

DIMENSIONS: Why do we need a new Data Handling architecture for Sensor Networks?

Deepak Ganesan
Dept of Computer Science
UCLA
Los Angeles, CA 90095
deepak@cs.ucla.edu

Deborah Estrin
Dept of Computer Science
UCLA
Los Angeles, CA 90095
destrin@cs.ucla.edu

John Heidemann
USC/ISI
4676, Admiralty Way
Marina Del Rey, CA 90292
johnh@isi.edu

ABSTRACT

An important class of networked systems is emerging that involve very large numbers of small, low-power, wireless devices. These systems offer the ability to sense the environment densely, offering unprecedented opportunities for many scientific disciplines to observe the physical world. In this paper, we argue that a data handling architecture for these devices should incorporate their extreme resource constraints - energy, storage and processing - and spatio-temporal interpretation of the physical world in the design, cost model, and metrics of evaluation. We describe DIMENSIONS, a system that provides a unified view of data handling in sensor networks, incorporating long-term storage, multi-resolution data access and spatio-temporal pattern mining.

1. INTRODUCTION

Wireless Sensor Networks offer new opportunities for pervasive monitoring of the environment and ultimately for studying previously unobservable phenomena. Using traditional techniques, the data handling requirements of these systems will overwhelm the stringent resource constraints on sensor nodes [1]. In this paper, we describe DIMENSIONS, a system to enable scientists to observe, analyze and query distributed sensor data at multiple resolutions, while exploiting spatio-temporal correlation.

Application	Observed phenomena
Building Health Monitoring [2]	response to earthquakes, strong winds
Contaminant Flow	Concentration pattern, pooling of contaminants, plume tracking
Habitat micro-climate monitoring [3]	spatial and temporal variations

Table 1: Example Scientific Applications

Sensor networks place several requirements on a distributed storage infrastructure. These systems are highly data-driven (Table 1): they are deployed to observe, analyze and understand the physical world. A data handling architecture must, therefore, reconcile conflicting requirements:

- A fully centralized data collection strategy is infeasible given

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 2002 ACM X-XXXXX-XX-X/XX/XX ...\$5.00.

the energy constraints on sensor node communication, and inefficient given that sensor data has significant redundancy.

- Many queries over these systems will be spatio-temporal in nature. The storage system should support efficient spatio-temporal querying and mining for events of interest. Such events exist at specific spatio-temporal scales, and therefore in order to extract information from data, one has to perform interpretation over a certain region. Local processing alone is not sufficient. For example, identifying pooling of contaminants in a region.
- Users will routinely require compressed summaries of large spatio-temporal sensor data. However, periodically or occasionally, users will require detailed datasets from a subset of sensors in the network.

In addressing the storage challenges of sensor networks, one question immediately comes to mind: *can we use existing distributed storage systems for our purpose?*. We argue that there are fundamental differences in the cost models, nature of the data, and intended forms of use of sensor networks, that motivate new approaches and metrics of evaluation.

- Hierarchical web caches [4] are designed to lower latency, network traffic and load. The cost models that drive their caching strategy is based on user web access patterns, strategically placing web pages that are frequently accessed. Peer-to-peer systems are designed for efficient lookup of files in a massively distributed database. These systems do not capture key challenges of sensor networks: (a) they are designed for a much less resource constrained infrastructure, unlike in sensor networks, where communication of every bit should be accounted for (b) they are optimized for bandwidth which, while limited, is a non-depletable resource, unlike energy on sensor nodes (c) the atomic unit of storage is a file, and unlike sensor data, different files are not expected to exhibit significant spatio-temporal correlations.
- Geographic Information Systems (GIS) deal with data that exhibit spatial correlations, but the processing is centralized, and algorithms are driven by the need to reduce search cost, typically by optimizing disk access latency.
- Streaming media in the internet uses a centralized approaches to compression of spatio-temporal streams such as MPEG-2, and are optimized for different cost functions. Consider the problem of compressing a 3 dimensional datacube (dimensions: $x,y,time$) corresponding to data from a single sensor type on a grid of nodes on a plane, much like a movie of

sensor data. MPEG-2 compresses first along the spatial axes (x, y), and uses motion vectors to compress along the temporal axis. The cost model driving such an approach is perceptual distortion and transmission latency. Communication constraints in sensor networks drive a time first, space next approach to compressing the datacube, since temporal compression is local and far cheaper than spatial compression.

- Wavelets [5, 6] are a popular signal processing technique for lossy compression. In a centralized setting, compression using wavelets can use entropy-based metrics to tradeoff compression benefit with reconstruction error. In a sensor network setting, pieces of the data are distributed among nodes in the network, and communication constraints force local cost metrics that tradeoff the **communication overhead** with the compression benefit.

Thus, large scale, untethered devices sensing the physical world call for building systems that incorporate their extreme resource constraints and spatio-temporal interpretation of the physical world in the design, cost model, and metrics of evaluation of a data handling architecture. DIMENSIONS constrains traditional distributed systems design with the need to make every bit of communication count, incorporates spatio-temporal data reduction to distributed storage architectures, introduces local cost functions to data compression techniques, and adds distributed decision making and communication cost to data mining paradigms. It provides unified view of data handling in sensor networks incorporating long-term storage, multi-resolution data access and spatio-temporal pattern mining.

2. DESIGN GOALS

The following design goals allow DIMENSIONS to minimize the bits communicated:

Multi-Resolution Data Storage: A fundamental design goal of DIMENSIONS is the ability to extract sensor data in a multi-resolution manner from a sensor network. Such a framework offers multiple benefits (a) it allows users to look at low-resolution data from a larger region cheaply, before deciding to obtain more detailed and potentially more expensive datasets (b) Compressed low-resolution sensor data from large number of nodes can often be sufficient for spatio-temporal querying to obtain statistical estimates of a large body of data [7].

Distributed: Design goals of distributed storage systems such as [8, 9] of designing scalable, load-balanced, and robust systems, are especially important for resource constrained distributed sensor networks. We have as a goal that the system balances communication and computation load of querying and multi-resolution data extraction from the network. In addition, it should leverage distributed storage resources to provide a long-term data storage capability. Robustness is critical given individual vulnerability of sensor nodes. Our system shares design goals of sensor network protocols that compensate for vulnerability by exploiting redundancy in communication and sensing.

Adapting to Correlations in sensor data: Correlations in sensor data can be expected along multiple axes: temporal, spatial and between multiple sensor modalities. These correlations can be exploited to reduce dimensionality. While temporal correlation can be exploited locally, the routing structure needs to be tailored to spatial correlation between sensor nodes for maximum data reduction. The correlation structure in data will vary over time, depending on the changing characteristics of the sensed field. For example, the correlation in acoustic signals depend on the source location and orientation, which can be time-varying for a mobile source. The

storage structure should be able to adapt to the correlation in sensor data.

3. APPROACH

The key components of our design are (a) temporal filtering (b) *wavRoute*, our routing protocol for spatial wavelet subband decomposition (c) distributed long-term storage through adaptive wavelet thresholding. We outline key features of the system, and discuss cost metrics that tradeoff communication, computation, storage complexity, and system performance. A few usage models of the storage system are described, including multi-resolution data extraction, spatio-temporal data mining, and feature routing. To facilitate the description, we use a simplified grid topology model, whose parameters are defined in Table 2. We expect to relax this assumption in future work (Section 8).

The approach to DIMENSIONS is based on wavelet subband coding, a popular signal processing technique for multiresolution analysis and compression [5, 6]. Wavelets offer numerous advantages over other signal processing techniques for viewing a spatio-temporal dataset (a) ability to view the data at multiple spatial and temporal scales (b) ability to extract important features in the data such as abrupt changes at various scales thereby obtaining good compression (c) easy distributed implementation and (d) low computation and memory overhead. The crucial observation behind wavelet thresholding when applied to compression is that for typical time-series signals, a few coefficients suffice for reasonably accurate signal reconstruction.

n	Number of Nodes in the network
R	Region participating in the wavelet decomposition. Simplified model assumes grid placement of $\sqrt{n_x} \times \sqrt{n_y}$ nodes
λ	Number of levels in the spatial wavelet decomposition. $\sqrt{n} = 2^\lambda$ and $\lambda = \frac{1}{2} \log n$
(X_{apex}, Y_{apex})	Location of the apex of the decomposition pyramid
D_0	Time-series data at each node, before the spatial decomposition
Huffman	Entropy encoding scheme used

Table 2: Parameters

3.1 Temporal decomposition

Temporal data reduction is cheap since it involves only computation at a single sensor, and incurs no communication overhead. The first step towards constructing a multi-resolution hierarchy, therefore, consists of each node reducing the time-series data as much as possible by exploiting temporal redundancy in the signal and a priori knowledge about signal characteristics. By reducing the temporal datastream to include only potentially interesting events, the communication overhead of spatial decomposition is reduced.

Consider the example of a multi-resolution hierarchy for building health monitoring (Table 1). Such a hierarchy is constructed to enable querying and data extraction of time-series signals corresponding to interesting vibration events. The local signal processing involves two steps: (a) Each node performs simple real-time filtering to extract time-series that may represent interesting events. The filtering could be a simple amplitude thresholding i.e. events that cross a pre-determined SNR threshold. The thresholding yields short time-series sequences of building vibrations. (b) These time-series snippets are compressed using wavelet subband coding to yield a sequence that capture as much energy of the signal as possible, given communication, computation or error constraints.

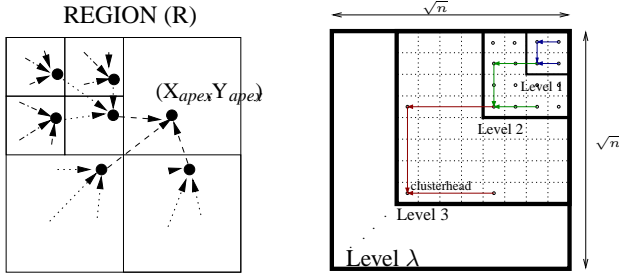


Fig 1(a) *wavRoute* hierarchy

Fig 1(b) Analytical Approximation

Figure 1: Grid Topology Model

Algorithm 1: ClusterWaveletTransform

Data : R : co-ordinates of region; λ : number of decomposition levels; t : current time
Result : Spatial wavelet decomposition of λ levels
 /* Location of apex of the pyramid */;
 $(X_{apex}, Y_{apex}) = \text{HashInRegion}(R, t)$;
 $l = 1$ /* initialize level */;
while $l \leq \lambda$ **do**
 $(X_l, Y_l) = \text{LocateClusterHead}(l, (X_{apex}, Y_{apex}))$;
 if *I am clusterhead for level $l - 1$* **then**
 GeoRoute Coefficients to (X_l, Y_l) ;
 current section becomes this one;
 if *I am clusterhead for level l* **then**
 Get coefficients from clusterheads at level $l - 1$;
 Perform a 2 dimensional Wavelet transform;
 Store Coefficients and Deltas Locally. Save Coefficients for next iteration;

3.2 wavRoute: A Routing Protocol for Spatial Wavelet Decomposition

Spatial data reduction involves applying a multi-level 2D wavelet transform on the coefficients obtained from 1D temporal data reduction described in Section 3.1. Our goals in designing this routing protocol are twofold: (a) minimize the communication overhead of performing a spatial wavelet decomposition (b) balance the communication, computation and storage load among nodes in the network. *wavRoute* uses a recursive grid decomposition of a physical space into tiles (such as the one proposed in [10]), in conjunction with geographic routing [11, 12] as shown in Algorithm 1. At each level of the decomposition, data from four tiles at the previous level are merged at a chosen node, which stores it in its local store. The merged streams are further subband coded, thresholded, and quantized to fit within specified optimization criteria (discussed in Section 3.4). The compressed stream is sent to the next higher level of the hierarchy as shown in Figure 2. The algorithm is executed recursively (Figure 1(a)): all nodes in the network participate in Step 1, in following steps, only clusterheads from the previous step participate. In the following sections, we elaborate on the algorithm used to select clusterhead location and the geographic forwarding protocol used.

Algorithm 2: LocateClusterHead

Data : l : current level of decomposition; (X_{apex}, Y_{apex}) : location of apex of Pyramid
Result : (X_{ch}, Y_{ch}) : Co-ordinates of clusterhead location for level i tile
 $X_{0\text{ tile}} = \lfloor \frac{X_{m_y}}{2^l} \rfloor$; $X_{1\text{ tile}} = X_{0\text{ tile}} + 2^l$;
 $Y_{0\text{ tile}} = \lfloor \frac{Y_{m_y}}{2^l} \rfloor$; $Y_{1\text{ tile}} = Y_{0\text{ tile}} + 2^l$;
 Compute Centroid of Tile;
 Compute Euclidean Shortest Path Vector L from Centroid to Storage Node;
 $(X_{sn}, Y_{sn}) = \text{Intersection Between Path Vector } L \text{ and Tile boundary}$;

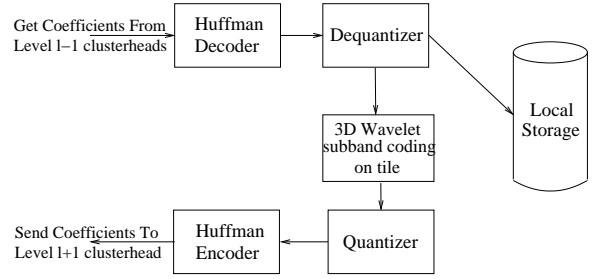


Figure 2: Protocol at clusterhead at depth i

Clusterhead Selection: To minimize communication cost, the choice of clusterhead should be tied into the routing structure. We use a simple clusterhead selection procedure that gives us good complexity properties (described in Section 5). First, the apex of the decomposition pyramid is first chosen by hashing into the geographic region, R (Figure 1(a)). Then, for each tile, the euclidean shortest path between the centroid of the tile and the apex location $((X_{apex}, Y_{apex}))$ is computed. The point where this path intersects the tile boundary is chosen to be the location of the clusterhead. Since each node can independently calculate the location of the tile based on its own geographic co-ordinates, the clusterhead location can be independently computed by each node.

Modified GPSR: We use a modified GPSR approach proposed in [11] to route packets to clusterheads. A brief review of the approach is described below, details can be obtained from [12] and [11]. GPSR is a *strongly geographic* routing protocol that takes a location rather than an address to deliver packets. [11] propose a modified GPSR protocol that ensures that packets are delivered to the node *closest* to the destination location.

3.3 Long-term Storage

Long term storage is provided in DIMENSIONS by exploiting the fact that thresholded wavelet coefficients lend themselves to good compression benefit [5, 6]. Our rationale in balancing the need to retain detailed datasets for multi-resolution data collection and to provide long-term storage is that if scientists were interested in detailed datasets, they would extract it within a reasonable interval (weeks). Long-term storage is primarily to enable spatio-temporal pattern mining, for which it is sufficient to store key features of data. Thus, the wavelet compression threshold is aged progressively as shown in Figure 3, lending older data to progressively better compression, but retaining key features of data.

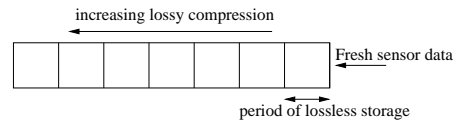


Figure 3: Long term storage

3.4 Choosing an Appropriate Cost Metric

An appropriate cost metric must weight multiple parameters: (a) communication overhead (b) query performance (c) error or signal distortion from lossy compression (d) computation overhead (e) storage cost. Combinations of the above parameters can be tuned to specific sensor network deployments.

We have explored configurations that compress based primarily on either communication cost or error. Bounding the communication cost at each level of the hierarchy ensures that clusterheads do

not incur significantly higher communication overhead than other nodes in the network. The compression error can, however, vary depending on the spatio-temporal correlation in the data. For example, an area with high correlation may be described with low error within the specified communication bound, but a region with low correlation would incur greater error. Conversely, bounding the compression error would result in the compression adapting to the correlation in different regions, but unbounded communication overhead if the data is highly uncorrelated. We are exploring joint optimization criteria that include both these parameters.

In a different scenario, a sensor network might consist of low-end, highly resource constrained devices (such as motes [13]), where computation, memory and storage are highly constrained, in addition to energy. In these networks, the hierarchy can be constructed by propagating approximation coefficients of the wavelet transform, which is computationally less expensive than subband coding.

3.5 Load Balancing and Robustness

Due to space constraints, we will introduce only a flavor of techniques that we are exploring in this area. As discussed above, bounding the communication overhead at each level of the decomposition can reduce the uneven distribution of energy consumption. Additional load-balancing can be provided by periodically hashing to a new apex location, implicitly changing the choice of clusterheads at different levels.

A strictly hierarchical configuration (as described above) is vulnerable to node failures, since loss of a node cuts off data from any of its children. We are therefore exploring decentralized, peer-based structures. One approach we are considering communicates summarized data to all members of the next higher level, rather than to just the clusterhead. Such an approach will have higher overall energy consumption and will require greater data compression, but it becomes immune to node failure and is naturally load balanced since data is replicated equally to all nodes.

4. USAGE MODELS

A distributed multi-resolution storage infrastructure benefits search and retrieval of datasets that exhibit spatial correlation, and applications that use such data.

Multi-Resolution Data Extraction: Multi-resolution data extraction can proceed along the hierarchical organization by first extracting and analyzing low-resolution data from higher level clusterheads. This analysis can be used to obtain high-resolution data from sub-regions of the network if necessary.

Querying for Spatio-Temporal features: The hierarchical organization of data can be used to search for spatio-temporal patterns efficiently by reducing the search space. For example, consider the search for pooling of contaminant flow. Such a feature has a large spatial span, and therefore significant energy benefit can be obtained by querying only a few clusterheads rather than the entire network. Temporal patterns can be efficiently queried by drilling down the wavelet hierarchy by eliminating branches whose wavelet coefficients do not partially match the pattern, thereby reducing the number of nodes queried. Summarized coefficients that result from wavelet decomposition have been found to be excellent for approximate querying [14, 7], and to obtain statistical estimates from large bodies of data (Section 6).

Feature Routing and Edge Detection: Target tracking and routing along spatio-temporal patterns such as temperature contours, have been identified as compelling sensor network applications. The problem of edge detection has similar requirements, and is important for applications like geographic routing, localization,

beacon placement and others, where explicit knowledge of edges can improve performance of the algorithm. Our architecture can be used to assist these applications, since it good at identifying discontinuities. By progressively querying for the specific features, communication overhead of searching for features can be restricted to only a few nodes in the network.

Debugging: Debugging data is another class of datasets that exhibits high correlation. Consider packet throughput data: throughput from a specific transmitter to two receivers that are spatially proximate are closely correlated; similarly, throughput from two proximate transmitters to a specific receiver are closely correlated. Our system serves two purposes for these datasets: (a) they can be used to extract aggregate statistics (Section 6) with low communication cost, and (b) discontinuities represent network hotspots, deep fades or effects of interference, which are important protocol parameters, and can be easily queried.

5. COMMUNICATION, COMPUTATION AND STORAGE COMPLEXITY

In this section, we provide a back-of-the-envelope comparison of the benefits of performing a hierarchical wavelet decomposition over a centralized data collection strategy. Our description will only address the average and worst case communication load on the nodes in the network, while Table 3 provides a more complete summary of the communication, computation and storage complexity.

A square grid topology with n nodes is considered (\sqrt{n} side as shown in Figure 1(b)), where $\sqrt{n} = 2^\lambda$. Clusterheads at each step are selected to be the one closest to the lower left-corner of the tile. Communication is only along edges of the grid, each of which is of length 1 unit. The cost of transmission and reception of one unit of data along a link of length 1, costs 1 unit of energy. The chosen cost metric is communication bandwidth, and each clusterhead is constrained transmit at most D_0 data. While realistic topologies are far from as regular as the one proposed [15], and the cost model can be more complicated, the simple case captures essential tradeoffs in construction of multi-resolution hierarchies.

Communication Overhead: The overhead for *centralized* decomposition can be computed from the total number of edges traversed on the grid ($n(\sqrt{n} - 1)$ for a $\sqrt{n} \times \sqrt{n}$ grid) The total cost, therefore, is $n(\sqrt{n} - 1)D_0$, giving an *Average-case* communication complexity of $O(\sqrt{n}D_0)$. The worst communication overhead is the storage node itself, or the sensor node(s) closest to the storage node, if the storage node is not power constrained. Thus, the *Worst-case* communication complexity is $O(n^{1.5}D_0)$.

To compute the overhead of the *hierarchical* decomposition, we use the fact that at level l of the decomposition, a communication edge is of length 2^{l-1} , and there are $2^{2\lambda-2l}$ tiles (computed as $\frac{\text{Area of Region}}{\text{Area of Tile}}$), giving a communication overhead of $2^{2\lambda+1-l}$.

Over λ levels of decomposition, the total cost is, thus, $\sum_{0 \leq l \leq \lambda} 2^{2\lambda+1-l} = 2^{\lambda+1}(2^\lambda - 1)$. Thus, the total communication cost is $O(nD_0)$ and the *Average-case* communication cost is $O(D_0)$. In the worst case, a node is on the forwarding path at every level of the hierarchy. Since there are λ levels and each level forwards data $3D_0$ to a clusterhead, worst case data passing through a node is $3\lambda D_0 = 3D_0 \log n$. Thus, *Worst-case* Communication Complexity = $O(D_0 \log n)$.

6. SAMPLE DATA ANALYSIS

Our system is currently under development, but following are some initial results from offline analysis of experimental data. We provide initial results from two different sensor datasets to demon-

	Centralized		Hierarchical	
	Avg. Case	Worst Case	Avg. Case	Worst Case
Communication	$O(\sqrt{n}D_0)$	$O(n^{1.5}D_0)$	$O(D_0)$	$O(D_0 \log n)$
Computation	$O(D_0 \log D_0)$	$O(nD_0)$	$O(D_0 \log D_0)$	$O(D_0 \log(nD_0))$
Storage	$O(D_0)$	$O(nD_0)$	$O(D_0)$	$O(D_0 \log n)$

Table 3: Communication, Computation, and Storage Overhead

strate the practical applicability of our system.

6.1 Precipitation Dataset

The first sensor dataset that we consider is a geospatial dataset that provides 50 km resolution daily precipitation for the Pacific NorthWest from 1949-1994 [16]. While the dataset is at a significantly larger scale than densely deployed sensor networks, the data exhibits spatio-temporal correlations, providing a useful performance case study.

The setup comprises a 15x12 grid of nodes, each recording daily precipitation values. A wide set of queries can be envisaged on such a dataset: (a) range-sum queries, such as total precipitation over a specified period from a single node or a region; (b) drill-down max queries, to efficiently access the node or region that receives maximum precipitation over a specified period; (c) drill-down edge queries, to efficiently access and query nodes on an edge between a high-precipitation and low-precipitation region.

The hierarchy construction proceeds as outlined in Section 3, with nodes at level 0 performing a one-dimensional temporal wavelet subband decomposition, and clusterheads at other levels combining data from the previous level, and performing a three-dimensional decomposition. We use communication bandwidth as the cost metric for the compression. Thus, at level i , data from four level $i - 1$ nodes are combined, subband coded, thresholded, quantized, and losslessly compressed using Run-Length encoding and Huffman encoding to fit within the specified target bandwidth. An approximate target communication bandwidth is chosen since it is computationally intensive to choose parameters to exactly fit the target bandwidth.

In the following example, we look at the performance of the hierarchy for a specific range-sum query: “Find the annual precipitation between 1949-1994 for each sensor node”. We use two performance metrics:

- *Compression Ratio* at level i is ratio of number of bits of transmitted data to the raw data size. Table 4 shows that the number of bits transmitted is approximately the same at each level of the hierarchy, giving us large compression ratios, especially at higher levels since more data is combined into the target bandwidth.
- *Query error* at level i is calculated as $\frac{|measured - true|}{true}$. This metric corresponds to the accuracy in the response for the range-sum query when clusterheads at level i are queried. While the error increases at higher levels of the hierarchy, it is still reasonably low at all layers. For example, 80% of measurements are within 30% of the true answer for level 1, and 80% of measurements are within 50% at level 3. We suggest that that such error is sufficient to make preliminary searches and then, if desired, drill-down with more detailed queries.

6.2 PacketLoss Dataset

We now look at the performance of our system on a packet throughput dataset from a 12x14 grid of nodes with grid spacing 2 feet

Level	Raw data size (Kbits)	Mean data sent to next level (Kbits)	Compression Ratio (Ratio of raw data size to transmitted data size)
1	262.5	5.6	46.8
2	984.4	3.8	257.2
3	3937.7	4.0	987
4	11813.2	5.2	2286.2

Table 4: Compression Result for Sum Query

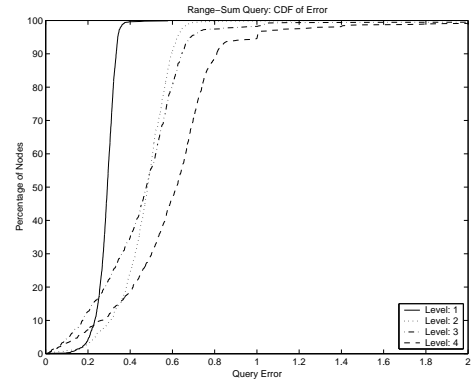


Figure 4: Query Error for Range-Sum Query

(detailed descriptions can be obtained from [15]). Each node has throughput data from all other transmitters, for twenty different settings of transmit signal strength from each node. Two forms of correlations can be exploited to reduce size of data: (a) correlation in throughput between adjacent transmitters to a particular receiver, and (b) correlation in throughput from a single transmitter to adjacent receivers.

Figure 5 shows the results of a query to obtain the throughput vs distance map from the compressed data. The error is small in the approximated data, and gives us large compression benefit. These results show that our algorithm works for a broad class of data with different correlation characteristics.

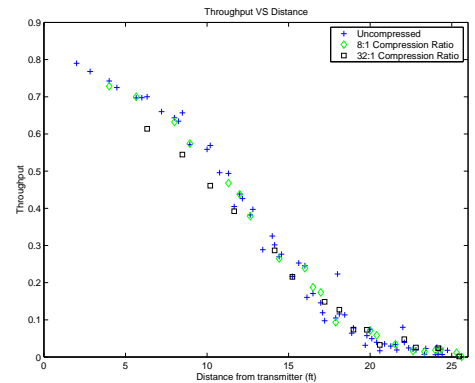


Figure 5: Throughput VS Distance

7. RELATED WORK

In building this system, we draw from significant research in information theory [5, 6, 17], databases [7, 14], spatio-temporal data mining [18, 19], and previous work in sensor networks [11, 20]. [14] and others have used wavelet-based synopsis data structures for approximate querying of massive data sets. Quadrees are popularly used in image processing and databases [19], and resemble the hierarchies that we build up. Other data structures such as R-trees, kd-trees and their variants [18] are used for spatial feature indexing. Triangular Irregular Networks (TIN [18]) is used in for multi-layered processing in cartography, and other geographical datasets. Some of these structures could find applicability in our context, and will be considered in future work. [11] proposes a sensor network storage architecture that leverage the excellent lookup complexity of distributed hash tables (DHT) for event storage. DHTs are useful when queries are not spatio-temporal in nature, while our system organizes data spatially to enable such queries.

8. SUMMARY AND RESEARCH AGENDA

This paper made the case for DIMENSIONS, a large-scale distributed multi-resolution storage system that exploits spatio-temporal correlations in sensor data. Many important research issues still need to be addressed:

What are the benefits from temporal and spatial data compression? To fully understand correlations in sensor data, and the benefits provided by temporal and spatial compression, we are deploying large scale measurement infrastructure, for sensor data collection in realistic settings,

What processing should be placed at different levels of the hierarchy? The clusterheads at various levels of the hierarchy can interpret the data at different scales, to obtain information about spatio-temporal patterns. A challenging problem is the development of algorithms for spatio-temporal interpretation of data.

How can we increasing compression benefit without sacrificing query performance? Our system can be augmented with other compression techniques such as delta coding, and recent work on blind source coding [17], to obtain better data compression. Correlation statistics can be learnt by clusterheads, and exploited to reduce data at the source.

How can we obtain energy savings from distributed compression? Better compression doesn't necessarily translate into energy savings in communication since the cost of passive listening is comparable to transmission [1]. Obtaining energy savings in communication of data involves (a) reducing size of data to be transmitted (b) scheduling communication to minimize listen time as well as transmit time. How do we schedule communication to translate compression benefit to energy benefit?

How can this research be applied to other networked systems? These techniques may have relevance to other systems where data is correlated, and massive distributed datasets need to be queried, such as scientific applications of Grid Computing [21].

Acknowledgements

We would like to thank Stefano Soatto for very useful feedback on theoretical aspects of our system. We would also like to thank members of the LECS lab for providing feedback on the paper.

9. REFERENCES

- [1] G.J. Pottie and W.J. Kaiser. Wireless integrated network sensors. *Communications of the ACM*, 43(5):51–58, May 2000.
- [2] M. Kohler. UCLA factor building. CENS Presentation.
- [3] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao. Habitat monitoring: Application driver for wireless communications technology. In *Proceedings of the 2001 ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, April 2001.
- [4] A. Chankhunthod, P.B. Danzig, C. Neerdaels, M.F. Schwartz, and K.J. Worrell. A hierarchical internet object cache. In *USENIX Annual Technical Conference*, pages 153–164, 1996.
- [5] R. M. Rao and A. S. Bopardikar. *Wavelet Transforms: Introduction to Theory and Applications*. Addison Wesley Publications, 1998.
- [6] M. Vetterli and J. Kovacevic. *Wavelets and Subband coding*. Prentice Hall, New Jersey, 1995.
- [7] J. S. Vitter, M. Wang, and B. Iyer. Data cube approximation and histograms via wavelets.. In D. Lomet, editor, *Proceedings of Seventh International Conference on Information and Knowledge Management (CIKM'98)*, pages 69–84, Washington D.C, November 1998.
- [8] J. Kubiatowicz, D. Bindel, Y. Chen, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. Oceanstore: An architecture for global-scale persistent storage. In *Proceedings of ACM ASPLOS*. ACM, November 2000.
- [9] A. Rowston and P. Druschel. Storage management and caching in past, a large-scale, persistent peer-to-peer storage utility. In *18th ACM SOSP*, volume 1, Lake Louise, Canada, October 2001.
- [10] I. Gupta, R. van Renesse, and K. Birman. Scalable fault-tolerant aggregation in large process groups, 2001.
- [11] S. Ratnasamy, D. Estrin, R. Govindan, B. Karp, L. Yin S. Shenker, and F. Yu. Data-centric storage in sensornets. In *ACM First Workshop on Hot Topics in Networks*, 2001.
- [12] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *Proceedings of Mobicom*, 2000.
- [13] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister. System architecture directions for networked sensors. In *Proceedings of ASPLOS-IX*, pages 93–104, Cambridge, MA, USA, November 2000. ACM.
- [14] K. Chakrabarti, M. Garofalakis, R. Rastogi, and K. Shim. Approximate query processing using wavelets. *VLDB Journal: Very Large Data Bases*, 10(2–3):199–223, 2001.
- [15] D. Ganesan, B. Krishnamachari, A. Woo, D. Culler, D. Estrin, and S. Wicker. Complex behavior at scale: An experimental study of low-power wireless sensor networks. Technical Report CSD-TR 02-0013, UCLA, 2002.
- [16] M. Widmann and C. Bretherton. 50 km resolution daily precipitation for the Pacific Northwest, 1949-94.
- [17] S. Pradhan, J. Kusuma, and K. Ramchandran. Distributed compression in a dense sensor network. *IEEE Signal Processing Magazine*, 1, March 2002.
- [18] M. Kreveld, J. Nievergelt, T. Roos, and P. Widmayer. *Algorithmic Foundations of Geographic Information Systems*. Springer-Verlag, 1997.
- [19] R. Agrawal, C. Faloutsos, and A.N. Swami. Efficient Similarity Search In Sequence Databases. In D. Lomet, editor, *Proceedings of the 4th International Conference of Foundations of Data Organization and Algorithms (FODO)*, pages 69–84, Chicago, Illinois, 1993. Springer Verlag.
- [20] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of Mobicom*, pages 56–67, Boston, MA, August 2000. ACM Press.
- [21] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. Grid services for distributed system integration. *Computer*, 35(6), 2002.