Ubiquitous Access to Distributed Data in Large-Scale Sensor Networks through Decentralized Erasure Codes

Alexandros G. Dimakis, Vinod Prabhakaran, and Kannan Ramchandran
Department of Electrical Engineering and Computer Science,
University of California, Berkeley, CA 94704.
Email: {adim, vinodmp, kannanr}@eecs.berkeley.edu

***Abstract*— Consider a large-scale wireless sensor network of n nodes, where a fraction k out of n generate data packets of global interest. Assuming that the individual nodes have limited storage and computational capabilities, we address the problem of how to enable ubiquitous access to the distributed data packets.**

Specifically, we assume that each node can store at most one data packet, and study the problem of diffusing the data so that by querying *any* k nodes, it is possible to retrieve all the k data packets of interest (with high probability).

We introduce a class of erasure codes and show how to solve this problem efficiently in a completely distributed and robust way. Specifically we show that we can efficiently diffuse the data by "pre-routing" only $O(\ln n)$ packets per data node to randomly selected storage nodes. By using the proposed scheme, the distributed data becomes available "at the fingertips" of a potential data collector located anywhere in the network

I. Introduction

The popular approach to retrieving data in wireless sensor networks is for the query unit to ask for the data from the sensor nodes of interest. The desired data is then routed from the source nodes to the data collector. This may be categorized as a "pull-based" strategy. In certain scenarios of interest, a pull-based approach at query time may have limitations. Primarily, there can potentially be a large latency in getting the desired data out of a multitude of source nodes scattered randomly across the network due to the multi-hop routing phase following the query. There is a tradeoff between the work performed at the time the data is generated relative to the work performed at query time. In general, processing done at query time introduces latency and unreliability that may not be acceptable for certain applications.

This work is accordingly motivated at trying to reduce latency and unreliability between query time and the time that the desired data is made available to the data collector. In the spirit of "smart dust" sensor networks [6], we consider a very large scale network with individual nodes severely constrained by communication, computation, and memory. When one envisions large numbers of cheap, unreliable sensors it is very reasonable to introduce redundancy to ensure that the whole network is acting as a robust distributed storage database. Of particular importance is the assumption that memory cannot scale with the size of the network.

We study the interesting case where a data collection query can be posed anywhere in the network (e.g. a mobile query unit) and require quick and reliable access to the data of interest, which is scattered randomly throughout the network. Clearly, this would require some sort of a "pre-routing" phase wherein the network needs to be prepared so as to allow for this low latency query answering capability. The key issue, of course, is whether it is possible to achieve this robust distributed storage under the communication and storage constraints imposed by the nature of wireless sensors.

This research was supported by NSF under grants CCR-0219722 and CCR-0330514. This work has been presented in part at Asilomar 2004 [4].

Our main result is a novel scheme for data dissemination that can guarantee ubiquitous access to distributed data by having only $O(\ln n)$ pre-routed packets per data node. In the proposed solution, each node operates autonomously without any central points of control. Our solution is based on a new class of erasure codes called *decentralized erasure codes*. These codes have the unique property that they can be created in a distributed way without gathering all the input data at one centralized location.

The remainder of the paper is organized as follows: In Section 2 we give the exact problem formulation, state our assumptions and discuss related work. In Section 3 we introduce decentralized erasure codes and present the theorems describing their performance. In Section 4 we give some examples of using decentralized erasure codes in specific network settings and present some experimental performance evaluation. We also present some interesting extensions of the basic model. Finally, Section 5 contains the proofs of our theorems.

II. Problem setting

We assume that there are k data-generating nodes that are measuring a physical quantity of interest (e.g. temperature). Without loss of generality we will assume that each data node generates one data packet of large size containing the measurements over some time interval.

In our initial problem setting we will assume the data packets are *independent*. In most interesting sensing scenarios the data will be highly correlated and we show how our scheme can perform distributed source coding to compress correlated data. Essentially, after distributed compression, the large correlated data packets can be replaced by smaller packets that are independent and have the theoretically smallest possible size, equal to the joint entropy of the physical sources.

Further, assume we have $n > k$ *storage* nodes that will be used as storage and relay devices. Sensor nodes have limited memory and we model that by assuming that each node (of any kind) can store, (wlog) only one data packet (or a combination having the same number of bits as a data packet). This is a key requirement to the scalability of the network. A data packet contains measurements over a time interval and can have significant size.

The ratio $k/n < 1$ is assumed fixed as k and n scale. For example we can assume that some fixed ratio (for example 10%) of nodes in a sensor network are measuring while the rest are used for storage. To avoid confusion one can think that the total number of sensors is $n + k$ or that some k out of the n sensors have an extra sensing device which is operating completely independently from the storage device. However, as will become apparent in subsequent sections, our framework is much more general and the k data nodes and n storage

nodes can be any arbitrary (possibly overlapping) subsets of nodes in a larger network of N nodes.

Throughout this paper we will be interested in answering *data collection queries* that involve gathering all k data packets to a collector. Finding efficient algorithms for answering other queries like aggregate and range queries under the proposed encoded storage scheme remains as future work.

We want to store the information contained in the k data nodes in a redundant way in all the n storage nodes. As there are k data packets of interest, and each node can store no more than 1 data packets worth of bits, it is clear that one has to query at least k nodes to get the desired data.

The problem is to store the data in such a way that users may query *any k nodes and use the results to reconstruct the original k packets* (with high probability). For instance, a data collector can get this data out of some k neighboring storage nodes in its immediate vicinity to minimize the latency.

Finally we assume that the data collector has enough memory to store k packets and enough processing power to run the maximum likelihood decoding algorithm, which as we will show corresponds to solving a system of k linear equations in a finite field.

Our proposed solution requires no routing tables, centralized processing or global knowledge of any sort. We only assume a packet routing layer that can route packets from point to point (based for example on geographic information). This can be easily achieved with only local geographic knowledge with greedy geographic or GPSR [16] routing. We further assume that there are packet acknowledgements and therefore, no packets are lost. This last assumption however can be very easily relaxed due to the completely randomized nature of the solution.

Our solution uses decentralized erasure codes, which we prove have minimal data node degree, which corresponds to a minimal number of pre-routed messages. Note however, that we do not claim optimality of the distributed storage system as a whole. This is because we rely on a packet routing layer instead of jointly optimizing across all network layers.

A. Comparison with Centralized Scheme

The problem of reconstructing the k measurements from any k out of n storage nodes can be seen as an erasure channel coding problem. If we assume the existence of a centralized super-node that can gather all the data, we could use any (n, k) capacity achieving erasure code and treat the data node packets as input symbols and the storage nodes as encoding symbols. More specifically, the centralized node would gather the k data packets, use an erasure code to generate n encoded packets (with the same size plus a small overhead for identification). Then, it would assign and send one encoded packet to each storing node. If we use a good erasure code we will be guaranteed to reconstruct the original packets by asking any k encoded storage nodes.

The most common erasure codes are Reed-Solomon which are very widely employed in many applications like computer network distributed storage systems [18], and redundant disk arrays [3].

Also, LDPC codes and more recently fountain codes [19] were proposed as alternatives with randomized construction and faster encoding and decoding times. See [22] for a practical investigation on using these codes for distributed storage.

The key feature of our problem that makes these codes unsuitable is the fact that *the data is distributed*. We could convert the problem to a centralized one by gathering all the data in one super-node and using any erasure code. This would have some advantages and some disadvantages compared with the decentralized scheme. On the positive side, a centralized solution would allow the use of Reed Solomon codes which have deterministic (as opposed to probabilistic) guarantees for decoding. Also, the use of LT or LDPC codes would allow for faster decoding times relative to the proposed codes. However, on the negative side, this does not constitute a scalable solution from a computational power, storage and communication point of view. Specifically gathering all the data in one place would create a network hotspot, and introduce security and unreliability problems if the super-node fails. In summary, the centralized solution might be preferable for small networks but would definitely have problems as the number of nodes increases.

B. Related work

The standard approach in query processing is to flood queries to all nodes, and construct a spanning tree by having each node maintain a routing table of their parents. This is the approach currently used in both TinyDB and Cougar [20]. Flooding can be pruned by constructing an analog to indexes in the network. An efficient indexing scheme is the Geographic Hash Table (GHT), which maps IDs and nodes to a metric space [26]. These approaches yield different tradeoffs between reliability over network changes, latency and communication cost. Decentralized erasure codes can be used to add storage redundancy in any existing query processing scheme when reliability is required.

There has been significant work on multiresolution distributed storage for sensor networks. Ganesan et al. [9], [10] propose the DIMENSIONS system which uses wavelets to efficiently summarize sensor data in a natural hierarchical structure. Gao et al. [12] exploit the principle of fractionally cascaded information to provide efficient algorithms and theoretical bounds for answering range queries.

Our work can be combined with multiresolution storage mechanisms as an interesting way of storing information in sensor networks. We briefly address this issue on Section 4 but more explicit investigations on jointly optimizing summarization and decentralized erasure code construction remain as part of our ongoing work.

The family of adaptive protocols called Sensor Protocols for Information via Negotiation (SPIN) [17] disseminates all the information to every node in the network. This enables a user to query any node and get the required information immediately by only communicating with one node. This is similar in spirit of the current work but relies on the assumption that each sensor node has enough memory to store all the data in the network, clearly not a scalable solution.

The proposed codes are linear codes with a structured randomized construction and are therefore related with LDPC [8] and LT codes [19]. In addition, there are some interesting connections with the rateless property of digital fountain codes. These connections are made explicit in a subsequent section.

C. Connections to Network Coding

Decentralized erasure codes can be seen as linear Network codes [13], [14], [15] on the (random) bipartite graph connecting the data and the storage nodes (where each edge corresponds to one pre-routed packet). Network coding is an exciting new paradigm for communication in networks. This paper investigates some connections between network coding and codes for distributed storage. A related investigation on applications of network coding on random graphs can be found in [7].

An equivalent way of thinking of the distributed storage problem is that of a random bipartite graph connecting the k data nodes with the n storage nodes and then adding a data collector for

every possible subset of size k of the n storage nodes. Then the problem of multicasting the k data packets to all the data collectors is equivalent to making sure that every collection of k storage nodes can reconstruct the original packets. It has been shown that random linear network codes [15] are sufficient for multicasting problems as long as the the underlying network can support the required throughput. The key difference between classical network coding and our problem is that in network coding, one is given a fixed multicasting network and tries to maximize throughput. In our problem, the code is created over the random, bipartite graph connecting data and storage nodes. Note that this graph does not correspond to any physical communication links but to virtual routing selections that are made by the algorithm. Therefore this graph is not given, but can be explicitly designed to minimize communication cost. Essentially we are trying to make this random bipartite graph as sparse as possible (because each edge corresponds to communication) while keeping the flow high enough and also enforcing each data node to act independently, without coordination.

The key technical condition we need to prove to establish that decentralized erasure codes will be decodable is that the random bipartite graphs we construct have a perfect matching [1] with high probability. The existence of a perfect matching guarantees that the max flow that can go through the network is sufficient. Our main theoretical contribution is in quantifying how sparse these random bipartite graphs can be under these constraints. The proof is obtained by using an extension of a combinatorial counting technique introduced by P. Erdős and A. Rényi in [5], [2] for analyzing matchings in random bipartite graphs. The extension stems from the dependencies on the data nodes which destroy the symmetry assumed in [5], [2] and thereby complicating matters (see Section 5 for more details).

III. Decentralized Erasure Codes

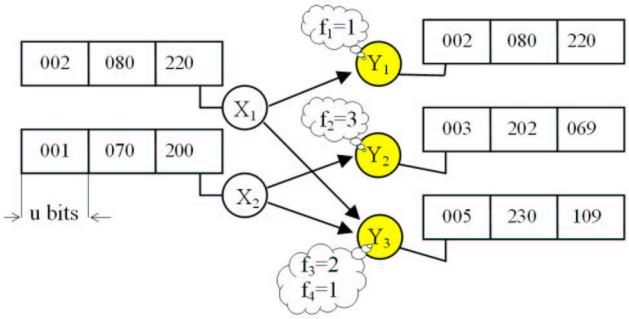

Fig. 1. A simple example of a linear code over $GF(2^8)$. Here $k = 2$, $n = 3, k/n = 2/3$, $q = 256$. The primitive polynomial of the field is $D^8 + D^4 + D^3 + D^2 + 1$

Decentralized Erasure Codes are random linear codes over a finite field $GF(q)$ with a specific generator matrix structure.

A toy example of a linear code over $GF(2^8)$ is given in figure (1). In the example there are two data nodes X_1 and X_2 and three storage nodes Y_1, Y_2, Y_3. We assume the data nodes have gathered a number of measurements. In the example we choose $u = 8$ bits to represent each number in our field which corresponds to $GF(2^8)$. The bits of the data measurements are divided into blocks of u bits which correspond to elements in $GF(2^8)$ (for example $X_1(1) = 002, X_1(2) = 080, X_1(3) = 220$). Now the data packet X_1 is pre-routed to storage nodes Y_1, Y_3 and X_2 to Y_2, Y_3. Once a storage node receives one or more data packets, it selects coefficients f_i uniformly and independently in $GF(2^8)$. Each coefficient then multiplies each blocks independently, multiple blocks are added (under the arithmetic of the Galois Field) and the results are cascaded into a new block packet Y_i that has exactly the same size as all the data packets. So for example Y_3 has stored a packet that corresponds to $2X_1 + 1X_2$. Using this notation we mean that $Y_3(i) = 2X_1(i) + 1X_2(i)$ for $i = 1, 2, 3$. Each storage node will also store the coefficients f_i that it selected. This introduces an overhead storage that can be made arbitrarily small by coding over larger blocks, this issue is discussed in more detail in a subsequent section.

Notice that in Figure 1 any two out of the three encoding packets can be used to reconstruct the original data. In general, linear codes can be represented using their generator matrix in the form $u = mG$ where u is an $1 \times n$ encoding vector, m is $1 \times k$ input vector and G is a $k \times n$ matrix. In general the vectors and the generator matrix are general field elements in some $GF(q)$. For the example

$$G = \begin{pmatrix} 1 & 0 & 2 \\ 0 & 3 & 1 \end{pmatrix} \quad (1)$$

to retain m the receiver must invert a $k \times k$ submatrix G' of G. The key property required for successful decoding (that is also true for the example) is that any selection of G' forms a *full rank matrix*.

A. Code Construction

To construct a decentralized erasure code, each data symbol is independently pre-routed to $d(k)$ storage nodes. More specifically, it picks one out of the n storage nodes, pre-routes the packet and repeats $d(k)$ times. Each storage node multiplies whatever it happens to receive with coefficients selected uniformly and independently in $GF(q)$. A schematic representation of this is given in figure 2. This corresponds to *generating each row of the G matrix independently* by placing at most $d(k)$ nonzero elements in random positions.

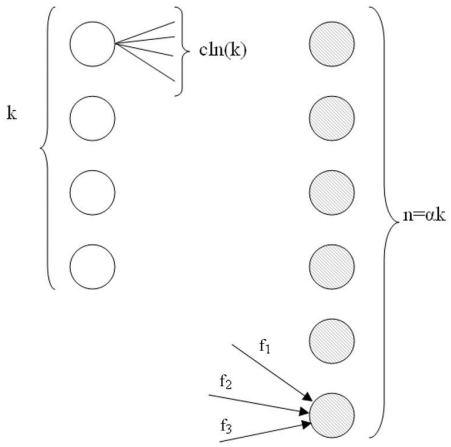

Fig. 2. Decentralized erasure codes construction. There are $d(k) = c\ln(k)$ edges starting from each data node and landing independently and uniformly on the storage nodes (If two edges have the same starting and ending point, we identify them)

The number of messages pre-routed by each data node is $d(k)$ and the total number of packets that will be pre-routed is $O(kd(k))$. To minimize the communication cost for a sensor network we want to find the smallest possible $d(k)$ that guarantees that $k \times k$ submatrices

G' of the random G we generate will be full rank with high probability.

The following theorems are the main results of the paper:

Theorem 1: $d(k) = c\ln(k)$ is sufficient so that collecting any random $k \times k$ submatrix G' of G is full rank with probability at least $(1 - \frac{k}{q})c_2(k)$. (where $c_2(k) \to 1$ as $k \to \infty$ and $c > 5\frac{n}{k}$)

Theorem 2: (converse) If each row of G is generated independently (Decentralized property), at least $d(k) = \Omega(\ln(k))$ is necessary to have G' invertible with high probability.

From the two theorems it follows that $d(k) = c\ln(k)$ is (order) optimal. Therefore, decentralized erasure codes have minimal data node degree.

Decentralized erasure codes can be decoded using ML decoding which corresponds to solving a linear system of k equations in $GF(q)$. This has a decoding complexity of $O(k^3)$. This is not as efficient as the belief propagation decoding used in $LDPC$ and LT codes. It is perhaps possible to have a faster decoding algorithms but the focus in this paper is the decentralized property since we assume that the data collector can have large computational power. Notice that if some data nodes have failed and only $k' < k$ nodes pre-route packets, asking any k' will still yield an invertible matrix. Therefore, the proposed scheme is robust in both data and storage node failures.

B. Distributed Digital Fountains

One key property of Digital Fountain codes (for example LT codes [19]), is the fact that they are rateless. This means that LT codes create every encoded packet independently. In other words, *every column* of the generator matrix of an LT code is independent of the others (with logarithmic average degree, very similarly to our result). In this context, one can think of the decentralized property as being the transpose of the rateless property of digital fountains. This is because in our case it is the rows of the the generator matrix that are independent and this corresponds to *having each data source* acting independently. The equivalence to the rateless property in our case is that sources can fail (or even be added) without affecting the performance of the code. Therefore each source is independently spraying the storage nodes with information. If a collector acquires enough encoded packets, its possible to retrieve all the sources.

C. Storage Overhead

Notice that other than storing the linear combination of the received data packets, each node must also store the randomly selected coefficients f_i. One can easily see that each storage node will be storing only $O(\ln k)$ coefficients w.h.p. Therefore, as long as the total data packet size is much larger than $u \ln k$ bits the overhead is negligible. The overhead is also related with the probability of decoding. Theorem 1 bounds the probability of failure by k/q. Recall that since $q = 2^u$ where u is the number of bits required to store each f_i, one can reduce the probability of error exponentially in the overhead bits. As a numerical example, say we set $k = 1000$ data nodes and $u = 20$ bits. Say, one data packet is 200 bytes of data. Therefore, it requires an overhead storage of approximately $138/1600 \approx 8.6\%$ to guarantee a probability of failure smaller than $\frac{1000}{2^{20}} \approx 9 \times 10^{-4}$.

IV. Sensor Network Scenarios

In this section we show how decentralized erasure codes can be applied to various sensor network scenarios and analyze their performance. It is very important to realize that one can pick the k data nodes and the n storage nodes to be *any arbitrary subsets of nodes* of a larger network. The exact choices depend on the specific sensing application. The only requirement that we impose is that n/k should remain fixed as the network scales.

In general it is easy to determine the total communication cost involved in creating a decentralized erasure code. Each data symbol pre-routes to $5\frac{n}{k}\ln k$ storage nodes, therefore the total number of packets sent will be $5n\ln k$. To determine the communication cost in terms of radio transmissions we need to impose a specific network model for routing. For example, if the diameter of the network is $D(n)$ then the total communication cost to build a decentralized erasure code will be at most $O(D(n)n\ln k)$. To become more specific we need to impose additional assumptions that depend on the specific application. If $D(n) = O(\sqrt{n})$ for example in a grid network, the total communication cost would be bounded by $O(n^{1.5}\ln k)$ to make the data available in $k = O(n)$ storage nodes.

Since each data node is essentially multicasting its packet to $O(\ln k)$ storage nodes, multicast trees can be used to minimize the communication cost. These issues depend on the specific network model and geometry and we do not address them in this paper.

A. Perimetric Storage

To perform some experimental evaluation and also to illustrate how the decentralized erasure codes can be used as a building block for more complex applications we consider the following scenario. Suppose we have N total nodes placed on a grid in the unit square (dense scaling) and we are only interested in storing information in the $4\sqrt{N}$ nodes on the perimeter of the square (see figure 3). This is an interesting extension since in most cases the sensor network will be monitoring an environment and potential users interested in the data will have easier access to the perimeter of this environment. Therefore we will have $n = 4\sqrt{N}$ storage nodes and $k = \rho\sqrt{N}$ data nodes for some constant $\rho < 4$. The k data nodes can be placed in the grid randomly or by using some optimized sensor placement strategy [11]. Notice that we only have $O(\sqrt{N})$ nodes measuring or storing. The rest are used as relays and perhaps it is more interesting to assume that the k data nodes are duty-cycled to elongate the lifetime of the network. Note that in a dense network scenario $\sqrt{N}$ can become sufficiently large to monitor the environment of interest. Again, we want to query any k nodes from the perimeter and be able to reconstruct the original k data packets w.h.p. The problem now is that the diameter of the network (assuming greedy geographic routing) is $O(\sqrt{N}) = O(n)$ as opposed to $\sqrt{n}$.

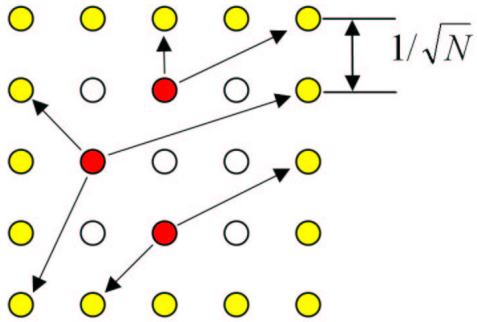

Fig. 3. Perimetric storage: The $n = 4\sqrt{N}$ nodes on the perimeter are used as storage, and $k = O(\sqrt{N})$ nodes inside the grid are the data nodes.

We assume that the transmission radius is scaling like $O(\frac{1}{\sqrt{N}})$ and measure communication cost as the total number of 1-hop radio transmissions (each transfers one packet for one hop) required to build the decentralized erasure code. It can be easily seen that the total communication cost is at most $O(N\ln N)$ which yields a logarithmic

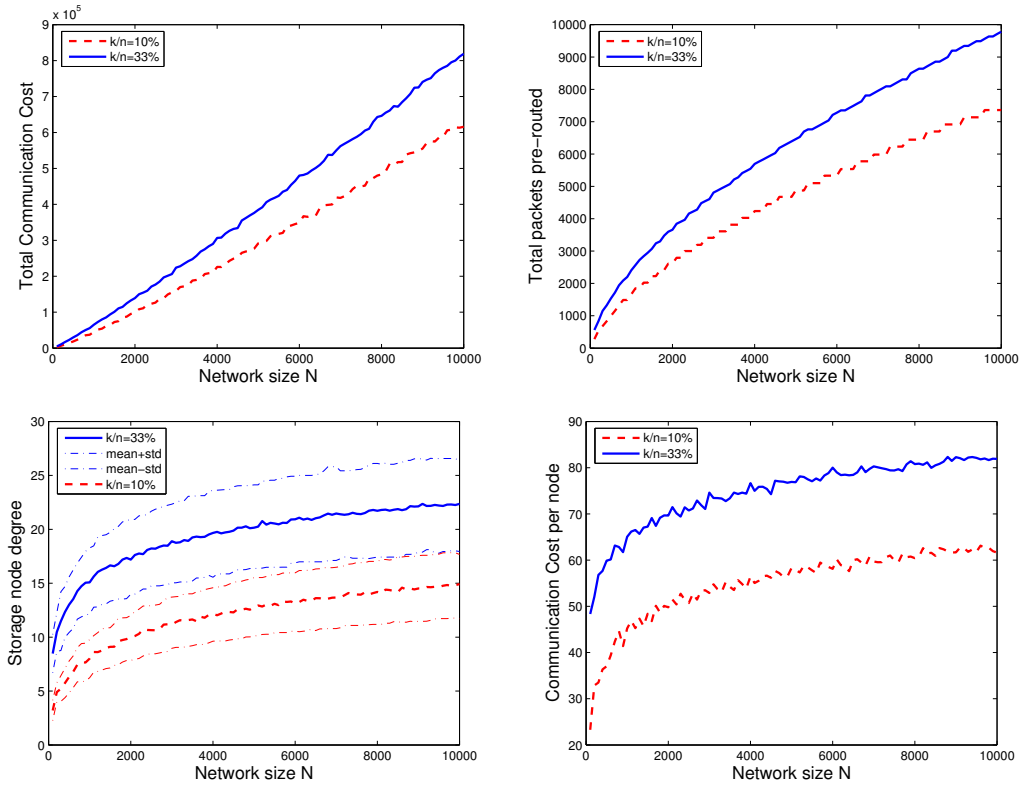

Fig. 4. Experiments for perimetric storage scenario. For each subgraph, we plot $k/n = 10\%$ (so $k = 0.4\sqrt{N}$) and $k/n = 33\%$ (so $k = 4/3\sqrt{N}$). In both cases $n = 4\sqrt{N}$
a) Total communication cost to prepare the decentralized erasure code. b) Total number of packets pre-routed. c) Average and standard deviation plots for the number of packets that are stored at storage nodes. e) Total communication cost per node.

bound $O(\ln N)$ on the transmissions per node. Figure 4 illustrates some experiments on the performance under the perimetric storage scenario. Notice that the communication cost per node is indeed growing very slowly in N.

B. Multiresolution

Assume we have a mechanism to create spatial summarization of data, perhaps using wavelets as in [9]. Then we can imagine a scenario where the network will be separated into regions $R_1, R_2, \cdots R_m$ and a summary data packet (corresponding to "headline news") is computed for each region at some fusion point. Then we can use decentralized erasure codes with $k = m$, i.e. setting the data nodes to be the fusion points and $n = N$. This will disseminate the summary data everywhere in the network. A potential user can then query any m nodes to obtain the summaries. Therefore, "headline news" for all the regions is available everywhere and if further details are required about one specific region, the user can query nodes from that region. This, of course can be done in many layers, yielding an interesting tradeoff between latency, reliability, and resolution.

C. Correlated Data

When the data at the measuring nodes are correlated, this correlation can be exploited to improve the performance. It was shown by Slepian and Wolf [27] that in theory no loss of performance results from the fact that the sources are distributed and the total rate of the code required is equal to the joint entropy rate of the sources. Distributed Source Coding Using Syndromes (DISCUS) [23] is a practical means of achieving this. The data nodes form the syndromes of the data packets they observe under suitable linear codes. These syndromes are treated as the data which the nodes pre-route to form the decentralized erasure codewords at the storage nodes. The data collector reconstructs the syndromes by gathering the packets from k storage nodes. Using DISCUS decoding the collector can recover the original data from the syndromes. The correlation statistics, which is required by DISCUS can be learned by observing previous data at the collection point. The data nodes only need to know the rates at which they will compress their packets. This can be either communicated to them or learned adaptively in a distributed network protocol. The syndromes can be considerably shorter than the original data packets if the data observed by the different nodes are significantly correlated as is usually the case in sensor networks. Note that this approach is separating the source coding problem from the storage problem and this is not be optimal in general [24]. However, this constitutes a practical and efficient way to considerably reduce the packet sizes.

Another possibility that remains to be investigated is for the nodes to send out the correlated packets they observe to the storage nodes as in section 3.1. The storage nodes could form random linear combinations of the packets they receive. Since the data packets are correlated it is possible for the collector to query fewer than k storage nodes to recover the data. Note however this would not reduce the network traffic but would help the data gathering procedure.

V. Analysis and Proofs

The main body of this section involves the proof of Theorem 1, stating that $d(k) = c\ln(k)$ is sufficient to guarantee that a random

$k \times k$ submatrix G' of G is full rank w.h.p. To prove that G' is full rank it suffices to show that the determinant $detG' \neq 0$. There is a close connection between determinants of matrices and graph matchings which for the bipartite case is given by Edmonds' theorem [21]. A bipartite graph will have a perfect matching (P.M.) if there exists a subset $E' \subseteq E$ of its edges so that no two edges in E' share a common vertex and all the vertices connect to an edge in E'. For our problem we have a random bipartite graph formed by the data nodes and the storage nodes as in figure 2. Each data node has $c \ln k$ edges that land randomly on the storage nodes. The combinatorial counting technique we use was introduced by P. Erdős and A. Rényi in [5], [2].

More specifically, we define the graph $B_{lnk-left-out}$ as the random bipartite graph with two sets of vertices, V_1, V_2, where $|V_1| = k$, $|V_2| = n$, $n = \alpha k, (\alpha > 1)$. Every vertex in V_1 connects with $c \ln(k)$ vertices of V_2 each one chosen independently and uniformly with replacement. If two edges connect the same two vertices we identify them. Then we pick a subset $V_2' \subset V_2$ where $|V_2'| = k$ and form the random bipartite graph $B'_{lnk-left-out} = |V_1| \cup |V_2'|$. Edges that connect to $V_2 \setminus V_2'$ are deleted.

This graph corresponds to the submatrix G' and the key property we require to establish our result is that $B'_{lnk-left-out}$ has a perfect matching w.h.p.

Theorem 3: *Let $B'_{\ln k-left-out}$ be a bipartite graph with $|V_1| = |V_2'| = k$ obtained from $B_{\ln k-left-out}$ by taking a random subset of k storage nodes. $B'_{\ln k-left-out}$ has a perfect matching with probability $1 - \mathbf{o}(1)$ as $k \to \infty$.*

Proof:

For a set of nodes $A \subset V_i$ of a bipartite graph B, we denote $\Gamma(A)$=$\{y : xy \in E(B)\ x \in A\}$. So $\Gamma(A)$ is simply the set of nodes that connect to nodes in A.

A key theorem used in this proof is Hall's theorem. We use it in the following form (which is easily derived from the standard theorem [1], [2]):

Lemma 1: Let B be a bipartite graph with vertex classes V_1,V_2' and $|V_1| = |V_2'| = k$. If B has no isolated vertices and no perfect matching, then there exists a set $A \subset V_i$ $(i = 1, 2)$ such that:
i) $|\Gamma(A)| = |A| - 1$
ii) The subgraph $A \cup \Gamma(A)$ is connected
iii) $2 \leq |A| \leq (n + 1)/2$.

The event that B has no perfect matching can be written as the union of two events. Specifically, let E_0 denote the event that B has one or more isolated vertices: $P(no\, P.M.) = P(E_0 \cup \exists A)$ (for some set A satisfying Lemma (1)) Therefore by a union bound we have: $P(no\, P.M.) \leq P(E_0) + P(\exists A)$. We will treat the isolated nodes event later. We know from Lemma (1) that the size of A can vary from 2 to $(k + 1)/2$, so we obtain the union bound:

$$P(\exists A) = P(\bigcup_{i=2}^{(k+1)/2} (\exists A, |A| = i)) \leq \sum_{i=2}^{(k+1)/2} P(\exists A, |A| = i). \quad (2)$$

We can further partition into two cases, that the set A belongs to V_1 (the data nodes) or V_2' (the k storage nodes used to decode).

$$P(\exists A) \leq \sum_{i=2}^{(k+1)/2} P(\exists A \subset V_1, |A| = i) + P(\exists A \subset V_2', |A| = i) \quad (3)$$

So we now bound the probabilities $P(\exists A \subset V_1, |A| = i)$ and $P(\exists A \subset V_2', |A| = i)$ using a combinatorial argument. *Case I: A belongs in the data nodes:* Suppose we fix i nodes $A_1 \subset V_1$ and $i - 1$ nodes on $A_2 \subset V_2'$. Then the probability that a set $A = A_1$ satisfies the conditions of lemma (1) with $\Gamma(A) = A_2$ is equal to the probability that all the edges starting from A_1 will end in A_2 or are deleted. Note however that every node in V_1 picks $c \ln(k)$ neighbors from the set V_2 (which is the large set of $n = \alpha k$ nodes). We bound the probability by allowing all edges starting from A_1 to land in $A_2 \cup V_2 \setminus V_2'$. Therefore we have $ci \ln(k)$ edges that must land in $A_2 \cup V_2 \setminus V_2'$ and $|A_2 \cup V_2 \setminus V_2'| = i - 1 + (\alpha - 1)k$. Note that all the other edges can land anywhere and that would not affect $|\Gamma(A)|$. Therefore, since there are $\binom{k}{i}$ choices for A_1 and $\binom{k}{i-1}$ choices for A_2 we have:

$$P(\exists A \subset V_1) \leq \sum_{i=2}^{(k+1)/2} \binom{k}{i}\binom{k}{i-1} (\frac{i - 1 + (\alpha - 1)k}{\alpha k})^{ci \ln(k)} \quad (4)$$

We can always bound this sum by its maximum value times k (since there are less than k positive quantities added up). Therefore it suffices to show that

$$kP(\exists A \subset V_1, |A| = i) = \mathbf{o}(1), \qquad \forall i \in [2, (k + 1)/2] \quad (5)$$

as $k \to \infty$.

From Stirling's approximation we obtain the bound [2] $\binom{k}{i} \leq (\frac{ek}{i})^i$ and also it is easy to see that $(\frac{ek}{i-1})^{i-1} \leq (\frac{ek}{i})^i$ when $i \leq k$.

If we denote $\mathcal{X} = \left(\frac{i-1+(\alpha-1)k}{\alpha k}\right) < 1$ and use these two bounds we obtain :

$$P(\exists A \subset V_1, |A| = i) \leq \exp\Big(\ln(k)(2i + ic \ln(\mathcal{X})) + 2i(1 - ln(i))\Big) \quad (6)$$

If we multiply by k we get from (5) that it suffices to show $\exp\Big(\ln(k)(2i + ic \ln(\mathcal{X}) + 1) + 2i(1 - ln(i))\Big) = \mathbf{o}(1)$ for all $i \in [2, (k + 1)/2]$, as $k \to \infty$. Therefore, for this exponential to vanish it is sufficient to have the coefficient of $\ln k$ be negative: $2i + ic \ln(\mathcal{X}) + 1 < 0$, which gives us a bound for c: $c > \frac{-(1+2i)}{i \ln(\mathcal{X})}$ Notice that $\mathcal{X} < 1$ and therefore it is possible to satisfy this inequality for positive c. This bound should be true for every $i \in [2, (k+1)/2]$.

Therefore, a sufficient condition for $P(\exists A \subset V_1)$ to vanish is

$$c > \frac{-5}{2 \ln(\frac{\alpha - 1/2}{\alpha})} \simeq 5\alpha. \quad (7)$$

Case II: A belongs in the encoding nodes: With the same technique, we obtain a bound if the set A is on the data nodes. This time we pick $A \subset V_2'$ with $|A| = i$ and we want $|\Gamma(A)| = i - 1$. So we require that all edges that connect to A end in a specific set $A_2 \in V_1$. The extra requirement that $A \cup \Gamma(A)$ should be connected, further reduces the probability and is bounded away. We therefore obtain the bound:

$$P(\exists A \subset V_2, |A| = i) \leq \binom{k}{i-1}\binom{k}{i}\Big(\frac{\alpha k - i}{\alpha k}\Big)^{c(k-(i-1)) \ln(k)} \quad (8)$$

Which yields the condition for $c : c > \alpha \frac{k}{i} \frac{2i+1}{k-i} = 2\alpha \frac{k}{k-i} + \alpha \frac{k}{i(k-i)}$ Now notice that this is a convex function of i so the maximum is obtained at $i = 2$ or $i = \frac{k+1}{2}$.

By substituting $i = 2$ and $i = \frac{k+1}{2}$ we find that these inequalities are always dominated by (7). So finally we require that $c > 5\alpha$.

Case III: There exist no isolated nodes: We will say that a data or storage node is isolated when it connects to no storage or data node respectively. Bounding the probability of this event $P(E_0)$ is easier to deal with. Notice that data nodes cannot be isolated by construction. The αk encoding nodes receive totally $kc \ln(k)$ independent connections and we need to show that they are all covered by at least one encoding node w.h.p. Using a standard bound

we obtain the following result: ([21]):Let C denote the number of connections required to cover all αk encoding symbols. then $P[C > \beta\alpha k \ln(\alpha k)] \leq (\alpha k)^{-(\beta-1)}$ Which shows that any $\beta > 1$ (we require $\beta > 5$) will suffice to cover all the encoding symbols with high probability.

Therefore from combining all the required bounds for c we find that $c > 5\alpha = 5\frac{n}{k}$ is sufficient for the bipartite graph to have a perfect matching with high probability. ■

Proof of Theorem 1 (Sketch) We want to show that the matrix G' is full rank with high probability. Recall that by construction every row of G' has a logarithmic number of non-zero coefficients chosen uniformly and independently from a finite field $GF(q)$. Denote these coefficients by $f_1, f_2, \cdots f_L$ Their actual number L is random and approximately equal (and in fact, smaller than) $ck\ln(k)$. It suffices to show that the determinant of G' is nonzero w.h.p. Note that $\det|G'| = \sum_\pi sgn(\pi)\prod_{i=1}^{k} g'_{i,\pi(i)}$ where we are summing over all the permutations of $\{1, 2, \cdots k\}$ and $g'_{i,j}$ is the i,jth element of G'. Notice that this is a multivariate polynomial $det(G') = P(f_1, f_2, \cdots, f_L)$. There are two fundamentally different cases for the determinant to be zero. If for each term corresponding to each permutation there existed one or more zero elements then the determinant would be identically zero (for any choice of $f_1, f_2, \cdots f_L$). Now the key step is to notice that each permutation corresponds to exactly one potential matching of the bipartite graph. Therefore if the graph has a perfect matching, $det(G')$ will not be identically zero. (Edmonds' Theorem [21]) The other case is when $deg(G')$ is a non-zero polynomial but the specific choices of $f_1, f_2, \cdots f_L$ correspond to one of its roots. It is clear that this is a rare event and we can bound its probability using the Schwartz-Zippel Theorem [21]. Notice that the degree of $det(G')$ is exactly k when there exists a perfect matching (which by Theorem 3 is true with probability $c_2(k) \to 1$ as $k \to \infty$) so we obtain a bound on the probability of failure: $Pr(det(G') = 0 | det(G') \not\equiv 0) \leq \frac{k}{q}$ and the event $det(G') \not\equiv 0$ is exactly the same as having a perfect matching which we have shown happens with high probability as k goes to infinity . ■

Proof of Theorem 2 (Converse) It is a standard result in balls and bins analysis [21] that in order to cover n bins w.h.p. one needs to throw $O(n \ln n)$ balls (See also case III in proof of Th. 1). Notice that in our case, covering all the storage nodes is necessary to have a full rank determinant (since not covering one corresponds to having a zero column in G). Therefore any scheme that connects data symbols to encoding symbols randomly and independently will require at least $\Omega(\ln k)$ connections per data node. ■

VI. Conclusions and Future Work

We have proposed decentralized erasure codes and shown how they can be used to introduce reliable distributed storage. Our future work involves jointly optimizing summarization and code construction for multiresolution storage scenarios. Other issues of interest involve investigating the effect of pre-routing a constant number of packets per data node as well as devising more efficient algorithms for decoding.

References

[1] B. Bollóbas "Modern Graph Theory", Springer 2000.

[2] B. Bollóbas "Random Graphs (second edition)", Cambridge University Press, 2001

[3] W. A. Burkhard and J. Menon. "Disk array storage system reliability". In 23rd Int. Symp. on Fault-Tolerant Comp.,1993.

[4] A. G. Dimakis, V. Prabhakaran and K. Ramchandran "Distributed Data Storage in Sensor Networks using Decentralized Erasure Codes", Asilomar Conference on Signals, Systems and Computers, November 2004.

[5] P. Erdős and A. Rényi, "On random matrices", Publ. Math. Inst. Hungar. Acad. of Sciences. 8, 1964.

[6] J. M. Kahn, R. H. Katz and K.S.J. Pister "Next century challenges: mobile networking for Smart Dust" Proc. of the 5th annual ACM/IEEE Int. Conf. on Mobile computing and networking, 271 - 278, 1999.

[7] D. Petrovic, K. Ramchandran, J. Rabaey, "Reliable Communication using Unreliable Radios", to appear in IEEE workshop on Network coding (NETCOD) 2005.

[8] R. G. Gallager. Low-Density Parity-Check Codes. MIT Press, Cambridge, MA, 1963.

[9] D. Ganesan, B. Greenstein, D. Perelyubskiy, D. Estrin and J. Heidemann, "An Evaluation of Multi-resolution Storage for Sensor Networks" Proceedings of the First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003).

[10] D. Ganesan, D. Estrin, J. Heidemann, "DIMENSIONS: Why do we need a new Data Handling architecture for Sensor Networks?", Proc. of ACM Computer Communication Rev., Volume 33, Number 1.

[11] D. Ganesan, R. Cristescu and B. Berefull-Lozano, "Power-Efficient Sensor Placement and Transmission Structure for Data Gathering under Distortion Constraints", Symposium on Information Processing in Sensor Networks (IPSN '04), April 2004.

[12] J.Gao L.J. Guibas, J. Hershberger, L. Zhang, "Fractionally Cascaded Information in a Sensor Network" Symposium on Information Processing in Sensor Networks (IPSN '04), Berkeley, California, April 2004.

[13] T. Ho, R. Koetter, M. Médard, D. R. Karger and M. Effros, "The Benefits of Coding over Routing in a Randomized Setting", International Symposium on Information Theory (ISIT) 2003.

[14] T. Ho, M. Médard, J. Shi, M. Effros and D. R. Karger, "On Randomized Network Coding", Invited Paper, 41st Allerton Annual Conference on Communication, Control, and Computing, 2003.

[15] T.Ho, M. Médard, R. Koetter, D. Karger, M. Effros, J.Shi and B. Leong "Toward a Random Operation of Networks". Submitted to Tran. Information Theory.

[16] B. Karp and H. Kung. "Greedy Perimeter Stateless Routing" In Proceedings of ACM Conf. on Mobile Computing and Networking (MOBICOM), Boston, MA, 243-254, 2000.

[17] J. Kulik, W. R. Heinzelman, and H. Balakrishnan, "Negotiation-based protocols for disseminating information in wireless sensor networks", Wireless Networks, Volume: 8, pp. 169-185, 2002.

[18] 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.

[19] M. Luby, LT Codes, In Proc. of IEEE FOCS, 2002.

[20] S. Madden and J. Gehrke. "Query processing in sensor networks". Pervasive Computing, 3(1), January-March 2004.

[21] R. Motwani and P. Raghavan, "Randomized Algorithms", Cambridge University Press, 1995.

[22] J.S. Plank, M.G. Thomason, "A practical analysis of low-density parity-check erasure codes for wide-area storage applications" 2004 International Conference on Dependable Systems and Networks, 101 - 110, 2004.

[23] S. S. Pradhan and K. Ramchandran, "Distributed source coding using Syndromes (DISCUS): Design and Construction", IEEE Tran. on Info. Theory, March, 2003.

[24] A. Ramamoorthy, K. Jain, P. A. Chou and M. Effros, "Separating Distributed Source Coding from Network Coding", 42nd Allerton Conference on Communication, Control and Computing, 2004.

[25] S. Ratnasamy, B. Karp, R. Govindan, D. Estrin, S. Shenker, "Data-Centric Storage in SensorNets" In First Workshop on Hot Topics in Networks (HotNets-I) 2002.

[26] S. Ratnasamy, B. Karp, L. Yin, F. Yu, D. Estrin, R. Govindan, and S. Shenker. "GHT: a geographic hash table for data-centric storage". In workshop on Wireless sensor nets and apps, 2002.

[27] D. Slepian, J.K. Wolf, "Noiseless Coding of Correlated Information Sources". IEEE Trans. Information Theory, IT-19, 1973, pp. 471-480.