# Frequent items over general data streams

Sumit Ganguly, Abhayendra Singh, Satyam Shankar

Indian Institute of Technology (IIT), Kanpur, India

# Overview

# Outline

# Data Stream Applications



- Network monitoring, web monitoring, financial data monitoring, etc..
- Characteristics:
  1. Records arrive continuously and at high rates.
  2. User defined queries encode alerts for exceptional conditions, anomalies or desirable scenarios, etc..
  3. Aggregate statistical analysis versus deep analysis.
- *Online computation using sub-linear space.*

# Data Stream Model

- Stream $\sigma$ is a sequence of records of the form $(pos, i, v)$, where, $i \in [n] = \{1, 2, \ldots, n\}$ and $v \in \mathbb{Z}$.
- $v$ is change in the *frequency* of $i$. *pos* is index of record in sequence.
- *frequency* of item $i$ in stream $\sigma$ is denoted by $f_i(\sigma)$.
- So

$$f_i(\sigma) = \sum_{(pos,i,v) \in \sigma} v$$

- $f_i(\sigma)$ is sum of changes to frequency of $i$ over all records in $\sigma$.

# Data Stream Models

- *General* stream: frequency vector $f(\sigma) \in \mathbb{Z}^n$.
- *Strict* stream: frequency vector $f(\sigma) \geq 0$.
- *Insert-only* stream: for each update $(i, v)$, $v = 1$.
- *Sliding window* stream: Stream defined as the sequence of records arriving in the time window $[\text{NOW}, \text{NOW} - w + 1]$.

# General Streams: Notation



- Let $\sigma$ be a general stream.
- $[n]$ is $\{1, 2, \ldots, n\}$.
- Define first and second moment of frequency vector.

$$F_1(\sigma) = \sum_{i \in \{1,\ldots,n\}} |f_i(\sigma)| = \|f(\sigma)\|_1 \quad .$$

$$F_2(\sigma) = \sum_{i \in \{1,\ldots,n\}} |f_i(\sigma)|^2 = \|f(\sigma)\|_2^2 \quad .$$

# Problem definition $\textsc{Frequent}_1(\epsilon, \phi)$



- Data structure can process data stream $\sigma$. Abbreviate $F_1(\sigma)$ by $F_1$ and $F_2(\sigma)$ by $F_2$.
- Problem $\textsc{Frequent}_1(\epsilon, \phi)$, $0 < \phi < \epsilon < 1$:
- Output all items $i \in [n]$ such that $f_i \geq \epsilon F_1$.
- Do not output any item $i \in [n]$ such that $f_i < (\epsilon - \phi) F_1$.

# Problem definition $\textsc{Frequent}_2(\epsilon, \phi)$



- Data structure must process data stream $\sigma$.
- Problem $\textsc{Frequent}_2(\epsilon, \phi)$, $0 < \phi < \epsilon < 1$:
- Output all items $i$ such that $f_i \geq (\epsilon F_2)^{1/2}$.
- Do not output any item $i$ if $f_i < ((\epsilon - \phi)F_2)^{1/2}$.

# Significance of FREQUENT

- Need low-space algorithms with space $o(n \log F_1(\sigma))$.
- Closely elated to problems of finding approximate frequent items, approx. quantiles, etc..
- Very popular and well-studied problem.
- In this work, *we consider problem* FREQUENT *for general streams.*

# Outline

# Measuring efficacy of algorithms for FREQUENT



- *Space* used during online computation.
- *Online processing time*: Time used to process each stream record and update data structure.
- *Retrieval Time*: Time taken to discover and output frequent items (and their estimated frequency).

# Review: Algorithms for $\textsc{Frequent}(\epsilon, \phi)$



*Insert-only* streams.

- $m = \|f(\sigma)\|_\infty = \max_{1 \le i \le n} |f_i(\sigma)|$.
- Deterministic/randomized space lower bound:

$$\Omega\left(\phi^{-1}\log(n\phi)\right)$$

[Bose et.al. SIROCCO 03, Cormode-Muthu PODS 03]

- Deterministic algorithm

$$\text{Space: } O(\phi^{-1}\log(mn))$$
$$\text{Online Processing time: } O(1)$$
$$\text{Retrieval time: } O(\phi^{-1}) .$$

[Misra-Gries82, Karp et.al. TODS03, Demaine et.al. ESA 02]

# Review: Algorithms for $\text{FREQUENT}_1(\epsilon, \phi)$: General Streams



- CountMin Algorithm [Cormode-Muthu *J. Algo.* 05], extension to general streams [Cormode-Muthu *IEEE/ACM Trans. Network.* 2005.]

$$\text{Space: } O(\phi^{-1}(\log F_1)(\log n)\log((\phi\delta)^{-1}))$$

$$\text{Online processing time: } O((\log n)\log((\phi\delta)^{-1}))$$

$$\text{Retrieval time: } O(\text{ Space}/\log(F_1)) \ .$$

## Review: Algorithms for $\textsc{Frequent}_2(\phi)$: General Streams



- $\textsc{CountSketch}$ algorithm[Charikar et.al. ICALP 02] extension [Cormode-Muthu *IEEE/ACM Trans. Network.* 2005.]

Space: $O(\phi^{-2}(\log n)(\log F_1)(\log((\phi\delta)^{-1})))$

Online processing time: $O((\log n)(\log((\phi\delta)^{-1})$

Retrieval time: $O($ Space$/\log(F_1)$ .

# Review: Algorithms for $\text{FREQUENT}(\epsilon, \phi)$



- General Streams: *Reversible sketches* [Schweller, Li, Chen+ *IEEE/ACM Trans. Network.* 2004.] for solving $\text{FREQUENT}_1(\epsilon, \phi)$.
- Specially constructed reversible hash functions.
- Optimizes space at expense of retrieval time.
- Space $O(\phi^{-1}(\log(\phi\delta)^{-1})(\log F_1))$.
- No bound given for retrieval time, (Typically $n^\alpha$, $\alpha > 0.5$).

# Review: Deterministic algorithms for $\textsc{Frequent}(\epsilon, \phi)$

- $\textsc{Frequent}_1(\epsilon, \phi)$:

  Space: $O(\phi^{-2}(\log^2 m)(\log^2 n)/(\log^2(\phi^{-1})))$

  Online processing time: $O(\phi^{-1}\log n/(\log \phi^{-1}))$

  CR-Precis [G-Majumder ESCAPE 07].
- $\textsc{Frequent}_1(\epsilon, \phi)$: Space lower bound [G CSR 08]

$$\Omega(\phi^{-2}(\log m)) \quad .$$

## Review: Deterministic algorithms for $\textsc{Frequent}(\epsilon, \phi)$



- $\textsc{Frequent}_2(\epsilon, \phi)$: Space lower bound $\Omega(n)$ [G 2008].
- Therefore, to obtain $\tilde{O}(\phi^{-1})$ space algorithm for $\textsc{Frequent}_1$ or $\textsc{Frequent}_2$, we consider only randomized algorithms.

# Contributions



- Novel algorithms for solving $\textsc{Frequent}_1(\epsilon, \phi)$ and $\textsc{Frequent}_2(\epsilon, \phi)$.
- We extend dyadic intervals technique to general streams.
- Worst case resource requirements.

$$\begin{aligned}
\text{Space: } & O\left(\phi^{-1}(\log(\epsilon-\phi)^{-1})(\log(n(\epsilon-\phi))) \right. \\
& \qquad \left. \log((\epsilon-\phi)^{-1}\delta^{-1})(\log F_1)\right) \\
\text{Update: } & O\left((\log(n(\epsilon-\phi)))\log((\epsilon-\phi)^{-1})\log((\epsilon-\phi)^{-1}\delta^{-}\right. \\
\text{Retrieval: } & O\left((\epsilon-\phi)^{-1}(\log(\epsilon-\phi)^{-1})(\log(n(\epsilon-\phi))) \right. \\
& \qquad \left. \log((\epsilon-\phi)^{-1}\delta^{-1})\right) \quad .
\end{aligned}$$

# Outline

# Dyadic intervals technique



- Assume $n$ is a power of 2.
- Level 0, interval length $2^0$: $[1,1][2,2]\ldots[n,n]$.
  Level 1, interval length $2^1$: $[1,2][3,4]\ldots[n-1,n]$.
  Level 2, interval length $2^2$: $[1,4][5,8]\ldots[n-3,n]$.

  $$\vdots$$

  Level $l$, interval length $2^l$:

  $$[1,2^l][2^l+1,2^{l+1}]\ldots[n-2^l+1,n].$$

# Dyadic intervals contd.

- Tree structure

$$[j2^l + 1, (j+1)2^l] =$$
$$[(2j)2^{l-1} + 1, (2j+1)2^{l-1}] \text{ (left child)} \quad +$$
$$[(2j+1)2^{l-1} + 1, (j+1)2^l] \text{ (right child)}$$

- Set of dyadic intervals at level $l$: $\{1, \ldots, n/2^l\}$.
- Dyadic interval $i$ at level $l$
  1. Parent$(i)$: $\lceil i/2 \rceil$.
  2. Left child$(i)$: $2i - 1$.
  3. Right child $(i)$: $2i$.

# Frequent items using Dyadic intervals



- Applicable for *strict streams*, i.e., *non-negative frequencies*.
- Interval frequency $I = [j2^l + 1, (j+1)2^l]$

$$f_I = \sum_{i \in I} f_i, \quad (\text{ denoted } f_j^{(l)}) \ .$$

- Sum of frequencies $F_1$ is invariant across level.

$$\sum_{\text{dyadic intervals } I \text{ at level } l} f_I = \sum_{i=1}^{n} f_i = F_1 \ .$$

- Call item/interval $I$ frequent if $f_I > \epsilon F_1$.
- If $i$ is frequent, then every dyadic interval containing $i$ (parent/ancestor) is also frequent

$$\epsilon F_1 \le f_i \le f^{(l)}_{\lceil i/2^l \rceil}$$

# Finding frequent items using frequency oracle



1. Suppose we are given all frequent intervals at level $l$. There are at most $\lfloor 1/\epsilon \rfloor$ such intervals.
2. Consider the set of left and right child intervals (at level $l-1$), check their frequencies, retain frequent intervals and recurse.
3. After level $0$ is processed, all frequent intervals are found.
4. Initialize: Highest level $\log n$ has 1 interval.
5. Slight improvement: start from level $l = \lceil \log \epsilon n \rceil$. Number of oracle calls $2\lfloor 1/\epsilon \rfloor \times \lceil \log \epsilon n \rceil$.

# Using frequency estimator



- Replace frequency oracle by a frequency estimator. Example, use CountMin structure at each level.
- Maintain frequency estimation structure, CountMin [CM 2004] at each level.
- Accuracy of frequency estimation:

$$|\hat{f}_i^{(l)} - f_i| \leq \phi F_1, \text{ with prob. } 1 - \delta \quad .$$

- In previous algorithm, replace frequent item/interval by *estimated frequent*:

$$\hat{f}_i^{(l)} > (\epsilon - \phi) F_1 \quad .$$

# Analysis



- Space requirement: $\log(n(\epsilon - \phi))$ frequency estimator structures with accuracy $\phi$.
- Space requirement is $O(\phi^{-1}(\log(1/\delta))(\log F_1))$ per level [CountMin].
- Total Space: $O(\phi^{-1}(\log(1/\delta))(\log F_1)(\log(n(\epsilon - \phi)))$ .
- Online processing time: time taken to propagate update to $\lfloor \log((\epsilon - \phi)n) \rfloor$ estimator structures.

$$O(\log(1/\delta) \times \lfloor \log((\epsilon - \phi)n) \rfloor) \ .$$

- Retrieval time: $2\lfloor 1/(\epsilon - \phi) \rfloor \times \lceil \log(\epsilon - \phi)n) \rceil$ calls to estimator. Each call typically takes $O(\log(1/\delta))$ time.
- Output is correct with confidence $1 - (\delta \times$ number of calls to estimator).

# Problem with general streams for dyadic technique

- Sum of absolute values of frequencies $F_1$ is *not invariant* across level.
- For example, $f_1 = 100, f_2 = -100$, $f_{[1,2]} = 0$. Both items may be $0.25$-frequent, but dyadic method fails to find it.

# Solution Strategy



- Choose random permutation $\pi : \{1, \ldots, n\} \rightarrow \{1, \ldots, n\}$.
- Fix item $i \in [n]$. Let $I(i, l)$ denote interval at level $l$ to which $i$ maps, i.e., $I(i, l) = \lceil i/2^l \rceil$.
- Each item except $i$ has probability $1/$ (no. of intervals) of mapping to interval $I(i, l)$. So

$$\mathsf{E}\left[ |f_{I(l,i)} - f_i| \right] \leq \sum_{j \neq i} \frac{|f_j|}{\text{no. of intervals}} \leq \frac{F_1}{(n/2^l)} \quad .$$

# Solution strategy-2



- By Markov's inequality,

$$\Pr\left\{|f_{I(l,i)} - f_i| > 8t\frac{F_1}{(n/2^l)}\right\} < 1/(8t) \quad .$$

- Let $\Delta_l = 8t/(n/2^l)$.
- Choose $t, l$ so that frequent and infrequent items are separated at each level.

$$(\epsilon - \Delta_l)F_1 > (\epsilon - \phi + \Delta_l)F_1$$

- Satisfied if $\phi > 2\Delta_l$ .

$$\text{Let } t = \log(n(\epsilon - \phi)), l \le \log(n(\epsilon - \phi)) \quad .$$

# Solution strategy-3



- Choose $t$ to be twice number of levels: $O(\log(\epsilon - \phi)n)$.
- Then, error probability: $t \cdot 1/(8t) = 1/8$ for each item/interval found as frequent/infrequent.
- Now use previous procedure. Each frequent item is found with probability $2/3$.
- Keep $O(\log(1/\delta))$ independent random permutations and structures associated with it.
- Return an item as frequent if it is discovered to be frequent in more than two-thirds of the structures.

# $F_2$-based frequent items and Dyadic intervals



Problems

- As before: Item frequencies may be high, Interval frequency may be 0 (low).
- New problem: Sum of squares of interval frequencies has no obvious relation to sum of squares of frequencies.
- Problem persists with a random permutation.

# Strategy

- Use random permutation as before.
- Also use random AMS sketches: $\xi : \{1, \ldots, n\} \rightarrow \{-1, +1\}$.
- Stream update $(i, v)$ mapped to $(\pi(i), v \cdot \xi(i))$.

$$f_i \mapsto f_{\pi(i)}\xi(i) \quad .$$

- Keep CountSketch structure at each level.
- Propagate update $(\pi(i), v \cdot \xi(i))$ to CountSketch structure at level $l$. Map $i$ to interval $\lceil \pi(i)/2^l \rceil$ at level $l$.

# $F_2$ at level $l$

- For $j \in [n]$, and $I$ interval at level $l$, let

$$x_{j,I} = \begin{cases} 1 & \text{if } j \in I \\ 0 & \text{otherwise.} \end{cases} \quad \text{So, } f_I = \sum_{j=1}^{n} f_j \xi_j x_{j,I} \ .$$

- So

$$F_2^{(l)} = \sum_I f_I^2 = \sum_I \left( \sum_{j=1}^{n} f_j \xi_j x_{j,I} \right)^2$$

- Taking expectation [Alon Matias Szegedy-like calculation]

$$\mathsf{E}\left[F_2^{(l)}\right] = F_2, \quad \mathsf{Var}\left[F_2^{(l)}\right) = 6F_2^2 \ .$$

# Data structure



Basic data structure

- Keep a random permutation $\pi$ and an $\mathrm{AMS}$ random map $\xi : [n] \rightarrow \{-1, +1\}$.
- $\textsc{CountSketch}$ structure at level $0, 1, \ldots, h$, $h = \lceil \log(n(\epsilon - \phi)) \rceil$.
- Accuracy parameter $\phi/4$, confidence $1 - \delta'$, $\delta' = 1/(8h(\epsilon - \phi)))$.
- $\textsc{CountSketch}$ argument can be repeated to obtain, with probability $1 - \delta'$,

$$|\hat{f}_I - f_I| \leq (\phi F_2)^{1/2} \quad .$$

- Basic data structure repeated $O\left(\log \frac{h}{(\epsilon - \phi)\delta}\right)$ times using independent random bits.

# Algorithm for $\textsc{Frequent}_2(\epsilon, \phi)$



For each copy of the basic data structure

1. *Initialize*: At level $h$, iterate over all $n/2^h \leq \frac{1}{\epsilon - \phi}$ intervals. Estimate $\hat{f}_I$ and retain $I$ if $\hat{f}_I > ((\epsilon - \phi)F_2)^{1/2}$.
2. *Recursion Step*: For each interval retained, estimate frequencies of left and right child intervals using the $\textsc{CountSketch}$ structure at that level. Retain if $\hat{f}_I > ((\epsilon - \phi)F_2)^{1/2}$.
3. Recurse downward, until level $0$ is processed.

Return items $i$ that cross threshold $\hat{f}_i > ((\epsilon - \phi)F_2)^{1/2}$ in at least two-thirds of the basic data structure copies.

# Outline

# Count-min dyadic versus Abs. Deltoids [Corm-Muthu 2005]



Distribution = zipfdiff $(0.1, 0.9)$, $n = 2^{20}$.

| Space (doubles) | Threshold $(\alpha)$ $\alpha F_1$ | Actual No. frequent | Recall Abs. Deltoids | Recall Dyadic |
|---|---|---|---|---|
| 210540 | $2^{-9}$ | 11 | 9 | 10 |
| | $2^{-10}$ | 20 | 14 | 16 |
| | $2^{-11}$ | 40 | 19 | 24 |
| 409600 | $2^{-9}$ | 11 | 10 | 11 |
| | $2^{-10}$ | 20 | 17 | 17 |
| | $2^{-11}$ | 40 | 24 | 29 |
| | $2^{-12}$ | 86 | 37 | 52 |
| 778240 | $2^{-9}$ | 11 | 11 | 11 |
| | $2^{-10}$ | 20 | 18 | 20 |
| | $2^{-11}$ | 40 | 29 | 32 |
| | $2^{-12}$ | 86 | 49 | 61 |
| | $2^{-13}$ | 179 | 73 | 100 |

Frequent items over general data streams

Sumit Ganguly, Abhayendra Singh, Satyam Shankar

Introduction and problem definition

Previous work

Algorithm: Basic ideas

**Experiments**

Conclusions

Distribution = zipfdiff $(0.3, 0.7)$

| Space (doubles) | Threshold $(\alpha)$ $\alpha F_1$ | Actual No. frequent | Recall Abs. Deltoids | Recall Dyadic |
|---|---|---|---|---|
| 210540 | $2^{-9}$ | 3 | 2 | 3 |
| | $2^{-10}$ | 7 | 4 | 4 |
| | $2^{-11}$ | 13 | 5 | 8 |
| 409600 | $2^{-9}$ | 3 | 3 | 3 |
| | $2^{-10}$ | 7 | 4 | 4 |
| | $2^{-11}$ | 13 | 8 | 9 |
| | $2^{-12}$ | 26 | 11 | 16 |
| 778240 | $2^{-9}$ | 3 | 3 | 3 |
| | $2^{-10}$ | 7 | 5 | 4 |
| | $2^{-11}$ | 13 | 10 | 11 |
| | $2^{-12}$ | 26 | 16 | 18 |
| | $2^{-13}$ | 72 | 22 | 26 |

# Dyadic method vs. Variational Deltoids [Cormode-Muthu 2005]



Distribution: zipfdiff $(0.3, 0.3)$, $n = 10^7$.

| Space (doubles) | Threshold $(\alpha F_2)^{1/2}$ | Actual No freq. items | Recall, Prec. Var. Delt. | Recall, Prec. Dyadic | Recal Prec Linea |
|---|---|---|---|---|---|
| 307240 | $2^{-9}$ | 2 | 0 | 0, 0 | 1,0 |
| | $2^{-10}$ | 8 | 0 | 3, 3 | 2,1 |
| | $2^{-11}$ | 24 | 0 | 4, 4 | 3,1 |
| | $2^{-12}$ | 76 | 0 | 10, 8 | 3,1 |
| | $2^{-13}$ | 232 | 0 | 26, 19 | 3,1 |
| 573440 | $2^{-9}$ | 2 | 0 | 0, 0 | 0 |
| | $2^{-10}$ | 8 | 0 | 4, 4 | 0 |
| | $2^{-11}$ | 24 | 0 | 7, 7 | 0 |
| | $2^{-12}$ | 76 | 0 | 18, 18 | 1,0 |
| | $2^{-13}$ | 232 | 0 | 38, 37 | 1,0 |

Distribution: zipfdiff $(0.3, 0.3)$

| Space (doubles) | Threshold $(\alpha F_2)^{1/2}$ | Actual No freq. items | Recall, Prec. Var. Delt. | Recall, Prec. Dyadic | Recal Prec Linea |
|---|---|---|---|---|---|
| 1064960 | $2^{-9}$ | 2 | 0 | 0, 0 | 1,1 |
| | $2^{-10}$ | 8 | 0 | 4, 4 | 1,1 |
| | $2^{-11}$ | 24 | 0 | 10, 10 | 3,2 |
| | $2^{-12}$ | 76 | 0 | 26, 26 | 3,2 |
| | $2^{-13}$ | 232 | 0 | 54, 53 | 3,2 |

Distribution: zipdiff$(0.4, 0.4)$

| Space (doubles) | Threshold $(\alpha F_2)^{1/2}$ | Actual No freq. items | Recall, Prec. Var. Delt. | Recall, Prec. Dyadic | Rec Pre Lin |
|---|---|---|---|---|---|
| 307240 | $2^{-9}$ | 17 | 0 | 8, 8 | 5, |
| | $2^{-10}$ | 42 | 0 | 19, 19 | 7, |
| | $2^{-11}$ | 99 | 0 | 39, 39 | 8, |
| | $2^{-12}$ | 232 | 0 | 60, 59 | 10 |
| | $2^{-13}$ | 540 | 0 | 115, 96 | 10 |
| 573440 | $2^{-9}$ | 17 | 2,2 | 11, 11 | 6, |
| | $2^{-10}$ | 42 | 3,3 | 24, 24 | 6, |
| | $2^{-11}$ | 99 | 0 | 44, 44 | 6, |
| | $2^{-12}$ | 232 | 0 | 91, 91 | 7, |
| | $2^{-13}$ | 540 | 0 | 154, 149 | 7, |

| Space (doubles) | Threshold $(\alpha F_2)^{1/2}$ | Actual No freq. items | Recall, Prec. Var. Delt. | Recall, Prec. Dyadic | Rec Pre Lin |
|---|---|---|---|---|---|
| 1064960 | $2^{-9}$ | 17 | 6 | 12, 12 | 16, |
| | $2^{-10}$ | 42 | 8 | 28, 28 | 21, |
| | $2^{-11}$ | 99 | 2 | 56, 56 | 21, |
| | $2^{-12}$ | 232 | 0 | 109, 109 | 22, |
| | $2^{-13}$ | 540 | 0 | 184, 184 | 24, |

Distribution: zipdiff$(0.5, 0.5)$.

| Space (doubles) | Threshold $(\alpha F_2)^{1/2}$ | Actual No freq. items | Recall, Prec. Var. Delt. | Recall, Prec. Dyadic | Rec Pre Lin |
|---|---|---|---|---|---|
| 307240 | $2^{-9}$ | 42 | 10, 10 | 27, 27 | 8, |
| | $2^{-10}$ | 84 | 4, 4 | 50, 50 | 9, |
| | $2^{-11}$ | 167 | 0 | 77, 77 | 9, |
| | $2^{-12}$ | 334 | 0 | 125, 122 | 9, |
| | $2^{-13}$ | 644 | 0 | 210, 183 | 10, |
| 573440 | $2^{-9}$ | 42 | 14, 14 | 29, 29 | 25, |
| | $2^{-10}$ | 84 | 16, 16 | 56, 56 | 29, |
| | $2^{-11}$ | 167 | 3 , 3 | 95, 95 | 30, |
| | $2^{-12}$ | 334 | 0 | 162, 162 | 31, |
| | $2^{-13}$ | 644 | 0 | 256, 256 | 31, |

Distribution: zipdiff$(0.5, 0.5)$.

| Space (doubles) | Threshold $(\alpha F_2)^{1/2}$ | Actual No freq. items | Recall, Prec. Var. Delt. | Recall, Prec. Dyadic | Rec Pre Lin |
|---|---|---|---|---|---|
| 1064960 | $2^{-10}$ | 84 | 26,26 | 66, 66 | 41, |
| | $2^{-11}$ | 167 | 20,20 | 119, 119 | 44, |
| | $2^{-12}$ | 334 | 7, 7 | 208, 208 | 47, |
| | $2^{-13}$ | 644 | 1, 1 | 359, 359 | 48, |

## Outline

# Conclusions

- Extended the dyadic intervals technique for finding frequent items over general streams.
- Proposed two algorithms for $\textsc{Frequent}_2$ and one algorithm for $\textsc{Frequent}_1$.
- Works well in practice.

THANK YOU!