Simple and Efficient Cardinality Estimation in Data Streams*

Seth Pettie
University of Michigan
pettie@umich.edu

Dingyu Wang
University of Michigan
wangdy@umich.edu

Longhui Yin
Tsinghua University
ylh17@mails.tsinghua.edu.cn

Abstract

We study sketching schemes for the cardinality estimation problem in data streams, and advocate for measuring the efficiency of such a scheme in terms of its MVP: Memory-Variance Product, i.e., the product of its space, in bits, and the relative variance of its estimates.

Under this natural metric, the celebrated HyperLogLog sketch of Flajolet et al. (2007) has an MVP approaching $6(3\ln 2 - 1) \approx 6.48$ for estimating cardinalities up to $2^{64}$. Applying the Cohen/Ting (2014) martingale transformation results in a sketch Martingale HyperLogLog with MVP $\approx 4.16$, though it is not composable. Recently Pettie and Wang (2020) proved that it is possible to achieve MVP approaching $\approx 1.98$ with a composable sketch called Fishmonger, though the time required to update this sketch is not constant.

Our aim in this paper is to strike a nice balance between extreme simplicity (exemplified by Martingale HyperLogLog) and extreme information-theoretic efficiency (exemplified by Fishmonger). We develop a new class of “curtain” sketches that are a bit more complex than Martingale LogLog but with substantially better MVPs, e.g., Martingale Curtain has MVP $\approx 2.31$. We also prove that Martingale Fishmonger has an MVP of around 1.63, and conjecture this to be an information-theoretic lower bound on the problem, independent of update time.

![Graph showing empirical distribution of estimates](image)

The figure shows the empirical distribution of estimates after 100,000 runs. All three sketches use 1,200 bits. Martingale Curtain exhibits significantly better concentration in its estimates relative to HyperLogLog or Martingale LogLog.

*This work was supported by NSF grants CCF-1637546 and CCF-1815316.
1 Introduction

The cardinality estimation (aka \( F_0 \) estimation or Distinct Elements) is a fundamental problem in streaming and sketching with diverse applications in databases\(^1\) [12, 22], network monitoring [5, 8, 11, 38], nearest neighbor search [32], caching [35], and genomics [2, 17, 30, 37]. In the sequential setting of this problem, we receive the elements of a multiset \( A = \{a_1, a_2, \ldots, a_N\} \) one at a time. We maintain a small sketch \( S \) of the elements seen so far, such that an estimate \( \hat{\lambda}(S) \) can be generated for the true cardinality \( \lambda = |A| \). The distributed setting is similar, except that streams \( A_1, \ldots, A_z \) are processed on distinct machines and sketched as \( S_1, \ldots, S_z \) such that the joint sketch \( S = A_1 \cup \cdots \cup A_z \) is a function of \( S_1, \ldots, S_z \). Until relatively recently [10, 15, 25, 34] there was no need to distinguish between the sequential and distributed settings because all sketches worked in both.

Models. Cardinality Estimation has been studied from two distinct perspectives: the random oracle model and what we call the standard model. In the random oracle model we have access to a uniformly random hash function \( h : [U] \to [0, 1] \) from the universe of elements \([U]\) (say \( U = 2^{64}\)) to \([0, 1]\). In the standard model we can generate uniformly random bits, but then need to explicitly store any hash functions, which counts towards the space usage of the sketch. The state-of-the-art in the standard model [7, 28] stores an \( O(\epsilon^{-2} \log \delta^{-1} + \log U)\)-bit sketch, and reports estimates such that \( \hat{\lambda} \in [(1 - \epsilon)\lambda, (1 + \epsilon)\lambda] \) with probability \( 1 - \delta \). According to [1, 26, 27], this is simultaneously asymptotically optimal in \( \epsilon, \delta, \) and \( U \). Sketches in the standard model [1, 3, 4, 7, 23, 28] typically use space that is measured asymptotically, with significant leading constants that make them unsuitable for practical applications.

In this paper we assume the random oracle model, and focus mainly on simple and practical sketches in the sequential setting.

Estimation Quality. In the random oracle model \( \hat{\lambda} \) is typically an unbiased (or asymptotically unbiased) estimate of \( \lambda \). Given that this holds, we measure the quality of the estimation by its relative variance \( \text{Var}(\hat{\lambda} | \lambda)/\lambda^2 \) or relative standard deviation \( \sqrt{\text{Var}(\hat{\lambda} | \lambda)}/\lambda \), often called the standard error.

The number of sketches analyzed in the random oracle model is quite large [6, 9, 10, 13, 15, 16, 18, 19, 21, 24, 25, 29, 31, 33, 34]. In many of these papers the focus is on the design and analysis of good estimators. The sketches themselves are typically quite simple, and involve variations on 2-3 recurring ideas. We explain how the LogLog [16, 20] and PCSA [21] sketches work, and illustrate how several other sketches are variations on these two.

A PCSA sketch \( S \in \{0, 1\}^{\log_2 U \times m} \) is an array of bits, initially zero. When processing \( a_i \), we interpret the hash value \( h(a_i) = (j, k) \in [\log_2 U] \times [m] \) as a pair of integers, which occurs with probability \( m^{-1} \cdot 2^{-(j+1)} \). (Here \([m] = \{0, \ldots, m - 1\}\).) The state of the sketch after processing \( \{a_1, \ldots, a_i\} \) is \( S_i \). The sketch bit \( S_i(j, k) = 1 \) iff there is some element \( a_{i'} \in \{a_1, \ldots, a_i\} \) for which

---

\(^1\)There has been widespread industrial adoption, e.g., https://looker.com/blog/practical-data-science-amazon-announces-hyperloglog, https://tech.nextroll.com/blog/data/2013/07/10/hll-minhash.html, http://content.research.neustar.biz/blog/hll.html, https://www.amobee.com/blog/counting-towards-infinity-next-generation-data-warehousing-part-i/, https://docs.aws.amazon.com/redshift/latest/dg/x_COUNT.html, https://medium.com/unsplash/hyperloglog-in-google-bigquery-7145821ac81b, https://thoughtbot.com/blog/hyperloglogs-in-redis, https://redislabs.com/redis-best-practices/counting/hyperloglog/, https://redditblog.com/2017/05/24/view-counting-at-reddit/
The LogLog sketch of Durand and Flajolet [16] is similar, except that we do not maintain the whole log \( U \)-bit vector for each column, just the position of its most significant 1-bit. In other words, \( S \in ([\log_2 U])^m \) and \( S_i(k) = \max\{j \mid \exists i' \in \{1, \ldots, i\}. h(a_{i'}) = (j, k)\}. \) Durand and Flajolet [16] showed that an estimation \( \hat{\lambda}(S) \propto m2^{m-1} \sum_k S(k) \) achieves standard error tending to \( \approx 1.3/\sqrt{m} \). Flajolet et al.’s HyperLogLog [20] uses the same sketch, but with a different estimation \( \tilde{\lambda}(S) \propto m^2(\sum_k 2^{-S(k)})^{-1} \), which has standard error tending to \( \approx 1.04/\sqrt{m} \). \(^2\)

Most other sketches are variations on these themes. For example, many sketches interpret the hash function as a real value (or integer from a large range) and store the minimum hash value in each of \( m \) substreams \([9,14,15,24,29]\). The LogLog sketch is essentially a base-2 discretization of this idea, with the stream split into \( m \) substreams. The Discrete Max Count [34] can be thought of as the analogue of LogLog without splitting into substreams, and with a different discretization. The Multi-resolution Bitmap [18] is essentially the same as PCSA.

**Composability.** Once the hash function \( h \) is fixed, the state of a PCSA or LogLog sketch after processing \( A \) is clearly a function of the set of distinct elements in \( A \), i.e., it is independent of the multiplicity of elements and the permutation in which they were processed. Formally, the transition function of the sketch is commutative and idempotent; see [31]. Any sketch satisfying these two properties is composable and can therefore be used to synthesize sketches from distributed data streams.

**Non-composable Sketching.** Considering the 40-year history of the Cardinality Estimation problem, the idea of non-composable sketching has appeared quite recently, and to very little fanfare. This is utterly astounding, considering the simplicity of the sketches and the superior quality of their estimates. Moreover, it will become clear below that adapting an existing implementation only requires adding a few lines of code.

Specific non-composable sketches were developed by Chen et al. [10] (the S-Bitmap) and Helmi et al. [25] (Recordinality). In 2014, Cohen [15] and Ting [34] independently described a mechanical method for transforming any composable sketch into a non-composable one for the sequential setting whose cardinality estimates are both unbiased and have lower variance than the original sketch. (The resulting sketch cannot be used in the distributed setting.)

The Cohen/Ting construction is, like all great inventions, obvious only in retrospect. Let \( S_i \) be the state of any composable (i.e., commutative and idempotent) sketch after processing elements \( \{a_1, \ldots, a_i\} \). The state of the transformed sketch is \( (S_i, \hat{\lambda}_i) \), i.e., the current cardinality estimate \( \hat{\lambda}_i \) is stored explicitly, where \( \hat{\lambda}_0 = 0 \). Define \( P_{i+1} = \Pr(S_{i+1} \neq S_i \mid S_i, a_{i+1} \notin \{a_1, \ldots, a_i\}) \) to be the probability that the \((i + 1)\)th element \( a_{i+1} \) changes the sketch, under the assumption that it has not been seen before. \(^3\) Then the next state of the transformed sketch is

\[
(S_{i+1}, \hat{\lambda}_i + P_{i+1}^{-1} \cdot \mathbb{I}[S_{i+1} \neq S_i]),
\]

where \( \mathbb{I}[E] \) is the indicator variable for event \( E \). Observe that if \( a_{i+1} \in \{a_1, \ldots, a_i\} \) then \( (S_{i+1}, \hat{\lambda}_{i+1}) = (S_i, \hat{\lambda}_i) \).

\(^2\)I.e., as \( m \to \infty \), the standard error tends these bounds from above.

\(^3\)Here we are assuming something slightly stronger, that \( S_{i+1} \) is a function of \( S_i \) and \( h(a_{i+1}) \) (the hash value of the \((i + 1)\)th distinct element), but not \( a_{i+1} \) per se. In the RANDOM ORACLE MODEL, the distribution of \( h(a_{i+1}) \) is independent of \( a_{i+1} \) and \( S_i \), and therefore \( P_{i+1} \) is well-defined.
(\(S_i, \hat{\lambda}_i\)) by commutativity and idempotence, but if \(a_{i+1} \not\in \{a_1, \ldots, a_i\}\), then
\[
\mathbb{E}(\hat{\lambda}_{i+1} \mid \hat{\lambda}_i) = \hat{\lambda}_i + P_{i+1}^{-1} \cdot P_{i+1} = \hat{\lambda}_i + 1.
\]

Thus, if \(\lambda_i = |\{a_1, \ldots, a_i\}|\) is the true cardinality of the sequence up to \(a_i\), the sequence \((\hat{\lambda}_i - \lambda_i)_{i \geq 0}\) is a martingale. We call this the martingale transform and prefix any sketch derived by this transform with the prefix Martingale, e.g., Martingale LogLog or Martingale PCSA.

Although Martingale sketches are trivially unbiased, analyzing their variance is not as simple. Cohen [15] and Ting [34] proved that the Martingale \(m\)-Min sketch (keep the \(m\) smallest real hash values) has standard error \(\sqrt{1/(2m)} = 0.71/\sqrt{m}\). Cohen [15] argued that the standard error of Martingale LogLog is \(\approx \sqrt{3/(4m)} \approx 0.866/\sqrt{m}\) whereas Ting [34] estimated it to be \(\approx \sqrt{1/(2\alpha_m m)}\), which tends to \(\sqrt{\ln 2/m} \approx 0.8326/\sqrt{m}\) as \(m \to \infty\). Here \(\alpha_m = \left(m \int_0^\infty (\log_2 \left(\frac{2+u}{1+u}\right))^m du\right)^{-1}\) is the coefficient of Flajolet et al.’s HyperLogLog estimator. We will discuss the variance of Martingale estimators in Section 3.

1.1 Towards Absolute Optimality in Cardinality Estimation

The astute reader will notice that looking at standard errors alone is insufficient for adjudicating which of two composable or non-composable sketches is better. For example, the PCSA sketch occupies \(m \log U \) bits with relative variance \(\approx (0.78)^2/m\) whereas the HyperLogLog sketch occupies \(m \log U \) bits with relative variance \(\approx (1.04)^2/m\). Despite the poorer performance in terms of the parameter “\(m\),” HyperLogLog is better than PCSA because the product of its memory and variance is smaller. But is this a fair comparison? It certainly seems fair because the actual memory footprints of PCSA and HyperLogLog are \(m \log U \) and \(m \log \log U\). However, accepting the nominal encoding of PCSA and HyperLogLog is somewhat unfair because the probability distribution over the state-space is highly concentrated. Indeed, both sketches have entropy \(O(m)\), so viewed through the prism of entropy it is not \(a\ priori\) clear which is superior.

In a companion paper [31], we captured this objective by defining the Fish-number (Fisher- Shannon number) of a sketch as the product of the reciprocal of its Fisher information (which controls the variance of an efficient estimator) and its Shannon entropy (which controls the memory of a compressed representation). The main result of [31] is that it is actually PCSA that is “better” than (Hyper)LogLog, having Fish-number \(H_0/I_0 \approx 1.98\), where
\[
H_0 = (\ln 2)^{-1} + \sum_{k=1}^{\infty} k^{-1} \log_2(1 + 1/k)
\]
and
\[
I_0 = \zeta(2) = \pi^2/6.
\]

All base-\(q\) versions of (Hyper)LogLog are worse than \(H_0/I_0\), but tend to \(H_0/I_0\) in the limit as \(q \to \infty\). The Fishmonger sketch introduced in [31] is composable (commutative and idempotent), occupies \(O(\log^2 \log U) + (1 + o(1))(H_0/I_0)b \approx 1.986b\) bits and has standard error \((1 + o(1))/\sqrt{b}\). It was conjectured in [31] that \(H_0/I_0\) is a lower bound on the Fish-number\(^6\) of any scale-invariant composable sketch.\(^7\)

---

\(^4\)Cohen [15] called these Historical Inverse Probability (HIP) sketches and Ting [34] applied the prefix Streaming to emphasize that they can be used in the single-stream setting, not the distributed setting.

\(^5\)In most sketches these quantities are not constant, but multiplicatively periodic with some period \(q > 1\). Some care is needed to “smooth” out these functions, making them constant in the limit. See [31] or Section 3.2.3.

\(^6\)(I.e., the memory-variance product, where “memory” is obtained from compression to the entropy bound.)

\(^7\)The scale-invariant assumption [31] basically says that the sketch must work equally well, regardless of the scale of \(\lambda\). It is straightforward to beat \(H_0/I_0\) if one assumes, for example, that we only care about \(\lambda \in [10^6, 2 \cdot 10^6]\).
The limiting memory-variance product $H_0/I_0 \approx 1.98$ is substantially better than a typical implementation of HyperLogLog with $U = 2^{64}$, which tends to roughly 6.46. However, Fishmonger is impractical in most settings, being based on a smoothed, entropy-compressed variant of PCSA, with a maximum-likelihood estimator. Using arithmetic coding [36] results in a sketch that requires linear time to decode/encode between updates. Just as Martingale LogLog improves the variance of (Hyper)LogLog, Martingale Fishmonger improves Fishmonger, but it is just as complicated in its encoding complexity.

Define the “MVP” (memory-variance product) of a sketch to be the product of its memory footprint (number of bits) and the relative variance of its cardinality estimates. For simplicity we often consider the limiting MVP, as $m \rightarrow \infty$, where $m$ is a parameter that controls the size of the sketch.

### 1.2 New Results

The goal of this work is to develop simple and efficient sketches for cardinality estimation, with the understanding that optimal simplicity and efficiency cannot be achieved simultaneously. At present, we have the following extreme options with little in between.

**Prioritize Simplicity.** Use HyperLogLog [20], which has an MVP approaching $\approx 6.46$ for $U = 2^{64}$ (6-bit counters), or in a sequential setting, Martingale LogLog [15,34], with MVP $\approx 4.16$. Each update to the sketch involves one evaluation of $h(\cdot)$ and a few random memory accesses.\(^8\)

**Prioritize Information Theoretic Efficiency.** Use Fishmonger, with limiting MVP $= H_0/I_0 \approx 1.98$, or in a sequential setting, Martingale Fishmonger, with some even better MVP (formally analyzed in this paper).

The first option is wasteful in terms of space and the second too slow and complicated for all but the most memory-conscious applications. Our contributions are as follows.

\(^8\)HyperLogLog reads and possibly writes $S(k)$, whereas Martingale LogLog may also have to read/write cells storing $\hat{\lambda}_i$ and $P_{i+1}$.
• We give a relatively simple way to analyze the limiting variance of generic Martingale-type sketches, which is inspired by Ting’s [34] perspective. For example, it is shown that the limiting MVP of Martingale Fishmonger is $H_0/2 < 1.63$. We conjecture that this is the optimal constant for cardinality estimation in the sequential setting.

• We introduce a new class of composable sketches called curtain sketches that slightly more complicated than PCSA or (Hyper)LogLog but with dramatically better MVPs. Martingale Curtain is relatively simple and has MVP ≈ 2.31. The SecondCurtain sketch is more complicated than Curtain but much simpler than Fishmonger. Martingale SecondCurtain has limiting MVP ≈ 2.06. See Table 1.

To ground the discussion, in order to guarantee a standard error of 1% for multisets of the universe $[2^{64}]$, HyperLogLog requires about 7.91 KiB, Martingale LogLog about 5.08 KiB, and Martingale Curtain about 2.82 KiB. Martingale SecondCurtain and Martingale Fishmonger are variable length sketches, whose expected sizes are around 2.52 KiB and 1.99 KiB, respectively; however to make them resilient to fluctuations in length over time, their memory allocation should be a bit larger, e.g., in this case an extra 0.1 kb should suffice.⁹

1.3 Related Work: Sketch Compression

It is well known that the $m \log \log U$-bit space of HyperLogLog is wasteful from an information-theoretic point of view. The state-of-the-art sketches in the standard model due to Kane, Nelson, and Woodruff [28] and Błasiok [7], compress this information to $O(m)$ bits. However, prior efforts to compress HyperLogLog from a practical perspective have not actually been successful.

Xiao, Zhou, Chen, and Luo [39] proposed a refinement of HyperLogLog called HLL-Tailcut+, which maintains the minimum counter value and a 3-bit offset for each counter; offset values {0, ..., 6} retain their nominal meaning but “7” indicates any counter value more than the minimum plus 6. They claimed that with suitable adjustments in the estimation function, this scheme has relative variance $1/m$. This claim is incorrect. HLL-Tailcut+ is neither commutative nor idempotent. The two sequences of [31, Appendix A] suffice to show that the relative bias and variance of HLL-Tailcut+ are lower bounded by absolute constants, independent of $m$.

Sedgewick [33] (unpublished) proposed a 134-bit sketch called HyperBitBit, which can be construed as a heuristic compression of HyperLogLog that tries to remember the two most informative counter-values of HyperLogLog. Sedgewick did not claim that HyperBitBit has any theoretical guarantees, but did observe that it usually gets less than 10% relative error. However, due to not being commutative or idempotent, this scheme can be made to have relative errors usually larger than 20%; see [31, Appendix A]. Moreover, the problem is not mitigated by making the sketch longer.

The moral here is that any heuristic compression of a sketch yields a new sketch-transition function, and the algebraic properties of this function matter. For composable sketches, it is critical that the transition function be commutative and idempotent. If we apply the Martingale transformation, then commutativity is unnecessary, but it is important that duplicate elements have no effect on the sketch.

Organization. In Section 2 we present the Curtain and SecondCurtain sketches. In Section 3 we give a framework for analyzing the limiting variance of Martingale sketches. Section 4 gives

⁹The crux of Fishmonger’s analysis [31] is to show that $O(\sqrt{m \log m + \log^2 \log U})$ bits suffices to avoid a memory overflow at all times, w.h.p.
closed form expressions for the limiting variance of base-q Martingale LogLog, Martingale PCSA, and Martingale Curtain. The analysis of Martingale SecondCurtain and some other proofs appear in the Appendix.

We give some empirical validation of our theoretical analysis in Section 5. Section 6 concludes with some open problems.

2 Curtain Sketches

All the sketches discussed in this paper can be regarded as coarsely representing the state of a dartboard after \( \lambda \) darts (hash values) have been thrown at it.\(^{10}\) The dartboard is the real unit square \([0, 1]^2\) and \(h : [U] \to [0, 1]^2\) assigns each element a point uniformly at random. The dartboard is partitioned into cells. A state of the sketch is just a partition of the cells into occupied and free, with the following constraints. (i) every cell containing at least one dart must be occupied, but occupied cells may contain no darts, (ii) if a dart is thrown at an occupied cell, the sketch may not change state. As a consequence of (i), if a dart is thrown at a free cell, the sketch must change state. Thus, the probability of seeing a state-change on the next dart is exactly the total area of all free cells. As a consequence of (i,ii), no cell, once occupied, can be made free.

Base-q PCSA and LogLog use the same partition into cells, as depicted in Figure 1. Let \(\text{Cell}(j, i)\) be the cell in column \(i\) covering the vertical interval \([q^{-(j+1)}, q^{-j})\). In a PCSA sketch, the occupied cells are precisely those with at least one dart. In LogLog, the occupied cells in each column are contiguous, extending to the highest cell containing a dart. Figure 2(a) depicts a PCSA state, and Figure 2(b) depicts the corresponding LogLog state. Cells are drawn with uniform sizes for clarity.

Consider the vector \(v = (g_0, g_1, \ldots, g_{m-1})\) where \(\text{Cell}(g_i, i)\) is the highest occupied cell in LogLog/PCSA. The curtain of \(v\) w.r.t. allowable offsets \(\emptyset\) is a vector \(v_{\text{curt}} = (\hat{g}_0, \hat{g}_1, \ldots, \hat{g}_{m-1})\) such that (i) \(\forall i \in [1, m - 1], \hat{g}_i - \hat{g}_{i-1} \in \emptyset\), and (ii) \(v_{\text{curt}}\) is the minimal such vector dominating \(v\), i.e., \(\forall i. \hat{g}_i \geq g_i\). Although we have described \(v_{\text{curt}}\) as a function of \(v\), it is clearly possible to maintain \(v_{\text{curt}}\) as darts are thrown, without knowing \(v\).

We have an interest in \(|\emptyset|\) being a power of 2 so that curtain vectors may be encoded efficiently, as a series of offsets. On the other hand, it is most efficient if \(\emptyset\) is symmetric around zero. For

\(^{10}\)This is just a more constrained version of Ting’s \([34]\) area-cutting process.
Figure 2: (a) The state of a PCSA sketch records precisely which cells contain a dart (gray); all others are empty (yellow). (b) The state of the corresponding LogLog sketch.

Figure 3: The “sawtooth” cell partition.

these reasons, we use a base-\(q\) “sawtooth” cell partition of the dartboard; see Figure 3. Henceforth Cell(\(j, i\)) is defined as usual, except \(j\) is an integer when \(i\) is even and a half-integer when \(i\) is odd. Then the allowable offsets are \(O_a = \{-a - 1/2, -a - 3/2, \ldots, -1/2, 1/2, \ldots, a - 3/2, a - 1/2\}\), for some \(a\) that is a power of 2.

2.1 The Curtain Sketch

Once again let Cell(\(g_i, i\)) be the highest cell containing a dart in column \(i\) in the sawtooth cell partition, and let \(v_\text{curt} = (\hat{g}_i)\) be the curtain of \(v = (g_i)\) w.r.t. \(O = O_a\). We say column \(i\) is in tension if \((\cdots, \hat{g}_{i-1}, \hat{g}_i - 1, \hat{g}_{i+1}, \cdots)\) is not a valid curtain, i.e., if \(\hat{g}_i - \hat{g}_{i-1} = \min(O)\) or \(\hat{g}_{i+1} - \hat{g}_i = \max(O)\). In particular, if column \(i\) is not in tension, then Cell(\(\hat{g}_i, i\)) must contain at least one dart, for if it contained no darts the curtain would be dropped to \(\hat{g}_i - 1\) at column \(i\). However, if column \(i\) is in tension, then Cell(\(\hat{g}_i, i\)) might not contain a dart.

The Curtain sketch encodes \(v_\text{curt} = (\hat{g}_i)\) w.r.t. the base-\(q\) sawtooth cell partition and offsets \(O_a\), and a bit-array \(b = \{0, 1\}^{h \times m}\). This sketch designates each cell occupied or free as follows.

Rule 1. If column \(i\) is not in tension then Cell(\(\hat{g}_i, i\)) is occupied, and \(b(\cdot, i)\) encodes the status of the \(h\) cells below the curtain, i.e., Cell(\(\hat{g}_i - (j + 1), i\)) is occupied iff \(b(j, i) = 1\), \(j \in \{0, \ldots, h - 1\}\).

Rule 2. If column \(i\) is in tension, then Cell(\(\hat{g}_i - j, i\)) is occupied iff \(b(j, i) = 1\), \(j \in \{0, \ldots, h - 1\}\).
Figure 4: A Curtain sketch w.r.t. $O = \{-3/2, -1/2, 1/2, 3/2\}$ and $h = 1$. (a): Gray cells contain at least one dart; light yellow cells contain none. The curtain $\nu_{\text{curt}} = (\hat{g}_i)$ is highlighted with a pink boundary. (b) Columns that are in tension have a $\star$ in their curtain cell. All dark gray cells are occupied and all dark yellow cells are free according to Rule 3. All other cells are occupied/free (light gray, light yellow) according to Rules 1 and 2.

**Rule 3.** Every cell above the curtain is free (Cell$(\hat{g}_i + j, i)$, when $j \geq 1$) and all remaining cells are occupied.

Figure 4 gives an example of a Curtain sketch, with $O = \{-3/2, -1/2, 1/2, 3/2\}$ and $h = 1$. (The base $q$ of the cell partition is unspecified in this example.)

**Theorem 1.** Consider the Martingale Curtain sketch with parameters $q, a, h$ (base $q$, $O_a = \{-(a - 1/2), \ldots, a - 1/2\}$, and $b \in \{0, 1\}^{h \times m}$), and let $\hat{\lambda}$ be its estimate of the true cardinality $\lambda$.

1. $\hat{\lambda}$ is an unbiased estimate of $\lambda$.

2. The relative variance of $\hat{\lambda}$ is:

$$\frac{1}{\lambda^2} \text{Var}(\hat{\lambda} | \lambda) = \frac{(1 + o_{\lambda/m}(1) + o_m(1))q \ln q}{2m(q - 1)} \left( \frac{q - 1}{q} + \frac{2}{q^h(q^a - 1/2 - 1)} + \frac{1}{q^{h+1}} \right),$$

As a result, the limiting MVP of Martingale Curtain is

$$\text{MVP} = (\log_2(2a) + h) \times \frac{q \ln q}{2(q - 1)} \left( \frac{q - 1}{q} + \frac{2}{q^h(q^a - 1/2 - 1)} + \frac{1}{q^{h+1}} \right).$$

**Proof.** Follows from Theorem 4 and 7.

Here $o_{\lambda/m}(1)$ and $o_m(1)$ are terms that go to zero as $m$ and $\lambda/m$ get large. The best parameterization of Theorem 1 (with $a$ a power of 2 and $h$ an integer) is to set $q = 2.91$, $a = 2$, and $h = 1$, exactly as in the example in Figure 4. This uses $\log \log U + 3(m - 1)$ bits to store the sketch proper, $\log U$ bits\(^\text{11}\) to store $\hat{\lambda}$, and achieves a limiting MVP $\approx 2.31$. In other words, to achieve a standard error $1/\sqrt{b}$, we need about $2.31b$ bits.

\(^\text{11}\)It is fine to store an approximation $\tilde{\lambda}$ of $\hat{\lambda}$ with $O(\log m)$ bits of precision.
Figure 5: (a) Gray cells contain at least one dart; yellow cells contain none. The second curtain $v_{\text{curt}} = (\hat{g}_i)$ is indicated by a pink boundary. (b) Columns that are in tension are marked with a $\star$. The $\delta$-vector is indicated below.

2.2 The SecondCurtain Sketch

The SecondCurtain sketch is a variable-length sketch that lies between Curtain and Fishmonger, both in terms of conceptual complexity and MVP. It is based on the observation that the positions of the second highest cell hit by a dart in each column exhibits considerably less variation than the highest.

Once again, consider a sawtooth cell partition. Initially all cells of the form $\text{Cell}(j, \cdot), j \in \{0, 1/2, 1, 3/2\}$ are regarded as containing darts. Define $v = (g_i)$ to be such that $\text{Cell}(g_i, i)$ is the second highest cell in column $i$ containing a dart, and let $v_{\text{curt}} = (\hat{g}_i)$ be the curtain of $v$ w.r.t. $\mathcal{O}_a = \{-a - 1/2, \ldots, a - 1/2\}$.

Observe that if column $i$ is not in tension, then it must be the case that some $\text{Cell}(\hat{g}_i + j, i)$ with $j \geq 1$ contains at least one dart, for otherwise the curtain height $\hat{g}_i$ at column $i$ should be lower. On the other hand, if column $i$ is in tension, then it may be that for all $j \geq 1$, $\text{Cell}(\hat{g}_i + j, i)$ contains no dart. The SecondCurtain sketch consists of $v_{\text{curt}} = (\hat{g}_i)$, and a variable-length encoding of $\delta = (\delta_0, \ldots, \delta_{m-1})$, where each $\delta_i$ encodes a positive integer in unary, i.e., $1=1$, $2=01$, $3=001$, and so on. The interpretation of the sketch is as follows.

Rule 1. If column $i$ is not in tension then $\text{Cell}(j, i)$ is occupied iff $j \leq \hat{g}_i$ or $j = \hat{g}_i + \delta_i$.

Rule 2. If column $i$ is in tension, then $\text{Cell}(j, i)$ is occupied iff $j \leq \hat{g}_i$ or $j = \hat{g}_i + (\delta_i - 1)$. I.e., $\delta_i = 1$ if all darts in column $i$ are in cells at or below $\text{Cell}(\hat{g}_i, i)$.

Figure 5 gives an example of a SecondCurtain sketch.

Theorem 2. Consider the Martingale SecondCurtain sketch with parameters $q, a$, and let $\hat{\lambda}$ be its estimate of the true cardinality $\lambda$.

1. $\hat{\lambda}$ is an unbiased estimator of $\lambda$.

2. Define $\varphi(x) = \sum_{i=0}^\infty \left( e^{x \frac{q^{i+1} - 1}{q} - 1} \right)$. The relative variance $\frac{1}{\lambda^2} \text{Var}(\hat{\lambda} | \lambda)$ is

$$\frac{(1 + o_{\lambda/m}(1) + o_m(1))}{2m} \left( \int_{-\infty}^{\infty} e^{-\frac{1}{q} q^{-a-1/2+1}} \left(1 + \varphi\left(\frac{1}{q^{\ell+1}}\right)\right) \prod_{j=1}^{\infty} \left(1 + \varphi\left(\frac{1}{q^{\ell+j(a-1/2)}}\right)\right)^2 \frac{1}{q} \frac{q - 1}{q} dt \right)^{-1}.$$
The memory (in bits) used by the sketch is a random variable, having expectation
\[
(1 + o_{\lambda/m}(1) + o_m(1))m \left( \log_2(2a) + 1 \right)
+ \int_{-\infty}^{\infty} e^{-\frac{\lambda}{q} t^{a-1/2}} \prod_{j=1}^{\infty} \left( 1 + \phi\left( \frac{\lambda}{qt^{(a-1)/2}} \right) \right)^2 \left( e^{-\frac{\lambda}{q} t} \varphi\left( \frac{\lambda}{qt} \right) - \left( e^{-\frac{\lambda}{q} t} - e^{-\frac{\lambda}{q}} \right) \varphi\left( \frac{\lambda}{qt+1} \right) \right) dt.
\]

Proof. Follows from Theorem 4, 8 and 9.

The best parameterization of Theorem 2 uses $q = 2.2$ and $a = 2$, with a limiting MVP $\approx 2.06$.

2.3 Implementation Considerations

To save space we store a curtain $(\hat{g}_0, \hat{g}_1, \ldots)$ as an offset vector $(\hat{g}_0, o_1, o_2, \ldots, o_{m-1})$, $o_i = \hat{g}_i - \hat{g}_{i-1}$, where $\hat{g}_0$ takes $\log_2 \log_q U$ bits and $o_i$ takes $\log_2 |0|$ bits. Both Curtain and SecondCurtain have $\log_2 |0| = \log_2(2a) = 2$, so we fix this constant below, and fix $\log_2 \log_q U = 6$.

Clearly, to evaluate $\hat{g}_i$, we need to compute the prefix sum $\hat{g}_0 + \sum_{i' \leq i} o_{i'}$.

Lemma 1. Given a $w$-bit machine word of $t$-bit integers $(x_0, \ldots, x_{w/t-1})$, the prefix sum $x_0 + \cdots + x_i$ can be evaluated in $O(\log(\log(w/t)))$ time.

Proof. Mask $(x_0, \ldots, x_i)$, then iteratively halve the number of remaining integers by masking those at odd and even positions, shifting, and adding.

Lemma 1 implies that $\hat{g}_i$ can be evaluated in $O(m(\log w)/w)$ time, which is still linear in $m$. We can reduce this to $(C/w)(\log w)$ time by encoding $\hat{g}_0, \hat{g}_{C-1}, \hat{g}_{2C-1}, \ldots$ directly, with $\log_2 \log_q U$ bits each. A more sophisticated implementation can reduce this to $\log(C/w) \log w$ as follows. Curtain can be regarded as arranging the $m$ sketches in a rooted tree, where $\hat{g}_0$ is at the root, and the difference between any node’s curtain value and its parent must be in $\emptyset$. The evaluation time of $\hat{g}_i$ depends on the distance from $\hat{g}_i$ to the root, namely $i$. However, using a tree topology closer to a binary tree, the evaluation time is faster. See Figure 6 for one example, where a curtain on $C = 449$ columns can be encoded with $6 + 2(C - 1)$ bits ($6 = \log \log U$ for $\hat{g}_0$) such that $\hat{g}_i$ can be evaluated by doing prefix-sums on at most 3 64-bit machine words. A sketch of $m$ columns could then be partitioned into $m/C$ such trees. The overhead for storing $\hat{g}_0$ explicitly is less than 1\% of the overall space.\(^\text{12}\)

\(^{12}\)Strictly speaking Theorem 1 does not apply to this tree topology. However, the effect on the variance should be negligible.
SecondCurtain is more difficult to implement efficiently. The length of the encoding for $\delta$ varies over time, so it is necessary to allocate enough space to meet a likely high-water mark. The analysis of Fishmonger (also a variable length code) indicates that $O(\sqrt{m \log m} + \log^2 \log U)$ extra bits suffices [31, §5]. Although both SecondCurtain and Fishmonger require linear time to update the sketch, SecondCurtain is more streamlined and conceptually trivial in comparison. Incrementing or decrementing any $\delta_i$ simply involves inserting a 0 or splicing out a 0 from the encoding of $\delta$.

The focus of this paper is on the theoretical analysis of Martingale sketches, Martingale Curtain and SecondCurtain in particular. We leave an exhaustive experimental evaluation of these sketches for future work.

3 Foundations of the Martingale Transform

In this section we present a simple framework for analyzing the limiting variance of Martingale sketches, which is inspired by Ting’s [34] perspective. Theorem 3 gives simple unbiased estimators for the cardinality and the variance of the cardinality estimator. The upshot of Theorem 3 is that to analyze the variance of the estimator, we only need to bound $E(\frac{1}{P_k})$, where $P_k$ is the probability the $k$th distinct element changes the sketch. Theorem 4 further shows that for sketches composed of $m$ subsketches (like Curtain, HyperLogLog, and PCSA), the limiting variance tends to $\frac{1}{2\kappa m}$, where $\kappa$ is a constant that depends on the sketch scheme. Section 4 analyzes the constant $\kappa$ for each of PCSA, LogLog, Curtain, and SecondCurtain.

3.1 Martingale Estimators and Retrospective Variance

Consider an arbitrary sketch with state space $S$. We assume the sketch state does not change upon seeing duplicated elements, hence it suffices to consider streams of distinct elements. We model the evolution of the sketch as a Markov chain $(S_k)_{k \geq 0} \in S^*$, where $S_k$ is the state after seeing $k$ distinct elements. Define $P_k = Pr(S_k \neq S_{k-1} \mid S_{k-1})$ to be the state changing probability, which depends only on $S_{k-1}$.

**Definition 1.** Let $[\mathcal{E}]$ be the indicator variable for event $\mathcal{E}$. For any $\lambda \geq 0$, define:

$$E_\lambda = \sum_{k=1}^{\lambda} [S_k \neq S_{k-1}] \cdot \frac{1}{P_k},$$

the martingale estimator,

and $V_\lambda = \sum_{k=1}^{\lambda} [S_k \neq S_{k-1}] \cdot \frac{1 - P_k}{P_k^2}$, the “retrospective” variance.

Note that $E_0 = V_0 = 0$.

The Martingale transform of this sketch stores $\hat{\lambda} = E_\lambda$ in one machine word and returns it as a cardinality estimate. It can also store $V_\lambda$ in one machine word as well. Theorem 3 shows that the retrospective variance $V_\lambda$ is a good running estimate of the empirical squared error $(E_\lambda - \lambda)^2$.

**Theorem 3.** The martingale estimator $E_\lambda$ is an unbiased estimator of $\lambda$ and the retrospective variance $V_\lambda$ is an unbiased estimator of $\text{Var}(E_\lambda)$. Specifically, we have,

$$E(E_\lambda) = \lambda, \text{ and } \text{Var}(E_\lambda) = E(V_\lambda) = \sum_{k=1}^{\lambda} E\left(\frac{1}{P_k}\right) - \lambda.$$
Remark 1. Theorem 3 contradicts Ting’s claim [34], that $\Lambda$ is unbiased only at “jump” times, i.e., those $\lambda$ for which $S_\lambda \neq S_{\lambda-1}$, and therefore inadequate to estimate the variance. In order to correct for this, Ting introduced a Bayesian method for estimating the time that has passed since the last jump time. The reason for thinking that jump times are different is actually quite natural. Suppose we record the list of distinct states $s_0, \ldots, s_k$ encountered while inserting $\lambda$ elements, $\lambda$ being unknown, and let $p_i$ be the probability of changing from $s_i$ to some other state. The amount of time spent in state $s_i$ is a geometric random variable with mean $p_i^{-1}$ and variance $(1 - p_i)/p_i^2$. Furthermore, these waiting times are independent. Thus, $\sum_{i \in \{0, k\}} p_i^{-1}$ and $\sum_{i \in \{0, k\}} (1 - p_i^{-1})/p_i^2$ are unbiased estimates of the cardinality $\lambda'$ and squared error upon entering state $s_k$. These exactly correspond to $E_{\lambda}$ and $\Lambda$, but they should be biased since they do not take into account the $\lambda - \lambda'$ elements that had no effect on $s_k$. As Theorem 3 shows, this is a mathematical optical illusion. The history is a random variable, and although the last $\lambda - \lambda'$ elements did not change the state, they could have, which would have altered the history and hence the estimates $E_{\lambda}$ and $\Lambda$.

Proof. Note that $P_k$ is a function of $S_{k-1}$. By the linearity of expectation and the law of total expectation, we have

$$E(E_k) = E(E(E_k \mid S_{k-1})) = E\left( E(E_{k-1} \mid S_{k-1}) + E\left( \left[ S_k \neq S_{k-1} \right] \cdot \frac{1}{P_k} \mid S_{k-1} \right) \right)$$

$$= E(E_{k-1}) + 1 = E(E_{k-2}) + 2 = \ldots = E(E_0) + k = k.$$

and

$$E(V_k) = E(E(V_k \mid S_{k-1})) = E\left( E(V_{k-1} \mid S_{k-1}) + E\left( \left[ S_k \neq S_{k-1} \right] \cdot \frac{1 - P_k}{P_k^2} \mid S_{k-1} \right) \right)$$

$$= E(V_{k-1}) + E\left( \frac{1 - P_k}{P_k} \right) = E(V_{k-2}) + E\left( \frac{1 - P_k}{P_k} \right) + E\left( \frac{1 - P_{k-1}}{P_{k-1}} \right) = \ldots$$

$$= E(V_0) + \sum_{i=1}^{k} E\left( \frac{1 - P_i}{P_i} \right) = \sum_{i=1}^{k} E\left( \frac{1}{P_i} \right) - k.$$

For the variance, we have

$$\text{Var}(E_{\lambda}) = E(E^2_{\lambda}) - (E(E_{\lambda}))^2 = E(E^2_{\lambda}) - \lambda^2.$$

Note that

$$E(E^2_k \mid S_{k-1}) = E\left( \left( E_{k-1} + \left[ S_k \neq S_{k-1} \right] \cdot \frac{1}{P_k} \right)^2 \mid S_{k-1} \right)$$

$$= E^2_{k-1} + 2E_{k-1} \cdot E\left( \left[ S_k \neq S_{k-1} \right] \mid S_{k-1} \right) + \frac{1}{P_k^2} \cdot E\left( \left[ S_k \neq S_{k-1} \right]^2 \mid S_{k-1} \right)$$

$$= E^2_{k-1} + 2E_{k-1} + \frac{1}{P_k}.$$

Then by the law of total expectation and the linearity of expectation, we have

$$E\left( E^2_{k} \right) = E\left( E\left( E^2_{k} \mid S_{k-1} \right) \right) = E\left( E^2_{k-1} + 2E_{k-1} + \frac{1}{P_k} \right) = E\left( E^2_{k-1} \right) + 2(k - 1) + E\left( \frac{1}{P_k} \right).$$

From this recurrence relation, we have

$$E\left( E^2_{\lambda} \right) = E\left( E^2_0 \right) + 2 \sum_{k=1}^{\lambda} (k - 1) + \sum_{k=1}^{\lambda} E\left( \frac{1}{P_k} \right) = \sum_{k=1}^{\lambda} E\left( \frac{1}{P_k} \right) + \lambda(\lambda - 1).$$
We conclude that
\[
\text{Var}(E_\lambda) = \sum_{k=1}^{\lambda} \mathbb{E}\left(\frac{1}{F_k}\right) + \lambda(\lambda - 1) - \lambda^2 = \sum_{k=1}^{\lambda} \mathbb{E}\left(\frac{1}{F_k}\right) - \lambda = \mathbb{E}(V_\lambda).
\]

3.2 Asymptotic Relative Variance

3.2.1 The ARV Factor

We consider classes of sketches composed of \(m\) subsketches, which controls the size and variance. In LogLog, PCSA, and Curtain these subsketches are the \(m\) columns. When considering a sketch with \(m\) subsketches, instead of using \(\lambda\) as the total number of insertions, we always use \(\lambda\) to denote the number of insertions \textit{per subsketch} and therefore the total number of insertions is \(\lambda m\). We care about the \textit{asymptotic relative variance} (ARV) as \(m\) and \(\lambda\) both go to infinity (defined below). A reasonable sketch should have variance \(O(1/m)\). Informally, the ARV factor is just the leading constant of this expression.

\textbf{Definition 2} (ARV factor). Consider a class of sketches whose size is parameterized by \(m\). For any \(k \geq 0\), define \(P_{m,k}\) to be the probability the sketch changes state upon the \(k\)th insertion and \(E_{m,k}\) the martingale estimator. The \textit{ARV factor} of this class of sketches is defined as
\[
\lim_{\lambda \to \infty} \lim_{m \to \infty} m \cdot \frac{\text{Var}(E_{m,\lambda m})}{(\lambda m)^2}.
\]

3.2.2 Scale-Invariance and the Constant \(\kappa\)

Few sketches have \textit{strictly} well-defined ARV factors. In Martingale LogLog, for example, the quantity \(\left(\lim_{m \to \infty} m \cdot \frac{\text{Var}(E_{m,\lambda m})}{(\lambda m)^2}\right)\) is not constant, but periodic in \(\log_2 \lambda\); it does not converge as \(\lambda \to \infty\). We explain how to fix this issue using smoothing in Section 3.2.3. Scale-invariant sketches must have well-defined ARV factors.

\textbf{Definition 3} (scale-invariance and constant \(\kappa\)). A combined sketch is \textit{scale-invariant} if

1. For any \(\lambda\), there exists a constant \(\kappa_\lambda\) such that \(\lambda \cdot P_{m,\lambda m}\) converges to \(\kappa_\lambda\) almost surely as \(m \to \infty\).

2. The limit of \(\kappa_\lambda\) as \(\lambda \to \infty\) exists, and \(\kappa \overset{\text{def}}{=} \lim_{\lambda \to \infty} \kappa_\lambda\).

The constant of a sketch \(A\) is denoted as \(\kappa_A\), where the subscript \(A\) is often dropped when the context is clear.

The next theorem proves that under mild regularity conditions, all scale-invariant sketches have well defined ARV factors and there is a direct relation between the ARV factor and the constant \(\kappa\). (See Appendix A for proof.)

\textbf{Theorem 4} (ARV factor of a scale-invariant sketch). Consider a sketching scheme satisfying the following properties.

1. It is scale-invariant with constant \(\kappa\).
2. For any $\lambda > 0$, the limit operator and the expectation operator of $\{\frac{1}{P_{m,\lambda m}}\}_m$ can be interchanged.

Then the ARV factor of the sketch exists and equals $\frac{1}{2\kappa}$.

The constant $\kappa$ together with Theorem 4 is useful in that it gives a simple and systematic way to evaluate the asymptotic performance of a well behaved (scale-invariant) sketch scheme.

MinCount [9,24,29] is an example of a scale-invariant sketch. The function $h(a) = (i,v) \in [m] \times [0,1]$ is interpreted as a pair containing a bucket index and a real hash value. A $(k,m)$-MinCount sketch stores the smallest $k$ hash values in each bucket.

**Theorem 5.** $(k,m)$-MinCount is scale-invariant and $\kappa_{(k,m)} = k$.

**Proof.** When a total of $\lambda m$ elements are inserted to the combined sketch, each subsketch receives $(1+o(1))\lambda$ elements as $\lambda \to \infty$. Since we only care the asymptotic behavior, we assume for simplicity that each subsketch receives exactly $\lambda$ elements.

Let $P^{(i)}_\lambda$ be the probability that the sketch of the $i$th bucket changes after the $\lambda$th element is thrown into the $i$th bucket. Then by definition, we have

$$P_{m,\lambda m} = \sum_{i=1}^{m} P^{(i)}_\lambda.$$  

Since all the subsketches are i.i.d., by the law of large numbers, $\lambda \cdot P_{m,\lambda} \to \lambda \cdot \mathbb{E}(P^{(1)}_\lambda)$ almost surely as $m \to \infty$.

Let $X$ be the $k$th smallest hash value among $\lambda$ uniformly random numbers in $[0,1]$, which distributes identically with $P^{(1)}_\lambda$. By standard order statistics, $X$ is a Beta random variable $\text{Beta}(k, \lambda - 1 + k)$ which has mean $\frac{k\lambda}{\lambda+1}$. Thus $\kappa_\lambda = \lambda \cdot \mathbb{E}(X) = \frac{k\lambda}{\lambda+1}$. We conclude that

$$\kappa = \lim_{\lambda \to \infty} \kappa_\lambda = \lim_{\lambda \to \infty} \frac{k\lambda}{\lambda + 1} = k.$$

Applying Theorem 4 to $(k,m)$-MinCount, we see its ARV is $\frac{1}{2km}$ matching Cohen [15] and Ting [34]. Technically its MVP is unbounded since hash values were real numbers, but any realistic implementation would store them as $\log U$-bit integers, for a total of $km \log U$ bits. Hence we regard its MVP to be $\frac{1}{2} \cdot \log_2 U$.

### 3.2.3 Smoothing Discrete Sketches

Sketches that partition the dartboard in some exponential fashion with base $q$ (like LogLog, PCSA, and Curtain) have the property that their estimates and variance are periodic in $\log q \lambda$. Pettie and Wang [31] proposed a simple method to smooth these sketches and make them truly scale-invariant as $m \to \infty$.

We assume that the dartboard is partitioned into $m$ columns. The base-$q$ smoothing operation uses an offset vector $\vec{r} = (r_0, \ldots, r_{m-1})$. We scale down all the cells in column $i$ by the factor $q^{-r_i}$, then add a dummy cell spanning $[q^{-r_i}, 1)$ which is always occupied. (Phrased algorithmically, if a dart is destined for column $i$, we filter it out with probability $1 - q^{-r_i}$ and insert it into the sketch with probability $q^{-r_i}$.) When analyzing variants of (Hyper)LogLog and PCSA, we use the uniform

---

13For simplicity, we assume the second condition of Theorem 4 holds for all the sketches analyzed in this paper.
offset vector \((0, 1/m, 2/m, \ldots, (m - 1)/m)\). The Curtain sketch can be viewed as having a built-in offset vector of \((0, 1/2, 0, 1/2, 0, 1/2, \ldots)\) which effects the “sawtooth” cell partition. To smooth it, we use the offset vector\(^{14}\)

\[
(0, 1/2, 1/m, 1/2 + 1/m, 2/m, 1/2 + 2/m, \ldots, 1/2 - 1/m, 1 - 1/m).
\]

As \(m \to \infty\), \(\vec{r}\) becomes uniformly dense in \([0, 1]\).

The smoothing technique makes the empirical estimation more scale-invariant (see [31, Figs. 1&2]) but also makes the sketch theoretically scale-invariant according to Definition 3. Thus, in the analysis, we will always assume the sketches are smoothed. However, in practice it is probably not necessary to do smoothing if \(q < 3\).

In the next section, we will prove that smoothed \(q\)-LL, \(q\)-PCSA, Curtain, and SecondCurtain are all scale-invariant.

4 Analysis of Dartboard Based Sketches

Consider a dartboard cell that covers the vertical interval \([q^{-t-1}, q^{-t})\). We define the height of the cell to be \(t\). In a smoothed cell partition, no two cells have the same height and all heights are of the form \(t = j/m\), for some integer \(j\). Thus, we may refer to it unambiguously as cell \(t\). Note that cell \(t\) is an \(m^{-1} \times \frac{1}{q^t-1} q\) rectangle.

4.1 Poissonized Dartboard

Since we care about the asymptotic case where \(\lambda \to \infty\), we model the process of “throwing darts” by a Poisson point process on the dart board (similar to the “poissonization” in the analysis of HyperLogLog [20]). Specifically, after throwing \(\lambda m\) darts (events) to the dartboard, we assume the number of darts in cell \(t\) is a Poisson random variable with mean \(\lambda \frac{1}{q^t-1} q\) and the number of darts in different cells are independent. For the poissonized dartboard, the range of height of cells naturally extend to the whole set of real numbers, instead of just having cells with positive height.

For any \(t \in \mathbb{R}\), let \(Y_{t,\lambda}\) be the indicator whether cell \(t\) contains at least one dart. Note that the probability that a Poisson random variable with mean \(\lambda'\) is zero is \(e^{-\lambda'}\). Thus we have,

\[
\Pr(Y_{t,\lambda} = 0) = e^{-\frac{\lambda}{q^t-1} q}.
\]

Here, we note some simple identities for integrals that we will use frequently in the analysis.

**Lemma 2.** For any \(q > 1\), we have

\[
\int \frac{1}{q^t} e^{-\frac{1}{q^t}} dt = \frac{1}{\ln q} e^{-\frac{1}{q^t}} + C.
\]

Furthermore, let \(c_0, c_1\) be any positive numbers, we have

\[
\int_{-\infty}^{\infty} \frac{c_0}{q^t} e^{-\frac{c_1}{q^t}} dt = \frac{c_0}{c_1 \ln q}.
\]

**Proof.** Use standard calculus. \(\square\)

\(^{14}\)In [31], the smoothing was implemented via random offsetting, instead of the uniform offsetting. In Curtain and SecondCurtain we need use uniform offsetting so that the offset values of columns are similar to their neighbors.
4.2 The Constant $\kappa$

Let $Z_{t,\lambda}$ be the indicator of whether the cell $t$ is free. Unlike $Y_{t,\lambda}$, $Z_{t,\lambda}$ depends on which sketching algorithm we are analyzing. Since the state changing probability is equal to the sum of the area of free cells, we have

$$P_{m,\lambda m} = \sum_{j=0}^{\infty} \frac{1}{m} \left( \frac{1}{q^j/m} - \frac{1}{q^{j+1/m+1}} \right) Z_{j/m,\lambda}. \quad (2)$$

If $P_{m,\lambda m}$ converges to $\kappa_\lambda/\lambda$ almost surely as $m \to \infty$, then $\mathbb{E}(P_{m,\lambda m})$ also converges to $\kappa_\lambda/\lambda$ as $m \to \infty$. Thus we have, from (2),

$$\kappa_\lambda/\lambda = \lim_{m \to \infty} \mathbb{E}(P_{m,\lambda m}) = \lim_{m \to \infty} \sum_{j=0}^{\infty} \frac{1}{m} \left( \frac{1}{q^j/m} - \frac{1}{q^{j+1/m+1}} \right) \mathbb{E}(Z_{j/m,\lambda})$$

$$= \int_{0}^{\infty} \left( \frac{1}{q^t} - \frac{1}{q^{t+1}} \right) \mathbb{E}(Z_{t,\lambda})dt \approx \int_{-\infty}^{\infty} \left( \frac{1}{q^t} - \frac{1}{q^{t+1}} \right) \mathbb{E}(Z_{t,\lambda})dt, \quad (3)$$

where we can extend the integration range to negative infinity without affecting the limit of $\kappa_\lambda$ as $\lambda \to \infty$.\(^\text{15}\) We conclude that

$$\kappa = \lim_{\lambda \to \infty} \kappa_\lambda = \lim_{\lambda \to \infty} \lambda \int_{-\infty}^{\infty} \left( \frac{1}{q^t} - \frac{1}{q^{t+1}} \right) \mathbb{E}(Z_{t,\lambda})dt. \quad (4)$$

The formula (4) is novel in the sense that, in order to evaluate $\kappa$, we now only need to understand the probability that $Z_{t,\lambda}$ is 1 for fixed $t$ and $\lambda$.\(^\text{16}\)

4.3 Analysis of Smoothed $q$-PCSA and $q$-LL

The sketches $q$-PCSA and $q$-LL are the natural smoothed base-$q$ generalizations of PCSA [21] and LogLog [16].

**Theorem 6.** $q$-PCSA and $q$-LL are scale-invariant. In particular, we have,

$$\kappa_{q,\text{PCSA}} = \frac{1}{\ln q}, \text{ and } \kappa_{q,\text{LL}} = \frac{1}{\ln q} \frac{q - 1}{q}.$$

**Proof.** For $q$-LL, cell $t$ is free iff both itself and all the cells above it in its column contain no darts. Thus we have

$$\mathbb{E}(Z_{t,\lambda}) = \prod_{i=0}^{\infty} \Pr(Y_{t+i,\lambda} = 0) = \prod_{i=0}^{\infty} e^{-\frac{\lambda}{q^{i+1}q}} = e^{-\frac{\lambda}{q^t}}.$$

Insert it to formula (4) and we get

$$\kappa_{q,\text{LL}} = \lim_{\lambda \to \infty} \lambda \int_{-\infty}^{\infty} \left( \frac{1}{q^t} - \frac{1}{q^{t+1}} \right) e^{-\frac{\lambda}{q^t}} dt = \frac{1}{\ln q} \frac{q - 1}{q}.$$

\(^{15}\)Note that for any $t, \lambda$, we all have $\mathbb{E}(Z_{t,\lambda}) \leq \mathbb{E}(1 - Y_{t,\lambda})$ (free cell has no dart). Therefore, by extending the integration (3) to the whole real line, the increment is bounded by $\int_{-\infty}^{0} \left( \frac{1}{q^t} - \frac{1}{q^{t+1}} \right) \mathbb{E}(1 - Y_{t,\lambda})dt = \int_{-\infty}^{\infty} \left( \frac{1}{q^t} - \frac{1}{q^{t+1}} \right) e^{-\lambda(1/q^{t+1} - 1/q^{t+1})}dt = \frac{1}{\lambda} \int_{-\infty}^{0} e^{-\frac{\lambda t}{q^t}} \log_q(\lambda(q-1)/q) 1/q^t e^{-1/q^t} dt = \frac{1}{\lambda} \frac{\lambda q + 1}{q^t} \ln q = \frac{\lambda q + 1}{q^t \ln q} \to 0$ as $\lambda \to \infty$.

\(^{16}\)Technically, to apply formula (4) one needs to first prove that the state changing probability $P_{m,\lambda m}$ converges almost surely to some constant $\kappa_\lambda/\lambda$ for any $\lambda$, which is a mild regularity condition for any reasonable sketch. Thus in this paper we will assume the sketches in the analysis all satisfy this regularity condition and claim that a sketch is scale-invariant if formula (4) converges.

16
For \( q\)-PCSA, cell \( t \) is free iff it has no dart. Thus \( Z_{t,\lambda} = 1 - Y_{t,\lambda} \) and by formula (4) we have

\[
\kappa_{q\text{-}PCSA} = \lim_{\lambda \to \infty} \lambda \int_{-\infty}^{\infty} \left( \frac{1}{q^t} - \frac{1}{q^{t+1}} \right) e^{-\frac{\lambda}{q^{t+1}} q^{-\frac{1}{q}}} dt = \frac{1}{\ln q}.
\]

The Fishmonger [31] sketch is based on a smoothed, entropy compressed version of base-2 PCSA. The memory footprint of Fishmonger approaches its entropy as \( m \to \infty \), which was calculated to be \( mH_0/\ln 2 \) [31, Lemma 4]. From Theorem 6, we know \( \kappa_{2\text{-}PCSA} = \frac{1}{\ln 2} \).

**Corollary 1.** Fishmonger has limiting MVP \( H_0/2 \approx 1.63 \).

**Proof.** By Theorem 4, limiting MVP equals \( \frac{mH_0}{\ln 2} \cdot \frac{1}{2^{m/2}} = \frac{H_0}{2} \).

### 4.4 Asymptotic Local View

For any \( t \) and \( \lambda \), since we want to evaluate \( Z_{t,\lambda} \), whose value may depend on its “neighbors” on the dartboard, we need to understand the configurations of the cells near cell \( t \). Since we consider the case where \( m \) goes to infinity, we may ignore the effect of smoothing to the cells in the immediate vicinity of cell \( t \).

After taking these asymptotic approximations, we can index the cells near cell \( t \) as follows.

**Definition 4** (neighbors of cell \( t \)). Fix a cell \( t \). Let \( i \in \mathbb{Z} \) and \( c \in \mathbb{R} \). The \((i, c)\)-neighbor of cell \( t \) is a cell whose column index differs by \( i \) (negative \( i \) means to the left, positive to the right) and has height \( t + c \), it covers the vertical interval \([q^{-t+c+1}, q^{-t+c})\). In the sawtooth partition, \( c \) is an integer when \( i \) is even and a half-integer when \( i \) is odd. (Note that we are locally ignoring the effect of smoothing.)

Once cell \( t \) is fixed, define \( W(i, c) \) to be the indicator for whether the \((i, c)\)-neighbor of cell \( t \) has at least one dart in it. Thus, for fixed \( t, \lambda \), we have

\[
\Pr(W(i, c) = 0) = \Pr(Y_{t+c,\lambda} = 0) = e^{-\frac{\lambda}{q^{t+c+1}} q^{-\frac{1}{q}}}.
\]

In the asymptotic local view, we lose the property that a cell can be uniquely identified by its height, hence the need to refer to nearby cells by their position relative to cell \( t \).

### 4.5 Analysis of Curtain

We first briefly state some properties of curtain. For any \( a \geq 1 \), recall that \( \Theta_a = \{-(a-1/2), -(a-3/2), \ldots, -1/2, 1/2, \ldots, a-3/2, a-1/2\} \). It is easy to see that for any vector \( v = (g_0, g_1, \ldots, g_{m-1}) \), \( v_{\text{curt}} = (\hat{g}_i) \) can be expressed as

\[
\hat{g}_i = \max_{j \in [0, m-1]} \{g_j - |i - j|(a-1/2)\}.
\]

For each \( i \), we define the tension point \( \tau_i \) to be the lowest allowable value of \( \hat{g}_i \), given the context of its neighboring columns.

\[
\tau_i = \max_{j \in [0, m-1] \setminus \{i\}} \{g_j - |i - j|(a-1/2)\},
\]

and thus we have \( \hat{g}_i = \max(g_i, \tau_i) \). We see that the column \( i \) is in tension iff \( g_i \leq \tau_i \), that is, \( \hat{g}_i = \tau_i \).
Theorem 7. Curtain is scale-invariant with

\[ \kappa_{\text{Curtain}} = \frac{1}{\ln q} \left( \frac{q-1}{q} \frac{q^{-1}}{q} + \frac{1}{2} \frac{1}{q^h(q^{a-1/2} - 1)} + \frac{1}{q^{h+1}} \right) \]

Proof. Fix cell \( t \) and \( \lambda \). Define \( W_1(k) \) to be the height of the highest cell containing darts in the column \( k \) away from \( t \)'s column. I.e., define \( t = \lceil k \text{ is odd} \rceil / 2 \) to be \( 1/2 \) if \( k \) is odd and zero if \( k \) is even, and \( W_1(k) \overset{\text{def}}{=} \max\{t + i + \varepsilon \mid i \in \mathbb{Z} \text{ and } W(k, i + \varepsilon) = 1\} \).

We have for any \( i \in \mathbb{Z} \),

\[ \Pr(W_1(k) \leq t + i + \varepsilon) = \prod_{j=1}^{\infty} \Pr(W(k, i + j + \varepsilon) = 0) = e^{-\frac{\lambda}{q^{t+1} + 1(q^{a/2} - 1)}}. \]

Let \( T_1 \) be the tension point of the column of cell \( t \), which equals \( \max_{j \in \mathbb{Z} \setminus \{0\}} \{W_1(j) - |j|(a-1/2)\} \). We have for any \( i \in \mathbb{Z} \),

\[ \Pr(T_1 \leq t + i) = \Pr\left( \max_{j \in \mathbb{Z} \setminus \{0\}} \{W_1(j) - |j|(a-1/2)\} \leq i + t \right) \]

\[ = \prod_{j \in \mathbb{Z} \setminus \{0\}} \Pr(W_1(j) - |j|(a-1/2) \leq i + t) \]

\[ = \left( \prod_{j=1}^{\infty} e^{-\frac{\lambda}{q^{t+1} + 1(q^{a/2} - 1)}} \right)^2 = e^{-\frac{\lambda}{q^{t+1} + 1(q^{a/2} - 1)}}. \]

From the rules of Curtain, we know that a cell is free iff it contains no dart, it is at most \( h - 1 \) below its column’s tension point, and at most \( h \) below the highest cell in its column containing darts. Thus,

\[ Z_{t,\lambda} = \left[ Y_{t,\lambda} = 0 \right] \cdot \left[ \begin{array}{l} t \geq T_1 - (h - 1) \end{array} \right] \cdot \left[ \begin{array}{l} t \geq W_1(0) - h \end{array} \right] , \]

Note that \( T_1 \) is independent from \( Y_{t,\lambda} \) and \( W_1(0) \). In addition, \( Y_{t,\lambda} \) is also independent from \( \left[ t \geq W_1(0) - h \right] \), since the latter only depends on \( Y_{t',\lambda} \) with \( t' \geq h + t + 1 \). Thus, we have

\[ E(Z_{t,\lambda}) = \Pr(Y_{t,\lambda} = 0) \cdot \Pr(T_1 \leq t + h - 1) \cdot \Pr(W_1(0) \leq t + h) \]

\[ = e^{-\frac{\lambda}{q^{t+1} + 1}} e^{-\frac{\lambda}{q^{t+1} + 1} q^{a/2} - 1} e^{-\frac{\lambda}{q^{t+1} + 1} q^{a/2} - 1} \]

\[ = \exp\left( -\frac{\lambda}{q^t} \left( \frac{q-1}{q} + \frac{2}{q^h(q^{a-1/2} - 1)} + \frac{1}{q^{h+1}} \right) \right) . \]

Thus by formula (4), we have

\[ \kappa_{\text{Curtain}} = \lim_{\lambda \to \infty} \frac{1}{\ln q} \int_{-\infty}^{\infty} \left( \frac{1}{q^t} - \frac{1}{q^{t+1}} \right) \exp\left( -\frac{\lambda}{q^t} \left( \frac{q-1}{q} + \frac{2}{q^h(q^{a-1/2} - 1)} + \frac{1}{q^{h+1}} \right) \right) dt \]

\[ = \frac{1}{\ln q} \left( \frac{q-1}{q} \frac{q^{-1}}{q} + \frac{1}{2} \frac{1}{q^h(q^{a-1/2} - 1)} + \frac{1}{q^{h+1}} \right) . \]

Refer to Appendix B for the analysis of SecondCurtain.
5 Experimental Validation

Throughout the paper we have maintained a possibly unhealthy devotion to asymptotic analysis, taking $m \to \infty$ whenever it was convenient. In practice $m$ will be a constant, and possibly a smallish constant. How do the sketches perform in the pre-asymptotic region?

In turns out that the theoretical analysis predicts the performance of Martingale sketches pretty well, even when $m$ is small. In the experiment of Figure 7, we fixed the sketch size at a tiny 128 bits. Therefore HyperLogLog uses $m_1 = \lfloor 128/6 \rfloor = 21$ counters. The Martingale LogLog and Martingale Curtain sketches encode the martingale estimator with a floating point approximation of $\hat{\lambda}$ in 14 bits, with a 6-bit exponent and 8-bit mantissa. Thus, Martingale LogLog uses $m_2 = (128 - 14)/6 = 19$ counters, and Martingale Curtain uses $m_3 = 37$.\(^{17}\)

For larger sketch sizes, the distribution of $\hat{\lambda}/\lambda$ is more symmetric, and closer to the predicted performance. Figure 8 gives the empirical distribution of $\hat{\lambda}/\lambda$ over 100,000 runs when $\lambda = 10^6$ and the sketch size is fixed at 1,200 bits. Here Martingale Curtain uses $m = 400$, and both Martingale LogLog and HyperLogLog use $m = 200$.

The experimental and predicted relative variances and standard errors are given in Table 2.

\(^{17}\)It uses the optimal parameterization $(q, a, h) = (2.91, 2, 1)$ of Theorem 1.
6 Conclusion

Cardinality estimation is an unusual problem in that it admits some notion of absolute optimality, which is captured by the memory-variance product (MVP). We have conjectured [31] that Fishmonger is optimal for composable sketches (limiting MVP = $H_0/I_2 \approx 1.98$), and conjecture that Martingale Fishmonger is optimal in the sequential (non-composable) setting, with limiting MVP = $H_0/2 \approx 1.63$.

In the composable setting, the meaty intellectual problems revolve around analyzing the bias and variance of novel estimators [9, 16, 19–21, 24, 29]. However, in the non-composable setting, applying the Cohen/Ting [15, 34] martingale transformation seems to be the only rational way to do cardinality estimation. If there are difficult problems here, they are in sketch design, not statistical estimation. The framework of Theorems 3 and 4 simplifies Cohen [15] and Ting [34], and gives a user-friendly formula for the asymptotic relative variance (ARV) of the martingale estimator, as a function of the sketch’s constant $\kappa$.

In this paper we aimed to design realistically implementable sketches that are both analytically tractable and approach the conjectured information-theoretic lower bounds. The sketch Martingale Curtain has an MVP $\approx 2.31$ and strikes a nice balance between efficiency and complexity. It can be updated with a few memory probes; see Section 2.3. The SecondCurtain sketch is much more efficient (limiting MVP $\approx 2.06$) but less practical as it encodes integers with variable length codes. Indeed, avoiding variable-length encodings seems to be essential for having worst case $O(1)$ update time.

We think it likely that there are yet-simpler and more efficient sketches still to be discovered. Is it possible to get close to MVP $\approx 2$ with a simple sketch in the sequential setting?

References

[1] N. Alon, P. B. Gibbons, Y. Matias, and M. Szegedy. Tracking join and self-join sizes in limited storage. In Proceedings 18th ACM Symposium on Principles of Database Systems (PODS), pages 10–20, 1999.

[2] D. N. Baker and B. Langmead. Dashing: Fast and accurate genomic distances with hyperloglog. bioRxiv, 2019.

[3] Z. Bar-Yossef, T. S. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan. Counting distinct elements in a data stream. In Proceedings 6th International Workshop on Randomization and Approximation Techniques (RANDOM), volume 2483 of Lecture Notes in Computer Science, pages 1–10, 2002.

[4] Z. Bar-Yossef, R. Kumar, and D. Sivakumar. Reductions in streaming algorithms, with an application to counting triangles in graphs. In Proceedings 13th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pages 623–632, 2002.

[5] R. Ben-Basat, G. Einziger, S. L. Feibish, J. Moraney, and D. Raz. Network-wide routing-oblivious heavy hitters. In Proceedings of the 2018 Symposium on Architectures for Networking and Communications Systems (ANCS), pages 66–73, 2018.

[6] K. S. Beyer, R. Gemulla, P. J. Haas, B. Reinwald, and Y. Sismanis. Distinct-value synopses for multiset operations. Commun. ACM, 52(10):87–95, 2009.
[7] J. Blasiok. Optimal streaming and tracking distinct elements with high probability. *ACM Trans. Algorithms*, 16(1):3:1–3:28, 2020.

[8] T. Buddhika, M. Malensek, S. L. Pallickara, and S. Pallickara. Synopsis: A distributed sketch over voluminous spatiotemporal observational streams. *IEEE Trans. Knowl. Data Eng.*, 29(11):2552–2566, 2017.

[9] P. Chassaing and L. Gerin. Efficient estimation of the cardinality of large data sets. In *Proceedings of the 4th Colloquium on Mathematics and Computer Science Algorithms, Trees, Combinatorics and Probabilities*, 2006.

[10] A. Chen, J. Cao, L. Shepp, and T. Nguyen. Distinct counting with a self-learning bitmap. *Journal of the American Statistical Association*, 106(495):879–890, 2011.

[11] M. Chen, S. Chen, and Z. Cai. Counter tree: A scalable counter architecture for per-flow traffic measurement. *IEEE/ACM Trans. Netw.*, 25(2):1249–1262, 2017.

[12] P. H. Chia, D. Desfontaines, I. M. Perera, D. Simmons-Marengo, C. Li, W. Day, Q. Wang, and M. Guevara. KHyperLogLog: Estimating reidentifiability and joinability of large data at scale. In *Proceedings of the 2019 IEEE Symposium on Security and Privacy*, pages 350–364, 2019.

[13] P. Clifford and I. A. Cosma. A statistical analysis of probabilistic counting algorithms. *Scandinavian Journal of Statistics*, 39(1):1–14, 2012.

[14] E. Cohen. Size-estimation framework with applications to transitive closure and reachability. *J. Comput. Syst. Sci.*, 55(3):441–453, 1997.

[15] E. Cohen. All-distances sketches, revisited: HIP estimators for massive graphs analysis. *IEEE Trans. Knowl. Data Eng.*, 27(9):2320–2334, 2015.

[16] M. Durand and P. Flajolet. Loglog counting of large cardinalities. In *Proceedings 11th Annual European Symposium on Algorithms (ESA)*, volume 2832 of *Lecture Notes in Computer Science*, pages 605–617. Springer, 2003.

[17] R. A. L. Elworth, Q. Wang, P. K. Kota, C. J. Barberan, B. Coleman, A. Balaji, G. Gupta, R. G. Baraniuk, A. Shrivastava, and T. J. Treangen. To petabytes and beyond: recent advances in probabilistic and signal processing algorithms and their application to metagenomics. *Nucleic Acids Research*, 48(10):5217–5234, 2020.

[18] C. Estan, G. Varghese, and M. E. Fisk. Bitmap algorithms for counting active flows on high-speed links. *IEEE/ACM Trans. Netw.*, 14(5):925–937, 2006.

[19] P. Flajolet. On adaptive sampling. *Computing*, 43(4):391–400, 1990.

[20] P. Flajolet, É. Fusy, O. Gandouet, and F. Meunier. HyperLogLog: the analysis of a near-optimal cardinality estimation algorithm. In *Proceedings of the 18th International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods for the Analysis of Algorithms (AofA)*, 2007.

[21] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31(2):182–209, 1985.
[22] M. J. Freitag and T. Neumann. Every row counts: Combining sketches and sampling for accurate group-by result estimates. In Proceedings of the 9th Biennial Conference on Innovative Data Systems Research (CIDR), 2019.

[23] P. B. Gibbons and S. Tirthapura. Estimating simple functions on the union of data streams. In Proceedings 13th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA), pages 281–291, 2001.

[24] F. Giroire. Order statistics and estimating cardinalities of massive data sets. Discret. Appl. Math., 157(2):406–427, 2009.

[25] A. Helmi, J. Lumbroso, C. Martínez, and A. Viola. Data Streams as Random Permutations: the Distinct Element Problem. In Proceedings of the 23rd International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods for the Analysis of Algorithms (AofA), pages 323–338, 2012.

[26] P. Indyk and D. P. Woodruff. Tight lower bounds for the distinct elements problem. In Proceedings 44th IEEE Symposium on Foundations of Computer Science (FOCS), October 2003, Cambridge, MA, USA, Proceedings, pages 283–288, 2003.

[27] T. S. Jayram and D. P. Woodruff. Optimal bounds for Johnson-Lindenstrauss transforms and streaming problems with subconstant error. ACM Trans. Algorithms, 9(3):26:1–26:17, 2013.

[28] D. M. Kane, J. Nelson, and D. P. Woodruff. An optimal algorithm for the distinct elements problem. In Proceedings 29th ACM Symposium on Principles of Database Systems (PODS), pages 41–52, 2010.

[29] J. Lumbroso. An optimal cardinality estimation algorithm based on order statistics and its full analysis. In Proceedings of the 21st International Meeting on Probabilistic, Combinatorial, and Asymptotic Methods in the Analysis of Algorithms (AofA), pages 489–504, 2010.

[30] G. Marçais, B. Solomon, R. Patro, and C. Kingsford. Sketching and sublinear data structures in genomics. Annual Review of Biomedical Data Science, 2(1):93–118, 2019.

[31] S. Pettie and D. Wang. Information theoretic limits of cardinality estimation: Fisher meets Shannon. CoRR, abs/2007.08051, 2020.

[32] N. Pham. Hybrid LSH: faster near neighbors reporting in high-dimensional space. In Proceedings of the 20th International Conference on Extending Database Technology (EDBT), pages 454–457, 2017.

[33] R. Sedgewick. Cardinality estimation. Presentation delivered at AofA (2016), Knuth-80 (2018), and Dagstuhl 19051 (2019).

[34] D. Ting. Streamed approximate counting of distinct elements: beating optimal batch methods. In Proceedings 20th ACM Conference on Knowledge Discovery and Data Mining (KDD), pages 442–451, 2014.

[35] J. Wires, S. Ingram, Z. Drudi, N. J. A. Harvey, and A. Warfield. Characterizing storage workloads with counter stacks. In Proceedings of the 11th USENIX Symposium on Operating Systems Design and Implementation (OSDI), pages 335–349, 2014.
[36] I. H. Witten, R. M. Neal, and J. G. Cleary. Arithmetic coding for data compression. Commun. ACM, 30(6):520–540, 1987.

[37] D. E. Wood, J. Lu, and B. Langmead. Improved metagenomic analysis with Kraken 2. bioRxiv, 2019.

[38] Q. Xiao, S. Chen, Y. Zhou, M. Chen, J. Luo, T. Li, and Y. Ling. Cardinality estimation for elephant flows: A compact solution based on virtual register sharing. IEEE/ACM Trans. Netw., 25(6):3738–3752, 2017.

[39] Q. Xiao, S. Chen, Y. Zhou, and J. Luo. Estimating cardinality for arbitrarily large data stream with improved memory efficiency. IEEE/ACM Trans. Netw., 28(2):433–446, 2020.

A Proof of Theorem 4

Theorem 4. First note that, by the assumptions, we have that

\[
\lim_{m \to \infty} \mathbb{E} \left( \frac{1}{P_{m,\lambda m}} \right) = \mathbb{E} \left( \lim_{m \to \infty} \frac{1}{P_{m,\lambda m}} \right) = \mathbb{E} \left( \frac{\lambda}{\kappa \lambda} \right) = \frac{\lambda}{\kappa \lambda}.
\]

Also note that since \( P_{m,k} \) are non-increasing as \( k \) increases, by simple coupling argument, we see that for any \( k \leq k' \), \( \mathbb{E}(1/P_{m,k}) \leq \mathbb{E}(1/P_{m,k'}) \) and \( \frac{k}{k'} \geq \frac{k'}{k} \).

Fix \( \lambda > 0 \), we have, by Theorem 3,

\[
\lim_{m \to \infty} \frac{1}{\lambda^2 m} \text{Var}(E_{m,\lambda m}) = \lim_{m \to \infty} \frac{1}{\lambda^2 m} \sum_{k=1}^{\lambda m} \mathbb{E} \left( \frac{1}{P_{m,k}} - \frac{1}{\lambda} \right)
\]

\[
= \lim_{m \to \infty} \frac{1}{\lambda^2 m} \sum_{i=0}^{\lambda-1} \sum_{j=1}^{m} \mathbb{E} \left( \frac{1}{P_{m,(i+1)m+j}} - \frac{1}{\lambda} \right)
\]

(5)

Since for any \( j \in [1, m] \), \( \mathbb{E} \left( \frac{1}{P_{m,(i+1)m+j}} \right) \leq \mathbb{E} \left( \frac{1}{P_{m,(i+1)m}} \right) \), we have

\[
\lim_{m \to \infty} \frac{1}{\lambda^2 m} \text{Var}(E_{m,\lambda m}) \leq \lim_{m \to \infty} \frac{1}{\lambda^2 m} \sum_{i=0}^{\lambda-1} \sum_{j=1}^{m} \mathbb{E} \left( \frac{1}{P_{m,(i+1)m}} - \frac{1}{\lambda} \right)
\]

\[
= \frac{1}{\lambda^2} \sum_{i=0}^{\lambda-1} \lim_{m \to \infty} \mathbb{E} \left( \frac{1}{P_{m,(i+1)m}} - \frac{1}{\lambda} \right)
\]

\[
= \frac{1}{\lambda^2} \sum_{i=0}^{\lambda-1} \frac{i + 1}{\kappa_{i+1}} - \frac{1}{\lambda}
\]

Denote the ARV factor as \( v \). Fix \( W > 0 \). Note that for any \( i \in [0, \lambda/W - 1] \), \( \frac{\lambda/W + i + 1}{\kappa_{\lambda/W + i + 1}} \leq \frac{(k+1)\lambda/W}{\kappa_{(k+1)\lambda/W}} \)

\[
v \leq \lim_{\lambda \to \infty} \left( \frac{1}{\lambda^2} \sum_{i=0}^{\lambda-1} \frac{i + 1}{\kappa_{i+1}} - \frac{1}{\lambda} \right) = \lim_{\lambda \to \infty} \left( \frac{1}{\lambda^2} \sum_{k=0}^{W-1} \sum_{i=0}^{\lambda/W - 1} \frac{k\lambda/W + i + 1}{\kappa_{k\lambda/W + i + 1}} \right)
\]

\[
\leq \lim_{\lambda \to \infty} \left( \frac{1}{\lambda^2} \sum_{k=0}^{W-1} \sum_{i=0}^{\lambda/W - 1} \frac{(k+1)\lambda/W}{\kappa_{(k+1)\lambda/W}} \right) = \frac{1}{W^2} \sum_{k=0}^{W-1} \lim_{\lambda \to \infty} \frac{k + 1}{\kappa_{(k+1)\lambda/W}}
\]
note that $\lim_{\lambda \to \infty} \kappa^{(k+1)\lambda/W} = \kappa$ by the definition of scale-invariance,

$$
= \frac{1}{W^2} \sum_{k=0}^{W-1} \frac{k+1}{\kappa} = \frac{1}{2\kappa} \frac{W(W+1)}{W^2}.
$$

(6)

On the other hand, we can bound it from below similarly. We will only outline the key steps since it is almost identical to the previous one. Note that for any $j \in [1, m]$, $\mathbb{E}(\frac{1}{P_{m,im+j}}) \geq \mathbb{E}(\frac{1}{P_{m,im}})$. Using this inequality in (5), we have

$$
\lim_{m \to \infty} \frac{1}{\lambda^2m} \text{Var}(E_{m,\lambda m}) \geq \lim_{m \to \infty} \frac{1}{\lambda^2m} \sum_{i=0}^{m-1} \sum_{j=1}^{m} \mathbb{E}\left(\frac{1}{P_{m,im}}\right) - 1 = \frac{1}{\lambda^2} \sum_{i=0}^{m-1} \kappa_i - 1 = \lambda.
$$

Thus by combining (6) and (7), we have

$$
v \geq \lim_{\lambda \to \infty} \left( \frac{\sum_{k=0}^{W-1} \frac{W-1}{\lambda^2m} \kappa \lambda/W + i}{\sum_{i=0}^{\lambda^2m} \kappa \lambda/W} \right) \geq \lim_{\lambda \to \infty} \left( \frac{1}{\lambda^2} \sum_{k=0}^{W-1} \frac{W-1}{\kappa \lambda/W} \right) = \frac{1}{W^2} \sum_{k=0}^{W-1} \frac{W-1}{\kappa} = \frac{1}{2\kappa} \frac{W(W-1)}{W^2}.
$$

(7)

Thus by combining (6) and (7), we have

$$
\frac{1}{2\kappa} \frac{W(W-1)}{W^2} \leq v \leq \frac{1}{2\kappa} \frac{W(W+1)}{W^2}.
$$

Since the choice of $W$ is arbitrary, we conclude that the ARV factor $v$ is well-defined and $v = \frac{1}{2\kappa}$. \qed

**B Analysis of SecondCurtain**

**B.1 Constant $\kappa$**

**Theorem 8.** SecondCurtain is scale-invariant with

$$
\kappa_{\text{SecondCurtain}} = \int_{-\infty}^{\infty} e^{-\frac{1}{4} \frac{1}{q^{a-1/2}+1}} \left(1 + \varphi\left(\frac{1}{q^{a-1}}\right)\right) \prod_{j=1}^{\infty} \left(1 + \varphi\left(\frac{1}{q^{a+j/2}}\right)\right)^2 \frac{1}{q^a} \frac{1}{q} dt,
$$

where $\varphi(x) = \sum_{i=0}^{\infty} \left(e^{-x} \frac{x^i}{i!} - 1\right)$.

**Proof.** Fix a cell $t$. Let $W_2(k)$ be the height of the second highest cell containing a dart in the $k$th column relative to column of cell $t$. Let $\ell = \left[k \text{ is odd}\right]/2$. We have for any $i \in \mathbb{Z},$

$$
\Pr(W_2(k) \leq t + i + \ell) = \prod_{j \geq 1} \Pr(Y_{t+i+j,\lambda} = 0) + \sum_{j \geq 1} \Pr(Y_{t+i+j,\lambda} = 1) \prod_{j \geq 1} \Pr(Y_{t+i+j+w,\lambda} = 0)
$$

$$
= \left(\prod_{j \geq 1} \Pr(Y_{t+i+j,\lambda} = 0) \right) \left(1 + \sum_{j \geq 1} \frac{\Pr(Y_{t+i+j,\lambda} = 1)}{\Pr(Y_{t+i+j,\lambda} = 0)}\right)
$$

$$
= e^{-\frac{\lambda}{q^{t+i+j+1}}} \left(1 + \sum_{j \geq 1} \frac{1 - e^{-\frac{\lambda}{q^{t+i+j}q^{-1}}}}{e^{-\frac{\lambda}{q^{t+i+j}q^{-1}}}}\right).
$$

24
Now we can express the cumulative expression for the tension point $T_2$ of SecondCurtain:

$$\Pr(T_2 \leq i + t) = \Pr \left( \max_{j \in \mathbb{Z} \setminus \{0\}} \{W_2(j) - |j|(a - 1/2)\} \leq i + t \right)$$

$$= \prod_{j \in \mathbb{Z} \setminus \{0\}} \Pr(W_2(j) - |j|(a - 1/2) \leq i + t)$$

$$= \left( \prod_{j=1}^{\infty} e^{-\frac{\lambda}{q^{t+i+1+j}(a-1/2)}} \left( 1 + \varphi\left( \frac{\lambda}{q^{t+i+1+j}(a-1/2)} \right) \right) \right)^2$$

$$= e^{-\frac{\lambda}{q^{t+i+1}} \frac{\lambda^2}{q^{2-1/2}}} \prod_{j=1}^{\infty} \left( 1 + \varphi\left( \frac{\lambda}{q^{t+i+1+j}(a-1/2)} \right) \right)^2 \text{.} \quad (8)$$

From the rules of SecondCurtain, we know

$$Z_{t,\lambda} = [Y_{t,\lambda=0}] \cdot [t > T_2] \cdot [t > W_2(0)].$$

Note that $T_2$ are independent from $Y_{t,\lambda}$ and $W_2$. We have

$$\mathbb{E}(Z_{t,\lambda}) = \Pr(Y_{t,\lambda} = 0, t > W_2(0)) \cdot \Pr(t > T_2)$$

$$= (\Pr(t > W_1(0)) + \Pr(W_1(0) > t > W_2(0))) \cdot \Pr(T_2 \leq t - 1). \quad (9)$$

Note that

$$\Pr(W_1(0) > t > W_2(0)) = \Pr(Y_{t,\lambda} = 0) \cdot \sum_{j \geq 1} \Pr(Y_{t+j,\lambda} = 1) \prod_{k \geq 1, k \neq j} \Pr(Y_{t+k,\lambda} = 0)$$

$$= e^{-\frac{\lambda}{q}} \varphi\left( \frac{\lambda}{q^{t+1}} \right), \quad (10)$$

and from the proof of Theorem 7 we know

$$\Pr(t > W_1(0)) = \Pr(W_1(0) \leq t - 1) = e^{-\frac{\lambda}{q}}. \quad (11)$$

Apply the expressions (8), (10) and (11) to (9) and we have

$$\mathbb{E}(Z_{t,\lambda}) = e^{-\frac{\lambda}{q}} \frac{\lambda^0}{q^{0-1/2}} \prod_{j=1}^{\infty} \left( 1 + \varphi\left( \frac{\lambda}{q^{t+j(a-1/2)}} \right) \right)^2 \left( e^{-\frac{\lambda}{q^j}} + e^{-\frac{\lambda}{q^{j+1}}} \varphi\left( \frac{\lambda}{q^{j+1}} \right) \right)$$

$$= e^{-\frac{\lambda}{q}} \frac{\lambda^0}{q^{0-1/2}} \prod_{j=1}^{\infty} \left( 1 + \varphi\left( \frac{\lambda}{q^{t+j(a-1/2)}} \right) \right)^2 \left( e^{-\frac{\lambda}{q^j}} + e^{-\frac{\lambda}{q^{j+1}}} \varphi\left( \frac{\lambda}{q^{j+1}} \right) \right) \text{.}$$

Then we have, from formula (4),

$$\kappa_{\text{SecondCurtain}} = \lim_{\lambda \to \infty} \lambda \int_{-\infty}^{\infty} e^{-\frac{\lambda}{q^j} \frac{\lambda^0}{q^{0-1/2}}} \left( 1 + \varphi\left( \frac{\lambda}{q^{t+j(a-1/2)}} \right) \right)^2 \left( e^{-\frac{\lambda}{q^j}} + e^{-\frac{\lambda}{q^{j+1}}} \varphi\left( \frac{\lambda}{q^{j+1}} \right) \right) \frac{1}{q^j} \frac{1}{q} dt$$

$$= \int_{-\infty}^{\infty} e^{-\frac{\lambda}{q^j} \frac{\lambda^0}{q^{0-1/2}}} \left( 1 + \varphi\left( \frac{1}{q^{t+i+1}} \right) \right)^2 \left( e^{-\frac{\lambda}{q^j}} + e^{-\frac{\lambda}{q^{j+1}}} \varphi\left( \frac{1}{q^{j+1}} \right) \right) \frac{1}{q^j} \frac{1}{q} dt. \quad \square$$
B.2 Average Number of Bits Per column

Each column uses \( \log_2(2a) \) bits to store its offset in the second curtain. Note that \texttt{SecondCurtain} uses the unary encoding to store the distance between the highest cell and the curtain. Next, we analyze the average number of bits \texttt{SecondCurtain} uses per column for the unary encoding.

Let \( r \in [0, 1) \) be the offset. We call the column with cells of height \( \{i + r \mid i \in \mathbb{Z}\} \) as column \( r \). Note that \( W_1(0) \), \( W_2(0) \) and \( T_2 \) can be defined for column \( r \) by using any cell in that column as the reference cell. When in tension, it takes \( \max(W_1(0) - T_2 + 1, 1) \) bits to store the distance in the \( \delta \) vector. When not in tension, it takes \( (W_1(0) - W_2(0)) \) bits. Let \( K_r \) be the number of bits for the unary coding for a column \( r \). Thus we have

\[
K_r = \left[ W_2(0) \leq T_2 \right] \max(W_1(0) - T_2 + 1, 1) + \left[ W_2(0) > T_2 \right] (W_1(0) - W_2(0)).
\]

**Theorem 9.** The average unary encoding length is independent from \( \lambda \). Specifically,

\[
\int_0^1 \mathbb{E}(K_r) \, dr = 1 + \int_{-\infty}^{\infty} e^{-\frac{\lambda}{q^r+1/2-1}} \prod_{j=1}^{\infty} \left( 1 + \varphi\left( \frac{\lambda}{q^{r+j/(a-1/2)} + 1} \right) \right)^2 \left( e^{-\frac{\lambda}{q^r+1} - e^{-\frac{\lambda}{q^r+2}}} \varphi\left( \frac{\lambda}{q^{r+1}} \right) \right) \, dt.
\]

**Proof.** For simplicity, we write \( W_1(0) \) and \( W_2(0) \) as \( W_1 \) and \( W_2 \). We also write \( T_2 \) as \( T \). Note that \( T \) is independent from \( W_1 \) and \( W_2 \).

First we note that

\[
\max(W_1 - T + 1, 1) = 1 + \left[ W_1 \geq T \right] (W_1 - T).
\]

Then we have

\[
K_r = \left[ W_1 \geq T \right] (W_1 - T + 1) \left[ W_2 \leq T \right] + (W_1 - W_2) \left[ W_2 > T \right]
\]

\[
= \left[ W_2 \leq T \right] + \sum_{i=-\infty}^{\infty} \left[ W_2 \leq T \right] \left[ W_1 \geq r + i > T \right] + \left[ W_2 > T \right] \left[ W_1 \geq r + i > W_2 \right]
\]

\[
= \left[ W_2 \leq T \right] + \sum_{i=-\infty}^{\infty} \left[ r + i > T \right] \left[ W_1 \geq r + i > W_2 \right]. \tag{12}
\]

Note that we have

\[
\Pr(W_1 \geq r + i > W_2) = \sum_{j \geq 0} \Pr(Y_{i+r+j, \lambda} = 1) \prod_{k \geq 0, k \neq j} \Pr(Y_{i+r+k, \lambda} = 0) = e^{-\frac{\lambda}{q^{r+i}}} \varphi\left( \frac{\lambda}{q^{r+i}} \right), \tag{13}
\]

\[
\Pr(W_2 \leq T) = \sum_{i=-\infty}^{\infty} \Pr(T \geq r + i) \Pr(W_2 = r + i) = 1 - \sum_{i=-\infty}^{\infty} \Pr(T < r + i) \Pr(W_2 = r + i), \tag{14}
\]

and

\[
\Pr(W_2 = r + i) = \Pr(Y_{r+i}, \lambda) = 1 \sum_{j \geq 1} \Pr(Y_{r+i+j, \lambda} = 1) \prod_{k \geq 1, k \neq j} \Pr(Y_{r+i+k, \lambda} = 0)
\]

\[
= \left( 1 - e^{-\frac{\lambda}{q^{r+i}}} \varphi\left( \frac{\lambda}{q^{r+i}} \right) \right) e^{-\frac{\lambda}{q^{r+i+1}}} \varphi\left( \frac{\lambda}{q^{r+i+1}} \right)
\]

\[
= \left( e^{-\frac{\lambda}{q^{r+i+1}} - e^{-\frac{\lambda}{q^r}}} \right) \varphi\left( \frac{\lambda}{q^{r+i+1}} \right). \tag{15}
\]
Apply the expressions (13), (14) and (15) to (12) and we have,

\[
\mathbb{E}(K_r) = \mathbb{E} \left( [W_2 \leq T] + \sum_{i=-\infty}^{\infty} [r + i > T] [W_1 \geq r + i > W_2] \right)
\]

\[
= \Pr(W_2 \leq T) + \sum_{i=-\infty}^{\infty} \Pr(i > T) \Pr(W_1 \geq i > W_2)
\]

\[
= 1 + \sum_{i=-\infty}^{\infty} \Pr(r + i > T) (\Pr(W_1 \geq r + i > W_2) - \Pr(W_2 = r + i))
\]

\[
= 1 + \sum_{i=-\infty}^{\infty} e^{-\frac{\lambda}{q^{i+r}} \frac{q^{a-1/2}}{a-1}} \prod_{j=1}^{\infty} \left( 1 + \varphi \left( \frac{\lambda}{q^{i+r} + j(a-1/2)} \right) \right)^2 \left( e^{-\frac{\lambda}{q^{i+r}}} \varphi \left( \frac{\lambda}{q^{i+r}} \right) - \left( e^{-\frac{\lambda}{q^{i+r+1}}} - e^{-\frac{\lambda}{q^{i+r}}} \right) \varphi \left( \frac{\lambda}{q^{i+r+1}} \right) \right).
\]

Finally, due to smoothing, we have (set \( t = i + r \))

\[
\int_{0}^{1} \mathbb{E}(K_r) \, dr = 1 + \int_{-\infty}^{\infty} e^{-\frac{\lambda}{q^{i+r}} \frac{q^{a-1/2}}{a-1}} \prod_{j=1}^{\infty} \left( 1 + \varphi \left( \frac{\lambda}{q^{i+r} + j(a-1/2)} \right) \right)^2 \left( e^{-\frac{\lambda}{q^{i+r}}} \varphi \left( \frac{\lambda}{q^{i+r}} \right) - \left( e^{-\frac{\lambda}{q^{i+r+1}}} - e^{-\frac{\lambda}{q^{i+r}}} \right) \varphi \left( \frac{\lambda}{q^{i+r+1}} \right) \right) \, dt.
\]

By numerical evaluations, the optimal parametrization such that \( a \) is a power of 2 is about \((q,a) = (2.2, 2)\). Taking these parameters, we have \( \kappa_{\text{SecondCurtain}} \approx 0.937 \) and the average number of bits per sketch 3.86. Applying Theorem 4, we get its MVP \(\frac{3.86}{0.937} \approx 2.06\).