A Novel Repairable Fountain Codes Scheme Based on Partial Duplication Technique

Anan Zhou¹,² · Benshun Yi¹,² · Mian Xiang¹,² · Laigan Luo¹,²

Accepted: 7 May 2022 / Published online: 12 June 2022
© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2022

Abstract
Distributed storage system (DSS) is an emerging paradigm which provides reliable storage services for various source data. As the fault-tolerance scheme for DSS, erasure codes are required to provide redundancy service with high fault-tolerance and low cost. However, the existing coding scheme cannot provide these requirements well. Thus, it becomes an important yet challenging issue to find a code for storing various source data with high fault-tolerance and low cost. In this paper, a novel construction of repairable fountain codes with unequal locality is proposed by combining with partial duplication technique, namely the PD-ULRFC scheme. We construct a multi-tier heterogeneous storage network, where data core, processing units and storage nodes collaboratively store and transmit data. Moreover, the proposed PD-ULRFC scheme can reduce the repair and download cost by sacrificing a little extra storage occupation. Furthermore, the expressions of the repair cost and download cost are derived to analyze the performance of PD-ULRFC scheme. The simulation results demonstrate that the PD-ULRFC scheme significantly outperforms other redundant schemes in communication cost saving.

Keywords Heterogeneous distributed storage network · Repairable fountain code · Partial duplication technique · Communication cost

✉ Benshun Yi
yibs@whu.edu.cn

Anan Zhou
zhouanan@whu.edu.cn

Mian Xiang
xmhmuzu@qq.com

Laigan Luo
luolaigan@whu.edu.cn

¹ Wuhan University, Wuhan, China
² Hubei Minzu University, Enshi, China
1 Introduction

With the ever-increasing amount of multimedia data, data traffic has exponentially increased recently. The distributed storage systems (DSSs) have gained increasing attention in various fields due to its potential to meet the storage requirements of massive data. However, the stored data cannot keep permanent stability in DSSs because of the failures of storage devices. It is worth to tackle the problem of reconstructing the lost data when storage nodes fail. Currently, the replications and erasure codes are two common redundancy technologies which are employed to ensure the reliability of DSSs [1]. Unfortunately, the two techniques cannot deal with the problem of storage space-bandwidth overhead trade-off well. Especially for the erasure codes, although it occupies less storage space, it needs to transfer a large amount of data to reconstruct the source file. Therefore, it is critical to develop a new redundancy scheme to achieve low storage space while consuming low bandwidth resources.

Conventional Reed-Solomon codes are a best fault tolerance codes because of their maximum distance separable (MDS) structure. However, they still have some drawback, e.g., inefficient repair and high implementation complexity [2, 3]. To improve the repair efficiency, the concept of repair locality is proposed and attracted research focus [4]. The repair locality is the number of drives that will need to be contacted in order to recover a lost encoded symbol. There are some locality-aware code constructions suggested and even implemented in production environments [5, 6]. However, these code construction is start with a base MDS code, then still possess the disadvantage in terms of high encoding and decoding complexities.

As a subclass of erasure codes with several attractive properties, fountain codes [7–9] have attracted an increasing attention of scholars. For a given set of \( k \) message blocks, a fountain code generates a potentially limitless steam of coded symbols, each created independently of others as a random combination of message blocks according to a given degree distribution. Ideally, the original \( k \) message blocks can be reconstructed by selecting randomly \( (1 + \varepsilon)k \) coded symbols in a high probability, where \( \varepsilon > 0 \). The design of fountain code makes it possess low encoding and decoding complexity.

Therefore, fountain codes have been recognized as repair codes for DSSs to maintain the reliability of stored data [10–19]. Early work introducing fountain codes to DSSs provided the high efficiency of data transmission and the low latency of data retrieval [10–13]. However, conventional fountain codes, such as LT or Raptor codes [7, 8], these codes are not systematic and the low degree of the encoded symbols does not imply low locality. Thus, it requires more communication cost of transmitting data for repairing lost data in DSSs. For DSS, it expects that the repair code scheme is systematic form and has low locality, where the locality is defined as the number of symbols that are required to contact to recover a lost symbol. It Recently, the problem of fountain code has been considered in [14–19]. In [14], a novel family of fountain codes, namely repairable fountain code (RFC) was proposed, which has two properties, i.e., systematic form and low locality. The former enables the reading of source data without decoding and the latter can decrease the recovery cost of reconstructing a single lost symbol by local repair. In [15], the author analyzed the repair complexity of LT code and Raptor code and introduced systematic fountain code for DSSs. However, the results shown that there is still a large decoding overhead. The [16] investigated the locality of systematic fountain code under a given degree distribution. However, it could not obtain small decoding overhead because of the BP decoding scheme. With the consideration of systematic fountain codes with BP decoding, the authors in [17]
proposed truncated Poisson distribution to decrease the decoding overhead. Furthermore, Y. Wang et al. [18] proposed an improve repairable fountain codes with unequal repair locality, termed URL-RFC, which reduced the cost of repairing and downloading in multi-tier heterogeneous mobile edge caching networks. In [19], J. Baik et al. proposed a new systematic fountain codes with reduced locality, which is called locality-improved repairable fountain codes (LIRFC). Although the above schemes perform well in reducing the amount of data transmission, it still has the disadvantages of high repair cost compared with the replica technology.

In this work, to further decrease the cost, we propose a novel RFC scheme with unequal locality (UL) by combining with partial duplication technique (PD-ULRFC). Our main contribution is to suggest the use of partial duplication technique, which can increase the chances of repairing and downloading files through the links between storage nodes, thereby reducing the repair cost and download cost. Also, the closed-form expressions of repair cost and download cost are derived to analyze the performance of the PD-ULRFC. By comparing to other redundant schemes, we show that significant gains can be achieved in the reduction of repair cost and download cost.

The organization of the rest of this work is as follows. A brief background on the system network model and the repairable fountain codes are presented in Sect. 2. Afterwards, the detail encoding scheme of PD-ULRFC and the repair and download mechanism is provided in Sect. 3. In Sect. 4, we verify the significant improvements of the proposed scheme by simulation. Finally, we conclude our work in Sect. 5.

2 Background and Network Model

In this section, we present the heterogeneous storage network model and the background of repairable fountain codes.

2.1 The Heterogeneous Storage Network Model

As shown in Fig. 1, we consider a heterogeneous multi-tier storage network, including a data core, Processing Units (PUs), and storage nodes. The data core is responsible for storing all the original data and distributing it to each PUs. In addition, the PUs set is divided into $M$ units, i.e., $PU_1$, $PU_2$, ... , $PU_M$, which are responsible for providing service coverage for all storage nodes. Furthermore, the whole coverage area is partitioned into $M$ areas, i.e., $Area_1$, $Area_2$, ..., $Area_M$, where each of them is serviced by a PU. Moreover, those nodes in the same area can communicate with each other using wireless links. As a result, a multi-tier storage network architecture is formed. In the meantime, it is noted that the data repair and download processes can be cooperatively implemented by PUs and storage nodes.

In the network model, the storage nodes’ failure follows a Poisson process. Therefore, the lifespan of a live node in system can be modeled by an exponentially distributed random variable with parameter $N_m\lambda_m$ which denotes the expected survival rate of a node in $Area_m$, where $N_m$ is supposed as the expected number of nodes in $m$th area, $m \in \{1, 2, \ldots, M\}$ and $t$ is the time. Thus, the probability density function (PDF) of the lifespan $T_{lm}$ is as follows.

\[
f_{T_{lm}}(t) = N_m\lambda_m \cdot e^{N_m\lambda_m t}, t \geq 0
\]  

(1)
Similarly, we can also model the nodes repair time $T_{rm}$ as an exponentially distributed random variable with parameter $\mu_m$. And the PDF of $T_{rm}$ is as follows.

$$f_{T_{rm}}(t) = \mu_m \cdot e^{-\mu_m t}, \ t \geq 0 \quad (2)$$

where $\mu_m$ denotes the repair rate of a node in $Aream$. In $Aream$, the number of storage nodes $n_m$ satisfies $n_m \ll N_m$, which indicates the probability that the number of nodes is smaller than $n_m$ is negligible.

In addition, denote by $\rho_{PU_m}$ and $\rho_m$ the energy cost of transmitting 1-bit information from the $m$th PU and the nodes in $Aream$, respectively. According to the different path losses, i.e., a larger distance will incur a greater cost, we assume that $\rho_{PU_m} > \rho_m$ on average. For simplicity, we mainly focus on storing and distribution of one general file of size $F$ bits. The file is divided into multiple blocks and these blocks can be stored in the PUs. Furthermore, in $Aream$, the nodes randomly request the file at a rate $\omega_m$ with independent identically distributed random interval time $T_{dm}$ with PDF

$$f_{T_{dm}}(t) = \omega_m \cdot e^{-\omega_m t}, \ t \geq 0 \quad (3)$$

The requester node can retrieve the whole file via the links between nodes in $Aream$ and PUs links cooperatively.

In the network model, the file is divided into $M$ unequal blocks and stored into PUs. For any $Aream$, the $F_m$ bits data stored in $PU_m$ are divided into $k_m$ blocks to generate $n_m$ encoded blocks, while one encoded block is stored in a storage node. Under these conditions, we consider uniform allocation, where each node in $Aarea_m$ stores $a_m$ bits data.

Data repairing and downloading with cooperative policy in this network architecture comprises the following two components:

**Data Repairing** When a storage node failure from the network, the data stored in the node will be lost. The lost data have to be recovered in time and stored on a new available node. The transmitted cost generated by accessing available node for repairing the lost data is called repair cost. If there are enough available node in $Aarea_m$, the lost data can be repaired
by links between storage nodes. The amount of data transmitted through node links is $d_m \alpha_m$, where $d_m$ is the repair locality. If the number of available node is insufficient, the lost data can only be reconstructed by the $PU_m$, and the amount of transmitted data is $\alpha_m$. 

**Data Downloading** When node requests a file, the whole file can be transmitted from different storage nodes through wireless links or from PUs directly. The transfer cost incurred by download file is called download cost. The amount of data transmitted through the links between storage nodes is $h_m \alpha_m \geq F_m$, where $h_m$ is download locality. It means that the links between storage nodes are available if $h_m$ or more nodes still alive in $Area_m$. If the number of nodes is smaller than $h_m$, $F_m$ bits are downloaded from the $PU_m$.

### 2.2 The Repairable Fountain Codes

As described in [14], the RFC encoding process can be simplified as follows. We represent $k$ message symbols as $U = (u_1, u_2, \ldots, u_k)$ in $GF(q^p)$, where $q$ is a prime and $p$ is a positive integer, and denote by $V = (v_1, v_2, \ldots, v_n)$ the sequence of $n$ encoded symbols generated by $U$. The encoded symbols contain a non-coded copy of the $k$ message symbols. It means that the first $k$ symbols of $V$ are equal to $U$. The parity symbols $v_j, j = k + 1, k + 2, \ldots, n$ are constructed in the following three-step processes.

1) $d(k)$ message symbols are sequentially chosen independently and uniformly at random with replacement from $k$ message symbols, where $d(k) = c \cdot \log(k)$, and $c$ is constant.

2) For each message symbol chose in 1), a coefficient is uniformly selected from $GF(q^p)$.

3) Each parity symbol can be generated from the linear combination of the message symbols selected in 1), weighted with the coefficients chosen in 2).

Therefore, the rest of the parity symbols can be obtained by repeating steps 1)–3). As the $j$th encoded symbol, the $v_j$ can be calculated as follows.

$$v_j = uG(j) = \sum_1^k \alpha_{ij}u_i$$

where $G(j)$ is the $j$th column of the generator matrix $G$, $u_i$ is the $i$th original message symbol, and $\alpha_{ij}$ is the probability of the $j$th encoded symbol choosing the $i$th original message symbol.

From the encoding process, each parity symbol is the weighted sum of at most $d(k)$ message blocks. We call a parity symbol and its choosing $d(k)$ message blocks as a local group of $d(k) + 1$ cardinality. This means that the repairing of a single failed node can be achieved by downloading at most $d(k)$ symbols from its local group. Therefore, the RFCs scheme get a low locality, $d(k)$.

Furthermore, the summary of mentioned symbols in this paper is listed in Table 1.

---

### 3 The Construction and Analysis of PD-ULRFC Scheme

We consider that original data is stored in different areas of the heterogeneous storage network. The failure/request rates of storage nodes, the energy cost of storage nodes and PUs are different in $M$ areas, which brings about different repairing and downloading
constraints to the fault-tolerant coding scheme design. Therefore, to further decrease the repair cost and download cost, we design a PD-ULRFC scheme in this section.

### 3.1 The Construction of PD-ULRFC

In this subsection, we present the construction of PD-ULRFC for heterogeneous storage network. The proposed PD-ULRFC are obtained by combining the RFC and the partial duplication technique. Assume that the number of input symbols in $Area_m$ is $k_m$. Denote by $U_m = \{u_{m,1}, u_{m,2}, \ldots, u_{m,k_m}\}$ the vector of input symbols in $Area_m$. The encoding algorithm of PD-ULRFC is shown as Algorithm 1:

#### Algorithm 1 The encoding process of PD-ULRFC

**Input:** The number of data blocks, $k_m$; The data resources, from $u_{m,1}$ to $u_{m,k_m}$

**Output:** $(n_m + k_m)$ parity symbols

**Step 1:** The original file is partitioned into $M$ groups of input symbols of different lengths $k_m$, and then transmitted to the $M$ PUs;

**Step 2:** Choose appropriate parameters $(n_m, k_m, d(k_m))$ based on the requirements of $Area_m$;

**Step 3:** $M$ groups of encoded symbols are generated by utilizing RFC encoding method, where the number of encoded symbols in $m$th group is $n_m$, including $k_m$ systematic symbols and $(n_m - k_m)$ parity symbols.

**Step 4:** Subsequently, the $k_m$ systematic symbols are duplicated to obtain double replicas of systematic symbols.

### Table 1 Summary symbols

| Symbols | Explanation |
|---------|-------------|
| $F_m$ | File size of original data in $Area_m$. |
| $N_m$ | Expected number of nodes in $Area_m$. |
| $\mu_m$ | Failure rate of nodes in $Area_m$. |
| $\alpha_m$ | Request rate of nodes in $Area_m$. |
| $\sigma_m$ | Amount of data in one node of $Area_m$. |
| $\rho_m$ | Cost of 1-bit data transmission through links between nodes of $Area_m$. |
| $\rho_{PU_m}$ | Cost of 1-bit data transmission through links between processing units and nodes in $Area_m$. |
| $T_{lm}$ | The lifespan of nodes staying in $Area_m$. |
| $T_{rm}$ | The repair time of nodes in $Area_m$. |
| $T_{dm}$ | The interval time of requesting file of nodes in $Area_m$. |
| $T_m$ | Repair bandwidth of links between nodes of $Area_m$. |
| $T_{PU_m}$ | Repair bandwidth of links of processing units $PU_m$. |
| $k_m$ | Number of source file packets stored in nodes of $Area_m$. |
| $n_m$ | Number of storage nodes in $Area_m$. |
| $d_m$ | The repairing Locality in $Area_m$. |
| $f_{T_{lm}}(t)$ | The pdf of lifespan $T_{lm}$. |
| $f_{T_{rm}}(t)$ | The pdf of repair time $T_{rm}$. |
| $f_{T_{dm}}(t)$ | The pdf of interval time $T_{dm}$. |
Algorithm 1 The encoding process of PD-ULRFC

**Step 5:** Finally, \((n_m + k_m)\) encoded symbols will be obtained, and the encoding parameters of PD-ULRFC scheme are indicated as \((n_m + k_m, k_m, d(k_m))\).

The generator matrix \(G\) of the proposed PD-ULRFC when \(M = 3\) is shown in Fig. 2. The identity group part confers the systematic form, which is composed of \(2k_1 + 2k_2 + 2k_3\) columns. \(P_1, P_2\) and \(P_3\) are random matrix whose columns are sparse, each bearing at most \(d(k_1), d(k_2)\) and \(d(k_3)\) nonzero elements, respectively. It is worth noting that the length of \(k_1, k_2\) and \(k_3\) are unequal, resulting in the unequal lengths of \(n_1, n_2\) and \(n_3\).

As can be seen from the bipartite graph in Fig. 3, the encoded symbols of \(Area_m\) include double replicas of input symbols as the systematic form and a set of parity symbols with locality \(d(k_m)\) generated by input symbols. We can also calculate the vector of the \(m\)th group of symbols as follows.

\[
V_m = U_m G(I_m | P_m)
\]

(5)

where \(I_m = \begin{bmatrix} I_{k_1} \\ I_{k_2} \end{bmatrix}\) denotes the identity matrix group. Assume that \(v_{m,i}\) is denoted as the \(i\)th encoded symbol in \(V_m\), the \(o_{m,i}\) is denoted as the probability of the \(i\)th encoded symbol in \(V_m\) chooses the \(i\)th input symbol from \(U_m\). The \(G(l)\) indicates the \(l\)th column of the generator matrix. Therefore, we can calculate the encoded symbol \(v_{m,i}\) as follows.

\[
v_{m,i} = U_m G(l) = \sum o_{m,i,j} v_{m,i}
\]

(6)

In the following theorem, we prove that the proposed PD-ULRFC scheme has the high full rank probability and the low repair locality in any \(k_m \times n_m\) sub-matrix formed by the corresponding columns of the matrix \(G\).

**Theorem 1** Consider a sub-matrix \(G_m = [I_m | P_m]\), where \(I_m = \begin{bmatrix} I_{k_1} \\ I_{k_2} \end{bmatrix}\) is an identity matrix group. In addition, each column of \(P_m\) has average \(d(k_m) = c \cdot \log(k_m)\) values drawn...
uniformly at random over $GF(q^p)$. Then, for constant $c > 0$ and $q^p > k_m$, a randomly selected $k_m \times (1 + \epsilon)k_m$ sub-matrix $G_s$ is full rank with high probability (w.h.p.).

Proof: $k_m$ input symbols and $(1 + \epsilon)k_m$ coded symbols, there is a random bipartite graph $G_m = (U_m, V_m, E_m)$, where each coded symbol has a degree at most $d(k_m) = c \cdot \log(k_m)$. One input symbol is covered if and only if it is adjacent to at least one coded symbol. The lower bound of the probability of decoding failure is defined as the probability that an uncovered input node exist in bipartite graph $G_m$. According to classical problem of throwing balls into bins, $O(k_m \log(k_m))$ balls are randomly throw to $k_m$ bins to ensure that no bin is empty. Therefore, the decoding failure probability is arbitrarily small, since $(1 + \epsilon)k_m > k_m$. \qed

From the notation $(n_m + k_m, k_m, d(k_m))$ of PD-ULRFC, we can see that the parameter $n_m + k_m$ is referred to as the number of storage nodes in $Area_m$, while the parameter $d(k_m)$ is the number of storage nodes connected for repairing a failed node. Taking $(9+3,3,2)$ PD-ULRFC scheme in $Area_1$ as an example to illustrate the storage and repair processes as shown in Fig. 4. Primarily, the file of size $F_1$ is divided into $k_1 = 3$ blocks of size $\alpha_1 = F_1/k_1$ with encoding them into $n_1 + k_1 = 9 + 3$ coded symbols through PD-ULRFC scheme, where the number of the systematic symbols (i.e., $\{v_{1,1}, v_{1,2}, v_{1,3}\}$) and their replicas (i.e., $\{v'_{1,1}, v'_{1,2}, v'_{1,3}\}$) is 6. Then, storing the 12 coded symbols into 12 storage nodes. In the case that any one of the systematic symbols is lost, it can be repaired by its replica symbol and obviously the size of transmitted data is $\alpha_1$. However, when two systematic nodes with the same data are failed, we must use RFC scheme to repair a node and then repair the other by replica. Accordingly, the total size of transmitted data is $3\alpha_1$. In addition, any one of the parity symbols is lost, it must be repair through RFC scheme and the size of transmitted data is $2\alpha_1$. Moreover, the request node can retrieve the original file by connecting any $(1 + \epsilon)k_1$ nodes.

![Bipartite graph](image-url)
3.2 The Analysis of PD-ULRFC

We will investigate the performance of the PD-ULRFC and derive the closed-form expressions for the repair cost and download cost in the following subsection.

Repair Cost During the repair process, the proposed scheme has three repairing methods, replication repair, RFC repair and PUs repair. Indicated by \( p_{\text{rep}} \), \( p_{\text{RFC}} \), and \( p_{\text{PU}} \) the probabilities of the above repair methods when there are \( i \) node failures in \( \text{Area}_m \), respectively. The \( \gamma_{\text{rep}}, \gamma_{\text{RFC}}, \) and \( \gamma_{\text{PU}} \) are denoted as the repair bandwidth of a single failure node, respectively. Moreover, \( \alpha_m = F_m / k_m \) denotes the amount of data stored in one node and \( \beta_m \) is the amount of data transmitted from each node.

The number of failed nodes in \( \text{Area}_m \) is indicated as \( i_m = i_{m_s} + i_{m_p} \), where \( i_{m_s} \) denotes the number of failed systematic nodes and \( i_{m_p} \) denotes the number of failed parity nodes. Denote by \( S_{\text{min}} \) and \( S_{\text{max}} \) as the minimum and maximum of systematic node pairs stored the same data in the failed nodes, respectively. The RFC repair is possible if a pair of systematic nodes that have the identical data are failed. \( S_{\text{max}} \) can be calculated that \( S_{\text{max}} = \lfloor \frac{i_m}{2} \rfloor \). In the situation of \( i_{m_s} \leq k_m \), there are at least zero node pairs among the \( i_{m_s} \) failed systematic nodes storing the identical data. If \( i_{m_s} > k_m \), there are \( k_m \) failed systematic nodes that contain different data and the rest are stored identical data. Therefore, the size of \( S_{\text{min}} \) can be calculated as follows.

\[
S_{\text{min}} = \begin{cases} 
0, & i_{m_s} \leq k_m \\
 i_{m_s} - k_m, & i_{m_s} > k_m 
\end{cases}
\]  

Assume that the \( p_{\text{rep}} \) is denoted as the probability that \( j \) node pairs store the identical data among \( i_{m_s} \) failed systematic nodes, and \( \sum_{j=S_{\text{min}}}^{i_{m_s}} p_{\text{rep}} = 1 \). If \( j \) systematic node pairs are failed, there will have \( n_m - j \) storage nodes containing different data in \( \text{Area}_m \). When \( n_m - j > d(k_m) \), the system can repair the failed nodes by using replication repair and RFC repair, and there are \( i_{m_s} - 2j \) failed systematic nodes repaired by replication repair. Thus, the probability of replication repair \( p_{\text{rep}} \) is as follows.
where \( f(n_m - j) \) denotes an indicator function, and the corresponding expression is

\[
f(n_m - j) = \begin{cases} 
1, & n_m - j \geq d(k_m) \\
0, & n_m - j < d(k_m)
\end{cases}
\]

In the meantime, the probability of using RFC repair is denoted as follows.

\[
p_{im}^{RFC} = \sum_{j=S_{min}}^{S_{max}} p_{ij} \left( \frac{i_{mp} + 2j}{i_{m}} \right) f(n_m - j)
\]

It is noted that for a pair of equivalent failed nodes with the same data, the RFC repair is first used to repair one node, and then the other can be repaired by replica repair. Therefore, the cost of repairing two equivalent nodes in \( \text{Area}_m \) is

\[
\frac{1}{u_{\text{RFC}} m} + \frac{1}{u_{\text{rep}} m}.
\]

When \( n_m - j < d(k_m) \), the system must recover the lost data by using PUs repair, and the probability of PUs repair is as follows.

\[
p_{im}^{PU} = \sum_{j=S_{min}}^{S_{max}} p_{ij} (1 - f(n_m - j))
\]

We indicate the repair cost of \( i_m \) failed nodes by \( C_{ri_m} \), and calculate the expression of \( C_{ri_m} \) as follows.

\[
C_{ri_m} = \rho_m (i_{mp} \gamma^{RFC}_{m} p_{im}^{rep} + (i_{mp} \gamma^{RFC}_{m} + \gamma^{rep}_{m}) p_{im}^{RFC} + i_{mp} \gamma^{PU}_{m} p_{im}^{PU} \rho^{PU}_{m})
\]

According to the previous knowledge, there are \( (n_m + k_m) \) storage nodes in \( \text{Area}_m \). Denote by \( p_m \) as the probability of single node staying in \( \text{Area}_m \) within repair period \( T \), i.e.,

\[
p_m = e^{-\mu T},
\]

\( b_i(n_m + k_m, p_m) \) is the probability mass function (pmf) of the Bernoulli distribution with the number of storage nodes \( n_m + k_m \) and the node available probability \( p_m \).

\[
b_i(n_m + k_m, p_m) = \binom{n_m + k_m}{i} p_m^i (1 - p_m)^{n_m + k_m - i}, 0 \leq i \leq n_m + k_m
\]

If \( i < d(k_m) \), it must recover the lost data by using PUs repair, otherwise by using other repair ways. Thus, we can get

\[
C_{rm} = \frac{1}{F_m T} \left( \rho^{PU}_m \gamma^{PU}_m \sum_{i=d(k_m)}^{n_m+k_m} (n_m + k_m - i) b_i(n_m + k_m, p_m) \right.
\]

\[
+ \sum_{i=d(k_m)}^{n_m+k_m} b_i(n_m + k_m, p_m) C_{r(n_m+k_m-i)}
\]

Therefore, the total average repair cost of the system is the sum of the repair cost of \( M \) edge areas, i.e.,
\[ C_r = \sum_{m=1}^{M} C_{rm} \] (15)

**Download Cost** According to the decoding rules of the RFC, the whole file can be downloaded by the links between nodes if there are at least \((1 + \varepsilon)k_m\) different available nodes in \(Area_m\), where \(\varepsilon > 0\), otherwise from PUs. In addition, there must be \((1 + \varepsilon)k_m\) different available nodes when there are more than \(2(1 + \varepsilon)k_m - 2\) nodes available in \(Area_m\). Therefore, it is necessary to consider the scenario whether there are \((1 + \varepsilon)k_m\) different available nodes in \(Area_m\) when the number of available nodes is \((1 + \varepsilon)k_m + 1\), \((1 + \varepsilon)k_m + 2\), \ldots, \(2(1 + \varepsilon)k_m - 2\).

When the number of available nodes is \(i_m\) in \(Area_m\), the number of failed nodes can be denoted as \((n_m + k_m - i_m)\). Thus, the probability of \((1 + \varepsilon)k_m\) different available nodes is \(p_{n_m+k_m-i_m}^{rep} + p_{n_m+k_m-i_m}^{RFC}\), and the probability of less than \((1 + \varepsilon)k_m\) different available nodes is \(p_{n_m+k_m-i_m}^{rep} + p_{n_m+k_m-i_m}^{RFC} + p_{n_m+k_m-i_m}^{PU} = 1\).

Therefore, for the case that \(i_m\) is equal to \((1 + \varepsilon)k_m + 1\), \((1 + \varepsilon)k_m + 2\), \ldots, \(2(1 + \varepsilon)k_m - 2\), the download cost is

\[ C_{di_m} = \rho_m (1 + \varepsilon)k_m \alpha_m (p_{n_m+k_m-i_m}^{rep} + p_{n_m+k_m-i_m}^{RFC}) + \rho_{PU} F_m p_{n_m+k_m-i_m}^{PU} \] (16)

During a repair period \(T\), the number of available nodes in \(Area_m\) can be described by a Poisson birth-death process [20]. Thus, the probability of more than \(i_m\) available nodes in \(Area_m\) is

\[ S_{i_m} = \frac{1}{T} \left( \sum_{i=i_m}^{n_m+k_m} \frac{1 - p_{i'}}{\mu_i} \prod_{j=i_m,j \neq i}^{j} \left( j - i \right) \right) \] (17)

where \(\mu_i = \mu'\) for \(i = i_m, \ldots, n_m + k_m\), and \(p_{i'} = e^{-\mu_i T}\).

Therefore, the probability of \(i_m\) available nodes in \(Area_m\) at file request time is \(s_{i_m} = S_{i_m} - S_{i_m+1}\). Finally, the average cost of downloading is

\[ C_{dm} = N_m \omega_m (\rho_{PU} F_m (1 - S_{(1+\varepsilon)k_m})) + \sum_{i_m=(1+\varepsilon)k_m}^{2(1+\varepsilon)k_m-2} s_{i_m} C_{di_m} + \rho_m (1 + \varepsilon)k_m \alpha_m S_{2(1+\varepsilon)k_m-1} \] (18)

where \(N_m \omega_m\) denotes the overall request rate per time.

The total average download cost of the system is the sum of the download costs of \(M\) edge areas, i.e.,

\[ C_d = \sum_{m=1}^{M} C_{dm} \] (19)

Therefore, the total communication cost is defined as the sum of the average repair cost and the download cost, i.e.,

\[ C = C_r + C_d \] (20)
### 3.3 The Availability Analysis of Links Between Storage Nodes in *Area*$_m$

The proposed PD-ULRFC scheme can improve the possibility of using the links between storage nodes. For PD-ULRFC scheme, the repair and download can be achieved by using the links between storage nodes or the PUs links. The $P^{SN}_{rm}$ and $P^{PU}_{rm}$ are denoted as the successful repair probability by using the links between storage nodes and PUs links, respectively. Similarly, the $P^{SN}_{dm}$ and $P^{PU}_{dm}$ are denoted as the successful download probability by through the links between storage nodes and PUs links, respectively. For the $(n_m + k_m, d(k_m))$ PD-ULRFC scheme, we can calculate the above probability as follows.

\[
P^{SN}_{rm} = \sum_{i_m=d(k_m)}^{n_m+k_m} b_i(n_m + k_m, p_m)(p^{rep}_{n_m+k_m-i_m} + p^{RFC}_{n_m+k_m-i_m})
\]

\[
P^{PU}_{rm} = \sum_{i_m=0}^{d(k_m)-1} b_i(n_m + k_m, p_m) + \sum_{i_m=d(k_m)}^{n_m+k_m} b_i(n_m + k_m, p_m)p^{PU}_{n_m+k_m-i_m}
\]

\[
P^{SN}_{dm} = S_{2(1+\epsilon)k_m-2} + \sum_{i_m=(1+\epsilon)k_m}^{2(1+\epsilon)k_m-2} s_i p^{rep}_{n_m+k_m-i_m} + p^{RFC}_{n_m+k_m-i_m}
\]

\[
P^{PU}_{dm} = 1 - S_{2(1+\epsilon)k_m-2} + \sum_{i_m=(1+\epsilon)k_m}^{2(1+\epsilon)k_m-2} s_i p^{PU}_{n_m+k_m-i_m}
\]

### 4 Numerical Results

We evaluate the communication of the PD-ULRFC scheme by simulations. At first, we compare the communication cost of RFC scheme and PD-ULRFC scheme in one area. We take the *Area*$_1$ as the example. We set $\mu_1 = 1$, $\omega_1 = 0.02$, $F_1 = 1$, $N_1 = 50$, and $\rho_{PU_1} : \rho_1 = 5 : 1$. Under different coding parameters, the cost of RFC and PD-ULRFC is shown as Fig. 5.

Figure 5 shows the comparison of repair cost for RFC and PD-ULRFC. Apparently, the PD-ULRFC scheme is superior to the RFC scheme when $T$ less than or equals a certain value. Moreover, the PD-ULRFC scheme's cost is at the same level with the RFC scheme after a certain value. This is the reason why double systematic nodes can decrease the average number of nodes repaired by PUs. Figure 6 shows the change of downloading cost about PD-ULRFC scheme and RFC scheme. Significantly, the downloading cost of the PD-ULRFC scheme is smaller than the RFC scheme. This is because the PD-ULRFC scheme raises the probability of downloading stored files by using the links between nodes when fixed repair period $T$, thereby decreasing the downloading cost. Figure 7 illustrates how the repair period $T$ affects the total communication cost for the PD-ULRFC scheme and RFC scheme. The total communication cost of PD-ULRFC scheme is always lower than RFC scheme at any value of repair period, which shows that the design of PD-ULRFC
Fig. 5 The comparison of the average repair cost for the PD-ULRFC scheme and RFC scheme

Fig. 6 The comparison of the average download cost for the PD-ULRFC scheme and RFC scheme
scheme for distributed storage systems can significantly decrease the total communication cost while the cost of downloading occupies the central part.

Figure 8 shows the successful repair probability of RFC scheme and PD-ULRFC scheme. We choose (20,8,4) RFC and (20+8,8,4) PD-ULRFC for analysis. The blue curve denotes the probability of repairing data by using the links between storage nodes and the red curve denotes through the PUs links. It can be seen that the probability of repairing by using the links between storage nodes becomes smaller as the repair period $T$ increases, and the probability equals almost 0 when $T > 3.5$, i.e., the data can only be repaired through PUs links. In addition, we can see that the probability that PD-ULRFC repairs lost data through the links between storage nodes is higher than the RFC at same $T$, which indicates the cost of the PD-ULRFC scheme is less than the RFC scheme when the $T$ is small. From Fig. 9, we can see that the proposed PD-ULRFC scheme significantly improves the successful download probability through the links between storage nodes. Since the amount of transmitted data during the RFC scheme and PD-ULRFC scheme download is $F_1$ in $Area_1$, there is a conclusion that the greater the probability of downloading by using the links between storage nodes, the smaller the download cost at the same repair period.

Then, we compare the communication cost of PD-ULRFC with other redundancy schemes in a heterogeneous network. In order to accurately evaluate the performance, we set different parameters to construct a heterogeneous network, including different failure rate $\mu_m$, different request rate $\omega_m$, different unit cost of data transmission and different encode parameters. In additions, some systematic parameters are given as $F = 1$, $M = 3$, $N_m = 100$. Other parameters are given in Table 2.

As shown in Fig. 10, we compare the cost overhead of the proposed PD-ULRFC scheme and other redundancy schemes. It can be observed that the PD-ULRFC scheme...
Fig. 8  The analysis of successful repair probability

Fig. 9  The analysis of successful download probability
obtains a lower repair cost than the 3-Replication while the $T$ ranges from 0.2 to 6. However, the 3-Replecation achieves better performance in the near-instantaneous repair duration (i.e., $T < 0.15$). As can be seen from Fig. 11, the download cost of the PD-ULRFC is lower than the URL-RFC and the MDS. This is because the proposed scheme increases the chances of downloading files by utilizing the links between nodes in areas, thereby reducing download cost. From Fig. 12, we can see that the PD-ULRFC scheme is superior to the MDS code and the URL-RFC at any value of $T$ in the simulations, which shows that the PD-ULRFC can effectively decrease the communication cost in heterogeneous storage systems.

| Parameters | Value |
|------------|-------|
| $F_m$      | $F_1 = 0.3, F_2 = 0.4, F_3 = 0.3$ |
| $\mu_m$    | $\mu_1 = 3, \mu_2 = 6, \mu_3 = 8$ |
| $\omega_m$ | $\omega_1 = 0.02, \omega_2 = 0.05, \omega_3 = 0.1$ |
| $\rho_m$   | $\rho_1 = 1, \rho_2 = 3, \rho_3 = 5$ |
| $\rho_{PU_m}$ | $\rho_{PU_1} = 10, \rho_{PU_2} = 30, \rho_{PU_3} = 50$ |
| $k_m$      | $k_1 = 30, k_2 = 20, k_3 = 15$ |
| $n_m$      | $n_1 = 80, n_2 = 65, n_3 = 45$ |
| $d_m$      | $d_1 = 7, d_2 = 6, d_3 = 5$ |

Fig. 10 The comparison of average repair cost of various schemes
Fig. 11  The comparison of average download cost of various schemes

Fig. 12  The comparison of average total communication cost of various schemes
5 Conclusion

In this paper, a novel construction of the RFC scheme was proposed, namely PD-ULRFC scheme, for heterogeneous distributed storage network. The proposed scheme could enhance the availability of links between storage nodes and decrease the repair cost and download cost. In addition, we derived the closed-form expressions of repair cost and download cost of the proposed scheme. Finally, the proposed scheme was compared with other redundancy schemes in terms of repair cost, download cost and total communication cost. Simulation results shown that, the PD-ULRFC scheme was promising to lower the communication cost than other redundancy scheme under the scenarios with different parameter configurations. The practicability and the feasibility of the PD-ULRFC scheme were fully validated in this work. In future work, we will investigate the applications of the PD-ULRFC scheme in realistic storage networks to evaluate the performance of the PD-ULRFC scheme.

Funding This research was supported in part by the National Natural Science Foundation of China [Grant No. 61371125] and the Young Talents Scientific Research Project of Education Department of Hubei Province [Grant No. Q20201902].

Data Availability This research utilizes no data or dataset.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Weatherspoon, H., & Kubiatowicz, J. D. (2002). Erasure coding vs. replication: A quantitative comparison. In Proceedings of the international workshop on peer-to-peer system (pp. 328–337).
2. Dimakis, A. G., Godfrey, P. B., Wu, Y., Wainwright, M. J., & Ramchandran, K. (2010). Network coding for distributed storage systems. IEEE Transactions on Information Theory, 56(9), 4539–4551.
3. Facebook Corp. (2013, September 16). A focus on efficiency. Retrieved June 2 2021, from http://newsroom.fb.com/news/2013/09/focusing-on-efficiency/
4. Gopalan, P., Huang, C., Simitci, H., & Yekhanin, S. (2012). On the locality of codeword symbols. IEEE Transactions on Information Theory, 58(11), 6925–6934.
5. Papailiopoulos, D. S., & Dimakis, A. G. (2012). Locally repairable codes. In Proceedings of the IEEE international symposium on information theory proceeding (ISIT) (pp. 2771–2775).
6. Huang, C., Simitci, H., Xu, Y. K., Ogus, A., Calder, B., Gopalan, P., Li, J., & Yekhanim, S. (2012). Erasure coding in windows azure storage. In Proceedings of the USENIX conferences annual technical conference (pp. 1–12).
7. Luby, M. (2002). LT Codes. In Proceedings of the 43rd Annual IEEE symposium on foundations computer science (pp. 271–280).
8. Shokrollahi, A. (2006). Raptor codes. IEEE Transactions on Information Theory, 52(6), 2551–2567.
9. MacKay, D. J. C. (2005). Fountain Codes. IEEE Proceeding, Part I, Communications, 152(6), 1062–1068.
10. Cao, N., Wang, Q., Ren, K., & Lou, W.J. (2010). Distributed storage coding for flexible and efficient data dissemination and retrieval in wireless sensor networks. In Proceedings of the IEEE international conference on communications (ICC).
11. Lu, H.F., Foh, C. H., Wen, Y.G., & Cai, J.F. (2012). Optimizing content retrieval delay for LT-based distributed cloud storage systems. In Proceedings of the IEEE global communications conference (GLOBECOM).
12. Ye, X. C., Chen, W. T., & Tang, F. L. (2015). LT codes based distributed coding for efficient distributed storage in wireless sensor networks. In Proceedings of the IFIP networking conference.
13. Janet, J., Balakrishnan, S., & Somasekhar, K. (2016). Fountain code based cloud storage mechanism for optimal file retrieval delay. In Proceedings of the international conference on information communication and embedded systems (ICICES).
14. Asteris, M., & Dimakis, A. G. (2014). Repairable fountain codes. IEEE Journal on Selected Areas in Communications, 32(5), 1037–1047.
15. Gummadi, R. (2011). Coding and scheduling in networks for erasures and broadcast. Ph.D. dissertation, University of Illinois at Urbana Champaign.
16. Okpotse, T., & Yousefi, S. (2015). Locality-aware fountain codes for massive distributed storage systems. In Proceedings of the 2015 IEEE 14th Canadian workshop on information theory (CWIT) (pp. 18–21).
17. Okpotse, T., & Yousefi, S. (2019). Systematic fountain codes for massive storage using the truncated Poisson distribution. IEEE Transactions on Communications, 67(2), 943–954.
18. Wang, Y., Gu, S. S., Zhao, L., Zhang, N., Xiang, W., & Zhang, Q. Y. (2020). Repairable fountain coded storage systems for multi-tier mobile edge caching networks. IEEE Transactions on Network Science and Engineering, 7(4), 2310–2322.
19. Baik, J., Suh, Y., Shin, M., Kim, S., & Kim, J. (2020). Locality-improved repairable fountain codes for distributed storage systems. In Proceedings of the IEEE international conference on communication (ICC).
20. Elishco, O., & Barg, A. (2021). Capacity of dynamical storage systems. IEEE Transactions on Information Theory, 67(1), 329–346.

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Anan Zhou was born in 1991. He received the M.S. degree from the Wuhan University of Technology, Wuhan, China, in 2016. He is currently pursuing the Ph.D. degree with the Department of Electronic Information School, Wuhan University, Wuhan, China. His research interests include distributed storage and fountain codes.

Benshun Yi was born in 1965. He received the Ph.D. degrees in electrical engineering from Huazhong University of Science and Technology. He is currently a professor of Electronic Information school of Wuhan University, Wuhan, China. His research interests fall in Power Quality, wireless network and channel coding and he has published extensively in these areas.
Mian Xiang was born in 1992. She received the Ph.D. degree in communication and information system from Wuhan University. She is currently a teacher of Information Engineering School of Hubei Minzu University, Enshi, China. Her research interests fall in wireless network and channel coding.

Laigan Luo was born in 1995. He received the M.S. degree from the Wuhan University of Technology, Wuhan, China, in 2020. He is currently pursuing the Ph.D. degree with the Department of Electronic Information School, Wuhan University, Wuhan, China. His research interests include deep learning and computer vision.