Efficient Distributed Storage for Space Information Network Based on Fountain Codes and Probabilistic Broadcasting

Bo Kong[1], Gengxin Zhang[1], Wei Zhang[1] and Feihong Dong[1]
[1] College of Communication Engineering, PLA University of Science and Technology
Nanjing, China
[e-mail: kbvx_123@163.com, satlab@126.com, zev@msn.com, dfh_sinlab@hotmail.com]
*Corresponding author: Gengxin Zhang

Received December 3, 2015; revised February 29, 2016; revised April 6, 2016; accepted April 26, 2016; published June 30, 2016

Abstract

This article investigates the distributed data storage problem in the space information network (SIN) using distributed fountain codes. Since space nodes in the SIN are resource-limited, in order to reduce energy consumption while improving the storage reliability, an efficient distributed storage based on fountain codes and probabilistic broadcasting (DSFPB) strategy is proposed. In the proposed strategy, source packets are disseminated among the entire network according to probabilistic broadcasting (PBcast), and the final degree distribution is close to the desired robust soliton distribution (RSD), this is benefited from the appropriate packets encoding procedure of the proposed strategy. As presented by the analysis and simulations, the total cost of data dissemination is greatly reduced compared with existing representative strategies, while improving the decoding performance.

Keywords: Space information network, distributed data storage, probabilistic broadcasting, fountain codes, LT codes

This work is supported by National Natural Science Foundation of China under Grant No. (91338201, 91438109, 61571464 and 61401507).

http://dx.doi.org/10.3837/tiis.2016.06.009 **ISSN : 1976-7277**

1. Introduction

The space information network (SIN) is a new self-organizing infrastructure constituted by various information systems of space, land, air and sea. It is proposed to settle the problems that different space systems' built separately and the inconvenience in cooperation among various space missions [1-3]. Unlike traditional satellite networks, the network status of SIN is changing dynamically due to its distinguishing characteristics such as high component complexity and large scale [4]. In our recent research which was published in [5], we divided the SIN into a series of hierarchical autonomous system (AS) networks based on the property of space nodes. Each AS network can be modeled as a homogeneous network constituted by similar space nodes. In this way, the complex SIN is decoupled into quasi-static sub-networks, which makes the control easier to carry out.

In recent years, space information is increasing substantially. According to the satellite database of union of concerned scientists (UCS), the amount of space information in 2030 will be 66 times than that in 2010 [6]. However, space nodes in the SIN are bandwidth-limited and energy-limited; these limitations can be jointly described as resource-limited. Sometimes they may even become invisible for a long time due to the shadow effect. Therefore, how to prolong the lifetime of space information in SIN is challenging. Distributed storage is an effective solution, and it adds redundancy into storage systems by combining network technologies [7]. Replication and coding are two main redundancy methods. Replication is the simplest form of redundancy, but it consumes too much storage overheads. As the development of replication, coding techniques has been widely used in many practical storage systems [8]. In order to reduce the repair bandwidth, [9-11] proposed the regenerating codes by introducing network coding into storage systems. Fountain codes [12] is an important class of rateless codes with low encoding and decoding complexity, which is suitable for distributed data storage in wireless networks.

We consider the distributed data storage problems in the SIN, and the main objective is to improve the reliability of the AS network. To the best of our knowledge, the distributed storage of the SIN has not been investigated yet. Since space nodes in the SIN are resource-limited, it is significant if the energy consumption of the data dissemination process is also reduced. In this article, redundant packets are generated based on Luby transform (LT) codes [13] and source packets are disseminated among storage nodes according to probabilistic broadcasting (PBcast) manner. We denote our strategy by distributed storage based on fountain codes and probabilistic broadcasting (DSFPB) strategy. DSFPB is the first strategy using PBcast to minimize the cost of data dissemination. The final degree distribution is close to the desired robust solition distribution (RSD), which is benefited from the encoding procedure. The DSFPB strategy does not require any routing table and it is scalable to any network topology. The energy consumption and decoding performance are analyzed. Simulation results validate that the DSFPB strategy greatly reduces the data dissemination cost compared with existing representative strategies, while improving the decoding performance.

The remainder of this article is organized as follows. In Section 2, we review some related work about coding redundancy technologies in distributed storage systems. In Section 3, we define the network model and provide some preliminaries about fountain codes and PBcast. In Section 4, we propose the DSFPB strategy for increasing data reliability in AS network. The data dissemination cost and successful decoding probability are analyzed in Section 5. In

Section 6, simulation results and discussion are presented. Finally, we make conclusion in Section 7.

2. Related Work

Since the distributed storage in the SIN has not been investigated, we present some related work about the distributed data storage with coding redundancy technologies in wireless networks and satellite networks.

The coding redundancy technologies has been widely studied in the distributed storage of wireless networks, which achieves better storage efficiency for the same redundancy order of replication. We divide an initial file into k source pieces with equal size, and encode them into n encoded pieces using a specific coding algorithm. These encoded pieces are disseminated among storage nodes and original file can be recovered from any k (or slightly larger than k) encoded pieces. For instance, Reed-Solomon (RS) codes and Low-Density Parity Check (LDPC) codes based distributed storage schemes were proposed in [14] and [15], respectively. Fountain codes provide a potential way to reduce the decoding complexity for its well-known rateless property. Dimakis et al. [16] proposed a randomized scheme for networked storage based on decentralized fountain codes, where the greedy routing algorithm is used to construct the encoded packets. Kamra et al. [17] proposed a novel growth codes which maximizes the partial decoding probability at decoder, the decoding process is similar to the traditional fountain codes. LT codes is the first practical fountain codes, which can be used in a decentralized way for distributed data storage in wireless networks.

Existing works about distributed storage based on LT codes mainly focus on two aspects [18-22]. The first is how to disseminate source packets over network. The deterministic routing table cannot be adopted since dynamic topology and limited resource of network. Lin et al. [18] proposed the exact decentralized fountain codes (EDFC) strategy to increase the lifetime of data in wireless networks. Random walk [23] was used in EDFC strategy to disseminate source packets among the entire network. The encoding process of EDFC strategy was analogous to the conventional manner. The main contribution of [18] is the usage of random walk. Random walk does not need any location information and can be used in arbitrary topology. Inspired by [18], random walk was used in most existing distributed storage strategies for data dissemination. Ye et al. [19] proposed the LT-codes based distributed coding (LTDC) strategy for efficient distributed storage, where a variant of Metropolis-Hastings (MH) algorithm was used to reduce the mixing time of random walk. However, in the dissemination process, each node should calculate the transition probability of neighbors, which consumes additional energy. Aly et al. [20-21] proposed the LT-codes based distributed storage (LTCDS) strategy employing simple random walk to disseminate source packets. Each node uniformly selected one of neighbors as its next "walk". However, most existing strategies ignore the inherent broadcast property of wireless channels. As shown in [23], the data dissemination cost of distributed storage includes both data transmission and data reception. Therefore, the data dissemination cost can be reduced if the broadcast property is utilized. Liang et al. [22] proposed the LT-codes based distributed strategy for improving data persistence (LTSIDP). The overhearing was used to get the information whether a particular packet has been received. However, the improvement of energy efficiency was not considered. In this article, we use the PBcast manner to disseminate source packets.

The process of forming encoded packets is another important issue. Encoding procedure in [18] and [19] performs only after all random walks are finished. Each node randomly

exclusive-or (XOR) a few source packets based on its code degree. Therefore, each node should temporarily manage a cache buffer to save all received source packets, which induces excessive energy consumption. Unlike [18] and [19], encoding procedure in [20-22] is performed on the fly and it requires less cache buffer. However, encoding procedure is performed only at source packets' first visit, regardless of whether nodes have XORed enough packets. As a result, the final degree distribution was not matched and the decoding performance was poor when queries few storage nodes. In addition, to run the "on the fly" encoding procedure, each source packet should visit all storage nodes at least once. This is done by setting a threshold to each random walk according to the cover time [25]. During the data dissemination process, each source packet would visit the same storage node several times. If each storage node runs the encoding procedure at source packets' every visit, unless it has been XORed or the code degree has fulfilled, the final degree distribution will be close to the desired degree distribution with high probability (w.h.p.).

For distributed storage in satellite networks, Suto et al. [26] proposed a hybrid system which integrates satellite networks with a control center. They mainly focus on the information distribution problems and replication is used for data transmission. However, the reliability of distributed data storage is not considered.

3. Preliminaries

In this section, we firstly provide the AS network model of SIN, then we review the basic knowledge about LT codes and PBcast.

3.1 Network Model

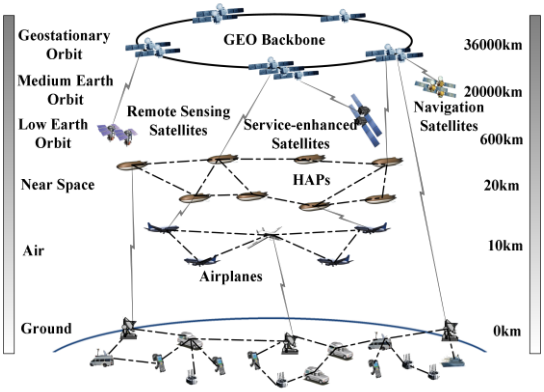

Fig. 1. The architecture of SIN.

The SIN is a space-based information infrastructure that contains different kinds of space nodes. As shown in **Fig. 1**, those nodes are located in different altitude of space orbits, and work within various circumstances with either dynamic (e.g., low earth orbit satellites) or static (e.g., grid topology on the earth) status. The unified management approach is not efficient for SIN since it induces lots of control messages, which consume excessive energy and network capacities. Therefore, as shown in **Fig. 2**, we decouple the SIN into a series of AS networks, each AS network contains a collection of similar space nodes. Individual AS network can be further divided into sub-AS networks, and independent manage strategy is

adopted in each AS (sub-AS) network. This will decouple the complex SIN into quasi-static subnets, thus unified control is easier.

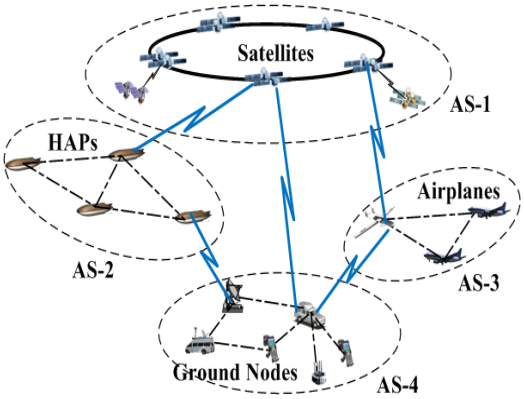

Fig. 2. AS network model of SIN.

In this article, we mainly focus on the distributed data storage problem in an AS network. As shown in **Fig. 3**, the AS network consists n similar nodes distributed uniformly at random with omnidirectional antennas. All nodes have an identical maximal transmission radius R. Two nodes are called neighbors and can communicate directly if and only if their distance is less than R. In order to remain the connectivity of AS network, R should satisfy $R^2 \geq bS \log n / n$, where S is the area of AS network and b is a positive constant [27]. Then the AS network can be modeled as an undirected random geometric graph (RGG) $G(n, R)$ with the maximal transmission radius R.

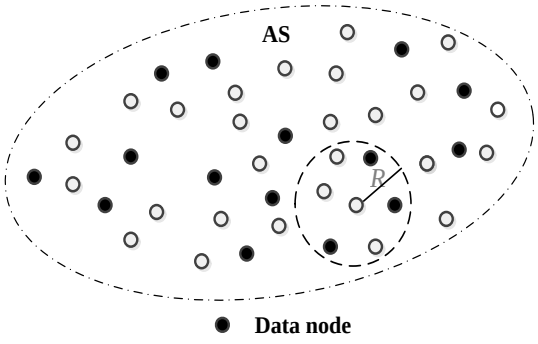

Fig. 3. Example of AS network.

Each node is assigned a unique ID based on its property, e.g., MAC address. For simplify and fair comparison with other strategies, all n nodes has a limited memory to save only one encoded packet and served as *storage nodes*. We assume that among n nodes, there are k nodes generate the original data periodically, which are referred to as *data nodes*. Data nodes are randomly chosen from the AS network and the number is related to the network scale, i.e., $k=\Theta(n)$. Then, we define the node degree of AS network.

Definition 1: Node Degree. Denote by $N(u)$ the set of neighbors of node u. The number of neighbors is defined as the node degree of node u, and is denoted by $d_n(u)$, i.e.,

$d_n(u) = |N(u)|$. The average node degree of all nodes in AS network is called the *density* of network.

3.2 Fountain codes and LT codes

Fountain codes are a class of rateless codes which can potentially generate unlimited encoded packets without a pre-designed coding rate. The original information can be recovered w.h.p. as long as the decoder receives enough encoded packets [13]. LT codes is the first practical fountain codes with low encoding and decoding complexities, which has been widely used in wireless multimedia, satellite broadcast and many other fields. The key property of LT codes is the online encoding process, and the belief propagation (BP) algorithm is used for efficient decoding.

Definition 2: Code Degree. In LT codes, each encoded (output) packet is obtained by XORing a few source packets which are chosen randomly and independently. The number of XORed source packets is defined as the code degree and it is sampled from the degree distribution function.

The key factor of affecting the performance of LT codes is the degree distribution, and the Robust Soliton Distribution (RSD) is considered as the standard distribution of LT codes. The RSD is defined as follows.

$$\Omega_{RSD}(d) = \frac{\rho(d) + \tau(d)}{\sum_{d=1}^{k} (\rho(d) + \tau(d))}, \ d = 1,...,k \tag{1}$$

where $\rho(d)$ denotes the degree distribution under ideal condition and it is defined as

$$\rho(d) = \begin{cases} 1/k, & d = 1 \\ 1/(k(k-1)), & 2 \le d \le k \end{cases} \tag{2}$$

Let $R = c \cdot \ln(k/\delta)\sqrt{k}$ for some constant $c > 0$ and

$$\tau(d) = \begin{cases} R/dk, & 1 \le d \le (k/R - 1) \\ R\ln(R/\delta)/k, & d = k/R \\ 0, & \text{others} \end{cases} \tag{3}$$

where $0 < \delta < 1$ is the probability of decoding failure. For LT codes with RSD, k source packets can be recovered from any $k + O(\sqrt{k}\ln^2(k/\delta))$ encoded packets with probability $1-\delta$. Both the encoding complexity and decoding complexity is $O(k\ln(k/\delta))$.

3.3 Probabilistic Broadcasting: PBcast

As presented above, random walks are widely used in most existing distributed storage strategies to disseminating source packets. To ensure the overall coverage, the length of random walk is set to the cover time.

Definition 3: Cover Time. Given an RGG $G(n,R)$, the cover time of a random walk is defined as the expected length that enable a source packet visiting all storage nodes in AS network at least once, and denote by T_c. As in [25], w.h.p., T_c is bounded by

$$T_c = Cn\ln n \tag{4}$$

where C is a positive constant. As a result, tens of thousands steps (of the order of thousand times than the number of storage nodes) are needed in the data dissemination process, which necessitates high energy consumptions. If the inherent broadcast property of wireless channels is well utilized, data dissemination cost can be largely reduced.

Flooding is the simplest form of broadcasting. In the flooding process, each node simply forwards (rebroadcasts) the packets that it receives for the first time. But the major drawback of flooding is the broadcast storm [28], which consumes excessive resource. PBcast is a reliable and energy-efficient variation of flooding in wireless networks. In the PBcast method, each node forwards the packets that it receives for the first time with probability p, where p is a positive constant. As in [29], there exists a threshold p^{th} such that almost all storage nodes would receive a particular source packet w.h.p. if and only if $p > p^{th}$. Consider a network with n nodes randomly distributed in the region S, p^{th} can be calculated as follows.

$$p^{th} \approx 1.44S / nR^2 \tag{5}$$

where R is the maximal transmission radius of network.

4. Proposed DSFPB Strategy

In this section, we present our proposed DSFPB strategy for distributed data storage in the AS network. We firstly describe the strategy generally in Section 4.1, and then the encoding procedure is presented in Section 4.2. Finally, we discuss the selection of parameters b and p in Section 4.3.

4.1 Description of the DSFPB Strategy

The main objective of the DSFPB strategy is to have all n storage nodes in the AS network store an LT encoded packet corresponding to the k source packets in a decentralized manner. If the number of source packets that each node XORed is close to its code degree, the final degree distribution will be matched with the desired RSD w.h.p. The code degree is generated according to $\Omega_{RSD}(d)$ and the XORed packets are randomly chosen from k source packets. To approach this target, it is crucial that each source packet should visit all storage nodes in the AS network at least once, and each node XORs a visited packet with probability d_c/k. This is done by the choice of forwarding probability p of PBcast and will be discussed in Section 4.3. The torus convention [30] is used to avoid the edge effect, i.e., storage node that far away from center has few neighbors than the density of AS network.

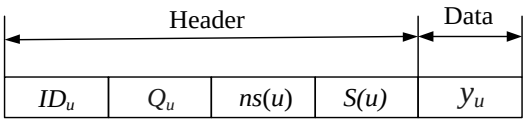

Fig. 4. Storage packet structure of node u.

We assume that the transmission in the AS network is synchronized and slotted. At the beginning, i.e., the initialization phase, each data node $s, s = 1,...,k$ generates a source data x_s, and adds its ID to packet head to form its initial source packet P_s. Then, each storage node $u, u = 1,...,n$ generate its code degree $d_c(u)$ according to the degree distribution Ω_{RSD}. After

that, node u generates its initial storage packet of the form $\left(ID_u, Q_u, ns(u), S(u), y_u\right)$ as shown in **Fig. 4**. The explanation of the packet structure is as follows. 1) ID_u: the unique ID of storage node u to break ties. 2) Q_u: the queue records the set of visited source packets without iteration. 3) $ns(u)$: the number of XORed source packets in node u. 4) $S(u)$: the set of XORed source packets in node u, i.e., $|S(u)| = ns(u)$. 5) y_u: the storage data of node u.

Then, k data nodes disseminate their source packets to the AS network according to PBcast mechanism. In the second phase, i.e., data nodes broadcasting phase, k data nodes broadcast their source packets with probability 1. Each reception node $u, u = 1,...,n$ will run the encoding procedure (will be discussed in Section 4.2) and update its own storage packet with probability $d_c(u)/k$. Subsequently, in the third phase, i.e., intermediate nodes forwarding phase, the above reception nodes forward (rebroadcast) their received source packets with probability p. Again, each node $u, u = 1,...,n$ that receives a source packet will run the same encoding procedure and update its storage packet with probability $d_c(u)/k$. This phase terminates until no node forwards source packets any more. In this article, we do not consider the packet loss during the data dissemination process because it can be handled at the lower layer. In addition, due to the small value of p (will be discussed in Section 4.3), the date dissemination process ends after the forwarding procedure repeats a few times.

In this article, we use the term "visit" refers to the packets either received or overheard by a storage node u, and node u choose itself as an intermediate node with probability p only when it receives the packet for the first time. The reason to utilize the overhearing property of wireless communications is to enhance the number of encoding procedures of each storage node. As a result, the probability that each node fulfills its code degree is increased.

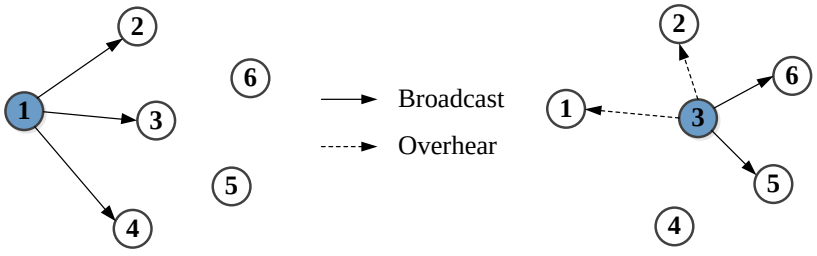

Fig. 5. The overhearing property increases the number of encoding procedures. (a) Broadcasting phase. (b) Forwarding phase.

Consider an AS network with 6 storage nodes as shown in **Fig. 5**, node 1 is the date node and broadcasts its source packet P_1 to node 2, 3 and 4 in the data node broadcasting phase. After run the encoding procedure, these three nodes will choose themselves independently with probability p as the intermediate node to rebroadcast packet P_1. Assume that node 3 is chosen to forward packet P_1 to node 5 and 6 in the third phase. Because node 1 and 2 are also within node 3's transmission radius, they will overhear P_1 and run the encoding procedure if P_1 has not be XORed previously. To run the repetitive encoding procedure, P_1 can be acquired in the queue Q_1 (Q_2), thus additional energy consumption is not needed any more. To save the

energy consumption of overhearing, we assume that the energy conservation mechanism (e.g. APPL [22]) is applied in the MAC layer.

Finally, after all PBcast are terminated, each source packet has visited the entire AS network at least once. After storage node $u, u=1,...,n$ runs the encoding procedure for all source packets with probability, in the storage phase, it will store the current packet Y_u temporarily. When a user node (either fixed or mobile) needs the original data, it will query any $k(1+\varepsilon)$ storage nodes and collect their encoded packets, where $\varepsilon > 0$ is the decoding overhead. The BP decoding algorithm can be used to decode the original data efficiently.

The pseudo-code of the DSFPB strategy is shown in Algorithm 1, which is constituted by four phases: the initialization phase, the data nodes broadcasting phase, the intermediate nodes forwarding phase and the storage phase.

Algorithm 1 *: DSFPB Strategy*

Input:
source packets $x_s, s=1,...,k$, positive constant p.
Output:
storage packets $Y_u, u=1,...,n$.
Begin:
/*Phase I: Initialization Phase*/
for all data nodes $s, s=1,...,k$ **do**
$P_s=[ID_s, x_s]$;
end
for all storage nodes $u, u=1,...,n$ **do**
$d_c(u) \leftarrow \Omega_{\mathrm{RSD}}(d)$;
$Y_u=[ID_u, Q_u, ns(u), S_u, y_u]$;
$Q_u, S_u, y_u=[]$;
end
/* Phase II: Date nodes Broadcasting Phase*/
k data nodes broadcast source packets with probability 1;
for $s=1$ **to** $s=k$ **do**
for each node $u, u=1,...,n$ receives P_s **do**
$Q_u=Q_u \cup P_s$;
run the encoding procedure (Algorithm 2);
end
end
/* Phase III: Intermediate Nodes Forwarding Phase*/
The reception nodes in Phase II forward the visited source packet with probability p ;
while (There exists a node s that forward source packet P_s) **do**
for each node $u, u=1,...,n$ receives/overhears P_s **do**
if P_s visits node u for the first time **then**
$Q_u=Q_u \cup P_s$;
end
run the encoding procedure (Algorithm 2);
end
The reception storage nodes froward visited packet with probality p ;
end
/* Phase IV: Storage Phase*/
for all storage nodes $u, u=1,...,n$ **do**
store Y_u ;
end

Return Y_u

4.2 Encoding Procedure

Once a source packet visits the storage nodes of AS network, it will run the encoding procedure. Here we discuss the process of encoding procedure, i.e., how in DSFPB strategy,

each storage node made decisions about the visited source packets and XORed it with its current data. The pseudo-code of the encoding procedure and the subsequent XORing procedure are described in Algorithm 2 and Algorithm 3, respectively. The main objective of the encoding procedure in DSFPB strategy is to increase the successful encoding probability, i.e., the probability that each storage node fulfills its code degree.

Algorithm 2 *: Encoding procedure*

Input:
visited packets $P_s, s=1,...,k$ for node u
Output:
storage packet Y_u
Begin:

for $s=1$ **to** $s=k$ **do**
  if P_s is the first visited packet **then**
    temporary store P_s with probability 1;
  elseif P_s is the second visited packet **then**
    run the XORing procedure (Algorithm 3) for the first visited packet;
    run the XORing procedure (Algorithm 3) for the second visited packet;
  else
    run the XORing procedure (Algorithm 3) for P_s ;
  end
end
Return Y_u

Algorithm 3 *: XORing procedure*

Input:
visited packets $P_s, s=1,...,k$ for node u
Output:
storage packet Y_u
Begin:

if $ns(u) < d_c(u)$ **then**
  if $P_s \notin S(u)$ **then**
    coin=rand(1);
    if coin $\le d_c(u)/k$ **then**
      $S_u = S_u \cup s$;
      $y_u = y_u \oplus x_s$;
      $ns(u) = ns(u)+1$;
    end
  end
else
  continue;
end
Return Y_u

Once a source packet $P_s, s=1,...,k$ visits the storage node u, it will run the encoding procedure (Algorithm 2). If it is the first source packet that visits node u, this packet will be temporarily stored in its buffer with probability 1. After node u receives (or overhears) the second source packet (whether the same with the first source packet or not), it will run the XORing procedure (Algorithm 3) for the first visited packet. In this way it runs a Bernoulli trial and with probability $d_c(u)/k$ accepts the source packet. The Bernoulli trial is explained as follows. Each storage node u randomly selects a positive constant $coin$ ($0 < coin < 1$), if $coin \le d_c(u)/k$, storage node u will accept the visited packet and XOR it with its current data y_u. Otherwise node u will discard the first packet and run the encoding procedure for the

second visited source packet. For other visited source packets, storage node u will make its decision as follows.

When a source packet $P_s, s=1,...,k$ visits the storage node u, it will check the value of $ns(u)$. If $ns(u) < d_c(u)$ and $P_s \notin S(u)$, i.e., the number of source packets that node u has XORed is less than its code degree, meanwhile the packet P_s has not been XORed. Then storage node u will run the encoding procedure and XOR P_s with its current data with probability $d_c(u)/k$. This is done to ensure that the number of XORed source packets does not exceed the code degree. However, in [20-22], storage nodes only run the encoding procedure at source packet's first visit. This process still continues even after the node has fulfilled its code degree. As a result, the code degree might not be fulfilled since storage nodes with low $d_c(u)$ may XOR less source packets than expected, and vice versa. We can conclude that the final degree distribution of the proposed DSFPB strategy is much close to the desired RSD, because each source packet runs the encoding procedure at its every visit. In the DSFPB strategy, we assume that each data node takes its own source packet as the first visited packet. Therefore, they will run the same encoding procedure for other received source packets.

4.3 Selection of the Parameters b and p

Section 3.1 shows that to ensure the connectivity of AS network, the maximal transmission radius R should satisfy $R^2 \geq bS\log n/n$, where b is a positive constant. For a given graph $G(n,R)$, both the density (average node degree) and the total number of edges are affected by the radius R. Hence, a well-chosen parameter b will reduce the energy consumption while remaining the network connectivity.

In the intermediate nodes forwarding phase of DSFPB strategy, storage nodes that receive a source packet for the first time will forward it with probability p. To ensure the overall coverage, each source packet should visit all storage nodes in the AS network at least once. Note that $p > p^{th}$ is the necessary condition that most storage nodes are visited by a particular source packet. However, as we will show in Section 5, an excessive forwarding probability p largely increases the number of transmissions and receptions among the data dissemination process. Therefore, the forwarding probability p is required to close the threshold p^{th} while ensuring the reliability of PBcast, i.e., almost all storage nodes receive each source packet at least once.

We investigate the appropriate values of parameter b and p through simulations. n nodes are randomly deployed in the region $S = 2000 \times 2000\text{km}^2$, where $n = 100$ and 300 respectively. According to (5), the threshold of forwarding probability is $p^{th} \approx 0.25$. **Fig. 6** shows the average fraction of storage nodes (denote by F) in the AS network that receive a particular source packet versus the forwarding probability p after the PBcast process is finished. The total number of transmissions N_t and receptions N_r to disseminate a source packet when $n = 100$ are shown in **Fig. 7(a)** and **Fig. 7(b)**, respectively. We omit the case of $n = 300$ because the results are similar.

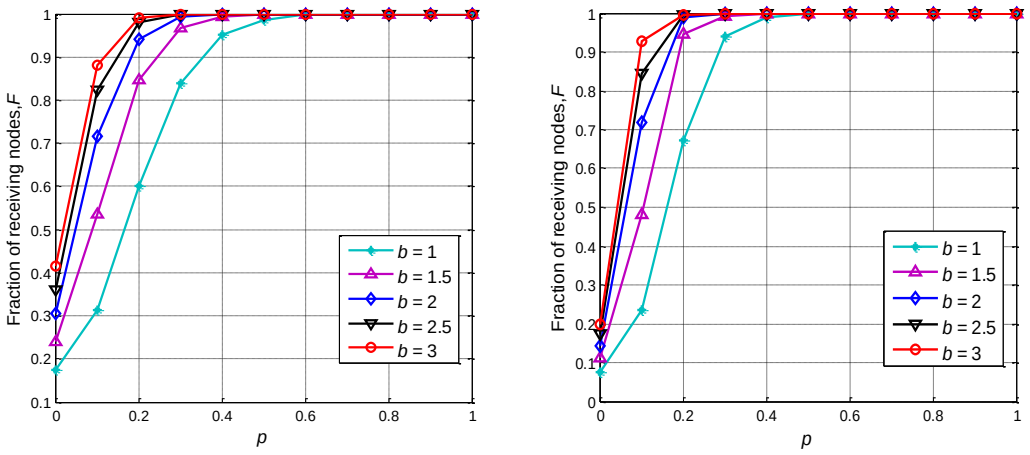

Fig. 6. The fraction of nodes receiving a source packet (F) versus p. (a) $n=100$. (b) $n=300$.

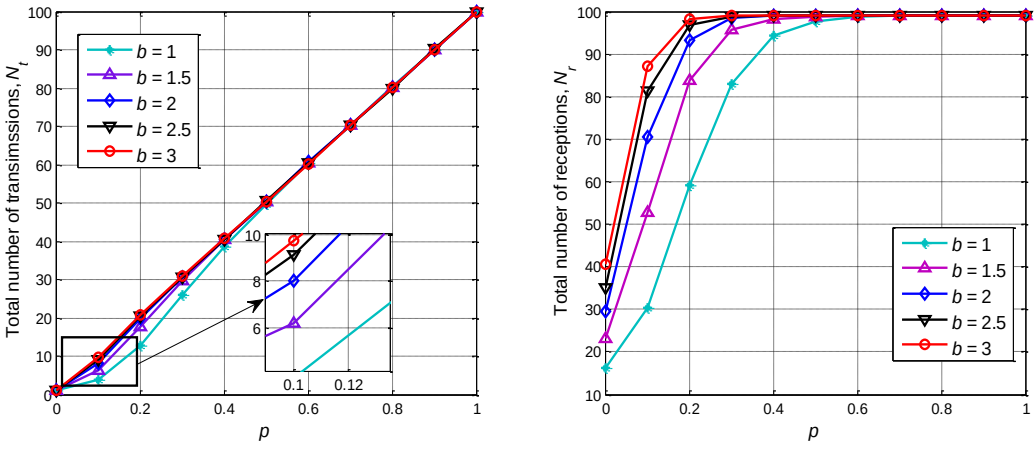

Fig. 7. Total number of transmissions N_t and receptions N_r versus p ($n=100$). (a) N_t. (b) N_r.

Fig. 6 shows that when $p=p^{th}\approx 0.25$, most storage nodes ($F>0.8$) received the particular source packet. The average fraction F increases as the parameter p get larger and converges to 1 when p is large enough. Therefore, we can select a large p to ensure the reliability of PBcast. However, as shown in **Fig. 7**, the number of transmissions N_t almost linearly increases with the parameter p, while the number of receptions N_r increases as p get larger and converges after $F=1$. For a given value of b, there exists an corresponding optimal probability p to ensure the reliability of PBcast. For example, when $n=100$, we should choose $p=0.6$ when $b=1$, while only $p=0.3$ can meet our goal when $b=2$ as shown in **Fig. 6(a)**.

Table 1 shows the optimal configuration (b, p) and the corresponding number of transmissions when $F=1$ and $n=100$. Note that we omit the number of receptions since the results are equal when $F=1$. The average node degree of these cases is also illustrated. It is

evident that $p=0.3$ is the minimum value to ensure the reliability of PBcast. The number of transmissions is not further reduced when $b \geq 2$, while the average node degree increases with the parameter b. To remain the network connectivity while reducing the energy cost, we set $b=2$ and $p=0.3$ in the following simulations.

Table 1. Average node degree versus optimal configuration of parameters in the AS network.

(b, p)	N_t	**Average node degree**
(1,0.6)	60	16.3
(1.5,0.5)	50	23.2
(2,0.3)	**30**	**29.5**
(2.5,0.3)	30	35.1
(3,0.3)	30	40.4

5. Dissemination Cost and Successful Decoding Probability Analysis

Two metrics are chosen to evaluate the tradeoff between the decoding performance and the energy efficiency of the proposed DSFPB strategy, i.e., the probability of successful decoding and the data dissemination cost, respectively. In this section, we firstly analyze the data dissemination cost, and then investigate the probability of successful decoding.

5.1 Data Dissemination Cost

Data dissemination includes both data transmission and data reception, and it is the most energy consuming process of the DSFPB strategy. Note that data transmission and data reception consumes almost equal energy in wireless communications [23]. Therefore, the data dissemination cost is proportional to the total number of data transmissions and data receptions. We firstly investigate the expression for data receptions. During the encoding process, all k source packets should visit the entire AS network at least once. It means that all storage nodes except k data nodes would receive them during the dissemination process. Hence, the total number of receptions N_r is

$$N_r = k(n-1) \tag{6}$$

In phase II, k data nodes broadcast their source packets with probability 1. In addition, in phase III, storage nodes that receive a source packet for the first time would forward it to its neighbors with probability p. Hence, the total number of transmissions N_t can be expressed as follows.

$$\begin{aligned} N_t &= N_t^{II} + N_t^{III} \\ &= k + k(n-1)p \end{aligned} \tag{7}$$

where N_t^{II} and N_t^{III} denote the number of transmissions in phase II and phase III, respectively.

Fig. 7 has shown the total number of required transmissions and receptions to disseminating a particular source packet, which is coincided with our analysis. If we assume that each data transmission and reception consumes unit of energy, the data dissemination cost (denote by

N_{tot}) of the DSFPB strategy can be expressed as the total number of data transmissions and data receptions, i.e.,

$$\begin{aligned} N_{tot} &= N_t + N_r \\ &= k + k(n-1)p + k(n-1) \\ &= k + k(n-1)(1+p) \end{aligned} \tag{8}$$

5.2 Successful Decoding Probability

Here we investigate the successful decoding probability of the proposed DSFPB strategy. Note that the performance of fountain codes is affected by the degree distribution, and the RSD is considered as the standard degree distribution of LT codes. In particular, if each encoded packet fulfills its code degree which is sampled from Ω_{RSD}, original k source packets can be successfully recovered from any $k + O(\sqrt{k}\ln^2(k/\delta))$ encoded packets with probability $1-\delta$.

We assume that the source packet s visits storage node u $c_s(u)$ times during the encoding process of the DSFPB strategy, the probability that source packet s would not be XORed by encoded packet Y_u is

$$P_s^R(u) = \prod_{i=1}^{c_s(u)} (1 - \frac{d_c(u)}{k} \times \mathrm{sgn}(d_c(u) - ns_i(u))) \tag{9}$$

where $ns_i(u)$ is the number of source packets that storage node u has XORed at packet s's ith visit, $\mathrm{sgn}(x)$ is the Sign function and it is defined as follows.

$$\mathrm{sgn}(x) = \begin{cases} 1, & x > 0 \\ 0, & x \le 0 \end{cases} \tag{10}$$

The purpose of Sign function is explained as follows. When a storage node has XORed enough source packets according to its code degree, it would not run the encoding procedure on other visited source packets. In this way, the code degree would not be exceeded by the number of XORed source packets in each storage node. If we relax this constraint, then,

$$P_s^R(u) = \prod_{c_s(u)} (1 - \frac{d_c(u)}{k}) = \left(1 - \frac{d_c(u)}{k}\right)^{c_s(u)} \tag{11}$$

Hence, after the encoding process, the probability that source packet s would be XORed by the encoded packet Y_u is

$$P_s^X(u) = 1 - P_s^R(u) = 1 - \left(1 - \frac{d_c(u)}{k}\right)^{c_s(u)} \tag{12}$$

During the encoding process of the proposed DSFPB strategy, all source packets are disseminated through independent PBcast mechanism. Hence, after the encoding process, the probability that just $d_c(u)$ source packets would be XORed by the encoded packet Y_u is

$$\begin{aligned}\Pr\left(ns(u)=d_c(u)\right)&=\Omega_{RSD}(d)\times\prod_{i=1}^{d_c(u)}P_s^X(u)\\&=\Omega_{RSD}(d)\times\prod_{i=1}^{d_c(u)}1-\left(1-\frac{d_c(u)}{k}\right)^{c_{s_i}(u)}\end{aligned}\tag{13}$$

where $i=1,...,d_c(u)$ is the XORed source packets of node u. Neglecting the effect of transmission time of PBcast, after the encoding process, the expected number that source packet s visits node u is

$$\begin{aligned}E[c_s(u)]&=T\times d_c(u)\Big/\sum d\\&=T\times d_c(u)\Big/\sum_{v=1}^{n}d_c(v)\end{aligned}\tag{14}$$

where T is the total number of transmissions during the DSFPB strategy, $\sum d=\sum_{v=1}^{n}d_c(v)$ is the total number of XORed source packets in the entire AS network. In particular, there exist $\binom{k}{d_c(u)}$ combinations for choosing $d_c(u)$ source packets from all k source packets. Then,

$$\Pr\left(ns(u)=d_c(u)\right)=\Omega_{RSD}(d)\times\binom{k}{d_c(u)}\times\left(1-\left(1-\frac{d_c(u)}{k}\right)^{T\times\frac{d_c(u)}{\sum d}}\right)^{d_c(u)}\tag{15}$$

We should notice that (15) is the probability that the number of XORed source packets at packet Y_u equals to the code degree $d_c(u)$. However, as shown in (11), it relaxes the constraint of Sign function. That is, each storage node cannot XOR any source packets once they have XORed enough source packets. The Sign function does not work if storage node u XORs less than $d_c(u)$ source packets. Therefore, (15) is the upper bounds of probability that storage node u would fulfill its code degree.

6. Simulation Results and Discussions

In this section, we present several sets of Monte Carlo simulations to evaluate the effectiveness of the proposed DSFPB strategy aiming to reduce the data dissemination cost and improve the decoding performance. We compare it with the EDFC strategy [18] and the LTCDS strategy [20]. We chose these two representative strategies because EDFC is the first distributed storage strategy using fountain codes, while LTCDS is a typical strategy that runs the "on-the-fly" encoding procedure. All simulations results are carried out using the MTALAB simulator.

For the simulation parameters, there are n nodes randomly distributed in a normalized $S=2000\times2000\text{km}^2$ region, where $k=0.1n$ data nodes are randomly selected among the entire AS network. In this article we consider two network size, i.e., $n=100$ and 300, respectively. Each storage node has the same maximal transmission radius R, and we set $R^2=2S\log n/n$ to ensure the network connectivity. In EDFC strategy the length of each random walk is set to 200, while in LTCDS strategy the parameter C is set to 5, i.e., the length of each random walk is $5n\ln n$. In the proposed DSFPB strategy, the forwarding probability p of PBcast is set to 0.3. For all three strategies, RSD is chosen as the degree distribution and

the parameters are set to δ=0.5 and c=0.05. Belief propagations (BP) algorithm is employed for LT decoding, and two main performance metrics are successful decoding probability and data dissemination cost, respectively.

6.1 Successful Decoding Probability

To compare the decoding performance of the proposed DSFPB strategy with EDFC and LTCDS strategies, we evaluate the successful decoding probability P_s versus the decoding ratio η. The *decoding ratio* is defined as the ratio between the number of queried storage nodes h and the number of data nodes k. i.e., $\eta = h/k$. In particular, P_s is defined as the ratio between the number of successfully decoding all k source packets (denote by M_s) and the number of Monte Carlo trials M, i.e., $P_s = M_s / M$. We use the torus convention to avoid the edge effect, and the results are obtained based on 1000 Monte Carlo trials, i.e., $M = 1000$.

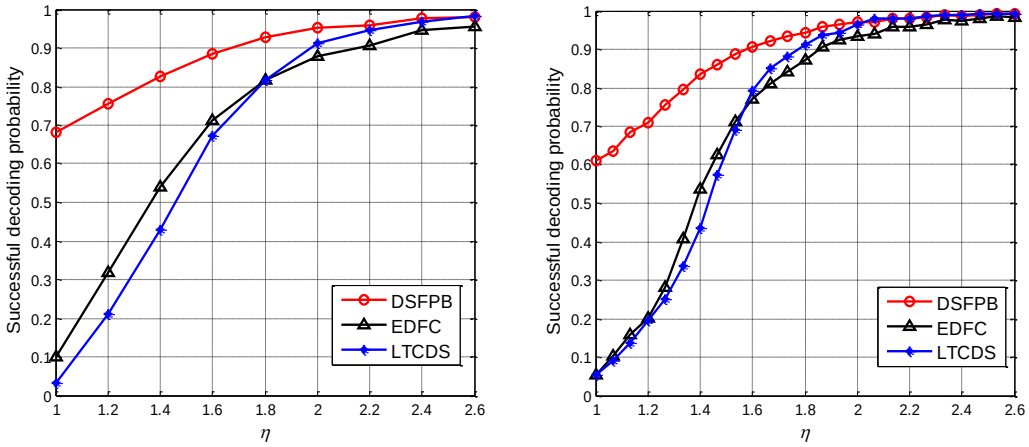

Fig. 8. The probability of successful decoding versus η. (a) $n = 100$. (b) $n = 300$.

Fig. 8 shows the successful decoding probability of the proposed DSFPB strategy in comparison with the EDFC strategy and the LTCDS strategy versus the decoding ratio η for $n = 100$ and 300, respectively. It is evident from the results the proposed DSFPB strategy has a better decoding performance, especially when η is small (e.g., $\eta \le 2$), while the EDFC and LTCDS strategies have almost the same performance. The performance difference decreases as η get larger and finally all three strategies have almost the same performance. For example, when $n = 100$ (**Fig. 8(a)**) and $\eta = 1$, the largest difference between the DSFPB strategy and others is 0.65, while only less than 0.1 when $\eta > 2$. This is because for high decoding ratio, i.e., the user node queries more storage nodes, the probability of fulfilling the desired degree distribution is increasing.

The improvement of decoding performance is benefited from the appropriate encoding procedure of the DSFPB strategy, since each source packet runs the encoding procedure upon its every visit to a storage node. **Fig. 9** shows the final degree distribution of three distributed storage strategies (the proposed DSFPB strategy, EDFC strategy and LTCDS strategy) when $n = 100$. It is evident from the results that the final degree distribution of the proposed DSFPB strategy (**Fig. 9(b)**) is much closer to the desired RSD distribution (**Fig. 9(a)**).

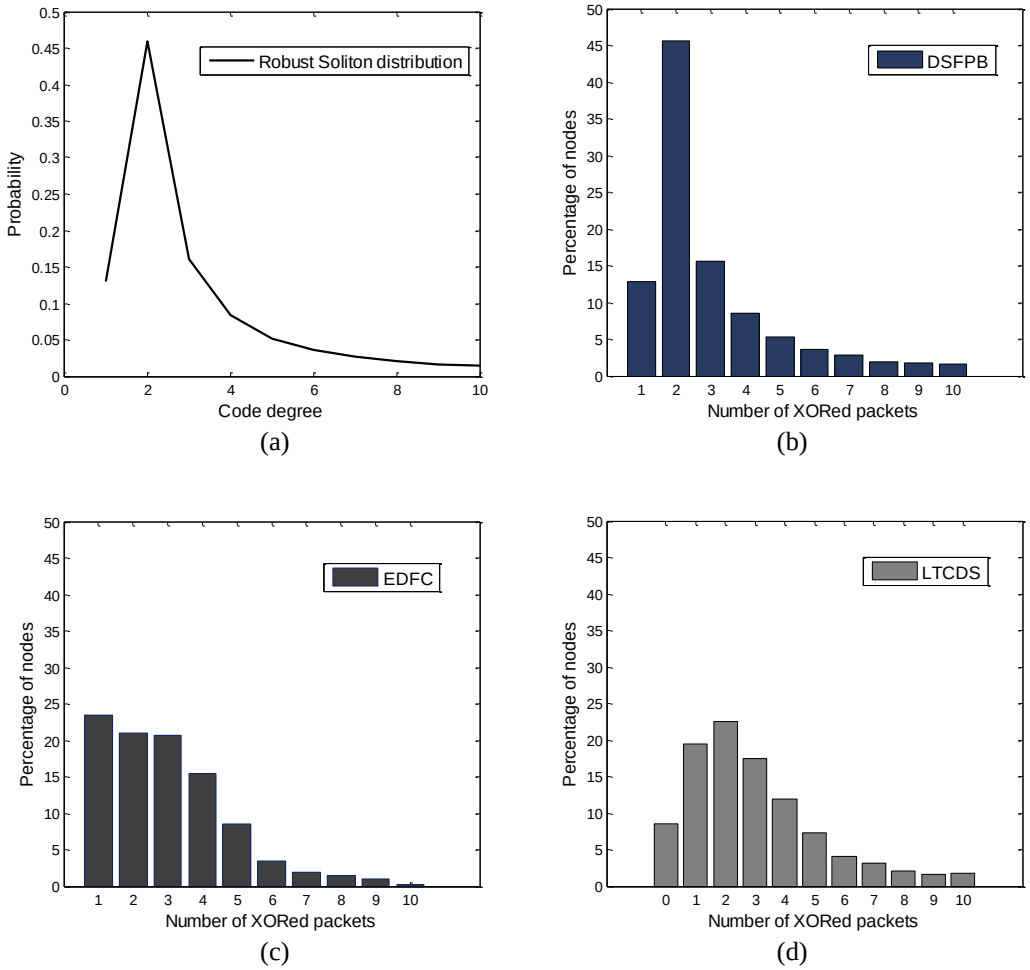

Fig. 9. Final degree distribution of each distributed storage strategy ($n=100$).
(a) Example of RSD. (b) DSFPB. (c) EDFC. (d) LTCDS.

Poor decoding performance of the EDFC and LTCDS strategies are caused by their irregular final degree distribution. This is mainly because their encoding procedure and packets' dissemination way. In the EDFC strategy, encoding procedure is only performed after all random walks are finished. As a result, storage nodes with high code degree might not XOR sufficient source packets as shown in **Fig. 9(c)**. In addition, each data node should launch several random walks during the packets dissemination process. Since each random walk terminates after a few walks (shorter than cover time), storage nodes that far away from the data nodes might not be visited. In the LTCDS strategy, packets run the encoding procedure only at its first visit to a storage node, regardless of the number of XORed source packets. As a result, most storage nodes might not XOR sufficient source packets. In the extreme case almost 10 percent of storage nodes do not XOR any source packets as shown in **Fig. 9(d)**. Therefore the final degree distribution of EDFC and LTCDS strategies are not matched with the RSD distribution, which results in poor decoding performance when queries few storage nodes.

6.2 Data Dissemination Cost

In this section, we compare the data dissemination cost of the proposed DSFPB strategy with the EDFC and LTCDS strategies. As presented above, we assume that each data transmission and reception consumes unit of energy, then the data dissemination cost can be described as the total number of data transmissions and data receptions. **Table 2** shows the number of transmissions N_t, the number of receptions N_r and the data dissemination cost N_{tot} of all three distributed storage strategies (the proposed DSFPB strategy, EDFC strategy and LTCDS strategy) when $n=100$ and 300, respectively. It is obvious from the results that the DSFPB strategy consumes the least energy compared with other two strategies. By exploiting the broadcast property, the data dissemination cost can be reduced more than an order of magnitude using the PBcast manner. In addition, the data dissemination cost of the EDFC strategy is higher than the LTCDS strategy, although the length of each random walk is shorter. This is because that in the EDFC strategy, source packets are only considered for XORing in the last step of random walks. To ensure that most storage nodes receive enough source packets w.h.p., each data node should launch multiple random walks according to the storage nodes n and the degree distribution. Despite shorter length of each random walk, the total data dissemination cost is much higher than the LTCDS and the proposed DSFPB strategies.

Table 2. Total data dissemination cost in each strategy.

Strategy	$n=100$			$n=300$		
	N_t	N_r	N_{tot}	N_t	N_r	N_{tot}
DSFPB	3.07e2	9.90e2	**1.29e3**	2.72e3	8.97e3	**1.16e4**
EDFC	2.93e4	2.93e4	5.86e4	7.03e5	7.03e5	1.40e6
LTCDS	2.30e4	2.30e4	4.60e4	2.56e5	2.56e5	5.13e5

We should mention that during the encoding phase of the DSFPB strategy, in order to utilize the overhearing property without inducing additional energy, each node should maintain a queue Q to record the set of visited source packets. Obviously, Q contains all source packets at the end of the DSFPB strategy. To reduce the storage overhead without sacrificing the decoding performance, it is sensible to discard the packet content if it has been XORed or the code degree has fulfilled. In summary, the proposed DSFPB strategy improves the data dissemination efficiency, while increasing the successful decoding probability.

7. Conclusion

We investigated the distributed data storage problem in the AS network of SIN, a simple but efficient DSFPB strategy was proposed based on fountain codes and PBcast. Compared with most existing strategies, the PBcast was utilized to disseminate source packets instead of random walks. As a result, the dissemination cost was reduced significantly. The theoretic analyses have been studied and we evaluated the performance with Monte Carlo trials. Our simulation results indicated that the DSFPB strategy was improved in both data dissemination process and decoding performance. Therefore, it is significant for the resource-limited SIN.

Despite the assumptions presented in the network model have been widely used in existing distributed data storage strategies, some of them may not practical in reality. In our future work, we will mainly focus on how to improve the storage efficiency under more practical network conditions (e.g., packets drop, transmission model, routing protocols, etc.). Furthermore, various metrics like the transmission delay and the throughput should be used to evaluate the performance of SIN.

References

[1] F.-H. Dong, Q.-F. Huang, H.-J. Li, B. Kong and W. Zhang, "A novel M2M backbone network architecture," *International Journal of Distributed Sensor Networks*, vol. 15, no. 11, pp. 1-10, November, 2015. Article (CrossRef Link).

[2] J. Mukherjee and B. Ramamurthy, "Communication technologies and architectures for space network and Interplanetary Internet," *IEEE Commun. Surveys and Tutorials*, vol. 15, no. 2, pp. 881-897, July, 2013. Article (CrossRef Link).

[3] H.-F. Hu and Y.-A. Liu, "A feasible mesh-based architecture and protocol model of space information network," in *Proc. of 2nd IITA Int. Conf. on Geoscience and Remote Sensing*, pp. 529-531, August 28-31, 2010. Article (CrossRef Link).

[4] F. Ren and J.-L. Fan, "An adaptive distributed certificate management scheme for space information network," *IET Inf. Security*, vol. 7, no. 4, pp. 318-326, December, 2013. Article (CrossRef Link).

[5] W. Zhang, G.-X. Zhang. L. Gou, B. Kong and D.-M. Bian, "A hierarchical autonomous system based topology control algorithm in space Information network," *KSII Trans. on Internet and Inf. Sys.*, vol. 9, no. 9, pp. 3572-3593, September, 2015. Article (CrossRef Link).

[6] G.-X. Zhang, W. Zhang, H. Zhang and Z.-D. Xie, "A novel proposal of architecture and network model for space communication network," in *Proc. of the IAF 65th Int. Astron. Congress*, pp. 1-7, October 1-3, 2014. Article (CrossRef Link).

[7] D. Leong and A. G. Dimakis, "Distributed storage allocations," *IEEE Trans. on Inf. Theory*, vol. 58, no. 7, pp. 4733-4752, March, 2012. Article (CrossRef Link).

[8] W.-D. Sun, Y.-J. Wang and X.-Q. Xiao, "Tree-structured parallel regeneration for multiple data losses in distributed storage systems based on erasure codes," *China Communications*, vol. 10, no. 4, pp. 113-125, April, 2013. Article (CrossRef Link).

[9] A. G. Dimakis, P. B. Godfery, Y.-N. Wu, M. J. Wainwright and K. Ramchandran, "Network coding for distributed storage systems," *IEEE Trans. on Inf. Theory*, vol. 56, no. 9, pp. 4539-4551, September, 2010. Article (CrossRef Link).

[10] S.-J. Lin, W.-H. Chung, Y. S. Han and T. Y. Naffouri, "A unified form of exact-MSR codes via product-matrix frameworks," *IEEE Trans. on Inf. Theory*, vol. 61, no. 2, pp. 873-886, December, 2015. Article (CrossRef Link).

[11] S.-J. Lin and W.-H. Chung, "Novel repair-by-transfer codes and systematic exact-MBR codes with lower complexities and smaller field sizes," *IEEE Trans. on Inf. Theory*, vol. 25, no. 12, pp. 3232-3241, January, 2014. Article (CrossRef Link).

[12] S. Puducheri, "Fountain codes," *IEE Proceeding Communications*, vol. 152, no. 6, pp. 1062-1068, December, 2005. Article (CrossRef Link).

[13] S. Puducheri, J. Kliewer and T. E. Fuja, "The design and performance of distributed LT codes," *IEEE Trans. on Inf. Theory*, vol. 53, no. 10, pp. 3740-3754, October, 2007. Article (CrossRef Link).

[14] A. G. Dimakis, V. Prabhakaran and K. Ramchandran, "Decentralized erasure codes for distributed networked storage," *IEEE Trans. on Inf. Theory*, vol. 52, no.6, pp.2809-2816, June, 2006. Article (CrossRef Link).

[15] Y.-M. Wei, Y. W. FOO and K. C. Lim, "The auto-configurable LDPC codes for distributed storage," in *Proc. of IEEE 17th Int. Conf. on Computational Sci. and Engineering*, pp.1332-1338, December 19-21, 2014. Article (CrossRef Link).

[16] A. G. Dimakis, V. Prabhakaran and K. Ramchandran, "Distributed fountain codes for networked storage," in *Proc. of the IEEE Int. Conf. on Acoustics, Speech and Signal Process. Proceedings*, May 14-19, 2006. Article (CrossRef Link).
[17] A. Kamra, V. Misra, J. Feldman and D. Rubenstein, "Growth codes: maximizing sensor network data persistence," in *Proc. of ACM SIGCOMM*, pp. 255-266, 2006. Article (CrossRef Link).
[18] Y.-F. Lin, B. Liang and B.-C. Li, "Data persistence in large-scale sensor networks with decentralized fountain codes," in *Proc. of 26th IEEE Int. Conf. Comput. Commun.*, pp. 1658-1666, May 6-12, 2007. Article (CrossRef Link).
[19] X.-C. Ye, J. Li, W.-T. Chen and F.-L. Tang, "LT codes based distributed coding for efficient distributed storage in wireless sensor networks," in *Proc. of IFIP Networking Conf.*, pp. 1-9, May 20-22, 2015. Article (CrossRef Link).
[20] Z.-N. Kong, S. A. Aly and E. Soljanin, "Decentralized coding algorithms for distributed storage in wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 28, no.2, pp.261-267, February 2010. Article (CrossRef Link).
[21] S. A. Aly, Z.-N. Kong and E. Soljanin, "Fountain codes based distributed storage algorithms for large-scale wireless sensor networks," in *Proc. of Int. Conf. on Inf. Process. in Sensor Networks*, pp. 171-182, April 22-24, 2008. Article (CrossRef Link).
[22] J.-B. Liang, J.-X. Wang and J. E. Chen, "An overhearing-based scheme for improving data persistence in wireless sensor networks," in *Proc. of IEEE Int. Conf. on Commun.*, pp. 1-5, May 23-27, 2010. Article (CrossRef Link).
[23] J. Zhang, L. Fu, X. Tian, Y. Cui and X. Wang, "Analysis of random walk mobility models with location heterogeneity," *IEEE Trans. Parallel and Distrib. Syst.*, vol. 26, no. 10, pp. 2657-2670, October, 2015. Article (CrossRef Link).
[24] C. Mobius, W. Dargie and A. Schill, "Power Consumption Estimation Models for Processors, Virtual Machines, and Servers," *IEEE Trans. Parallel and Distrib. Syst.*, vol. 25, no. 6, pp. 1600-1614, July, 2014. Article (CrossRef Link).
[25] G. Gianini and E. Damiani, "The cover time of neighbor-avoiding gossiping on geometric random networks," in *Proc. of 7th IEEE Int. Conf. on Digital Ecosystems and Technologies*, pp. 7-12, July 24-26, 2013. Article (CrossRef Link).
[26] K. Suto, P. Avakul, H. Nishiyama and N. Kato, "An efficient data transfer method for distributed storage system over satellite networks," in *Proc. of IEEE 77th Vehicular Technology Conf.*, pp. 1-5, June 2-5, 2013. Article (CrossRef Link).
[27] P. Gupta, P. R. Kumar, "The capacity of wireless networks," *IEEE Trans. Inf. Theory*, vol. 46, no.2, pp. 388-404, March, 2000. Article (CrossRef Link).
[28] S. Subramanian, S. Shakkottai and A. Arapostathis, "Broadcasting in Sensor Networks: The Role of Local Information," *IEEE Trans. Netw.*, vol. 16, no.5, pp. 1133-1146, March, 2008. Article (CrossRef Link).
[29] N. Rahnavard and F. Fekri, "CRBcast: A reliable and energy-efficient broadcast scheme for wireless sensor networks using rateless codes," *IEEE Trans. Wireless Commun.*, vol. 7, no.12, pp. 5390-5400, December, 2008. Article (CrossRef Link).
[30] R. Zheng and R. Barton, "Toward optimal data aggregation in random wireless sensor networks," in *Proc. of 26th IEEE Int. Conf. on Commun.*, pp. 249-257, May 6-12, 2007. Article (CrossRef Link).

Bo Kong received the B.S. degree (with honors) in communication engineering from Xidian University, Xi'an, China, in 2010 and the M.S. degree from College of Communication Engineering, PLA University of Science and Technology, Nanjing, China, in 2013. He is currently pursuing the Ph.D. degree in communications and information systems in College of Communications Engineering, PLA University of Science and Technology. His research interests include satellite communications, space information networks and distributed data storage. He currently serves as a regular reviewer for International Journal of Distributed Sensor Networks.

Gengxin Zhang received his M.S. and Ph.D. degrees from the Institute of Communication Engineer, Nanjing, China in 1990 and 1993 respectively. He is currently a Professor in PLA University of Science and Technology (PLAUST), Nanjing, China. His research interests include the design of communication systems, satellite and deep space communications.

Wei Zhang received the B.S. degree from Xidian University, Xi'an, China in 2009, and his M.S. degree from PLA University of Science and Technology (PLAUST), Nanjing, China in 2012. He is currently a Ph.D. candidate in PLAUST. His research interests are focused on satellite communications, deep space communications, and network capacity. He currently serves as a regular reviewer for International Journal of Distributed Sensor Networks.

Feihong Dong received the B.E. degree in information system engineering from PLA University of Science and Technology, Nanjing, China in 2010. He is currently working toward the Ph.D. degree with the College of Communications Engineering, PLA University of Science and Technology. His research interests include high altitude platform communication networks, satellite communication networks and space information networks.