Distinct Elements in Streams: An Algorithm for the (Text) Book

Sourav Chakraborty ⓡ[1]
Indian Statistical Institute Kolkata, India

N. V. Vinodchandran ⓡ[1]
University of Nebraska-Lincoln, NE, USA

Kuldeep S. Meel
National University of Singapore, Singapore

Abstract

Given a data stream $\mathcal{D} = \langle a_1, a_2, \ldots, a_m \rangle$ of m elements where each $a_i \in [n]$, the Distinct Elements problem is to estimate the number of distinct elements in $\mathcal{D}$. Distinct Elements has been a subject of theoretical and empirical investigations over the past four decades resulting in space optimal algorithms for it. All the current state-of-the-art algorithms are, however, beyond the reach of an undergraduate textbook owing to their reliance on the usage of notions such as pairwise independence and universal hash functions. We present a simple, intuitive, sampling-based space-efficient algorithm whose description and the proof are accessible to undergraduates with the knowledge of basic probability theory.

2012 ACM Subject Classification Theory of computation → Sketching and sampling

Keywords and phrases F_0 Estimation, Streaming, Sampling

Digital Object Identifier 10.4230/LIPIcs.ESA.2022.34

Funding This work was supported in part by National Research Foundation Singapore under its NRF Fellowship Programme (NRF-NRFFAI1-2019-0004), Singapore Ministry of Education Academic Research Fund Tier 1, Ministry of Education Singapore Tier 2 grant (MOE-T2EP20121-0011), and Amazon Faculty Research Awards. Vinod was supported in part by NSF CCF-2130608 and NSF HDR:TRIPODS-1934884 awards.

1 Introduction

We consider the fundamental problem of estimating the number of distinct elements in a data stream (Distinct Elements problem or the F_0 estimation problem). For a data stream $\mathcal{D} = \langle a_1, a_2, \ldots, a_m \rangle$, where each $a_i \in [n]$, $\mathsf{F}_0(\mathcal{D})$ is the number of distinct elements in $\mathcal{D}$: $\mathsf{F}_0(\mathcal{D}) = |\{a_1, a_2, \ldots, a_m\}|$.

▶ **Problem 1.** *Given a stream $\mathcal{D} = \langle a_1, a_2, \ldots, a_m \rangle$ of m elements where each $a_i \in [n]$, parameters ε, δ, output an (ε, δ)-approximation of $\mathsf{F}_0(\mathcal{D})$. That is, output c such that $\Pr[(1-\varepsilon) \cdot \mathsf{F}_0(\mathcal{D}) \leq c \leq (1+\varepsilon) \cdot \mathsf{F}_0(\mathcal{D})] \geq 1 - \delta$.*

We are interested in streaming algorithms that uses $\mathsf{poly}(\log m, \log n, \varepsilon^{-1}, \log \delta^{-1})$ bits of memory. Since $\mathcal{D}$ is clear from the context, we also use F_0 to refer to $\mathsf{F}_0(\mathcal{D})$.

F_0 estimation problem is a fundamental problem with a long history of theoretical and practical investigations. The seminal work of Flajolet and Martin [9] provided the first algorithm assuming the existence of hash functions with full independence. Subsequent

[1] The authors decided to forgo the old convention of alphabetical ordering of authors in favor of a randomized ordering, denoted by ⓡ. The publicly verifiable record of the randomization is available at `https://www.aeaweb.org/journals/policies/random-author-order/search`

investigations relying on the usage of limited-independence hash functions have led to design of algorithms with optimal space complexity $O(\log n + \frac{\log \delta^{-1}}{\varepsilon^2})$. We defer detailed bibliographical remarks to Section 3. However, all the current space-efficient algorithms are beyond the reach of an undergraduate textbook due to their reliance on notions such as pairwise independence and universal hash functions.

We present a very simple algorithm for the F_0 estimation problem using a sampling strategy that only relies on basic probability for its analysis. In particular, it does not use universal hash functions. While the simplicity of the code makes it appealing to be used in practical implementation, we believe that only using basic probability theory for the analysis makes the algorithm presentable to undergraduates right after the introduction of basic tail bounds. Our algorithm builds and refines ideas introduced in the recent work on estimating the size of the union of sets in the general setting of *Delphic sets* [13].

2 F_0-Estimator: A simple algorithm for F_0 estimation

Algorithm 1 F_0-Estimator.

Input Stream $\mathcal{D} = \langle a_1, a_2, \ldots, a_m \rangle$, ε, δ
Initialize $p \leftarrow 1$; $\mathcal{X} \leftarrow \emptyset$; thresh $\leftarrow \frac{12}{\varepsilon^2} \log(\frac{8m}{\delta})$
for $i = 1$ to m **do**
  $\mathcal{X} \leftarrow \mathcal{X} \setminus \{a_i\}$
  With probability p, $\mathcal{X} \leftarrow \mathcal{X} \cup \{a_i\}$
  if $|\mathcal{X}| =$ thresh **then**
    Throw away each element of $\mathcal{X}$ with probability $\frac{1}{2}$
    $p \leftarrow \frac{p}{2}$
    if $|\mathcal{X}| =$ thresh **then**
      Output $\perp$
Output $\frac{|\mathcal{X}|}{p}$

The algorithm F_0-Estimator uses a simple sampling strategy: it keeps a set $\mathcal{X}$ of samples at all times such that every element seen so far is independently in $\mathcal{X}$ with equal probability. In order to keep the set of samples small, it makes sure that $\mathcal{X}$ does not grow beyond the value thresh by adjusting the sampling rate p accordingly. After all the elements of the stream are processed, it outputs $\frac{|\mathcal{X}|}{p}$ where p is the final sampling rate.

2.1 Theoretical Analysis

We present the theoretical analysis entirely based on first principles, which adds to its length. For readers who are familiar with randomized algorithms, the proof is standard.

We state the following well-known concentration bound, Chernoff bound, for completeness.

▶ **Fact 2** (Chernoff's Bound). *Let $v_1, ..., v_k$ be independent random variables taking values in $\{0, 1\}$. Let $V = \sum_{i=1}^{k} v_i$ and $\mu = \mathbb{E}[V]$. Then $\Pr(|V - \mu| \geq \delta\mu) \leq 2e^{-\frac{\delta^2\mu}{3}}$*

The following theorem captures the correctness and space complexity guarantee of F_0-Estimator.

▶ **Theorem 3.** *For any data stream $\mathcal{D}$ and any $0 < \delta, \varepsilon < 1$, the algorithm F_0-Estimator outputs an (ε, δ)-approximation of $\mathsf{F}_0(\mathcal{D})$. The algorithm uses $O(\frac{1}{\varepsilon^2} \cdot \log n \cdot (\log m + \log 1/\delta))$ space in the worst case.*

Proof. The stated space complexity bound of the algorithm follows because, from the description, it is clear that the size of the set of samples kept by the algorithm is always $\leq \mathsf{thresh}$, and each item requires $\lceil \log_2 n \rceil$ bits to store.

In order to prove the algorithm outputs the correct estimate with high probability, we show that when the algorithm terminates, every distinct element of stream $\mathcal{D}$ is in $\mathcal{X}$ with a probability p, where $p \geq \frac{\mathsf{thresh}}{4\mathsf{F}_0}$. This guarantee, together with the Chernoff bound, implies the correctness of the algorithm. We give a formal proof of correctness below.

Consider the following two events:

Error : 'The algorithm F_0-$\mathsf{Estimator}$ does not return a value in the range $[(1-\varepsilon)\mathsf{F}_0, (1+\varepsilon)\mathsf{F}_0]$'
Fail : "The algorithm F_0-$\mathsf{Estimator}$ outputs $\perp$."

We will bound $\Pr[\mathsf{Error}]$ by δ. Observe that $\Pr[\mathsf{Error}] \leq \Pr[\mathsf{Fail}] + \Pr[\mathsf{Error} \cap \overline{\mathsf{Fail}}]$.

▷ Claim 4. $\Pr[\mathsf{Fail}] \leq \frac{\delta}{8}$

Proof of Claim. Let Fail_j denote the event that Algorithm 1 returns $\perp$ when $i = j$. Formally, Fail_j: '$|\mathcal{X}| = \mathsf{thresh}$ and none of the elements of $\mathcal{X}$ are thrown away at line 6' for $i = j$. The probability that Fail_j happens is $\left(\frac{1}{2}\right)^{\mathsf{thresh}}$. Therefore,

$$\Pr[\mathsf{Fail}] \leq \sum_{j=1}^{m} \Pr[\mathsf{Fail}_j] \leq m \cdot \left(\frac{1}{2}\right)^{\mathsf{thresh}} \leq \frac{\delta}{8}$$ ◁

▷ Claim 5. $\Pr[\mathsf{Error} \cap \overline{\mathsf{Fail}}] \leq \frac{\delta}{2}$.

We give a detailed proof of this claim below. Theorem follows from the above two claims. ◀

Proof of Claim 5

To bound $\Pr[\mathsf{Error} \cap \overline{\mathsf{Fail}}]$, we consider a relaxed version of Algorithm F_0-$\mathsf{Estimator}$, which is stated as Algorithm 2. Algorithm 2 is nothing but F_0-$\mathsf{Estimator}$ with lines 8 and 9 removed. Observe that for a given input, the algorithm F_0-$\mathsf{Estimator}$ behaves identically to Algorithm 2 as long as $|\mathcal{X}| \leq \mathsf{thresh}$ after each element of $\mathcal{X}$ is thrown away with probability $\frac{1}{2}$ (i.e., the event Fail does not happen). Now, we consider the following event:

Error_2 : 'The Algorithm 2 does not output a value in the range $[(1-\varepsilon)\mathsf{F}_0, (1+\varepsilon)\mathsf{F}_0]$.'

Observe that $\Pr[\mathsf{Error} \cap \overline{\mathsf{Fail}}] \leq \Pr[\mathsf{Error}_2]$. In Claim 7, we obtain the desired bound on $\Pr[\mathsf{Error}_2]$ and hence on $\Pr[\mathsf{Error} \cap \overline{\mathsf{Fail}}]$.

To prove an upper bound on $\Pr[\mathsf{Error}_2]$ in Claim 7, we will need the following claim. In the following, we use S_j to denote $\{a_1, a_2, \ldots, a_j\}$ – distinct elements that appear in the first j items in the stream.

▷ Claim 6. The following loop invariant holds in the **for** loop (lines 2– 7) of Algorithm 2:

Every element in S_j is in $\mathcal{X}$ independently with probability p.

Proof. First, we show that if the loop invariant holds after execution of line 4, then it holds after the execution of if-then-block (line 5–7). Since every element of $\mathcal{X}$ is thrown away independently with probability $\frac{1}{2}$ and p is updated to $p/2$, the invariant holds after the execution of the if-then-block (line 5–7).

Now we return our attention to proving that the invariant holds after the execution of line 4 for every iteration j. The proof proceeds via induction.

Algorithm 2

Input Stream $\mathcal{D} = \langle a_1, a_2, \ldots, a_m \rangle$, ε, δ
1: **Initialize** $p \leftarrow 1$; $\mathcal{X} \leftarrow \emptyset$; $\mathsf{thresh} \leftarrow \frac{12}{\varepsilon^2} \log(\frac{8m}{\delta})$
2: **for** $i = 1$ to m **do**
3: $\quad \mathcal{X} \leftarrow \mathcal{X} \setminus \{a_i\}$
4: $\quad$ With probability p, $\mathcal{X} \leftarrow \mathcal{X} \cup \{a_i\}$
5: $\quad$ **if** $|\mathcal{X}| = \mathsf{thresh}$ **then**
6: $\quad\quad$ Throw away each element of $\mathcal{X}$ with probability $\frac{1}{2}$
7: $\quad\quad$ $p \leftarrow \frac{p}{2}$
8: **Output** $\frac{|\mathcal{X}|}{p}$

Base Case. After line 4, $\Pr[a_1 \in \mathcal{X}] = p$. Since $\mathsf{thresh} > 1$, the condition in line 5 is not satisfied, therefore the desired invariant holds true.

Inductive Step. Let us assume by induction hypothesis, the desired invariant holds true after iteration $j-1$. Note that the execution of line 3–line 4 only affects whether $a_j \in \mathcal{X}$ independently of all $a_k \in S_j \setminus a_j$. There are two cases:

- $a_j \notin S_{j-1}$, then after the execution of line 4, we have $\Pr[a_j \in \mathcal{X}] = p$.
- $a_j \in S_{j-1}$, then after line 3, we have $\Pr[a_j \in \mathcal{X}] = 0$. After line 4, we have $\Pr[a_j \in \mathcal{X}] = p$.

◁

▷ Claim 7. $\Pr[\mathsf{Error}_2] \leq \frac{\delta}{2}$

Proof. Given the loop invariant stated in Claim 6, we have that every element of S_m is in $\mathcal{X}$ with probability p. In order to upper bound that the probability that $\frac{|\mathcal{X}|}{p}$ lies outside $[(1-\varepsilon)\mathsf{F}_0, (1+\varepsilon)\mathsf{F}_0]$ we will use Chernoff's bound. For this we first establish a lower bound on p. To this end, we decompose $\Pr[\mathsf{Error}_2]$ conditioned on lower bound on p. In particular, we define the following event:

Bad: "The value of p at line 8 in Algorithm 2 is less than $\frac{\mathsf{thresh}}{4\mathsf{F}_0}$."
Let $\ell = \lceil \log(\frac{\mathsf{thresh}}{4\mathsf{F}_0}) \rceil$. Since every value of p can be expressed as power of 2, we have that $p < 2^\ell$ if and only if $p < \frac{\mathsf{thresh}}{4\mathsf{F}_0}$. Observe that $\Pr[\mathsf{Error}_2] \leq \Pr[\mathsf{Bad}] + \Pr[\mathsf{Error}_2 \mid \overline{\mathsf{Bad}}]$. We will upper bound $\Pr[\mathsf{Bad}]$ and $\Pr[\mathsf{Error}_2 \mid \overline{\mathsf{Bad}}]$ separately.

Bounding $\Pr[\mathsf{Bad}]$. For $j \in [1, m]$, let Bad_j denote the event that 'jth iteration of the **for** loop is the first iteration where the value of p goes below 2^ℓ' i.e., the value of p at the end of iteration $(j-1)$ is 2^ℓ and the value of p is $2^{\ell-1}$ at the end of iteration j. Therefore, by definition of Bad_j, we have $|\mathcal{X}| = \mathsf{thresh}$ and $p = 2^\ell$ in line 5 of Algorithm 2. Recall that in every iteration of the loop, the value of p can decrease at most by a factor of $\frac{1}{2}$ and cannot increase. Therefore, we have $\Pr[\mathsf{Bad}] \leq \sum_{j=1}^{m} \Pr[\mathsf{Bad}_j]$. We will now compute $\Pr[\mathsf{Bad}_j]$ for a fixed j.

For $a \in S_m$ let r_a denote the indicator random variable indicating whether a is in the set $\mathcal{X}$. By Claim 6 the random variables $\{r_a\}_{a \in S_m}$ are independent and for all $a \in S_m$ $\Pr[r_a = 1] = p$. Since, $|\mathcal{X}| = \sum_{a \in S_j} r_a$ we have $\mathbb{E}[|\mathcal{X}|] = \mathbb{E}\left[\sum_{a \in S_j} r_a\right] = \sum_{a \in S_j} \Pr[r_a = 1] = p \cdot |S_j| = p \cdot \mathsf{F}_0$. Thus,

$$\Pr[\mathsf{Bad}_j] \leq \Pr[|\mathcal{X}| = \mathsf{thresh} \mid p = 2^\ell] \leq \Pr[|\mathcal{X}| \geq \mathsf{thresh} \mid p = 2^\ell] \leq 2e^{-\frac{\mathsf{thresh}^2}{3 \cdot p\mathsf{F}_0}}$$
$$= 2e^{-\frac{\mathsf{thresh}^2}{3 \cdot 2^\ell \mathsf{F}_0}} \leq \frac{\delta}{4m},$$

the last inequality follows from the values of ℓ and thresh. Therefore, $\Pr[\mathsf{Bad}] \leq \frac{\delta}{4}$.

Bounding $\Pr[\mathsf{Error}_2 \mid \overline{\mathsf{Bad}}]$. Similar to the above, for $a \in S_m$, let r_a denote the indicator random variable indicating whether a is in the set $\mathcal{X}$. By Claim 6 the random variables $\{r_a\}_{a \in S_m}$ are independent and for all $a \in S_m$, we have $\Pr[r_a = 1] = p$. Thus $|\mathcal{X}| = \sum_{a \in \mathcal{D}} r_a$ and thus $\mathbb{E}[|\mathcal{X}|] = p\mathsf{F}_0$. Conditioned on $\overline{\mathsf{Bad}}$, we have $p \geq \frac{\mathsf{thresh}}{4\mathsf{F}_0}$. Thus,

$$\begin{aligned}
\Pr[\mathsf{Error}_2 \mid \overline{\mathsf{Bad}}] &= \Pr\left[(1-\epsilon)p\mathsf{F}_0 \leq |\mathcal{X}| \leq (1+\varepsilon)p\mathsf{F}_0 \mid \overline{\mathsf{Bad}}\right] \\
&\leq 2e^{-\frac{\varepsilon^2 \mathsf{thresh}}{12}} \qquad \left[\text{Using Chernoff bound with } p \geq \frac{\mathsf{thresh}}{4\mathsf{F}_0}\right] \\
&\leq \frac{\delta}{4m} \leq \frac{\delta}{4}.
\end{aligned}$$

◁

3 Bibliographic Remarks

Distinct Elements problem (or F_0 estimation problem) is one of the most investigated problem in the data streaming model [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]. While the Distinct Elements problem has a wide range of applications in several areas of computing, it was first investigated in the algorithms community by Flajolet and Martin [9]. They provided the first approximation under the assumption of the existence of hash functions with full independence. The seminal work of Alon, Matias, and Szegedy [1] that introduced the data streaming model of computation revisited this problem as a special case of F_k estimation problem and achieved space complexity of $O(\log n)$ for $\varepsilon > 1$ and constant δ. The first (ε, δ) approximation for Distinct Elements problem was Gibbson and Tirthpura who achieved $O(\frac{\log n}{\varepsilon^2})$ space complexity [10]. Bar-Yossef, Jayram, Kumar, Sivakumar and Trevisan improved the space complexity bound to $\tilde{O}(\log n + 1/\varepsilon^2)$ [2]. Subsequently, Kane, Nelson, and Woodruff achieved $O(\log n + 1/\varepsilon^2)$ which is optimal in n and ε [12]. All the above bounds are for a fixed confidence parameter δ, which can be amplified to achieve confidence bounds for arbitrary δ by simply running $\log(1/\delta)$-estimators in parallel and returning the median. This incurs a multiplicative factor of $\log(1/\delta)$. Błasiok designed an (ε, δ) approximation algorithm for F_0 estimation problem with space complexity of $O(\frac{\log \delta^{-1}}{\varepsilon^2} + \log n)$, thereby matching the lower bound in all the three parameters n, ε and δ [4]. As is expected, every subsequent improvement added to the complexity of the algorithm or the analysis, and a majority of these work remain beyond the reach of non-experts. A crucial technical ingredient for all the works mentioned above is their careful usage of limited-independence hash functions in order to make space $\mathsf{poly}(\log n)$. Monte Carlo-based approaches have been utilized in the context of size estimation of the union of sets, but their straightforward adaptation to the streaming setting did not seem to yield progress. Recently, a new sampling-based approach was proposed in the context of estimating the size of the union of sets in the streaming model that achieves space complexity with $\log m$-dependence [13]. The algorithm we presented adapts ideas from this work to the context of F_0 estimation.

References

1 Noga Alon, Yossi Matias, and Mario Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Syst. Sci.*, 58(1):137–147, 1999. doi:10.1006/jcss.1997.1545.

2 Ziv Bar-Yossef, T. S. Jayram, Ravi Kumar, D. Sivakumar, and Luca Trevisan. Counting distinct elements in a data stream. In *Proc. of RANDOM*, pages 1–10, 2002. doi:10.1007/3-540-45726-7_1.

3 Ziv Bar-Yossef, Ravi Kumar, and D. Sivakumar. Reductions in streaming algorithms, with an application to counting triangles in graphs. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms,*, pages 623–632, 2002. URL: http://dl.acm.org/citation.cfm?id=545381.545464.

4 Jaroslaw Blasiok. Optimal streaming and tracking distinct elements with high probability. In *Proc. of SODA*, 2018. doi:10.1137/1.9781611975031.156.

5 Joshua Brody and Amit Chakrabarti. A multi-round communication lower bound for gap hamming and some consequences. In *Proceedings of the 24th Annual IEEE Conference on Computational Complexity, CCC 2009*, pages 358–368. IEEE Computer Society, 2009. doi:10.1109/CCC.2009.31.

6 Marianne Durand and Philippe Flajolet. Loglog counting of large cardinalities (extended abstract). In *Algorithms – ESA 2003, 11th Annual European Symposium*, volume 2832, pages 605–617, 2003. doi:10.1007/978-3-540-39658-1_55.

7 Cristian Estan, George Varghese, and Michael E. Fisk. Bitmap algorithms for counting active flows on high-speed links. *IEEE/ACM Trans. Netw.*, 14(5):925–937, 2006. doi:10.1145/1217709.

8 Philippe Flajolet, Éric Fusy, Olivier Gandouet, and Frédéric Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *Discrete Mathematics and Theoretical Computer Science*, pages 137–156, 2007. doi:10.46298/dmtcs.3545.

9 Philippe Flajolet and G. Nigel Martin. Probabilistic counting algorithms for data base applications. *J. Comput. Syst. Sci.*, 31(2):182–209, 1985. doi:10.1016/0022-0000(85)90041-8.

10 Phillip B. Gibbons and Srikanta Tirthapura. Estimating simple functions on the union of data streams. In *Proceedings of the Thirteenth Annual ACM Symposium on Parallel Algorithms and Architectures, SPAA*, pages 281–291, 2001. doi:10.1145/378580.378687.

11 Piotr Indyk and David P. Woodruff. Tight lower bounds for the distinct elements problem. In *Proc. of FOCS*, pages 283–288. IEEE Computer Society, 2003. doi:10.1109/SFCS.2003.1238202.

12 Daniel M. Kane, Jelani Nelson, and David P. Woodruff. An optimal algorithm for the distinct elements problem. In *Proceedings of the Twenty-Ninth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS*, pages 41–52, 2010. doi:10.1145/1807085.1807094.

13 Kuldeep S. Meel, N. V. Vinodchandran, and Sourav Chakraborty. Estimating the size of union of sets in streaming models. In *PODS'21: Proceedings of the 40th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 126–137, 2021. doi:10.1145/3452021.3458333.