Accuracy Vs. Lifetime: Linear Sketches for Approximate Aggregate Range Queries in Sensor Networks

Konstantinos Kalpakis kalpakis@csee.umbc.edu
Vasundhara Puttagunta vputta1@csee.umbc.edu
Parag Namjoshi nam1@csee.umbc.edu

Computer Science & Electrical Engineering Department
University of Maryland, Baltimore County
1000 Hilltop Circle, Baltimore, MD 21250

ABSTRACT

The in–network aggregation paradigm in sensor networks provides a versatile approach for evaluating aggregate queries. Traditional approaches need a separate aggregate to be computed and communicated for each query and hence do not scale well with the number of queries. Since approximate query results are sufficient for many applications, we use an alternate approach based on summary data–structures. We consider two kinds of aggregate queries: *value range queries* that compute the number of sensors that report values in the given range, and *location range queries* that compute the sum of values reported by sensors in a given location range. We construct summary data–structures called *linear sketches*, over the sensor data using in–network aggregation and use them to answer aggregate queries in an approximate manner at the base–station. There is a trade–off between accuracy of the query results and lifetime of the sensor network that can be exploited to achieve increased lifetimes for a small loss in accuracy. Experimental results show that linear sketching achieves significant improvements in lifetime of sensor networks for only a small loss in accuracy of the queries. Further, our approach achieves more accurate query results than the other classical techniques using Discrete Fourier Transform and Discrete Wavelet Transform.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems—*query processing*; I.1.2 [**Symbolic and Algebraic Manipulation**]: Algorithms—*Algebraic Algorithms*

General Terms

Algorithms, Performance

Keywords

Sensor Networks, In–network Aggregation, Linear Sketching, Approximate Query Answering

DIALM-POMC'04, October 1, 2004, Philadelphia, Pennsylvania, USA.

1. INTRODUCTION

Wireless sensor networks consist of hundreds of low cost nodes that come with wireless communication and certain processing and storage capabilities in addition to sensing capabilities, and have severe energy constraints. These nodes are deployed in a *sensor field* and can be thought of as distributed streaming data sources. The goal of the sensor network is to collect information from the sensors that is required for various applications at a resource rich base–station. In doing so it should work in an energy efficient manner so that it survives for the longest period of time.

To this end, several energy aware mechanisms for collecting data have been developed. The data–gathering approaches focus on systematically collecting raw data from all the sensors to the base–station. This approach involves communicating large amounts of raw data leading to shorter lifetimes. Fortunately, it is usually not the individual raw data from the sensors that matters to many applications, but certain information in the form of summaries or aggregation of this data that is more relevant [13]. As a result, we have data–fusion or aggregation approaches that make use of the processing power of the sensors to aggregate the data from different nodes as it gets forwarded to the base–station, thereby reducing the communication costs greatly. This mechanism of *in–network aggregation* of data has been shown to dramatically increase lifetimes of sensor networks (Krishnamachari et al. [11]) and has emerged into a core service supported in them (e.g. TAG [14]).

Queries are considered a natural way for users to interact with the sensor network [17]. Further, sensor network applications rely heavily on query processing, particularly of aggregate queries. In this paper we are interested in evaluating aggregate queries over sensor networks in an energy–efficient manner.

1.1 Query Processing in Sensor Networks

From the database perspective, in–network aggregation is a mode for evaluating aggregate queries over the sensor network. The common approach to (aggregate) query processing in sensor networks has two phases: the *dissemination phase* and the *aggregation phase*. For example consider a query that finds the number of sensors in the sensor field that record temperature between 10^oF and 30^oF.

select count(*) from sensors
where $10 \leq$ temperature ≤ 30.

During the dissemination phase, the query is injected into

the sensor network to all the sensors and during the aggregation phase, an aggregation–tree is imposed on the sensor network along which in–network aggregation is performed as follows. Every node waits for the counts from each of its child–nodes and adds them up. It contributes to the COUNT only if the where–clause is satisfied and forwards the aggregated count to its parent–node. Eventually, the result of the query is computed at the root (base–station). This approach answers queries accurately. However, it has several drawbacks. Every query needs to be disseminated to the sensors and this is an expensive operation because of the communication involved. Further, when we have multiple aggregate queries (for e.g. queries with different ranges in the where–clause), multiple aggregates need to be maintained and communicated, one for each query. This severely restricts the number of queries that can be answered before the sensor network dies.

To overcome these drawbacks, we propose an alternate approach that is inspired by *OnLine Analytical Processing* (OLAP) and query processing over streams. The idea is to maintain and update a small space summary data–structure in one–pass over the data or as data appear in the stream. The summary data–structure is then used to directly answer queries quickly in an approximate manner. The constraints that are imposed by streams (see Babu and Widom [1]) make techniques for stream computations suitable for sensors. These summary data–structures (also known as synopses) are typically computed as decomposable aggregates, i.e. they can be expressed as an aggregation function f over sets a and b so that $f(a \cup b) = g(f(a), f(b))$, where g is called the combine–function that is computable in small space and time. Since decomposable functions are suitable for the in–network aggregation paradigm, a similar approach can be taken for answering queries in sensor networks. During each round, the summary data structure can be computed over the data from all the sensors using in–network aggregation. The user queries can then be answered directly from this summary data–structure at the base–station at the end of each round. This way, the lifetime of the sensor network is not limited by the number of queries. However, it is limited by (a) the amount of computation at each sensor to compute the synopses which determines the number of CPU cycles and the corresponding energy, (b) the space complexity for computing the synopses which determines the energy required to power memory on the sensor, and most importantly (c) the size of the synopses which determines the energy for transmitting and receiving using wireless radio. Usually there is a direct relationship between the size of the synopses and the accuracy of the queries. Since most monitoring applications are interested in approximate query answers, **this approach offers a viable option for increasing the lifetime of the sensor network for a small loss in accuracy of the query results**.

It is important to note that the synopses data structure to be computed depends on the kind of queries we wish to answer. We consider the following two important aggregate range queries.

Value Range Queries: How many sensors recorded values in a particular range?

select count(*) from sensors
where $10 \leq$ reading ≤ 20.

Location Range Queries: What is the aggregate of values recorded by sensors located in a given rectangular region(range)?

select average(reading) from sensors
where $10 \leq x \leq 20$ and $10 \leq y \leq 20$.

Histograms are popularly used as summary data structures for answering such queries and several sketch based approaches for histograms have been suggested [16, 6]. However these methods are designed to capture the norm of the histogram. They depend solely on the data and do not consider the queries at all. We often have some information about the queries. For example certain regions (ranges) could be more interesting and hence queried more often. Further, in the case of monitoring applications, it is common to have continuous queries, in which case, the same range query is posed in every round. Therefore it is important to design sketches for a specific set of aggregate range queries. In this paper we propose to use linear sketches [15] that we developed for answering approximate aggregate range queries effectively and efficiently over sensor networks to get improved lifetimes for a small loss in accuracy of the results.

2. LINEAR SKETCHING APPROACH

In this section we present the theory of linear sketches (refer to [15] for details) and show how they can be used to answer aggregate range queries in sensor networks.

2.1 Linear Sketching Preliminaries

DEFINITION 1. *Let A be a matrix over the field of complex numbers, $\mathcal{C}$. The complex conjugate of A is denoted by $\overline{A}$. The conjugate transpose A^* of A is defined as $A^* = \overline{A}^T$.* ■

DEFINITION 2. (STANDARD INNER PRODUCT, NORM) *For any two complex vectors $x, y \in \mathcal{C}^n$, the standard inner product is the scalar given by $\langle x, y \rangle = y^* x$. Norm of x is given by $\|x\|^2 = \langle x, x \rangle = x^* x$.* ■

DEFINITION 3. (LINEAR SKETCH) *Let P be an $N \times k$ complex projection matrix, with columns $p_1, p_2, \cdots, p_k$ so that $P = [p_1 p_2 \cdots p_k]$. We call P a sketching matrix. Let x be a vector in $\mathcal{C}^N$. The projection of x onto p_i is given by $\langle x, p_i \rangle$, $i = 1, 2, \ldots, k$. The (linear) sketch or projection of x with respect to the sketching matrix P is the k–dimensional vector given by $\widehat{x} = [\langle x, p_1 \rangle \langle x, p_2 \rangle \cdots \langle x, p_k \rangle]^T$. Equivalently,*

$$\widehat{x} = P^* x \tag{1}$$

The sketch of an $N \times m$ matrix A with respect to the sketching matrix P is a matrix whose columns are sketches of the corresponding columns of A, i.e. $\widehat{A} = P^* A$. ■

Let x be a $N \times 1$ data vector. Consider a query vector q of the same length that is evaluated against x using the standard inner product as follows.

$$a = \langle x, q \rangle = q^* x \tag{2}$$

For example, when q is a 0–1 vector, Equation 2 simply sums the components of x where there are 1's in q. We refer to such queries as *sum–queries*. Let Q be an $N \times m$ matrix whose columns correspond to m such sum–queries. Then from Equations 1 and 2, we can see that $\widehat{x} = Q^* x$ yields a vector of length m, whose components are exact answers to the m queries in the query matrix Q. In other words,

when we use the query matrix Q as the sketching matrix, the sketch $\widehat{x}$ of the data vector gives the exact answers to all the queries. In general, the number of queries m can be very large making the sketch too big. The cost of computing the entire sketch as well as communicating it increases with the number of queries. For this reason we explore the following idea: *Can we find just a small number (k) of special queries, whose answers we can maintain accurately, so that we can use their answers to answer all the queries in Q in an approximate manner?*

Suppose that these k special queries form our sketching matrix P. Then the sketch of the data vector contains exact answers to the special queries. An approximate answer to the query q with respect to vector x can be estimated using their sketches $\widehat{q}$ and $\widehat{x}$ respectively as follows.

$$\widetilde{a} = \langle \widehat{x}, \widehat{q} \rangle = \widehat{q}^* \widehat{x} \tag{3}$$

Further the error in estimating the answer to the queries is given by the error vector, $e = a - \widetilde{a}$ and the sum of squared errors (SSE) over all the queries is given by $SSE = \|e\|^2 = e^* e$.

Definition 4. (Fourier Matrix)
A Fourier matrix F of order n is a square matrix for which

$$F^*_{(j,k)} = \frac{1}{\sqrt{n}} \omega^{(j-1)(k-1)} \tag{4}$$

where ω^i for $i = 0 \ldots n-1$ are the n distinct complex roots of unity. Hence,

$$F^* = \frac{1}{\sqrt{n}} \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \omega & \omega^2 & \cdots & \omega^{n-1} \\ 1 & \omega^2 & \omega^4 & \cdots & \omega^{n-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & \omega^{n-1} & \omega^{n-2} & \cdots & \omega \end{bmatrix} \tag{5}$$

We refer to the columns of F^* as Fourier vectors. ■

Definition 5. (Discrete Fourier Transform)
The Discrete Fourier Transform (DFT) of a vector $x \in \mathcal{C}^n$ is given by

$$\widehat{x} = Fx, \tag{6}$$

■

From the signal processing perspective, when the sketching matrix P contains the Fourier vectors, then $\widehat{x}$ contains simply the Fourier coefficients of x (by comparing Equations 1 and 6). Similarly when P contains the wavelet vectors, $\widehat{x}$ gives the respective wavelet coefficients of x. The standard practice is to maintain the top–k coefficients of the data vector which implicitly means having only vectors corresponding to the top–k coefficients as the sketching matrix. These approaches rely on the ability to capture the norm of the data vector using the top–k coefficients and do not make use of the queries at all. (The projection matrix depends only on the data vector.) Further, in the case of streaming data, the errors in estimating query results escalate with the error in estimating the top–k coefficients.

However, queries often have certain patterns and structures that can be exploited to get better results. For this reason, we propose to use an alternate approach of using sketching matrices that depend on the queries. We have the following theorem on sketching matrices [15].

Theorem 1. *Let Q be an $N \times m$ query matrix, x a non–zero data vector, and P be an $N \times k$ sketching matrix with orthonormal columns. Let $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$ be the eigenvalues of the matrix QQ^*, and let $v_1, v_2, \ldots, v_N$ be its corresponding orthonormal eigenvectors. Then we can achieve the following bound on the SSE by choosing P to have as columns the top–k eigenvectors of the matrix QQ^*, i.e. when $P = [v_{N-k+1} v_{N-k+2} \ldots v_N]$.*

$$\|e\|^2 \leq \lambda_{N-k} \|x\|^2 \tag{7}$$

■

Therefore we use the sketching matrix whose vectors are the top–k eigenvectors of the matrix QQ^*. Such a sketching matrix minimizes (in a certain sense) the SSE and thus the mean squared error with respect to the queries at hand.

Further, in the case of range queries, the query vectors contain blocks of consecutive ones. If Q contains all range queries of a fixed extent (i.e. query vectors with fixed number of consecutive ones) then both Q and QQ^* are circulant matrices.

Definition 6. *(*Circulant Matrix)
A circulant (matrix) of order n, is a square matrix of the form

$$\begin{aligned} C &= circ(c_0, c_1, \ldots, c_{n-1}) \\ &= \begin{bmatrix} c_0 & c_1 & \cdots & c_{n-1} \\ c_{n-1} & c_0 & \cdots & c_{n-2} \\ \vdots & \vdots & \vdots & \vdots \\ c_1 & c_2 & \cdots & c_0 \end{bmatrix} \end{aligned} \tag{8}$$

Each row of C is a circular right shift of its preceding row. ■

For example, the following query matrix Q has as columns, all range queries of fixed extent 2 with $N = 4$.

$$Q = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

Note that we consider the domain values N and 1 to be adjacent. The last column of Q is a query that wraps around the boundary. We shall refer to such queries as wrap–around queries. Although wrap–around queries may be uncommon, we include them here so that the query matrix is circulant and for the algebraic properties that follow.

We have the following theorem on circulant matrices.

Theorem 2 (See Davis [4]). *(Universal Eigenvectors of Circulants)*
Let C be a circulant matrix of order n. Matrix C is diagonalized by the Fourier matrix F of order n,

$$C = F^* \Lambda F, \tag{9}$$

where Λ is a diagonal matrix with the eigenvalues of C as its diagonal elements. Thus, each column of F^* is a (right) eigenvector of every circulant matrix, i.e. the columns of F^* is a universal set of eigenvectors for for all the circulant matrices. ■

Therefore by Theorem 2, the eigenvectors of QQ^* are in fact the Fourier vectors. Thus we can have the sketching

matrix P to be the certain Fourier vectors corresponding to the top–k eigenvalues of QQ^* and then the sketch $\widehat{x}$ is simply the corresponding Fourier coefficients of x. Note that the choice of the Fourier coefficients depends solely on the query matrix Q and may not be the same as the top–k Fourier coefficients of the data vector x. The advantage of Fourier vectors is that they have a succinct representation and do not have to be explicitly stored and any element of the sketching matrix can be generated on the fly using small space and time. Note that even if the query that needs to be evaluated is different from those in the query matrix Q, we can still evaluate it approximately from the sketches with reasonable errors (see the next section for experimental results and [15] for more detailed experiments). Further, there are ways to extend an existing sketching matrix to improve the accuracy of a given query matrix (see [15]).

In the remaining part of the section, we show how the above results can be applied to answer value range queries and location range queries over sensor networks.

2.2 Answering Value Range Queries

The frequency distribution of the sensor data gives the number of sensors recording a particular value. Maintaining the frequency distribution of values from all the sensors allows us to answer value range queries accurately by simply adding the frequencies reported at the values in the query range.

We assume that the values recorded by sensors take integer values between 1 and M. Therefore, the frequency distribution can be thought of as an $M \times 1$ column vector, h. Similarly, a value range query can then be thought of as a 0–1 column vector q of the same size, with 1's in the range of the query and 0's everywhere else. Then the exact answer to the range query can be answered using Equation 2 as $a = \langle h, q \rangle = q^* h$. To use this approach we need to be able to compute the frequency distribution, h over the sensor network. We observe that, if we consider each of the sensors generating an $M \times 1$ column vector with a 1 at the index corresponding to the value it measures, then h is simply the sum of such vectors from all the sensors. Let $h^{(i)}$ be the vector generated by the i^{th} sensor, then $h = \sum_{i=1}^{n} h^{(i)}$. In other words, h is a simple *SUM* aggregate of the vectors generated at each sensor and can therefore be computed using any in–network data–aggregation scheme in sensor–networks. However, this approach will need packets of size $O(M)$ to be communicated and aggregated which may be infeasible due to energy constraints.

We propose to apply the linear sketching approach mentioned previously for this problem. Because sketching is a linear operator, the sketch $\widehat{h}$ using the sketching matrix P is simply the sum of sketches of the individual vectors generated at each sensor. i.e. $\widehat{h} = P^* h = P^* \sum_{i=1}^{n} h^{(i)} = \sum_{i=1}^{n} P^* h^{(i)} = \sum_{i=1}^{n} \widehat{h}^{(i)}$. Therefore, just as h is a sum–aggregate over the vectors generated at each sensor, $\widehat{h}$ is also a sum–aggregate over the sketches of these vectors using the sketching matrix P and can therefore be carried out over the sensor–network using any data–aggregation service.

Also note that computing the individual sketch at each sensor, requires storing the matrix P at each sensor. Further, since there is only a single non–zero component in any $h^{(i)}$, computing the sketch requires only $O(k)$ arithmetic multiplications. Further, in the case of all range queries of fixed size, the projection matrix P is simply certain Fourier vectors that have a succinct representation, and therefore does not have to be explicitly stored. Any element of P can be generated on the fly using very little memory and computing. Using a b–byte representation for each component of the sketch, we need to communicate and aggregate packets of size $(bk + p)$ bytes where p is the header size. Typically $k << M$. We consider $p = 8$ and $b = 2$. The actual queries are evaluated at the base–station using the sketch $\widehat{h}$ and the sketch of the query vector as shown in Equation 3.

2.3 Answering Location Range Queries

Location range queries are the queries that ask for the sum of values reported by sensors located in a particular (spatial) range. In this case, we consider data generated by all the sensors as a single vector that is indexed by sensor–id. Therefore the size of the vector is the same as the number of sensors n. When we consider any ordering of the sensors, the range query translates into a sensor-query, i.e. a 0–1 vector with 1s in the place where the corresponding sensor is located within the query range. The sensor–queries need not contain blocks of consecutive 1's. Therefore, the sketching matrix in this case is not the Fourier matrix and may not have a succinct representation. We use the sketching matrix that corresponds to the top–k eigenvectors of QQ^*, where Q is a a query matrix obtained sensor–queries corresponding to the location range queries.

3. EXPERIMENTAL EVALUATION

We perform experiments to observe the trade-off between the gain in lifetime of a sensor network and loss in accuracy in estimating aggregate queries over sensor networks using in–network aggregation. The results presented in this section are based on the following experimental setup.

We have a synthetically generated field in which sensors are randomly placed and there is a single base–station. Queries are posed at the base–station. The readings of the sensors vary over time. In every round, data from the sensors is gathered or aggregated (depending on the approach) and the queries are evaluated at the base–station. We estimate the lifetimes of the sensor networks. We assume that the energy for communication dominates the energy used for processing and ignore the computation cost. The detailed description of the experiments are as follows.

3.1 Sensor Fields

We consider a $100m \times 100m$ field with 200 sensors that are randomly placed and the base station located at (50m,300m). All sensors are assumed to have equal initial energy of 1 Joule. Sensor fields are usually spatially correlated and exhibit periodicity. We generate synthetic data for the sensor fields that exhibit these properties using intuitive techniques. Photographs exhibit strong spatial correlation among their pixel values. So, we consider the pixel values of a picture to be the initial field distribution to account for the spatial correlation of the data. More specifically we considered a 100×100 subimage of the second band from the Landsat image of Miami [18] to be the initial field value distribution for our synthetic sensor field. Further, this is used to generate different values over time to exhibit periodicity as well as randomness. The value sensed by a sensor is the value of the field at the cell in which it is located. The value sensed by the $i^t h$ sensor at time t is given by $X_i(t)$.

We represent the values at time $t = 0$, $X_i(0)$ as μ_i. These come from the initial fields. Then the values at each sensor change with time as a sine curve along with some noise according to $X_i(t) = \mu_i + a_i \sin(\frac{2\pi t}{\tau_i} + \phi_i) + \varepsilon_i$. Here a_i, τ_i and ϕ_i are amplitude, frequency and phase for the change in values sensed by sensor i. These are time invariant for all the sensors and are chosen as follows. $a_i = \mu_i/3$. $\tau_i = 60$. ϕ_i is chosen randomly according to a uniform distribution in $(0, \pi)$. ε_i is a random variable that corresponds to white noise with variance $(\mu_i/10)^2$.

3.2 Queries

The queries are posed at the base–station. All queries that arrived during the current round are evaluated at the end of the current round. In the case of continuous queries, the same query is posed in every round as long as the query is active. We consider two kinds of aggregate range queries for our experiments. **Value Range Queries** are queries that ask for the number of sensors that record values in the specified value range. We consider *fixed extent* value queries $ValQFixed(v)$, which is a set all queries whose range is of fixed extent v. In reality, wrap–around queries are not expected and are not included here. We also consider value range queries of *random extent* $ValQRand(\mu, \sigma^2)$, which is a set of 100 queries whose extent is chosen randomly from a normal distribution with a mean μ and variance σ^2. **Location Range Queries** ask for sum of values recorded by sensors located in a particular rectangle in the field. We consider location queries of fixed extent ($LocQFixed(v)$) which is the set of all queries of fixed $v \times v$ spatial extent with at least 1 sensor.

3.3 Quantitative Performance Measures

We use the following quantities to measure the performance of different techniques:

MLT: Mean LifeTime of the sensor network as the number of rounds it lasts until the first sensor drains out of energy, where the mean is computed over 20 different placements of sensors in the field.

MSE: Mean of Squared Errors of results to all the queries over all the rounds and over all the placements.

RLE: Mean Relative Errors of results to all the queries over all the rounds and over all the placements.

REN: Relative Error in the norm of the frequency distribution over all the rounds and over all the placements.

3.4 Approaches

We compare results from the following approaches for evaluating queries.

No Aggregation (NoA): This is a data–gathering approach where values recorded by each sensor are forwarded without any aggregation. We use the algorithm due to Chang and Tassiulas [2] for routing. The packet size is 10 bytes with 2 bytes for data and 8 bytes for header. A complete histogram is computed at the base–station to answer the queries. This approach gives exact answers to all the queries as well as the exact norm.

The summary data–structure based approaches using in–network aggregation for estimating queries are described next. We use the data–aggregation technique by Kalpakis et al [10] for evaluating the lifetimes for these approaches

Naive Aggregation (NA): In this approach, each packet has n slots, one for each sensor. Each sensor generates a packet with its own value in the reserved slot and zeros elsewhere. Data aggregation is performed by addition of the corresponding $n \times 1$ vectors. This approach maintains the exact histogram, therefore the queries will be answered exactly. We assume two bytes for each sensor value and 8 bytes for the header. Therefore, the packet size is $2n + 8$ bytes.

Haar Wavelets (DWT): We maintain the top–k Haar wavelet coefficients. Normally we do not know which coefficients are the the top–k coefficients. Thus they must be estimated using in–network aggregation [8, 6]. We present results using the true top–k DWT coefficients. (The results with the estimated top–k DWT coefficients are not reported due to space constraints.) With header size of 8 bytes and 2 byte representation for each coefficient, the packet size in this case is $2k + 8$ bytes, where k is the size of the sketch.

Discrete Fourier Transform (DFT): Here we use the top–k DFT coefficients of the data vector.

Linear Sketching (LS): Here, we use the linear sketching approach described in Section 2 to compute the sketch of of size k. In the case of fixed extent queries we use the sketching matrix due to all queries of the same fixed size. In the case of random queries, we use the sketching matrix due to all queries of a fixed extent that is the same as the mean extent used to generate the queries. We use 2 byte representation for each component of the sketch. Therefore the size of the packet in this case will be $2k + 8$ bytes.

3.5 Experiments

Experiment 1: We consider 20 different sensor placements and report the mean lifetime. For each placement we consider fixed size value range queries $ValQFixed(v)$ for 60 rounds. We chose $v = 35$ that is 10% of the total range (354). We report MLT, MSE, RLE and REN for the different approaches in Table 1 and Figures 1 and 2.

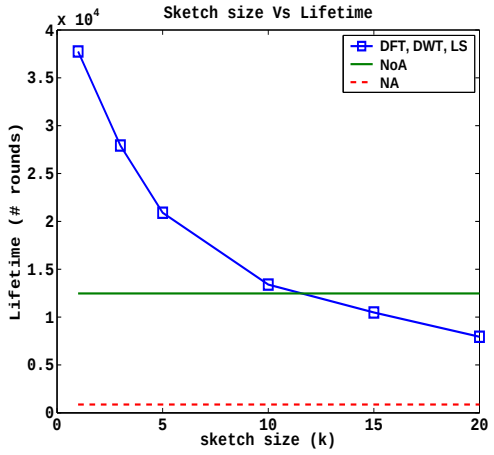

Figure 1: Sketch size Vs. Lifetime

NoA and NA approaches give exact results, therefore, the MSE, REL and REN are always zero. For the remaining aggregate based approaches we present the results with different sketch sizes. We note that as the sketch size increases, the lifetime of the sensor networks decreases and the errors in estimating the range queries decrease. The LS method gives significantly lesser errors in estimating answers than DFT and DWT. We also observe that although the DFT approach is better at capturing the norm, the accuracy of

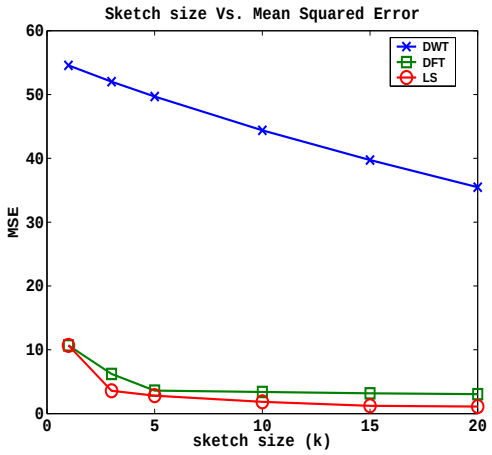

Figure 2: Sketch size Vs. Mean Squared Error in Experiment 1. $M = 354$; Query set: $ValQFixed(35)$; Mean exact answer: 56.03.

query answers is worse than LS. Further, with a sketch size of 3, LS achieved a MLT of 27, 925 which is 124% improvement over NoA and 6.48% RLE. Therefore it is evident that the LS method improves lifetime of the sensor network while incurring only a small loss in accuracy.

Experiment 2: The set up for this experiment is the same as above, except that we use a different query set in each round. We use value range queries with ranges of random size, $ValQRand(v, \sigma^2)$ with v as 10% of the range that the values that the sensors record and σ^2 as 16, i.e. $ValQRand(35, 16)$. For the LS method we use the $ValQFixed(v)$ set of queries for only forming the sketching matrix and the random queries are used for evaluation. The results are reported in Table 1. Note that although LS uses a different set of queries for the sketching matrix, it is still able to estimate answers to the original queries very well. LS gives significantly more accurate query results than DFT or DWT. We observe LS with sketch size of 3 gives 6.93% RLE and significant improvement in MLT. Therefore, it is clear that the LS sketching method works even for queries that were not used to prepare the sketching matrix.

Experiment 3: Finally we perform experiments with location range queries. For each placement over the sensor field we consider the location query set $LocQFixed(30)$. The average number of sensors in the location queries is 5.12. For the LS method, we use the top–k eigenvectors of the matrix QQ^* as the sketching matrix. The results are summarized in Table 1. Again, LS achieves better accuracy than DWT and DFT. Using a sketch size of 10, the LS method achieves a MLT of 13,398 that is 7.4% better than NoA, with a RLE of 11.85%. Therefore, from this experiment, we observe that LS achieve better estimates for the location queries than what DWT or DFT achieve and that too using small sketch sizes that gives improved lifetimes.

4. PREVIOUS WORK

Several studies have been carried out towards performing in–network aggregation in sensor networks [7, 9, 10]. The main focus here is to come up with good data aggregation trees along which in–network aggregation can be carried out. Kalpakis et al. [9] propose the Maximum Lifetime Data Aggregation algorithm for data aggregation that gives a near optimal data–aggregation schedule consisting of aggregation trees. They present a computationally efficient approximate scheme for maximizing lifetime [10]. Chang et al. [2] give a flow–based approach to gather data (without aggregation) in an optimal manner.

Aggregate query processing in sensor networks has received a lot of attention recently [12, 14, 17]. Cosidine et. al. [3] present approximate sketches for COUNT, SUM and AVG that are robust with respect to node failures,etc. However these approaches maintain a separate aggregate for each query with a different where–clause (range) and do not scale well with the number of queries. Our approach works by maintaining summary data–structures using in–network aggregation. Several such data structures have been proposed for query answering over streaming data (see for e.g. Gibbons and Matias [5]). Small space samples and histograms are popularly used for answering selectivity and aggregate range queries and to compute the size of joins and number of distinct elements. Recently, several sketch based summary data structures have been proposed [16, 6]. In the sensors domain, Hellerstein et al. [8] argue that monitoring applications demand more sophisticated aggregate query processing over sensor networks. They compute wavelets over the sensor data that can be used to answer approximate aggregate range queries. However, these summary data–structures are not optimized for aggregate range queries.

5. CONCLUSION

Techniques based on summary data–structures for approximate query answering fit well with the in–network aggregation paradigm in sensor networks. In this paper we propose summary data–structures called *linear sketches* for answering aggregate range queries over sensor networks. While the accuracy of the query results increases with the size of the sketch, the lifetime of the sensor network decreases. Therefore there is a trade–off between accuracy of the query results and lifetime of the sensor network that we exploit to achieve significant increase in lifetimes while incurring a small loss in accuracy of query results. We performed several experiments to compare the accuracy of queries as well as lifetime of the sensor networks. The proposed method of linear sketching achieves much better accuracy compared to results using classical techniques such as DFT and DWT and significant gains in lifetimes compared to the naive aggregation and no aggregation schemes, in the case of both value range queries as well as location range queries.

6. ACKNOWLEDGMENTS

This work was supported in part by NASA under Cooperative Agreement NCC5–315.

7. REFERENCES

[1] S. Babu and J. Widom. Continuous queries over data streams. *SIGMOD Rec.*, 30(3):109–120, 2001.

[2] J.-H. Chang and L. Tassiulas. Energy conserving routing in wireless ad-hoc networks. In *Proceedings of IEEE INFOCOM*, volume 1, pages 22–31, 2000.

[3] J. Considine, F. Li, G. Kollios, and J. W. Byers. Approximate aggregation techniques for sensor databases. In *Proceedings of the 20th International*

Table 1: Experimental Results

Notation

p	Packet size in bytes
k	Sketch size
MLT	Mean life time over 20 different random placements of the sensor network
MSE	Mean squared error over all the queries
RLE	Mean relative error over all the queries
REN	Relative error in the norm of frequency distribution

Experiment 1: Results with fixed size value range queries
M = 354; Query set: $ValQFixed(35)$; Mean exact answer: 56.03.

Method	p	MLT	MSE			RLE			REN		
NoA	10	12,472	0			0			0		
NA	408	865	0			0			0		
Other Aggregation methods (DWT/DFT/LS)											
k	p	MLT	MSE			RLE %			REN %		
			DWT	DFT	LS	DWT	DFT	LS	DWT	DFT	LS
1	10	37,749	54.57	10.68	10.68	97.98	20.03	20.03	96.44	26.59	26.59
3	14	27,925	52.02	6.21	3.57	94.32	11.53	6.48	90.87	22.31	23.14
5	18	20,901	49.68	3.60	2.81	90.91	6.55	4.99	86.20	19.94	22.93
10	28	13,398	44.39	3.39	1.84	83.00	6.16	3.28	76.45	17.72	22.63
15	38	10,487	39.73	3.18	1.21	75.81	5.78	2.12	68.41	16.46	22.38
20	48	7,945	35.48	3.05	1.10	69.18	5.54	1.93	61.45	15.47	22.14

Experiment 2: Results with random size value range queries.
$M = 354$; Query Set: $ValQRand(35, 16)$; Mean exact answer: 53.80.

Method	p	MLT	MSE			RLE			REN		
NoA	10	12,472	0			0			0		
NA	408	865	0			0			0		
Other Aggregation methods (DWT/DFT/LS)											
k	p	MLT	MSE			RLE %			REN %		
			DWT	DFT	LS	DWT	DFT	LS	DWT	DFT	LS
1	10	37,749	52.48	10.53	10.53	98.17	21.06	21.06	96.44	26.59	26.59
3	14	27,925	50.16	6.15	3.53	94.84	12.24	6.93	90.87	22.31	22.14
5	18	20,901	48.03	3.57	2.74	91.71	7.01	5.20	86.20	19.94	22.93
10	28	13,398	43.15	3.35	1.80	84.38	6.58	3.44	76.45	17.72	22.63
15	38	10,487	38.82	3.14	1.20	77.64	6.16	2.26	68.41	16.46	22.38
20	48	7,945	34.85	3.01	1.10	71.39	5.89	2.07	61.45	15.47	22.14

Experiment 3: Results with location range queries of fixed extent.
Query Set: $LocQFixed(30)$; Mean exact answer: 1446.

Method	p	MLT	MSE			RLE			REN		
NoA	10	12,472	0			0			0		
NA	408	865	0			0			0		
Other Aggregation methods (DWT/DFT/LS)											
k	p	MLT	MSE			RLE %			REN %		
			DWT	DFT	LS	DWT	DFT	LS	DWT	DFT	LS
1	10	37,749	1250	294	919	86.23	21.17	67.18	85.90	14.72	85.38
3	14	27,925	826	277	503	55.44	20.30	37.01	57.83	13.95	71.16
5	18	20,901	475	266	329	31.11	19.47	25.33	33.27	13.34	62.17
10	28	13,398	264	244	159	19.27	17.82	11.85	12.76	12.06	49.55
15	38	10,487	235	223	100	17.25	16.31	7.46	10.95	10.98	43.34
20	48	7,945	212	207	75	15.62	15.13	5.67	9.55	10.05	40.32

Conference on Data Engineering (ICDE), pages 449–460. IEEE Computer Society, 2004.

[4] P. J. Davis. *Circulant Matrices*. John Wiley & Sons, Inc., 1979.

[5] P. B. Gibbons and Y. Matias. Synopsis data structures for massive data sets. *DIMACS: Series in Discrete Mathematics and Theoretical Computer Science: Special Issue on External Memory Algorithms and Visualization*, A:39–70, 1999.

[6] S. Guha, P. Indyk, S. Muthukrishnan, and M. Strauss. Histogramming data streams with fast per-item processing. In *Proceedings of the 29th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 681–692, 2002.

[7] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-efficient communication protocol for wireless microsensor networks. In *Proceedings of the 33rd Hawaii International Conference on System Sciences-Volume 8*. IEEE Computer Society, 2000.

[8] J. M. Hellerstein, W. Hong, S. R. Madden, and K. Stanek. Beyond Average: Towards Sophisticated Sensing with Queries. In *Proceedings of 2nd International Workshop on Information Processing in Sensor Networks (IPSN)*, 2003.

[9] K. Kalpakis, K. Dasgupta, and P. Namjoshi. Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks. *Computer Networks*, 42(6):697–716, 2003.

[10] K. Kalpakis, K. Dasgupta, and P. Namjoshi. Improving the lifetime of sensor networks via intelligent selection of data aggregation trees. In *Proceedings of Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS'03)*, 2003.

[11] B. Krishnamachari, D. Estrin, and S. Wicker. The Impact of Data Aggregation in Wireless Sensor Networks. In *Proceedings of International Workshop on Distributed Event-Based Systems*, 2002.

[12] S. Madden and M. J. Franklin. Fjording the stream: An architecture for queries over streaming sensor data. In *ICDE*, 2002.

[13] S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler. Supporting Aggregate Queries Over Ad-Hoc Wireless Sensor Networks. In *Proceedings of 4th IEEE Workshop on Mobile Computing and Systems Applications*, 2002.

[14] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. In *Proceedings of the ACM Symposium on Operating System Design and Implementation (OSDI)*, December 2002.

[15] V. Puttagunta and K. Kalpakis. Answering approximate aggregate queries using linear sketches. Technical Report TR-CS-03-29, University of Maryland Baltimore County, December 2003.

[16] N. Thaper, S. Guha, P. Indyk, and N. Koudas. Dynamic multidimensional histograms. In *Proceedings of the ACM SIGMOD international conference on Management of data*, pages 428–439, 2002.

[17] Y. Yao and J. E. Gehrke. Query processing in sensor networks. In *Proceedings of the First Biennial Conference on Innovative Data Systems Research (CIDR)*, 2003.

[18] http://www.nnic.noaa.gov/socc/gallery.htm. Web page.