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

Deepak Ganesan, Deborah Estrin

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 obtain detailed datasets for analysis. 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

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 obtain detailed datasets for analysis. The data handling requirements from these systems will be massive, and disproportionate to the stringent resource constraints on sensor nodes ([4]). 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 ([3])	response to earthquakes, strong winds
Contaminant Flow	Concentration pattern, pooling of contaminants, plume tracking
Habitat microclimate monitoring	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 the energy constraints on sensor nodes, and impractical 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, to identify pooling of contaminants, spatio-temporal interpretation is done over data from 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 ([20]) 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 two 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) the atomic unit of storage is a file, and unlike sensor data, files do not exhibit 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.
- Centralized approaches to compression of spatio-temporal streams such as MPEG-2, are optimized for dif-

ferent 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, which is important in transmission of a movie. 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 ([1, 2]) 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 [10].

Distributed: Design goals of distributed storage systems such as [11, 12] 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 decomposition (c) distributed long-term storage through adaptive wavelet thresholding. We describe a few usage models of the storage system, 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.

The approach to DIMENSIONS is based on wavelet thresholding, a popular signal processing technique for multiresolution analysis and compression [1, 2]. 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$\sqrt{n}$ nodes
λ	Number of levels in the spatial wavelet decomposition. $\sqrt{n} = 2^{\lambda}$ and $\lambda = \frac{1}{2}logn$
(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

Given that communication is the most expensive operation in terms of energy, local data reduction is performed as the first step. By reducing the temporal datastream to include only potentially interesting events, and compressing the time-series, the overhead of spatial decomposition is reduced. The local signal processing involves two steps: (a) Each node performs simple real-time filtering to extract time-series that may represent interesting events (b) These time-series snippets are compressed using wavelet thresholding to yield a set of coefficients that capture most of the energy of the signal. In the example of building health monitoring (Table 1), the filtering is a simple amplitude thresholding i.e. events that crosses a pre-determined SNR threshold. The thresholding yields short time-series sequences of building vibrations.

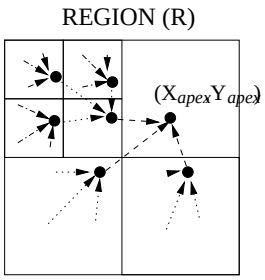

Fig 1(a) $wavRoute$ hierarchy

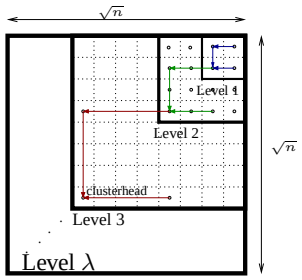

Fig 1(b) Analytical Approximation

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}) = HashInRegion(R,t);
l = 1 /* initialize level */;
while $l \leq \lambda$ **do**
  (X_l, Y_l) = 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 [17]), in conjunction with geographic routing ([15, 16]) as shown in Algorithm 1. At each level of the decomposition, coefficients from each tile are compressed using 2D wavelet transform at a selected clusterhead, which locally stores the coefficients and details from the decomposition, and forwards the resulting coefficients to the clusterhead at the next level of decomposition (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 the select clusterhead location and the geographic forwarding protocol used.

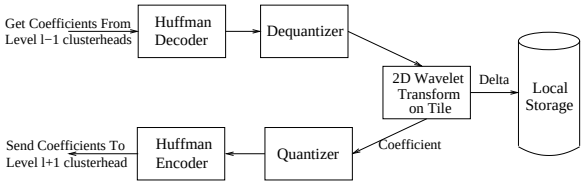

Figure 2: Protocol at clusterhead at depth i

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
$X_{0\ tile} = \lfloor \frac{X_{my}}{2^l} \rfloor$; $X_{1\ tile} = X_{0\ tile} + 2^l$;
$Y_{0\ tile} = \lfloor \frac{Y_{my}}{2^l} \rfloor$; $Y_{1\ tile} = Y_{0\ tile} + 2^l$;
Compute Centroid of Tile;
Compute Euclidean Shortest Path Vector L from Centroid to Storage Node;
(X_{sn}, Y_{sn}) = Intersection Between Path Vector L and Tile boundary;

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 4). 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 inde-

pendently computed by each node.

Modified GPSR: We use a modified GPSR approach proposed in [15] to route packets to clusterheads. A brief review of the approach is described below, details can be obtained from [16] and [15]. GPSR is a *strongly geographic* routing protocol that takes a location rather than an address to deliver packets. [15] propose a modified GPSR protocol that ensures that packets are delivered to the node *closest* to the destination location. When no node is located at the given location, GPSR finds the packet along the 'perimeter' of a planar subgraph of the original graph, using the right hand rule. Thus, GPSR uses a combination of greedy, and if required, perimeter forwarding, to reach the node closest to the destination.

Balancing Load: The combination of small tile size and wavelet decomposition at each clusterhead in *wavRoute* reduces the penalty incurred by being a clusterhead greatly. Section 4 shows that under certain assumptions of topology and routing, the worst case overhead of storage and communication at any clusterhead is within $log(n)$ (n = network size) times the overhead at any other node, the case occurring when the same node is chosen as a clusterhead at every level. To further balance this overhead, *wavRoute* periodically changes the routing structure, by hashing to different apex locations. Changing the apex location implicitly changes the clusterheads selected at each level of the routing hierarchy, and balances the load among nodes in the network.

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 ([1, 2]). 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, lending older data to progressively better compression, but retaining key features of data.

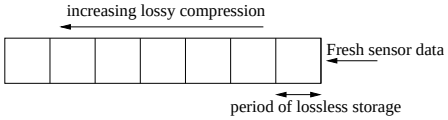

Figure 3: Long term storage

3.4 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: Data from a specified region and time-frame can be extracted in a multi-resolution manner by specifying a threshold for the detail coefficients. The query is flooded to the specified region using a protocol such as diffusion ([9]) and geo-routing. Nodes apply the specified threshold in to the stored details, and send the compressed sparse vector back to the querying node.

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 in a drill-down manner on 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 ([6, 10]), and to obtain statistical estimates from large bodies of data (Section 5).

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: Other datasets besides sensor data exhibit spatial correlations. One such example is 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 5) with low communication cost (b) discontinuities represent network hotspots, deep fades or effects of interference, which are important protocol parameters, and can be easily queried.

4 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 and storage load on the nodes in the network, while Table 3 provides a more complete summary of the average and worst-case 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. Each node has initially D_0 data to send. While realistic topologies are far from as regular as the one proposed ([8]), and the cost model is 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 $\sqrt{n}$x$\sqrt{n}$ 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 decomposing a 2x2 tile in which each node has data D_0, results in D_0 of coefficients, that are both sent to the next level clusterhead, and stored locally, and $3D_0$ of deltas that are only stored locally. The amount of data generated and stored at a clusterhead at any level of the decomposition, is therefore, the same. In reality, quantization and huffman encoding would change these values depending on the nature of the dataset, yet, the raw size of the data is a reasonable indicator of the communication overhead.

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 $\frac{Area\ of\ Region}{Area\ of\ Tile}$), 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 $3\lambda D_0 = 3D_0 logn$.

Worst-case Communication Complexity = $O(D_0 logn)$.

5 Sample Data Analysis

Our system is currently under development. The following are some initial results from offline analysis of experimental data. The dataset that we consider is a packet throughput dataset from a 12x14 grid of nodes (detailed descriptions can be obtained from [8]). Each node has throughput data from all other transmitters, for twenty different settings of transmit signal strength from each node. Figure 4 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.

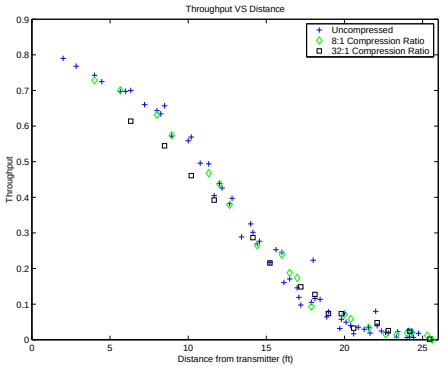

Figure 4: Approximate Queries on Compressed coefficients

6 Related Work

In building this system, we draw from significant research in information theory [1, 2, 19], databases [10, 6], spatio-temporal data mining [21, 5], and previous work in sensor networks [15, 9]. [6] and others have used wavelet-based synopsis data structures for approximate querying of massive data sets. Quadtrees are popularly used in image processing and databases ([5]), and resemble the hierarchies that we build up. Other data structures such as R-trees, kd-trees and their variants ([21]) are used for spatial feature indexing. Triangular Irregular Networks (TIN [21]) 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. [15] 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.

	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

7 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 learn correlations? Recent work in information theory ([19]) proposes practical codes to exploit the theoretical benefits of blind source coding known as the Slepian Wolf theorem. They suggest that joint correlation statistics can be used to compress data at the source, without requiring any form of communication. The caveat is the joint correlation statistics are considered known and are stable over periods of time. Clusterheads in our architecture are in a position to study the correlation statistics of wavelet coefficients over time, and incorporate these statistics to enable further compression benefit. How stable are these statistics over time?

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 ([4]). 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?

Acknowledgements We would like to thank Stefano Soatto for useful feedback and taking the time to provide suggestions on the paper.

References

[1] R. M. Rao and A. S. Bopardikar, "Wavelet Transforms: Introduction to Theory and Applications", Addison Wesley Publications

[2] M. Vetterli and J. Kovacevic, "Wavelets and Subband coding", Prentice Hall, 1995

[3] UCLA Factor Building, *http://www.ess.ucla.edu/ kohler/CENS/CENS_Seismic_Slides.ppt*

[4] G.Pottie, W. Kaiser: "Wireless Integrated Network Sensors (WINS): Principles and Approach." Communications of the ACM. Vol 43, Number 5. May 2000.

[5] Rakesh Agrawal, Christos Faloutsos and Arun Swami, "Efficient Similarity Search In Sequence" Databases FODO conference, Evanston, Illinois, Oct. 13-15, 1993

[6] K. Chakrabarti, M. Garofalakis, R. Rastogi, and K. Shim, "Approximate Query Processing Using Wavelets" VLDB 2000, Cairo, Egypt, September 2000, pp. 111-122.

[7] M. Morehart, F. Murtagh, and J. Starck, "Multiresolution spatial analysis", GeoComputation 1999

[8] D. Ganesan, A. Woo, B. Krishnamachari, D. Culler, D. Estrin, S. Wicker, "An Empirical Study of Epidemic Algorithms in Large Scale Multihop Wireless Networks". Under submission.

[9] C. Intanagonwiwat, R. Govindan and D. Estrin, "Directed Diffusion: A Scalable and Robust Communication Paradigm for Sensor Networks," MobiCom 2000, August 2000, Boston, Massachusetts

[10] J. S. Vitter, M. Wang, and B. Iyer. "Data Cube Approximation and Histograms via Wavelets," Proceedings of Seventh International Conference on Information and Knowledge Management (CIKM'98), Washington D.C., November 1998.

[11] J. Kubiatowicz et al, "OceanStore: An Architecture for Global-Scale Persistent Storage" , ASPLOS, November 2000.

[12] A. Rowstron and P. Druschel, "Storage management and caching in PAST, a large-scale, persistent peer-to-peer storage utility", SOSP, Banff, Canada, October 2001.

[13] I. Stoica, R. Morris, D. Karger, F. Kaashoek, H. Balakrishnan, Chord: A scalable peer-to-peer lookup service for Internet applications, SIGCOMM 2001

[14] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Schenker, Scalable Content-Addressable Network, SIGCOMM 2001

[15] S. Ratnasamy, D. Estrin, R. Govindan, B. Karp, S. Shenker, L. Yin and F. Yu, Data-centric storage in Sensornets, Submitted for review. February 1st, 2002.

[16] Karp, B. and Kung, H.T., Greedy Perimeter Stateless Routing for Wireless Networks, MobiCom 2000, Boston, MA, August, 2000, pp. 243-254.

[17] I. Gupta, R.V Renesse and K.P. Birman, "Scalable Fault-tolerant Aggregation in Large Process Groups", In The International Conference on Dependable Systems and Networks (DSN 01), Goteborg, Sweden, July, 2001.

[18] JPEG2000 Final Committee Draft. http://www.jpeg2000.org

[19] J. Kusuma, L. Doherty and K. Ramchandran, Distributed source coding for sensor networks, Proc. IEEE Conf. on Image Processing (ICIP), October 2001

[20] A. Chankhunthod, M. Schwartz, P. Danzig, K. Worrell, C. Neerdaels, A Hierarchical Internet Object Cache, USENIX Annual Technical Conference 1995.

[21] M. Krevald, J. Nievergelt, T. Roos, P. Widmayer, "Algorithmic Foundations of Geographic Information Systems", Springer, 1997