Structure-Aware Sampling: Flexible and Accurate Summarization

Edith Cohen, Graham Cormode, Nick Duffield
AT&T Labs–Research
180 Park Avenue
Florham Park, NJ 07932, USA
{edith,graham,duffield}@research.att.com

ABSTRACT

In processing large quantities of data, a fundamental problem is to obtain a summary which supports approximate query answering. Random sampling yields flexible summaries which naturally support subset-sum queries with unbiased estimators and well-understood confidence bounds. Classic sample-based summaries, however, are designed for arbitrary subset queries and are oblivious to the structure in the set of keys. The particular structure, such as hierarchy, order, or product space (multi-dimensional), makes *range queries* much more relevant for most analysis of the data.

Dedicated summarization algorithms for range-sum queries have also been extensively studied. They can outperform existing sampling schemes in terms of accuracy on range queries per summary size. Their accuracy, however, rapidly degrades when, as is often the case, the query spans multiple ranges. They are also less flexible—being targeted for range sum queries alone—and are often quite costly to build and use.

In this paper we propose and evaluate variance optimal sampling schemes that are *structure-aware*. These summaries improve over the accuracy of existing *structure-oblivious* sampling schemes on range queries while retaining the benefits of sample-based summaries: flexible summaries, with high accuracy on both range queries and arbitrary subset queries.

1. INTRODUCTION

Consider a scenario where a large volume of data is collected on a daily basis: for example, sales records in a retailer, or network activity in a telecoms company. This activity will be archived in a warehouse or other storage mechanism, but the size of the data is too large for data analysts to keep in memory. Rather than go out to the full archive for every query, it is natural to retain accurate summaries of each data table, and use these queries for data exploration and analysis, reducing the need to read through the full history for each query. Since there can be many tables (say, one for every day at each store, in the retailer case, or one for every hour and every router in the network case), we want to keep a very compact summary of each table, but still guarantee accurate answers to any query. The summary allows approximate processing of queries, in place of the original data (which may be slow to access or even no longer available); it also allows fast 'previews' of computations which are slow or resource hungry to perform exactly.

Proceedings of the VLDB Endowment, Vol. 4, No. 11

EXAMPLE 1. *As a motivating example, consider network data in the form of IP flow records. Each record has a source and destination IP address, a port number, and size (number of bytes). IP addresses form a natural* hierarchy*, where prefixes or sets of prefixes define the ranges of interest. Port numbers indicate the generating application, and related applications use* ranges *of port numbers. Summaries of IP flows are used for many network management tasks, including planning routing strategies, and traffic anomaly detection. Typical ad hoc analysis tasks may involve estimating the amount of traffic between different subnetworks, or the fraction of VoIP traffic on a certain network. Resources for collection, transport, storage and analysis of network measurements are expensive; therefore, accurate summaries are needed by network operators to understand the behavior of their network.*

Such scenarios have motivated a wealth of work on data summarization and approximation. There are two main themes: methods based on random sampling, and algorithms that build more complex summaries (often deterministic, but also randomized). Both have their pros and cons. Sampling is fast and efficient, and has useful guaranteed properties. Dedicated summaries can offer greater accuracy for the kind of range queries which are most common over large data, albeit at a greater cost to compute, and providing less flexibility for other query types. Our goal in this work is to provide summaries which combine the best of both worlds: fast, flexible summaries which are very accurate for the all-important range queries. To attain this goal, we must understand existing methods in detail to see how to improve on their properties.

Summaries which are based on random sampling allow us to build (unbiased) estimates of properties of the data set, such as counts of individual identifiers ("keys"), sums of weights for particular subsets of keys, and so on, all specified after the data has been seen. Having high-quality estimates of these primitives allows us to implement higher-level applications over samples, such as computing order statistics over subsets of the data, heavy hitters detection, longitudinal studies of trends and correlations, and so on.

Summarization of items with weights traditionally uses Poisson sampling, where each item is sampled independently. The approach which sets the probability of including an item in the sample to be proportional to its weight (IPPS) [12] enables us to use the Horvitz-Thompson estimator [15], which minimizes the sum of per-item variances. "VAROPT" samples [3, 26, 7] improve on Poisson samples in that the sample size is fixed and they are more accurate on subset-sum queries. In particular VAROPT samples have *variance optimality*: they achieve variance over the queries that is provably the smallest possible for any sample of that size.

Since sampling is simple to implement and flexible to use, it is the default summarization method for large data sets. Samples support a rich class of possible queries directly, such as those men-

tioned in Example 1: evaluating the query over the sampled data (with appropriately scaled weights) usually provides an unbiased, low variance estimate of the true answer, while not requiring any new code to be written. These summaries provide not only estimates of aggregate values but also a representative sample of keys that satisfy a selection criteria. The fact that estimates are unbiased also means that relative error decreases for queries that span multiple samples or larger subsets and the estimation error is governed by exponential tail bounds: the estimation error, in terms of the number of samples from any particular subset, is highly concentrated around the square root of the expected number of samples.

We observe, however, that traditionally sampling has neglected the inherent structure that is present, and which is known before the data is observed. That is, data typically exists within a well-understood schema that exhibits considerable structure. Common structures include *order* where there is a natural ordering over keys; *hierarchy* where keys are leaves within a hierarchy (e.g. geographic hierarchy, network hierarchy); and combinations of these where keys are multi-dimensional points in a *product structure*. Over such data, queries are often *structure-respecting*. For example, on ordered data with n possible key-values, although there are $O(2^n)$ possible subset-sum queries, the most relevant queries may be the $O(n^2)$ possible range queries. In a hierarchy, relevant queries may correspond to particular nodes in the hierarchy (geographic areas, IP address prefixes), which represent $O(n \log n)$ possible ranges. In a product structure, likely queries are boxes—intersections of ranges of each dimension. This is observed in Example 1: the queries mentioned are based on the network hierarchy.

While samples have been shown to work very well for queries which resemble the sums of *arbitrary* subsets of keys, they tend to be less satisfactory when restricted to range queries. Given the same summary size, samples can be out-performed in accuracy by dedicated methods such as (multi-dimensional) histograms [11, 20, 16], wavelet transforms [17, 28], and geometric summaries [24, 14, 1, 9, 29] including the popular Q-digest [22].

These dedicated summaries, however, have inherent drawbacks: they primarily support queries that are sum aggregates over the original weights, and so other queries must be expressed in terms of this primitive. Their accuracy rapidly degrades when the query spans multiple ranges—a limitation since natural queries may span several (time, geographic) ranges within the same summary and across multiple summaries. Dedicated summaries do not provide "representative" keys of selected subsets, and require changes to existing code to utilize. Of most concern is that they can be very slow to compute, requiring a lot of I/O (especially as the dimensionality of the data grows): a method which gives a highly accurate summary of each hour's data is of little use if it takes a day to build! Lastly, the quality of the summary may rely on certain structure being present in the data, which is not always the case. While these summaries have shown their value in efficiently summarizing one-dimensional data (essentially, arrays of counts), their behavior on even two-dimensional data is less satisfying: troubling since this is where accurate summaries are most needed. For example, in the network data example, we are often interested in the traffic volume between (collections of) various source and destination ranges.

Motivated by the limitations of dedicated summaries, and the potential for improvement over existing (structure-oblivious) sampling schemes, we aim to design sampling schemes that are both VAROPT and *structure-aware*. At the same time, we aim to match the accuracy of deterministic summaries on range sum queries and retain the desirable properties of existing sample-based summaries: unbiasedness, tail bounds on arbitrary subset-sums, flexibility and support for representative samples, and good I/O performance.

1.1 Our Contributions

We introduce a novel algorithmic sampling framework, which we refer to as *probabilistic aggregation*, for deriving VAROPT samples. This framework makes explicit the freedom of choice in building a VAROPT summary which has previously been overlooked. Working within this framework, we design *structure-aware* VAROPT sampling schemes which exploit this freedom to be much more accurate on ranges than their structure-oblivious counterparts.

- For hierarchies, we design an efficient algorithm that constructs VAROPT summaries with bounded "range discrepancy". That is, for any range, the number of samples deviates from the expectation by less than 1. This scheme has the minimum possible variance on ranges of any unbiased sample-based summary.

- For ordered sets, where the ranges consist of all intervals, we provide a sampling algorithm which builds a VAROPT summary with range discrepancy less than 2. We prove that this is the best possible for any VAROPT sample.

- For d-dimensional datasets, we propose sampling algorithms where the discrepancy between $p(R)$, the expected number of sample points in the range R, and the actual number is $O(\min\{ds^{\frac{d-1}{2d}}, \sqrt{p(R)}\})$, where s is the sample size.

This improves over structure-oblivious random sampling, where the corresponding discrepancy is $O(\sqrt{p(R)})$.

Discrepany corresponds to the error of range-sum queries, but sampling has an advantage over other summaries with similar error bounds: The error on queries Q which span multiple ranges grows linearly with the number of ranges for other summaries but has square root dependence for samples. Moreover, for samples the expected error never exceeds $O(\sqrt{p(Q)})$ (in expectation) regardless of the number of ranges.

Construction Cost. For a summary structure to be effective, it must be possible to construct quickly, and with small space requirements. Our main-memory sampling algorithms perform tasks such as sorting keys or (for multidimensional data) building a kd-tree. We propose even cheaper alternatives which perform two read-only passes over the dataset using memory that depends on the desired summary size s (and is independent of the size of the data set). When the available memory is $O(s \log s)$, we obtain a VAROPT sample that with high probability $1 - O(1/\text{poly}s)$ is close in quality to the algorithms which store and manipulate the full data set.

Empirical study. To demonstrate the value of our new structure-aware sampling algorithms, we perform experiments comparing to popular summaries, in particular the wavelet transform [28], ϵ-approximations [14], randomized sketches [4] and to structure-oblivious random sampling. These experiments show that it is possible to have the best of both worlds: summaries with equal or better accuracy than the best-in-class, which are flexible and dramatically more efficient to construct and work with.

2. PROBABILISTIC AGGREGATION

This section introduces the "probabilistic aggregation" technique. For more background, see the review of core concepts from sampling and summarization in Appendix A.

Our data is modeled as a set of (key, weight) pairs: each key $i \in K$ has weight $w_i \geq 0$. A sample is a random subset $S \subset K$. A sampling scheme is IPPS when, for expected sample size s and derived threshold τ_s, the sample includes key i with probability $\min\{w_i/\tau_s, 1\}$. IPPS can be achieved with *Poisson* sampling (by

Algorithm 1 PAIR-AGGREGATE(p, i, j)

Require: $0 < p_i, p_j < 1$

if $p_i + p_j < 1$ **then**
  if $rand() < \frac{p_i}{p_i+p_j}$ **then** $p_i \leftarrow p_i + p_j$; $p_j \leftarrow 0$
  else $p_j \leftarrow p_i + p_j$; $p_i \leftarrow 0$
else ▷ $p_i + p_j \geq 1$
  if $rand() < \frac{1-p_j}{2-p_i-p_j}$ **then** $p_i \leftarrow 1$; $p_j \leftarrow p_i + p_j - 1$
  else $p_i \leftarrow p_i + p_j - 1$; $p_j \leftarrow 1$ ▷ w/prob $\frac{1-p_i}{2-p_i-p_j}$
return p

including keys independently) or VAROPT sampling, which allows correlations between key inclusions to achieve improved variance and fixed sample size of exactly s. There is not a unique VAROPT sampling scheme, but rather there is a large family of VAROPT sampling distributions: the well-known "reservoir sampling" is a special case of VAROPT on a data stream with uniform weights. Classic tail bounds, including Chernoff bounds, apply both to VAROPT and Poisson sampling.

Structure is specified as a *range space* $(\mathcal{K}, \mathcal{R})$ with $\mathcal{K}$ being the key domain and *ranges* $\mathcal{R}$ that are subsets of $\mathcal{K}$. The *discrepancy* $\Delta(S, R)$ of a sample S on a range $R \in \mathcal{R}$ is the difference between the number of sampled keys $S \cap R$ and its expectation $p(R)$. We use Δ to denote the maximum discrepancy over all ranges $\mathcal{R}$. Discrepancy Δ means that the error of range-sum queries is at most $\Delta\tau_s$. If a sample is Poisson or VAROPT, it follows from Chernoff bounds that the expected discrepancy is $O(\sqrt{p(R)})$ and (from bounded VC dimension of our range spaces) that the maximum range discrepancy is $O(\sqrt{s \log s})$ with probability $O(1 - \text{poly}(1/s))$. With structure-aware sampling, we aim for much lower discrepancy.

Defining probabilistic aggregation. Let p be the vector of sampling probabilities. We can view a sampling scheme that picks a set of keys S as operating on p. Vector p is incrementally modified: setting p_i to 1 means i is included in the sample, while $p_i = 0$ means it is omitted. When all entries are set to 0 or 1, the sample is chosen (e.g. Poisson sampling independently sets each entry to 1 with probability p_i). To ensure a VAROPT sample, the current vector p' must be a *probabilistic aggregate* of the original p.

A random vector $p^{(1)} \in [0,1]^n$ is a *probabilistic aggregate* of a vector $p^{(0)} \in [0,1]^n$ if the following conditions are satisfied:

(i) (*Agreement in Expectation*) $\forall i, \mathsf{E}[p_i^{(1)}] = p_i^{(0)}$,

(ii) (*Agreement in Sum*) $\sum_i p_i^{(1)} = \sum_i p_i^{(0)}$, and

(iii) (*Inclusion-Exclusion Bounds*)

$$\text{(I):} \quad \mathsf{E}[\prod_{i \in J} p_i^{(1)}] \leq \prod_{i \in J} p_i^{(0)}$$

$$\text{(E):} \quad \mathsf{E}[\prod_{i \in J} (1 - p_i^{(1)})] \leq \prod_{i \in J} (1 - p_i^{(0)}) \,.$$

We obtain VAROPT samples by performing a sequence of probabilistic aggregations, each setting at least one of the probabilities to 1 or 0. In Appendix B we show that probablistic aggregations are transitive and that set entries remain set. Thus, such a process must terminate with a VAROPT sample.

Pair Aggregation. Our summarization algorithms perform a sequence of simple aggregation steps which we refer to as *pair aggregations* (Algorithm 1). Each pair aggregation step modifies only two entries and sets at least one of them to $\{0, 1\}$. The input to *pair aggregation* is a vector p and a pair i, j with each $p_i, p_j \in (0, 1)$. The output vector agrees with p on all entries except i, j and one of the entries i, j is set to 0 or 1. It is not hard to verify, separately considering cases $p_i + p_j < 1$ and $p_i + p_j \geq 1$, that PAIR-AGGREGATE(p, i, j) correctly computes a probabilistic aggregate of its input, and hence the sample is VAROPT.

Pair aggregation is a powerful primitive. It produces a sample of size exactly $s = \sum_i p_i^{(0)}$ [1]. Observe that the choice of which pair i, j to aggregate at any point can be arbitrary—and the result is still a VAROPT sample. This observation is what enables our approach. We harness this freedom in pair selection to obtain VAROPT samples that are structure aware: Intuitively, by choosing to aggregate pairs that are "close" to each other with respect to the structure, we control the range impact of the "movement" of probability mass.

3. ONE DIMENSIONAL STRUCTURES

We use pair aggregation to make sampling structure-aware by describing ways to pick which pair of items to aggregate at each step. For now, we assume the data fits in main memory, and our input is the list of keys and their associated IPPS probabilities p_i. We later discuss the case when the data exceeds the available memory.

For hierarchy structures (keys $\mathcal{K}$ are associated with leaves of a tree and $\mathcal{R}$ contains all sets of keys under some internal node) we show how to obtain VAROPT samples with (optimal) maximum range discrepancy $\Delta < 1$. There are two special cases of hierarchies: (i) *disjoint ranges* (where $\mathcal{R}$ is a partition of $\mathcal{K}$)—captured by a flat 2-level hierarchy with parent nodes corresponding to ranges and (ii) *order* where there is a linear order on keys and $\mathcal{R}$ is the set of all prefixes—the corresponding hierarchy is a path with single leaf below each internal node. For order structures where $\mathcal{R}$ is the set of "intervals" (all consecutive sets of keys) we show that there is always a VAROPT sample with maximum range discrepancy $\Delta < 2$ and prove that this is the best possible.

- **Disjoint ranges:** Pair selection picks pairs where both keys belong to the same range R. When there are multiple choices, we may choose one arbitrarily. Only when there are none do we select a pair that spans two different ranges (arbitrarily if there are multiple choices).
- **Hierarchy:** Pair selection picks pairs with lowest LCA (lowest common ancestor). That is, we pair aggregate (i, j) if there are no other pairs with an LCA that is a descendant of LCA(i, j).

Following these rules guarantees low range discrepancy: they ensure that for all ranges $R \in \mathcal{R}$ and for all iterations h where R has at least one entry which is not set, we have $\sum_{i \in R} p_i^{(h)} \equiv \sum_{i \in R} p_i^{(0)}$. So, at termination, when all entries in R are set:

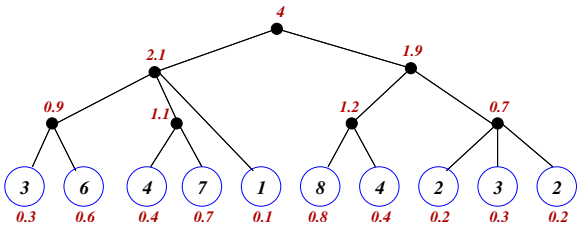
$$|S \cap R| \in \{\lfloor \textstyle\sum_{i \in R} p_i^{(0)} \rfloor, \lceil \sum_{i \in R} p_i^{(0)} \rceil\}.$$

Hence, the maximum range discrepancy is $\Delta < 1$. In Appendix C we bound the discrepancy of multi-range queries.

EXAMPLE 2. *Figure 1 demonstrates sampling over a hierarchical space. The input provides a weight associated with each input item (key). Each key corresponds to a leaf node which shows its weight and IPPS sampling probability for sample size $s = 4$. Each internal tree node shows the expected number of samples under it.*

All VAROPT *samples include exactly 4 keys but if not structure aware then the number of samples under internal nodes can significantly deviate from their expectation. A Poisson IPPS sample has 4 keys in expectation, and is oblivious to structure.*

[1] Assuming that $\sum_i p_i^{(0)}$ is integral. This can be ensured (deterministically) by choosing τ as described in Algorithm 4.



leaf	1	2	3	4	5	6	7	8	9	10
IPPS	0.3	0.6	0.4	0.7	0.1	0.8	0.4	0.2	0.3	0.2
(1)+(2),(3)+(4), (6)+(7),(8)+(9)	0	0.9	1	0.1	0.1	0.2	1	0.5	0	0.2
(2)+(4), (8)+(10)	0	1	1	0	0.1	0.2	1	0	0	0.7
(6)+(10)	0	1	1	0	0.1	0	1	0	0	0.9
(5)+(10)	0	1	1	0	0	0	1	0	0	1

Figure 1: Sampling over a hierarchy structure.

The table shows a sequence of pairwise aggregations which follows the hierarchy pair selection rule. The result is a structure-aware VarOpt *sample, consisting of the leaves $S = \{2, 3, 7, 10\}$. One can verify that the number of samples under each internal nodes is indeed the floor or ceiling of its expectation.*

- **Order structures:** In Appendix D we establish the following:

THEOREM 1. *For the order structure (all intervals of ordered keys), (i) there always exists a* VarOpt *sample distribution with maximum range discrepancy $\Delta \leq 2$. (ii) For any fixed $\Delta < 2$, there exist inputs for which a* VarOpt *sample distribution with maximum range discrepancy $\leq \Delta$ does not exist.*

4. PRODUCT STRUCTURES

We now consider summarizing d-dimensional data, where the key structure projected on each axis is an order or a hierarchy. Ranges are axis parallel hyper rectangles: a product of one-dimensional key ranges (order) and/or internal nodes of a hierarchy.

We develop a VarOpt sampling algorithm where the discrepancy on a range R is that of a (structure oblivious) VarOpt sample on a subset with $\mu \leq \min\{p(R), 2ds^{\frac{d-1}{d}}\}$. Hence, the estimation error is subject to tail bounds (2) and (3) with this value of μ and concentrated around $\sqrt{\mu} \leq \min\{\sqrt{p(R)}, \sqrt{2d}s^{\frac{d-1}{2d}}\}$.

As in the one-dimensional case, the intuition behind our approach is to limit range discrepancy by preferring pairwise aggregations that result in "localized" movement of "probability mass."

Uniform case. We start with the special case of a uniform distribution over a d-dimensional hyper cube with measure $s = h^d$. Our algorithm partitions the hypercube into s unit cells and selects the sample by independently picking a single point uniformly from each unit cell. The resulting sample S is a VarOpt sample (of size s) of the uniform hypercube. For analysis, observe that any axis-parallel hyperplane intersects at most $h^{d-1} = s^{\frac{d-1}{d}}$ unit cells. Therefore, any axis-parallel box query R intersects at most $2ds^{(d-1)/d}$ cells that are not contained in R. The only error in our estimation comes from these "boundary" cells which we denote $B(R)$: all other cells are either fully inside or fully outside R, and so do not contribute to the discrepancy. We map each boundary cell $C \in B$ to a 0/1 random variable which is 1 with probability proportional to the size of the overlap, $|C \cap R|$. These random variable are independent Poisson with $\mu = \sum_{C \in B} |C \cap R| \leq \min\{p(R), |B(R)|\}$, and so the tail bounds hold.

Algorithm 2 KD-HIERARCHY$(depth, key_set)$

if |key_set| $= 1$ **then**
  h.val $\leftarrow$ key_set
  h.left $\leftarrow$ null; *h.right* $\leftarrow$ null;
  return h ▷ kd-hierarchy h is a leaf node
else
  h.val $\leftarrow$ null
  a $\leftarrow$ depth mod d ▷ axis on which partition is made
  if axis a has an order structure **then**
    $m \leftarrow \arg\min_m \left|\sum_{i|key_a(i)\leq m} p_i - \sum_{i|key_a(i)>m} p_i\right|$
    ▷ m is weighted median of key_set ordered on axis a
    left_set $\leftarrow \{i|key_a(i) \leq m\}$;
    right_set $\leftarrow \{i|key_a(i) > m\}$;
  else ▷ axis a has a hierarchy structure H_a
    Find the partition of key_set into left_set and right_set over all linearizations of the hierarchy to minimize $\left|\sum_{i\in\text{left\_set}} p_i - \sum_{i\in\text{right\_set}} p_i\right|$
  h.left$\leftarrow$ KD-HIERARCHY$(depth+1,$ left_set$)$
  h.right$\leftarrow$ KD-HIERARCHY$(depth+1,$ right_set$)$
  return h

General Case. In general, the probability mass is not distributed uniformly throughout the space as in the previous case. So instead, we seek to build a partition of the space into regions so that the probability mass is (approximately) equal. In particular, we consider using kd-trees to obtain a partition into cells containing keys whose sum of probabilities is $\Theta(1)$ (in general it is not possible to partition the discrete probabilities to sum to exactly 1). Choosing kd-trees means that every axis-parallel hyperplane intersects $O(s^{\frac{d-1}{d}})$ cells. Since cells are not exact units, we have to carefully account for aggregations of the "leftover" probabilities.

Let K be the input set of weighted d-dimensional keys. Then:

- Compute IPPS inclusion probabilities for K and set aside all keys with $p_i = 1$ (they must all be included in the summary). Hence, wlog, we have that all keys in K have $p_i < 1$.
- Compute a hierarchy T over the (multidimensional) keys in K: $T \leftarrow$ KDHIERARCHY$(0, K)$.
- Apply the hierarchy summarization algorithm (Section 3) to T.

Algorithm KD-HIERARCHY builds a kd-tree, splitting on each dimension in turn. At each internal node we select a hyperplane perpendicular to the current axis that partitions the probability weight in half (or as equally as possible.) Each leaf of the tree then corresponds to an undivided rectangle containing approximately unit probability mass. Analysis and examples are given in Appendix E.

5. I/O EFFICIENT SAMPLING

The algorithms presented in previous sections assume that we can hold the full data set in memory to generate the summary. As data sets grow, we require summarization methods that are more I/O efficient. In particular, the reliance on being able to sort data, locate data in hierarchies, and build kd-trees over the whole data may not be realistic for large data sets. In this section, we present I/O efficient alternatives that generate a structure-aware VarOpt sample while only slightly compromising on range discrepancy with respect to the main-memory variants. The intuition behind our approach is that a structure-oblivious VarOpt sample of sufficient

Algorithm 3 IO-AGGREGATE(i)

Process key i:

$p_i \leftarrow \min\{1, w_i/\tau_s\}$ ▷ IPPS sampling prob
if $p_i = 1$ **then**
  $S \leftarrow S \cup \{i\}$ ▷ i is placed in the sample
else ▷ $p_i < 1$
  $L \leftarrow L(i)$ ▷ L is the cell that contains i
  if $a(L) = \emptyset$ **then** ▷ Cell L has no key with $p_a \in \{0, 1\}$)
    $a(L) \leftarrow i$ ▷ i becomes the active key of its cell
  else ▷ L has an existing active key
    $a \leftarrow a(L)$
    PAIR-AGGREGATE(p, i, a) ▷ One of p_i, p_a is set
    $a(L) \leftarrow \emptyset$
    if $p_a = 1$ **then**
      $S \leftarrow S \cup \{a\}$ ▷ a is placed in the sample
    if $0 < p_a < 1$ **then**
      $a(L) \leftarrow a$ ▷ a remains the active key of L
    if $p_i = 1$ **then**
      $S \leftarrow S \cup \{i\}$ ▷ i is placed in the sample
    if $0 < p_i < 1$ **then**
      $a(L) \leftarrow i$ ▷ i becomes the active key of L

size $\tilde{O}(s)$ is useful to guide the construction of a structure-aware summary because with high probability it hits *all* sufficiently large ranges (those with $p(R) \geq 1$) (In geometric terms, it forms an ϵ-net of the range space [13]). Once built, the summary can be kept in fast-access storage while the original data is archived or deleted.

Description. Our algorithms perform two read-only streaming passes over the (unsorted) input dataset. When using memory of size $\tilde{O}(s)$ (where s is the desired sample size), the range discrepancy is similar to that of the main memory algorithm with high probability. In the first pass we compute a random sample S' of size $s' > s$ using memory s' via Poisson IPPS or stream VAROPT sampling (reservoir sampling if keys have uniform weights). We also compute the IPPS threshold value τ_s for a sample of size s (described in Appendix A). After completion of the first pass (in main memory) we use S' to construct a partition $\mathcal{L}$ of the key domain. The partition has the property that with high probability $p(L) \leq 1$ for all $L \in \mathcal{L}$.

In the second pass, we incrementally build the sample S, initialized to $\emptyset$. We perform probabilistic aggregations, guided by the partition $\mathcal{L}$, using IPPS probabilities for a sample of size s. We maintain at most one *active key* $a(L)$ for each cell $L \in \mathcal{L}$, which is initialized to null. Each key i is processed using IO-AGGREGATE (Algorithm 3): if $p = \min\{1, w_i/\tau_s\} = 1$, then i is added to S. Otherwise, if there is no active key in the cell $L(i)$ of i, then i becomes the active key. If there is an active key a, we PAIR-AGGREGATE i and a. If the updated p value of one of them becomes 1, we include this key in the sample S. The key with p value in $(0, 1)$ (if there is one) is the new active key for the cell. The storage used is $O(s + |\mathcal{L}|)$, since we maintain at most one active key for each part and the number of keys included in S is at most s. Finally, after completing the pass, we PAIR-AGGREGATE the $\leq |\mathcal{L}|$ active keys, placing all keys with final $p_i = 1$ in S.

Partition and aggregation choices. The specifics of the main-memory operations—the construction of the partition and the final aggregation of active keys—depend on the range structure. We start with product structures and then refine the construction to obtain stronger results for one-dimensional structures. Note that keys in S' with $\min\{1, w_i/\tau_s\} = 1$ must be included in S. Moreover, S' must include all such keys—as S' includes all keys with $w_i \geq \tau_{s'} < \tau_s$, it therefore includes all keys with $w_i \geq \tau_s$. These keys can thus be excluded from consideration after the first phase.

Product structures: We compute $h \leftarrow$ KD-HIERARCHY$(0, S')$ (for S' with all keys with $w_i \geq \tau_s$ removed). This hierarchy h induces a partition of the key domain according to the splitting criteria in each node. The partition $\mathcal{L}$ corresponds to the leaves of h. The aggregation of active keys in the final phase follows the hierarchy h (as in Section 3).

Disjoint ranges: There is a cell in the partition for every range from $\mathcal{R}$ that contains a key from S'. We then induce an arbitrary order over the ranges and put a cell for each union of ranges in $\mathcal{R}$ which lies between two consecutive ranges represented in the sample. In total we obtain $2s'$ cells. In the final phase, active keys can be aggregated arbitrarily.

When $s' = \Omega(s \log s)$, with probability $1 - \text{poly}(1/s)$, all ranges of size ≥ 1 are intersected by S' and each cell L that is a union of ranges not seen in S' has size at most 1 (thus, each range R with probability mass $p(R) < 1$ can obtain at most one sample in S, and so will not be over-represented in the final sample). Thus, the maximum discrepancy is $\Delta < 1$ with probability $1 - \text{poly}(1/s)$.

Order: We sort S' according to the order (excluding keys with $w_i \geq \tau_s$). If $i_1, \ldots, i_t$ are the remaining keys in sorted order, there is a cell L for each subinterval $(i_j, i_{j+1}]$ and two more, one for all keys $\leq i_1$ and the other for keys $> i_t$. The final aggregation of active keys follows the main-memory aggregation of ordered keys.

When $s' = \Omega(s \log s)$, with high probability, the maximum probability distance between consecutive keys is 1 and therefore, the maximum range discrepancy is $\Delta < 2$.

Hierarchy: A natural solution is to linearize the hierarchy, i.e. generate a total order consistent with the hierarchy, and then apply the algorithm for this order structure. This obtains $\Delta < 2$ with high probability. Alternatively, we can select all ancestors in the hierarchy of keys in S and form a partition by matching each key to its lowest selected ancestor. This will give us maximum range discrepancy $\Delta < 1$ with high probability. The number of ancestors, however, can be large and therefore this approach is best for shallow hierarchies.

6. EXPERIMENTAL STUDY

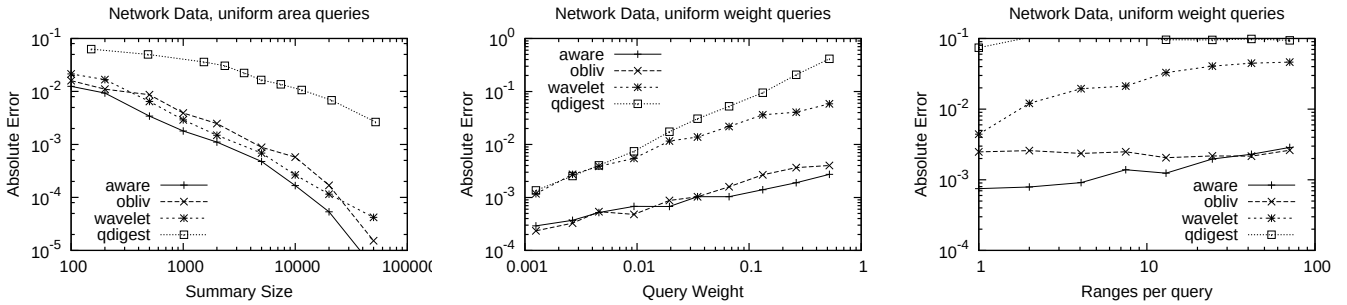
We conducted an experimental study of our methods, and compared them with existing approaches to summarizing large data sets. We consider three performance aspects: building the summary, query processing, and query accuracy. We vary the weight of queries and the number of ranges in each query.

6.1 Experimental Environment

Data Sets and Query Sets. We compared our approach on a variety of data sets, and present results on two examples:

The *Network* dataset consists of network flow data summarizing traffic volumes exchanged between a large number of sources and destination, observed at a network peering point. Each of the 196K input tuples gives the total volume associated with a (source, destination) pair. In total, there are 63K distinct sources and 50K distinct destinations. The keys are drawn from the two-dimensional IP hierarchy, i.e. X=2^{32} and Y=2^{32} (i.e., a product of hierarchies).

The *Technical Ticket* data is derived from calls to a customer care center of a broadband wireline access network that resulted in a technical trouble ticket. Each key consists of: (i) an anonymous



(a) Accuracy vs Space on Network (b) Accuracy vs Query weight on Network (c) Accuracy vs Ranges per Query on Network

Figure 2: Experimental results on Network Data set.

identifier for unique customers; (ii) a trouble code, representing a point in a hierarchy determining the nature of the problem identified from a predetermined set by the customer care staff; and (iii) a network code, indicating points on the network path to the customer's location. Both attributes are hierarchical with varying branching factor at each level, representing a total of approximately 2^{24} possibilities in each dimension, i.e. $X = 2^{24}$ and $Y = 2^{24}$. There are 4.8K distinct trouble codes present in the data, 80K distinct network locations, and 500K distinct observed combinations.

Each query is produced as a collection of non-overlapping rectangles in the data space. To study the behavior of different summaries over different conditions, we generated a variety of queries of two types. In the *uniform area* case, each rectangle is placed randomly, with height and width chosen uniformly within a range $[0, h]$ and $[0, w]$. We tested a variety of scales to determine h and w, varying these from covering almost the whole space, to covering only a very small fraction of the data (down to a 10^{-4} fraction). In the *uniform weight* case, each rectangle is chosen to cover roughly the same fraction of the total weight of keys. We implement this by building a kd-tree over the whole data, and picking cells from the same level (note, this is independent of any kd-tree built over sampled data by our sampling methods). For each query, we compute the exact range sum over the data, and compare the absolute, sum-squared and relative errors of our methods across a collection of queries. In our plots below, we show results from a battery of 50 queries with varying number of rectangles.

Methods. We compared structure-aware sampling to best-in-class examples of the various approaches described in Appendix A:

Obliv, is a structure-oblivious sampling method. We implemented VAROPT sampling to give a sample size of exactly s.

Aware, the structure aware sampling method. We follow the 2 pass algorithm (Section 5 and Section 4), and first draw a sample of size several times larger than s, the target size (in our experiments, we set $s' = 5s$: increasing the factor did not significantly improve the accuracy). We then built the kd-tree on this sample, and took a second pass over the data to populate the tree. Lastly, we perform pair aggregation within the tree structure to produce a sample of size exactly s. Although the algorithm is more involved than straight VAROPT, both are implemented in fewer than 200 lines of code.

Wavelet, implements the (two-dimensional) standard Haar wavelet scheme. In a single pass we compute the full wavelet transform of the data: each input data point contributes to $\log X \cdot \log Y$ wavelet coefficients. We then prune these coefficients to retain only the s largest (normalized) coefficients for query answering.

Qdigest, implements the (two-dimensional) q-digest data structure [14]. This deterministically builds a summary of the data. Given a parameter ϵ, the structure is promised to be at worst $O(\frac{1}{\epsilon^2} \log X \cdot \log Y)$, but in practice materializes much fewer nodes than this, so we count the number of materialized nodes as its size.

Sketch, implements the Count-sketch, a randomized summary of the data [4]. Each input item contributes to $O(\log X \cdot \log Y)$ sketches of dyadic rectangles. We choose the size of each sketch so that the total size is bounded by a parameter s.

Our implementations in Python were run on the same machine, on a 2.4GHz core with access to 4GB of memory. For most methods, we perform static memory allocation as a function of summary size in advance. The exception is wavelets, which needs a lot of space to build the transform before pruning.

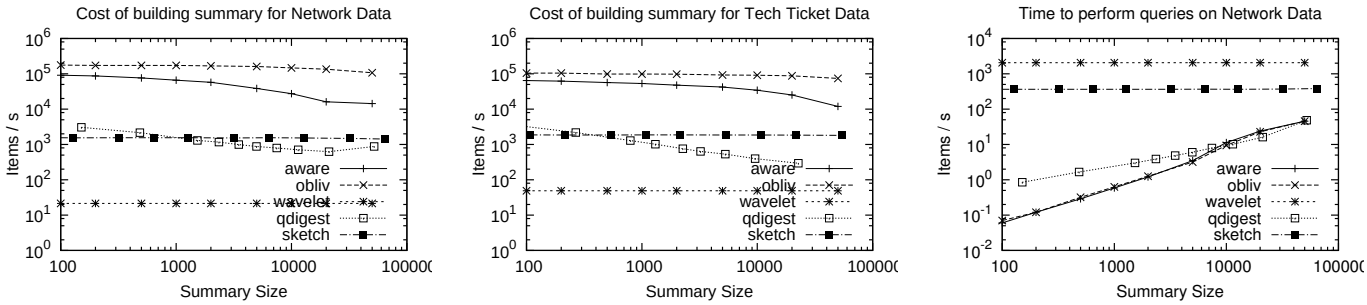
6.2 Network Data Accuracy

Figure 2 shows accuracy results on network data. The y-axis shows accuracy measured as the error in the query results divided by the total weight of all data (the absolute error). Our experiments which computed other metrics such as sum-squared error showed the same trends, and so are omitted.

On this data, the structure-aware sampling typically achieved the least error. Figure 2(a) shows this behavior across a variety of summary sizes with uniform area queries each containing 25 ranges. For comparison, we measure the space used by each summary in terms of elements on the original data, so in this case the smallest sample contains 100 IP address pairs (and weights). This is compared to keeping the 100 largest wavelet coefficients, and a q-digest of 100 nodes. We also compared to sketch summaries with an equivalent amount of space. However, the space required before a sketch of two-dimensional data becomes accurate is much larger than for the other summaries considered. The total error for the queries shown was off the scale on the graphs, so we omit sketches from further accuracy comparison.

Across most summary sizes, the structure-aware sampling is several times more accurate than its structure-oblivious counterpart: Figure 2(a), which is in log-scale on both axes, shows that the error of the structure-aware method is half to a third that of the oblivious method given the same space: a significant improvement. The deterministic q-digest structure is one to two orders of magnitude less accurate in the same space. Only the Haar wavelet comes close to structure-aware sampling in terms of accuracy. This is partly due to the nature of uniform area queries: on this data, these correspond to either very small or very large weight.

Figure 2(b) shows the accuracy under uniform weight queries, where each query contains 10 ranges of approximately equal



(a) Construction throughput: network data (b) Construction throughput: tech ticket data (c) Query time vs summary size

Figure 3: Time costs as summary size varies.

weight. The graph shows the results for a fixed summary size of 2700 keys corresponding to a total summary size of about 32KB. Here we see a clear benefit of sampling methods compared to wavelets and q-digest: note that the error of q-digest is close to the total weight of the query. For those "light" queries which comprise a small fraction of the data, there is little to choose between the two sampling methods. But for queries which touch more of the data, structure-awareness is able to obtain half the error of its oblivious cousin.

The general trend is for the absolute error to increase with the fraction of data included in the sample. However, note that the gradient of these lines is shallow, meaning that the *relative* error is actually improving, as our analysis indicates: structure aware sampling obtains 0.001 error on a query that covers more than 0.1 of the whole data weight, i.e., the observed relative error is less than a 1% fraction.

Figure 2(c) shows the case when we hold the total weight of the query steady (in this case, at approximately 0.12 of the total data weight), and vary the number of ranges in each query. We see that the structure oblivious accuracy does not vary much: as far as the sample is concerned, all the queries are essentially subset queries with similar weight. However, when the query has fewer ranges, the structure aware sampling can be several times better. As the number of ranges per query increase, each range becomes correspondingly smaller, and so for queries with 40 or more ranges there is minimal difference between the structure aware and structure oblivious. On this query set, wavelets are an order of magnitude less accurate, given the same summary size.

6.3 Scalability

To study scalability of the different methods, we measured the time to build the summaries, and the time to answer 2500 range queries. Figure 3 shows these results as we vary the size of the summary. Although our implementations are not optimized for performance, we believe that these results show the general trend of the costs. Structure-oblivious is the fastest, whose cost is essentially bounded by the time to take a pass through the data (Figures 3(a) and 3(b)). Structure-aware sampling requires a second pass through the data, and for large summaries the extra time to locate nodes in the kd-tree reduces the throughput. We expect that a more engineered implementation could reduce this building cost.

The q-digest and sketch summaries are both around 2 orders of magnitude slower to build the summary. These structures are quite fast in one-dimension, but have to do more work in higher dimensions. For example, the sketch needs to update a number of locations which grows with the product of the logarithm of the dimension sizes. On pairs of 32 bit addresses, this factor is proportional to 32^2=1024. The cost of building the 2D Haar wavelet summary is 4 orders of magnitude more than sampling. The reason is that each point in the data contributes to 1024 coefficients, leading to millions of values before thresholding.

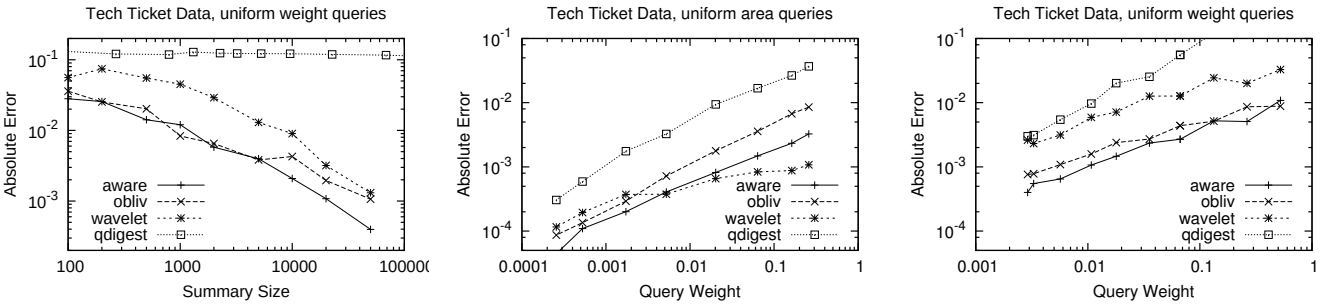
Since the samples, once built, have the same form, query answering takes the same time with both obliv and aware (Figure 3(c)): we just compute the intersection of the sample with each query rectangle. The cost grows with increasing sample size, as we are just scanning through the sample to find which points fall within each rectangle. Still, this naive approach to query answering can process thousands of query rectangles per second (recall, the y-axis is showing the time to complete all 2500 rectangle queries).

In comparison, asking this many queries over the full data takes 2 minutes. Again, we see the disadvantage of the wavelet approach: each rectangle query takes of the order of a second—in other words, it is about 1000 times slower than using a sample. The reason is that each rectangle is decomposed into dyadic rectangles. In the worst case, there are 1000 dyadic rectangles formed, and each dyadic rectangle requires the value of 1000 coefficients. The effect as we go to higher dimensions only gets worse, growing exponentially with the dimension. While there should be more efficient ways to use wavelets, the overall cost is offputtingly high.

6.4 Tech Ticket Data Accuracy

The plots in Figure 4 show accuracy experiments on the tech ticket data set. Figure 4(a) shows that structure-aware and structure-oblivious sampling behave similarly for smaller sample sizes: this is because this data set has many high weight keys which must be included in both samples. For large sample sizes, the methods diverge, and the benefit of structure awareness is seen: the error is less than half that for the same sized obliv summary for samples that are between 1% and 10% of the data size.

The next two plots compare the case for uniform area queries (over 25 ranges, Figure 4(b)) and uniform weight queries (10 ranges, Figure 4(c)). We see that on uniform area queries, wavelets can become competitive for higher weights, but this is not seen when the weight of each range is controlled. In either case, for these queries of several ranges, structure-aware sampling seems to give the best results overall . Certainly, wavelets do not seem to scale with this type of data: tens to hundreds of millions of coefficients are generated before thresholding, leading to a high time and space cost. Figure 3(b) emphasises the impracticality of wavelets on this data: generating and using samples takes seconds, while using wavelets takes (literally) hours.



(a) Accuracy vs space on Tech Ticket (b) Accuracy vs query weight on Tech Ticket (c) Accuracy vs query weight on Tech Ticket

Figure 4: Accuracy on the Tech Ticket Data.

7. CONCLUDING REMARKS

We introduce structure-aware sampling as an alternative to structure-oblivious sampling and tailored deterministic summaries. Our structure-aware samples are VAROPT—they retain the full benefits of state-of-the-art sampling over deterministic summaries: flexibility and support for arbitrary subset queries, accuracy on these queries, unbiased estimation, and exponential tail bounds on the error. By optimizing the sample distribution for range queries, we obtain superior accuracy with respect to structure-oblivious samples and match or surpass the accuracy of tailored deterministic summaries. Our proposed algorithms are simple to implement and are I/O efficient, requiring only two sequential read passes over the data and memory which is independent of the input size. Going to only a single (streaming) pass requires quite different ideas, since in this case the VAROPT sample is unique, and hence structure oblivious. Instead, it is necessary to relax the VAROPT requirement to allow the freedom to exploit structure; our initial results in this direction are presented in [6].

8. REFERENCES

[1] A. Bagchi, A. Chaudhary, D. Eppstein, and M. T. Goodrich. Deterministic sampling and range counting in geometric data streams. *ACM Trans. Algorithms*, 3(2):16, 2007.

[2] H. Brönnimann, B. Chen, M. Dash, P. Haas, and P. Scheuermann. Efficient data reduction with EASE. In *KDD*, 59–68, 2003.

[3] M. T. Chao. A general purpose unequal probability sampling plan. *Biometrika*, 69(3):653–656, 1982.

[4] M. Charikar, K. Chen, and M. Farach-Colton. Finding frequent items in data streams. In *ICALP*, 693–703, 2002.

[5] H. Chernoff. A measure of the asymptotic efficiency for test of a hypothesis based on the sum of observations. *Annals of Math. Statistics*, 23:493–509, 1952.

[6] E. Cohen, G. Cormode, N. Duffield. Structure-Aware Sampling on Data Streams In *SIGMETRICS*, 197–208, 2011

[7] E. Cohen, N. Duffield, H. Kaplan, C. Lund, and M. Thorup. Stream sampling for variance-optimal estimation of subset sums. In *SODA*, 1255–1264, 2009.

[8] E. Cohen, N. Duffield, C. Lund, M. Thorup, and H. Kaplan. Variance optimal sampling based estimation of subset sums. arXiv:0803.0473v2, 2010.

[9] G. Cormode, F. Korn, S. Muthukrishnan, and D. Srivastava. Finding hierarchical heavy hitters in streaming data. *ACM Trans. Knowl. Discov. Data*, 1(4):1–48, 2008.

[10] R. Gandhi, S. Khuller, S. Parthasarathy, and A. Srinivasan. Dependent rounding and its applications to approximation algorithms. *J. Assoc. Comput. Mach.*, 53(3):324–360, 2006.

[11] D. Gunopulos, G. Kollios, V. J. Tsotras, and C. Domeniconi. Approximating multi-dimensional aggregate range queries over real attributes. In *SIGMOD*, 463–474, 2000.

[12] J. Hájek. *Sampling from a finite population*. Marcel Dekker, New York, 1981.

[13] D. Haussler and E. Welzl. Epsilon nets and simplex range queries. *Discrete Comput. Geom.*, 2, 1987.

[14] J. Hershberger, N. Shrivastava, S. Suri, and C. D. Tóth. Adaptive spatial partitioning for multidimensional data streams. In *ISAAC*, 522–533, 2004.

[15] D. G. Horvitz and D. J. Thompson. A generalization of sampling without replacement from a finite universe. *J. Amer. Stat. Assoc.*, 47(260):663–685, 1952.

[16] J.-H. Lee, D.-H. Kim, and C.-W. Chung. Multi-dimensional selectivity estimation using compressed histogram information. *SIGMOD Rec.*, 28(2):205–214, 1999.

[17] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. In *SIGMOD*, 448–459, 1998.

[18] A. Panconesi and A. Srinivasan. Randomized distributed edge coloring via an extension of the chernoff-hoeffding bounds. *SIAM J. Comput.*, 26(2):350–368, 1997.

[19] J. M. Phillips. Algorithms for epsilon-approximations of terrains. In *ICALP*, 447–458, 2008.

[20] V. Poosala and Y. Ioannidis. Selectivity estimation without the attribute value independence assumption. In *VLDB*, 486–495, 1997.

[21] C.-E. Särndal, B. Swensson, and J. Wretman. *Model Assisted Survey Sampling*. Springer, 1992.

[22] N. Shrivastava, C. Buragohain, D. Agrawal, and S. Suri. Medians and beyond: new aggregation techniques for sensor networks. In *SenSys*, 239–249, 2004.

[23] A. Srinivasan. Distributions on level-sets with applications to approximation algorithms. In *FOCS*, 588-597, 2001.

[24] S. Suri, C. D. Tóth, and Y. Zhou. Range counting over multidimensional data streams. In *SCG*, 160–169, 2004.

[25] M. Szegedy and M. Thorup. On the variance of subset sum estimation. In *ESA*, 75–86, 2007.

[26] Y. Tillé. *Sampling Algorithms*. Springer, 2006.

[27] V. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilites. *Theory of Probability and its applications*, 16:264–280, 1971.

[28] J. S. Vitter, M. Wang, and B. Iyer. Data cube approximation and histograms via wavelets. In *CIKM*, 96–104, 1998.

[29] Y. Zhang, S. Singh, S. Sen, N. Duffield, and C. Lund. Online identification of hierarchical heavy hitters: algorithms, evaluation, and applications. In *IMC*, 101–114, 2004.

APPENDIX

A. BACKGROUND ON SUMMARIZATION

Sampling Techniques

A sample S is a random subset drawn from the space of keys K. In our context, the keys K are members of a structured domain $\mathcal{K}$.

Inclusion Probability Proportional to Size (IPPS) [12]: Many sampling schemes set the inclusion probability of each key in S proportional to its weight, but truncated so as not to exceed 1. When defined with respect to a *threshold* parameter $\tau > 0$, the inclusion probability of i is $p_i = \min\{1, w_i/\tau\}$.

The expected size of the sample is just the sum of the p_i's, so we can achieve a sample of expected size s by choosing an appropriate threshold τ_s. This τ_s can be found by solving the equation:

$$\textstyle\sum_i \min\{1, w_i/\tau_s\} = s\ .$$

τ_s can be computed via a linear pass on the data, using a heap H of size at most s. Let L denote the total weight of keys processed that are not present in the heap H. Algorithm 4 presents the algorithm to update the value of τ for each new key. If the weight of the new key is below the current value of τ, then we add it to L, the sum of weights; else, we include the weight in the heap H. Then adjust the heap: if the heap has s items, or the smallest weight in the heap falls below the new value of τ (Line 3), we move the smallest weight from H to L. Finally, we compute the new τ value in Line 6. After processing all keys, we have found τ_s for this data.

The Horvitz Thompson (HT) estimator [15] allows us to accurately answer queries from a sample, by providing *adjusted weights* to use for each key in the sample. For a key i of weight w_i and inclusion probability p_i the adjusted weight is $a(i) = w_i/p_i$ if i is included in the sample and $a(i) \equiv 0$ otherwise. For all i, $a(i)$ is an optimal estimator of w_i in that it minimizes the variance $\mathsf{Var}[a(i)]$.

A summary that includes the keys in S and their HT adjusted weights can be used to estimate the weight $w(J)$ of any subset of keys $J \subset K$: $a(J) = \sum_{i\in J} a(i) = \sum_{i\in S\cap J} a(i)$. The HT estimates are clearly unbiased: for all i, $\mathsf{E}[a(i)] = w_i$ and from linearity of expectation, for all subsets J, $\mathsf{E}[a(J)] = w(J)$.

With IPPS, the HT adjusted weight of an included key is τ if $w_i \le \tau$ and w_i otherwise. Hence, for any subset J we have

$$a(J) = \textstyle\sum_{i\in J|w_i\ge\tau} w_i + |\{i \in J \cap S : w_i < \tau\}| \cdot \tau\ . \qquad (1)$$

We can store either the adjusted weights for each key, or the original weights (and compute the adjusted weights via τ on demand). The variance of adjusted weight a_i is $\mathsf{Var}[a_i] = w_i^2(1/p_i - 1) \equiv w_i(\tau - w_i)$ if $w_i \le \tau$ and 0 otherwise. Using IPPS inclusion probabilities with HT estimates minimizes the sum $\Sigma V[a] = \sum_i \mathsf{Var}[a(i)]$ of per-key variances over all inclusion probabilities and estimators with the same (expected) sample size $s = \sum_i p_i$.

Poisson sampling is where the choice of whether to sample any key is made independently of all other choices. A Poisson IPPS sample of a specified expected size s can be computed efficiently with a single pass over the data.

VarOpt sampling [3, 26, 7] improves over Poisson sampling by guaranteeing a fixed sample size (exactly s) and giving tighter estimates [7, 8, 3, 26, 10]: the variance of any subset-sum estimate is at most that obtained by a Poisson IPPS sample. VarOpt samples are optimal in that for *any* subset size, they minimize the average variance of the estimates [25, 7]. A VarOpt sample of size s, denoted VarOpt$_s$, can be computed with a single pass over the data [8], generalizing the notion of reservoir sampling.

A sample distribution over $[n]$ is VarOpt for a parameter τ if:

Algorithm 4 STREAM_$\tau(i)$: processing item i

1: **if** $w_i < \tau$ **then** $L \leftarrow L + w_i$
2: **else** Insert$((i, w_i), H)$;
3: **while** $(|H| = s)$ **or** $(\min(H) < \tau)$ **do**
4: $\quad a \leftarrow$ delete_min(H).
5: $\quad L \leftarrow L + w_a$
6: $\quad \tau \leftarrow \frac{L}{s-|H|}$

(i) Inclusion probability is IPPS, i.e., for key i, $p_i = \min\{1, w_i/\tau\}$.

(ii) The sample size is exactly $s = \sum_{i\in[n]} p_i$.

(iii) *High-order inclusion & exclusion probabilities are bounded by respective products of first-order probabilities*, so, for any $J \subseteq [n]$,

$$\text{(I):}\quad \mathsf{E}[\prod_{i\in J} X_i] \le \prod_{i\in J} p_i \qquad \text{(E):}\quad \mathsf{E}[\prod_{i\in J}(1 - X_i)] \le \prod_{i\in J}(1 - p_i)$$

where p_i is the probability that key i is included in the sample S and $\mathsf{E}[\prod_{i\in J} X_i]$ is the probability that all $i \in J$ are included in the sample. Symmetrically $\mathsf{E}[\prod_{i\in J}(1 - X_i)]$ is the probability that all $i \in J$ are excluded from the sample.

Tail bounds. For both Poisson [5] and VarOpt [18, 23, 10, 8] samples we have strong tail bounds on the number of samples $J \cap S$ from a subset J. We state the basic Chernoff bounds on this quantity (other more familiar bounds are derivable from them). Let X_i be an indicator variable for i being in the sample, so $X_i = 1$ if $i \in S$ and 0 otherwise. Then X_J, the number of samples intersecting J is $X_J = \sum_{i\in J} X_i$, with mean $\mu = \mathsf{E}[X_J] = \sum_{i\in J} p_i$.

If $\mu \le a \le s$, the probability of getting more than a samples out of s in the subset J is

$$\Pr[X_J \ge a] \le \left(\frac{s-\mu}{s-a}\right)^{m-a}\left(\frac{\mu}{a}\right)^{a} \quad \left[\le e^{a-\mu}\left(\frac{\mu}{a}\right)^{a}\right]\ . \quad (2)$$

For $0 \le a \le \mu$, the probability of fewer than a samples in J is

$$\Pr[X_J \le a] \le \left(\frac{s-\mu}{s-a}\right)^{m-a}\left(\frac{\mu}{a}\right)^{a} \quad \left[\le e^{a-\mu}\left(\frac{\mu}{a}\right)^{a}\right]\ . \quad (3)$$

When sampling with IPPS, these bounds on the number of samples also imply tight bounds on the estimated weight $a(J)$: It suffices to consider J such that $\forall i \in J, p_i < 1$ (as we have the exact weight of keys with $p_i = 1$). Then the HT estimate is $a(J) = \tau X_J = \tau|J \cap S|$ and thus the estimate is guaranteed to be accurate:

$$\Pr[a(J) \le h],\ \Pr[a(J) \ge h] \le e^{(h-w(J))/\tau}(w(J)/h)^{h/\tau}\ . \quad (4)$$

Range discrepancy. The *discrepancy* of a sample measures the difference between the number of keys sampled in a range to the number expected to be there. Formally, consider keys with attached IPPS inclusion probabilities p_i over a structure with a set of ranges $\mathcal{R}$. The discrepancy Δ of a set of keys S with respect to range R is

$$\Delta(S, R) = \big||S \cap R| - \textstyle\sum_{i\in R} p_i\big|.$$

The *maximum range discrepancy* of S is accordingly the maximum discrepancy over $R \in \mathcal{R}$. For a sample distribution Ω, we define the maximum range discrepancy as

$$\Delta = \max_{S\in\Omega}\max_{R\in\mathcal{R}}\big||S \cap R| - \textstyle\sum_{i\in R} p_i\big|$$

The value of the discrepancy has implications for the accuracy of query answering. The absolute error of the HT estimator (1) on R is the product of τ and the discrepancy: $\tau \cdot \Delta(S, R)$. We therefore seek sampling schemes which have a small discrepancy.

For a Poisson IPPS or VAROPT sample, the discrepancy on a range R is subject to tail bounds (Eqns. (2) and (3)) with $\mu = p(R)$ and has expectation $O(\sqrt{p(R)})$.

ϵ-approximations. Maximum range discrepancy is closely related to the concept of an *ϵ-approximation* [27]. A set of points A is an ϵ-approximation of the range space $(X, \mathcal{R})$, if for *every* range R,

$$\left| \frac{|A \cap R|}{|A|} - \frac{|X \cap R|}{|X|} \right| < \epsilon \, .$$

(This is stated for uniform weights but easily generalizes). An ϵ-approximation of size s has maximum range discrepancy $\Delta = \epsilon s$. A seminal result by Vapnik and Chervonenkis bounds the maximum estimation error of a random sample over ranges when the VC dimension is small:

THEOREM 2 (FROM [27]). *For any range space with VC dimension d, there is a constant c, so that a random sample of size*

$$s = c\epsilon^{-2}(d \log(d/\epsilon) + \log(1/\delta))$$

is an ϵ-approximation with probability $1 - \delta$.

The structures we consider have constant VC dimension, and the theorem can be proved from a direct application of Chernoff bounds. Because VAROPT samples satisfy Chernoff bounds, they also satisfy the theorem statement. By rearranging and substituting the bound on ϵ, we conclude that a Poisson IPPS or a VAROPT sample of size s has maximum range discrepancy $\Delta = O(\sqrt{s \log s})$. with probability $(1 - \text{poly}(1/s))$

Other Summarization Methods

In addition to sampling methods such as Poisson IPPS and VAROPT there have been many other approaches to summarizing data in range spaces. We provide a quick review of the rich literature on summarization methods specifically designed for range-sum queries. Some methods use random sampling in their construction, although the resulting summary is not itself a VAROPT sample.

ϵ-approximations. As noted above, ϵ-approximations accurately estimate the number of points falling in a range. For axis-parallel hyper-rectangles, Suri, Toth, and Zhou presented randomized constructions that with constant probability generates an ϵ-approximation of size $O(\epsilon^{\frac{-2d}{d+1}} \log^d(\epsilon^{\frac{2}{d+1}} n))$ and an alternative but computationally intensive construction with much better asymptotic dependence on ϵ: $O(\frac{1}{\epsilon} \log^{2d+1} \frac{1}{\epsilon})$ [24]. The best space upper bound we are aware of is $O(\frac{1}{\epsilon} \log^{2d}(\frac{1}{\epsilon}))$ [19].

A proposal in [2] is to construct an ϵ-approximation of a random sample of the dataset instead of the full dataset. This is somewhat related to our I/O efficient constructions that utilize an initial larger random sample. The differences are that we use the sample only as a guide—the final summary is not necessarily a subset of the initial sample—and that the result of our construction is a structure-aware VAROPT sample of the full data set.

Another construction [14] trades better dependence on ϵ with logarithmic dependence on domain size. The data structure is built deterministically by dividing on each dimension in turn, and retaining "heavy" ranges. This can be seen as a multi-dimensional variant of the related q-digest data structure [22].

Sketches and Projections. Random projections are a key tool in dimensionality reduction, which allows large data sets to be compactly summarized and queried. Sketches are particular kinds of random projections, which can be computed in small space [4]. By keeping multiple sketches of the data at multiple levels of granularity, we can provide ϵ-approximation-like bounds in space that depends linearly on ϵ^{-1} and logarithmically on the domain size.

Wavelet transforms and deterministic histograms. A natural approach to summarizing large data for range queries is to decompose the range space into "heavy" rectangles. The answer to any range query is the sum of weights of all rectangles fully contained in by the query, plus partial contributions from those partly overlapped by the query. The accuracy then depends on the number (and weight) of rectangles overlapped by the query. This intuition underlies various attempts based on building (multi-dimensional) histograms [11, 20, 16]. These tend to be somewhat heuristic in nature, and offer little by way of guaranteed accuracy.

More generally, we can represent the data in terms of objects other than rectangles: this yields transformations such as DCT, Fourier transforms and wavelet representations. Of these, wavelet representations are most popular for representing range data [17, 28]. Given data of domain size u, the transform generates u coefficients, which are then *thresholded*: we pick the s largest coefficients (after normalization) to represent the data. When the data is dense, we can compute the transform in time $O(u)$, but when the domain is large and the data sparse, it is more efficient to generate the transform of each key, in time proportional to the product of the logarithms of the size of each dimension per key.

B. SEQUENCES OF AGGREGATIONS

Our algorithms repeatedly apply probabilistic aggregation:

LEMMA 3. *Consider a sequence $p^{(0)}, p^{(1)}, p^{(2)}, p^{(3)}, \ldots$ where $p^{(h)}$ is a probabilistic aggregate of $p^{(h-1)}$. Probabilistic aggregation is* transitive*, that is, if $h_1 < h_2$ then $p^{(h_2)}$ is a probabilistic aggregate of $p^{(h_1)}$.*

PROOF. We show property (I) (see Section 2) holds under transitivity. The proof for (E) is similar, and the other properties are immediate. We show that if $p^{(h+2)}$ is an aggregate of $p^{(h+1)}$, and $p^{(h+1)}$ is an aggregate of $p^{(h)}$, then $p^{(h+2)}$ is an aggregate of $p^{(h)}$.

$$\begin{aligned} \mathsf{E}_{p^{(2)}|p^{(0)}}[\prod_{i \in J} p_i^{(2)}] &= \mathsf{E}_{p^{(1)}|p^{(0)}}\left[\mathsf{E}_{p^{(2)}|p^{(1)}}[\prod_{i \in J} p_i^{(2)}]\right] \\ &\leq \mathsf{E}_{p^{(1)}|p^{(0)}}[\prod_{i \in J} p_i^{(1)}] \leq \prod_{i \in J} p_i^{(0)} \end{aligned}$$

□

Note that $p_i^{(h)} \in \{0, 1\}$ implies $p_i^{(h+1)} \equiv p_i^{(h)}$. Thus in a sequence of aggregations, any entry that is set remains set, so the number of positive entries in the output is at most that in the input.

C. MULTIPLE RANGES IN A HIERARCHY

We show that the discrepancy of a query that spans multiple ranges in a hierarchy is bounded by the number of ranges.

LEMMA 4. *Let Q be a union of ℓ disjoint ranges $R_1, \ldots, R_\ell$ on a hierarchy structure. The discrepancy is at most ℓ and is distributed as the error of a* VAROPT *sample on a subset of size $\mu = \sum_{h=1}^{\ell} (p(R_h) - \lfloor p(R_h) \rfloor) \leq \ell$.*

PROOF. Consider a truncated hierarchy where the nodes $R_1, \ldots, R_\ell$ are leaves. Include other nodes to form a set L' of disjoint nodes which covers all original leaves. For each leaf node in the truncated hierarchy $R_h \in L'$ consider a corresponding "left-over" 0/1 random variable with probabilities $p(R_h) - \lfloor p(R_h) \rfloor$: its value is 1 if the range R_h has $\lceil R_h \rceil$ samples, and its value is 0 if there are $\lfloor R_h \rfloor$ samples. It follows directly from our construction that the sample with respect to these leftovers is a VAROPT sample, since the original summarization from L' up proceeds like a hierarchy summarization, treating the leftovers as leaves. □

Applying Chernoff bounds, we obtain that the discrepancy on Q is at most $\sqrt{\ell}$ with high probability.

D. ORDER STRUCTURES

Recall that order structures consist of all intervals of keys.

THEOREM 1 (RESTATED). *For the order structure (i) there always exists a* VAROPT *sample distribution with maximum range discrepancy $\Delta \leq 2$. (ii) For any fixed $\Delta < 2$, there exist inputs for which a* VAROPT *sample distribution with maximum range discrepancy $\leq \Delta$ does not exist.*

Order Structure Summarization. To gain intuition, first consider inputs where there is a partition $\mathcal{L}$ of the ordered keys into non-overlapping intervals such that for each interval $J \in \mathcal{L}$, $p(J) = \sum_{i \in J} p_i \equiv 1$, i.e. the initial probabilities sum to 1. In this case, there are VAROPT samples which pick one key from each interval $J \in \mathcal{L}$. Now observe that any query interval R covers some number of full unit intervals. The only discrepancy comes from the at most 2 end intervals, and so the maximum range discrepancy is bounded by the probability mass therein, $\Delta < 2$. Therefore, this sample has maximum range discrepancy $\Delta < 2$.

The general case is handled by OSSUMMARIZE$(p_1, \ldots, p_n)$. (Algorithm 5). Without loss of generality, key i is the ith key in the sorted order, and p_i is its inclusion probability. The algorithm processes the keys in sorted order, maintaining an active key a that is the only key that is not set from the prefix processed so far. At any step, if there is an active key a, the selected pair for the aggregation consists of a and the current key. Otherwise, the aggregation involves the current and the next key. The final sample S is the set of keys with $p_i = 1$. We now argue it has bounded discrepancy.

PROOF OF THEOREM 1 (i). The algorithm can be viewed as a special case of a hierarchy summarization algorithm where the ordered keys are arranged as a hierarchy which is a long path with a single leaf hanging out of each path node. The internal nodes in the hierarchy correspond to prefixes of the sorted order, and thus they are estimated optimally (the number of samples is the floor or ceiling of the expectation): For any i, the number of members of S amongst the first i keys is

$$|S \cap [1,i]| \in \{\lfloor \sum_{j \leq i} p_i \rfloor, \lceil \sum_{j \leq i} p_i \rceil\} .$$

For a key range $R = [i_1, i_2]$ that is not a prefix ($i_2 \geq i_1 > 1$), we can express it as the difference of two prefixes:

$$\begin{aligned}
|S \cap R| &= |S \cap [1,i_2]| - |S \cap [1, i_1 - 1]| \\
&\leq \lceil \sum_{j \leq i_2} p_i \rceil - \lfloor \sum_{j \leq i_1 - 1} p_i \rfloor \leq 1 + \sum_{j \leq i_2} p_i - (-1 + \sum_{j \leq i_1 - 1} p_i) \\
&= 2 + \sum_{i_1 \leq j \leq i_2} p_i \, . \\
&\geq \lfloor \sum_{j \leq i_2} p_i \rfloor - \lceil \sum_{j \leq i_1 - 1} p_i \rceil \geq -1 + \sum_{j \leq i_2} p_i - (1 + \sum_{j \leq i_1 - 1} p_i) \\
&= -2 + \sum_{i_1 \leq j \leq i_2} p_i \, .
\end{aligned}$$

Hence the maximum discrepancy is at most 2, as claimed. □

Summaries with smaller discrepancy. Requiring the summary to be VAROPT means that it may not be feasible to guarantee Δ *strictly* less than 2. We can, however, obtain a *deterministic* set with maximum range discrepancy $\Delta < 1$: Associate key i with the interval $H_i = (\sum_{j<i} p_j, \sum_{j \leq i} p_j]$ on the positive axis (with respect to the original vector of inclusion probabilities) and simply include in S all keys where the H_i interval contains an integer. In fact, we can obtain a sample which satisfies the VAROPT requirements (i) and (ii) but not (iii) with $\Delta < 1$: Uniformly pick $\alpha \in [0,1]$ and store all keys i so that $h + \alpha \in H_i$ for each integer h. This sampling method is known as *systematic sampling* [21]. Systematic samples, however, suffer from positive correlations which mean that estimates on some subsets have high variance (and Chernoff tail bounds do not apply).

Algorithm 5 OSSUMMARIZE$(p_1, \ldots, p_n)$

```
1:  a ← 1                                          ▷ leftover key
2:  i = 2                                          ▷ current key
3:  while i ≤ n do                                 ▷ left to right scan of keys
4:      while p_a = 1 and a < n do
5:          a++;
6:      i ← a + 1
7:      while p_i = 1 and i < n do
8:          i++;
9:      if p_a < 1 and p_i < 1 then
10:         PAIR-AGGREGATE(a, i)
11:         if p_a = 1 or p_a = 0 then             ▷ p_a is set
12:             a ← i                              ▷ i is the new leftover key
13:         i++
```

Lower bound on discrepancy. We show that there are ordered weighted sets for which we can not obtain a VAROPT summary with maximum range discrepancy $\Delta < 2$.

PROOF OF THEOREM 1 (ii). For any positive integer m, we show that for some sequence, there is no VAROPT sample with $\Delta \leq 2 - 1/m$.

We use a sequence where $p_i = \epsilon \ll 1/(4m)$ and $\sum_i p_i \geq 5m$. Let $i_1 < i_2 < i_3, \cdots$ be the included keys, sorted by order. With each key i_ℓ we associate the position $s(i_\ell) = \sum_{j \leq i_\ell} p_j$.

We give a proof by contradiction. Consider keys i_ℓ, i_j with $\ell < j$. If an interval contains at most h sampled keys, it must be of size at most $h + \Delta$. If an interval contains at least h sampled keys, it must be of size at least $h - \Delta$.

The interval $[s(i_\ell) - \epsilon, s(i_j)]$, which is of size $s(i_j) - s(i_\ell) + \epsilon$, contains $j - \ell + 1$ sampled keys.

We obtain that $s(i_j) - s(i_\ell) + \epsilon \geq j - \ell + 1 - \Delta$. Rearranging, $s(i_j) \geq s(i_\ell) - \epsilon - \Delta + 1 + j - \ell$.

The interval $(s(i_\ell), s(i_j) - \epsilon)$, which is of size $s(i_j) - s(i_\ell) - \epsilon$, contains $j - \ell - 1$ sampled keys. Hence, $s(i_j) - s(i_\ell) - \epsilon \leq j - \ell - 1 + \Delta$. Rearranging, $s(i_j) \leq s(i_\ell) + \epsilon + j - \ell + \Delta - 1$.

From the above, for $j > \ell$, we have

$$s(i_j) \in (s(i_\ell) + j - \ell - \Delta + 1 - \epsilon, s(i_\ell) + j - \ell + \Delta + 1 + \epsilon) \, . \quad (5)$$

For $j \geq 2$,

$$s(i_j) \in (s(i_1) - \epsilon + j - \Delta, s(i_1) + j - 2 + \Delta + \epsilon) \, . \quad (6)$$

Fixing the first j included keys, $i_1, \ldots, i_j$, the conditional probability on $i_{j+1} = h$ is at most p_h. We have $s(i_{j+1}) \in (s(i_j) + 2 - \Delta - \epsilon, s(i_j) + \Delta + \epsilon)$. Therefore, there must be a positive probability for the event

$$s(i_{j+1}) < s(i_j) + \Delta + \epsilon - (1 - \epsilon) = s(i_j) + 1 - 1/m + 2\epsilon$$

which is contained in the event $s(i_{j+1}) \leq s(i_j) + 1 - 1/(2m)$.

Iterating this argument, we obtain that for all $k > 1$, we must have positive probability for $s(i_k) < s(i_1) + (k-1)(1 - 1/(2m)) = s(i_1) + k - 1 - (k-1)/2m$. From (6) we have $s(i_k) \geq s(i_1) + k - 1 - (1 - 1/m) = i_1 + k - 2 + 1/m$. Taking $k = 4m$, we get a contradiction. □

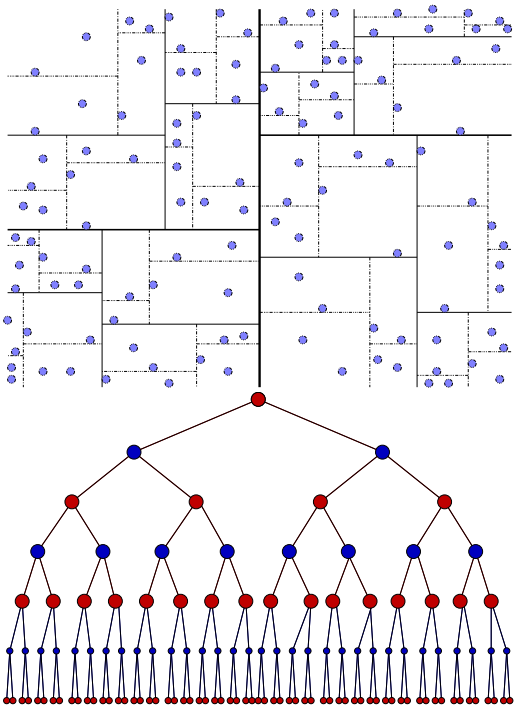


Figure 5: KD Hierarchical partition of two dimensional data.

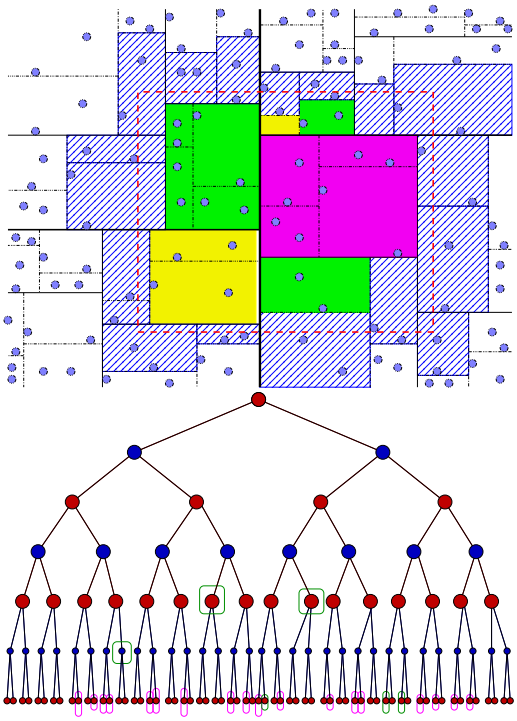


Figure 6: Query rectangle on a hierarchical partition.

E. ANALYSIS OF KD-HIERARCHY

The KD-HIERARCHY algorithm (Algorithm 2) aims to minimize the discrepancy within a product space. Figure 5 shows a two-dimensional set of 64 keys that are uniformly weighted, with sampling probabilities $1/2$, and the corresponding kd-tree: a balanced binary tree of depth 6. The cuts alternate vertically (red tree nodes) and horizontally (blue nodes). Right-hand children in tree correspond to right/upper parts of cuts and left-hand children to left/lower parts.

We now analyze the resulting summary, based on the properties of the space partitioning performed by the kd-tree. We use v to refer interchangeably to a node in the tree and the hyperrectangle induced by node v in the tree. A node v at depth d in the tree has probability mass $p(v) \leq s/2^d + 2$. We refer to the set of minimum depth nodes that satisfy $p(v) \leq 1$ as s-leaves (for *super leaves*) (v is an s-leaf iff $p(v) \leq 1$ and its immediate ancestor $a(v)$ has $p(a(v)) > 1$. The depth of an s-leaf (and of the hierarchy when truncated at s-leaves) is at most $D = 2 + \lceil \log_2 s \rceil$.

LEMMA 5. *Any axis-parallel hyperplane cuts $O(s^{\frac{d-1}{d}})$ s-leaves.*

PROOF. Consider the hierarchy level-by-level, top to bottom. Each level where the axis was not perpendicular to the hyperplane at most doubles the number of nodes that intersect the hyperplane. Levels where the partition axis is perpendicular to the hyperplane do not increase the number of intersecting nodes. Because axes were used in a round-robin fashion, the fraction of levels that can double the number of intersecting nodes is $\frac{d-1}{d}$. Hence, when we reach the s-leaf level, the number of intersecting nodes is at most $2^{\frac{d-1}{d}D} = O(s^{\frac{d-1}{d}})$. □

An immediate corollary is that the boundary of an axis-parallel box R may intersect at most $2ds^{\frac{d-1}{d}}$ s-leaves. We denote by $B(R)$ this set of boundary s-leaves. Let $U(R)$ be a minimum size collection of nodes in the hierarchy such that no internal node contains a leaf from $B(R)$. Informally, $U(R)$ consists of the (maximal) hyperrectangles which are fully contained in R or fully disjoint from R. Figure 6 illustrates a query rectangle R (dotted red line) over the data set of Figure 5. The maximal interior nodes contained in R ($v \in U(R) | v \subset R$) are marked in solid colors (and green circles in the tree layout) and the boundary s-leaves $B(R)$ in light stripes (magenta circles in the tree layouts). For example, the magenta rectangle corresponds to the R-L-L-R path.

LEMMA 6. *The size of $U(R)$ is at most $O(ds^{\frac{d-1}{d}} \log s)$.*

PROOF. Each node in U must have a sibling such that the sibling, or some of its descendants, are in $B(R)$. If this is not the case, then the two siblings can be replaced by their parent, decreasing the size of $U(R)$, which contradicts its minimality. We bound the size of $U(R)$ by bounding the number of potential siblings. The number of ancestors of each boundary leaf is at most the depth which is $\leq 2 + \lceil \log_2 s \rceil$. Thus, the number of potential siblings is at most the number of boundary leaves times the depth. By substituting a bound on $|B(R)|$, we obtain the stated upper bound. □

These lemmas allow us to bound the estimation error, by applying Lemma 4. That is, for each $v \in U(R)$ such that $v \subset R$ we have a 0/1 random variable that is 1 with probability $p_v - \lfloor p_v \rfloor$ and is 0 otherwise (The value is 0 if v includes $\lfloor p_v \rfloor$ samples and 1 otherwise). For each $v \in B(R)$, we have a random variable that is 1 with probability $p(v \cap R)$. This is the probability that S contains one key from $v \cap R$ (S can contain at most one key from each s-leaf). The sample is VAROPT over these random variables with

$$\mu = \sum_{v \in U(R) | v \subset R} (p(v) - \lfloor p(v) \rfloor) + \sum_{v \in B(R)} p(v \cap R) \leq |U(R)| + |B(R)| \,.$$

Substituting our bounds on $|U(R)|$ and $|B(R)|$ from Lemmas 5 and 6 gives accuracy bound claimed at the start of the section.