Privacy-Preserving Indexing for eHealth Information Networks

Yuzhe Tang, Ting Wang, Ling Liu,
Shicong Meng, and Balaji Palanisamy
College of Computing,
Georgia Institute of Technology

Talk Overview

- **Motivation**
  - **Content Privacy issues in sharing access-controlled content**
  - **State of Art research**
- **Our approach ss-PPI**
  - **Data structure for search on access-controlled content**
  - **Algorithm for building such a data structure**
  - **Experiments**

Introduction

- e-Health systems today
  - A network of multiple healthcare providers
    - Physicans' offices, hospitals, labs, insurance companies, etc
  - Collectively provides large-scale information sharing over distributed, access controlled content.
  - Example: Nationwide Health Information Network (NHIN).
- **Problems of Sharing Private Content**
  - Rapid growth in Private & Semi-Private information on the network
    - Experimental results of drug tests
  - Mechanisms to search information have failed to keep pace
    - Public Information: Google, Yahoo!
    - Private Information: ???

Problem Statement

- Healthcare Providers
  - **Hospitals are willing to share documents about patients only to those with access control such as family doctors of the patients and the list of people to which the patient has grants the access.**
    - **Alzheimer's Disease *(Alice, Bob),* AIDS *(Alice),* Diabetics*(Alice, Bob, Lisa, …)***
  - **Need to enforces *access policy***

- Searchers
  - Wants documents that match her keyword query Q
  - Has an identity

- New problem raised
  - Users want effective and efficient search facility.
  - Providers don't want to disclose their content (i.e., content privacy).
  - How to facilitate effective search while minimally revealing content privacy.

Assumptions and Data Structures

- A search vocabulary of size M shared across N providers
- A network of N providers P1, P2, … PN.
- **Provider**: each publishes their access controlled contents with two vectors:
  - **Content vector**, one per provider:
    - a vector of M binary elements with 1 denoting match and 0 denoting unmatch andM is the size of the search vocabulary.
  - **Access Control vector**, one per legitimate user for a given provider
    - a vector of M binary elements, with 1 denoting allow access and 0 denoting denying access.
- **Searcher**: each can send keyword search any to any providers with its ID using terms from the vocabulary

Search Problem Definition

- Search Correctness
  - A *searcher* **s** issues a *query* **q** expecting a set of *documents* **d** such that
    1. d is shared by some provider p
    2. d matches the query q
    3. d is accessible to s as dictated by p's access policy

- Content Privacy
  - **An adversary A should *not* be able to deduce, using the search mechanism, that provider P is sharing document d with keywords q *unless A has been granted access to d by P***

Two-Step Search Process

- Step 1
  - Each query returns a list of providers who have a match and grant the access to the searcher
  - **Our problem: How to provide the search efficient and privacy preserving.**
- Step 2
  - Each matching provider will provide the set of documents that meet the two conditions:
    - The provider has them and they match the search keyword(s)
    - The searcher also has the permission to access them

Baseline Approaches

- Brute-force search by query broadcasting
  - Good to preserve content privacy
  - Inefficient in search performance

- Search by indexing
  - Efficient search performance
  - Reveal content privacy

- Probabilistic PPI (VLDB'03)
  - balance between privacy preservation & search performance
  - Suffer from
    - Inefficient index construction
    - Vulnerability to colluding attacks

State of Art: Brute-Force

- Query broadcasting
  - Each search query is sent to all N providers
  - Only providers who have the match docs respond
- Content Privacy
  - Good when many providers have matching docs
  - Bad when only one or small number of providers have the match
  - **Problem Cause**
    - **Every term is mapped precisely**
- Search Efficiency
  - Inefficient and worst in search performance
  - Not scalable for large N

State of Art: Search by Indexing

- Provider Index
  - Maintaining a keyword-provider inverted index
  - Each search query has a matching index entry of the providers who have the matching docs
- Content Privacy
  - Good when the index is constructed and maintained safely, thus need a trusted third party
  - Trusted third party is not realistic and not scalable
    - Need Privacy Preserving Indexing
- Search Efficiency
  - Highly efficient (best in search performance)
  - Scalable for large N

State of Art: Privacy Preserving Index

- No need for trusted third party
- Intuition
  - **Add sufficient "false positives" in such a way that filtering of "noise" is impossible or very hard**
  - **Example**

    Diabetics {…, P1, P2,…}

    Prostate cancer {…, P1, P2,…}
- Key challenge
  - Given a search term, how to determine the right amount of false positives?
    - Too much false positives poor search performance
    - Too few false positive poor content privacy

Privacy vs Performance Tradeoff

State of Art: Privacy Preserving Index

Definition

Let t_i denote the search term, P denote the set of N providers and M denote the set of providers returned by PPI. A PPI takes an input t_i and returns M, a subset of P, such that one of the following is true:

(i) M is empty if no matching document is found;

$$M = \phi \quad \text{only if} \quad \forall\, t_i \quad M \subseteq P \quad d_j : t_i \notin d_j$$

(ii) M contains a set of providers and more than 50% of M are false positives;

$$M = P_{true} \cup P_{false}, \; |P_{false}| \geq |P_{true}|$$

[Bawa et.al VLDB 2003]

(iii) M = P.

Correctness: No true positives excluded; provider enforces access control

Privacy Guarantee: Quantifiable Privacy on Reiter-Rubin scale

Accuracy/Performance Penalty: Loss in Selectivity

State of Art: Probabilistic Approach

Main ideas of Probabilistic PPI [Bawa et.al VLDB 2003]

1. Partitioning the set of N providers into random Groups of fix size
2. Keyword search returns the number of matching groups instead of matching providers

   A group is a match if one of its providers is a match

   Each group will process the query in r round and in each round a provider with a match will lie with a probability (½)^r and tell the truth with probability 1- (½)^r .

   As r increases, the probability of telling the truth increases.

   Errors are introduced with finite r .

Problems:

Inefficient index construction: higher privacy requires higher number of rounds for each group

Vulnerable to colluding attacks

Members in a group do not have the same level of privacy: The providers participate in a round earlier leaks more

Our solution: Secret-sharing Privacy Preserving Index

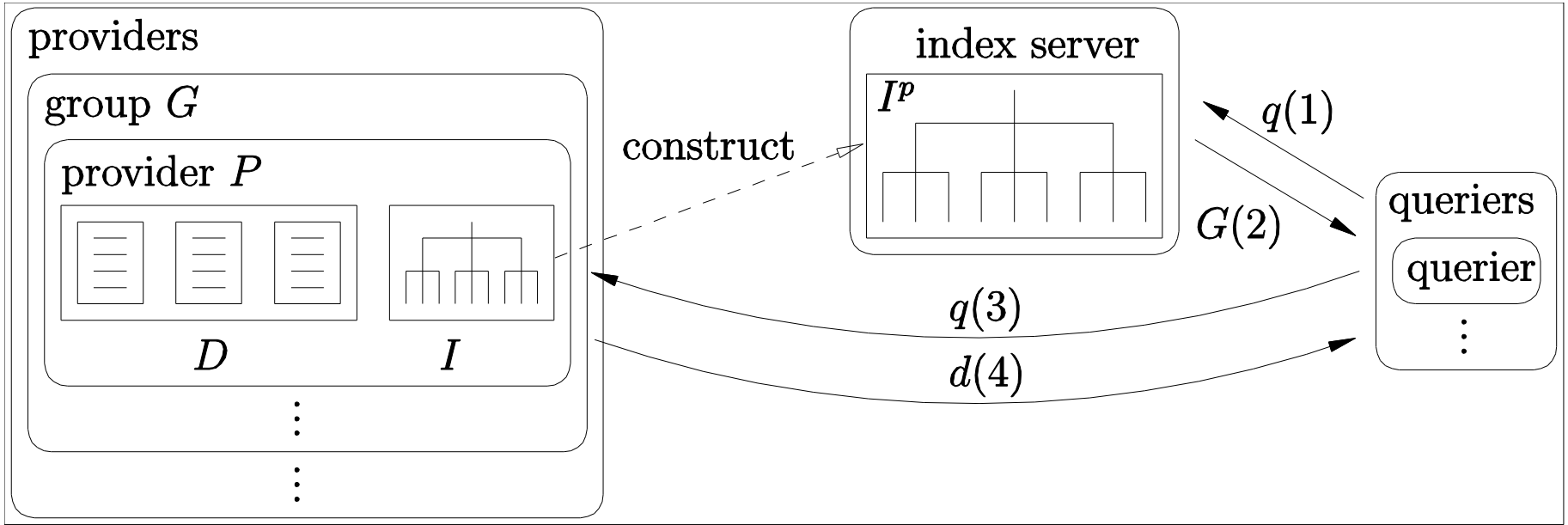
- ss-PPI: Resistant to colluding attacks
  - It achieves information-theoretic security.
  - Resistant to 2c − 2 adversaries (parameter c is tunable)

- Efficient index construction
  - Index construction done in 2 rounds (constant).
  - Parallel computation based on secret sharing

- Fine grained privacy preservation
  - Sensitive to role: query forwarded to different sets of providers for different access roles of users (query issuers).
  - Preserve both content privacy and access policy privacy.
    - members are *indistinguishable*

ss-PPI: System Overview

- Architecture
  - The ss-PPI index server is public and untrusted
  - Providers are autonomous
  - Users (searchers) directly pose queries to the ss-PPI index server.



ss-PPI: Index Construction

- Step 1: Random Group Formation.
  - Organize providers into group by universal hashing.

- ***Step 2: Secure Group Index Construction***
  - A novel **secret sharing** based protocol for secure aggregation

- Step 3: Global index construction
  - Distributed scheme to produce global index vector

ss-PPI: Index Construction

- ***Secure Group Index Construction***
  - A novel **secret sharing** based protocol which takes the search vocabulary of size M and produce a group vector of size M
    - For each search term, its corresponding element is set to 1 if at least one member has a match; and otherwise it sets to zero.
  - Goal: Secure aggregation
    - Member providers provide their matching for each term as a sub-secrete
    - Subsecrete is securely packaged such that it can be aggregated with other members without leakage of provider identity
    - Secure aggregation produces group index for each term without disclosing which members have the match.

Secure Group Index Construction

- **Main idea**: Smart use of Secrete Sharing
  - Given a group of n providers and M search terms
- **Algorithm Design:**
  - Every member provider provides a secrete vote based on each term *i* and its access role, called *sub-secret* v_i.
  - Thus the ith element in the group vector for this specific term is called *super-value v, and* $v = v_1 + v_2 + \ldots + v_n$.
  - The super-value *v* equals to the number of providers with $v_i = 1$. Thus, *v* spans from 0 to n.
- **Secure aggregation Goal**
  - The super-value should be computed accurately and securely.
  - Given a search term, each member provider has the equal probability to contribute to the aggregate super-value in the group index vector.

Secure Group Index Construction

- **Algorithm**
  - Input: *sub-secrets*
    - A bit indicating if each provider possesses each term.
  - Output*: super-value*
    - The total number of providers in the group who have a match to the term.
- **A four-step protocol**
  - Generating *sub-packets* from sub-secrets
  - Distributing sub-packets
  - Computing *super-packets* from sub-packets
  - Aggregating super-shares to construct super-secret

Transform Sub-secrete into Sub-packet

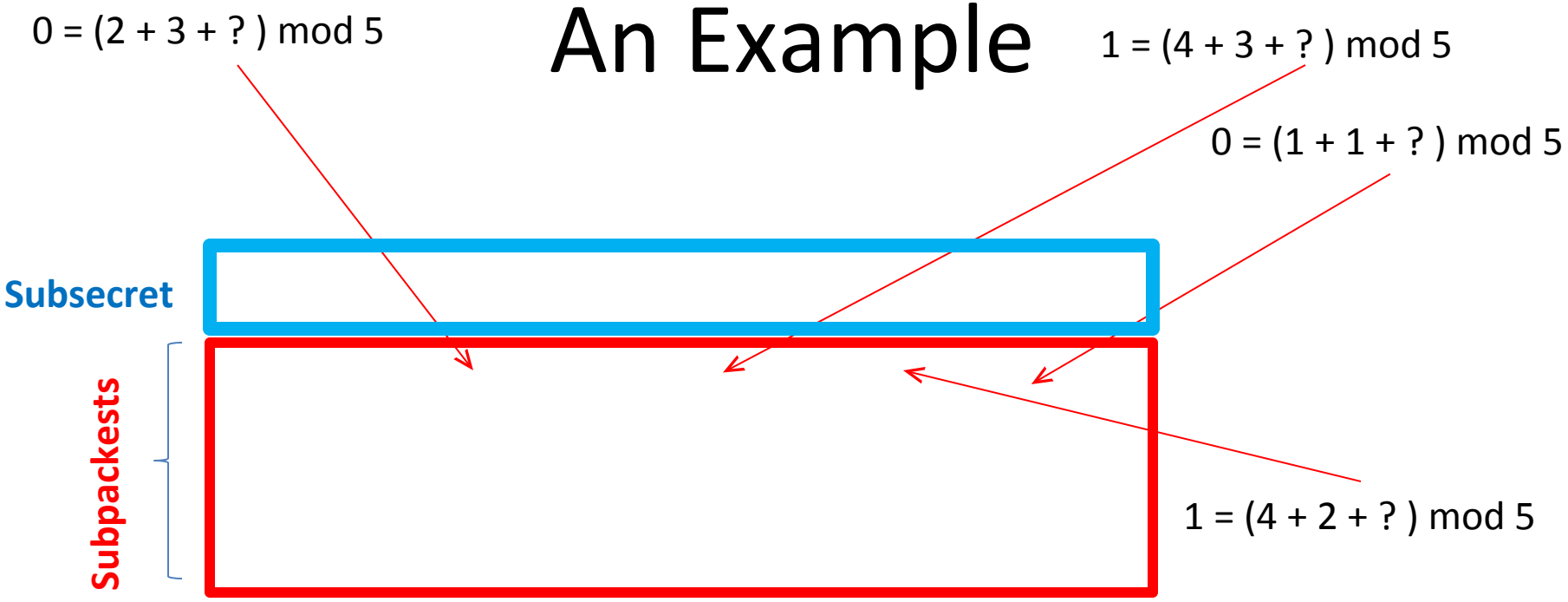
- Goal:
  - Keeping a subsecrete private while participating in group aggregation
  - A method to allow each provider to package its sub-secrete into c shares such that even obtaining c-1 shares, one cannot construct the sub-secrete.
- Two system defined parameters
  - q is the modulus with q ≫ n
  - c indicates the number of sub-packets used to represent a sub-secret.

$$v_i = f(u_{i,0}, u_{i,1}, \ldots, u_{i,c-1}) = \sum_{j=0}^{c-1} u_{i,j} \mod q$$

- How
  - The packet-generating process generates (c, c)-secret packets: given any less than c sub-packets, the sub-secret v_i is still completely undeterminable.
  - One implementation
    - The first c-1 sub-packets are randomly selected from the domain of [0,q-1]
    - The last sub-packet is computed by

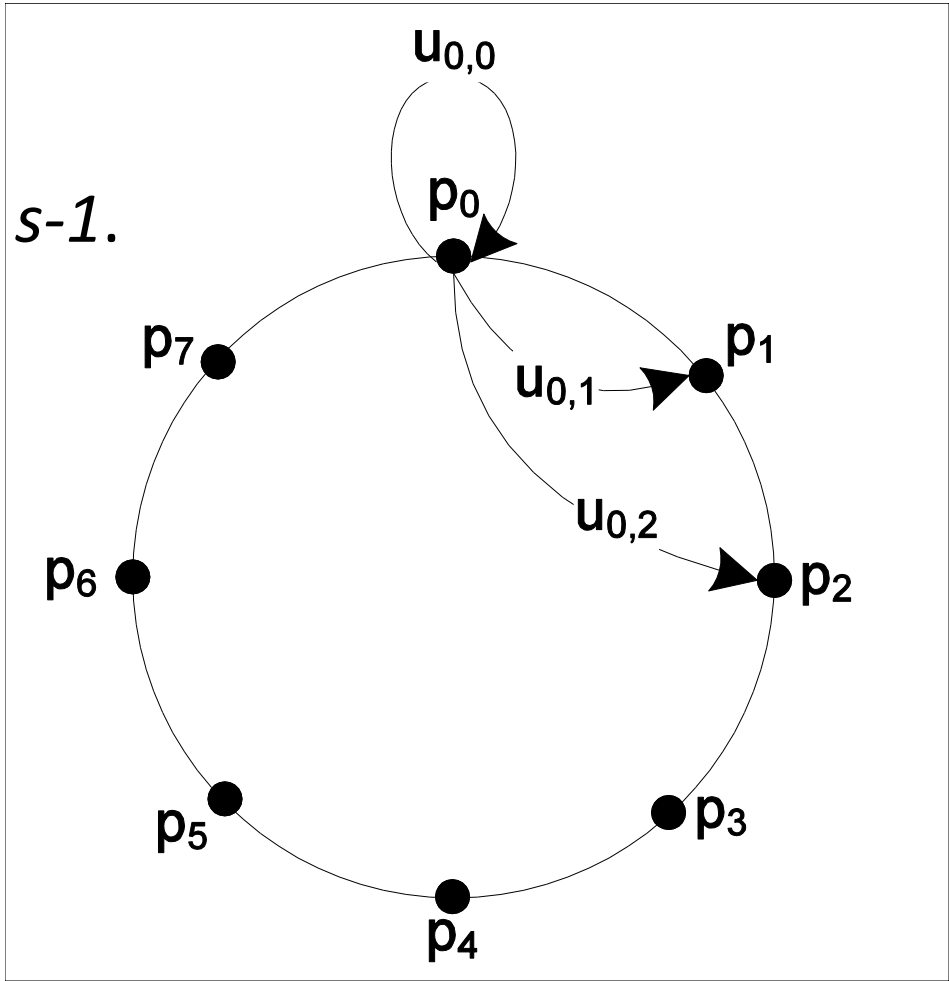
$$u_{i,c} = \left(v_i - \sum_{j=1}^{c-1} u_{i,j}\right) \mod q.$$

An Example

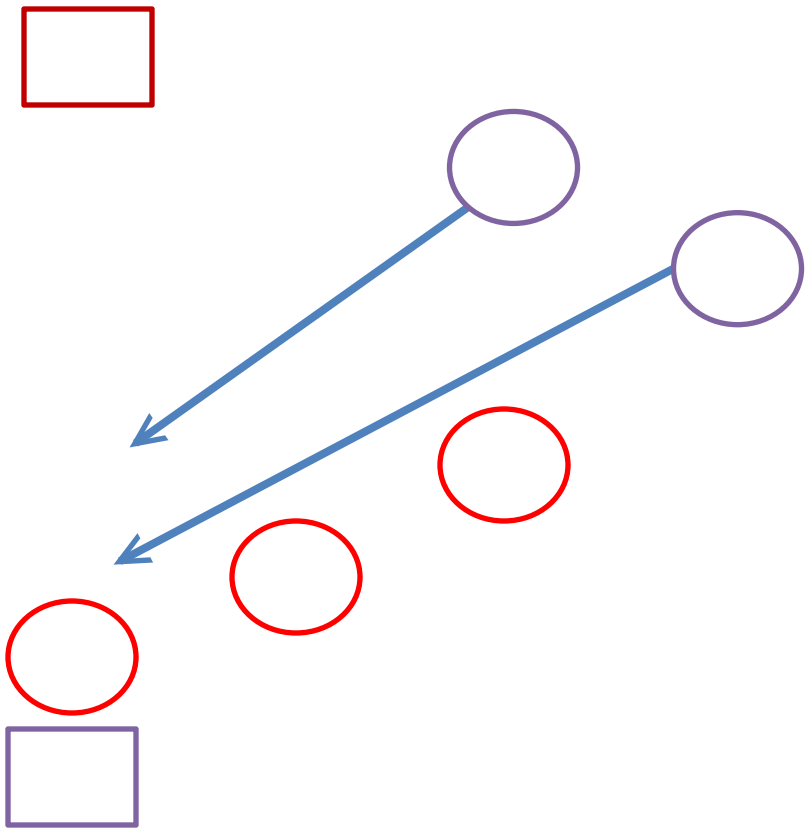


Distributing Sub-packets

- Parameter c indicates number of sub-packets generated for one sub-secret.
- Providers are organized in a ring, numbering from *0* to *s-1*.
- Provider p_i generating *c* sub-packets keeps one packet and sends the j-th sub-packet to the next (j-1)-th provider p_{i+j-1}.



Illustrating the 4 steps: An Example



Protocol Complexity

- Time: Constant number of rounds
  - 2 rounds
  - Linear to the size of the group

- Bandwidth:
  - O(n*c)

    n: group size

    c: # shares

Security Analysis

- Network eavesdropper:
  - Computational security provided by secure channel.

- Colluding providers
  - Information the
    from inspecting
  - Resistant to up-
  - For >(2c-2) collu
    privacy to be preser

By contrast, probabilistic PPI (VLDB'03)

1. Can't survive even when there are only 2 colluding providers
2. Need run >10 rounds to guarantee correctness.

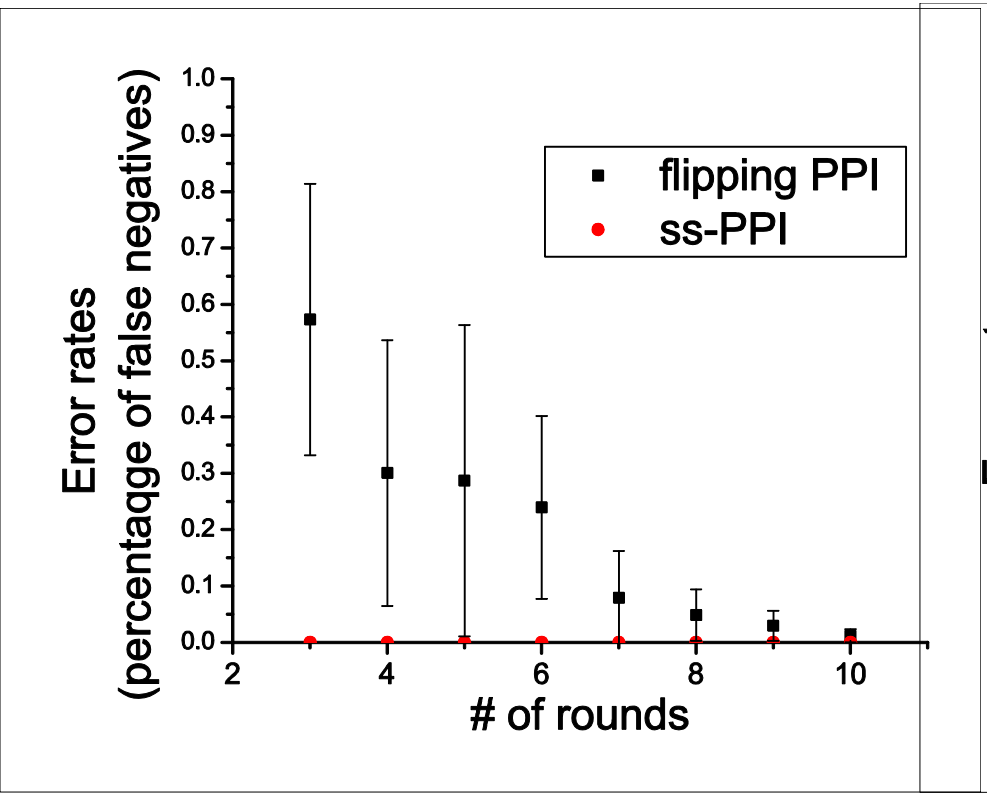
Experiments

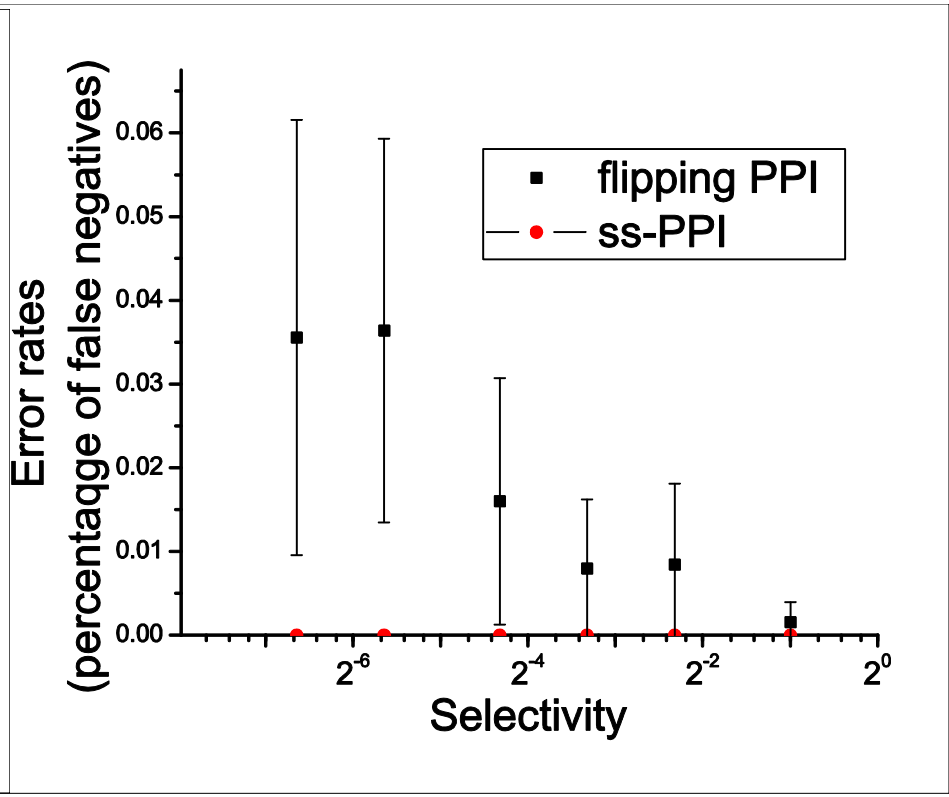
- Simulation based experiments
  - Compared against probabilistic PPI [bawa et.al VLDB03].
  - Synthetic dataset of 100,000 (10^5) providers in 1000 groups.
  - Content model by a p2p dataset[16], based on TREC WT10g collection.

- Evaluation is conducted in terms of
  - Privacy level
  - Index construction efficiency
  - Correctness

Correctness

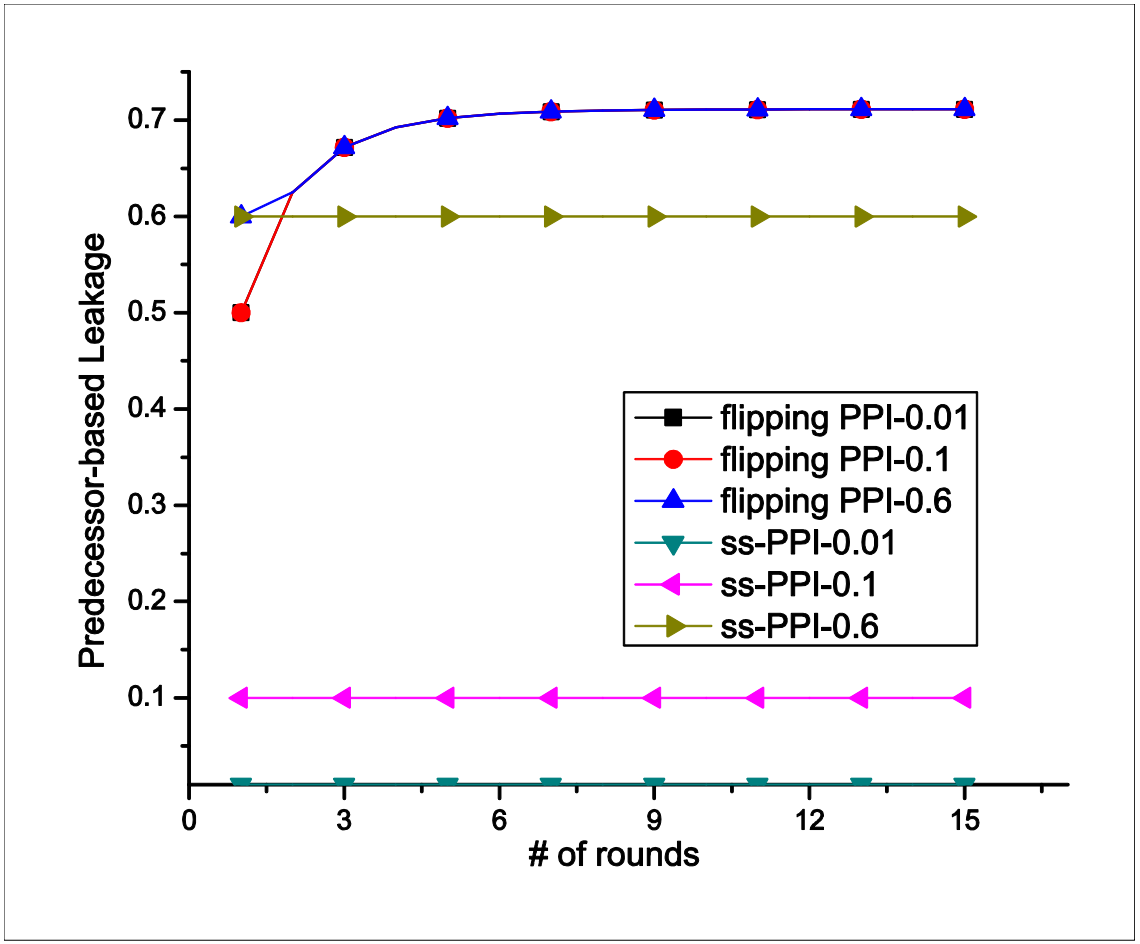
- Our ss-PPI always achieve 100% accuracy.



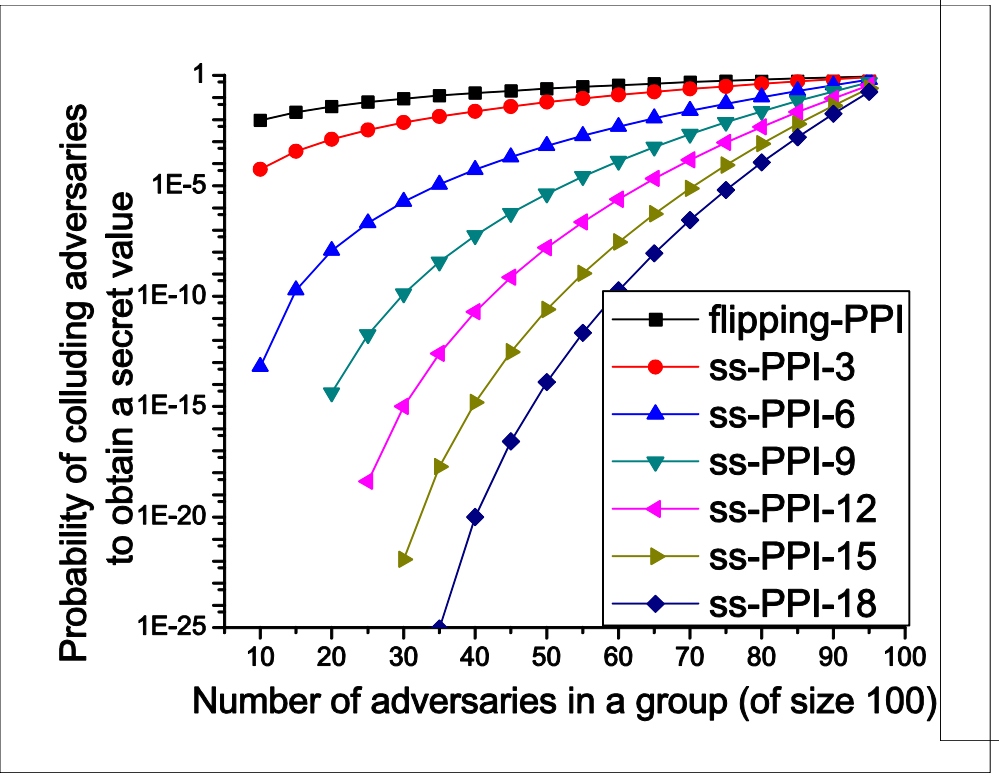


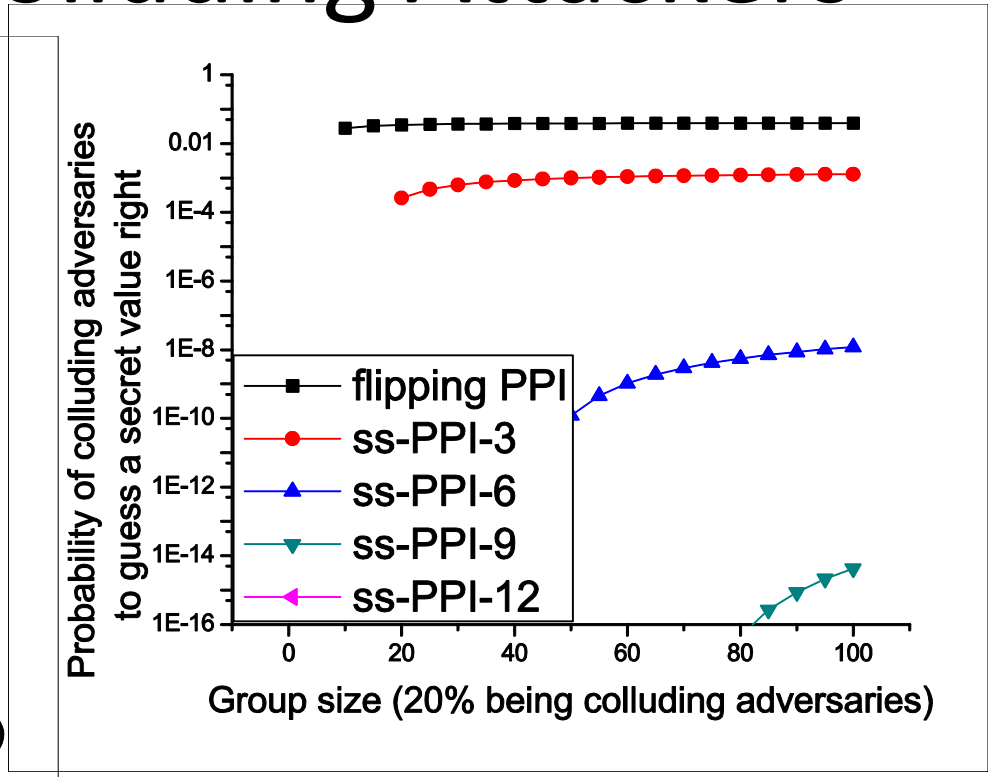
Privacy against non-colluding adversary

- Different selectivity is tested against leakage
  - ss-PPI does not expose any information other than that in priori (selectivity at 1%, 10%, 60%).
  - while the flipping-PPI leads to privacy leakage up to 71%.



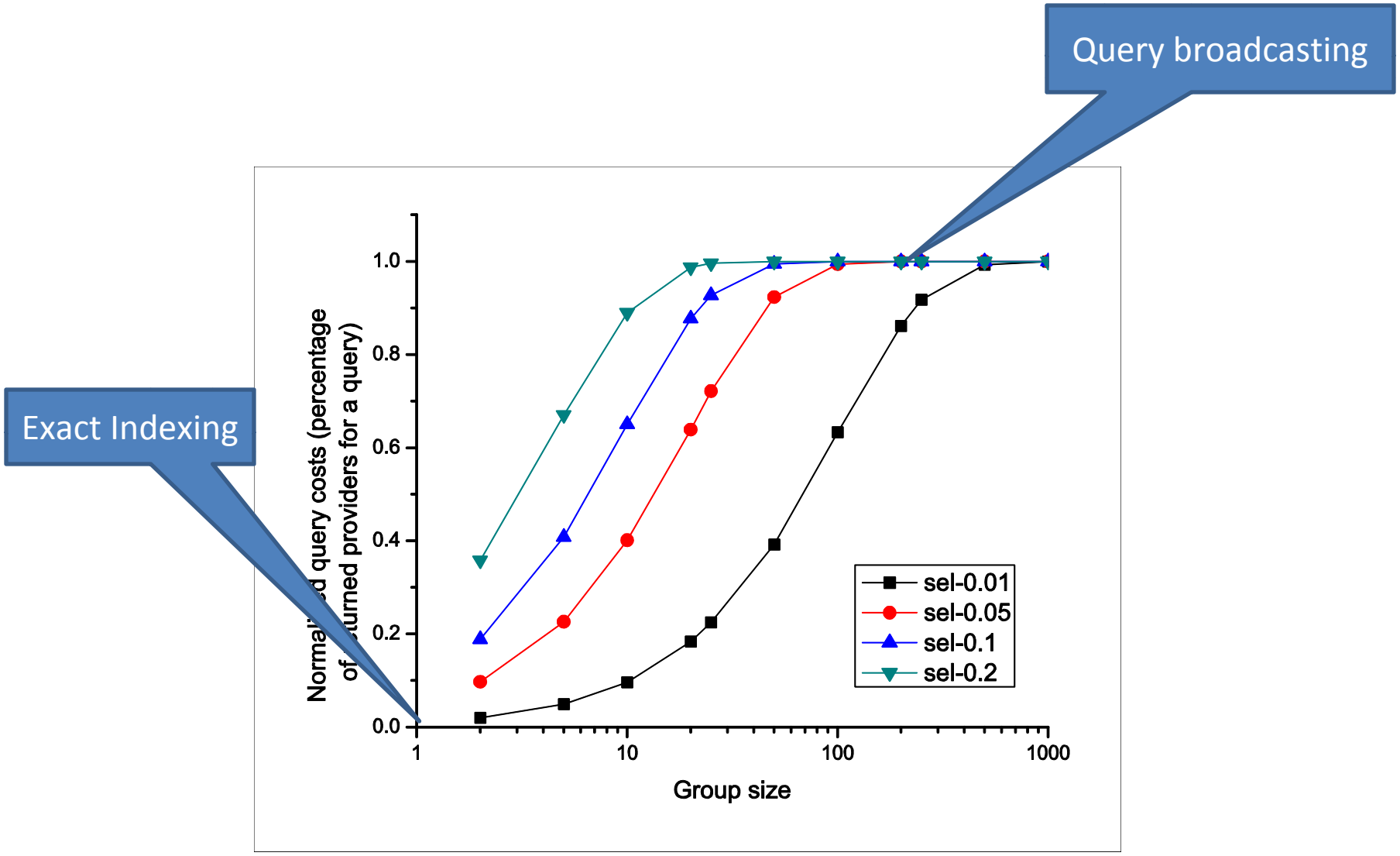
Privacy against Colluding Attackers





- Left figure: varying the number of colluding adversaries and fixed group size of 100 and the adversaries accounting for 20% of the group.
- There exist a threshold on number of adversaries under which SS-PPI's leaking probability is 0. For example, in the plot, the curve for SS-PPI-18 shows up only when adversaries are more than 35.
- Right figure: SS-PPI slightly increase privacy leaking probability until certain converging value as group size goes up, while flipping PPI stays constant.
- Note that the probability (y axis) is plotted in log scale, the differences between two protocols in comparison are significant.

Query processing costs



Conclusion

- Proposed ss-PPI to ensure that the search does not reveal the specific association between contents and providers (a.k.a. content privacy).

- *ss-PPI outperforms existing approaches*
  - *High privacy guarantee against collusion attacks*
  - *Fast PPI construction algorithm*
  - *Search efficiency.*

Thanks!
QA

Preliminary: Secret Sharing

- In a (n,k) secret sharing scheme, a secret s is decomposed into n shares.
  - Secrecy: given any k' shares with k' < k, one can not obtain any valid information on the value (distribution) of secret *s*.
  - Recoverability: Given k' shares (k' >= k), any one can reconstruct the exact value of secret *s*.
- Additive homomorphism
  - Sub-secret s, t are respectively decomposed to sub-shares r1(s), r2(s), … rn(s), and r1(t), r2(t), … rn(t).
  - For any i in [1,n], $ri(s + t) = ri(s) + ri(t)$
  - Shares of super-secret(s+t) equal to sum of shares of sub-secret.