Analysing Access Pattern and Volume Leakage from Range Queries on Encrypted Data

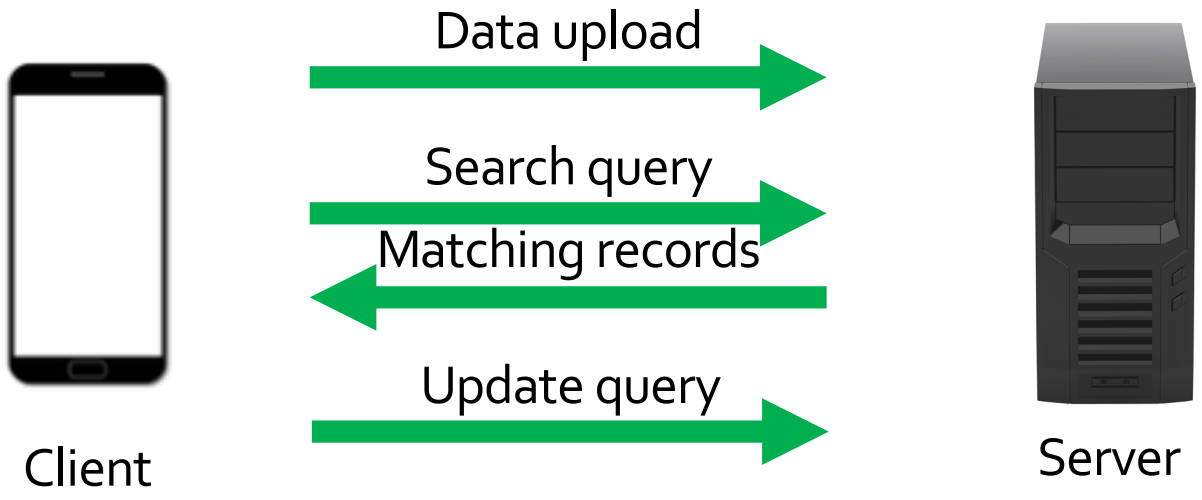
Kenny Paterson @kennyog
based on joint work with
Paul Grubbs, Marie-Sarah Lacharité, Brice Minaud

Information Security Group

ROYAL HOLLOWAY UNIVERSITY OF LONDON

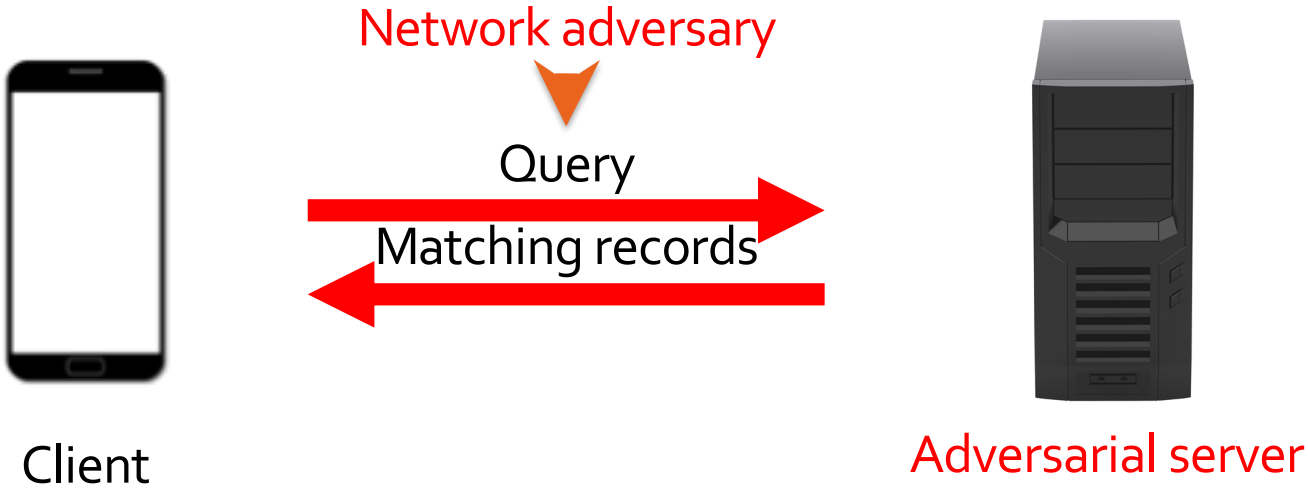
SuRI – EPFL – June 18, 2018

Outsourcing Data to the Cloud



- For **encrypted database systems**:
  - Data = collection of records in a database (e.g. health records).
  - Query examples =
    - Find records with a given value (e.g. patients aged 57).
    - Find records within a given range (e.g. patients aged 55 to 65).
    - …

Security of Data Outsourcing Solutions



- **Adversaries:**
  - **Network** adversary = observes traffic on network.
  - **Snapshot** adversary = breaks into server, gets snapshot of memory.
  - **Persistent** adversary = corrupts the server for a period of time; sees all communication transcripts. Can be server itself.
- **Security goal = privacy:**

  Adversary learns as little as possible about the client's data and queries.

State of the Art

- **Network attacker apparently easy to defeat using network encryption, e.g. TLS.**
- **For snapshot and persistent attackers: no perfect solution.**

  Every solution is a trade-off between **functionality** and **security**.
- **Huge amount of literature.**

  [AKSX04], [BCLO09], [PKV+14] , [BLR+15], [NKW15], [K15], [CLWW16], [KKNO16] , [RACY16], [LW16] …
- **A few "complete" solutions**:

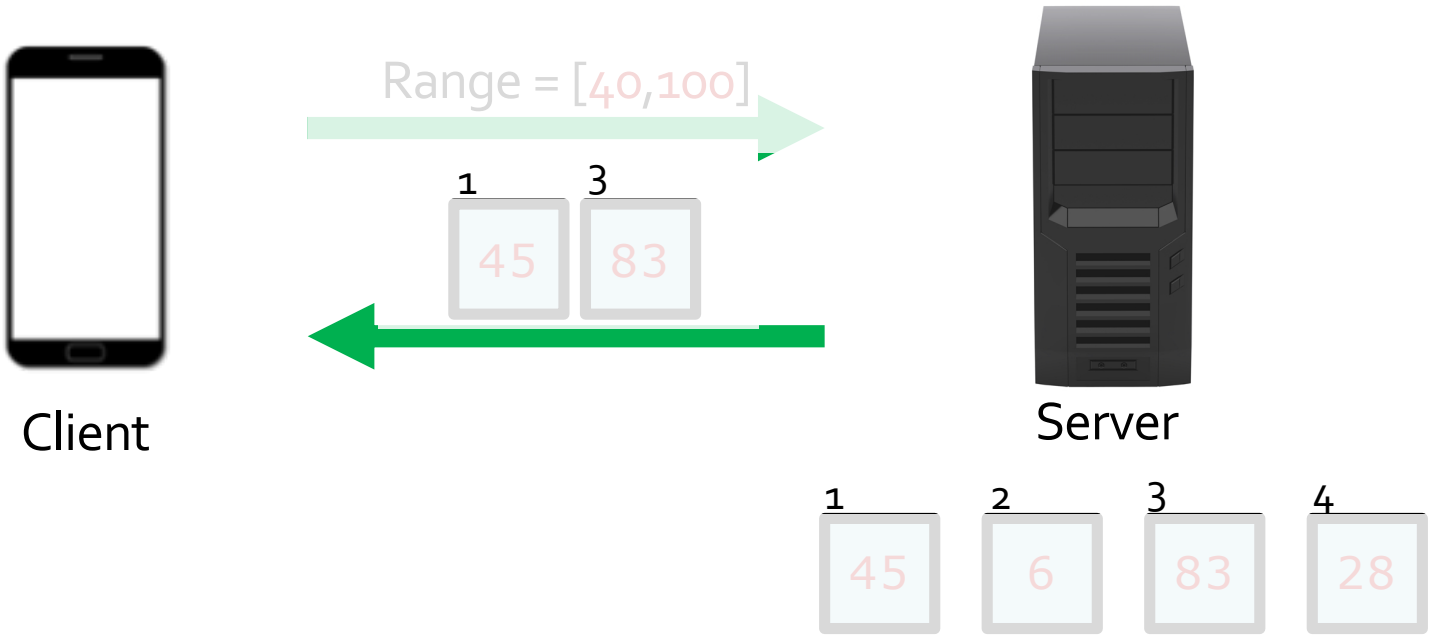
  Mylar (for web apps)

  CryptDB (handles most of SQL)

  ⚠ *Controversial!*

  ➔ Cipherbase (Microsoft), Encrypted BigQuery (Google), …
- **Very active area of research.**

Setting for this Talk: Schemes Supporting Range Queries



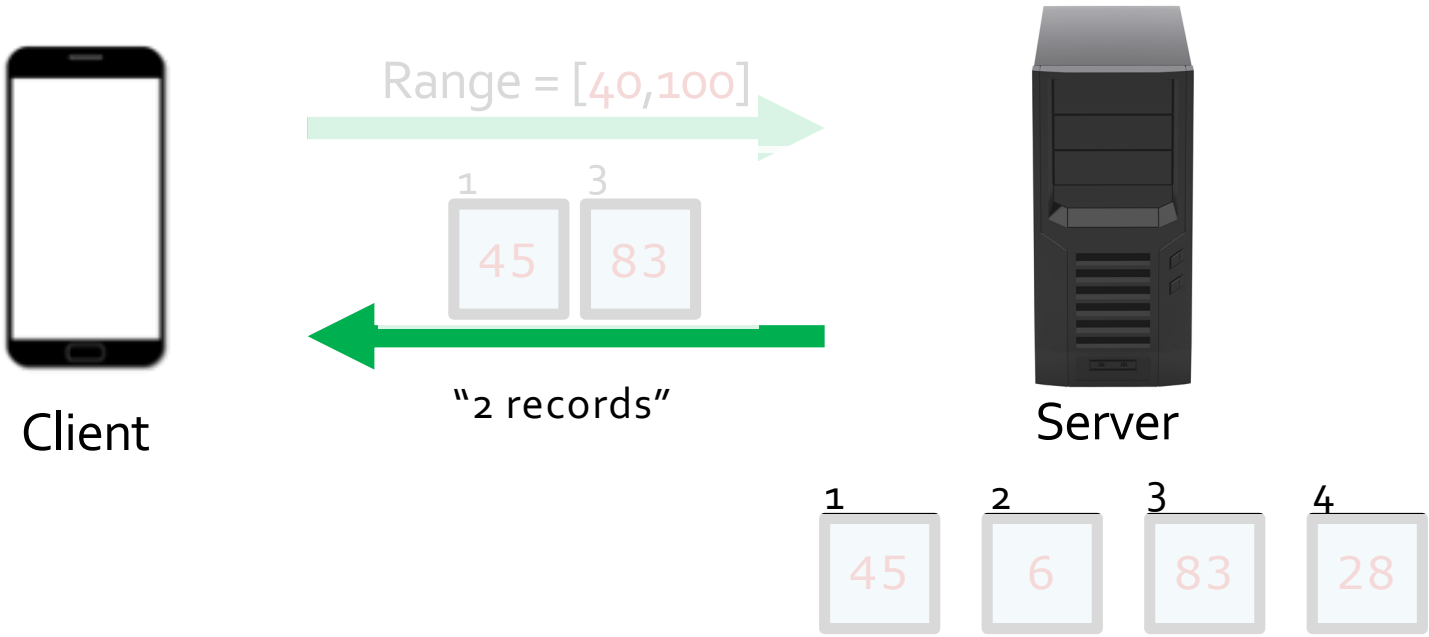
- All known schemes leak to **the server** the set of matching records = **access pattern**.

  OPE, ORE schemes, POPE, [HK16], Blind seer, [Lu12], [FJKNRS15],…

- Some schemes also leak # records below queried range endpoints = **rank**.

  FH-OPE, Lewi-Wu, Arx, Cipherbase, EncKV,…

Setting for this Talk: Schemes Supporting Range Queries



- Could hide access pattern from server by using ORAM (at huge cost).
- But volume of responses (number of records) would still leak to server.
- Volume would also leak to network adversary unless traffic padding mechanisms were used; these are rare in practice (cf. AES-GCM in TLS).
- Motivates consideration of **volume attacks**.

Exploiting Leakage

- Most schemes prove that nothing more leaks than their leakage model allows.
- For example, leakage = volume, access pattern, or access pattern + rank.
- *What can we really learn from this leakage?*

Our goals:

- Volume leakage only: **distribution** reconstruction (DR) = recover the number of times each value occurs in the database.
- Access pattern (+ rank): **full** reconstruction = recover the exact value for every record.

Exploiting Leakage – State of the Art

[KKNO16]: If N denotes the number of distinct data items, then:

- $O(N^2 \log N)$ queries suffice for full reconstruction, using only access pattern leakage.
- $O(N^4 \log N)$ queries suffice for distribution reconstruction, using only volume leakage.

(NB: In both cases, because of inherent symmetry, only reconstruction up to reflection is possible.)

Exploiting Leakage – Highlights of Our Results

[LMP18] (eprint 2017/701; S&P18):

- O(N log N) queries suffice for full reconstruction, using only access pattern leakage.
  - where N is the number of possible values (e.g. 125 for age in years).
  - provided data is **dense** (every value occurs at least once).

[GLMP]:

- $O(N^2 \log N)$ queries suffice for distribution reconstruction, using only volume leakage.
  - provided the number of records R is larger than about $N^2/2$.

Attacks from Access Pattern Leakage [LMP18]

Assumptions for Analysis

1. Data is **dense:** all values appear in at least one record.

   Can be relaxed in some of our attacks.

2. Range queries are **uniformly distributed.**

   Our algorithms don't actually care though – the assumption is only used for computing upper bounds on required number of queries.

Main Results from [LMP18]

1. **Full reconstruction** with O(N logN) queries from access pattern leakage
   - in fact, $N \cdot (3 + \log N)$.
2. **Approximate reconstruction** with relative accuracy ε with $O(N \cdot (\log 1/\varepsilon))$ queries.
3. **Approximate reconstruction** using an *auxiliary distribution* and rank leakage.
   - more efficient in practice, evaluation via simulation.

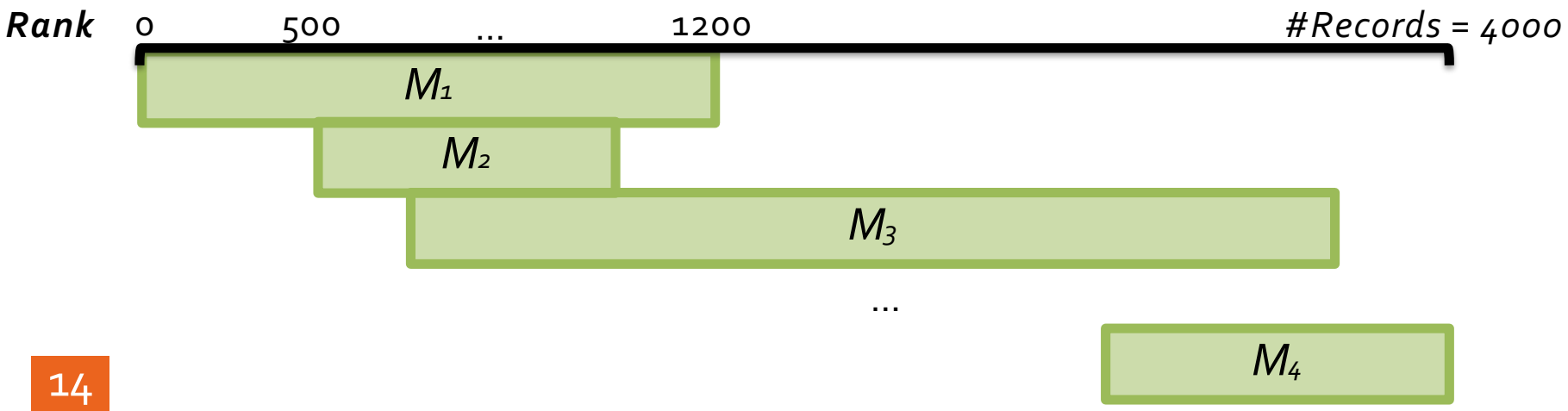
Attack 1: Full Reconstruction

Full Reconstruction **with Rank Leakage**

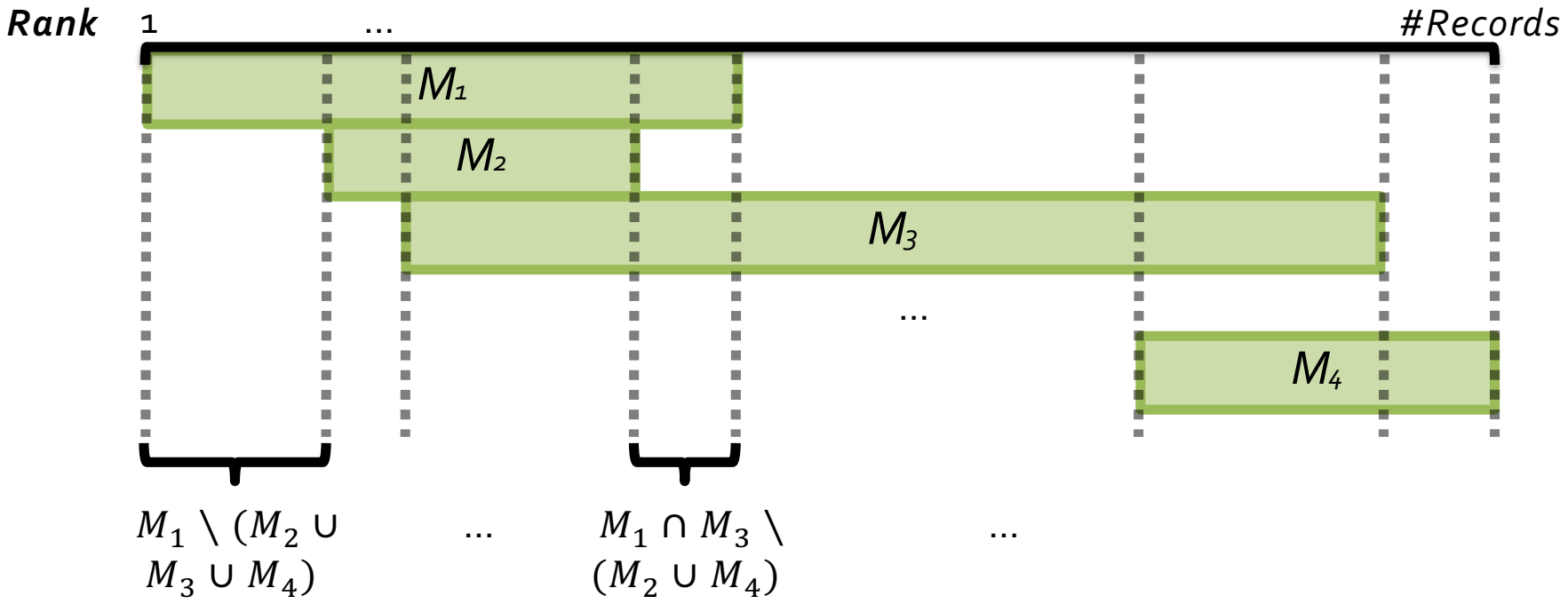
- Adversary is observing query leakage…

(Reordered for convenience)

Hidden	Leaked		
Query [x,y]	a = rank(x-1)	b = rank(y)	Matching IDs
[1,18]	0	1200	M_1
[2,10]	500	800	M_2
[7,98]	600	3000	M_3
[55,125]	2000	4000	M_4



Full Reconstruction with Rank Leakage



- Order sets by rank.
- Partition records into smallest possible sets using access pattern leakage.
- If this partitions records into N sets, **win**! Just match minimal sets with values.

Full Reconstruction with Rank Leakage

- Expected number of queries **sufficient** for full reconstruction is at most:

$$N \cdot (2 + \log N) \quad \text{for } N \geq 27.$$

  - Essentially a coupon collector's problem.
- Expected number of **necessary** queries is at least:

$$1/2 \cdot N \cdot \log N - O(N)$$

  for *any* algorithm.
- This algorithm is "data-optimal", i.e. it fails iff full reconstruction is impossible for *any* algorithm given the input data.

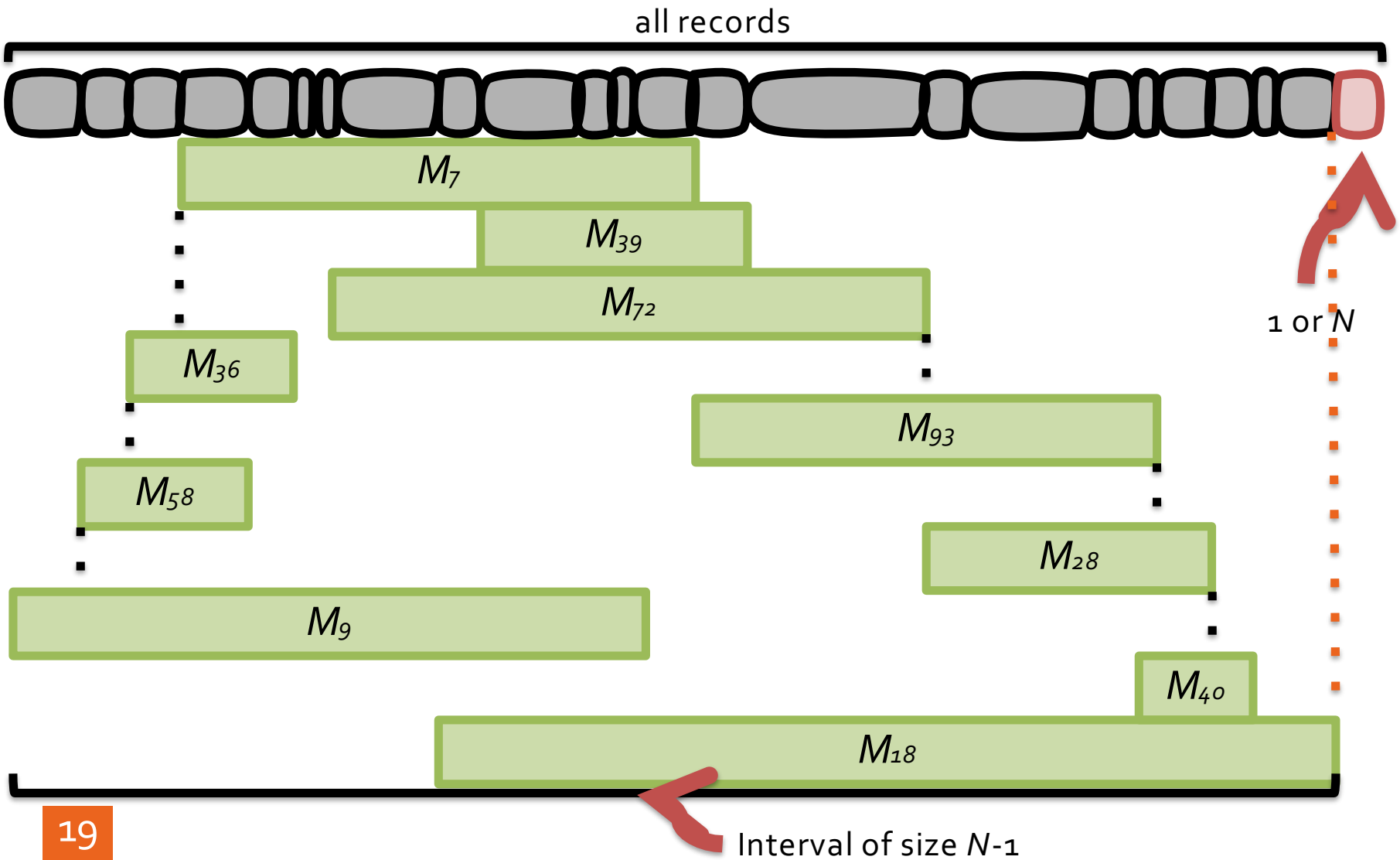
Full Reconstruction without Rank Leakage

- **More general setting:** now use only access pattern leakage.
- **Partition** (as before), then **sort** (see slides ahead).
- Expected number of **sufficient** queries is at most:
  $N \cdot (3 + \log N)$ for $N \geq 26$

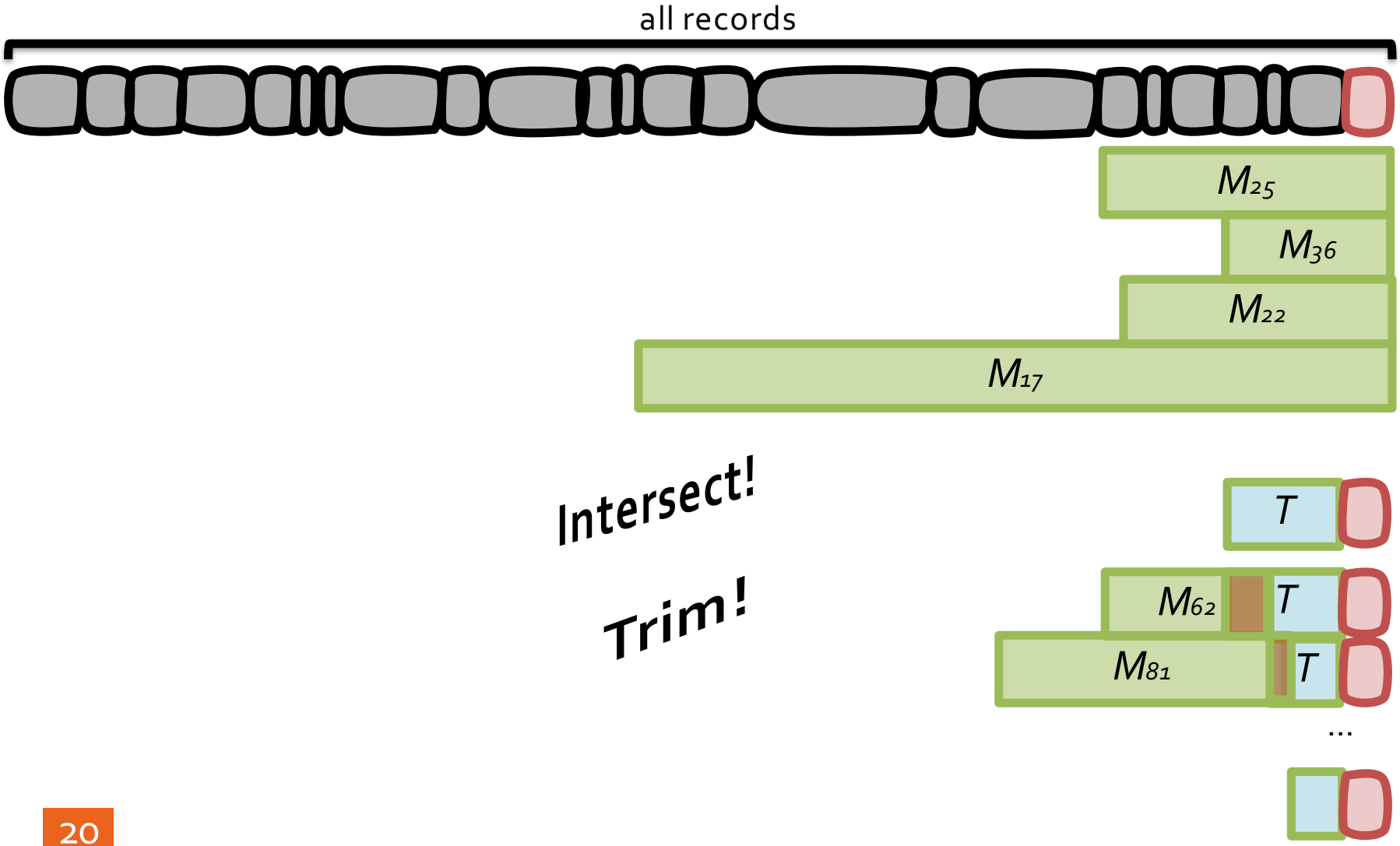
  - i.e. new sorting step is very cheap in terms of data.
- Expected number of **necessary** queries is at least:
  $1/2 \cdot N \cdot \log N - O(N)$

  for any algorithm.
- Still data-optimal!

Full Reconstruction (without Rank Leakage): Sorting Step

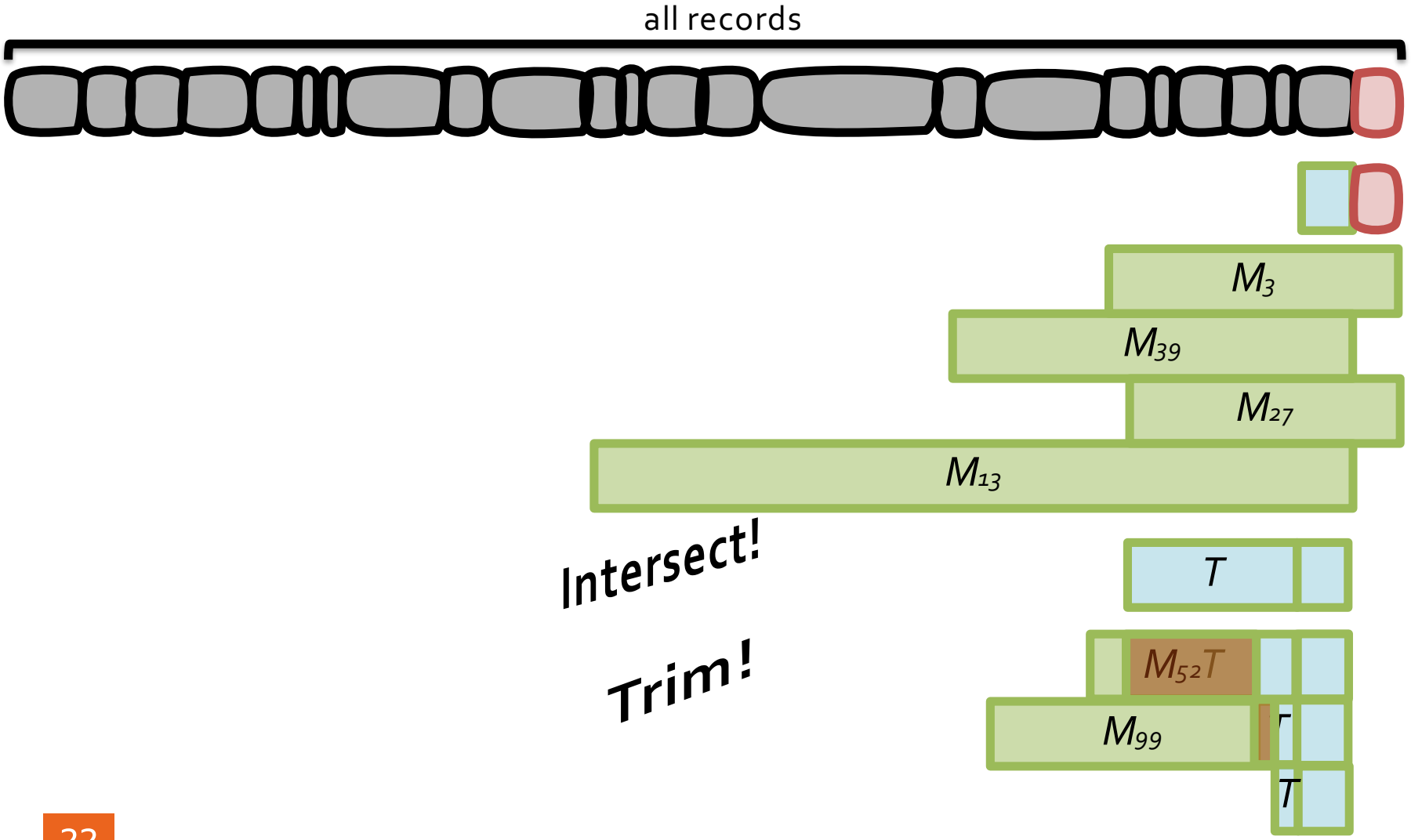


Full Reconstruction (without Rank Leakage): Sorting Step – Extending

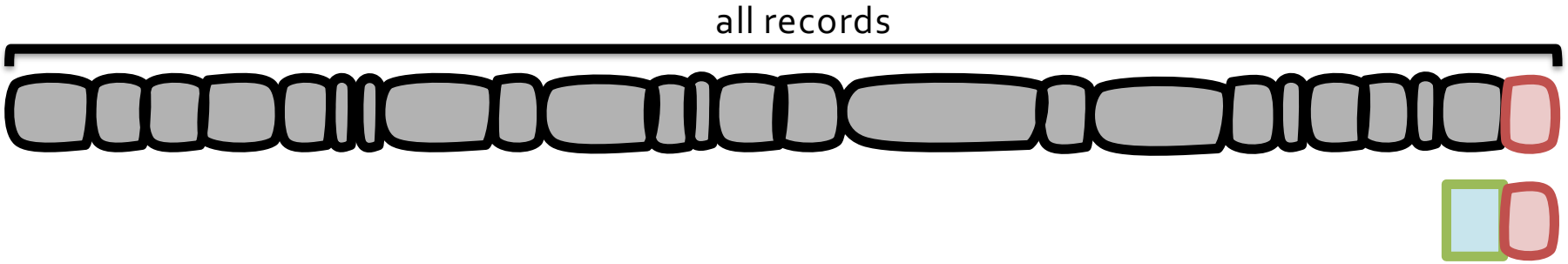


Full Reconstruction (without Rank Leakage): Sorting Step – Extending

Full Reconstruction (without Rank Leakage): Sorting Step



Full Reconstruction (without Rank Leakage): Sorting Step



Full Reconstruction (without Rank Leakage): Proof Intuition

- Hard part is to show that O(N log N) queries suffice, with a small constant.
- Proof consists of showing that **if** certain favourable range queries are made, then partitioning succeeds in constructing N classes, and sorting succeeds in ordering them.
- Coupon collecting bounds then establish that O(N log N) queries are enough.

Attack 3: Reconstruction with Auxiliary Data

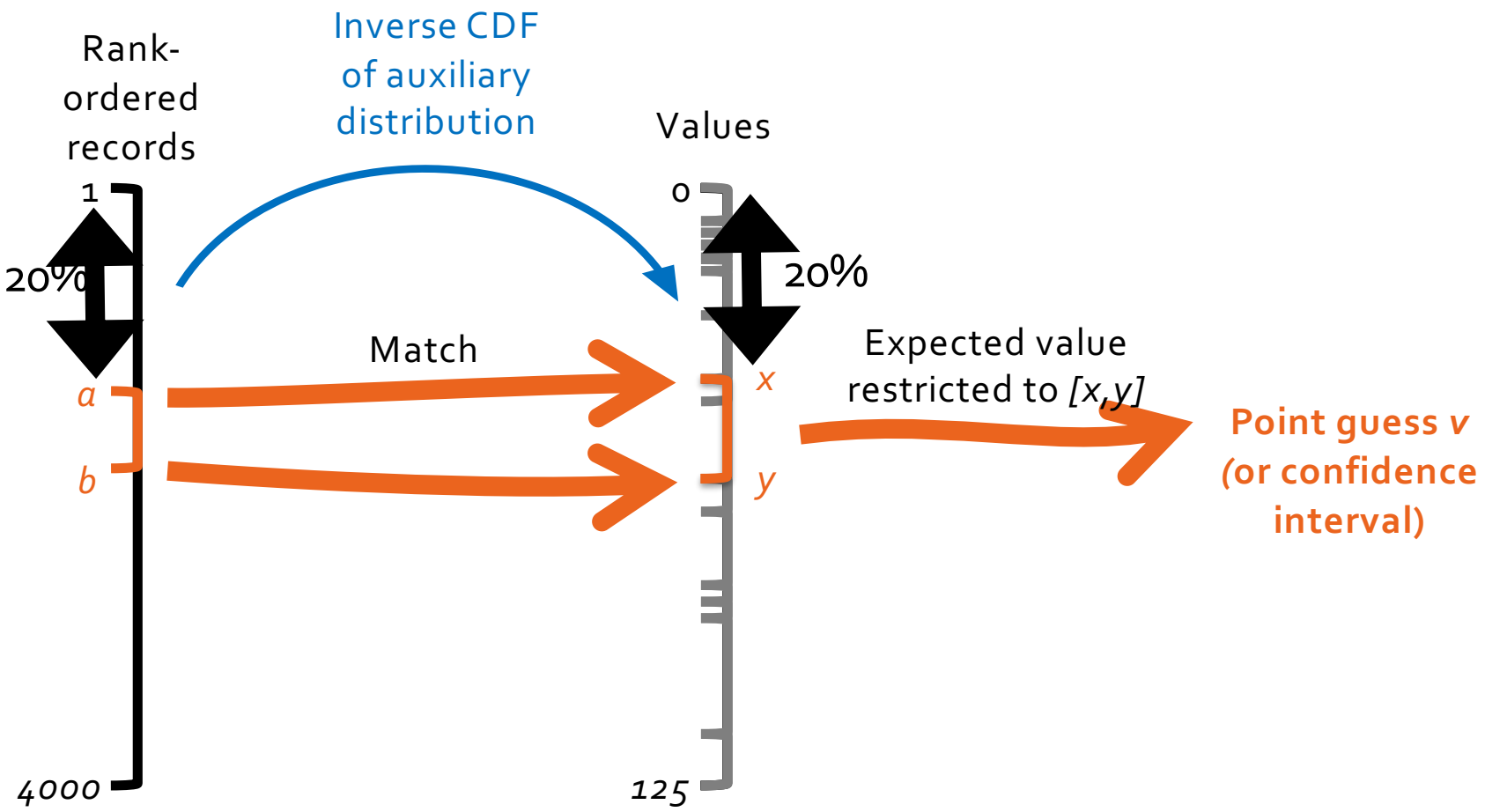
Reconstruction with Auxiliary Data and Rank Leakage

- As before, queries have ranges chosen uniformly at random.
- Assume access pattern and rank are leaked.
- We now also assume that an **approximation to the distribution on values** is known.

  "Auxiliary distribution".

  From aggregate data, or from another reference source.
- We show experimentally that, under these assumptions, **far fewer queries** are needed.

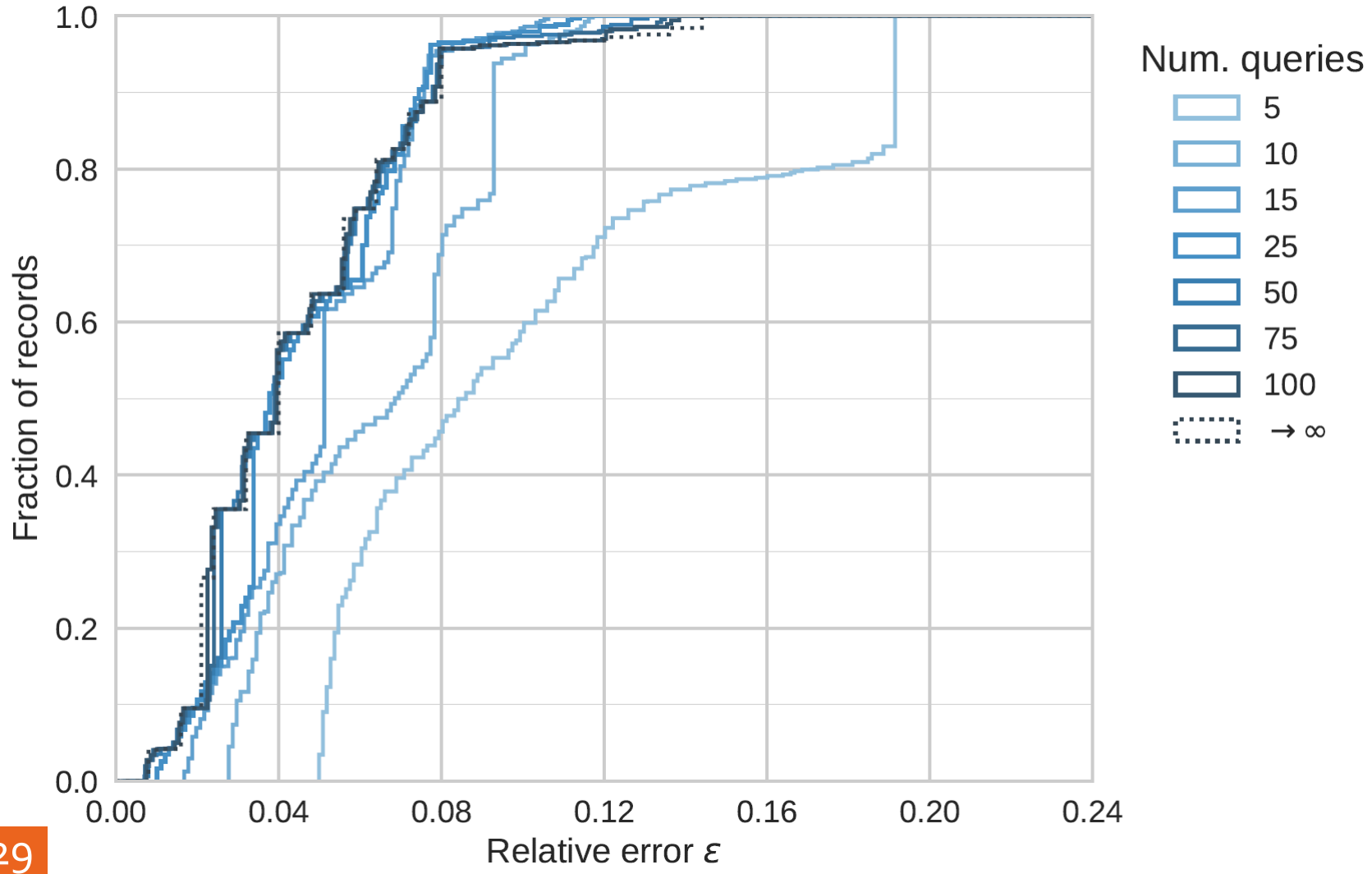
Auxiliary Data Attack: Estimating Step



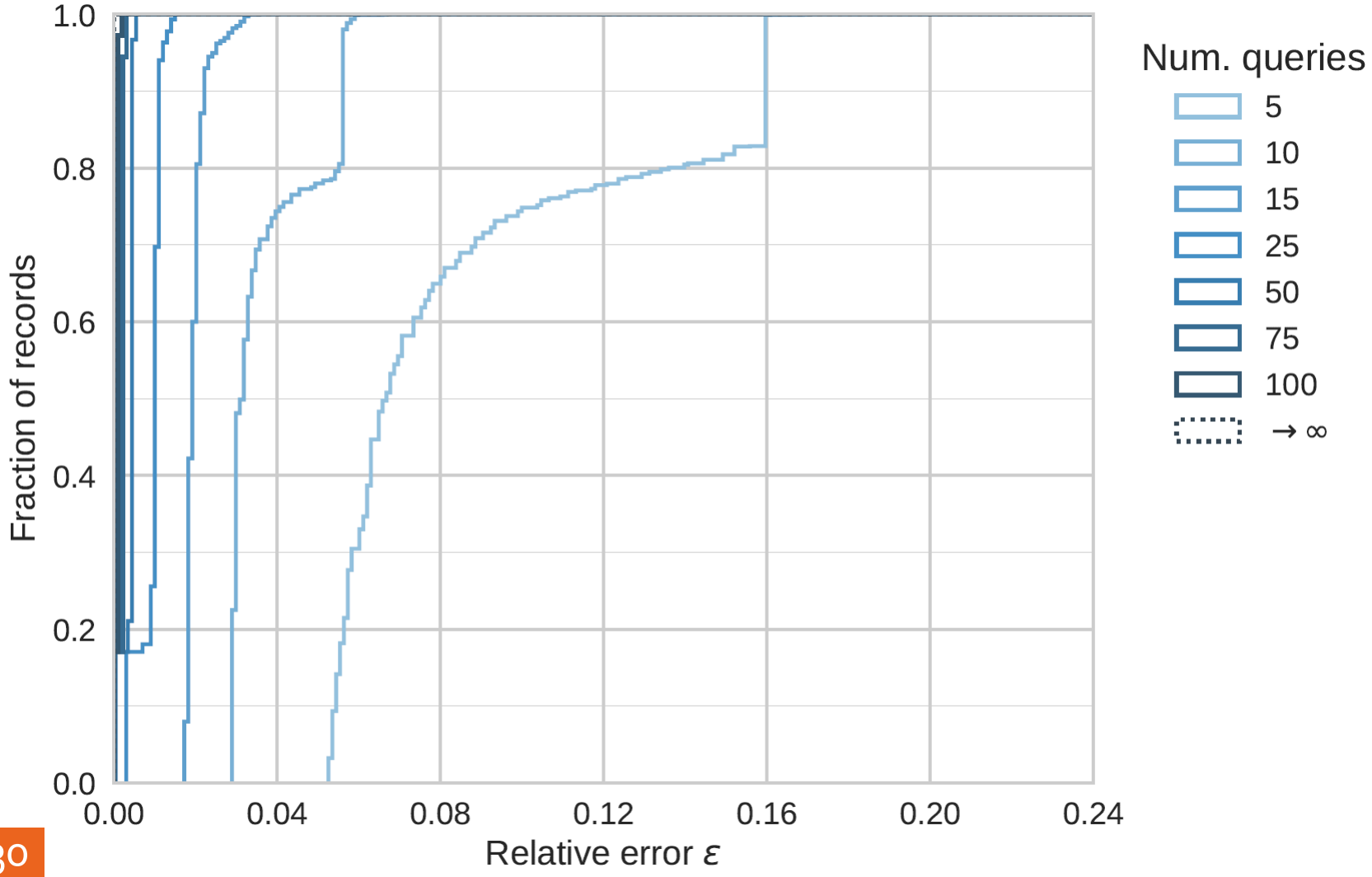
Auxiliary Data Attack: Experimental Evaluation

- Ages, N = 125 (0 to 124).
- Health records from US hospitals (NIS HCUP 2009).
- **Target:** age of individual hospitals' records.
- **Auxiliary data:** aggregate of 200 hospitals' records.
- **Measure of success:** proportion of records with value guessed within ε.

Auxiliary Data Attack: Results for Typical Target Hospital



Auxiliary Data Attack: Results with Perfect Auxiliary Distribution



Summary of Attacks from [LMP18]

Full reconstruction in ≈N log N queries with only access pattern.

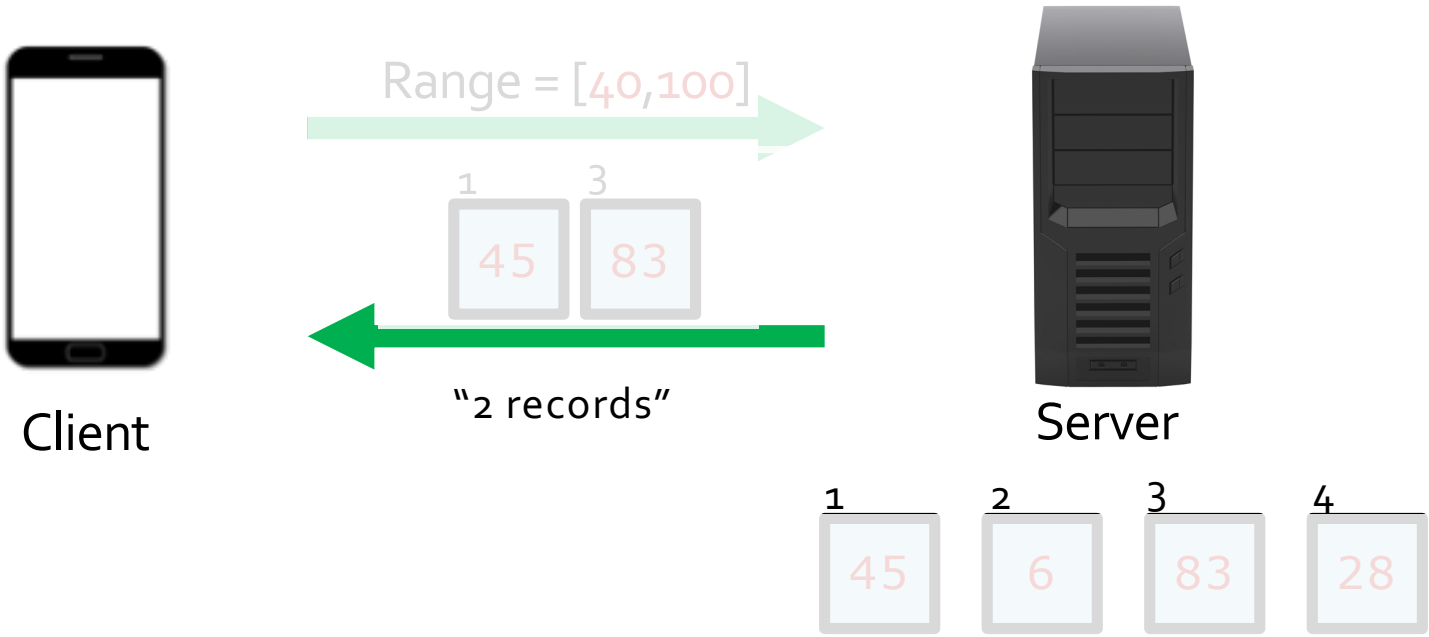
Efficient, data-optimal algorithms + matching lower bound.

Attack	Req'd leakage	Other req'ts	Suff. # queries
KKNO16	AP	Density	$O(N^2 \log N)$
Full	AP + rank	Density	$N \cdot (\log N + 2)$
	AP	Density	$N \cdot (\log N + 3)$
ε-approximate	AP	Density	$5/4\ N \cdot (\log 1/\varepsilon) + O(N)$
Auxiliary	AP + rank	Auxiliary dist.	Experimental

- For N = 125, about 800 queries suffice for full reconstruction!
- If an auxiliary distribution + rank leakage is available, after only 25 queries, 55% of records can be reconstructed to within 5 years.

Attacks based on Volume Leakage

Volume Leakage



- Now only volume of responses (number of records) leaks to server or network adversary.
- Much tougher attack setting.
- Target is **distribution reconstruction**: how many records have each value.

Exploiting Volume Leakage – State of the Art

[KKNO16]:

- $O(N^4 \log N)$ queries suffice for distribution reconstruction, using volume leakage.
- Two attacks: polynomial factorisation and heuristic assignment algorithm.
- Complexity of former scales badly with N.
- Both attacks rely heavily on assumption that range queries are **uniformly random**, and fail badly if there is any deviation from this assumption.
- [KKNO16] also show that $\Omega(N^4)$ queries are **required** for certain pathological distributions.

Exploiting Volume Leakage – Main Results from [GLMP]

- **Distribution Reconstruction** from volume leakage, provided R, the number of records is large enough (about N^2).
  - Attack only needs to see each query once.
  - It then needs $O(N^2 \log N)$ queries under a uniform query assumption; more generally, the coupon-collector number for the query distribution.
- Subsequent recovery of value of any new record added to the database using volume leakage from O(N) queries .
- **Online query reconstruction** using an *auxiliary distribution* (or the distribution recovered in the first attack).

Distribution Reconstruction from Volume Leakage

Distribution Reconstruction from Volume Leakage

- Adversary is observing volume leakage…

Hidden				Leaked
Query [x,y]	a = rank(x-1)	b = rank(y)	Matching IDs	Volume
[1,18]	0	1200	M_1	1200
[2,10]	500	800	M_2	300
[7,98]	600	3000	M_3	2400
[55,125]	2000	4000	M_4	2000

Key considerations:

- For uniformly random range queries, after $O(N^2 \log N)$ queries, all volumes will have been observed.
- This set of volumes has a lot of additive structure.

Distribution Reconstruction from Volume Leakage

- Suppose enough queries have been made that all possible volumes have been observed ($O(N^2 \log N)$ queries for uniform distribution).
- Can deduce R, the total number of records (it's the largest volume).
- Consider volumes for the set of ranges [1,1], [1,2],….[1,N]: **elementary ranges/volumes**.
  - If we can identify these, then DR becomes easy: just do pairwise subtractions.
- On the other hand, the elementary volumes are very special:
  - They are complemented: if V is elementary, then R-V must also be a volume.
  - Every volume arises as an elementary volume or the difference of two elementary volumes: Vol([i,j]) = Vol([1,j]) – Vol([1,i]).
  - So the (absolute value of the) difference of elementary volumes is always a volume.

Distribution Reconstruction by Clique Finding

Let's build a graph!

- Vertices are identified with complemented volumes (includes elementary volumes but maybe more).
- Add an edge between two vertices if the difference in volumes of vertices is also a volume.
- Recall: "The (absolute value of the) difference of elementary volumes is always a volume".
- This implies that the set of elementary ranges forms an N-clique in the graph.
- **Basic idea**: build the graph and use your favourite clique-finding algorithm to identify an N-clique!
- (But clique-finding is hard in general - NP-complete.)
- (And there may be many additional vertices and edges in the graph not arising from elementary volumes.)

Distribution Reconstruction by Clique Finding

Graph pre-processing:

- Certain vertices and edges **must** be in the N-clique: any volumes occurring at a single edge/vertex.
- Certain vertices **cannot** be in the clique: vertices not connected to **all** of these necessary vertices by an edge.
- **Iterate** based on these two properties, maximum $O(N^2)$ iterations.
- **Bootstrapping**: smallest complemented volume **must** be in clique, as must largest volume R (corresponding to range [1,N]).
- Our experiments with real databases show that, very often, preprocessing finds the required clique (or its symmetric complement).
  - Doing actual clique-finding is redundant in these cases!

Example of Distribution Reconstruction by Preprocessing

Example: N=4, R=20, record values:

1,1,1,2,2,2,2,2,3,3,3,3,3,3, 3,3,3,3,3,4

Elementary volumes:

[1,1]: 3
[1,2]: 8
[1,3]: 19
[1,4]: 20

Other volumes:

[2,2]: 5
[2,3]: 16
[2,4]: 17
[3,3]: 11
[3,4]: 12
[4,4]: 1

Volume leakage: {1,3,5,8,11,12,16,17,19,20}

Example of Distribution Reconstruction by Preprocessing

Volume leakage:
{1,3,5,8,11,12,16,17,19,20}

Complemented volumes give initial vertex set:

{1, 3, 8, 12, 17, 19, 20*}

x 5 (15 not a volume)
x 11 (9 not a volume)
x 16 (4 not a volume)

*included by definition; complement is 0.

Bootstrapping:

1 and 20 must be in the clique (smallest complemented volume, largest volume).
(1,3) is not an edge – eliminate 3;
(1,8) is not an edge – eliminate 8;
(1,19) is not an edge – eliminate 19.

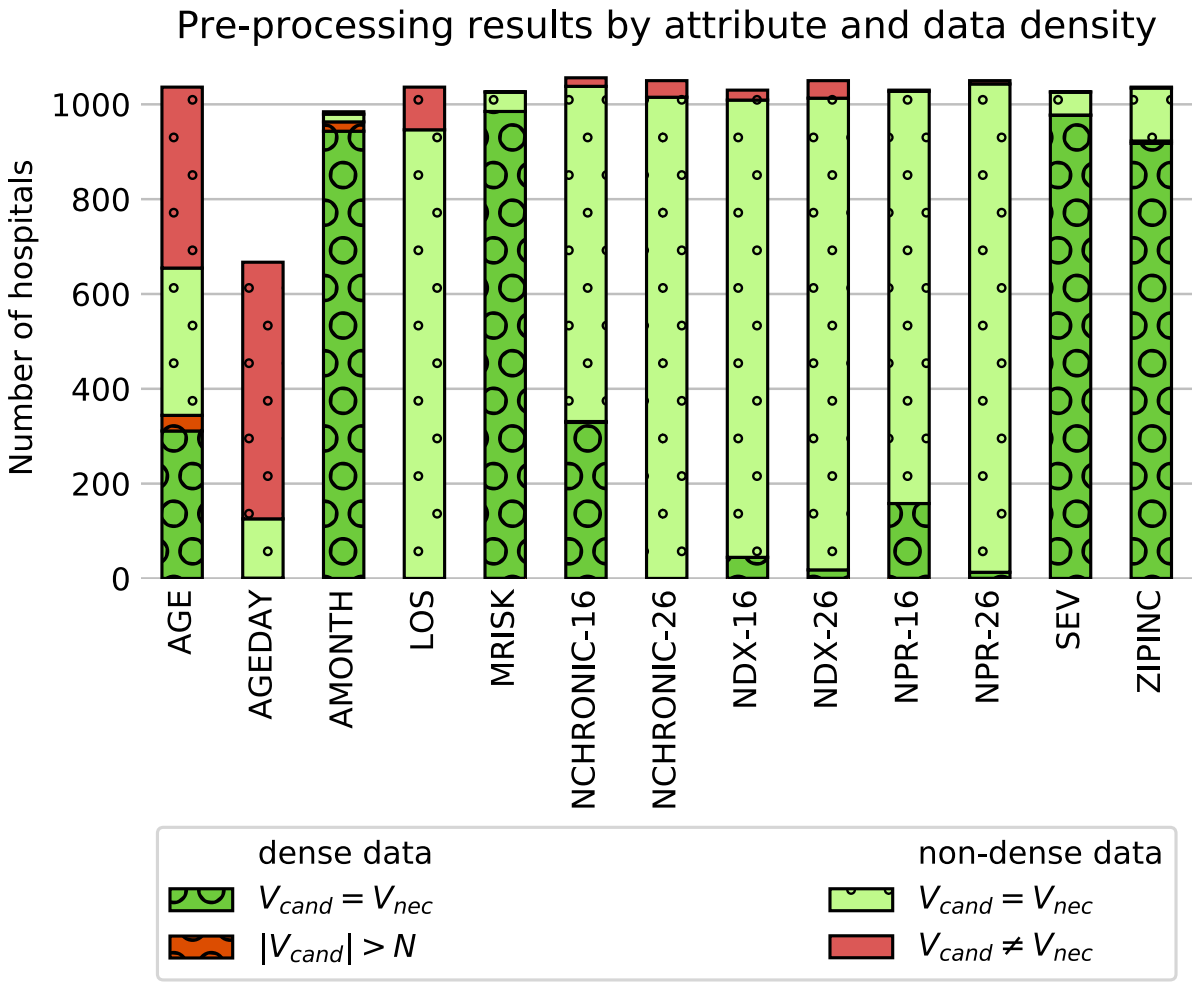
This leaves {1, 12, 17, 20}

Recovering the database counts:

	1
12-1 =	11
17-12=	5
20-17=	3

which is correct up to reflection!

Distribution Reconstruction by Preprocessing: Experimental Evaluation

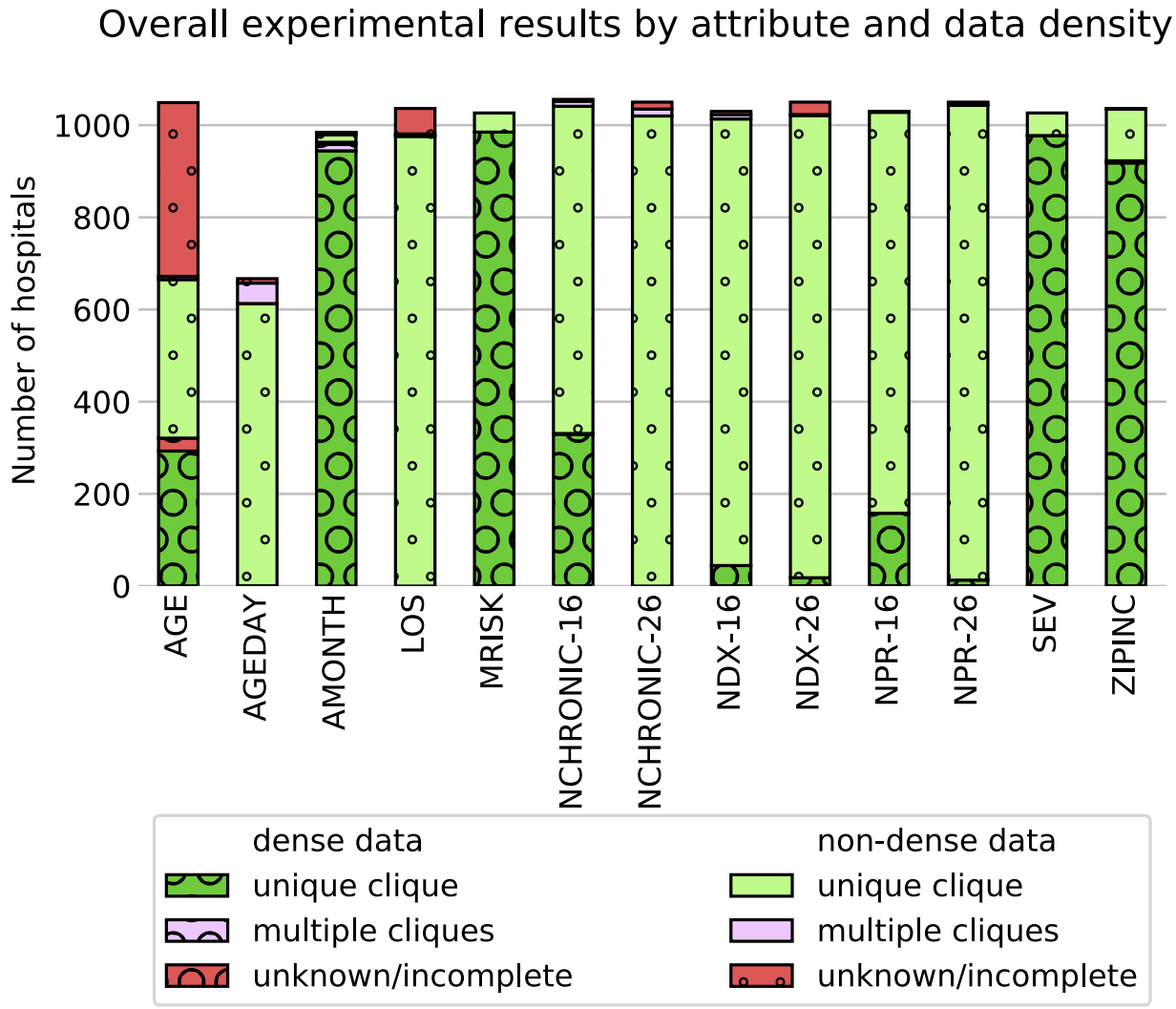


Distribution Reconstruction by Clique Finding

Clique finding:

- Pre-processing starts with a set of necessary vertices V_{nec} and a set of possible candidate vertices V_{cand} for the clique.
- It grows V_{nec} and shrinks V_{cand} ending with $V_{nec} \subseteq V_{elem} \subseteq V_{cand}$, where V_{elem} is the set of elementary vertices.
- If $V_{nec} = V_{cand}$, then we are done (special case for sparse data, where o can arise as a volume).
- Otherwise, we extend the sub-clique on V_{nec} to a larger one using a special-purpose algorithm (target is clique on N vertices).
- Several heuristics are employed in our algorithm; these rely on various graph algorithms as sub-steps, including Luby's algorithm for finding maximal independent sets.

Distribution Reconstruction by Preprocessing: Experimental Evaluation



A Random Graph Model for Distribution Reconstruction

- We can also build a probabilistic model of the graph in our attack.
  - Assume data is uniformly distributed, so database counts follow a multinomial distribution.
  - Approximate each count by a Poisson distribution; volumes of ranges are also then Poissonian.
- From this we can estimate that the initial graph has about $2N + N^3/8(\pi R)^{1/2}$ vertices.
- We can also show that the graph has about $N^2 + N^7/80(\pi R^3)^{1/2}$ edges.
- Edge density is then $O(N/R^{1/2})$.
- Applying results from random graph theory we find that, to ensure $O(1)$ cliques, we need $R=\Omega(N^2)$.
  - This assumes we have a random graph – we manifestly do not!
- This bound on R matches well with what we observe in our experiments with HCUP data: for R above $N^2/2$, the attack works well; for R below $N^2/2$, it tends to fail.

Summary of Attacks from [GLMP]

Distribution reconstruction in $\approx N^2 \log N$ queries for uniform ranges, using only volume leakage, provided $R = O(N^2)$.

Attack	Req'd leakage	Other req'ts	Suff. # queries
KKNO16 - DR	Volume	Uniform queries	$O(N^4 \log N)$
DR	Volume	$R = O(N^2)$	$O(N^2 \log N)$ for uniform queries
Update data recovery	Volume	$R = O(N^2)$	$O(N)$ (random graph model)
Online query recon	Volume	Auxiliary dist.	Experimental

Conclusions

Conclusions

- Many clever schemes have been designed, enabling range queries on encrypted data.
  - OPE, ORE schemes, POPE, [HK16], Blind seer, [Lu12], [FJKNRS15], FH-OPE, Lewi-Wu, Arx, Cipherbase, EncKV,…
- Second-generation schemes defeat the snapshot adversary (with caveats).
- It is important to analyse impact of leakage of these schemes.
- No known scheme offers meaningful privacy against a persistent adversary (including server itself).
  - In realistic settings, N logN queries suffice; even less if auxiliary distribution + rank leakage is known.
- One can apply ORAM to hide the access pattern leakage, but then performance suffers and volume attacks are still possible.
  - And were already possible for a network attacker!

Future Work

- More research is needed!
- Overall goal: since perfect security is too expensive, we need to raise the bar for the attacker without hurting performance too much.
- And for schemes supporting richer classes of queries than just range queries.
- Some kind of ORAM with limited locality? (Sacrificing ORAM's strong obliviousness guarantees for better performance.)
- Exploration of the effectiveness of adding padding and/or noise in preventing attacks.