In-Network Coding for Resilient Sensor Data Storage and Efficient Data Mule Collection

Michele Albano[1] and Jie Gao[2]

[1] Instituto de Telecomunicações (IT), Aveiro, Portugal,
michele@av.it.pt
[2] Department of Computer Science, Stony Brook University, USA,
jgao@cs.sunysb.edu

Abstract. In a sensor network of n nodes in which k of them have sensed interesting data, we perform in-network erasure coding such that each node stores a linear combination of all the network data with random coefficients. This scheme greatly improves data resilience to node failures: as long as there are k nodes that survive an attack, all the data produced in the sensor network can be recovered with high probability. The in-network coding storage scheme also improves data collection rate by mobile mules and allows for easy scheduling of data mules.
We show that using spatial gossip we can compute the erasure codes for the entire network with a total of near linear message transmissions, thus improving substantially the communication cost in previous scheme [5]. We also extend the scheme to allow for online data reconstruction, by interleaving spatial gossip steps with mule collection. We present simulation results to demonstrate the performance improvement using erasure codes.

Key words: Sensor Networks, Erasure Coding, Resilient Storage, Data Mule

1 Introduction

Data representation is a central problem in sensor network design. How to represent sensor data and where to store it greatly influence the design of other operation modules. In the most straightforward scheme, the sensor readings are simply kept in the sensor's local flash drives. However, this scheme is vulnerable to node failures: when a node stops being functional (by natural disasters, human/animal intrusion, or due to energy depletion), the data stored on the broken nodes is lost as well.

The issue of data storage and representation is also closely related to the data collection mechanism that gathers the sensor data to a base station. Earlier schemes have mostly adopted data collection trees and used multi-hop routing, e.g., in TinyDB [19]. This scheme presents the problem that nodes near the base station relay more traffic than an average node, and thus would use up their energy sooner. This will create a network hole around the base station that isolates the base station from the rest of functioning sensor network. For

this reason, data collection by a mobile base station, called *data mule*, becomes attractive [8, 10, 16, 23, 26].

A data mule tours around in the network and retrieves data through direct communication with a sensor in close proximity. The mule can be either a vehicle/robot specifically employed for this task, or existing mobile entities (e.g., wild animals) that collect data in an opportunistic manner. For dedicated mules, the problem of motion planning and coordination is not trivial. A natural metric to plan routes is to minimize the total travel distance (proportional to delay or energy consumption) of the data mules. This becomes the multiple traveling salesman problem (mTSP), which is NP-complete and does not have any efficient approximation algorithms [4]. Existing solutions for mule planning are all heuristic schemes [8, 10, 16, 23, 26]. Alternatively, a practically appealing solution is to let data mules follow random movement, which requires minimum coordination. The downside is that we encounter the coupon collector problem. Specifically, if the data mule visits a random node each time, initially it picks up new data with high probability from each visited node. After the data mule has collected a substantial fraction of all the data from the network, it is highly likely that the next random node encountered has been visited before. Thus it takes the mule a long time to aimlessly walk in the network and hope to find the last few pieces. Theoretically for a random walk to cover a grid-like network, the number of steps is quadratic in the size of the network [17].

In many applications the sensor network is to detect target presence or other discrete events. Only a subset of sensors may have interesting data to report (we will denote these sensors as *data nodes* and the other ones as *storage nodes*) and the locations of these sensors are uncertain before hand (see [6]). Before the data mule visits the data nodes they have no idea where they are. Thus any pre-defined motion planning is inefficient as the mules may visit many nodes without anything to report. One can alleviate this problem by data replication (s.t. some storage nodes hold data copies). But one data mule may collect data that has been collected before; and multiple mules may collect the same piece of data due to lack of coordination.

In-network coding. Consider the following simplified setting with k sensors with data and altogether n nodes that can possibly store it. Typically k is a fraction of n. We apply *erasure coding*, which transforms k data blocks into n codewords such that the original data can be recovered from any k codewords. Thus, as long as data mules collectively gather k codewords, the original data can be successfully reconstructed at the base station.

Using erasure codes for data storage brings two benefits for data mule collection: (1) sensor data is preprocessed in the network. Nodes with interesting data will initiate the encoding procedure and will preload useful information

into other nodes in the network. Should only a small set of unknown sensors hold data to report, the data mule would not need to acquire the location of these sensors and can pick up data from any sensor; (2) Using erasure codes solves the coupon collector problem. The mules only needs to pick up a sufficient number of codewords, and it takes only linear time for random walk to achieve that [3]. The challenge of using erasure codes for sensor data storage is to develop distributed encoding algorithm with low communication cost.

Our contribution. In this work, we construct the erasure codes with *near linear* message cost. In particular, we use spatial gossip in which each node p chooses another node q with probability proportional to $1/|pq|^3$, where $|pq|$ is the Euclidean distance between p and q. The gossip proceeds in synchronous rounds. In each round, every node with data (either its own data or data received in previous rounds) multiplies its data by a random coefficient and sends it to another node chosen with the above spatial distribution. A node may receive messages from multiple other nodes, in this case it will store the summation of the received data and its current data. After $O(\text{polylog}n)$ rounds, each node in the network stores a random linear combination of the original data. Once a node i receives data from node j, either directly or indirectly, some linear combination with the data of node i will be delivered to all other nodes that j communicates with in the following rounds. Thus data will be disseminated in an exponential rate. Using spatial gossip to build the in-network codes limits the total number of transmissions needed to build the erasure codes to $O(n \text{ polylog}n)$, that is substantially smaller than the cost $O(n\sqrt{n})$ of the state of the art [5].

In addition to the basic mule collection, we also consider time critical applications for which we would like to reconstruct data as soon as the data mule acquires new information. For this, we keep all the computed coded values in the sensors, instead of replacing them with the newly computed codeword. If we denote the number of original data pieces in a linear combination as the *degree* of the codeword, this strategy considers increasing the degree of the codewords "slowly". In the first rounds, the sensors exchange the original data, so that initially the mule will pick up data in the original form. Gradually when the mule has collected a sufficiently large subset of data, it becomes harder to encounter new original data. Now we use gossip algorithm to exchange codewords of degree 2, and 3, and so on. For each piece of coded data collected, the mule will use its available data to reconstruct the original data. This is motivated by the idea of growth code developed by Kamra et al [9].

1.1 Related work

In-network coding. In the past, random linear codes have been proposed for improving sensor data resilience [5]. In that scheme each node that produced

some data sends its data to $O(\log k)$ randomly chosen storage nodes. A storage node that receives multiple data from different data nodes saves a linear combination of them with random coefficients. The paper showed that a random set of k storage nodes can recover the original data with high probability. But the total message cost is $\widetilde{O}(n\sqrt{n})$ for uniformly randomly distributed sensors.

Fountain codes and other erasure codes have been used for in-network coding [1, 2, 13, 14] as well. These papers use random walk with the Metropolis algorithm to disseminate packets from data nodes. More variations have also been introduced such that small node failures can be recovered locally [15], or more important data is recovered with higher probability. For all these schemes, the message cost is super linear.

Data collection by mules. Most prior work on using mobile mules for data collection has focused on the system issues [8, 10, 16, 23, 26]. The scheduling problem is more challenging if both latency and energy consumption are under consideration. A lot of interesting work has been done on mule scheduling [24, 25]. We will not survey them here, since our objective is to prepare the data in the network in a nice format such that data collection is easy even for a mule performing a random walk.

Gossip algorithms. Gossip algorithms have been extensively studied. See the survey [7] and the book [22] for references. There have been lots of variations depending on which node is selected to gossip to, and what information is exchanged between two gossip partners. Of particular relevance to our work is spatial gossip proposed by Kempe et al [12]. Our analysis uses a theorem proved in [12] to prove an upper bound on its convergence rate. The application of gossip algorithm in computing codewords in sensor network with near linear message cost, to our knowledge, has not been done before.

2 In-network Coding by Spatial Gossip

We assume that n sensors are in the network but only k of them have data to report at any moment. These nodes are called *data nodes* and the rest are denoted as *storage nodes*. We assume that the storage quota at each node is limited. For simplicity we assume that the data at a storage node almost hit the quota, so we can not put more data than one data symbol into a storage node.

Distributed erasure codes. *Erasure coding* transforms a message of m blocks into a longer message with M blocks, $M > m$, such that the original message can be recovered from a subset of the M blocks. In our case we use random linear codes over a finite Galois field $GF(q)$, $q = 2^b$. An original data piece is a vector of $GF(q)$ and is called a *symbol*. There are k symbols $\{s_1, s_2, \cdots, s_k\}$

distributed in the network. A *codeword* is a linear combination of the k symbols, denoted as $w = \sum_{i=1}^{k} \lambda_i s_i$, where λ_i's are coefficients. The calculation above is under the arithmetic of the Galois Field. The size of a codeword w is the same as the size of the symbols (b-bits long). The *degree* of a codeword w is the number of non-zero coefficients. Now suppose we have n codewords $w_1, w_2, \cdots, w_n$, with $w_j = \sum_{i=1}^{k} \lambda_{ij} s_i$. This coding scheme can be represented by the k by n generator matrix $G = \{\lambda_{ij}\}$. Take $w = (w_1, w_2, \cdots, w_n)$, $s = (s_1, s_2, \cdots, s_k)$. Thus $w = sG$.

The property required for decoding the symbols by using any k codewords is that any k columns from G form a full rank matrix G'. The decoding procedure is essentially solving a linear system $w' = sG'$ in $GF(q)$ and recover s, e.g., by Gaussian elimination.

The coefficients are taken as elements of $GF(q)$ as well and have b bits. The coefficients are delivered and stored along with the codeword. This causes storage/transmission overhead. However, we can make the overhead arbitrarily small by amortization over time by coding a long stream of data with the same set of coefficients.

Erasure code constructed by spatial gossip. We use spatial gossip to construct an erasure code in a distributed manner. At the beginning some subset of k nodes in the network has interesting data — the symbols s_i. We take t as the indicator of the number of rounds that have been executed. The current codeword at node j is denoted as $w_j^t = \sum_{i=1}^{k} \lambda_{ij}^t s_i$.

At round t, sensor x takes a random coefficient λ_x^t and updates its own codeword as $w_x \leftarrow \lambda_x^t \cdot w_x$. Suppose at this point w_x is $\sum_{i=1}^{k} \beta_i s_i$. The node x will choose a geographical location y^* and sends its current codeword w_x, as well as the current coefficients of the codeword $\{\beta_i\}$, to the node y closest to y^* (for example, by using geographical routing [11, 21]). In particular, y^* is selected with probability $p(x, y^*) = 1/(2\pi r^3)$, where $r = |xy^*| \geq 1$. Since the sensors are distributed nearly uniformly, the probability that a node y is chosen is also proportional to $1/|xy|^3$. The proof is given in [21] and is not repeated here.

A node at round t may receive multiple messages from different nodes. It simply stores the summation of the incoming data and its own codeword as its current codeword. The coefficients of w_j^t are updated by the summation of the according coefficients. Notice that the degree of the codeword at any node is monotonically non-decreasing. The following theorem has been proved in [12] regarding spatial gossip:

Theorem 1. *The symbol from a data node x is propagated to a node y with probability $1 - O(1/d)$ after $O(\log^{3.4} d)$ rounds, where $d = |xy|$.*

We run the spatial gossip algorithm for $m = O(\log^{3.4} n)$ rounds so that for each data node x and any storage node y, y's codeword contains the symbol from x with high probability $1 - O(1/n)$. Recall that at round t, the node x takes a random number λ_x^t and multiplies it with the current codeword. The coefficient for symbol i in the final erasure code at node j is the multiplication of m random numbers corresponding to the nodes on the path of propagation from i to j, and is null with a very small probability $O(1/n)$.

3 Data Recovery and Mule Collection

Data recovery upon node failures. The goal of the spatial gossip is to construct erasure codes on the sensors. The codewords from any k nodes can be used to recover the original symbols. That is, take the matrix G' such that the j-th column is the vector of the coefficients of the j-th node. We would like G' to have full rank. From the discussion in the previous section, each element of G' is 0 with very small probability $O(1/n)$ and is otherwise the multiplication of m random numbers. We show in the following theorem that the probability of G' having full rank is very high. The proof of the following theorem is similar to the one in [5].

Theorem 2. *Take the codewords from any k nodes in the network and the corresponding k by k generator matrix G', G' has full rank with high probability $(1 - k/q)c(k)$, where $c(k) \to 1$ when $k \to \infty$.*

Proof. G' is a matrix with each element as some random variable determined by the gossip process. We just need to show that the determinant of G' is not zero. There are two possibilities for $\det(G')$ to be zero. In the first case, $\det(G')$ is identically zero regardless of the random coefficients selected in the gossip algorithm – for example, the entries of one column are all zero. To analyze the chance for this to happen, we take a bipartite graph β on vertex set X and Y, where X represents the data nodes and Y represents the storage nodes taken to recover the original data. $|X| = |Y| = k$. There is an edge between vertex x_i and y_j if there is a non-zero element at position (i, j) of G'. Edmonds' Theorem [20] says that if there is a perfect matching in β then $\det(G')$ is not identically zero. By our gossip algorithm, each edge ij is present with probability $1 - O(1/n)$. By the same analysis as in [5] we can see that the graph β has a perfect matching with high probability. Thus $\det(G')$ is not identically zero w.h.p.

In the second case, the determinant of G' happens to be zero for the specific random coefficients that we choose. This is a rare event and the probability that this happens can be bounded with Schwartz-Zippel Theorem [20]. The degree

of $\det(G')$ is k. All the random coefficients are chosen from the field $GF(q)$. Thus prob$\{\det(G') = 0\} \leq k/q$.

In-network coding facilitating mule collection. When the data in the network is stored in terms of erasure codes, data collection by data mules is much easier. This scheme works very well with opportunistic data mules, whose movements are not under our control. As long as the opportunistic data mules visit k different nodes somehow, they can reconstruct the data from the entire network.

Moreover, with network coding a dedicated data mule does not need to know in advance which nodes have data and thus must be visited (in fact, it is impossible to know this in advance unless these data nodes report to the base station, in which case they can just report the data instead). It is also not necessary to plan very carefully what routes to follow so as to visit all the nodes in the network.

Online mule collection and data reconstruction. In the basic mule collection we first perform the in-network coding before mule collection. When events are time-critical, we would like to reconstruct data as soon as the mule collects something new. Initially we prefer data stored in its original form. At later data collection rounds it is difficult to discover a new data and higher degree symbols will become useful. We put a cap on the degree of the codewords in the gossip rounds. The cap grows with the number of gossip rounds.

What is the optimal mechanism to increase the cap of the degree of the codewords with the number of rounds remains an open question and our future work. For centralized LT codes, it has been proved that *Soliton* distribution [18] achieves the optimal rate and in practice the *robust Soliton* distribution has a more stable performance. We are not able to directly apply these results in our case due to the complex nature of nodes gossiping (codewords are moved around, instead of original data symbols). We learned through the simulations presented in next section that a practical scheme is to initially set the cap as a constant (in fact 1) to allow the data mule to collect enough original data, then let the cap grow as a linear function, so that the data mule can reconstruct data from almost each new data symbol.

4 Simulations

All the simulations were performed on sensor networks featuring 700 nodes including 100 data nodes, all uniformly randomly distributed inside a 50 by 50 square. The communication graph is modeled as a unit disk graph with communication range 3. For each data collection trial, a full reconstruction of data means that the base station received from the data mules enough codewords to be able to decode all the produced data.

In each set of simulations, $10 \sim 20$ sensor networks were randomly created. For each of them 100 different sets of data nodes were simulated. For each data production scenario $45 \sim 50$ data dissemination rounds were completed. After each round of data dissemination, 100 data collection experiments were performed, each one featuring a single data mule collecting codewords.

We evaluate different data collection strategies: *Random polling:* the data mule collects data from nodes selected uniformly at random in the network, this is implemented by the data mule doing random walk and picking up codewords during the motion; *Random straight line walk:* the data mule walks along a random line in the sensor network and collects data from the nodes that are closest to the line; *Clustered collection:* the data mule chooses a random location in the sensor network and collects data from near the location until k codewords are collected.

Communication cost and data recovery rate. We tested four different schemes, depending on what format of the data is exchanged and which node to send. A node with data may send either a single original data symbol or a codeword to a recipient. We denote the corresponding scheme by *non-coded gossip* or *coded gossip* method. In both cases, a storage node keeps a codeword (using random linear codes) of the incoming data packets. A node may send its data to a recipient chosen uniformly at random over all the nodes, or with some spatial distribution. We call the corresponding scheme *uniform gossip* or *spatial gossip*. Combining the choices, four different strategies are evaluated in our simulations. Specifically the non-coded uniform gossip method is the one in [5]. Our scheme is spatial coded gossip. The data collection scheme uses *clustered collection* in this set of experiments.

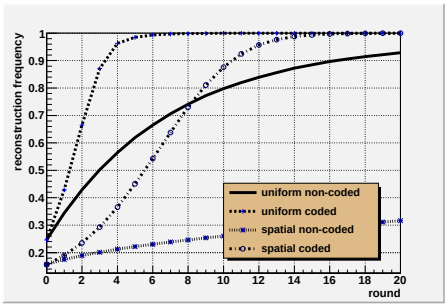

Fig. 1. Frequency of correct reconstruction with different data dissemination strategies, against round number.

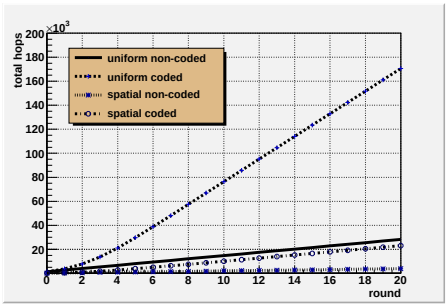

Fig. 2. Total number of communication hops for different data dissemination strategies, against round number.

Figure 1 shows data recovery rate. The uniform non-coded gossip strategy initially performs better, similarly to the uniform coded gossip strategy. Then the spatial coded gossip strategy outperforms the uniform non-coded gossip strategy

and behaves similar to the uniform coded gossip strategy. The number of rounds that were necessary to perform data dissemination for uniform non-coded gossip is 230 (not shown in the figure). 20 rounds are enough to reconstruct data most of the times for spatial coded and non-coded gossip strategies. Figure 2 shows the total communication cost to move data around, against the round number. The uniform coded gossip strategy, with the best data recovery performance, is extremely expensive, as almost everyone sends data to random others. In summary, our scheme has the best combined performance than the other strategies in terms of lower message cost and higher data recovery rate.

Data collection strategies. We tested the three different data collection strategies introduced at beginning of this section. Figure 3 shows the frequency that the mules were able to correctly reconstruct all the data, against the number of rounds that were performed. The random polling method is the most expensive one for the data mule. It also has the highest reconstruction rate with a smaller standard deviation when the number of gossip rounds is small. Nevertheless, when we execute about 18 gossip rounds, all three data collection movement schemes can fully reconstruct the network data for all the trials we tested. This shows the erasure codes constructed with spatial gossip allow for flexible choice of data mule movement patterns.

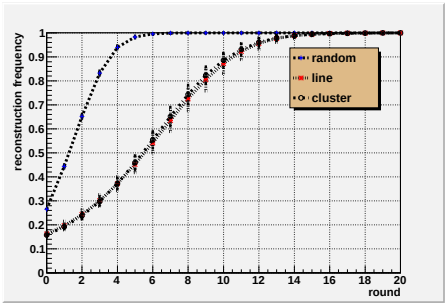

Fig. 3. Frequency of correct reconstruction with different collection strategies, against round number.

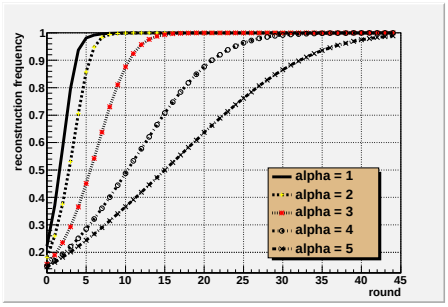

Fig. 4. Frequency of correct reconstruction with different Spatial Gossip exponents, against round number.

Varying the exponent for spatial gossip. We vary the exponent of the spatial distribution: in each round each node p selects one receiver node q, with probability proportional to $1/(d_{i,j})^{\alpha}$, and the analyzed αs were $1, 2, 3, 4, 5$. The simulations aim at finding out which exponent performs better in terms of reconstruction frequency and communication cost.

Figure 4 shows the frequency that the mules were able to correctly reconstruct all the data, against the round number. Figure 5 shows the total communication cost to move data around, against the round number. With a smaller α, the data storage scheme has higher communication cost and higher recon-

struction probability. The communication cost that the network incurs to ensure correct data reconstruction is presented in the table in Figure 6. With the same reconstruction performance, the total message cost is the smallest when $\alpha = 3$, which corroborates our theoretical analysis.

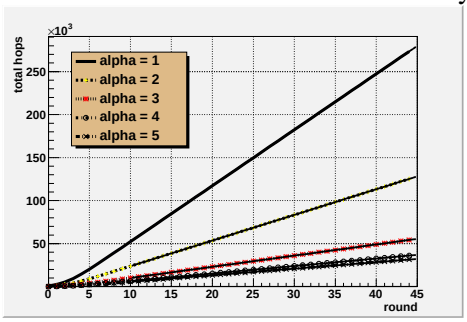

Fig. 5. Total number of communication hops with different Spatial Gossip exponents, against round number.

α	cost per round	# rounds for successful reconstruction	total cost
1	6500	10	65000
2	2980	11	32780
3	1300	19	24700
4	867	39	33813
5	758	50	37900

Fig. 6. Total communication cost for correct data reconstruction with different Spatial Gossip exponents.

Online reconstruction. We introduce a network wide parameter called the "maximum codeword degree" to restrict the degree of the codeword at each round. The parameter can be changed after every round. Figure 7 shows this parameter as a function of the gossip round used in the simulation. The parameter is 1 for the first 40 rounds, then it increases in a sublinear manner up to round 80, after which it becomes a linear function of the round number.

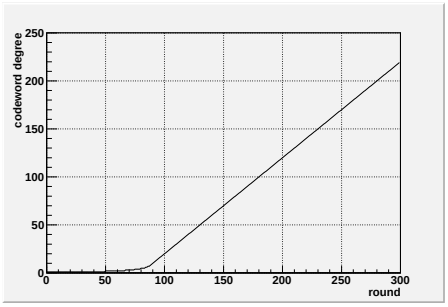

Fig. 7. Maximum codeword degree as function of the current gossiping round.

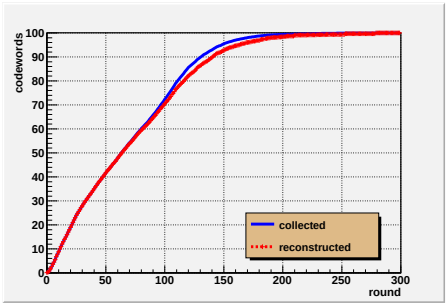

Fig. 8. Number of data reconstructed vs collected against the round number.

From the point of view of the sensors, the algorithm is the same for each node. A node has a "current codeword", and a storage list. Initially, the "current codeword" is initialized to a produced data if the sensor is a data node, else it is initialized to null. The storage list is initialized with the "current codeword". In each gossip round, each sensor selects a recipient with spatial distribution, then it sends to it its "current codeword". Each sensor puts all the received codewords into its storage list, then it creates its next "current codeword" combining ele-

ments of the storage list. Elements chosen at random from the storage list are combined to create a new "current codeword" that has a degree equal or less to the "maximum codeword degree". After each gossip and random walk round, the data mule downloads from a sensor all the codewords it has into its storage list, then the data mule reconstructs all the original data symbols it can.

Figure 8 compares the number of reconstructed data against the number of independent codewords collected by the data mule, for each round. Simulations show that this strategy is able to collect a large number of data as soon as codewords are downloaded. On the other hand, the data mule is less efficient in reconstructing all the data than in the previous scenario, since it needs to collect more than 100 codewords to complete its job. We put a cap on the current codeword's degree, and this leads to a slowdown in information dispersal with respect to a scenario where the codeword degree is as high as it can get. The maximum degree of codewords initially stays 1, to help with the online reconstruction, since a data mule can extract automatically original data from codewords of degree 1. Later the maximum codeword degree increases quickly, thus the mule can cope with high degree codewords leveraging on all the original data it has already reconstructed.

Future work. In our future work, we would like to explore two directions. First, what is the optimal strategy for the cap of the codeword degree for online data reconstruction? Second, if the network data has spatial correlation, how do we exploit it in the network coding scheme?

References

1. S. A. Aly, Z. Kong, and E. Soljanin. Fountain codes based distributed storage algorithms for large-scale wireless sensor networks. In *IPSN '08: Proceedings of the 7th international conference on Information processing in sensor networks*, pages 171–182, 2008.
2. S. A. Aly, Z. Kong, and E. Soljanin. Raptor codes based distributed storage algorithms for wireless sensor networks. In *Proc. of IEEE International Symposium on Information Theory*, pages 2051–2055, July 2008.
3. C. Avin and C. Brito. Efficient and robust query processing in dynamic environments using random walk techniques. In *IPSN '04: Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 277–286, New York, NY, USA, 2004. ACM.
4. T. Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34:209–219, June 2006.
5. A. G. Dimakis, V. Prabhakaran, and K. Ramchandran. Ubiquitous access to distributed data in large-scale sensor networks through decentralized erasure codes. In *Proc. Symposium on Information Processing in Sensor Networks (IPSN'05)*, pages 111–117, April 2005.
6. J. Gao, L. J. Guibas, J. Hershberger, and N. Milosavljević. Sparse data aggregation in sensor networks. In *Proc. of the International Conference on Information Processing in Sensor Networks (IPSN'07)*, pages 430–439, April 2007.

7. S. M. Hedetniemi, S. T. Hedetniemi, and A. Liestman. A survey of gossiping and broadcasting in communication networks. *Networks*, 18(4):319–349, 1988.
8. D. Jea, A. A. Somasundara, and M. B. Srivastava. Multiple controlled mobile elements (data mules) for data collection in sensor networks. In *IEEE International Conference on Distributed Computing in Sensor System (DCOSS)*, pages 244–257, 2005.
9. A. Kamra, V. Misra, J. Feldman, and D. Rubenstein. Growth codes: maximizing sensor network data persistence. In *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 255–266, New York, NY, USA, 2006. ACM.
10. A. Kansal, M. Rahimi, W. J. Kaiser, M. B. Srivastava, G. J. Pottie, and D. Estrin. Controlled mobility for sustainable wireless networks. In *IEEE Sensor and Ad Hoc Communications and Networks (SECON'04)*, 2004.
11. B. Karp and H. Kung. GPSR: Greedy perimeter stateless routing for wireless networks. In *Proc. of the ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom)*, pages 243–254, 2000.
12. D. Kempe, J. Kleinberg, and A. Demers. Spatial gossip and resource location protocols. In *STOC '01: Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 163–172, New York, NY, USA, 2001. ACM Press.
13. Y. Lin, B. Li, and B. Liang. Differentiated data persistence with priority random linear codes. In *ICDCS '07: Proceedings of the 27th International Conference on Distributed Computing Systems*, page 47, Washington, DC, USA, 2007. IEEE Computer Society.
14. Y. Lin, B. Liang, and B. Li. Data persistence in large-scale sensor networks with decentralized fountain codes. In *Proc. of the 26th IEEE INFOCOM07*, May 2007.
15. Y. Lin, B. Liang, and B. Li. Geometric random linear codes in sensor networks. In *Proc. IEEE International Conference on Communications ICC '08*, pages 2298–2303, 19–23 May 2008.
16. W. Lindner and S. Madden. Data management issues in periodically disconnected sensor networks. In *Proceedings of Workshop on Sensor Networks at Informatik*, 2004.
17. L. Lovasz. Random walks on graphs: A survey. *Bolyai Soc. Math. Stud.*, 2:353–397, 1996.
18. M. Luby. Lt codes. In *FOCS '02: Proceedings of the 43rd Symposium on Foundations of Computer Science*, page 271, Washington, DC, USA, 2002. IEEE Computer Society.
19. S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. TAG: a tiny aggregation service for ad-hoc sensor networks. *SIGOPS Oper. Syst. Rev.*, 36(SI):131–146, 2002.
20. R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.
21. R. Sarkar, X. Zhu, and J. Gao. Hierarchical spatial gossip for multi-resolution representations in sensor networks. In *Proc. of the International Conference on Information Processing in Sensor Networks (IPSN'07)*, pages 420–429, April 2007.
22. D. Shah. *Gossip Algorithms*. Foundations and Trends in Networking. Now Publishers Inc, 2008.
23. R. Shah, S. Roy, S. Jain, and W. Brunette. Data MULEs: Modeling a three-tier architecture for sparse sensor networks. In *IEEE SNPA Workshop*, May 2003.
24. R. Sugihara and R. K. Gupta. Improving the data delivery latency in sensor networks with controlled mobility. In *DCOSS '08: Proceedings of the 4th IEEE international conference on Distributed Computing in Sensor Systems*, pages 386–399, 2008.
25. R. Sugihara and R. K. Gupta. Optimizing energy-latency trade-off in sensor networks with controlled mobility. In *INFOCOM'09*, 2009.
26. Z. Vincze and R. Vida. Multi-hop wireless sensor networks with mobile sink. In *CoNEXT'05: Proceedings of the 2005 ACM conference on Emerging network experiment and technology*, pages 302–303, New York, NY, USA, 2005. ACM Press.