Resizable sketches

Rasmus Pagh

Sketching and Algorithm Design workshop

Simons Institute for the Theory of Computing, 2023-10-13

A gap between theory and practice

- Most sketches in theory
  - Space parameter *s*, fixed for the lifetime of the sketch
  - Accuracy is a function of the space and other parameters

- Most sketches in practice
  - Want to be able to change space over time to adjust accuracy
  - In Lee Rhodes' talk yesterday:

> Insight: This requires that the sketch size must start very small and grow sublinearly, and ultimately end its growth at a fixed size, or grow very very slowly.

merge

```
public ItemsSketch<T> merge(ItemsSketch<T> other)
```

This function merges the other sketch into this one. The other sketch may be of a different size.

Parameters:

`other` - sketch of this class

Returns:

a sketch whose estimates are within the guarantees of the largest error tolerance of the two merged sketches.

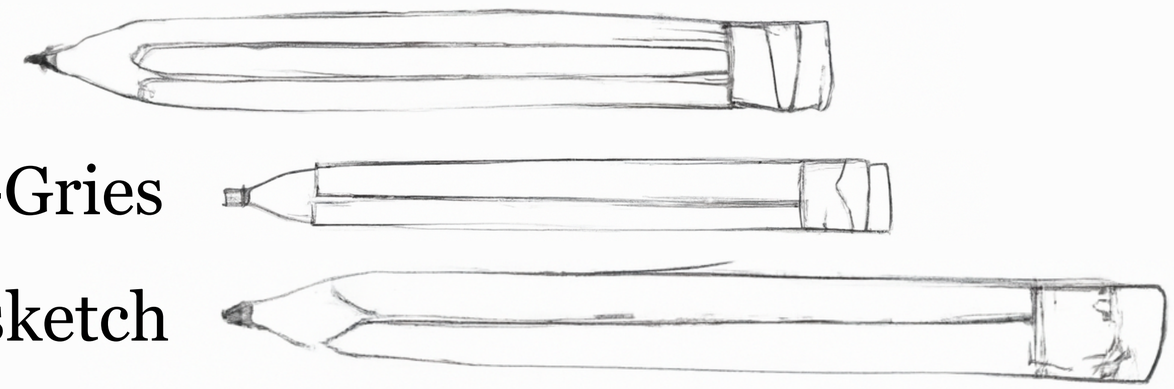
Merging sketches with different configured lgConfigK

This enables a user to union a HLL sketch that was configured with, say, *lgConfigK = 12* with another loaded HLL sketch that was configured with, say, *lgConfigK = 14*.

Why is this important? Suppose you have been building a history of sketches of your customer's data that go back a full year (or 5 or 10!) that were all configured with *lgConfigK = 12*. Because sketches are so much smaller than the raw data it is possible that the raw data was discarded keeping only the sketches. Even if you have the raw data, it might be very expensive and time consuming to reload and rebuild all your sketches with a larger more accurate size, say, *lgConfigK = 14*. This capability enables you to merge last year's data with this year's data built with larger sketches and still have meaningful results.

This talk

- Three examples:
  - Expandable filters
  - Expandable Misra-Gries
  - Expandable KMV sketch



- Few answers, many questions…

Expandable filters

Joint work with Gil Segev, Udi Wieder (FOCS '13), Niv Dayan, Ioana Bercea, Pedro Reviriego (SIGMOD '23)

- Filter (aka. "approximate membership") data structure [Bloom '70]:
  - **Input**: $\varepsilon > 0$, set $S \subseteq U$ of size n
  - **Queries**: membership in superset $S' \supseteq S$ where for each $x \notin S$, $\Pr[x \in S'] \leq \varepsilon$
  - [Carter & Wegman '78]: Possible with $n \log_2(1/\varepsilon) + O(n)$ bits, which is optimal

- Filter without a bound on n [PSW '13]:
  - Elements in S presented one by one, must maintain the approximation $S' \supseteq S$ at all times
  - Lower bound for n insertions to S: must use $\Omega(|S| \log \log n)$ bits at *some* point
  - Upper bound: $O(n \log \log n + n \log(1/\varepsilon))$ bits suffice

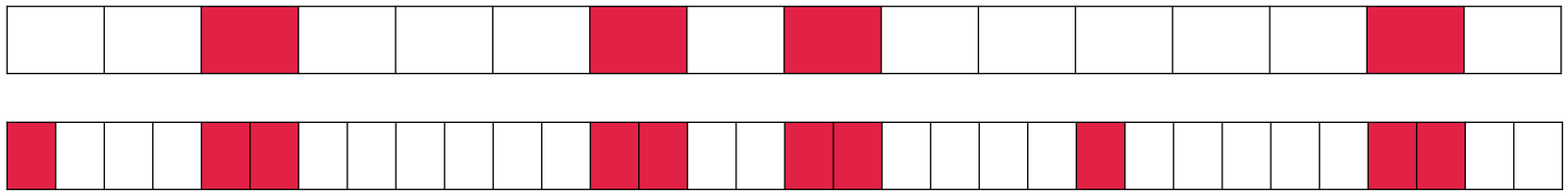
Expandable filters: Upper bound

- **Idea**: Hash function $h : U \to [0,1]$, discretized to m values, $h_m(x) = \lfloor mh(x) \rfloor$
  - $h_m(x)$ represents the $\log_2 m$ most significant bits of $h(x)$
- If n is known: Store $h_{n/\varepsilon}(S)$, compressed to $n \log_2(1/\varepsilon) + O(n)$ bits
  - $S' = \{x \in U \mid h_{n/\varepsilon}(x) \in h_{n/\varepsilon}(S)\}$
  - Does not require S to be known in advance

- For unknown n:
  - First idea: Store $h_m(S)$ where m is smallest power-of-2 $> n/\varepsilon$, double m when needed
  - But: Given $h_m(x)$ there are *two* possible values of $h_{2m}(x)$
  - Can get a *superset* of $h_{2m}(S)$, unfortunately error grows

Expandable filters: Upper bound

- Problem: Keeping false positive rate bounded



$\supseteq h_{16}(S)$

$\supseteq h_{32}(S)$

- Unbounded setting, better idea:
  - Store $h_m(S)$ where m is smallest power-of-2 $> n \log^2(n)/\varepsilon$, double m when needed
  - Collision probability with items inserted in ith expansion phase is $\lesssim \varepsilon/i^2$
  - Total collision probability $\lesssim \sum_i \varepsilon/i^2 \lesssim \varepsilon$
  - $h_m(S)$ now has sparsity $\approx \varepsilon/\log^2(n)$, so need $O(\log(\log(n)/\varepsilon))$ bits per element

Expandable filters in practice

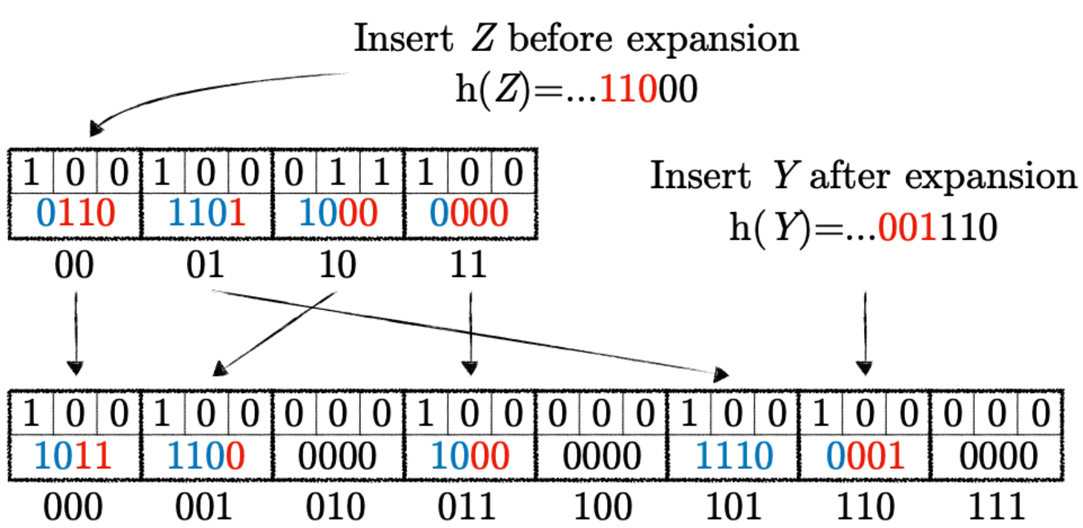
- Issues for practical application:
  - Need a high-performance dynamic succinct membership data structure
  - Supporting deletions, which requires storing a *multiset* (small multiplicities)
  - Adjusting parameters to make lower order terms insignificant
  - …

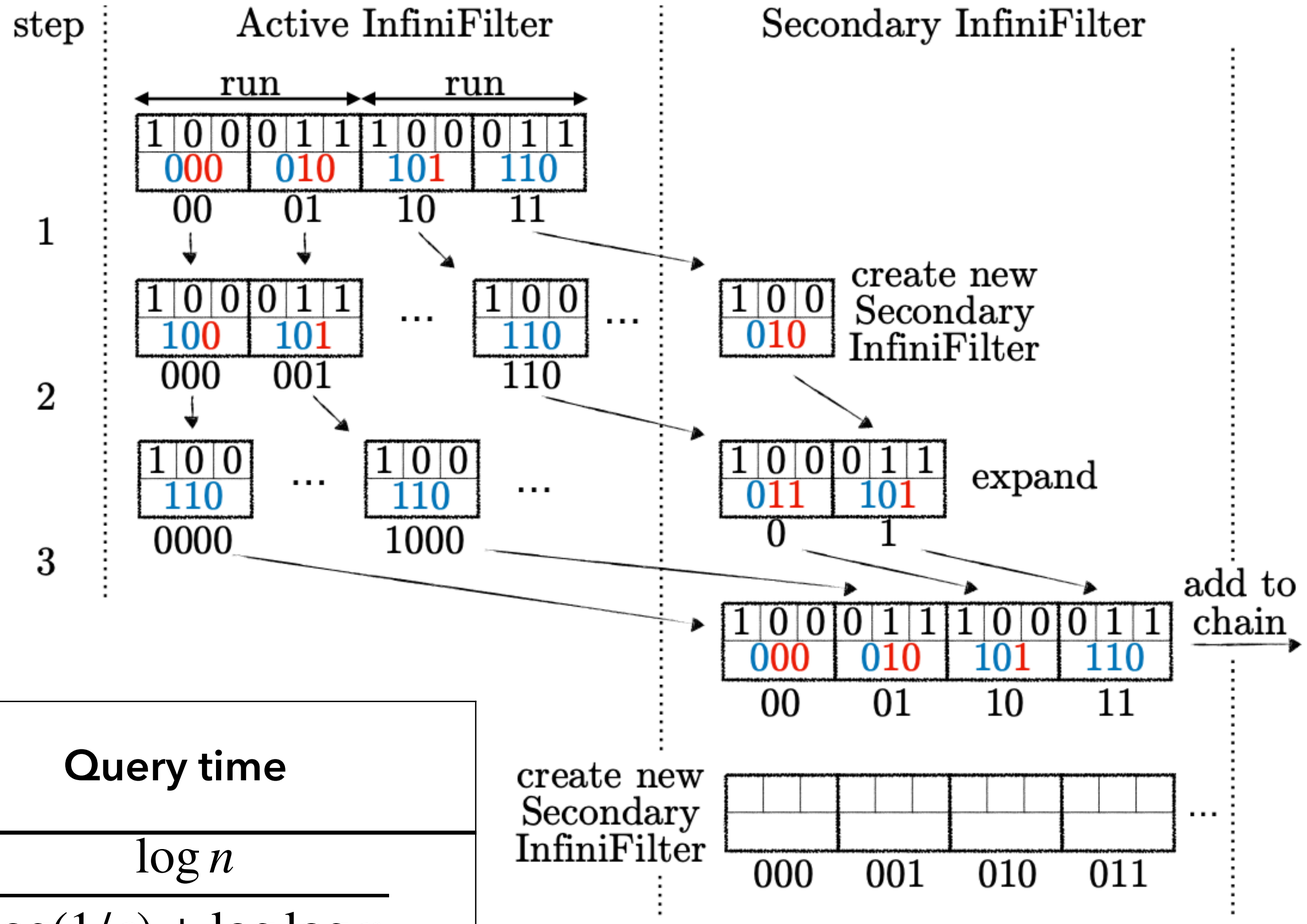
SIGMOD ‘23

InfiniFilter: Expanding Filters to Infinity and Beyond

NIV DAYAN, University of Toronto, Canada
IOANA BERCEA, BARC, IT University of Copenhagen, Denmark
PEDRO REVIRIEGO, Universidad Politécnica de Madrid, Spain
RASMUS PAGH, BARC, University of Copenhagen, Denmark

A peek inside InfiniFilter





Algorithm	Bits/key	Update time	Query time
InfiniFilter	$\log_2(1/\varepsilon) + \log\log n$	$O(1)$	$\dfrac{\log n}{\log(1/\varepsilon) + \log\log n}$
new Aleph Filter	$\log_2(1/\varepsilon) + \log\log n$	$O(1)$	$O(1)$

Misra-Gries sketch

- **Heavy hitters problem**:
  - Stream of items $x_1, x_2, \ldots, x_m \in U$
  - Maintain information about frequent items using space s

- Misra-Gries sketch:
  - Store set of s pairs (x, ℓ_x), where ℓ_x is a counter lower bounding the frequency c_x of x
  - Space usage reduced to $\leq s$ by decrementing all counters and discarding pairs $(x,0)$
  - Invariant: $\ell_x \geq c_x - m/s$

Expandable Misra-Gries sketch

- Expandable sketch:
  - In phase i, insert 2^i elements into Misra-Gries sketch of size s_i, $s_1 \leq s_2 \leq s_3 \leq \ldots$
  - Invariant: $\ell_x \geq c_x - \sum_i 2^i/s_i$
- Example parameterizations:
  - Choose $s_i \approx i^2 2^i/\Delta \lesssim m \log^2(m)/\Delta$ to bound the maximum error to $\sum_i \Delta/i^2 \lesssim \Delta$
  - Choose $s_i \approx 2^{i/2} \lesssim \sqrt{m}$ to bound the maximum error to $\sum_{i=1}^{\lceil \log m \rceil} 2^{i/2} \lesssim \sqrt{m}$

necessary?

Expandable KMV sketch

bottom-k

- **Distinct elements problem**:
  - Stream of items $x_1, x_2, \ldots, x_m \in U$
  - Maintain information about #distinct items
- KMV sketch: For a hash function $h : U \to [0,1]$ maintain the sample $S = \{x_i \mid h(x_i) < \tau\}$, where the threshold τ decreases s.t. at most k elements are stored
  - For #distinct items: $|S|/\max(h(S))$ is a good estimator

- Expandable version:
  - Increasing k does not immediately give us a larger sample S, *but* when sufficiently many new elements have appeared in the stream, the size of S will become k

Expandable KMV sketch example

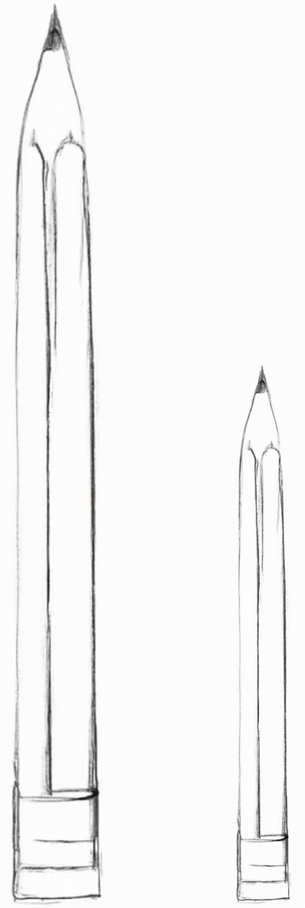
- **Distinct elements problem, increasing precision**:
  - Stream of items $x_1, x_2, \ldots, x_m \in U$
  - Estimate number of distinct elements with additive error $O(m^{2/3})$
  - Application: Estimate $|X \cap Y| = |X| + |Y| - |X \cup Y|$ from sketches of X, Y

- Expandable version (sketch):
  - Let $k = m^{2/3}$
  - Relative error $1 \pm 1/\sqrt{k} = 1 \pm m^{-1/3}$
  - Absolute error at most $m/m^{1/3} = m^{2/3}$

Reducing sketch size

Often much simpler than increasing, e.g.:

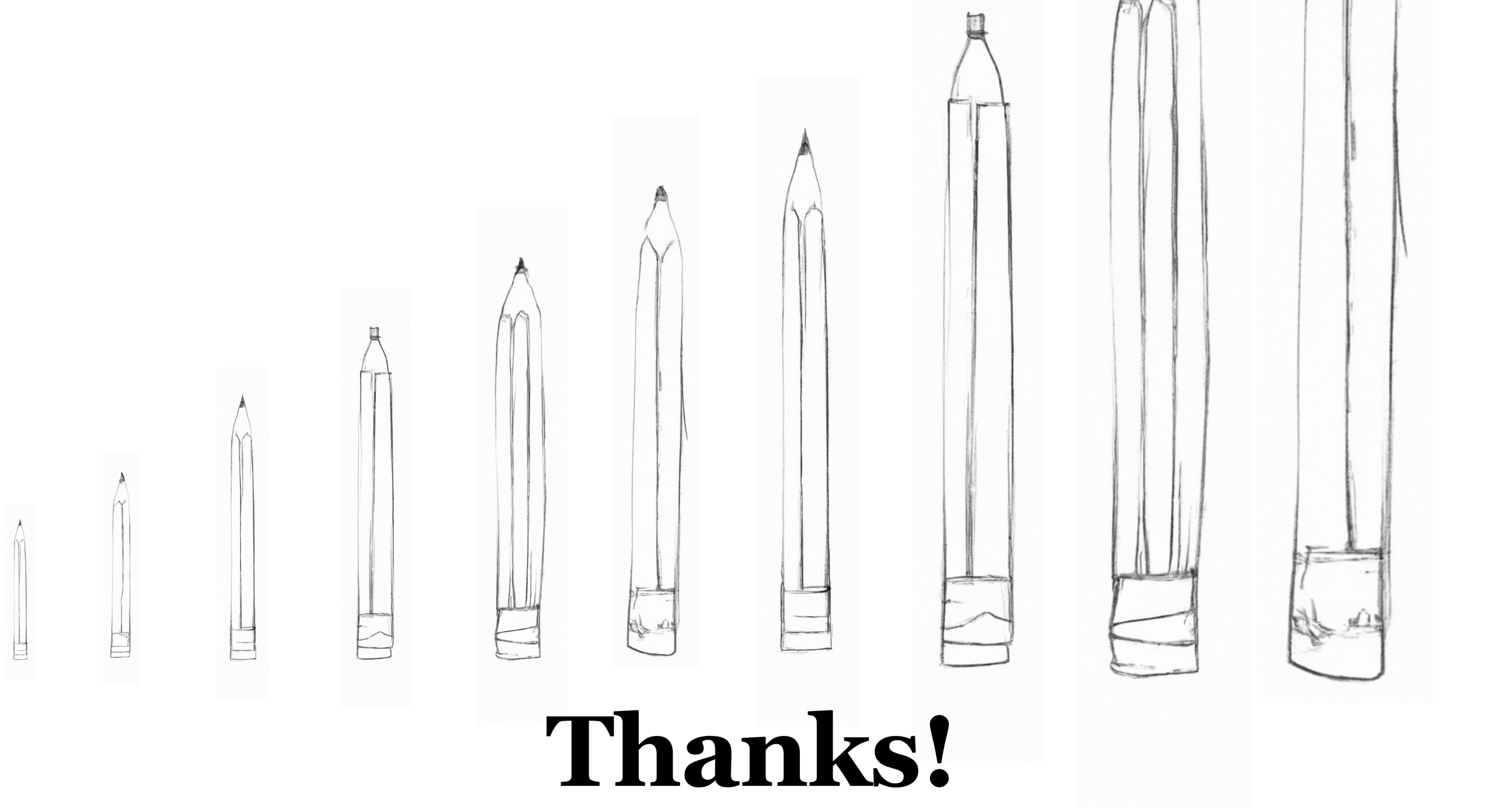
- Filters: Can recompute $h_m(S)$ from $h_{2m}(S)$
- Misra-Gries: Keep decreasing counters until only s pairs left
- KMV: Decrease the threshold τ and recompute S of size k
- HLL: Compute pairwise max of adjacent counters
- …



Many open questions

- Better upper bounds?
  - expandable HLL?
  - error depending on #distinct elements rather than stream length?
  - bounded error heavy hitters with $o(\log m)$ multiplicative space overhead?
  - …

- Considering other sketching problems: Quantiles, graphs, matrices,…
- Lower bounds?
- Handling more general settings, e.g., turnstile streams?



Thanks!