DOI:10.1145/1562764.1562787

# Distinct-Value Synopses for Multiset Operations

By Kevin Beyer, Rainer Gemulla, Peter J. Haas, Berthold Reinwald, and Yannis Sismanis

**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 for the case in which the dataset of interest is split into partitions. We create for each partition a synopsis that can be used to estimate the number of DVs in the partition. By combining and extending a number of results in the literature, we obtain both suitable synopses and DV estimators. The synopses can be created in parallel, and can be easily combined to yield synopses and DV estimates for "compound" partitions that are created from the base partitions via arbitrary multiset union, intersection, or difference operations. Our synopses can also handle deletions of individual partition elements. We prove that our DV estimators are unbiased, provide error bounds, and show how to select synopsis sizes in order to achieve a desired estimation accuracy. Experiments and theory indicate that our synopses and estimators lead to lower computational costs and more accurate DV estimates than previous approaches.**

## 1. INTRODUCTION

The task of determining the number of distinct values (DVs) in a large dataset arises in a wide variety of settings. One classical application is population biology, where the goal is to determine the number of distinct species, based on observations of many individual animals. In computer science, applications include network monitoring, document search, predicate-selectivity estimation for database query optimization, storage-size estimation for physical database design, and discovery of metadata features such as keys and duplicates.

The number of DVs 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 DVs. 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 25 years has therefore focused on scalable approximate methods. These methods work either by drawing a random sample of the data items and statistically extrapolating the number of DVs, or by taking a single pass through the data and using hashing techniques to compute an estimate using a small, bounded amount of memory.

Almost all of this work has focused on producing a given synopsis of the dataset, such as a random sample or bit vector, and then using the synopsis to obtain a DV estimate. Issues related to combining and exploiting synopses in the presence of union, intersection, and difference operations on multiple datasets have been largely unexplored, as has the problem of handling deletions of items from the dataset. Such issues are the focus of this paper, which is about DV estimation methods when the dataset of interest is split into disjoint partitions, i.e., disjoint multisets.[a] The idea is to create a synopsis for each partition so that (i) the synopsis can be used to estimate the number of DVs in the partition and (ii) the synopses can be combined to create synopses for "compound" partitions that are created from the base partitions using multiset union, intersection, or difference operations.

This approach permits parallel processing, and hence scalability of the DV-estimation task to massive datasets, as well as flexible processing of DV-estimation queries and graceful handling of fluctuating data-arrival rates. The partitioning approach can also be used to support automated data integration by discovering relationships between partitions. For example, suppose that the data is partitioned by its source: Amazon customers versus YouTube downloaders. Then DV estimates can be used to help discover subset-inclusion and functional-dependency relationships, as well as to approximate the Jaccard distance or other similarity metrics between the domains of two partitions; see Brown and Haas and Dasu et al.[4, 6]

Our goal is therefore to provide "partition-aware" synopses for DV estimation, as well as corresponding DV estimators that exploit these synopses. 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 to obtain a solution to our problem, resulting in best-of-breed DV estimation methods that can also handle multiset operations and deletions. We first consider,

[a] Recall that a *multiset*, also called a *bag*, is an unordered collection of values, where a given value may appear multiple times, for example, {3,3,2,3,7,3,2}. Multiset union, intersection, and difference are defined in a natural way: if $n_A(v)$ and $n_B(v)$ denote the multiplicities of value $v$ in multisets $A$ and $B$, respectively, then the multiplicities of $v$ in $A \uplus B$, $A \cap_+ B$, and $A \setminus_+ B$ are given respectively by $n_A(v) + n_B(v)$, $\min(n_A(v), n_B(v))$, and $\max(n_A(v) - n_B(v), 0)$.

A previous version of this research paper was published in *Proceedings of the 2007 ACM SIGMOD Conference*.

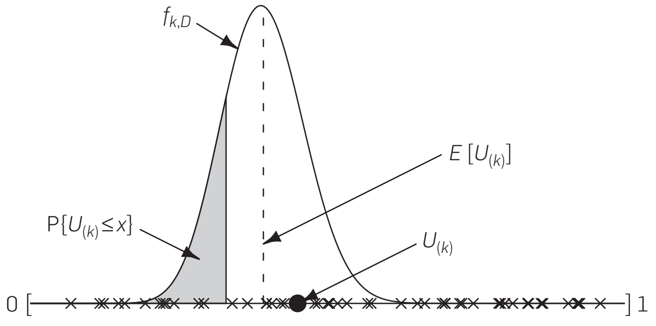
in Section 2, a simple "KMV" (*K* Minimum hash Values) synopsis[2] for a single partition—obtained by hashing the DVs in the partition and recording the *K* smallest hash values—along with a "basic" DV estimator based on the synopsis; see Equation 1. In Section 3, we briefly review prior work and show that virtually all prior DV estimators can be viewed as versions of, or approximations to, the basic estimator. In Section 4, we propose a new DV estimator—see Equation 2—that improves upon the basic estimator. The new estimator also uses the KMV synopsis and is a deceptively simple modification of the basic estimator. Under a probabilistic model of hashing, we show that the new estimator is unbiased and has lower mean-squared error than the basic estimator. Moreover, when there are many DVs and the synopsis size is large, we show that the new unbiased estimator has essentially the minimal possible variance of any DV estimator. To help users assess the precision of specific DV estimates produced by the unbiased estimator, we provide probabilistic error bounds. We also show how to determine appropriate synopsis sizes for achieving a desired error level.

In Section 5, we augment the KMV synopsis with counters—in the spirit of Ganguly et al. and Shukla et al.[10, 18]—to obtain an "AKMV synopsis." 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. We also show how to extend our simple unbiased estimator to exploit the AKMV synopsis and provide unbiased estimates in the presence of multiset operations, obtaining an unbiased estimate of Jaccard distance in the process; see Equations 7 and 8. Section 6 concerns some recent complements to, and extensions of, our original results in Beyer et al.[3]

## 2. A BASIC ESTIMATOR AND SYNOPSIS

The idea behind virtually all DV estimators can be viewed as follows. Each of the *D* DVs in a dataset is mapped to a random location on the unit interval, and we look at the position $U_{(k)}$ of the *k*th point from the left, for some fixed value of *k*; see Figure 1. The larger the value of *D*, i.e., the greater the number of points on the unit interval, the smaller the value of $U_{(k)}$. Thus *D* can plausibly be estimated by a decreasing function of $U_{(k)}$.

**Figure 1. 50 random points on the unit interval (*D* = 50, *k* = 20).**



Specifically, 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 *k*th smallest point, is $E[U_{(k)}] \approx \Sigma_{j=1}^{k} (1/D) = k/D$. Thus $D \approx k/E[U_{(k)}]$. The simplest estimator of $E[U_{(k)}]$ is simply $U_{(k)}$ itself—the so-called "method-of-moments" estimator—and yields the *basic estimator*

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

The above 1-to-1 mapping from the *D* DVs to a set of *D* uniform random numbers can be constructed perfectly using $O(D \log D)$ memory, but this memory requirement is clearly infeasible for very large datasets. Fortunately, a hash function—which typically only requires an amount of memory logarithmic in *D*—often "looks like" a uniform random number generator. In particular, let $\mathscr{D}(A) = \{v_1, v_2, \ldots, v_D\}$ be the *domain* of multiset *A*, i.e., the set of DVs in *A*, and let *h* be a hash function from $\mathscr{D}(A)$ to $\{0, 1, \ldots, M\}$, where *M* is a large positive integer. For many hash functions, the sequence $h(v_1), h(v_2), \ldots, h(v_D)$ looks 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 $U_1 = h(v_1)/M$, $U_2 = h(v_2)/M$, ..., $U_D = h(v_D)/M$ 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 [16, p. 45] shows that collisions will be avoided when $M = \Omega(D^2)$. 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 *k*th smallest hash value (normalized by a factor of $1/M$). In general, $E[1/X] > 1/E[X]$ for a non-negative random variable *X*,[17, p. 351] and hence

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

**Algorithm 1 (KMV Computation).**

```
h: hash function from domain of dataset to {0, 1, ..., M}
L: list of k smallest hash values seen so far
maxVal(L): returns the largest value in L

for each item x in the dataset do
    $v = h(x)$
    if $v \notin L$ then
        if $|L| < k$ then
            insert v into L
        else if $v <$ maxVal(L) then
            insert v into L
            remove largest element of L
        end if
    end if
end for
```

i.e., the estimator $\hat{D}_k^{\mathrm{BE}}$ is biased upwards for each possible value of $D$, so that it overestimates $D$ on average. Indeed, it follows from the results in Section 4.1 that $E[\hat{D}_k^{\mathrm{BE}}] = \infty$ for $k = 1$. 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 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.[3]

The foregoing discussion of the basic estimator immediately implies a choice of synopsis for a partition $A$. Using a hash function as above, hash all of the DVs in $A$ and then record the $k$ smallest hash values. We call this synopsis a *KMV synopsis* (for $k$ minimum values). The KMV synopsis can be viewed as originating in Bar-Yossef et al.,[2] but there is no discussion in Bar-Yossef et al.[2] about implementing, constructing, or combining such synopses.

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

A KMV synopsis can be computed from a single scan of the data partition, using Algorithm 1. The algorithm uses a sorted list of $k$ hash 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 items' hash values, 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 $i$th 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, the conditional probability of this event, given the value of $M_i$, is $P\{U_i < M_i | M_i\} = M_i$. By the law of total expectation, $P\{U_i < M_i\} = E[P\{U_i < M_i \mid M_i\}] = E[M_i] = k/i$. Thus the expected cost for handling the remaining $D - k$ DVs is

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

since $\Sigma_{i=1}^{D}(\mathrm{l}/i) = O(\log D)$. The overall expected cost is thus $O(N + D + k \log k + k \log k \log D) = O(N + k \log k \log D)$. □

We show in Section 5 that adding counters to the KMV synopsis has a negligible effect on the construction cost, and results in a desirable "closure" property that permits efficient DV estimation under multiset operations.

## 3. PRIOR WORK

We now give a unified view of prior synopses and DV estimators, and discuss prior methods for handling compound partitions.

### 3.1. 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, infer potential candidate synopses from the various algorithm descriptions. The literature on DV estimation is enormous, so we content ourselves with giving highlights and pointers to further references; for some helpful literature reviews, see Beyer et al.,[3] Gemulla,[11] Gibbons[13] and Haas and Stokes.[14]

**Random Samples**: Historically, the problem of DV estimation was originally considered for the case in which the synopsis comprises a random sample of the data items. Applications included estimation of species diversity (as discussed in the introduction), determining the number of distinct Roman coins based on the sample of coins that have survived, estimating the size of Shakespeare's vocabulary based on his extant writings, estimating the number of distinct individuals in a set of merged census lists, and so forth; see Haas and Stokes[14] and references therein. The key drawback is that DV estimates computed from such a synopsis can be very inaccurate, especially when the dataset is dominated by a few highly frequent values, or when there are many DVs, each having a low frequency (but not all unique). With high probability, a sample of size $k$ will have only one or two DVs in the former case—leading to severe underestimates—and $k$ DVs in the latter case—leading to severe overestimates. For this reason, especially in the computer science literature, the emphasis has been on algorithms that take a complete pass through the dataset, but use limited memory. When the dataset is massive, our results are of particular interest, since we can parallelize the DV-estimate computation.

Note that if we modify the KMV synopsis to record not the hash of a value, but the value itself, then we are in effect maintaining a uniform sample of the DVs in the dataset, thereby avoiding the problems mentioned above. See Gemulla[11] for a thorough discussion of "distinct-value sampling" and its relationship to the DV-estimation problem.

**Bit-Vector Synopses**: The oldest class of synopses based on single-pass, limited-memory processing comprises various types of bit vectors. The "linear counting" technique [1, 21] hashes each DV to a position in a bit vector $V$ of length $M = O(D)$, and uses the number of 1-bits to estimate the DV count. Its $O(D)$ storage requirement is typically unacceptable for modern datasets.

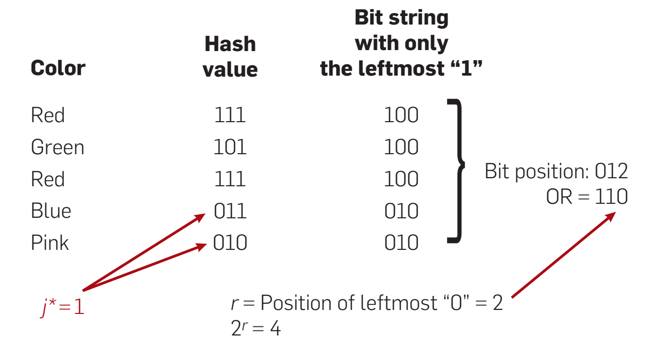
The "logarithmic counting" method of Flajolet and Martin [1,9] uses a bit vector of length $L = O(\log D)$. The idea is to hash each of the DVs in $A$ to the set $\{0, 1\}^L$ of binary strings of length $L$, and look for patterns of the form $0^j1$ in the leftmost bits of the hash values. For a given value of $j$, the probability of such a pattern is $1/2^{j+1}$, so the expected observed number of such patterns after $D$ DVs have been hashed is $D/2^{j+1}$. Assuming that the longest observed pattern, say with $j^*$ leading 0's, is expected to occur once, we set $D/2^{j^*+1} = 1$, so that $D = 2^{j^*+1}$; see Figure 2, which has been adapted from Astrahan et al.[1] The value of $j^*$ is determined approximately, by taking each hash value, transforming the value by zeroing out all but the leftmost 1, and computing the logarithmic-counting synopsis as the bitwise-OR of the transformed values. Let $r$ denote the position (counting from the left, starting at 0) of the leftmost 0 bit in the synopsis. Then $r$ is an upper bound for $j^*$, and typically a lower bound for $j^* + 1$, leading to a crude (under)estimate of $2^r$. For example, if $r = 2$, so that the leftmost bits of the synopsis are 110 (as in Figure 2), we know that the pattern 001 did not occur in any of the hash values, so that $j^* < 2$.

The actual DV estimate is obtained by multiplying $2^r$ by a factor that corrects for the downward bias, as well as for hash collisions. In the complete algorithm, several independent values of $r$ are, in effect, averaged together (using a technique called "stochastic averaging") and then exponentiated. Subsequent work by Durand and Flajolet[8] improves on the storage requirement of the logarithmic counting algorithm by tracking and maintaining $j^*$ directly. The number of bits needed to encode $j^*$ 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 partitioned-data 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 [10,18] 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.

**Figure 2. Logarithmic counting.**



| Color | Hash value | Bit string with only the leftmost "1" |
|---|---|---|
| Red | 111 | 100 |
| Green | 101 | 100 |
| Red | 111 | 100 |
| Blue | 011 | 010 |
| Pink | 010 | 010 |

**Sample-Counting Synopsis**: Another type of synopsis arises from the "sample counting" DV-estimation method—also called "adaptive sampling"—credited to Wegman.[1] Here the synopsis for partition $A$ comprises a subset of $\{h(v): v \in \mathscr{D}(A)\}$, where $h: \mathscr{D}(A) \mapsto \{0, 1, \dots, M\}$ is a hash function as before. 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 DVs in the partition, as in logarithmic counting, and insert the hash values into a buffer that can hold up to $k > 0$ hash values; the buffer tracks only the distinct hash values inserted into it. When the buffer fills up, it is purged by removing all hash values whose leftmost bit is not equal to the leftmost bit of $s$; this operation removes roughly half of the hash values in the buffer. From this point on, a hash 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 hash 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. For sample-counting algorithms with reference string equal to $00\cdots0$, the synopsis holds the $K$ smallest hash values encountered, where $K$ lies roughly between $k/2$ and $k$.

**The Bellman Synopsis**: In the context of the Bellman system, the authors in Dasu et al.[6] propose a synopsis related to DV estimation. This synopsis comprises $k$ entries and uses independent hash functions $h_1, h_2, \dots, h_k$; the $i$th synopsis entry is given by the $i$th *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 partition domains; see Section 3.3. (The *Jaccard distance* between ordinary sets $A$ and $B$ is defined as $J(A, B) = |A \cap B|/|A \cup B|$. If $J(A, B) = 1$, then $A = B$; if $J(A, B) = 0$, then $A$ and $B$ are disjoint.) Indeed, this synopsis cannot be used directly for DV estimation because the associated DV estimator is basically $\hat{D}_1^{\mathrm{BE}}$, which has infinite expectation; see Section 2. When constructing the synopsis, each scanned data item must be hashed $k$ times for comparison to the $k$ current minHash values; for the KMV synopsis, each scanned item need only be hashed once.

### 3.2. DV estimators

The basic estimator $\hat{D}_k^{\mathrm{BE}}$ was proposed in Bar-Yosssef et al.,[2] along with conservative error bounds based on Chebyshev's inequality. 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 $00 \cdots 0$ and, as before, consider the normalized binary decimal representation of the hash values. Thus the first purge leaves behind normalized values of the form $0.0\cdots$, the second purge leaves behind values of the form $0.00\cdots$, and so forth, the last ($r$th) purge leaving behind only normalized hash 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 hash values in the size-$k$ buffer, so that the estimate $K/2^{-r}$ is roughly equal to $\hat{D}_k^{\mathrm{BE}}$.

### 3.3. Estimates for compound partitions

To our knowledge, the only prior discussion of how to construct DV-related estimates for compound partitions is found in Dasu et al.[6] DV estimation for the intersection of partitions $A$ and $B$ is not computed directly. Instead, the Jaccard distance $\rho = J(\mathscr{D}(A), \mathscr{D}(B))$ is estimated first by an estimator $\hat{\rho}$, and then the number of values in the intersection of $\mathscr{D}(A)$ and $\mathscr{D}(B)$ is estimated as

$$\hat{D} = \frac{\hat{\rho}}{\hat{\rho}+1}\left(|\mathscr{D}(A)| + |\mathscr{D}(B)|\right)$$

The quantities $|\mathscr{D}(A)|$ and $|\mathscr{D}(B)|$ are computed exactly, by means of GROUP BY relational queries; our proposed estimators avoid the need to compute or estimate these quantities. There is no discussion in Dasu et al.[6] 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.

## 4. AN IMPROVED DV ESTIMATOR

As discussed previously, the basic estimator $\hat{D}_k^{\mathrm{BE}}$ is biased upwards for the true number of DVs $D$, and so somehow needs to be adjusted downward. We therefore consider the estimator

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

and show that, both compared to the basic estimator and in a certain absolute sense, $\hat{D}_k^{\mathrm{UB}}$ has superior statistical properties, including unbiasedness. 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. Henceforth, we assume without further comment that $D > k$; if $D \le k$, then we can easily detect this situation and compute the exact value of $D$ from the synopsis.

### 4.1. Moments and error bounds

Let $U_1, U_2, \ldots, U_D$ be the normalized hash values of the $D$ distinct items in the dataset; for our analysis, we model these values as a sequence of i.i.d. random variables from the uniform [0, 1] distribution—see the discussion in Section 2. As before, denote by $U_{(k)}$ the $k$th smallest of $U_1, U_2, \ldots, U_D$, that is, $U_{(k)}$ is the $k$th uniform *order statistic*. We can now apply results from the classical theory of order statistics to establish properties of the estimator $\hat{D}_k^{\mathrm{UB}} = (k-1)/U_{(k)}$. We focus on moments and error bounds; additional analysis of $\hat{D}_k^{\mathrm{UB}}$ can be found in Beyer et al.[3]

Our analysis rests on the fact that 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 standard beta function; see Figure 1. To verify (3), fix $x \in [0, 1]$ and observe that, for each $1 \le i \le D$, we have $P\{U_i \le x\} = x$ by definition of the uniform distribution. The probability that exactly $k$ of the $D$ uniform random variables are less than or equal to $x$ is a binomial probability: $\binom{D}{k}x^k(1-x)^{D-k}$. Thus the probability that $U_{(k)} \le x$ is equal to the probability that *at least* $k$ random variables are less than or equal to $x$, which is a sum of binomial probabilities:

$$P\{X_{(k)} \le x\} = \sum_{j=k}^{D}\binom{D}{j}x^j(1-x)^{D-j} = \int_0^x f_{k,D}(t)\,dt$$

The second equality can be established by integrating the rightmost term by parts repeatedly and using the identity

$$B(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)} = \frac{(a-1)!(b-1)!}{(a+b-1)!} \tag{4}$$

where $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}\,dt$ is the standard gamma function and the rightmost equality is valid for integer $a$ and $b$. The result in (3) now follows by differentiation.

With (3) in hand, we can now determine the moments of $\hat{D}_k^{\mathrm{UB}}$, in particular, the mean and variance. Denote by $a^{\underline{b}}$ the falling power $a(a-1)\cdots(a-b+1)$.

Theorem 2. *Suppose that $r \ge 0$ is an integer with $r < k$. Then*

$$E[(\hat{D}_k^{UB})^r] = (k-1)^r D^{\underline{r}}/(k-1)^{\underline{r}} \tag{5}$$

*In particular, $E[\hat{D}_k^{UB}] = D$ provided that $k > 1$, and* $\mathrm{Var}[\hat{D}_k^{UB}] =$ *$D(D-k+1)/(k-2)$ provided that $k > 2$.*

Proof. For $k > r \ge 0$, if follows from (3) that

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

and the first assertion of the theorem follows directly from (4). Setting $r = 1$ in (5) yields the next assertion, and the final assertion follows from (5) and the relation $\mathrm{Var}[\hat{D}_k^{UB}] = E[(\hat{D}_k^{\mathrm{UB}})^2] - E^2[\hat{D}_k^{\mathrm{UB}}]$. □

Recall that the mean squared error (MSE) of a statistical estimator $X$ of an unknown quantity $\mu$ is defined as $\mathrm{MSE}[X] = E[(X-\mu)^2] = \mathrm{Var}[X] + \mathrm{Bias}^2[X]$. To compare the MSE of the basic and unbiased estimators, note that, by (5), $E[\hat{D}_k^{\text{BE}}] = kD/(k-1)$ and

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

Thus $\hat{D}_k^{\text{BE}}$ is biased high for $D$, as discussed earlier, and has higher MSE than $\hat{D}_k^{\text{UB}}$.

We can also use the result in (3) to obtain probabilistic (relative) error bounds for the estimator $\hat{D}_k^{\text{UB}}$. Specifically, set $I_x(a, b) = \int_0^x t^{a-1}(1-t)^{b-1}\,dt/B(a, b)$, so that $P\{U_{(k)} \le x\} = I_x(k, D-k+1)$. Then, for $0 < \varepsilon < 1$ and $k \ge 1$, we have

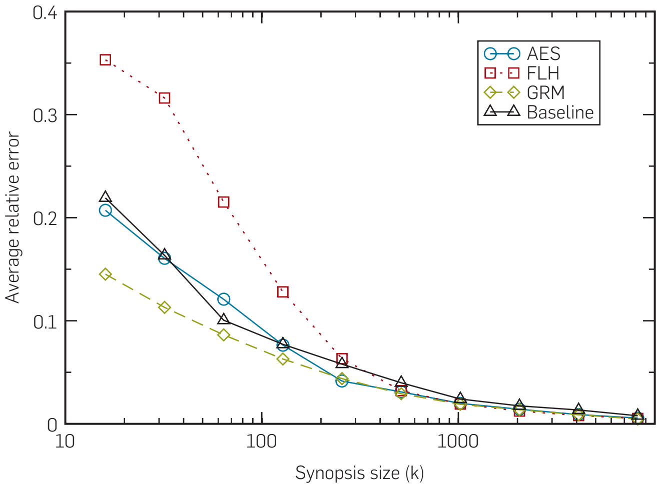
$$P\left\{\frac{|\hat{D}_k^{\text{UB}} - D|}{D} \le \varepsilon\right\} = P\left\{\frac{k-1}{(1+\varepsilon)D} \le U_{(k)} \le \frac{k-1}{(1-\varepsilon)D}\right\}$$
$$= I_{\frac{k-1}{(1-\varepsilon)D}}(k, D-k+1) - I_{\frac{k-1}{(1+\varepsilon)D}}(k, D-k+1). \tag{6}$$

For example, if $D = 10^6$ and the KMV synopsis size is $k = 1024$, then, with probability $\delta = 0.95$, the estimator $\hat{D}_k^{\text{UB}}$ will be within ±4% of the true value; this result is obtained by equating the right side of (6) to $\delta$ and solving for $\varepsilon$ numerically. In practice, of course, $D$ will be unknown, but we can assess the precision of $\hat{D}_k^{\text{UB}}$ approximately by replacing $D$ with $\hat{D}_k^{\text{UB}}$ in the right side of (6) prior to solving for $\varepsilon$.

Error bounds can also be derived for $\hat{D}_k^{\text{BE}}$. As discussed in Beyer et al.,[3] $\hat{D}_k^{\text{UB}}$ is noticeably superior to $\hat{D}_k^{\text{BE}}$ 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 $\varepsilon$. As $k$ increases, $k - 1 \approx k$ and both estimators perform similarly.

The foregoing development assumes that the hash function behaves as a 1-to-1 mapping from the $D$ DVs in the dataset to a set of $D$ uniform random numbers. In practice, we must use real-world hash functions that only approximate this ideal; see Section 2. Figure 3 shows the effect of using real hash functions on real data. The RDW database was obtained from the data warehouse of a large financial company, and consists of 24 relational tables, with a total of 504 attributes and roughly 2.6 million tuples. For several different hash functions, we computed the average value of the relative error $\text{RE}(\hat{D}_k^{\text{UB}}) = |\hat{D}_k^{\text{UB}} - D|/D$ over multiple datasets in the database. The hash functions are described in detail in Beyer et al.[3]; for example, the Advanced Encryption Standard (AES) hash function is a well established cipher function that has been studied extensively. The "baseline" curve corresponds to an idealized hash function as used in our analysis. As can be seen, the real-world accuracies are consistent with the idealized results, and reasonable accuracy can be obtained even for synopsis sizes of $k < 100$. In Beyer et al.,[3] we found that the relative performance of different hash functions was sensitive to the degree of regularity in the data; the AES hash function is relatively robust to such data properties, and is our recommended hash function.

**Figure 3. Hashing Effect on the RDW Dataset.**



### 4.2. Analysis with many DVs

When the number of DVs is known to be large, we can establish a minimum-variance property of the $\hat{D}_k^{\text{UB}}$ estimator, and also develop useful approximate error bounds that can be used to select a synopsis size prior to data processing.

**Minimum Variance Property**: The classical statistical approach to estimating unknown parameters based on a data sample is the method of maximum likelihood.[7, Sec. 4.2] A basic result for maximum-likelihood estimators[17, Sec. 4.2.2] asserts that an MLE of an unknown parameter has the minimum variance over all possible parameter estimates as the sample size becomes large. We show that $\hat{D}_k^{\text{UB}}$ is asymptotically equivalent to the maximum-likelihood estimator (MLE) as $D$ and $k$ become large. Thus, for $D \gg k \gg 1$, the estimator $\hat{D}_k^{\text{UB}}$ has, to a good approximation, the minimal possible variance for any estimator of $D$.

To find the MLE, we cast 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 $L(D;U_{(k)}) = f_{k,D}(U_{(k)})$. That is, roughly speaking, $\hat{D}$ maximizes the probability of seeing the value of $U_{(k)}$ that was actually observed. 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 $\Psi$ denotes the digamma function. If $x$ is sufficiently large, then $\Psi(x) \approx \ln(x-1) + \gamma$, where $\gamma \approx 0.5772$ denotes Euler's constant. Applying this approximation, we obtain $\hat{D}_k^{\text{MLE}} \approx k/U_{(k)}$, so that the MLE estimator roughly resembles the basic estimator $\hat{D}_k^{\text{BE}}$ provided that $D \gg k \gg 1$. In fact, our experiments indicated that $\hat{D}_k^{\text{MLE}}$ and $\hat{D}_k^{\text{BE}}$ are indistinguishable from a practical point of view. It follows that $\hat{D}_k^{\text{MLE}} \approx (k/k-1)\hat{D}_k^{\text{UB}}$ is asymptotically equivalent to $\hat{D}_k^{\text{UB}}$ as $k \to \infty$.

**Approximate Error Bounds**: To obtain approximate probabilistic error bounds when the number of DVs is large, we simply let $D \to \infty$ in (6), yielding

$$P\{\text{RE}(\hat{D}_k^{\text{UB}}) \le \varepsilon\} \approx e^{-\frac{k-1}{1+\varepsilon}}\left(1 + \sum_{i=1}^{k-1}\frac{(k-1)^i}{(1+\varepsilon)^i\, i!}\right)$$
$$- e^{-\frac{k-1}{1-\varepsilon}}\left(1 + \sum_{i=1}^{k-1}\frac{(k-1)^i}{(1-\varepsilon)^i\, i!}\right).$$

An alternative derivation is given in Beyer et al.[3] using a powerful proof technique that exploits well known properties of the exponential distribution. The approximate error bounds have the advantageous property that, unlike the exact bounds, they do not involve the unknown quantity $D$. Thus, given desired values of $\varepsilon$ and $\delta$, they can be used to help determine target synopsis sizes prior to processing the data. When $D$ is not large, the resulting recommended synopsis sizes are slightly larger than necessary, but not too wasteful. It is also shown in Beyer et al.[3] that $E[\mathrm{RE}(\hat{D}_k^{\mathrm{UB}})] \approx (\pi(k-2)/2)^{-1/2}$ for large $D$, further clarifying the relationship between synopsis size and accuracy.

## 5. FLEXIBLE AND SCALABLE ESTIMATION

The discussion up until now has focused on improving the process of creating and using a synopsis to estimate the number of DVs in a single base partition, i.e., in a single dataset. As discussed in the introduction, however, the true power of our techniques lies in the ability to split up a massive dataset into partitions, create synopses for the partitions in parallel, and then compose the synopses to obtain DV estimates for arbitrary combinations of partitions, e.g., if the combination of interest is in fact the union of all the partitions, then, as in the classical setting, we can estimate the number of DVs in the entire dataset, while parallelizing the traditional sequential scan of the data. On the other hand, estimates of the number of DVs in the intersection of partitions can be used to estimate similarity measures such as Jaccard distance.

We therefore shift our focus to DV estimation for a compound partition that is created from a set of base partitions using the multiset operations of intersection, union, and difference. To handle compound partitions, we 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 $A$ and $B$ are compound partitions and $E$ is obtained from $A$ and $B$ via a multiset operation, then we can compute an AKMV synopsis for $E$ from the corresponding AKMV synopses for $A$ and $B$, and unbiasedly estimate the number of DVs in $E$ from this resulting synopsis. This procedure avoids the need to access the synopsis for each of the base partitions that were used to create $A$ and $B$. The AKMV synopsis can also handle deletions of individual items from the dataset. 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.

### 5.1. AKMV synopses

We assume throughout that all synopses are created using the same hash function $h: \mathscr{D} \mapsto \{0, 1, \ldots, M\}$, where $\mathscr{D}$ denotes the domain of the data values that appear in the partitions and $M = \Omega(|\mathscr{D}|^2)$ as discussed in Section 2. We start by focusing on insertion-only environments, and discuss deletions in Section 5.3.

We first define the AKMV synopsis of a base partition $A$, where $A$ has a KMV synopsis $L = (h(v_1), h(v_2), \ldots, h(v_k))$ of size $k$, with $h(v_1) < h(v_2) < \cdots < h(v_k)$. The AKMV synopsis of $A$ is defined as $L^+ = (L, c)$, where $c = (c(v_1), c(v_2), \ldots, c(v_k))$ is a set of $k$ non-negative counters. The quantity $c(v)$ is the multiplicity in $A$ of the value $v$. The first two lines in Figure 4 show the normalized hash values and corresponding counter values in the AKMV synopses $L_A^+$ and $L_B^+$ of two base partitions $A$ and $B$, respectively. (Circles and squares represent normalized hash values; inscribed numbers represent counter values.)

The size of the AKMV synopsis is $O(k \log D + k \log N)$, where $N$ is the number of data items in $A$. It is easy to modify Algorithm 1 to create and maintain counters via $O(1)$ operations. The modified synopsis retains the original construction complexity of $O(N + k \log k \log D)$.

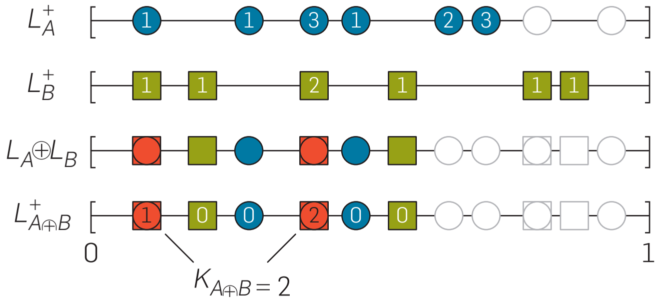
To define an AKMV synopsis for compound partitions, we first define an operator $\oplus$ for combining KMV synopses. Consider two partitions $A$ and $B$, along with their KMV synopses $L_A$ and $L_B$ of sizes $k_A$ and $k_B$, respectively. Define $L_A \oplus L_B$ to be the ordered list comprising the $k$ smallest values in $L_A \cup L_B$, where $k = \min(k_A, k_B)$ and we temporarily treat $L_A$ and $L_B$ as sets rather than ordered lists. (See line 3 of Figure 4; the yellow points correspond to values that occur in both $L_A$ and $L_B$.) Observe that the $\oplus$ operator is symmetric and associative.

THEOREM 3. *The ordered list* $L = L_A \oplus L_B$ *is the size-k KMV synopsis of* $A \uplus B$, *where* $k = \min(k_A, k_B)$.

PROOF. We again temporarily treat $L_A$ and $L_B$ as sets. 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 $h(A \uplus B)$. 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 $h$ i∈ $(L_A \cup L_b)\backslash G$, we have that $h > \max_{h' \in G} h'$ by definition of $G$, because $h \in h(A \uplus B)$. 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 $A \uplus B$. □

We next define the AKMV synopsis for a compound partition $E$ created from $n \ge 2$ base partitions $A_1, A_2, \ldots, A_n$ using the multiset union, intersection, and set-difference operators. Some examples are $E = A_1 \backslash\!+ A_2$ and $E = ((A_1 \uplus A_2) ⩀ (A_3 \uplus A_4)) \backslash\!+ A_5$. The AKMV synopsis for $E$ is defined as $L_E^+ = (L_E, C_E)$, where $L_E = L_{A_1} \oplus L_{A_2} \oplus \cdots \oplus L_{A_n}$ is of size $k = \min(k_{A_1}, k_{A_2}, \cdots, k_{A_n})$, and, for $v \in \cup_{i=1}^{n} \mathscr{D}(A_i)$, the counter $c_E(v)$ is the multiplicity

**Figure 4. Combining two AKMV synopses of size $k = 6$ (synopsis elements are colored).**



of $v$ in $E$; observe that $c_E(v) = 0$ if $v$ is not present in $E$, so that there may be one or more zero-valued counters; see line 4 of Figure 4 for the case $E = A \Cap B$.

With these definitions, the collection of AKMV synopses over compound partitions is closed under multiset operations, and an AKMV synopsis can be built up incrementally from the synopses for the base partitions. Specifically, suppose that we combine (base or compound) partitions $A$ and $B$—having respective AKMV synopses $L_A^+ = (L_A, c_A)$ and $L_B^+ = (L_B, c_B)$—to create $E = A \diamond B$, where $\diamond \in \{\uplus, \Cap, \setminus_+\}$. Then the AKMV synopsis for $E$ is $(L_A \oplus L_B, c_E)$, where

$$c_E(v) = \begin{cases} c_A(v) + c_B(v) & \text{if } \diamond = \uplus; \\ \min(c_A(v), c_B(v)) & \text{if } \diamond = \Cap; \\ \max(c_A(v) - c_B(v), 0) & \text{if } \diamond = \setminus_+. \end{cases}$$

As discussed in Beyer et al. and Gemulla,[3, 11] the AKMV synopsis can be sometimes be simplified. If, for example, all partitions are ordinary sets (not multisets) and ordinary set operations are used to create compound partitions, then the AKMV counters can be replaced by a compact bit vector, yielding an $O(k \log D)$ synopsis size.

### 5.2. DV estimator for AKMV synopsis

We now show how to estimate the number of DVs for a compound partition using the partition's AKMV synopsis; to this end, we need to generalize the unbiased DV estimator of Section 4. Consider a compound partition $E$ created from $n \geq 2$ base partitions $A_1, A_2, \ldots, A_n$ using multiset operations, along with the AKMV synopsis $L_E^+ = (L_E, c_E)$, where $L_E = (h(v_1), h(v_2), \ldots, h(v_k))$. Denote by $V_L = \{v_1, v_2, \ldots, v_k\}$ the set of data values corresponding to the elements of $L_E$. (Recall our assumption that there are no hash collisions.) Set $K_E = |\{v \in V_L: c_E(v) > 0\}|$; for the example $E = A \Cap B$ in Figure 4, $K_E = 2$. It follows from Theorem 3 that $L_E$ is a size-$k$ KMV synopsis of the multiset $A_\uplus = A_1 \uplus A_2 \uplus \cdots \uplus A_n$. The key observation is that, under our random hashing model, $V_L$ can be viewed as a random sample of size $k$ drawn uniformly and without replacement from $\mathscr{D}(A_\uplus)$; denote by $D_\uplus = |\mathscr{D}(A_\uplus)|$ the number of DVs in $A_\uplus$. The quantity $K_E$ is a random variable that represents the number of elements in $V_L$ that also belong to the set $\mathscr{D}(E)$. It follows that $K_E$ has a hypergeometric distribution: setting $D_E = |\mathscr{D}(E)|$, we have $P\{K_E = j\} = \binom{D_E}{j}\binom{D_\uplus - D_E}{k-j} / \binom{D_\uplus}{k}$.

We now estimate $D_E$ using $K_E$, as defined above, and $U_{(k)}$, the largest hash value in $L_E$. From Section 4.1 and Theorem 3, we know that $\hat{D}_\uplus = (k-1)/U_{(k)}$ is an unbiased estimator of $D_\uplus$; we would like to "correct" this estimator via multiplication by the ratio $\rho = D_E/D_\uplus$. We do not know $\rho$, but a reasonable estimate is

$$\hat{\rho} = K_E/k \qquad (7)$$

the fraction of elements in the sample $V_L \subseteq \mathscr{D}(A_\uplus)$ that belong to $\mathscr{D}(E)$. This leads to our proposed estimator

$$\hat{D}_E = \frac{K_E}{k}\left(\frac{k-1}{U_{(k)}}\right) \qquad (8)$$

THEOREM 4. *If $k > 1$ then $E[\hat{D}_E] = D_E$. If $k > 2$, then*

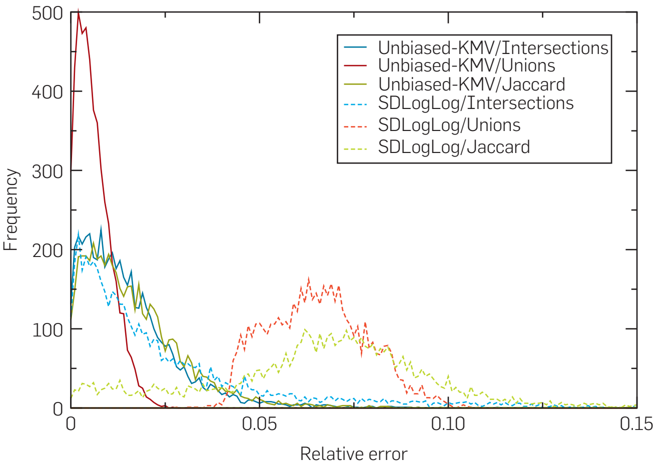
$$\mathrm{Var}[\hat{D}_E] = \frac{D_E(kD_\uplus - k^2 - D_\uplus + k + D_E)}{k(k-2)}$$

PROOF. The distribution of $K_E$ does not depend on the hash values $\{h(v): v \in \mathscr{D}(A_\uplus)\}$. It follows that the random variables $K_E$ and $U_{(k)}$, and hence the variables $\hat{p}$ and $U_{(k)}$, are statistically independent. By standard properties of the hypergeometric distribution, $E[K_E] = kD_E/D_\uplus$, so that $E[\hat{D}_E] = E[\hat{\rho}\,\hat{D}_\uplus] = E[\hat{\rho}]E[\hat{D}_\uplus] = \rho\, D_\uplus = D_E$. The second assertion follows in a similar manner. ❑

Thus $\hat{D}_E$ is unbiased for $D_E$. It also follows from the proof that the estimator $\hat{\rho}$ is unbiased for $D_E/D_\uplus$. In the special case where $E = A \cap B$ for two ordinary sets $A$ and $B$, the ratio $\rho$ corresponds to the Jaccard distance $J(A, B)$, and we obtain an unbiased estimator of this quantity. We can also obtain probability bounds that generalize the bounds in (6); see Beyer et al.[3] for details.

Figure 5 displays the accuracy of the AKMV estimator when estimating the number of DVs in a compound partition. 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 $\hat{D}_E$ to estimate the DV count for the union and intersection, and also estimated the Jaccard distance between set domains using our new unbiased estimator $\hat{\rho}$ defined in (7). We also estimated these quantities using a best-of-breed estimator: the SDLogLog estimator, which is a highly tuned implementation of the LogLog estimator given in Durand and Flajolet.[8] For this latter estimator, 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. The figure displays, for each estimator, a relative-error histogram for each of these three multiset operations. (The histogram shows, for each possible RE value, the number of dataset pairs for which the DV estimate yielded that value.) For the

**Figure 5. Accuracy Comparison for Union, Intersection, and Jaccard Distance on the RDW Dataset.**



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.

### 5.3. 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 $A_i \setminus_+ A_d$ yields the AKMV synopsis $L^+_A$ of the true multiset $A$. We do not actually need two synopses: 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 $L^+_A$ described above. If too many counters become equal to 0, the quality of synopsis-based DV estimates will deteriorate. Whenever the number of deletions causes the error bounds to become unacceptable, the data can be scanned to compute a fresh synopsis.

## 6. RECENT WORK

A number of developments have occurred both in parallel with, and subsequent to, the work described in Beyer et al.[3] Duffield et al.[7] devised a sampling scheme for weighted items, called "priority sampling," for the purpose of estimating "subset sums," i.e., sums of weights over subsets of items. Priority sampling can be viewed as assigning a priority to an item $i$ with weight $w_i$ by generating random number $U_i$ uniformly on $[0, 1/w_i]$, and storing the $k$ smallest-priority items. In the special case of unit weights, if we use a hash function instead of generating random numbers and if we eliminate duplicate priorities, the synopsis reduces to the KMV synopsis and, when the subset in question corresponds to the entire population, the subset-sum estimator coincides with $\hat{D}_k^{\mathrm{UB}}$. If duplicate priorities are counted rather than eliminated, the AKMV synopsis is obtained (but we then use different estimators). An "almost minimum" variance property of the priority-sampling-based estimate has been established in Szegedy[20]; this result carries over to the KMV synopsis, and strengthens our asymptotic minimum-variance result. In Cohen et al.,[5] the priority-sampling approach has recently been generalized to a variety of priority-assignment schemes.

Since the publication of Beyer et al.,[3] there have been several applications of KMV-type synopses and corresponding estimators to estimate the number of items in a time-based sliding window,[12] the selectivity of set-similarity queries,[15] the intersection sizes of posting lists for purposes of OLAP-style querying of content-management systems,[19] and document-key statistics that are useful for identifying promising publishers in a publish-subscribe system.[22]

## 7. SUMMARY AND CONCLUSION

We have revisited the classical problem of DV estimation, but from a synopsis-oriented point of view. By combining and extending previous results on DV estimation, we have obtained the AKMV synopsis, along with corresponding unbiased DV estimators. Our synopses are relatively inexpensive to construct, yield superior accuracy, and can be combined to easily handle compound partitions, permitting flexible and scalable DV estimation.

### References

1. Astrahan, M., Schkolnick, M., Whang, K. Approximating the number of unique values of an attribute without sorting. *Inf. Sys. 12* (1987), 11–15.
2. Bar-Yossef, Z., Jayram, T.S., Kumar, R., Sivakumar, D., Trevisan, L. Counting distinct elements in a data stream. In *Proc. RANDOM* (2002), 1–10.
3. Beyer, K.S., Haas, P.J., Reinwald, B., Sismanis, Y., Gemulla, R. On synopses for distinct-value estimation under multiset operations. In *Proc. ACM SIGMOD* (2007), 199–210.
4. Brown, P.G., Haas, P.J. Techniques for warehousing of sample data. In *Proc. ICDE* (2006).
5. Cohen, E., Kaplan, H. Tighter estimation using bottom k sketches. *Proc. VLDB Endow. 1*, 1 (2008), 213–224.
6. Dasu, T., Johnson, T., Muthukrishnan, S., Shkapenyuk, V. Mining database structure; or, how to build a data quality browser. In *Proc. ACM SIGMOD* (2002), 240–251.
7. Duffield, N., Lund, C., Thorup, M. Priority sampling for estimation of arbitrary subset sums. *J. ACM 54*, 6 (2007), 32.
8. Durand, M., Flajolet, P. Loglog counting of large cardinalities. In *Proc. ESA* (2003), 605–617.
9. Flajolet, P., Martin, G.N. Probabilistic counting algorithms for data base applications. *J. Comp. Sys. Sci. 31* (1985),182–209.
10. Ganguly, S., Garofalakis, M., Rastogi, R. Tracking set-expression cardinalities over continuous update streams. *VLDB J. 13* (2004), 354–369.
11. Gemulla, R. *Sampling Algorithms for Evolving Datasets.* Ph.D. thesis, TU Dresden, Dept. of CS, 2008. http://nbn-resolving.de/urn:nbn:de:bsz:14-ds-1224861856184-11644.
12. Gemulla, R., Lehner, W. Sampling time-based sliding windows in bounded space. In *Proc. SIGMOD* (2008), 379–392.
13. Gibbons, P. Distinct-values estimation over data streams. In M. Garofalakis, J. Gehrke, and R. Rastogi, editors, *Data Stream Management: Processing High-Speed Data Streams*. Springer, 2009. To appear.
14. Haas, P.J., Stokes, L. Estimating the number of classes in a finite population. *J. Amer. Statist. Assoc. 93* (1998), 1475–1487.
15. Hadjieleftheriou, M., Yu, X., Koudas, N., Srivastava, D. Hashed samples: selectivity estimators for set similarity selection queries. *Proc. VLDB Endow. 1*, 1 (2008), 201–212.
16. Motwani, R., Raghavan, P. *Randomized Algorithms*. Cambridge University Press (1995).
17. Serfling, R.J. *Approximation Theorems of Mathematical Statistics*. Wiley, New York (1980).
18. Shukla, A., Deshpande, P., Naughton, J.F., Ramasamy, K. Storage estimation for multidimensional aggregates in the presence of hierarchies. In *Proc. VLDB* (1996), 522–531.
19. Simitsis, A., Baid, A., Sismanis, Y., Reinwald, B. Multidimensional content eXploration. *Proc. VLDB Endow. 1*, 1 (2008), 660–671.
20. Szegedy, M. The DLT priority sampling is essentially optimal. In *Proc. STOC* (2006), 150–158.
21. Whang, K., Vander-Zanden, B.T., Taylor, H.M. A linear-time probabilistic counting algorithm for database applications. *ACM Trans. Database Sys. 15* (1990), 208–229.
22. Zimmer, C., Tryfonopoulos, C., Weikum, G. Exploiting correlated keywords to improve approximate information filtering. In *Proc. SIGIR* (2008), 323–330.

**Kevin Beyer, Rainer Gemulla, Peter J. Haas, Berthold Reinwald, and Yannis Sismanis** ({kbeyer,rgemull,phaas,reinwald, syannis}@us.ibm.com),
IBM Almaden Research Center,
San Jose, CA.