On Wavelet Decomposition of Uncertain Time Series Data Sets

Yuchen Zhao
University of Illinois at Chicago
Chicago, Illinois
yzhao@cs.uic.edu

Charu C. Aggarwal
IBM T. J. Watson Research Ctr
Hawthorne, NY
charu@us.ibm.com

Philip S. Yu
University of Illinois at Chicago
Chicago, Illinois
psyu@cs.uic.edu

ABSTRACT

In this paper, we will explore the construction of wavelet decompositions of uncertain data. Uncertain representations of data sets require significantly more space, and it is therefore even more important to construct compressed representations for such cases. We will use a hierarchical optimization technique in order to construct the most effective partitioning for our wavelet representation. We explore two different schemes which optimize the uncertainty in the resulting representation. We will show that the incorporation of uncertainty into the design of the wavelet representations significantly improves the compression rate of the representation. We present experimental results illustrating the effectiveness of our approach.

Categories and Subject Descriptors

H.2.8 [**Information Systems**]: Database Applications

General Terms

Algorithms

1. INTRODUCTION

With advances in hardware and software technology, many new applications create data which is inherently uncertain. Typically, uncertain data may be created as a result of many different kinds of data collection or creation methods:
(1) When the data is collected because of imprecise instruments, the error can be measured from the characteristics of the instrumentation. In some cases, repeated measurements can be used in order to estimate the probability distribution of the underlying data. For example, in sensor networks, the data can often be collected only approximately.
(2) In methods such as privacy-preserving data mining, uncertainty is intentionally added to the data in order to protect sensitive information. In some cases [5], the data may be explicitly modeled as uncertain data.
(3) In many cases, the data is generated by artificial techniques such as forecasting. Such data can be statistically modeled as uncertain data.

The problem of uncertain data management has been studied in the traditional database literature [14], though the issue has seen a revival in recent years [6, 7, 4, 5, 10, 11, 13]. The driving force behind this revival has been the evolution of new techniques and technologies which result in uncertain data collection. Data mining and management techniques need to be carefully re-designed in order to work effectively with the case of uncertain data. One such data management application is that of synopsis construction. A survey of synopsis construction algorithms may be found in [3]. An important problem in synopsis construction is that of wavelet decomposition [18]. Wavelet decomposition techniques are widely used in a number of indexing and retrieval applications. Since uncertain data has additional volume (because of uncertainty information), the process of synopsis construction is especially important in order to compress the corresponding representation. In this paper, we will design an effective technique for performing such a compression and discuss its applicability to query estimation.

The particular form of the wavelet which we will examine is the *Haar Wavelet.* Haar wavelets are particularly useful because of their additive and multi-resolution representation of the underlying data. However, existing approaches on one-dimensional and two-dimensional Haar wavelets [18] are especially not effective for the case of uncertain time series data. We will design an uncertain representation of the Haar decomposition and show its utility for representing the data in compressed format. We will show that a carefully designed decomposition can provide accurate representations of the underlying time series, and is significantly superior to more direct solutions for performing the decomposition.

This paper is organized as follows. We will discuss related work in the remainder of this section. The next section discusses the details of the wavelet-based decomposition technique. We will examine the properties of this decomposition, and its utility for accurate time series representation. In section 3, we will present the experimental results illustrating the effectiveness of the technique. Section 4 contains the conclusions and summary.

1.1 Related Work

The problem of uncertain data management and mining has been explored extensively in recent years [1, 2]. Uncertain data is created by numerous applications in data forecasting, privacy, bio-medical data and mobile applications [2, 5]. Numerous algorithms have been developed recently

Order k	**Averages** Φ values	**DWT Coeff.** ψ values
$k = 4$	(8, 6, 2, 3, 4, 6, 6, 5)	-
$k = 3$	(7, 2.5, 5, 5.5)	(1, -0.5,-1, 0.5)
$k = 2$	(4.75, 5.25)	(2.25, -0.25)
$k = 1$	(5)	(-0.25)

Table 1: Example of Wavelet Computation

for data management and mining problems [4, 2, 13]. In particular, considerable effort has been devoted to the problem query processing for uncertain data sets [2, 10]. For many aggregate kinds of queries, it may be desirable to construct a compact summary of the data in order to provide summary responses to queries. Recently, summary techniques have been developed for aggregate queries in probabilistic data. A survey of these techniques may be found in [19].

A well known technique for summarization of deterministic time series data sets is that of wavelet decomposition. A survey of recent methods for one-dimensional and multidimensional wavelet decomposition may be found in [3, 18]. The problem of wavelet decomposition was studied in the context of database query processing in [8, 15, 16, 20]. Recently, a number of sketching techniques have been developed for probabilistic data streams [12]. A recent technique [9] designs histograms and wavelets for probabilistic data under certain classes of probabilistic models. However, none of these techniques are applicable to wavelet decomposition of arbitrary data-driven probability distributions. In order to retain generality, this paper will use a general probability density function in the form of data-driven distribution. In practice, precise probability models for data are rarely available, and these can only be approximated in a data driven way. Such probabilistic representations are often space-intensive; a characteristic which further increases the need to have an efficient data reduction process.

2. UNCERTAIN WAVELETS

In this section, we present the algorithm for constructing the wavelet decomposition of the underlying data representation. The primary representation which we will use for the underlying representation is the *Haar Wavelet*. Before discussing the decomposition method in further detail, we will review the details of Haar wavelet construction. This technique is particularly simple to implement, and is widely used in the literature for hierarchical decomposition and summarization. The basic idea in the wavelet technique is to create a decomposition of the data characteristics into a set of wavelet functions and basis functions. The property of the wavelet method is that the higher order coefficients of the decomposition illustrate the broad trends in the data, whereas the more localized trends are captured by the lower order coefficients.

We assume for ease in description that the length q of the series is a power of 2. This is without loss of generality, because it is always possible to decompose a series into segments, each of which has a length that is a power of two. The Haar Wavelet decomposition defines 2^{k-1} coefficients of order k. Each of these 2^{k-1} coefficients corresponds to a contiguous portion of the time series of length $q/2^{k-1}$. The ith of these 2^{k-1} coefficients corresponds to the segment in

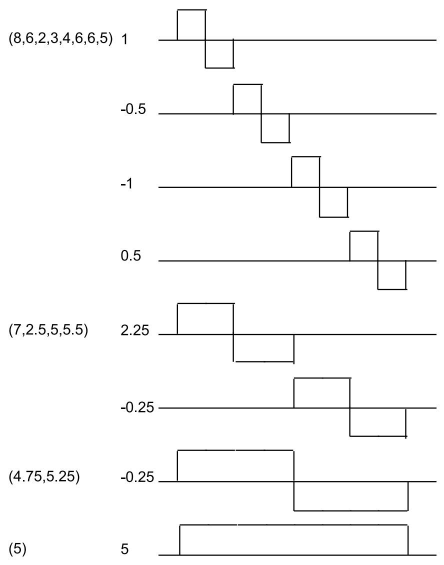

Figure 1: Illustration of Wavelet Decomposition

the series starting from position $(i-1) \cdot q/2^{k-1} + 1$ to position $i * q/2^{k-1}$. Let us denote this coefficient by ψ_k^i and the corresponding time series segment by S_k^i. At the same time, let us define the average value of the first half of the S_k^i by a_k^i and the second half by b_k^i. Then, the value of ψ_k^i is given by $(a_k^i - b_k^i)/2$. More formally, if Φ_k^i denote the average value of the S_k^i, then the value of ψ_k^i can be defined recursively as follows:

$$\psi_k^i = (\Phi_{k+1}^{2 \cdot i-1} - \Phi_{k+1}^{2 \cdot i})/2 \qquad (1)$$

The set of Haar coefficients is defined by the Ψ_k^i coefficients of order 1 to $\log_2(q)$. In addition, the global average Φ_1^1 is required for the purpose of perfect reconstruction. We note that the coefficients of different order provide an understanding of the major trends in the data at a particular level of granularity. For example, the coefficient ψ_k^i is half the quantity by which the first half of the segment S_k^i is larger than the second half of the same segment. Since larger values of k correspond to geometrically reducing segment sizes, one can obtain an understanding of the basic trends at different levels of granularity. We note that this definition of the Haar wavelet makes it very easy to compute by a sequence of averaging and differencing operations. In Table 1, we have illustrated how the wavelet coefficients are computed for the case of the sequence $(8, 6, 2, 3, 4, 6, 6, 5)$. This decomposition is illustrated in graphical form in Figure 1. We also note that each value can be represented as a sum of $\log_2(8) = 3$ linear decomposition components. In general, the entire decomposition may be represented as a tree of depth 3, which represents the hierarchical decomposition of the entire series. This is also referred to as the *error tree*. In Figure 2, we have illustrated the error tree for the wavelet decomposition illustrated in Table 1. The nodes in the tree contain the values of the wavelet coefficients, except for a special *super-root* node which contains the series average. This super-root node is not necessary if we are only considering the relative values in the series, or

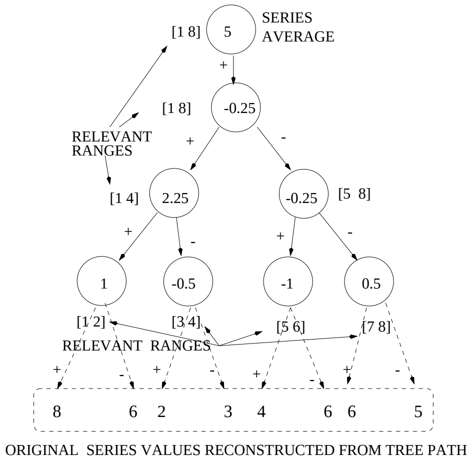

Figure 2: Error Tree from Wavelet Decomposition

the series values have been normalized so that the average is already zero. We further note that the number of wavelet coefficients in this series is 8, which is also the length of the original series. The original series has been replicated just below the error-tree in Figure 2, and it can be reconstructed by adding or subtracting the values in the nodes along the path leading to that value. We note that each coefficient in a node should be added, if we use the left branch below it to reach to the series values. Otherwise, it should be subtracted. This natural decomposition means that an entire contiguous range along the series can be reconstructed by using only the portion of the error-tree which is relevant to it. Furthermore, we only need to retain those coefficients whose values are significantly large, and therefore affect the values of the underlying series. In general, we would like to minimize the reconstruction error by retaining only a fixed number of coefficients, as defined by the space constraints.

2.1 Defining the Uncertain Case

In the case of uncertain data, the wavelet decomposition algorithm is defined with the use of uncertain variables along the time series. It is assumed that the time-series has a length of N, and the ith element along the time series is defined by the probability density function $f_i(\cdot)$. We make no assumption about the nature of the probability density function $f_i(\cdot)$. The function may not even be available in the closed form, but may only be representable in the form of bucket histograms. This may often be the case, when the data point is obtained by observing multiple instantiations of a particular event. The use of the bucketized histogram representation of the probability density function is the most general representation, since it does not depend upon any canonical form of the underlying representation. Therefore, we will use it for our analysis. This assumption is also useful for modeling the general case where each value in the time series is defined by sampling over a large set of data values. For example, consider the case in which we are tracking the scores of a large number of games in the NBA basketball league. Each time-stamp may correspond to the games which occur in a particular period such as a day or a week. A particular frequency statistic (eg. margin by which game was won or total points scored) of the games in a given period may define the histogram for the corresponding uncertain representation. In other applications such as surveys, the statistics collected during a particular period may represent the histogram for the uncertain representation. Similarly, for many privacy-preserving data mining applications, the data may be available only on an aggregated basis.

Let us assume that each of the functions $f_i(\cdot)$ is represented in the form of frequency values along w buckets. For ease of exposition, we will assume that w is a power of 2. This assumption is useful for describing the wavelet decomposition cleanly, though the general decomposition can be constructed by expressing the number of buckets as the sums of powers of two. We assume that the length of the time series is N. As in the previous case, we assume that N is a power of 2 for ease in exposition. It is assumed that the jth bucket corresponds to the range $[l(j), u(j)]$. We assume that the relative frequencies along the buckets for $f_i(\cdot)$ are denoted by $p(i,1) \ldots p(i,w)$. The value of $p(i,j)$ denotes the probability that the ith element of the time-series lies in the range $[l(j), u(j)]$. Since, the values of $p(i,j)$ over the different buckets must sum to 1, we have:

$$\sum_{j=1}^{w} p(i,j) = 1 \qquad (2)$$

We note that a straightforward solution of the wavelet decomposition problem is to independently decompose each of the w components. However, this can be a poor solution when the correlation between adjacent buckets in the wavelet decomposition is very high. This is often the case in real applications. Furthermore, in such cases, it is more desirable to incorporate the probabilistic component directly in the decomposition. In the next section, we will describe the method for performing the decomposition.

2.2 Wavelet Decomposition Algorithm

In the case of the decomposition of a deterministic time series, we create the wavelet decomposition by cutting each time interval at the middle of the corresponding range. However, in the case of uncertain data, this is not possible since, we also have a probabilistic component to the data behavior. This probabilistic component corresponds to the w different buckets along which the frequencies are expressed. Therefore, it is important to take the use the probabilistic behavior of the data during the decomposition process.

In the case of deterministic wavelet decomposition, each coefficient describes the behavior of the data along a particular range of time. Correspondingly, in the case of the uncertain representation, each coefficient corresponds to the behavior along a particular time series segment *and* probabilistic range (contiguous group of histogram buckets). We note that a division of a particular order may occur along either the time component or the probabilistic component. The information as to whether the cut occurs along the time component or the probabilistic component is additional overhead which needs to be stored independently. This overhead can be substantial if it needs to be stored at each node of the wavelet tree. One possibility for reducing the overhead is *to assume a single global ordering of the cuts along the time dimension or the probabilistic dimension.* Thus, all cuts along a given level of the wavelet tree are either a cut along the time component or a cut along the probabilistic component. By adding this constraint, the only additional overhead we need to store is the global ordering of cuts along either the time or the probabilistic component. This overhead has the same order as the height of the wavelet tree or $h = \log_2(N)$. Therefore, we can define the cut-order in the form of a bit-string of length h, where a 0 at a particular position in the string corresponds to a probability cut, whereas a 1 in a particular position corresponds to time cut. We define the *global cut-order* for a given wavelet decomposition as follows:

Definition 1. *The cut-order for a given wavelet decomposition is a 0-1 bit-string $q_1 q_2 \ldots q_h$ of length h, where h is the height of the wavelet tree. The bit q_r is 1, if all cuts of the rth order are along the time-series component, and the bit q_r is 0, if all cuts of the rth order are along the probabilistic component.*

We note that since some of the cuts are along the time component, and other cuts are along the probability component, we need to define the time-order and the probability-order of the wavelet coefficients. We note that the use of a global-ordering introduces some inflexibility into the decomposition in order to save on overhead. If we had used a node-specific cut-strategy, then we would have had to store this information at each node. On the other hand, a node-specific cut-strategy would also result in fewer number of nodes because of better local optimization. Therefore, we will also examine the methodology for the local-decomposition slightly later, and compare the two methods in the experimental section. For the time being, we will only discuss the case for defining the global cut-order, since the broad framework can be easily extended to the local case. Next, we will establish some further definitions on defining the *time-order* for a global wavelet coefficient.

Definition 2. *The time-order for an rth order wavelet coefficient with cut-order $q_1 \ldots q_r$ is given by $\sum_{i=1}^{r} q_i$. This simply represents the number of cuts along the time dimension. The time order for a wavelet coefficient of order r is denoted by $t(r)$.*

Correspondingly, the probability-order for the cuts are defined in terms of the number of cuts along the probability-dimension.

Definition 3. *The probability-order for an rth order wavelet coefficient is given by $r - \sum_{i=1}^{r} q_i$. The probability-order for a wavelet coefficient of order r is denoted by $s(r)$.*

Since the number of cuts for a wavelet coefficient of order r sum to r, it easily follows that the sum of $t(r)$ and $s(r)$ must be r.

We will first define the concept of associating an uncertain wavelet coefficient with both the time-series and probability space. This will also help in explaining how the $(i+1)$th order wavelet coefficients are generated from the ith order coefficient.

Definition 4. *An uncertain wavelet coefficient of order r is associated with the time-series domain range $[a, b]$, and probability domain range*[1] *$[c, d]$. The length of the range $[a, b]$ is given by $N/2^{t(r)-q_r}$ and the length of the range $[c, d]$ is given by $w/2^{s(r)-1+q_r}$. The magnitude of the wavelet coefficient is defined as follows:*
(1) *If q_r is 0, then the value of the wavelet-coefficient is defined by subtracting the average values of the probabilities in $[a, b] \times [\lceil (c + d)/2 \rceil, d]$ from the average values of the probabilities in $[a, b] \times [c, d]$.*
(2) *If q_r is 1, then the value of the wavelet-coefficient is defined by subtracting the average values of the probabilities in $[\lceil (a + b)/2 \rceil, b] \times [c, d]$ from the average values of the probabilities in $[a, b] \times [c, d]$.*

In addition, we have a special coefficient at the root of the tree representation which corresponds to the average probability values across all $N \times w$ buckets. In the event that the tree is built to full height, and no coefficients are discarded, the total number of coefficients including this special coefficient is equal to $N * w$. We note that if the wavelet-tree is constructed to completion, then its height will be given by $\log_2(N) + \log_2(w)$. In practice, it may not be necessary to construct the tree to full height. The total number of passes required over the database is given by this height over the tree.

Each node at the leaf level of the tree corresponds to a pair of adjacent buckets in the $w \times N$ histogram represented by the wavelet, and the coefficient value corresponds to half the difference in the probability values of these two buckets. The exact value of any bucket may be reconstructed by summing the wavelet coefficients along the height of the entire tree, where a left branch corresponds to an addition and a right branch corresponds to a subtraction.

Observation 1. *The probability value along a given bucket may be reconstructed by summing the wavelet-coefficients along the height of the entire tree.*

In general, it is possible to obtain a fairly accurate reconstruction by storing only the largest wavelet coefficients. Ideally, for a given wavelet tree, we would like to pick the wavelet coefficients in such a way that the error of the representation is optimized. We define the error as the mean-square error of the uncertain data representation:

Definition 5. *Let the original probabilities of representation be denoted by $p(i, j)$, for the jth bucket of data point i. Let the probabilities obtained from the wavelet decomposition (after removing the smaller coefficients) be denoted by $p'(i, j)$. Then, the mean square error E is defined as follows:*

$$E = \sum_{i=1}^{N} \sum_{j=1}^{w} (p(i, j) - p'(i, j))^2 \tag{3}$$

We would like to pick the wavelet coefficients, so that the error of representation is minimized. First we define the *representation vector of each wavelet coefficient.*

Definition 6. *The representation vector of a wavelet coefficient of order r for region $[a, b] \times [c, d]$ is defined as follows:*
(a) *The vector has length $N \times w$, with one element for each position in the time series.*
(b) *All positions outside the positions corresponding to $[a, b] \times [c, d]$ are set to 0.*
(c) *if q_r is 0, then all positions in $[a, b] \times [\lceil (c + d)/2 \rceil, d]$ are set to -1, and all positions in $[a, b] \times [c, \lfloor (c + d)/2 \rfloor]$ are set to 1.*

[1]We note that c and d are expressed in terms of the index of the histogram buckets, which may range between 1 and w.

```
**Algorithm** *CutOrder*(Database: $\mathcal{D}$, Lookahead: $r$);
**begin**
  **while** entire tree is not constructed
  $m = 0$;
    **begin**
    Try different combinations for $q_{m*r+1} \dots q_{m*r+r}$
      in order to construct the next $m$ levels optimally;
    m=m+1;
    **end**
**end**
```

Figure 3: Determination of Cut Order

(d) *if q_r is 1, then all positions in $[a, \lfloor (a+b)/2 \rfloor] \times [c, d]$ are set to 1, and all positions in $[\lceil (a+b)/2 \rceil, b] \times [c, d]$ are set to -1.*

We can construct a representation vector $\overline{e_i}$ for each node of the wavelet tree. Thus, there are a total of $M = N * w$ representation vectors along with corresponding wavelet coefficients. Let the wavelet coefficient (as computed above) for the representation vector $\overline{e_i}$ be denoted by w_i. Then, we make the following observation:

OBSERVATION 2. *The probability histogram for the uncertain time series can be computed as the wavelet weighted sum of the representation vectors, which is denoted $\sum_{i=1}^{M} w_i \cdot \overline{e_i}$.*

We further note that the vectors $\overline{e_1} \dots \overline{e_M}$ are orthogonal to one another because they represent hierarchically organized nodes of the wavelet tree.

OBSERVATION 3. *The vectors $\overline{e_1}/|\overline{e_1}| \dots \overline{e_M}/|\overline{e_M}|$ form an orthonormal basis system of the probability space of size $w * N$.*

Therefore, the *normalized representation* of the wavelet decomposition of the time series T is denoted as follows:

$$T = \sum_{i=1}^{M} w_i \cdot |\overline{e_i}| \cdot \frac{\overline{e_i}}{|\overline{e_i}|} \tag{4}$$

The normalized representation simply divides each vector by its corresponding modulus. Thus, the normalized wavelet coefficients are $w_1 \cdot |\overline{e_1}| \dots w_M \cdot |\overline{e_M}|$ respectively. Clearly, the normalization factor $|\overline{e_i}|$ is greater for higher level nodes in the tree, since a greater number of entries are non-zero in these cases. Since the wavelet coefficients provide an orthonormal basis system, we can obtain an analogous result to the deterministic case of wavelet decomposition.

LEMMA 1. *The total square error on discarding the subset of wavelet coefficients $i_1 \dots i_k$ is given by $\sum_{j=1}^{k} w_{i_j}^2 \cdot |\overline{e_{i_j}}|^2$. Therefore, the mean square error is minimized by selecting the largest (normalized) wavelet coefficients.*

PROOF. This result directly follows from the orthonormality of the corresponding vectors. The representation error in vector form is given by $\sum_{j=1}^{k} w_{i_j} \cdot \overline{e_{i_j}}$. By taking the modulus of this vector, we get the desired result. $\square$

2.3 Determining the Cut Order

We have described the entire construction of the wavelet decomposition, except for how the cut order is determined. Clearly, the effectiveness of the wavelet decomposition is defined by the global cut-order $q_1 \dots q_r$. We would like to pick the cut-order in such a way that only a small number of the wavelet coefficients have large values. We note that since q_i is a global cut-order, all nodes at level of r of the tree have a cut which is defined by q_r. Therefore, the natural strategy is to construct the wavelet tree level by level, and to choose q_i at each level such that our objective of keeping large wavelet coefficients at higher levels of the tree is achieved.

In order to achieve this goal, we first pick $q_i = 0$ and compute the coefficients $f_1 \dots f_l$ at a given level of the tree. Then, we pick $q_i = 1$, and compute the coefficients $g_1 \dots g_l$ for the same level of the tree. If the value of $\sum_{i=1}^{l} f_i^2$ is greater, then we pick $q_i = 0$, otherwise we pick $q_i = 1$. This approach is used in top-down fashion in order to build the tree level by level. We note that this procedure determines the cut-order by using a *level-lookahead* of 1. This approach does not have significant asymptotic overhead over picking a random value of q_i, since only two possibilities need to be tested at a given time. We can construct the same algorithm by using a level-lookahead of 2, by constructing the 4 possible combinations of successive levels by picking q_i and q_{i+1} in different ways. The construction of two levels at one time creates a superior wavelet decomposition, but at higher cost. By increasing the lookahead to the entire height of the tree, it is possible to construct the optimal tree, but the cost of exploring so many combinations would be computationally prohibitive. In practice, the use of a small lookahead provides an effective decomposition. The procedure for cut-order determination is illustrated in Figure 3.

2.4 Local Optimization of Wavelet Coefficients

Instead of using a global cut-order $q_1 \dots q_r$, it is possible to locally optimize the cut-order at each node. In this case, we test different kinds of cuts at each node to determine whether to cut along the temporal dimension, or to cut along the probabilistic dimension. As in the previous case, we pick the choice for which the sum of the squares of the corresponding wavelet coefficients for a given node are optimized. We note that the possible choices wavelet coefficients for a given level of the tree can be computed simultaneously in a single scan over the database. Subsequently, the wavelet coefficients can be picked optimally for each node. The main difference from the global approach is that a different decision on choice of coefficients is made for each node depending upon the magnitude of the corresponding coefficients. We note that the wavelet basis continues to maintain the orthonormality property. Since Lemma 1 is dependent on the orthonormality property, it follows that it continues to hold even in the case of the locally optimized wavelet tree. Therefore, once the wavelet-tree is constructed, it suffices to pick the largest wavelet coefficients.

We also need to store additional information about the local cut-order. We store an additional bit corresponding to each node. This bit takes on the value of 0 or 1 depending upon the choice of the direction along which the cut is made. Since only an additional bit needs to be stored at each node, this results in modest storage overhead. As we will see in the experimental section, this overhead is more than compensated for by the better optimization of the cuts at different levels of the tree.

2.5 Application to Query Resolution

Since the wavelet representation provides a compressed representation of the underlying time-series, it can be used

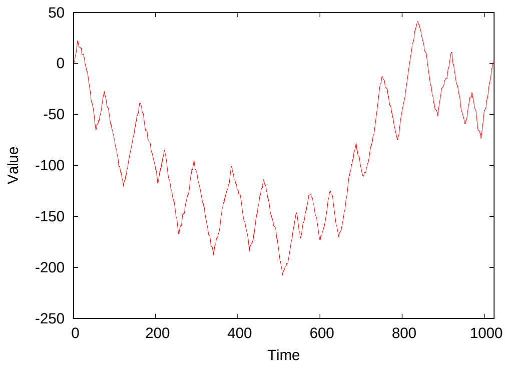

Figure 4: Illustration of Synthetic Time Series

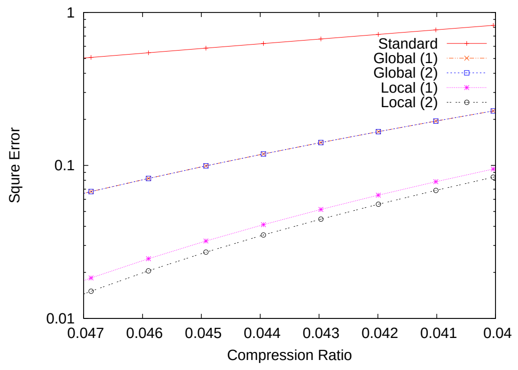

Figure 5: Accuracy on Synthetic Data Set

for effective query resolution. Some examples of queries in which the wavelet representation can be used are as follows: **(1)** Given a target series S, determine its expected distance to T. **(2)**Given a target series S, determine the expected number of points which lie within a threshold ϵ to T. **(3)** Given a target series S, determine the number of points for which the probability of lying within the threshold ϵ is greater than δ.

We note that S need not be of the length of the entire series T, but may comprise only a small contiguous portion of the series T. In each of these cases, it is possible to use the wavelet tree in order to efficiently respond to the corresponding queries. While most of the coefficients in the wavelet tree may already have been pruned, it is possible to improve the efficiency of query answering further by carefully selecting which branches are explored. In each case, the query can be efficiently answered by using the following approach on the wavelet tree:

• Only those portions of the wavelet tree which intersect with the query segment S are explored. Since it is assumed that S comprises only a small portion of the entire series, it follows that this helps in pruning off a large portion of the tree.

• For queries which have a threshold distance ϵ, only those buckets need to be explored which lie within this threshold distance of ϵ. Therefore, portions of the wavelet tree which correspond to irrelevant buckets can be pruned. If desired, the wavelet tree can be explored hierarchically in order to prune off large portions of the tree at higher levels.

By using this approach, the efficiency of query processing can be significantly improved. Thus, the wavelet representation provides an effective representation which can be leveraged for efficient query processing.

3. EXPERIMENTAL RESULTS

In the experimental section, we will examine the effectiveness and efficiency of the wavelet decomposition technique. In order to have fairness in comparison, we fixed the amount of space required by each technique. As a baseline, we use the *standard decomposition* of two-dimensional Haar wavelet transform as proposed in [18]. In the subsequent plots, we denote this technique as the *standard decomposition* approach. In order to illustrate the effectiveness of our approach, we can use either the global or the local approach for wavelet decomposition. In addition, the amount of lookahead can be varied in order to explore the effectiveness of various scenarios. Therefore, we will test the following four variations: **(1)** The global approach using a level-lookahead of 1. **(2)** The global approach using a level-lookahead of 2. **(3)** The local approach using a level-lookahead of 1. **(4)** The local approach using a level-lookahead of 2.

In addition, we compare the natural technique of using the standard decomposable approach to these four approaches. In all our tests, we used $w = 512$ in order to represent the buckets for the different histograms.

All experiments were done on an Intel Core 2 Duo processor, 2.1GHz, 3G memory), running Windows Vista. All approaches were implemented in C++ using Visual Studio 2005.

3.1 Evaluation Measures

All methods were compared with the assumption of a fixed budget B on the storage space. The use of a fixed amount B of storage space on any of the approaches creates a different reconstruction of the underlying data. It is desirable to compare the accuracy of the different reconstructions as compared to the original data. Let the reconstructed value of $p(i,j)$ be denoted by $p'(i,j)$. Then, we use the mean square error in order to compute the accuracy of the representation. As discussed in Equation 3, the mean square error E is defined as follows:

$$E = \sum_{i=1}^{N} \sum_{j=1}^{w} (p(i,j) - p'(i,j))^2$$

Clearly, the error increases when the value of B is reduced for any method. This baseline error is an intuitive way of measuring the effectiveness of different kinds of applications on the underlying data, since it reflects the core representational error. In the experimental section, we will express B as the compression ratio between the original and the compressed data.

3.2 Uncertain Data Generation

As baseline, we used both real and synthetic data sets. First, we will describe the base data sets, and then we will discuss how noise was added to each of these data sets. The synthetic data set was a time-series is generated using the following canonical form:

$$t_{i+1} = t_i + pole \times rand \qquad (5)$$

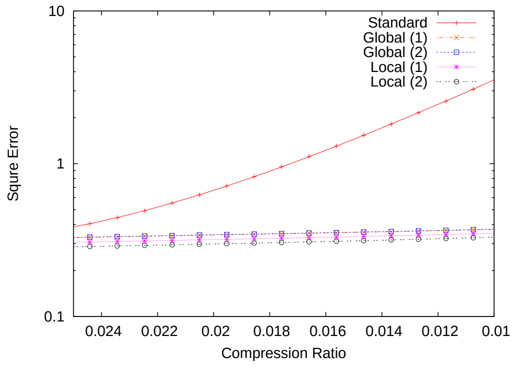

Figure 6: Accuracy on Chlorine Data

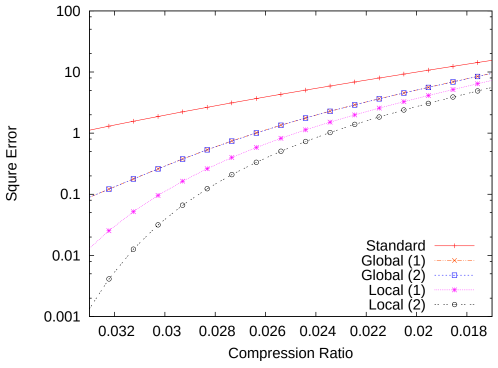

Figure 7: Accuracy on Sensor Data (Humidity)

The quantity t_i represents the value of tick i in the time series. The parameter *pole* could be either 1 or -1, which changes periodically. The period of *pole* is randomly selected from 10 to 50. The value of *rand* is the gap between two adjacent ticks (t_i and t_{i+1}), and it is random number which is randomly selected from 0 to 5. The time series of synthetic data set is shown in Figure 4. We note that this time series does not have any noise added to it. We will discuss the methodology for adding the noise slightly later.

Next, we describe the real data sets. The data set was constructed from sensors deployed in Intel Berkeley Research Labs [17, 22]. These data sets correspond to light, temperature, and humidity readings from the sensors. We will pick some of the streams from these data sets to test with our approach. The second data set was the **Chlorine** Data Set, and was generated by EPANET 2.0 [21]. This data stream contained 166 different sensor streams corresponding to chlorine readings at different junctions. As in the case of the synthetic data sets, artificial noise was added to the time series data stream. For all data sets, the methodology used to inject noise was the same.

We inject noise into the time series which is of a similar order as the variance of the values. The noise is modeled as a mixture modeling distribution, though we store it in the form of a histogram with w buckets, where w is fairly fine grained, e.g., w=512. In order to model the noise in the time series, we use a cluster scenario to generate the offset from the true time series value. For each time position, we generate an instantiation of the offset from the true value by picking one of the clusters. This may provide a positive or negative offset. We add this offset to the time series value, and this provides one possible sample of the uncertain value at that point. We generate k cluster centers to represent a k-modal distribution. Each cluster center was drawn from a normal distribution with mean equal to zero and variance v. The value of v is picked of the same order as the time series variance. Each cluster itself has a variance of v_2 and the distribution of the points in it is also modeled as a normal distribution with variance v_2. In our tests, v_2 is set to 2 percent of v. For each time series position, we create the histograms with the use of random sampling. In order to represent this sample, we increment the corresponding bucket in the histogram by 1. We use 4000 samples for each time series position in order to fill up the large number of possible buckets (w buckets) in a reasonable way.

3.3 Results

We first present the results for the synthetic data set. We set the value of v to be 5, which is of the same order as the time series variance. The accuracy of each approach is shown in Figure 5. The compression ratio is illustrated on the X-axis, and the square error is illustrated on the Y-axis. We further note that the Y-axis is drawn on the logarithmic scale. Therefore, the difference between the different approaches is much greater than might appear visually. In each of the different methods, the error reduces as the compression ratio increases. This is natural, since a higher number of coefficients can be retained with increased compression ratio. We note that all the four variations of our uncertain wavelet decomposition approach significantly outperform the approach which treats the different wavelet coefficients separately. Among the four approaches using the wavelet decomposition technique, the local approach with a level-lookahead 2 achieves the highest accuracy, followed by the local approach with a level-lookahead 1, and then the two global approaches, which get the same cut order. We note that the local approach provides higher accuracy because of the better optimization in individual nodes. Even though each coefficient requires more space to store, the number of coefficients required for the same accuracy is far fewer.

We test the approaches on the Chlorine data set [21], which was generated using EPANET. As in the previous case, we set the value of v to be of the same order as the time-series variance, which in this case happened to be 2.0. The accuracies of each approach are shown in Figure 6. As in the previous case, the compression ratio is illustrated on the X-axis, whereas the accuracy is illustrated on a logarithmic scale on the Y-axis. From Figure 6, one can see that the local approach with a level-lookahead 2 is most accurate, while the accuracy of the standard approach is much lower than the others. Since the Y-axis is on a logarithmic scale, the differences are actually much greater than might appear visually in Figure 6.

We also tested the accuracy of each approach on sensor motes data set [22]. We first test on the time-series stream containing the humidity data. We set the value of v to be 0.1. The accuracy of each approach is shown in Figure 7. The accuracy of the standard approach is significantly lower than other approaches using the wavelet decomposition technique. As in the previous case, the Y-axis is set on a logarithmic scale. As in other cases, the local approach with

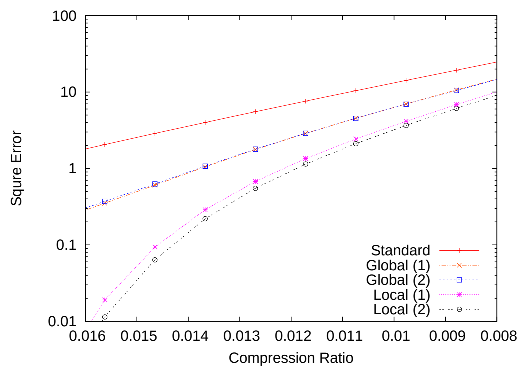

Figure 8: Accuracy on Sensor Data (Humid Temp)

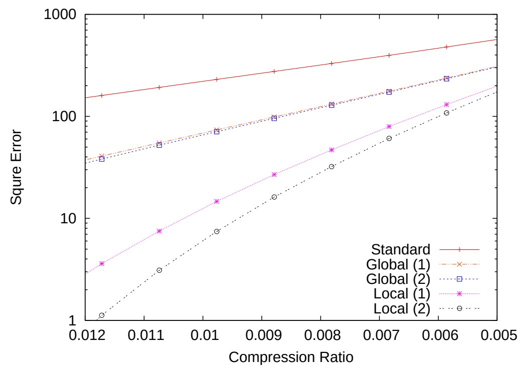

Figure 9: Accuracy on Sensor Data (Light)

a level-lookahead of 2 is still the most accurate. One interesting observation is that the global approach with a level-lookahead 1 achieves almost the same accuracy as the global approach with a level-lookahead 2; but by local approach, increasing the lookahead notably provides higher effective decomposition. We also note that the global technique provides fairly respectable accuracy across the different data streams.

We also tested the results on the Humid Temp streams. We set the value of v to be 0.1. As in the previous case, we illustrate results with logarithmic scaling on the Y-axis. The accuracies of all five approaches are illustrated in Figure 8. As in the previous case, all of our wavelet decomposition approaches were significantly superior to the standard decomposition approach. One can see that the approaches using the two proposed techniques show the same trends as we mentioned above.

Finally we test our approach on the light stream from the sensor data set. We set the value of v to be 1. We report the accuracy of each approach in Figure 9. As in the previous cases, the X-axis illustrates the compression ratio, whereas the Y-axis contains the behavior of the different techniques. Again, the local approach with a level-lookahead of 2 is most accurate. In this case, both the local approaches were significantly superior to the global approach to wavelet decomposition. Thus, the local approach with look-ahead of 1, is competitive with the global approach with look-ahead 2 in the sense that the relative ordering between these two approaches varies over the different data sets. As in all the previous cases, our wavelet decomposition technique significantly outperforms the standard decomposition technique. Thus, the proposed approach maintains its effectiveness over a wide variety of real and synthetic data sets.

Method	Synth.	Chlor.	Hum.	HumTemp	Light
Stand.	0.421	1.966	2.933	3.105	3.057
Glo(1)	0.297	1.045	1.981	1.840	1.685
Glo(2)	0.577	1.888	4.134	3.588	3.354
Loc(1)	0.296	1.451	2.543	2.653	2.512
Loc(2)	0.499	2.620	3.556	3.416	3.651

Table 2: Running Time on Different Data Sets

We also tested the efficiency of the different techniques. We note that the approach with standard decomposition is extremely simple because it requires no optimization. Therefore, this approach serves as a baseline in order to test the efficiency of the other approaches. The running times in seconds for each approach are presented in Table 2. We notice that the standard decomposition approach consumes more time than level-lookahead 1 techniques, but less than level-lookahead 2 techniques. The reason is that although the standard decomposition is a quite simple and direct extension of the Haar decomposition, it generates more coefficients than our proposed techniques by cutting on each rows and columns, which offsets the benefit of no optimization. For our proposed four approaches, in each case, it is clear that the local approaches are competitive with the global approaches in terms of running time. Therefore, the local method is much more desirable since it provides greater effectiveness at competitive efficiency. On the other hand, the level-lookahead 2 approaches are more time consuming than the level-lookahead 1 approaches. The running time of level-lookahead 2 techniques are about two times the running times of level-lookahead 1 approaches. One can also observe that the use of a global or local approach does not affect running time as much as the lookahead level. Whether we cut the nodes globally or locally, we need to compute the coefficients of each node in both cases, and the running time for a particular coefficient computation is not very different. But approaches with a level-lookahead 2 need to compute the coefficients of level $i + 1$ while we are working on level i, and there are four possible combinations which need to be computed. That is why a level-lookahead 2 approach is more time consuming than a level-lookahead 1 approach. In spite of these additional computations, our proposed methods are more efficient than the standard method. The additional cost is well within practical limits as a tradeoff for the tremendous advantages of the more sophisticated tree-based techniques in terms of effectiveness.

3.4 Sensitivity Analysis

In the previous experiments, we fixed the uncertainty variance for each data set in order to evaluate the efficiency and effectiveness of different approaches. In this section, we will first conduct a sensitivity analysis on different uncertainty levels for wavelet decomposition. We used the Sensor data (Light Stream) for the purpose of sensitivity analysis.

As we have mentioned, the *normalized representation* of the wavelet decomposition of the time series T is defined

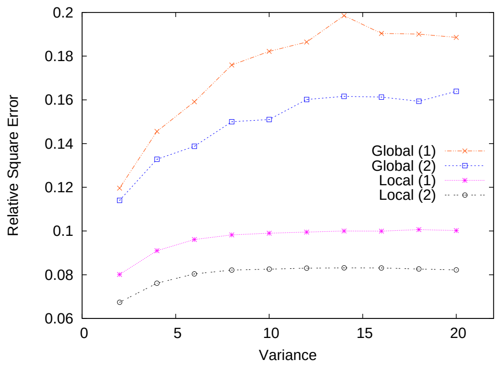

Figure 10: Sensitivity to Variance (Light)

according to Equation 4 as follows:

$$T = \sum_{i=1}^{M} w_i \cdot |\overline{e_i}| \cdot \frac{\overline{e_i}}{|\overline{e_i}|}$$

The total square error E on discarding all the wavelet coefficients is given by $E = \sum_{i=1}^{M} w_i^2 \cdot |\overline{e_i}|^2$. While the variance increases, the total absolute square error E decreases. In this case, the use of the absolute value of square error might not be appropriate for sensitivity analysis. In order to measure fairly, we use the relative value of square error to measure the performance. The relative square error R on discarding the subset of wavelet coefficients $i_1 \ldots i_k$ is defined as follows:

$$R = \frac{\sum_{j=1}^{k} w_{i_j}^2 \cdot |\overline{e_{i_j}}|^2}{E} \qquad (6)$$

Figure 10 illustrates the sensitivity analysis of different approaches with increasing variance v of the probability distribution function. The compression ratio was fixed at 0.488%. The variance v is illustrated on the X-axis, and the relative square error is illustrated on the Y-axis. Higher values of v correspond to probability density functions with greater uncertainty. As the variance increases, the relative square errors of all approaches increase. However, the local approaches increase only slightly whereas the square error trends of global approaches are more sensitive. The local approaches are less likely to be influenced by the fluctuations in the variance than the global approaches. This is because local approaches are much more flexible in optimizing different localities of the data irrespective of the underlying uncertainty level.

Since this paper addresses the problem of arbitrary probability distributions which are represented as histograms, the number of buckets defines the accuracy of representation. Clearly, greater granularity of representation results in greater computational constraints. Therefore, it is useful to test the effectiveness of the technique with increasing number of buckets. We note that the use of a larger number of buckets can represent the probability density function more accurately. However, larger number of buckets also tends to consume more time since it results in a larger wavelet tree. In order to understand the influence of the number of buckets on the efficiency, we test the execution time of all techniques with increasing number of buckets. The results for the light stream data are illustrated in Figure 11.

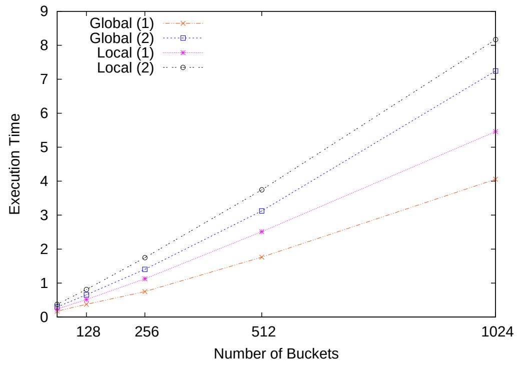

Figure 11: Scalability with Buckets (Light)

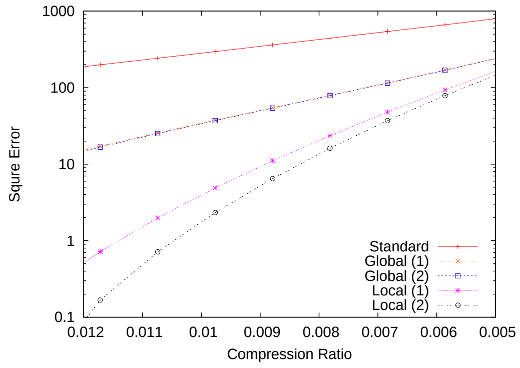

Figure 12: Accuracy on Exponential Dist. Mixture

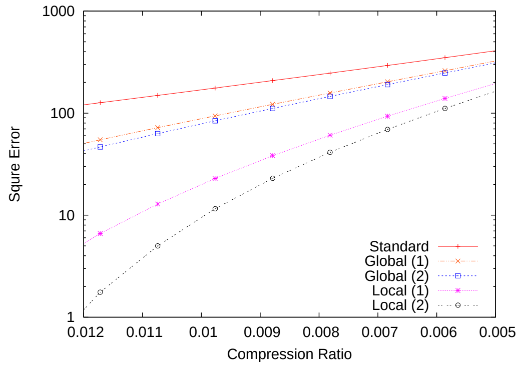

Figure 13: Accuracy on Uniform Dist. Mixture

We have illustrated the number of buckets on the X-axis, whereas the execution time is illustrated on the Y-axis. All techniques scale linearly with increasing number of buckets since the size of the constructed wavelet tree scales linearly with increasing number of buckets.

It is also valuable to test the effectiveness of different techniques on other probability distributions. In all the experiments presented here, the noise injected in the data was generated by a mixture of normal distributions. In this part, we will test the sensitivity by modeling the noise as a mixture of other kinds of distributions. The broad data of the generation technique remains the same, except that a different method is used to model the noise. In particular, we will test the effectiveness of the method on (1) a mixture of exponential distributions, and (2) a mixture of uniform distributions. In Figure 12, we have illustrated the accuracy of the method on a mixture of exponential distributions on sensor data (light stream). We set the exponential distribution rate parameter λ for each of the exponential distributions to be 1.5. We also illustrate the accuracy of the methods on a mixture of uniform distributions on the same data set in Figure 13. The uniform distributions are drawn from $[-2.5, 2.5]$ in each case. In both figures, the X-axis illustrates the compression ratio, and the Y-axis represents the accuracy on a logarithmic scale. The results are quite similar to the case of the normal distribution. In each case, our proposed wavelet techniques maintained an order of magnitude advantage over the baseline technique. Therefore, the proposed approaches are extremely robust and capable of maintaining their effectiveness and efficiency over a wide range of parameter settings and probability distributions.

4. CONCLUSIONS AND SUMMARY

In this paper, we discussed a new method for wavelet decomposition of uncertain data. We construct a method for wavelet decomposition on both the temporal and probabilistic aspects of the data, and design a strategy for optimizing the relative effect of both components. We show that such an approach is much more effective than direct applications of wavelet decomposition. This is because of a choice of careful optimization strategy which uses both the temporal and uncertain aspects carefully. We show that the approach is effective and efficient on real and synthetic data sets.

Acknowledgements

The work of the first and third authors are supported in part by NSF through grant IIS-0905215, IIS-0914934, and DBI-0960443.
The research of the second author was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-09-2-0053. The views and conclusions in this document are those of the authors and should not be interpreted as representing the official policies, either express or implied, of the Army Research Laboratory or the United States Government. The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

5. REFERENCES

[1] C.C. Aggarwal, P. S. Yu. A Survey of Uncertain Data Algorithms and Applications, *IEEE Trans. on Knowledge and Data Engineering*, 21(5), May 2009.

[2] C. C. Aggarwal (ed.) *Managing and Mining Uncertain Data*, Springer, 2009.

[3] C. C. Aggarwal (ed.) *Data Streams: Models and Algorithms*, Springer, 2007.

[4] C. C. Aggarwal. On Density Based Transformations for Uncertain Data Mining. *ICDE Conference*, 2007.

[5] C. C. Aggarwal. On Unifying Privacy and Uncertain Data Models. *ICDE Conference*, 2008.

[6] C. C. Aggarwal, Y. Li, J. Wang, J. Wang. Frequent Pattern Mining with Uncertain Data. *KDD Conference*, 2009.

[7] C. C. Aggarwal, P. S. Yu. Outlier Detection with Uncertain Data, *SDM Conference*, 2008.

[8] K. Chakrabarti, M. N. Garofalakis, R. Rastogi, K. Shim. Approximate query processing using wavelets. *VLDB Journal*, 10(2-3): 199-223, 2001.

[9] G. Cormode, M. Garofalakis: Histograms and Wavelets on Probabilistic Data, *ICDE Conference*, 2009.

[10] N. Dalvi, D. Suciu: Efficient Query Evaluation on Probabilistic Databases. *VLDB Conference*, 2004.

[11] A. Das Sarma, O. Benjelloun, A. Halevy, J. Widom: Working Models for Uncertain Data. *ICDE Conference*, 2006.

[12] T. S. Jayram, A. McGregor, S. Muthukrishnan, E. Vee. Estimating Statitistical Aggregates on Probabilistic Data Streams, *PODS Conference*, 2007.

[13] H.-P. Kriegel, M. Pfeifle: Density-based clustering of uncertain data. *KDD Conference*, 2005.

[14] L. V. S Lakshmanan, N. Leone, R. Ross, V. S. Subrahmanian. ProbView: A Flexible Database System. *ACM TODS*, 22(3):419–469, 1997.

[15] Y. Matias, J. S. Vitter, M. Wang, Wavelet-based histograms for selectivity estimation. *ACM SIGMOD Conference*, 1998.

[16] Y. Matias, J. S. Vitter, M. Wang, Dynamic Maintenance of Wavelet based histograms. *VLDB Conference*, 2000.

[17] S. Papadimitriou, J. Sun, C. Faloutsos: Streaming Pattern Discovery in Multiple Time-Series, *VLDB Conference*, 2005.

[18] E. Stolnitz, T. DeRose, T. Salesin. *Wavelets for Computer Graphics: Theory and Applications*, Morgan Kaufmann, 1996.

[19] E. Vee. Sketching Aggregates over Probabilistic Data Streams, *Managing and Mining Uncertain Data, (ed. C. Aggarwal)*, Springer, 2009.

[20] J. Vitter, M. Wang. Appproximate Computation of Multi-dimensional Aggregates of Sparse Data Using Wavelets. *ACM SIGMOD Conference*, 1999.

[21] **URL:** http://www.epa.gov/ORD/NRMRL/ /waswrd/epanet.html

[22] **URL:** http://berkeley.intel-research.net/labdata/