Lattice Histograms: a Resilient Synopsis Structure

Panagiotis Karras [#], Nikos Mamoulis [*]

[#]*Department of Informatics, University of Zurich*
CH-8050 Zurich, Switzerland
karras@ifi.uzh.ch

[*]*Department of Computer Science, University of Hong Kong*
Pokfulam Road, Hong Kong, China
nikos@cs.hku.hk

***Abstract*— Despite the surge of interest in data reduction techniques over the past years, no method has been proposed to date that can always achieve approximation quality preferable to that of the optimal plain histogram for a target error metric. In this paper, we introduce the Lattice Histogram: a novel data reduction method that discovers and exploits any arbitrary hierarchy in the data, and achieves approximation quality provably at least as high as an optimal histogram for any data reduction problem. We formulate LH construction techniques with approximation guarantees for general error metrics. We show that the case of minimizing a maximum-error metric can be solved by a specialized, memory-sparing approach; we exploit this solution to design reduced-space heuristics for the general-error case. We develop a mixed synopsis approach, applicable to the space-efficient high-quality summarization of very large data sets. We experimentally corroborate the superiority of LHs in approximation quality over previous techniques with representative error metrics and diverse data sets.**

I. Introduction

A variety of data management applications call for the quick and efficient reduction of a very large amount of data into a compact synopsis that achieves high accuracy of approximation. Specific applications that give rise to a sustained interest in the area include OLAP/DSS systems [1], approximate query answering [2], [3], [4], [5], cost-based query optimization [6], time-series indexing [7], data mining [8] and, most recently, distributed stream monitoring [9]. In all problem formulations, the goal is to minimize an approximation error over the original data within a given space budget. Past research has led the way from conventional approximation techniques such as histograms [10], [11], [12], [13], [14], [4], [3], [15], [16], [17], [18], [19] and Haar wavelets [6], [1], [5], [20], [21], [22], [23], [24], [25], [18], [26], [27] to more sophisticated structures such as compact hierarchical histograms [9] and the Haar$^+$ tree [28]. However, existing research has largely neglected a cardinal question of the area, well expressed in a call to arms by Ioannidis [29]: the interaction between histograms and indices presents opportunities but also several technical challenges that need to be investigated. The main advantage of a plain histogram in relation to an index structure is its freedom to choose the most appropriate bucket boundaries for the data at hand. Still, the strength of an index structure is its ability to exploit an underlying hierarchy in the summarized data set, looking beyond local interrelations. Recent research has attempted to create less restrictive synopsis structures in two independent routes [9], [28]; both experimentally demonstrate that the structures they propose can, in certain circumstances, achieve higher approximation quality than an optimal histogram. Yet their results are valid only for data with particular characteristics. The quality of approximation they achieve in relation to an optimal histogram is not *provably* superior for any data set; indeed, as we show, it can be much worse. Besides, both [9] and [28] impose an ad hoc *predefined* hierarchy on the data, by default a hierarchy of dyadic intervals. Such a hierarchy may not be the most appropriate.

In this paper, we introduce Lattice Histograms (LH): a resilient index structure for data compaction that addresses the above shortcomings. It allows for the detection of a most suitable hierarchy in the data set under compaction and the derivation of a high-accuracy approximation based on that hierarchy; hence it is superior to fixed-hierarchy methods [9]; moreover, it achieves approximation quality at least as high as an optimal plain histogram. We propose an approximation scheme and a reduced-memory heuristic for synopsis construction with this structure, and we experimentally verify the advantage of LH over previous techniques.

II. Background and Related Work

Past research has established two principal methods for the construction of high-quality data approximations with deterministic guarantees. The former, *histograms*, creates buckets of contiguous values that are approximated by a representative value. The latter utilizes an appropriate hierarchical *data structure* for concise data representation. Under both approaches, given an n-size data vector $\mathbf{D} = \langle d_0, d_1, \ldots, d_{n-1} \rangle$, the problem is to devise a representation $\hat{\mathbf{D}}$ of $\mathbf{D}$ using at most B space, so that a given error metric in the approximation is minimized. A *normalized*, weighted Minkowski-norm error metric, $\mathcal{L}_p^w(\hat{\mathbf{D}}, \mathbf{D}) = \left(\sum_i \frac{(w_i |\hat{d}_i - d_i|)^p}{n} \right)^{\frac{1}{p}}$, covers most practically interesting point-wise error metrics; $\hat{d}_i$ is the reconstructed value for d_i and w_i a related weight; for relative error, $w_i = \frac{1}{\max\{|d_i|, S\}}$, where $S > 0$ is a sanity bound that prevents small values from dominating the result [20], [22]. The techniques in this paper are applicable to any monotonic distributive error metric, defined as in [22], [28].

A. Plain One-dimensional Histograms

A plain *histogram* divides $\mathbf{D}$ into $B \ll n$ disjoint intervals $[b_i, e_i]$, $1 \leq i \leq B$ called *buckets* or *segments* and attributes a single value v_i to each of them that approximates all consecutive values therein, d_j, $j \in [b_i, e_i]$. In a *dense* histogram these intervals are successive; in a *sparse* histogram there may be void areas between them. A single bucket (segment) can be expressed by the triplet $s_i = \{b_i, e_i, v_i\}$. Given a target metric, the best value of v_i is defined as a function[1] of the data in $[b_i, e_i]$. $3B$ numbers suffice to represent a sparse B-bucket histogram; $2B$ values represent a dense B-bucket histogram. Initial work on histograms focused on heuristics [11], [12], [13]. [14] presented an $O(n^2B)$ dynamic-programming scheme that derives $\mathcal{L}_2$-error-optimal (dense) bucket boundaries. Its basic observation is that the b-optimal histogram for a data vector $\mathbf{D}$ can be recursively derived from the space of $(b-1)$-optimal partitionings of all prefix vectors of $\mathbf{D}$. In fact, the solution of [14] is a special case of the line-segmentation algorithm introduced in [30]. We emphasize that, for an *arbitrary* error metric, this algorithm needs $O(n^3B)$ time, higher than the $O(n^2B)$ of the $\mathcal{L}_2$ case; this issue re-appears in Table II, Section V-E. Still, [17] proposed efficient methods specialized for several specific metrics. Later, [18] introduced a generic space-efficiency paradigm applicable on these histogram construction algorithms.

B. Hierarchical Synopsis Structures

Alternative research has studied index structures that represent the data in consecutive hierarchical levels of detail.

1) The Haar Wavelet Hierarchy: The Haar wavelet hierarchy can be visualized through a complete binary tree, the *Haar tree* [6]. The coefficient in the Haar tree root node contains the overall average value and each other coefficient value c_i contributes the value $+c_i$ to all data values (leaves) in its *left* sub-tree and $-c_i$ to those in its *right* sub-tree. Hence each original data value is reconstructed by adding/subtracting the coefficients in the path towards its position. A *Haar wavelet synopsis* of $\mathbf{D}$ is a vector $\hat{\mathbf{Z}}$ of $B \ll n$ non-zero $\langle i, c_i \rangle$ terms, such that its inverse wavelet transform $\hat{\mathbf{D}} = \mathcal{W}^{-1}\left(\hat{\mathbf{Z}}\right)$ approximates $\mathbf{D}$. The computation of an optimal Haar wavelet synopsis is computationally easiest for the $\mathcal{L}_2$ error. Still, past research has tackled the more demanding version of the problem for non-Euclidean error metrics [20], [21], [22], [23], [25], [18], [26], leading to the Haar$^+$ tree, which supersedes Haar wavelet models.

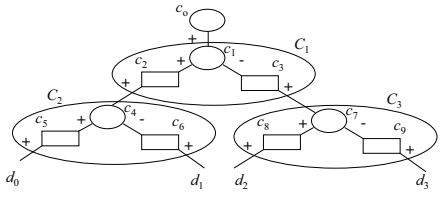


Fig. 1. An One-Dimensional Haar$^+$ Tree

[1]For $\mathcal{L}_1$ it is the median of the values in $[b_i, e_i]$ [19], for $\mathcal{L}_2$ their mean [14], for $\mathcal{L}_\infty$ the mean of the maximum and minimum value among them, while [17] analyzes the respective relative error cases.

2) The Haar$^+$ Tree: The Haar$^+$ tree [28] extended the Haar wavelet hierarchy by allowing for extra coefficient values in the structure, which contribute their (signed) value to a single dyadic interval alone. Figure 1 depicts a simple one-dimensional Haar$^+$ tree that may approximate a four-element data set $\{d_0, d_1, d_2, d_3\}$; it contains a single root coefficient node c_0 that contributes its value to all approximated data values, followed by a binary tree of triads (C_1, C_2 and C_3), which substitute the single non-root coefficients of the classical Haar tree. In each triad (e.g., C_1), the *head coefficient* (e.g., c_1) contributes its value positively to its left sub-tree and negatively to its right sub-tree. The left (e.g., c_2) and right (e.g., c_3) *supplementary coefficients* contribute their values positively only in the single subinterval that they affect (e.g., c_2 contributes positively to d_0 and d_1 only). An *optimal* synopsis of space B for an error metric $\mathcal{E}$ places B non-zero coefficient values at any positions in the Haar$^+$ tree so that $\mathcal{E}$ is minimized. Haar$^+$ not only increases the accuracy of approximation in relation to the simple Haar tree, but also allows for faster synopsis construction [28].

3) Compact Hierarchical Histograms: The Compact Hierarchical Histogram (CHH), proposed by [9], defines a binary hierarchy of intervals and selects an optimal subset of nodes to represent a data set. In fact, it can be easily shown that a binary CHH is equivalent to a Haar$^+$ tree with only supplementary (and root) coefficients. [9] observed that the calculation of the optimal value to retain on a node per se is computationally hard, and proposed CHH construction heuristics. This approach comes in contrast to the one of [26], [28], in which a quantized set of possible values is examined, allowing for an approximation guarantee. The winning Greedy CHH heuristic [9] improves upon an overlapping partitioning, in which the candidate assigned value at a CHH node is the optimal value for the whole data interval under its scope (as in a plain histogram bucket [14], [17], [19]), but not for the value set it *actually* approximates; the heuristic uses the optimal occupied node positions for such an overlapping partitioning for the target error metric, but adjusts the values assigned to them so as to be optimal for the actually approximated data set (i.e., a subset of the data under the node's scope); the result is a *longest-prefix-match partitioning*. [9] tested the CHH at approximating Internet traffic data, but emphasized that it can be used in a broad range of applications. In fact, as pointed out in [31], [28], hierarchical synopsis structures (such as the Haar$^+$ tree and its special case, the CHH) are most suitable at approximating *discontinuous* data sets. [9] suggested that the inherent IP address hierarchy provides a predefined hierarchy suitable for the representation of quantities measured over them. However, the IP address hierarchy is not necessarily related to the relationship of data values measured over them. In fact, any *predefined* hierarchy (i.e., structure where larger buckets contain smaller ones) imposes an arbitrary constraint on the approximation problem. The most appropriate hierarchical (i.e, bucket-overlap) pattern for a given data set is an *unknown*, not a *given*, of the problem.

C. Multidimensional Histograms

Related research has strived to extend the histogram idea to multiple dimensions, and to build hierarchical structures at that. [32] introduced a multidimensional version of the equi-depth histogram of [33]. [13] introduced MHist, a multidimensional histogram generalizing the one-dimensional MaxDiff heuristic of [12]. [34] introduced GenHist, which allows unrestricted overlap among multidimensional buckets extracted from progressively coarser grids over the data set. [35] presented a multidimensional histogram built by analyzing query results, and [36] extended this work with STHoles, which allows a bucket to contain another. Still, these techniques are based on heuristics, and do not provide approximation guarantees for general queries. Indeed, even in the two-dimensional case, constructing an optimal partitioning into arbitrary non-overlapping rectangular buckets is NP-hard [37]; algorithms with approximation guarantees have been provided for limited versions of the problem, with non-arbitrary buckets [38], [39], [40]; the problem with arbitrary *overlapping* buckets, as in [34], [36] is even harder. This work handles the static synopsis construction problem in the one dimension, as [14], [22], [28], [9], allowing bucket overlap, as [34], [36], [9]. Still, unlike [36], it does not depend on query feedback, hence it is *not* susceptible to errors for queries that target unseen data regions; it allows for *arbitrary* bucket sizes and positions, constrained neither by the imposition of grid structures over the data (as in [34]), nor by a predefined hierarchy (as in [9]); and, unlike all these works, it can provide tight approximation guarantees.

III. Motivation

Plain histograms are advantaged by their freedom to choose the most appropriate partitioning into buckets for the problem at hand, with no restrictions on their relative locations and sizes. On the other hand, they are constrained by their underlying *locality* assumption; a bucket is supposed to approximate *neighboring* values, which are expected to exhibit small variations. Thus, histograms are unable to exploit non-local interrelations. By contrast, hierarchical data compaction is advantaged by its ability to exploit non-local interrelations. Besides, a B-term Haar$^+$ representation defines B to $3B+1$ distinct consecutive intervals; a B-term CHH defines B to $2B+1$ intervals; in contrast, a B-sized histogram defines only B distinct intervals. Still, hierarchical representations are constrained by the predefined nature of their hierarchies; such hierarchies delimit the allowed buckets to a restrictive set; a hierarchy that does not fit into the predefined mold cannot be exploited by those structures. In consequence, the advantage of a hierarchical structure over a plain histogram does not apply at the task of approximating *continuous* data, such as those generated by natural processes or economic phenomena (e.g., a series of currency exchange rates or stock exchange indices). A plain histogram approximates such data more effectively than fixed-hierarchy structures For example, consider the data set $\mathbf{D} = \{4, 3, 5, 10, 12, 11, 11, 4\}$. A 3-term plain histogram can represent it as $\{4, 4, 4, 11, 11, 11, 11, 4\}$ with $\mathcal{L}_\infty = 1$ and $\mathcal{L}_1 = 0.5$. Neither a Haar$^+$ tree nor, therefore, a CHH can achieve this accuracy. An $\mathcal{L}_\infty$-optimal 3-term Haar$^+$ synopsis consists of the coefficients $\{c_0 = 6.5, c_8 = 4.5, c_{19} = 3.5\}$, producing the approximation $\{6.5, 6.5, 6.5, 6.5, 11, 11, 10, 3\}$ with $\mathcal{L}_\infty = 3.5$; an $\mathcal{L}_1$-optimal 3-term Haar$^+$ synopsis consists of the coefficients $\{c_0 = 4, c_8 = 7, c_{20} = 7\}$, producing the approximation $\{4, 4, 4, 4, 11, 11, 11, 4\}$ with $\mathcal{L}_1 = 1.125$; the same results can be achieved in this case by a CHH. Both [28] and [9] strived to annul the quality tradeoff between histograms and index structures, from different points of departure. [28] inserted histogram-like buckets into a Haar tree; still, it did not escape from the constraints imposed by the Haar hierarchy. Likewise, [9] built hierarchical interrelations on a plain histogram; thus, it imposed a fixed hierarchy anew at the expense of flexibility. In this paper, we introduce a structure that eliminates this quality tradeoff.

IV. The Lattice Histogram

We now introduce the Lattice Histogram (LH), a resilient synopsis structure that combines the strengths of, and goes beyond, plain histograms and fixed-hierarchy techniques. Figure 2 depicts an LH structure that may be used for summarization of an eight-element data set $\{d_0, \ldots, d_7\}$. The structure contains $\frac{n(n+1)}{2} = 36$ nodes, $\{c_0, \ldots, d_{35}\}$, layered in $n = 8$ levels, $\{\ell_1, \ldots, \ell_8\}$. A node c_i at level ℓ_k ($k = 1, \ldots, n$, counting from the top) *affects* an interval $\mathcal{I}_i$ of length $n-k+1$. There are k intervals of size $n-k+1$ in a data vector of size n, hence the k^{th} level contains k nodes. Each node has two children in the successor level, if such exists, and two parent nodes in the predecessor level, except for *edge* nodes, who have one parent node only. For instance, the single parent of node c_6 (Figure 2) is c_3; c_{10} and c_{11} are its children; and it affects the interval $\mathcal{I}_6 = \{d_0, \ldots, d_4\}$. On the other hand, c_7 has parents c_3 and c_4, children c_{11} and c_{12}, and affects $\mathcal{I}_7 = \{d_1, \ldots, d_5\}$.

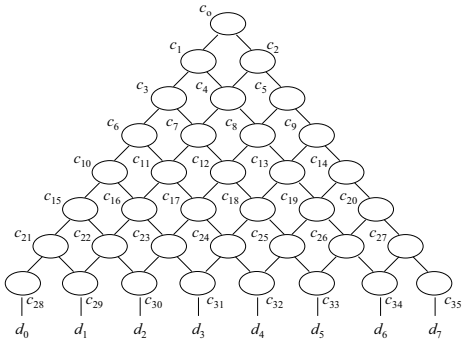
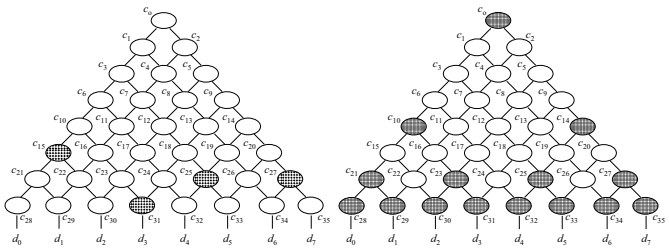


Fig. 2. A Lattice Histogram Structure

An LH representation of a given data vector $\mathbf{D}$ in a space budget B assigns non-zero values to, or *occupies*, a set of B nodes in the lattice structure. In our data representation mechanism, the intervals $\mathcal{I}_i$, $\mathcal{I}_j$ defined by two occupied nodes c_i, c_j may *contain* each other; however, non-containing *overlap* between occupied node intervals is not allowed; hence, no node is allowed to have two occupied ancestors such that one of them is not ancestor of the other. For example, nodes c_9 and c_{11} cannot be both occupied, as they are both ancestors of c_{24} without one of them being an ancestor of the other; if

node c_4 in Figure 2 is occupied, then any of its *descendant* nodes can be occupied; apart from these, only those nodes that either contain $\mathcal{I}_4 = \{d_1, \ldots, d_6\}$ or are disjoint to it may be occupied: c_0, c_1, c_2, c_{28} and c_{35}. A data item in an LH can be reconstructed as the value of the lowest occupied node affecting it in $O(\log B)$, using an appropriate interval tree. The *optimal* LH representation of $\mathbf{D}$ in a space budget B is the one that minimizes a given error metric $\mathcal{E}$. For example, for the data vector $\mathbf{D} = \{4,3,5,10,12,11,11,4\}$ the 2-term LH representation that minimizes $\mathcal{L}_1$, $\mathcal{L}_2$, and $\mathcal{L}_\infty$, consists of the occupied node set $\{c_0{=}4, c_{13}{=}11\}$ and yields the approximation $\hat{\mathbf{D}} = \{4,4,4,11,11,11,11,4\}$ with $\mathcal{L}_1 = 0.5$, $\mathcal{L}_2 = \sqrt{.5}$, $\mathcal{L}_\infty = 1$. Neither the Haar$^+$ (or CHH) structure, nor a plain histogram can achieve such accuracy of approximation. For example, the $\mathcal{L}_\infty$-optimal 2-term histogram for this vector approximates it as $\{4,4,4,8,8,8,8,8\}$ with $\mathcal{L}_\infty = 4$; an $\mathcal{L}_1$-optimal 2-bucket histogram is $\{4,4,4,11,11,11,11,11\}$ with $\mathcal{L}_1 = 1.375$. Likewise, an $\mathcal{L}_\infty$-optimal 2-term Haar$^+$ approximation (see Section II-B.2) is $\{c_0 = 6, c_3 = 2\}$ with $\mathcal{L}_\infty = 4$, while an $\mathcal{L}_1$-optimal 2-term Haar$^+$ approximation is $\{c_0 = 4, c_8 = 7\}$ with $\mathcal{L}_1 = 2$. The same results hold for a CHH. We emphasize the following points:

- A plain histogram [11], [14], [17] is a special case of an LH. In particular, it is an LH in which containment between occupied node intervals is disallowed. Figure 3a shows a subset of four occupied nodes in the LH of Figure 2 that make a (dense) plain histogram.



(a) (b)

Fig. 3. LH nodes that make (a) a plain histogram, (b) a full binary CHH

- A CHH [9] is also a special case of an LH. In particular, it is an LH limited to a specific, pre-selected, hierarchy of nodes. Figure 3b shows the subset of nodes in the LH of Figure 2 that form a full binary CHH.
- In terms of sparse approximation theory [41], [25], the LH provides an approximation using a *redundant* dictionary of $O(n^2)$ vectors; it requires $2\log n$ bits per index. A plain histogram that approximates a *sparse* one-dimensional array also requires $2\log n$ bits to store the boundaries of each bucket.

A. Definitions and Properties

A Lattice Histogram is a sparse array $\mathbf{L}$ of $N = \frac{n(n+1)}{2}$ elements, arranged in a lattice, that represents a data vector $\mathbf{D}$ of n elements $\{d_0, \ldots, d_{n-1}\}$. The lattice is configured so that the index f_k of the first node c_{f_k} in level k is the $(k{-}1)^{\text{th}}$ triangular number, $f_k = \frac{k(k-1)}{2}$. Therefore, node c_i resides in level $k = \lfloor \frac{1+\sqrt{8i+1}}{2} \rfloor$ and its children are nodes c_{i+k} and c_{i+k+1}. A data item d_j of the represented data vector $\mathbf{D}$ has a parent node c_i in the last level n of the lattice, such that $i{+}n{=}j{+}N$. An LH obeys the following property.

Property 1: **LH Property** Let c_i and c_j be two occupied nodes in an LH $\mathbf{L}$, and $\mathcal{I}_i$, $\mathcal{I}_j$ be their respective intervals. If $\mathcal{I}_i \cap \mathcal{I}_j \neq \emptyset$, then $\mathcal{I}_i \subset \mathcal{I}_j$ or $\mathcal{I}_j \subset \mathcal{I}_i$.

Nepots and Nepotic Descendants The *nepot* of a node c_i in level $k{=}\lfloor \frac{1+\sqrt{8i+1}}{2} \rfloor$ of an LH is the node $c_{i_{nep}} {=} c_{i+2k+2}$, that is, the middle of the three descendants of c_i in level $k{+}2$, if such exists. The nepot of c_i and the descendants thereof are called *nepotic descendants* of c_i. The only items in $\mathcal{I}_i$ which are not affected by the nepot of c_i are the leftmost $c_{i_{lm}}$ and rightmost $c_{i_{rm}}$ item in that interval.

Complementary nodes and Linear Descendants A pair of LH nodes c_j and c_k is *complementary* with respect to another node c_i if and only if $\mathcal{I}_j \cap \mathcal{I}_k = \emptyset$ and $\mathcal{I}_i = \mathcal{I}_j \cup \mathcal{I}_k$. A node c_j that is member of a complementary pair with respect to c_i is a *linear descendant* of c_i, and c_i is a *linear ancestor* of c_j. A node c_i in level $k = \lfloor \frac{1+\sqrt{8i+1}}{2} \rfloor$ has $n - k$ pairs of linear descendants; we denote the leftward member of such a pair in level $k + \ell$ as $c_{i_{L(k+\ell)}}$ and its rightward complementary node as $c_{i_{R(n-\ell+1)}}$, for $\ell \in \{k+1, \ldots, n\}$.

In Figure 2, c_{13} is the nepot of c_5, and nodes c_{16} and c_{25} are complementary with respect to node c_7. All descendants of a node are divided into a group of linear and one of nepotic descendants. Figure 4 illustrates these groups for a selected node, along with an example pair of complementary nodes.

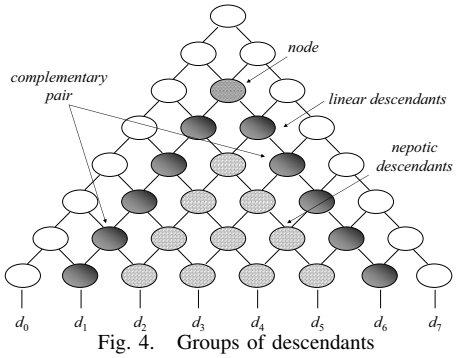


Fig. 4. Groups of descendants

Given an LH $\mathbf{L}$, let $\|\mathbf{L}\|$ be the number of occupied nodes in it. The following theorem shows the redundancy of occupying the linear descendants (hence, symmetrically, the linear ancestors) of an occupied node.

Theorem 1: Any LH representation $\mathbf{L}$ is reducible to an at least equally sparse LH $\mathbf{L}'$, in which no linear descendant of an occupied node is occupied, and $\|\mathbf{L}'\| \leq \|\mathbf{L}\|$.

Proof: Let c_i be an occupied node in $\mathbf{L}$ and c_j be an occupied linear descendant thereof. Let c_k be the complementary node of c_j with respect to c_i. Then, given that node c_j is occupied, no node which is a linear descendant of c_i and an ancestor of c_k can be occupied; if such a node were occupied, then its affected interval would overlap with but neither contain nor be contained by the affected interval $\mathcal{I}_j$ of occupied node c_j; hence the defining property of an LH (Property 1) would be violated. Concerning node c_k itself, we distinguish the following cases: (i) if c_k is occupied, then the configuration $\{c_i = a, c_j = b, c_k = c\}$ is equivalent to

$\{c_i = 0, c_j = b, c_k = c\}$; in this case, the occupation of c_i is of no consequence for the approximation, as all the values in its scope have occupied affecting nodes in lower levels of the lattice; (ii) if c_k is not occupied, then the configuration $\{c_i = a, c_j = b, c_k = 0\}$ is equivalent to $\{c_i = 0, c_j = b, c_k = a\}$. The value assigned to c_i can be simply forwarded to c_k (recall that linear descendants of c_i which are ancestors of c_k cannot be occupied); given that c_j is occupied, the value interval c_i actually affects is exactly the interval affected by c_k. This substitution process is repeated for all other cases, terminating when the bottom of the lattice is reached. Hence, any instance of an occupied node c_i with an occupied linear descendant can be reduced to an equivalent, not less sparse, configuration where c_i is not occupied. In effect, any LH $\mathbf{L}$ can be reduced to an at least equally sparse LH $\mathbf{L}'$, in which no linear descendant of an occupied node is occupied and $\|\mathbf{L}'\| \leq \|\mathbf{L}\|$. ■

Corollary 1: The optimal B-term LH representation $\mathbf{L}$ of a data vector $\mathbf{D}$ that minimizes a given error measure $\mathcal{E}$ can be expressed as an LH in which occupied nodes have no occupied linear descendants.

In the next section, we proceed to construct a dynamic programming approximation scheme for the optimal LH representation of a data vector $\mathbf{D}$ based on Corollary 1.

Incoming value An *incoming value* v to an LH node c_i with affected interval $\mathcal{I}_i$ is the value assigned to the lowest occupied ancestor node of c_i, whose affected interval contains $\mathcal{I}_i$. For a data item, the incoming value definition corresponds to a reconstructed value. The incoming value to the descendants of an occupied LH node c_i is defined by c_i alone. Other ancestors of those descendants do *not* contribute to that incoming value; their contribution is either canceled by c_i, or prohibited by the LH Property, or would be redundant according to Theorem 1.

Figure 5 depicts the implications of occupying a node to nodes in its periphery with whom it shares descendants.

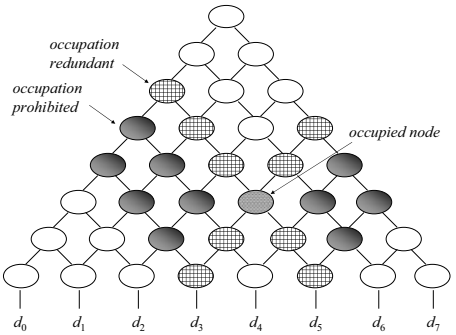


Fig. 5. Prohibited and redundant occupation of nodes

V. Lattice Histogram Computation

Problem 1: Given a data vector $\mathbf{D}$ and a monotonic distributive error metric $\mathcal{E}$, construct an LH representation $\mathbf{L}$ of $\mathbf{D}$ with B occupied nodes that produces an approximation $\hat{\mathbf{D}}$ of minimal error $f_{\mathcal{E}}(\|\mathbf{D} - \hat{\mathbf{D}}\|)$.

To solve this problem, we have to determine the optimal positions and values for B occupied nodes. As an occupied node need not have occupied linear descendants, two strategic choices are available on a node c_i: either c_i is occupied and the rest of the problem is solved on its nepotic descendants, or c_i is not not occupied and the rest of the problem is solved among a complementary pair of its linear descendants; this approach can result into *any* eligible combination of occupied descendants nodes of c_i conforming to the LH property. Let v be an incoming value at node c_i in level k, b be an amount of available space allocated to c_i and its descendants and $Q(i, v, b)$ the optimal choice to be made at c_i in that case. The full optimal solution can be derived in a bottom-up process that calculates $Q(i, v, b)$ on each node, for each possible v and b. The interval affected by c_i contains $n-k+1$ elements, hence c_i and its descendants do not need to use more than $n - k + 1$ occupied nodes; thus the domain of b for c_i is $|D_i| = \min\{B, n-k+1\}$. We delimit the domain of v by quantizing it into multiples of a resolution step δ. Let M (m) be the maximum (minimum) value in $\mathbf{D}$, and $\Delta = |M-m|$. Then the domain of v for any c_i is $\mathcal{S} \subset [m, M]$; values outside this interval are not useful for approximation; hence $|\mathcal{S}| = \frac{\Delta}{\delta}$. Similarly, the cardinality of the domain of allowed assigned values z at node c_i is $|\mathcal{S}_i| = O(\frac{\Delta}{\delta})$. Table I summarizes our notation. We now define dynamic programming recursive schemes for LH construction. We distinguish a general-purpose algorithm for any monotonic distributive error metric, and a specialized algorithm that achieves higher time- and space-efficiency for maximum-error metrics.

symbol	meaning
$\mathbf{D}$	Summarized data vector
$\mathbf{L}$	Optimized LH representation of $\mathbf{D}$
c_i	Node in $\mathbf{L}$
$\mathcal{I}_i$	Data interval affected by c_i
v	Incoming value to c_i
z	Value assigned to c_i
m (M)	Global minimum (maximum) in $\mathbf{D}$

TABLE I
Notation Used

A. General Case

Our general algorithm computes and tabulates all values of the fundamental $Q(i, v, b)$ function It examines all allowed distributions of the allocated space b on node c_i with incoming value v, and selects the best of them. The solution is established after $Q(0, 0, B)$ is computed, at which point all choices corresponding to that solution can be traced within the tabulation for the extraction of $\mathbf{L}$. As we have discussed, if a node c_i is occupied, then no linear descendant of c_i need be occupied. Therefore, the $b-1$ available space left can be all allocated to the nepotic descendants of c_i, i.e., to $c_{i_{nep}}$. In that case, all possible (quantized) assignments of an approximation value z at c_i need[2] to be examined. The incoming value to $c_{i_{nep}}$ is z and the leftmost (rightmost) item in $\mathcal{I}_i$, c_{il_m} (c_{ir_m}), which is not affected by the nepot, is approximated by z. On the other hand, if c_i is left unoccupied, then b space can be shared between any of the $n-k$ complementary pairs of its linear descendants, $k = \lfloor \frac{1+\sqrt{8i+1}}{2} \rfloor$. Then the same incoming value v is forwarded to the two nodes of the chosen pair

[2]The values this node will eventually approximate cannot be known in advance, hence all possible assigned values are examined

that shares the allocated space. Hence $Q(i,v,b)$ entries are polymorphic; an entry contains: (i) the δ-optimal value z to assign at c_i (possibly none); (ii) the minimum error $E(i,v,b)$ thus achieved; when $z{=}0$, it also contains (iii) a level index $\hat{\ell}$ for the chosen best complementary linear descendants pair $c_{i_{L(k+\hat{\ell})}}$ and $c_{i_{R(n-\hat{\ell}+1)}}$, $\hat{\ell}\in\{1,\ldots,n{-}k\}$; and (iv) the amount of space b_L out of b allocated to $c_{i_{L(k+\hat{\ell})}}$. A recursive procedure MinError emerges, which computes $E(i,v,b)$ as:

$$E(i,v,b)=\min\begin{cases}\min\limits_{z\in\mathcal{S}_i}\begin{cases}E\left(i_{lm},z,0\right)+\\E\left(i_{nep},z,b{-}1\right)+\\E\left(i_{rm},z,0\right)\end{cases}\\\min\limits_{1\le\ell\le n-k,b'\in D_i}\begin{cases}E\left(i_{L(k+\ell)},v,b'\right)+\\E\left(i_{R(n-\ell+1)},v,b{-}b'\right)\end{cases}\end{cases}$$

Error addition is used for the sake of simplicity; any distributive function G can be applied. This recurrence computes the least of two minima, one for each choice available at c_i. Eventually, the computed minimal error value is tabulated along with its accompanying choices of z, $\hat{\ell}$ and b_L. The recursion reaches its end cases in the two last LH levels. Error values are directly computed at the last level; the level before the last is also special, as it does not involve nepots. Error calculation is also straightforward for $b{=}0$.

Complexity Analysis There are k nodes in level k; each node has $O(\frac{\Delta}{\delta}\min\{B,n-k\})$ $Q(i,v,b)$ entries; all z values need to be checked only once in $O(\frac{\Delta}{\delta})$ time, since the exact value of v has no consequence on the optimal z. Hence, the time required for nepotic descendant computations is $O\left(\frac{\Delta}{\delta}\sum_{k=1}^{n}k\min\{B,n{-}k\}\right)=O\left(\frac{\Delta}{\delta}n^2B\right)$. The time for the computations involving linear descendants is $O\left(\frac{\Delta}{\delta}\sum_{k=1}^{n}k\sum_{b=1}^{\min\{B,n-k+1\}}\sum_{\ell=1}^{n-k}\min\{\ell,b\}\right)$, which adds up to $O\left(\frac{\Delta}{\delta}n^3B^2\right)$. In conclusion, the total time complexity is $O\left(\frac{\Delta}{\delta}n^3B^2\right)$. Similarly, the space complexity is $O\left(\frac{\Delta}{\delta}n^2B\right)$. If a distinction between *total space* and *working space* complexity is meaningful, as in [21], [22], we need only keep the arrays of all linear descendants plus the nepot of a node in the main memory at any time, hence the working space complexity becomes $O\left(\frac{\Delta}{\delta}nB\right)$.

B. Maximum-Error Case

The problem of minimizing a maximum-error metric, such as $\mathcal{L}_\infty$, has a special practical interest, since such metrics provide intuitive *deterministic error guarantees* for independent approximate values [21], [22], [23]. Moreover, with this problem we can follow a more memory-sparing approach; we exploit the solution to the dual, *error-bounded* problem in order to solve the *space-bounded* problem that we are interested in. Such an approach was used in [25] for the restricted Haar wavelet synopsis problem; in that case, it did not furnish a space complexity benefit; the algorithm for the space-bounded problem was already $O(n)$ (see Table II that follows). Still, in the present case of Lattice Histograms this technique delivers a crucial complexity advantage. The potential of this technique to deliver such an advantage in other synopsis construction problems has been outlined in [42].

Problem 2: Given a data vector $\mathbf{D}$ and an error bound ϵ for a maximum-error metric $\mathcal{E}_{max}$, construct an LH $\mathbf{L}$ of $\mathbf{D}$ that produces an approximation $\hat{\mathbf{D}}$, such that $f_{\mathcal{E}_{max}}(\|\mathbf{D}{-}\hat{\mathbf{D}}\|)\le\epsilon$ and the number of occupied nodes B^* in $\hat{\mathbf{D}}$ is minimized. Of all representations with B^* non-zero terms satisfying ϵ, select the one with the minimal actual error $\epsilon^*\le\epsilon$.

This problem can be solved by a dynamic-programming recurrence analogous to the one of Section V-A. Let $S(i,v)$ be the minimum space that should be allocated to node c_i and its descendants in order for the error bound ϵ to be satisfied with incoming value v at c_i. The tabulation is now simpler; no distributions of allocated space need to be examined; we only tabulate over bucket values; this convenience renders both the time and, most significantly, the space complexity of this algorithm lower than the one of Section V-A. The solution is established after $S(0,0)$ is computed, at which point all choices corresponding to that solution can be traced within the tabulation for the extraction of $\mathbf{L}$. Each $S(i,v)$ entry now contains: (i) the δ-optimal value z to assign at c_i (possibly none); (ii) in the case that $z=0$, a level index ℓ for the chosen pair of complementary linear descendants $c_{i_{L(k+\ell)}}$ and $c_{i_{R(n-\ell+1)}}$, $\ell\in\{1,\ldots,n-k\}$; and (iii) the minimum space thus achieved. A recursive procedure MinSpace emerges, in which the value of $S(i,v)$ is computed as:

$$S(i,v)=\min\left\{\begin{array}{l}\min\limits_{z\in\mathcal{S}_i\left|\begin{smallmatrix}S(i_{lm},z)=0,\\S(i_{rm},z)=0\end{smallmatrix}\right.}\left\{S\left(i_{nep},z\right)\right\}+1,\\\min\limits_{1\le\ell\le n-k}\left\{\begin{array}{l}S\left(i_{L(k+\ell)},v\right)+\\S\left(i_{R(n-\ell+1)},v\right)\end{array}\right\}\end{array}\right\}$$

This recurrence follows the same pattern as the one of Section V-A. It differs in the absence of a b parameter and in the simplification of the case where the node c_i is occupied. Now the occupation of node c_i brings a $+1$ term in the space equation; besides, according to Theorem 1, node c_i may be occupied only by an assigned value z that approximates the values of its leftmost and rightmost linear descendants within the given error bound ϵ (otherwise one of its linear descendants would need to be occupied as well); this condition is expressed in the equation; the incoming value to the nepotic descendants of c_i is again its assigned value z. In the case that node c_i is not occupied, the recurrence follows the pattern of Section V-A, searching for the pair of complementary linear descendants that minimizes the required space; still, a b parameter does not exist. In addition, the recurrence optimizes, in secondary priority, the actual error achieved within the minimal space.

Our LH construction algorithm for maximum-error metrics invokes the MinSpace module by binary search in the domain of error. This method avoids a tabulation with respect to space B, hence pays in terms of both space- and time-efficiency. In our implementation, the seed value of the fluctuating error bound ϵ for the target maximum-error metric $\mathcal{E}_{max}$ is obtained as the $\mathcal{E}_{max}$-error of an equi-width B-bucket plain histogram of $\mathbf{D}$; this provides an upper bound for the B-optimal $\mathcal{E}_{max}$-error of an LH over $\mathbf{D}$. Since our solution minimizes the error within the δ-optimal space, this binary search yields the δ-optimal error when it converges to the space budget B. Still, space *less* than B may also achieve the B-optimal error. Thus,

in order to ensure the convergence of the search, our procedure performs an *optimality test* for each examined error bound that requires *less* than B space with actual error $\bar{\epsilon}$; it re-runs a variant of MinSpace in which the condition to be satisfied on each approximated value d_i is $\mathcal{E}_{max}(|\hat{d}_i - d_i|) < \bar{\epsilon}$ (with $<$ instead of $\leq$); if this variant requires *more* than B space, then the search can safely terminate; otherwise, it proceeds. Hence, the search terminates when it reaches an error bound that either requires an LH of exactly B space, or requires an LH of $\bar{B} < B$ space and actual error $\bar{\epsilon}$, while any error bound $\epsilon < \bar{\epsilon}$ requires $\tilde{B} > B$ space. When the tested bound ϵ is decreased, the minimum error derived for the previous bound is taken into account for determining the new bound. Figure 6 shows a pseudocode for this IndirectLattice algorithm.

Algorithm IndirectLattice(B)
Input: space bound B, n-data vector $[d_0, \ldots, d_{n-1}]$
Output: $\mathcal{E}_{max}$-error optimal B-sized LH
1. ϵ_u = $\mathcal{E}_{max}$-error of B-term equi-width histogram;
2. ϵ_l = 0;
3. $e_{low} = \epsilon_l$; $e_{high} = \epsilon_u$;
4. **while** (not finished)
5. $e_{mid} = (e_{high} + e_{low})/2$;
6. $\mathbf{L}$=MinSpace($\leq e_{mid}$); $\bar{B}$ = size of $\mathbf{L}$;
7. $\bar{\epsilon}$ = actual $\mathcal{E}_{max}$-error of $\mathbf{L}$; /* $\bar{\epsilon} \leq \epsilon$ */
8. **if** ($\bar{B} < B$)
9. $\tilde{\mathbf{L}}$=MinSpace($< \bar{\epsilon}$); $\tilde{B}$ = size of $\tilde{\mathbf{L}}$;
10. **if** ($\tilde{B} > B$) finished := 1; /* optimal result found */
11. **else** $e_{high} = \bar{\epsilon}$;
12. **else if** ($\bar{B} > B$) $e_{low} = e_{mid}$
13. **else** finished := 1; /* $\bar{B} = B$ */
14. **return** $\mathbf{L}$;

Fig. 6. Indirect LH construction

Complexity Analysis. MinSpace tabulates $O(\frac{\Delta}{\delta})$ entries per node; again, the optimal z value needs to be computed once for all v in $O(\frac{\Delta}{\delta})$ time. Hence, the time needed to range through all computations involving nepotic descendants is $O\left(\frac{\Delta}{\delta}n^2\right)$; likewise, the time needed to range through linear descendant computations is $O\left(\frac{\Delta}{\delta}\sum_{k=1}^{n} k(n-k)\right) = O\left(\frac{\Delta}{\delta}n^3\right)$. In conclusion, the time complexity of MinSpace is $O\left(\frac{\Delta}{\delta}n^3\right)$. The binary search adds an $O(\log \epsilon^*)$ factor to this complexity, where ϵ^* is the final optimal error[3] value; thus the total time complexity becomes $O\left(\frac{\Delta}{\delta}n^3 \log \epsilon^*\right)$. The space complexity is $O\left(\frac{\Delta}{\delta}n^2\right)$, and the working space is $O\left(\frac{\Delta}{\delta}n\right)$. Crucially, these complexities are independent of B.

C. Approximation Guarantee

The following theorem provides a guarantee for the approximation achieved with the presented LH algorithms for normalized Minkowski-distance error metrics, in the spirit of [26], [28].

Theorem 2: Consider a data set $\mathbf{D}$ of size n, summarized by an LH for the normalized Minkowski-distance $\mathcal{L}_p$ error in B terms. Let $\mathbf{L}^*$ be the optimal LH representation in $\mathbb{R}$. Let $\mathbf{L}_\delta$ be the optimal LH representation in the domain of multiples of δ. Let the derived error values be $\mathcal{E}^*$ and $\mathcal{E}_\delta$, respectively. Then, $\mathcal{E}_\delta \leq \mathcal{E}^* + \frac{\delta}{2}$.

Proof: Let $\mathbf{D}^*$ denote the approximation of $\mathbf{D}$ produced by $\mathbf{L}^*$. Let $\hat{\mathbf{L}}_\delta$ be the LH representation derived after rounding all coefficients in $\mathbf{L}^*$ to the nearest multiple of δ, and $\hat{\mathcal{E}}_\delta$ be the $\mathcal{L}_p$ error of the approximation $\hat{\mathbf{D}}$ produced by $\hat{\mathbf{L}}_\delta$. Since $\mathbf{L}_\delta$ is the $\mathcal{L}_p$-optimal δ-step representation, it follows that $\mathcal{E}_\delta \leq \hat{\mathcal{E}}_\delta$. However, according to the triangle inequality, $\hat{\mathcal{E}}_\delta \leq \mathcal{E}^* + \mathcal{L}_p(\mathbf{D}^*, \hat{\mathbf{D}})$. As each approximated data value is the single value assigned to the lowest ancestor node, each such value in $\hat{\mathbf{L}}_\delta$ has been rounded from its value in $\mathbf{L}^*$ by at most $\frac{\delta}{2}$. Therefore, $\mathcal{L}_\infty(\mathbf{D}^*, \hat{\mathbf{D}}) \leq \frac{\delta}{2}$. From the definition of the *normalized* Minkowski-distance norm it follows that $\mathcal{L}_p(\mathbf{D}^*, \hat{\mathbf{D}}) \leq \mathcal{L}_\infty(\mathbf{D}^*, \hat{\mathbf{D}})$. Thus, $\mathcal{E}_\delta \leq \mathcal{E}^* + \frac{\delta}{2}$. ■

D. Heuristic LH Computation

Our algorithm for general-error LH computation (Section V-A) provides the good approximation guarantees analyzed in Section V-C. However, its space complexity will engender difficulties with sizeable data sets. On the other hand, the space complexity for our maximum-error algorithm (Section V-B) is lower, as it evades the B factor. Still, we can exploit the maximum-error algorithm in order to build a heuristic solution for general error metrics. This solution runs the maximum-error algorithm for an appropriate[4] maximum-error metric. After the set of occupied nodes for the best maximum-error solution is established, the values assigned to these nodes are adjusted, so as to be optimal for the target error at hand. The value adjustment step of our heuristic is reminiscent of that performed by the greedy heuristic of [9]. However, our heuristic does not involve a low-quality overlapping partitioning; it utilizes a partitioning which is already of the longest-prefix-match type, but optimized for an associated maximum-error metric instead of the general error metric at hand.

E. Theoretical Comparison

Table II summarizes the evolution of complexity for synopsis construction algorithms. The practical $\mathcal{L}_1$ and $\mathcal{L}_\infty$ metrics are used for illustration; n is the data set size, B the space bound, δ the resolution step, $\mathcal{E}$ an upper bound for the target normalized Minkowski-norm error, Δ the difference of the minimum from the maximum value in the data set, and ϵ^* the optimal $\mathcal{L}_\infty$ error. The fractions with denominator δ express the cardinality of the examined set of incoming or assigned values. Space complexity expressions for [26], [9], [28] take into account their use of the space-efficiency technique of [18]. The cubic complexity of Lattice Histograms is comparable to original approaches for other techniques, as [14], [21], [26], and competitive towards the lower-quality k-holes CHH heuristic [9]; still, as we will see in the next section, the LH achieves consistently higher quality than previous approaches; hence, its cubic complexity is worthwhile in terms of synopsis quality. Moreover, in contrast to [26], [28], the time complexity of our LH construction algorithms does *not* depend quadratically on the cardinality factor $\frac{\Delta}{\delta}$; it grows linearly with it.

Figure 7 depicts a genealogy of synopsis structures. An arrow denotes that the destination structure *contains* the structure of origin: any representation that can be achieved with

[3]The log function expresses the dependence of running time on the derived error value; it is to be understood as a *growth* function, as in [25]; not as an algebraic function; $\epsilon^* \leq 1$ does *not* imply non-positive time.

[4]For example, the maximum relative error is an appropriate target metric if we wish to minimize the average relative error.

Technique	Time Complexity ($\mathcal{L}_1$)	Time Complexity ($\mathcal{L}_\infty$)	Space Complexity		Reference
Plain Histogram	$O\left(n^3 B\right)$		$O\left(nB\right)$		[14]
Plain Histogram	$O\left(n^2(B+\log n)\right)$	$O\left(nB\log^2 n\right)$	$O(n)$		[17], [18]
Restricted Haar	$O\left(n^2 B^2\right)$	$O\left(n^2 B\log B\right)$	$O\left(n^2 B\right)$		[21]
Restricted Haar	$O\left(n^2\log B\right)$	$O\left(n^2\right)$, $O\left(n^2\frac{\log\epsilon^*}{\log n}\right)$	$O(n)$		[18], [25]
Unrestricted Haar	$O\left(\left(\frac{\varepsilon}{\delta}\right)^2 n^3 B\right)$	$O\left(\left(\frac{\varepsilon}{\delta}\right)^2 n\log^2 B\right)$	$O\left(\frac{\varepsilon}{\delta}B\log\frac{n}{B}+n\right)$		[26]
CHH (k-holes)	$O\left(n^{k+1}B^2\log n\right)$	$O\left(n^{k+1}B\log B\log n\right)$	$O\left(n^k B\log^2 n\right)$		[9]
CHH (Greedy)	$O\left(nB^2\log n\right)$	$O\left(nB\log n\log B\right)$	$O\left(B\log^2 n+n\right)$		[9]
Haar$^+$	$O\left(\left(\frac{\Delta}{\delta}\right)^2 nB\right)$	$O\left(\left(\frac{\Delta}{\delta}\right)^2 n\log^2 B\right)$	$O\left(\frac{\Delta}{\delta}B\log\frac{n}{B}+n\right)$		[28]
Lattice Histogram	$O\left(\frac{\Delta}{\delta}n^3B^2\right)$	$O\left(\frac{\Delta}{\delta}n^3\log\epsilon^*\right)$	$O\left(\frac{\Delta}{\delta}n^2B\right)$	$O\left(\frac{\Delta}{\delta}n^2\right)$	**This work**

TABLE II

SUMMARY OF RESULTS FOR QUALITY-AWARE ONE-DIMENSIONAL SYNOPSIS CONSTRUCTION

the latter can also be achieved with the former. Hence, a restricted Haar wavelet synopsis [21], [22] is a special case of an unrestricted one [26]; a unrestricted Haar wavelet synopsis is a special case of a Haar$^+$ representation [28]; a CHH [9] is a special case of a Haar$^+$ representation as well. Similarly, a plain histogram [14], [17] is a special case of an LH; a CHH is a special case of an LH as well. We infer that the approximation quality achieved with an LH is bound to be at least as good as that achieved with a plain histogram or a CHH, subject to a sufficiently small resolution step δ. The quality comparison of an LH to the structurally independent Haar$^+$ approximation is of greatest experimental interest.

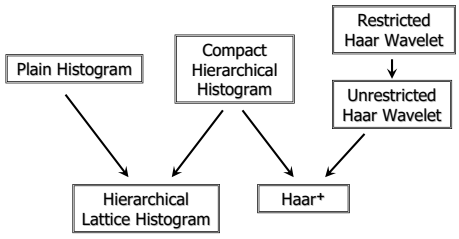


Fig. 7. Genealogy of Synopsis Structures

F. Approximating Very Large Data Sets

The space complexity for LH computation is quadratic in n; although lower than the the high-polynomial space complexity for an inferior-quality k-holes CHH [9] (Table II), this complexity is still bound to drain the memory resources when summarizing very large data sets. Therefore, we propose a more appropriate and practical approach: a large data set is first summarized by a *primary* space-efficient approximation technique, such as a plain histogram or a Haar$^+$ synopsis [14], [17], [18], [28]. In case the primary technique is a plain histogram, the data set is subsequently divided into smaller segments, using a selection of the boundaries established by it, and LH synopses are constructed for each of those segments, using the space budget that the primary histogram allocated to them. This process results into a quality enhancement in each segment, hence for the total data set. The size of the chosen segments is determined by the available memory resources. An integrated LH representation is derived by concatenating the representations of all segments. Figure 8 depicts schematically the segmentation of a data set in this case.

Alternatively, if the primary technique is a Haar$^+$ synopsis, then the lowest levels of the Haar$^+$ hierarchy are substituted by local LH structures. An LH synopsis is separately built for each of them, again using the space budget the primary Haar$^+$ approximation has allocated to each A *mixed* synopsis structure results, in which the overall approximation quality is increased. Figure 9 depicts such a *mixed* structure.

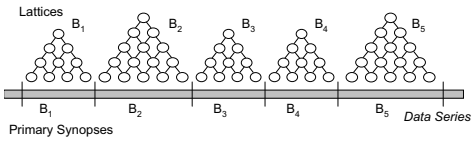


Fig. 8. Piece-wise LH application

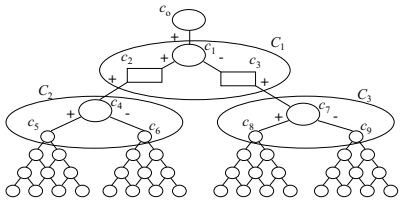


Fig. 9. Mixed Haar$^+$/LH structure

VI. EXPERIMENTAL VERIFICATION

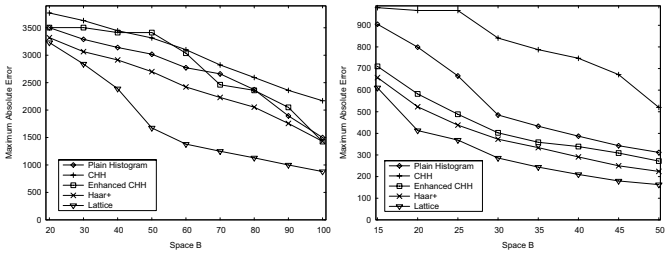
We now compare the LH to alternative synopsis techniques in terms of our focal question of quality. The quality achieved with a mixed synopsis (Section V-F) directly depends on the performance of the LH synopses it contains in relation to the structures they substitute. The following methods were implemented with g++ 3.4.3 and run on a 3.5GB machine:

- **Plain Histogram** The optimal histogram algorithms [14], [17], which provide an upper bound to the quality of approximate histograms [11], [12], [13], [15], [16], [19].
- **CHH** The winning Greedy CHH heuristic [9], which initially computes an overlapping partitioning (see Section II-B.3), where the candidate assigned value on a node is the optimal value for the data interval under its scope with the target metric. We have observed that a quality improvement can occur if the *median* values in those intervals (which are actually optimal for the $\mathcal{L}_1$ metric) are used instead. Medians guide the algorithm more robustly towards the occupation of good *positions*. We call this variant **Enhanced CHH**. We include these CHH schemes in our study for the sake of completeness, even though Haar$^+$ is bound to outperform them for sufficiently small δ. To our knowledge, this is the first experimental comparison between CHH and *optimal* plain histograms for non-$\mathcal{L}_2$ metrics [17], as well as between CHH and any other hierarchical synopsis; thus it supplements [9].
- **Haar$^+$** The synopsis construction model based on the Haar$^+$ tree [28], which supersedes [21], [22], [26].

- **Lattice** Our LH techniques.

Description of Data In order to assess the quality achieved with diverse synopses in several real-world environments, we have used two real-life data sets with hard to approximate bursts and discontinuities, as well as a real-world data set with continuity features. In order to allow the binary-interval-based Haar$^+$ and CHH techniques to perform at their best, we have used binary data sizes. The first data set[5] (FR), discussed in [43], is a sequence of the mean monthly flows for the Fraser River at Hope, B.C. The flows present periodic autoregression features, while they average at 2709 (standard deviation: 2123) and feature discontinuities (min value: 482, max value: 10800). We have used a 512-value prefix of it. The second data set[6] (FC) is extracted from a relation of 581,012 tuples describing the forest cover type for 30 x 30 meter cells, obtained from US Forest Service. FC contains the frequencies of the distinct values of attribute `aspect` in the relation. The frequencies average at 1613 (standard deviation: 730) and feature spikes of large values (min value: 499, max value: 6308). We have used a 256-value prefix of it. The third data set (DJIA) is the Dow-Jones Industrial Average (DJIA) data set[7] that contains closing values of the Dow-Jones Industrial Average index from 1900 to 1993. Negative values were removed. We used a 512-value subset of closing values from Apr. 14th, 1948 to Feb. 8th, 1950. The closing values average at 182 (standard deviation: 8.73) and exhibit both continuities and hierarchical patterns (min value: 161.6, max value: 205.03).

A. Synopsis Quality with non-Smooth Data

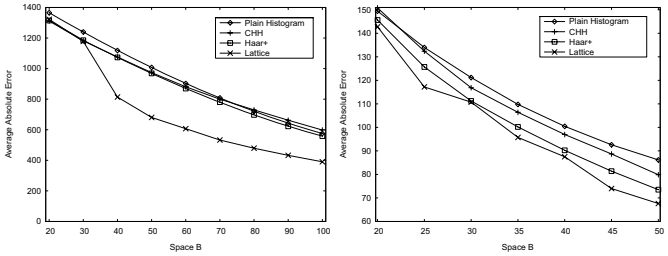


(a) FR (b) FC

Fig. 10. Quality comparison: $\mathcal{L}_\infty$

1) Maximum Absolute Error: In this experiment we evaluate the accuracy achieved with the $\mathcal{L}_\infty$ metric on the the FR and FC data sets. Figure 10 shows the results, in which the resolution value has been set[8] at $\delta = 50$ for the FR and $\delta = 10$ for the FC data set with both the Haar$^+$ and Lattice (LH) techniques. The Lattice histogram achieves the highest quality for both data sets; its advantage is particularly clear with the FR data set. As expected, the accuracy achieved with the CHH techniques is lower than that of LH and Haar$^+$. Our Enhanced CHH algorithm could outperform the regular CHH; with the FC data set, it outperforms the optimal histogram too, while the regular CHH does not; its performance is not as stable with the FR data set. The $\mathcal{L}_\infty$-optimal plain histogram quality itself is poor in relation to an LH, as expected. Most interestingly, the LH quality is consistently higher than that of the Haar$^+$ representation.

2) Average Error: We now evaluate the accuracy achieved with the $\mathcal{L}_1$ metric on the the same data sets and with the same resolutions. Figure 11 shows the results. Now the Lattice technique is the heuristic of Section V-D. Although this heuristic does not confer the same quality guarantees as its memory-intensive counterpart (Section V-A), it still achieves the highest quality in this experiment. The quality achieved with this heuristic on FR comes closer to that of other hierarchical techniques for the smallest values of B; this is due to the fact that the positions selected for a small space budget, albeit optimal for a maximum error metric, may not perform as well for another metric. However, this behavior is annulled for larger values of B, denoting that the heuristic selects well-chosen bucket positions. In this experiment there is no Enhanced CHH, as the regular uses the median values by default. With the FR data set, the CHH accuracy deteriorates as B grows, eventually becoming the lowest.

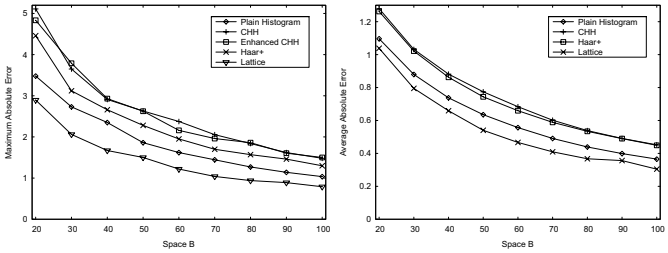


(a) FR (b) FC

Fig. 11. Quality comparison: $\mathcal{L}_1$

B. Synopsis Quality with Smooth Data

1) Maximum Absolute Error: We now examine the DJIA data set, which does not present as sharp discontinuities as those we have examined heretofore. We first assess the quality of approximation with $\mathcal{L}_\infty$ (Figure 12a); the resolution value was set at $\delta = 0.5$ with both Haar$^+$ and LH. Again, LH achieves the highest quality. Interestingly, the other techniques cannot match the optimal plain histogram.



(a) $\mathcal{L}_\infty$ (b) $\mathcal{L}_1$

Fig. 12. Quality comparison: DJIA

2) Average Error: Figure 12b shows the results with $\mathcal{L}_1$. The LH heuristic used in this experiment achieves the top quality again, while the disadvantage of the other hierarchical techniques in relation to the optimal plain histogram is clear. Results with other metrics, such as $\mathcal{L}_2$, were similar.

[5]Available at http://lib.stat.cmu.edu/datasets/fraser-river

[6]Available at http://kdd.ics.uci.edu/

[7]Available at http://lib.stat.cmu.edu/datasets/djdc0093

[8]Smaller values burdened the running time without significant quality increase; larger values were undermining the quality of the synopses.

VII. Discussion

The results on higher LH accuracy in relation to plain histograms and CHH were expected. The results in relation to the structurally independent Haar$^+$ tree are more interesting. Moroever, while the performance of the Haar$^+$ tree (and its simplified form, the CHH) in relation to a plain histogram varies depending on the nature of the summarized data, the LH achieves invariably higher quality than all contenders.

VIII. Conclusions

In this paper we have introduced the Lattice Histogram: a robust, resilient data structure for data approximation. The LH answers to the call for an investigation of the opportunities and challenges posed by the interaction between histograms and index structures. An LH identifies a most suitable hierarchy in a data set and uses it in order to approximate the given data. This structure combines both the advantages of a plain histogram over a hierarchical index structure, and those of an index structure over a histogram, in a single synopsis model; hence, it annuls the quality tradeoff between them. We designed approximation schemes for LH computation for both general error metrics and the special case of maximum-error metrics; we have employed the latter solution for a reduced-memory, high-quality heuristic for the general-error case. We have demonstrated that Lattice Histograms consistently achieve higher quality than the recently proposed Compact Hierarchical Histograms and the well-established optimal plain histograms, regardless of the nature of the summarized data set. In fact, both these models are special cases of an LH. Still, an LH can achieve higher quality than the structurally independent Haar$^+$ tree too. In the future we plan to design streaming heuristics for LH-based data approximation.

Acknowledgment

This work was supported by grant HKU 7155/06E from Hong Kong RGC.

References

[1] J. S. Vitter and M. Wang, "Approximate computation of multidimensional aggregates of sparse data using wavelets," in *SIGMOD*, 1999.
[2] S. Acharya, P. B. Gibbons, V. Poosala, and S. Ramaswamy, "Join synopses for approximate query answering," in *SIGMOD*, 1999.
[3] V. Poosala, V. Ganti, and Y. E. Ioannidis, "Approximate query answering using histograms," *IEEE Data Eng. Bull.*, vol. 22, no. 4, pp. 5–14, 1999.
[4] Y. E. Ioannidis and V. Poosala, "Histogram-based approximation of set-valued query-answers," in *VLDB*, 1999.
[5] K. Chakrabarti, M. Garofalakis, R. Rastogi, and K. Shim, "Approximate query processing using wavelets," *VLDB Journal*, vol. 10, no. 2-3, pp. 199–223, 2001 (also VLDB 2000).
[6] Y. Matias, J. S. Vitter, and M. Wang, "Wavelet-based histograms for selectivity estimation," in *SIGMOD*, 1998.
[7] K. Chakrabarti, E. Keogh, S. Mehrotra, and M. Pazzani, "Locally adaptive dimensionality reduction for indexing large time series databases," *TODS*, vol. 27, no. 2, pp. 188–228, 2002 (also SIGMOD 2001).
[8] T. Li, Q. Li, S. Zhu, and M. Ogihara, "A survey on wavelet applications in data mining," *SIGKDD Explorations Newsletter*, vol. 4, no. 2, pp. 49–68, 2002.
[9] F. Reiss, M. Garofalakis, and J. M. Hellerstein, "Compact histograms for hierarchical identifiers," in *VLDB*, 2006.
[10] Y. E. Ioannidis, "Universality of serial histograms," in *VLDB*, 1993.
[11] Y. E. Ioannidis and V. Poosala, "Balancing histogram optimality and practicality for query result size estimation." in *SIGMOD*, 1995.
[12] V. Poosala, Y. E. Ioannidis, P. J. Haas, and E. J. Shekita, "Improved histograms for selectivity estimation of range predicates," in *SIGMOD*, 1996.
[13] V. Poosala and Y. E. Ioannidis, "Selectivity estimation without the attribute value independence assumption," in *VLDB*, 1997.
[14] H. V. Jagadish, N. Koudas, S. Muthukrishnan, V. Poosala, K. C. Sevcik, and T. Suel, "Optimal histograms with quality guarantees." in *VLDB*, 1998.
[15] P. B. Gibbons, Y. Matias, and V. Poosala, "Fast incremental maintenance of approximate histograms," *TODS*, vol. 27, no. 3, pp. 261–298, 2002.
[16] A. C. Gilbert, S. Guha, P. Indyk, Y. Kotidis, S. Muthukrishnan, and M. J. Strauss, "Fast, small-space algorithms for approximate histogram maintenance," in *STOC*, 2002.
[17] S. Guha, K. Shim, and J. Woo, "REHIST: Relative error histogram construction algorithms." in *VLDB*, 2004.
[18] S. Guha, "Space efficiency in synopsis construction algorithms," in *VLDB*, 2005.
[19] E. Terzi and P. Tsaparas, "Efficient algorithms for sequence segmentation." in *SIAM SDM*, 2006.
[20] M. Garofalakis and P. B. Gibbons, "Probabilistic wavelet synopses," *TODS*, vol. 29, no. 1, pp. 43–90, 2004 (also SIGMOD 2002).
[21] M. Garofalakis and A. Kumar, "Deterministic wavelet thresholding for maximum-error metrics." in *PODS*, 2004.
[22] ——, "Wavelet synopses for general error metrics," *TODS*, vol. 30, no. 4, pp. 888–928, 2005.
[23] P. Karras and N. Mamoulis, "One-pass wavelet synopses for maximum-error metrics," in *VLDB*, 2005.
[24] Y. Matias and D. Urieli, "Optimal workload-based weighted wavelet synopses," *Theoretical Computer Science*, vol. 371, no. 3, pp. 227–246, 2007 (also ICDT 2005).
[25] S. Muthukrishnan, "Subquadratic algorithms for workload-aware Haar wavelet synopses," in *FSTTCS*, 2005.
[26] S. Guha and B. Harb, "Approximation algorithms for wavelet transform coding of data streams," in *SODA*, 2006 (also eprint arXiv:cs/0604097).
[27] A. Deligiannakis, M. Garofalakis, and N. Roussopoulos, "Extended wavelets for multiple measures," *TODS*, vol. 32, no. 1, 2007.
[28] P. Karras and N. Mamoulis, "The Haar$^+$ tree: a refined synopsis data structure," in *ICDE*, 2007.
[29] Y. E. Ioannidis, "Approximations in database systems," in *ICDT*, 2003.
[30] R. Bellman, "On the approximation of curves by line segments using dynamic programming," *Communications of the ACM*, vol. 4, no. 6, p. 284, 1961.
[31] A. Graps, "An introduction to wavelets," *IEEE Computational Sciences and Engineering*, vol. 2, no. 2, pp. 50–61, 1995.
[32] M. Muralikrishna and D. J. DeWitt, "Equi-depth histograms for estimating selectivity factors for multi-dimensional queries," in *SIGMOD*, 1988.
[33] G. Piatetsky-Shapiro and C. Connell, "Accurate estimation of the number of tuples satisfying a condition," in *SIGMOD*, 1984.
[34] D. Gunopulos, G. Kollios, V. J. Tsotras, and C. Domeniconi, "Selectivity estimators for multidimensional range queries over real attributes," *The VLDB Journal*, vol. 14, no. 2, pp. 137–154, 2005 (also SIGMOD 2000).
[35] A. Aboulnaga and S. Chaudhuri, "Self-tuning histograms: building histograms without looking at data," in *SIGMOD*, 1999.
[36] N. Bruno, S. Chaudhuri, and L. Gravano, "STHoles: a multidimensional workload-aware histogram," in *SIGMOD*, 2001.
[37] S. Muthukrishnan, V. Poosala, and T. Suel, "On rectangular partitionings in two dimensions: Algorithms, complexity, and applications," in *ICDT*, 1999.
[38] S. Khanna, S. Muthukrishnan, and S. Skiena, "Efficient array partitioning," in *ICALP*, 1997.
[39] S. Muthukrishnan and T. Suel, "Approximation algorithms for array partitioning problems," *Journal of Algorithms*, vol. 54, no. 1, pp. 85–104, 2005.
[40] F. Furfaro, G. M. Mazzeo, D. Saccà, and C. Sirangelo, "Hierarchical binary histograms for summarizing multi-dimensional data," in *ACM SAC*, 2005.
[41] A. C. Gilbert, S. Muthukrishnan, and M. J. Strauss, "Approximation of functions over redundant dictionaries using coherence," in *SODA*, 2003.
[42] P. Karras, D. Sacharidis, and N. Mamoulis, "Exploiting duality in summarization with deterministic guarantees," in *KDD*, 2007.
[43] A. McLeod, "Diagnostic checking of periodic autoregression models with application," *Journal of Time Series Analysis*, vol. 15, no. 2, pp. 221–233, 1994.