Ravine Streams: Persistent Data Streams in Disruptive Sensor Networks

Mingsen Xu, Wen-Zhan Song
Department of Computer Science, Georgia State University
Atlanta, Georgia 30303, USA
Email: {mxu4@student.gsu.edu, wsong@gsu.edu}

***Abstract*—Opportunistic network coding has been developed and applied in disruptive networks to provide optimal data delivery. Though network coding system utilizes coding opportunities among multiple paths, its application in data collection suffers from a disconnected sink node and the limited storage space available for data cache. The state-of-the-art approach has studied preserving data persistence as an optimization problem under storage and energy constraints, without considering disruptive network dynamics during data redistribution. In this paper, we propose Ravine Streams (RS) to maximize data preservation under the constraints of limited storage and probabilistic node failure throughout data redistribution. Our RS approach leverages adaptive power control to achieve ensured storage of each redistribution data. Meanwhile, in the course of data redistribution, distributed coding-based rebroadcast strategy not only reduces the data duplication, but also improves the statistical property of symbol randomness. We show that the performance of preserving data persistence of proposed RS is approximately bounded by the optimal solutions. The experimental evaluations demonstrate that RS increases data delivery ratio, consumes even less communication energy with only comparable storage cost, when compared with existing data preserving algorithms.**

***Index Terms*—Persistent Data Collection, Network Erasure Coding, Transmission Power Control, Disruptive Sensor Network**

I. Introduction

In a large-scale sensor network deployment, one of major reasons contributing to intermittent connections and disruptive communication is sensor nodes failure. Note that the reasons for node failure have a quite broad range, which can be either power depletion, unable to route packet to destination or overflowed receiving and transmitting buffer. Each node failure influences the data collection flow routed through it. In the worst case, sink node failure impairs the entire network traffic. To preserve data persistence, data stream has to be rerouted to other paths or redistributed to other storage nodes if none active routes are found. Existing algorithms from the literature studied this data preserving as an network flow optimization problem, like [1] maximizes the minimum residue energy in the network for the next data redistribution to arise; [2] combines minimizing data redistribution and retrieval cost into a single problem.

Though optimal solutions are devised under either energy or storage space constraints, most of the previous works ignore the fact that node failures could happen in the course of data rerouting or redistribution. The determined approach is not capable of scheduling data redistribution without the global knowledge of which part of nodes could fail in the middle. Moreover, the data redistribution is conducted in the network of heterogeneous node failure probabilities. This fact implies that data should move towards more reliable nodes with sufficient residue energy and storage space, such that higher data delivery rate can be achieved.

In this paper, we propose *Ravine Streams* to preserve data stream persistence in disruptive sensor networks. Source data is initiated and delivered in its encode form by network erasure codes *OnCode* [20]. For data preservation, receiving nodes distributively make the acceptance decision based on local failure probability and storage space. Note that here nodal failure probability has taken the residual energy into account and dynamically update across the process. Adaptive transmission power control in *RS* ensures that data acceptance probability by neighbor nodes is expected at 1 for each broadcasting under minimum transmission power. With data packet recoding, the dispensable data content redundancy is constrained for more energy efficiency. Moreover, we show that the delivery performance of proposed *RS* is bounded by *Optimal* solution: $\eta = \frac{OPT \cdot \ln(\frac{B}{\delta})}{2\lambda}[1 + \omega(1) \cdot \frac{1}{B} + 2\omega(2) \cdot \frac{B-2}{\binom{B}{2}}]$.

In summary, this work has three contributions: (1) RS leverages probabilistic transmission power control and network erasure recoding to preserve data throughout disruptive network connection. Additionally, the data redundancy is largely reduced, increasing algorithm energy efficiency. (2) RS makes distributed probabilistic decision that directs redistribution data to more reliable storage nodes, without global knowledge of density and node failure probabilities, smoothing data collection traffic under disruptive connections. (3) Our analysis shows that the proposed RS algorithm has bounded performance compared with optimal solution.

The rest of the paper is organized as follows. In section II, we summarize and compare related works. Then we present the design and analysis of *Ravine Streams* protocol in section III. In section IV, we present experimental evaluation results. We conclude our work in section V.

II. Related Works

A. Data persistence without storage constraint

GROWTH [3] ensures data reliability in a zero-configuration network with no storage constraints. Growth code evolves its code degree over time. For each code degree,

it leads to a better decoding probability than LT codes when partial encoded packets are received. MORE [4], SlideOR [5] and [6] leverage random linear coding to encode innovative received packets. All of them have better performance than the ExOR [7], which is the opportunistic routing protocol without network coding. MORE encode symbols by random linear coding, without needs of selecting next-hop forwarder list and improving spatial usage in multiple transmissions. CCACK [8] uses a null-space based ACK to notify neighbor nodes about the reception of encoded packets. CCACK suppress redundant packets, which carry linearly dependent encoded symbols. Distributed LT Codes [9], [10] decomposes original degree distribution into different probability distributions for sources only one-hop away from data relay based on a two-branch relay model. The degree deconvolution with selective XOR in forwarders theoretically yield a degree distribution close to RSD in the receiver. Xu et al. proposes ONEC [11] to explore network erasure coding in disruptive sensor networks. ONEC leverages a strict structure, e.g. tree structure, to conduct recursive deconvolution of degree distribution. And opportunistic recoding is exploited to ensure more symbols can seep through network to receiver.

All the aforementioned works do not consider storage constraint and the scenario of sink failure. Sink failure and $O(1)$ storage space constraints can pose a big challenge to network coding based methods, which merely encode and forward packets to sink, in either opportunistic or ACK-control manner.

B. Data persistence with storage and energy constraint

Lin et al. [12] propose a LT codes based erasure coding for preserving data in a large-scale network. The key idea is to randomly walk data from redistribution nodes to storage nodes to emulate the random selection in LT coding. Multiple redundant random walks are initiated by each of data which stop in a storage node with specific probability derived by its selected code degree. A optimal redundancy factor for each data symbol is derived offline by linear programming. In [13], Aly et al. suggest to construct LT codes without knowledge of maximum node degree. In [14], Aly et al. employ random walk based scheme to distributively construct the Raptor Codes, which is of scalable decoding complexity $(1+\epsilon)n$ with respect to the network size and symbol size. [15] presents a packet-centric approach to ensure that degree distribution conform with final degree distribution. Every encoded packet is associated with a selected code degree. The packet collects and encodes the symbols from passing nodes uniformly distributed in the network. The packets terminate once associated code degree is satisfied.

The above methods preserve data persistence in a robust manner by using network coding, however, they ignore the storage and energy constraints. Tang et al. [16] formalizes storage depletion induced data redistribution as the minimum cost flow problem, and devises a distributed data redistribution algorithm (*PoF*). [17] consider the energy and storage depletion to maximize data preservation time. Valero et al. [2] formulates the problem as an optimization problem and use Linear Programming to find the optimal solution. A distributed algorithm (*EDR2*) is implemented for in-network storage and later data retrieval. [18] proposes a probabilistic broadcasting based data redistribution scheme. Their contribution is to derive a proper probability distribution for rebroadcasting packet. Hou et al. [1] seeks to maximize the minimum remaining energy among the nodes storing data items. It presents a centralized greedy heuristic to approximate the optimal solution under certain conditions.

The above optimization algorithms, though considering energy and storage constraint, ignore an important fact that disruptive condition may happen in the course of data redistribution. The disruptive network seriously challenges data redistribution. Distinct from the existing works, our contributions of *RS* are three fold: *first*, *RS* conducts probabilistic broadcasting with adaptive transmission power to overcome the disruptive connection during data redistribution. It has high energy efficiency and low message redundancy. *Second*, *in-situ* recoding can reduce data redundancy in symbol wise. *Third*, *RS* generalizes the energy and storage constraints to nodal utility, which includes failure probability and storage constraint. According to the nodal utility, data storage decision can be made distributively.

III. Algorithms and Analysis

Ravine Streams is an integrated set of distributed algorithms that protect data collection stream from disruptive network nodes and connections. The data persistence preservation of *RS* is built upon the opportunistic in-network coding and delivery (OnCode) [20]. Data packets are initially encoded and transmit by *OnCode*. When local storage space vanishes, *RS* redistributes excessive received data packets to more reliable nodes to preserve data persistence. RS considers both heterogeneous failure probabilities (due to energy depletion or other factors) and storage constraints, and addresses the problem of disruptive connection during data redistribution. RS leverages network erasure coding in data redistribution, and determines the rebroadcasting probability and transmission power level according to nodal utilities. It is worth to mention that RS algorithms are probabilistic, adapting to the dynamic network conditions with minimum execution overhead.

We describe the properties of an ideal data recode and redistribution algorithm, which is referred to as *OPT*. As the optimal benchmark solution, *OPT* presents the following properties that our algorithm design concerns. First, *OPT* does know how much data can be accepted and how much data needed to be redistributed if the node happens to be disruptive at a certain time. Here, we define a node is disruptive as it can not forward any data towards sink, and data stream is disconnected hereafter. In other words, *OPT* determines the *data acceptance* in an optimal way so that the storage use are balanced across the network. Second, for a data redistribution in a time range ahead, *OPT* can always find a determined path from disruptive node to destination storage node with stable communication link and ensured data delivery. And only

one copy of source data packet needs to be maintained inside network. In practice, it is unlikely to predict a stable communication link across multi-hop connection in the disruptive network conditions, which means the data redistribution is not guaranteed to be successful every time. Thus, multiple copies of data packet are necessary to reduce the risk of data loss. Under this condition, data redundancy needs to be reduced between multiple survived data copies. Third, *OPT* can observe if the disruptive node is trapped in bad region or not, so as to adjust the appropriate radio transmission power to assist the communication. In fact, without perfect knowledge ahead of time, it consumes considerable overhead to search different levels of transmission powers until a proper power is found.

RS adopts two integrated mechanisms to tackle the unideal situations: *Determine Data Acceptance*, *Probabilistic Data Redistribution*. RS is able to preserve data persistence in heterogeneous situations subject to node failure probability and storage constraints. The mathematical symbol and notations are described in Table I.

TABLE I
NOTATION IN ALGORITHM

Notation	Explanation
N_i^m	Neighbor set of node i under power level m
U_i^m	Nodal utility under TX power m
δ_i	Failure probability of node i
C_i	Free storage space on node i
C_{max}	Maximum storage space
β	System parameter for data acceptance
α	Replacement factor
q_i	Probability of accepting incoming packet in node i
p_m	Selection probability for power level m
τ	Exponential index of power adjustment

A. Determine Data Acceptance

Due to the opportunistic nature of encoded network traffic, it is unlikely to predict the exact amount of incoming packet in a certain time window. Therefore, when disruption occurs, there is hardly a determined way to calculate the exact amount of data needed to be redistributed. The best approach is to make this decision based on the dynamic available storage space in each node. RS adopts the reactive methods to discern available storage in node i, defined as C_i. We first look at an example of a data stream from $i \rightarrow j \rightarrow k$, where node k becomes disconnected, making node j not able to forward any incoming traffic. Node j stores data packets unsent in its own buffer. To preserve data persistence, node j has to redistribute the excessive data to other nodes for storage before its space is used up.

If data packets are redistributed only when storage space is full, the network will experience a sudden and unexpected traffic burst. To avoid this dramatic traffic change, RS adopts probabilistic method to pre-estimate the probability q_j for each incoming packet. The probability q_j indicates the probability with which node j accept the received packet. With this smoothing technique, network traffic can remain relatively stable even under unforeseen disruptive conditions.

The probability q_j is determined distributively, based on local ratio of nodal available storage space C_j and maximum storage space C_{max}. With less available storage ratio, incoming packet will be less likely to be accepted. Moreover, each node i has a certain failure probability δ_j due to either energy depletion or other external factors. Therefore, final data acceptance probability is estimated against factors $\frac{C_j}{C_{max}}$ and δ_j. We combine these two factors by *multiplicative* operation.

$$q_j = \beta \cdot U_j = \beta \cdot (\frac{\delta_j \cdot C_j}{C_{max}}) \tag{1}$$

where U_j represents the available nodal utility, and β is introduced by RS algorithm to hold a tolerance guard, which is tunable. If node decides not to save the received packet, it will proceed to redistribute the packet to other nodes for storage.

B. Probabilistic Data Redistribution

The reason of redistributing data packet following a probabilistic manner is two fold. First, having packets probabilistically broadcast consume significantly less communication cost for offloading data to other storage nodes than that of gossip flooding. Second, statistically such a probabilistic redistribution can mitigate the risk of data loss despite the fact part of delivery paths may fail in the course of redistribution.

Two main challenges arise in design. First is about how to make data redistributed towards reliable storage nodes instead of nodes with higher failure probability. Due to heterogeneous node densities, rebroadcast message can be heavily stored in nodes with denser vicinity but unreliable nodes. Data is unlikely to be redistributed to other nodes, which might have better connection to sink node for collection purpose. *RS* integrates the adaptive power control to change the outcome of data redistribution towards more reliable nodes.

Second challenge is to reduce data redundancy in redistribution. More redundant data packets broadcast in the network implies more energy are consumed to achieve this process. Although energy cost for preserving data persistence is the extra overhead must paid, this cost needs to be carefully conserved. Excessive waste of energy severely causes nodes to fail due to power depletion.

Probabilistic data redistribution is articulated in Algorithm 1, which describes the transmission power control based rebroadcast and recoding based data redundancy constraining. When the node with decreasing storage space decides not to save the incoming packet, it starts to redistribute the data to other storage nodes with adaptively controlled transmission power. In line $5-8$ of Algorithm 1, node i selects a certain power level for broadcasting, according to a posterior probability distribution of transmission power, i.e. $\{p_1, p_2, ..., p_m\}$. This power distribution is dynamically updated based on available storage space among neighbor nodes. This rebroadcast is invoked when the message is not accepted by the node. Since it is the source of data redistribution, there is no duplicate copy of this packet in the network yet.

Algorithm 1 Probabilistic Data Redistribution

Input: *Effective storage space* C_i, β, *max storage* C_{max}, *replacement factor* α, *Power distribution* $\{p_1, p_2, ..., p_m\}$

1: Draw random variable ρ from range $(0, 1)$
2: Set $q_i = \beta \cdot U_i = \beta \cdot (\frac{\delta_i \cdot C_i}{C_{max}})$
3: **if** $\rho < q_i$ **then**
4: Save this data packet to local storage
5: **else**
6: Node i select power level m with probability p_m
7: Rebroadcast the data packet κ using selected transmission power m.
8: **end if**
9: Node j upon receiving a rebroadcast data packet κ
10: Draw random variable ρ from range $(0, 1)$
11: **if** $\rho < q_j$ **then**
12: Save the data packet to local storage flash
13: **else**
14: **if** Check coding distance$(\kappa, \mathcal{S}) > 0$ **then**
15: Select symbol S_l with smallest distance with other symbols in storage
16: Replace S_l with κ with probability α
17: **if** ISRECODABLE$(\kappa, \mathcal{S})$ == TRUE **then**
18: Recode κ by XORing symbols in storage
19: **end if**
20: Rebroadcast packet κ' with probability $(1 - \alpha)$
21: **else**
22: **if** ISRECODABLE$(\kappa, \mathcal{S})$ == TRUE **then**
23: Recode κ into κ' by XORing symbols from storage
24: **end if**
25: Rebroadcast new packet κ'
26: **end if**
27: **end if**

Then the rebroadcast packets are treated differently. When a rebroadcast packet arrives at a node with sufficient storage, the message will be accepted and stored (line 11-13). In the other hand, when no sufficient storage is available, receiving node will operate recoding before it rebroadcast out the message with a selected probability. In line 14, it examines the code distance between received packet κ and encoded symbols in the storage space with the same code degree. If there exists data of the same degree which has different code symbols, the packet will replace the one with smallest coding distance from other symbols with a probability α. The purpose of replacement is to maximize the decoding probability, since the encoded symbol with minimum code distance contribute little to the decoding.

In case that no symbol with the same degree is found in the storage, algorithm evaluates the opportunity that rebroadcast packet can be recoded by XOR operation with stored symbols to generate a new symbol. Note that the recoding does not change the code degree associated with the packet (line 23-25). Recoding the same rebroadcast message in different nodes can mix the message randomly with diverse symbols, generating non-duplicate encoded data. Hence, it reduces the data redundancy in rebroadcast messages, avoiding wasting energy in broadcasting duplicate messages.

Adaptive power control and *constrained data redundancy* are two major technical building blocks in the probabilistic data redistribution, which are described as follows.

1) Adaptive Power Control: Without adjusting radio transmission power, packets may be trapped in low storage region. Adjusting transmission power moves packet out to the nodes with more sufficient and reliable storage space. The transmission power adjustment procedure itself is probabilistic. An intuitive way for adjusting probability of transmission power m is to make it proportional to the total available nodal utilities $\sum U_j^m$ in node j:

$$p_m = c \cdot \sum U_j^m \tag{2}$$

The above equation implies that transmission power that reach larger amount of $\sum U_j^{P_i}$ has a larger probability to be selected for data rebroadcast. However, there exists one drawback of directly using the proportional equation to determine the probability of transmission power. Applying this strategy, each node greedily selects the maximum power to rebroadcast throughout the process, leading to significant message flooding and useless data redundancy, though the local data entrapment is relieved. Figure 1 shows an example depicting this problem. In the figure, nodes from A to G select their maximum power respectively, i.e. $m = 3$, for it can reach nodes with most nodal utilities. However, each node, like node F, receives 6 copies of rebroadcast messages, which are mostly duplication. Meanwhile, transmission with power m can consume quadratically more energy than that of power $m - 1$. The energy efficiency can be enhanced by letting each node selects their transmission power conservatively. Meanwhile, we still wish to preserve the data persistence, which means that rebroadcast message is stored by at least one storage node. Adaptive power control has been designed to achieve both of goals.

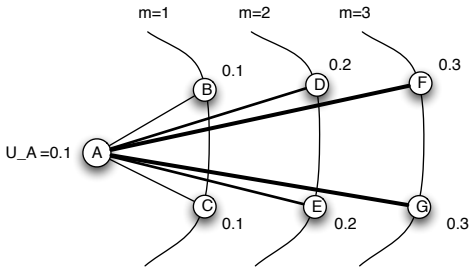


Fig. 1. Example of transmission power control

The probability of selecting transmission power level is redistributed to be exponentially proportional to the total nodal utilities among rebroadcasting node's neighbors.

$$p_m = (\frac{\bar{U}_j^m}{\bar{U}_j^{m-1}} \sum_j (U_j^m))^\tau \tag{3}$$

where τ is the exponential system parameter tunable between 0 and 1 and $\bar{U}_j^m$ is the average nodal utilities of node j and node j's neighbors, when power level m is applied in node j's transmission. When the sum of nodal utilities equals to 1, the expected probability of storing the rebroadcast message is ≥ 1. This ensures that the corresponding power level is selected to guarantee the data persistence. Rather than linear proportional relationship, the above equation has the power exponent $\tau \in (0, 1)$. With system exponent τ, the power level p_m, under which the sum of nodal utilities U_j^m is closest to 1, has highest probability in all power levels. The reasons of adding coefficients $\bar{U}_j^m / \bar{U}_j^{m-1}$ are two fold. First, the power level m, which increases not only total nodal utilities but also the average values, should be given a higher probability in the power distribution. Second, p_m needs to approach to 1 more aggressively once the next power level m leads to a larger average utility amount, so that cut-off power level may appear smaller. A smaller cut-off power level indicates exponential energy conservation. In Equation 3, if the calculated $p_m \geq 1$, then let $p_m = 1$. Notice that m, in which $p_m \geq 1$, is the cut-off power level in power probability distribution. Through probability normalization, it ensures that $\sum_m p_m$ equal to 1. The average nodal utility follows that:

$$\bar{U}_j^m = \frac{1}{|N_j^m| + 1} \sum U_j^m$$

In order to calculate the nodal utility summation and average value, node j needs to know its neighbors' utilities. Thus, RS algorithm adds two extra fields in each broadcasting packets: power level indicator and available nodal utility. Upon receiving rebroadcast message, node j records both power level received and available nodal utility from neighbor node.

For instance, in Figure 1, before adaptive power control, node A choose power level based on Equation 2, so that a total nodal utility can reach or exceed 1. Since $(0.1 + 0.1 * 2 + 0.2 * 2 + 0.3 * 2) = 1.3 > 1$, the most likely power level for node A is $m = 3$. We set $\tau = 0.4$. With adaptive power control, when $m = 1$, it gives $p_1 = (0.1 * 3)^{0.4} = 0.618$. Set $m = 2$, then $p_2 = (\frac{0.14}{0.1} \cdot 0.7)^{0.4} = 1$. Thus, node A's cut-off power is reduced from level 3 to level 2.

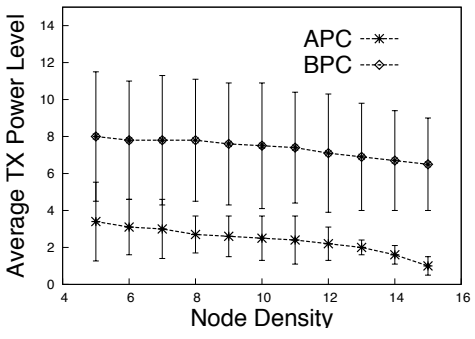


Fig. 2. Average TX power level for data redistribution.

Figure 2 shows the average TX power levels selected by Equation 2 (BPC) and Equation 3 (APC) under different node densities respectively, within the TOSSIM [19] network simulator. From the results, it shows that adaptive power control (APC) reduces the power level by 60% on average compared with basic power control (BPC). In addition, the smaller deviation of power level among nodes indicate that energy consumption of TX power is relatively more balanced if executed by *APC*. Moreover, the value of power level decreases slowly in *BPC*. The reason is that each node greedily selects its power level, so that storage space is consumed faster than necessary, making other nodes have to take higher transmission power to discover more storage nodes.

2) Data Redundancy Constrain: Adaptive power control (APC) reduces the energy consumption of rebroadcast as well as the total amount of rebroadcast data replication. However, redistribution packets consist of duplicate data due to the rebroadcasting process. Data content redundancy wastes communication energy as well as the storage space. Therefore, we further constrain the content redundancy of rebroadcast message by conducting symbol recoding without impairing the original code degree distribution.

Recoding procedure is shown in Algorithm 2. Recoding step takes as input the rebroadcast encoded packet S, a set X of both decoded symbols and encoded packets of degree less than 2 and regenerates a fresh encoded packet with the same degree as d. This recoding step mixes the encoded symbol of rebroadcast packet with local symbols in storage, such that multiple copies of broadcast packets can be stored with distinct encoding contents after their redistribution process.

Algorithm 2 Data Recoding

Input: *Received rebroadcast packet S, a set X of both decoded symbols and encoded packets of degree ≤ 2*
Output: *Recoded packet Y*

```
for all s ∈ S do
    B ← ∅
    for all x ∈ X do
        B ← x, if{x ⇔ s & freq(x) < freq(B)}
    end for
    if B ≠ ∅ then
        Y ← S ⊕ x
    end if
end for
```

The basic operation in network erasure coding is *XOR*, which is effective and of low computation cost. It verifies the properties that swapping symbol only requires another *XOR* operation. For instance, a native symbol s_1 in S can be replaced with s_2 if target pair $(s_1 \oplus s_2)$ is available, where s_2 is not contained in S. That is, $(s... \oplus s_1) \oplus (s_1 \oplus s_2) = (s... \oplus s_2) = S'$. We denote this relationship as "$s_1 \Leftrightarrow s_2$". Since there are more than 50% encoded packet of degree equals to or less than 2 in LT Codes and Raptor Codes and the search for degree 2 packets are faster than higher degree packets, we mainly leverage those local packets with degree $d \leq 2$ to recode rebroadcast packets. When multiple target

pairs are found in the storage or can be generated from native symbols, like $(s_1 \oplus s_3)$ and $(s_1 \oplus s_4)$, the symbol s_i, with less appearance frequency is selected.

Illustrated in Figure 3, conduction of symbol recoding constrains the data redundancy. In Figure 3, packet of $\{x1 + x2 + x3\}$ is broadcast from node C, and two duplicate copies are received by node A and B respectively. There exist different groups of encoded symbols previously cached in node A and B storage. For example, node A has 3 decoded native symbols $x3, x4, x6$ (degree 1), which has different appearance frequency in the prior symbol recoding, shown in number marked in the right-top corner. According to line 4 in Algorithm 2, node A selects $\{x2 + x5\}$ to recode, because $x5$ satisfies "$x5 \Leftrightarrow x2$" and $\{x2 + x5\}$ has the lower frequency than $\{x3\}$ and $\{x4\}$ as well. The recoded packets from node A and B become distinct and carry different native symbols.

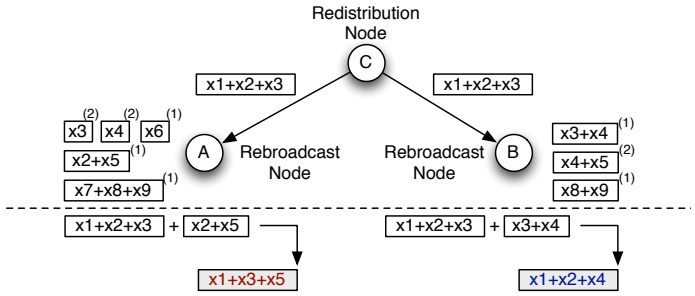


Fig. 3. Illustration of symbol recoding in rebroadcast node.

Although a successful decoding probability depends on the code degree distribution adopted during encoding, symbol recoding does increase the decoding probability. From LT encoding point of view, our data redundancy constrain approach builds stronger and more reliable connections between left and right components in the Tanner Graph. Introducing more connections between symbol and encoded packet prevent decoding from early failure, in the case of encoded packet loss.

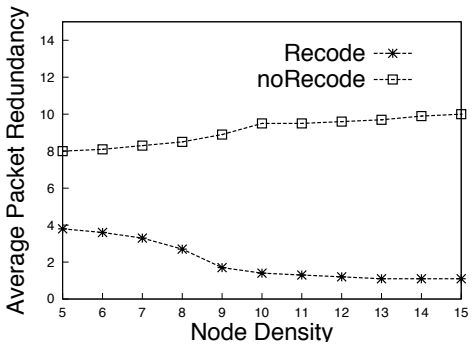


Fig. 4. Data redundancy under various node densities.

Figure 4 evaluates the data redundancy against different node densities. With our recoding on rebroadcast packet, the redundancy of data content has been reduced by at least 50%. With node deployment becomes denser the data redundancy with recoding can be further lowered to approach to 1, which means that there is almost no identical rebroadcast message on the network. In contrast, without recoding the data redundancy increases as the node density, because more nodes are involved in packet rebroadcasting, causing more duplicate packets in the network.

C. Algorithm Analysis

RS preserves data when disruptive connection causes buffer to overflow. This data quality preservation layer is built upon opportunistic in-network coding-based (*OnCode* [20]) data delivery. The packets delivered in the network before redistribution are handled by *OnCode* protocol. The deliverable symbol size B for unit time span is based on the disruption probability distribution Π, shown in [20].

Lemma 1: [20] Given a disruption probability distribution $\Pi = \{\pi_{w_1}, \pi_{w_2}, ..., \pi_{w_n}\}$, with π_{w_i} denoting the probability for discretized disruption probability w_i, then by recursive hitting time estimation, the deliverable symbol size $B = \sum_{i=1}^{n} B_i = \sum_{i=1}^{n} \frac{R_i}{(1+\epsilon)\lambda} = \sum_{i=1}^{n} \frac{1/(1+\epsilon)\lambda}{L(i,sink)} = \sum_{i=1}^{n} \frac{1/(1+\epsilon)\lambda}{\sum_{k \in Nbr(i)} P_{ik}(T_{ik}+L(k,sink))}$, where both P_{ik} and T_{ik} are determined by distribution Π.

Hence, the performance bound of *OnCode* is described as:

Theorem 2: [20] Given the size of symbol set as B and degree distribution of Raptor Codes $\omega(*)$, the delivery ratio of *OnCode* is OPT/λ, where $\lambda = \frac{1+\sqrt{1+4\cdot(1-e^{-\epsilon B}/\delta)}}{2\cdot\omega(1)\cdot(1+\epsilon)}$ $(\delta, \epsilon \in (0, 1])$, with a successful decoding probability of all symbols $\geq (1 - e^{-\epsilon B})$.

[20] shows that *OnCode* provides a OPT/λ performance bound. We continue to consider the optimal solution of data preserving in the course of disruptive network connection: *OPT*. A stable path from distribution node to storage node is always perceived in *OPT* for distributing data, and exact one copy of data distribution is sufficient for preserving persistence. Therefore, the storage cost for *OPT* is $O(1)$. For *RS*, adaptive power control in section III-B1 ensures that in each broadcast the expected acceptance probability of distribution data in storage node approximates 1, and *RS* adopts 2-hop rebroadcast to make sure the event that storage nodes accept distributed data with probability equivalent to 1.

We assume that there are available B storage space for B innovative data symbols. *OPT* can exactly leverages all of these space to store redistribution data for persistence. Then, the delivery ratio of *RS* is derived as follows.

Theorem 3: Given the size of source data set B, and degree distribution of Raptor Codes $\omega(*)$, the final delivery ratio of *RS*: $\eta = OPT \cdot \frac{\ln(\frac{B}{\delta})}{2\lambda}[1 + \omega(1) \cdot \frac{1}{B} + 2\omega(2) \cdot \frac{B-2}{\binom{B}{2}}]$, where $\lambda = \frac{1+\sqrt{1+4\cdot(1-e^{-\epsilon B}/\delta)}}{2\cdot\omega(1)\cdot(1+\epsilon)}$ and $\delta, \epsilon \in (0, 1]$, with a successful decoding probability of B symbols $\geq (1 - e^{-\epsilon B})$.

Proof: RS adopts 2-hop redistribution to ensure the acceptance of data, therefore only $\frac{B}{2}$ can be accepted in storage nodes expectedly without recoding. In other words, the final delivery ratio of *RS* becomes $\frac{1}{\lambda \cdot 2}$. However, *RS* leverages recoding to eliminate data duplication, so that any 2 copies

are distinct from each node when they are stored. Therefore, the final delivery ratio is determined by the portion of data that can be recoded shown in Algorithm 1.

Only packets of degree 1 and 2 are utilized in recoding. And we define the event *A* as that packets are recodable when there exists a symbol x in the packet that can find $x \Leftrightarrow y$ in the storage symbol set of either degree 1 or 2. The probability of event *A* is $\ln(\frac{B}{\delta}) \cdot (\omega(1) \cdot \frac{1}{B} + \omega(2) \cdot \frac{B-2}{\binom{B}{2}})$. If this event occurs, the corresponding delivery ratio of *RS* is $1/\lambda$, otherwise $1/2\lambda$. Therefore, the final expected delivery ratio equals to: $OPT \cdot \frac{\ln(\frac{B}{\delta})}{2\lambda}[1 + \omega(1) \cdot \frac{1}{B} + 2\omega(2) \cdot \frac{B-2}{\binom{B}{2}}]$.

■

IV. Performance Evaluation

In this section we present the performance evaluation of RS on TOSSIM [19] simulation. The performance of RS is compared against other existing data collection and redistribution algorithms from the literature. The experimental results demonstrate that data delivery ratio under disruptive networks is substantially improved by our *RS* algorithm, with an efficient energy cost and moderate storage cost. Experimental setup and result analysis are discussed as follows.

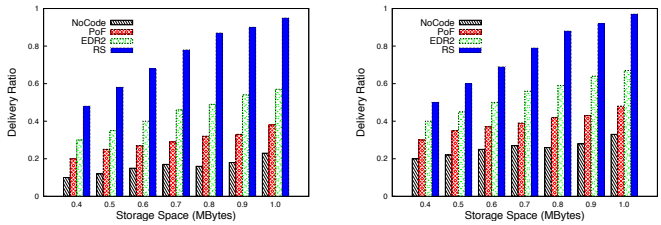
A. Experimental setup

To illustrate the advantages of *Ravine Streams* in improving data delivery ratio, hence enhancing data persistence over disruptive sensor networks, this work implements and compares *RS* with three other existing data collection and redistribution protocols: *EDR2* [2], *PoF* [16] and *NoCode*. *EDR2* is proposed to optimally solve the data redistribution problem under intermittent network connection, i.e. minimizing the data redistribution and retrieval cost. *EDR2* is implemented in a distributed manner by "push-relabel" network flow methods. *PoF* devises an index based potential field, which is used to determine the amount of data redistributed to storage nodes. It shows the optimality of data redistribution without data retrieval cost. *NoCode* is a plain data collection protocol, without specific redistribution and coding strategy. Once node disruption happens, data is redistributed randomly to other storage nodes by broadcasting.

The experiment is carried out in $100m \times 100m$ square area with nodes randomly deployed. We test the algorithms in a network of size 100 and 500 respectively, and the total storage space, nodal failure probability and number of data generators inside network vary to compare the performances under different scenarios. Moreover, energy cost and flash storage cost are also evaluated among different algorithms. For *RS*, the total power levels are set as 8, each of which can reach areas of different radii. Set nodal utility factor $\beta = 0.9$, and replacement factor $\alpha = 0.5$. We show the experimental results in the following sections.

B. Data persistence under disruptive networks

For numerical evaluation, we use data delivery ratio to represent data persistence. Data delivery ratio is defined as the ratio between the amount of data sent and the actual amount of data recovered by the destinations. Figure 5 shows the data delivery ratio for multiple algorithms under variable total storage spaces. The average nodal failure probability is 20% and there are 50 and 200 data generators in network size of 100 and 500 respectively. Each of data generators has $1kbps$ data rate. In Figure 5, y-axis denotes the percentage of data sent which can be preserved throughout the network disruption, with x-axis showing different total storage space in nodes. *RS* outperforms other algorithms for all cases of different storage spaces. From both of experiments, the ratio of data persistence improves along with the increasing storage space. For example, it can be observed that with increasing nodal storage space ($0.4M$bytes to $1.0M$bytes), the ratio of data persistence is significantly improved by RS, increasing from 50% to 93%, which is much more than the increments in other algorithms. Compared with the *EDR2* and *PoF*, the application of network erasure coding in *RS* makes packet redistribution more robust. Unlike the *EDR2* and *PoF*, which does not has reliable data redistribution to storage node, *RS* adopts probabilistic broadcasting together with randomized recoding for data redistribution. Therefore, more data packets are preserved even in the present of disruptive nodes and network conditions. Compared with the *NoCode*, *RS* eliminates the duplicate packet retransmissions. The recoding of redistributed packet with innovative data in *RS* is beneficial for mitigating the data content redundancy. Each packet carries different innovative symbols which contribute to the final decoding. Recoding not only reduces the data redundancy, but also promotes the randomness in mixing symbols. The network erasure coding based data delivery can maximize its coding gains when the symbol selection is pure random across the entire source symbol set.

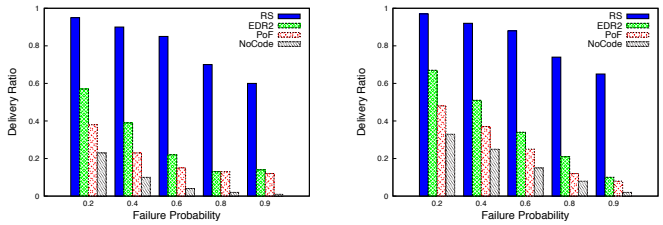


(a) Network Size 100 (b) Network Size 500

Fig. 5. Data delivery ratio over different storage spaces (failure prob. = 20%).

Figure 6 demonstrates the delivery ratio under different node failure probabilities. The storage space of each node is set to $1M$bytes. In the left figure, it shows that RS stills has the best performance in terms of delivery ratio. More importantly, when average nodal failure probability increases, the delivery ratio of *RS* only reduces by 30%, and still maintains above 60% delivery ratio under 90% failure probability. On the contrary, the delivery ratio of *EDR2* and *PoF* fall below 20%, and *NoCode* has even less than 5% data delivery ratio. The right figure in Figure 6 demonstrates the same trend. An essential reason is that algorithms other than *RS* fail to consider disruptive network condition during data distribution,

and hence algorithms are not able to adapt to intermittent connection and disruptive communication link. For example, after redistribution node in *PoF* determines how many amount of data should be rerouted to the destination storage node, based on the potential index, redistribution nodes take it for granted that the redistribution data can reach or be stored in the destination. Unfortunately, it is not always true, especially in disruptive networks where the intermittent connection can happen any time during the entire data delivery. This experiment manifests that disruptive communication poses a big challenge to existing data redistribution protocols.



(a) Network Size 100 (b) Network Size 500

Fig. 6. Data delivery ratio over different failure probabilities (storage=1MByte).

Figure 7 demonstrate the delivery ratio under different data generators in the network. With increasing number of data generator, the network traffic becomes heavier and preserving data persistence is hence more challenging. The figure shows that when there are 100 data generators in a network of 500 nodes, *EDR2* and *PoF* can still preserve more 50% data, with 45% data persistence for *NoCode*. However, the data delivery ratio of all three approaches drop significantly as the number of data generator increases from 100 to 350. In the worse case, all three data preserving algorithm only manage to preserve less than 20% amount of innovative data. By comparison, it shows the advantages of *RS*, which retains a steady performance of above 90% data persistence across different number of data generators. Efficient data duplication avoidance (by recoding) and message redundancy reduction (by opportunistic rebroadcasting and adaptive power control) in *RS* play an important role in maximizing data delivery ratio.

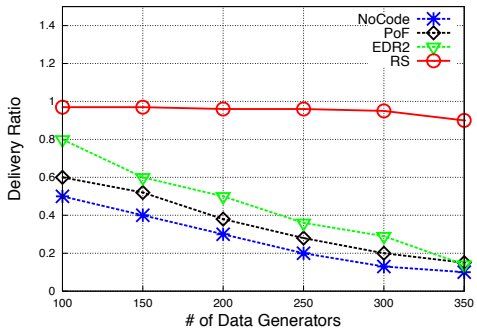


Fig. 7. Data delivery ratio v.s. amount of data generators (storage=1MByte, failure prob. = 20%).

C. Storage cost

We evaluate the storage cost for each of the algorithms in Figure 8. We observe that *NoCode* method consumes more storage space than others when number of data generator increases higher than 100. Under 350 data generators, there are only 6% storage space on average is available in each of nodes. This implies that the uncontrolled rebroadcasting of *NoCode* results in much redundant packets which requires much more storage space. Although *RS* consumes less storage space than *PoF* and *EDR2* in the network of light traffic, *RS* requires more storage space as the network traffic becomes higher. This is because *RS* adopts the opportunistic rebroadcasting to accommodate the disruptive communication link which introduces moderate duplicate messages. *PoF* and *EDR2* consume less storage space but introduce much more energy cost.

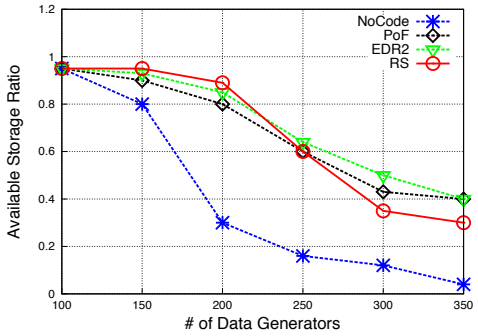


Fig. 8. Available storage space ratio v.s. amount of data generators (storage=1MByte, failure prob. = 20%).

D. Energy cost

Energy cost is critical to the data preserving, since excessive energy consumption will increase the nodal failure probability due to energy depletion. Illustrated in Figure 6, all the algorithms experience a performance degradation under higher failure probability. Therefore, energy conservation is not only for the sake of efficiency, but also for maximizing data persistence.

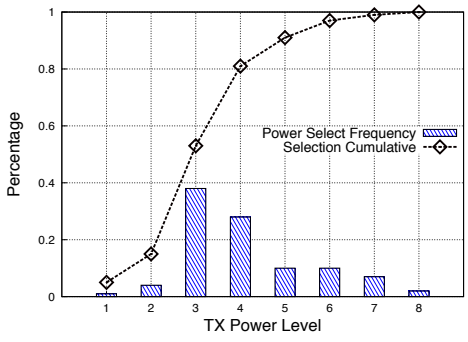


Fig. 9. Transmission Power Level Distribution.

The energy cost increment is exponentially proportional to transmission power level. Thus, we evaluate the experimental output of power level adaptation algorithm in Figure 9. Figure

9 represents the transmission power probability distribution according to Equation 3. The diamond-dotted curve is the cumulative probability of power selection. It shows that more than 70% probability fall into power level 3 and 4, which are low-moderate power level for the radio. This results is consistent with the preliminary results from Figure 2. It also depicts that adaptive power control can adapt to network dynamics, including neighbor node failure, node storage space changes and node density alteration, with small transmission power cost.

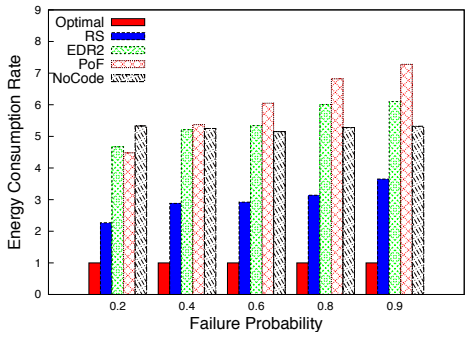


Fig. 10. Energy Consumption Rate (storage=$1M$Byte).

Figure 10 demonstrates total energy consumption of different algorithms, normalized against the *Optimal* solution. The optimal solution is assumed to establish a stable delivery path from distribution node to storage node, which only consumes minimum energy cost. In low failure probability, *RS* only expend 2.3 times as *Optimal* solution. It is due to that in *RS* probabilistic rebroadcasting is applied to accommodate the disruptive connection. Compared with *Optimal*, more redundant nodes are required to rebroadcast packets in the absence of global knowledge of node failure prediction. In the worse case, *RS* consumes 3.5 times as much as *Optimal* solution. Although *RS* adjusts its transmission power for better expected coverage, its total energy consumption is 40% less than *EDR2* and *PoF*. The packet retransmission in those two approaches consume considerable energy when the path for data redistribution becomes intermittent. *NoCode* randomly rebroadcasts the packet for redistribution, which is unaware of network dynamics, so that energy consumption of *NoCode* remains 5 times as much as the *Optimal* solution. But the data preserving performance of *NoCode* is dramatically influenced by failure probability as in Figure 6.

V. Conclusion

An integrated network coding based approach, *Ravine Streams* (*RS*), has been proposed in this work to achieve persistent data streams in disruptive wireless sensor networks. RS maximizes the probability that source symbols seep through multiple disruptive paths, via probabilistic rebroadcasting with adaptive power control and recoding, preserving the persistence of data stream. The experimental results reveal that the delivery ratio of source symbols (data stream persistence) is improved considerably with efficient energy cost and modest storage cost.

References

[1] X. Hou, Z. Sumpter, L. Burson, X. Xue, and B. Tang, "Maximizing Data Preservation in IntermittentlyConnected Sensor Networks," in *IEEE International Conference on Mobile Ad hoc and Sensor Systems*, Oct. 2012.

[2] M. Valero, M. Xu, N. A. Mancuso, W.-Z. Song, and R. Beyah, "EDR2: A Sink Failure Resilient Approach for WSNs," in *IEEE International Conference on Communications*, Jun. 2012.

[3] A. Kamra, V. Misra, J. Feldman, and D. Rubenstein, "Growth Codes: Maximizing Sensor Network Data Persistence," in *ACM SIGCOMM*, Sep. 2006.

[4] S. Chachulski, M. Jennings, S. Katti, and D. Katabi, "Trading structure for randomness in wireless opportunistic routing." in *ACM SIGCOMM*, Aug. 2007.

[5] Y. Lin, B. Liang, and B. Li, "SlideOR: Online Opportunistic Network Coding in Wireless Mesh Networks," in *Mini-Conference at IEEE INFOCOM 2010*, Mar. 2010.

[6] A. G. Dimakis, V. Prabhakaran, and K. Ramchandran, "Distributed fountain codes for networked storage," in *Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2006.

[7] S. Biswas and R. Morris, "ExOR: Opportunistic Multi-hop Routing for Wireless Networks," in *Proceedings of the Special Interest Group on Data Communication (SIGCOMM'05)*, ser. SIGCOMM '05, vol. 35, no. 4. Philadelphia, Pennsylvania, USA: ACM, Oct. 2005, pp. 133–144. [Online]. Available: http://dx.doi.org/10.1145/1080091.1080108

[8] D. Koutsonikolas, C.-C. Wang, and Y. C. Hu, "CCACK: Efficient Network Coding Based Opportunistic Routing Through Cumulative Coded Acknowledgments," in *IEEE INFOCOM*, Mar. 2010.

[9] S. Puducheri, J. Kliewer, and T. E. Fuja, "Distributed LT Codes," in *International Symposium on Information and Theory*, Sep. 2006.

[10] ——, "On the Performance of Distributed LT Codes," in *Allerton Conf. Communications, Control and Computing*, Sep. 2006.

[11] M. Xu, W.-Z. Song, and Y. Zhao, "Opportunistic Network Erasure Coding in Disruptive Sensor Networks," in *The 8th IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (IEEE MASS'11)*, 2011.

[12] Y. Lin, B. Liang, and B. Li, "Data Persistence in Large-scale Sensor Networks with Decentralized Fountain Codes," in *Proceedings of the 26th Annual IEEE International Conference on Computer Communications (IEEE INFOCOM)*, May 2007.

[13] S. A. Aly, Z. Kong, and E. Soljanin, "Fountain codes based distributedstorage algorithms for large-scale wireless sensor networks," in *Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN)*, Apr. 2008.

[14] ——, "Raptor Codes Based Distributed Storage Algorithms for Wireless Sensor Networks," in *IEEE International Symposium on Information Theory*, Jul. 2008.

[15] D. Vukobratovic, C. Stefanovic, V. Crnojevic, F. Chiti, and R. Fantacci, "A Packet-Centric Approach to Distributed Rateless Coding in Wireless Sensor Networks," in *Proceedings of IEEE SECON*, Jun. 2009.

[16] B. Tang, N. Jaggi, H. Wu, and R. Kurkal, "Energy-Efficient Data Redistribution in Sensor Networks," in *MASS*, Aug. 2010.

[17] M. Takahashi, B. Tang, and N. Jaggi, "Energy-Efficient Data Preservation in Intermittently Connected Sensor Networks," in *The Third International Workshop on Wireless Sensor, Actuator and Robot Networks*, Apr. 2011.

[18] J. Liang, J. Wang, X. Zhang, and J. Chen, "An Adaptive Probability Broadcast-based Data Preservation Protocol in Wireless Sensor Networks," in *IEEE International Conference on Communications*, Jun. 2011.

[19] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications," in *Proceedings of the 1st ACM Conference on Embedded Networked Sensor Systems (SenSys'03)*, 2003.

[20] M. Xu, W.-Z. Song, and D. Heo, "OnCode: Opportunistic In-Network Coding and Delivery in Energy-Synchronized Sensor Networks," available at http://sensorweb.cs.gsu.edu/~xum/pub/OnCodeTR.pdf, Tech. Rep., May 2012.