SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

February 2020, Vol. 63 122301:1–122301:15
https://doi.org/10.1007/s11432-019-9937-7

Storage and repair bandwidth tradeoff for heterogeneous cluster distributed storage systems

Jingzhao WANG[1], Yuan LUO[1*] & Kenneth W. SHUM[2]

[1]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai* 200240*, China;*
[2]*Institute of Network Coding, The Chinese University of Hong Kong, Hong Kong, China*

Received 22 April 2019/Revised 31 May 2019/Accepted 3 July 2019/Published online 15 January 2020

Abstract The storage and repair bandwidth tradeoff is an important issue in distributed storage systems (DSSs) where large scale data are stored in multiple nodes with erasure coding to ensure reliability. There are lots of studies on the DSS model with multiple clusters where the repair bandwidths from intra-cluster and cross-cluster nodes are differentiated to improve repair efficiency based on the realistic network topological structures. At the same time, separate nodes are also prevalent due to the variety of practical networks, but the work on the cluster DSS model with multiple separate nodes is insufficient, which is a main motivation of this paper. We formulate the tradeoff bound between storage repair bandwidth for a heterogeneous DSS model consisting of clusters and separate nodes by analyzing the min-cuts of heterogeneous information flow graphs corresponding to the orders of failed nodes. Additionally, the tradeoff bounds are investigated in multiple aspects when the repair bandwidth constraints and the amount of separate nodes vary, respectively. Moreover, a class of regenerating codes are illustrated to achieve the tradeoff in the heterogeneous cases.

Keywords distributed storage system, cluster, separate node, repair bandwidth, information flow graph

Citation Wang J Z, Luo Y, Shum K W. Storage and repair bandwidth tradeoff for heterogeneous cluster distributed storage systems. Sci China Inf Sci, 2020, 63(2): 122301, https://doi.org/10.1007/s11432-019-9937-7

1 Introduction

In modern data centers, large scale data are stored in multiple servers (nodes), where node failures are prevalent [1]. To ensure data reliability with low redundancy, erasure coding is widely used in practical distributed storage systems (DSSs) [2–5], where reducing the repair bandwidth incurred by node failures is an important issue. Dimakis et al. [6] analyzed the DSS model with network coding [7] and formulated the tradeoff between node storage and repair bandwidth, breeding lots of studies on regenerating code constructions [5, 8–12], where product matrix [10] and piggybacking design [8, 11, 12] are typical and powerful construction techniques. In the homogeneous DSS model [6], each node stores the same amount of data, and the newcomer downloads the same amount of data from each helper nodes when repairing a fail one.

In realistic storage systems, the servers are generally organized by clusters (racks) [13, 14], where the networks between storage nodes are different and the intra-cluster data transmission is faster and cheaper than cross-cluster. Additionally, the cross-cluster bandwidth is often limited. In some cases, the available cross-cluster bandwidth for each node is only 1/5 to 1/20 of the intra-cluster bandwidth [15]. In order to improve the transmission efficiency, there are lots of studies on the DSS model with cluster structures differentiating the intra-cluster and cross-cluster bandwidth [1, 16, 17]. These studies can be classified into two types based on whether there is a relay node in each cluster.

* Corresponding author (email: yuanluo@sjtu.edu.cn)

In [1,18,19], the authors focused on minimizing the cross-cluster bandwidth in the multi-cluster model, where the relay node in each cluster was used to collect data from intra-cluster nodes and transmit to the newcomer in the repair procedure. In [16,20], Prakash et al. analyzed the tradeoff bounds in detail when the relay nodes were used both in the repair and data collection procedures and connected to different amounts of intra-cluster nodes. On the other hand, the authors of [17,21,22] considered the cluster DSS model without relay nodes, where the intra-cluster and cross-cluster repair bandwidth were analyzed as two parameters in the repair procedure. In [21, 23], the authors considered the two-rack model and analyzed the storage and repair bandwidth tradeoff. Sohn et al. [17, 24] formulated the tradeoff bound theoretically for the multiple cluster DSS model, and the newcomer would download data from all the alive nodes in the repair procedure.

In [22], we investigated the cluster DSS model when the number of help nodes varied, which is different with the constraints in [17, 24]. In heterogeneous storage systems, such as peer-to-peer (P2P) networks, sensor networks, hybrid cloud storage systems and content delivery networks [25–27], separate nodes are also prevalent. For example, in large scale data centers consisting of many racks (clusters) and separate servers, the storage system can be modeled with the cluster DSS model with multiple separate nodes as proposed in [22, 28]. In [28], we theoretically analyzed the tradeoff bounds of the cluster DSS model after one separate node was added. However, the tradeoff bound for the cluster DSS model with multiple separate nodes has not been characterized and proved theoretically, which is a main motivation of this paper. As the separate nodes can be seen as special clusters with a single node, the model in this paper is regarded as the heterogeneous cluster DSS (HC-DSS) model.

The structure and contributions of this paper are as follows. In Section 2, we introduce the heterogeneous cluster DSS model with multiple separates and formulate the capacity problem with information flow graph. Main definitions used to analyze the HC-DSS model are introduced in Subsection 2.3. Unlike analyzing the cluster DSS model with one separate node [28], we need to investigate all the repair sequences corresponding to different amounts and locations of separate nodes in the HC-DSS model. The main challenge is to find an efficient comparison procedure considering the relationship among multiple separate nodes, which is the main contribution of this paper. In Section 3, we generalize traditional algorithms to investigate the capacity of the heterogeneous model. The final capacity is obtained by analyzing all the node repair sequences step by step corresponding to Theorems 1–3, respectively. In Subsection 3.2, the tradeoff bounds between node storage and repair bandwidth are formulated and analyzed numerically in multiple aspects when the repair bandwidth constraints and the number of separate nodes vary, respectively. In Section 4, a class of constructions on regenerating code are investigated to achieve the tradeoff bound of the HC-DSS model.

2 System model and preliminaries

Subsection 2.1 introduces the heterogeneous cluster DSS model. In Subsection 2.2, we introduce the information flow graph, which is used to characterize the system capacity of the DSS model. The main definitions used to analyse information flow graphs are presented in Subsection 2.3.

2.1 The heterogeneous cluster DSS (HC-DSS) model

Figure 1 illustrates the HC-DSS model comprised with E separate nodes and LR cluster nodes (L clusters with R nodes in each one). Obviously, there are $n = LR + E$ storage nodes in total. The original file of size $\mathcal{M}$ is encoded and stored in n nodes each of size α. We can retrieve the original file by downloading data from any k nodes out of n, which is defined in [29] as the (n, k)-MDS property[1)]. If a node has failed, we can recover it exactly or functionally. This paper considers functional repair, unlike exact repair, the newcomer need not to reconstruct the failed node exactly only if the (n, k)-MDS property is satisfied after repair procedure. In addition, we only consider one node failure and assume that the newcomer locates

1) This name is from the $[n, k]$ maximum distance separable (MDS) codes in coding theory.

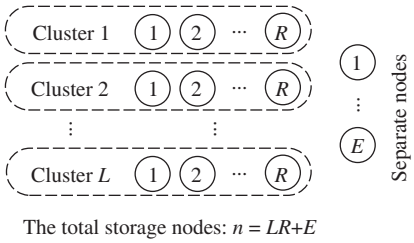

Figure 1 The HC-DSS model.

at the same place with the failed node. Hence, the system model will not change with the failure/repair procedures. We will consider the failure/repair procedures of cluster and separate nodes, respectively.

After a cluster node has failed, the newcomer downloads β_I symbols from each of the d_I intra-cluster nodes and β_C symbols from each of the d_C cross-cluster nodes. In this paper, we denote the total number of helper nodes as $d \triangleq d_I + d_C$, and assume

$$k \leqslant d \leqslant n-1 \tag{1}$$

as introduced in [10].

Upon the failure of a separate node, the newcomer downloads β_C symbols from each of d cross-cluster nodes. It is because the separate nodes can be seen as special clusters with a single node. As introduced in [22], the intra-cluster networks are generally cheaper and faster than cross-cluster, thus more data are downloaded from intra-cluster nodes. As a result, we assume $\beta_I \geqslant \beta_C$ throughout the paper. Additionally, we assume the newcomer will download data from all the intra-cluster nodes, namely,

$$d_I = R-1.$$

Hence,

$$k-R+1 \leqslant d_C \leqslant n-R, \tag{2}$$

based on (1). For simplicity, we use storage/repair parameters and node parameters to denote $(\alpha, d_I, \beta_I, d_C, \beta_C)$ and (n, k, L, R, E), respectively in the HC-DSS model. If $\beta_I = \beta_C$, the homogeneous DSS model in [6] is retrieved.

2.2 Problem formulation

In [6], DSSs are modeled with the information flow graph (IFG) comprised with three kinds of nodes: data source S, data collector DC and storage nodes x^i (represented by a pair of input and output nodes $(x^i_{\text{in}}, x^i_{\text{out}})$ in the graph). The capacity of edge $x^i_{\text{in}} \to x^i_{\text{out}}$ represents the node storage α. The original data are stored in n nodes represented by n edges (with infinite capacity) emanating from S to $\{x^i_{\text{in}}\}_{i=1}^n$. The failure/repair procedure is represented by a newcomer x^{n+1} connecting d alive nodes with edges with capacity β, which means that the failed node is repaired through downloading β data from d helper nodes. However, the capacities of incoming edge for a newcomer are different in the heterogeneous cases. The total alive nodes remain n when each failure/repair procedure is finished. Meanwhile, the (n,k)-MDS property is kept by DC connecting arbitrary k alive nodes to retrieve the original file. In Figure 2, the original file is stored in five nodes (x^1, x^2 in cluster 1; x^3, x^4 in cluster 2; separate node x^5). Firstly, x^1 has failed and is repaired by the newcomer x^6 which connects with one intra-cluster node x^2 and three cross-cluster nodes x^3, x^4 and x^5. Subsequently, a separate node x^5 has failed and is recovered by x^7 connecting with four cross-cluster nodes. The DC connects $k=4$ alive nodes to reconstruct the original file.

In the IFGs, a cut set between S and DC is an edge set that contains at least one edge of every paths from S to DC. The min-cut is a cut set between S and DC with the smallest total sum of the edge capacities [6]. In this paper, we also use min-cut to denote the total sum capacity of the edges in

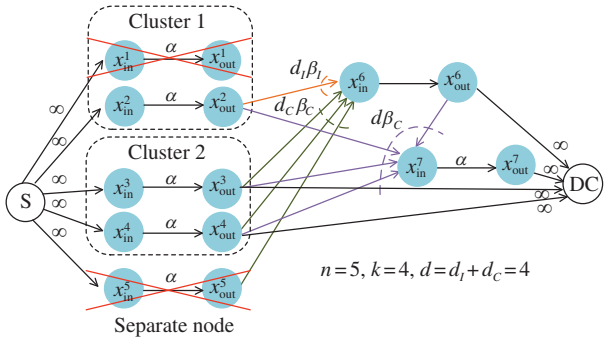

Figure 2 (Color online) An information flow graph.

the min-cut with out incurring confusion. As introduced in [6], the problem that any DC could retrieve the original data $\mathcal{M}$ are equivalent to S multicasting $\mathcal{M}$ to every DC. Based on the max-flow bound in network coding [7,30], the original data $\mathcal{M}$ cannot be greater than the minimum min-cut which is defined as the system capacity and denoted by $\mathbb{C}$, this is to say,

$$\mathbb{C} \geqslant \mathcal{M} \tag{3}$$

need to be satisfied to maintain the (n,k)-MDS property. As introduced in [22], for an (n,k,L,R,E) HC-DSS model, we need to characterize the feasible region of points $(\alpha,d_I,\beta_I,d_C,\beta_C)$ to reliably store the data file of size $\mathcal{M}$. As capacity $\mathbb{C}$ is formulated as a function of $(\alpha,d_I,\beta_I,d_C,\beta_C)$, the tradeoff between α and $(d_I,\beta_I,d_C,\beta_C)$ can be derived with (3). Obviously, the capacity of a given HC-DSS is characterized by comparing the min-cuts of all possible IFGs.

2.3 Main definitions

To analyse the min-cut of the HC-DSS model, we introduce the main definitions in this section such as repair sequence and the min-cut, selected node distribution, cluster order, and separate location vector, parts of which are already defined in [22].

Repair sequence and the min-cut. Since the IFG is acyclic, the k output nodes connected by DC can be topologically ordered as $\{x_{\text{out}}^{t_i}\}_{i=1}^{k}$, where the topological ordering [31] means that if there exists a path from $x_{\text{out}}^{t_i}$ to $x_{\text{out}}^{t_j}$, then $i<j$. It is proved in [6,17] that the min-cut may get smaller if $\{x^{t_i}\}_{i=1}^{k}$ are all newcomers and x^{t_i} connects to the $i-1$ former newcomers $\{x^{t_j}\}_{j=1}^{i-1}$, which is also satisfied in the HC-DSS model. We use min-cut to specify that under the above constraint in the following parts. Obviously, the k newcomers $\{x^{t_i}\}_{i=1}^{k}$ correspond to a repair sequence of k original nodes called selected nodes. As analyzed in [22,24], the min-cut in the HC-DSS model is obtained by cutting $\{x^{t_i}\}_{i=1}^{k}$ one by one in the topological ordering as

$$\text{MC}(\boldsymbol{s},\boldsymbol{\pi}) = \sum_{i=1}^{k} \min\left\{\alpha, w_i(\boldsymbol{s},\boldsymbol{\pi})\right\}, \tag{4}$$

where cluster order $\boldsymbol{\pi}$ and selected node distribution $\boldsymbol{s}$ are defined subsequently to represent the repair sequence. In fact, as illustrated in Figure 3, when cutting x^{t_i}, since the capacity of the edge from $x_{\text{out}}^{t_i}$ to DC is infinite, we only need to consider the capacity of the following two parts of edges:

(1) The capacity of edge $x_{\text{in}}^{t_i} \to x_{\text{out}}^{t_i}$, which equals α.

(2) The total capacity of the edges terminating at $x_{\text{in}}^{t_i}$, which is called the part incoming weight and denoted by

$$w_i(\boldsymbol{s},\boldsymbol{\pi}) = \begin{cases} a_i(\boldsymbol{s},\boldsymbol{\pi})\beta_I + b_i(\boldsymbol{s},\boldsymbol{\pi})\beta_C, & \text{if node } i \text{ is in cluster,} \\ c_i(\boldsymbol{s},\boldsymbol{\pi})\beta_C, & \text{if node } i \text{ is separate,} \end{cases} \tag{5}$$

where $c_i(\boldsymbol{s},\boldsymbol{\pi}) = d+1-i$ as introduced in [28].

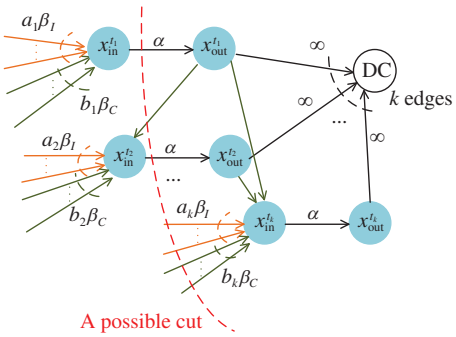

Figure 3 (Color online) The part-cut values of cutting $\{x^{t_i}\}_{i=1}^{k}$ in the topological ordering.

The part-cut value $\min\{\alpha, w_i(\boldsymbol{s},\boldsymbol{\pi})\}$ is obtained by cutting the edges corresponding to the minor one. Obviously, the min-cut $\text{MC}(\boldsymbol{s},\boldsymbol{\pi})$ is formulated with the k part-cut values as shown in (4). Since a cluster order is enough to determine a repair sequence, $\text{MC}(\boldsymbol{s},\boldsymbol{\pi})$, $w_i(\boldsymbol{s},\boldsymbol{\pi})$, $a_i(\boldsymbol{s},\boldsymbol{\pi})$, $b_i(\boldsymbol{s},\boldsymbol{\pi})$ and $c_i(\boldsymbol{s},\boldsymbol{\pi})$ are also written as $\text{MC}(\boldsymbol{\pi})$, $w_i(\boldsymbol{\pi})$, $a_i(\boldsymbol{\pi})$, $b_i(\boldsymbol{\pi})$ and $c_i(\boldsymbol{\pi})$, respectively for simplicity, when we do not analyse $\boldsymbol{s}$ or $\boldsymbol{s}$ is fixed beforehand without incurring confusion.

Selected node distribution. As the L clusters are indiscriminate for an (n,k,L,R,E) HC-DSS model, we can always order and relabel the clusters by the number of selected nodes in a non-increasing order for any given repair sequence. The selected nodes are represented with $\boldsymbol{s}=(s_0,s_1,\ldots,s_L)$, where s_0 and s_i $(1\leqslant i\leqslant L)$ denote the amount of selected separate nodes and nodes in the ith cluster, respectively. Obviously, the selected node distribution $\boldsymbol{s}$ satisfies that $0\leqslant s_0\leqslant E$ and $s_L\leqslant\cdots\leqslant s_1$ as the clusters are relabeled for convenience. We also represent the set of all possible $\boldsymbol{s}$ as

$$\mathcal{S}\triangleq\left\{\boldsymbol{s}=(s_0,s_1,\ldots,s_L):\ s_{i+1}\leqslant s_i,\ 0\leqslant s_i\leqslant R,\ \text{for } 1\leqslant i\leqslant L;\ 0\leqslant s_0\leqslant E;\ \sum_{i=0}^{L}s_i=k\right\}. \tag{6}$$

Cluster order. Additionally, we label the selected nodes from 1 to k for the simplicity of description by the repair sequence, which is denoted by cluster order $\boldsymbol{\pi}=(\pi_1,\pi_2,\ldots,\pi_k)$ and π_i $(1\leqslant i\leqslant k)$ is the label of the cluster where x^{t_i} belongs to. It is enough to denote cluster label of a selected node because there is no difference among the nodes in one cluster. If the selected node i is separate, we denote π_i as 0, then $0\leqslant\pi_i\leqslant L$. Actually, the selected node distribution focuses on the amount while the cluster order characterizes the order of selected nodes. Thus for a given selected node distribution $\boldsymbol{s}$, there are multiple possible cluster orders represented by the set

$$\Pi(\boldsymbol{s})\triangleq\left\{\boldsymbol{\pi}=(\pi_1,\ldots,\pi_k):\ \sum_{j=1}^{k}\mathbb{I}(\pi_j=i)=s_i\ \text{for } 0\leqslant i\leqslant L\right\}, \tag{7}$$

where $\mathbb{I}(\pi_j=i)$ equals 1 if $\pi_j=i$, and 0 otherwise.

Separate location vector. The amount and locations of selected separate nodes are denoted with N_{sep} $(0\leqslant N_{\text{sep}}\leqslant k)$ and separate location vector $\boldsymbol{v}=(v_1,\ldots,v_k)$, respectively, where v_i $(1\leqslant i\leqslant N_{\text{sep}})$ is the location number of the i-th selected separate node and $v_i=0$ $(N_{\text{sep}}<i\leqslant k)$. For convenience, the components of $\boldsymbol{v}$ is also set to satisfy $v_{i-1}\leqslant v_i$ for $1\leqslant i\leqslant N_{\text{sep}}$ without loss of generality. Obviously, there are $k-N_{\text{sep}}$ selected cluster nodes. When $N_{\text{sep}}=0$, there is no selected separate node and $\boldsymbol{v}=(0,\ldots,0)$. When $N_{\text{sep}}=k$, all the selected nodes are separate and $\boldsymbol{v}=(1,2,\ldots,k)$.

For example, in Figure 4(a), there are eight selected nodes numbered from 1 to 8 by the repair sequence corresponding to $\boldsymbol{s}=(2,4,2,0)$ and $\boldsymbol{\pi}=(1,2,0,1,2,1,0,1)$. The amount of selected separate nodes is $N_{\text{sep}}=2$ and the corresponding separate location vector is $\boldsymbol{v}=(3,7,0,\ldots,0)$.

As introduced in Subsection 2.2, for a given HC-DSS model, we need to find the IFG with the minimum min-cut (defined as the system capacity). With (4), the min-cuts corresponding to different repair sequences can be obtained. In Section 3, we will compare the min-cuts from the aspects of cluster order and selected node distribution (determining the repair sequence), and characterize the system capacity finally.

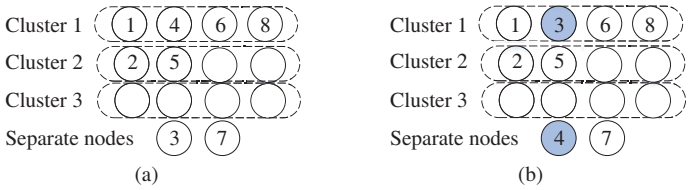

Figure 4 (Color online) As the numbered nodes show, $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ are generated by Algorithm 1 for the same $\boldsymbol{s} = (2, 4, 2, 0)$ and different separate location vectors. (a) $\boldsymbol{\pi} = (1, 2, 0, 1, 2, 1, 0, 1)$; (b) $\boldsymbol{\pi}' = (1, 2, 1, 0, 2, 1, 0, 1)$.

3 The capacity of the HC-DSS model

In Subsection 3.1, we introduce the main challenge in analyzing the min-cuts of the HC-DSS model comparing with the model with only one separate node. Heterogeneous vertical order algorithm and horizontal selection algorithms are proposed to investigate the repair sequences with multiple separate nodes, which focus on the cluster order and selected node distribution, respectively. Subsequently, we compare the min-cuts by reducing the scope of the repair sequences step by step corresponding to Theorems 1–3, respectively. The final capacity is proved in Theorem 3, with which the tradeoff bounds is formulated and analyzed in Subsection 3.2.

3.1 Heterogeneous vertical order algorithm and horizontal selection algorithm

As introduced in Subsection 2.3, there are multiple selected node distributions $\boldsymbol{s} \in \mathcal{S}$ (see (6)) for a given HC-DSS model. At the same time, there are different cluster orders $\boldsymbol{\pi} \in \Pi(\boldsymbol{s})$ (see (7)) for a fixed $\boldsymbol{s}$. Thus the min-cuts are analyzed in the following two aspects.

(1) Fix $\boldsymbol{s}$ and compare the min-cuts corresponding to different $\boldsymbol{\pi} \in \Pi(\boldsymbol{s})$.

(2) Fix the cluster order generating algorithm and compare the min-cuts corresponding to different selected node distributions $\boldsymbol{s} \in \mathcal{S}$.

In [17, 24], the authors proposed algorithms for the cluster DSS model corresponding to the above two steps under the assumption that the newcomer connects to all the alive nodes, namely, $d = n - 1$. However, this assumption cannot be used to analyse the cases with multiple node failures, which is also frequent in realistic storage systems [20, 32]. Thus we investigate more flexible situations in [22, 28], where the applicability of the algorithms are proved under more general assumption $d_I = R - 1$ and $k - R + 1 \leqslant d_C \leqslant n - R$. Additionally, we generalized the algorithms to the case with one separate node. In this paper, we also analyse the min-cuts under the general assumptions in the above two steps and propose the heterogeneous algorithms for the HC-DSS model with multiple separate nodes.

The vertical order and horizontal selection algorithms in [28] only need to handle the location of one separate node, while Algorithms 1 and 2 consider the locations of multiple separate nodes, which are generalizations of the former algorithms and more flexible to analyse the min-cuts. When there is only one separate node in the model of [28], the min-cuts are compared in two aspects with or without the selected separate node. Although the proof methods in [28] are inductive for the multiple separate node cases as Theorem 1, Lemma 1 and Theorem 3 show, it is more complicated to analyse the min-cuts of the HC-DSS model. We need to compare the min-cuts corresponding to different numbers and locations of selected separate nodes, which are represented by N_{sep} and the separate location vector as defined in Subsection 2.3. The main challenge is to find an efficient comparison procedure considering the relationship among multiple selected separate nodes. In the following parts, the final capacity of the HC-DSS model is characterized by reducing the comparing scope corresponding to different selected separate nodes step by step, which is the main contribution of this paper. Parts of the proof procedures can be deduced by the case with one separate node and are omitted for simplicity, but we introduce the main ideas and differences in the model with multiple separate nodes.

In Algorithm 1, the inputs are selected node distribution $\boldsymbol{s}$ and separate location vector $\boldsymbol{v}$. The output cluster order is generated vertically as shown in Figure 4(a), where $\boldsymbol{s} = (2, 4, 2, 0)$ and the separate location vector is $\boldsymbol{v} = (3, 7, 0, 0, 0, 0, 0, 0)$. The selected cluster nodes are generated column by column

from left to right. The first two selected nodes are in clusters 1 and 2 in the first column, namely, $\pi_1 = 1$ and $\pi_2 = 2$. The third selected node is separate as $\boldsymbol{v}$ shows, then $\pi_3 = 0$. The fourth and fifth selected nodes are in the second column and generated vertically, thus $\pi_4 = 1$ and $\pi_5 = 2$. Similarly, Algorithm 1 outputs $\boldsymbol{\pi} = (1, 2, 0, 1, 2, 1, 0, 1)$ finally. Let cluster order

$$\boldsymbol{\pi}^*(\boldsymbol{s}, \boldsymbol{v}) = (\pi^*(\boldsymbol{s}, \boldsymbol{v})_1, \pi^*(\boldsymbol{s}, \boldsymbol{v})_2, \ldots, \pi^*(\boldsymbol{s}, \boldsymbol{v})_k) \tag{8}$$

denote the output of Algorithm 1 with input separate location vector $\boldsymbol{v}$ and $\boldsymbol{s} \in \mathcal{S}$.

Algorithm 1 Heterogeneous vertical order algorithm

Input: Selected node distribution $\boldsymbol{s} = (s_0, s_1, \ldots, s_L)$; separate location vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_k)$.
Output: Cluster order $\boldsymbol{\pi}^* = (\pi_1^*, \ldots, \pi_k^*)$.

Initial cluster label $j \leftarrow 1$ and count variable $t \leftarrow 1$;
for $i = 1$ to k **do**
  if $i = v_t$ **then**
    $\pi_i^* \leftarrow 0$; $t \leftarrow t + 1$; continue;
  end if
  if $s_j = 0$ **then**
    $j \leftarrow 1$;
  else
    $\pi_i^* \leftarrow j$; $s_j \leftarrow s_j - 1$; $j \leftarrow (j \mod L) + 1$;
  end if
end for

In Algorithm 2, N_{sep} separate nodes are selected firstly. Subsequently the cluster nodes are selected from cluster 1 to cluster L until $k - N_{\text{sep}}$ nodes are selected. As shown in Figure 4(a), the algorithm inputs are $N_{\text{sep}} = 2$ and $(n = 14, k = 8, R = 4, L = 4, E = 2)$. First, the $E = 2$ separate nodes are selected, namely, $s_0^* = 2$. Second, we select all the nodes in cluster 1, namely, $s_1^* = R = 4$. Then two nodes in cluster 2 are selected, namely, $s_2^* = 2$. We already get $k = 8$ selected nodes in total. Hence, no more nodes are selected in the remaining clusters and the final output is $\boldsymbol{s}^* = (2, 4, 2, 0)$.

Algorithm 2 Heterogeneous horizontal selection algorithm

Input: Node parameters (n, k, L, R, E); the amount of selected separate nodes N_{sep} $(1 \leqslant N_{\text{sep}} \leqslant E)$
Output: Selected node distribution $\boldsymbol{s}^* = (s_0^*, s_1^*, \ldots, s_L^*)$.

$s_0^* \leftarrow N_{\text{sep}}$;
for $i = 1$ to L **do**
  if $i \leqslant \left\lfloor \frac{k - s_0^*}{R} \right\rfloor$ **then**
    $s_i^* \leftarrow R$;
  end if
  if $i = \left\lfloor \frac{k - s_0^*}{R} \right\rfloor + 1$ **then**
    $s_i^* \leftarrow k - s_0^* - \left\lfloor \frac{k - s_0^*}{R} \right\rfloor R$;
  end if
  if $i > \left\lfloor \frac{k - s_0^*}{R} \right\rfloor + 1$ **then**
    $s_i^* \leftarrow 0$;
  end if
end for

When the separate location vector is set as $\boldsymbol{v} = (0, \ldots, 0)$ and $N_{\text{sep}} = 0$, the algorithms for the cluster DSS model in [17, 24] are retrieved from Algorithms 1 and 2. Similarly, if $N_{\text{sep}} = 1$ and $\boldsymbol{v} = (j, 0, \ldots, 0)$ $(1 \leqslant j \leqslant k)$, we obtain the algorithms for the case with only one separate node in [22]. In the cluster DSS model, Algorithms 1 and 2 directly generate the cluster order achieving the minimum min-cut. However, in the HC-DSS model, since the selected separate nodes will influence the min-cuts, the algorithms only generate the cluster order achieving the local minimum min-cut for an input separate location vector which is proved in Theorem 1. In order to find the cluster order achieving the global minimum min-cut and obtain the capacity, we need to compare the local minimum min-cuts for different separate location vectors, which is investigated in Lemma 1 and Theorem 2.

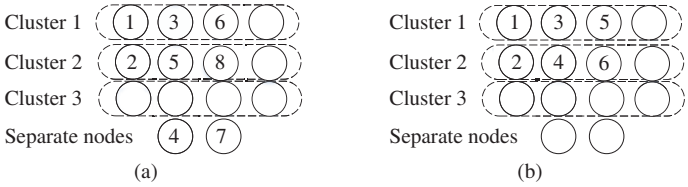

Figure 5 The cluster nodes of $\boldsymbol{s} = (2, 3, 3, 0)$ and $\boldsymbol{\pi} = (1, 2, 1, 0, 2, 1, 0, 2)$ shown in (a) are represented with $\overline{\boldsymbol{s}} = (0, 3, 3, 0)$ and $\overline{\boldsymbol{\pi}} = (1, 2, 1, 2, 1, 2)$ respectively in (b).

Theorem 1. In the HC-DSS model, for a given separate location vector $\boldsymbol{v}$ with N_{sep} separate nodes, $\boldsymbol{\pi}^*(\boldsymbol{s}^*, \boldsymbol{v})$ minimizes the min-cut, that is to say,

$$\text{MC}(\boldsymbol{s}^*, \boldsymbol{\pi}^*(\boldsymbol{s}^*, \boldsymbol{v})) \leqslant \text{MC}(\boldsymbol{s}, \boldsymbol{\pi})$$

holds for all $\boldsymbol{s} \in \mathcal{S}$ and $\boldsymbol{\pi} \in \Pi(\boldsymbol{s})$ with N_{sep} and separate location vector $\boldsymbol{v}$, where $\boldsymbol{\pi}^*(\boldsymbol{s}^*, \boldsymbol{v})$ is given by (8) with $\boldsymbol{s}^*$ generated by Algorithm 2.

Proof. As proved in [28], when comparing the part incoming weights (defined by (5)) of $\boldsymbol{\pi}^*(\boldsymbol{s}^*, \boldsymbol{v})$ and $\boldsymbol{\pi}$, we only need to consider the selected cluster nodes because the part incoming weights of the selected separate nodes are equal. Thus, the selected cluster nodes are represented by another cluster order without separate nodes. As Figure 5 shows, $\boldsymbol{s} = (2, 3, 3, 0)$ and $\boldsymbol{\pi} = (1, 2, 1, 0, 2, 1, 0, 2)$ with separate nodes are represented with $\overline{\boldsymbol{s}} = (0, 3, 3, 0)$ and $\overline{\boldsymbol{\pi}} = (1, 2, 1, 2, 1, 2)$ in Figure 5(b). The original cluster order $\boldsymbol{\pi}$ with separate location vector $\boldsymbol{v}$ and the contrast cluster order $\overline{\boldsymbol{\pi}}$ without separate nodes satisfy that

$$\pi_i = \begin{cases} \overline{\pi}_i, & \text{if } 1 \leqslant i < v_1, \\ \overline{\pi}_{i-1}, & \text{if } v_1 < i < v_2, \\ \overline{\pi}_{i-2}, & \text{if } v_2 < i < v_3, \\ \vdots & \vdots \\ \overline{\pi}_{i-N_{\text{sep}}}, & \text{if } v_{N_{\text{sep}}} < i \leqslant k, \\ 0, & \text{if } i = v_1, \ldots, v_{N_{\text{sep}}}. \end{cases}$$

For example, in Figure 5, $v_1 = 4$, thus $\pi_i = \overline{\pi}_i$ for $1 \leqslant i \leqslant 3$. As the 4th and 7th selected nodes in Figure 5(a) are separate, $\pi_4 = 0$ and $\pi_7 = 0$. When $i > 4$, it is obvious that $\pi_5 = \overline{\pi}_4$, $\pi_6 = \overline{\pi}_5$ and $\pi_8 = \overline{\pi}_6$. Then we can prove this theorem with similar methods in [28], which is omitted here due to the space limitation.

In a separate location vector $\boldsymbol{v} = (v_1, \ldots, v_k)$, there are N_{sep} nonzero components satisfying $v_i < v_{i+1}$ $(1 \leqslant i \leqslant N_{\text{sep}} - 1)$ based on the definitions. In the HC-DSS model, we assume $N_{\text{sep}} \geqslant 1$ and analyse the change of the min-cut when one nonzero component v_j $(1 \leqslant j \leqslant N_{\text{sep}})$ changes, while others remain unchanged. It is obvious that v_j $(1 \leqslant j \leqslant N_{\text{sep}} - 1$) can increase at least one so long as $v_{j+1} \geqslant v_j + 2$, and $v_{N_{\text{sep}}}$ can increase to k at most. For example, in Figure 4(a), the separate location vector is $\boldsymbol{v} = (3, 7, 0, \ldots, 0)$ and $N_{\text{sep}} = 2$. Figure 4(b) shows the repair sequence where v_1 increases by 1 and the new separate location vector changes to $(4, 7, 0, \ldots, 0)$.

Let $\text{MC}^*_{v_j}$ denote the local minimum min-cut corresponding to the cluster order generate by Algorithms 1 and 2 with input separate location vector $\boldsymbol{v}$. Thus

$$\text{MC}^*_{v_j} \triangleq \text{MC}(\boldsymbol{\pi}^*(\boldsymbol{s}^*, \boldsymbol{v})).$$

Note that the subscript of $\text{MC}^*_{v_j}$ could be any nonzero component of $\boldsymbol{v}$, while we only analyse one component v_j at a time. Meanwhile, we use $\text{MC}^*_{v_j+1}$ to represent the min-cut after v_j increasing by 1, in which case the cluster order and separate location vector for $\text{MC}^*_{v_j}$ are changed while the selected node distribution remains unchanged. Assume the new separate location vector is $\overline{\boldsymbol{v}}$, thus $\overline{v}_j = v_j + 1$, and $\overline{v}_t = v_t$ for $t \in [k] \setminus j$ [2)]. Then we have

$$\text{MC}^*_{v_j+1} \triangleq \text{MC}(\boldsymbol{\pi}^*(\boldsymbol{s}^*, \overline{\boldsymbol{v}})).$$

2) $[k]$ represents integer set $\{1, 2, \ldots, k\}$.

Note that the cluster orders and separate location vectors corresponding to $\mathrm{MC}^*_{v_j+1}$ and $\mathrm{MC}^*_{v_j}$ are not expressed explicitly for the simplicity. In the following Lemma 1, $\mathrm{MC}^*_{v_j+1}$ and $\mathrm{MC}^*_{v_j}$ are compared by analyzing the corresponding cluster orders under the following implicit constraints:

- $v_j + 1 < v_{j+1}$ for $1 \leqslant j \leqslant N_{\mathrm{sep}} - 1$,
- $v_{N_{\mathrm{sep}}} \leqslant k$

based on the definition of separate location vector. These constraints are to ensure component v_j can increase by 1 and will not conflict with the definitions.

Lemma 1. In the HC-DSS model, for a given separate location vector $\boldsymbol{v} = (v_1, \ldots, v_k)$ with N_{sep} selected separate nodes, the local minimum min-cuts satisfy that

$$\mathrm{MC}^*_{v_j} \geqslant \mathrm{MC}^*_{v_j+1} \tag{9}$$

for $1 \leqslant j \leqslant N_{\mathrm{sep}}$ under the above implicit constraints.

Proof. Assume the corresponding clusters of $\mathrm{MC}^*_{v_j}$ and $\mathrm{MC}^*_{v_j+1}$ are $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$, respectively. Based on (4), we need to compare

$$\mathrm{MC}^*_{v_j} = \sum_{i=1}^{k} \min\left\{\alpha, w_i(\boldsymbol{\pi})\right\} \text{ and } \mathrm{MC}^*_{v_j+1} = \sum_{i=1}^{k} \min\left\{\alpha, w_i(\boldsymbol{\pi}')\right\}.$$

Then we have

$$\pi_i = \begin{cases} \pi'_{i+1} = 0, & \text{if } i = v_j, \\ \pi'_{i-1}, & \text{if } i = v_j + 1, \\ \pi'_i, & \text{otherwise}, \end{cases} \tag{10}$$

which means that $\pi_i = \pi'_i$ for $i \in [k] \backslash \{v_j, v_j + 1\}$. For example, in Figure 4(a), the separate location vector is $\boldsymbol{v} = (3, 7, 0, \ldots, 0)$ and $N_{\mathrm{sep}} = 2$. When increasing $v_1 = 3$ by one in Figure 4(a), we get the new cluster order $\boldsymbol{\pi}'$ in Figure 4(b). It is obvious that only the two colored nodes 3 and 4 of $\boldsymbol{\pi}'$ in Figure 4(b) are changed comparing with $\boldsymbol{\pi}$. Hence, the part incoming weights satisfy

$$w_i(\boldsymbol{\pi}) = w_i(\boldsymbol{\pi}')$$

for $i \in [k] \backslash \{v_j, v_j + 1\}$. We only need to compare

$$w_{v_j}(\boldsymbol{\pi}) = (d_I + d_C + 1 - v_j)\beta_C,\ w_{v_j+1}(\boldsymbol{\pi}) = a_{v_j+1}(\boldsymbol{\pi})\beta_I + b_{v_j+1}(\boldsymbol{\pi})\beta_C$$

and

$$w_{v_j}(\boldsymbol{\pi}') = a_{v_j}(\boldsymbol{\pi}')\beta_I + b_{v_j}(\boldsymbol{\pi}')\beta_C,\ w_{v_j+1}(\boldsymbol{\pi}') = (d_I + d_C - v_j)\beta_C.$$

Similarly to the situation with one selected separate node proved in [28], we can prove this theorem by enumerating all the possible cases of $w_{v_j}(\boldsymbol{\pi})$, $w_{v_j+1}(\boldsymbol{\pi})$, $w_{v_j}(\boldsymbol{\pi}')$, $w_{v_j+1}(\boldsymbol{\pi}')$ and α, which is omitted here due to the space limitation.

Theorem 2. In the HC-DSS model with N_{sep} ($1 \leqslant N_{\mathrm{sep}} \leqslant E$) selected separate nodes, the separate location vector $\boldsymbol{v}^* = (v_1^*, \ldots, v_k^*)$ minimizes the local minimum min-cut, where

$$v_i^* = \begin{cases} k - N_{\mathrm{sep}} + i, & \text{if } 1 \leqslant i \leqslant N_{\mathrm{sep}}, \\ 0, & \text{if } N_{\mathrm{sep}} + 1 \leqslant i \leqslant k \end{cases} \tag{11}$$

meaning that the N_{sep} separate nodes are at the end of the cluster order.

Proof. For any given cluster order achieving the local minimum min-cut with N_{sep} separate nodes, assume the corresponding separate location vector is $\boldsymbol{v}$ and consider the components of $\boldsymbol{v}$ from $v_{N_{\mathrm{sep}}}$ to v_1 iteratively. Based on the definition of separate location vector in Subsection 2.3, v_i ($1 \leqslant i \leqslant N_{\mathrm{sep}}$) is

at most $k - N_{\text{sep}} + i$ and these values are achieved when the N_{sep} separate nodes are at the end of the cluster order. In this case, the cluster order with ($v_{N_{\text{sep}}} = k$, $v_{N_{\text{sep}}-1} = k-1, \ldots, v_1 = k - N_{\text{sep}} + 1$) minimizes the local minimum min-cut, which is proved iteratively in the following steps.

Step 1. Increase $v_{N_{\text{sep}}}$ one by one until $v_{N_{\text{sep}}} = k$ while the other components remain unchanged. It can be proved that

$$\text{MC}^*_{v_{N_{\text{sep}}}} \geqslant \text{MC}^*_{v_{N_{\text{sep}}}+1} \geqslant \text{MC}^*_{v_{N_{\text{sep}}}+2} \geqslant \cdots \geqslant \text{MC}^*_k \tag{12}$$

based on Lemma 1.

Step 2. Consider $v_{N_{\text{sep}}-1}$ of separate location vector $\boldsymbol{v}$ with $v_{N_{\text{sep}}} = k$ achieving the local minimum min-cut. We can also increase $v_{N_{\text{sep}}-1}$ one by one until $v_{N_{\text{sep}}-1} = k-1$. Base on Lemma 1, we can also get

$$\text{MC}^*_{v_{N_{\text{sep}}-1}} \geqslant \text{MC}^*_{v_{N_{\text{sep}}}} \geqslant \text{MC}^*_{v_{N_{\text{sep}}}+1} \geqslant \cdots \geqslant \text{MC}^*_{k-1}. \tag{13}$$

Step 3. Now consider v_i ($1 \leqslant i \leqslant N_{\text{sep}}$) of separate location vector $\boldsymbol{v}$ with $v_j = k - N_{\text{sep}} + j$ for $i+1 \leqslant j \leqslant N_{\text{sep}}$ achieving the local minimum min-cut. After increasing v_i one by one until $v_i = k - N_{\text{sep}} + i$, we will get

$$\text{MC}^*_{v_i} \geqslant \text{MC}^*_{v_i+1} \geqslant \cdots \geqslant \text{MC}^*_{k-N_{\text{sep}}+i} \tag{14}$$

with Lemma 1. Note that although the expressions of formulas (12)–(14) are very similar, they represents totally different min-cuts of different cluster orders. These expressions are used for simplicity without incurring confusion. For any given cluster order achieving the local minimum min-cut with N_{sep} selected separate nodes and separate location vector $\boldsymbol{v}$, we can always find another cluster order with smaller local minimum min-cut by increasing the components of $\boldsymbol{v}$ until $\boldsymbol{v}^*$ is achieved by the above steps. Hence, we finish the proof.

In Theorem 2, the cluster order corresponding to separate location vector $\boldsymbol{v}^*$ minimizes the local minimum min-cut when the number of selected separate nodes N_{sep} is fixed. Let

$$\boldsymbol{v}^*(j) = (v^*(j)_1, \ldots, v^*(j)_k) \tag{15}$$

denote the separate location vector satisfying (11) with $N_{\text{sep}} = j$. We will compare the min-cuts corresponding to $\boldsymbol{\pi}^*(\boldsymbol{s}^*, \boldsymbol{v}^*(j))$, when j varies in Theorem 3. For simplicity, we define

$$\boldsymbol{\pi}^*(j) \triangleq \boldsymbol{\pi}^*(\boldsymbol{s}^*, \boldsymbol{v}^*(j)), \tag{16}$$

where $0 \leqslant j \leqslant k$ and $\boldsymbol{\pi}^*(\boldsymbol{s}^*, \boldsymbol{v}^*(j))$ is given by (8) with $\boldsymbol{s}^*$ generated by Algorithm 2.

Theorem 3. The capacity of the (n, k, L, R, E) HC-DSS model is achieved by $\boldsymbol{\pi}^*(E)$, meaning that all the separate nodes are selected ($N_{\text{sep}} = E$) and the E selected separate nodes are at the end of $\boldsymbol{\pi}^*(E)$, where the separate location vector $\boldsymbol{v}^*$ satisfies

$$v_i^* = \begin{cases} k - E + i, & \text{if } 1 \leqslant i \leqslant E, \\ 0, & \text{otherwise}, \end{cases} \tag{17}$$

and $\boldsymbol{\pi}^*(E)$ is defined by (16).

Proof. We first compare the min-cuts of $\boldsymbol{\pi}^*(j)$ and $\boldsymbol{\pi}^*(j+1)$, which means that N_{sep} increases by 1. For example in Figure 6(a), the cluster order with two separate nodes is $\boldsymbol{\pi}^*(2) = (1, 2, 1, 2, 1, 1, 0, 0)$. In Figure 6(b), a selected separate node is added and the cluster order changes to $\boldsymbol{\pi}^*(3) = (1, 2, 1, 1, 1, 0, 0, 0)$. In [28], we investigate the situation of adding one separate node to the cluster DSS model, which is inductive to this proof. In order to compare

$$\text{MC}(\boldsymbol{\pi}^*(j)) = \sum_{i=1}^{k} \min\left\{\alpha, w_i(\boldsymbol{\pi}^*(j))\right\} \text{ and } \text{MC}(\boldsymbol{\pi}^*(j+1)) = \sum_{i=1}^{k} \min\left\{\alpha, w_i(\boldsymbol{\pi}^*(j+1))\right\},$$

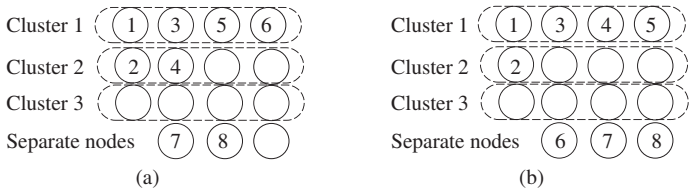

Figure 6 In (a), there are two selected separate node and the cluster order is $\boldsymbol{\pi}^*(2) = (1, 2, 1, 2, 1, 1, 0, 0)$. In (b), a selected separate node is added, and the cluster changes to $\boldsymbol{\pi}^*(3) = (1, 2, 1, 1, 1, 0, 0, 0)$

we also compare $w_i(\boldsymbol{\pi}^*(j))$ and $w_i(\boldsymbol{\pi}^*(j+1))$ for $1 \leqslant i \leqslant k$ similarly to [28]. Base on (5), the part incoming weights satisfy

$$w_i(\boldsymbol{\pi}^*(j)) = w_i(\boldsymbol{\pi}^*(j+1)) = (d+1-i)\beta_C \tag{18}$$

for $k-j+1 \leqslant i \leqslant k$, because the last j selected nodes in $\boldsymbol{\pi}^*(j)$ and $\boldsymbol{\pi}^*(j+1)$ are all separate. For instance, the last two selected nodes (nodes 7 and 8) in Figure 6(a) and (b) are all separate.

Now consider the first $k-j$ selected nodes in $\boldsymbol{\pi}^*(j)$ and $\boldsymbol{\pi}^*(j+1)$. It is easy to verify that this situation is equivalent to adding one selected separate node to the cluster DSS model investigated in [28], where we have proved that

$$\sum_{i=1}^{k-j} w_i(\boldsymbol{\pi}^*(j+1)) \leqslant \sum_{i=1}^{k-j} w_i(\boldsymbol{\pi}^*(j)). \tag{19}$$

Therefore, $\mathrm{MC}(\boldsymbol{\pi}^*(j+1)) \leqslant \mathrm{MC}(\boldsymbol{\pi}^*(j))$ for $1 \leqslant j \leqslant E-1$ by summing (18) and (19). Therefore

$$\mathrm{MC}(\boldsymbol{\pi}^*(E)) \leqslant \cdots \leqslant \mathrm{MC}(\boldsymbol{\pi}^*(1)).$$

In [28], the min-cuts without selected separate nodes are compared with those with only one, and we have proved that $\mathrm{MC}(\boldsymbol{\pi}^*(0)) \geqslant \mathrm{MC}(\boldsymbol{\pi}^*(1))$. Hence, $\mathrm{MC}(\boldsymbol{\pi}^*(E))$ is the system capacity achieved by cluster order $\boldsymbol{\pi}^*(E)$.

In Subsection 3.1, for a given HC-DSS model, we find the cluster order achieving the capacity by comparing the min-cuts step by step. First, we compare the min-cuts of cluster orders for a fixed separate location vector and prove that the local minimum min-cut is achieved by the output cluster order of Algorithms 1 and 2 in Theorem 1. Second, we compare the local minimum min-cuts of different separate location vectors with fixed number of separate nodes, among which the minimum one is achieved by separate location vector $\boldsymbol{v}^*$ satisfying (11) proved by Lemma 1 and Theorem 2. Finally, the global minimum min-cut, namely the capacity, is obtained by comparing the min-cuts of clusters with different amounts of separate nodes, which is proved in Theorem 3.

3.2 Tradeoff between storage and repair bandwidth for the HC-DSS model

As proved in Subsection 3.1, the capacity of a given HC-DSS model is $\mathrm{MC}(\boldsymbol{\pi}^*(E))$ achieved by cluster order $\boldsymbol{\pi}^*(E)$ (see Theorem 3). Thus the tradeoff between storage α and repair bandwidth parameters $(d_I, \beta_I, d_C, \beta_C)$ is formulated with (3) as

$$\mathrm{MC}(\boldsymbol{\pi}^*(E)) = \sum_{i=1}^{k} \min\left\{w_i(\boldsymbol{\pi}^*(E)), \alpha\right\} \geqslant M, \tag{20}$$

which is calculated by (21) and (22), and illustrated in Figure 7.

Let $w_i^* \triangleq w_i(\boldsymbol{\pi}^*(E))$ for simplicity, thus w_i^* is formulated as

$$w_{k-i+1}^* = \begin{cases} \left(R - \left\lceil \frac{i}{\left\lfloor \frac{k-E}{R} \right\rfloor + 1} \right\rceil\right)\beta_I + \left(d_C - i + \left\lceil \frac{i}{\left\lfloor \frac{k-E}{R} \right\rfloor + 1} \right\rceil\right)\beta_C, & 1 \leqslant i \leqslant \left(\lfloor \frac{k-E}{R} \rfloor + 1\right)\left(k - 1 - \lfloor \frac{k-E}{R} \rfloor R\right), \\ \left(\left\lfloor \frac{k-i-E}{\left\lfloor \frac{k-E}{R} \right\rfloor} \right\rfloor\right)\beta_I + \left(d_C + R - i - \left\lfloor \frac{k-i-E}{\left\lfloor \frac{k-E}{R} \right\rfloor} \right\rfloor\right)\beta_C, & \left(\lfloor \frac{k-E}{R} \rfloor + 1\right)\left(k - 1 - \lfloor \frac{k-E}{R} \rfloor R\right) < i \leqslant k - E, \\ (R + d_C - i)\beta_C, & i = (k - E + 1), \ldots, k. \end{cases} \tag{21}$$

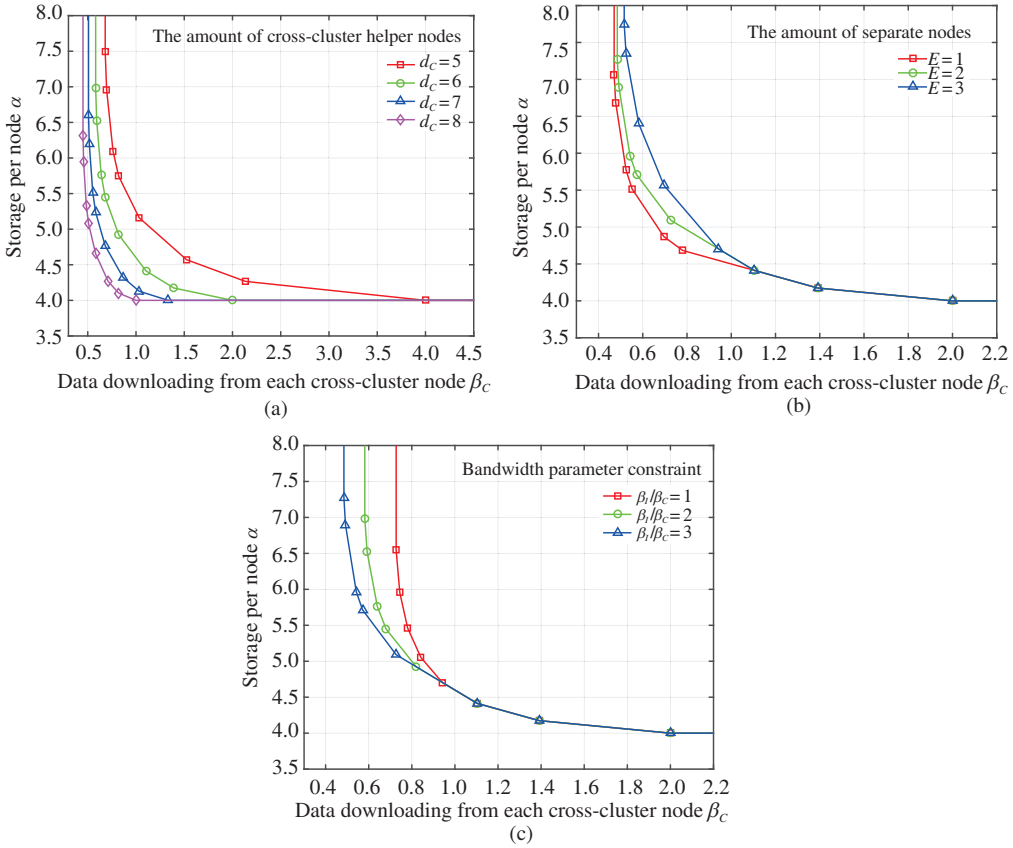

Figure 7 (Color online) Optimal tradeoff bounds between node storage α and repair bandwidth parameter β_C for the $(n = 14, k = 8, L = 3, R = 4, E)$ HC-DSS model with the original file size $\mathcal{M} = 32$. (a) shows the tradeoffs of various d_C when the amount of separate nodes $E = 2$ and the bandwidth parameter constraint $\tau = \beta_I/\beta_C = 2$ are fixed. (b) illustrates the tradeoffs of various E when $d_C =$ and $\tau = 2$ are fixed. (c) compares the tradeoffs of various $\tau = \beta_I/\beta_C$ when $d_C = 6$ and $E = 2$ are fixed.

For ease of illustration, we set the subscript of w^* as $k - i + 1$ such that

$$w_1^* \leqslant w_2^* \leqslant \cdots \leqslant w_k^*$$

without changing the final tradeoff bound. Let α^* denote the minimum α satisfying (20), the tradeoff bound is formulated as follows with similar methods introduced in [6,28]. For $i = 2, \ldots, k$,

$$\alpha^* = \frac{\mathcal{M} - \sum_{j=1}^{i-1} w_j^*}{k - i + 1}, \tag{22}$$

for $\mathcal{M} \in (\sum_{j=1}^{i-1} w_j^* + (k - i + 1)w_{i-1}^*, \sum_{j=1}^{i} w_j^* + (k - i)w_i^*]$. When $i = 1$, $\alpha^* = \mathcal{M}/k$ for $\mathcal{M} \in [0, w_1^*]$.

Remark 1. When $E = 0$ and $E = 1$, the tradeoff bounds for the cluster cases in [17, 22] and the model with clusters and one separate node in [28] can be obtained. When $\beta_I = \beta_C$, the cluster and separate nodes are undifferentiated and the traditional homogeneous DSS model in [6] is retrieved. The corresponding tradeoff bounds are derived with (21) and (22), where $w_{k-i+1} = (R + d_C - i)\beta_C$ for $1 \leqslant i \leqslant k$.

We compare the tradeoff bounds in multiple aspects as shown in Figure 7(a)–(c), respectively. Because the total repair bandwidths of cluster and separate nodes are different ($d_I\beta_I + d_C\beta_C$ and $d\beta_C$, respectively) in the HC-DSS model, we characterize the repair bandwidth by the cross-cluster bandwidth parameter β_C rather than the total repair bandwidth as shown in [6,17]. When the bandwidth parameter constraint $\tau = \beta_I/\beta_C$ is given, the total repair bandwidths of cluster and separate nodes can be obtained with β_C. In Figure 7(a), we show the tradeoff bounds when the amount of cross-cluster helper nodes d_C varies.

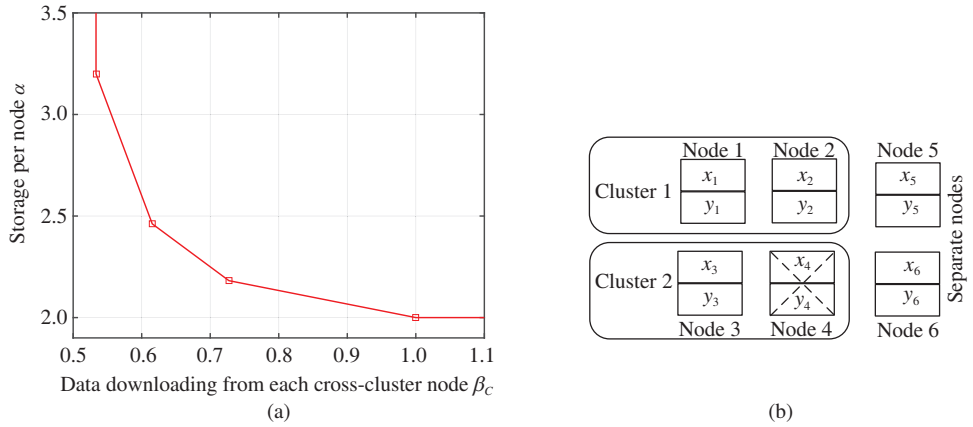

Figure 8 (Color online) In the $(n = 6, k = 4, L = 2, R = 2, E = 2)$ HC-DSS model with $\tau = \beta_I/\beta_C = 2$, $d_I = 1$, $d_C = 4$ and $\mathcal{M} = 8$, (a) illustrates the optimal tradeoff bound between α and β_C, (b) shows the data assignment in the MSR code construction example.

The curves move left as d_C increases, meaning that the capacity increases as the amount of helper nodes increases which is consistent with the conclusion in [6]. In Figure 7(b), the number of separate nodes E changes while other parameters remain unchanged. When E increases, parts of the tradeoff bound curves move right, meaning that the capacity reduces. When the node storage α is fixed (i.e., $\alpha = 5$ in Figure 7(b)), the more separate nodes are (E gets larger), the more repair bandwidth is needed (β_C gets larger). However, when $\alpha = 4$, the repair bandwidth is not affected by E. Figure 7(c) illustrates the influence of bandwidth parameter constraint $\tau = \beta_I/\beta_C$. When $\tau = 1$, we obtain the traditional DSS model in [6] as introduced in Remark 1. When τ increases, more data are downloaded from intra-cluster nodes. For example, when $\alpha = 7$ is fixed, the bigger τ is, the smaller β_C gets.

In Subsection 3.2, the storage and repair bandwidth tradeoff bound is formulated with (21) and (22) for the HC-DSS model. Numerical results are illustrated in Figure 7 in three aspects. In Section 4, a class of regenerating code constructions for the HC-DSS model is investigated.

4 Code constructions for the HC-DSS model

Constructing coding strategies achieving the points in the tradeoff bound is an important issue, where the minimum storage regenerating (MSR) code achieving the MSR point in the tradeoff bound is widely investigated [6,8–12,29]. Product matrix [10] and piggybacking design [8,11,12] are powerful construction techniques based on the idea of interference alignment [33]. We give a class of MSR code constructions for the HC-DSS model with the interference alignment scheme in the following parts.

As shown in Figure 8(a), $(\beta_C = 1, \alpha = 2)$ is the MSR point of the tradeoff curve for the HC-DSS model with storage/repair constraints $\tau = \beta_I/\beta_C = 2$, $d_I = 1$ $d_C = 4$ and original file size $\mathcal{M} = 8$. The encoding and repair procedures are as follows.

Encoding procedure. We represent the original file of size $M = 8$ with x_i $(1 \leqslant i \leqslant 4)$ and y_i $(1 \leqslant i \leqslant 4)$. (x_1, x_2, x_3, x_4) and (y_1, y_2, y_3, y_4) are encoded with two $(6, 4)$-MDS codes as

$$(x_1, \ldots, x_6) = (x_1, \ldots, x_4)[I_{4\times4}|G] \text{ and } (y_1, \ldots, y_6) = (y_1, \ldots, y_4)[I_{4\times4}|F],$$

where $I_{4\times4}$ is the identity matrix. $G = (g_{ij})_{4\times2}$ and $F = (f_{ij})_{4\times2}$ are the encoding matrices. The encoded symbols are stored as Figure 8(b) shows. Then

$$x_5 = g_{11}x_1 + g_{21}x_2 + g_{31}x_3 + g_{41}x_4,\ x_6 = g_{12}x_1 + g_{22}x_2 + g_{32}x_3 + g_{42}x_4,$$
$$y_5 = f_{11}y_1 + f_{21}y_2 + f_{31}y_3 + f_{41}y_4,\ y_6 = f_{12}y_1 + f_{22}y_2 + f_{32}y_3 + f_{42}y_4.$$

Obviously, the six storage nodes satisfy the (6, 4)-MDS property. The encoding matrices G and F need to be designed to satisfy the storage/repair constraints.

Repair procedure. When node 4 is failed, the newcomer need to download 1 symbol from each of the 4 cross-cluster nodes (nodes 1, 2, 5, 6) and 2 symbols from the intra-cluster node 3 based on the storage/repair constraints ($d_C = 4,\ \beta_C = 1,\ d_I = R - 1 = 1,\ \beta_I = 2\beta_C = 2$). Assume the download symbols satisfy that

$$s_1 = r_{11}x_1 + r_{12}y_1,\ s_2 = r_{21}x_2 + r_{22}y_2,\ s_3 = (x_3, y_3),\ s_5 = r_{51}x_5 + r_{52}y_5,\ s_6 = r_{61}x_6 + r_{62}y_6,$$

where r_{i1} and r_{i2} are called the repair download parameters, and s_i ($1 \leqslant i \leqslant 6,\ i \neq 4$) represents the symbols downloading from node i. Note that s_3 consists of two symbols as node 3 is an intra-cluster node. The newcomer acquires the following equations:

$$\begin{cases} s_1 = r_{11}x_1 + r_{12}y_1, \\ s_2 = r_{21}x_2 + r_{22}y_2, \\ s_5 - r_{51}g_{31}x_3 - r_{52}f_{31}y_3 = r_{51}g_{11}x_1 + r_{52}f_{11}y_1 + r_{51}g_{21}x_2 + r_{52}f_{21}y_2 + r_{51}g_{41}x_4 + r_{52}f_{41}y_4, \\ s_6 - r_{61}g_{32}x_3 - r_{62}f_{32}y_3 = r_{61}g_{12}x_1 + r_{62}f_{12}y_1 + r_{61}g_{22}x_2 + r_{62}f_{22}y_2 + r_{61}g_{42}x_4 + r_{62}f_{42}y_4, \end{cases} \tag{23}$$

where the left parts of equations are known values and the right parts are unknown variables. Hence, if the coefficients of x_i and y_i ($i = 1, 2, 4$) satisfy that

$$\mathbf{rank}\left(\begin{bmatrix} r_{11} & r_{12} \\ r_{51}g_{11} & r_{52}f_{11} \\ r_{61}g_{12} & r_{62}f_{12} \end{bmatrix}\right) = 1,\ \mathbf{rank}\left(\begin{bmatrix} r_{21} & r_{22} \\ r_{51}g_{21} & r_{52}f_{21} \\ r_{61}g_{22} & r_{62}f_{22} \end{bmatrix}\right) = 1 \text{ and } \mathbf{rank}\left(\begin{bmatrix} r_{51}g_{41} & r_{52}f_{41} \\ r_{61}g_{42} & r_{62}f_{42} \end{bmatrix}\right) = 2, \tag{24}$$

the variables x_1, y_1, x_2, y_2 are eliminated and x_4, y_4 are figured out. If another node is failed, we can use similar methods to repair it, but the repair download parameters and encoding matrices G, F need to be designed deliberately to recover all possible node failures. As proved in [29], there exist MDS codes and repair download parameters satisfying (24). The construction of G and F involves more matrix techniques and will not introduced here.

From this MSR construction instance, we can find that regenerating codes for the traditional DSSs can be generalized to the HC-DSS model where the downloading constraints are relaxed as the newcomer could download more data from intra-cluster nodes. Since the node parameters of the HC-DSS model vary as the number of separate nodes changes, finding general code constructions is complicated and remains a problem, where the tradeoff bound is inductive for the HC-DSS model with fixed amount of separate nodes. Based on the tradeoff bound, we can also investigate the code construction problem from the perspective of reducing the cross-cluster bandwidth, as introduced in [18]. Meanwhile, more future studies are needed to construct efficient regenerating codes which is adaptive to the structure of HC-DSSs.

5 Conclusion

In this paper, we investigate the heterogeneous cluster DSS model, where the capacity is characterized. Additionally, the tradeoff bounds are formulated and analyzed in multiple aspects when the repair bandwidth constraints and the amount of separate nodes change, respectively. An MSR coding strategy for the HC-DSS model is constructed with the interference alignment scheme.

Acknowledgements This work was partially supported by National Natural Science Foundation of China (Grant No. 61571293), China Program of International S&T Cooperation (Grant No. 2016YFE0100300), and SJTU-CUHK Joint Research Collaboration Fund 2018.

References

1 Hu Y C, Li X L, Zhang M, et al. Optimal repair layering for erasure-coded data centers. ACM Trans Storage, 2017, 13: 1–24

2 Huang C, Chen M H, Li J. Pyramid codes: flexible schemes to trade space for access efficiency in reliable data storage systems. ACM Trans Storage, 2013, 9: 3
3 Zhang H Y, Li H, Li S Y R. Repair tree: fast repair for single failure in erasure-coded distributed storage systems. IEEE Trans Parallel Distrib Syst, 2017, 28: 1728–1739
4 Balaji S B, Krishnan M N, Vajha M, et al. Erasure coding for distributed storage: an overview. Sci China Inf Sci, 2018, 61: 100301
5 Hou H X, Han Y S. A class of binary MDS array codes with asymptotically weak-optimal repair. Sci China Inf Sci, 2018, 61: 100302
6 Dimakis A G, Godfrey P B, Wu Y N, et al. Network coding for distributed storage systems. IEEE Trans Inform Theor, 2010, 56: 4539–4551
7 Li S Y R, Yeung R W, Cai N. Linear network coding. IEEE Trans Inform Theor, 2003, 49: 371–381
8 Yang B, Tang X, Li J. A systematic piggybacking design for minimum storage regenerating codes. IEEE Trans Inform Theor, 2015, 61: 5779–5786
9 Goparaju S, Fazeli A, Vardy A. Minimum storage regenerating codes for all parameters. IEEE Trans Inform Theor, 2017, 63: 6318–6328
10 Rashmi K V, Shah N B, Kumar P V. Optimal exact-regenerating codes for distributed storage at the MSR and MBR points via a product-matrix construction. IEEE Trans Inform Theor, 2011, 57: 5227–5239
11 Rashmi K V, Shah N B, Ramchandran K. A piggybacking design framework for read-and download-efficient distributed storage codes. IEEE Trans Inform Theor, 2017, 63: 5802–5820
12 Tang X, Yang B, Li J, et al. A new repair strategy for the hadamard minimum storage regenerating codes for distributed storage systems. IEEE Trans Inform Theor, 2015, 61: 5271–5279
13 Ford D, Labelle F, Popovici F I, et al. Availability in globally distributed storage systems. In: Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation, Vancouver, 2010. 61–74
14 Kubiatowicz J, Bindel D, Chen Y, et al. OceanStore: an architecture for global-scale persistent storage. ACM SIGOPS Oper Syst Rev, 2000, 34: 190–201
15 Ahmad F, Chakradhar S T, Raghunathan A, et al. Shufflewatcher: shuffle-aware scheduling in multi-tenant mapreduce clusters. In: Proceedings of 2014 USENIX Annual Technical Conference, Philadelphia, 2014. 1–13
16 Prakash N, Abdrashitov V, Medard M. The storage versus repair-bandwidth trade-off for clustered storage systems. IEEE Trans Inform Theor, 2018, 64: 5783–5805
17 Sohn J Y, Choi B, Yoon S W, et al. Capacity of clustered distributed storage. IEEE Trans Inform Theor, 2019, 65: 81–107
18 Hou H X, Lee P P C, Shum K W, et al. Rack-aware regenerating codes for data centers. IEEE Trans Inform Theor, 2019, 65: 4730–4745
19 Hu Y C, Lee P P C, Zhang X. Double regenerating codes for hierarchical data centers. In: Proceedings of 2016 IEEE International Symposium on Information Theory (ISIT), 2016. 245–249
20 Abdrashitov V, Prakash N, Médard M. The storage vs repair bandwidth trade-off for multiple failures in clustered storage networks. In: Proceedings of 2017 IEEE Information Theory Workshop (ITW), 2017. 46–50
21 Akhlaghi S, Kiani A, Ghanavati M R. Cost-bandwidth tradeoff in distributed storage systems. Comput Commun, 2010, 33: 2105–2115
22 Wang J Z, Wang T H, Luo Y. Storage and repair bandwidth tradeoff for distributed storage systems with clusters and separate nodes. Sci China Inf Sci, 2018, 61: 100303
23 Pernas J, Yuen C, Gastón B, et al. Non-homogeneous two-rack model for distributed storage systems. In: Proceedings of 2013 IEEE International Symposium on Information Theory (ISIT), 2013. 1237–1241
24 Sohn J Y, Choi B, Yoon S W, et al. Capacity of clustered distributed storage. In: Proceedings of 2017 IEEE International Conference on Communications (ICC), Paris, 2017
25 Ernvall T, El Rouayheb S, Hollanti C, et al. Capacity and security of heterogeneous distributed storage systems. IEEE J Sel Areas Commun, 2013, 31: 2701–2709
26 Golrezaei N, Dimakis A G, Molisch A F. Wireless device-to-device communications with distributed caching. In: Proceedings of IEEE International Symposium on Information Theory, Cambridge, 2012
27 Yu Q, Shum K W, Sung C W. Tradeoff between storage cost and repair cost in heterogeneous distributed storage systems. Trans Emerging Tel Tech, 2015, 26: 1201–1211
28 Wang J Z, Wang T H, Luo Y, et al. Capacity of distributed storage systems with clusters and separate nodes. 2019. ArXiv: 1901.03000
29 Dimakis A G, Ramchandran K, Wu Y N, et al. A survey on network codes for distributed storage. Proc IEEE, 2011, 99: 476–489
30 Ahlswede R, Cai N, Li S Y R, et al. Network information flow. IEEE Trans Inform Theor, 2000, 46: 1204–1216
31 Ahuja R K, Magnanti T L, Orlin J B. Network Flows: Theory, Algorithms, and Applications. Upper Saddle River: Prentice Hall, 1993. 77–78
32 Zorgui M, Wang Z Y. Code constructions for multi-node exact repair in distributed storage. Sci China Inf Sci, 2018, 61: 100304
33 Wu Y N, Dimakis A G. Reducing repair traffic for erasure coding-based storage via interference alignment. In: Proceedings of 2009 IEEE International Symposium on Information Theory (ISIT), 2009. 2276–2280