Research on Data Storage Scheme Under Sink Failures in Wireless Sensor Networks

Yue Wang and Jun Wang(✉)

Department of Communication and Information Engineering,
Nanjing University of Posts and Telecommunications, Nanjing 210003, Jiangsu, China
980941416@qq.com, wang_jun@njupt.edu.cn

Abstract. In remote and inaccessible environment, sensory data must be stored inside the network in case of sink failures. Since all sensor nodes have limited storage capacity and energy, so we need to ensure that the most important and urgent data can be stored and decoded first. In this paper, we studied the data storage problem in sink-failures sensor networks. Considering that most existing algorithms mainly focus on how to maximized number of stored data, which makes the loss of the most critical data, so we design a novel network coding data storage scheme based on priority named NCSP (Network Coding Storage with Priority). In this scheme, in order to prevent the loss of the most critical information, data in the networks are divided into independent priority groups. Different network coding schemes are used in different groups. Aggressiveness mechanism is also considered in this paper. Finally, MATLAB simulations results demonstrate that NCSP outperforms than other algorithms in terms of decoding priority.

Keywords: Wireless sensor networks · Network Coding Storage
Priority mechanism · Fountain code · LT code · iLT code

1 Introduction

Many sensor network applications are deployed in hash environment (such as remote or unattended regions) and need to take a long time to monitor a large number of data, audio, video, images, etc. The network lifetime should be as long as possible until the specific monitoring tasks are completed. However some nodes including sink nodes may fail due to a variety of reasons (battery depletion, enemy attacks and challenge environment). In the case of sink nodes failure, connections between common sensor nodes and sink nodes are intermittent, therefore common sensor nodes are required to collect and store data before replacing or renewing the failed sinks. Therefore the research on the failure of nodes in wireless sensor networks has aroused people's concern and attention.[1]

[1] This work is supported by National Nature Science Foundation of China 61401234, 61271234, and PAPD Project of Jiangsu Higher Education Institutions.

M. Huang et al. (Eds.): WICON 2016, LNICST 214, pp. 27–35, 2018.
https://doi.org/10.1007/978-3-319-72998-5_4

In order to find a better solution of the data storage problem of sink-failures sensor networks in challenging environment, in this paper we propose a more applicable distributed data storage scheme. The main contribution of this work is as follows. Firstly, we study different storage schemes for wireless sensor networks (WSNs) in hostile environments and find the distributed coding scheme based on fountain codes is the most suitable scheme for data storage in wireless sensor networks. Secondly, in order to protect the most critical data, we design a network coding data storage scheme based on the data's priority, NCSP. The simulation results show it is more effective than other schemes in terms of decoding priority. Lastly, we also consider aggressiveness mechanism to subdivide the priorities of data.

The rest of this paper is organized as follows. Section 2 presents the related works. In Sect. 3, a new network coding data storage scheme based on priority is proposed. In Sect. 4, simulation results and analysis are given. Section 5 concludes the paper and gives suggestions for further study.

2 Related Works

In this paper, we divide different storage schemes into several classes, as shown in Fig. 1.

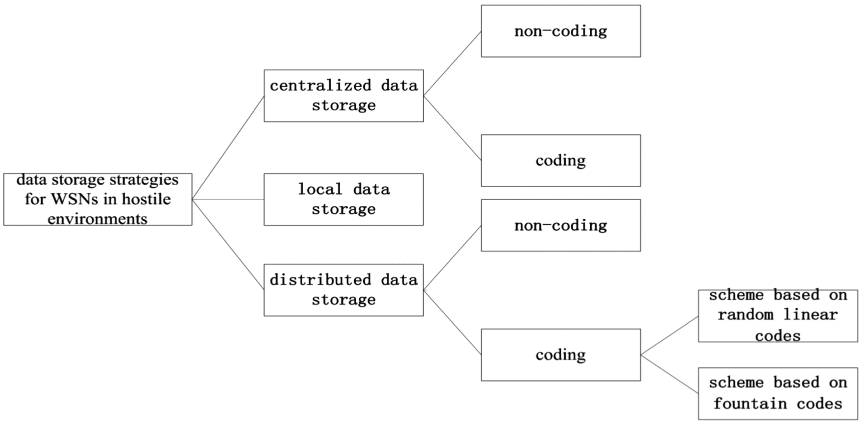
data storage strategies for WSNs in hostile environments
centralized data storage
local data storage
distributed data storage
non-coding
coding
non-coding
coding
scheme based on random linear codes
scheme based on fountain codes

Fig. 1. Classification of WSNs storage schemes.

The centralized non-coding data storage scheme is controlled by the central coordinator. In [1], authors tried to assign different sensory data with different priorities and aim to maximize the preserved data priorities. Besides, data priorities were specified in advance. In reality, the data characteristics of different sensor networks are different. The priority of the data should be arranged according to the unique characteristics of the data. In the local data storage scheme, sensing data are only stored by sensor nodes in their own memories. The user will broadcast the querying request to the whole sensor network. In the distributed non-coding data storage scheme, data distribution is dependent on the cooperative communication mechanism between nodes. So, it is suitable for the scenario where the network topology is changed dynamically. Reference [2] mainly focused on how to increase the lifetime of the data stored in the network. By redistributing data from the nodes with low energy to those with high energy, the algorithm slowed the loss of data to a certain extent. But they assumed that the sensor nodes only had a unit of storage and the source node only had one item of data. Such a network was too simple to exist in the reality.

In some cases, such as the limited resources of sensor nodes, the distributed code data storage is easier and more efficient than the centralized code data storage. According to the different coding method, the distributed coding data storage is classified into two categories: schemes based on random linear codes and schemes based on fountain codes.

The complexity of encoding and decoding is also high. Therefore, random linear codes [3, 4] are not suitable for the sensor nodes with limited energy and computing power. PRLC [3] proposed a random linear coding scheme based on priority to process data with different priorities. The main idea was to assign different coding rates for data with different priorities. That is, the lower coding rate was distributed to the data with higher priority. But in this scheme, the transmission of data packets was dependent on geographic routing. In addition, PRLC could not guarantee that all data were successfully decoded.

The retrieval of data only occurs in a small area at most time. GRLC [4] proposed a random linear code scheme based on the geographical location. According to the size of the geographic area, data in different geographic regions were stored by hierarchical coding. But GRLC was also dependent on geographic routing.

Fountain codes are easy to realize in wireless sensor networks, because the complexity of coding and decoding is relatively low. Therefore, fountain codes are very suitable for data storage in wireless sensor networks.

We show the comparison of different data storage schemes as follows (Table 1):

Table 1. Comparison of different data storage schemes.

Classification	Data storage scheme		
	Centralized data storage	*Local data storage*	*Distributed data storage*
Advantages	Simple structure, convenient query	Simple structure and query, no data forwarding overhead	Simple and fast query, suitable for distributed sensor networks
Disadvantages	A bottleneck, vulnerability	High communication overhead, short network lifetime, easily loss of data	Mapping to generate additional overhead

In summary, the distributed coding scheme based on fountain codes is the most suitable scheme for data storage in wireless sensor networks. So we use different fountain codes with different characteristics in our proposed scheme. In the existing algorithms, the priority of the data was always arranged in advance. And they also assumed that the energy is infinite. But in this paper, data priorities are dynamically allocated in accordance with specific conditions. The initial energy and storage space of nodes are limited and can't be added. In additional, it is dependent on cooperative communication between nodes instead of geographic routing.

3 Network Coding Data Storage Scheme Based on Priority (NCSP)

Network model: There are n sensor nodes randomly deployed in a square area of L * L, and an intermittent connection of sink node and sensor nodes. The packet size is equal in length. The number of iterations is k. The channel of the coding data packets is BEC. The packet loss rate is q = 0.05. Belief Propagation is used.

In this paper we made the following assumptions:

(1) The source data package is modeled as a sequence of data items and each data item has the same unit size.
(2) Since sensor nodes are uniformly distributed, the energy of transfer any one data between any one hop is regarded as 1 unit in the entire network. For each node, sending or receiving a data item consumes the energy of 0.50 unit. If it is the intermediate node, forwarding a data item consumes the energy of 1 unit (both receiving and sending).
(3) For an arbitrary sensor node *i*, its initial energy and storage space are limited. Due to harsh external environment, the node also has a failure ratio of 0.40.

According to the idea of the algorithm, the flow chart is constructed as shown in Fig. 2:

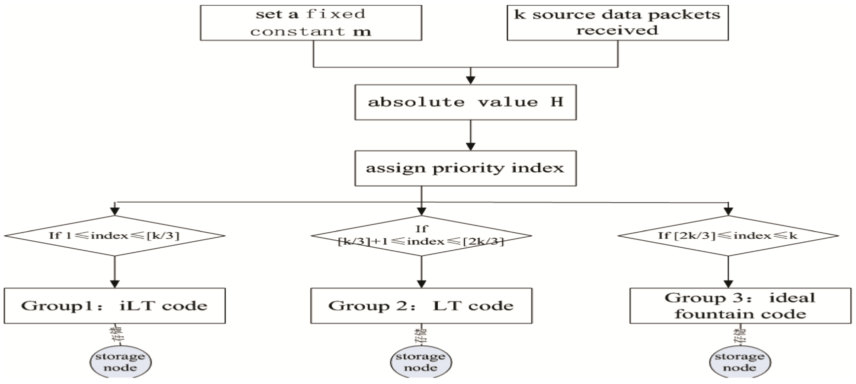

Fig. 2. Flow chart of the algorithm.

In this paper, in order to maximize the delivery ratio of the most important data, we propose a new data storage scheme based on both network code and data priority. We divide data into different groups according to the priority, and use different coding methods in the different group, so as to ensure the important data to be better protected.

The process of the algorithm is divided into the following stages:

3.1 Coding Preparation Stage

According to the different information that source data packets carried, different priorities for each packet are assigned.

According to the different quantities and values of information that each packet carried, we assign corresponding priority for each packet. The priority is stored in the head of each packet. Priority is in reverse proportion to the importance of the packet which means that the most important data will have the minimum priority and will be stored and decoded firstly.

We use the sorting method to assign the priority of data. In initial state, a fixed constant *m* is given, which indicates the average or normal value of a certain measurement (the value is related to the specific application of the sensor networks). And the first received data's priority is set to 1 (index = 1). Then the second received data's priority is set to 2 (index = 2). The priority is related to the data's deviation from the standard value. Defining the value of *H* is equal to the absolute value of the difference between the measurement value of each data packet and *m*. If the *H* value of the second data packet is greater than that of the first data packet, the priorities of the data are exchanged which means the priority of the first packet is set to 2. If the *H* of the second packet is smaller than that of the first packet, the priority of the second packet is set to 2. In the same way, we can assign corresponding priority for each packet.

3.2 Network Coding Stage

The priorities of the data are classified into three levels, which are coded by different coding scheme. And then they are stored in the sensor nodes in the network.

Unlike other network coding methods based on priority, we divide them into different groups according to the priority of sensor data. Encoding and decoding of important data with low decoding overhead and low error rate can ensure the success ratio of storage and retrieval of important data. According to the priority ranking, the data is divided into three groups. In different groups, we use different coding methods. Each packet is encoded and decoded in its own group, and is not affected by the other groups. We divide the data into three groups, and adopt ideal fountain code, LT [5] code and iLT [6] code respectively to encode data. For the data sets with the highest priority, the iLT code with the highest decoding rate and the lowest overhead is adopted. For the data sets with the second highest priority, the LT code with higher decoding rate and lower overhead is adopted. For the data sets with the lowest priority, the ideal fountain code with the lowest decoding rate and the highest overhead is adopted. In harsh environments, we can't completely and accurately decode all sensor data. In our scheme, it can be guaranteed that the data with highest priority will be accurately decoded with the fastest speed. It can also make the network decoding more efficiently.

3.3 Decoding Preparation Stage

The corresponding aggressiveness is assigned to each node.

In order to differentiate the data in the same group, and to optimize coding efficiency, the concept of "aggressiveness" is introduced in our scheme. "Aggressiveness" indicates the buffer size at the moment the node starts decoding. It is defined by the number of blocks it needs to buffer before starting to generate encoded blocks, normalized by the buffer size. By dynamically adjusting the "aggressiveness" value of higher priority data can shorten buffering latency caused by data accumulation. The aggressiveness value in our scheme is a variable. The value of priority is proportional to that of the aggressiveness. So we set the value of aggressiveness(i) as:

$$aggressiveness(i) = t \bullet index(i) \tag{1}$$

In formula (1) t is a constant (t = 2 in simulation). Since the coding data packet is always encoded by several source data packets, and each source packet carries an aggressiveness value. The total aggressiveness value follows the formula (2):

$$aggressiveness = \sum 2^{aggressiveness(i)} \tag{2}$$

The buffer size of storing the value of aggressiveness is relatively small compared with the whole storage size, so it can be neglected (Fig. 3).

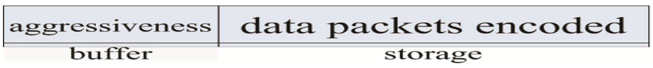

Fig. 3. Data structure of node.

3.4 Decoding Stage

After processing the corresponding aggressiveness, nodes start to decode the coding packets according to their encoding method.

3.5 Analysis of the Algorithms

The proposed algorithm has a great performance in terms of decoding ratio of data with higher priorities. The performance can be improved by introducing the aggressiveness mechanism. Besides, as the number of nodes increases, the improvement on decoding rate becomes more obvious. Meanwhile, the promotion on decoding rate remains at a high level. It indicates when there are only several nodes in the WSNs, the contention between different priorities is not as tough as that in a large number of nodes. The results indicate that the proposed algorithm performs better in a multi-node environment.

From a theoretical analysis, different coding methods may bring some extra overhead, but the number of important data which are successfully stored and decoded will be significantly improved.

4 Performance Evaluation and Analysis

To verify the performance of the proposed algorithm, we conduct simulation experiments in the MATLAB environment.

The parameters are set as follows: Source data packets are generated randomly for encoding and transmission, which the length of source data packets is 500. The number of iteration is 5000, c = 0.20, $\delta = 0.80$. The channel of the coding data packets is BEC and the packet loss rate is q = 0.05. Belief Propagation is used. The node failure rate is set as 0.40.

4.1 Influence of the Value of Aggressiveness

Average download time is the ratio of the average decoding time to the total time. Decoding overhead is the ratio of the decoding overhead to the total overhead.

The parameters are set as follows: Source data packets are generated randomly for encoding and transmission, the length of source data packets k = 3000, constants of fountain codes c = 0.20, $\delta = 0.80$. The channel of the coding data packets is BEC channel. The packet loss rate is q = 0.05. Belief Propagation is used. The simulation result is showed in Fig. 4.

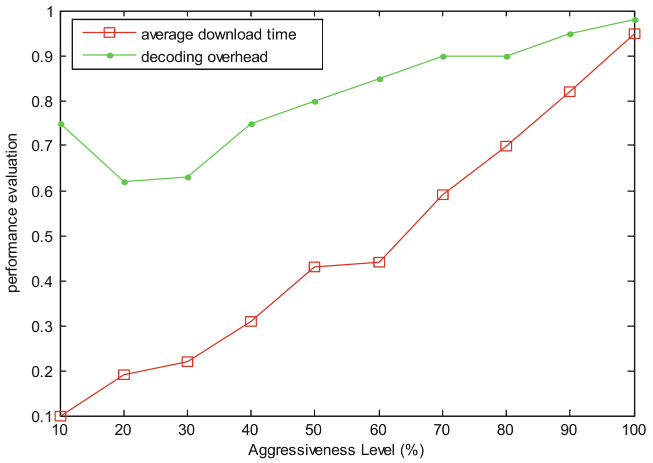

Fig. 4. Influence of aggressiveness.

In Fig. 4, we set aggressiveness from 10% to 100% to observe its impact to the above two parameters. As the value of aggressiveness increases, the average download time linearly increases which verified our analysis above—the nodes with smaller "aggressiveness" value will be decoded firstly. Also, decoding overhead will increase with the increase of aggressiveness. However, an extremely low aggressiveness value is also unsuitable. As shown in the graph, when the aggressiveness value is below 20%, the decoding overhead shows speed-up deterioration with the decrease of aggressiveness, which will accelerate the energy consumption. From the above analysis, we can conclude that the system have the best performance with the aggressiveness value ranges from 20% to 50%.

4.2 Comparison of Different Coding Schemes

Next, we compare the performance of the following 5 schemes: only ideal fountain code (fountain), only LT code (LT), only iLT code (iLT), the coding scheme in this paper but no aggressiveness (ft+LT+iLT), the coding scheme in this paper and aggressiveness (ft +LT+iLT+aggr). The simulation result is as follows:

Shown in Fig. 5, the decoding ratio of LT code is better than that of ideal fountain code. iLT code outperforms the LT code in terms of decoding ratio. There is no obvious advantage in the decoding ratio of ft+LT+iLT when very few data packets have been transmitted. As the number of data packets sent increases, the performance on decoding ratio becomes more obvious. When more than 4000 packets have been sent, the decoding ratio is better than that of the other three methods. When the aggressiveness is introduced in the algorithm, the decoding performance can be improved.

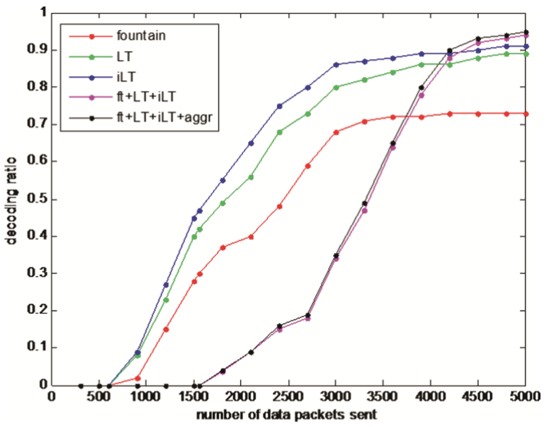

Fig. 5. Decoding rate of different coding schemes.

In fact, our new scheme is aim to optimize the decoding ratio of the higher priority data. Therefore, we test the decoding data packet ratio (the ratio of the data packets decoded successfully to the number of source data packets) and the decoding priority ratio (the ratio of the priorities decoded successfully to the number of total priorities) in different scenarios.

As it can be seen from Fig. 6, the proposed algorithm has a great performance on decoding ratio of higher priority data. Compared with the ideal fountain code and LT code, the total data storage capacity of iLT code is the largest. Meanwhile, the decoding ratio of the high priority data has also been increased. The proportion of high priority data which successfully decoded is significantly improved in ft+LT+iLT compared with iLT. But the total amount of data stored has decreased. When the aggressiveness is introduced, it provides more protection of the important data. When the data in the same priority group can't be completely decoded, it adjusts the "aggressiveness" value of the important data in order to have a higher probability of successful decoding. So the successful decoding ratio of high priority data is improved further.

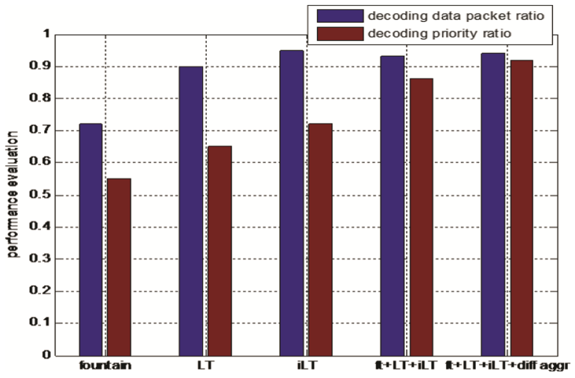

Fig. 6. Storage performance of different schemes.

5 Conclusion and Future Work

In this paper, we propose a new network coding data storage scheme based on priority to ensure the decoding ratio of the highest priority data. To a certain extent, it improves the total storage capacity, and ensures the priority storage of important data. But when the amount of data is very small, there is no obvious advantage in the decoding rate. We will explore new scheme to improve the overall decoding rate in our future work.

References

1. Xue, X., Hou, X., Tang, B., Bagai, R.: Data preservation in intermittently connected sensor networks with data priority. In: 10th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, pp. 122–130 (2013)
2. Takahashi, M., Tang, B., Jaggi, N.: Energy-efficient data preservation in intermittently connected sensor networks. In: IEEE Conference on Computer Communications Workshops, Shanghai, pp. 590–595 (2011)
3. Lin, Y., Li, B., Liang, B.: Differentiated data persistence with priority random linear code. In: Proceedings of the 27th International Conference on Distributed Computing Systems, Piscataway, NJ, pp. 47–56. IEEE (2007)
4. Lin, Y., Liang, B., Li, B.: Geometric random linear codes in sensor networks. In: Proceedings of IEEE International Conference on Communication, Piscataway, NJ, pp. 2298–2303. IEEE (2008)
5. Ye, X., Li, J., Chen, W.-T., Tang, F.: LT codes based distributed coding for efficient distributed storage in wireless sensor networks. In: 2015 IFIP (2015). ISBN 978-3-901882-68-5
6. Yang, J., Zhao, Z., Zhang, H.: Energy efficient data gathering based on distributed iLT coding. In: The 11th International Symposium on Communications & Information Technologies (ISCIT 2011) (2011)