A Multi-Loss Flexible Regenerating Code Recovery Scheme Using Network Coding For Distributed Storage Wireless Sensor Network
(複数消失に対応する再生成符号を用いたワイヤレスセンサネットワーク上の分散ストレージの復元方法)

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL OF
THE UNIVERSITY OF TOKYO
BY

Chenchao Li

李 晨超

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF PHILOSOPHY

Kaoru Sezaki

指導教員 瀬崎 薫

February 2013

ACKNOWLEDGEMENT

I would like to express my great appreciation to my supervisor Professor Sezaki Kaoru and Teaching Assistant Masayuki Iwai. They guided me all the way to the summit of this research with valuable motivation and advice. They also helped me publishing my paper in domestic conferences It is great that they are advisors of this thesis and they had patience to hear my research and also have detailed worthwhile discussions which improved thesis quality.

I am thankful to Visiting Professor Dr. Yunlong Zhao and Dr. Hongyang Chen, who helped me a lot at the very beginning of my research life.

I am grateful to Dr. Arnold S. K. Han, Dr. Hiroki Ishizuka, Dr. Khan Muhammad Asif Hossain, Dr. Dang Congwei, Dr. Liu Guangwen and Dr. Niuhao, who gave me lots of advice and keen comments in the meeting.

I never forget to express my gratitude to cohorts Mr. Kazuto Shimizu, Mr. Junya Okuno, Mr. Shunpei Nakayama and Mr. Wang ShaoZhe. We always give support and encouragement to each other.

I wish to express my appreciation to Mr. Jose Alvarez Pabl, Mr. Yuichiro Kida, Mr. Shunsuke Aoki and Mr. Keijiro Nakagawa who give me great support in the spirit.

Moreover, I wish to send my best wishes to two secretaries Ms. Kaho Matsumoto, Ms. Jun Naito and Ms. Asuka Tsubono. In their efforts to work, they developed a good research environment for students.

Finally, I am much obliged to my family for caring about my health and my friends for bring happy day and night. They always stood by me and supported me throughout my entire life.

Table of Contents

CONTENTS

CONTENTS

1. Introduction

Wireless sensor network (WSN) arises rapidly during these years. With the help of WSN, we can enjoy a more comfortable and convenient life than previous. And recently, with the advent of network coding, we can make nodes to not only do normal forward but also process the incoming the independent information flows which can make a totally revolution against the traditional way.

We start our chapter one from the brief introduction of WSN, the security concern and requirement need to solve. Then in order to help the understanding of network coding, we give a short passage to explain the idea. In the end, we propose our research goal, mention our contribution and give a construction of our paper.

1.1. Wireless Sensor Network (WSN)

With the development of Micro-Electro-Mechanical-System (MEMS) technology, a number of sensors, limited-range wireless communication chip, Micro Control Unit (MCU) and memory, we generate a new technology named as Wireless Sensor Network (WSN) as illustrated in Fig. 1. [1]

WSN integrates sensor technology, embedded technology, computer technology and communication technologies which means it can cooperatively monitor environmental conditions like temperature, humidity or pressure in real-time, process retrieved sensing data, obtain detailed and accurate information, transfer processed or unprocessed information to one or more base stations, which act as gateway between sensor nodes and end-users or administrators.

In WSN, sensor nodes work under resource-constraint conditions, for instance, limited battery power, computing power, communication range and wireless traffic. Since the sensor nodes have limited memory and are typically deployed in difficult-to-access locations, a radio is implemented for wireless communication to transfer the data to a base station. Battery is the main power source in a sensor node. Secondary power supply that harvests power from the environment such as solar panels may be added to the node. Depending on the application and the type of sensors used, actuators may be incorporated in the sensors.

Recently, WSNs have shown great potential for many applications in lots of scenarios such as military target tracking and surveillance, natural disaster relief, biomedical health monitoring,

1. INTRODUCTION

and hazardous environment exploration and seismic sensing. For instance, with natural disasters, nodes can sense and detect the environment to forecast disasters before they occur. In biomedical applications, surgical implants of sensors can help monitor a patient's health. For seismic sensing, ad hoc deployment of sensors along the volcanic area can detect the development of earthquakes and eruptions. All these fields are so interesting and important that need us to pay plenty of focus on.

Unfortunately, WSN has its own design and resource constraints when compared with traditional network. The main resource constraints include a limited amount of energy, short communication range, low traffic, and limited processing and storage in each node. On the other side, design constraints are application dependent and are based on the monitored environment. The environment plays a sticking point in determining the size of the network, the deployment scheme, and the network topology. The size of the network varies with the monitored environment. An ad hoc deployment is preferred over pre-planned deployment when the environment is inaccessible by humans or when the network is composed of hundreds to thousands of nodes. Obstructions in the environment can also limit communication between nodes, which in turn affects the network connectivity.

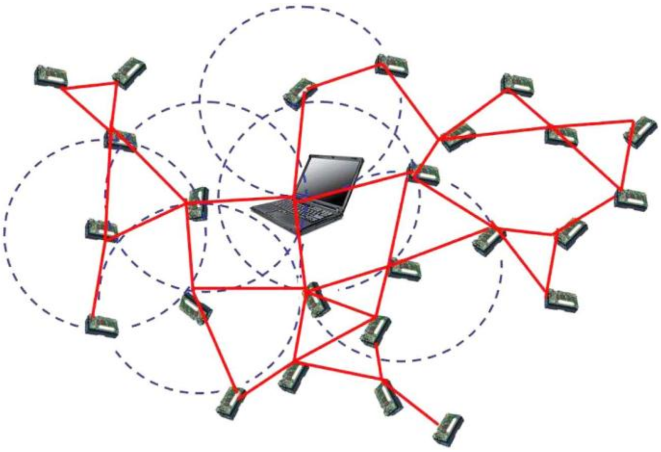


Fig.1. Wireless sensor network is a collection of small devices

1.2. Security Concern and Requirement in WSN

Wireless sensor networks (WSNs) have recently attracted a lot of interest in the research community because of their wide range of applications. Since distributed nature of these networks and their deployment in remote areas, these networks are vulnerable to numerous security threats that can adversely affect their proper functioning. This problem is more critical if the network is deployed for some mission-critical applications. Random failure of nodes is also very likely in real-life deployment scenarios. Due to resource constraints in the sensor nodes, traditional security mechanisms with large overhead of computation and communication are infeasible in WSNs. Security in sensor networks is, therefore, a particularly challenging task.

1.2.1. Security Requirements in WSN

As we mentioned above, WSN is a very special network. Compared with the traditional network, it has lots of limitations and constraints. Due to these disadvantages, it is difficult to directly employ the conventional security mechanisms in WSN. In order to optimize the conventional security algorithms for WSNs, it is necessary to be aware about the constraints of sensor nodes. We need to change the tactics and algorithms in order to adjust to the needs of WSN. However, WSN also have high requirements for security parts just same as normal networks. The most important security requirements in WSN are listed below:

(1) Data confidentiality: In WSN, the issue of confidentiality should address the following requirements: (i) a sensor node should not allow its readings to be accessed by its neighbors unless they are authorized to do so, (ii) key distribution mechanism should be extremely robust, (iii) public information such as sensor identities, and public keys of the nodes should also be encrypted in certain cases to protect against traffic analysis attacks.

(2) Data integrity: The mechanism should ensure that no message can be altered by an entity as it traverses from the sender to the recipient.

(3) Data freshness: It implies that the data is recent and ensures that no adversary can replay old messages. This requirement is especially important when the WSN nodes use shared-keys for message communication, where a potential adversary can launch a replay attack using the old key as the new key is being refreshed and propagated to all the nodes in the WSN. A nonce or time-specific counter may be added to each packet to check the freshness of the packet.

(4) Time synchronization: Most of the applications in sensor networks require time synchronization. Any security mechanism for WSN should also be time-synchronized. A collaborative WSN may require synchronization among a group of sensors.

1.2.2. Privacy Vulnerabilities in WSN

Sensor network technology promises a vast increase in automatic data collection capabilities through efficient deployment of tiny sensor devices. While these technologies offer great benefits to users, they also show significant potential for abuse. Particularly relevant concerns are privacy problems, since sensor networks provide increased data collection capabilities. Adversary users can use seemingly innocuous data to derive sensitive information if they know how to correlate multiple sensor inputs. For example, For example, in a houses sensor network, we use nodes to collect statistics about water, humidity and electricity consumption within a large neighborhood in order to get resources planning purposes and usage advice. But try to imagine that a bad guy attacks your nodes and your private information is leaked. The adversaries will know your personal activities just like when you take a shower, when all family members are gone.

The main privacy problem is not that WSN enable the collection of information. In fact, much information from sensor networks could probably be collected through direct site surveillance. Furthermore, WSNs aggravate the privacy problem because they make large volumes of information easily available through remote access. Hence, adversaries need not be physically present to maintain surveillance. They can collect, rewrite or overhear information in a low-risk, anonymous way. Remote access also allows a single adversary to monitor multiple sites simultaneously. Some of the more common attacks against sensor privacy in WSN are:

(1) Monitor and Eavesdropping: This is the most obvious attack to privacy. By listening to the data, the adversary could easily discover the communication contents. When the traffic conveys the control information about the sensor network configuration, which contains potentially more detailed information than accessible through the location server, the eavesdropping can act effectively against the privacy protection.

(2) Traffic Analysis: It typically combines with monitoring and eavesdropping. An increase in the number of transmitted packets between certain nodes could signal that a specific sensor has registered activity. Through the analysis on the traffic, some sensors with special roles or activities can be effectively identified.

(3) Camouflage: Adversaries can insert their node or compromise the nodes to hide in the sensor network. After that these nodes can masquerade as a normal node to attract the packets, then misroute the packets, for instance, forward the packets to the nodes conducting the privacy analysis.

1. INTRODUCTION

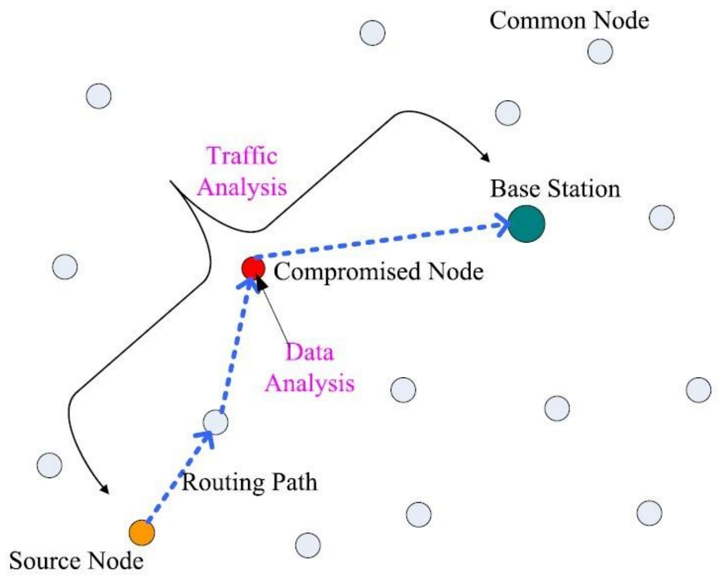


Fig.2. Two scenarios for privacy attack in WSN

1.3. Network Coding

1.3.1. Theory of Network Coding

Network coding (NC) was a new coding conception first proposed in 2000 [4], which has attracted much attention in coding field. The basic principle of network coding is illustrated in Fig. 3. Network coding breaks the traditional routing principle in current communication network that information can only be stored and forwarded separately without superposition, and allows network nodes to code the incoming data with appropriate coding methods, such as exclusive or (XOR) operation, linear operation [5] and so on, so as to achieve the maximum transmission capacity defined by the "Max-flow Min-cut" theorem of Shannon. In Fig.2, we can observe that to exchange messages a, b, nodes A and B must route their packets through node S. Clearly, the traditional scheme in Fig.3 (a) would require four transmissions. However, if S is allowed to perform network coding with simple XOR operations, $a \oplus b$ can be sent, as shown in Fig. (b), in one single broadcast transmission (instead of one transmission with b followed by another one with a). By combining the received data with the stored message, A, which possesses a, can recover b and B can recover a using b. Thus, network coding saves one transmission.

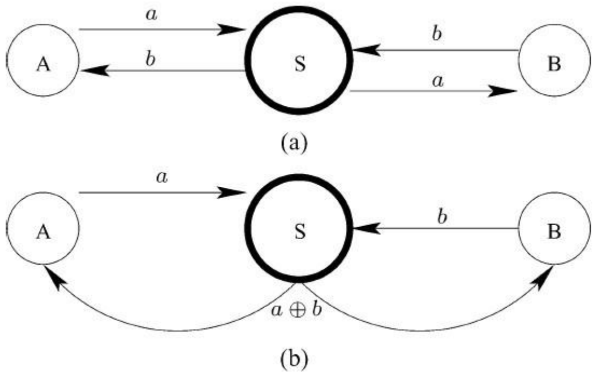


Fig.3. Traditional transmission and network coding transmission

1.3.2. Benefit about Network Coding

Based on developments of different network coding methods with stored and forwarded routing, NC can bring in lots of benefits, especially in WSN applications:

(1) Energy saving: A multi-node wireless broadcast retransmission scheme, using NC to code the required retransmission packets as one NC packet according to the statistics for packet loss of different nodes, can greatly reduce the retransmission time thus saves huge transmission energy [6]. NC was applied to the 2-hop information exchange in WSN to promote the efficiency of 2-hop coverage during the information exchange, which then saves energy. Xiong et al. proposed an intra-cluster information exchange method based on the analysis of a cluster topology model. All the simulation results show that NC based information exchange can save a lot of energy compared with other methods, which then prolong the lifetime of energy limited WSN.

(2) Throughput enhancement: It is proved that NC has great effect on increasing the network throughput [4]. The simple example is as shown in Fig.3. Besides, Petrovic et al. proposed a distributed RNC based strategy without signal modulation, which can achieve the same throughput as the modulated strategy and save the energy consumptions of modulation. Alberto et al. developed a new routing method for WSN by combining RNC with diffuse algorithm, which can adjust and promote the throughput and bit rate of WSN meanwhile reducing the time of packet retransmission so as to save energy. All those results illuminate that no matter uniform or non-uniform link it is, NC can achieve a higher multicast capacity.

(3) The robustness: NC can enhance the robustness and error restoration of WSN by storing

NC coded packets in distributed network storage, by enhancing the error correction ability according to the packet loss rate, by coping with the dynamic node joining and leaving and so on.

(4) Network load balance: NC can efficiently use the rest links other than multicast tree paths, to distribute the network traffic in a wider range of network, so as to balance the network load and prolong the lifetime of each node. As shown in Fig.4, in (a), all the paths` capacity is 2 and by routing multicast shown in (b), in order to make all sinks reach the max capacity, SU, UX, UY, SW,WZ are utilized. Each feasible flow is 2 and the number of paths is 5. In (c), by multicast using NC, we assume that the information in SV is coded as $a \oplus b$, then all the information in SV, VX, VZ is $a \oplus b$ and each sink can get a, b finally. Compared to (b), we use 4 more paths. In other words, we balance the network load.

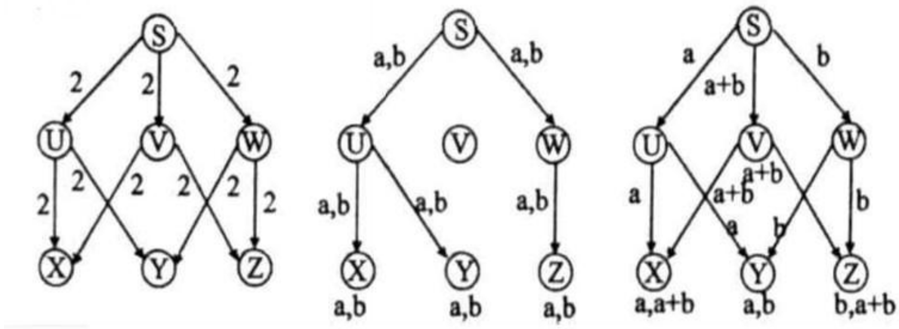


Figure 4.one source and three sinks networks

(5) The security: Without complex encryption algorithm, NC can improve the network security easily. And this is also the main part we want to apply in our research, I will explain this part in detail in the future chapters.

1.4. Objective and Contribution of Research

In order to address some security issues existing in wireless sensor distributed storage network, in our master thesis, we try to use network coding in WSN.

Objective of the research: The goal of our research is to apply network coding with some existing codes to achieve the secure and dependable distributed data storage scheme in order to increase the data reliability, reduce traffic cost and easy to implement in wireless sensor

networks.
Contributions: We propose a data distribution scheme including our particular network flow diagram and network coding. Our scheme is resilient for data fault and node compromise, as well as maintaining storage efficiency. In detail, by taking advantages of above schemes, we have computational security and maintain the optimal data size. The extensive analysis verifies our scheme can provide secure and dependable data storage in WSNs in the presence of node compromise, and remain efficiency in terms of storage, computation and communication.

1.5. Organization of Thesis

The remainder of our rest paper is organized as follows:
Chapter two: we will describe the network coding theory mathematically and briefly. By given the explanation of linear network coding and random network coding, the principle and working mechanism of network coding will be shown. Then followed by two applications using practical network coding in WSN environment, we will demonstrate how network coding performances over different fields in practical world.
Chapter three: By using network flow theory, we will turn the distributed storage recovery problem into the network flow model. Different from the former researches, we will introduce virtual source node to turn repeatedly recovery problem into simple multi independent nodes one-loss network model.
Chapter four: we will propose our scheme and break it down in detail. In order to solve some problems in conventional storage system, we will demonstrate our algorithm which combines Replica, MDS codes, network coding and network flow diagram. By the generation example, we will show a more general generation construction in common situation.
Chapter five: we will analyse our scheme. We will do the exactly simulation by NECO 2.0, a special simulator designed to emulate the network coding environment. From the result, we will summary our scheme having a good reliability and efficient performance.
Chapter six: we will make a conclusion of this master research. After summary up the advantages and drawbacks of our system, we give a brief view about our future works.

2. NETWORK CODING

2. Network coding

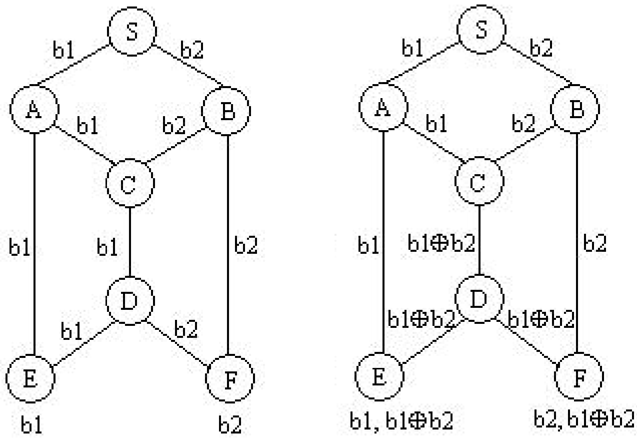
After the brief introduction of above contents, we can get a simple recognition that network coding does help a lot for WSN which can save energy, enhance throughput, increase robustness of network, balance network load and help the security. We concern with whether the theorem is sufficient and necessary condition for WSN. The fascinating fact is that the main network coding theorem is suitable for WSN. We start this chapter with the explanation of the network coding theory. By the introduction of some important definition and notification, we elaborate the conception by algebraic approach, mathematical formulas and graph theory. Followed by math part, we give some applications for network coding in WSN to understand network coding more directly and easily.

2.1. Theory of Network Coding

Using "butterfly network" as an example as shown in Fig.5 (a), consider the source S multicast two unit data, b1 and b2, to information sinks E and F. Suppose the channel capacity of each single data link between two nodes (such as link SA, AC, CD, DE and so on) in this network is one unit data. So in traditional routing method, as shown in Fig.5 (a), E can only receive data b1 in one unit time because only one unit data can be transmitted per unit time over link CD, which therefore cannot achieve the maximum transmission capacity of multicast.

However, if we introduce the NC into the traditional routing method, as shown in Fig.5 (b), node W can code the incoming data b1 and b2 using XOR operation and then transmit the coded data $b1 \oplus b2$ over link CD, so as to make E and F both receive $b1 \oplus b2$ at the same time [6]. Finally, by using corresponding decode method (For E: $b2 = b1 \oplus (b1 \oplus b2)$; For F: $b1 = b2 \oplus (b1 \oplus b2)$), E and F can obtain b1 and b2 in one unit time, which then achieve the maximum multicast transmission capacity [5, 6].

2. NETWORK CODING



(a) Conventional routing (b) NC based routing

Figure 5.Butterfly network with one source and two sinks

All the proof can be found in [4], in this epochal paper, we can easily find the worth of network coding. By max-flow min-cut theorem, networks should be able to achieve the max-flow bound on the information transmission rate in a multicast scenario in theoretical. However, in the practical world, it is impossible to achieve that by the conventional forward routing. It is in general not optimal to consider information to be multicast in a network as a fluid which can be routed or replicated at the intermediate nodes. On the contrary, network coding had to be employed to achieve optimality which makes a great progress.

2.1.1. Linear Network Coding

In [4], a state-of-the-art technology has been demonstrated, the author showed us some conjecture, and then concentrate on the one source communication. They left multisource network information flow problems as the future work. In [5], the author solved the problem that a communication network in which certain source nodes multicast information to other nodes on the network in the multi-hop fashion where every node can pass on any of its received data to others. We will explain this important algorithm below.

Linear code multicast, LCM in short, is the processing that in particular, all the nodes except source and sink nodes in the network, do the linear operation such as addition and multiplication for all the received packets, when receiving enough independent packets, nodes can decoding the original information.

2. NETWORK CODING

In a LCM problem, a group of nodes P are involved in moving the data from S source nodes to K sink nodes. Each node generates totally new packets which are linear combinations of earlier received packets, multiplying them by the coefficients chosen from a finite field, typically of size $GF(2^s)$, which is Galois Field at most common situation. Each node, P_k with in degree, $InDeg(\mathrm{p}_k)=\mathrm{S}$, generates a message X_k from the linear combination of received messages $\{M_i\}_{i=1}^{S}$ by the relation: $X_k = \sum_{i=1}^{S} g_k^i \cdot M_i$,where the values g_k^i are the coefficients selected from $GF(2^s)$. Note that, since operations are computed in a finite field, the generated message is of the same length as the original messages. Each node forwards the computed value X_k along with the coefficients, g_k^i, used in the k^{th} level, g_k^i.

Sink nodes receive these network coded messages, and collect them in a matrix. The original messages can be recovered by performing Gaussian elimination on the matrix. In reduced row echelon form, decoded packets correspond to the rows of the form $e_i = [00...010...00]$.

As the evolution of conventional network coding, LCM is sufficient for achieving asymptotic optimality for single source network coding. It is not clear whether this continues to hold for multi-source network coding. In this section, we present a discussion and explore a potential gap between the asymptotic performance of linear codes and nonlinear codes.

We model the communication network by a directed graph $G(\mathrm{V,E})$, where G the set of nodes and E the set of edges in G .The information between network nodes transmitted in packets. We assume that each packet is an element of some finite field $F_q = GF(q)$ and can be represented by a binary vector of length $n = \log_2(q)$ bits. The communication is performed in rounds, so that at each round, every edge in the network can transmit a single packet. The definitions are below:

(1) $v(S) \in \Omega$, there symbol Ω denotes a fixed-dimensional vector space over a sufficiently large base field.

2. NETWORK CODING

(2) For each channel XY , $v(XY) \in v(X)$

(3) For any collection of non-source nodes in the network, $\langle \{v(T): T \in \wp\} \rangle = \langle \{v(XY): X \notin \wp, Y \in \wp\} \rangle$

(4) The vector assigned to an outgoing channel from T must be a linear combination of the vectors assigned to the incoming channels of T .

Let $h(i)$ be the maximum number of packets that can be delivered from s to all terminals in all destination nodes T in i rounds. So that the capacity h^* of the network is defined as $h^* = \lim_{i \to \infty} \sup \frac{h(i)}{i}$. At the same time, let $\vec{b} = [b_1, b_2, b_3, \ldots b_k]$ be the set send by source nodes, so the matrix $\vec{\beta} = \left[\beta_1^1, \beta_2^1, \beta_3^1 \ldots \beta_{h-1}^1, \beta_h^1\right]^T$ means the information set vector send on the first hoop in the topology. For the set for all global encoding vectors, we define a $h \times h$ matrix as follow:
$\vec{m} = \begin{pmatrix} m_1^1 & \cdots & m_h^1 \\ \vdots & \ddots & \vdots \\ m_1^h & \cdots & m_h^h \end{pmatrix}$

For example, the butterfly network shown in Fig.5, if the coding coefficients in node A, B, C are sent as $(m_1^1, m_1^2), (m_2^1, m_2^2), (m_3^1, m_3^2)$ separately while the packets received by node A, B,C are notified as y_1, y_2, y_3.

In node E, the received packet can be represented as:
$\begin{pmatrix} \beta_1 \\ \beta_3 \end{pmatrix} = \begin{pmatrix} m_1^1 & m_1^2 \\ m_1^1 m_3^1 + m_2^1 m_3^2 & m_1^2 m_3^1 + m_2^1 m_3^2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$

Similarly, in node F, the received packet is: $\begin{pmatrix} \beta_2 \\ \beta_3 \end{pmatrix} = \begin{pmatrix} m_2^1 & m_2^2 \\ m_1^1 m_3^1 + m_2^1 m_3^2 & m_1^2 m_3^1 + m_2^1 m_3^2 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$

For the common network, the symbol vector is $\vec{\beta} = M_j^i \cdot \vec{b}^T$

When M_j^i is full of rank, at the j-th path after received enough packets

$\vec{\beta} = \left[\beta_1^j, \beta_2^j, \beta_3^j \ldots \beta_{h-1}^j, \beta_h^j\right]^T$ the sink node t_j can decode the original message $\vec{b} = [b_1, b_2, b_3, \ldots b_k]$.

It has been proven for a long time that linear coding is enough to achieve the upper bound in multicast problems. However, LCM is not sufficient in general (e.g. multisource, multi-sink with arbitrary demands), even for more general versions of linearity such as convolutional coding. Besides, it is centralized and only knowing all the network topology, we can decode the information. But in WSN, it is difficult to achieve because with the limited energy and computation ability, it is almost impossible for every node to know the whole network topology and coding coefficient [9].

2.1.2. Random Linear Network Coding

In [10], the authors presented a new coding algorithm named random network coding which is based on LCM. In random network coding for multicast, (content distribution, distributed storage, broadcast and so on) the information packets are linearly combined with randomly chosen coefficients from a field. The approach will achieve the capacity region of the network with high probability. We should note that firstly, in random network coding the choice of code and operation of each node is chosen completely decentralized and does not require any centralized management. Secondly, the combined effect of the random choices is communicated by pilot tones at a controllable overhead. But if we can combine network coding with an efficient resource allocation scheme that only forwards information on properly chosen links with proper rates we can avoid the flooding problem.

Moreover, random network coding has many advantages in practical settings. In particular, it allows each node in the network to choose a suitable encoding coefficient in a decentralized manner without prior coordination with other nodes in the network. Random coding has been used in several practical implementation schemes [11]. Now, we will introduce random network coding (RNC) below.

First of all, let us reconsider the most important parameters of a network coding scheme, which is the minimum required size of a finite field. The field size determines the number of available linear combinations. The number of such combinations and the required field size are determined by the combinatorial structure of the underlying communication network. For example, in the network shown in Figure.5, $GF(2)$ big enough for achieving the network capacity. However, the practical network topology is much bigger than the butterfly network using in the theoretical analysis. There is another example shown in Figure.6. Let $\Gamma_{e1},\Gamma_{e2},\Gamma_{e3},\Gamma_{e4}$ be the global encoding vectors of edges e_1,e_2,e_3,e_4. We can notice that in this network, each pair of (v_i,v_j) of the intermediate nodes is connected to a terminal. Hence any two of the global encoding vectors $\Gamma_{e1},\Gamma_{e2},\Gamma_{e3},\Gamma_{e4}$ must be linearly independent. Similarly that with $GF(2)$ there exist only three non-zero pairwise linearly independent vectors of size two: $\begin{pmatrix}1\\0\end{pmatrix},\begin{pmatrix}0\\1\end{pmatrix}$ and $\begin{pmatrix}1\\1\end{pmatrix}$, so it is obvious that $F_2=GF(2)$ is insufficient for achieving network capacity. However, it is possible to find a network coding solution over $F_3=GF(3)$ or a larger field. For example, by using $F_3=GF(3)$ the following global encoding coefficients are feasible: $\begin{pmatrix}1\\0\end{pmatrix},\begin{pmatrix}0\\1\end{pmatrix},\begin{pmatrix}1\\1\end{pmatrix},\begin{pmatrix}1\\2\end{pmatrix}$.

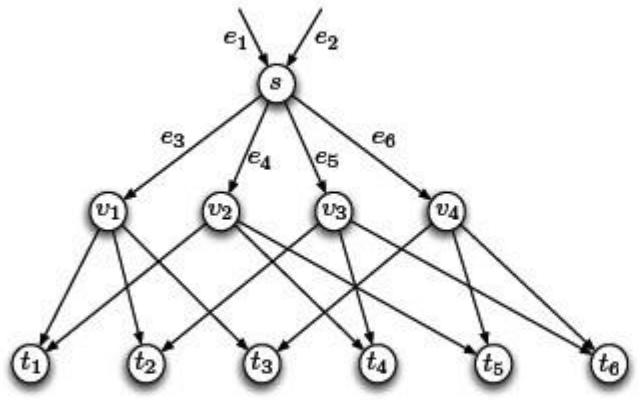


Figure.6.A complicated coding network

So we know that it is important to choose an appropriate parameter $F_q=GF(\mathrm{q})$ to make suitable to the network. However, proofing the theorem is not the emphasis of our research. In [13], shown in Table 1, the probability of linear independency is over 99.6% for $q = 2^8$, and this is almost independent of q. So in our research, we also set the size of finite field as $F_8 = GF(8)$.

q	Probability	q	Probability	q	Probability
2^1	0.288788	2^5	0.967773	2^9	0.998043
2^2	0.688538	2^6	0.984131	2^{10}	0.999022
2^3	0.859406	2^7	0.992126	2^{11}	0.999511
2^4	0.933595	2^8	0.996078	2^{12}	0.999756

Table.1. Probability of Linear Independency as a Function of Finite Field Size (q).

After understood of finite field size, we can comprehend random network coding (RNC) more comfortable. RNC pick up the parameters randomly and independently from finite field size for all the intermediate nodes except sink nodes. As long as $F_q = GF(\mathrm{q})$ big enough, it will be a higher probability to get full rank of $\vec{m} = \begin{pmatrix} m_1^1 & \cdots & m_h^1 \\ \vdots & \ddots & \vdots \\ m_1^h & \cdots & m_h^h \end{pmatrix}$ in order to decode the packets successfully.

The probability of success decoding is $P_{success} \geq \left(1\text{-}\frac{d}{q}\right)^v$ where d means the number of sink nodes, v means the number of edges with independent randomized coefficients, q means Galois field size here. The first two coefficients d and v are the smaller, the better. Conversely, the bigger q is, the better.

2.2. Some Applications for Network Coding over WSN

In this section, we would introduce some applications for network coding over WSN. From three different fields, three interesting and distinctive parts of network coding application can be noted. They can give a more intuitive view for the advantage of NC in the practical world.

2.2.1. Opportunistic Routing& Optimal Coding

In [14], the author proposes COPE, a new architecture for wireless networks. Their work aims to bridge theory with practice; it addresses the common case of unicast traffic, dynamic and potentially flows, and practical issues facing the integration of network coding in the current network stack. By three main techniques, opportunistic listening, opportunistic coding and learning neighbor state, COPE inserts a coding layer between the IP and MAC layers, which detects coding opportunities and exploits them to forward multiple packets in a single transmission.

COPE sets the nodes in promiscuous mode, makes them snoop on all communications over the wireless medium and store the overheard packets for a limited period. In addition, each node broadcasts reception reports to tell its neighbors which packets it has stored. Reception reports are sent by annotating the data packets the node transmits. A node that has no data packets to transmit periodically sends the reception reports in special control packets.

It is best illustrated with an example. In Fig. 7(a), node B has 4 packets in its output queue p_1, p_2, p_3, and p_4. Its neighbors have overheard some of these packets. The table in Fig 7(b) shows the next hop of each packet in B's queue. When the MAC permits B to transmit, B takes packet p1 from the head of the queue. Assuming that B knows which packets each neighbor has, it has a few coding options as shown in Fig. 7(c). It could send $p_1 \oplus p_2$. Since node C has p1 in store, it could XOR p_1 with $p_1 \oplus p_2$ to obtain the native packet sent to it, for instance p_2. However, node A does not have p_2, and so cannot decode the XOR-ed packet. Thus, sending $p_1 \oplus p_2$ would be a bad coding decision for B, because only one neighbor can benefit from this transmission. The second option in Fig. 7(c) shows a better coding decision for B. Sending $p_1 \oplus p_3$ would allow both neighbors C and A to decode and obtain their intended packets from a single transmission. Yet the best coding decision for B would be to send $p_1 \oplus p_3 \oplus p_4$, which would allow all three neighbors to receive their respective packets all at once.

Each node announces to its neighbors the packets it stores in reception reports. However, at times of severe congestion, reception reports may get lost in collisions, while at times of light traffic, they may arrive too late, after the node has already made a suboptimal coding decision. In the absence of deterministic information, COPE estimates the probability that a particular neighbor has a packet as the delivery probability of the link between the packet's previous hop and the neighbor. Occasionally, a node may make an incorrect guess, which causes the coded packet to be un-decodable at some next hop. In this case, the relevant native packet is retransmitted, potentially encoded with a new set of native packets.

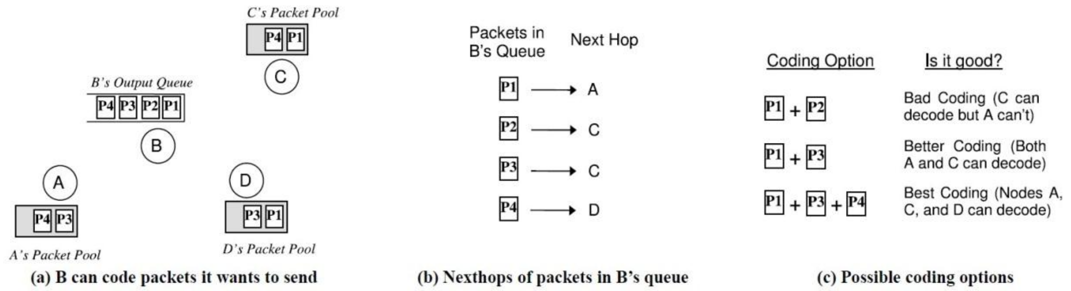


Figure 7.example of Opportunistic Coding

With a few fundamental approaches, COPE can improve wireless throughput, including more accurate congestion control, better routing, and efficient MAC protocols. COPE is an important step forward in our understanding of the potential of wireless networks because it presents a new orthogonal axis that can be manipulated to extract more through put.

2.2.2. Efficient Error Recovery Underwater

In [15], the author used network coding to solve error recovery in the underwater environment. Using common error-recovery techniques such as Automatic Repeat re-Quest (ARQ) and Forward Error Correction (FEC) in underwater sensor networks has the following drawbacks. ARQ-based schemes require the receiver to detect losses and then request the sender to retransmit packets. This may lead to long delays. FEC-based schemes proactively add redundant packets to eliminate retransmission from the source. The FEC can be applied on an end-to-end or hop-by-hop basis (as in [16]). However, in either case, the proper amount of redundancy is hard to decide due to the difficulty of obtaining accurate error-rate estimates.

Why network coding is suitable for underwater sensor networks because (1) underwater sensor nodes are usually larger than land-based sensors and they posse more computational capabilities [15]; (2) the broadcast property of acoustic channels naturally renders multiple highly interleaved routes from a source to a sink. The computational power at the sensor nodes coupled with the multiple routes provides ample opportunity to apply network coding.

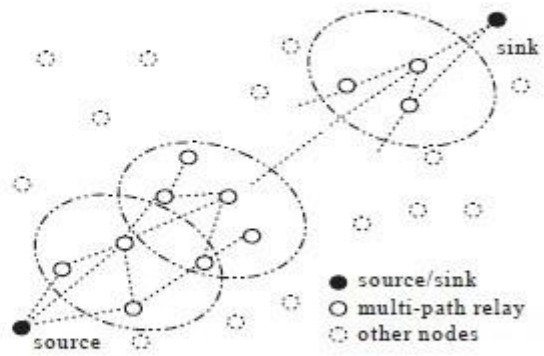


Fig.8. Illustration of transmitting a packet along multiple paths from the source to the sink.

At source nod: Packets from the source are divided into generations, each generation contains K packets. The source linearly combines K packets in a generation using randomly generated coefficients. More specifically, let $X_1, X_2, \ldots X_k$ denote the K packets in a generation. The source linearly combines these K packets to compute $K^{'}$ outgoing packets, denoted as $Y_1, Y_2, \ldots, Y_k$ where $Y_i = \sum_{j=1}^{K} g_{ij} X_j$. The coefficient g_{ij} is picked randomly from a finite field $F_q = GF\left(2^q\right)$. The set of coefficients $\left(g_{i1}, g_{i2}, \ldots g_{ik}\right)$ is referred as the encoding vector for Yi [15] and are carried in a packet as overhead. We choose $K^{'} \geq K$ since adding a small amount of redundancy at the source (e.g., $K^{'} = K + 2$) reduces the impact of packet loss on the first hop (which cannot be recovered at later hops) and improves error recovery at the sink.

At relay nod: In forwarding paths relay nod stores incoming packets from different routes in a local buffer for a certain period of time, linearly combines the buffered packets which belong to the same generation. Suppose a relay, γ, receives M incoming packets, $X_1^{\gamma}, X_2^{\gamma}, \ldots X_M^{\gamma}$. If we use $\left(f_{i1}, f_{i2}, \ldots f_{ik}\right)$ to denote the encoding vector carried by X_i^{γ} .Because transmitting dependent packets is not useful for decoding at the sink, relay γ computes $M^{'}$ outgoing packets, where $M^{'}$ is the rank of the coefficient matrix $\left(f_{i1},f_{i2},...f_{ik}\right)$. Moreover, since $M^{'} \leq \min\left(M,K\right)$.Let $Y_1^{\gamma},Y_2^{\gamma},...Y_{M^{'}}^{\gamma}$ denote the outgoing packets, so $Y_i^{\gamma}=\sum_{j=1}^{M} h_{ij}^{\gamma} X_j^{\gamma}$, where h_{ij} is chosen from $F_q=GF\left(2^q\right)$, too. Let $\left(g_{i1}^{\gamma},g_{i2}^{\gamma},...g_{iK}^{\gamma}\right)$ denote the encoding vector of Y_i^{γ}.

At sink nod: After receiving K packets with linearly independent encoding vectors, it recovers the original packets by matrix inversion. The complexity is $O\left(K^3\right)$.

After analytically studied the performance of this scheme along with several other error-recovery schemes, the authors provided guidance on how to choose parameters in our scheme and demonstrated their scheme is the most efficient among the multiple schemes in both error recovery and energy consumption.

2.3. Summary

In this chapter, we have described the network coding theory step by step. Started with brief introduction, followed by linear network coding (LNC) and random network coding (RNC), we demonstrate the principle and working mechanism of network coding. Then by two applications using practical network coding in WSN environment, first one-opportunistic routing& optimal coding, second one- efficient error recovery underwater, we show how network coding performances over different fields in practical world.

3. Network Model for WSDSN

In chapter two, we theoretically introduced network coding by linear network coding multicast (LCM) and random network coding (RNC). By following the practical examples of opportunistic routing& optimal coding and efficient error recovery underwater, it is obvious that network coding can be applied over different fields in real world. In so many research fields, we focus on how network coding performs in distributed storage systems. In WSN, distributed storage systems provide reliable access to data through redundancy spread over individually unreliable nodes. Many researches have been done to solve the problem. In this chapter, we will start the analysis with a mathematical modelling the network flow graph.

3.1. Existing Problems in Distributed Storage System

In recent years, the demand for large scale data storage has increased significantly and rapidly, with a lot of applications like social networks, file, and video sharing demanding seamless storage, access and security for massive amounts of data [18]. If in modern data centers and peer-to-peer networks, when the deployed storage nodes are individually unreliable, redundancy must be introduced into the system to improve reliability against node failures. However, in WSN, there are lots of restricts existed such as computational capacity, traffic, storage space cost, transmission delay and so on. All these limits make the problem in WSN more complicated than in common situation. There is a typical motivation desired situation shown in Figure.9. Smart phone users want to store their files while keeping security.

3. NETWORK MODEL FOR WSDSN

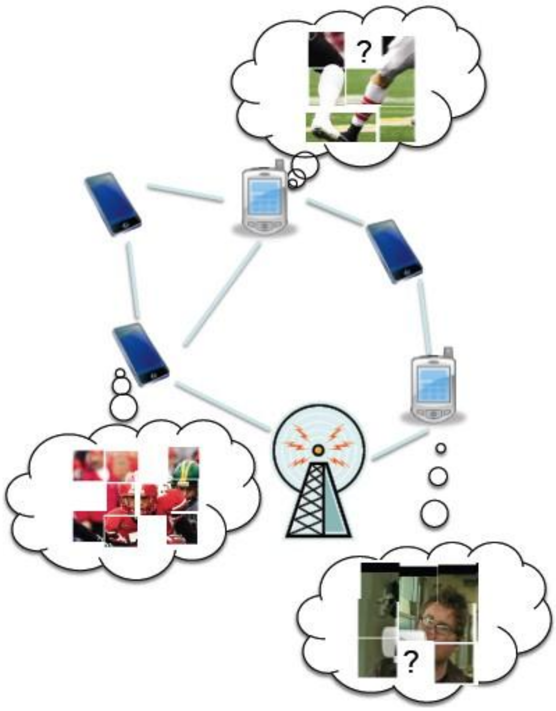


Figure.9 a typical motivation desired in data storage for WSN

In detail, typical sensor nodes are wireless nodes with limited storage and computational power. Furthermore, they are prone to "failure", by going out of wireless range, interference, running out of battery etc. When a sensor node fails, data itself was storing is lost. In a cooperative sensor network, it is a good idea to have nodes' data duplicated and spread around the network so it can be recovered from other nodes in case of failure, but not for WSN. If part of the sensor network fails, the data stored in the failed sensor nodes is lost. The only data which remain is the data remaining at the surviving nodes. The surviving nodes have symbols encoded from the data produced by all the sensor nodes, so there is still a chance of recovering data produced by failed nodes using the surviving encoding symbols.

When in a distributed storage system with mass of nodes, we choose n nodes to store a file of size M. Design a fault recovery scheme based on network coding, use this scheme to encode the file into n pieces, transmit them evenly to n chosen nodes, which means every nodes have one piece. The basic goal of our scheme is guarantee any k nodes ($k<n$) can decode the original file, each decoding packet has a storage size of $\frac{M}{k}$. Not like other models, in our

system, after running a period of time, not one node, but r nodes become malfunction. At this time, in our network model, the system chose some more r nodes, utilize them (we call them new nodes) getting data pieces from n-r alive nodes (we call them old nodes) to recover the lost part. And after the recovery, the whole system still remains n nodes as many as the very beginning. The whole process is shown in Fig.10.

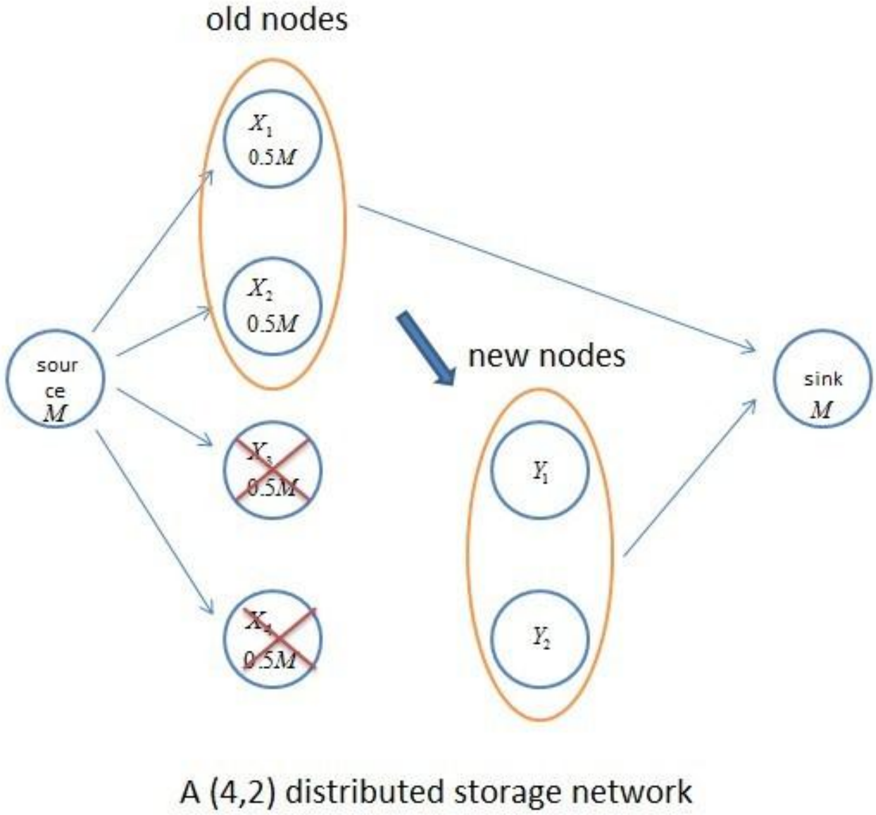


Fig.10 A simple example of (4, 2) coding distributed network system

According to above, we make the definition for our wireless sensor distributed storage network (WSDSN) model.

Definition 3.1 Assume a file of size M being upload to n chosen nodes while maintain the characteristic of (n, k) coding. When r nodes become malfunction, system will chose other r nodes to recover and maintain the coding function. Finally, sink node can pick up any k nodes ($k<n$) to decode the original file. In detail, we break the process in WSDSN into 3 stages: data upload, data recovery, data download.

3. NETWORK MODEL FOR WSDSN

(1) Data upload

At the very beginning, the source node store a file of size M, then encodes the file into n pieces, transmit them evenly to n chosen nodes $X_1, X_2, \dots X_{n\text{-}1}, X_n$. Any k nodes ($k<n$) can decode the original file, each decoding packet has a storage size of $\frac{M}{k}$.

(2) Data recovery

When r nodes become malfunction, in order to maintain the reliability of network while keep the whole redundancy, system chose some more r nodes $Y_1, Y_2, \dots Y_{\text{r-1}}, Y_r$, utilize them (we call them new nodes) getting data pieces from $n\text{-}r$ alive nodes $X_1, X_2, \dots X_{n\text{-r-1}}, X_{n\text{-}r}$ (we call them old nodes) to recover the lost part. More specific is shown below:

Step one: Every old node $X_i \left(1 \le i \le n\text{-}r\right)$ can connect to other available nodes $X_1, X_2, \dots X_{i\text{-}1,} X_{i+1}, \dots X_{n\text{-r-1}}, X_{n\text{-}r}$ in order to download the data piece $\beta_{1,i}^{oo}, \dots, \beta_{\text{n-r},i}^{oo}$, $\beta_{i',i}^{oo}$ is the data traffic from old node $X_{i'}$ to old node X_i. If node $X_{i'}$ do not link to node X_i, $\beta_{i',i}^{oo}=0$

Step two: Every new node $Y_j \left(1 \le j \le r\right)$ can connect to other rest available old nodes $X_1, X_2, \dots, X_{n\text{-r-1}}, X_{n\text{-}r}$ in older to download the data piece $\beta_{1,\text{j}}^{on}, \dots, \beta_{\text{n-r,j}}^{on}$, $\beta_{i',\text{j}}^{on}$ is the data traffic from old node $X_{i'}$ to new node Y_i. If node $X_{i'}$ do not link to node Y_j, $\beta_{i,\text{j}}^{on}=0$. At the same time, every new node $Y_j \left(1 \le j \le r\right)$ can also connect to other rest new available nodes $Y_1, Y_2, \dots Y_{i\text{-}1,} Y_{i+1}, \dots Y_{n\text{-r-1}}, Y_{n\text{-}r}$ to download the data piece $\beta_{1,\text{j}}^{nn}, \dots, \beta_{j\text{-1,j}}^{nn}, \beta_{j\text{+1,j}}^{nn}, \dots \beta_{\text{n-r,j}}^{nn}$. $\beta_{j',\text{j}}^{nn}$ is the data traffic from new node $Y_{j'}$ to new node Y_j. If node $Y_{j'}$ do not link to node Y_j, $\beta_{\text{j}',\text{j}}^{nn}=0$.

Step three: Every new node $Y_j \left(1 \le j \le r\right)$ creates a new coded packet of size $\frac{M}{k}$ by received data pieces $\beta_{1,\text{j}}^{on}, \dots, \beta_{\text{n-r,j}}^{on}$ and $\beta_{1,\text{j}}^{nn}, \dots, \beta_{j\text{-1,j}}^{nn}, \beta_{j\text{+1,j}}^{nn}, \dots \beta_{\text{n-r,j}}^{nn}$. Notice, every new node can only receive

3. NETWORK MODEL FOR WSDSN

parts of data from $\beta^{on}_{1,j},...,\beta^{on}_{n-r,j}$ and $\beta^{nn}_{1,j},...,\beta^{nn}_{j-1,j},\beta^{nn}_{j+1,j},...\beta^{nn}_{n-r,j}$.

(3) Data download

After the data recovery, the system stores the data by n-r old nodes $X_1,X_2,...X_{n\text{-}r\text{-}1},X_{n\text{-}r}$ and r new nodes $Y_1,Y_2,...Y_r$. Any sink node can decode the original data from any k nodes ($k<n$).

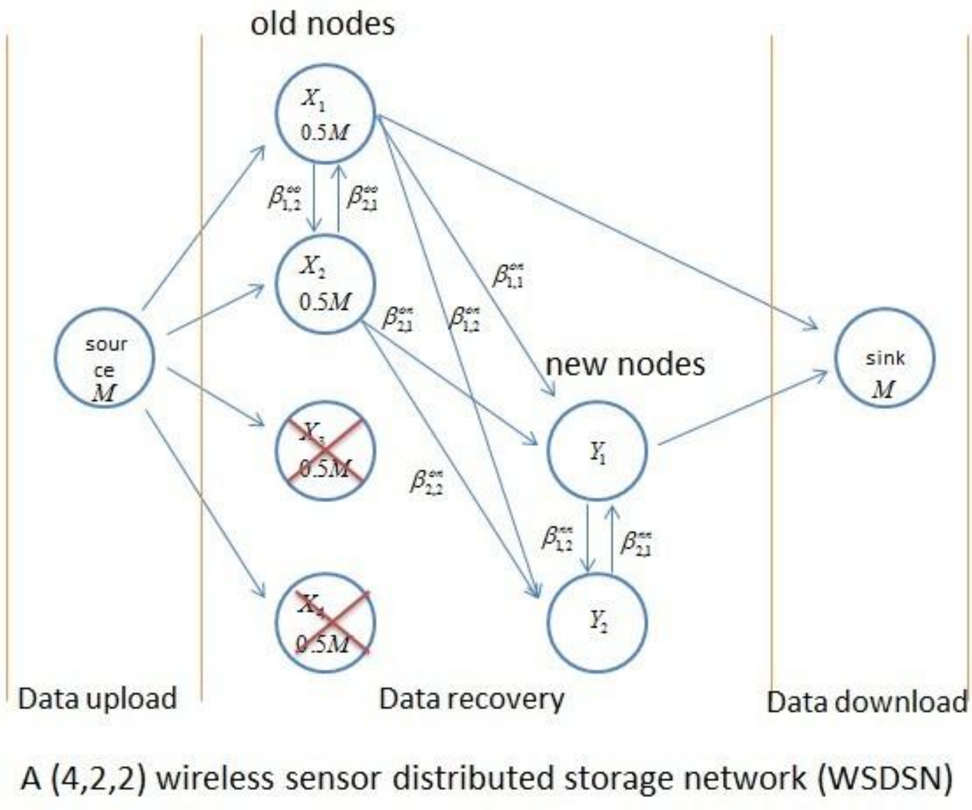


Fig.11 Three stages in WSDSN

We define the repair traffic as the amount of outbound data being read from other surviving clouds during the single-cloud failure recovery. Our goal is to minimize the repair traffic for cost-effective repair.

3. NETWORK MODEL FOR WSDSN

The whole traffic during three stages: $\gamma = \sum_{i'=1 and i' \neq i}^{n-r} \sum_{i=1}^{n-r} \beta_{i',i}^{oo} + \sum_{i=1}^{n-r} \sum_{j=1}^{r} \beta_{i,j}^{on} + \sum_{j'=1 and j' \neq j}^{r} \sum_{j=1}^{r} \beta_{j',j}^{nn}$

The traffic is a big index to judge a scheme whether performs good or not in any distributed storage system. In order to analyze the problem simple with network coding, we should extend WSDSN into network flow graph. Later, we will introduce some other networks in section.2, give our model in section.3.

3.2. Conventional Models

3.2.1. One-node Loss Recovery Network Model

In [19], the authors set their network environment on DSN ($n,k,1$) model, which means in their model, assuming only one node become malfunction at one time unit. They make the network model into information flow graph in Fig.12.

The authors defined it as a directed acyclic graph consisting of three kinds of nodes: a single data source S , storage nodes X_{in}^{i}, X_{out}^{i} and data collectors DC_i .

(1) The single node S corresponds to the source of the original data.

(2) Storage nodes X_{in}^{i}, X_{out}^{i} are represented by a storage input node X_{in}^{i} , and a storage output node X_{out}^{i} . These two nodes are connected by a directed edge $X_{in}^{i} \rightarrow X_{out}^{i}$ with capacity equal to the amount of data stored at node i .

(3) Data collectors DC_i have $\binom{n}{k}$ nodes. The system can recover the original data from any k nodes in n nodes. According to one scheme, there is always a received node.

The authors also defined the system consisting of three edges of nodes:

(1) The edge from source S to initial set of storage nodes is the first type. The bound in this type is infinite ∞ .

(2) The edge from the old node X_i to the new node X_j . The bound in this type is α .

(3) The edge from data collectors DC_i to the data piece provider node. The bound in this type is also infinite ∞.

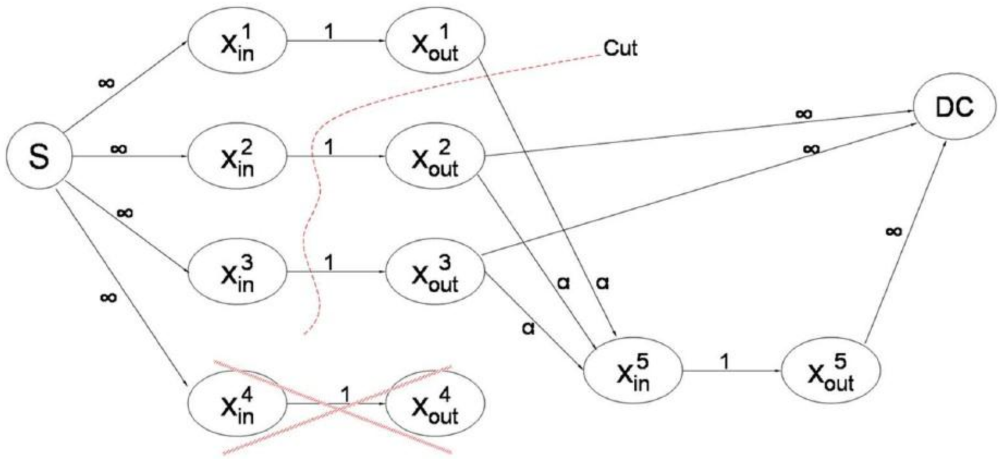


Fig.12 a DSN (4, 3, 1) information flow graph model

Obviously, in [19], the authors focus on the one-node loss recovery network model. In their model, they did not concern on the complicated situation that multi-nodes loss which is very common on the practical world. However, it is still a good model since it firstly gave a clear definition model on DSN.

3.2.2. Potentially Infinite Information Flow Model

In [20], the authors extended their model based on [19]. They supposed a repeatedly one-node loss recovery network model in Fig.13. In their model, every storage node gets α information, once if need, every new nodes will get data pieces, download β bits from d surviving nodes ($d < n$). Here, they used "immediate repeat" scheme, which means when a node is lost, a new node will fill in the recovery system immediately. They further restricted their attention to the symmetric setup where it is required that any *k* storage nodes can recover the original file and the traffic in every edge is the same so that it was a symmetric recovery system. Since the recovery process is potentially infinite, so the model is a potentially infinite information flow network model. They denote this family of directed acyclic graphs by $G(n,k,d,\alpha,\beta)$.

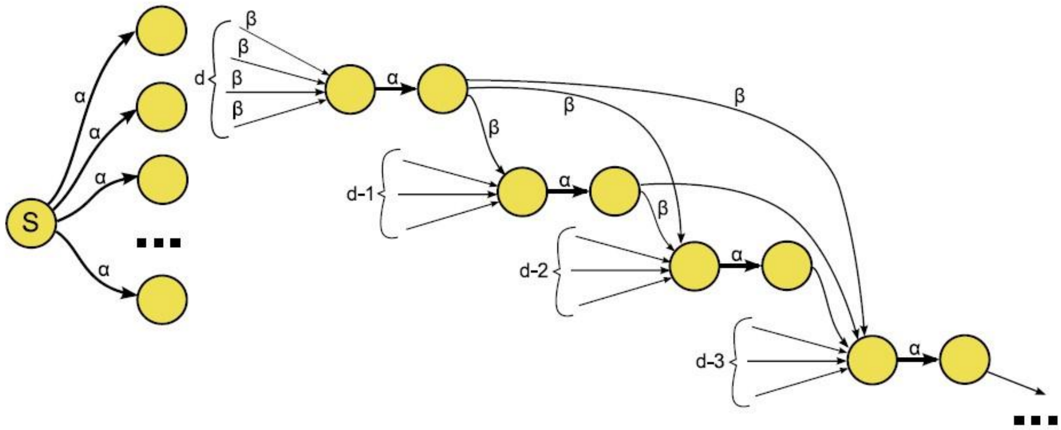


Fig. 13 a DSN (n, k, 1) repeatedly one-node loss recovery network model

In their assumption, their work is based on repeatedly one-node loss recovery, which is a special situation for multi-nodes-loss. They did not put their concern on more common circumstances. Moreover, it is not suitable for WSN since the nodes cannot link to each other all the time. Sometimes some nodes would hard to access or become malfunction.

3.3. Proposed Network Model

3.3.1. Ideal Goal for WSDSN

If in modern data centers and peer-to-peer networks, when the deployed storage nodes are individually unreliable, redundancy must be introduced into the system to improve reliability against node failures. However, in WSN, there are lots of restrictions existed such as computational capacity, traffic, storage space cost, transmission delay and so on. All these limits make the problem in WSN more complicated than in common situation. The former two models are obviously not suitable for WSDSN since these restrictions. Moreover, those restrictions limit the developing of the performances. We should balance all these parameters to provide an optimal recovery performance for WSDSN.

3.3.2. Virtual Source Node

Before explaining our network model, we want to introduce virtual source node (VSN) to simply the recovery problem.

Definition 3.2 When WSDSN finish recovery once, we assume there is a virtual source node (VSN) to connect all the repaired nodes.

We will explain this idea with Fig.14. The recovery process is on the left. The source nodes X_1, X_2, X_3, X_4 store a file of size M, when X_3, X_4 are lost, the system chose Y_1, Y_2 to reconstruct the lost parts. After the data recovery, two old nodes X_1, X_2 and two new nodes Y_1, Y_2 keep the original data M .We can notice that the after-recovery system just like a virtual source node upload four data pieces to these four nodes.

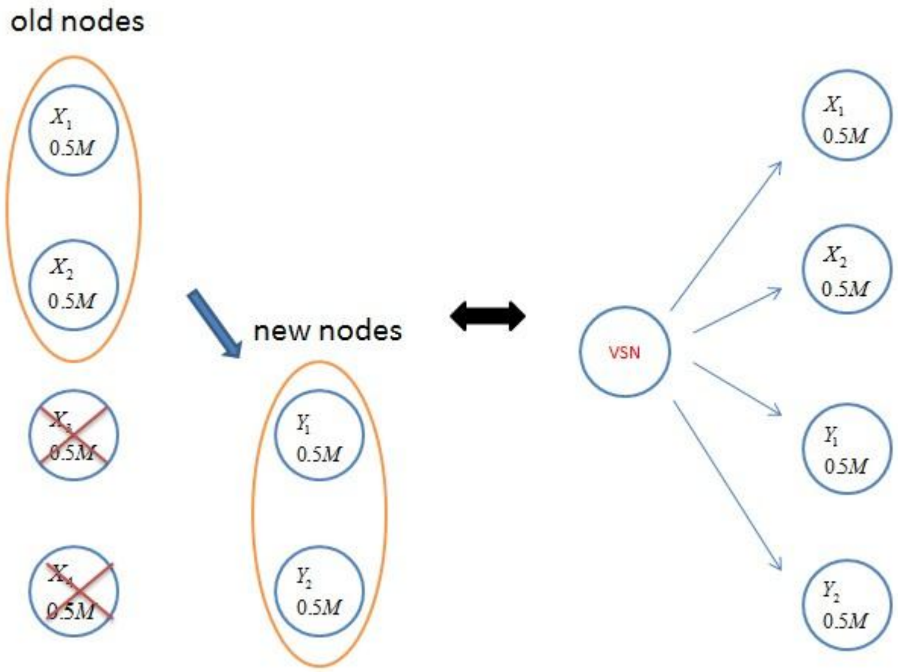


Fig.14 WSDSN (4, 2, 2) after-recovery situation and its corresponding storage status

Virtual sink nodes connect to any k node subsets and ensure that the code has the MDS property that any k out of n suffices to recover. If we use the model in [19], the unit-capacity edges sum from input to output for any k out of n should be larger than the file of size M , otherwise the max-flow in DSN (n,k,r) would be disable to reconstruct the original data. So no matter the after-recovery situation or the corresponding storage status, both of them should satisfy the unit-capacity edges sum from input to output for any k out of n should be larger than the file of size M . We can get the conclusion that the virtual source node can simplify the model.

Now we will describe how the virtual source nodes predigest the model by the multi-nodes one-loss recovery network model. Assuming two multi-recoveries happens: 1. DSN (4, 2, 1) have four storage nodes X_1,X_2,X_3,X_4 , when X_4 loss, X_1,X_2,X_3 send data pieces to reconstruct Y_4. 2. DSN (4, 2, 1) now have four storage nodes X_1,X_2,X_3,Y_4, when X_3 loss, X_1,X_2,Y_1 send data pieces to reconstruct Y_2 .The whole process is in Fig.15 and Fig.16 shows the network model if we import the virtual source node. By observation of these two graphs, the recovery edges (red edges) are the same. Now we can analyze the model layer by layer, each layer can be consider as a network flow graph by a VSN. Then, a multi-recovery model will be transfer into plenty of simple multi independent nodes one-loss network model.

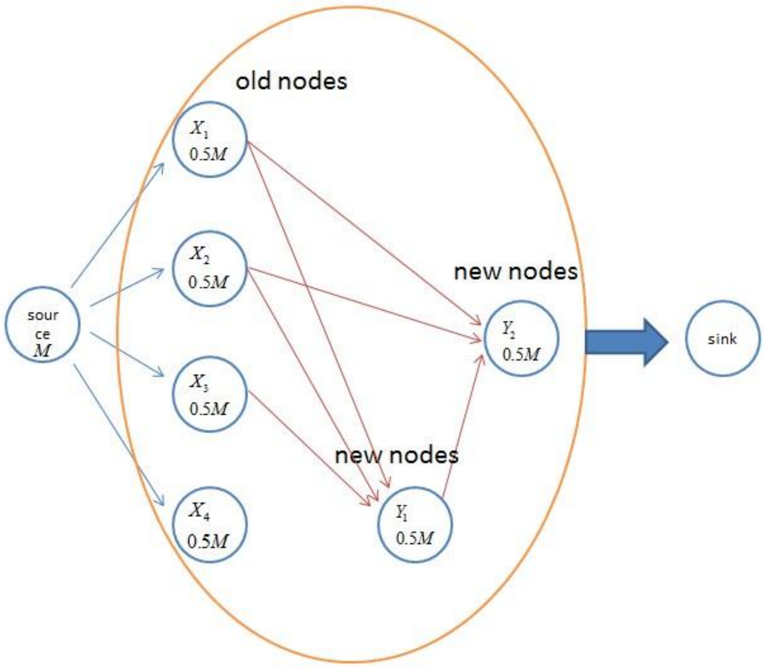
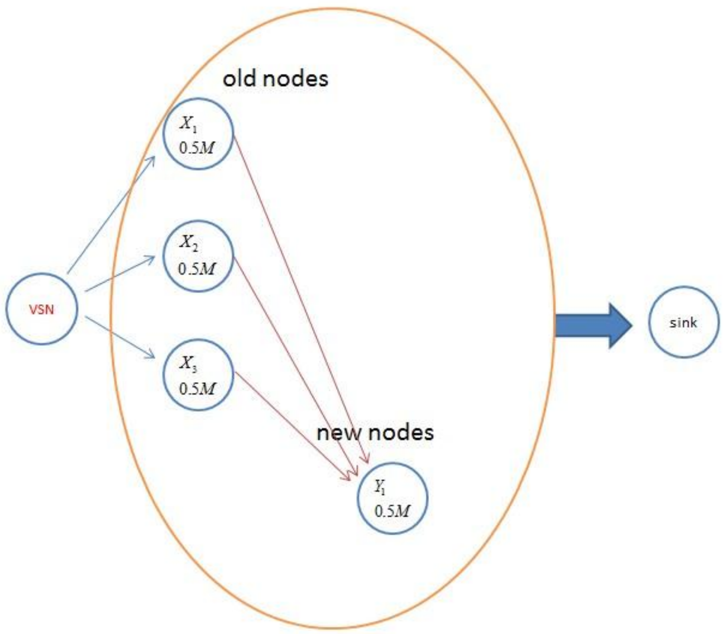


Fig. 15 WSDSN (4, 2, 2) two-nodes one-loss repeatedly recovery



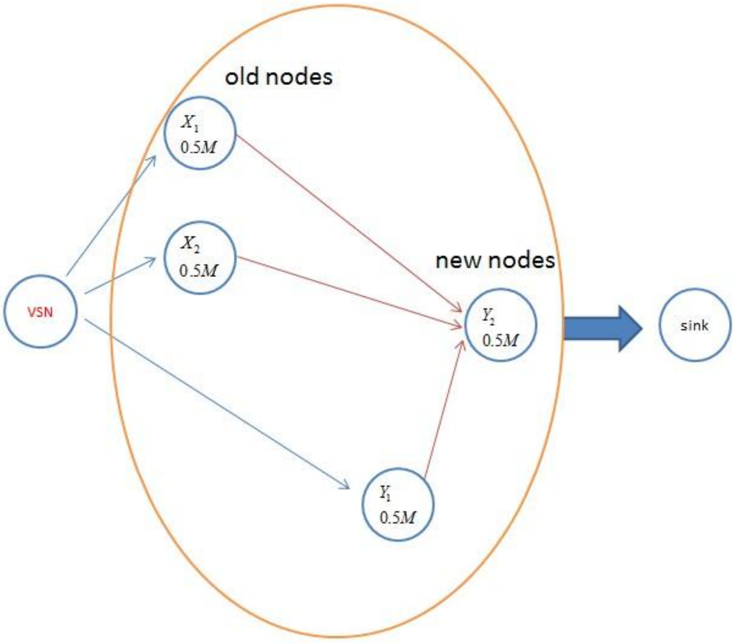


Fig. 16 (a) the first recovery, (b) the second recovery

3.3.3. Acyclic Network with Three Types of Nodes

In the former researches, they only assumed the transmission existed between old nodes to new nodes without considering the transmission inside the old nodes group or the new nodes group. Whether these types of transmission can decrease the traffic or not is still unclear. For example, in [20], we mentioned that in their works, they defined Storage nodes X_i^{in}, X_i^{out} are represented by a storage input node X_i^{in}, and a storage output node X_i^{out}. These two nodes are connected by a directed edge $X_i^{in} \rightarrow X_i^{out}$ with capacity equal to the amount of data stored at node i. Based on this definition, in our research, we define our storage node with three functions $X_i^{in}, X_i^{mid}, X_i^{out}$ in order to avoid cyclic network that complex the model.

When there is transmission inside the old nodes group or the new nodes group, the network model would become a cyclic graph just like the Fig.17 (a). If both of the nodes X_1, X_2 are old nodes and there also exist data transmission. $\frac{M}{k}$ is the value of storage. Obviously, it is a cyclic graph which will make the model extremely complex. To improve this issue, we design a storage node with three functions $X_i^{in}, X_i^{mid}, X_i^{out}$ shown in Fig.17 (b).

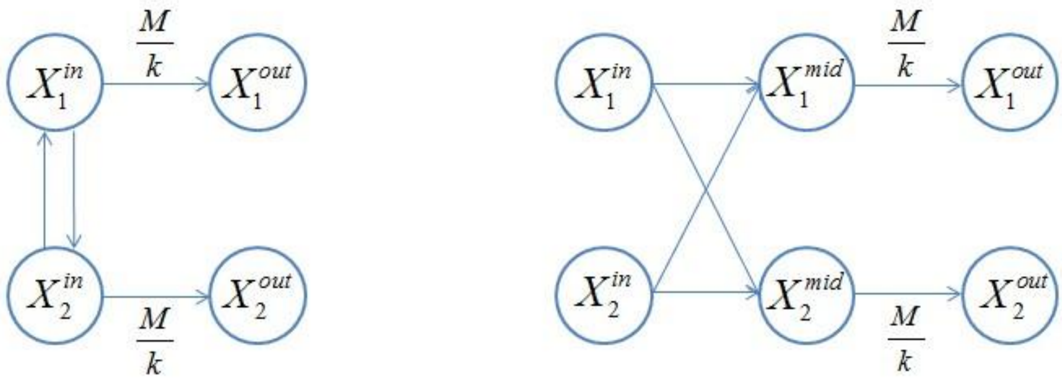


Fig.17 (a) storage node with X_i^{in}, X_i^{out}, (b) storage node with $X_i^{in}, X_i^{mid}, X_i^{out}$

Definition 3.3 we set one storage node into three parts $X_i^{in}, X_i^{mid}, X_i^{out}$, where the bound between X_i^{in}, X_i^{mid} is set as infinite ∞ while the bound between X_i^{mid}, X_i^{out} is set as the value of storage $\frac{M}{k}$.

By this definition, we assume each old/new node can get as many data pieces as they wish from other old/new node, but only remain $\frac{M}{k}$ in its buffer and discard the rest. The other scheme is to do network coding process, combine the received pieces and the existing pieces, then to keep in buffer. We will introduce this scheme in chapter four in detail.

3.3.4. Multi-Nodes-Loss Recovery Network Model

Combining the whole ideas above, we extend our WSDSN (n,k,r) into the information flow network model $G(n,k,r,\varphi)$. The example of $G(4,2,2,\varphi)$

3. NETWORK MODEL FOR WSDSN

Node:

(1) Each old node is represented by three functions $X_i^{in}, X_i^{mid}, X_i^{out}$. The bound between X_i^{in}, X_i^{mid} is set as infinite ∞ while the bound between X_i^{mid}, X_i^{out} is set as the value of storage $\frac{M}{k}$.

(2) Each new node is represented by three functions $Y_j^{in}, Y_j^{mid}, Y_j^{out}$. The bound between Y_j^{in}, Y_j^{mid} is set as infinite ∞ while the bound between Y_i^{mid}, Y_i^{out} is set as the value of storage $\frac{M}{k}$.

Edge (φ is the set for total value of all edges):

(1) The edge from virtual source node (VSN) to old node is represented as $VSN \rightarrow X_i^{in}$. The bound in this type is infinite ∞.

(2) The edge from old node X_i to new node Y_j is represented as $X_i^{out} \rightarrow Y_j^{in}$. The bound in this type is $\beta_{i,j}^{on}$. If old node X_i cannot link to new node Y_j, $\beta_{i,j}^{on} = 0$.

(3) The edge from old node X_i to another old node $X_{i'}$ is represented as $X_i^{in} \rightarrow X_i^{mid}$. The bound in this type is $\beta_{i,i'}^{oo}$. If old node X_i cannot link to rest old node $X_{i'}$, $\beta_{i,i'}^{oo} = 0$.

(4) The edge from new node Y_j to another new node $Y_{j'}$ is represented as $Y_j^{in} \rightarrow Y_{j'}^{mid}$. The bound in this type is $\beta_{j,j'}^{nn}$. If old node X_i cannot link to new node Y_j, $\beta_{j,j'}^{nn} = 0$.

(5) The sink node is represented as D. D links to k nodes to reconstruct the original files. The bound in this type is also is infinite ∞.

3. NETWORK MODEL FOR WSDSN

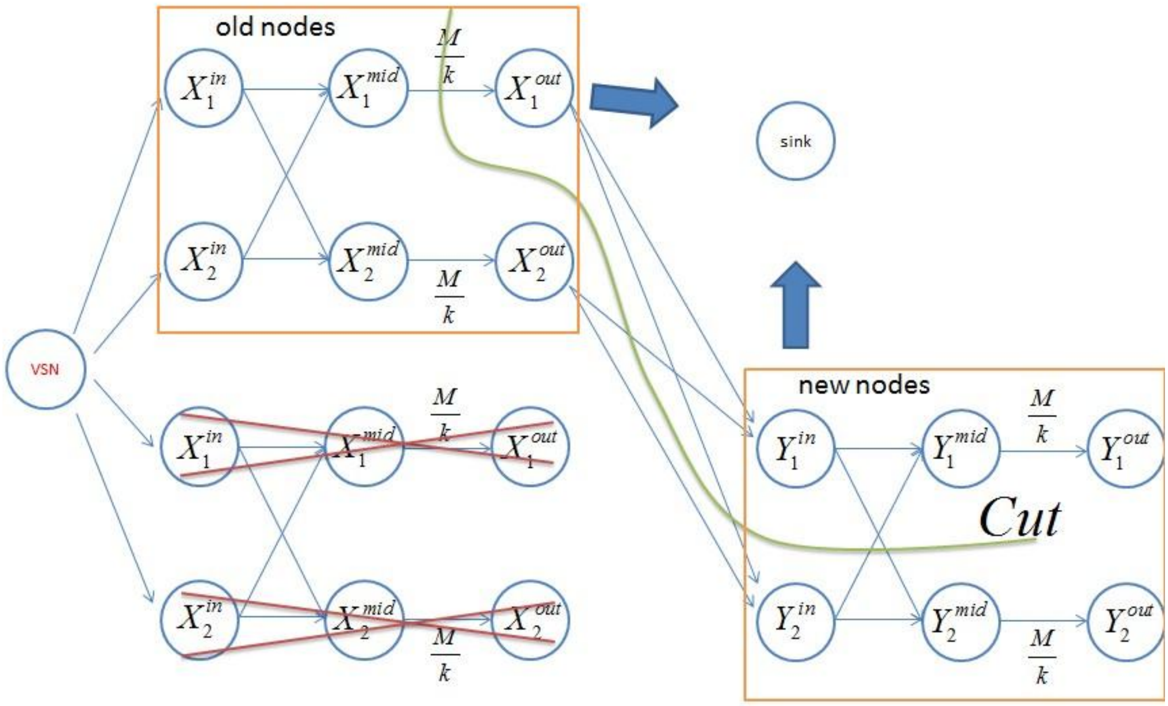


Fig.18 network flow model for $G(4,2,2,\varphi)$

According to our model, we will give our fist theorem.

Theorem3.1 During data recovery, the data transmission inside the old nodes group cannot decrease the traffic.

Proof sketch: From max-flow min-cut theorem, if the max-cut is not less than the original file M , there is a coding scheme which can reconstruct the file to provide recovery success. And if the min-cut is not more than the original file M , the max-cut is not less than the original file M . So we set $Min\text{-}cut(G) \geq M$, then the recovery will complete eventually. Since the bound $VSN \rightarrow X_i^{in}$ is infinite ∞ and the bound $X_i^{in} \rightarrow X_i^{mid}$ is also infinite ∞, it is impossible that min-cut will not cut $VSN \rightarrow X_i^{in}$ or $X_i^{in} \rightarrow X_i^{mid}$, which means $X_i^{in} \rightarrow X_i^{mid}$ will never be selected.

By considering the research goal for WSDSN that decrease the traffic as much as possible, we get the theorem that $X_i^{in} \rightarrow X_i^{mid}$ has no relation to the value of the min-cut in the model. $\beta_{i,i'}^{oo} = 0$. It is unnecessary for the data transmission inside old nodes group.

3.4. Summary

In this chapter, in order to analysis the most important indicators traffic cost, by using network flow theory, we turn the distributed storage recovery problem into the network flow model. Different from the former researches, we introduced virtual source node to turn repeatedly recovery problem into simple multi independent nodes one-loss network model. Moreover, by the storage node with $X_i^{in}, X_i^{mid}, X_i^{out}$, we make our model become an acyclic graph. By this conception, we get the theorem that it is unnecessary for the data transmission inside old nodes group.

However, in our model, we did not limit the bound for each edge independently. In practical world, the traffic for each edge is always different and restricted. We hope we can fix this issue as the future work.

4. Proposed System and Scheme for WSDSN

In chapter three, we introduced our distributed storage recovery model. By using network flow theory, our model is a network flow model which is easy to analysis the traffic cost. We also introduced virtual source node to turn repeatedly recovery problem into simple multi independent nodes one-loss network model. By the storage node with $X_i^{in}, X_i^{mid}, X_i^{out}$, we get the theorem that it is unnecessary for the data transmission inside old nodes group. In this chapter, we will present our scheme for WSDSN and compare it with former algorithms to demonstrate its advantages.

4.1. Conventional Algorithms in WSDSN

In this section, we will firstly give an overview of the main approaches proposed in distributed storage systems to store data redundantly across multiple storage servers: replication, erasure coding, and network coding. For each, we will show the storage space cost to store data redundantly and the network cost to restore the desired level of redundancy.

4.1.1. Replica

Replica (Replication) is a fundamental principle in ensuring the data availability and durability of data for any data storage system. Managing the number and placement of replicas is critical to this process. Systems re-replicate data when replicas fail, evaluate the correctness of replicas in the system, and move replicas among sites to meet availability goals [21].

In detail, the client stores one file replica at each of nodes. Thus, the original file can be recovered from any of the nodes. The storage space cost is the size of file M across all nodes per node. Upon detecting corruption of a replica, the sink node can use any one of the healthy replicas to create a new replica. As part of the repair component, in order to create a new replica of size M, the nodes need to retrieve a replica of size M. In [22], the authors introduced a typical scheme about Replica. You can find more details about this scheme.

However, in WSDSN environment, there lack constructs that allow them to securely determine the number and location of replicas in the system. The restrictions in WSDSN such as computational power, energy consumption, make it very unsuitable for WSDSN. It is certain that make hundreds of copies can provide the reliability of data but obviously it is not the optimal choice.

4.1.2. Decentralized Erasure Codes

Decentralized Erasure Codes are random linear codes over a finite field $F_q = GF(q)$ with a specific generator matrix structure. Once a storage node receives one or more data packets, it selects coefficients uniformly and independently from $GF(2^8)$ in common situation. Each coefficient then multiplies each block independently, multiple blocks are added and the results are cascaded into a new block packet that has exactly the same size as all the data packets [23].

In detail, in decentralized erasure coding, given a file $F_q = GF(q)$ of k blocks, the system uses an (n,k) maximum distance separable erasure code (MDS codes) to create n coded blocks out of the original k file blocks, and stores them at n nodes (one coded block per server). Thus, the original file can be recovered from any k out of the n servers. Whenever the client detects corruption of one of the coded blocks, it can use the remaining healthy blocks to regenerate the corrupted coded block. In [24], the authors introduced a typical scheme about Replica. You can find more details about this scheme.

However, compared with Replica, decentralized erasure coding has a higher network overhead cost for the repair component. Since to create one new coded block, the system has to first reconstruct the entire file (i.e., retrieve k coded blocks), thus incurring a network overhead factor of k . The traffic is extremely high and unacceptable in WSDSN.

4.1.3. Former Network Coding Based Scheme

We have introduced enough about the network coding, so we want to save the room for contents. Recently, researchers did lots of works to improve the network coding performance on distributed storage system. In [25] and [26], authors have shown that the network overhead factor k for the repair component is not unavoidable and in most cases, it was commonly believed. Given a file represented by input blocks with the size of M , the client uses network coding to generate coded blocks as linear combinations of the original m file blocks. In [18], authors introduced us two schemes. First one is OMMDS (Optimally Maintained MDS), the second one is regenerating codes. Both of these schemes can reduce traffic use by 25% or more compared with the best previous design—a hybrid of Replica and erasure codes system.

However, there are still some drawbacks in the former research. For example, in [18], the authors restricted that all the new nodes should connect to same amounts of old nodes to doing recovery. This is effective when in some stable network like Google, Amazon or something but in WSDSN. If all the nodes can only connect to the minimum amounts of nodes, some routing paths will inevitable to be wasted. In [27], the authors gave us a scheme named simple regenerating codes, by combining MDS codes and simple locally decodable parities, their scheme is easily implemented into practical world. But their core idea is a RAID 5, only works good in peer-to-peer network or cloud system.

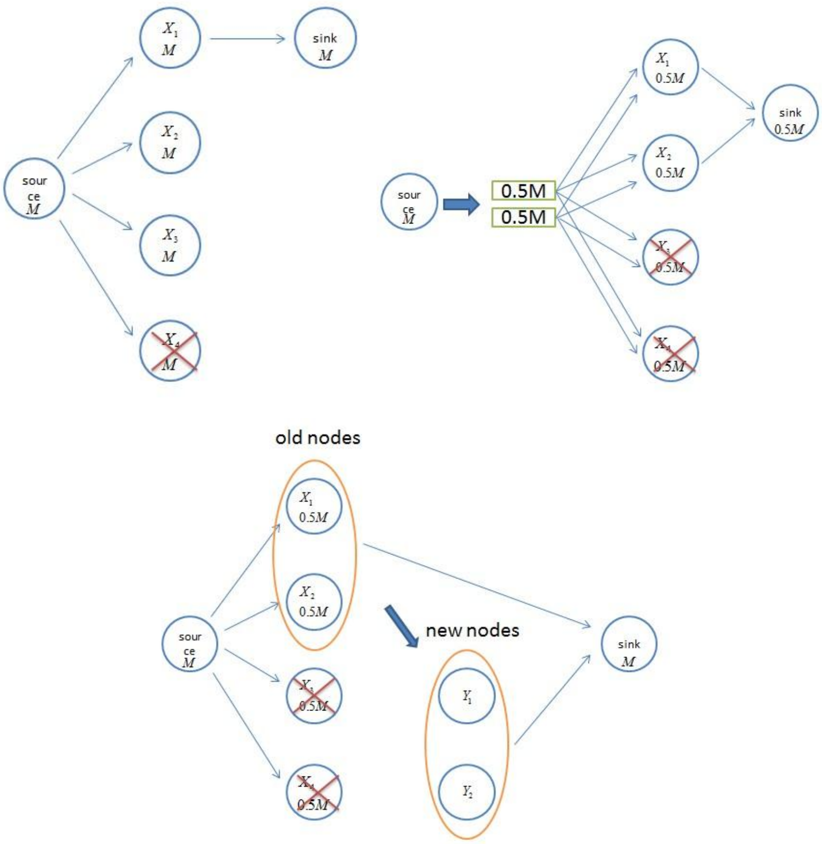


Fig.19 Example of various approaches for redundantly storing a file with size of 1MB, (a) replica, (b) decentralized erasure codes, (c) network coding

4.2. MFRC-WSDSN Network Model

As mentioned above, former researches have mainly two problems: 1. their schemes are not suitable or difficult to apply for WSDSN environment 2. Their schemes are not efficient or need to be modified to get a better performance.

For the first type of problems, there is nothing we can do to improve since no matter how we modify the scheme cannot make it become suitable for WSDSN. Our research focus on the second issue: how to improve the performance.

In most of former studies, their schemes restricted that all the new nodes should connect to same amounts of old nodes to doing recovery. It is too idealized that be resource-prodigal to achieve in WSDSN. In WSDSN, some nodes can link with many nodes while some nodes can only link to a few nodes. During the data recovery, if we enforce all the nodes link with the minimum amount of node, $d_{\min}$, those nodes which can connect to more other nodes would waste many paths to decrease the recovery efficiency.

For instance, in a WSDSN (5, 2, 3) network, among nodes X_1,X_2,X_3,X_4,X_5, X_3,X_4,X_5 become failed, the system chose another three nodes Y_1,Y_2,Y_3 to recovery the data. However, the number of nodes that they can link with is different. Node Y_1 can link to X_1,X_2, node Y_2 can link to X_1,X_2,Y_1 and node Y_3 can link to X_1,X_2,Y_2. By former researches, $d_{\min}=2$, we enforce Y_1,Y_2,Y_3 to link with two nodes to recover data so that there is one paths been wasted for Y_2,Y_3 respectively.

In order to fix this situation, we give a multi-loss flexible regenerating code recovery (MFRC) scheme. This scheme can make new node Y_j to link with any d_j nodes to recover the data, d_j is independent with node Y_j, which makes recovery process more flexible to adapt WSDSN environment. And with the Simple regenerating codes, our scheme is also easy to implement and work well in the simulation.

Definition 4.1 MFRC-WSDSN network model

4. PROPOSED SYSTEM AND SCHEME FOR WSDSN

Assume a file of size M being upload to n chosen nodes $X_1,X_2,...X_{n\text{-}1},X_n$ while maintain the characteristic of (n, k) coding. When r nodes $X_{n\text{-r-}1},X_{n\text{-}r},...X_n$ failed, system will chose other r nodes $Y_1,Y_2,...Y_{\text{r-}1},Y_r$ to recover and maintain the coding function. Moreover, $Y_1,Y_2,...Y_{\text{r-}1},Y_r$ can link to $d_1,d_2,..d_r$ nodes to do the data recovery, we define $\delta=\{d_1,d_2,..d_r\}$. In the end, sink node can pick up any k nodes ($k<n$) to decode the original file. We call this network as a MFRC-WSDSN (n,k,r,δ) network. According to Theorem3.1, the data transmission inside the old nodes group is no longer involved in our model.

Just like definition 3.1, we also make the process in MFRC-WSDSN into 3 stages: data upload, data recovery, data download.

(1) Data upload

At the very beginning, the source node store a file of size M, then encodes the file into n pieces, transmit them evenly to n chosen nodes $X_1,X_2,...X_{n\text{-}1},X_n$. Any k nodes ($k<n$) can decode the original file, each decoding packet has a storage size of $\frac{M}{k}$.

(2) Data recovery

When r nodes $X_{n\text{-r-}1},X_{n\text{-}r},...X_n$ become malfunction, in order to maintain the reliability of network while keep the whole redundancy, system chose some more r nodes $Y_1,Y_2,...Y_{\text{r-}1},Y_r$, utilize them (we call them new nodes) getting data pieces from $n\text{-}r$ alive nodes $X_1,X_2,...X_{n\text{-r-}1},X_{n\text{-}r}$ (we call them old nodes) to recover the lost part. More specific is shown below:

Step one: Every new node $Y_j\left(1\le j\le r\right)$ can connect to other d_j rest available old and new nodes to download the data piece. $d^{old}+d^{new}=d_j$. Let $\beta^{on}_{i^{'},\text{j}}$ be the data traffic from old node $X_{i^{'}}$ to new node Y_i while $\beta^{nn}_{j^{'},j}$ be the data traffic from new node $Y_{j^{'}}$ to new node Y_j.

4. PROPOSED SYSTEM AND SCHEME FOR WSDSN

Step two: Every new node $Y_j\left(1\leq j\leq r\right)$ creates a new coded packet of size $\frac{M}{k}$ by received data pieces $\beta^{on}_{1,j},\ldots,\beta^{on}_{n\text{-}r,j}$ and $\beta^{nn}_{1,j},\ldots,\beta^{nn}_{j\text{-}1,j},\beta^{nn}_{j+1,j},\ldots\beta^{nn}_{n\text{-}r,j}$.

(3) Data download

After the data recovery, the system stores the data by n-r old nodes $X_1,X_2,\ldots X_{n\text{-}r\text{-}1},X_{n\text{-}r}$ and r new nodes $Y_1,Y_2,\ldots Y_r$. Any sink node can decode the original data from any k nodes ($k<n$).

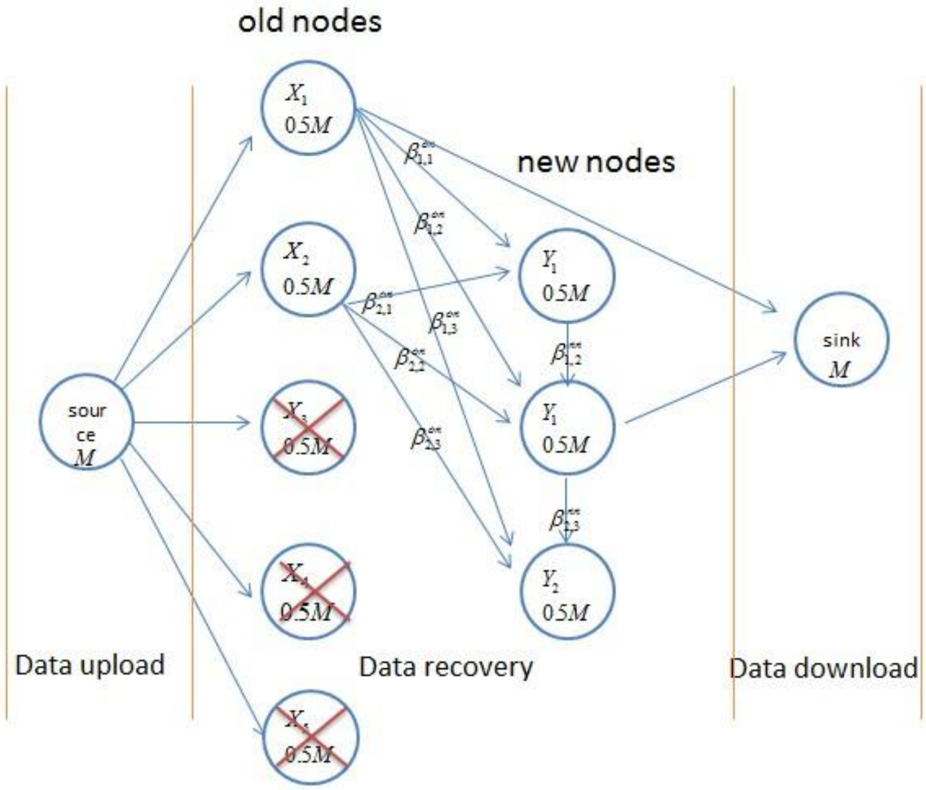


Fig.20 MFRC-WSDSN (5, 2, 3, {2, 3, 3}) model

For MFRC-WSDSN model, we set as following:

(1) The recovery capability from any nodes to new node $Y_{j'}$ is same. $\beta^{on}_{i,j}=\beta^{nn}_{j',j}=\beta_j$.

$\left(1 \le i \le n-r, 1 \le j^{'} \le r\right)$. We make this presumption since normal system are symmetrical, if the each node`s capacity is different, it is extremely complicated to analysis the network.

(2) For new node $Y_{j^{'}}$, old nodes $X_1, X_2, \ldots X_{n\text{-r-1}}, X_{n\text{-}r}$ and d_j new nodes $Y_1, Y_2, \ldots Y_{j-1}$ are its upper nodes. During the recovery, the process is one node by one node. New node only chose old nodes and recovered new nodes to get data pieces. Those new nodes that in the middle of recovery would not be linked with. This presumption also simply the flow graph and make the model easier to understood.

Node in MFRC-WSDSN (n,k,r,δ):

(1) Each old node is represented by three functions $X_i^{in}, X_i^{mid}, X_i^{out}$. The bound between X_i^{in}, X_i^{mid} is set as infinite ∞ while the bound between X_i^{mid}, X_i^{out} is set as the value of storage $\frac{M}{k}$.

(2) Each new node is represented by three functions $Y_j^{in}, Y_j^{mid}, Y_j^{out}$. The bound between Y_j^{in}, Y_j^{mid} is set as infinite ∞ while the bound between Y_i^{mid}, Y_i^{out} is set as the value of storage $\frac{M}{k}$.

Edge (φ is the set for total value of all edges) in MFRC-WSDSN (n,k,r,δ):

(1) The edge from virtual source node (VSN) to old node is represented as $VSN \to X_i^{in}$. The bound in this type is infinite ∞.

(2) The edge from old node X_i to new node Y_j is represented as $X_i^{out} \to Y_j^{in}$ while the edge from $Y_{j^{'}}$ to Y_j is $Y_{j^{'}}^{mid} \to Y_j^{in}$. The bound in both of these types is β_j.

(3) The sink node is represented as D. D links to k nodes to reconstruct the original files. The bound in this type is also is infinite ∞.

4.3. Network Flow Model Analysis

The goal for this section is to find low bound β_j for each new node $Y_{j'}$ to promise the data collector node D can always find k nodes ($k<n$) to reconstruct the original data.

Theorem4.1 For the data of size M in in MFRC-WSDSN (n,k,r,δ), if there is a linear network coding scheme with $\beta_j = \dfrac{M}{k\left(d_j\text{-}k+1\right)}$, it is the optimal scheme.

Proof: In order to simplify the proof, assume that he data collector node D link with $X_1^{out}, X_2^{out}, \ldots X_{k\text{-}1}^{out}, Y_j^{out}, Y_j^{in}$ have downloaded β_j data from d_j nodes. If the reconstruction is successful, the value of min-cut is not less than M to promise all the cuts is larger than M. We can make a variable cut valued β_j to calculate the lower bound.

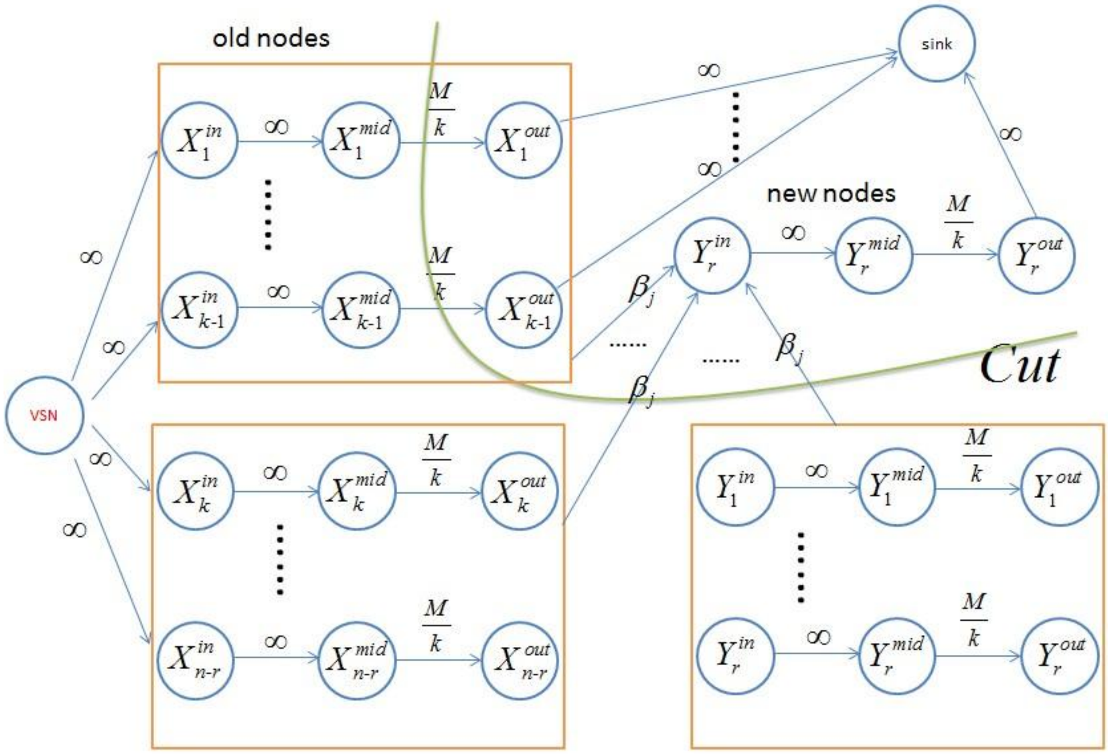


Fig.21 a cut in MFRC-WSDSN (n,k,r,δ) model

We select a cut in MFRC-WSDSN (n,k,r,δ), it cross the edges between X_i^{mid} and X_i^{out}, the edges which end at Y_j^{in} and not start at $X_i^{out}\left(1 \le i \le k\text{-}1\right)$ like in Fig.21. There are (k-1) edges between X_i^{mid} and X_i^{out}, the value in this type is $\frac{M}{k}$. Assume there are t (t $\le$ k-1) edges from $X_1^{out}, X_2^{out}, \ldots X_{k\text{-}1}^{out}$ to Y_j^{in}, this cut will cross the edges $\left(d_j\text{-}t\right)$ to Y_j^{in}, all the value in this type is β_j.

The value of this cut is: $cut\left(S,D\right) = \frac{\text{(k-1)M}}{k} + \left(d_j\text{-}t\right)\beta_j$

From max-flow min-cut theorem: $cut\left(S,D\right) \ge \min\text{-}cut\left(G\right) \ge M$

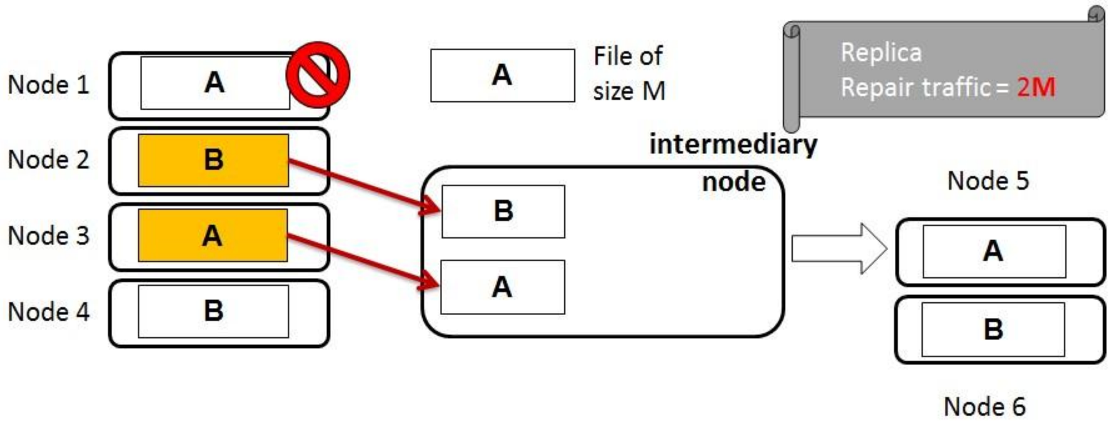
Since $t \le k\text{-}1$, we get $\beta_j \ge \frac{M\left(d_j\text{-}t\right)}{k}$

If the reconstruction is successful, $\left(\beta_j\right)_{\min} = \frac{M\left(d_j\text{-}t\right)}{k}$

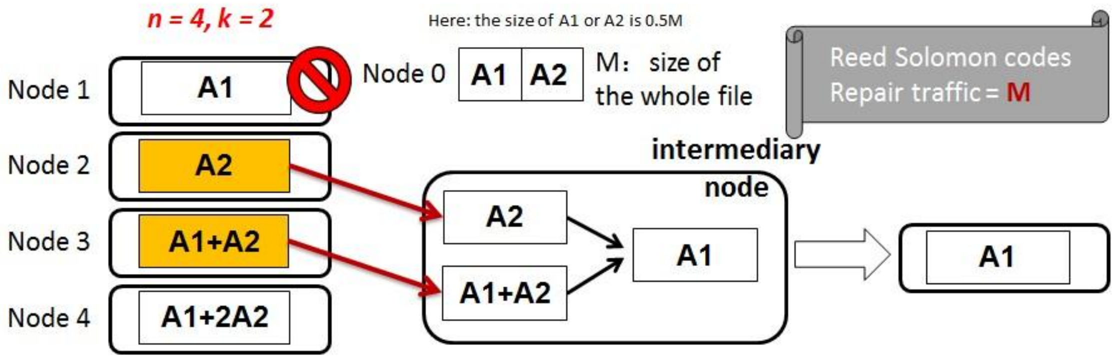
In other word, if there is one linear network coding scheme can maintain $\beta_j = \left(\beta_j\right)_{\min}$, it is the optimal scheme to get the best traffic in MFRC-WSDSN (n,k,r,δ). In next section, we will introduce our scheme in detail.

4.4. Multi-Loss Flexible Regenerating Code Recovery (MFRC) Scheme

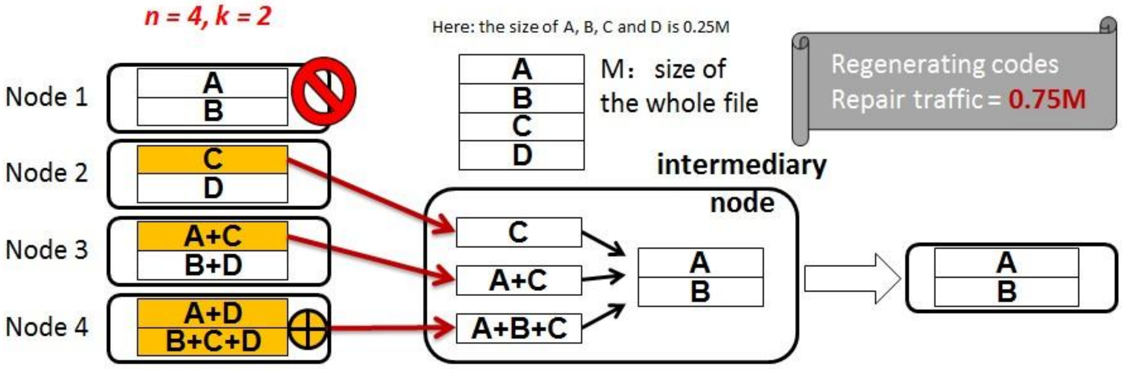
Before this core section of our paper, it is necessary to review the existing schemes again to make a deeper impression of ours when comparing them at later. Since I have introduced part of them above, this part we will only give a brief graphic illustration in Fig.22.



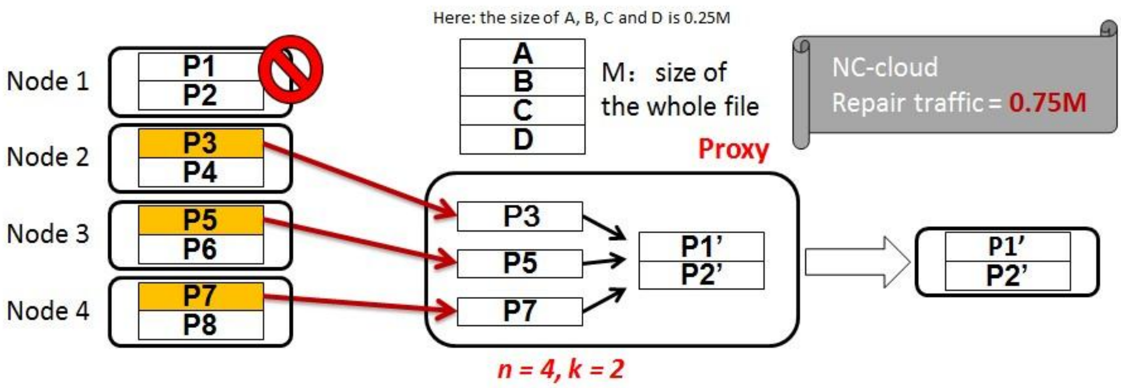
(a) Replica



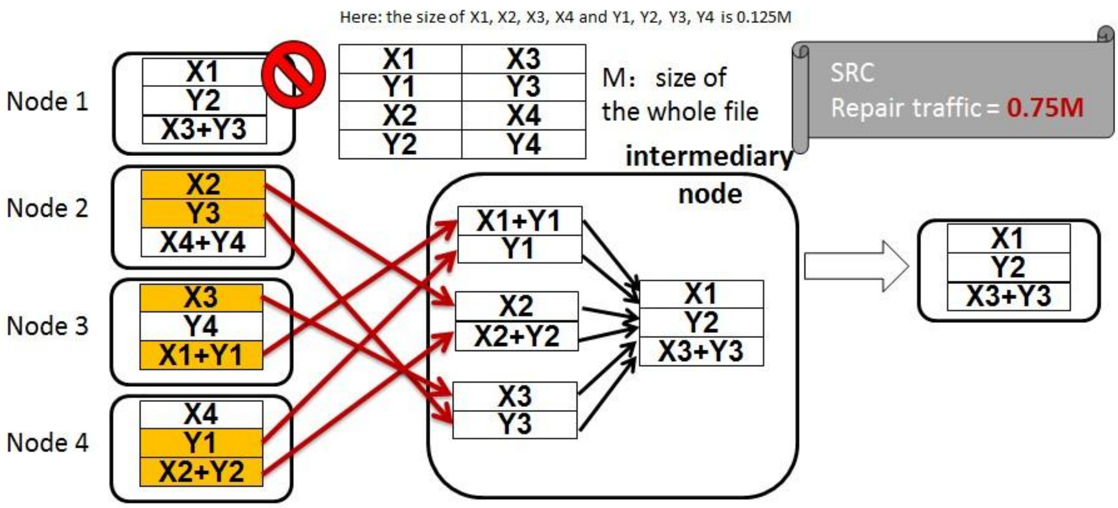
(b) Reed Solomon Code



(c) Regenerating Code



(d) Network Coding Cloud



(e) Simple Regenerating Code

Fig.22 examples of several existing storage mechanism: (a) Replica [21] (b) MDS code [23] (c) Regenerating codes [25] (d) NC-cloud [28] (e) SRC [27]

4.4.1. Important Coefficients

In order to evaluate the performance of above algorithms, we need to introduce some coefficients.

(1) S

S stands for the ratio of original data value to total stored data per node. S= (original data/ total coded data) for each storage node.

(2) F

F stands for the maximum of tolerant invalid node.

(3) C

C stands for the traffic cost to repair one invalid node. C = (recovery traffic cost/ original data) for each storage node.

(4) A

A stands for the amount nodes been used to repair one invalid node.

(5) R

R stands for the coding rate. As we all know, R=k/n.

name	(n, k)	R	(S, F, C, A)	Conclusion(merits & demerits)
Replica	(3,1)	33.3%	1, 2, 1, 1	Easy to implement, Waste storage space
MDS code	(4,2)	50%	0.5, 2, 2, 2	High fault-tolerant, Waste some repair traffic
NC-cloud	(8,4)	50%	0.5, 2, 1.5, 2	Less traffic cost, Based on cloud storage system
SRC	(12,8)	66.7%	0.375, 1, 2, 2	flexible, high coding rate, not suitable for multi-loss situation
RAID5	(3,2)	66.7%	0.5, 1, 2, 2	Common RAID, Recovery nodes should on same strip

Table.2 conclusion of existing mechanism

The all-known (n, k) property: a code will be storing information in n storage nodes and should be able to tolerate any combination of n-k failures without data loss. We refer to this property as R, the coding rate. Normally, the higher R, the higher storage efficiency is. However, if the system has a high storage efficient, the fault-tolerant is normally low. In other word, R is in direct proportion to storage efficient while it is also in indirect proportion to fault-tolerant. This subtle tradeoff can be observed in Table.2.

When one node failed, system needs to repair the data integrity by using live nodes. It is sure that Replica has the best performance in traffic cost field. The other schemes` traffic costs are all greater than or equal to 2. This issue is related to Exclusive OR operation. OR needs at least two parameters, so one recovery need at least two data pieces.

Inspired by these existing scheme, we develop our MFRC scheme---easy to implement, have a good tradeoff between storage efficiency and fault-tolerant.

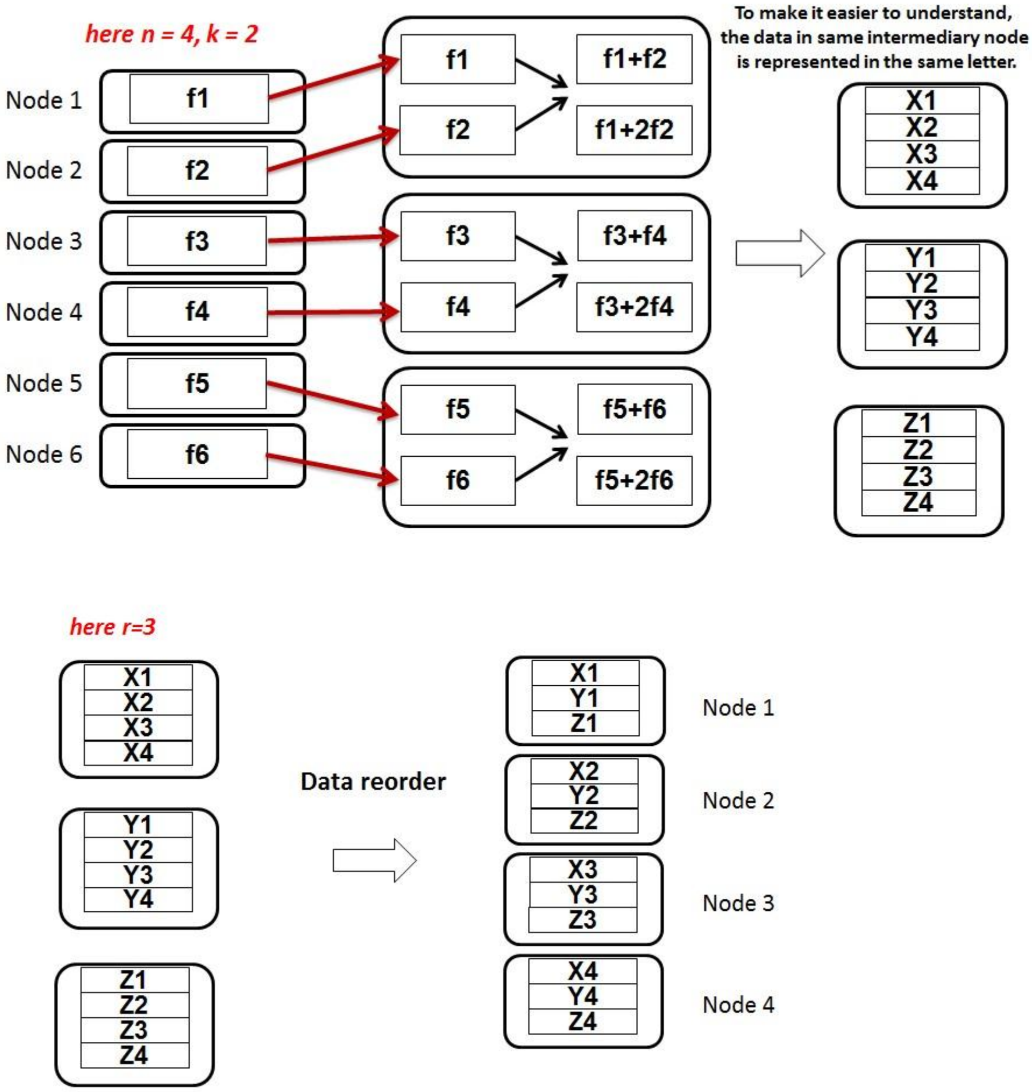
4.4.2. Simple Generation Example

In short, our system is a hybrid system. It combines several schemes with their merit: Replica can save traffic cost and easy to implement, MDS code can have high fault-tolerant, network coding improves the storage efficiency. In this section, I will give a simple example by MFRC (4, 2, 3) scheme.

(1) Data Upload Process

First, the row data collected by sensor nodes do (n, k) MDS code to have the multi-loss recovery property. Then in each storage node, system uses some proportion of memory to store the

Replica, some space to store the encoded pieces.
We give a first overview of our encode construction through a simple example in Fig.23, which shoes an (n=4, k=2, r=3) - MFRC scheme.



4. PROPOSED SYSTEM AND SCHEME FOR WSDSN

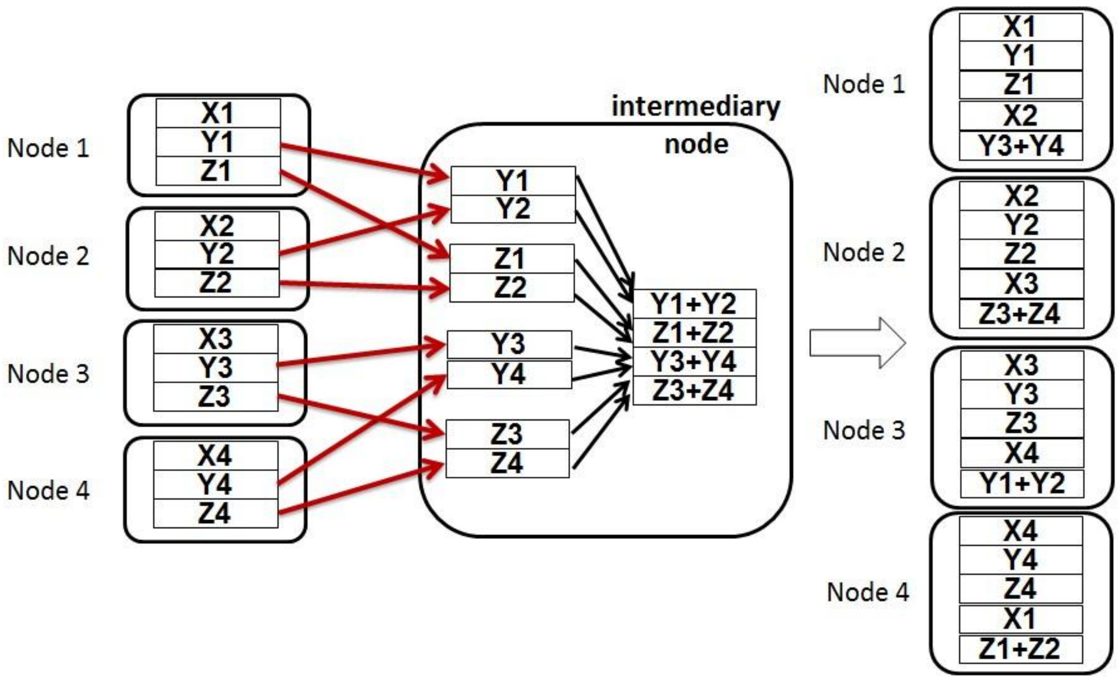


Fig.23 Example of a (n=4, k=2, r=3) - MFRC encoded scheme

The row data collected by sensor nodes is split in 6 chunks f1; f2; f3; f4. We first encode $[f_1\ f_2]$ in $[X_1\ X_2 X_3\ X_4]$, in $[f_3\ f_4]$ in $[Y_1 Y_2 Y_3 Y_4]$ and $[f_5\ f_6]$ in $[Z_1\ Z_2 Z_3\ Z_4]$ using any standard (4, 2) MDS code. This can be easily done by multiplication of the data with the 2×4 generator matrix G of the MDS code to form $[X_1\ X_2 X_3\ X_4]=[f_1\ f_2]$G, $[Y_1 Y_2 Y_3 Y_4]=$ $[f_3\ f_4]$G and $[Z_1\ Z_2 Z_3\ Z_4]=[f_5\ f_6]$G. Now we get twelve data pieces.

Next, the system encodes the data pieces. The core idea is using other pieces to encoding and the more strips, the better. For instinct, in node no.1, the 60% part of memory stores replica, other part stores the encoded pieces. The encoded piece X_2 and $Y_3 \oplus Y_4$ are in 2 chunks which is from 3 strips. We circularly place these chunks in 4 nodes, each storing 5 consisted with three parts, replica, other strips replica and encoded pieces.

(2) Data Recovery Process

We show an example by failing node1 in Fig.24. When it becomes invalid, the system first checks neighbour nodes memory to find any replica directly. Then by XOR operation, system can decode rest pieces. Any two nodes contain X_i, Y_i, Z_i chunks which through the outer MDS codes can be used to recover the original data object. We note that the parity chunks are not used in this process, which shows the sub-optimality of our construction. In short, the newcomer reconstructs each lost chunk by downloading, accessing, and XORing 2 other chunks.

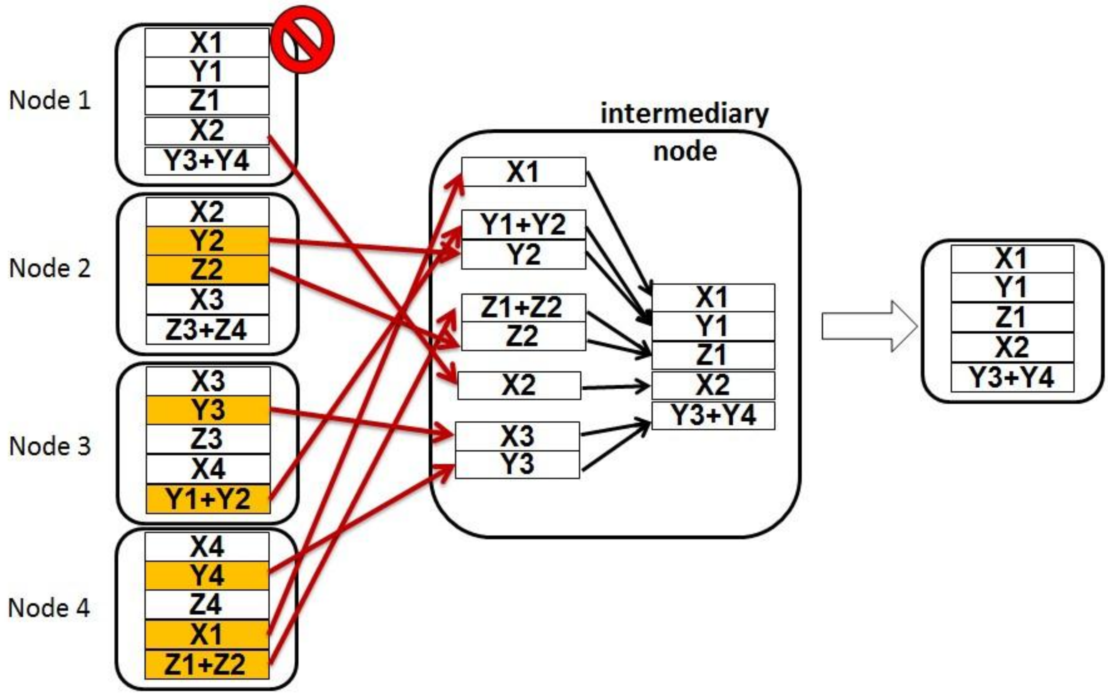


Fig. 24 Example of a (n=4, k=2, r=3) - MFRC decoded scheme

In Fig. 24, we give an example of a single node repair of the (4, 2, 2) - MFRC. We assume that node1 is lost and a newcomer joins the system. To reconstruct, the newcomer has to download X_2 and Y_1 from nodes 4 and 2. The rest parts should be recovered by XOR operation. For example, $Y_1 = Y_2 \oplus (Y_1 \oplus Y_2)$, two pieces are individually from node 2 and 3.

$Z_1 = Z_2 \oplus \left(Z_1 \oplus Z_2\right), Y_3 \oplus Y_4 = Y_3 \oplus Y_4$.

(3) Data Download Process

After the data recovery, any sink node can decode the original data from any k data pieces ($k<n$).In short our codes combine outer MDS codes and simple parities to provide fault tolerance and efficient repair respectively. Due to this separation of duties, our codes are suboptimal.

4.4.2. General Generation Construction

Now we will present our general MFRC construction. We use r parallel and identical MDS outer pre-codes and generate a single parity vector from f encoded parts. We circularly place the generated chunks in n storage nodes. The (n, k, r)-MFRC is an (n, k) erasure code with coding rate R. Our scheme always attains a $\frac{r}{r+1}$ fraction of the space efficiency of an (n, k) MDS code, for the same reliability, but with simple and low cost node repair. Last section, we perform single node repairs in the same manner as the r=3. In order to repair a chunk, we access r nodes and perform a simple addition. For any (n, k, r), we proceed by introducing the general code construction and showing its properties in this section.

(1) Data Upload Process

Let a file of size M, that is packetized in n parts, $M = \left[n^1, n^2, ...n^n\right]$, with each $n^i, i \in \{n\}$, have the size of k. We encode each of the n file parts independently, into vectors X^i of length n, using an outer (n, k) MDS code. That is, we have $X^1 = n^1 \cdot G, X^2 = n^2 \cdot G, ..., X^n = n^n \cdot G$, where G is the $n \times k$ MDS generator matrix. The outer MDS code can be any scalar or array (n, k) MDS code, i.e., we pose no requirements on its design or finite field size.

Then we generate a single parity sum vector from all the coded vectors $S = \sum_{i=1}^{n}\sum_{j=1}^{r} X_j^{(i+1)}$.

This process yields a total coded chunks in the $X_j^{(i+1)}$ vectors and n parity chunks in S.

Now we will circularly place these $(r+2)n$ chunks in n storage nodes, with each node storing

$r+1$ coded chunks and 1 sum chunk. Hence each node expends $\alpha = r+2 = \frac{r+2}{r}\frac{M}{k}$ in storage capacity. The placement will again obey the property that enables easy repair: no two chunks within a storage node should share the same subscript. To ensure successful repair we also require that r must be odd. We state the circular placement of chunks in the $i-th$ node, for $i \in \{n\}$.

Node 1	Node 2	...	Node n-1	Node n
$X_1^{(1)}$	$X_1^{(2)}$	...	$X_1^{(n-1)}$	$X_1^{(n)}$
$X_2^{(1)}$	$X_2^{(2)}$	...	$X_2^{(n-1)}$	$X_2^{(n)}$
$\vdots$	$\vdots$	...	$\vdots$	$\vdots$
$X_r^{(1)}$	$X_r^{(2)}$	...	$X_r^{(n-1)}$	$X_r^{(n)}$
$X_1^{(2)}$	$X_1^{(3)}$	...	$X_1^{(n)}$	$X_1^{(1)}$
S_1	S_2	...	S_{n-1}	S_n

Table.3 array chunks of n storage nodes

Theorem4.2: The (n, k, r)-MFRC can tolerate any combination of n-k node erasures and has coding rate $R_{MFRC} = \frac{r}{r+2} \cdot \frac{k}{n}$.

Proof sketch: The MDS pre-codes guarantee perfect file reconstruction posterior to any n-k erasures. The file can certainly be reconstructed by connecting to any k nodes: any collection of k nodes contain $(r+1)k$ distinct coded chunks, k of each file part. Each of these k-tuples of coded chunks can give back the information chunks of a single file part due to the $r+1$ outer MDS codes.

The effective coding rate of the (n, k, r)-MFRC is equal to the ratio of the initial file size to the expedited storage, that is $R_{MFRC} = \frac{file \quad size}{storage \; spent} = \frac{r}{r+2} \cdot \frac{k}{n}$

By the above theorem we can claim that the rate of the MFRC is a fraction $\frac{r}{r+2}$ of the coding rate of an (n, k) MDS code, hence is upper bounded by $\frac{r \cdot k}{(r+2) \cdot n} = \frac{r \cdot k}{(r+2) \cdot (\mathrm{k+m})} \xrightarrow{k \to \infty} \frac{r}{r+2}$

(2) Data Recovery Process

In this subsection we prove the repair properties of the MFRC, which are summarized in the following theorem.

Theorem 4.3: The repair of a single chunk of the (n, k, r)-MFRC, where each node stores $\alpha = r+2 = \frac{r+2}{r} \frac{M}{k}$, costs $\frac{M}{k}$ in repair traffic and $(r+1)$ in chunk reads, and disk accesses. The repair of a single node failure costs $\beta_{MFRC} = (\mathrm{r}+2) \frac{M}{k}$ in repair traffic.

Proof sketch: Let node $i \in [n]$ fail. A new comer node can reconstruct the lost chunk by accessing all other nodes and downloading the chunk of each node that has the same subscript $i \oplus (l\text{-}1)$ as the lost chunk. For example to reconstruct $X_j^{(1)}$, we need to perform the following steps:

Step	Repair chunk $X_j^{(1)}$
1	Access $Disk_{\mathrm{j-1}}$ and download $X_j^{(1)}$
2	Access $Disk_{\mathrm{j-2}}$ and download $X_j^{(2)}$
⋮	⋮
r	Access $Disk_{\mathrm{j}}$ and download S_j
r+1	Restore $X_j = \mathrm{S}_j - \sum_{l=2}^{r} X_j^{(l)}$

Table.4 repair step for one storage node

(3) Data Download Process

After the data recovery, any sink node can decode the original data from any k data pieces ($k<n$). MFRC has the property that a single node failure can be repaired by any subset of d remaining nodes, and $k \leq d \leq n\text{-}1$ is fixed by the specific code design.

4.5. Summary

In this chapter, we presented our MFRC scheme for WSDSN and compare it with former algorithms to demonstrate its advantages. Our scheme is hybrid by severe existing schemes. We theoretically show that our codes have the (n, k) reliability, have asymptotically optimal storage and are within a logarithmic factor from optimality in repair traffic. One significant benefit is that the number of nodes that need to be contacted for repair can be made a small constant, independent of n, k. Further, MFRC can be easily implemented by combining any prior MDS code implementation with XORing of coded chunks and the appropriate chunk placement into nodes.

5. Simulation Results

In chapter four, we introduced our MFRC scheme for WSDSN. In this chapter, we will present a comparison of the proposed codes with replication and Reed-Solomon codes using a network coding simulator. Firstly, we will give a brief introduction of this simulator and our implement environment. Then compared by repair performance such as repairing rate, average packets and processing time, we will give the merits of our scheme. Finally, we will give performance estimation for super-large-scale WSDSN.

5.1. Simulation Platform

We chose NECO (Network Coding Simulator) as our simulation platform [30]. NECO is a simple high-performance simulation framework dedicated to the evaluation of Network Coding based protocols. Its main features include (1) definition of graphs representing the topology (which can be randomly generated by NECO or given through a standard representation), (2) the modular specification of network coding protocols, (3) visualization of network operations and seamless statistics module. The simulator is entirely written in Python and is easily extensible to account for extra modules [31]. The simulator is entirely written in Python and is easily extensible to account for extra modules.

Although implementations in NS [32], [33], Op-net [34] and other general network simulators have the significant advantages of using well-known frameworks and accessing a wide array of available libraries, there are also some underlying disadvantages. First, there is no standard network coding library for these platforms. Second, since network coding is most beneficial for unreliable and large networks, a crucial feature of such a simulator is to be able to simulate complex networks. These also are the important reasons for us to choose NECO as our platform.

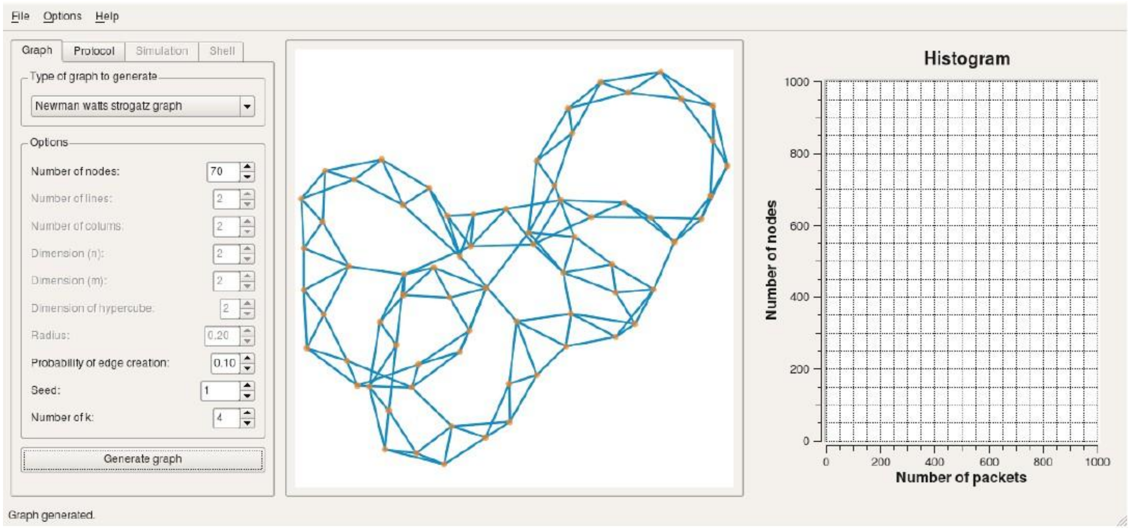


Fig. 25 Screenshot of the graph tab

5.2. Simulation Results

We first present the architecture of the cloud storage system that our simulator is modeling. The architecture contains one sink node as a master data server and rest nodes to collect and transmit data. As a wireless sensor distributed storage system may store up to tens of petabytes of data, we expect numerous failures and hence fault tolerance and high availability are critical. To offer high data reliability, the master server needs to monitor the health status of each storage server and detect failures promptly.

We implemented a discrete-event simulator of a storage system using a similar architecture and data repair mechanism. To provide accurate simulation results, when performing repair jobs, the simulator keeps track of the details of each repair process which gives us a detailed performance analysis.

We manually set the r=8, loss rate at 1% for the source nodes and 3% for the sink node because of the massive data transmission in the final data download. Then we ran the system by severe schemes including replica, MDS code and our MFRC code in our simulator. After the simulation, we analyzed log file and called the severe parameters, we also collected the repair time of each chunk from the simulation.

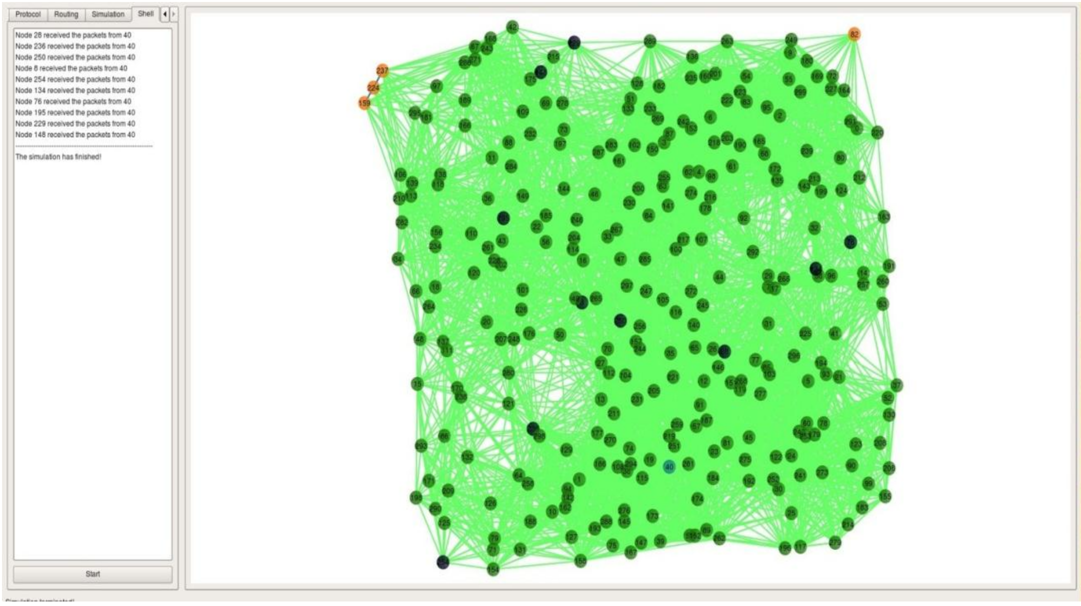


Fig. 26 network topology in our simulator

In Fig.26, we show the network topology in our simulator, NECO randomly choses node No.40th as the sink node, the rest 90% nodes as the data collector nodes. At the beginning, the most nodes are yellow, after successful coding and transmission, they turned into gray. If there is a failed routing path, the original blue line will maintain blue instead of turning into green.

5.2.1. Packets per Node Performance

Besides the repair traffic, which is the most important parameter in WSDSN, packets per node during the transmission is also a significant one to evaluate the performance. It can evaluate how a algorithm can balance the network traffic. The x-axis stands by number of packets each node delivered, the y-axis stands by number of nodes. We compare three codes: 3-way replication, Reed-Solomon (16, 8) codes and our scheme. The result is presented in Fig.27, 28, 29, respectively.

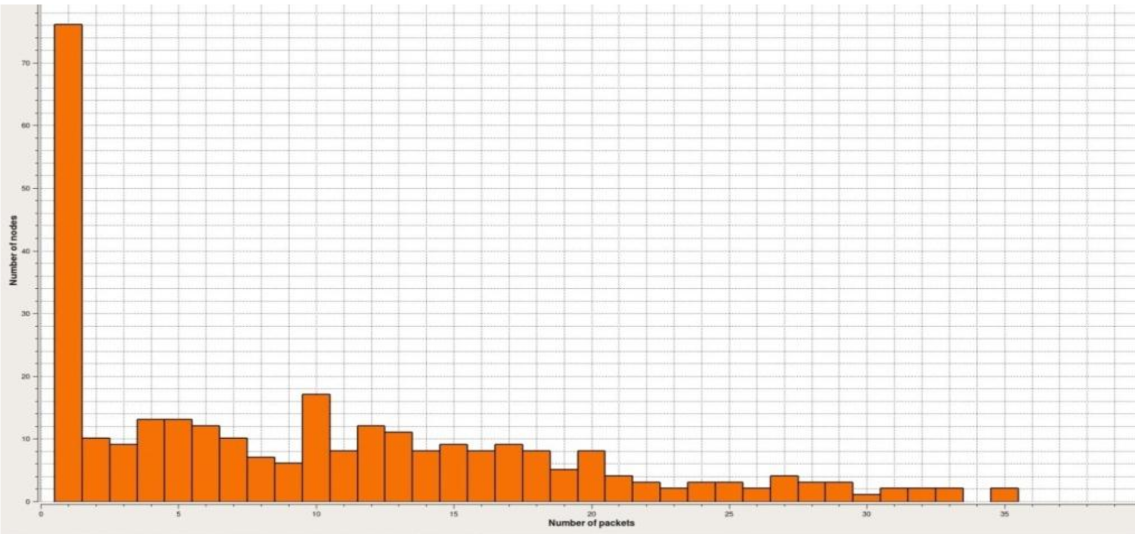


Fig.27 Packets per node for 3-way replica

This is the result for Replica. As I mentioned before, this method will give the most reliable assurance for the transmission in theory. But when in the simulation, the distribution of data is not ideal. Almost 30% nodes deliver the data packets only once. This means in these 30% nodes, lots of potential have been wasted. For the rest 70% nodes, their works become much heavier corresponding to the rest.

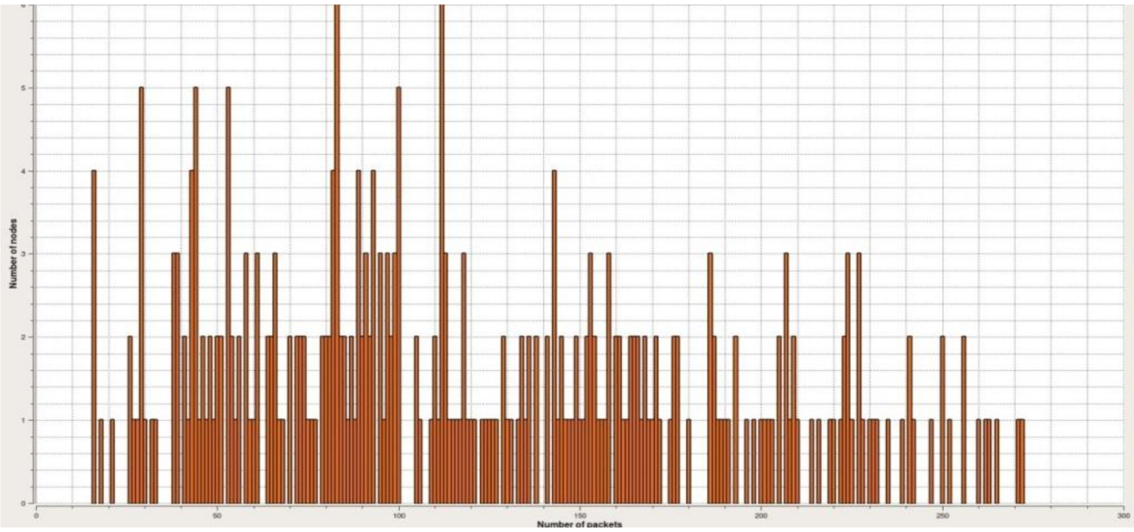


Fig.28 Packets per node for the (16, 8) RS code

This is the result for RS code. Of course, it balance the distribution of packets when compared with the former one, but we should notice that the max number of packets increase very fast. Almost eight times of replication. When considering the overhead of every packet, the total number is still unacceptable

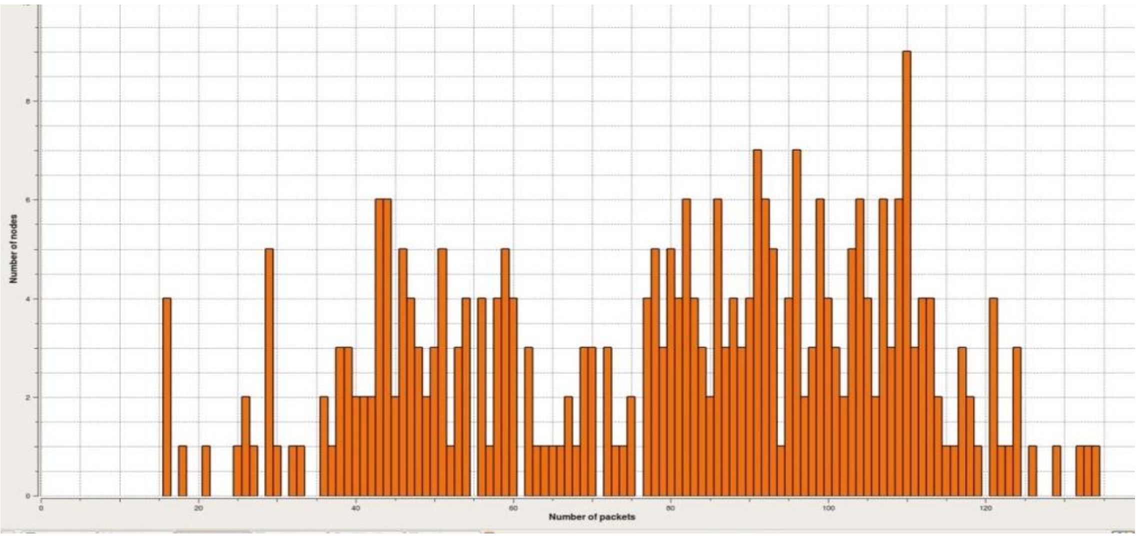


Fig.29 Packets per node for the (16, 8, 8) MFRC code

Here is the result for our MFRC scheme. We can easily notice that in our algorithm, the max packets a node deliver is only 140, but the number in reed solomon code is 300. This means the busiest node can save over 50% traffic overhead. And when compared with the traditional RS code, it reduced 34% packets at all.

5.2.2. Successful Repair Performance

Now we observe how much percent can our algorithm to do successful repair when we grow (n, k). The result is presented in Fig.29. We can make two observations. First, 3-way replication has the best repair performance followed by MFRC, while the RS-code offers the worst performance. Second, the repair performance of MFRC has a higher upward trend than RS-code. When (n, k) grows to (32, 28), RS-code just has the same successful repair performance as (8, 4, 8) MFRC schem. Furthermore, we can improve the repair performance by rising the (n, k), but it should be cautious due to the limits of nodes.

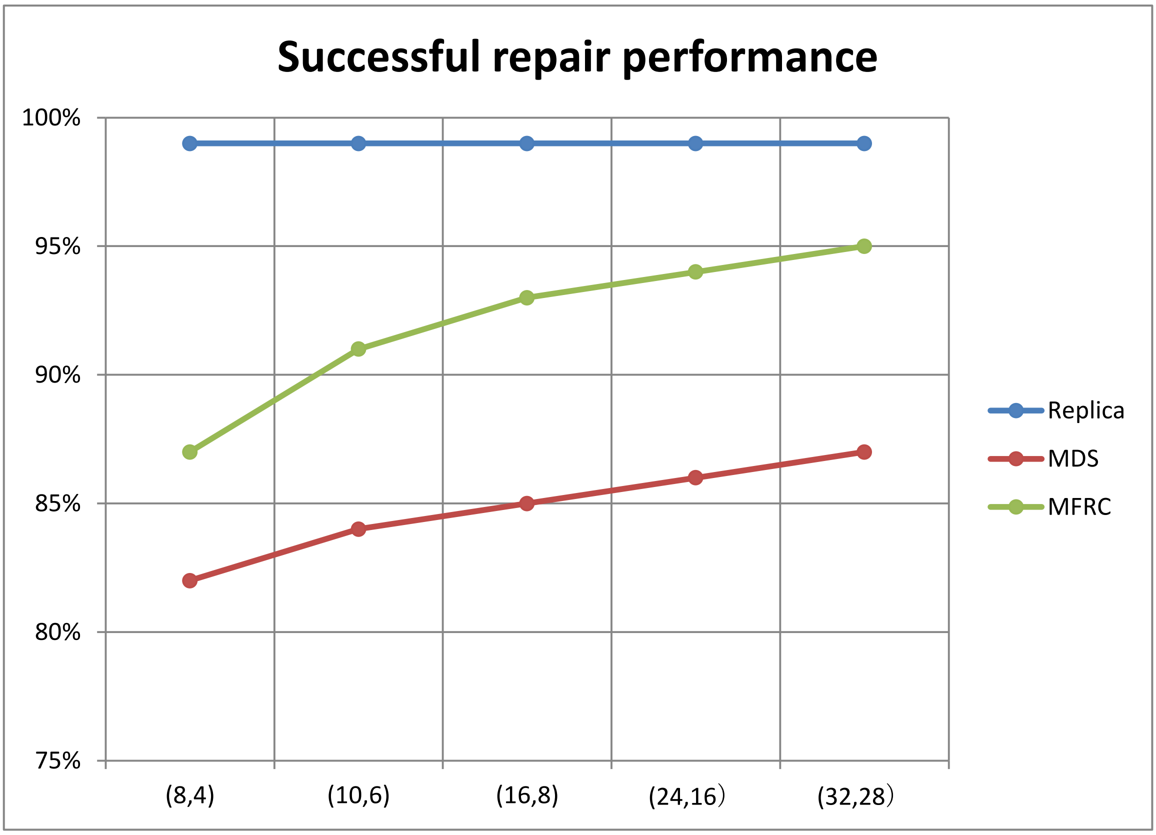


Fig.30 Successful repair performance

5.2.3. Storage Space Cost Performance

Now we observe how storage overhead varies when we grow (n, k). Obviously, high cost always results in high storage overhead. As 3- way replication is a popularly used approach, we use it as the base line for comparison and always set it as 1. The result is presented in Fig.31. When (n, k) is fixed, the normalized cost of both the RS-code and the MFRC decreases respectively. When (n, k) grows to (32, 28), the normalized cost of MFRC is 0.62 while that of RS-code is 0.4. In other words, (32, 28, 8) MFRC only need approximately 60% the storage of 3-way replication. MFRC still costs more storage space than MDS-codes, but it is acceptable because if we change larger values of r, but at the cost of slower repair.

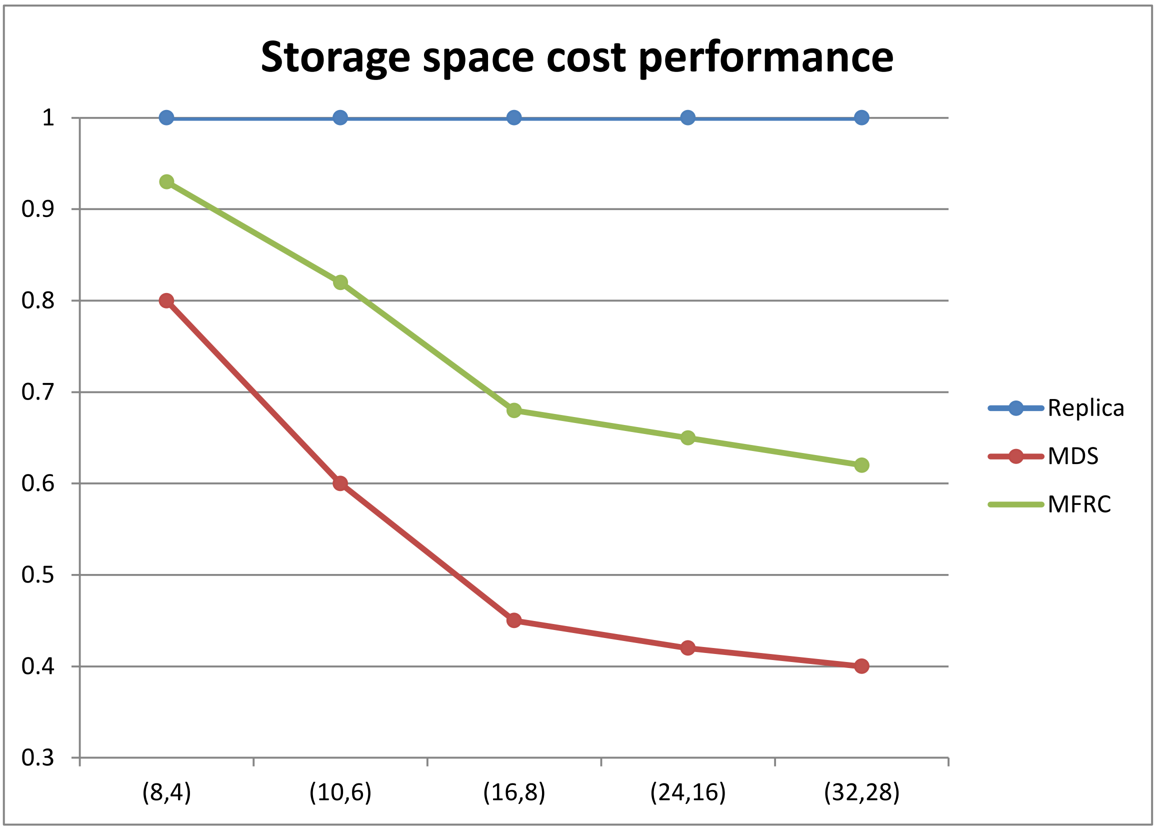


Fig.31 Storage space cost performance

As conclusion, we presented a comparison of the proposed codes with replication and Reed-Solomon codes using a storage simulator-NECO in this section. They deliver comparable performance to 3-way replication and significantly higher data reliability at a lower storage space cost.

5.2.4. Estimation for Large-Scale WSDSN

In this section, we focus on MFRC by compared its performance in severe size of network. We started from 30nodes, then 100, 300 nodes. Finally we will make the estimation for large-scale network, give a mathematical relationship for our scheme.

In Fig.33, 34, 35, we show the network topology in severe number of nodes which contains 30, 100 and 300 nodes respectively. We can find when the network size is small, our scheme can provide all the transmission is unobstructed and packages can be delivered in 100% without lost due to the failure of paths. However, when nodes keeping increasing to 300, several paths became malfunction just like shown in Fig.35, four nodes in the left-up and right-up became dead due to the failed paths. We can get the conclusion that when network size become 10,000 or 100,000 of magnitude, it is inevitable that some paths and nodes would die, especially in the edge of network, due to lack of neighbors to do the collaborated coding. If nodes can move, the situation is easy to fix because of the obvious reason that edge nodes will move into core-area with some probability. If nodes are stationary, we can expand the sensing radius to make sure more neighbor nodes involved to doing coding.

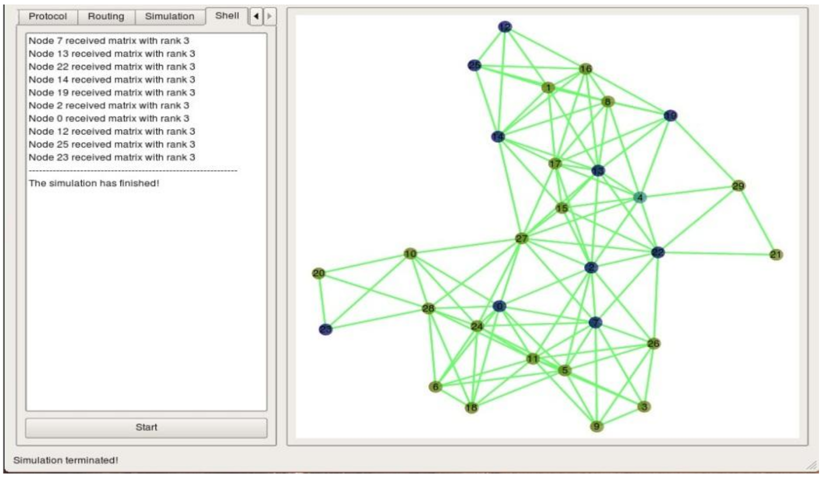
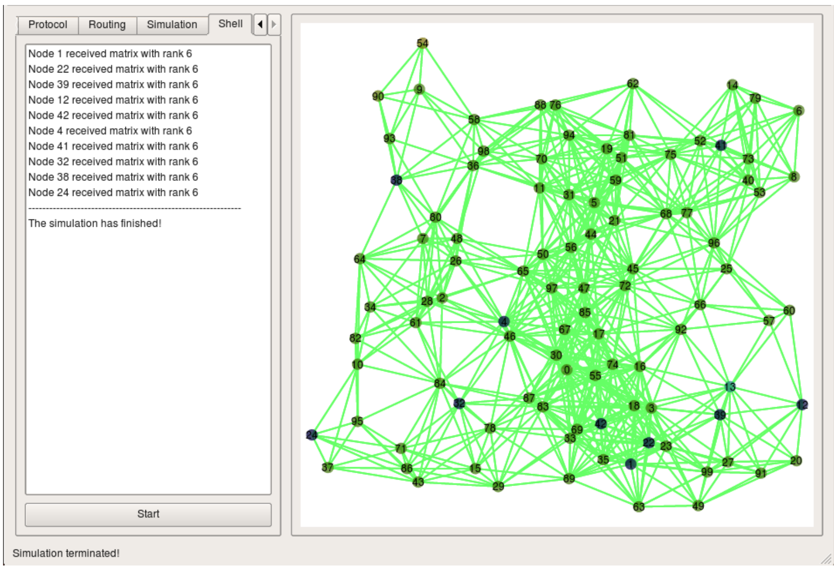


Fig.33 30 nodes network topology



5. SIMULATION RESULTS

Fig.34 100 nodes network topology

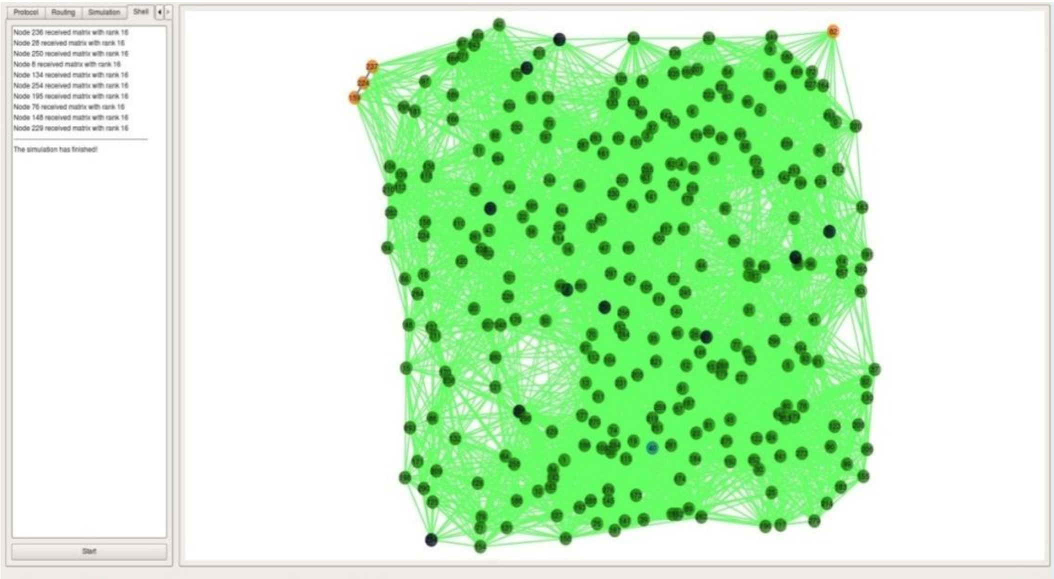


Fig.35 300 nodes network topology

Besides the failed paths, there is also a main change in parameters which is the time consumption performance. From Fig.36, basically, system use more time to doing coding operation and the consumption has the linear relationship with the change of (n, k). However, when network size becomes larger and larger, the time consumption increases by geometrically. It is a tough dilemma because when network size becomes 10,000 or 100,000 of magnitude, it maybe would take a long period of time to doing the collaborated coding. By improving the capability of individual nodes, this problem would be easily solved in the near future.

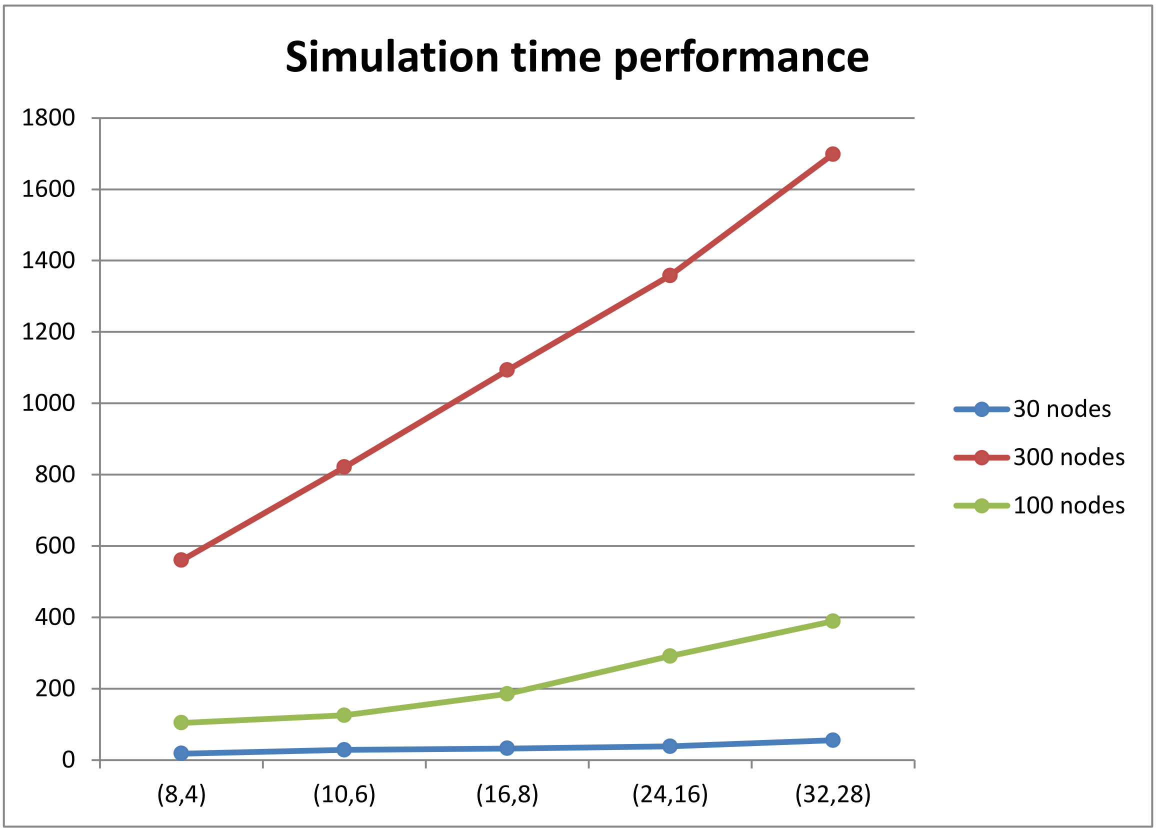


Fig.36 simulation time for different size network

5.3. Summary

In this chapter, we firstly compared our MFRC scheme with Replica and MDS. From the simulation result, our scheme can achieve very good storage space cost with good performance. As an example, MFRC reduced 34% total packets in the 300 nodes environment. It also approximately needs only 38% the storage of 3-way replication when in (32, 28, 8). Although our scheme cannot achieve the perfect repair performances like Replica, but it can save time consumption by choosing appropriate indicators while keeping security doing the coding operation. Then we compared the performance when our scheme working in different scale of network. Most performance indicators increased by linear relationship. Some edge paths might fail, the time consumption is also increasing by geometrically. It is also acceptable due to the analysis above. As conclusion, our scheme adds new feasible points in the tradeoff space of distributed storage codes. They deliver comparable performance to 3-way replication and significantly higher data reliability at a lower storage space cost and we get confident that it might be useful in the practical world.

6. Conclusions

6.1. Summary

In this thesis, we proposed a multi-loss flexible regenerating code recovery scheme using network coding for distributed storage wireless sensor network.

In the practical world, it is impossible to achieve the max-flow bound on the information transmission rate by the conventional forward routing. However, with the collaborated coding operations, random linear network coding can achieve the capacity region of the network with high probability. We applied this born in nature characteristic into the distributed storage system.

In order to analysis the traffic cost, by using network flow theory, we turn the distributed storage recovery problem into the network flow model. Different from the former researches, we introduced virtual source node to turn repeatedly recovery problem into simple multi independent nodes one-loss network model. Inspired by the several former existing researches, we presented our MFRC scheme for WSDSN and compare it with former algorithms to demonstrate its advantages. Our codes have the (n, k) reliability, have asymptotically optimal storage and are within a logarithmic factor from optimality in repair traffic. One significant benefit is that the number of nodes that need to be contacted for repair can be made a small constant, independent of n, k. At the same time, our scheme is also easy to implement.

From the simulation result, our scheme can achieve very good storage space cost with good performance. As an example, MFRC reduced 34% total packets in the 300 nodes environment. It also approximately needs only 38% the storage of 3-way replication when in (32, 28, 8). Although our scheme cannot achieve the perfect repair performances like Replica, but it can close the suboptimal performance by choosing appropriate indicators while keeping security doing the coding operation. Then we compared the performance when our scheme working in different scale of network. Most performance indicators increased by linear relationship. Some edge paths might fail, the time consumption is also increasing by geometrically. It is also acceptable due to the analysis above. As conclusion, our scheme adds new feasible points in the tradeoff space of distributed storage codes. They deliver comparable performance to 3-way replication and significantly higher data reliability at a lower storage space cost and we get confident that it might be useful in the practical world.

6.2. Future work

We think there are at least two important things worth trying as future work. First one, our model has some drawbacks such as we did not limit the bound for each edge independently. In practical world, the traffic for each edge is always different and restricted. However, it is sure that it makes the problem easier. Whether our scheme did its best performance is mainly related to the network model. For example, if we consider mutually cooperative recovery just like in Fig.36, it is sure that more traffic will be saved due to the transmission among the new nodes, but it is hard to get the mathematical analysis.

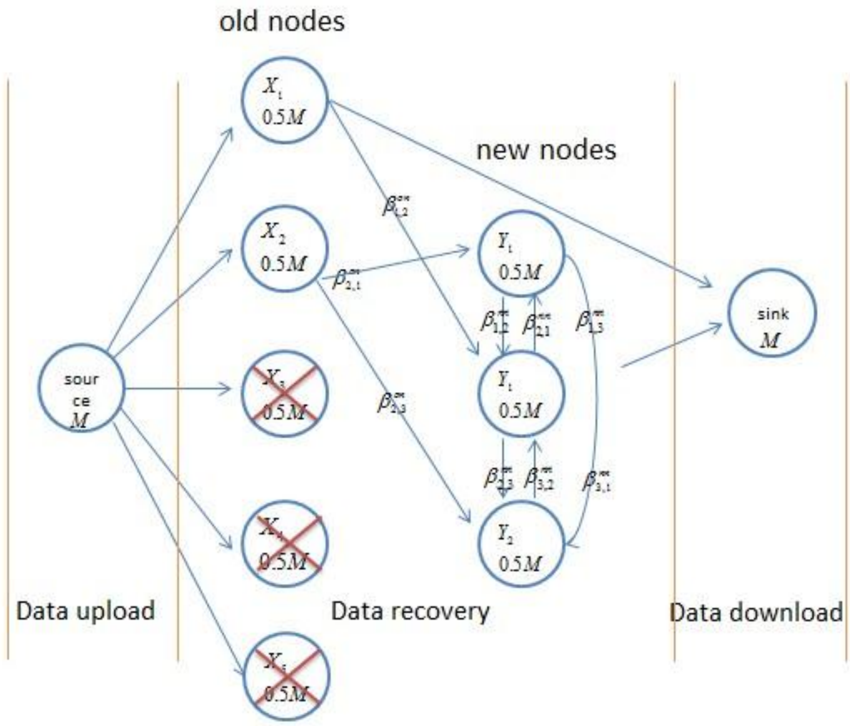


Fig.37 a mutually cooperative recovery model

The other big issue is lack of practical word experiment. Whether our scheme can perform well or not is still untested. For cloud system, there are lots of complex scheme to store the files. Maybe in the near future, there might be an appropriate environment for our scheme.

Publications

1. Li Chenchao, Iwai Masayuki, Sezaki Kaoru, "The Data Generation and Distribution using the Streaming Network Coding in WSN", in IEICE Technical Committee on Human Probes (HPB),Tottori,2012.2

2. Li Chenchao, Iwai Masayuki, Sezaki Kaoru, "A Simulator for Streaming Network Coding on Privacy Preservation WSN“, in IPSJ SIG MoBiLe computing and ubiquitous communications (MBL),Okinawa,2012.5

3. Li Chenchao, Iwai Masayuki, Sezaki Kaoru, " A Multi-Loss Flexible Regenerating Code Recovery Scheme For WSN“, in IEICE General Conference,Gifu,2013.3

References

[1] Jennifer Yick, Biswanath Mukherjee, Dipak Ghosal. "Wireless sensor network survey". http://wiki-castelldefels.cimne.upc.edu/images/2/20/Wireless_sensor_network_survey.pdf

[2] Jaydip Sen. "A Survey on Wireless Sensor Network Security". International Journal of Communication Networks and Information Security (IJCNIS), pages 55-78, 2009. http://arxiv.org/pdf/1011.1529.pdf

[3] John Paul Walters, Zhengqiang Liang, Weisong Shi, and Vipin Chaudhary. "Wireless Sensor Network Security: A Survey", http://www.eecis.udel.edu/~fei/reading/070426.wsn.security.survey.pdf

[4] R.Ahlswede, N. Cai, S. Y. R. Li and R. W. Yeung, "Network information flow," IEEE. Trans. Inform. Theory, 46(4), pp. 1204-1216, July 2000.

[5] S. Y. R. Li, R. W. Yeung and N. Cai, "Linear Network Coding," IEEE. Trans. Inform. Theory, 49(2), pp. 371-381, Feb. 2003.

[6] X. Xiao, L. M. Yang and B. X. Pu, "Multi-node Wireless Broadcasting Retransmission Scheme based on Network Coding," Computer Application, 28(4), pp. 849-852, Apr. 2008.

[7] Na LI. Nan Zhang and Sajal K.Das,Bhavani Thurais-ingham, "Privacy presenvation in wireless sensor net-works : A state-of-the-art survey," Ad Hoc Network 7(2009)1501-1514

[8] Christina Fragouli and Emina Soljanin, "Network Coding Fundamentals". http://books.google.co.jp/books?id=jFCAn8My7SYC&pg=PA12&lpg=PA12&dq=max-flow+min-cut+network+coding&source=bl&ots=kkE5xepS-X&sig=oz4CVB64sGVyU-4fWZFOsirneHE&hl=ja&sa=X&ei=B8_CUI7pMY-ImQWbxIHgBQ&redir_esc=y

[9] T. Ho, R. Koetter, M. Medard, D. R. Karger, and M. Effros, "The benefits of coding over routing in a randomized setting," in International Symposium on Information Theory (ISIT),2003.

[10] S. Katti, D. Katabi, W. Hu, H. S. Rahul, and M. M ́edard. "The Importance of Being Opportunistic: Practical Network coding for Wireless Environments". In 43rd Annual Allerton Conference on Communication, Control, and Computing, Allerton, 2005

[11] Philip A. Chou, Yunnan Wu, and Kamal Jain. "Practical network coding", Allerton Conference on Communication, Control, and Computing, October 2003,

[12] Tracey Ho, Muriel Médard, Ralf Koetter, David R. Karger, Michelle Effros, Jun Shi, and Ben Leong. "A Random Linear Network Coding Approach to Multicast". IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 52, NO. 10, OCTOBER 2006

[13] Liu Yunyi, Qin Tuanfa, Liu Yunyi, Qin Tuanfa, "Network Coding (NetCod) ", 2011 International Symposium, 25-27 July 2011.

REFERENCES

[14] Sachin Katti, HariharanRahul, WenjunHu, Dina Katabi, Muriel Medard, Jon Crowcroft , “ XORs in the Air: Practical Wireless Network Coding” SIGCOMM’06, September 11–15, 2006, Pisa, Italy

[15] Z. Guo, B.Wang, and J.-H. Cui, “ Efficient error recovery using network coding in underwater sensor networks”, Journal Ad Hoc Networks Volume 7 Issue 4, 791-802,June, 2009

[16] J. Y. Wang, G. Yang, X. H. Lin, Z. Q. He and J. R. Lin, “Continuous Data Collection in Wireless Sensor Networks through PNC and Distributed Storage,” Int. Conf. Wireless Communications, Networking and Mobile Computing, Shanghai, China, pp. 2568-2571, 21-25 Sept. 2007

[17] Y. N. Wu, P. A. Chou and S. Y. Kung, “Information exchange in wireless networks with network coding and physical-layer broadcast,” Int. Conf. on Information Sciences and Systems, The Johns Hopkins University, March 16-18, 2005.

[18] Alexandros G. Dimakis, Kannan Ramchandran, Yunnan Wu, Changho Suh, “A Survey on Network Codes for Distributed Storage”, IEEE Journal on Selected Areas in Communications 28(2): 277-288 (2010).

[19] Alexandros G. Dimakis, P. Brighten Godfrey, Martin J. Wainwright and Kannan Ramchandran, “The Benefits of Network Coding for Peer-to-Peer Storage Systems”, INFOCOM 2007, 6-12 May 2007.

[20] Yunnan Wu , Ros Dimakis , Kannan Ramchandran, “Deterministic regenerating codes for distributed storage”, allerton conference on control, computing, and communication, urbana-champaign, 2007.

[21] B.-G. Chun, F. Dabek, A. Haeberlen, E. Sit, H. Weatherspoon, M. F. Kaashoek, J. Kubiatowicz, and R. Morris. “Efficient replica maintenance for distributed storage systems”. In Proc. of NSDI ’06, 2006.

[22] R. Curtmola, O. Khan, R. Burns, and G. Ateniese. “MR-PDP: Multiple-replica provable data possession”. In Proc. of ICDCS, 2008.

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

[24] K. Bowers, A. Oprea, and A. Juels. “HAIL: A high-availability and integrity layer for cloud storage”. In Proc. of ACM CCS, 2009.

[25] A. G. Dimakis, B. Godfrey, M. J. Wainwright, and K. Ramchandran. “Network coding for distributed storage systems”. In INFOCOM, 2007.

[26] A. G. Dimakis, P. B. Godfrey, Y. Wu, M. O. Wainwright, and K. Ramchandran. “Network

coding for distributed storage systems”. IEEE Transactions on Information Theory, 2010.

[27] Dimitris S. Papailiopoulos, Jianqiang Luo, Alexandros G. Dimakis, Cheng Huang, and Jin Li, “Simple Regenerating Codes: Network Coding for Cloud Storage”. INFOCOM 2012

[28] Yuchong Hu , Henry C. H. Chen , Patrick P. C. Lee , Yang Tang, “NCCloud: Applying Network Coding for the Storage Repair in a Cloud-of-Clouds”, FAST 2012

[29] Alexander Thomasian, “Comment on “RAID5 Performance with Distributed Sparing”, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, VOL. 17, NO. 4, APRIL 2006

[30] Diogo Ferreira, Luisa Lima and Joao Barros, “NECO: NEtwork COding simulator”, http://www.dcc.fc.up.pt/~neco/Docs/report.pdf

[31] http://www.dcc.fc.up.pt/~neco/wiki/index.php/NECO_NEtwork_COding_Simulator_Documentation

[32] “The Network Simulator - ns-2,” http://www.isi.edu/nsnam/ns/.

[33] “The Network Simulator - ns-3,” http://www.nsnam.org.

[34] “Op-net Simulator,” http://www.opnet.com/.