On synopses for distinct-value estimation under multiset operations

Conference Paper · June 2007

DOI: 10.1145/1247480.1247504 · Source: DBLP

5 authors, including:

Berthold Reinwald
IBM
110 PUBLICATIONS **2,770** CITATIONS
SEE PROFILE

Yannis Sismanis
Google Inc.
46 PUBLICATIONS **1,828** CITATIONS
SEE PROFILE

Rainer Gemulla
Universität Mannheim
73 PUBLICATIONS **3,404** CITATIONS
SEE PROFILE

On Synopses for Distinct-Value Estimation Under Multiset Operations

Kevin Beyer[1] Peter J. Haas[1] Berthold Reinwald[1] Yannis Sismanis[1] Rainer Gemulla[2]

[1]IBM Almaden Research Center
San Jose, CA, USA
{kbeyer,phaas,reinwald,syannis}@us.ibm.com

[2]Technische Universität Dresden
Dresden, Germany
gemulla@inf.tu-dresden.de

ABSTRACT

The task of estimating the number of distinct values (DVs) in a large dataset arises in a wide variety of settings in computer science and elsewhere. We provide DV estimation techniques that are designed for use within a flexible and scalable "synopsis warehouse" architecture. In this setting, incoming data is split into partitions and a synopsis is created for each partition; each synopsis can then be used to quickly estimate the number of DVs in its corresponding partition. By combining and extending a number of results in the literature, we obtain both appropriate synopses and novel DV estimators to use in conjunction with these synopses. Our synopses can be created in parallel, and can then be easily combined to yield synopses and DV estimates for arbitrary unions, intersections or differences of partitions. Our synopses can also handle deletions of individual partition elements. We use the theory of order statistics to show that our DV estimators are unbiased, and to establish moment formulas and sharp error bounds. Based on a novel limit theorem, we can exploit results due to Cohen in order to select synopsis sizes when initially designing the warehouse. Experiments and theory indicate that our synopses and estimators lead to lower computational costs and more accurate DV estimates than previous approaches.

Categories and Subject Descriptors

H.2 [**Database Management**]: Miscellaneous

General Terms

Algorithms, theory

Keywords

distinct-value estimation, synopsis warehouse

1. INTRODUCTION

The task of determining the number of distinct values (DVs) in a large dataset arises in a wide variety of settings in computer science, including data integration [4, 8], query optimization [22, 29], network monitoring [12], and OLAP [28, 31]. The number of distinct values can be computed exactly by sorting the dataset and then executing a straightforward scan-and-count pass over the data; alternatively, a hash table can be constructed and used to compute the number of distinct values. Neither of these approaches scales well to the massive datasets often encountered in practice, because of heavy time and memory requirements. A great deal of research over the past twenty five years has therefore focused on approximate methods that scale to very large datasets. These methods work either by drawing a random sample of the data items and using the observed frequencies of the values in the sample as a basis for estimation [6, 19, 20] or by taking a single pass through the data and using hashing techniques to compute an estimate using a bounded amount of memory [1, 2, 3, 11, 12, 13, 14, 16, 18, 33].

Almost all of this work has focused on producing a given synopsis of the dataset and then using the synopsis to obtain a DV estimate; issues related to combining and exploiting synopses in the presence of set operations on multiple datasets have gone largely unexplored. Such issues are the focus of this paper, which is about DV estimation methods in the context of a "synopsis warehouse" environment as described in [5].[1] In a synopsis warehouse, incoming data is split into *partitions*, i.e., multisets of values, and a synopsis is created for each partition; the synopses are used to quickly estimate various partition properties. As partitions are rolled in and out of a full-scale warehouse, the corresponding synopses are rolled in and out of the synopsis warehouse. The architecture requires that synopses can be created in parallel, ensuring scalability, and that synopses can be combined to create a synopsis corresponding to the union, intersection, or difference of the corresponding partitions, providing flexibility. We use the term "partition" here in a very general sense. Data may be partitioned — e.g., by timestamp, by data value, and so forth — for purposes of parallel processing and dealing with fluctuating data-arrival rates. Data may also, however, be partitioned by its source — e.g., SAP customer addresses versus PeopleSoft customer addresses. In the latter scenario, comparison of data characteristics in different partitions may be of interest for purposes of metadata discovery and automated data integration [4]. For example, DV estimates can be used to detect keys and duplicates in a partition, can help discover subset-inclusion and functional-dependency relationships, and can be used to approximate the Jaccard distance or other similarity metrics between the domains of two partitions [4, 8].

Our goal is therefore to provide "warehouse-ready" synopses for DV estimation, as well as corresponding DV estimators that exploit these synopses. As indicated above, it is essential that, whenever

[1]Similar ideas appear in [8], where the synopses are called "signatures."

SIGMOD'07, June 12–14, 2007, Beijing, China.

a compound data partition is created via multiset operations on a collection of input partitions, the synopsis for the compound partition can be easily obtained by combining the synopses of the input partitions. We also strive to maintain the best possible accuracy in our DV estimates, especially when the size of the synopsis is small: as discussed in the sequel, the size of the synopsis for a compound partition is limited by the size of the smallest input synopsis.

We bring together a variety of ideas from the literature — see Section 2 — to obtain a solution to our problem, resulting in best-of-breed DV estimation methods. More specifically, we propose the use of an "AKMV synopsis," extending an idea in [3] in order to handle multiset operations gracefully; our extension involves adding counters to the basic synopsis, in the spirit of [15, 16, 31]. We then provide methods for combining AKMV synopses such that the collection of these synopses is "closed" under multiset operations on the parent partitions. The AKMV synopsis can also handle deletions of individual partition elements. Our new DV estimator is a deceptively simple modification of an estimator proposed in [3]. Using a probabilistic model of hashing, we apply results from the theory of order statistics to show that our proposed estimator is unbiased and has lower mean-squared error than the "basic" DV estimator that underlies most current methods. We also derive exact moment formulas and probabilistic error bounds for the unbiased estimator, along with asymptotic approximations to these quantities that can be employed when the number of DVs is known to be large. Our asymptotic analysis rests on a new limit theorem that allows us to exploit results in [7], which were originally developed for estimating the size of a transitive closure. The asymptotic error bounds can be used to help determine appropriate synopsis sizes when designing the synopsis warehouse. We also provide a maximum-likelihood estimator that is asymptotically equivalent to our unbiased estimator as the synopsis size and number of DVs becomes large; it follows that when there are many distinct values and the synopsis size is large, the unbiased estimator has essentially the minimal possible variance of any DV estimator. We then show how our new estimator can be modified to obtain unbiased DV estimates in the presence of multiset operations.

The remainder of the paper is organized as follows. We review previous approaches to DV estimation in Section 2. In Section 3, we describe the KMV synopsis, a simpler version of our ultimate AKMV synopsis that was essentially proposed in [3], and discuss its space requirements and construction cost. We then introduce our unbiased estimator in Section 4, and give both an exact and an asymptotic analysis of the estimator's behavior. The issues involved in combining synopses are covered in Section 5. We show that the KMV synopsis can be used successfully in certain estimation scenarios, but needs to be augmented in order to deal with multiset difference operations; the resulting AKMV synopsis has the desirable closure property mentioned above. We also show how to extend our simple unbiased estimator to provide unbiased estimates in the presence of set operations, obtaining a direct unbiased estimate of the Jaccard distance in the process. We provide an empirical evaluation of our methods in Section 6 and conclude in Section 7.

2. RELATED WORK

We now discuss previously-proposed synopses, DV estimators, and methods for handling compound partitions.

2.1 Synopses for DV Estimation

In general, the literature on DV estimation does not discuss synopses explicitly, and hence does not discuss issues related to combining synopses in the presence of set operations on the corresponding partitions. We can, however, directly infer potential candidate synopses from the various algorithm descriptions.

2.1.1 Bit-Vector Synopses

The oldest class of synopses comprises various types of bit vectors. The "linear counting" technique [2, 12, 33] uses a bit vector V of length $M = O(D)$, together with a hash function $h\colon \mathscr{D}(A) \mapsto \{1, 2, \ldots, M\}$, where $\mathscr{D}(A)$ denotes the domain of the partition A of interest. The function h is applied to each element $v \in A$, and the $h(v)$th bit of V is set to 1. After A has been scanned, the estimate of D, the number of distinct values in A, is the total number of 1-bits in V, multiplied by a correction factor. The correction factor compensates for undercounting due to "hash collisions" in which $h(v) = h(v')$ for $v \neq v'$; see, for example, [2]. The $O(D)$ storage requirement for linear counting is often prohibitive in applications where D can be very large and multiple DV estimators must be maintained.

The "logarithmic counting" method of Flajolet and Martin [2, 14] uses a bit vector of length $L = O(\log D)$. The idea is to hash each of the distinct values in A to the set $\{0,1\}^L$ of binary strings of length L, and keep track of r, the position (counting from the left, starting at 0) of the leftmost 0 bit over all of the hashed values. The estimate is roughly of the form 2^r (multiplied by a certain factor that corrects for "bias" and hash collisions). This tracking of r is achieved by taking each hashed value, transforming the value by zeroing out all but the leftmost 1, and computing the bitwise-OR of the transformed values. The value of r is then given by the leftmost 0 bit in the resulting bit vector. In the complete algorithm, several independent values of r are, in effect, averaged together (using a technique called "stochastic averaging") and then exponentiated. Alon et al. [1] analyze a variant of the logarithmic counting algorithm under an assumption of pairwise-independent hashing. Recent work by Durand and Flajolet [11] improves on the storage requirement of the logarithmic counting algorithm by tracking and maintaining r, the position of the leftmost 0, directly. The number of bits needed to encode r is $O(\log \log D)$, and hence the technique is called LogLog counting.

The main drawback of the above bit-vector data structures, when used as synopses in our warehouse setting, is that union is the only supported set operation. One must, e.g., resort to the inclusion/exclusion formula to handle set intersections. As the number of set operations increases, this approach becomes extremely cumbersome, expensive, and inaccurate.

Several authors [15, 31] have proposed replacing each bit in the logarithmic-counting bit vector by an exact or approximate counter, in order to permit DV estimation in the presence of both insertions and deletions to the dataset. This modification does not ameliorate the inclusion/exclusion problem, however.

2.1.2 Random Samples

Another synopsis possibility is to use a random sample of the data items in the specified partition [6, 19, 20]. The key drawback is that DV estimates computed from such a synopsis can be very inaccurate, especially when the data is skewed or when there are many distinct values, each having a low frequency (but not all unique); see [6] for a negative result on the performance of sample-based estimators. Moreover, combining synopses to handle unions of partitions can be expensive [5], and it appears that the inclusion/exclusion formula is needed to handle intersections.

2.1.3 Sample-Counting Synopsis

Another type of synopsis arises from the "sample counting" DV-estimation method — also called "adaptive sampling" — credited

to Wegman [2, 13]. Here the synopsis for partition A comprises a subset of $\{h(v): v \in \mathscr{D}(A)\}$, where $h: \mathscr{D}(A) \mapsto \{0,1,\ldots,M\}$ is a hash function. In more detail, the synopsis comprises a fixed-size buffer that holds binary strings of length $L = \log(M)$, together with a "reference" binary string s, also of length L. The idea is to hash the distinct values in the partition, as in logarithmic counting, and insert the hashed values into a buffer that can hold up to $k > 0$ hashed values; the buffer tracks only the distinct hash values inserted into it. When the buffer fills up, it is purged by removing all hashed values whose leftmost bit is not equal to the leftmost bit of s; this operation removes roughly half of the hashed values in the buffer. From this point on, a hashed value is inserted into the buffer if and only if the first bit matches the first bit of s. The next time the buffer fills up, a purge step (with subsequent filtering) is performed by requiring that the two leftmost bits of each hashed value in the buffer match the two leftmost bits of the reference string. This process continues until all the values in the partition have been hashed. The final DV estimate is roughly equal to $K2^r$, where r is the total number of purges that have occurred and K is the final number of values in the buffer.

The algorithms in [3, 16, 17] embody the same idea, essentially with a "reference string" equal to $00\cdots0$. Indeed, the number of purges in the sample-counting algorithm corresponds to the "die level" used in [3, 16, 17]. One difference in these algorithms is that the actual data values, and not the hashed values, are stored: the level at which a data value is stored encodes the number of leading 0's in its hashed representation. In [16], the stored values are augmented with additional information. Specifically, for each distinct value in the buffer, the algorithm maintains the number of instances of the value in the dataset (here a relational table) and also maintains a reservoir sample [32] of the rows in the table that contain the value. This additional information can be exploited to obtain approximate answers, with probabilistic error guarantees, to a variety of SELECT DISTINCT queries over a partition. Such queries include, as a special case, the SELECT COUNT(DISTINCT) query that corresponds to our desired DV estimate. In [3], the basic sample-counting algorithm is enhanced by compressing the stored values.

For sample-counting algorithms with reference string equal to $00\cdots0$, the synopsis holds the K smallest hashed values encountered, where K lies roughly between $k/2$ and k.

2.1.4 The Bellman Synopsis

In the context of the Bellman system, the authors in [8] propose a synopsis related to DV estimation. This synopsis comprises k entries and uses independent hash functions $h_1, h_2, \ldots, h_k$; the ith synopsis entry is given by the ith *minHash* value $m_i = \min_{v \in \mathscr{D}(A)} h_i(v)$. The synopsis for a partition is not actually used to directly compute the number of DVs in the partition, but rather to compute the Jaccard distance between partitions (see Section 2.3 below). When constructing the synopsis, each scanned data item in the partition incurs a cost of $O(k)$, since the item must be hashed k times for comparison to the k current minHash values.

2.2 DV Estimators

The motivation behind virtually all DV estimators can be viewed as follows. If $D \gg 1$ points are placed randomly and uniformly on the unit interval, then, by symmetry, the expected distance between any two neighboring points is $1/(D+1) \approx 1/D$, so that the expected value of $U_{(k)}$, the kth smallest point, is $E[U_{(k)}] \approx \sum_{j=1}^{k}(1/D) = k/D$. Thus $D \approx k/E[U_{(k)}]$. The simplest estimator of $E[U_{(k)}]$ is simply $U_{(k)}$ itself,[2] and yields the *basic estimator*

$$\hat{D}_k^{\mathrm{BE}} = k/U_{(k)}.$$

The simplest connection between the above idea and the DV-estimation problem rests on the observation that a hash function often "looks like" a uniform random number generator. In particular, let $v_1, v_2, \ldots, v_D$ be an enumeration of the distinct values in dataset A and let h be a hash function as before. For many hash functions, the sequence $h(v_1), h(v_2), \ldots, h(v_D)$ will look like the realization of a sequence of independent and identically distributed (i.i.d.) samples from the discrete uniform distribution on $\{0,1,\ldots,M\}$. Provided that M is sufficiently greater than D, the sequence $U_1 = h(v_1)/M, U_2 = h(v_2)/M, \ldots, U_D = h(v_D)/M$ will approximate the realization of a sequence of i.i.d. samples from the continuous uniform distribution on $[0,1]$. This assertion requires that M be much larger than D to avoid collisions, i.e., to ensure that, with high probability, $h(v_i) \neq h(v_j)$ for all $i \neq j$. A "birthday problem" argument [27, p. 45] shows that collisions will be avoided when $M = O(D^2)$. We assume henceforth that, for all practical purposes, any hash function that arises in our discussion avoids collisions. We use the term "looks like" in an empirical sense, which suffices for applications. Thus, in practice, the estimator $\hat{D}_k^{\mathrm{BE}}$ can be applied with $U_{(k)}$ taken as the kth smallest hash value (normalized by a factor of $1/M$). Note that the function $f(x) = 1/x$ is strictly convex on $(0,\infty)$, so that

$$E[\hat{D}_k^{\mathrm{BE}}] = E\left[k/U_{(k)}\right] > k/E\left[U_{(k)}\right] \approx D$$

by Jensen's inequality. That is, the estimator $\hat{D}_k^{\mathrm{BE}}$ is biased upwards for each possible value of D. In Section 4, we provide an unbiased estimator that also has lower mean-squared error than $\hat{D}_k^{\mathrm{BE}}$.

Note that, in a certain sense, the foregoing view of hash functions — as algorithms that effectively place points on the unit interval according to a uniform distribution — represents a worst-case scenario with respect to the basic estimator. To the extent that a hash function spreads points *evenly* on $[0,1]$, i.e., without the slight clumping that is a byproduct of randomness, the estimator $\hat{D}_k^{\mathrm{BE}}$ will yield more accurate estimates. We have observed this phenomenon experimentally; see Section 6.

The estimator $\hat{D}_k^{\mathrm{BE}}$ was proposed in [3], along with conservative error bounds based on Chebyshev's inequality; the motivation behind the estimator is essentially the one given above. Interestingly, both the logarithmic and sample-counting estimators can be viewed as approximations to the basic estimator. For logarithmic counting — specifically the Flajolet-Martin algorithm — consider the binary decimal representation of the normalized hash values $h(v)/M$, where $M = 2^L$. E.g., a hash value $h(v) = 00100110$, after normalization, will have the binary decimal representation 0.00100110. It can be seen that the smallest normalized hash value is approximately equal to 2^{-r}, so that, modulo the correction factor, the Flajolet-Martin estimator (without stochastic averaging) is $1/2^{-r}$, which roughly corresponds to $\hat{D}_1^{\mathrm{BE}}$. The final F-M estimator uses stochastic averaging to average independent values of r and hence compute an estimator $\hat{E}$ of $E[\log_2 \hat{D}_1^{\mathrm{BE}}]$, leading to a final estimate of $\hat{D} = c2^{\hat{E}}$, where the constant c approximately unbiases the estimator. (Our new estimators are exactly unbiased.) For sample counting, suppose, without loss of generality, that the reference string is $00\cdots0$ and, as before, consider the normalized binary decimal representation of the hashed values. Thus the first purge leaves behind normalized values of the form $0.0\cdots$, the second

[2] In the statistical literature, this estimator is called the *method-of-moments* estimator of $E[U_{(k)}]$.

purge leaves behind values of the form $0.00\cdots$, and so forth, the last (rth) purge leaving behind only normalized hashed values with r leading 0's. Thus the number 2^{-r} (which has $r-1$ leading 0's) is roughly equal to the largest of the K normalized hashed values in the size-k buffer, so that the estimate $K/2^{-r}$ is roughly equal to $\hat{D}_k^{\mathrm{BE}}$.

Giroire [18] studies a variant of $\hat{D}_k^{\mathrm{BE}}$ in which the hashed values are divided into $m > 1$ subsets, leading to m i.i.d. copies of the basic estimator. These copies are obtained by dividing the unit interval into m equal segments; the jth copy of the basic estimator is based on all of the hashed values that lie in the jth segment, after shifting and scaling the segment (and the points therein) into a copy of the unit interval. (Note that for a fixed synopsis size k, each copy of the basic estimator is based on approximately k/m observations.) Each copy of the basic estimator is then subjected to a nonlinear transformation g, and multiplied by a correction factor c. The function g is chosen to "stabilize" the estimator, and the constant c is chosen to ensure that the estimator is asymptotically unbiased as k becomes large. Finally, the i.i.d. copies of the transformed estimators are averaged together. The motivation behind the transformation g is to avoid the instability problem, discussed previously, that arises when $k = 1$. In Section 4.1, we show that our proposed estimator is unbiased for any values of D and k, while being less cumbersome to compute. Moreover, when $D \gg k \gg 0$, our estimator has approximately the minimal possible variance (see Section 4.2), and hence is at least as statistically efficient as any estimator in [18].

2.3 Estimates for Compound Partitions

To our knowledge, the only prior discussion of how to construct DV-related estimates for compound partitions is found in [8]. DV estimation for the intersection of partitions A and B is not computed directly. Instead, the Jaccard distance[3] $\rho = D(A \cap B)/D(A \cup B)$ (called the "resemblance" in [8]) is estimated first and then, using the estimator $\hat{\rho}$, the number of values in the intersection is estimated as

$$\hat{D}(A \cap B) = \frac{\hat{\rho}}{\hat{\rho}+1}\big(D(A)+D(B)\big). \tag{1}$$

The quantities $D(A)$ and $D(B)$ are computed exactly, by means of GROUP BY queries; our proposed estimators avoid the need to compute or estimate these quantities. There is no discussion in [8] of how to handle any set operations other than the intersection of two partitions. If one uses the principle of inclusion/exclusion to handle other set operations, the resulting estimation procedure will not scale well as the number of operations increases. Our new techniques handle arbitrarily complex combinations of set operations (multiset unions, intersections, and differences) in an efficient manner.

3. THE KMV SYNOPSIS

As indicated above, we will use estimators that are closely related to the basic estimator $\hat{D}_k^{\mathrm{BE}}$ (see Section 4). This relationship immediately implies a choice of synopsis for a partition A: we first hash each value in $\mathscr{D}(A)$ using a hash function h with domain $\{0,1,\ldots,M\}$, and then we record the k smallest of the hashed values. We call this synopsis a *KMV synopsis* (for k minimum values). The KMV synopsis can be viewed as originating in [3], but there is no discussion in [3] about implementing, constructing, or combining such synopses.

As discussed previously, we need to have $M = O(D^2)$ to avoid collisions. Thus each of the k hashed values requires $O(\log M) = O(\log D)$ bits of storage, and the required size of the KMV synopsis is $O(k \log D)$.

[3]Here $D(S)$ denotes the number of distinct values in the set S.

Algorithm 1 (KMV Computation)

```
 1: h: hash function from domain of dataset to {0,1,...,M}
 2: L: list of k smallest hashed values seen so far
 3: maxVal(L): returns the largest value in L
 4:
 5: for each item x in the dataset do
 6:    v = h(x)
 7:    if v ∉ L then
 8:       if |L| < k then
 9:          insert v into L
10:       else if v < maxVal(L) then
11:          insert v into L
12:          remove largest element of L
13:       end if
14:    end if
15: end for
```

A KMV synopsis can be computed from a single scan of the data partition, using Algorithm 1. The algorithm uses a sorted list of k hashed values, which can be implemented using, e.g., a priority queue. The membership check in line 7 avoids unnecessary processing of duplicate values in the input data partition, and can be implemented using a temporary hash table that is discarded after the synopsis has been built.

Assuming that the scan order of the items in a partition is independent of the hashed item values, and using reasoning similar to [17], we obtain the following result.

THEOREM 1. *The expected cost to construct a KMV synopsis of size k from a partition A comprising N data items having D distinct values is $O(N + k \log k \log D)$*

PROOF. The hashing step and membership check in lines 6 and 7 incur a cost of $O(1)$ for each of the N items in A, for a total cost of $O(N)$. To compute the expected cost of executing the remaining steps of the algorithm, observe that the first k DVs encountered are inserted into the priority queue (line 9), and each such insertion has a cost of at most $O(\log k)$, for an overall cost of $O(k \log k)$. Each subsequent new DV encountered will incur an $O(\log k)$ cost if it is inserted (line 11), or an $O(1)$ cost otherwise. (Note that a given DV will be inserted at most once, at the time it is first encountered, regardless of the number of times that it appears in A.) The ith new DV encountered is inserted only if its normalized hash value U_i is less than M_i, the largest normalized hash value currently in the synopsis. Because points are placed uniformly, we have $P\{U_i < M_i \mid M_i\} = M_i$, so that, by the law of total expectation,

$$P\{U_i < M_i\} = E\big[P\{U_i < M_i \mid M_i\}\big] = E[M_i] \approx k/i.$$

Thus, writing H_D for the Dth harmonic number, the expected cost for handling the remaining $D-k$ distinct values is

$$\begin{aligned} E[\mathrm{Cost}] &\approx \sum_{i=k+1}^{D} \big[(k/i)O(\log k) + \big(1-(k/i)\big)O(1)\big] \\ &< \sum_{i=1}^{D} [(k/i)O(\log k)] + O(D) = O(D) + O(k \log k) \sum_{i=1}^{D} (1/i) \\ &= O(D) + O(k \log k) H_D = O(D + k \log k \log D), \end{aligned}$$

and the overall expected cost is $O(N + D + k \log k + k \log k \log D) = O(N + k \log k \log D)$. □

The foregoing construction cost compares favorably to the $O(kN)$ cost for the Bellman synopsis. Moreover, when D is small, the

KMV synopsis contains more information in a statistical sense than the Bellman synopsis, since the former synopsis essentially samples distinct values without replacement, whereas the latter synopsis samples distinct values with replacement. The cost for the KMV synopsis is comparable to that of the sample-counting synopsis [17]. Indeed, the sample-counting synopsis is very similar to KMV, except that the size is a random variable K whose value ranges between roughly $k/2$ and k. Thus the KMV synopsis contains more statistical information for a given space requirement, and yields DV estimators that are statistically more stable.

We show in Section 5 that the KMV synopsis gracefully handles a variety of set operations, but needs to be augmented with counters to handle multiset differences. This augmentation has a negligible effect on the construction cost, and results in a desirable "closure" property that permits efficient estimation.

4. NEW DV ESTIMATORS

As discussed previously, the basic estimator $\hat{D}_k^{\mathrm{BE}}$ is biased upwards for the true number of distinct values D. Inspired by results in [7], we consider the estimator

$$\hat{D}_k^{\mathrm{UB}} = (k-1)/U_{(k)} \tag{2}$$

and show that $\hat{D}_k^{\mathrm{UB}}$ is unbiased. The $\hat{D}_k^{\mathrm{UB}}$ estimator forms the basis for the extended DV estimators, discussed in Section 5, used to estimate the number of DVs in a compound partition. Another standard estimation approach is the method of maximum likelihood — we therefore also develop a maximum-likelihood estimator (MLE) for our problem. Finally, we analyze $\hat{D}_k^{\mathrm{UB}}$ as D becomes large, and show that our results coincide asymptotically with those in [7], yielding convenient error bounds. This "large D" analysis facilitates the choice of k when initially designing a synopsis warehouse. Henceforth, we assume without further comment that $D > k$; if $D \leq k$, then we can easily detect this situation and compute the exact value of D from the synopsis.

4.1 Analysis of the Unbiased Estimator

Let $U_1, U_2, \ldots, U_D$ be the normalized hash values of the distinct items in the dataset; we model these values as a sequence of independent and identically distributed (i.i.d.) random variables from the uniform$[0,1]$ distribution — see the discussion in Section 2.2. Denote by $U_{(k)}$ the kth smallest of $U_1, U_2, \ldots, U_D$, that is, $U_{(k)}$ is the kth order statistic. Results from the theory of order statistics [9, Sec. 2.1] imply that $U_{(k)}$ follows the beta distribution with parameters k and $D-k+1$. That is, the probability density function (pdf) of $U_{(k)}$ is given by

$$f_{k,D}(t) = t^{k-1}(1-t)^{D-k}/B(k, D-k+1), \tag{3}$$

where $B(a,b) = \int_0^1 t^{a-1}(1-t)^{b-1}\,dt$ denotes the beta function [23]. Moreover,

$$P\{U_{(k)} \leq x\} = \int_0^x f_{k,D}(t)dt = I_x(k, D-k+1) \tag{4}$$

$$= \sum_{i=k}^{D} \binom{D}{i} x^i (1-x)^{D-i} \tag{5}$$

where $I_x(a,b)$ is the regularized incomplete beta function and is defined for all real a and b. An efficient algorithm for the computation of the most significant digits of $I_x(a,b)$ is given in [10]. Observe that the sum in (5) is simply the probability that at least k of the U_i values are smaller than x.

To facilitate the analysis of $\hat{D}_k^{\mathrm{UB}}$, we first derive the moments of $1/U_{(k)}$. For $k > r \geq 0$, we have

$$E[U_{(k)}^{-r}] = \int_0^1 \frac{f_{k,D}(t)}{t^r}dt = \frac{B(k-r, D-k+1)}{B(k, D-k+1)}.$$

If r is an integer, we can exploit the identity $B(a,b) = a^{-1}\binom{a+b-1}{a}^{-1}$ to obtain

$$E[U_{(k)}^{-r}] = D^{\underline{r}}/(k-1)^{\underline{r}}, \tag{6}$$

where $a^{\underline{b}}$ denotes the falling power $a(a-1)\cdots(a-b+1)$. Regarding $\hat{D}_k^{\mathrm{UB}}$, we find that

$$E[\hat{D}_k^{\mathrm{UB}}] = E[(k-1)/U_{(k)}] = (k-1)E[U_{(k)}^{-1}] = D,$$

so that $\hat{D}_k^{\mathrm{UB}}$ is indeed unbiased for D, and

$$\mathrm{Var}[\hat{D}_k^{\mathrm{UB}}] = (k-1)^2 E[U_{(k)}^{-2}] - (k-1)^2 E[U_{(k)}^{-1}]^2 = \frac{D(D-k+1)}{k-2}.$$

Because $\hat{D}_k^{\mathrm{UB}}$ is unbiased, its mean-squared error (MSE) is equal to its variance.

For comparison, note that, by (6), $E[\hat{D}_k^{\mathrm{BE}}] = kD/(k-1)$ and

$$\mathrm{MSE}[\hat{D}_k^{\mathrm{BE}}] = \left(\frac{k}{k-1}\right)^2 \mathrm{MSE}[\hat{D}_k^{\mathrm{UB}}] + \left(\frac{D}{k-1}\right)^2.$$

Thus, as discussed earlier, $\hat{D}_k^{\mathrm{BE}}$ is biased high for D, and has infinite mean when $k = 1$, as observed in [16]. Moreover, it can be seen that $\hat{D}_k^{\mathrm{UB}}$ has lower MSE than $\hat{D}_k^{\mathrm{BE}}$.

We now provide probabilistic (relative) error bounds for the estimator $\hat{D}_k^{\mathrm{UB}}$. Specifically, given $0 < \delta < 1$, we give a value of ε such that $\hat{D}_k^{\mathrm{UB}}$ lies in the interval $[(1-\varepsilon)D, (1+\varepsilon)D]$ with probability δ.

THEOREM 2. *For $0 < \varepsilon < 1$ and $k \geq 1$,*

$$P\left\{\frac{|\hat{D}_k^{UB} - D|}{D} \leq \varepsilon\right\} = I_{u(D,k,\varepsilon)}(k, D-k+1) - I_{l(D,k,\varepsilon)}(k, D-k+1), \tag{7}$$

where

$$u(D,k,\varepsilon) = \frac{k-1}{(1-\varepsilon)D} \quad \textit{and} \quad l(D,k,\varepsilon) = \frac{k-1}{(1+\varepsilon)D}. \tag{8}$$

PROOF. The desired result follows directly from (4) after using (2) to obtain

$$P\left\{\frac{|\hat{D}_k^{\mathrm{UB}} - D|}{D} \leq \varepsilon\right\} = P\left\{(1-\varepsilon)D \leq \hat{D}_k^{\mathrm{UB}} \leq (1+\varepsilon)D\right\} = P\left\{\frac{k-1}{(1+\varepsilon)D} \leq U_{(k)} \leq \frac{k-1}{(1-\varepsilon)D}\right\}.$$

□

Error bounds for a given value of δ can be obtained by equating the right side of (7) to δ and solving for ε using a root-finding algorithm. Although useful for theoretical analysis, these bounds cannot be used directly in practice, since they involve the unknown parameter D. Using a standard approach from statistics, practical approximate error bounds based on the observed value of $U_{(k)}$ can be obtained by replacing D with $\hat{D}_k^{\mathrm{UB}}$ in the above formulas.

Figure 1 displays the error bound ε as a function of δ for $D = 1{,}000{,}000$ and several values of k. The dashed and solid curves represent the confidence intervals for the basic estimator $\hat{D}_k^{\mathrm{BE}}$ and the unbiased estimator $\hat{D}_k^{\mathrm{UB}}$, respectively. As expected, $\hat{D}_k^{\mathrm{UB}}$ is

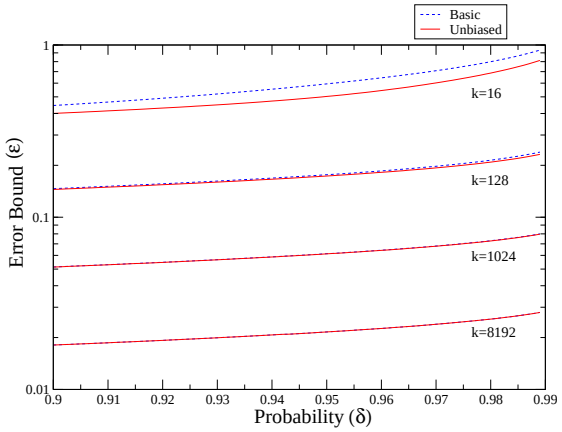


Figure 1: Error bounds for $D = 1,000,000$

superior to $\hat{D}_k^{\mathrm{BE}}$ when k is small; for example, when $k = 16$ and $\delta = 0.95$, use of the unbiased estimator yields close to a 20% reduction in ε. As k increases, $k-1 \approx k$ and both estimators perform similarly. Note that the error bound is very stable even for large values of D. E.g., it follows the results of Section 4.3 that, for $\delta = 0.95$ and $k = 1024$, the upper bound on ε as $D \to \infty$ is $\varepsilon \approx 0.06127$, whereas we observe a value of $\varepsilon \approx 0.06124$ for $D = 1,000,000$.

To further examine the behavior of the unbiased estimator, we derive the expected value of the absolute ratio error (ARE), where the ARE is defined as $|\hat{D}_k^{\mathrm{UB}} - D|/D$. The expected ARE is a common metric for comparing the performance of statistical estimators.

THEOREM 3. *The expected ARE of $\hat{D}_k^{UB}$ is given by*

$$E\left[\frac{|\hat{D}_k^{UB} - D|}{D}\right] = 2\binom{D}{k-1}\left(\frac{k-1}{D}\right)^{k-1}\left(1-\frac{k-1}{D}\right)^{D-k+2}$$

PROOF. We have

$$\begin{aligned} E[\mathrm{ARE}] &= \frac{1}{D}\int_0^1 \left|\frac{k-1}{t} - D\right| f_{k,D}(t)\,dt \\ &= \frac{1}{D}\int_0^{(k-1)/D} \left(\frac{k-1}{t} - D\right) f_{k,D}(t)\,dt \\ &\qquad + \frac{1}{D}\int_{(k-1)/D}^1 \left(D - \frac{k-1}{t}\right) f_{k,D}(t)\,dt \\ &= 2I_{(k-1)/D}(k-1, D-k+1) - 2I_{(k-1)/D}(k, D-k+1), \end{aligned}$$

where the last equality is obtained after expanding the integrals and applying the identity $(k-1)(tD)^{-1} f_{k,D}(t) = f_{k-1,D-1}(t)$. The desired result now follows after applying (5). □

4.2 Maximum Likelihood Estimator

The classical statistical approach to estimating unknown parameters is the method of maximum likelihood [30, Sec. 4.2]. We apply this approach by casting our DV-estimation problem as a parameter estimation problem. Specifically, recall that $U_{(k)}$ has the pdf $f_{k,D}$ given in (3). The MLE estimate of D is defined as the value $\hat{D}$ that maximizes the likelihood $L(D; U_{(k)})$ of the observation $U_{(k)}$, defined as $L(D; U_{(k)}) = f_{k,D}(U_{(k)})$. We find this maximizing value by solving the equation $L'(D; U_{(k)}) = 0$, where the prime denotes differentiation with respect to D. We have

$$L'(D; U_{(k)}) = \ln(1 - U_{(k)}) - \Psi(D-k+1) + \Psi(D+1),$$

where Ψ denotes the digamma function. If x is sufficiently large, then $\Psi(x) \approx \ln(x-1) + \gamma$, where γ denotes Euler's constant. Applying this approximation, we obtain

$$\hat{D}_k^{\mathrm{MLE}} \approx \frac{k}{U_{(k)}},$$

so that the MLE estimator roughly resembles the basic estimator $\hat{D}_k^{\mathrm{BE}}$ provided that $D \gg k$. In fact, our experiments indicated that $\hat{D}_k^{\mathrm{MLE}}$ and $\hat{D}_k^{\mathrm{BE}}$ are indistinguishable from a practical point of view. It follows that $\hat{D}_k^{\mathrm{MLE}}$ is asymptotically equivalent to $\hat{D}_k^{\mathrm{UB}}$ as $k \to \infty$. A basic result for MLE estimators [30, Sec. 4.2.2] implies that, for $D \gg k \gg 0$, the estimator $\hat{D}_k^{\mathrm{UB}}$ has, to a good approximation, the minimal possible variance for any estimator of D.

4.3 Analysis with Many DVs

Our asymptotic analysis for large D rests on Theorem 4 below. As before, for a sequence $U_1, U_2, \ldots, U_n$ of i.i.d. uniform[0,1] random variables, denote by $U_{(1)} < U_{(2)} < \cdots < U_{(n)}$ the order statistics of the sequence, and define the *spacings* $W_1 = U_{(1)}, W_2 = U_{(2)} - U_{(1)}, \ldots, W_n = U_{(n)} - U_{(n-1)}$. Write $X_n \Rightarrow X$ if and only if the sequence $\{X_n : n \geq 1\}$ *converges in distribution* to X, that is,

$$\lim_{n\to\infty} P\{X_n \leq x\} = P\{X \leq x\}$$

for all x at which the function $F(x) = P\{X \leq x\}$ is continuous. We say that a random variable Y has an *exponential distribution* with rate parameter λ, denoted $\mathrm{Exp}(\lambda)$, if

$$P\{Y \leq y\} = \begin{cases} 1 - e^{-\lambda y} & \text{if } y \geq 0; \\ 0 & \text{if } y < 0. \end{cases}$$

THEOREM 4. *Let $U_1, U_2, \ldots, U_n$ be a sequence of i.i.d. uniform[0,1] random variables, define $W_1, W_2, \ldots, W_n$ as above, and fix $k \geq 1$. Then*

$$\lim_{n\to\infty} (nW_1, nW_2, \ldots, nW_k) \Rightarrow (Y_1, Y_2, \ldots, Y_k),$$

where $Y_1, Y_2, \ldots, Y_k$ are i.i.d. with each Y_i having an Exp(1) distribution.

PROOF. Let $\{Y_n : n \geq 1\}$ be an infinite sequence of i.i.d. Exp(1) random variables, and write $W = (W_1, W_2, \ldots, W_k)$ and $Y = (Y_1, Y_2, \ldots, Y_k)$. It follows from a well known result for order statistics — see [9, p. 134] or [24, p. 105–107] — that, for any fixed $n \geq k$, we have $W \stackrel{\mathscr{D}}{=} Y/S_n$, where $S_n = Y_1 + Y_2 + \cdots + Y_{n+1}$ and $\stackrel{\mathscr{D}}{=}$ denotes equality in distribution. Since $nW \stackrel{\mathscr{D}}{=} Y/(S_n/n)$ and $\lim_{n\to\infty} S_n/n = E[Y_1] = 1$ with probability 1 by the strong law of large numbers, Slutsky's theorem [30, p. 19] implies that $Y/(S_n/n) \Rightarrow Y/1 = Y$, and hence $nW \Rightarrow Y$, as $n \to \infty$. □

Thus, for large D and fixed $k \leq D$, the scaled spacings $DW_1, DW_2, \ldots, DW_k$ are approximately i.i.d. Exp(1). Since

$$P\{W_i \leq x\} = P\{DW_i \leq Dx\} \approx 1 - e^{-Dx}$$

for $1 \leq i \leq n$, it follows that the unscaled spacings $W_1, W_2, \ldots, W_k$ are approximately i.i.d. Exp(D), so that $U_{(k)}$ is distributed approximately as the sum of k i.i.d. Exp(D) random variables. This exponential scenario is precisely the one analyzed in [7], in the context of estimating the size of a transitive closure. It follows from [7] that

$$\begin{aligned} P\left\{\frac{|\hat{D}_k^{\mathrm{UB}} - D|}{D} \leq \varepsilon\right\} &\approx e^{-\frac{k-1}{1+\varepsilon}}\left(1 + \sum_{i=1}^{k-1}\frac{(k-1)^i}{(1+\varepsilon)^i i!}\right) \\ &\quad - e^{-\frac{k-1}{1-\varepsilon}}\left(1 + \sum_{i=1}^{k-1}\frac{(k-1)^i}{(1-\varepsilon)^i i!}\right) \end{aligned}$$

and

$$E\Big[\frac{|\hat{D}_k^{\mathrm{UB}} - D|}{D}\Big] \approx \frac{2(k-1)^{k-2}}{(k-2)!e^{k-1}} \approx \sqrt{\frac{2}{\pi(k-2)}}.$$

As might be expected, the above formulas can also be obtained by letting $D \to \infty$ in the corresponding formulas from Section 4.1. Though slightly conservative, the asymptotic error bounds have the advantageous property that, unlike the exact bounds, they do not involve the unknown quantity D. Thus, given desired values of ε and δ, they can be used to help determine target synopsis sizes when initially designing a synopsis warehouse.

5. COMBINING SYNOPSES

The discussion up until now has focused on creating and using a synopsis to estimate the number of DVs in a single base partition. We now focus on DV estimation for a *compound* partition, i.e., a partition that is created from a set of base partitions using the multiset operations of intersection, union, and difference. When a compound partition G has been created from base partitions using only union and intersection, or when each base partition contains no duplicates, we can estimate the number of DVs in G directly from the KMV synopses for the base partitions. To handle multiset difference, however, we need to augment our KMV synopses with counters; we show that the resulting AKMV synopses are "closed" under multiset operations on the parent partitions. The closure property implies that if E and F are compound partitions and G is obtained from E and F via a set operation, then we can compute an AKMV synopsis for G from the corresponding AKMV synopses for E and F, and unbiasedly estimate the number of DVs in G from this resulting synopsis. This procedure avoids the need to access the synopsis for each of the base partitions that were used to create E and F. The AKMV synopsis can also handle deletions of individual items from the warehouse. As discussed below, the actual DV estimators that we use for compound partitions are, in general, extensions of the simple $\hat{D}_k^{\mathrm{UB}}$ estimator developed in Section 4.

We assume throughout that all synopses are created using the same hash function $h\colon \mathscr{D} \mapsto \{0,1,\ldots,M\}$, where $\mathscr{D}$ denotes the domain of the data values that appear in the partitions and $M = O(|\mathscr{D}|^2)$ as discussed previously. We denote ordinary set-union, set-intersection, and set-difference operators by $\{\cup,\cap,\setminus\}$ and the corresponding multiset operators by $\{\cup_{\mathrm{m}},\cap_{\mathrm{m}},\setminus_{\mathrm{m}}\}$.[4]

5.1 Union Operations

Consider two partitions A and B, along with their KMV synopses L_A and L_B of sizes k_A and k_B, respectively. (For purposes of this discussion, we view the synopses as sets of hashed values.) We wish to estimate $D_\cup = |\mathscr{D}(A \cup_{\mathrm{m}} B)|$, where, as before, $\mathscr{D}(S)$ denotes the set of DVs in multiset S. Observe that $\mathscr{D}(A \cup_{\mathrm{m}} B) = \mathscr{D}(A) \cup \mathscr{D}(B)$, so that $D_\cup$ can also be interpreted as $|\mathscr{D}(A) \cup \mathscr{D}(B)|$.

Define $L_A \oplus L_B$ to be the set comprising the k smallest values in $L_A \cup L_B$, where $k = \min(k_A, k_B)$. Observe that the $\oplus$ operator is symmetric and associative.

THEOREM 5. *The set $L = L_A \oplus L_B$ is the size-k KMV synopsis of $A \cup_m B$, where $k = \min(k_A, k_B)$.*

PROOF. For a multiset S with $\mathscr{D}(S) \subseteq \mathscr{D}$, write $h(S) = \{h(v)\colon v \in \mathscr{D}(S)\}$, and denote by G the set of k smallest values in $h(A \cup_{\mathrm{m}} B)$. Observe that G contains the k' smallest values in $h(A)$ for some $k' \le k$, and these k' values therefore are also contained in L_A, i.e., $G \cap h(A) \subseteq L_A$. Similarly, $G \cap h(B) \subseteq L_B$, so that $G \subseteq L_A \cup L_B$. For any $h \in (L_A \cup L_b) \setminus G$, we have that $h > \max_{h' \in G} h'$ by definition of G, because $h \in h(A \cup B)$. Thus G in fact comprises the k smallest values in $L_A \cup L_B$, so that $L = G$. Now observe that, by definition, G is precisely the size-k KMV synopsis of $A \cup_{\mathrm{m}} B$. □

Thus we can immediately apply the results of the Section 4 to estimate $D_\cup$ by $\hat{D}_\cup = (k-1)/U_{(k)}$ where $\hat{D}_\cup$ is computed from the size-k KMV synopsis $L = L_A \oplus L_B$. This result extends immediately to multiple partitions: the number of DVs in $A_1 \cup_{\mathrm{m}} A_2 \cup_{\mathrm{m}} \cdots \cup_{\mathrm{m}} A_n$ can be estimated from $L = L_{A_1} \oplus L_{A_2} \oplus \cdots \oplus L_{A_n}$.

5.2 Intersection Operations

As before, consider two partitions A and B, with corresponding KMV synopses L_A and L_B of sizes k_A and k_B, respectively. Our goal now is to estimate $D_\cap = |\mathscr{D}(A \cap_{\mathrm{m}} B)| = |\mathscr{D}(A) \cap \mathscr{D}(B)|$. Set $L = L_A \oplus L_B$ and write $L = \{h(v_1), h(v_2), \ldots, h(v_k)\}$, where $k = \min(k_A, k_B)$ as before and each distinct value v_i is an element of $\mathscr{D}(A) \cup \mathscr{D}(B)$. Also write $V_L = \{v_1, v_2, \ldots, v_k\}$, and set

$$K_\cap = |\{v \in V_L\colon v \in \mathscr{D}(A) \cap \mathscr{D}(B)\}|.$$

LEMMA 1. *For each $v \in V_L$, we have $v \in \mathscr{D}(A)$ (resp., $v \in \mathscr{D}(B)$) if and only if $h(v) \in L_A$ (resp., $h(v) \in L_B$).*

PROOF. Let $v \in V_L$, so that $h(v)$ is among the k smallest values of $h(A \cup_{\mathrm{m}} B)$. Then $h(v)$ is among the k smallest values of $h(A)$ if $v \in \mathscr{D}(A)$, so that $h(v) \in L_A$ if $v \in \mathscr{D}(A)$. Conversely, if $h(v) \in L_A$, then it follows immediately from our running assumption of no hash collisions that $v \in \mathscr{D}(A)$. An analogous argument holds for partition B. □

Lemma 1 implies that $v \in \mathscr{D}(A) \cap \mathscr{D}(B)$ if and only if $h(v) \in L_A \cap L_B$, and we can compute $K_\cap$ from L_A and L_B alone. Observe that, under our random hashing model, V_L can be viewed as a uniform random sample of size k drawn from $\mathscr{D}(A \cup_{\mathrm{m}} B)$. The quantity $K_\cap$ is a random variable that represents the number of elements in V_L that also belong to the set $\mathscr{D}(A \cap_{\mathrm{m}} B)$. It follows that $K_\cap$ has a hypergeometric distribution: setting $D_\cap = |\mathscr{D}(A \cap_{\mathrm{m}} B)|$ and $D_\cup = |\mathscr{D}(A \cup_{\mathrm{m}} B)|$ as before, we have

$$P\{K_\cap = j\} = \binom{D_\cap}{j}\binom{D_\cup - D_\cap}{k-j} \Big/ \binom{D_\cup}{k}. \tag{9}$$

We now use $K_\cap$ to estimate $D_\cap$. From Section 5.1, we know that $\hat{D}_\cup = (k-1)/U_{(k)}$ is an unbiased estimator of $D_\cup$; we would like to "correct" this estimator via multiplication by the Jaccard distance $\rho = D_\cap / D_\cup$. We do not know ρ, but a reasonable estimate is

$$\hat{\rho} = K_\cap / k, \tag{10}$$

the fraction of sample elements in $V_L \subseteq \mathscr{D}(A \cup B)$ that belong to $\mathscr{D}(A \cap B)$. This leads to our proposed estimator

$$\hat{D}_\cap = \frac{K_\cap}{k}\Big(\frac{k-1}{U_{(k)}}\Big).$$

We now establish some basic properties of the estimator. For $n \ge k \ge 1$, set

$$\Delta(n,k,\varepsilon) = I_{u(n,k,\varepsilon)}(k, n-k+1) - I_{l(n,k,\varepsilon)}(k, n-k+1),$$

where $I_x(a,b)$ is the regularized incomplete beta function, and $u(n,k,\varepsilon)$ and $l(n,k,\varepsilon)$ are defined as in (8). Take $\Delta(\infty,k,\varepsilon) = 0$.

[4] Recall that if $n_A(v)$ and $n_B(v)$ denote the multiplicities of value v in multisets A and B, respectively, then the multiplicity of v in $A \cup_{\mathrm{m}} B$, $A \cap_{\mathrm{m}} B$, and $A \setminus_{\mathrm{m}} B$ are given respectively by $n_A(v) + n_B(v)$, $\min\big(n_A(v), n_B(v)\big)$, and $\max\big(n_A(v) - n_B(v), 0\big)$.

Denote by

$$H(j;N,M,n) = \binom{M}{j}\binom{N-M}{n-j}\Big/\binom{N}{n}$$

the hypergeometric probability distribution function.

THEOREM 6. *When based on a combined synopsis of size k, the estimator $\hat{D}_\cap$ satisfies $E[\hat{D}_\cap] = D_\cap$ if $k > 1$,*

$$\mathrm{Var}[\hat{D}_\cap] = \frac{D_\cap(kD_\cup - k^2 - D_\cup + k + D_\cap)}{k(k-2)}$$

if $k > 2$, and, if $D_\cap > 0$, $\varepsilon \in (0,1)$, and $k \geq 1$,

$$P\left\{ \frac{|\hat{D}_\cap - D_\cap|}{D_\cap} \leq \varepsilon \;\middle|\; K_\cap = j \right\} = \Delta(kD_\cap/j, k, \varepsilon) \tag{11}$$

for $0 \leq j \leq \min(k, D_\cap)$, and

$$\begin{aligned} &P\left\{ \frac{|\hat{D}_\cap - D_\cap|}{D_\cap} \leq \varepsilon \right\} \\ &\quad = \sum_{j=0}^{\min(k,D_\cap)} \Delta(kD_\cap/j, k, \varepsilon) H(j; D_\cup, D_\cap, k). \end{aligned} \tag{12}$$

PROOF. A can be seen from (9), the distribution of $K_\cap$ does not depend on the hash values $\{h(v): v \in \mathscr{D}(A \cup B)\}$. It follows that the random variables $K_\cap$ and $U_{(k)}$ are statistically independent, as are $\hat{\rho}$ and $U_{(k)}$, where $\hat{\rho} = K_\cap/k$ as above. By (9) and standard properties of the hypergeometric distribution, we have

$$E[K_\cap] = k\frac{D_\cap}{D_\cup} \tag{13}$$

and

$$\mathrm{Var}[K_\cap] = \frac{D_\cap(D_\cup - D_\cap)k(D_\cup - k)}{D_\cup^2(D_\cup - 1)}. \tag{14}$$

It follows from (13) that $E[\hat{\rho}] = \rho$. Using independence and the unbiasedness of $\hat{D}_\cup$, we find that

$$E[\hat{D}_\cap] = E[\hat{\rho}\hat{D}_\cup] = E[\hat{\rho}]E[\hat{D}_\cup] = \rho D_\cup = D_\cap.$$

The formula for $\mathrm{Var}[\hat{D}_\cap]$ follows from (6), (13), and (14), after some straightforward algebra. To obtain the relation in (11), use the fact that K and $\hat{D}_\cup$ are independent, and write

$$\begin{aligned} P\left\{ \frac{|\hat{D}_\cap - D_\cap|}{D_\cap} \leq \varepsilon \;\middle|\; K_\cap = j \right\} &= P\left\{ \frac{|(j/k)\hat{D}_\cup - D_\cap|}{D_\cap} \leq \varepsilon \right\} \\ &= P\left\{ \frac{|\hat{D}_\cup - D^*|}{D^*} \leq \varepsilon \right\}, \end{aligned}$$

where $D^* = (k/j)D_\cap$. The desired result then follows by mimicking the proof of Theorem 2. The final relation in (12) follows from (11) by unconditioning on $K_\cap$ and using (9). □

Thus $\hat{D}_\cap$ is unbiased for $D_\cap$. It also follows from the proof that the estimator $\hat{\rho}$ is unbiased for the Jaccard distance ρ. Using (12), we can compute exact confidence bounds numerically, analogously to the single-partition case. To obtain practical approximate bounds based on observation of $K_\cap$ and $U_{(k)}$, use the representation in (11), but replace $D_\cap$ by $\hat{D}_\cap$.

Interestingly, $\hat{D}_\cap$ can be viewed as being in the spirit of the Bellman estimator (1). Specifically, the quantity $\big(D(A) + D(B)\big)/(\hat{\rho} + 1)$ in (1), after some algebra, can be viewed as an estimator of $D_\cup$, so the overall estimator can be viewed as an estimator of ρ times an estimator of $D_\cup$. We emphasize, however, that $\hat{D}_\cap$ is based on very different estimators of ρ and $D_\cup$.

5.3 Compound Partitions with No Duplicates

We now consider the more general problem of estimating D_E, the number of DVs in E, where E is a compound partition created from $n \geq 2$ base partitions $A_1, A_2, \ldots, A_n$, all of which are ordinary sets, using the ordinary union, intersection, and set-difference operators. Some examples are $E = A_1 \setminus A_2$ and $E = \big((A_1 \cup A_2) \cap (A_3 \cup A_4)\big) \setminus A_5$. Note that E is also an ordinary set, so that $E = \mathscr{D}(E)$, and estimating the number of DVs in E is equivalent to estimating the cardinality of E. Our key observation is that the foregoing analysis for the intersection operator applies essentially unchanged in the current setting. Specifically, we form the synopsis $L = L_{A_1} \oplus L_{A_2} \oplus \cdots \oplus L_{A_n}$ of size $k = \min(k_{A_1}, k_{A_2}, \ldots, k_{A_n})$. Define V_L as before and set

$$K_E = |\{ v \in V_L : v \in \mathscr{D}(E) \}|. \tag{15}$$

By a trivial extension of Lemma 1, we can compute K_E from L_{A_1}, $L_{A_2}, \ldots, L_{A_n}$ alone. The estimator

$$\hat{D}_E = \frac{K_E}{k}\left(\frac{k-1}{U_{(k)}}\right) \tag{16}$$

is unbiased for D_E, and the other properties of $\hat{D}_E$ are derived exactly as for $\hat{D}_\cap$. For the special case $E = A_1 \cup A_2$, we have $K_E = k$ with probability 1, and $\hat{D}_E$ reduces to $\hat{D}_\cup$.

Note that the approach of this section can also be applied when the base partitions are multisets, provided that the only operations used are $\cup_\mathrm{m}$ and $\cap_\mathrm{m}$. To see this, observe that the computation in (15) effectively replaces each A_i with $\mathscr{D}(A_i)$; as indicated previously, D_E is unchanged by this transformation. Of course, we can also estimate D_E when E is any ordinary set expression involving the $\mathscr{D}(A_i)$ sets, even if the A_i's are themselves multisets.

5.4 Multiset Difference

Multiset difference is more complicated than union or intersection in that there are two possible, nonequivalent quantities to estimate: $D_-^* = |\mathscr{D}(A) \setminus \mathscr{D}(B)|$ or $D_- = |\mathscr{D}(A \setminus_\mathrm{m} B)|$. The quantity D_-^* can be estimated as in Section 5.3 by taking $E = \mathscr{D}(A) \setminus \mathscr{D}(B)$, and so we concentrate on the estimation of D_-.

This latter task requires that we augment our KMV synopsis. We define an augmented synopsis of a base partition A, which we call an *AKMV synopsis*, as $L_A^+ = (L_A, c_A)$, where $L_A = \{h(v_1), h(v_2), \ldots, h(v_k)\}$ is a KMV synopsis of size k, and $c_A = \{c_A(v_1), c_A(v_2), \ldots, c_A(v_k)\}$ is a set of k nonnegative counters. The quantity $c_A(v)$ is the multiplicity in A of the value v; this use of counters is in the spirit of [15, 16, 31]. The size of the AKMV synopsis is $O(k \log D + k \log M)$, where M is the maximum multiplicity of a value in the multiset A. Note that, if A is a set, then it suffices to maintain a bit vector rather than a vector of counts, so that the size of the synopsis is $O(k \log D)$, just as with an ordinary KMV synopsis. It is easy to modify Algorithm 1 to create and maintain counters via $O(1)$ operations. The modified algorithm retains the original algorithm complexity of $O(N + k \log k \log D)$.

We now proceed almost exactly as in Section 5.3, taking $E = A \setminus_\mathrm{m} B$ in (15). Observe that a value $v \in V_L$ is an element of K_E if and only if $h(v) \in L_A$ and either (i) $h(v) \notin L_B$ (which implies that $v \notin \mathscr{D}(B)$ by Lemma 1) or (ii) $h(v) \in L_B$ with $c_A(v) - c_B(v) > 0$. Thus we can determine K_E from L_A^+ and L_B^+ alone. This approach extends to any quantity of the form D_E, where E is a compound (multiset) partition created by applying the operations $\cup_\mathrm{m}$, $\cap_\mathrm{m}$, and $\setminus_\mathrm{m}$ to $n \geq 2$ base partitions $A_1, A_2, \ldots, A_n$, each of which may be a multiset. As with multiset difference, we use the count vectors to compute K_E; see the following section for another example of such a computation.

5.5 The Closure Property

In the previous section, we defined the AKMV synopsis of a base partition. We extend this definition to the AKMV synopsis $L_E^+ = (L_E, c_E)$ of a compound partition E by taking $L_E = L_{A_1} \oplus L_{A_2} \oplus \cdots \oplus L_{A_n}$, where $A_1, A_2, \ldots, A_n$ are the base partitions used to construct E, and $c_E(v)$ is the multiplicity of v in E.

With this definition, the collection of AKMV synopses over compound partitions is closed under multiset operations. For example, if we combine compound partitions E and F — having respective AKMV synopses $L_E^+ = (L_E, c_E)$ and $L_F^+ = (L_F, c_F)$ — to create $G = E \cap_{\mathrm{m}} F$, then the combined synopsis is $L_G^+ = (L_E \oplus L_F, c_G)$. Here $c_G(v) = \min\big(c_E(v), c_F(v)\big)$ for each $v \in V_{L_G}$, where we take $c_X(v) = 0$ if $h(v) \notin L_X$. The cases $G = E \cup_{\mathrm{m}} F$ and $G = E \setminus_{\mathrm{m}} F$ are handled similarly. Then K_E is computed as the number of non-zero elements in c_G, and the number of DVs in G is estimated as in (16).

5.6 Deletions

We now show how AKMV synopses can easily support deletions of individual items. Consider a partition A that receives a stream of transactions of the form $+v$ or $-v$, corresponding to the insertion or deletion, respectively, of value v.

A naive approach maintains two AKMV synopses: a synopsis L_i^+ for the multiset A_i of inserted items and a synopsis L_d^+ for the multiset A_d of deleted items. Computing the AKMV synopsis of the multiset difference $A_i \setminus_{\mathrm{m}} A_d$ yields the AKVM synopsis L_A^+ of the true multiset A. Because A_i and A_d are always growing with time, they can become significantly larger than the actual partition A, so that DV estimates based on (15) will be of low quality. Therefore, whenever the number of deletions causes the error bounds to become unacceptable, a fresh scan of the data can be used to produce a new pair of synopses L_i and L_d corresponding to $A_i = A$ and $A_d = \emptyset$.

This method does not actually require two synopses. We can simply maintain the counters in a single AKMV synopsis L by incrementing the counter at each insertion and decrementing at each deletion. If we retain synopsis entries having counter values equal to 0, we produce precisely the synopsis L described above.

We conjecture that the above procedure can be further improved as follows. Whenever an insertion transaction $+v'$ arrives and $v' < \mathrm{maxVal}(L)$, remove a 0-item (i.e., an item with a 0-valued counter) from L and insert the new value v', with corresponding counter value $c_A(v') = 1$. The victim 0-item can be chosen arbitrarily from among all 0-items in the synopsis, except that a 0-item with hash value equal to $\mathrm{maxVal}(L)$ should be removed only if it is the sole 0-item in the synopsis. If the synopsis contains no 0-items when the insertion transaction arrives, then remove the maximum item as usual and insert the new item. Thus, if the synopsis contains at least one 0-item with hash value less than $\mathrm{maxVal}(L)$, then $\mathrm{maxVal}(L)$ stays the same after the insertion; otherwise, $\mathrm{maxVal}(L)$ decreases. Using this process, L produces its best estimates when there are very few 0-items. This situation occurs whenever the current size of $\mathscr{D}(A)$ is roughly the maximum size that $\mathscr{D}(A)$ has ever attained. For example, when insertions significantly outnumber deletions, L will yield reasonably accurate estimates. If, however, A shrinks significantly, a fresh synopsis may be required to achieve the desired accuracy.

6. EXPERIMENTAL EVALUATION

We implemented the AKMV synopsis and the corresponding DV estimators, and applied our prototype to various real-world and synthetic datasets. First we examined the average ARE of the new estimators over synthetic data in order to establish a baseline; the synthetic datasets correspond to a scenario in which the hash function behaves *exactly* like a random number generator, in accordance with the assumptions underlying the derivation of our methodology. Then, using real data, we examined the impact on estimation accuracy of using different hash functions when creating the synopsis. We next compared the accuracy of the new estimators to that of the current best-of-breed estimators. Finally, we investigated the accuracy of our DV estimators on compound partitions of the form $A \cup_{\mathrm{m}} B$ and $A \cap_{\mathrm{m}} B$, as well as the accuracy of our Jaccard-distance estimator.

6.1 Experimental Setup

We evaluated our KMV prototype using three real-world datasets. The OPIC dataset contains product information for a large computer company. The BASEBALL dataset contains data about baseball players, teams, awards, hall-of-fame memberships, and both game and player statistics for the baseball championship in Australia. The RDW dataset was obtained from the data warehouse of a large financial company. Table 1 displays some summary characteristics of these datasets. All experiments were performed on a UNIX machine with one 2.8 GHz processor and 1GB of RAM. Unless stated otherwise, results are reported for experiments on the RDW dataset; the results for other datasets are similar.

6.2 Accuracy Comparison

We first compared the expected accuracy of the KMV-based estimators using a synthetic baseline dataset. Then we compared these estimators with the current best-of-breed DV estimators on real data, using various hashing methods.

6.2.1 Baseline Comparison on Synthetic Data

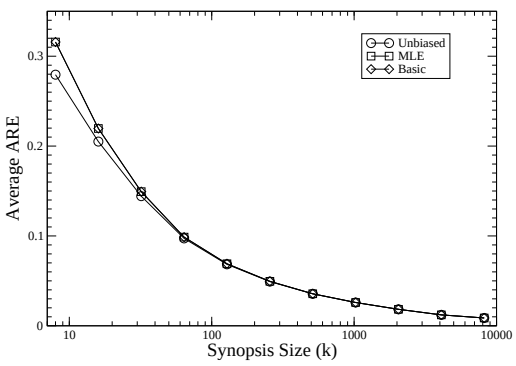
In order to compare the DV estimators while controlling for hashing effects, we used a high-quality pseudorandom number generator [26] to produce synthetic datasets, each with a specified number of distinct values. In effect, we generated the hashed values directly, according to a "truly" uniform[0,1] probability distribution. Since we derived our estimators under precisely this uniform-distribution assumption, the synthetic dataset provides a baseline for performance, allowing us, in subsequent experiments, to clearly see the effects on accuracy of using non-ideal hash functions.

We studied the behavior of the various estimators we have discussed: the basic estimator $\hat{D}_k^{\mathrm{BE}}$, the unbiased estimator $\hat{D}_k^{\mathrm{UB}}$ and the MLE estimator $\hat{D}_k^{\mathrm{MLE}}$. To gauge the accuracy of a given estimator $\hat{D}$ for a specified number D of distinct values, we generated 1000 datasets, each containing D distinct values, using a different pseudorandom number seed each time ensure independence between the datasets. For each estimator, we computed the ARE on each dataset and then averaged these 1000 ARE values.

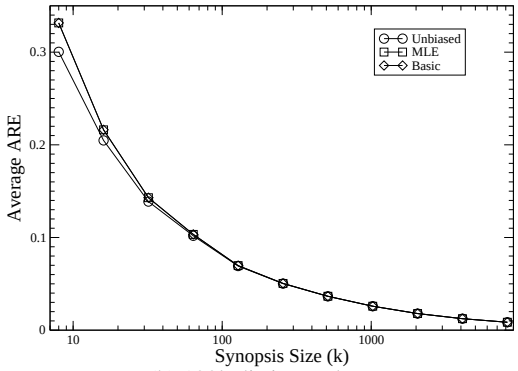
Figure 2 shows the average ARE as a function of the synopsis size k when the true numbers of distinct values are $D = 1$ million and $D = 100,000$, respectively. Not surprisingly, the primary factor that affects the accuracy is the size of the KMV synopsis, with relative estimation errors decreasing as k increases. As expected from theory, the relative estimation error is relatively insensitive to the true number of distinct values. Observe that the unbiased esti-

Dataset	#Tables	Total #Attributes	# Tuples
OPIC	106	1802	27,757,807
BASEBALL	12	192	262,432
RDW	24	504	2,661,506

Table 1: Real Dataset Characteristics



(a) 1 million distinct values



(b) 100k distinct values

Figure 2: Baseline: Accuracy vs. KMV Synopsis size

mator consistently provides the best accuracy for all synopsis sizes, and that the basic and MLE estimators are indistinguishable. For small synopses, the benefit of the unbiased estimator increases as the true number of distinct values increases. Since the unbiased estimator introduces no overhead with respect to the basic estimator, we recommend using it for all synopsis sizes.

6.2.2 Hashing Effect

The development in Section 4 assumes that the hash function can be viewed as a 1-to-1 mapping from the D distinct values in the dataset to a set of D uniform random numbers. Such a mapping can be constructed perfectly using $O(D)$ memory, but this memory requirement is infeasible for very large datasets. For practical purposes, we need to *approximate* such a mapping using a hash function that requires a small amount of memory (logarithmic in D).

In this section we study how the use of real-world hash functions effects the accuracy of our estimators. More specifically, we compute estimates from real data, using various hash functions, and compare the accuracy of these results to the accuracy of the baseline estimates from the previous section. Our accuracy measure is again the ARE, averaged over all of the datasets in the database. We used three different hashing methods:

AES The Advanced Encryption Standard (AES) hash function is a well established cipher function that has been studied extensively. For example, Hellekalek and Wegenkittl [21] showed that AES behaves empirically like a high-quality random-number generator when applied in an iterative fashion, making it a promising candidate for DV estimation (although we use AES in a slightly different fashion here than in [21]). Since AES is a cipher function, its output size is equal to the input size. In our implementation, we only used the most significant 32 bits of the output as the hash value.

FLH This hash function, due to Wegman,[5] typifies the sort of hash function used in current computer systems. The function is rather complicated, so we omit details here. Note that FLH is designed merely to avoid collisions, and so does not provide any guarantees on the distribution of its output.

GRM This Golden-Ratio Multiplicative (GRM) hash function is based on classical results of Knuth [25]. The method is based on the observation that multiplying each member of the sequence $1, 2, \ldots, n$ by the golden ratio, and keeping the fractional part of the results, yields real numbers that are very evenly spread over the unit interval.

[5]Personal communication.

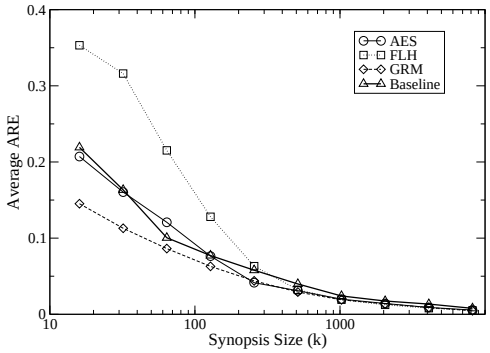


Figure 3: Hashing Effect on the RDW Dataset

Figures 3–5 display the average ARE of the unbiased KMV estimator as a function of synopsis size for the three hash functions described above, as well as for the ideal hash function implicit in the baseline scenario of Figure 2. The three figures correspond to the three real-world datasets described previously.

The FLH hash function is dominated by the other hash functions on all datasets. For the RDW and OPIC datasets, the GRM hashing function is clearly superior, even outperforming the ideal baseline in certain cases. The main reason behind GRM's high accuracy, however, is that the RDW and OPIC datasets contain many machine-generated surrogate integer keys of the form $1, 2, \ldots, n$. As discussed above, the hashed values are very evenly distributed for such input. Thus, even when the synopsis size k is quite small, the spacings are very stable and even, leading to very accurate DV estimates. The BASEBALL dataset, on the other hand, contains almost no surrogate keys, and the accuracy of GRM drops significantly. For this dataset we see that the AES hash function is reasonably close to the baseline ARE, performs comparably to the other hash functions for all values of k, and has superior performance for small values of k. Overall, we recommend using AES as a hash function, because it tracks the baseline output reasonably closely for all datasets, which results in relatively reliable accuracy that is independent of the presence of surrogate keys.

6.2.3 Comparison with Best-of-Breed Estimators

We next compared the unbiased KMV estimator to two current best-of-breed estimators: the SDLogLog estimator, which is a highly

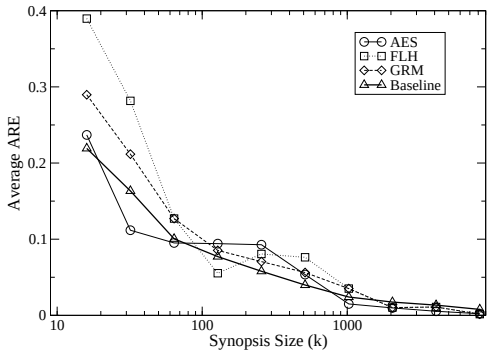


Figure 4: Hashing Effect on the BASEBALL Dataset

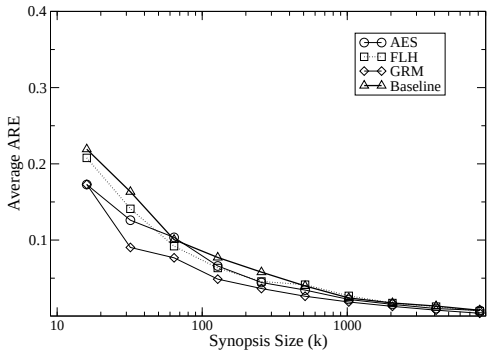


Figure 5: Hashing Effect on the OPIC Dataset

tuned implementation of the loglog estimator given in [11], and a variant of the sample-counting algorithm described in [3]. In preliminary experiments, we found that these latter two estimators were the best of the probabilistic-counting and sample-counting types, respectively. All of the algorithms used exactly the same amount of available memory, which corresponded to a synopsis size of $k = 8192$. We chose this value because it maximized the performance of our own hand-tuned optimized SDLogLog estimator.

The box plot in Figure 6 summarizes, for each estimator, the distribution of the ARE values over all of the datasets in the RDW database.[6] For comparison, we also plotted the ARE distribution for the baseline scenario when using the unbiased KMV estimator $\hat{D}_k^{\text{UB}}$ together with a synopsis of size $k = 8192$. As can be seen, the unbiased KMV estimator is significantly more accurate than both SDLogLog and sample-counting on real datasets. The main reason is that both SDLogLog and sample-counting merely approximate the basic estimator — and hence the MLE estimator — even when k is large, whereas the unbiased KMV estimator essentially coincides with the MLE estimator for large k. If the synopsis size were small, so that bias effects were important, then the unbiased KMV estimator would have a further accuracy advantage. In these experiments, we found that, for the large value of k that we used, the AES hash function distributed points somewhat more evenly than we would expect from a true random number generator, and this phenomenon resulted in a slight accuracy improvement; see Section 2.2. Thus these results provide further empirical evidence for the suitability of the AES hash function for DV estimation.

[6]The top, midpoint, and bottom of a box represent the 75th, 50th, and 25th percentiles of the ARE values, and the top of the thin line corresponds to the maximum ARE value that was observed.

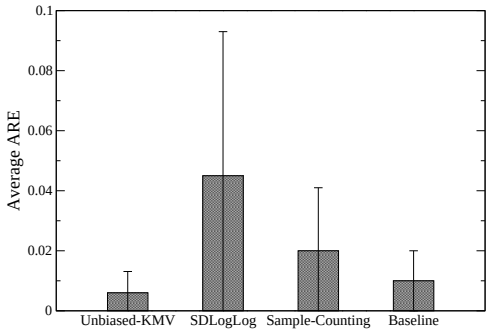


Figure 6: Accuracy Comparison for the RDW Dataset

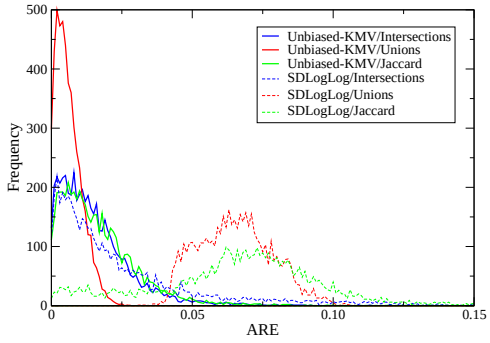


Figure 7: Accuracy Comparison for Union, Intersection, and Jaccard Distance on the RDW Dataset

6.3 Compound Partitions

The main motivation behind our work on the AKMV synopsis was the desire for efficient and accurate DV estimates within the synopsis-warehouse scenario, where different synopses need to be combined in order to efficiently extract interesting information about corresponding compound partitions. In practice, the most common operations in this setting are multiset union and intersection. We therefore evaluated the accuracy of the KMV-synopsis estimators in the presence of these operations, as well as the accuracy of our Jaccard-distance estimator.

For this experiment, we computed a KMV synopsis of size $k = 8192$ for each dataset in the RDW database. Then, for every possible pair of synopses, we used the unbiased estimator in (16) to estimate the DV count for the union and intersection, and also estimated the Jaccard distance using our new estimator $\hat{\rho}$ defined in (10). We also estimated these quantities using the SDLogLog estimator; specifically, we estimated the number of DVs in the union directly, and then used the inclusion/exclusion rule to estimate the DV count for the intersection and then for the Jaccard distance.

Figure 7 displays, for each estimator, a histogram for each of these three multiset operations. (The histogram shows, for each possible value of ARE, the number of dataset pairs for which the DV estimate yielded that specific ARE value.) For the majority of the datasets, the unbiased estimator based on the KMV synopsis provides estimates that are almost ten times more accurate than the SDLogLog estimates, even though both methods used exactly the same amount of available memory.

7. SUMMARY AND CONCLUSIONS

We have revisited the classical problem of DV estimation, but from a synopsis-oriented point of view. By combining and extend-

ing previous results on DV estimation, we have obtained a solution that is well suited to the synopsis warehouse architecture. Our solution comprises the AKMV synopsis, along with novel unbiased DV estimators that exploit an AKMV synopsis.

Our theoretical contributions include (1) using the theory of order statistics to derive a new class of unbiased DV estimators and to provide error bounds, (2) providing a connection to the results in [7] via an asymptotic analysis, and (3) providing a connection to the theory of maximum likelihood estimators, thereby establishing asymptotic efficiency. From a practical point of view, we have shown empirically that the AKMV synopsis, in combination with our new unbiased estimators, provides superior accuracy, especially when estimating the number of distinct values in compound partitions. Moreover, because our methods require only a single hash function, constructing the synopsis is relatively inexpensive. We have shown how to combine synopses in order to handle compound partitions. Based on our experiments, we have also provided guidance in selecting a hash function, and have identified the AES hash function as a reasonably good choice.

8. REFERENCES

[1] N. Alon, Y. Matias, and M. Szegedy. The space complexity of approximating the frequency moments. *J. Comput. Sys. Sci.*, 58:137–147, 1999.

[2] M. Astrahan, M. Schkolnick, and K. Whang. Approximating the number of unique values of an attribute without sorting. *Inf. Sys.*, 12:11–15, 1987.

[3] Z. Bar-Yossef, T. S. Jayram, R. Kumar, D. Sivakumar, and L. Trevisan. Counting distinct elements in a data stream. In *Proc. RANDOM*, pages 1–10, 2002.

[4] P. Brown, P. J. Haas, J. Myllymaki, H. Pirahesh, B. Reinwald, and Y. Sismanis. Toward automated large-scale information integration and discovery. In *Data Management in a Connected World*, pages 161–180. Springer, 2005.

[5] P. G. Brown and P. J. Haas. Techniques for warehousing of sample data. In *Proc. ICDE*, 2006.

[6] M. Charikar, S. Chaudhuri, R. Motwani, and V. R. Narasayya. Towards estimation error guarantees for distinct values. In *Proc. ACM PODS*, pages 268–279, 2000.

[7] E. Cohen. Size-estimation framework with applications to transitive closure and reachability. *J. Comput. Sys. Sci.*, 55:441–453, 1997.

[8] T. Dasu, T. Johnson, S. Muthukrishnan, and V. Shkapenyuk. Mining database structure; or, how to build a data quality browser. In *Proc. ACM SIGMOD*, pages 240–251, 2002.

[9] H. A. David and H. N. Nagaraja. *Order Statistics*. Wiley, third edition, 2003.

[10] A. R. Didonato and A. H. Morris, Jr. Algorithm 708; significant digit computation of the incomplete beta function ratios. *ACM Trans. Math. Software*, 18(3):360–373, 1992.

[11] M. Durand and P. Flajolet. Loglog counting of large cardinalities. In *Proc. 11th Eur. Symp. Algorithms (ESA 2003)*, volume 2832 of *Lecture Notes in Computer Science*. Springer, 2003.

[12] C. Estan, G. Varghese, and M. Fisk. Bitmap algorithms for counting active flows on high speed links. In *Proc. SIGCOMM 02*, pages 323–336, 2002.

[13] P. Flajolet. Adaptive sampling. In M. Hazewinkel, editor, *Encyclopaedia of Mathematics, Supplement I*. Kluwer, 1997.

[14] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *J. Computer Sys. Sci.*, 31:182–209, 1985.

[15] S. Ganguly, M. Garofalakis, and R. Rastogi. Tracking set-expression cardinalities over continuous update streams. *VLDB J.*, 13:354–369, 2004.

[16] P. B. Gibbons. Distinct sampling for highly-accurate answers to distinct values queries and event reports. In *Proc. VLDB*, pages 541–550, 2001.

[17] P. B. Gibbons and S. Tirthapura. Estimating simple functions on the union of data streams. In *Proc. ACM Symp. Parallel Algorithms and Architecture*, pages 281–291, 2001.

[18] F. Giroire. Order statistics and estimating cardinalities of massive data sets. In *Proc. Intl. Conf. Analysis Algorithms*, pages 157–166, 2005.

[19] P. J. Haas, Y. Liu, and L. Stokes. An estimator of the number of species from quadrat sampling. *Biometrics*, 62:135–141, 2006.

[20] P. J. Haas and L. Stokes. Estimating the number of classes in a finite population. *J. Amer. Statist. Assoc.*, 93:1475–1487, 1998.

[21] P. Hellekalek and S. Wegenkittl. Empirical evidence concerning AES. *ACM Trans. Modelling Comput. Simulation*, 13:322–333, 2003.

[22] Y. E. Ioannidis. The history of histograms (abridged). In *Proc. VLDB*, pages 19–30, 2003.

[23] N. L. Johnson, S. Kotz, and N. Balakrishnan. *Continuous Univeriate Distributions – 2*. Wiley, 2nd edition, 1995.

[24] S. Karlin and H. M. Taylor. *A Second Course in Stochastic Processes*. Academic Press, 1981.

[25] D. E. Knuth. *Sorting and Searching*, volume 3 of *The Art of Computer Programming*. Addison-Wesley, 1973.

[26] M. Matsumoto and T. Nishimura. Mersenne twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. Modeling Computer Simulation*, 8(1):3–30, 1998.

[27] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[28] S. Padmanabhan, B. Bhattacharjee, T. Malkemus, L. Cranston, and M. Huras. Multi-dimensional clustering: a new data layout scheme in DB2. In *Proc. ACM SIGMOD*, pages 637–641, 2003.

[29] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, and T. G. Price. Access path selection in a relational database management system. In *Proc. ACM SIGMOD*, pages 23–34, 1979.

[30] R. J. Serfling. *Approximation Theorems of Mathematical Statistics*. Wiley, New York, 1980.

[31] A. Shukla, P. Deshpande, J. F. Naughton, and K. Ramasamy. Storage estimation for multidimensional aggregates in the presence of hierarchies. In *Proc. VLDB*, pages 522–531, 1996.

[32] J. Vitter. Random Sampling with a Reservoir. *ACM Trans. Math. Software*, 11(1):37–57, 1985.

[33] K. Whang, B. T. Vander-Zanden, and H. M. Taylor. A linear-time probabilistic counting algorithm for database applications. *ACM Trans. Database Sys.*, 15:208–229, 1990.