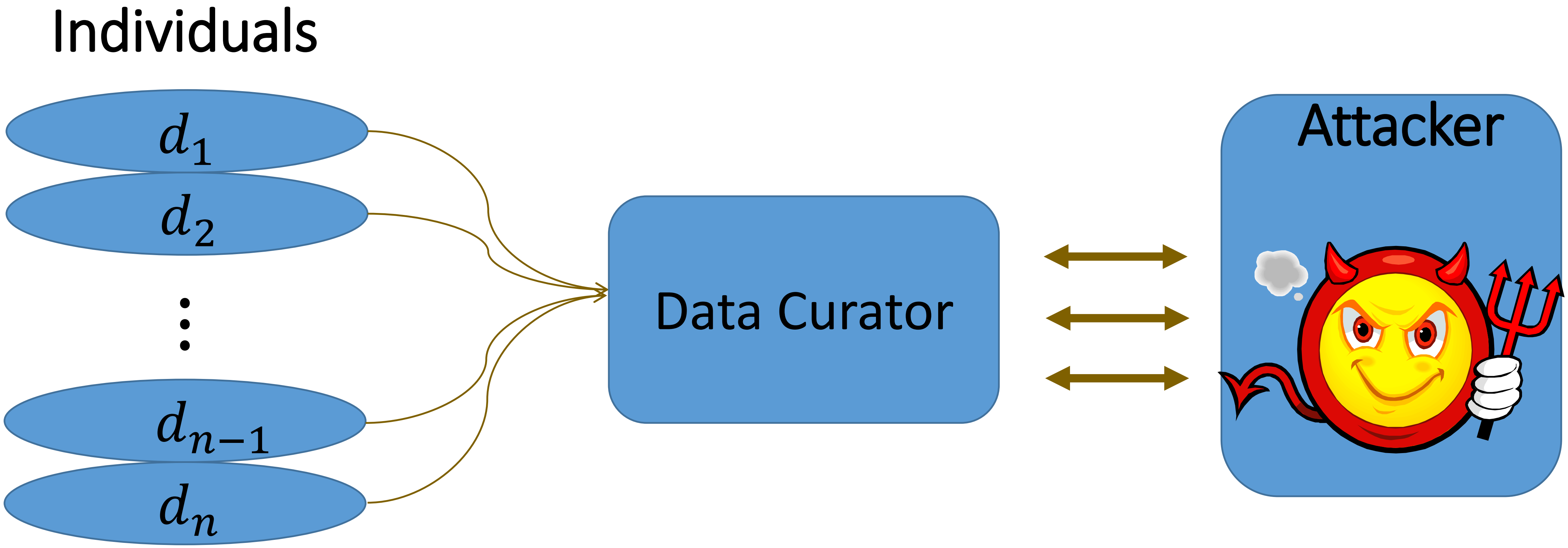
Understanding the Sparse Vector Technique for Differential Privacy

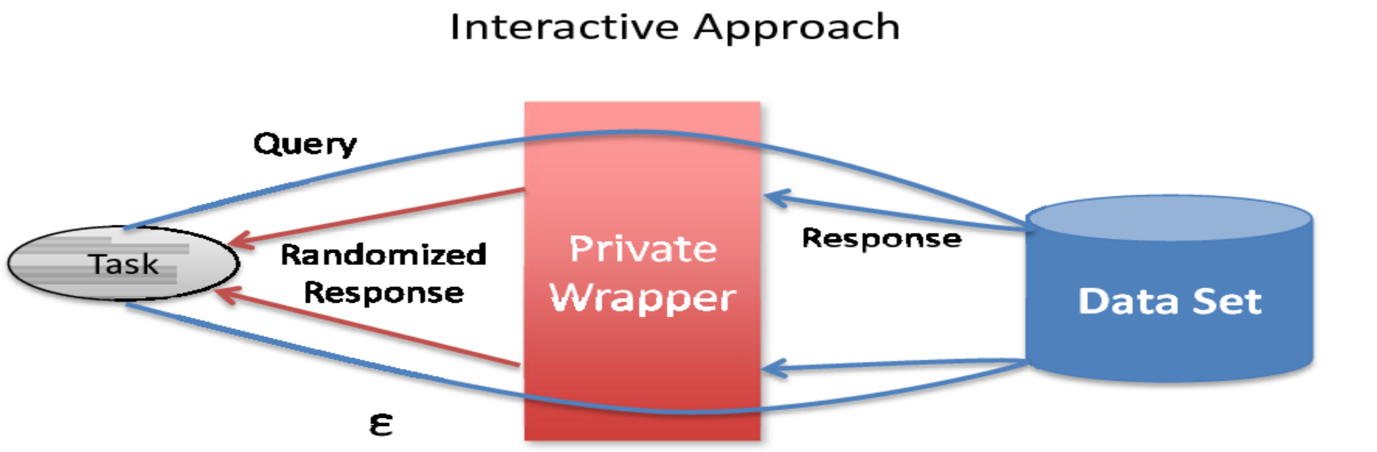
Min Lyu, Dong Su, Ninghui Li

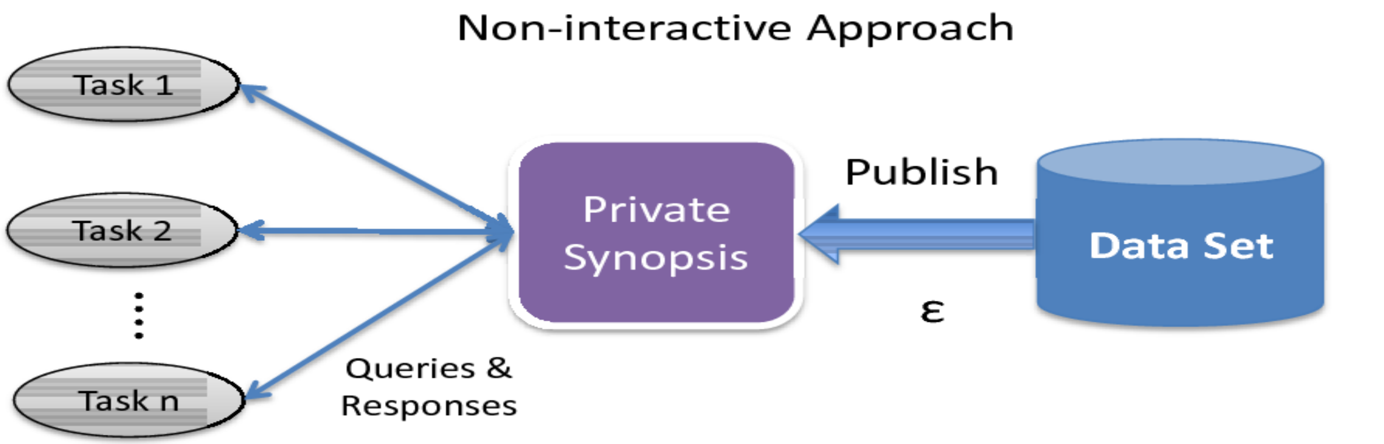
Learning from Private Data



Interactive Setting versus. Non-interactive Setting

- Interactive setting
  - Answer queries as they come, not knowing what the rest of the queries ar
- Non-interactive setting
  - The set of all queries tha one wants to provide utility are known





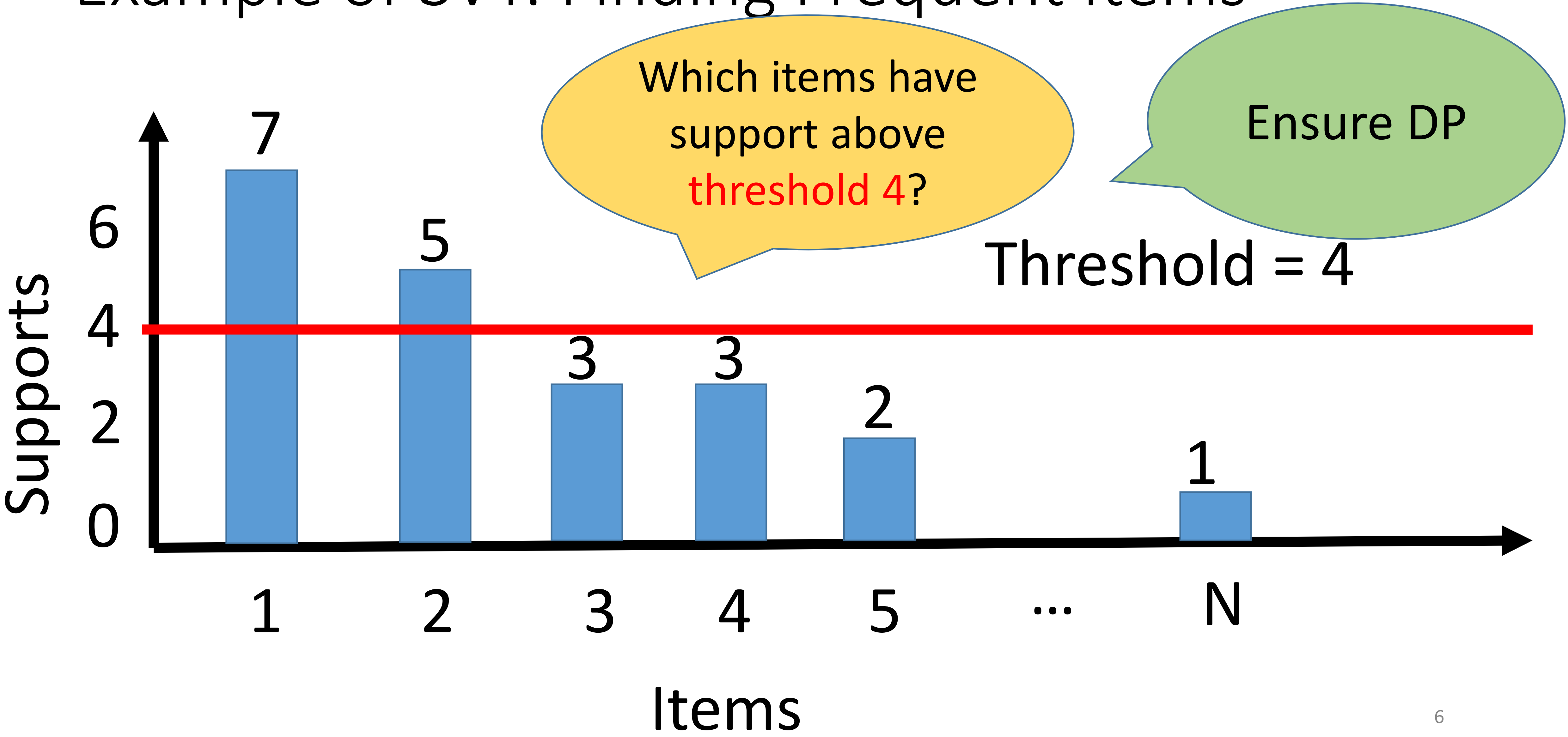
Limitation of Interactive Setting

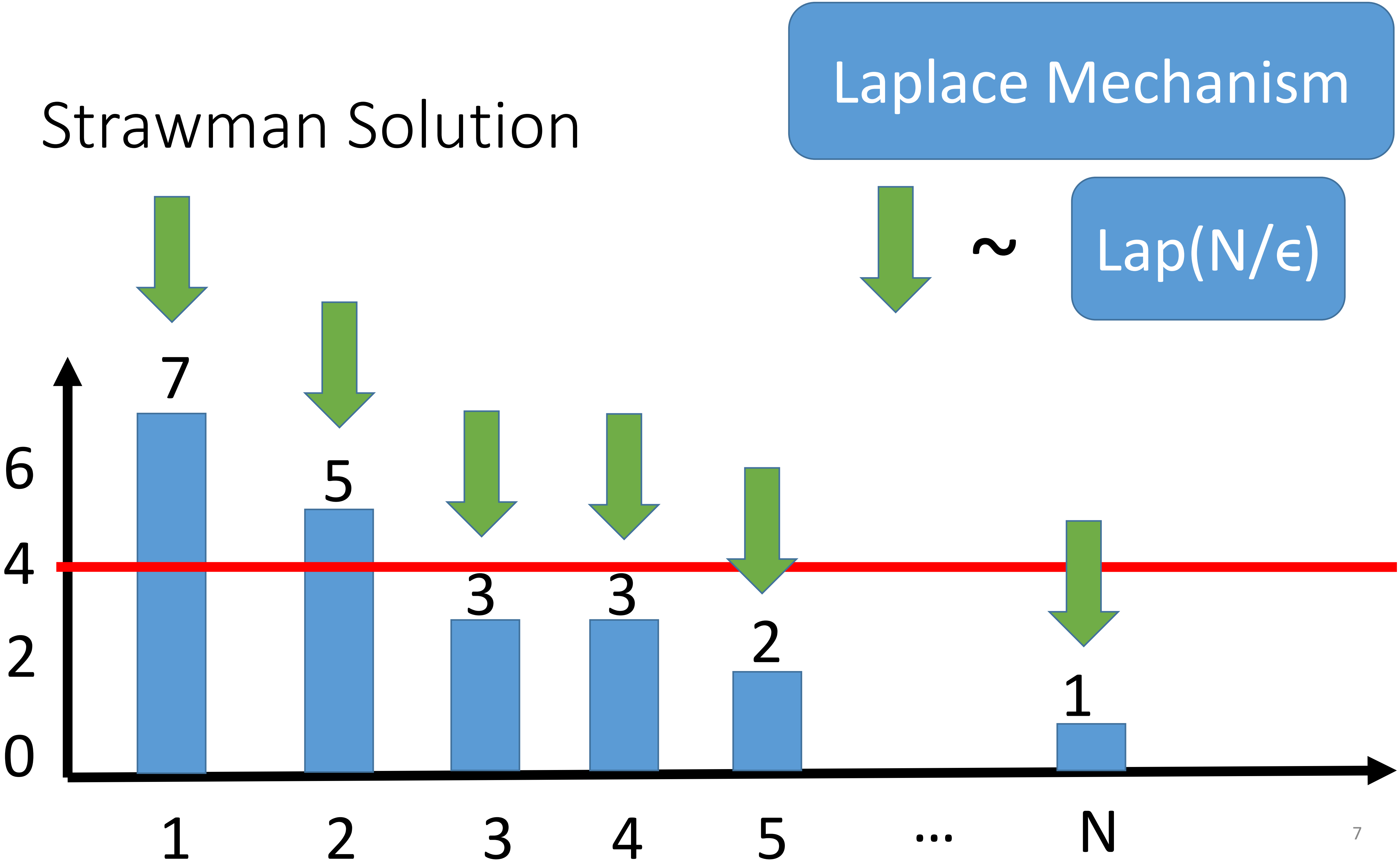
- Answering each query consumes some privacy budget
- After answering a pre-determined number of queries, one exhausts the privacy budget, and cannot answer any question anymore
- Problem especially intractable when dealing with multiple users of data

Using the Sparse Vector Technique in Interactive Setting

- For each new query,
  - Use past queries/answers to generate an simulated answer
  - Check whether the error of simulated answer is above some (noisy) threshold
  - If error is below threshold, then return simulated answer
  - If error is above threshold, then query the data to answer the query (consumes privacy budget), returns the answer and store the query/answer
- If threshold is perturbed, then answering with simulated answer is “free” (i.e., not consuming privacy budget)

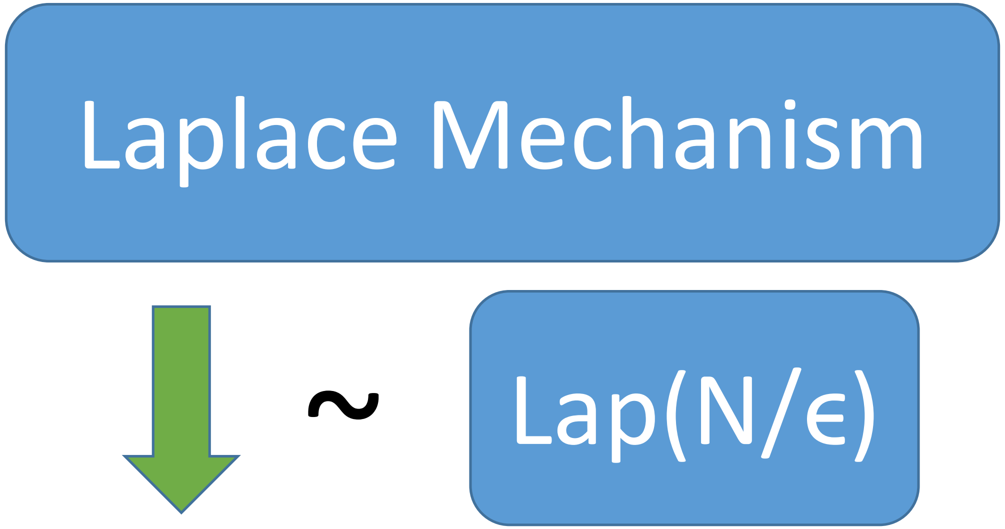
Example of SVT: Finding Frequent Items

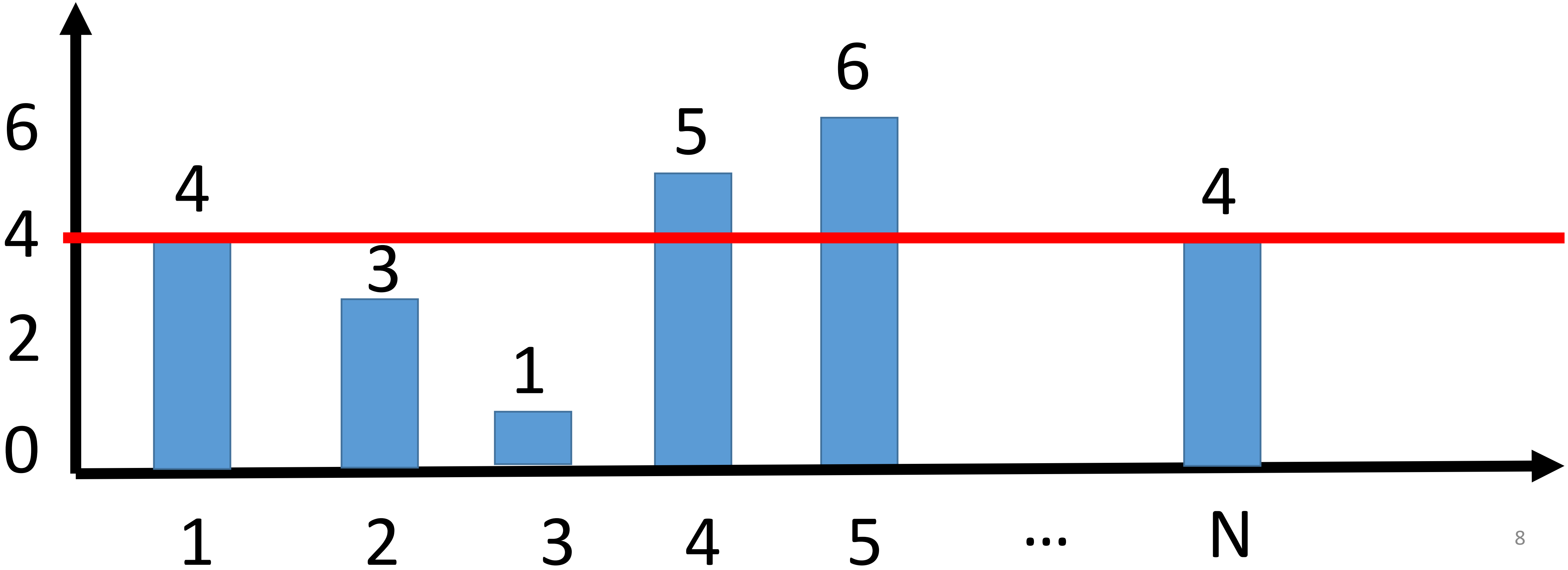




Strawman Solution
Laplace Mechanism
~
Lap(N/ϵ)
7
5
3
3
2
1
6
4
2
0
1
2
3
4
5
…
N

Strawman Solution



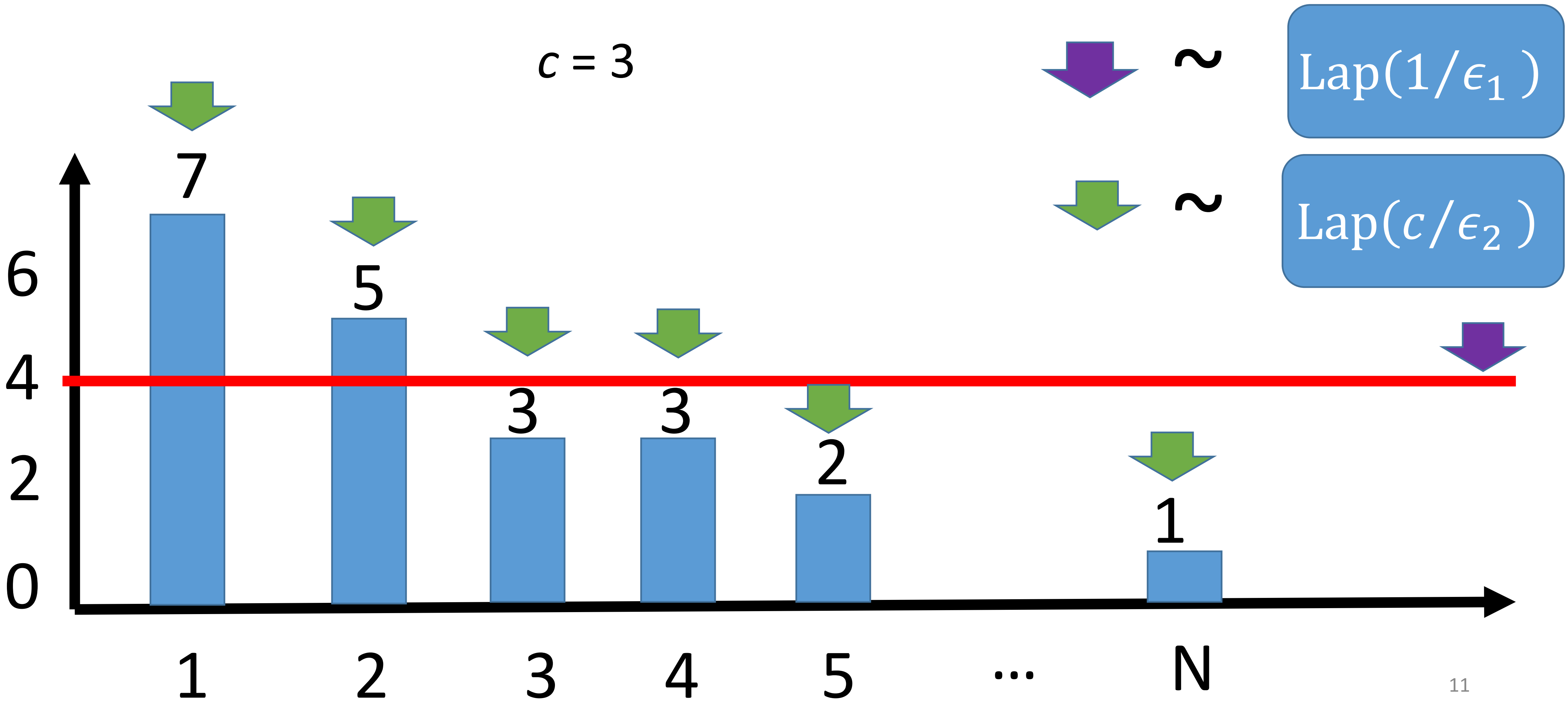


Sparse Vector Technique

Given a sequence of queries and a certain threshold *T*,

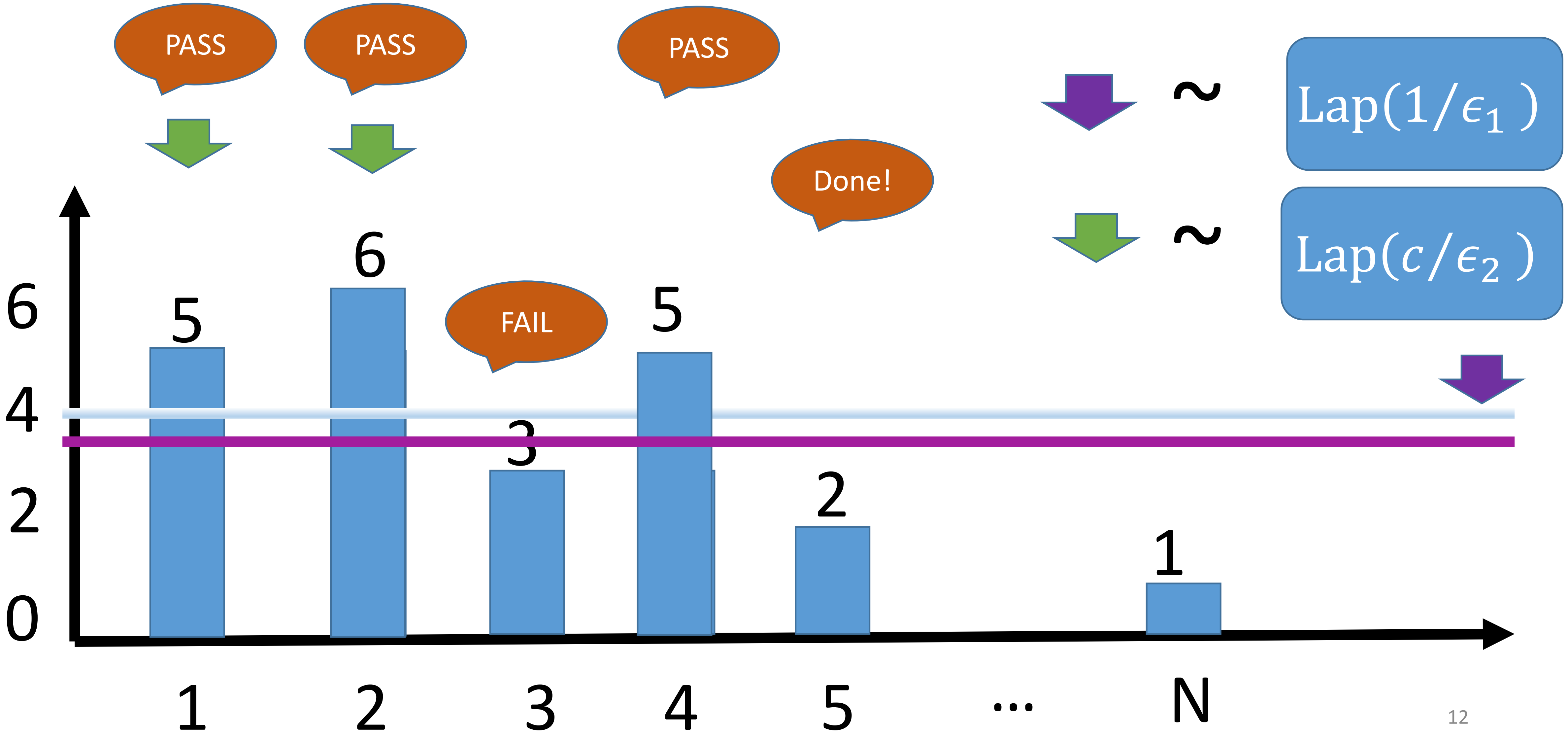
- Perturb the threshold
- Compare each perturbed query answer against the noisy threshold
- Output a vector indicating whether each query answer is above or below *T*, denoted by $\top$ and $\bot$
- Output noisy counts for positive queries (optional)

Sparse Vector Technique [DNR+09, HR10, RR10]



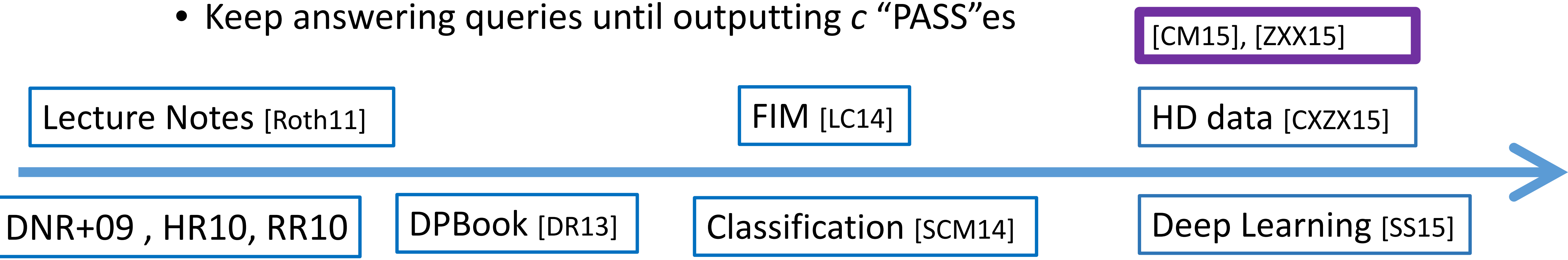
Sparse Vector Technique

$c = 3$



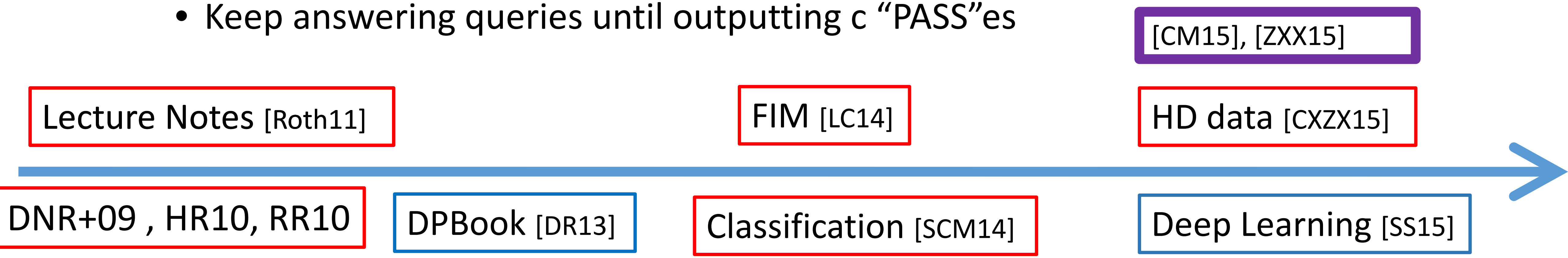
Sparse Vector Technique

- Input: stream of queries and threshold
- Output: vector of indicators
- Key Points
  - Perturbing threshold
  - Expect predominant majority of queries are below threshold
  - Only outputting “PASS” consumes privacy budget
  - Keep answering queries until outputting *c* “PASS”es



Sparse Vector Technique

- Input: stream of queries and threshold
- Output: vector of indicators
- Key Points
  - Perturbing threshold
  - Expect predominant majority of queries are below threshold
  - Only outputting “PASS” consumes privacy budget
  - Keep answering queries until outputting c “PASS”es



Contribution

- A new version of SVT that provides better utility
  - Optimized privacy budget allocation
  - Reduce sensitivity noise scale by half for monotonic queries
  - Retraversal with higher threshold
- Rigorous proof of SVT's privacy
  - Identify misunderstandings that likely caused the different non-private versions
  - Pointed out the error in the proof of [CM15]
- In non-interactive setting, SVT can be replaced by EM

Our Proposed Standard SVT (SVT-S)

Require: $D, Q, \Delta, \epsilon, \mathbf{T} = T_1, T_2, \cdots.$

$\rho = \text{Lap}\left(\frac{\Delta}{\epsilon_1}\right)$
count $= 0$
for each query $q_i \in Q$ **do**
    $\nu_i = \text{Lap}\left(\frac{2c\Delta}{\epsilon_2}\right)$
    if $q_i(D) + \nu_i \geq T_i + \rho$ **then**
        if $\epsilon_3 > 0$ **then**
            Output $a_i = q_i + \text{Lap}\left(\frac{c\Delta}{\epsilon_3}\right)$
        else
            Output $a_i = \top$
        count $=$ count $+ 1$,
        if count $\geq c$ **then**
            Abort
    else
        Output $a_i = \bot$

Perturb threshold once

Perturb each query with noise scaling with c

Pay extra budget for outputting numeric answers

Stop after getting c positive answers

How to ensure DP?

- Perturb the threshold:

    mask the difference of negative queries on D and D', no matter how many negative queries there are.

- Perturb the query:

    bound the probability ratio for positive queries

- Stop after getting target amount of positive answers:

    noise $\propto c$

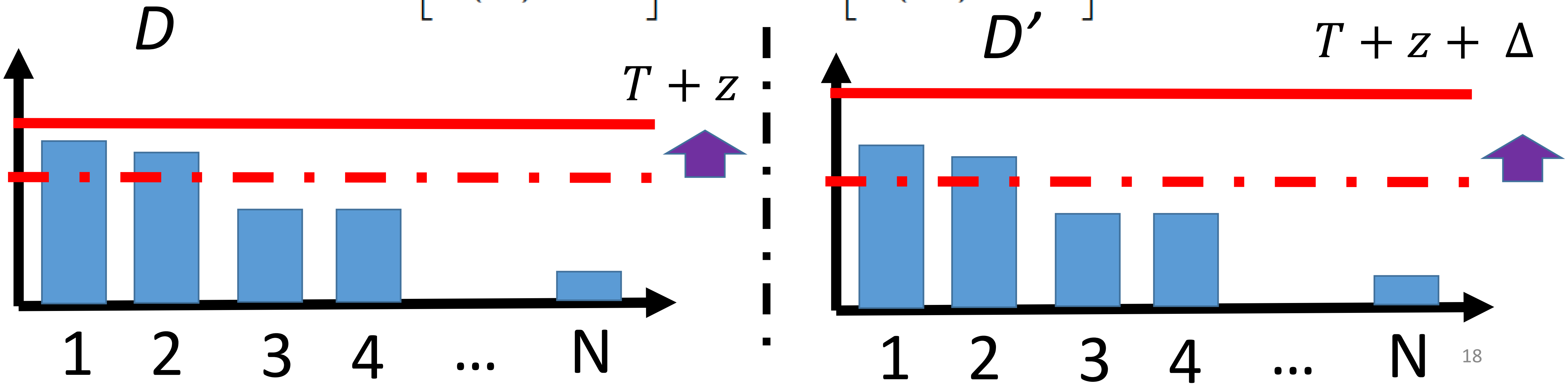
How to Prove Privacy?

- First, analyze the situation that all outputs are negative.

Lemma

Let $\mathcal{A}$ be SVT-S. For any neighboring datasets D and D', and any integer ℓ, we have

$$\Pr\left[\mathcal{A}(D) = \perp^{\ell}\right] \leq e^{\epsilon_1} \Pr\left[\mathcal{A}(D') = \perp^{\ell}\right].$$



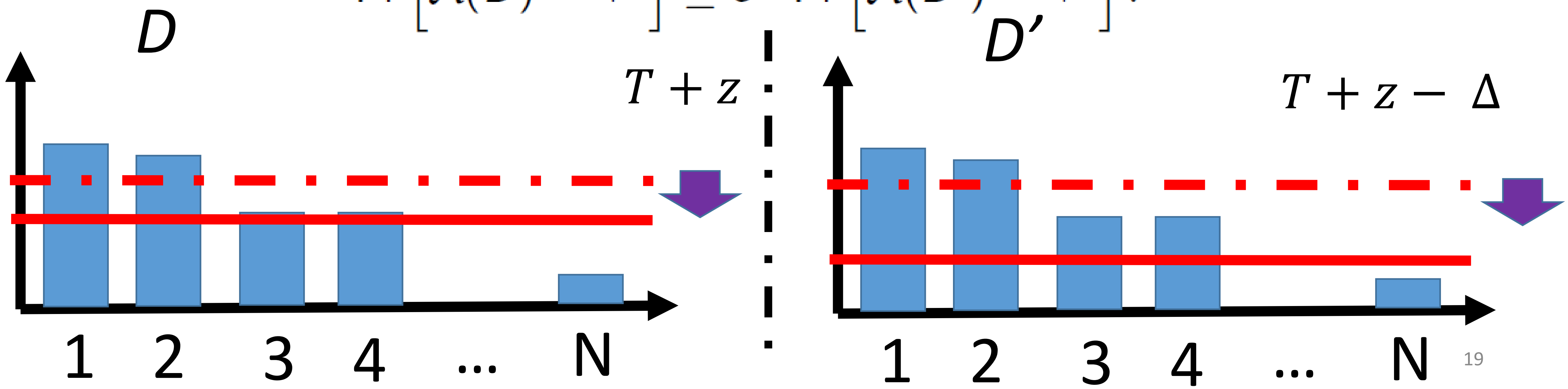
How to Prove Privacy?

- Second, analyze the situation that all outputs are positive.

Lemma

Let $\mathcal{A}$ be SVT-S. For any neighboring datasets D and D', and any integer ℓ, we have

$$\Pr\left[\mathcal{A}(D) = \top^{\ell}\right] \leq e^{\epsilon_1} \Pr\left[\mathcal{A}(D') = \top^{\ell}\right].$$

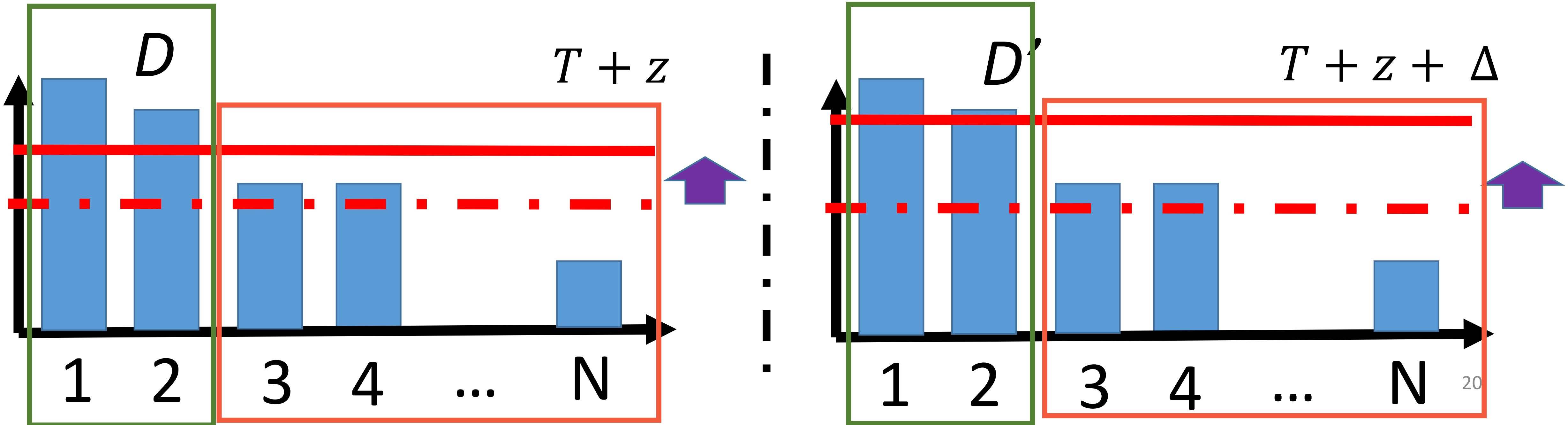


How to Prove Privacy?

- Third, combine them together, **but have to choose one direction**

For negative outputs, budget on perturbing threshold is enough

For positive outputs, need to add $\mathrm{Lap}(2c/\varepsilon_2)$ to each query and stop after outputting c of them



Improving Utility of SVT-S

- Optimizing budget allocation between query perturbation and threshold perturbation: $\epsilon_1 : \epsilon_2 = 1 : (2c)^{2/3}$

- For monotonic queries:
  - query noise is $Lap(\frac{c\Delta}{\epsilon_2})$ instead of $Lap(\frac{2c\Delta}{\epsilon_2})$
  - Optimization of privacy budget allocation: $\epsilon_1 : \epsilon_2 = 1 : c^{2/3}$

- For non-interactive setting, SVT with retraversal:
  - Increase the threshold
  - Retraverse the list of queries until *c* queries are selected.

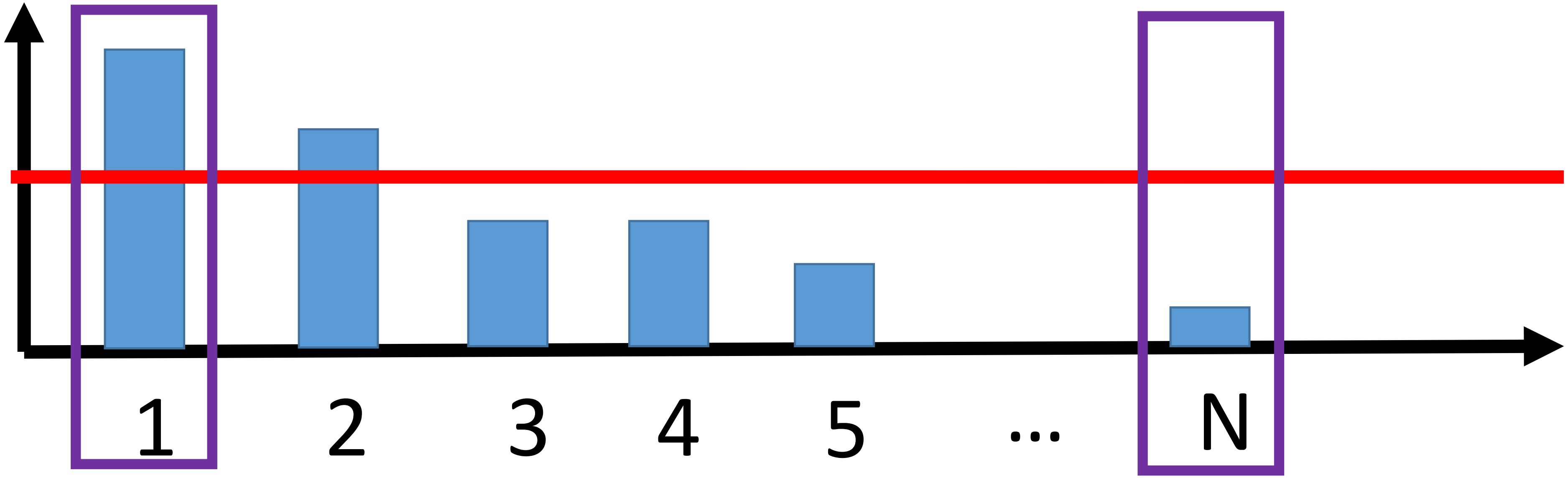
Experiment

Dataset	#Records	#Items
BMS-POS	515,597	1,657
aol	647,337	2,290,685
kosarak	990,002	41,270
zipf	10,00,000	10,000

Settings	Methods	Description
Interactive	SVT-DPBook	DPBook SVT
	SVT-S	Standard SVT
Non-interactive	SVT-ReTr	Standard SVT with Retraversal
	EM	Exponential Mechanism

Evaluation Metrics

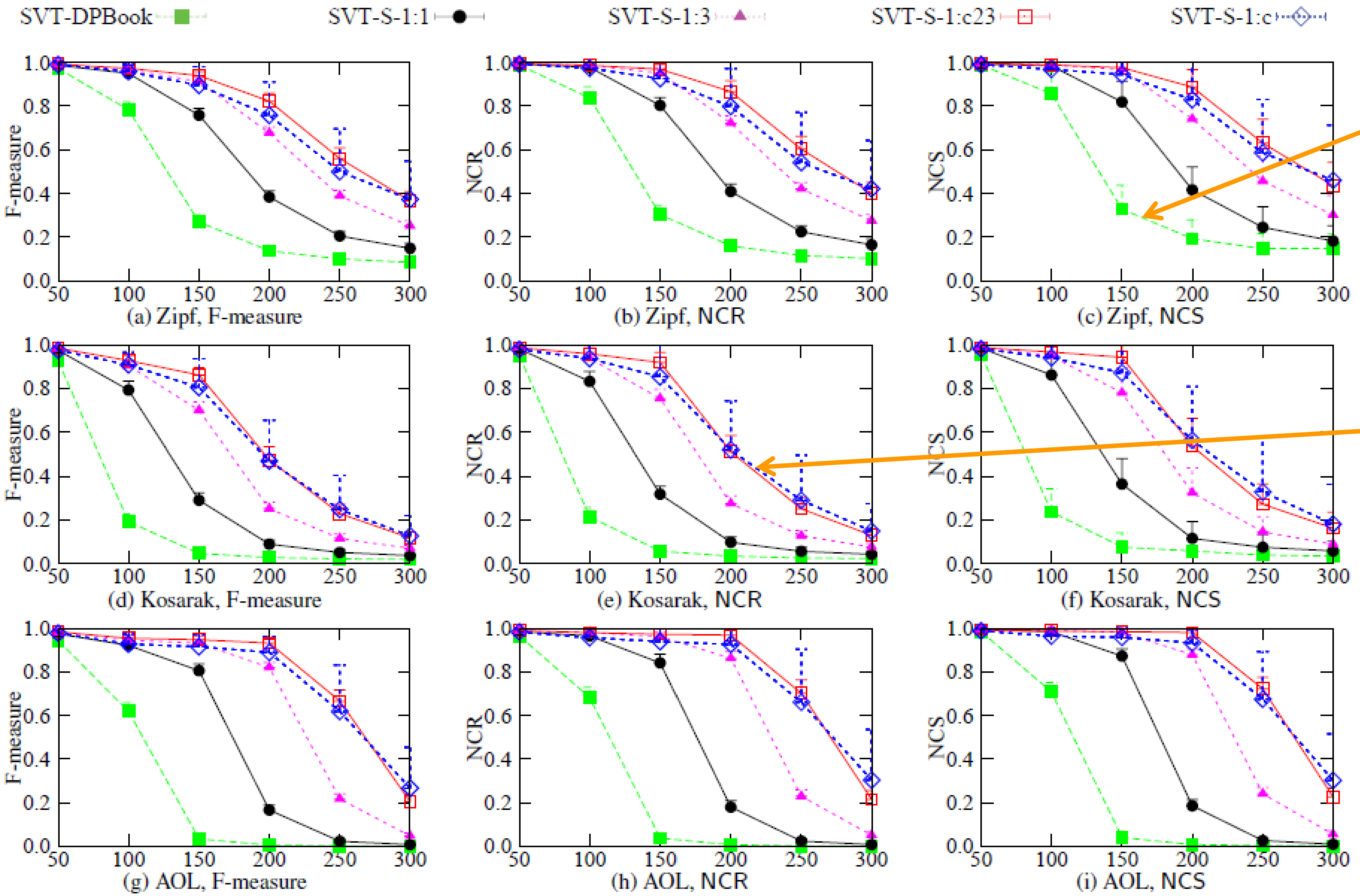
- F-Measure
  - Harmonic mean of precision and recall of the computed item set and the ground truth item set
  - Uniform penalization for all queries
    - missing the top most item is penalized the same way as missing the N-th item.



Evaluation Metrics

- Normalized Cumulative Gain
  - Consider both **membership** and **query score**
  - $$\text{NCG}(U_{\mathcal{A}}(D)) = \frac{\sum_{q \in U_{\mathcal{A}}(D)} \text{rel}(q)}{c}$$
  - $\text{rel}(q)$ is the relevance score for the query q. We derive two instantiations of NCG by choosing two different relevance score functions.
    - Normalized Cumulative Rank (NCR): $\text{rel}(q)$ is q's rank
      - Highest one has rank N, and the next one has rank $N - 1$
      - Normalized by the maximum score $N(N + 1)/2$
    - Normalized Cumulative Support (NCS): $\text{rel}(q)$ is true answer of q
      - $$\text{NCS}(U_{\mathcal{A}(D)}) = \frac{\sum_{q \in U_{\mathcal{A}(D)}} q(D)}{\sum_{q \in U_T} q(D)}$$

Comparison on Interactive Approaches

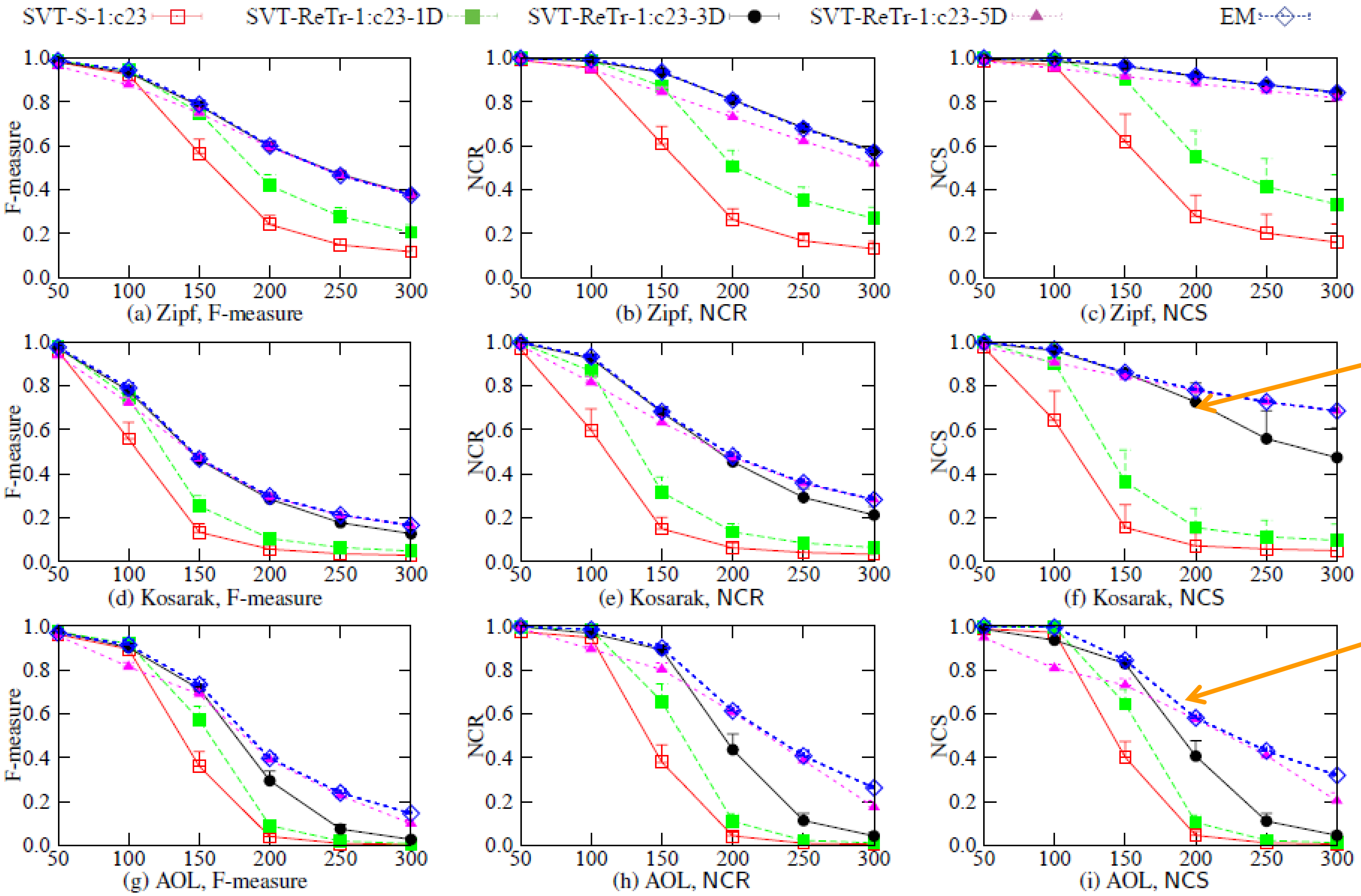


DPBook performs worst

$1:c$ and $1:c^{2/3}$ are much better than 1:1 allocations

Privacy budget 0.25

Comparison on Non-Interactive Approaches



Increasing threshold improve accuracy

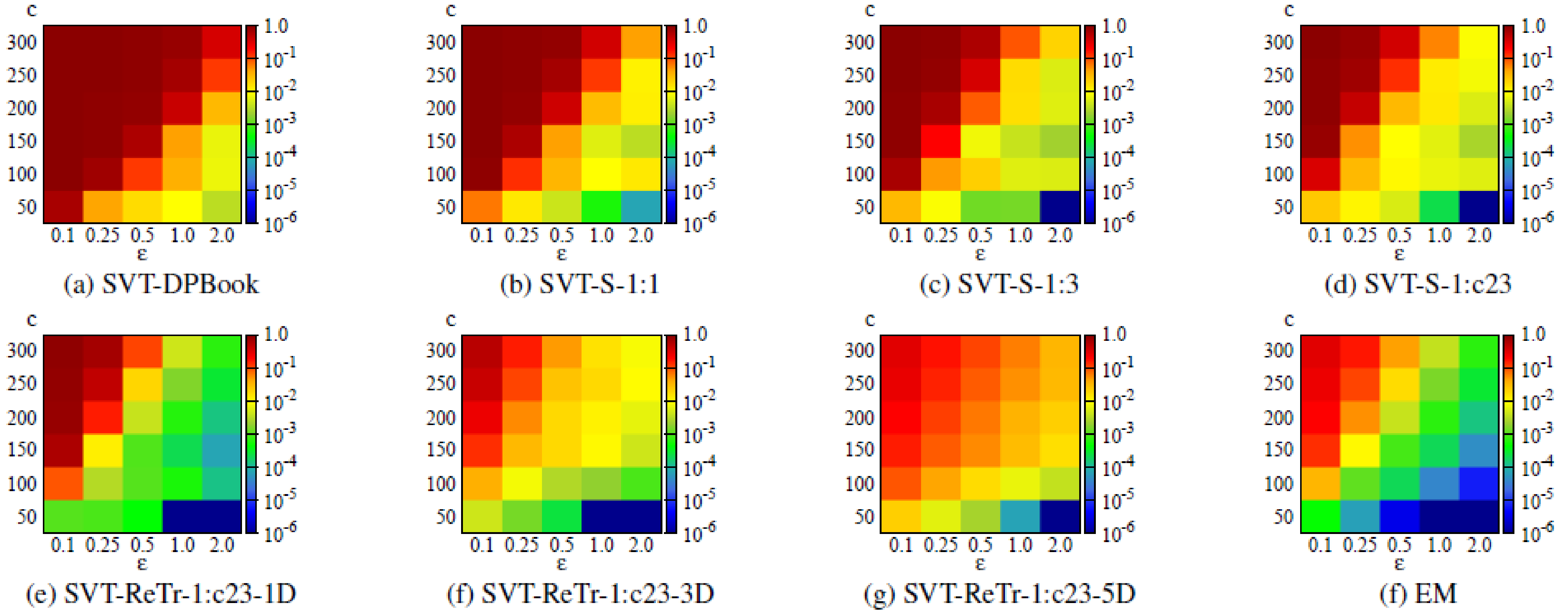
EM is clearly better than 1: $c^{2/3}$

Privacy budget 0.1

Varying ϵ and Maximum Number of Positive Queries

Interactive

Non-interactive



Dataset: Kosarak

Metric: 1.0-NCS

Recommendations

- In the interactive settings, use our proposed standard SVT
  - For general queries, uses the $1/(2c)^{2/3}$ to allocate privacy budget between $\epsilon_1$and ϵ_2
  - For monotonic queries, uses the $1/c^{2/3}$ to allocate privacy budget between $\epsilon_1$and ϵ_2
- In the non-interactive settings, do not use SVT and use EM instead
  - If one gets better performing using SVT than using EM, then it is likely that one's usage of SVT is non-private

Q & A?