A Study on Workload-aware Wavelet Synopses for Point and Range-Sum Queries *

Michalis Mathioudakis
National Technical University of Athens
mathiou@dblab.ntua.gr

Dimitris Sacharidis
National Technical University of Athens
dsachar@dblab.ntua.gr

Timos Sellis
National Technical University of Athens
timos@dblab.ntua.gr

ABSTRACT

In this paper, we perform an extensive theoretical and experimental study on common synopsis construction algorithms, with emphasis on wavelet based techniques, that take under consideration query workload statistics. Our goal is to compare, "expensive" quadratic time algorithms with "cheap" near-linear time algorithms, particularly when the latter are not optimal and/or not workload-aware for the problem at hand. Further, we present the first known algorithm for constructing wavelet synopses for a special class of range-sum query workloads. Our experimental results, clearly justify the necessity for designing workload-aware algorithms, especially in the case of range-sum queries.

1. INTRODUCTION

Compact synopses have proven very popular, recently, as an effective means of dealing with massive multi-dimensional data-sets, such as those typically encountered in modern data intensive scenarios, e.g. OLAP queries. Such synopses serve two goals. First, synopses can assist in the query optimization process by providing highly accurate selectivity estimates, as they efficiently summarize the joint data distribution. Second, synopses can be used instead of the actual data providing with approximate answers to large costly range aggregate queries. Indeed, it can be argued that users, such as data analysts, exhibit exploratory behavior: they pose complex queries over large parts of the data, which would require considerable resources to process, and yet, they can tolerate some imprecision in the answers as they try to identify/extract interesting patterns, as long as these approximate answers are fast and are accompanied with some error guarantees.

In particular, summarization techniques that take user behavior under consideration are very common. User behavior is collected in terms of workload statistics by logging the queries posed to the system. The argument here is that these statistics accurately predict future behavior and thus, can be used to identify "hot" areas of the data. The synopsis construction algorithm is then designed so as to pay closer attention to these hot areas by providing increased accuracy, at the expense of less interesting areas. We refer to such algorithms as being *workload-aware*.

Among the most popular data summarization techniques proposed in literature are space partitioning approaches, such as histograms and wavelet synopses. Briefly, for a domain of size N, a histogram construction algorithm searches for a partition of the domain space in B arbitrarily sized buckets and approximates all data inside a bucket by a single value; the optimal partition is chosen so as to minimize some error metric, while not exceeding the available budget B. On the other hand, the wavelet decomposition employs a fixed, data-independent hierarchical partitioning of the domain space (better known as a linear transformation) into N intervals, each having a single value, a *coefficient*, associated with; the wavelet synopsis algorithm optimizes for some error metric by choosing to maintain only B of the resulting coefficients, in accordance to a space budget. The emphasis of this study is on space partitioning approaches and, in particular, on workload-aware wavelet synopsis construction algorithms.

Various algorithms for constructing wavelet synopses can be found in the database research literature; some of them can also be tuned to become workload-aware. These algorithms try to minimize the aggregate error incurred when all possible point, or equality, queries are considered, given a space budget. Interestingly enough, their algorithmic requirements range from (pseudo-)linear in domain size[1] and budget up to quadratic in domain size. A natural question that arises is how "cheap", linear algorithms fare against "expensive", quadratic algorithms in terms of the accuracy achieved by the resulting synopsis.

Surprisingly, the problem of minimizing the aggregate error when range queries are involved has attracted little attention. Clearly, workload-aware algorithms should take into consideration such queries, in order to be more applicable to real-life scenarios. In this work, we propose a novel workload-aware algorithm, RangeWave, that optimizes for a

*This work has been funded by the project PENED 2003. The project is cofinanced 75% of public expenditure through EC - European Social Fund, 25% of public expenditure through Ministry of Development - General Secretariat of Research and Technology and through private sector, under measure 8.3 of OPERATIONAL PROGRAMME "COMPETITIVENESS" in the 3rd Community Support Programme.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
DOLAP'06, November 10, 2006, Arlington, Virginia, USA.
Copyright 2006 ACM 1-59593-530-4/06/0011 ...$5.00.

[1] or tuple count, in the case of sparse data-sets

special class of hierarchical range queries. To the best of our knowledge, this is the first treatment for range queries given workload statistics.

Our Contribution Our study is focused primarily on wavelet synopsis construction algorithms and presents the following contributions:

1. We provide a detailed theoretical presentation of the time and space complexities associated with the most common algorithms found in literature, with particular interest in the algorithms that can become workload aware. In the extensive experimental study, our goal is to seek for the algorithm that offers the optimal compromise in terms of achieved accuracy compared to its running time overhead. For the sake of comparison, we also include in our study the optimal histogram construction algorithm.

2. We present a novel workload-aware synopsis construction algorithm, RangeWave, for a special class of range-sum queries. We experimentally validate its increased accuracy and further investigate the performance overhead it introduces, against previous non workload-aware wavelet techniques.

1.1 Outline

The remainder of this paper is organized as follows. Next, Section 2 discusses related work. Section 3 provides the necessary background for the wavelet decomposition and introduces weighted $\mathcal{L}_p^w$ norms for measuring synopsis accuracy for workloads of both point and range-sum queries. A detailed presentation and categorization of existing algorithms is presented in Section 4. Our proposed algorithm, RangeWave, is formally introduced in Section 5. Finally, we present an experimental evaluation in Section 6 and conclude in Section 7.

2. RELATED WORK

Histograms have a long tradition as an effective synopsis technique and are, thus, used widely in commercial systems. The bulk of early research focused on heuristically minimizing approximation errors in some estimation problem with the validity of the heuristics shown only experimentally. The interested reader can refer to the work of Ioannidis [9] for detailed discussion. A special class of heuristic algorithms is those where the histogram is constructed solely by query answer feedback; these include self-tuning, or workload-aware histograms [1, 2].

The first work to tackle the problem of constructing an optimal histogram from an algorithmic standpoint is [10]. The authors make the observation that once the optimal bucket boundaries have been found the value to assign to each bucket can be independently derived. Their optimization is for the sum-squared-error of point queries, yet extensions to other metrics such as weighted variations and maximum absolute/relative error are possible, albeit only for point queries. In the case of aggregating range-sum errors, the pioneering work of [12] dealt with workloads of hierarchical range queries, minimizing the (weighted) sum-squared-error. For arbitrary range queries approximation algorithms were provided in [16], assuming, however, uniform workload distribution.

The wavelet decomposition has been applied successfully as a data reduction mechanism in a wide variety of applications. To name a few, wavelets have been used in answering range-sum aggregate queries over data cubes [19, 18] and in selectivity estimation [14]. The effectiveness of Haar wavelets as a general-purpose approximate query processing tool was demonstrated in [3]. For the case of data sets with multiple measures the authors in [4] introduce the notion of extended wavelets, while I/O efficient algorithms for common maintenance tasks were presented in [11].

A lot of recent work focuses on developing optimal algorithms for constructing wavelet synopses that minimize more meaningful than the sum-squared-error metrics, taking into consideration workload statistics. The work in [5] constructs wavelet synopses that probabilistically minimize the maximum relative or absolute error incurred for reconstructing any data value. The work in [6] showed that this is also possible for deterministically constructed synopses and provided a novel dynamic programming recurrence, applicable [7] to incorporating workload statistics. Similar ideas were employed in [15] to construct optimal synopses in near-quadratic time for a particular class of workload statistics. Further, the work in [8] provides with improvements to the space requirements of the aforementioned dynamic programming algorithms. For the same problem, that of constructing an optimal workload-aware wavelet synopses, the work in [17] constructs a novel workload-dependant Haar-like basis by changing the decomposition process so that the conventional greedy thresholding technique applies. Assuming a uniform workload, the authors in [13] proved that the heuristics employed in [14] were in fact optimal — however, their approach requires the construction of the prefix-sum array/cube.

Let us note that various approximation schemes exist for fast construction of histograms and wavelet synopses, better suited for low-resource settings such as data stream environments. Such schemes, however, are beyond the scope of this study where we primarily focus on *optimal* workload-aware wavelet synopses.

3. PRELIMINARIES

Wavelets are a useful mathematical tool for hierarchically decomposing discrete functions (vectors) in ways that are both efficient and theoretically sound. Broadly speaking, the wavelet decomposition of a vector consists of a coarse overall approximation along with detail coefficients that influence the vector at various scales.

Haar decomposition. Let us describe the Haar transformation, the simplest type of wavelet decompositions. Let A be a data vector of size N; $N = 2^j$ for some $j \in \mathbb{N}$. We create a new vector $S_1[0 \dots (N/2-1)]$ of size $N/2$ by averaging the values together pairwise to get a new lower-resolution representation of the data. Similarly, we create a vector of differences, or details, $D_1[0 \dots (N/2-1)]$, consisting of the differences of the (second) of the averaged values from the computed pairwise average. Notice that there is no loss of information so far, since the value of every element of the original data vector can be reconstructed from the elements of the new vectors, S_1 and D_1. For example $A[0] = S[0] + D[0] = (A[0] + A[1])/2 + (A[0] - A[1])/2$, $A[1] = S[0] - D[0] = (A[0] + A[1])/2 - (A[0] - A[1])/2$, etc. The process we described so far comprises the first

level of the hierarchical decomposition and is recursively repeated for the vector of averages, until we produce vectors of size 1. The total number of details we compute is $1+2+\ldots+N/2 = N-1$. In the end, we construct a vector C of size N, consisting of the average of all the original values and the $N-1$ details. C is called the non-normalized Haar transformation of A. Intuitively, coefficients that are computed by averaging or differencing larger areas of the data vector are more important than those involving smaller areas. In order to get the normalized Haar transformation of A we multiply each coefficient computed during the k-th step of the decomposition by $\sqrt{N/2^{logN-k}}$. Furthermore, the Haar wavelet decomposition can also be extended to *multi-dimensional* data arrays through natural generalizations of the one-dimensional decomposition process described above.

Figure 1(a) illustrates a simple numeric example. Suppose we have the data vector $a = [2, 2, 0, 2, 3, 5, 4, 4]$ of size $N = 8$. The non-normalized tranformation of a is the vector $w_a = [11/4, -5/4, 1/2, 0, 0, -1, -1, 0]$ of size $N = 8$. It consists of the average value of a and the $N-1$ detail coefficients we computed during the decomposition. In our example, the average value of a is $11/4$, the single level 0 detail is $-5/4$, level 1 details are $1/2$ and 0 and level 2 details are $0, -1, -1$ and 0.

The error tree, seen on Figure 1(b), provides a good means of visualizing the hierarchical nature of Haar transformation. The root of the tree, c_0, is the average of all data values. Its inner nodes correspond to the detail coefficients and its leaves correspond to the original data values. Notice that a leaf-value can be reconstructed from the values of $\log N + 1$ inner nodes, one per level of the tree, lying on the path from the root to the leaf. For example, $a[5] = c_0 - c_1 + c_3 - c_6 \Leftrightarrow 5 = \frac{11}{4} - \left(-\frac{5}{4}\right) + 0 - (-1)$. A coefficient contributes positively to the leaves on its left and negatively to the leaves on its right.

Wavelet Synopses. A B-term wavelet synopsis is constructed by choosing a subset $\Lambda \subset C$ of wavelet coefficients; typically $B = |\Lambda| \ll N$. Implicitly, the remaining $N - B$ coefficients are considered zero and, thus, not stored.

A wavelet synopsis is a lossy form of compression; that is, by inverting the decomposition process we obtain only an approximation of the original data incurring, thus, an approximation error. The overall approximation error is calculated by aggregating individual errors, which are of two kinds: (i) point errors; and (ii) range-sum errors.

A point error is the error incurred in answering a point, or equality, query using the synopsis: i.e., what is the value of the i-th element. Let A represent the original data of length N and $\hat{A}$ the data reconstructed by the synopsis; clearly, there are N point errors: $E^p[i] = \left|A[i] - \hat{A}[i]\right|$, for $0 \le i < N$.

On the other hand, a range-sum error is incurred when answering a range-sum query: i.e., what is the sum of values in a contiguous range of the data. Assuming, as before, a vector A of length N and its approximation $\hat{A}$, there are $N(N+1)/2$ distinct range-sum queries and respective errors: $E^{rs}[l:r] = \left|\sum_{i=l}^{r}(A[i] - \hat{A}[i])\right|$, for $0 \le l \le r < N$. For ease of presentation and without loss of generality, we assume that the range-sum errors are sorted in a vector $E^{rs}[i]$, for $0 \le i < N(N+1)/2$.

A special class of range-sum queries is that which involves dyadic ranges. Given a vector A of length N, a power of 2, the dyadic ranges of A are the ranges $[k2^j, (k+1)2^j - 1]$, for $0 \le j < \log N$ and $0 \le k < N/2^j$. There are $2N-1$ dyadic ranges, conceptually arranged in a full binary tree, and therefore as many dyadic range-sum queries and errors possible: $E^{dr}[j,k] = \left|\sum_{i=k2^j}^{(k+1)2^j-1}(A[i] - \hat{A}[i])\right|$.

In order to measure the total approximation error we need to aggregate the individual point or range-sum errors by devising an appropriate metric. In this study we adopt the usage of weighted $\mathcal{L}_p^w$ norms, as they can incorporate all error metrics that appear in literature.

Formally, an $\mathcal{L}_p^w$ is a measure of the magnitude of a vector. Assuming an input data vector A and a weight vector w, both of length N, $\mathcal{L}_p^w$ is defined as:

$$\mathcal{L}_p^w(A) \hat{=} \left(\sum_{i=0}^{N-1} w[i](A[i])^p\right)^{1/p}, \text{ for } 0 < p < \infty.$$

Additionally, for $p = \infty$ we can define $\mathcal{L}_\infty^w$ to be the maximum value of $w[]A[]$: $\mathcal{L}_\infty^w = \max_i w[i]A[i]$. Note that, for the purpose of optimizing for a minimum (or a maximum) value, one can ignore the $1/p$ exponent, a convention which we follow for the remainder of this paper.

Weighted norms can be applied to both point and range-sum error vectors. The most widely used case is that of the $\mathcal{L}_2$ norm of the point error vector; this gives the ubiquitous sum-squared-error (SSE) metric. For the case of constructing an optimal wavelet synopsis in terms of minimizing the SSE it can be shown that one only needs to keep the largest in absolute normalized value coefficients[2].

$\mathcal{L}_p^w$ norms are general enough to include relative error metrics, such as the maximum relative point error (with sanity bound s) $\max_i \frac{E^p[i]}{\max\{A[i],s\}}$. One simply needs to incorporate the denominator in the weight: $w[i] = 1/\max\{A[i], s\}$.

Workload-aware algorithms. The focus on this study will be on algorithms that minimize the *expected sum-squared-error* when (point or range-sum) queries are drawn from the query workload. Assuming a query q_i appears in the collected workload statistics with probability p_i and setting $w[i] = p_i$, it is straightforward to see that the $\mathcal{L}_2^w$ norm equals the expected value of the sum-squared-error [17]. Note, that relative errors are also possible, e.g., by setting $w[i] = p_i/\max\{A[i], s\}$.

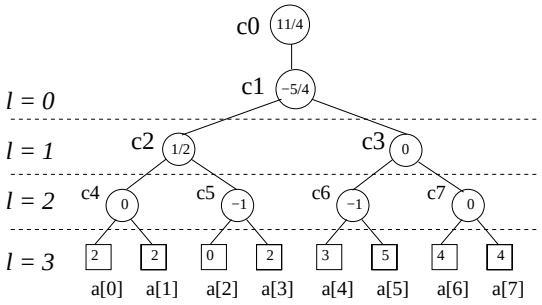
For the remainder of this paper, *workload-aware* synopsis construction algorithms are considered those that can optimize for the $\mathcal{L}_2^w$ norm.

4. SYNOPSIS CONSTRUCTION ALGORITHMS

In this section we present algorithms found in the literature for constructing wavelet synopses, as well as briefly discuss some optimal histogram algorithms. Our aim is to establish their space and time requirements and examine whether they can become workload-aware. An experimental study of these algorithms can be found in Section 6. We begin our discussion with algorithms designed for point queries in Section 4.1 and continue with those designed for the more general category of range-sum queries in Section 4.2.

[2]The wavelet decomposition is an orthonormal transformation and by the Parseval theorem the $\mathcal{L}_2$ norm is preserved.

Resolution	Averages	Detail Coefficients
3	[2, 2, 0, 2, 3, 5, 4, 4]	−−−−−
2	[2, 1, 4, 4]	[0, −1, −1, 0]
1	[3/2, 4]	[1/2, 0]
0	[11/4]	[−5/4]



(a) **(b)**

Figure 1: Example wavelet decomposition and error tree for vector a.

4.1 Point Query Workloads

The Classic Algorithm. For the case of constructing a B-term wavelet synopsis minimizing the sum-squared-error (SSE, or unweighted $\mathcal{L}_2$ norm) the Classic algorithm selects the B largest in absolute normalized value coefficients, as discussed in Section 3. To this end it makes use of a heap to construct the synopsis in $O(N)$ space and $O(N + B \log N)$ time.

The GaKu Algorithm. The most influential work on wavelet synopses is that of Garofalakis and Kumar [6]. It has served as the basis for subsequent work [7, 15, 8], as well as our RangeWave algorithm introduced in Section 5. The authors in [6] proposed a dynamic programming algorithm optimizing for the maximum relative error ($\mathcal{L}_\infty^w$), which can be generalized to any distributive error metric [7], such as $\mathcal{L}_p^w$ norms of the point error vector.

In the following, we discuss the variation that minimizes the $\mathcal{L}_2^w$ norm of the point errors vector [7], which we denote as GaKu. The algorithm works on the Haar error tree recursively, starting from the root node c_0. Let $path(c_j)$ represent the set of all ancestors of c_j in the error tree, $coeff(c_j)$ denote the set of coefficients in the subtree rooted at c_j and $data(c_j)$ denote the data values (leaves) in the c_j subtree. Furthermore, let $sign(i, j)$ be 1 (-1) when the i-th data value lies in the first (second) half of $data(c_j)$.

The basic idea of GaKu is that for every node c_j with a fixed set S of its ancestors being considered part of the synopsis, the subproblem of minimizing the $\mathcal{L}_2^w$ error introduced in $data(c_j)$ can be found recursively by combining solutions for the subproblems at children nodes c_{2j}, c_{2j+1}. A three dimensional array M is required to store the solution of every dynamic programming subproblem. $M[j, b, S]$ denotes the minimum value of the $\mathcal{L}_2^w$ error in $data(c_j)$, assuming a space of b coefficients has been alloted to $coeff(c_j)$ and that a subset $S \in path(c_j)$ of coefficients is to be included in the synopsis. The base case of the recursion occurs for the leaf nodes of the error tree (i.e., the data values). In this case, $M[j, b, S]$, for $N \leq j < 2N$, is defined only for $b = 0$ and denotes the weighted square of the point error for $c_j = d_{j-N}$ when S is chosen for the synopsis.

$$M[j, 0, S] = w_i \cdot |d_{j-N} - \sum_{c_k \in S} sign(j - N, k) \cdot c_k|^2$$

For inner nodes, $M[j, b, S]$ is computed for two cases depending on whether coefficient c_j is kept or discarded, and for every case of allocating available synopsis space to its children.

$$M_{drop}[j, b, S] = \min_{0 \leq b' \leq b} (M[2j, b', S] + M[2j + 1, b - b', S])$$

$$M_{keep}[j, b, S] = \min_{0 \leq b' \leq b-1} (M[2j, b', S \cup \{c_j\}], M[2j+1, b-b'-1, S \cup \{c_j\}])$$

Then, the minimum of the two cases is chosen.

$$M[j, b, S] = \min \{M_{drop}[j, b, S], M_{keep}[j, b, S]\}$$

The minimum $\mathcal{L}_2^w$ error of the entire data set can be found in $M[root, B, \emptyset]$. The space and time complexities of GaKu are $O(N^2)$ and $O(N^2 \log B)$ respectively as shown in [8, 7].

The UrMa Algorithm. A fundamentally different approach is followed for the same $\mathcal{L}_2^w$ optimization problem by Matias and Urieli in [17]. Their technique, however, cannot be generalized to other $\mathcal{L}_p^w$ norms. The main idea is to employ a modified workload-dependant Haar decomposition, that embodies the weights directly into the decomposition process.

Instead of computing pairwise averages and differences, UrMa computes weighted averages and weighted differences over the data, and repeats this process over the weighted averages until the overall average is obtained. Let a_j and c_j denote the weighted averages and detail coefficients computed at the j-th node of the error tree. These are computed as:

$$a_j = \frac{y_j a_{2j} + x_j a_{2j+1}}{x_j + y_j} \quad \text{and} \quad c_j = \frac{a_{2j} - a_{2j+1}}{x_j + y_j},$$

where x_j, y_j are computed using the sum of weights l_k, r_k corresponding to the left and the right half of the interval under c_j in the error tree, respectively:

$$x_j = \sqrt{\frac{r_j}{l_j r_j + l_j^2}} \quad \text{and} \quad y_j = \sqrt{\frac{l_j}{l_j r_j + r_j^2}}$$

For the special case of the root of the error tree, which represents the total weighted average, we have:

$$x_0 = y_0 = \sqrt{\frac{1}{l_0 + r_0}}$$

The calculations for the sum of weights l_j, r_j can be done in parallel to the decomposition process in one pass using $O(N)$ space and time. The resulting transformation is shown to be orthonormal with respect to the $\mathcal{L}_2^w$ norm[17]. Therefore, picking the B largest in absolute value coefficients minimizes the $\mathcal{L}_2^w$ error and the Classic technique can be applied to construct the synopsis. Using a heap of size N, the

time and space complexity of UrMa is $O(N + B \log N)$ and $O(N)$, respectively.

The Vopt Algorithm. The work in [10] introduced an optimal histogram construction algorithm for minimizing the sum-squared-error. The authors decouple the problem of selecting bucket boundaries and assigning a single value per bucket by making the observation that once the optimal bucket boundaries have been found the value to assign to each bucket can be independently derived. In this study, we employ a modified workload-aware version of this algorithm, denoted as Vopt, that minimizes the $\mathcal{L}_2^w$ norm of the point error vector in $O(N^2B)$ time using $O(NB)$ space. This algorithm serves mainly as a comparison for the wavelet synopses techniques previously described.

4.2 Range-Sum Query Workloads

The MaUr Algorithm. The work in [14] proposed a greedy heuristic for the problem of minimizing the sum-squared-error over the range-sum error vector. Matias and Urieli in [13] proved the validity of that heuristic. They showed that the Haar decomposition deployed over the prefix sum array/cube of the data is in fact orthogonal with respect to the $\mathcal{L}_2$ norm over range-sum queries — this is not the case for the weighted $\mathcal{L}_2^w$ norm and, therefore, this algorithm cannot be made workload-aware. They also showed that the orthogonality does not hold in the case of computing the transformation over the unprocessed, raw data. The MaUr algorithm employs a heap similar to the Classic algorithm and, thus, needs $O(N + B \log N)$ time and $O(N)$ space.

The KMS Algorithm. The work in [12] proposes a dynamic programming workload-aware algorithm, denoted as KMS, for constructing a histogram that minimizes the $\mathcal{L}_2^w$ norm over a class of range-sum queries, the hierarchical range-sum queries. For any two such queries the ranges involved are either disjoint or contained in each other (i.e., their boundaries do not cross), conceptually constructing a tree hierarchy. The dynamic programming algortihm searches among the different ways the interval of a range-sum query can be partially overlapped from left and right by histogram buckets. There are $O(N^2)$ possible buckets that can overlap such an interval from the left and as many from the right; KMS needs to tabulate over all $O(N^4)$ possible choices. This results in impracticably large time and space complexities, $O(N^7B^2)$ and $O(N^5B)$ respectively.

5. RangeWave: AN OPTIMAL WORKLOAD-AWARE ALGORITHM FOR DYADIC RANGE-SUM QUERIES

In this section we present the first workload-aware algorithm for constructing wavelet synopses assuming workloads of range-sum queries, termed RangeWave. Our algorithm has the following characteristics:

1. It works entirely on the unprocessed/raw data, thus, making maintenance tasks simpler. Note that the only other known wavelet synopsis algorithm (MaUr [13]) requires the construction of the prefix sum array/cube.

2. It is workload-aware and optimizes for the class of dyadic range-sum query workloads, but can use any $\mathcal{L}_p^w$ norm to aggregate the dyadic range-sum errors — in comparison, MaUr is not workload-aware and only limited to the sum-squared-error metric.

3. It exhibits time and space requirements on par with the generalized GaKu algorithm [7], which targets only point query errors.

The RangeWave algorithm builds upon the basic dynamic programming recurrence of Garofalakis and Kumar and makes an important observation: since the dyadic range-sum queries can be organized in a dyadic hierarchy (binary tree) which coincides with the Haar error tree, to answer a dyadic range-sum query q_i that resides at node c_i of the error tree, one needs to examine only the ancestors of c_i, that is, only those coefficients in $path(c_i)$.

Briefly, RangeWave at any node c_i of the error tree given a space budget b and a subset S of coefficients in $path(c_i)$ must (i) calculate the error associated with dyadic range-sum q_i, (ii) distinguish two cases of keeping or not coefficient c_i, and (iii) decide the best allocation of remaining space ($b-1$ or b) to its children. This process proceeds recursively reaching the leaves/data values.

More formally, suppose that the wavelet coefficients and the weights are stored in arrays $C[0 \ldots N\text{–}1]$ and $W[1 \ldots 2N\text{–}1]$, respectively. As far as the weights array is concerned, $W[N \ldots 2N-1]$ correspond to weights for the point queries (dyadic ranges of length 1), while $W[1 \ldots N-1]$ corresponds to the weights of dyadic range-sum queries with length larger than 1. With every node c_i a two-dimensional dynamic programming array $E_i[b, S]$ is associated, which stores the optimal solution (minimum $\mathcal{L}_2^w$) when space of b coefficients is alloted to the subtree rooted at c_i and when a set S of coefficients in $path(c_i)$ are stored.

The base case of the recursion occurs for the leaf nodes of the error tree, when $N \leq i < 2N$. Then, the $E_i[0, \cdot]$ entries (defined only for zero space alloted to the leaves) store the weighted sum-squared error for reconstructing a data value (i.e., a point query):

$$E_i[0,S] = W[i]\left(\sum_{j\in path(i)} sign(i,j)\cdot C[j] - \sum_{j\in S} sign(i,j)\cdot C[j]\right)^2$$
$$= W[i]\left(\sum_{j\in path(i)\setminus S} sign(i,j)\cdot C[j]\right)^2$$

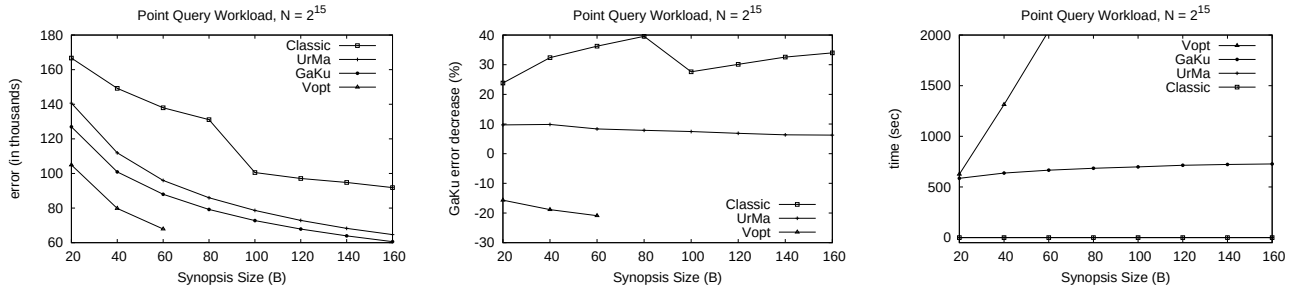
For any error tree node c_i other than the root, RangeWave chooses the minimum error among including or not the coefficient c_i:

$$E_i[b,S] = \min\{E_i^{keep}[b,S], E_i^{drop}[b,S]\} \text{ , where,}$$

$$E_i^{keep}[b,S] = W[i]\left(\sum_{j\in path(i)\setminus S} sign(i,j)\cdot C[j]\right)^2 + \min_{0\leq b'\leq b-1}\left\{E_{2i}[b', S\cup\{c_i\}] + E_{2i+1}[b-b'-1, S\cup\{c_i\}]\right\}$$

and

$$E_i^{drop}[b,S] = W[i]\left(\sum_{j\in path(i)\setminus S} sign(i,j)\cdot C[j]\right)^2 + \min_{0\leq b'\leq b}\left\{E_{2i}[b', S] + E_{2i+1}[b-b', S]\right\}.$$



(a) $\mathcal{L}_2^w$ Error (b) $\mathcal{L}_2^w$ Error Decrease for GaKu (c) Running Time

Figure 2: Accuracy and Running Time vs. Synopsis Size (B) for Point Query Workloads

At the root node c_0, there is no corresponding dyadic range-sum query and further there is a single child. Therefore, RangeWave only needs to examine whether the overall average c_0 should be kept in the synopsis:

$$\begin{aligned} E_0[B,\emptyset] &= \min\left\{E_0^{keep}[B,\emptyset], E_0^{drop}[B,\emptyset]\right\} \\ E_0^{keep}[B,\emptyset] &= \min_{0\le b'\le B-1} E_1[b',\{c_0\}] \\ E_0^{drop}[B,\emptyset] &= \min_{0\le b'\le B} E_1[b',\emptyset]. \end{aligned}$$

The mimimum $\mathcal{L}_2^w$ error is $E_0[B,\emptyset]$ and the optimal synopsis can be reconstructed using standard dynamic programming techniques (e.g., by storing the decisions made by the algorithm at each node).

Complexity Analysis. There are 2^l nodes at the l-th level of the error tree. Further, for each node c_i at the l-th level, the array $E_i[\cdot,\cdot]$ contains $\min\{B, 2^{\log N-l}-1\}\times 2^{l+1}$ entries. This means, that for nodes at levels larger than $\log N-\log B$ the array is of size $(2^{\log N-l}-1)\times 2^{l+1} = O(N)$. Summing for all nodes at level $l > \log N - \log B$ we get a space of

$$\sum_{l=\log N-\log B+1}^{\log N} 2^l \cdot N = O(N^2).$$

Similarly, for nodes at levels smaller or equal to $\log N-\log B$, the array size is $B\times 2^{l+1}$. Summing for all nodes at levels $l \le \log N - \log B$ we get a space of

$$\sum_{l=0}^{\log N-\log B} 2^l\cdot B\cdot 2^{l+1} = B\cdot \sum_{l=0}^{\log N-\log B} 2^{2l+1} = O\left(\frac{N^2}{B}\right).$$

Therefore, the space complexity of RangeWave is $O(N^2)$.

Each entry in the dynamic programming array requires examining all space allocation to the children nodes and thus $O(\min\{B, 2^{logN-l+1}-1\})$ time. Using similar arguments as before we obtain:

$$\begin{aligned} &\sum_{l=0}^{\log N-\log B} 2^l\cdot B^2\cdot 2^l + \sum_{l=\log N-\log B}^{\log N} 2^l\cdot\left(2^{\log N-l}\right)^2\cdot 2^l \\ &= O(N^2) + O(N^2\log B) = O(N^2\log B) \end{aligned}$$

Therefore, the time complexity of RangeWave is $O(N^2\log B)$.

6. EXPERIMENTAL STUDY

In this section we perform an extensive experimental evaluation on the algorithms discussed in Section 4, as well as our RangeWave algorithm. The comparison is made for each of the two families of algorithms: (i) assuming point query workloads, and (ii) assuming dyadic range-sum query workloads. Our objective is to evaluate the scalability and the obtained accuracy of the algorithms at hand. One question that usually arises in such settings and that we try to answer is whether one should choose expensive quadratic time algorithms in favor of faster near-linear time counterparts.

6.1 Testbed and Methodology

We have implemented the algorithms discussed in Sections 4 and 5 in C++ and the experiments reported here were performed on a 2GHz machine with 2GB of main memory. We used a Zipfian frequency generator to construct data sets of average skew ($z = 0.6$) by randomly permuting the generated frequencies. In the majority of experiments we used a dataset with $2^{15} = 32,768$ data values.

For constructing the workloads we used the Zipfian generator to assign weights to all possible queries. For the point query workload, denoted as W^{pq}, we chose relatively high skew ($z = 1.2$) and randomly permuted the frequencies. For the dyadic range-sum query workload, we used high skew ($z = 1.2$) to construct two workloads, one biased over larger ranges, W_b^{di}, and one with no particular bias W^{di} in which small and large ranges were equally likely to appear.

In our experimental study, we measure the $\mathcal{L}_2^w$ norm of the algorithms; to stress the accuracy of certain algorithms we explicitly state the error decrease they achieve. Finally, we measure the running time and show the scalability of all algorithms as N and B increase — note that we aborted the execution of algorithms in settings when they required more than an hour to finish.

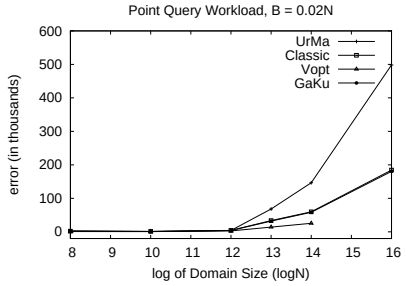
6.2 Point Query Workloads

We compare the two workload-aware wavelet algorithms GaKu and UrMa against Classic, which minimizes the sum-squared-error, and the workload-aware histogram construction algorithm Vopt. Recall that GaKu and Vopt are quadratic time algorithms compared to the near-linear time algorithms UrMa and Classic. In all experiments the workload used is W^{pq}.

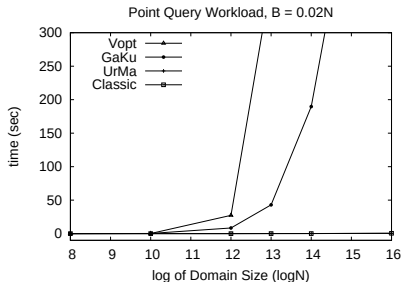
Figure 2 studies performance as the synopsis size increases. In particular, Figure 2(a) shows the $\mathcal{L}_2^w$ norm for all algorithms and Figure 2(b) emphasizes on the error decrease of GaKu compared to the other three algorithms. Note that the error in Figure 2(a) is measured in thousands with the exact magnitude having no particular meaning, as only rel-

ative numbers are interesting. The important thing to notice is that the histograms algorithm Vopt exhibits superior performance, around 10%-15% more accurate than GaKu, at a large cost, however; as Figure 2(c) shows, its running time becomes immediately prohibitive. Among the wavelet synopsis algorithms, GaKu is the winner in terms of accuracy, achieving around 10% and 30% more accurate synopses than UrMa and Classic, respectively, but at a cost due to its quadratic in N time requirement. Further, GaKu scales well as synopsis space increases due its logarithmic in B complexity. Overall, the UrMa algorithm seems the better choice as it constructs a relative accurate wavelet synopsis in really short time.

Figure 3 studies performance as the domain size increases from 2^8 up to 2^{16}, while the synopsis size was kept at 2% of the data size. As Figure 3(b) shows the time complexity of Vopt for $B = 0.02N$ essentially becomes cubic in N and, thus, we could not obtain measurements for 2^{16}. Also, the running time of GaKu increases quickly to impractical values. As far as accuracy is concerned, Figure 3(a) shows that among the pseudo-linear time UrMa scales badly in contrast to Classic; still, the quadratic time algorithms construct more accurate synopses.



(a) $\mathcal{L}_2^w$ Error



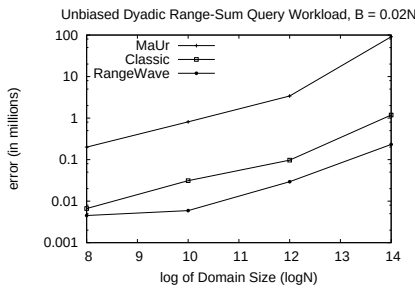
(b) Running Time

Figure 3: Accuracy and Running Time vs. Domain Size (N) for Point Query Workloads

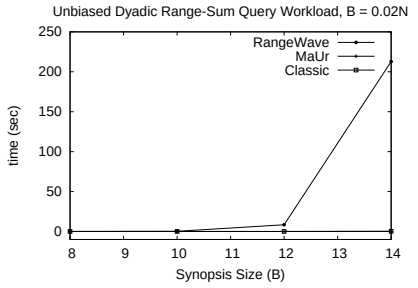
6.3 Dyadic Range-Sum Query Workloads

In this section we compare our RangeWave algorithm to the near-linear time, non workload-aware wavelet algorithms MaUr and Classic. Let us note that we do not include measurements of the workload-aware histogram construction algorithm KMS as it could only give results for a modest dataset of 128 values.

Unbiased Workload. We start our discussion with experiments on the unbiased workload W^{di}. Figure 4 studies



(a) $\mathcal{L}_2^w$ Error



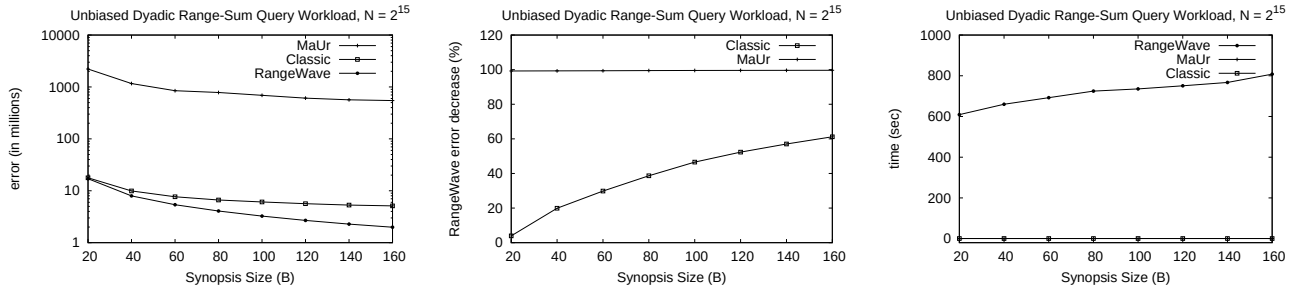
(b) Running Time

Figure 5: Accuracy and Running Time vs. Domain Size (N) for Unbiased Dyadic Range-Sum Query Workload

performance as the synopsis size increases. In particular, Figure 4(a) shows the $\mathcal{L}_2^w$ norm for all algorithms in logarithmic scale, Figure 4(b) emphasizes on the improvement of RangeWave against the other algorithms, and Figure 4(c) shows the running time. Notice, in Figure 4(b) that RangeWave offers an almost 100% improvement compared to MaUr as it produces orders of magnitude more accurate synopses, as seen in Figure 4(a). Further, RangeWave offers an improvement of up to 60% over Classic, as the former increases its accuracy at an larger rate as B increases. As expected, Figure 4(c) shows that RangeWave scales well as synopsis size increases, due to its logarithmic dependecy on B.

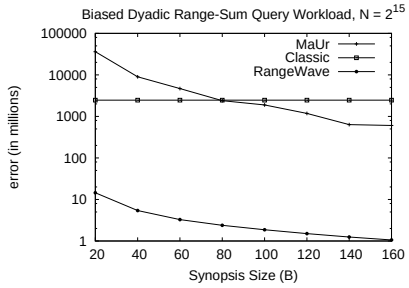
In Figure 5, we increase domain size from 2^8 up to 2^{14}, while the synopsis size was kept at 2% of the data size. Figure 5(a) shows that RangeWave is constantly significantly more accurate than Classic and orders of magnitude more accurate than MaUr. On the other hand, its quadratic time complexity is depicted in Figure 5(b).

Biased Workload. Here, we experiment with workload W_b^{di} which assigns more significance to larger range-sum queries. Figure 6 plots the accuracy achieved by all algorithms. It is important to notice that for such a biased workload Classic, in contrast to the unbiased workload case, performs poorly. Figure 6(a) shows that its accuracy does not increase with synopsis size. In general, both near-linear non workload-aware algorithms perform orders of magnitude worse than RangeWave; in fact, Figure 6(b) shows that RangeWave enjoys a performance gain of almost 100%. Also, observe that, as Figure 7 portrays, the accuracy of RangeWave is consistently orders of magnitude higher, as the domain size increases.

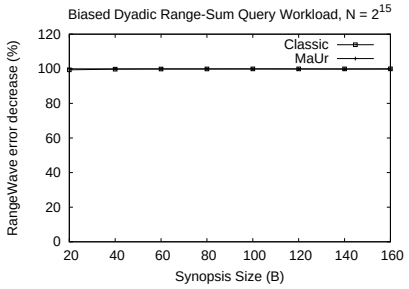


(a) $\mathcal{L}_2^w$ Error (b) $\mathcal{L}_2^w$ Error Decrease for RangeWave (c) Running Time

Figure 4: Accuracy and Running Time vs. Synopsis Size (B) for Unbiased Dyadic Range-Sum Query Workload



(a) L_2^w Error



(b) L_2^w Error Decrease for RangeWave

Figure 6: Accuracy vs. Synopsis Size (B) for Biased Dyadic Range-Sum Query Workload

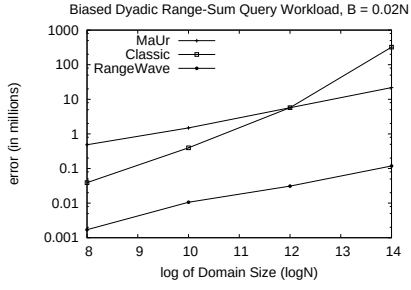


Figure 7: Accuracy vs. Domain Size (N) for Biased Dyadic Range-Sum Query Workload

7. CONCLUSIONS AND DISCUSSION

In this paper, we have presented an extensive theoretical and experimental study on wavelet synopsis algorithms that take under consideration either point or range-sum query workloads. In particular, for the latter case, we have devised the first known algorithm RangeWave for constructing optimal wavelet synopses, albeit restricted to a particular class of range-sum queries. Our experimental results for the case of point query workloads have shown that in practice, one gets what one pays for: quadratic time synopsis algorithms consistently outperform pseudo-linear counterparts, at a high price, though. As for range-sum query workloads, RangeWave is orders of magnitude more accurate, especially with skewed workloads, justifying the necessity for designing workload-aware algorithms for range-sum queries.

Our future plans, include the extension of RangeWave to more general range-sum workloads and the design of fast approximation schemes for the same problem, more appropriate to larger data-sets or to a streaming environment.

8. REFERENCES

[1] A. Aboulnaga and S. Chaudhuri. Self-tuning histograms: Building histograms without looking at data. In *SIGMOD Conference*, pages 181–192, 1999.

[2] N. Bruno, S. Chaudhuri, and L. Gravano. Stholes: A multidimensional workload-aware histogram. In *SIGMOD Conference*, pages 211–222, 2001.

[3] K. Chakrabarti, M. N. Garofalakis, R. Rastogi, and K. Shim. Approximate query processing using wavelets. In *Proceedings of 26th International Conference on Very Large Data Bases (VLDB)*, pages 111–122, 2000.

[4] A. Deligiannakis and N. Roussopoulos. Extended wavelets for multiple measures. In *Proceedings ACM SIGMOD International Conference on Management of Data*, pages 229–240, 2003.

[5] M. Garofalakis and P. B. Gibbons. Wavelet synopses with error guarantees. In *Proceedings ACM SIGMOD International Conference on Management of Data*, pages 476–487. ACM Press, 2002.

[6] M. Garofalakis and A. Kumar. Deterministic wavelet thresholding for maximum-error metrics. In *Proceedings ACM Principles of Database Systems (PODS)*, pages 166–176, 2004.

[7] M. Garofalakis and A. Kumar. Wavelet synopses for general error metrics. *ACM Transactions on Database Systems*, 30(4):888–928, 2005.

[8] S. Guha. Space efficiency in synopsis construction algorithms. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages

409–420. VLDB Endowment, 2005.

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

[10] H. V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. C. Sevcik, and T. Suel. Optimal histograms with quality guarantees. In *VLDB*, pages 275–286, 1998.

[11] M. Jahangiri, D. Sacharidis, and C. Shahabi. Shift-Split: I/O efficient maintenance of wavelet-transformed multidimensional data. In *Proceedings ACM SIG on Management of Data (SIGMOD)*, June, 2005.

[12] N. Koudas, S. Muthukrishnan, and D. Srivastava. Optimal histograms for hierarchical range queries. In *PODS*, pages 196–204, 2000.

[13] Y. Matias and D. Urieli. On the optimality of the greedy heuristic in wavelet synopses for range queries. Technical Report TR-TAU, 2005.

[14] Y. Matias, J. S. Vitter, and M. Wang. Wavelet-based histograms for selectivity estimation. In *Proceedings ACM SIGMOD International Conference on Management of Data*, pages 448–459, 1998.

[15] S. Muthukrishnan. Subquadratic algorithms for workload-aware haar wavelet synopses. In *FSTTCS*, 2005.

[16] S. Muthukrishnan and M. Strauss. Rangesum histograms. In *SODA*, pages 233–242, 2003.

[17] D. Urieli and Y. Matias. Optimal workload-based weighted wavelet synopses. In *Proceedings of International Conference on Database Theory (ICDT)*, 2005.

[18] J. S. Vitter and M. Wang. Approximate computation of multidimensional aggregates of sparse data using wavelets. In *Proceedings ACM SIGMOD International Conference on Management of Data*, pages 193–204. ACM Press, 1999.

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