Privacy in Social Networks: Introduction

Model: Social Graph

Profiles + relationships with other users + exchange of information

Social networks model social relationships by **graph structures** using vertices and edges.

Vertices model individual social actors in a network, while edges model relationships between social actors.

Labels (type of edges, vertices)

Directed/undirected

$G = (V, E, L, L_V, L_E)$ V: set of vertices (nodes), $E \subseteq V \times V$, set of edges, L set of labels, $L_V: V \rightarrow L$, $L_{E:} E \rightarrow L$

- **Some Statistics:**
  - **MySpace -- 206,304,468 user accounts**
  - **Windows Live (MSN) Spaces -- 120,000,000 user accounts**
  - **Orkut (by Google) -- 67,962,551 user accounts**
  - **Hi5 -- 50,000,000 user accounts**
  - **Friendster -- 50,000,000 user accounts**
  - **Facebook -- 48,000,000 user accounts**
  - **LiveJournal -- 12,900,000 user accounts**

Privacy Preserving Publishing

Digital traces in a wide variety of on-line settings =>

rich sources of data for large-scale studies of social networks

Some made based on publicly crawlable blocking and social networking sites =>

users have explicitly "chosen" to publish their links to others

Focus first on domains where users have strong expectations of privacy

Privacy Preserving Publishing

Why publishing?

analysis of networks:

study disease transmission, measure the influence of a publication, evaluate the networks resiliency to faults and attacks

examples of sensitive data:

the sole publicly available of email communications was published because of goverment litigation (enron dataset)

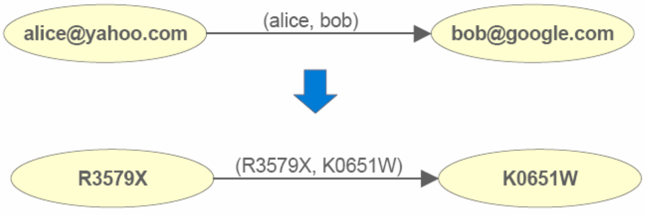
a social network which shows a set of indivical related to sexual contacts and shared drug injections, analysis about how HIV spreads

Goal: Permit useful analysis yet avoid disclosing sensistive information

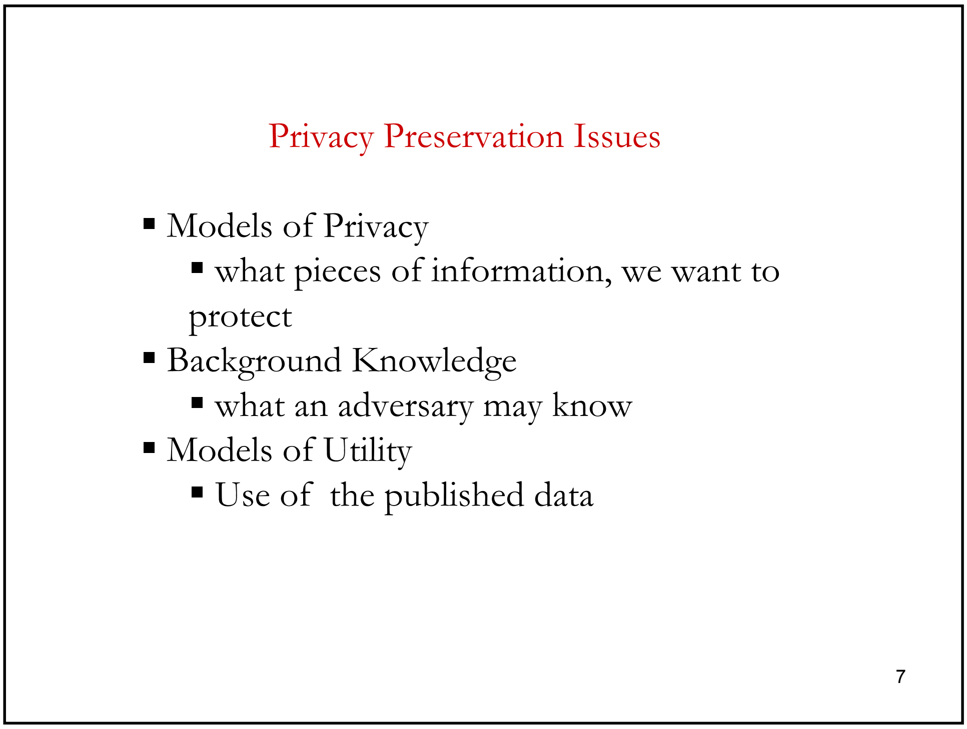
Privacy Preserving Publishing

Psedo-anonymization or Naive anonymization

replace identifying attributes with synthetic (encrypting) identifiers



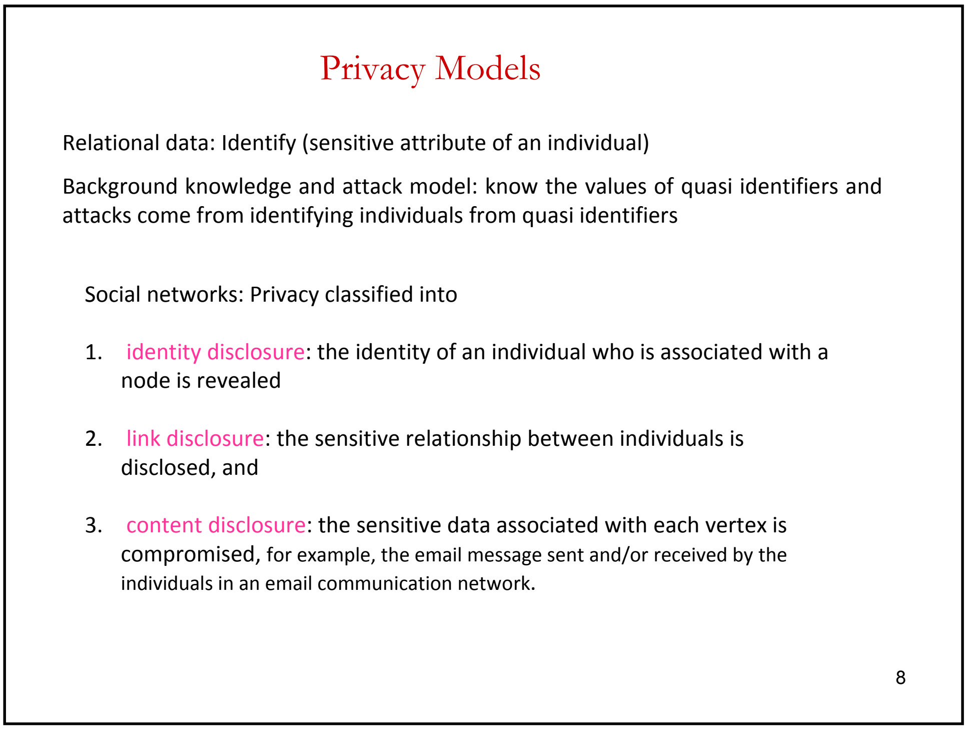
- **Is this enough? Can anonymization protect users' privacy?**



Privacy Preservation Issues

- Models of Privacy
  - what pieces of information, we want to protect
- Background Knowledge
  - what an adversary may know
- Models of Utility
  - Use of the published data

7



Privacy Models

Relational data: Identify (sensitive attribute of an individual)

Background knowledge and attack model: know the values of quasi identifiers and attacks come from identifying individuals from quasi identifiers

Social networks: Privacy classified into

1. identity disclosure: the identity of an individual who is associated with a node is revealed
2. link disclosure: the sensitive relationship between individuals is disclosed, and
3. content disclosure: the sensitive data associated with each vertex is compromised, for example, the email message sent and/or received by the individuals in an email communication network.

8

Privacy Models (more)

Examples of pieces of information:

- Vertex existence: *whether a target individual appears in the network or not*.

Examples: a social network of millionaires or a disease infection network

- Vertex properties: *such as the degree of the vertex*.

Examples: in a financial support network, how many support sources, whether the victim is a community leader can be derived.

- Sensitive vertex labels: *labels can be divided into non-sensitive vertex and sensitive similar to the case of relational data*

For example, in a disease infection network, each individual may be associated with a sensitive label disease.

Privacy Models (more)

- Link relationship: *Whether a link exists between two vertices*

For example, in a finance transaction network, there is a financial transaction

- Link weight: The weights of edges can reflect affinity between two vertices or record the communication cost between two individuals or the communication frequency between two individuals

- Sensitive edge labels:

- Graph metrics: such as *betweenness* (that is, the degree an individual lies between other individuals in the network, in their shortest path), *closeness* (that is, the degree an individual is near to all other individuals in the network directly or indirectly – the shortest distance between the node and all other nodes reachable from it), *centrality* (that is, the count of the number of relationships to other individuals in the network), *path length* (that is, the distances between pairs of vertices in the network), *reachability* (that is, the degree any member of a network can reach other members of the network)

Privacy Models (model)

- Link relationship: *Whether a link exists between two vertices*

For example, in a finance transaction network, there is a financial transaction

- Link weight: The weights of edges can reflect affinity between two vertices or record the communication cost between two individuals or the communication frequency between two individuals

- Sensitive edge labels:

- Graph metrics: such as *betweenness* (that is, the degree an individual lies between other individuals in the network, in their shortest path), *closeness* (that is, the degree an individual is near to all other individuals in the network directly or indirectly – the shortest distance between the node and all other nodes reachable from it), *centrality* (that is, the count of the number of relationships to other individuals in the network), *path length* (that is, the distances between pairs of vertices in the network), *reachability* (that is, the degree any member of a network can reach other members of the network)

Models of Background Knowledge

Relational data: the values of the quasi identifiers

In general: the network structure around it

- Attributes of vertices
- Vertex degree
- Link relationship
- Neighborhood
- Graph metrics

Models of Background Knowledge

Type of attacks:

active: an adversary tries to compromise privacy by strategically creating new user accounts and links before the anonymized network is released, so that these new nodes and edges will then be present in the anonymized network.

passive: try to learn the identities of nodes only after the anonymized network has been released

Utility Models

Or information loss, Or anonymization quality

What type of analysis – how the anonymized network will be used

Relational: measured using the sum of information loss in individual tuples (distance of tuple in the original table from the anonymized tuple in the released table)

- General graph properties (diameter, distribution of vertex degrees, etc)
- Aggregate network queries: compute the aggregate on some path or subgraph that satisfies some given condition

Example: the average distance from a medical doctor vertex to a teacher vertex

Useful in many applications, such as customer relationship management

Anonymization Methods

Relational data: generalization, noise (e.g., perturbations etc), anatomy

Initial Graph G -> Anonymized Graph G*

Pseudo-anonymization

Anonymization Methods

- Clustering-based approaches: clusters vertices and edges into groups and replaces a subgraph with a super-vertex (Generalization)
- Graph Modification approaches: modifies (inserts or deletes) edges and vertices in the graph (Perturbations)

Some Graph-Related Definitions

- A subgraph H of a graph G is said to be induced if, for any pair of vertices x and y of H, (x, y) is an edge of H if and only if (x, y) is an edge of G.

In other words, H is an induced subgraph of G if it has exactly the edges that appear in G over the same vertex set.

- If the vertex set of H is the subset S of V(G), then H can be written as G[S] and is said to be induced by S.

- Neighborhood

Mappings that preserve the graph structure

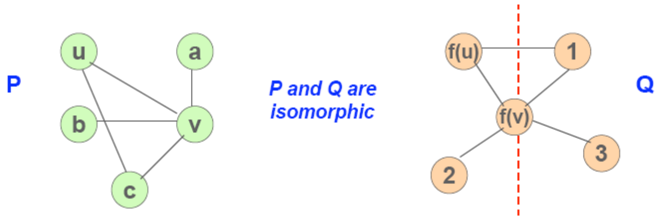
A **graph homomorphism** *f* from a graph *G* = (*V*, *E*) to a graph *G*' = (*V*', *E*'), is a mapping f: G $\rightarrow$ G', from the vertex set of G to the vertex set of G' such that (u, u') $\in$ G $\Rightarrow$ (f(u), f(u')) $\in$ G'

If the homomorphism is a bijection whose inverse function is also a graph homomorphism, then *f* is a graph **isomorphism** [(u, u') $\in$ G $\Leftrightarrow$ (f(u), f(u')) $\in$ G']

A graph **automorphism** is a graph isomorphism with itself, i.e, a mapping from the vertices of the given graph G back to vertices of G such that the resulting graph is isomorphic with G. An automorphism f is **non-trivial** if it is not identity function.

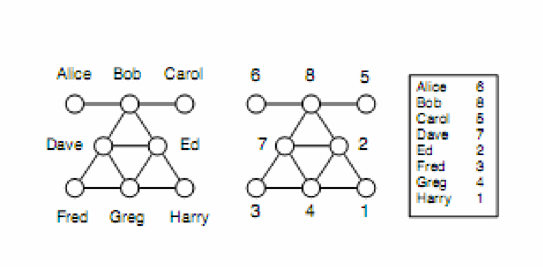
A **bijection**, or a bijective function, is a function *f* from a set X to a set Y with the property that, for every y in Y, there is exactly one x in X such that f(x) = y.

Alternatively, f is bijective if it is a one-to-one correspondence between those sets; i.e., both **one-to-one** (injective) and **onto** (surjective)).



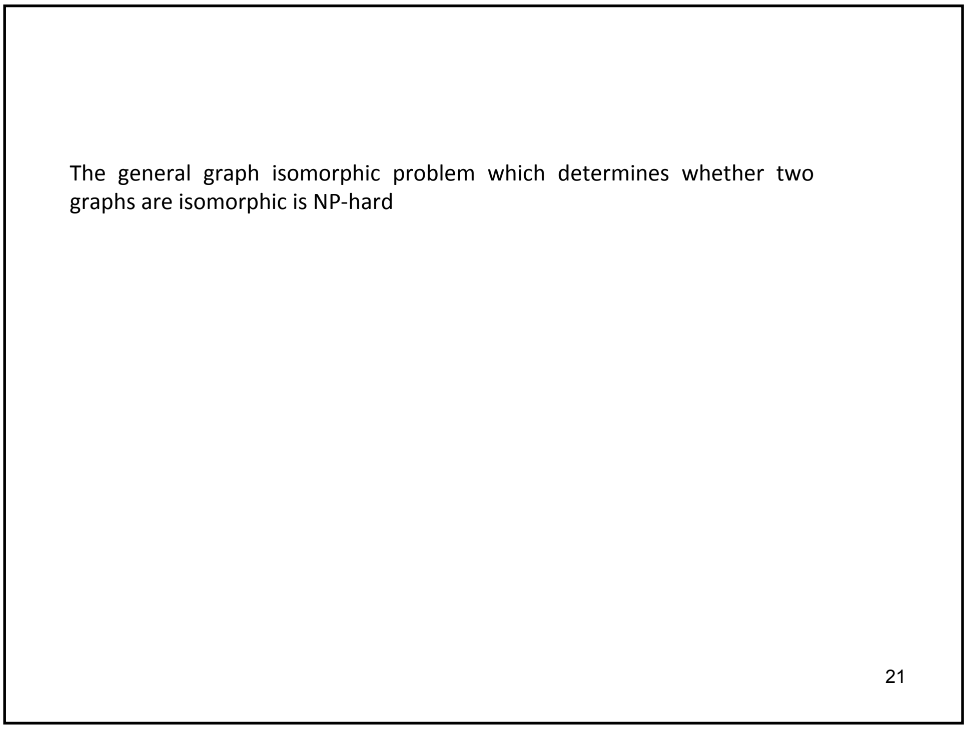
P and Q are isomorphic

the naive anonymization is an isomorphic graph

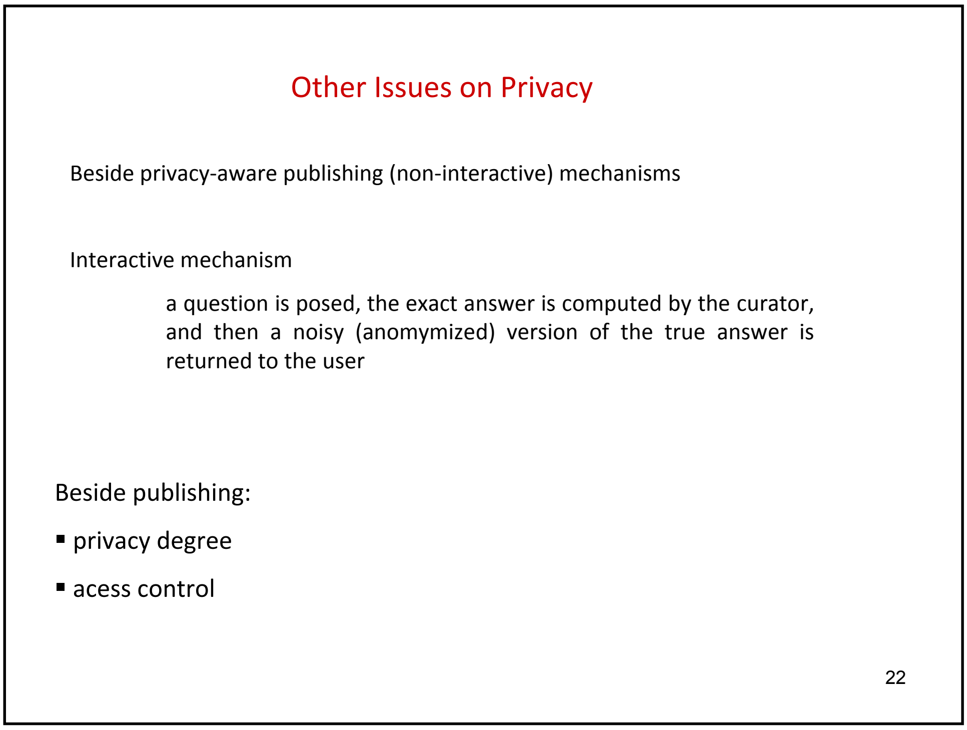


Alice	6
Bob	8
Carol	5
Dave	7
Ed	2
Fred	3
Greg	4
Harry	1

the anonymization mapping f is a random secret mapping



The general graph isomorphic problem which determines whether two graphs are isomorphic is NP-hard



Other Issues on Privacy

Beside privacy-aware publishing (non-interactive) mechanisms

Interactive mechanism

a question is posed, the exact answer is computed by the curator, and then a noisy (anomymized) version of the true answer is returned to the user

Beside publishing:

- privacy degree
- acess control

Type of Attacks

Active and Passive Attacks

Lars Backstrom, Cynthia Dwork and Jon Kleinberg, ***Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography***
Proceedings of the 16th international conference on World Wide Web, 2007 (WWW07)

Model

Purest form of social network:

Nodes corresponding to individuals

Edges indicating social interactions

(no labels, no directions, no annotations)

Simple Anonymization: Actual names are removed

Utility preserved: properties of the graph (such as connectivity, node-to-node distnaces, frequencies of small subgraphs, or the extend to which it can be clustered)

Can this work?

Privacy threat: De-anonymize 2 nodes and learn if connected

Focus of the paper:

Identify type of attacks that even from a single anonymized copy of a social network, it is possible for an adversary to learn whether edges exist or not between specific targeted pair of nodes

Note that the adversary may be a user of the system being anomymized

Passive Attacks

An adversary tries to learn the identities of the nodes only **after** the anonymized network has been released

Simply try to find themselves in the released network and from this to discover the existence of edges among users to whom they are linked

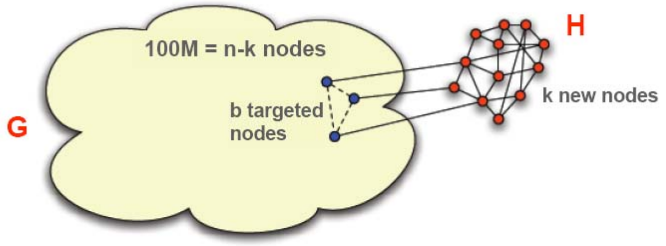
Based on the observation that most nodes in real social networks already belong to a small subgraph, thus if a user can collude with a coalition of k-1 friends after the release, he/she is able to identify additional nodes that are connected to this coalition

Active Attacks

An adversary tries to compromise privacy by strategically creating new user accounts and links **before** the anonymized network is released

Chooses an arbitrary set of users whose privacy it wishes to violate, creates a small number of new user accounts with edges to those targeted users and creates patterns of links amongst the new accounts to make it identifiable in the anomymized graph structure

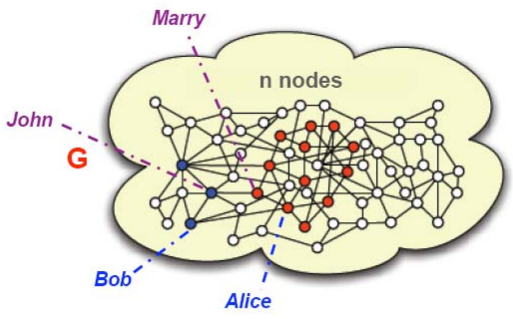
- **Active work in with high probability in any network – passive rely on the chance that a use can uniquely find themselves after the network is released**
- **Passive attacks can compromise the privacy of users liked to the attacker**



Before releasing the anonymized network G of n-k nodes, attacker:

- **Choose a set of b targeted users.**
- **Create a subgraph H containing k nodes.**
- **Attach H to the targeted nodes.**

Creating the subgraph H --> *structural steganography*

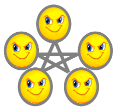


After the anonymized network is release:

- Find the subgraph H in the graph G
- Follow edges from H to locate b target nodes and their true location in G
- Determine all edges among these b nodes --> breach privacy

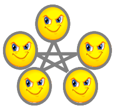
Active Attacks - Challenges

Let *G* be the network, *H* the subgraph
With high probability, *H* must be:



- Uniquely identifiable in *G*
  - For any *G*
- Efficiently locatable
  - Tractable instance of subgraph isomorphism
- But undetectable
  - From the point of view of the *data curator*

Active Attacks - Approaches



- Basic idea: *H* is *randomly generated*
  - Start with *k* nodes, add edges independently at random
- Two variants:
  - $k = \Theta(\log n)$ de-anonymizes $\Theta(\log^2 n)$ users
  - $k = \Theta(\sqrt{\log n})$ de-anonymizes $\Theta(\sqrt{\log n})$ users
    - H needs to be “more unique”
    - Achieved by “thin” attachment of H to G

The “Walk-based” attack – better in practice

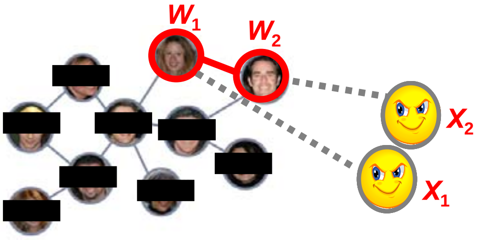
The “Cut-based” attack – matches theoretical bound

Outline

- Attacks on anonymized networks – high level description
- **The Walk-Based active attack**
  - Description
  - Analysis
  - Experiments
- Passive attack

The Walk-Based Attack – Simplified Version

- Construction:
  - Pick target users $W = \{w_1,\ldots,w_k\}$
  - Create new users $X = \{x_1,\ldots,x_k\}$ and random subgraph ***G*[*X*] = *H***
  - Add edges (x_i, w_i)



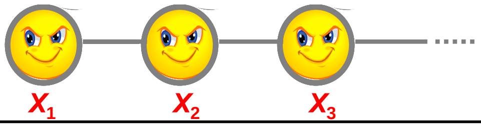
[redacted]

- Recovery
  - Find *H* in *G* ↔ No subgraph of G isomorphic to H
  - Label *H* as $x_1,\ldots,x_k$ ↔ No automorphisms
  - Find $w_1,\ldots,w_k$

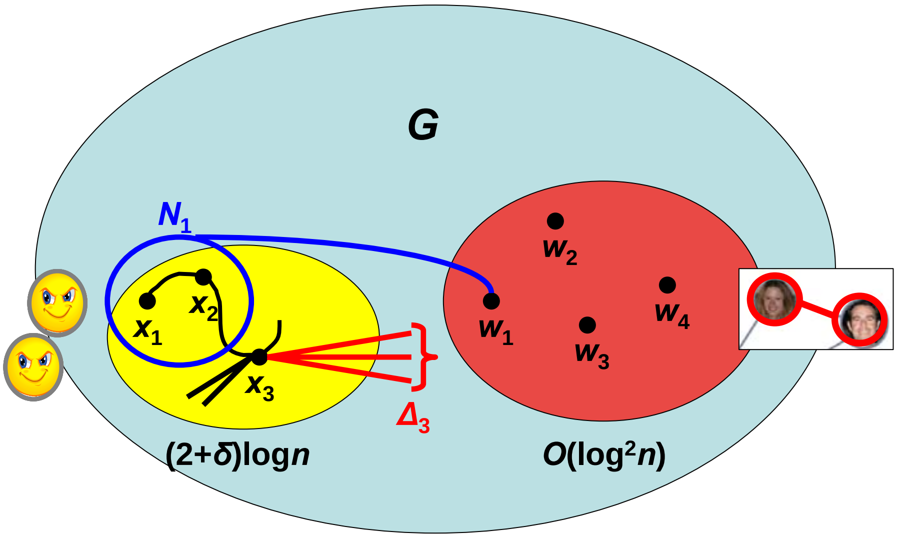
The Walk-Based Attack – Full Version

- Construction:
  - Pick target users $W = \{w_1,\ldots,w_b\}$
  - Create new users $X = \{x_1,\ldots,x_k\}$ and *H*
  - Connect w_i to a *unique subset* N_i of *X*
  - Between *H* and *G – H*
    - Add Δ_i edges from x_i where $d_0 \le \Delta_i \le d_1 = O(\log n)$
  - Inside *H*, add edges (x_i, x_{i+1})

To help find *H*

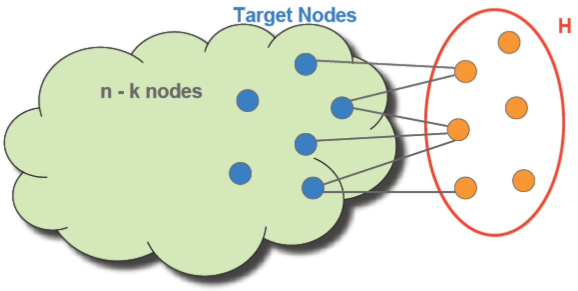


Construction of *H*

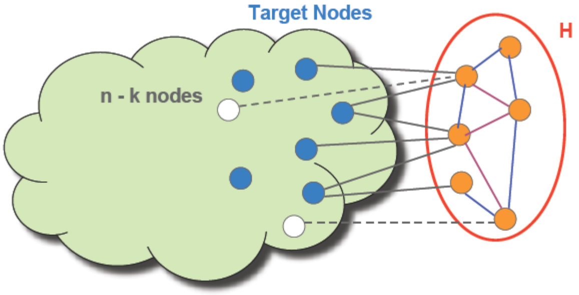


- Total degree of x_i is Δ'_i

- H = set of nodes X size k = (2+δ) log n (δ > 0)
- W = set of targeted users size b = O(log² n)
  - e.g. n = 1000M, b = 900, k ≈ 30
- External degree for node x_i: $\Delta_i \in [d_0, d_1]$ for $d_0 \leq d_1 = O(\log n)$
- Each w_j connects to a set of nodes $N_j \subseteq X$. Set
- N_j must be of size at most c=3 and are distinct across all nodes w_j.



- Add arbitrary edges from H to G-H to make it Δ_i for all x_i.
- Add internal edges in H: edge (x_i, x_{i+1})
- Add additional internal edges connecting (x_i, x_j) with probability 0.5
- Therefore, each node x_i has total degrees of $\Delta'_i = \Delta_i$ + (#internal edges)

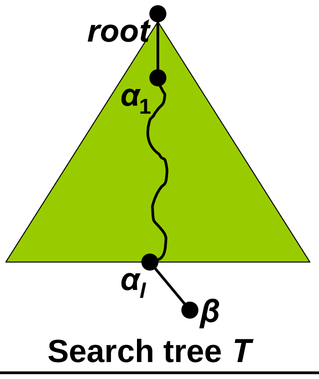


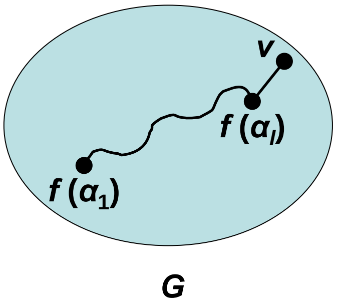
11

37

Recovering *H*

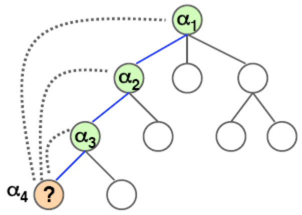
- Search G based on:
  - Degrees Δ'_i
  - Internal structure of *H*





38

- **Degree Test:** Node x_i has total degrees of $\Delta'_i = \Delta_i$ + (#internal edges)
- **Internal Structure Test:** Node x_i links to correct subset of $\{x_1, x_2, \ldots, x_{i-1}\}$
- **Search tree T:** All nodes α_i in T has corresponding node $f(\alpha_i)$ in G.
- Every path of nodes $\alpha_1, \alpha_2, \ldots, \alpha_j$ from the root must have corresponding path in G formed by nodes $f(\alpha_1), f(\alpha_2), \ldots, f(\alpha_j)$ with the same degree sequence $x_1, x_2, \ldots, x_j$.
- The probability of a false path surviving to depth $l \approx 2^{-l^2/2}$



Analysis

- Theorem 1 [*Correctness*]:
  With high probability, *H* is unique in *G*. Formally:
  - *H* is a **random** subgraph
  - *G* is **arbitrary**
  - Edges between H and G – H are **arbitrary**
  - There are edges (x_i, x_{i+1})
  - ➢ Then WHP no subgraph of *G* is isomorphic to *H*.

- Theorem 2 [*Efficiency*]:
  Search tree *T* does not grow too large. Formally:
  - For every ε, WHP the size of *T* is $O(n^{1+\varepsilon})$

Theorem 1 [*Correctness*]

- *H* is unique in *G.* Two cases:
  - For no *disjoint* subset *S*, *G*[*S*] isomorphic to *H*
  - For no *overlapping* *S*, *G*[*S*] isomorphic to *H*

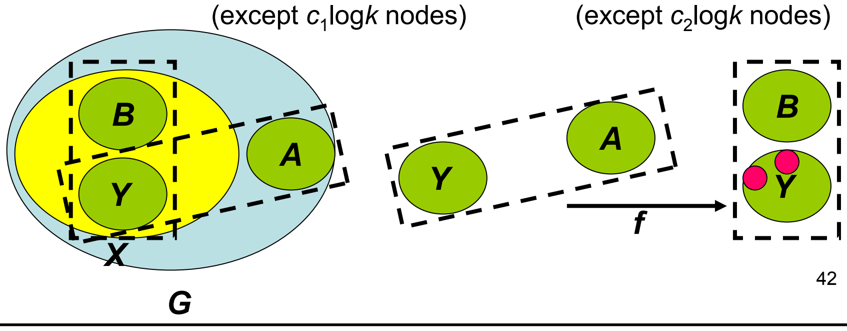
- **Case 1:**
  - $S = <s_1,\ldots,s_k>$ nodes in $G - H$
  - ε_S – the event that $s_i \leftrightarrow x_i$ is an isomorphism
  - $\Pr[\varepsilon_S] \le 2^{k-1}/2^{k(k-1)/2}$
  - By Union Bound,

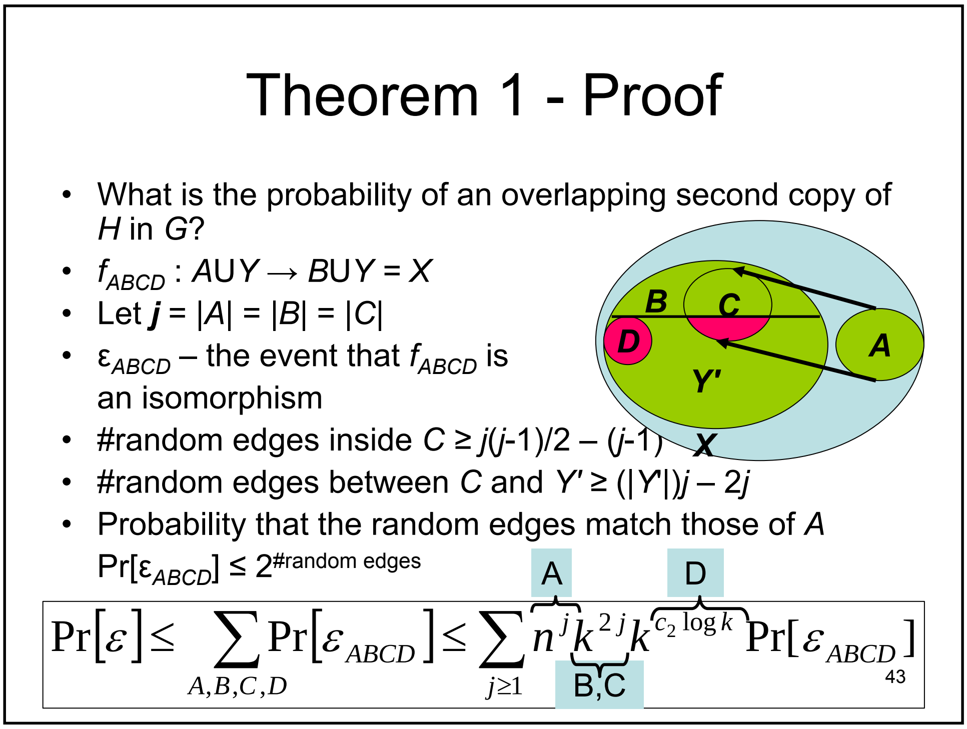
$$\boxed{\Pr[\bigcup_S \varepsilon_S] < n^k \cdot 2^{k-1}/2^{k(k-1)/2}}$$

Theorem 1 continued

- **Case 2:** S and X overlap. Observation – *H* does no have much internal symmetry
- **Claim (a):** WHP, there are no *disjoint* isomorphic subgraphs of size $c_1$log*k* in *H*. Assume this from now on.
- **Claim (b):** Most of A goes to B, most of Y is fixed under *f*

(except $c_1$log*k* nodes) (except $c_2$log*k* nodes)

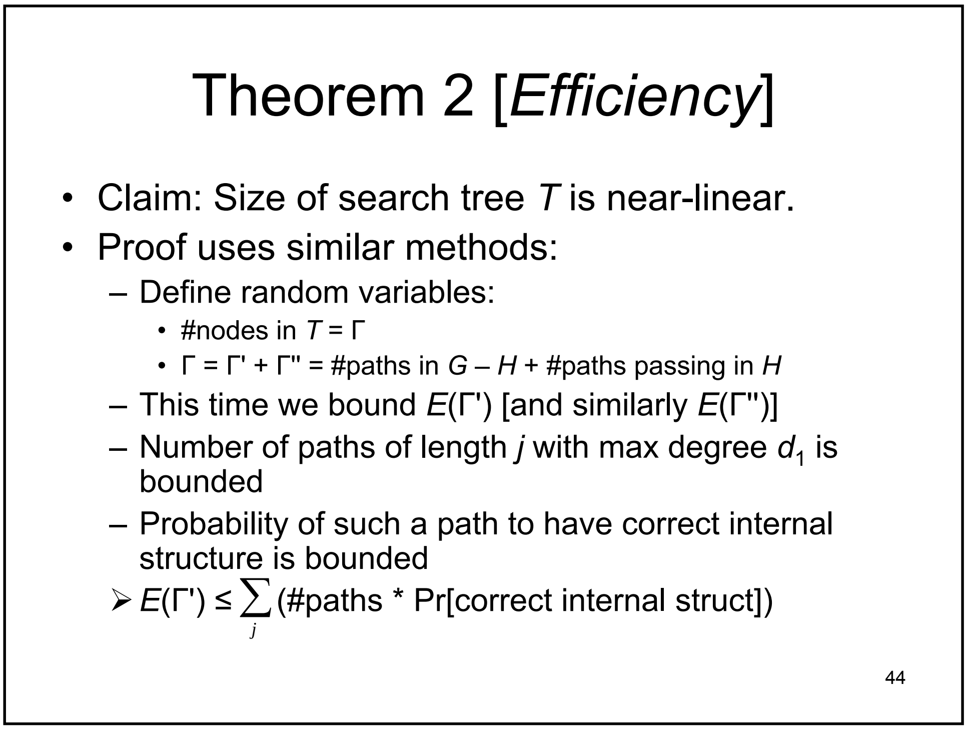




Theorem 1 - Proof

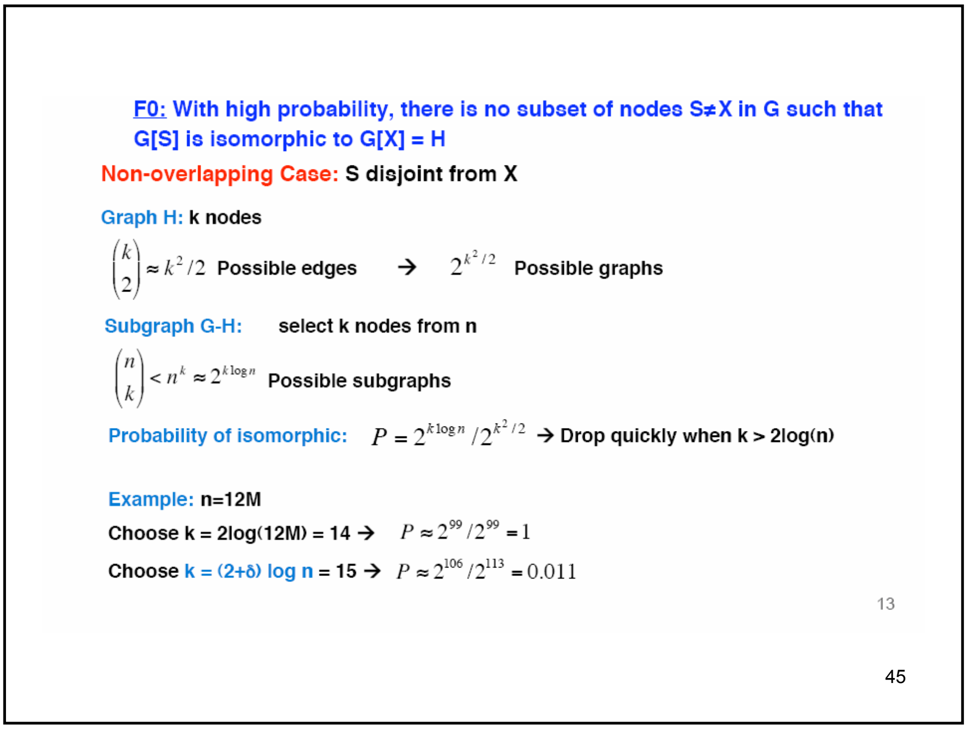
- What is the probability of an overlapping second copy of *H* in *G*?
- $f_{ABCD} : A \cup Y \rightarrow B \cup Y = X$
- Let $\boldsymbol{j} = |A| = |B| = |C|$
- ε_{ABCD} – the event that f_{ABCD} is an isomorphism
- #random edges inside $C \geq j(j-1)/2 - (j-1)$
- #random edges between C and $Y' \geq (|Y'|)j - 2j$
- Probability that the random edges match those of *A* $\Pr[\varepsilon_{ABCD}] \leq 2^{\text{\#random edges}}$

$$\Pr[\varepsilon] \leq \sum_{A,B,C,D} \Pr[\varepsilon_{ABCD}] \leq \sum_{j\geq 1} \overbrace{n^j}^{A} \underbrace{k^{2j}}_{B,C} \overbrace{k^{c_2 \log k}}^{D} \Pr[\varepsilon_{ABCD}]$$



Theorem 2 [*Efficiency*]

- Claim: Size of search tree *T* is near-linear.
- Proof uses similar methods:
  - Define random variables:
    - #nodes in $T = \Gamma$
    - $\Gamma = \Gamma' + \Gamma''$ = #paths in $G - H$ + #paths passing in H
  - This time we bound $E(\Gamma')$ [and similarly $E(\Gamma'')$]
  - Number of paths of length *j* with max degree d_1 is bounded
  - Probability of such a path to have correct internal structure is bounded
  - ➢ $E(\Gamma') \leq \sum_j$(#paths * Pr[correct internal struct])



F0: With high probability, there is no subset of nodes S≠X in G such that G[S] is isomorphic to G[X] = H

Non-overlapping Case: S disjoint from X

Graph H: k nodes

$\binom{k}{2} \approx k^2/2$ Possible edges → $2^{k^2/2}$ Possible graphs

Subgraph G-H: select k nodes from n

$\binom{n}{k} < n^k \approx 2^{k\log n}$ Possible subgraphs

Probability of isomorphic: $P = 2^{k\log n}/2^{k^2/2}$ → Drop quickly when k > 2log(n)

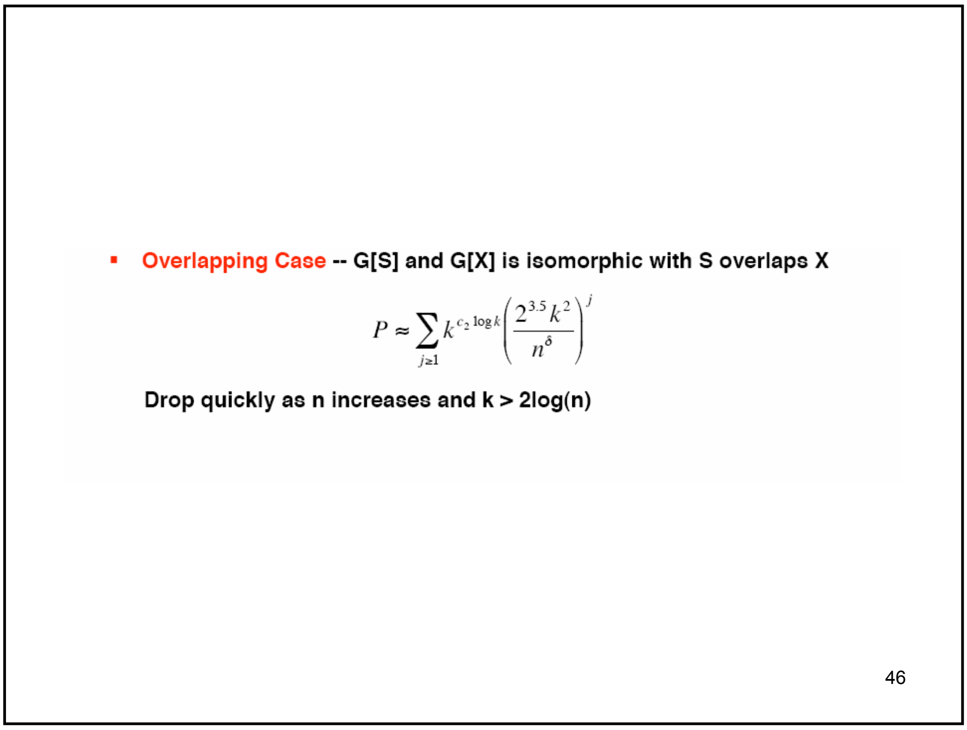
Example: n=12M

Choose k = 2log(12M) = 14 → $P \approx 2^{99}/2^{99} = 1$

Choose k = (2+δ) log n = 15 → $P \approx 2^{106}/2^{113} = 0.011$

13

45



- Overlapping Case -- G[S] and G[X] is isomorphic with S overlaps X

$$P \approx \sum_{j\geq 1} k^{c_2 \log k}\left(\frac{2^{3.5}k^2}{n^{\delta}}\right)^j$$

Drop quickly as n increases and k > 2log(n)

46

F1: For c_1>4, there is no disjoint sets of nodes Y and Z in H, each of size c1log(k), such that H[Y] and H[Z] are isomorphic

- Scope down what we had from F0:
  - Graph G size n ⇒ Subgraph H size k
  - Sets of nodes size (2+δ) log n ⇒ Sets of nodes size (c_1>4)log(k)

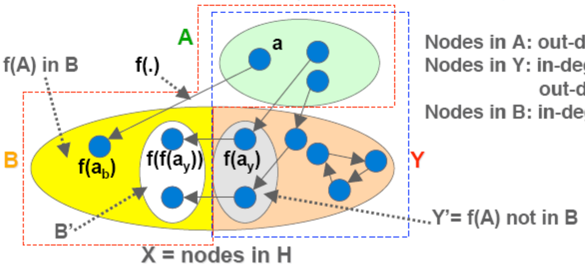
Fixed Point of a Isomorphism

For Isomorphism mapping S to S' (f: S → S')

- A fixed point is in both S and S'
- A fixed point maps to itself

15

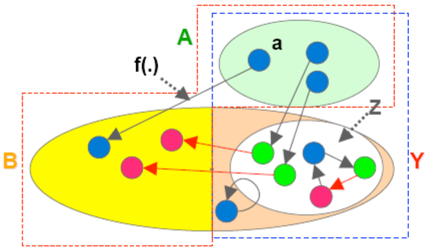
Claim3.1: Let A, B and Y are disjoint sets of nodes in G with B,Y ⊆ X. With isomorphism f: A∪Y → B∪Y, |{ f(A) not in B }| ≤ $c_1$log(k) nodes.



Nodes in A: out-deg =1 (to B or Y)
Nodes in Y: in-deg=1 (from Y or itself)
out-deg=1 (to B or itself)
Nodes in B: in-deg=1 (from A or Y)

- Consider path from A→Y→B: |Y'| and |B'| are disjoint and H[B'] and H[Y'] are isomorphic. ➔ |Y'| ≤ c1log(k) ➔ # paths A→Y→B ≤ c1log(k)
- Y' = f(A) not in B ➔ |f(A) not in B| ≤ c1log(k)

Claim3.2: Let A, B and Y are disjoint sets of nodes in G with $B, Y \subseteq X$. With isomorphism $f: A \cup Y \rightarrow B \cup Y$, set Y has at most $c2\log(k)$ nodes that are not fixed point of f, where $c2 \ge 3c1$



Choose every other edge in the path or cycle. In cycle, choose 1 edges from 3 z-nodes. → Worst case $|Z|/3$ edges

Z1 = Nodes on tail of selected edges
Z2 = Nodes on head of selected edges

- Z1 and Z2 are disjoint subset of X; and G[Z1] and G[Z2] are isomorphic
- ➔ From F1, $|Z1| = |Z2| \le c1\log(k)$
- $|Z1| = |Z2| =$ #selected edges $= |Z|/3$ ➔ $|Z|/3 \le c1\log(k)$
- $|Z| \le 3c1\log(k) \le c2\log(k)$

Experiments

- Data: Network of friends on *LiveJournal*
  - $4.4 \cdot 10^6$ nodes, $77 \cdot 10^6$ edges

Welcome to LiveJournal

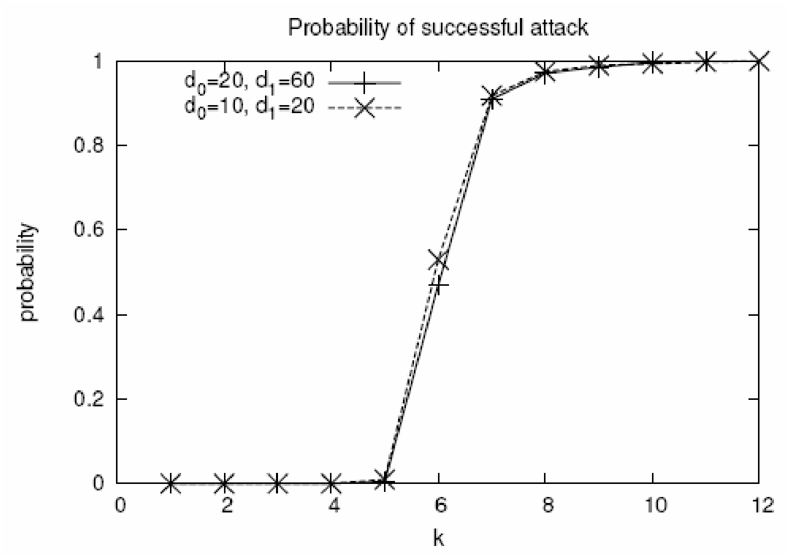
LiveJournal lets you express yourself, share your life, and connect with friends online.

You can use LiveJournal in many different ways: as a private journal, a blog, a discussion forum, a social network, and more.

Joining LiveJournal is completely free.

- Uniqueness: With 7 nodes, an average of 70 nodes can be de-anonymized
  - Although $\log(4.4 \cdot 10^6) \approx 15$
- Efficiency: $|T|$ is typically $\sim 9 \cdot 10^4$
- Detectability:
  - Only 7 nodes
  - Many subgraphs of 7 nodes in G are dense and well-connected

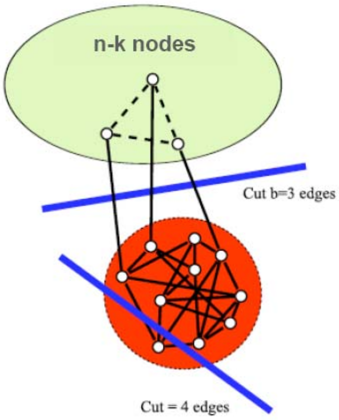
Probability that *H* is Unique



- **Theoretical asymptotic lower bound for #new nodes: $\Omega(\sqrt{\log n})$**
- **Randomly generate subgraph H = (x_1, x_2, …, x_k) with $k = O(\sqrt{\log n})$**
- **Number of compromised nodes $b = \Theta(\sqrt{\log n})$**

Construction of H

- **For W=(w_1, w_2,…, w_b) b targeted users, create X= (x_1, x_2,…, x_k) where k = 3b+3 nodes**
- **Create links between each pair (x_i, x_j) with probability = 0.5**
- **Choose arbitrary b nodes (x_1, x_2,…, x_b); connect x_i to w_i**



- d(H) = min degree in H
- c(H) = min internal cut in H

Properties:

With high probability

- b = size of cut between H and G-H
- c(H) = d(H) ≥ k/3 > b
- H has non-trivial automorphism

Observe

- All internal cuts in H >b
- Cuts of size ≤ b are external cuts between H and G-H. They will never break H.

Recall: Cuts of size ≤ b are external cuts between H and G-H. They will never break H.

- Step1:
  - Use Gomory-Hu tree to break the graph along the cuts of size ≤ b
  - Finally, one of these chunks is H
- Step2:
  - Find which one is H
  - H needs to be unique

Graph H: k nodes

$\binom{k}{2} \approx k^2/2$ Possible edges → $2^{k^2/2}$ Possible graphs

Subgraph G-H:

- There are n/k sets.
- Each set has k! possible graphs

$(n/k)k!$ Possible subgraphs

Probability of isomorphic: $P = \frac{(n/k)k!}{2^{k^2/2}}$ → Drop quickly when k >√ log(n)

Example: n=1000M

Log(n) = 9

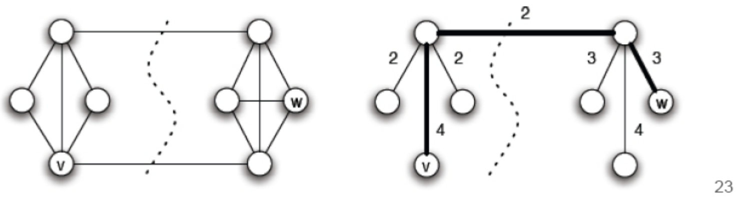
Choose k = 12 → $P = 8.45271119 \times 10^{-6}$

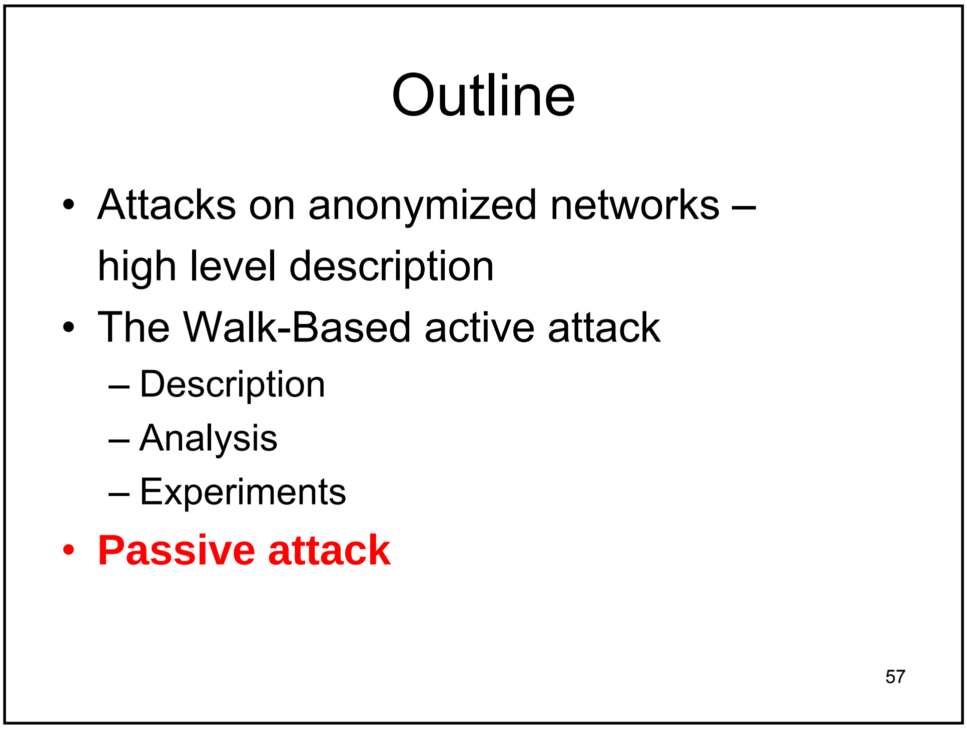
- Tree with the same set of nodes in G. Edge are labeled with weight.
- The value of min-cut for (u,v)

  = #edges on the smallest cut that will disconnect u and v

  = min-weight on the path between u and v in T
- Breaking graph G along the cuts of size ≤ b

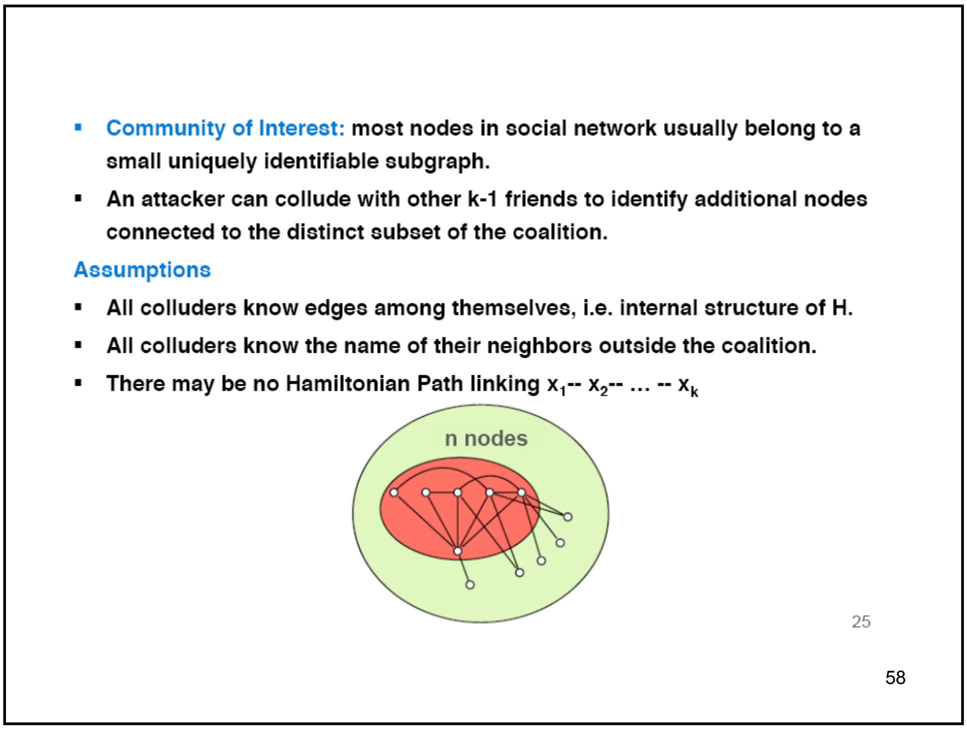
  = delete all edges of size ≤ b from T
- Repeat until all forests have size k
- Brute force to check whether each forest is isomorphic to H





Outline

- Attacks on anonymized networks – high level description
- The Walk-Based active attack
  - Description
  - Analysis
  - Experiments
- **Passive attack**



- **Community of Interest:** most nodes in social network usually belong to a small uniquely identifiable subgraph.
- An attacker can collude with other k-1 friends to identify additional nodes connected to the distinct subset of the coalition.

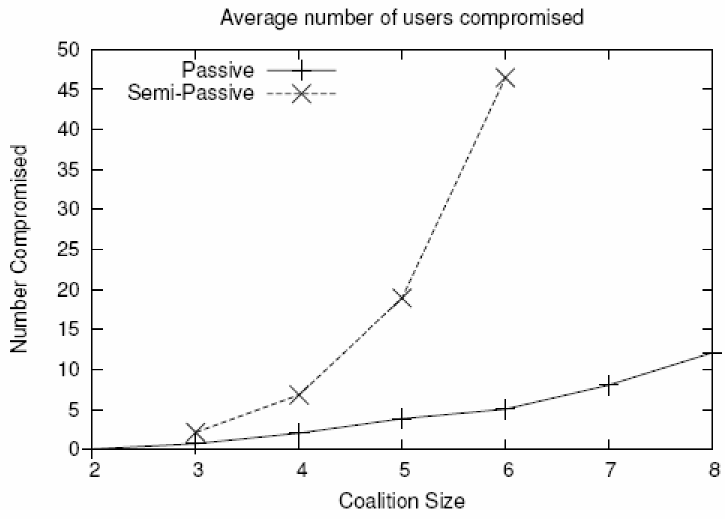
Assumptions

- All colluders know edges among themselves, i.e. internal structure of H.
- All colluders know the name of their neighbors outside the coalition.
- There may be no Hamiltonian Path linking x_1-- x_2-- … -- x_k

Search Tree T

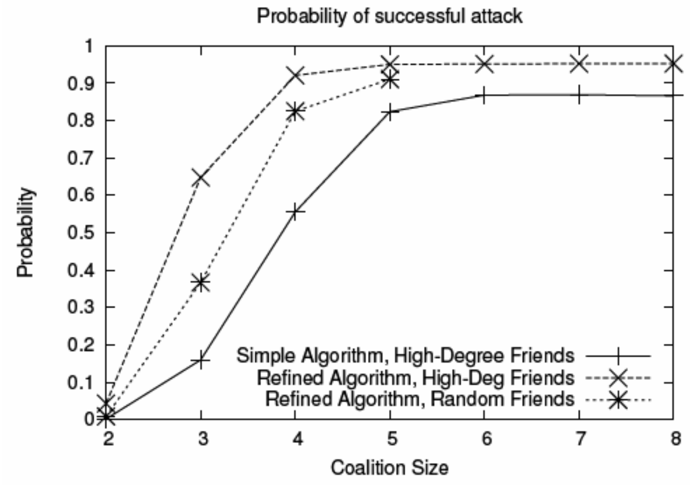
- Degree Test
- For ALL subset S $\subseteq$ {1,…,k, node α matching H must have $g_{\alpha}(S) = g(S)$
  - If we consider S = {1,3,5}
  - g(S) = q: There is q users that connects to x_1, x_3,and x_5.

Passive Attack - Results



Passive Attack

- *H* is a coalition, recovered by same search algorithm
- Nothing guaranteed, but works in practice



Active Attack	Passive Attack
▪ More effective. Work with high probability in any network. ▪ Can choose the victims ▪ Risk of being detected	▪ Attackers may not be able to identify themselves after seeing the released anonymized network. ▪ The victims are only those linked to the attackers. ▪ Harder to detect

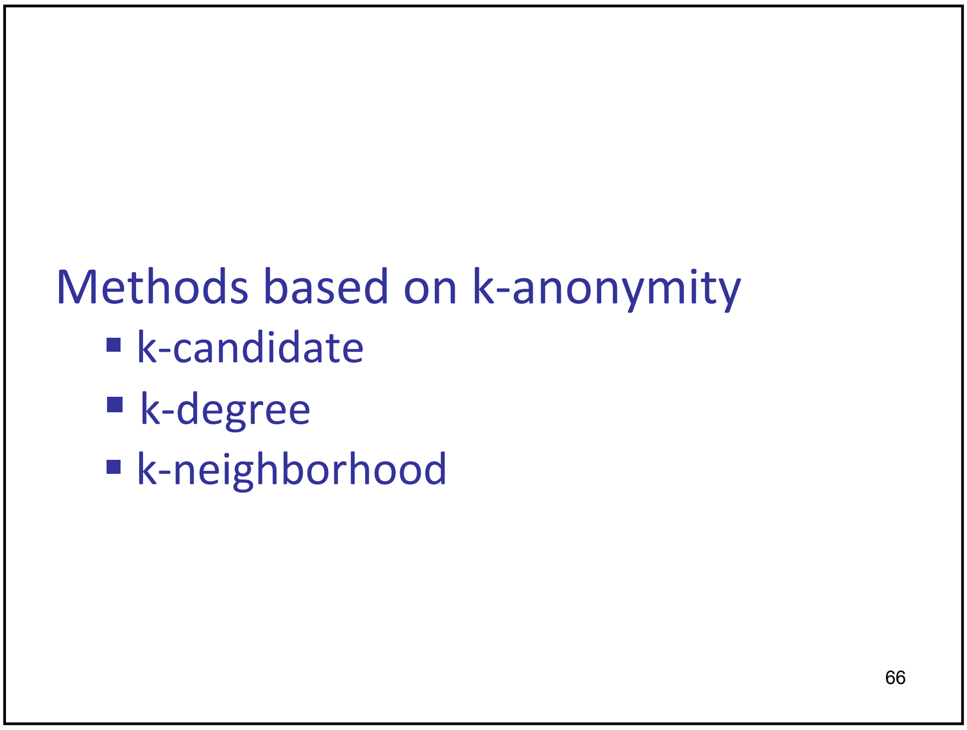
Semi-Passive Attack:

- Create only additional links to the targeted nodes. No additional node.
- Can breach privacy on the scale approaching that of the active attack.

Potential Solutions

- **Random Perturbation**
  - **m-perturbation**
  - **Randomly delete m edges and insert m edges**
- **Model-based Perturbation**
  - **Derive statistical model from original data**
  - **Develop model to bias the perturbation to give desire properties of the graph**
  - **Give better utility**

Privacy in Social Networks: k-anonymity



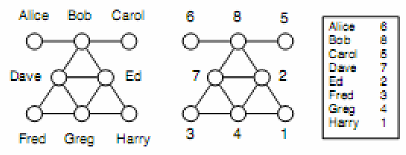
Methods based on k-anonymity

- k-candidate
- k-degree
- k-neighborhood

k-candidate Anonymity

M Hay et al, *Resisting Structural Re-identification in Anonymized Social Networks* VLDB 2008

G_a the naive anonymization of G through an anonymization mapping f



Alice	6
Bob	8
Carol	5
Dave	7
Ed	2
Fred	3
Greg	4
Harry	1

An individual x ∈ V called the target has a candidate set, denoted **cand(x)** which consists of the nodes of G_a that could possibly correspond to x

Given an uninformed adversary, each individual has the same risk of re-identification, cand(x) = V_a

In practice, some knowledge, examples:

Bob has three or more neigbhbors, cand(Bob) =?

Greg is connected to at least two nodes, each with degree 2, cand(Greg) =?

Focus on

structural re-identification where the information of the adversary is about graph structure

analysis about structural properties: finding communities, fitting power-law graph models, enumerating motifs, measuring diffusion, accessing resiliency

Two factors

- descriptive power of the external information
- structural similarity of nodes

Knowledge Acquisition in Practice

External information may be acquired through

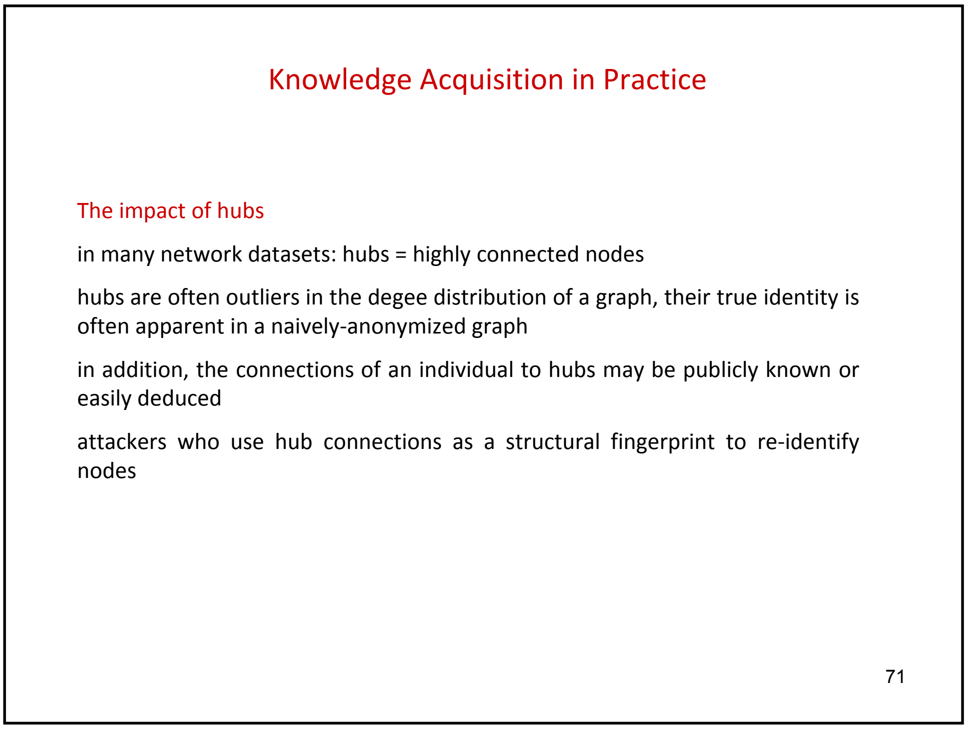
- malicious actions by the adversary or
- through public infomation sources

An adversary may be a participant in the network with some innate knowledge of entities and their relationships

Radius - neighborhood

Adversary knowledge about a targeted individual tends to be local to the targeted node

For the participant adversary there is a horizon of awareness of about distance two around most individuals



Knowledge Acquisition in Practice

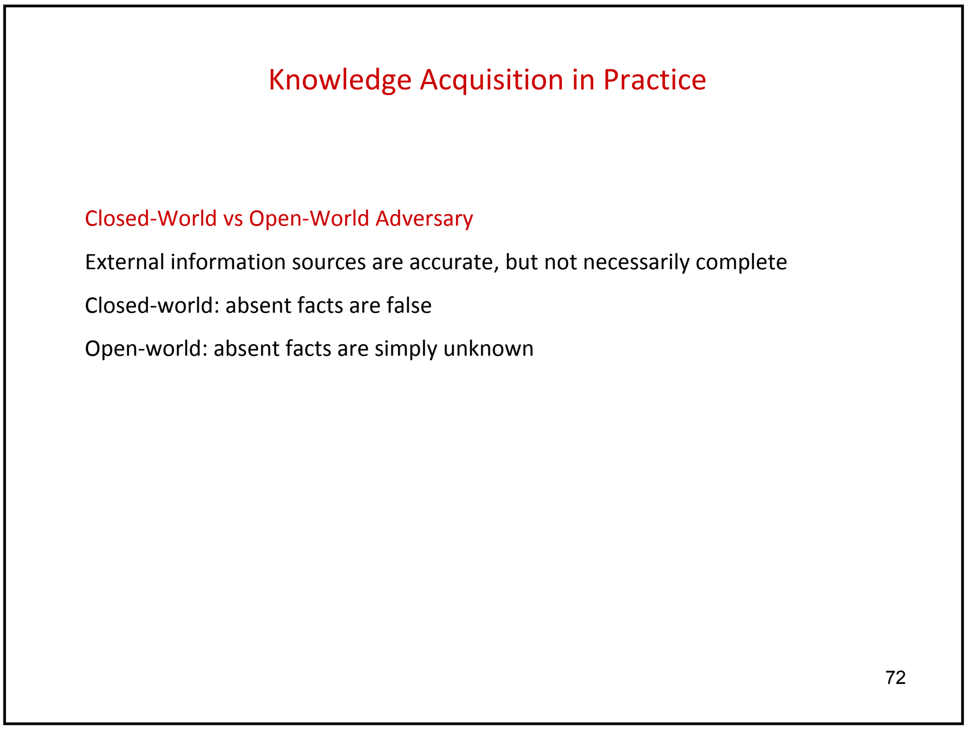
The impact of hubs

in many network datasets: hubs = highly connected nodes

hubs are often outliers in the degee distribution of a graph, their true identity is often apparent in a naively-anonymized graph

in addition, the connections of an individual to hubs may be publicly known or easily deduced

attackers who use hub connections as a structural fingerprint to re-identify nodes



Knowledge Acquisition in Practice

Closed-World vs Open-World Adversary

External information sources are accurate, but not necessarily complete

Closed-world: absent facts are false

Open-world: absent facts are simply unknown

Anonymity through Structural Similarity

A strong form of structural similarity between nodes is automorphic equivalence.

[automorphic equivalence]. Two nodes x, y ∈ V are automorphically equivalent (denoted x ≡ y) if there exists an isomorphism from the graph onto itself that maps x to y.

Example: Fred and Harry, but not Bob and Ed

Automorphic equivalence induces a partitioning on V into sets whose members have identical structural properties.

An adversary —even with exhaustive knowledge of the structural position of a target node — cannot identify an individual beyond the set of entities to which it is automorphically equivalent.

We say that these nodes are structurally indistinguishable [nodes in the graph achieve anonymity by being “hidden in the crowd” of its automorphic class members.]

Anonymity through Structural Similarity

Some special graphs have large automorphic equivalence classes.
For example, in a complete graph, or in a graph which forms a ring, all nodes are automorphically equivalent.

In most graphs we expect to find small automorphism classes, likely to be insufficient for protection against re-identification.

Automorphic equivalence is an extremely strong notion of structural similarity.

In order to distinguish two nodes in different automorphic equivalence classes, it may be necessary to use complete information about their positions in the graph.

We must consider the distinguishability of nodes to realistic adversaries with limited external information

Adversary Knowledge

Model the external information of an adversary as access to a source that provides answers to a restricted knowledge query Q evaluated for a single target node of the original graph G.
We always assume knowledge gathered by the adversary is accurate.

For a target node x, the adversary uses Q(x) to refine the feasible candidate set.
Since G_a is published, the adversary can easily evaluate any structural query directly on G_a.
Thus the adversary will compute the refined candidate set that contains all nodes in the published graph G_a that are consistent with answers to the knowledge query on the target node.

[CANDIDATE SET UNDER Q]. For a query Q over a graph, the candidate set of x w.r.t Q is candQ(x) = $\{y \in V_a \mid Q(x) = Q(y)\}$.

Adversary Knowledge

1. Vertex Refinement Queries
2. Subgraph Queries
3. Hub Fingerprint Queries

Vertex Refinement Queries I

A class of queries, of increasing power, which report on the local structure of the graph around a node.

- The weakest knowledge query, H_0, simply returns the label of the node. (We consider unlabeled graphs, so H0 returns ε on all input nodes; else the node label - to include attributes)

- $H_1(x)$ returns the degree of x,

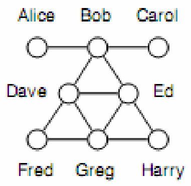
- $H_2(x)$ returns the multiset of each neighbors' degree,

- $H_i(x)$ returns the multiset of values which are the result of evaluating H_{i-1} on the nodes adjacent to x

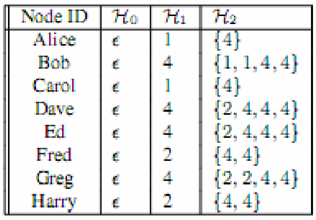
$$\mathcal{H}_i(x) = \{\mathcal{H}_{i-1}(z_1), \mathcal{H}_{i-1}(z_2) \ldots, \mathcal{H}_{i-1}(z_m)\}$$

where $z_1 \ldots z_m$ are the nodes adjacent to x.

Vertex Refinement Queries II



(a) graph

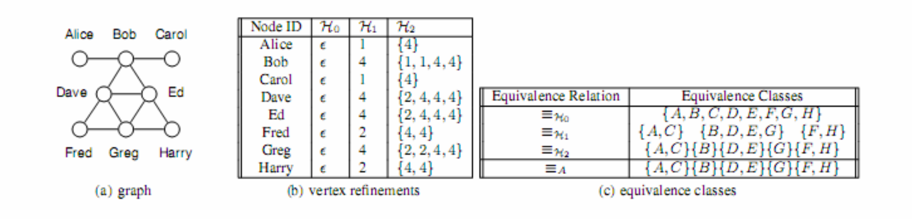


Node ID	$\mathcal{H}_0$	$\mathcal{H}_1$	$\mathcal{H}_2$
Alice	ϵ	1	$\{4\}$
Bob	ϵ	4	$\{1, 1, 4, 4\}$
Carol	ϵ	1	$\{4\}$
Dave	ϵ	4	$\{2, 4, 4, 4\}$
Ed	ϵ	4	$\{2, 4, 4, 4\}$
Fred	ϵ	2	$\{4, 4\}$
Greg	ϵ	4	$\{2, 2, 4, 4\}$
Harry	ϵ	2	$\{4, 4\}$

(b) vertex refinements

Vertex Refinement Queries III

DEFINITION 2 (RELATIVE EQUIVALENCE). Two nodes x, y in a graph are equivalent relative to H_i, denoted $x \equiv_{H_i} y$, if and only if $H_i(x) = H_i(y)$.



(a) graph

Node ID	$\mathcal{H}_0$	$\mathcal{H}_1$	$\mathcal{H}_2$
Alice	ϵ	1	$\{4\}$
Bob	ϵ	4	$\{1,1,4,4\}$
Carol	ϵ	1	$\{4\}$
Dave	ϵ	4	$\{2,4,4,4\}$
Ed	ϵ	4	$\{2,4,4,4\}$
Fred	ϵ	2	$\{4,4\}$
Greg	ϵ	4	$\{2,2,4,4\}$
Harry	ϵ	2	$\{4,4\}$

(b) vertex refinements

Equivalence Relation	Equivalence Classes
$\equiv_{\mathcal{H}_0}$	$\{A,B,C,D,E,F,G,H\}$
$\equiv_{\mathcal{H}_1}$	$\{A,C\}$ $\{B,D,E,G\}$ $\{F,H\}$
$\equiv_{\mathcal{H}_2}$	$\{A,C\}\{B\}\{D,E\}\{G\}\{F,H\}$
$\equiv_{A}$	$\{A,C\}\{B\}\{D,E\}\{G\}\{F,H\}$

(c) equivalence classes

Vertex Refinement Queries IV

To an adversary limited to knowledge query H_i, nodes equivalent with respect to H_i are indistinguishable.

Proposition: Let x, x' ∈ V, if $x \equiv_{H_i} x'$, then $cand_{H_i}(x) = cand_{H_i}(x')$

Iterative computation of H continues until no new vertices are distinguished.

We call this query H*.
In the example of Figure 2, H* = H_2.

Equivalence under H* is very likely to coincide with automorphic equivalence.

Subgraph Queries I

Vertex refinement queries are a concise way to describe locally expanding structural queries.
Two limitations.

- always provide complete information about the nodes adjacent to the target (i.e., an instance of closed-world knowledge).
- H queries can describe arbitrarily large subgraphs around a node if that node is highly connected, thus the index of H query may be a coarse measure of the information learned

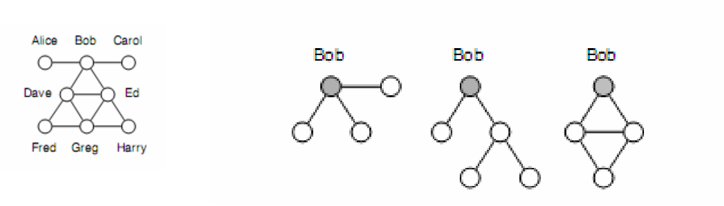
  For example, if $H_1(x) = 100$, the adversary learns about a large subgraph in G, whereas $H_1(y) = 2$ provides much less information.

As an alternative, we consider a very general class of queries which assert the existence of a subgraph around the target node.

We measure the descriptive power of a query by *counting the number of edges in the described subgraph*; we refer to these as **edge facts**.

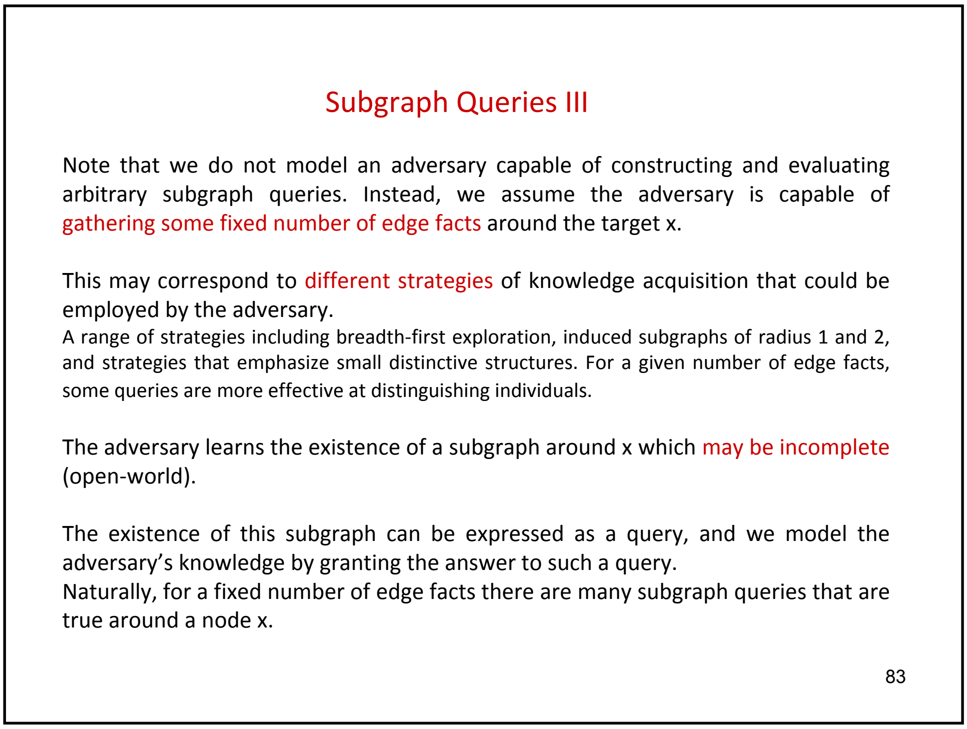
Subgraph Queries II

Example: Three subgraph queries centered around Bob.



The first simply asserts that Bob has (at least) three distinct neighbors, the second describes a tree of nodes near Bob, and the third relates nearby nodes in a subgraph.

These informal query patterns use 3, 4, and 5 edge facts, respectively.



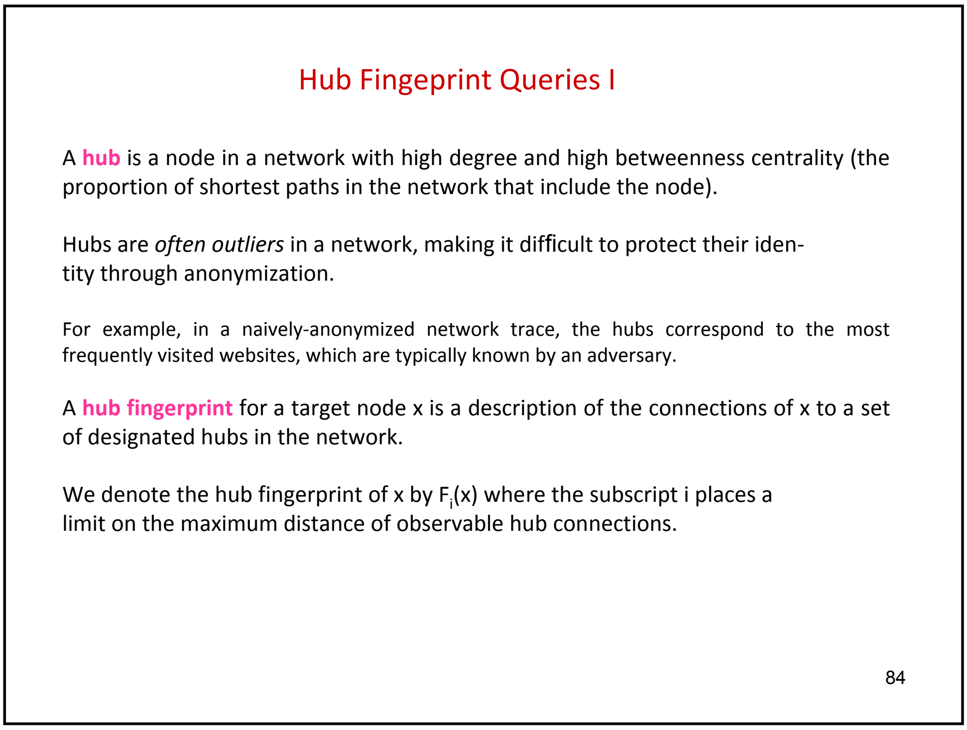
Subgraph Queries III

Note that we do not model an adversary capable of constructing and evaluating arbitrary subgraph queries. Instead, we assume the adversary is capable of gathering some fixed number of edge facts around the target x.

This may correspond to different strategies of knowledge acquisition that could be employed by the adversary.
A range of strategies including breadth-first exploration, induced subgraphs of radius 1 and 2, and strategies that emphasize small distinctive structures. For a given number of edge facts, some queries are more effective at distinguishing individuals.

The adversary learns the existence of a subgraph around x which may be incomplete (open-world).

The existence of this subgraph can be expressed as a query, and we model the adversary's knowledge by granting the answer to such a query.
Naturally, for a fixed number of edge facts there are many subgraph queries that are true around a node x.



Hub Fingerprint Queries I

A **hub** is a node in a network with high degree and high betweenness centrality (the proportion of shortest paths in the network that include the node).

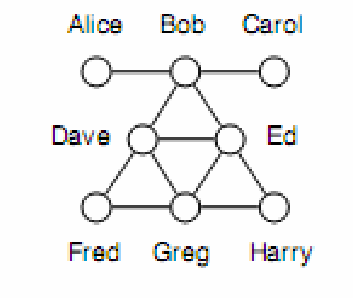
Hubs are *often outliers* in a network, making it difficult to protect their identity through anonymization.

For example, in a naively-anonymized network trace, the hubs correspond to the most frequently visited websites, which are typically known by an adversary.

A **hub fingerprint** for a target node x is a description of the connections of x to a set of designated hubs in the network.

We denote the hub fingerprint of x by $F_i(x)$ where the subscript i places a limit on the maximum distance of observable hub connections.

Hub Fingeprint Queries II



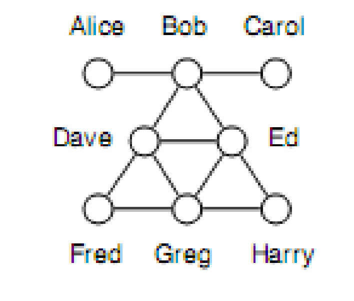
Forexample, if we consider Dave and Ed hubs,
The hub fingerprint of Fred is a vector of his shortest path lengths (bounded by i) to each hub.
F1(Fred) = (1; 0) because Fred is distance 1 from Dave but not connected to Ed in one hop or less;
F2(Fred) = (1; 2) because Fred is distance 1 from Dave and distance 2 from Ed.

Hub Fingeprint Queries III

We consider an adversary capable of gathering hub fingerprints in both an open and a closed world.

- In the closed world, the lack of a connection to a hub implies with certainty that no connection exists.
- In the open world, the absence of a connection in a hub fingerprint may simply represent incompleteness in the adversary's knowledge.

Example:
In the open world, if the adversary knows F_1(Fred) = (1; 0) then nodes in the anonymized graph with F_1 fingerprints of (1; 0) or (1; 1) are both candidates for Fred.



Comparison of the Knowledge Models I

Vertex refinement queries and subgraph queries are related, but differ

Expressiveness: Vertex refinement queries provide complete information about node degree. A subgraph query can never express H_i knowledge because subgraph queries are existential and cannot assert exact degree constraints or the absence of edges in a graph.

Complexity of computing H* is linear in the number of edges in the graph, and is therefore efficient even for large datasets. Evaluating subgraph queries, on the other hand, can be NP-hard in the number of edge facts, as computing candidate sets for subgraph queries requires finding all isomorphic subgraphs in the input graph.

Disclosure risk: Although we do not place computational restrictions on the adversary, the vertex refinement queries allow a data owner to efficiently assess disclosure risk.
Yet, the semantics of subgraph queries seem to model realistic adversary capabilities more accurately. It may be difficult for an adversary to acquire the detailed structural description of higher- order vertex refinement queries.

Vertex refinement queries offer an efficient and conservative measure of structural diversity in a graph. In addition, Hi queries are conceptually appealing as they represent a natural spectrum of structural knowledge, beginning with H_1 which reports node degree, and converging, as i increases, on automorphic equivalence.

Comparison of the Knowledge Models II

We note that both have well-studied logical foundations. H_i knowledge corresponds to first order logic with counting quantifiers, restricted to i variables.

Subgraph queries can be expressed as conjunctive queries with disequalities. The number of edge facts corresponds to the number of subgoals in the query

Disclosure in Real Networks

- Study three networked data sets, drawn from diverse domains.

- For each data set, consider each node in turn as a target.

- Assume the adversary computes a vertex refinement query, a subgraph query, or a hub fingerprint query on that node, and then compute the corresponding candidate set for that node.

- Report the distribution of candidate set sizes across the population of nodes to characterize how many nodes are protected and how many are identifiable.

Disclosure in Real Networks

Hep-Th database: papers and authors in theoretical high-energy physics, taken from the arXiv archive, linked if they wrote at least two papers together.

Enron dataset: derived from a corpus of email sent to and from managers at Enron Corporation, made public by the Federal Energy Regulatory Commission during its investigation of the company. Two individuals are connected if they corresponded at least 5 times.

Net-trace dataset: from an IP-level network trace collected at a major university. The trace monitors traffic at the gateway; it produces a bipartite graph between IP addresses internal to the institution, and external IP addresses. Restricted to 187 internal addresses from a single campus department and the 4026 external addresses to which at least 20 packets
were sent on port 80 (http traffic).

All datasets have undirected edges, with self-loops removed. Eliminated a small percentage of disconnected nodes in each dataset, focusing on the largest connected component in the graph.

Disclosure in Real Networks

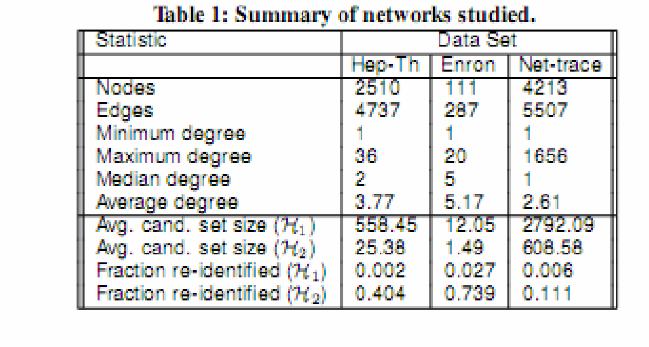


Table 1: Summary of networks studied.

Statistic	Data Set		
	Hep-Th	Enron	Net-trace
Nodes	2510	111	4213
Edges	4737	287	5507
Minimum degree	1	1	1
Maximum degree	36	20	1656
Median degree	2	5	1
Average degree	3.77	5.17	2.61
Avg. cand. set size ($\mathcal{H}_1$)	558.45	12.05	2792.09
Avg. cand. set size ($\mathcal{H}_2$)	25.38	1.49	608.58
Fraction re-identified ($\mathcal{H}_1$)	0.002	0.027	0.006
Fraction re-identified ($\mathcal{H}_2$)	0.404	0.739	0.111

Reidentification: Vertex Refinement I

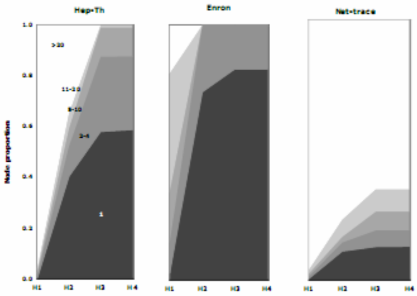


Figure 4: Relationship between candidate size and vertex refinement knowledge $\mathcal{H}_i$ for $i = 1..4$ for three network datasets. The trend lines show the percentage of nodes whose candidate sets have sizes in the following buckets: $[1]$ (black), $[2, 4]$, $[5, 10]$, $[11, 20]$, $[21, \infty]$ (white).

For the Hep-Th data, H1 leaves nearly all nodes at low risk for re-identification, and it requires H3 knowledge to uniquely re-identify a majority of nodes.

For Enron, under H1 about 15% of the nodes have candidate sets smaller than 5, while only 19% are protected in candidate sets greater than 20. Under H2, re-identification jumps dramatically so that virtually all nodes have candidate sets less than 5.

Net-trace has substantially lower disclosure overall, with very few identified nodes under H1, and even H4 knowledge does not uniquely identify more than 10% of the nodes. This results from the unique bipartite structure of the network trace dataset: many nodes in the trace have low degree, as they are unique or rare web destinations contacted by only one internal host.

Reidentification: Vertex Refinement II

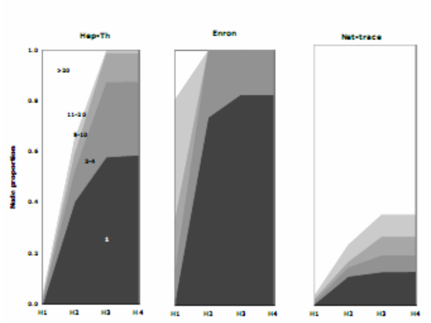


Figure 4: Relationship between candidate size and vertex refinement knowledge $\mathcal{H}_i$ for $i = 1..4$ for three network datasets. The trend lines show the percentage of nodes whose candidate sets have sizes in the following buckets: $[1]$ (black), $[2, 4]$, $[5, 10]$, $[11, 20]$, $[21, \infty]$ (white).

A natural precondition for publication is a very low percentage of high-risk nodes under a reasonable assumption about adversary knowledge. Two datasets meet that requirement for H1 (Hep-Th and Net-trace), but no datasets meet that requirement for H2.

Overall, there can be significant variance across different datasets in their vulnerability to different adversary knowledge.

However, across all datasets, the most significant change in re-identification is from H1 to H2, illustrating the increased power of adversaries that can explore beyond the target's immediate neighborhood. Re-identification tends to stabilize after H3 — more information in the form of H4 does not lead to an observable increase in re-identification in any dataset.

Finally, even though there are many re-identified nodes, a substantial number of nodes are not uniquely identified even with H4 knowledge.

Reidentification: Subgraph Queries I

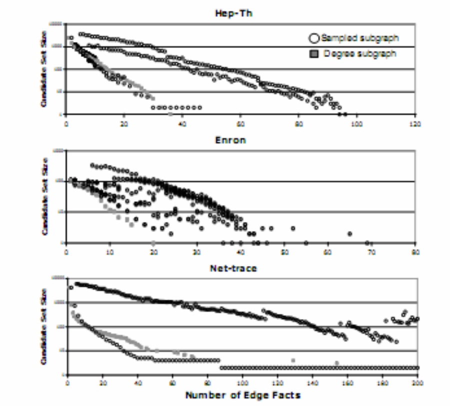


Figure 5: Candidate set sizes (on a log scale) for sampled subgraph queries consisting of specified number of edge facts. (Please note differences in scale.)

Figure 5 shows the relationship between the number of edge facts and re-identification success.

Each point represents a subgraph query of a specified size; the re-identification success is measured by the size of the candidate set (vertical axis).

For a fixed number of edge facts, there are many possible subgraph queries.

We simulated adversaries who gather facts around the target according to a variety of strategies: breadth-first exploration (labelled "Degree subgraphs"), random subgraphs, induced subgraphs of radius 1 and 2, and small dense structures (collectively referred to as "Sampled subgraphs")

Reidentification: Subgraph Queries II

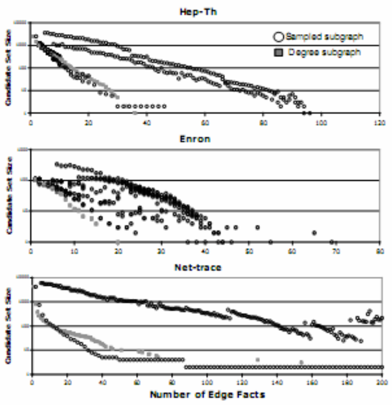


Figure 5: Candidate set sizes (on a log scale) for sampled subgraph queries consisting of specified number of edge facts. (Please note differences in scale.)

Overall, disclosure is substantially lower than for vertex refinement queries.

To select candidate sets of size less than 10 requires a subgraph query of size 24 for Hep-Th, size 12 for Enron, and size 32 for Net-trace.

The smallest subgraph query resulting in a unique disclosure was size 36 for Hep-Th and 20 for Enron. The smallest candidate set witnessed for Net-trace was size 2, which resulted from a query consisting of 88 edge facts.

Breadth-first exploration led to selective queries across all three datasets. Such a query explores all neighbors of a node and then starts to explore all neighbors of a randomly chosen neighbor, etc.

This asserts lower bounds on the degree of nodes. In Enron, these were the most selective subgraph queries witnessed; for Hep-Th and Net-trace, the more selective subgraph queries asserted the existence of two nodes with a large set of common neighbors.

Reidentification: Subgraph Queries III

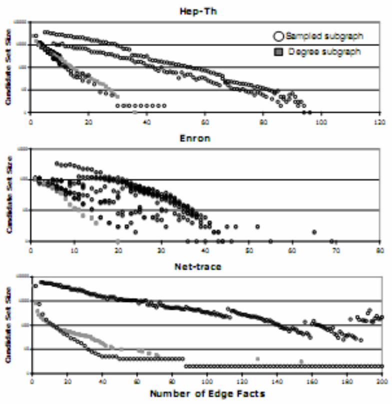


Figure 5: Candidate set sizes (on a log scale) for sampled subgraph queries consisting of specified number of edge facts. (Please note differences in scale.)

The results presented above illustrate the diverse subset of subgraph queries sampled.

While it is clearly intractable to perform an exhaustive search over all possible subgraphs and matching them to each node in the graph, it is an interesting open question to determine, given a graph, and a fixed number of edge facts, the subgraph query that will result in the smallest candidate set.

This would refect the worst-case disclosure possible from an adversary restricted to a specified number of edge facts.

Reidentification: Hub Fingerprints I

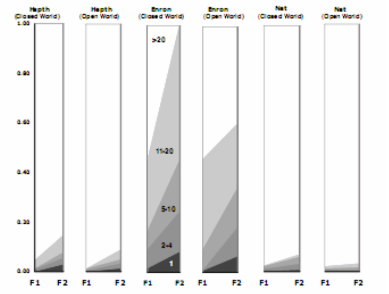


Figure 6: Candidate set sizes for hub fingerprint queries: $\mathcal{F}_1$ and $\mathcal{F}_2$ are shown for each dataset under a closed-world and open-world assumption.

Figure shows the candidate set sizes for hub fingerprints F1 and F2, choosing the 5 highest degree nodes as hubs for Enron, and the 10 highest degree nodes for both Hepth and Net-trace. *The choice of the number of hubs was made by considering whether the degree of the node was distinguishable in the degree distribution and therefore likely to be an outlier in the original graph.*

Under both the closed-world interpretation and the open-world interpretation.

Reidentification: Hub Fingerprints II

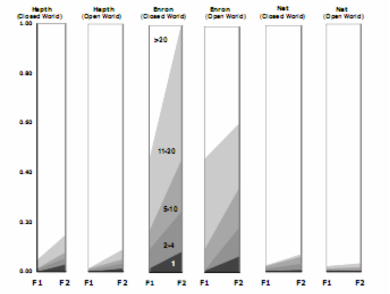


Figure 6: Candidate set sizes for hub fingerprint queries: $\mathcal{F}_1$ and $\mathcal{F}_2$ are shown for each dataset under a closed-world and open-world assumption.

Generally, disclosure is low using hub fingerprints. At distance 1, 54% of the nodes in Enron were not connected to any hub and therefore hub fingerprints provide no information. This statistic was 90% for Hepth and 28% for Net-trace.

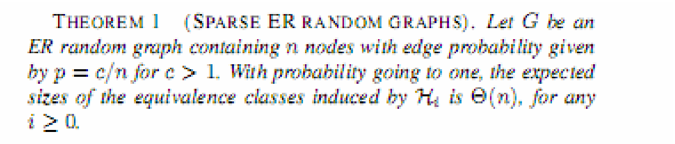
In addition, connectivity to hubs was fairly uniform across individuals. For example, the space of possible fingerprints at distance 1 for Hepth and Net-trace is 2^{10}= 1024. Of these, only 23 distinct fingerprints were observed for Hepth and only 46 for Net-trace.

While hubs themselves stand out, they have high-degrees, which means connections to a hub are shared by many. While hubs would appear to be a challenge for anonymization, this finding suggests that disguising hubs in published data may not be required to maintain anonymity.

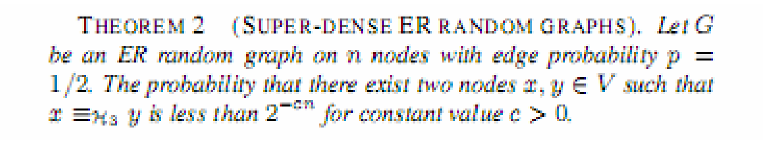
Anonymization in Random Graphs

Erdos-Renyi (RE) random graphs
n nodes by sampling each edge indpendently with probability p
sparce p = c/n, dense = clogn/n, super-dense p = c (c is a constant)
c>1,
include a giant connected component of size Θ(n), and a collection of smaller components (sparse)
completed connected (dense)

Reindification in Random Graphs



THEOREM 1 (SPARSE ER RANDOM GRAPHS). *Let G be an ER random graph containing n nodes with edge probability given by $p = c/n$ for $c > 1$. With probability going to one, the expected sizes of the equivalence classes induced by $\mathcal{H}_i$ is $\Theta(n)$, for any $i \geq 0$.*

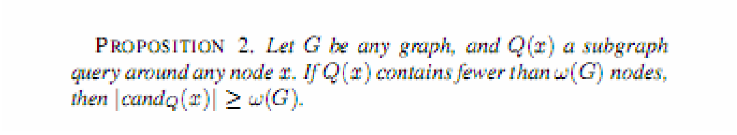


THEOREM 2 (SUPER-DENSE ER RANDOM GRAPHS). *Let G be an ER random graph on n nodes with edge probability $p = 1/2$. The probability that there exist two nodes $x, y \in V$ such that $x \equiv_{\mathcal{H}_3} y$ is less than 2^{-cn} for constant value $c > 0$.*

For dense, nodes cannot be identified for H_1 for any c>0, but all nodes are re-identifiable for H_2 for any c>1

Reindification in Random Graphs

ω(G) the number of nodes in the largest clique



PROPOSITION 2. *Let G be any graph, and $Q(x)$ a subgraph query around any node x. If $Q(x)$ contains fewer than $\omega(G)$ nodes, then $|cand_Q(x)| \geq \omega(G)$.*

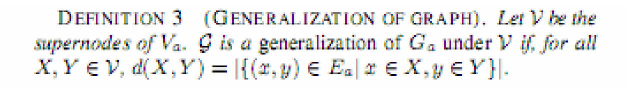
Any subgraph query matching fewer than ω(G) nodes, will match any node in the clique

Anonymization Algorithms

Partition/Cluster the nodes of Ga into disjoint sets

In the generalized graph,
supernodes: subsets of Va
edges with labels that report the density

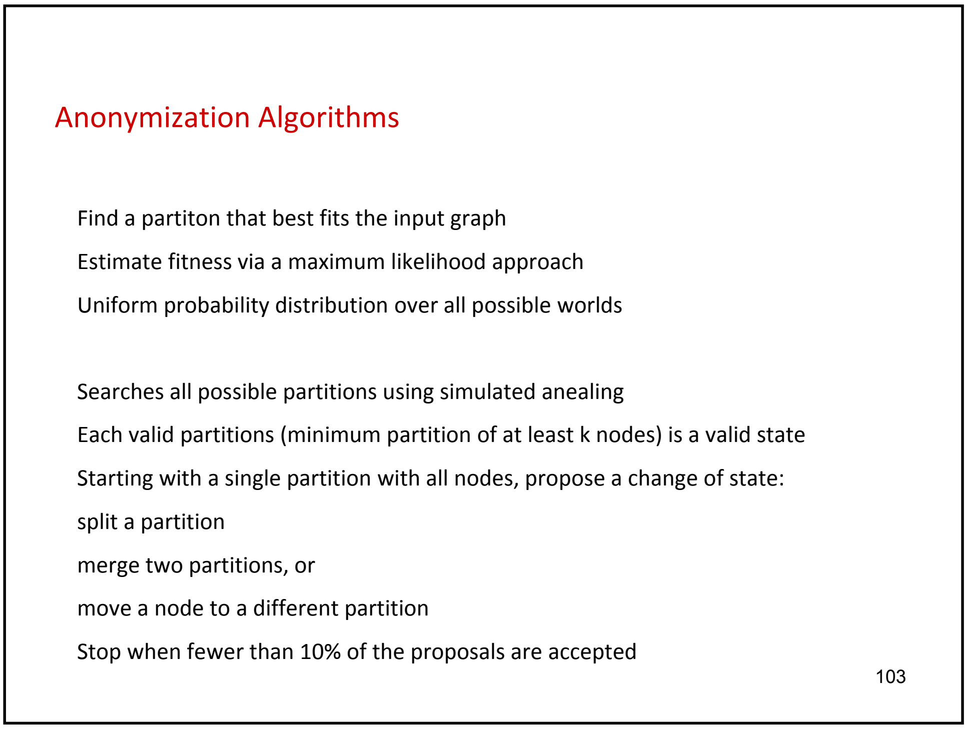
Partitions of size at least k



DEFINITION 3 (GENERALIZATION OF GRAPH). *Let $\mathcal{V}$ be the supernodes of V_a. $\mathcal{G}$ is a* generalization of G_a under $\mathcal{V}$ *if, for all* $X, Y \in \mathcal{V}$, $d(X,Y) = |\{(x,y) \in E_a | x \in X, y \in Y\}|$.

Extreme cases: a singe super-node with self-loop, Ga

Again: Privacy vs Utility



Anonymization Algorithms

Find a partiton that best fits the input graph

Estimate fitness via a maximum likelihood approach

Uniform probability distribution over all possible worlds

Searches all possible partitions using simulated anealing

Each valid partitions (minimum partition of at least k nodes) is a valid state

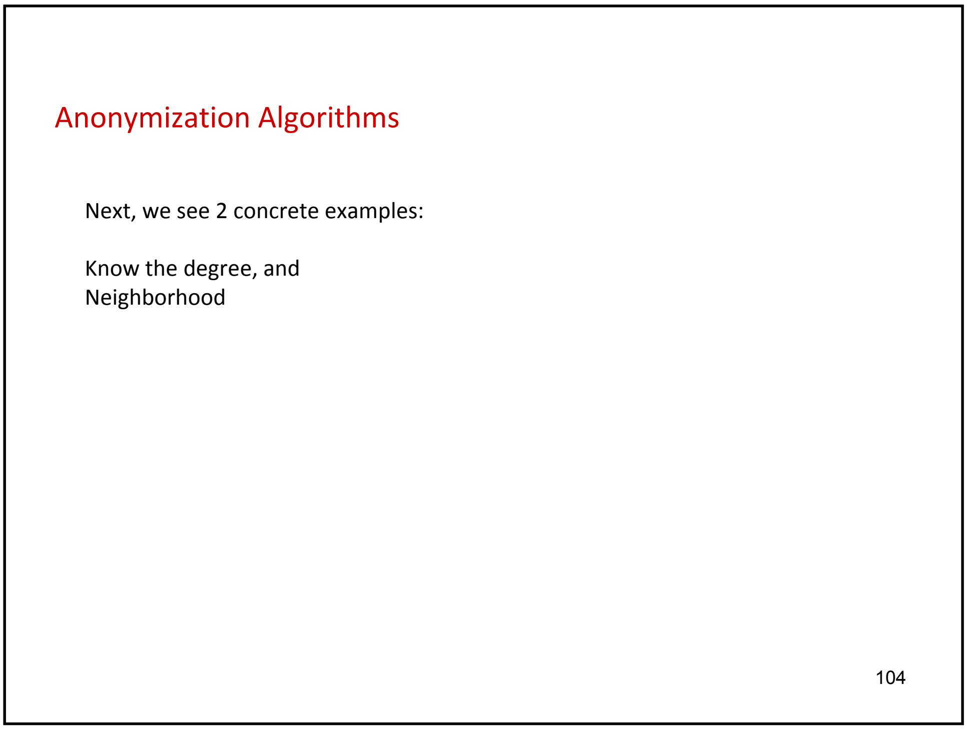
Starting with a single partition with all nodes, propose a change of state:

split a partition

merge two partitions, or

move a node to a different partition

Stop when fewer than 10% of the proposals are accepted



Anonymization Algorithms

Next, we see 2 concrete examples:

Know the degree, and
Neighborhood