The Discrete Wavelet Transform and Wavelet Synopses

Minos Garofalakis
Technical University of Crete
minos@acm.org

SYNONYMS

None.

DEFINITION

Wavelets are a useful mathematical tool for hierarchically decomposing functions in ways that are both efficient and theoretically sound. Broadly speaking, the wavelet transform of a function consists of a coarse overall approximation together with detail coefficients that influence the function at various scales. The wavelet transform has a long history of successful applications in signal and image processing [11, 12]. Several recent studies have also demonstrated the effectiveness of the wavelet transform (and *Haar wavelets*, in particular) as a tool for approximate query processing over massive relational tables [2, 7, 8] and continuous data streams [3, 9]. Briefly, the idea is to apply wavelet transform to the input relation to obtain a compact data synopsis that comprises a select small collection of *wavelet coefficients*. The excellent energy compaction and de-correlation properties of the wavelet transform allow for concise and effective approximate representations that exploit the structure of the data. Furthermore, wavelet transforms can generally be computed in linear time, thus allowing for very efficient algorithms.

HISTORICAL BACKGROUND

A growing number of database applications require on-line, interactive access to very large volumes of data to perform a variety of data-analysis tasks. As an example, large Internet Service Providers (ISPs) typically collect and store terabytes of detailed usage information (NetFlow/SNMP flow statistics, packet-header information, etc.) from the underlying network to satisfy the requirements of various network-management tasks, including billing, fraud/anomaly detection, and strategic planning. This data gives rise to massive, multi-dimensional *relational data tables* typically stored and queried/analyzed using commercial database engines (such as, Oracle, SQL Server, DB2). To handle the huge data volumes, high query complexities, and interactive response-time requirements characterizing these modern data-analysis applications, the idea of effective, easy-to-compute *approximate query answers* over precomputed, compact *data synopses* has recently emerged as a viable solution. Due to the exploratory nature of most target applications, there are a number of scenarios in which a (reasonably-accurate) fast approximate answer over a small-footprint summary of the database is actually preferable over an exact answer that takes hours or days to compute. For example, during a "drill-down" query sequence in ad-hoc data mining, initial queries in the sequence frequently have the sole purpose of determining the truly interesting queries and regions of the database. Providing fast approximate answers to these initial queries gives users the ability to focus their explorations quickly and effectively, without consuming inordinate amounts of valuable system resources.

The key behind such approximate techniques for dealing with massive data sets lies in the use of appropriate *data-reduction techniques* for constructing compact synopses that can accurately approximate the important features of the underlying data distribution. The *Haar wavelet decomposition* is one such technique with deep roots in the fields of signal and image processing, that has recently found its way into database applications as an important approximate query processing tool.

SCIENTIFIC FUNDAMENTALS

Haar Wavelet Basics. Haar wavelets are conceptually simple, easy to compute, and have been found to perform well in practice for a variety of applications, ranging from image editing and querying to database selectivity estimation tasks. Consider a one-dimensional data vector A containing the $N = 8$ data values $A = [2, 2, 0, 2, 3, 5, 4, 4]$. The Haar wavelet transform of A can be computed as follows. The values are first averaged together pairwise to get a new "lower-resolution" representation of the data with the following average values $[2, 1, 4, 4]$. To restore the original values of the data array, additional *detail coefficients* must be stored to capture the information lost due to this averaging. In Haar wavelets, these detail coefficients are simply the differences of the (second of the) averaged values from the computed pairwise average, that is, $[2 - 2, 1 - 2, 4 - 5, 4 - 4] = [0, -1, -1, 0]$. No information has been lost in this process – it is simple to reconstruct the eight values of the original data array from the lower-resolution array containing the four averages and the four detail coefficients. Recursively applying the above pairwise averaging and differencing process on the lower-resolution array containing the averages, gives the following full transform:

The *wavelet transform* W_A of A is the single coefficient representing the overall average of the data values followed by the detail coefficients in the order of increasing resolution, i.e., $W_A = [11/4, -5/4, 1/2, 0, 0, -1, -1, 0]$ (each entry is called a

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]

wavelet coefficient). For vectors containing similar values, most of the detail coefficients tend to be very small; thus, eliminating them from the wavelet transform (i.e., treating them as zeros) introduces only small errors when reconstructing the original data, resulting in a very effective form of lossy data compression [12].

A helpful tool for conceptualizing the recursive Haar wavelet transform process is the *error tree* structure (shown in Figure 1(a) for the example array A). Each internal node c_i ($i = 0, \ldots, 7$) is associated with a wavelet coefficient value, and each leaf d_i ($i = 0, \ldots, 7$) is associated with a value in the original data array; in both cases, the index i denotes the positions in the (data or wavelet transform) array. For instance, c_0 corresponds to the overall average of A. The resolution levels l for the coefficients (corresponding to levels in the tree) are also depicted.

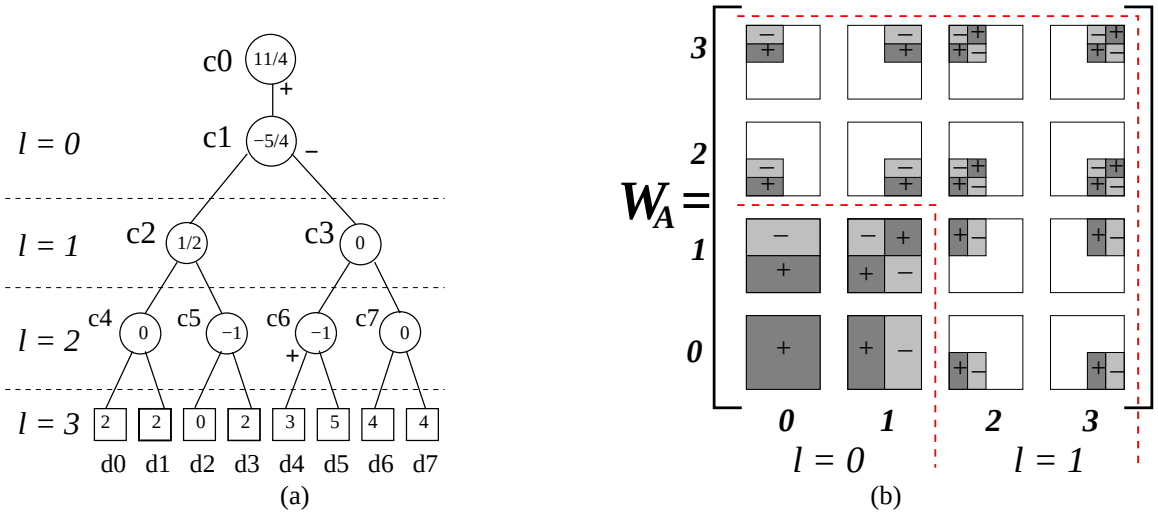

Figure 1: (a) Error-tree structure for the example data array A ($N = 8$). (b) Support regions and signs for the 16 nonstandard two-dimensional Haar basis functions.

Given an error tree T and an internal node t of T, $t \neq c_0$, $\texttt{leftleaves}(t)$ ($\texttt{rightleaves}(t)$) denotes the set of leaf (i.e., data) nodes in the subtree rooted at t's left (resp., right) child. Also, given any (internal or leaf) node u, $\texttt{path}(u)$ is the set of all (internal) nodes in T that are proper ancestors of u (i.e., the nodes on the path from u to the root of T, including the root but not u) with non-zero coefficients. Finally, for any two leaf nodes d_l and d_h, $d(l:h)$ denotes the range sum $\sum_{i=l}^{h} d_i$. Using the error tree representation T, the following important reconstruction properties of the Haar wavelet transform can be outlined.

• (P1) The reconstruction of any data value d_i depends only on the values of the nodes in $\texttt{path}(d_i)$. More specifically, $d_i = \sum_{c_j \in \texttt{path}(d_i)} \delta_{ij} \cdot c_j$, where $\delta_{ij} = +1$ if $d_i \in \texttt{leftleaves}(c_j)$ or $j = 0$, and $\delta_{ij} = -1$ otherwise; for example, $d_4 = c_0 - c_1 + c_6$ = $\frac{11}{4} - (-\frac{5}{4}) + (-1) = 3$.

• (P2) An internal node c_j contributes to the range sum $d(l:h)$ *only if* $c_j \in \texttt{path}(d_l) \cup \texttt{path}(d_h)$. More specifically, $d(l:h) = \sum_{c_j \in \texttt{path}(d_l) \cup \texttt{path}(d_h)} x_j$, where

$$x_j = \begin{cases} (h - l + 1) \cdot c_j, \text{ if } j = 0 \\ (|\texttt{leftleaves}(c_j, l:h)| - |\texttt{rightleaves}(c_j, l:h)|) \cdot c_j, \text{ otherwise.} \end{cases}$$

where $\texttt{leftleaves}(c_j, l:h) = \texttt{leftleaves}(c_j) \cap \{d_l, d_{l+1}, \ldots, d_h\}$ (i.e., the intersection of $\texttt{leftleaves}(c_j)$ with the summation range) and $\texttt{rightleaves}(c_j, l:h)$ is defined similarly. (Clearly, coefficients whose subtree is completely contained within the summation range have a net contribution of zero, and can be safely ignored.) For example, $d(2:6) = 5c_0 + (2-3)c_1 - 2c_2 = 5 \times \frac{11}{4} - (-\frac{5}{4}) - 1 = 14$.

Thus, reconstructing a single data value involves summing at most $\log N + 1$ coefficients and reconstructing a range sum

involves summing at most $2 \log N + 1$ coefficients, regardless of the width of the range. The *support region* for a coefficient c_i is defined as the set of (contiguous) data values that c_i is used to reconstruct.

The Haar wavelet transform can be naturally extended to *multi-dimensional* data arrays using two distinct methods, namely the *standard* and *nonstandard* Haar transform [12]. As in the one-dimensional case, the Haar transform of a d-dimensional data array A results in a d-dimensional wavelet-coefficient array W_A with the same dimension ranges and number of entries. Consider a d-dimensional wavelet coefficient W in the (standard or nonstandard) wavelet-coefficient array W_A. W contributes to the reconstruction of a d-dimensional rectangular region of cells in the original data array A (i.e., W's support region). Further, the sign of W's contribution ($+W$ or $-W$) can vary along the quadrants of W's support region in A.

As an example, Figure 1(b) depicts the support regions and signs of the sixteen nonstandard, two-dimensional Haar coefficients in the corresponding locations of a 4×4 wavelet-coefficient array W_A. The blank areas for each coefficient correspond to regions of A whose reconstruction is independent of the coefficient, i.e., the coefficient's contribution is 0. Thus, $W_A[0, 0]$ is the overall average that contributes positively (i.e.,"$+W_A[0, 0]$") to the reconstruction of all values in A, whereas $W_A[3, 3]$ is a detail coefficient that contributes (with the signs shown) only to values in A's upper right quadrant. Each data cell in A can be accurately reconstructed by adding up the contributions (with the appropriate signs) of those coefficients whose support regions include the cell. Error-tree structures for d-dimensional Haar coefficients are essentially *d-dimensional quadtrees*, where each internal node t corresponds to a *set* of (at most) $2^d - 1$ Haar coefficients, and has 2^d children corresponding to the quadrants of the (common) support region of all coefficients in t; furthermore, properties (P1) and (P2) can also be naturally extended to the multi-dimensional case [2, 7, 8].

Data Reduction and Approximate Query Processing. Consider a relational table R with d data attributes $X_1, X_2, \ldots X_d$. The information in R can be represented as a d-dimensional array A_R, whose j^{th} dimension is indexed by the values of attribute X_j and whose cells contain the count of tuples in R having the corresponding combination of attribute values. A_R is essentially the *joint frequency distribution* of all the data attributes of R. Given a limited amount of storage for building a *wavelet synopsis* of an input relation R, a thresholding procedure retains a certain number $B << N$ of the coefficients in the wavelet transform of A_R as a highly-compressed approximate representation of the original data (the remaining coefficients are implicitly set to 0). (The full details as well as efficient transform algorithms can be found in [2, 13].) The goal of *coefficient thresholding* is to determine the "best" subset of B coefficients to retain, so that some overall error measure in the approximation is minimized — the next subsection discusses different thresholding strategies proposed in the database literature.

The construction of wavelet synopses typically takes place during the statistics collection process, whose goal is to create concise statistical approximations for the value distributions of either individual attributes or combinations of attributes in the relations of a Database Management System (DBMS). Once created, a wavelet synopsis is typically stored (as a collection of B wavelet coefficients) as part of the *DBMS-catalog information*, and can be exploited for several different purposes. The primary (and, more conventional) use of such summaries is as a tool for enabling effective (compile-time) estimates of the result sizes of relational operators for the purpose of *cost-based query optimization*. (Accurate estimates of such result sizes play a critical role in choosing an effective physical execution plan for an input SQL query.) For instance, estimating the number of data tuples that satisfy a range-predicate selection like $l \leq X \leq h$ is equivalent to estimating the range summation $f(l : h) = \sum_{i=l}^{h} f_i$, where f is the frequency distribution array for attribute X. As mentioned earlier, given a B-coefficient synopsis of the f array, computing $f(l : h)$ only involves retained coefficients in $\texttt{path}(f_l) \cup \texttt{path}(f_h)$ and, thus, can be estimated by summing only $\min\{B, 2 \log N + 1\}$ synopsis coefficients [13]. A B-coefficient wavelet synopsis can also be easily expanded (in $O(B)$ time) into an $O(B)$-bucket *histogram* (i.e., piecewise-constant) approximation of the underlying data distribution with several possible uses (e.g., as a data visualization/approximation tool).

More generally, wavelet synopses can enable very fast and accurate approximate query answers [6] during interactive data-exploration sessions. As demonstrated by Chakrabarti et al. [2], an approximate query processing algebra (which includes all conventional aggregate and non-aggregate SQL operators, such as `select`, `project`, `join`, `sum`, and `average`) can operate *directly over the wavelet synopses of relations*, while guaranteeing the correct relational operator semantics. Query processing algorithms for these operators work *entirely* in the wavelet-coefficient domain. This allows for extremely fast response times, since the approximate query execution engine can do the bulk of its processing over compact wavelet synopses, essentially postponing the (expensive) expansion step into relational tuples until the end-result of the query.

Conventional and Advanced Wavelet Thresholding Schemes. Recall that coefficient thresholding achieves data reduction by retaining $B << N$ of the coefficients in the wavelet transform of A_R as a highly-compressed, lossy representation of the original relational data. The goal, of course, is to minimize the amount of "loss" quantified through some overall approximation error metric. Conventional wavelet thresholding (the method of choice for most studies on wavelet-based data reduction) greedily retains the *B largest Haar-wavelet coefficients in absolute value* after a simple normalization step (that divides each coefficient value at resolution level l by $\sqrt{2^l}$). It is a well-known fact that this thresholding method is in fact *provably optimal* with respect to minimizing the overall root-mean-squared error (i.e., *L_2-norm error*) in the data compression [12]. More

formally, letting $\hat{d}_i$ denote the (approximate) reconstructed data value for cell i, retaining the B largest normalized coefficients implies that the resulting synopsis minimizes $L_2(\hat{d}) = \sqrt{\sum_i (\hat{d}_i - d_i)^2}$ (for the given amount of space B).

Conventional wavelet synopses optimized for overall L_2 error may not always be the best choice for approximate query processing systems. The quality of the approximate answers such synopses provide can vary widely, and users have no way of knowing the accuracy of any particular answer. Even for the simplest case of approximating a value in the original data set, the absolute and relative errors can show wide variation. Consider the example depicted in Table 1. The first line shows the 16 original data values (the exact answer), whereas the second line shows the 16 approximate answers returned when using conventional wavelet synopses and storing 8 coefficients. Although the first half of the values is basically a mirror image of the second half, all the approximate answers for the first half are 65, whereas all the approximate answers for the second half are exact! Similar data values have widely different approximations, e.g., 30 and 31 have approximations 30 and 65, respectively. The approximate answers make the first half appear as a uniform distribution, with widely different values, e.g., 3 and 127, having the same approximate answer 65. Moreover, the results do not improve when one considers the presumably easier problem of approximating the sum over a range of values: for *all possible* ranges within the first half involving $x = 2$ to 7 of the values, the approximate answer will be $65 \cdot x$, while the actual answers vary widely. For example, for both the range d_0 to d_2 and the range d_3 to d_5, the approximate answer is 195, while the actual answer is 285 and 93, respectively. On the other hand, *exact* answers are provided for all possible ranges within the second half.

Original data values	127	71	87	31	59	3	43	99	100	42	0	58	30	88	72	130
Wavelet answers	65	65	65	65	65	65	65	65	100	42	0	58	30	88	72	130

Table 1: Errors with Conventional Wavelet Synopses.

The simple example above illustrates that conventional wavelet synopses suffer from several important problems, including the introduction of severe bias in the data reconstruction and wide variance in the quality of the data approximation, as well as the lack of non-trivial guarantees for individual approximate answers. To address these shortcomings, recent work has proposed novel thresholding schemes for building wavelet synopses that try to minimize different approximation-error metrics, such as the *maximum relative error* (with an appropriate *sanity bound* $\mathtt{s}$) in the approximation of individual data values based on the synopsis; that is, minimize $\max_i \left\{ \frac{|\hat{d}_i - d_i|}{\max\{|d_i|, \mathtt{s}\}} \right\}$. Such relative-error metrics are arguably the most important quality measures for approximate query answers. (The role of the sanity bound is to ensure that relative-error numbers are not unduly dominated by small data values.)

More specifically, Garofalakis and Gibbons [7] introduce *probabilistic* thresholding schemes based on ideas from randomized rounding, that probabilistically round coefficients either up to a larger rounding value (to be retained in the synopsis) or down to zero. Intuitively, their probabilistic schemes assign each non-zero coefficient *fractional storage* $y \in (0, 1]$ equal to its retention probability, and then flip independent, appropriately-biased coins to construct the synopsis. Their thresholding algorithms are based on *Dynamic-Programming (DP)* formulations that explicitly minimize appropriate probabilistic metrics (such as the maximum normalized standard error or the maximum normalized bias) in the randomized synopsis construction; these formulations are then combined with a *quantization* of the potential fractional-storage allotments to give combinatorial techniques [7].

In more recent work, Garofalakis and Kumar [8] show that the pitfalls of randomization can be avoided by introducing efficient schemes for *deterministic* wavelet thresholding with the objective of optimizing a *general class of error metrics* (e.g., maximum or mean relative error). Their optimal and approximate thresholding algorithms are based on novel DP techniques that take advantage of the Haar transform error-tree structure. In a nutshell, their DP algorithms tabulate the optimal solution for the subtree rooted at each error-tree node c_j *given the error contribution that "enters" that subtree through the choices made at all ancestor nodes of* c_j *in the tree* (i.e., the choice of coefficients on $\mathtt{path}(c_j)$). The key observation here is that, since the depth of the error tree is $O(\log N)$, all such possible selections can be tabulated while still keeping the running-time of the thresholding algorithm in the low-polynomial range. This turns out to be a fairly powerful idea for wavelet synopsis construction that can handle a broad, natural class of *distributive error metrics* (which includes several useful error measures for approximate query answers, such as maximum or mean weighted relative error and weighted L_p-norm error) [8].

The above wavelet thresholding algorithms for non-L_2 error metrics consider only the *restricted* version of the problem, where the algorithm is forced to select values for the synopsis from the standard Haar coefficient values. As observed by Guha and Harb [10], such a restriction makes little sense when optimizing for non-L_2 error, and can, in fact, lead to sub-optimal synopses. Their work considers *unrestricted* Haar wavelets, where the values retained in the synopsis are specifically chosen to optimize a general (weighted) L_p error metric. Their proposed thresholding schemes rely on a DP over the error tree (similar to that in [8]) that *also iterates over the range of possible coefficient values for each node*. To keep time and space complexities

manageable, techniques for bounding these coefficient-value ranges are also discussed [10].

Extended and Streaming Wavelet Synopses. Complex tabular data sets *with multiple measures* (multiple numeric entries for each table cell) introduce interesting challenges for wavelet-based data reduction. Such massive, multi-measure tables arise naturally in several application domains, including OLAP (On-Line Analytical Processing) environments and time-series analysis/correlation systems. As an example, a corporate sales database may tabulate, for each available product, (1) the number of items sold, (2) revenue and profit numbers for the product, and (3) costs associated with the product, such as shipping and storage costs. Similarly, real-life applications that monitor continuous time-series typically have to deal with several readings (measures) that evolve over time; for example, a network-traffic monitoring system takes readings on each time-tick from a number of distinct elements, such as routers and switches, in the underlying network and typically several measures of interest need to be monitored (e.g., input/output traffic numbers for each router or switch interface) even for a fixed network element. Deligiannakis et al. [4] show that obvious approaches for building wavelet synopses for such multi-measure data can lead to poor synopsis-storage utilization and suboptimal solutions even in very simple cases. Instead, their proposed solution is based on (1) *extended wavelet coefficients*, the first adaptive, efficient storage scheme for multi-measure wavelet coefficients; and, (2) novel algorithms for selecting the optimal subset of extended coefficients to retain for minimizing the weighted sum of L_2 errors across all measures under a given storage constraint.

Traditional database systems and approximation techniques are typically based on the ability to make multiple passes over *persistent data sets*, that are stored reliably in stable storage. For several emerging application domains, however, data arrives at high rates and needs to be processed on a continuous (24×7) basis, without the benefit of several passes over a static, persistent data image. Such *continuous data streams* arise naturally, for example, in the network installations of large Telecom and Internet service providers where detailed usage information (Call-Detail-Records (CDRs), SNMP/RMON packet-flow data, etc.) from different parts of the underlying network needs to be continuously collected and monitored for interesting trends and phenomena (e.g., fraud or Denial-of-Service attacks). Efficiently tracking an accurate wavelet synopsis over such massive streaming data, using only small space and time (per streaming update), poses a host of new challenges. Recently-proposed solutions [3, 9] rely on maintaining small-space, *pseudo-random AMS sketches* (essentially, random linear projections) over the input data stream [1]. These sketches can then be queried to efficiently recover the topmost wavelet coefficients of the underlying data distribution within provable error guarantees [3].

KEY APPLICATIONS

Wavelet synopses are a general data-reduction tool with several important applications, including statistics for query optimization, lossy data compression, OLAP cube summarization, and interactive data exploration, mining, and query processing.

DATA SETS

Several publicly-available real-life data collections have been used in the experimental study of wavelet synopses (and other data-reduction methods); examples include the US Census Bureau data sets (`http://www.census.gov/`), the UCI KDD Archive (`http://kdd.ics.uci.edu/`), and the UW Earth Climate and Weather Data Archive (`http://www-k12.atmos.washington.edu/k12/grayskies/`).

FUTURE DIRECTIONS

The area of wavelet-based data reduction is still rife with interesting algorithmic questions, including, for instance (1) designing efficient methods for building wavelet synopses that optimize different error metrics under general streaming models (e.g., allowing both item insertions and deletions), and (2) developing a sound foundation and appropriate summarization tools for approximate *set-valued* (i.e., non-aggregate) queries. Dealing with the *curse of dimensionality* that invariably haunts space-partitioning techniques (such as wavelets and histograms) is another big open issue; some initial ideas based on combining these techniques with statistical-correlation models appear in [5]. And, of course, from a systems perspective, the problem of incorporating wavelets and other approximate query processing tools in an industrial-strength database engine (that can, e.g., select and optimize the appropriate tools for each scenario) remains wide open.

CROSS REFERENCES

WAVELETS ON STREAMS, APPROXIMATE QUERY PROCESSING, DATA COMPRESSION, DATA REDUCTION, SYNOPSIS STRUCTURES, DATA SKETCH/SYNOPSIS

References

[1] Noga Alon, Yossi Matias, and Mario Szegedy. "The Space Complexity of Approximating the Frequency Moments". In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing*, pages 20–29, Philadelphia, Pennsylvania, May 1996.

[2] Kaushik Chakrabarti, Minos N. Garofalakis, Rajeev Rastogi, and Kyuseok Shim. "Approximate query processing using wavelets". *The VLDB Journal*, 10(2-3):199–223, September 2001. ("Best of VLDB'2000" Special Issue).

[3] Graham Cormode, Minos Garofalakis, and Dimitris Sacharidis. "Fast Approximate Wavelet Tracking on Streams". In *Proceedings of the 10th International Conference on Extending Database Technology (EDBT'2006)*, Munich, Germany, March 2006.

[4] Antonios Deligiannakis, Minos Garofalakis, and Nick Roussopoulos. "Extended Wavelets for Multiple Measures". *ACM Transactions on Database Systems*, 32(2), June 2007.

[5] Amol Deshpande, Minos Garofalakis, and Rajeev Rastogi. "Independence is Good: Dependency-Based Histogram Synopses for High-Dimensional Data". In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data*, Santa Barbara, California, May 2001.

[6] Minos Garofalakis and Phillip B. Gibbons. "Approximate Query Processing: Taming the Terabytes". Tutorial in *27th Intl. Conf. on Very Large Data Bases*, Roma, Italy, September 2001.

[7] Minos Garofalakis and Phillip B. Gibbons. "Probabilistic Wavelet Synopses". *ACM Transactions on Database Systems*, 29(1), March 2004. (SIGMOD/PODS'2002 Special Issue).

[8] Minos Garofalakis and Amit Kumar. "Wavelet Synopses for General Error Metrics". *ACM Transactions on Database Systems*, 30(4), December 2005. (SIGMOD/PODS'2004 Special Issue).

[9] Anna C. Gilbert, Yannis Kotidis, S. Muthukrishnan, and Martin J. Strauss. "One-pass wavelet decomposition of data streams". *IEEE Transactions on Knowledge and Data Engineering*, 15(3):541–554, May 2003.

[10] Sudipto Guha and Boulos Harb. "Wavelet synopsis for data streams: minimizing non-euclidean error". In *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Chicago, Illinois, August 2005.

[11] Björn Jawerth and Wim Sweldens. "An Overview of Wavelet Based Multiresolution Analyses". *SIAM Review*, 36(3):377–412, 1994.

[12] Eric J. Stollnitz, Tony D. DeRose, and David H. Salesin. *"Wavelets for Computer Graphics – Theory and Applications"*. Morgan Kaufmann Publishers, San Francisco, CA, 1996.

[13] Jeffrey Scott Vitter and Min Wang. "Approximate Computation of Multidimensional Aggregates of Sparse Data Using Wavelets". In *Proceedings of the 1999 ACM SIGMOD International Conference on Management of Data*, Philadelphia, Pennsylvania, May 1999.