Data Collection for Distributed Surveillance Sensor Networks in Disaster-hit Regions

Yao Zhao, Xin Wang, Jin Zhao
School of Computer Science
Fudan University, China
Email: {082024092, xinw, jzhao}@fudan.edu.cn

Azman Osman Lim
School of Information Science
Japan Advanced Institute of Science and Technology, Japan
Email: aolim@jaist.ac.jp

Abstract— **The objective of many applications with the surveillance missions in wireless sensor networks is to provide long-term monitoring of the specific environments, such as disaster-hit regions. These applications usually perform continuous monitoring without any maintenance, even if some sensor nodes fail. A significant challenge when designing the data collection approaches for such systems is that the conventional communication protocols for wireless sensor networks would present low efficiency, since the network topology changes rapidly due to the node failure. Thus the sensor nodes in such systems should use an automatic transmission approach to disseminate their sensed data to the sink in a distributed manner. In this paper, we propose a novel *C*oding-based *P*robabilistic *R*outing (*CPR*) to address this specific problem of data collection for distributed surveillance sensor networks in disaster-hit regions. CPR dynamically adapts to node failure to collect the maximum data in any given time and chooses an optimal probabilistic routing to decrease the transmission consumption. The extensive simulation results are presented to show that CPR outperforms other strategies.**

I. Introduction

In wireless sensor networks, many applications such as [1–5] have targeted to provide long-term monitoring of the specific environments, such as disaster-hit regions and hostile areas. In such applications, the sensor nodes which store the valuable data would fail frequently due to the disaster or enemy's breaking, and the routing trees would easily catch failure. However, these unpredictable failures are especially troubling because of the potential loss of valuable data collected by the sensor nodes.

During the process of data collection in such disaster-hit applications, the sink ought to maximize the amount of received data. However, the setup of routing trees usually produces much long delay, which implies the sink has to waste long time to wait for the setup of routing trees. It is contrary to the objective of maximizing the received data during limited time. In addition, each sensor node monitored the environment would be treated as a source, thus there would be quite a lot of routing trees. Moreover, the constructed routing trees may be likely to catch failure, especially in a disaster or hostile area. Thus it takes much consumption to maintain the routing trees and the conventional communication protocols for wireless sensor networks would present low efficiency in such systems.

Once the sink comes into the surveillance area, it sends out the query request, and then the nodes try to deliver their stored data to the sink. Often, the rate of data generated within a sensor network greatly exceeds the capacity available to deliver the data to the data sinks [6]. With so many nodes trying to channel data to the sink, there may be congestion and delay in the neighborhood around the sink, thus much of the sensed data trying to reach the sink will be stalled. During this time, data is especially vulnerable to loss if the nodes storing the data catch failure due to the disaster. Though some forwarding transmission of sensed data does not increase the rate at which data moves toward the sink, it does increase the likelihood that such data will survive as some nodes fail. Thus, when there is a chance to disseminate a packet, how to decide the transmission direction ("forward" or "backward") becomes a challenge when designing the probabilistic routing.

Though there are proposed methods to reduce the data congestion, they are not suitable for our problem. In [7], data congestion is detected by sensors signaling their upstream neighbors via a backpressure mechanism. This method does not take advantage of transmitting "backward" to increase the probability of combining a packet with others. In [8], congestion avoidance is achieved by data aggregation. However, the original data is needed in our problem. In [9], the authors suggest to use predictive routing to decrease the data congestion, but they assume a static sensor network.

A kind of coding strategies, called *Growth Codes* [10], employ a dynamically changing codeword degree distribution and promise to be able to decode a substantial number of the codewords at any given time. Growth Codes are especially useful in wireless sensor network where there is no information about the sink node. However, the original Growth Codes are not suitable for data collection of disaster-hit sensor networks we described above. On one hand, when using Growth Codes, the knowledge of the total number of sensor nodes is required for calculating the specified degree distribution. It is not practical in the applications of disaster sensor networks, since these networks should keep long-term running and thus the total node number would gradually decrease and become unknown to each individual node. On the other hand, the data collection problem we described above is query-based, the sensor nodes would know the direction of the sink after the query-broadcast phase.

In this paper, we propose a coding-based probabilistic routing (*CPR*), to address the problem of data collection for distributed surveillance sensor networks in disaster-hit regions.

COLLABORATECOM 2010, October 9-12, Chicago, USA

DOI 10.4108/icst.collaboratecom.2010.45

CPR chooses an optimal probabilistic routing to decrease the total transmission consumption and dynamically adapts to node failure to collect the maximum data in any given time.

The first contribution of CPR is to determine the optimal probabilistic routing. The proposed optimal transmission scheme strikes a balance between the incoming and outgoing data for the sink's neighborhood, thus can remove the heavy data congestion around the sink. Meanwhile, it is sufficient for CPR to facilitate mixing and encoding of symbols. By choosing the optimal transmission scheme, CPR maintains the same level of data collection but introduces lower transmission cost compared to the original Growth Codes.

The second contribution of CPR is to search the near-optimal degree transition points to avoid the performance degradation as the total node number decreases. Here, *degree* is referred to as the number of symbols encoded together to form a codeword and *degree transition points* is referred to as the time when a node should choose a higher codeword degree. In CPR, the sensors can dynamically chooses their optimal degree transition points when some neighbors are found to catch failure, since node failure leads to the dynamic degree transition points. By choosing the near-optimal degree transition points, CPR maintains a higher level of data collection compared to the original Growth Codes [10] and No Coding.

The rest of the paper is organized as follows: Section II introduces some previous related work. Section III describes the network model and provides the motivation of CPR. Section IV describes the design of CPR in detail, including a complete form of the proposed algorithm. Section V evaluates the performance of CPR by simulation. Finally, Section VI concludes this paper.

II. Related Work

Without coding, sensor nodes simply exchange the original symbols, so that the sink would be likely to receive many duplicates. In the well studied Coupon Collectors Problem, it has been shown that if the N original symbols are generated uniformly randomly, the sink needs to receive approximately $O(NlogN)$ symbols to recover all N original symbols. Network coding [11] is used for data storage in [12], the authors show that a simple distribution scheme using network coding and only based on local information can perform almost as well as the case where there is complete coordination among nodes.

In [13], a kind of decentralized erasure codes is proposed. Assuming a scenario where there are n storage nodes with limited memory, among them, k ($k < n$) sources can generate the data, the authors show the sink is able to retrieve all the data by querying any k nodes. Decentralized erasure codes are optimally sparse and lead to reduced communication, storage and computation costs over random linear coding. The algorithms for the extreme case where each storage node can only store a single information unit is investigated in [14]. Data is pre-routed to $log(N)$ of the N storage nodes and the storage nodes simply combine the incoming information with their existing information.

Persistence and reliability of cached data can be improved through Decentralized Fountain Codes [15] [16]. The authors address the problem that how to collectively store sensed data when the sink is not present in the network. They use Belief Propagation for the reason of low decoding complexity and Random Walks to disseminate coded data in a scalable way. The authors argue that all data could be recovered as long as a sufficient number of sensor nodes are alive. Channel codes such as *LT Codes* [17] start decoding only after accumulating a large number of received packets, is not suitable for our disaster sensor networks since the resource constrained sensor nodes are prone to fail at any time.

Growth Codes [10] are specifically designed to increase data persistence, *i.e.*, to maximize the amount of symbols that can be recovered at the sink at any time. Nodes send out one codeword to one random neighbor when there is a transmission chance, the neighbor node then combines received codeword with its stored local codeword(s). In Growth Codes, the number of original symbol units a codeword is coded over is referred to as *degree*. The authors propose to transmit codewords of degree one at early time, then gradually increase the degree with more and more codewords being received by the sink (hence the name *Growth Codes*). One drawback of Growth codes is that the optimal transition points, at which the codes switch to higher complexity codewords and which are pre-defined before disposing sensor networks, do not adjust adaptively even if parts of nodes fail. So that it is a critical challenge for Growth Codes to be migrated into the application of data collection for distributed surveillance sensor networks in disaster-hit regions.

The design of Growth Codes is generalized to the multi-snapshot scenarios in [18]. The authors aim to maximize the expected utility gain through joint coding and scheduling and propose two algorithms, with and without mixing different snapshots. When no mixing is used, it needs a schedule to improve the total data utility. They formalize the scheduling problem into a Multi-Armed Bandit Problem so as to derive the optimal solution using Gittins Indices and identify conditions under which a greedy algorithm is optimal. To improve Growth Codes under scenarios of different node mobility, resilient coding algorithms [19] [20] are proposed. They investigate the suitability of different codeword degree distributions with respect to the dynamics of the underlying wireless network and design a corresponding data management algorithm. To improve the performance of Growth Codes in large-scale sensor networks, a joint scheduling of packet encoding and priority broadcast is proposed in [21].

Notice that, the above coding strategies are designed for data storage or data persistence in different aspects but do not consider the problem of data collection for distributed surveillance sensor networks in disaster-hit regions. The goal of our proposed method of CPR is to investigate techniques to maximize the received data for data collection in the scenarios where the sink would send out the query-request but the stored nodes failed rapidly.

III. Problem Description

A. Network Model

We start with the description of the network model. A sensor network consists of N sensors distributed randomly in a monitored region, each sensor node has a unique identifier (ID) and is capable of sensing an area around itself (called the sensing region). Each sensor node also has a radio interface to communicate directly with neighboring sensors. A query from the sink would ask for a summarization of some sensed data over some time window. Such queries typically run multiple times, periodically for different time windows.

Given a data query over a sensor network, a naive way is to simply flood the network with the query. We add an item of hop number into the query message, which indicates the number of hops between the sink and a sensor node. Initially, the query message has a hop number of zero. Each intermediate node in the network increases the item of the hop number by one and then broadcasts the query message. During this query flooding phase, the sensors will get their hop number in a breadth-first manner. The parameter of hop number is used to estimate which node is "nearer" to the sink and which is "farther" from the sink.

As a result, the neighbor set of one node can be divided into three subsets: N_1 consists of neighbors with a smaller hop number, N_2 with a same hop number while N_3 with a larger hop number. Notice that, though some other methods of more efficient query execution than the above flooding approach were presented, such as the works in [22], there are also easy manners for the sensors to get their approximate hop number during the query execution phase. Moreover, even though the hop number changes when several intermediate sensors fail, the relationship of which is nearer to the sink between two nodes would not be changes drastically.

B. Transmission Scheme

This section addresses the problem of what affects the selection of transmitting "forward" or "backward". When there is a transmitting chance, a node needs to choose one neighbor as the receiver. We distinguish the probabilities of transmitting to different subsets: the probability of choosing a node in N_1 (*i.e.*, "forward") is p_1, the probability of choosing a node in N_2 is p_2, and the probability of choosing a node in N_3 (*i.e.*, "backward") is p_3. Growth Codes select $p_1 : p_2 : p_3$ as $1 : 1 : 1$, since they were designed for emergence applications where the sensors have little chance to know the direction of the sink. However, when addressing our problem of data collection in disaster scenarios, we should choose a more efficient transmission scheme. In this paper, we try to search the optimal transmission scheme of the distributed manner, which can be considered as a kind of probabilistic routing approach.

Intuitively, transmitting to N_1 ("forward") increases the probability of delivering a packet to the sink using fewer links. However, strict ("forward") would reinforce the funneling effect, a phenomenon that there is heavy congestion and long delay in the neighborhood of the sink when too many nodes try to channel data.

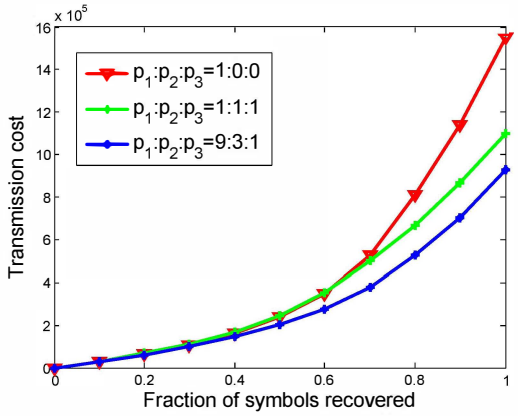


Fig. 1: Transmission consumption to recover certain fraction of symbols under different $p_1 : p_2 : p_3$ in a 500 node random network with node density of 15.

Here, we perform a simulation to observe that how the transmission scheme affects the total transmission consumption, where the simulation settings will be described in detail in Section V. Three transmission schemes (*i.e.*, $p_1 : p_2 : p_3$) are considered — $1 : 1 : 1$ (random way), $1 : 0 : 0$ (strict "forward") and $9 : 3 : 1$. Fig. 1 depicts the simulation results. Strict "forward" is not the optimal transmission scheme, since it reinforces the funneling effect. Thus, the optimal transmission is a tradeoff between strict "forward" and a random way.

IV. Design of CPR

Taking advantage of the information of hop number and the local neighbor table, we propose a coding-based probabilistic routing (*CPR*) to address the problem of data collection for distributed surveillance sensor networks in disaster-hit regions. First of all, we summarize the key factors a distributed protocol should pay attention to.

Data Congestion: As many nodes try to deliver their data to the sink and limited buffer size, there may be a heavy congestion in the neighborhood of the sink. Limited to the buffer size, a part of incoming data has to be deserted. Thus, a good protocol for sensor network should try to avoid heavy data congestion.

Node Failure: Node failure is inevitable in many applications, especially in a disaster or hostile environment. If some nodes fail, the total number of nodes N will decrease. Since the sensor nodes are deployed in a distributed manner, it is not feasible to update this global information of N to the whole network. Existing coding approaches of sensor networks, such as Growth Codes [10], fail to adjust adaptively and thus lead to a performance decrease.

Energy Consumption: The limited energy supply of sensing devices in sensor networks demands low energy consumption. Since data sending and receiving is a main consumption

of energy, the total transmission consumption should be minimized. In this paper, we assume that when any node delivers a packet, the total transmission consumption increases by 1.

A. Protocol Overview

In our protocol, we assume that there are N nodes in the network where each node takes a single snapshot of their environment and generates one unit of data message that can fit into a packet. For the purpose of decreasing total transmission cost, nodes do not exchange symbols until a query request is notified. Once the sink comes into sensing area, it disseminates a query message and then sensor nodes in the network start the performing of CPR.

Now, we regard $p_1 : p_2 : p_3$ defined in Section III as $\beta : \alpha : 1$. Furthermore, since it is difficult to specify the appropriate values of two variables (β and α) respectively in different scenarios, we simplified assume β as α^2.

On account of the effect that too many nodes are willing to deliver data to the sink, neighborhood of the sink may receive much data that exceed their transmitting capacity. As the process of data collection goes, more and more data will be stalled in the buffer and lead to data congestion. A quantitative measurement of data congestion of a sensor node is defined to be the quotient of received packet number and sent packet number. Based on this measurement, we perform a simulation of 500 nodes random network to analyze the relationship of congestion and node density. Fig. 2 demonstrates that it is the transmission scheme (*i.e.*, α) that mainly results in the data congestion rather than the other factors, such as node density.

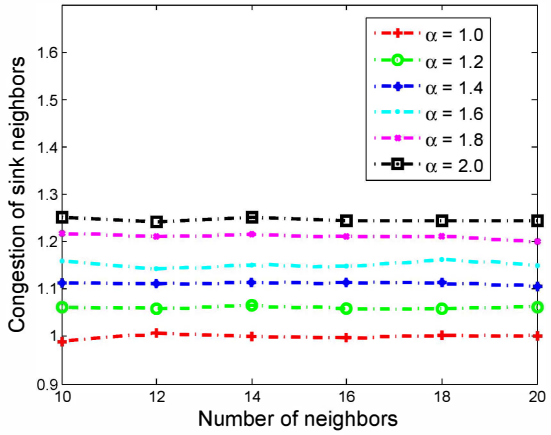


Fig. 2: Congestion changes little with node density but varies much with α.

It is not hard to understand that why node density does not affect congestion much. Taking an example, we compare two cases where α is fixed to be 1: (i) node A is located in a network of 10 neighbors in average, (ii) node B is located in a network of 20 neighbors in average. In the first case, the probability that A is chosen as the receiver by any of its neighbors is $1/10$, so that A receives $10{\times}1/10 = 1$ packet in expectation. In the second case, the probability that B is chosen as the receiver by any of its neighbors is $1/20$, so that B receives $20{\times}1/20 = 1$ packet in expectation, as well. As long as the transmission scheme (*i.e.*, α) is fixed, a single node will not receive any more data during a time slot. Therefore, by choosing an appropriate value of α, CPR maintains the same level of data collection as Growth Codes, while introducing lower transmission and avoiding the data congestion around the sink.

Growth Codes calculate the degree transition points $K_{i,N}$, but ignore the fact that when N decreases as parts of network fail, $K_{i,N}$ should also decrease, thus the degree transition points ought to be gradually advanced as N decreases. Notice that, this kind of "advance" is not important for Growth Codes but is indeed important for protocols designed for data collection in disaster-hit sensor networks. The reason is that Growth Codes are specially defined for high dynamical scenarios and the sensor nodes in these applications do not need to continuously work for a very long time. However, in the application of disaster-hit regions where sensor nodes need to work for a very long time, the node number N would inevitably decrease drastically at different time.

Now, we summarize the basic flow of the process of CPR: (i) nodes calculate the degree transition points as is shown in [10]; (ii) nodes modify the degree transition points as parts of nodes are found to catch failure; (iii) nodes adaptively determine the optimal transmission scheme (*i.e.*, α). The choosing of α and the modification of $K_{i,N}$ is described in detail in the next subsections, and the complete algorithm of CPR is given at the end of this section.

B. Determining Optimal Transmission Scheme

Clearly, nodes with larger buffer tolerate to data congestion better than ones with smaller buffer. To recover all N symbols, $K_{N,N}$ codewords are required to be received by the sink in expectation. Let con be the average value of the quotient of the number of received packets and the number of transmitted packets during one time slot, then each node may buffer $con - 1$ packets during one time slot. If there is heavy data congestion of some intermediate nodes, some packets should be deserted and removed from the buffer due to the limited buffer capacity. Let B be the buffer size of each sensor node.

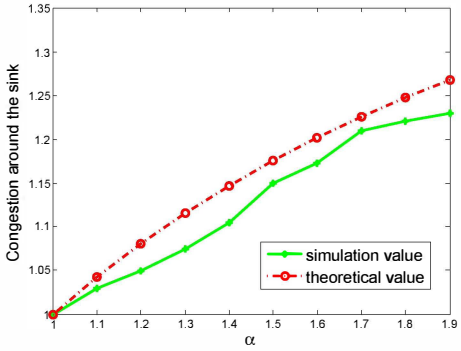
Since one node stores at most B packets in the buffer and sends out a random packet from the buffer at one time slot, according to the Coupon Collector Problem, it will take $Blog_2B$ time slots to disseminate all the B packets. During these time slots, any incoming packets would be little useful in some sense. Thus, to avoid the useless data desertion, the value of con should satisfy the following inequation:

$$(con - 1){\times}Blog_2B \leq B. \tag{1}$$

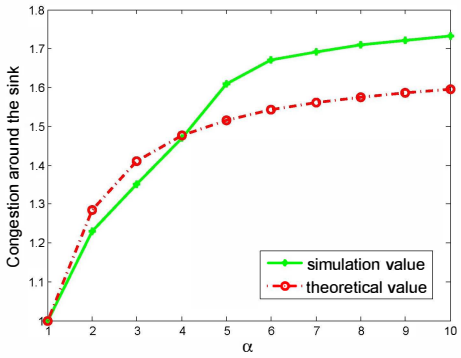
Assume con_{max} to be the largest value of con, then we get

$$con_{max} = \frac{1}{log_2B} + 1. \tag{2}$$

To determine the appropriate value of α, we first consider a centering network model where all the nodes are uniformly distributed in a circle filed and the sink is located at the center



(a) When 1≤con≤1.3.



(b) When 1≤con≤1.8.

Fig. 3: The optimal α at different data congestion in a 500 node random network.

position. Then, we consider a complete random network where both the sink and other nodes are randomly located in the network.

Here, the centering model can be thought as the theoretical value of the optimal transmission scheme, while the random model can be thought as the simulation value.

1) Centering Model: In a centering model, the sink is located in the center of a circle field and other nodes are randomly located. Assume the density of nodes to be d, then the average number of nodes with hop number k is $\pi[k^2-(k-1)^2]\times d$. Thus, the expected number of packets received at the nodes with hop number $k(k\geq 2)$ is given by

$$\frac{\pi d\alpha^2}{1+\alpha+\alpha^2}[(k+1)^2-k^2]+\frac{\pi d\alpha}{1+\alpha+\alpha^2}[k^2-(k-1)^2] + \frac{\pi d}{1+\alpha+\alpha^2}[(k-1)^2-(k-2)^2]. \quad (3)$$

The average number of transmitted packets is obtained as

$$\pi(2k-1)\times d. \quad (4)$$

Therefore, the average congestion is the quotient of Eq. (3) and Eq. (4), which can be transformed as follows:

$$1+\frac{\frac{\alpha^2-\alpha-3}{\alpha^2+\alpha+1}+1}{2k-1}. \quad (5)$$

As k $(k\geq 2)$ decreases, data congestion increases, so we especially consider the maximum congestion when k is 2. When $k=2$, then we get con_{max}

$$con_{max}=\frac{\alpha^2-\alpha-3}{3(\alpha^2+\alpha+1)}+\frac{4}{3}. \quad (6)$$

Settling Eq. (6), then we get

$$\alpha=\frac{(3con_{max}-3)+\sqrt{-27con_{max}^2+54con_{max}-11}}{2(5-3con_{max})}. \quad (7)$$

From Eq. (2), we know that con_{max} is $\frac{1}{log_2 B}+1$. Let θ be con_{max}, the appropriate value of α is determined as follows:

$$\alpha=\frac{(3\theta-3)+\sqrt{-27\theta^2+54\theta-11}}{2(5-3\theta)}. \quad (8)$$

2) Complete Random Network: In a complete random network where both the sink and other nodes are randomly located, it is hard to get the exact transmission scheme α by formulation. Therefore, we turn to derive some empirical values from simulation.

When performing our simulation, we choose a 500 node network with density of 15 neighbors on average, the sink is randomly attached to one sensor node. Since node density does not increase/decrease the congestion much, our selected setting of 15 neighbors on average is reasonable and representative.

Transmitting "forward" helps to reduce the total transmission cost, but increases data congestion around the sink. Once the congestion exceeds the buffer capacity, a part of data is thrown and removed from the buffer. Moreover, transmitting "backward" helps to mix the symbols. The optimal α enables us to balance the tradeoff between transmission cost and data congestion.

Fig. 3 depicts the optimal α at different data congestion within different ranges. Respectively, Fig. 3(a) depicts the result that when $1\leq con\leq 1.25$, the optimal α (*i.e.*, the optimal transmission scheme) should be between 1 and 2, Fig. 3(b) depicts that when $1\leq con\leq 1.8$, a wider range, the optimal α should be corresponding different. As a result, nodes in a complete random network can adaptively determine an appropriate α according to the empirical values derived from these two figures.

For practical issue, we give the resulting decision in TABLE I, which can be pre-stored before disposing the sensor nodes.

C. Determining Near-Optimal Degree Transition Points

1) Degree Transition Points of Growth Codes: The authors of [10] suggest that codewords in the network start with degree one and then gradually increase the degree over time. Growth Codes define the degree transition points as time points when a node should send out a codeword with a higher degree and give the optimal degree transition points as follows.

Let $R_{i,N}$ represent the number of symbols recovered by a sink when codewords of size greater than i provide a greater likelihood for providing recovery than those of degree less than i in a N node network.

$$R_{1,N}=\frac{N-1}{2},\ldots,R_{i,N}=\frac{iN-1}{i+1} \quad (9)$$

TABLE I: The optimal α in different data congestion.

con	1	1.05	1.1	1.15	1.2	1.3	1.4	1.5	1.6	1.7	1.8
α	1	1.2	1.35	1.5	1.65	2.5	3.2	4	5	6.5	10

To recover $R_{j,N}$= $\frac{jN-1}{j+1}$ symbols, at most $K_{j,N}$ codewords are required in expectation.

$$K_{j,N} = K_{j-1,N} + \sum_{i=R_{(j-1),N}}^{R_{j,N}-1} \frac{\binom{N}{j}}{\binom{i}{j-1}(N-i)} \tag{10}$$

2) Degree Transition Points of CPR: Growth Codes require $K_{N,N}$ codewords to recover all the symbols. In the application of data collection in disaster scenarios where sensor nodes need to work for a very long time, the node number N would inevitably decrease drastically at different time. Since it is unrealistic to get the global information of N, the decentralized nodes in sensor networks need to estimate the updated degree transition points $K_{i,N}$ if parts of network fail. In this subsection, we describe how to estimate the advance of degree transition points $K_{i,N}$. Here, we refer to the term of "adv" as the number of time slots that should be advanced for each sensor node to do certain coding operation.

When a node finds one of its neighbors fail, it implies that the advance value of each active node is $K_{i,N}-K_{i,N-1}$, since there are only $N-1$ active nodes. Thus the total advance of the whole network is $(K_{i,N}-K_{i,N-1})(N-1)$. Assume that $avgNeighbor$ is the average number of neighbors per node, so that an incident of one time of node failure can be found by $avgNeighbor$ nodes in average. Furthermore, we replace $avgNeighbor$ with $N_1+N_2+N_3$, since the former parameter is hard to get.

Therefore, when a node finds one of its neighbors failed, its average advance value (termed as adv) is:

$$adv = \frac{(K_{i,N}-K_{i,N-1})(N-1)}{N_1+N_2+N_3} \tag{11}$$

Though Eq. (11) describes the advance value of degree transition points, it is impractical for some kinds of sensor networks to calculate this value, since they may have much low computation capacity and $K_{i,N}$ involves the calculation of the combination number. So that, we need to estimate the advance value in Eq. (11).

Notice that, the right component of Eq. (10) is trivial if $R_{(i-1),N} \geq R_{i,N}-1$.

$$\begin{aligned} & R_{(i-1),N} > R_{i,N}-1 \\ \Rightarrow \quad & \frac{(i-1)N-1}{i} > \frac{iN-1}{i+1} - 1 \\ \Rightarrow \quad & i > \frac{\sqrt{4N+5}-1}{2} \approx \sqrt{N} \end{aligned} \tag{12}$$

When $i=1$, Growth Codes are equal to the coupon collector problem. So we can use the settlement of coupon collector problem to get the value of $K_{1,N}$. To recover $R_{1,N}$ symbols, the collecting times is calculated as below (the parameter γ below refers to the Euler-Mascheroni constant).

$$\begin{aligned} k_{1,N} &= \frac{N}{N} + \frac{N}{N-1} + \ldots + \frac{N}{N+1-R_{1,N}} \\ &= N[(\frac{1}{1} + \frac{1}{2} + \ldots + \frac{1}{N}) \\ &\quad -(\frac{1}{1} + \frac{1}{2} + \ldots + \frac{1}{N-R_{1,N}})] \\ &\approx N[(\ln N + \gamma) - (\ln(N-R_{1,N}) + \gamma)] \\ &= N \ln \frac{N}{N - \frac{N-1}{2}} \\ &= N \ln \frac{2N}{N+1} \end{aligned} \tag{13}$$

Let $p_{r,i}$ be the probability of successfully decoding a randomly chosen symbol of degree i when r symbols have already been recovered, then $p_{r,i}$ = $\frac{\binom{r}{i-1}(N-1)}{\binom{N}{i}}$. Thus,

$$\frac{p_{r+1,i}}{p_{r,i}} = \frac{\frac{\binom{r+1}{i-1}(N-1)}{\binom{N}{i}}}{\frac{\binom{r}{i-1}(N-1)}{\binom{N}{i}}} = \frac{r}{r+2-i}. \tag{14}$$

When $2 \leq i \leq \sqrt{N}$, $r \geq \frac{N-1}{2}$, we can know

$$\frac{p_{r+1,i}}{p_{r,i}} \approx 1. \tag{15}$$

Eq. (15) implies that when $2 \leq i \leq \sqrt{N}$, $p_{r,i}$ is approximately a const value. When $2 \leq i \leq \sqrt{N}$,

$$\begin{aligned} K_{i,N} &\approx K_{1,N} + \frac{R_{i,N}-R_{1,N}}{p_{R_{1,N},1}} \\ &= N \ln \frac{2N}{N+1} + \frac{(i-1)(N+1)}{i+1}. \end{aligned} \tag{16}$$

Therefore, we get the approximate value of $K_{i,N}$:

$$\begin{aligned} K_{i,N} &\approx N \ln \frac{2N}{N+1} + \frac{(i-1)(N+1)}{i+1} \\ &\approx N \ln 2 + \frac{(i-1)(N+1)}{i+1}. \end{aligned} \tag{17}$$

From Eq. (11) and Eq. (17), we finally get the approximate value of adv:

$$adv \approx \frac{(\ln 2 + \frac{i-1}{i+1})(N-1)}{N_1+N_2+N_3}. \tag{18}$$

By Eq. (18), we know the degree transition points should be advanced by $\frac{(\ln 2+\frac{i-1}{i+1})(N-1)}{N_1+N_2+N_3}$, when a node finds one of its neighbor failed.

The complete algorithm for the design of CPR is given as follows.

Algorithm of CPR

$N \leftarrow$ pre-deployed information of total node number
$K_{[1..N],N} \leftarrow$ derived from Eq. (10)
$N_1 \leftarrow$ neighbors with the smaller hop number
$N_2 \leftarrow$ neighbors with the same hop number
$N_3 \leftarrow$ neighbors with the larger hop number
$B \leftarrow$ buffer size
$con \leftarrow \frac{B}{K_{N,N}}+1$
determine α according to *con*
$\beta \leftarrow \alpha \times \alpha$
while this node keeps running **do**
  if one of my neighbors fails **then**
    update N_1,N_2,N_3
    avgNeighbor $\leftarrow N_1+N_2+N_3$
  for i $\leftarrow$ *1* **to** *N*
    adv $\leftarrow$ derived from Eq. (18)
    $K_{i,N} \leftarrow K_{i,N}$ - *adv*
deg $\leftarrow$ 0
while data collection is going on **do**
  if time now > $K_{deg,N}$ **then**
    deg $\leftarrow$ *deg*+1
  form a codeword *P* with degree of *deg*
  by XORing symbols in the local buffer
  rand $\leftarrow$ a random decimal between 0 and 1
  if *rand* < β **then**
    dest $\leftarrow N_1$
    if *rand* < (β + α) **then**
      if *dest* $\leftarrow N_2$ **then**
        dest $\leftarrow N_3$
  destNode $\leftarrow$ a random node from subset of *dest*
  send codeword *P* to *destNode*

V. Performance Evaluation

Since no existing strategies were designed to deal with the specific problem of data collection for distributed surveillance sensor networks in disaster-hit regions, where the sink presents to send the query-request but the storing nodes fail rapidly at unpredictable time, we compare CPR with the original Growth Codes and No Coding, to evaluate the correctness and effectiveness of the proposed CPR algorithm. In this section, we have implemented both the original Growth Codes and CPR, both of the two algorithms are implemented in C++. No Coding is also considered, where nodes simply exchange the uncoded symbols without doing any coding operation.

A. Simulation Settings

In our settings, 500 sensor nodes are randomly located in a 1×1 square field, the sink is attached to one of the nodes at random. Each sensor takes a single reading of the monitored region and stores only one packet initially. We assume that each sensor node has a storage capacity of $B = 40$ (*i.e.*, buffer size is 40 and one node can store at most 40 codewords at any given time). A maximum of $40N$ codewords can be temporarily stored in the network, so it is sufficient to facilitate mixing of symbols. We also performed simulations with $N = 1000$ and different values of B, but the results were not significantly different.

To present how CPR performs on transmission consumption in different scenarios, we vary the communicating radius of nodes (*i.e.*, R). As R increases, each node in the network has more neighbors to communicate with. Typically in sensor networks, nodes have several neighbors, so we choose the appropriate R to afford a range of $10 - 20$ neighbors in expectation.

To present how CPR performs on maximizing the received data during the process of data collection, we simulate two networks where there are 20 and 10 neighbors for each node on average, respectively. Thus, in the former case, the sensor nodes are deployed at a high density, while in the latter case at a low density. Besides, we consider different scenarios where 10%, 30%, 50% of the sensors nodes failed, respectively.

B. Lower Transmission Cost

One goal of CPR is to adaptively choose the optimal transmission scheme (*i.e.*, α) to achieve more efficient transmission. Since in CPR, the sensors select the value of α greater than 1, thus it is more likely for the transmission to deliver a packet to the sink in the "forward" direction. Moreover, the selected α is not too large to lead to heavy data congestion around the sink. Therefore, CPR outperforms the original Growth Codes in transmission consumption.

Fig. 4 depicts the transmission consumption of CPR and the original Growth Codes versus node density in different networks. Here, the transmission consumption refers to the total transmitting times to recover all $N = 500$ symbols. Once a node disseminates a packet, we regard that the transmission consumption increases by 1. In our simulation, the buffer size is fixed at 40. From Eq. (2), Eq. (17) and TABLE I, the value of *con* and α is also fixed to be 1.2 and 1.65, respectively.

As is shown in Fig. 4, CPR decreases the transmission consumption in different cases. When node density is 20, CPR can save up to 15% transmitting times. Notice that, CPR consumes a decreasing cost as node density increases. This can be explained by a fact that in a fixed field of 1×1 square, the increasing of node density implies fewer hops required for a symbol to be delivered to the sink, and hence the total transmission consumption decreases.

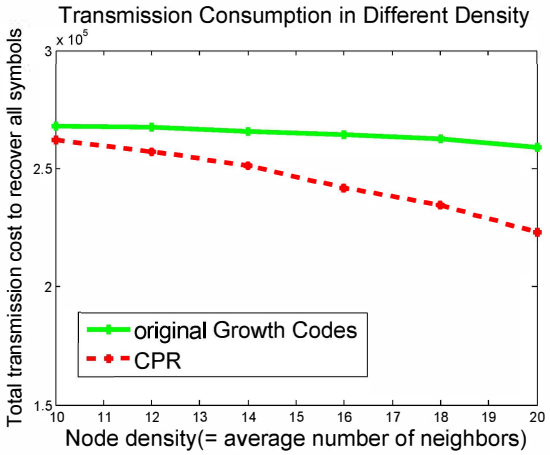
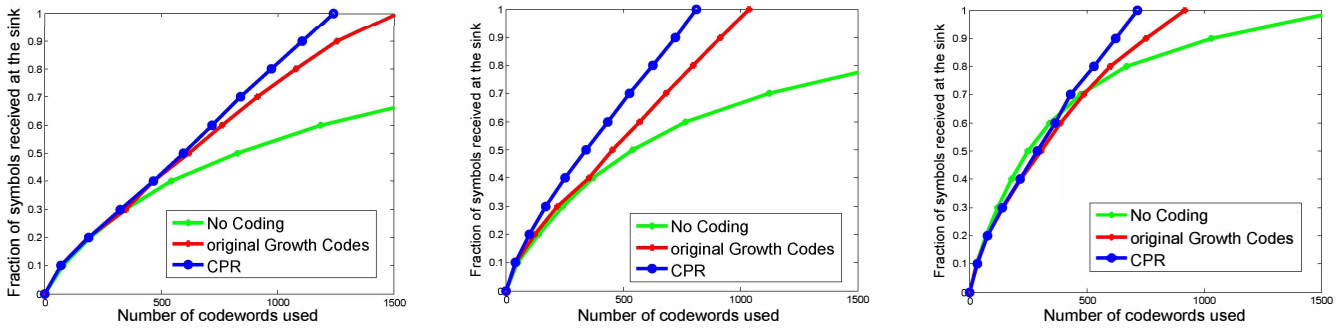
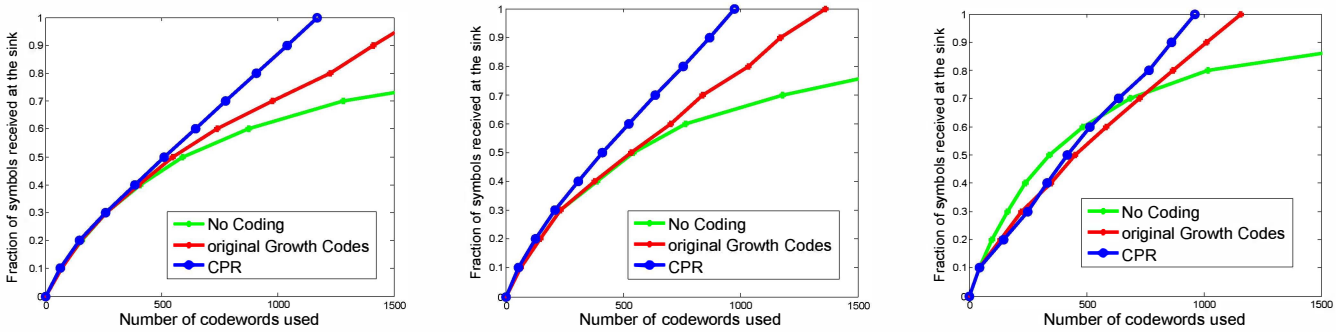


Fig. 4: CPR consumes lower transmission cost compared to the original Growth Codes.



(a) 10% of the nodes failed. (b) 30% of the nodes failed. (c) 50% of the nodes failed.

Fig. 5: Data collection for disaster-hit sensor networks with high density ($N = 500$, average number of neighbors is 20).



(a) 10% of the nodes failed. (b) 30% of the nodes failed. (c) 50% of the nodes failed.

Fig. 6: Data collection for disaster-hit sensor networks with low density ($N = 500$, average number of neighbors is 10).

C. Maximize Received Data Amount

The other goal of CPR is to maximize the received data for data collection, where the sink would send the query-request but the storing nodes fail rapidly at unpredictable time. We simulate two kinds of long-term running networks, where the node density is 20 (high density) and 10 (low density), respectively. Due to node failure, the total number of active nodes in a wireless sensor network will inevitably decrease. Thus, we consider a sensor network with $N = 500$ nodes in three cases where 10%, 30% and 50% nodes failed prior to the process of data collection, respectively.

Fig. 5 and Fig. 6 depicts the number of codewords used in the network to recover certain fraction of symbols. In Fig. 5, the performances in a high density sensor network are considered. Clearly, CPR outperforms the original Growth Codes and No Coding in all the cases of 10% nodes failed (*i.e.*, 450 active nodes) and 30% nodes failed (*i.e.*, 350 active nodes), as is shown in Fig. 5(a), Fig. 5(b) respectively. For example, to recover 280 symbols when 30% nodes failed, 628 codewords are required to reach the sink for CPR, while 800 codewords are required for the original Growth Codes and more than 1500 codewords are required for No Coding. In Fig. 6, the performances in a low density sensor network are depicted. Again, CPR is shown to outperform the original Growth Codes and No Coding.

Sometimes in the cases of 50% nodes failed (*i.e.*, 250 active nodes), No Coding would give better performance than both CPR and the original Growth Codes during the earlier phase of data collection (in Fig. 5(c) and Fig. 6(c)). It is not hard to understand, since both CPR and the original Growth Codes need a certain number of codewords to do the decoding operation. However, this can not be guaranteed when too many nodes caught failure.

Therefore, since CPR adaptively chooses the near-optimal degree transition points, it collects more data in data collection than the original Growth Codes.

VI. Conclusion

In this paper, we address the specific problem of data collection for distributed surveillance sensor networks in disaster-hit regions, where the sink send the query-request but the storing nodes fail rapidly at unpredictable time, and provide a novel coding-based probabilistic routing, called CPR. The CPR algorithm maximizes the data collection to recover more data and minimizes the total transmission cost. The core contribution of CPR can be summarized in two aspects: 1) it exploits an optimal probabilistic routing to choose the optimal transmission scheme according to the specific environments to decrease the total transmission cost, and 2) it searches for the near-optimal degree transition points to avoid the performance

degradation in the case of node failure. The proposed CPR algorithm has been compared analytically and empirically with the schemes with using the original Growth Codes and with No Coding in random networks, and the comparison results have shown the advantages of CPR in both data receiving and transmission cost.

VII. Acknowledgment

This work was supported in part by National Key S&T Project under Grant 2010ZX03003-003-03, Shanghai Municipal R&D Foundation under Grant No. 09511501200, the Shanghai Rising-Star Program under Grant No. 08QA14009. Jin Zhao is the corresponding author.

References

[1] E. Cayirci and T. Coplu, "Sendrom: Sensor networks for disaster relief operations management," in *Wireless Networks*, vol. 13, no. 3. Springer, 2007, pp. 409–423.

[2] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, R. Stoleru, G. Zhou, Q. Cao, P. Vicaire, J. A. Stankovic, T. F. Abdelzaher, J. Hui, and B. Krogh, "Vigilnet: An integrated sensor network system for energy-efficient surveillance," *ACM Trans. Sensor Network*, vol. 2, no. 1, 2006.

[3] K. Lorincz, D. J. Malan, T. R. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, M. Welsh, and S. Moulton, "Sensor networks for emergency response: Challenges and opportunities," *IEEE Pervasive Computing*, vol. 3, pp. 16–23, 2004.

[4] R. Di Pietro, L. Mancini, C. Soriente, A. Spognardi, and G. Tsudik, "Catch me (if you can): Data survival in unattended sensor networks," in *PerCom 2008*, March 2008, pp. 185–194.

[5] G. Barrenetxea, F. Ingelrest, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "Sensor scope: Out-of-the-box environmental monitoring," in *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, April 2008, pp. 332–343.

[6] C.-Y. Wan, S. B. Eisenman, A. T. Campbell, and J. Crowcroft, "Overload traffic management for sensor networks," *ACM Trans. Sen. Netw.*, vol. 3, no. 4, p. 18, 2007.

[7] C.-Y. Wan, S. B. Eisenman, and A. T. Campbell, "Coda: congestion detection and avoidance in sensor networks," in *SenSys '03*. New York, NY, USA: ACM, 2003, pp. 266–279.

[8] S.-R. R. K. R. J. Petrovic, D., "Data funneling: routing with aggregation and compression for wireless sensor networks," May 2003, pp. 156 – 162.

[9] B. Kusy, H. Lee, M. Wicke, N. Milosavljevic, and L. Guibas, "Predictive qos routing to mobile sinks in wireless sensor networks," in *IPSN '09*. Washington, DC, USA: IEEE Computer Society, 2009.

[10] 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*. New York, NY, USA: ACM, 2006, pp. 255–266.

[11] R. Ahlswede, N. Cai, S.-Y. Li, and R. Yeung, "Network information flow," *Information Theory, IEEE Transactions on*, vol. 46, no. 4, pp. 1204–1216, Jul 2000.

[12] S. Acedanski, S. Deb, M. Medard, and R. Koetter, "How good is random linear coding based distributed networked storage," in *In NetCod*, 2005.

[13] A. Dimakis, V. Prabhakaran, and K. Ramchandran, "Decentralized erasure codes for distributed networked storage," *Information Theory, IEEE Transactions on*, vol. 52, no. 6, pp. 2809–2816, June 2006.

[14] ——, "Ubiquitous access to distributed data in large-scale sensor networks through decentralized erasure codes," in *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*. Piscataway, NJ, USA: IEEE Press, 2005, p. 15.

[15] Y. Lin, B. Liang, and B. Li, "Data persistence in large-scale sensor networks with decentralized fountain codes," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, May 2007, pp. 1658–1666.

[16] 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*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 171–182.

[17] M. Luby, "Lt codes," in *FOCS '02: Proceedings of the 43rd Symposium on Foundations of Computer Science*. Washington DC, USA: IEEE Computer Society, 2002, p. 271.

[18] J. Liu, Z. Liu, D. Towsley, and C. H. Xia, "Maximizing the data utility of a data archiving & querying system through joint coding and scheduling," in *IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks*. New York, NY, USA: ACM, 2007, pp. 244–253.

[19] D. Munaretto, J. Widmer, M. Rossi, and M. Zorzi, "Network coding strategies for data persistence in static and mobile sensor networks," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007. WiOpt 2007. 5th International Symposium on*, April 2007, pp. 1–8.

[20] ——, "Resilient coding algorithms for sensor network data persistence," in *EWSN*, 2008, pp. 156–170.

[21] Y. Zhao, X. Wang, J. Zhao, and X. Xue, "Maximizing growth codes utility in large-scale wireless sensor networks," in *Euro-Par 2010: 16th International Euro-Par Conference*. Ischia, Italy: Springer, 2010.

[22] H. Gupta, S. R. Das, and Q. Gu, "Connected sensor cover: self-organization of sensor networks for efficient query execution," in *MobiHoc '03*. New York, NY, USA: ACM, 2003, pp. 189–200.