compute the set of candidate queries corresponding to each issued query.
  - The attacker knows the graph (e.g., public road network)
  - Attempts to reconstruct the queries (e.g., sensitive trip origin and destination)
  - Attack is within the security model of GKT
- Candidates determined by the isomorphisms between the plaintext trees and the query trees.
- **We show that query recovery is only possible up to rooted-tree isomorphisms between these two sets of trees.**

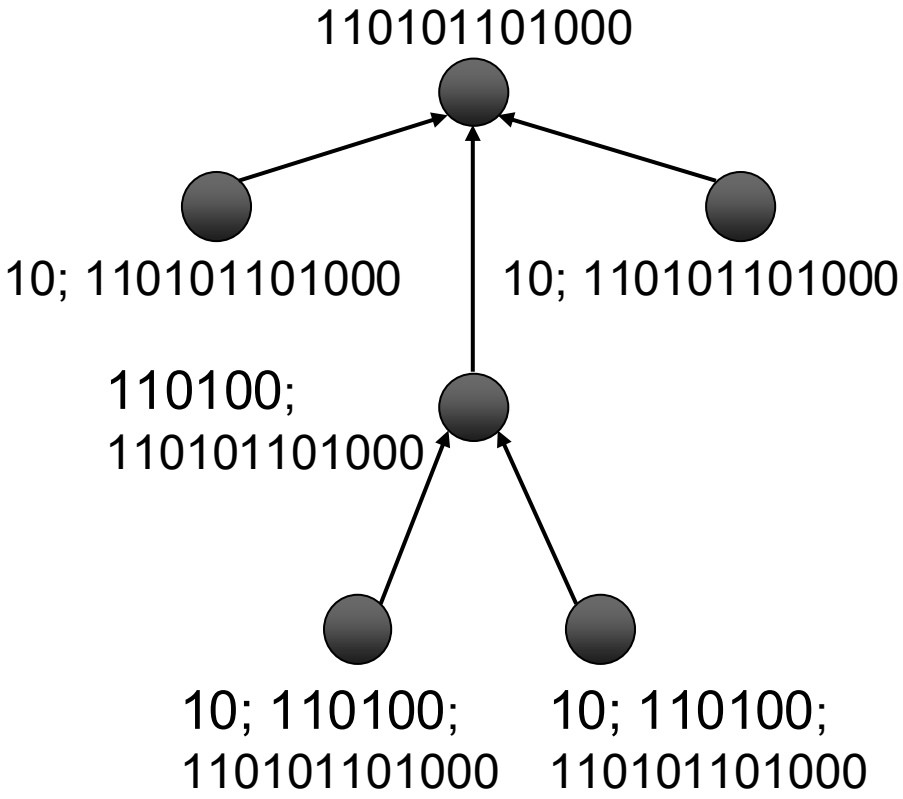
Talk Overview

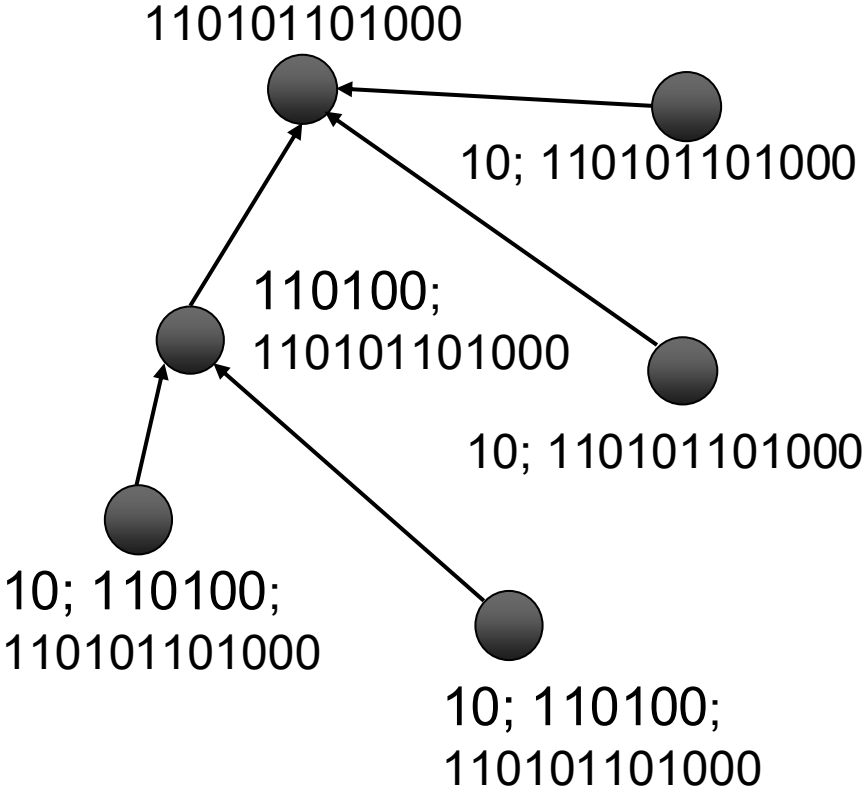
- Introduction
- **Attacking a Graph Encryption Scheme**
  - GKT Scheme and its Leakage
  - Extending a Tree Isomorphism Algorithm
  - **Query Recovery**
- Experiments

Our Solution: Path names

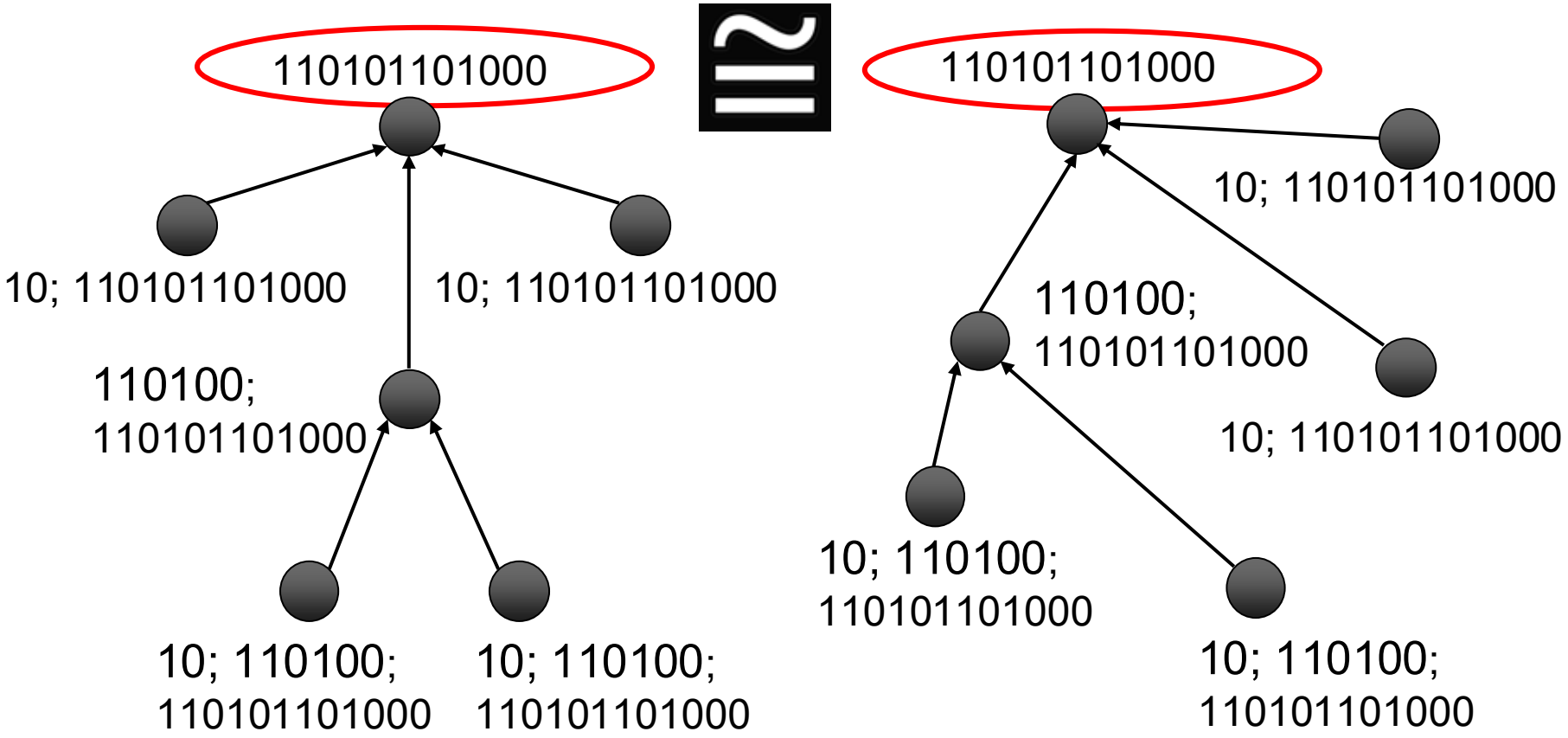


Our Solution: Path names

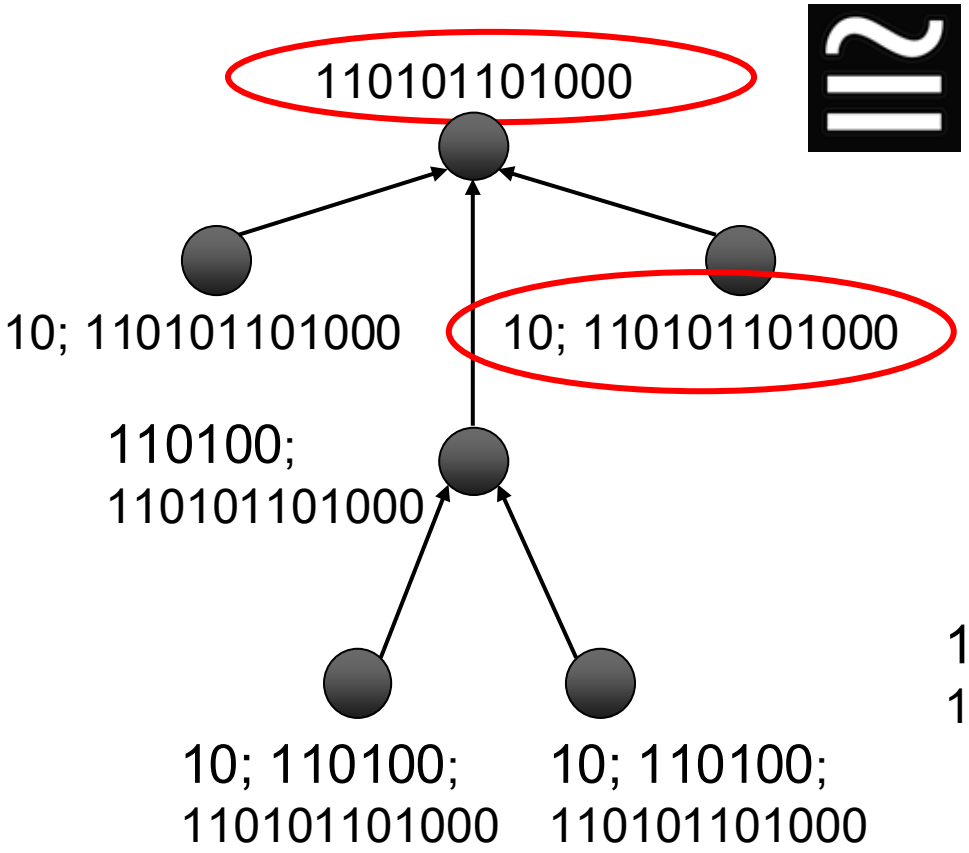


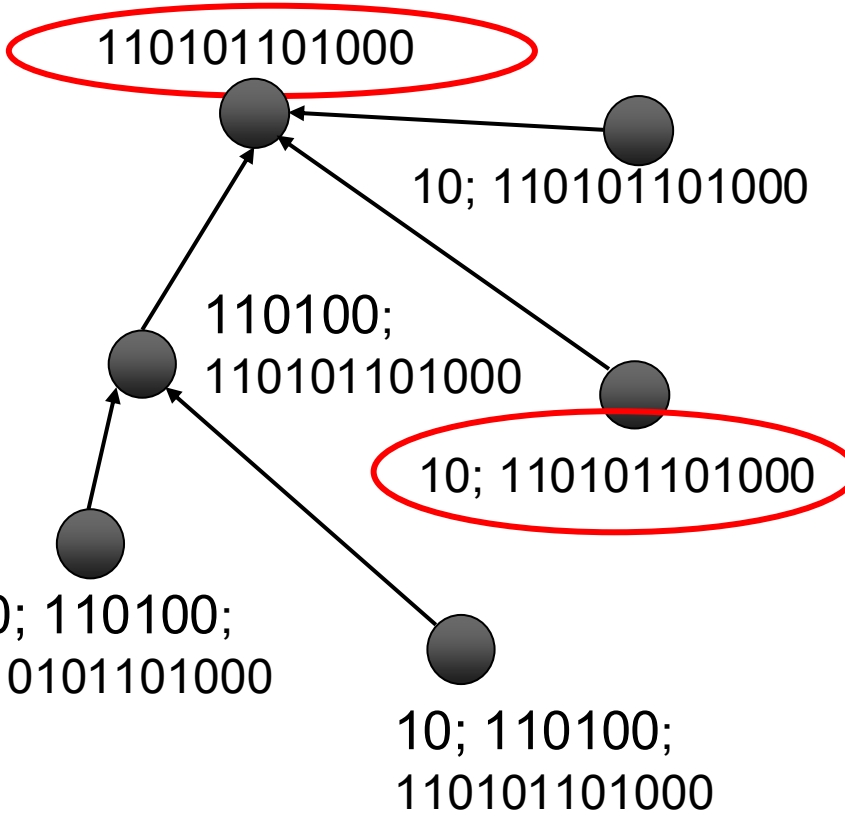


Our Solution: Path names

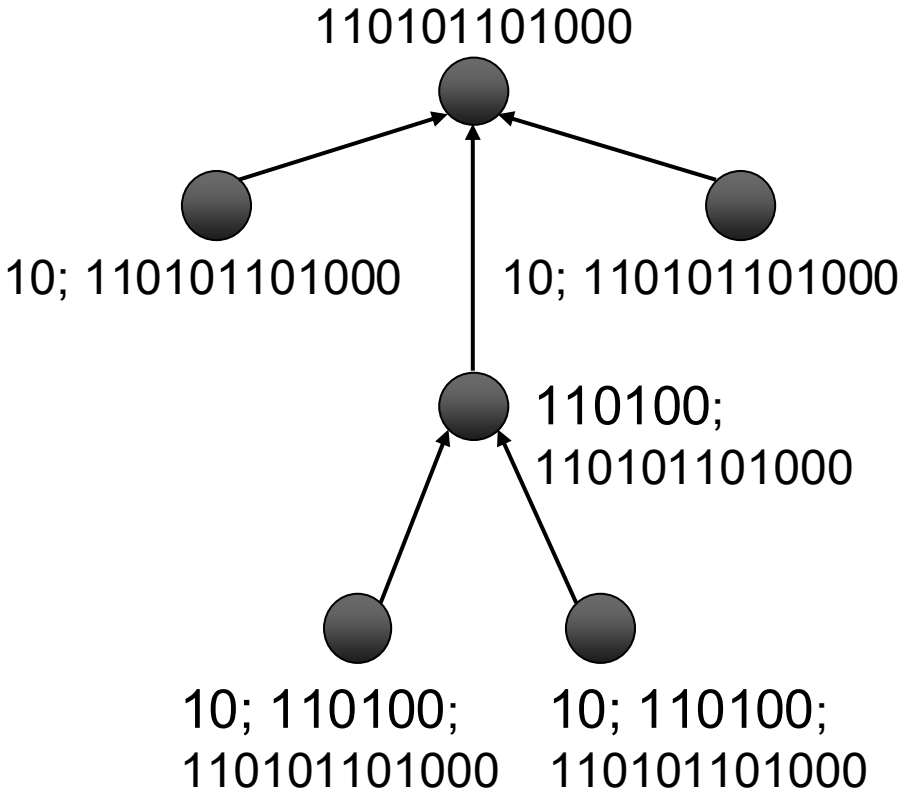


Our Solution: Path names





Our Solution: Path names



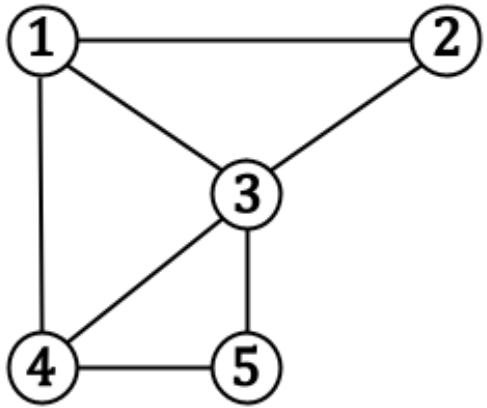
Main Theorem: Let T and T' be rooted trees and let u and v be vertices in T and T', respectively. Vertices u and v have the same path name **iff** there exists an isomorphism from T to T' mapping u to v.

Attack Overview

1. Preprocess the graph
2. Observe leakage and compute the query trees
3. Process the query trees
4. Compose the resulting maps from steps (1) and (3)

1. Preprocess the Graph

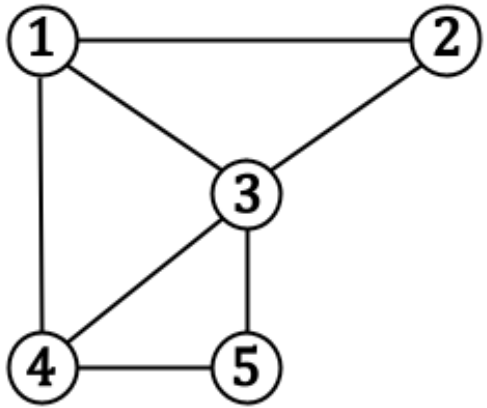
For each r in V compute the shortest path tree rooted at r.



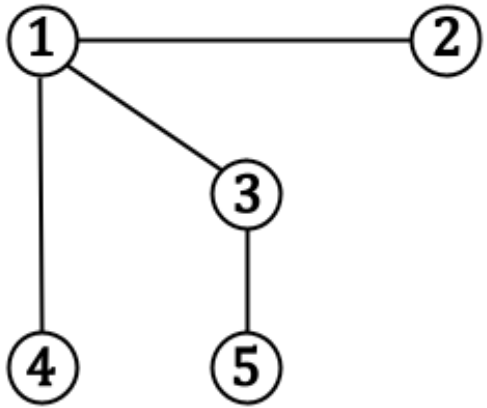
G

1. Preprocess the Graph

For each r in V compute the shortest path tree rooted at r.



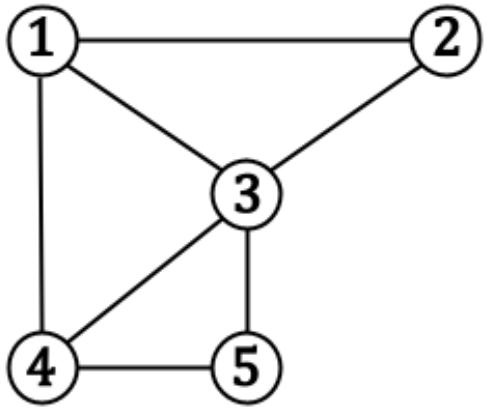
G



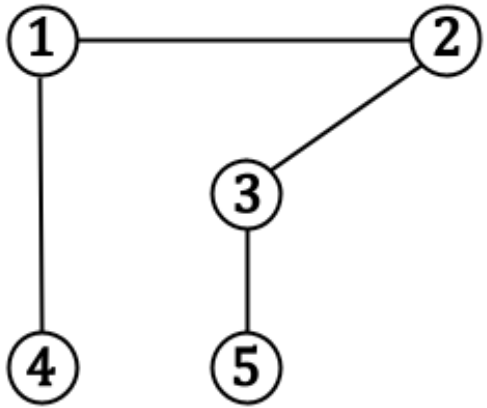
T_1

1. Preprocess the Graph

For each r in V compute the shortest path tree rooted at r.



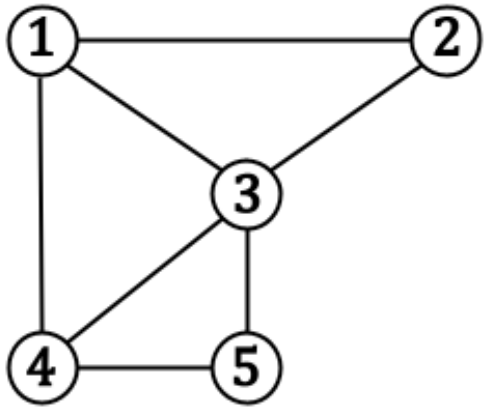
G



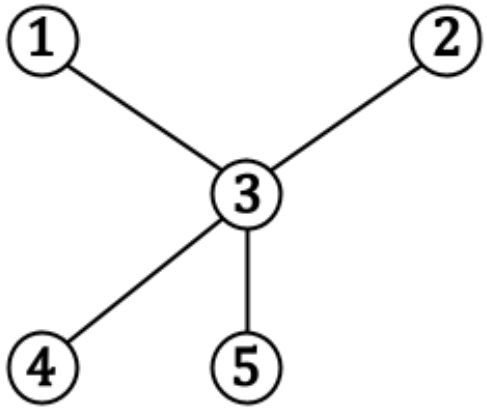
T_2

1. Preprocess the Graph

For each r in V compute the shortest path tree rooted at r.



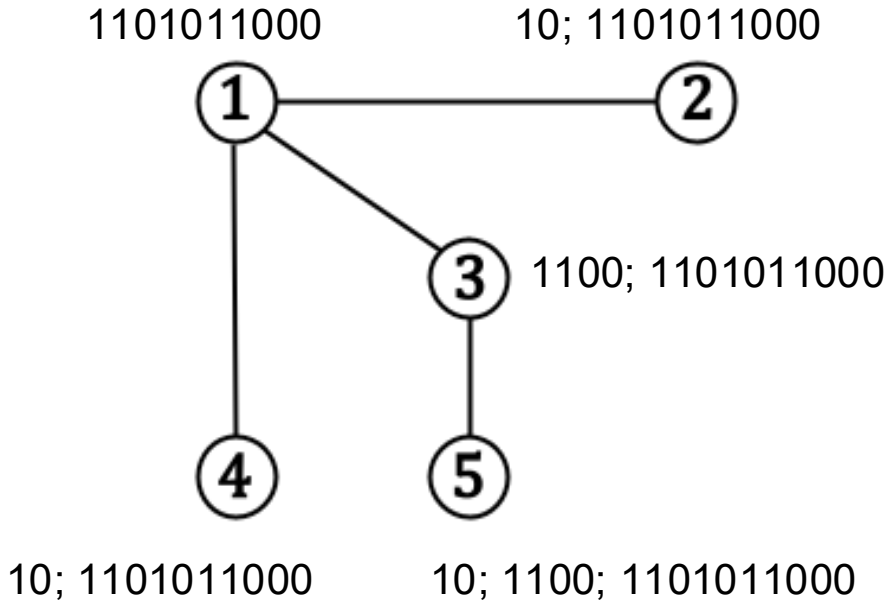
G



T_3

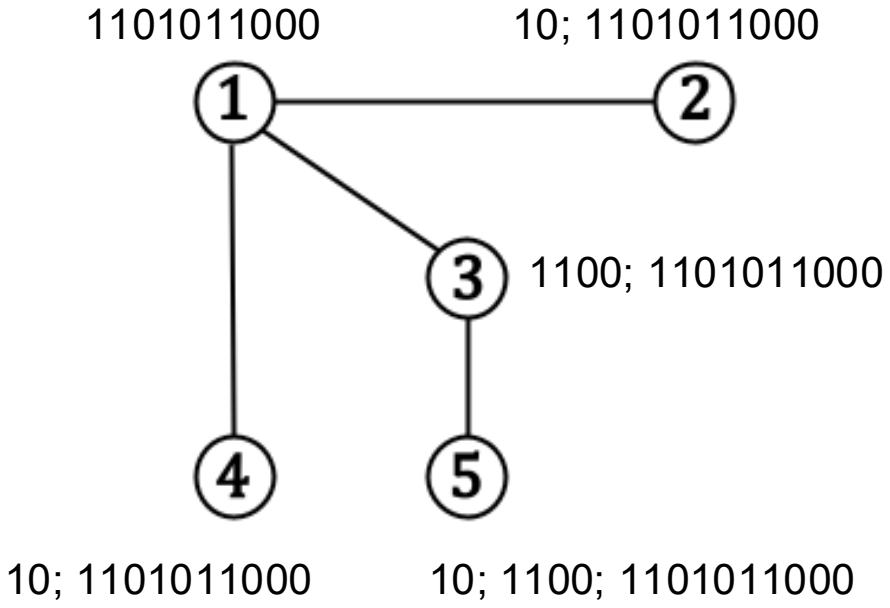
1. Preprocess the Graph

For each tree compute the (hashed) path names of all vertices.



1. Preprocess the Graph

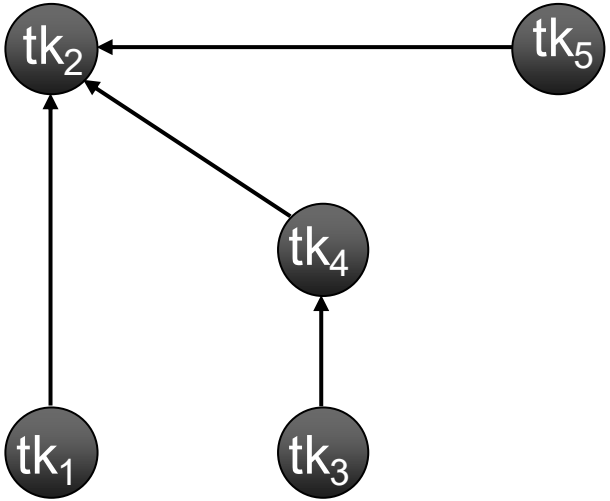
Construct map from path names to SPSP queries.



Path names	Queries
1101011000	(1, 1)
10; 1101011000	(2, 1), (4, 1)
10; 1100; 1101011000	(5, 1)
1100; 1101011000	(3, 1)

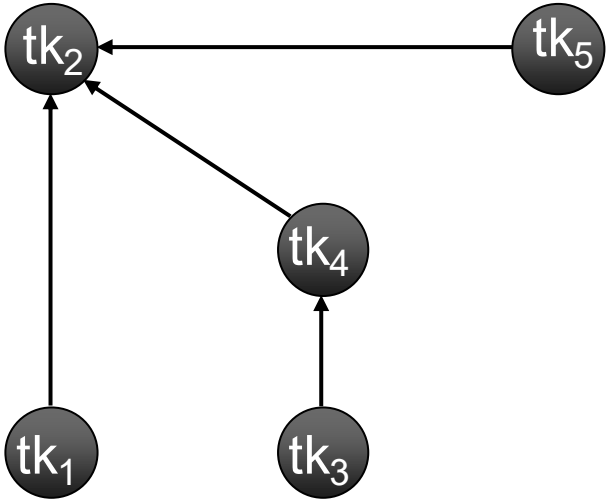
2. Compute the Query Tree

Need to observe all queries to a single destination.



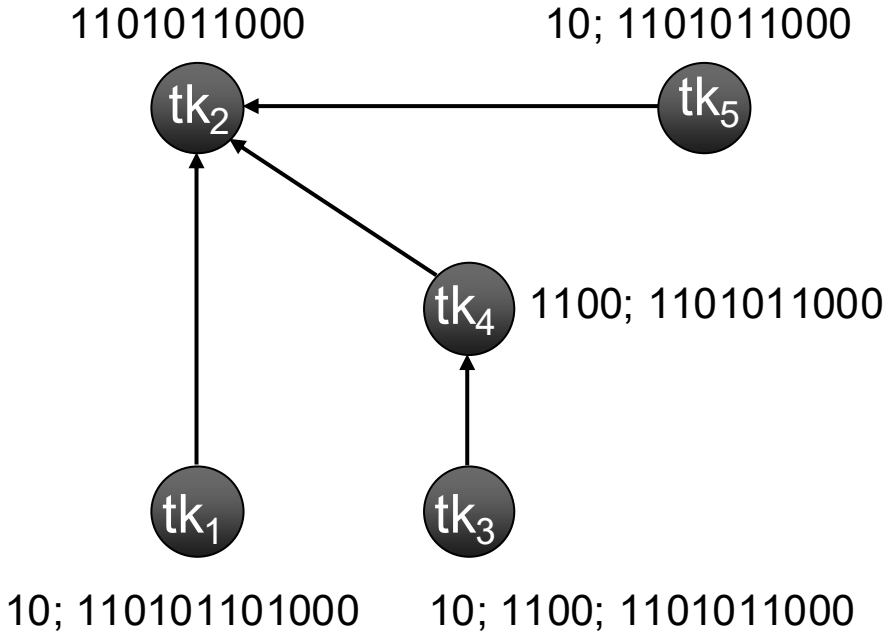
3. Process the Query Trees

For each query tree compute the (hashed) path names of all vertices.



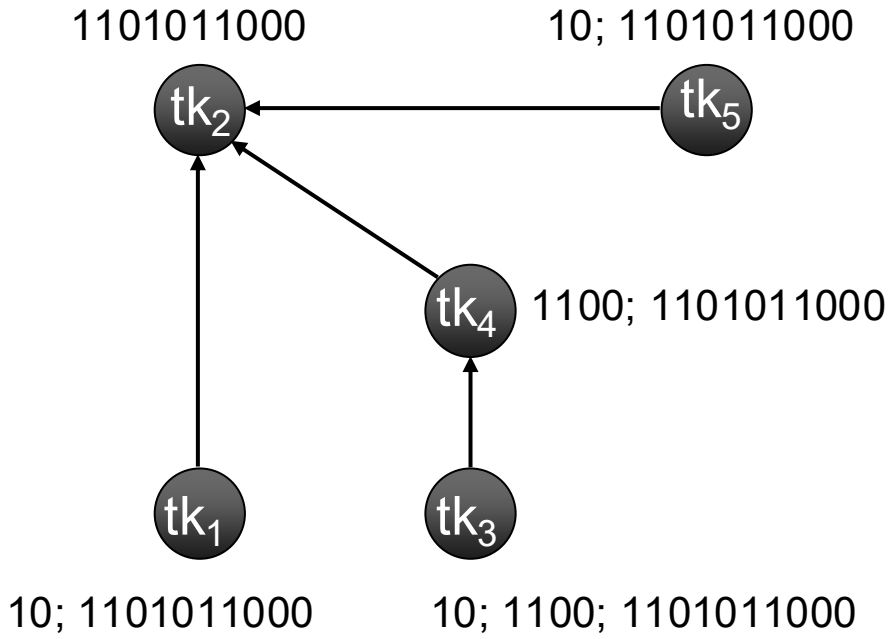
3. Process the Query Trees

For each query tree compute the (hashed) path names of all vertices.



3. Process the Query Trees

Construct map from tokens to path names.



Token	Path names
tk_2	1101011000
tk_5	10; 1101011000
tk_4	1100; 1101011000
tk_3	10; 1100; 1101011000
tk_1	10; 1101011000

4. Compose the Maps

Token	Path names
tk_2	1101011000
tk_5	10; 1101011000
tk_4	1100; 1101011000
tk_3	10; 1100; 1101011000
tk_1	10; 1101011000

Path names	Queries
1101011000	(1, 1)
10; 1101011000	(2, 1), (4, 1)
10; 1100; 1101011000	(5, 1)
1100; 1101011000	(3, 1)

4. Compose the Maps

Token	Path names
tk_2	1101011000
tk_5	10; 1101011000
tk_4	1100; 1101011000
tk_3	10; 1100; 1101011000
tk_1	10; 1101011000

Path names	Queries
1101011000	(1, 1)
10; 1101011000	(2, 1), (4, 1)
10; 1100; 1101011000	(5, 1)
1100; 1101011000	(3, 1)

4. Compose the Maps

Token	Path names
tk_2	1101011000
tk_5	10; 1101011000
tk_4	1100; 1101011000
tk_3	10; 1100; 1101011000
tk_1	10; 1101011000

Path names	Queries
1101011000	(1, 1)
10; 1101011000	(2, 1), (4, 1)
10; 1100; 1101011000	(5, 1)
1100; 1101011000	(3, 1)

4. Compose the Maps

The set of candidate queries always contains the correct query.

Token	Path names
tk_2	1101011000
tk_5	10; 1101011000
tk_4	1100; 1101011000
tk_3	10; 1100; 1101011000
tk_1	10; 1101011000

Path names	Queries
1101011000	(1, 1)
10; 1101011000	(2, 1), (4, 1)
10; 1100; 1101011000	(5, 1)
1100; 1101011000	(3, 1)

Talk Overview

- Introduction
- Attacking a Graph Encryption Scheme
  - GKT Scheme and its Leakage
  - Extending a Tree Isomorphism Algorithm
  - Query Recovery
- **Experiments**

Real World Datasets

Dataset	n	m	d	# Comp
InternetRouting	35	323	0.543	1
CA-GrQc	46	1030	0.995	1
email-Eu-core	1005	16,706	0.0331	20
facebook-combined	4039	88,234	0.011	1
p2p-Gnutella08	6301	20,777	0.001	2
p2p-Gnutella04	10,876	39,994	0.0006	1
p2p-Gnutella25	22687	54705	0.0002	13
p2p-Gnutella30	36682	88328	0.0001	12

6 of the same datasets used in the original GKT paper.

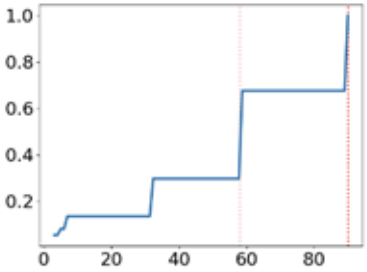
Real World Datasets

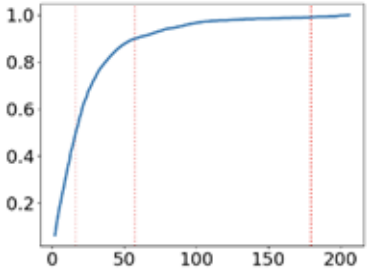
Dataset	n	m	d	% Unique	Percentile 50	90	99
InternetRouting	35	323	0.543	2.353 %	40	84	90
CA-GrQc	46	1030	0.995	0.145 %	1845	1845	1845
email-Eu-core	1005	16,706	0.0331	6.507 %	16	69	190
facebook-combined	4039	88,234	0.011	0.206 %	1826	11,424	20,480
p2p-Gnutella08	6301	20,777	0.001	21.463 %	4	12	64
p2p-Gnutella04	10,876	39,994	0.0006	21.911 %	3	9	32
p2p-Gnutella25	22687	54705	0.0002	16.075 %	5	18	54
p2p-Gnutella30	36682	88328	0.0001	14.671 %	5	24	60

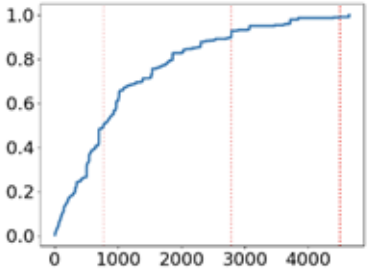
Candidates vs. % Queries

InternetRouting	Ca-GrQc	email-EU-core	facebook-combined

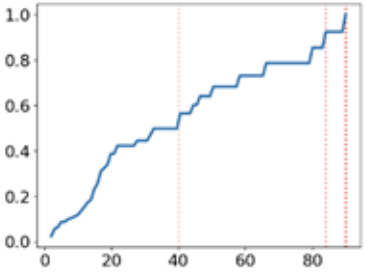
75% of queries (averaged over 10 runs)

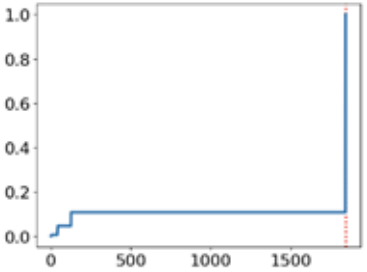


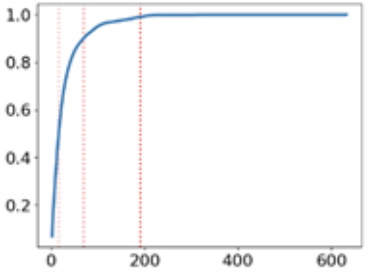


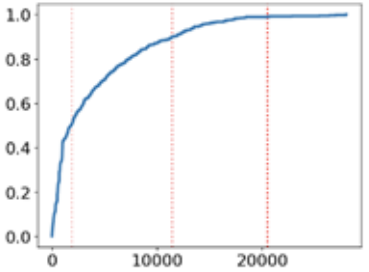


100% of queries.









Acknowledgements

Support: ETH Zürich, ThinkSwiss, and the NetApp University Research Fund