Map Phase: Each node k, using the Map functions, computes all IVs from a set of locally available files $\mathcal{M}_k \subseteq \{w_1, \ldots, w_N\}$. That is for each $w_n \in \mathcal{M}_k$, node k computes $v_{1,n}, \ldots, v_{Q,n}$.

Shuffle Phase: Each node k aims to collect all IVs $v_{q,1}, \ldots, v_{q,N}$ for each assigned output function $\phi_q : q \in \mathcal{W}_k \subseteq [Q]$ where $\mathcal{W}_k$ is the set of indexes of the output functions assigned to node k. The nodes transmit codewords derived from locally computed IVs such that, after the shuffle phase, each node k can recover all necessary IVs of the assigned output functions.

Reduce Phase: With the collected IVs from the shuffle phase, each node k uses Reduce function h_q to compute $u_q = h_q(v_{q,1}, \ldots, v_{q,N})$ for $q \in \mathcal{W}_k$.

Assume that each node computes all Q IVs from each of its locally available file. We define the computation load of the system as $r = \frac{1}{N}\sum_{k=1}^{K} |\mathcal{M}_k|$, which is the total number of IVs computed normalized by QN. Note that r represents both the average number of times that an IV is computed, and the average number of times that a file is mapped across the network. Then, the communication load L, where $0 \leq L \leq 1$, is defined as the total number of bits transmitted by all nodes normalized by the total number of bits of all IVs, QNT. In other words, $L = \frac{1}{QNT}\sum_{k=1}^{K} l_k$, where l_k is the number of bits transmitted by node k.

Moreover, in this work, we assume that each Reduce function is assigned to an average of s nodes, where $s = \frac{1}{Q}\sum_{k=1}^{K} \mathcal{W}_k$. We say (r, s, L) is feasible if there exist 1) a file mapping such that each file is mapped to an average of r nodes 2) a function assignment such that each function is assigned to an average of s nodes and 3) a shuffle phase design such that at most $QNTL$ bits are transmitted on the multicast channel and each node can decode all requested IVs for the assigned fucntions. Then, we define the optimal communication load as follows.

Definition 1: The optimal communication load is defined as

$$L^*(r, s) \triangleq \inf\{L : (r, s, L) \text{ is feasible}\}. \tag{2}$$

Remark 1: In general, r and s can take different values. However, in this work, we consider the case where $r = s$. This is motivated by the fact that r and s can be a design choice. For example, in [3], for $s = 1$, the authors discuss picking an optimal r to reduce the overall execution time on a real distributed computing platform. For $s > 1$, we note that, while allowing more (r, s) pairs may help optimize the execution time, limiting the pairs to $r = s$ can still leave a sufficient number of viable operating points. Hence, in the remainder of the paper, we will focus on the case where $r = s$. Nevertheless, we will keep both variables r and s because they each have a different meaning.

III. Hypercuboid Approach for Cascaded CDC

In this section, we present the proposed combinatorial design for general cascaded CDC networks that apply to both heterogeneous and homogeneous networks. We will begin with a simpler, two-dimensional example to introduce the basic ideas of the proposed approach. This is followed by a description of the general scheme that includes four key components: Generalized Node Grouping, Node Group Mapping, Cascaded Function Mapping, and Multi-round Shuffle Phase. We then present two three-dimensional examples of the proposed hypercuboid design, one for a homogeneous network, and one for a heterogeneous network, to further illustrate details of the proposed design and compute the achievable communication rates.

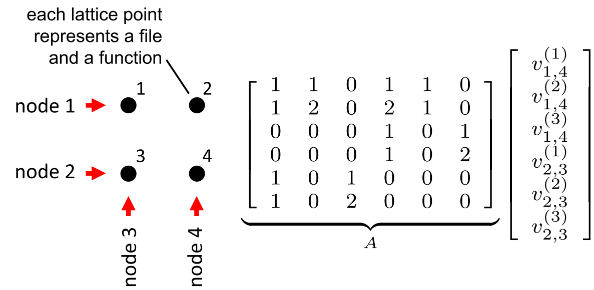


Fig. 1. (left) Lattice points that represent the file mapping and function assignment of the hypercube design with $r = 2$, $s = 2$ and $K = 4$. Each lattice point represents a file and a function and each node maps files and is assigned files based on a line of lattice points. (right) The linear combinations of packets received by node 1 in round 2 after cancelling out locally computed packets.

A. 2-Dimensional Homogeneous Example

Example 1: Consider $K = 4$ nodes that map $N = 4$ files and are assigned to compute $Q = 4$ functions. Fig. 1 and Table I show the file mapping and functions assignment. In Fig. 1, the nodes are aligned along a 2-by-2 lattice and then horizontal or vertical lines define the mapping and assignment at the nodes. For instance, node 1 maps files w_1 and w_2 and is assigned functions 1 and 2 represented by the top horizontal line of lattice points. Similarly, node 3 maps files w_1 and w_3 and is assigned functions 1 and 3 represented by the left vertical line of lattice points. As each lattice point intersects 2 lines, one vertical and one horizontal, then we find each file is mapped at 2 nodes, $r = 2$, and each function is assigned to 2 nodes, $s = 2$.

In the Map phase each node computes all IVs from each locally available file as shown in the third column of Table I. For example, node 1 computes $v_{1,1}$, $v_{2,1}$, $v_{3,1}$, and $v_{4,1}$ from file w_1 and $v_{1,2}$, $v_{2,2}$, $v_{3,2}$, and $v_{4,2}$ from file w_2. The IVs can be classified by the number of nodes that request them in the Shuffle phase. For instance, IV $v_{1,1}$ is only needed by nodes 1 and 3, but these nodes computes this IV from the locally available file w_1. Therefore, we say $v_{1,1}$ is requested by 0 nodes. Similarly, $v_{2,2}$, $v_{3,3}$, and $v_{4,4}$ are requested by 0 nodes. Then, since nodes 1 and 3 are the only nodes that are assigned function 1, we see that $v_{1,3}$ is only requested by one node (node 1) because w_3 is available at node 3, but not at node 1. Similarly, $v_{1,2}$ is only requested by one node (node 3) because w_2 is available at node 1, but not at node 3. On the other hand, $v_{1,4}$ is requested by 2 nodes, nodes 1 and 3, because neither node maps file w_4.

There are 2 rounds in the Shuffle phase where the nodes shuffle the IVs that are requested by 1 and 2 nodes,

TABLE I
FILES, FUNCTIONS, IVS AND TRANSMISSIONS OF EXAMPLE 1

node	files	computed IVs	funcs.	requested IVs	round 1	round 2
1	w_1	$v_{1,1}, v_{2,1}, v_{3,1}, v_{4,1}$	ϕ_1	$v_{1,3}, v_{1,4}$	$v_{1,2} \oplus v_{2,1}$	$v_{3,2}^{(1)} + v_{3,2}^{(2)} + v_{4,1}^{(1)} + v_{4,1}^{(2)}$
	w_2	$v_{1,2}, v_{2,2}, v_{3,2}, v_{4,2}$	ϕ_2	$v_{2,3}, v_{2,4}$		$v_{3,2}^{(1)} + 2v_{3,2}^{(2)} + 2v_{4,1}^{(1)} + v_{4,1}^{(2)}$
2	w_3	$v_{1,3}, v_{2,3}, v_{3,3}, v_{4,3}$	ϕ_3	$v_{3,1}, v_{3,2}$	$v_{3,4} \oplus v_{4,3}$	$v_{1,4}^{(1)} + v_{1,4}^{(2)} + v_{2,3}^{(1)} + v_{2,3}^{(2)}$
	w_4	$v_{1,4}, v_{2,4}, v_{3,4}, v_{4,4}$	ϕ_4	$v_{4,1}, v_{4,2}$		$v_{1,4}^{(1)} + 2v_{1,4}^{(2)} + 2v_{2,3}^{(1)} + v_{2,3}^{(2)}$
3	w_1	$v_{1,1}, v_{2,1}, v_{3,1}, v_{4,1}$	ϕ_1	$v_{1,2}, v_{1,4}$	$v_{1,3} \oplus v_{3,1}$	$v_{2,3}^{(1)} + v_{2,3}^{(3)} + v_{4,1}^{(1)} + v_{4,1}^{(3)}$
	w_3	$v_{1,3}, v_{2,3}, v_{3,3}, v_{4,3}$	ϕ_3	$v_{3,2}, v_{3,4}$		$v_{2,3}^{(1)} + 2v_{2,3}^{(3)} + 2v_{4,1}^{(1)} + v_{4,1}^{(3)}$
4	w_2	$v_{1,2}, v_{2,2}, v_{3,2}, v_{4,2}$	ϕ_2	$v_{2,1}, v_{2,3}$	$v_{2,4} \oplus v_{4,2}$	$v_{1,4}^{(1)} + v_{1,4}^{(3)} + v_{3,2}^{(1)} + v_{3,2}^{(3)}$
	w_4	$v_{1,4}, v_{2,4}, v_{3,4}, v_{4,4}$	ϕ_4	$v_{4,1}, v_{4,3}$		$v_{1,4}^{(1)} + 2v_{1,4}^{(3)} + 2v_{3,2}^{(1)} + v_{3,2}^{(3)}$

respectively. The messages transmitted by each node are shown in last two columns of Table I. In round 1, each node includes 2 IVs in a coded message to 2 other nodes. For instance, node 1 computes $v_{1,2}$ and $v_{2,1}$, where $v_{1,2}$ is requested by node 3 and is available at node 4, and the opposite is true for $v_{2,1}$. Therefore, node 1 transmits $v_{1,2} \oplus v_{2,1}$ to nodes 3 and 4.[1] In this example, each node transmits a coded message to serve 2 independent requests of nodes aligned along the other dimension.

In round 2, we consider all IVs requested by 2 nodes which are $v_{1,4}$, $v_{4,1}$, $v_{2,3}$ and $v_{3,2}$. These IVs are each available at the 2 nodes that do not request them. Each IV is split into 3 disjoint equally sized packets. For instance, for the IVs requested by node 1, $v_{1,4}$ is split into $v_{1,4}^{(1)}$, $v_{1,4}^{(2)}$ and $v_{1,4}^{(3)}$ and $v_{2,3}$ is split into $v_{2,3}^{(1)}$, $v_{2,3}^{(2)}$ and $v_{2,3}^{(3)}$. Each node sends two linear combinations of its available packets.[2] Accordingly, each node will receive a total of 6 linear combinations to solve for the 6 requested packets. The linear combinations are shown in the last column of Table I. We can see, for instance, after subtracting out available packets, node 1 receives the linear combinations shown by the matrix-vector multiplication on the right side of Fig. 1. Since, the matrix A is invertible, node 1 can solve for its requested packets and therefore all requested IVs. The messages of round 2 are deliberately designed so that the received messages at each node can be represented by a full rank matrix similar to matrix A for node 1. One can verify from the the table of Fig. 1, that each node can recover all requested packets from round 2.

We compute the communication load, $L_{\rm c}$, by counting all transmitted messages and considering their size. There are 4 messages of length T bits (size of a single IV), and 8 messages of length $\frac{T}{3}$ bits. After normalizing by the total bits over all IVs, QNT, the communication load is

$$L_{\rm c} = \frac{1}{QNT}\left(4T + 8\frac{T}{3}\right) = \frac{1}{16}\left(4 + \frac{8}{3}\right) \approx 0.417. \quad (3)$$

With an equivalent, r, s and K, we can compare $L_{\rm c}$ to the fundamental bound and scheme of [3] where the communication load is

$$L_1 = \frac{3\binom{K}{3}\binom{1}{r-1}\binom{r}{3-s}}{r\binom{K}{r}\binom{K}{s}} + \frac{4\binom{K}{4}\binom{2}{r-1}\binom{r}{4-s}}{r\binom{K}{r}\binom{K}{s}}$$
$$= \frac{3\cdot 4\cdot 1\cdot 2}{72} + \frac{4\cdot 1\cdot 2\cdot 1}{72} = \frac{32}{72} \approx 0.444. \quad (4)$$

Ultimately, we find $L_{\rm c} < L_1$, and our new design has a reduced communication load.

Remark 2: To the best of our knowledge, this is the first homogeneous example of CDC that has a communication load less than L_1. In [3], L_1 was shown to be the smallest achievable communication load given r, s and K, under an implicit assumption of the reduce function assignment that *every* set of s nodes must have have a common reduce function. This assumption was made in the proof of [3, Theorem 2]. Note that our proposed design does not impose such an assumption. For instance, neither node pairs $\{1, 2\}$ nor $\{3, 4\}$ in Example 1 have a shared assigned function. Example 1 shows that the more general function assignment proposed in this work allows us to achieve a lower communication load that is less than L_1, even for homogeneous networks. Similar observations were made for a heterogeneous network with $s = 1$ in [10], [11].

B. General Achievable Scheme

Next, we present the proposed general achievable scheme and describe its four key components. in detail.

1) Generalized Node Grouping: The Generalized Node Grouping lays the foundation of the proposed hypercuboid design. It consists of Single Node Grouping (equivalent to Node Grouping 2 in [9]), and Double Node Grouping. The latter is specifically designed for the cascaded CDC networks considered in this work.

Consider a general network of K nodes with varying storage capacity. To define a hypercuboid structure for this network, divide these K nodes into P disjoint sets, $\mathcal{C}_1, \ldots, \mathcal{C}_P$, each of size $|\mathcal{C}_p|$ and $\sum_{p=1}^{P} |\mathcal{C}_p| = K$. Nodes in the same set $\mathcal{C}_p$ have the same storage capacity and each stores $1/m_p$ of the entire file library. Furthermore, assume that nodes in $\mathcal{C}_p$ map the library r_p times so that $\sum_{p=1}^{P} r_p = r$. Apply the hypercube design in [9] to each $\mathcal{C}_p$ by splitting nodes in $\mathcal{C}_p$ into r_p disjoint subsets $\{\mathcal{K}_i, i \in \mathcal{I}_p\}$ of equal size m_p, where $|\mathcal{C}_p| = r_p m_p$, and the index set $\mathcal{I}_p = \{i : n_{p-1} + 1 \leq i \leq n_p\}$ and $n_j = \sum_{i=1}^{j} r_i, j \in [P]$. The entire network is comprised of r node

[1] Note that, each IV corresponds to an element in the field $\mathbb{F}_{2^T}$. To perform the bit-wise XOR operation, "$\oplus$", we translate each IV into an array of T bits in the field $\mathbb{F}_2^T$ and then perform XOR with each bit of each IV.

[2] We assume that T is large enough to allow for a large range of coefficients in the linear combinations.

sets, $\mathcal{K}_1, \ldots, \mathcal{K}_r$. Nodes in $\mathcal{K}_i, i \in [r]$ are aligned along the i-th dimension of the hypercuboid, and they collectively map the library exactly once.

2) Single Node Grouping: Given a subset $\mathcal{A} \subset [r]$, we say that $\mathcal{S} \subset [K]$ is an $(\mathcal{A}, 1)$ node group if it contains exactly one node from each $\mathcal{K}_i$, i.e., $|\mathcal{S} \cap \mathcal{K}_i| = 1$, for every $i \in \mathcal{A}$. In particular, consider all possible $([r], 1)$ node groups $\mathcal{T}_1, \ldots, \mathcal{T}_X$ of size r that each contains a single node from every node set $\mathcal{K}_1, \ldots, \mathcal{K}_r$, here $X = \prod_{i=1}^{r} |\mathcal{K}_i| = \prod_{p=1}^{P} m_p^{r_p}$. Denote $\mathcal{T}_{j,i} = \mathcal{T}_j \cap \mathcal{K}_i$, $\forall j \in [X]$ and $\forall i \in [r]$, as the node in $\mathcal{T}_j$ that is chosen from $\mathcal{K}_i$.

3) Double Node Grouping: Given a subset $\mathcal{A} \subset [r]$, we say that $\mathcal{S} \subset \mathcal{K}$ is an $(\mathcal{A}, 2)$ node group if it contains exactly two nodes from each $\mathcal{K}_i$, i.e., $|\mathcal{S} \cap \mathcal{K}_i| = 2$, for every $i \in \mathcal{A}$. Hence, the size of an $(\mathcal{A}, 2)$ node group is $|\mathcal{S}| = 2 \cdot |\mathcal{A}|$. Double Node Grouping is essential for the design of the Multi-round Shuffle phase.

4) Node Group (NG) File Mapping: Given all $([r], 1)$ node groups $\mathcal{T}_1, \ldots, \mathcal{T}_X$, we split the N files into X disjoint sets labeled as $\mathcal{B}_1, \ldots, \mathcal{B}_X$. These file sets are of size $\eta_1 \in \mathbb{Z}^+$ and $N = \eta_1 X$. Each file set $\mathcal{B}_i$ is only available to every node in the node group $\mathcal{T}_i$. It follows that if node $k \in [K]$ belongs to a node group $\mathcal{T}_i$, then the file set $\mathcal{B}_i$ is available to this node. Hence, by considering all possible node groups $\mathcal{T}_i$ that node k belongs to, its available files, denoted by $\mathcal{M}_k$, is expressed as

$$\mathcal{M}_k := \bigcup_{i: k \in \mathcal{T}_i} \mathcal{B}_i. \tag{5}$$

Note that, since each file belongs to a unique file set $\mathcal{B}_i$ and is mapped to a unique set of r nodes (in the node group $\mathcal{T}_i$), we must have $\frac{1}{N} \sum_{k=1}^{K} |\mathcal{M}_k| = \frac{Nr}{N} = r$.

The function assignment is defined as follows:

5) Cascaded Function Assignment: Given all $([r], 1)$ node groups $\mathcal{T}_1, \ldots, \mathcal{T}_X$, the Q files are split into X disjoint sets labeled as $\mathcal{D}_1, \ldots, \mathcal{D}_X$ and file set $\mathcal{D}_i$ is assigned exclusively to nodes of set $\mathcal{T}_i$. These function sets are of size $\eta_2 \in \mathbb{Z}^+$ and $Q = \eta_2 X$. For $k \in [K]$, define

$$\mathcal{W}_k := \bigcup_{i: k \in \mathcal{T}_i} \mathcal{D}_i \tag{6}$$

as the set of functions assigned to node k.

Remark 3: Note that the proposed Cascaded Function Assignment follows the same design principle as that of the NG File Mapping. As each file is mapped to r nodes in the network, the proposed design ensures that each reduce function is also mapped to r nodes. Thus, we assume that $r = s$ in our design. The proposed Cascaded Function Assignment serves as a building block for consecutive rounds of MapReduce iterations, where the reduce function outputs become the file inputs for the next iteration.

6) Map Phase: Each node $k \in [K]$ computes the set of IVs $\{v_{i,j} : i \in [Q], w_j \in \mathcal{M}_k\}$.

7) Multi-Round (MR) Shuffle Phase: We consider a Multi-round Shuffle Phase with r rounds, where in each round we use one of two methods termed the Inter-group (IG) Shuffle Method and the Linear Combination (LC) Shuffle Method. In the γ-th round, the nodes exchange IVs requested by γ nodes. The IG Shuffle Method is designed for $1 \le \gamma \le r - 1$ and forms groups of 2γ nodes. A node outside of each node group multicasts coded pairs of IVs to this node group. For the LC Shuffle Method, nodes also form groups of 2γ nodes; however, nodes of this group multicast linear combinations of packets among one another. The LC Shuffle Method is designed only for the r-th round.

8) Inter-Group (IG) Shuffle Method ($1 \le \gamma \le r - 1$): Consider $\mathcal{A} \subset [r]$ such that $|\mathcal{A}| = \gamma$. For each $\mathcal{A}$, let $\mathcal{S}$ be a $(\mathcal{A}, 2)$-node group with $|\mathcal{S}| = 2\gamma$ and $\mathcal{S}' \subset \mathcal{S}$ be a $(\mathcal{A}, 1)$-node group with $|\mathcal{S}'| = \gamma$. Assume $\mathcal{A}^c = [r] \setminus \mathcal{A}$ and let $\mathcal{Y}$ be a $(\mathcal{A}^c, 1)$-node group with $|\mathcal{Y}| = r - \gamma$. An arbitrary node in $\mathcal{Y}$ will multicast a summation of two sets of IVs, one for nodes in $\mathcal{S}'$ and one for nodes in $\mathcal{S} \setminus \mathcal{S}'$. To ensure that each node in $\mathcal{S}'$ (or $\mathcal{S} \setminus \mathcal{S}'$) can decode successfully from the multicast message, the set of IVs intended for nodes in $\mathcal{S}'$ (or $\mathcal{S} \setminus \mathcal{S}'$) must be available to nodes in $\mathcal{S} \setminus \mathcal{S}'$ (or $\mathcal{S}'$). To determine these IVs, letting $\mathcal{T}_\alpha = \{\mathcal{S} \setminus \mathcal{S}'\} \cup \mathcal{Y}$ and $\mathcal{T}_\ell = \mathcal{S}' \cup \mathcal{Y}$, we define

$$\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{S} \setminus \mathcal{S}'} := \{v_{i,j} : i \in \mathcal{D}_\alpha, w_j \in \mathcal{B}_\ell\} \tag{7}$$

and

$$\mathcal{V}_{\mathcal{T}_\alpha}^{\mathcal{S}'} = \{v_{i,j} : i \in \mathcal{D}_\ell, w_j \in \mathcal{B}_\alpha\Big\}. \tag{8}$$

By the definition of the NG File Mapping, nodes in $\mathcal{T}_\ell$ have access to files in $\mathcal{B}_\ell$. However, since nodes in $\mathcal{S} \setminus \mathcal{S}'$ are not in $\mathcal{T}_\ell$, they do not have access to files in $\mathcal{B}_\ell$. Thus, the set $\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{S} \setminus \mathcal{S}'}$ contains IVs that are requested by nodes in $\mathcal{S} \setminus \mathcal{S}'$ and can be computed at every node in $\mathcal{T}_\ell$. Similarly, $\mathcal{V}_{\mathcal{T}_\alpha}^{\mathcal{S}'}$ contains IVs that are requested by nodes in $\mathcal{S}'$ and can be computed at every node in $\mathcal{T}_\alpha$. The IVs of $\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{S} \setminus \mathcal{S}'}$ are concatenated to form the message $U_{\mathcal{T}_\ell}^{\mathcal{S} \setminus \mathcal{S}'}$ and similarly the IVs of $\mathcal{V}_{\mathcal{T}_\alpha}^{\mathcal{S}'}$ are concatenated to form the message $U_{\mathcal{T}_\alpha}^{\mathcal{S}'}$. For all possible choices of $\mathcal{A}, \mathcal{S}, \mathcal{S}', \mathcal{Y}$, an arbitrary node in $\mathcal{Y}$ multicasts

$$U_{\mathcal{T}_\ell}^{\mathcal{S} \setminus \mathcal{S}'} \oplus U_{\mathcal{T}_\alpha}^{\mathcal{S}'} \tag{9}$$

to the 2γ nodes in $\mathcal{S}$.

9) Linear Combination (LC) Shuffle Method ($\gamma = r$): This shuffle method is used for the r-th round only. Let $\mathcal{S}$ denote a $([r], 2)$-node group with $|\mathcal{S}| = 2r$. Each node $k \in \mathcal{S}$ will multicast linear combinations of IVs to the other $2r - 1$ nodes in $\mathcal{S}$. These IVs are defined as follows. Given $k \in \mathcal{S}$, let $\mathcal{T}_\ell$ denote a $([r], 1)$-node group such that $k \in \mathcal{T}_\ell \subset \mathcal{S}$. Let $\mathcal{T}_{\alpha_\ell} = \mathcal{S} \setminus \mathcal{T}_\ell$, which is also a $([r], 1)$-node group. Define

$$\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{T}_{\alpha_\ell}} = \{v_{i,j} : i \in \mathcal{D}_{\alpha_\ell}, w_j \in \mathcal{B}_\ell\}, \tag{10}$$

which are IVs requested by the r nodes in $\mathcal{T}_{\alpha_\ell}$ and are available at the r nodes in $\mathcal{T}_\ell$. For each $\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{T}_{\alpha_\ell}}$ we concatenate the IVs of this set into the message $U_{\mathcal{T}_\ell}^{\mathcal{T}_{\alpha_\ell}}$ and split it into $2r - 1$ equal size, disjoint sub-messages denoted by $U_{\mathcal{T}_\ell,1}^{\mathcal{T}_{\alpha_\ell}}, \ldots, U_{\mathcal{T}_\ell,2\ r-1}^{\mathcal{T}_{\alpha_\ell}}$. Let $\mathcal{L}_k = \{\ell : k \in \mathcal{T}_\ell \subset \mathcal{S}\}$. Then node k multicasts $2^{(r-1)}$ random linear combinations of the sub-messages in

$$\bigcup_{\ell \in \mathcal{L}_k} \bigcup_{i \in [2r-1]} U_{\mathcal{T}_\ell,i}^{\mathcal{T}_{\alpha_\ell}} \tag{11}$$

to the other $2r - 1$ nodes in $\mathcal{S}$.[3]

[3]In Appendix B, we prove that each node can decode its requested sub-messages with high probability.

10) Reduce Phase: For all $k \in [K]$, node k computes all output values u_q such that $q \in \mathcal{W}_k$.

Remark 4: The two Shuffle methods each have their own advantages. With the IG Shuffle Method, as shown in (9), a node outside a node group $\mathcal{S}$ transmits a coded message containing 2 IVs, one intended for γ nodes in $\mathcal{S}'$, and one for γ nodes in $\mathcal{S}$. Hence, each transmission serves 2γ nodes. Moreover, the IG Shuffle Method does not require the use of linear combinations or packetization of the IVs, and the sets of IVs can simply be XOR'd together. However, the IG Shuffle Method cannot be used in r-th Shuffle round since the node set $\mathcal{Y}$ would be empty. With the LC Shuffle Method, the node group shuffles linear combinations among one another and $2\gamma - 1$ nodes are served with each transmission. Then, after a node receives all the transmissions from other nodes of the node group, it can solve for all its requested IV packets. While the LC Shuffle Method can be generalized for any round γ, since we only use it for $\gamma = r$, its generalized form is not presented here.

Remark 5: While the proposed design only operates for $r = s$, it will be interesting to explore the possibility of expanding the proposed design to $r \neq s$. The main challenge stems from taking advantage of IVs that are requested by multiple nodes. In the shuffle method, it is ideal to group nodes which have similar requests as we have shown here for $r = s$ and as done in [3]. If the relationship between the file placement and function assignment is arbitrary, as in the case when $r \neq s$, this becomes a combinatorial optimization problem. There may not exist a systematic scheme to find a shuffle phase that minimizes the communication load.

Remark 6: When considering arithmetic complexity, we see the benefits of using bit-wise XOR for most of the Shuffle rounds. Here, we define arithmetic complexity as the number of addition and multiplications in the field $\mathbb{F}_{2^T}$ for encoding or decoding. Encoding or decoding bit-wise XOR has a complexity of 1 since two packets are simply added together. Then, in the final round, for a given shuffle group, a node generates $2^{(r-1)}$ linear combinations and decodes by solving a linear system of size $(2r - 1)2^{(r-1)}$. Using Gaussian elimination, solving such a system has an arithmetic complexity of $O(r^3 2^{3r})$. Considering all shuffle groups, the per node encode complexity for the first $r - 1$ rounds is $O\left(\left(\frac{K}{r}\right)^{2r-2}\right)$ and the decode complexity is $O\left(r\left(\frac{K}{r}\right)^{2r-2}\right)$. Then, for the last round, the encode complexity at each node is $O\left(\left(\frac{K}{r}\right)^{2r-1} 2^{r-2}\right)$ and the decode complexity is $O\left(r^3\left(\frac{K}{r}\right)^{2r-1} 2^{2r}\right)$.

C. 3-Dimension Homogeneous Example

Example 2: To demonstrate the general scheme, we construct a computing network using a 3-dimensional hypercube (or a cube) as shown in Fig. 2. First, we present the *NG File Mapping*.[4] Each lattice point in the cube represents a different file $\mathcal{B}_i = w_i$, $i \in [27]$ where $\eta_1 = 1$. The network has $K = 9$ nodes, partitioned into three sets: $\mathcal{K}_1 = \{1, 2, 3\}$, $\mathcal{K}_2 = \{4, 5, 6\}$, and $\mathcal{K}_3 = \{7, 8, 9\}$, aligned along each of the $r = 3$ dimensions of the cube. For example, the three nodes in $\mathcal{K}_1 = \{1, 2, 3\}$ are represented by three parallel planes, e.g., node 3 is represented by the green plane. For file mapping, each node is assigned all files indicated by the 9 lattice points on the corresponding plane. For instance, node 5, represented by the red plane, is assigned the file set $\mathcal{M}_5 = \{w_2, w_5, w_8, w_{11}, w_{14}, w_{17}, w_{20}, w_{23}, w_{26}\}$. For each $i \in [3]$, the size of $\mathcal{K}_i$ is $\frac{K}{r} = 3$, which is the number of lattice points in the i-th dimension. Since the three nodes in each set $\mathcal{K}_i$ are aligned along dimension i, they collectively stores the entire library of 27 files. Nodes compute every IV from each locally available file and therefore $r = 3$.

[4]While the File Mapping in this example is the same as that in [9], it is included here for completeness.

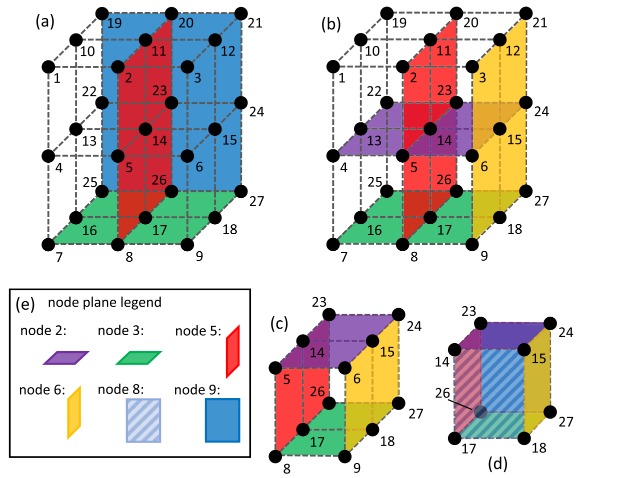


Fig. 2. The input files and output functions assigned to each node are represented by planes of lattice points from a cubic lattice. (a) planes assigned to nodes of a set $\mathcal{T}_{26} = \{3, 5, 9\}$. (b) planes assigned to a set of nodes $\mathcal{S} = \{2, 3, 5, 6\}$ in the 2nd round of the Shuffle phase and (c) lattice points which intersect 2 planes of from nodes in $\mathcal{S}$. (d) the lattice points which intersect 3 planes of nodes in set $\mathcal{S} = \{2, 3, 5, 6, 8, 9\}$. (e) legends for (a)-(d).

Next, we illustrate the *Cascaded Function Assignment*. The reduce functions are assigned to multiple nodes by the same process as the file mapping. The planes representing the reduce functions (and files) for 6 of 9 nodes are shown in Fig. 2. Each lattice point represents $\eta_2 = 1$ function (in addition to a file). For instance, node 2 stores files $\{w_i\}$ for all $i \in \{4, 5, 6, 13, 14, 15, 22, 23, 24\}$ (purple plane), and is assigned to compute reduce functions for all i in the same set. A depiction of these planes are shown in Fig. 2 (a) and (b). IVs can be categorized by the number (γ) of nodes which request them. Any IV of the form $v_{i,i}$ is only needed by nodes that can compute this IV themselves and are thus requested by $\gamma = 0$ nodes. Next, we consider round $\gamma = 1$ in which IVs which are requested by only 1 node and use the IG Shuffle Method. These IVs can be identified by considering node group $\mathcal{T}_\alpha$ ($\alpha \in \{1, 2, \cdots, 27\}$), consisting of 1 node from each set $\mathcal{K}_1$, $\mathcal{K}_2$ and $\mathcal{K}_3$. An example is $\mathcal{T}_\alpha = \{3, 5, 9\}$ whose planes are depicted in Fig. 2(a). Lattice points which fall on the intersection of exactly two of these planes represent input files that 2 out of the 3 nodes have available to it. As these 3 nodes are the only nodes that compute the 26-th reduce function, we see that $v_{26,23}$ is requested only by node 3 and available at nodes 5 and 9. Next, consider another node group

$\mathcal{T}_\ell = \{2,5,9\}$ that differs from $\mathcal{T}_\alpha$ by only in the first node (from $\mathcal{K}_1$). By observing the planes representing the nodes of $\mathcal{T}_\ell = \{2,5,9\}$, we find $v_{23,26}$ is requested only by node 2 and available at nodes 5 and 9. Therefore, either node 5 or 9 can transmit $v_{26,23} \oplus v_{23,26}$ to nodes 2 and 3 which can recover their requested IV. To match the description of the general scheme, we say $\mathcal{A} = \{1\}$, $\mathcal{S} = \{2,3\}$, $\mathcal{S}' = \{3\}$ and $\mathcal{Y} = \{5,9\}$.

Next, we consider Shuffle round $\gamma = 2$ in which IVs requested by 2 nodes are exchanged using the IG Shuffle Method. Given $\mathcal{T}_{26} = \{3,5,9\}$, whose planes are depicted in Fig. 2(a), we consider lattice points which intersect only 1 out of these 3 planes. For instance, w_{24} is available to node 9 and not nodes 3 or 5. Therefore, nodes 3 and 5 are the only nodes that request IV $v_{26,24}$. Since node 9 has this IV, it can multicast this IV to nodes 3 and 5. However, there is a way to serve two more nodes without increasing the communication load, recognizing that there are 2 other nodes, $\{2,6\}$, that have input file w_{24}. Given $\mathcal{S} = \{2,3,5,6\}$, there is a set of 4 files $\{w_{23}, w_{24}, w_{26}, w_{27}\}$ such that each file is only available to 2 nodes in $\mathcal{S}$ and all of these files are available to node 9. We define $\mathcal{S}' = \{3,5\}$ and $\mathcal{Y} = \{9\}$. Therefore, $\mathcal{V}^{\{3,5\}}_{\{2,6,9\}} = \{v_{26,24}\}$, $\mathcal{V}^{\{2,6\}}_{\{3,5,9\}} = \{v_{24,26}\}$ and node 9 transmits $v_{26,24} \oplus v_{24,26}$ to nodes $\{2,3,5,6\}$. Keeping $\mathcal{Y} = \{9\}$, we can also define $\mathcal{S}' = \{3,6\}$ to obtain $\mathcal{V}^{\{2,5\}}_{\{3,6,9\}} = \{v_{23,27}\}$ and $\mathcal{V}^{\{3,6\}}_{\{2,5,9\}} = \{v_{27,23}\}$ and node 9 also transmits $v_{23,27} \oplus v_{27,23}$ to nodes $\{2,3,5,6\}$. Continuing with $\mathcal{S} = \{2,3,5,6\}$, consider lattice points which are in the planes parallel to plane of node 9. These planes are defined by nodes $\{7,8\} \in \mathcal{K}_3$. The lattice points of interests in regards to $\mathcal{S}$ are highlighted in Fig. 2(c). We see that when $\mathcal{Y} = \{8\}$, node 8 transmits $v_{17,15} \oplus v_{15,17}$ and $v_{18,14} \oplus v_{14,18}$. When $\mathcal{Y} = \{7\}$, node 7 transmits $v_{5,9} \oplus v_{9,5}$ and $v_{6,8} \oplus v_{8,6}$. Each node of $\mathcal{S}$ has locally computed one IV and requests the other IV from each of the transmissions from nodes 7, 8 and 9.

Finally, we consider the last Shuffle round in which IVs requested by $\gamma = 3$ nodes are exchanged by the LC Shuffle Method. We see that none of the nodes in $\mathcal{T}_{26} = \{3,5,9\}$ have access to file w_{15} and therefore, they all request $v_{26,15}$. All 3 nodes in $\mathcal{T}_{15} = \{2,6,8\}$ have computed $v_{26,15}$, but request $v_{15,26}$ which nodes of $\mathcal{T}_{26}$ have computed. In fact, any node in a $([\gamma],1)$ node group $\mathcal{S}' \subset \mathcal{S} = \mathcal{T}_{26} \cup \mathcal{T}_{15}$ computes an IV that nodes in $\mathcal{S} \setminus \mathcal{S}'$ request. We consider the following sets of IVs: $\mathcal{V}^{\{3,5,9\}}_{\{2,6,8\}} = \{v_{26,15}\}$, $\mathcal{V}^{\{3,6,9\}}_{\{2,5,8\}} = \{v_{27,14}\}$, $\mathcal{V}^{\{2,5,9\}}_{\{3,6,8\}} = \{v_{23,18}\}$, $\mathcal{V}^{\{2,6,9\}}_{\{3,5,8\}} = \{v_{24,17}\}$, $\mathcal{V}^{\{2,6,8\}}_{\{3,5,9\}} = \{v_{15,26}\}$, $\mathcal{V}^{\{2,5,8\}}_{\{3,6,9\}} = \{v_{14,27}\}$, $\mathcal{V}^{\{3,6,8\}}_{\{2,5,9\}} = \{v_{18,23}\}$, and $\mathcal{V}^{\{3,5,8\}}_{\{2,6,9\}} = \{v_{17,24}\}$. The planes associated with each node of $\mathcal{S} = \{2,3,5,6,8,9\}$ are highlighted in Fig. 2(d). The IVs are split into 5 packets so that each node requests 20 unknown packets. Every node multicasts 4 linear combinations of its computed packets so that every node receives transmissions from 5 nodes and a total of 20 linear combinations to solve for the 20 requested packets. As proved in Appendix B, at the end of 3 shuffle rounds all node requests are satisfied.

In this example, each IV is computed at 3 nodes and $r = 3$. For round 1 ($\gamma = 1$), there are 9 node groups and nodes outside each group transmit an equivalent of 9 IVs. This results in $9 \cdot 9 = 81$ transmissions. For round 2 ($\gamma = 2$), nodes form 27 groups of 4 nodes and nodes outside each group transmit an equivalent of 6 IVs and leads to $27 \cdot 6 = 162$ transmissions. For round 3 ($\gamma = 3$), nodes form 27 groups of 6 nodes and each node in every group transmits an equivalent of $\frac{4}{5}$ IVs. This leads to $27 \cdot 6 \cdot 4/5 = 129.6$ transmissions of IVs. Collectively, the nodes transmit $(81 + 162 + 129.6)T = 372.6T$ bits and thus $L = \frac{372.6}{729} \approx 0.5111$.

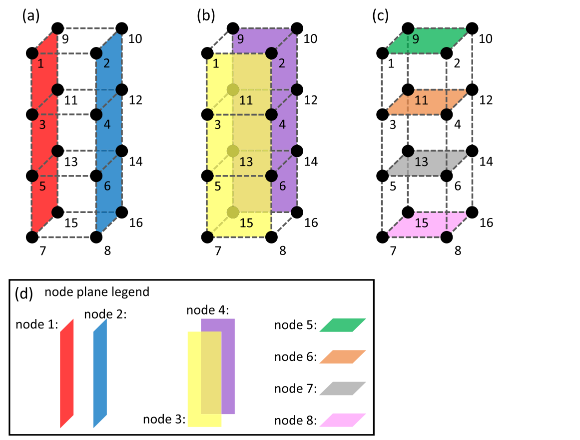


Fig. 3. A 3-dimensional lattice that defines the file availability and reduce function assignment of 8 nodes in a heterogeneous CDC network. Each lattice point represents both a file and a function. Nodes are assigned files and functions represented by planes in the lattice.

D. A 3-Dimensional (Cuboid) Heterogeneous Example

Example 3: Consider a heterogeneous network with $K = 8$ computing nodes where nodes $\mathcal{C}_1 = \{1,2,3,4\}$ have double the memory and computation power compared to nodes $\mathcal{C}_2 = \{5,6,7,8\}$. By General Node Grouping, nodes are split into 3 groups: $\mathcal{K}_1 = \{1,2\}$, $\mathcal{K}_2 = \{3,4\}$ and $\mathcal{K}_3 = \{5,6,7,8\}$. By NG File Mapping and Cascaded Function Assignment, there are $X = 16$ sets of functions and files and node assignments are represented by a lattice structure (a cuboid) in Fig. 3. Let $\eta_1 = \eta_2 = 1$ so that each file set only contains 1 file and each function set only contains 1 function. In the Map phase, every node computes every IV for each locally available file.

Next, we consider the Shuffle phase. In round 1 ($\gamma = 1$), we use the IG Shuffle Method and consider pairs of nodes that are from the same set $\mathcal{K}_i$ and aligned along the same dimension. Let $\mathcal{S} = \{1,2\}, \mathcal{S}' = \{1\}$, and $\mathcal{Y} = \{3,8\}$. We then have $\mathcal{S}' \cup \mathcal{Y} = \{1,3,8\} = \mathcal{T}_7$, and $(\mathcal{S} \setminus \mathcal{S}') \cup \mathcal{Y} = \{2,3,8\} = \mathcal{T}_8$. Note that node 1 is the only node that requests $v_{7,8}$ and node 2 is the only node that requests $v_{8,7}$. Hence, either node 3 or 8 from $\mathcal{Y}$ can transmit $v_{7,8} \oplus v_{8,7}$ to nodes 1 and 2 in $\mathcal{S}$. Continuing this process, we see that all IVs requested by a single node are transmitted in coded pairs.

Next, for round 2 ($\gamma = 2$) we use the IG Shuffle Method and consider groups of 4 nodes where 2 are from $\mathcal{K}_i$ and 2 are from $\mathcal{K}_j$ where $i \neq j$. For instance, let $\mathcal{S} = \{3,4,6,8\}$. If we let $\mathcal{S}' = \{3,6\}$, and $\mathcal{Y} = \{1\}$. We then have

$\mathcal{S}' \cup \mathcal{Y} = \{3,6,1\} = \mathcal{T}_3$, and $(\mathcal{S} \setminus \mathcal{S}') \cup \mathcal{Y} = \{4,8,1\} = \mathcal{T}_{15}$. Thus, node 1 from $\mathcal{Y}$ will transmit $v_{3,15} \oplus v_{15,3}$ to $\mathcal{S}$. For the same $\mathcal{S} = \{3,4,6,8\}, \mathcal{S}' = \{3,6\}$, if we let $\mathcal{Y} = \{2\}$, then we have $\mathcal{S}' \cup \mathcal{Y} = \{3,6,2\} = \mathcal{T}_4$, and $(\mathcal{S} \setminus \mathcal{S}') \cup \mathcal{Y} = \{4,8,2\} = \mathcal{T}_{16}$. Thus, node 2 from $\mathcal{Y}$ will transmit $v_{4,16} \oplus v_{16,4}$ to $\mathcal{S}$. Hence, IVs requested by 2 nodes can also be transmitted in coded pairs.

Finally, for 3 ($\gamma = 3$) we use the LC Shuffle Method and consider groups of 6 nodes that contains 2 nodes from each set $\mathcal{K}_1$, $\mathcal{K}_2$ and $\mathcal{K}_3$. For instance, consider $\mathcal{S} = \{1,2,3,4,5,6\}$. If we choose $\mathcal{S}' = \{1,3,5\} = \mathcal{T}_1$, then we have $\mathcal{S} \setminus \mathcal{S}' = \{2,4,6\} = \mathcal{T}_{12}$. We observe that $v_{1,12}$ is requested by three nodes in $\mathcal{S}'$ and is computed by all three nodes in $\mathcal{S} \setminus \mathcal{S}'$. Similarly, we consider the other three cases: $\mathcal{S}' = \{1,3,6\} = \mathcal{T}_3$, $\mathcal{S} \setminus \mathcal{S}' = \{2,4,5\} = \mathcal{T}_{10}$; $\mathcal{S}' = \{1,4,5\} = \mathcal{T}_9$, $\mathcal{S} \setminus \mathcal{S}' = \{2,3,6\} = \mathcal{T}_4$; and $\mathcal{S}' = \{1,4,6\} = \mathcal{T}_{11}$, $\mathcal{S} \setminus \mathcal{S}' = \{2,3,5\} = \mathcal{T}_2$. In this way, we identify 8 IVs which are requested by 3 nodes of $\mathcal{S}$ and locally computed at the 3 other nodes of $\mathcal{S}$. These IVs are: $v_{1,12}$, $v_{12,1}$, $v_{3,10}$, $v_{10,3}$, $v_{4,9}$, $v_{9,4}$, $v_{2,11}$ and $v_{11,2}$. Each IV is then split into $2\,r-1 = 2\cdot 3 - 1 = 5$ equal size packets and each node of $\mathcal{S}$ transmits $2^{r-1} = 2^2 = 4$ linear combinations of its locally available packets. Each node collectively receives $4 \cdot 5 = 20$ linear combinations from the other 5 nodes in $\mathcal{S}$ which are sufficient to solve for the requested 4 IVs or 20 unknown packets.

In this example, the computation load is $r = 3$ because every file is assigned to 3 nodes and every node locally computes all possible IVs. In order to compute the communication load, we can see that IVs requested by 0 nodes do not have to be transmitted. IVs requested by 1 or 2 nodes are transmitted in coded pairs, effectively reducing the communication load by half to shuffle these IVs. Hence, the number of transmissions in round 1 and 2 are given by $\frac{80}{2} = 40$, and $\frac{112}{2} = 56$, respectively. The number of transmissions in round 3 is $6 \cdot 6 \cdot \frac{4}{5} = 28.8$ because there are 6 choices of $\mathcal{S}$ of size 6 and each node transmit effectively $\frac{4}{5}$ of an IV. The communication load is thus given by $L = \frac{40+56+28.8}{256} = 0.4875$ where $QN = 16 \cdot 16 = 256$.

IV. Achievable Communication Load and Optimality

In this section, we present the achievable communication load of the proposed design for general cascaded CDC networks and discuss the optimality of the design given the proposed file and function assignment. An example is provided to illustrate the key steps in finding an information theoretic lower bound on the achievable communication load.

A. Achievable Communication Load

Theorem 1: Assume $r = s$. For the proposed hypercuboid scheme with NG File Mapping, Cascaded Function Assignment, and Multi-round Shuffle Phase, the following communication load is achievable

$$L_{\mathrm{c}} = \frac{X-1}{2X} + \frac{1}{X(4r-2)} \prod_{i=1}^{r} (|\mathcal{K}_i| - 1), \tag{12}$$

where $X = \prod_{i=1}^{r} |\mathcal{K}_i|$. An upper bound on L_c is obtained from (12) as

$$L_{\mathrm{c}} < \frac{1}{2} + \frac{1}{4r-2} = \frac{r}{2r-1} \leq \frac{2}{3}. \tag{13}$$

Proof: The proof of Theorem 1 is given in Appendix A. ■

Corollary 1: When setting $|\mathcal{K}_i| = \frac{K}{r}$ and $X = (\frac{K}{r})^r$, (12) gives the L_c of a homogeneous network with parameters K and r.

B. Optimality

In this section, we will show the optimality of the proposed hypercuboid approach for cascaded CDC. Note that, the fundamental computation-communication load tradeoff of [3] does not apply to the cascaded CDC design since it has a different reduce function assignment compared to that of [3]. We start by presenting the optimality of a homogeneous network using our proposed design and then for the more general heterogeneous design.

Theorem 2: Consider a homogeneous system with parameters K and r. Let L^* be the infimum of achievable communication load over all possible shuffle designs given the proposed NG File Mapping and Cascaded Function Assignment. Then, we have

$$L^* \geq \tfrac{1}{2} - \tfrac{1}{4m-2} - \left(\sum_{\hat{m}=1}^{m-1} \tfrac{\hat{m}^{2r}}{4\hat{m}^2-1}\right) m^{-2r}, \tag{14}$$

where $m = \frac{K}{r} \geq 2$. Furthermore, given L_c in (12), it follows from (14) that L_c is within a constant multiple of L^*

$$L_{\mathrm{c}} \leq \frac{64}{29} \cdot L^* \approx 2.207\ L^*, \tag{15}$$

for general K, r.

Proof: Theorem 2 is proved in Appendix C. ■

Remark 7: For the homogeneous network in Example 2, we have $L_c = 0.5111$. This is compared to the lower bound of (14) that gives $L^* \geq 0.3937$. In this case, we have $L_c \leq 1.2982\ L^*$, achieving a better constant than that of the general case given in (15).

Theorem 3: Consider a general heterogeneous system with parameters K and r. Let L^* be the infimum of achievable communication load over all possible shuffle designs given the proposed NG File Mapping and Cascaded Function Assignment. Let $x_i = |\mathcal{K}_i|$. Without loss of generality, assume that $x_1 \geq x_2 \cdots \geq x_s$. Then, we have

$$L^* \geq \max(L_{P1}, L_{P2}), \tag{16}$$

where

$$L_{P1} = \frac{x_1 - 1}{2x_1},$$
$$L_{P2} = \frac{1}{X^2} \sum_{i=1}^{r} \left(\prod_{j=1}^{i-1} x_j^2 \right) \sum_{\hat{m}=x_{i+1}+1}^{x_i} \sum_{\ell=1}^{i} (\hat{m}-1)^{2\ell-1} \hat{m}^{2(i-\ell)}. \tag{17}$$

Furthermore, for general K and r, we show that L_c is within a constant multiple of L^*,

$$L_{\mathrm{c}} < \frac{8}{3} \cdot L^*. \tag{18}$$

Proof: Theorem 3 is proved in Appendix D. ■

Remark 8: The two lower bounds L_{P1} and L_{P2} in (17) correspond to two different choices of permutations used to evaluate the right side of (29) of Lemma 2 in Appendix C. Extensive simulations suggest that the permutation used in L_{P2} is optimal in achieving the largest lower bound using Lemma 2. For instance, consider Example 4, we get $L_{P1} = 0.375$, which is less than $L_{P2} = 0.3945$. However, due to the complexity of (17), we use the simpler L_{P1} to determine the constant in (18). Note that the permutation used for L_{P2} matches with the permutation used in the homogeneous case to derive (14). Since L_{P1} is in general weaker than L_{P2}, we see that the constant in (18), derived using L_{P1}, is larger than that of (15) for the homogeneous case.

C. Optimality Example

Example 4: This example shows how to find a lower bound on the achievable communication load given the proposed NG Filing Mapping and Cascaded Function Assignment. Here, we use the homogeneous file mapping and function assignment of Example 2. Our approach builds upon an information theoretic lower bound (29) (see Lemma 2 and the notations therein in Appendix C), originally designed for $s = 1$ in [9], and extend it to the case of $s > 1$ for the case of cascaded CDC.

Lemma 2 requires that we pick a permutation of nodes and then use file and function counting arguments. The permutation we use here is $\{1, 7, 6, 2, 8, 5, 3, 9, 4\}$. To achieve a tighter bound, this permutation contains 3 sequential node groups where each node group contains one node aligned along each dimension of the cube. In order to calculate the terms of (29), for each node k_i, we count the number of files not available to the first i nodes of the permutation. This set of files is $\mathcal{M}_{\mathcal{K}} \setminus \mathcal{M}_{\{k_1,\ldots,k_i\}}$, called file of interests for node k_i. We also count the number of functions assigned to the i-th node of the permutation that are not assigned to the previous $i-1$ nodes. This set of functions is $\mathcal{W}_{k_i} \setminus \mathcal{W}_{\{k_1,\ldots,k_{i-1}\}}$, called functions of interests for node k_i. The product of these file and function counts represents the number of IVs of interests in Lemma 2. Moreover, since the IVs are independent and of size T bits, we have

$$H\left(V_{\mathcal{W}_{k_i},:}|V_{:,\mathcal{M}_{k_i}}, Y_{\{k_1,\ldots,k_{i-1}\}}\right) = T \cdot \left|\mathcal{M}_{\mathcal{K}} \setminus \mathcal{M}_{\{k_1,\ldots,k_i\}}\right| \cdot \left|\mathcal{W}_{k_i} \setminus \mathcal{W}_{\{k_1,\ldots,k_{i-1}\}}\right|, \quad (19)$$

where $H(\cdot)$ is the entropy function. In Fig. 4, we highlight the lattice points representing the sets of files and functions which are used to obtain the bound. Lattice points representing the files are highlighted in red and lattice points representing the functions are highlighted in green. First, we consider every function assigned to node 1 and every file not available to node 1, where node 1 is the first node in the permutation. This is shown in Fig. 4(a). We see that $H(V_{\mathcal{W}_1,:}|V_{:,\mathcal{M}_1}) = T \cdot 18 \cdot 9 = 162T$ since in this case each lattice point only represents 1 file and 1 function ($\eta_1 = \eta_2 = 1$).

Similarly, for node 7, we are count functions it computes and files it does not have locally available, except this time we do not count files available to node 1 or functions

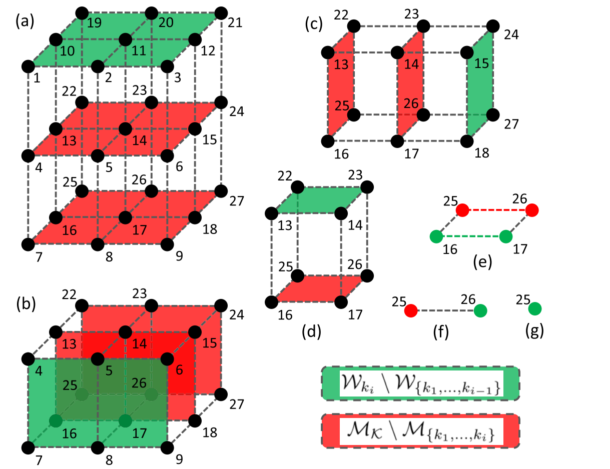


Fig. 4. A representation of Example 4 for a given permutation $\{1, 7, 6, 2, 8, 5, 3, 9, 4\}$. In (a)-(g), the i-th subfigure shows the functions of interests and files of interests for node k_i, highlighted in green and red, respectively. For instance, (c) shows the functions and files of interests for node 6, after accounting for functions and files of interests for node 1 (see (a)) and node 7 (see (b)). From (a)-(g), i increases by 1 at each step, and the lattice shrinks in one dimension by one unit. Refer to Fig. 2 for the definition of file mapping and function assignment.

assigned to node 1. Fig. 4(b) shows the files and functions we counting. Note that, we disregard the top layer of the cube which represents the files and functions assigned to node 1. We see that $H\left(V_{\mathcal{W}_7,:}|V_{:,\mathcal{M}_7}, Y_{\{1\}}\right) = T \cdot 6 \cdot 12 = 72T$. By continuing this process, from Fig. 4(c-f), we see that $H\left(V_{\mathcal{W}_6,:}|V_{:,\mathcal{M}_6}, Y_{\{1,7\}}\right) = 32T, H\left(V_{\mathcal{W}_2,:}|V_{:,\mathcal{M}_2}, Y_{\{1,7,6\}}\right) = 16T, H\left(V_{\mathcal{W}_8,:}|V_{:,\mathcal{M}_8}, Y_{\{1,7,6,2\}}\right) = 4T, H\left(V_{\mathcal{W}_5,:}|V_{:,\mathcal{M}_5}, Y_{\{1,7,6,2,8\}}\right) = T$. Finally, only 1 lattice point remains in Fig. 4 (g), representing a function assigned to node 3. However, there are no lattice points representing files node 3 does not have locally available. This occurs because the other two nodes aligned along the same dimension, nodes 1 and 2, have already been accounted for, and they collectively have all the files that node 3 does not have. Therefore, $H\left(V_{\mathcal{W}_3,:}|V_{:,\mathcal{M}_3}, Y_{\{1,7,6,2,8,5\}}\right) = 0$. Similarly, for the last two nodes of the permutation, nodes 9 and 4, there are no remaining files that are not locally available to them. In fact, there are also no functions assigned to nodes 9 and 4 which have not already been accounted for. Therefore, $H\left(V_{\mathcal{W}_9,:}|V_{:,\mathcal{M}_9}, Y_{\{1,7,6,2,8,5,3\}}\right) = H\left(V_{\mathcal{W}_4,:}|V_{:,\mathcal{M}_4}, Y_{\{1,7,6,2,8,5,3,9\}}\right) = 0$.

By taking the sum of (29), we directly compute the bound of (31) and find that

$$L^* \geq \frac{287T}{QNT} = \frac{287}{27 \cdot 27} \approx 0.3937. \quad (20)$$

V. Discussion

In this section, we compare the performance the proposed scheme with the state-of-the-art scheme of [3] and discuss design considerations of the proposed CDC scheme.

First, we compare the required number of files and functions for homogeneous designs. The scheme of [3] requires $N_1 = \binom{K}{r}\eta_1$ input files and $Q_1 = \binom{K}{s}\eta_2$ reduce functions. The proposed scheme requires $N_c = \left(\frac{K}{r}\right)^r \eta_1$ input files and $Q_c = \left(\frac{K}{s}\right)^s \eta_1$ output functions. Assuming $r = \Theta(K)$, by using Stirling's formula and a similar analysis found in [9], we directly compare the required number of input files to find

$$\frac{N_1}{N_c} = \Theta\left(\sqrt{\frac{K}{2\pi r(K-r)}} \cdot \left(\frac{K}{K-r}\right)^K\right) = \Theta\left(\sqrt{\frac{1}{K}} \cdot \left(\frac{K}{K-r}\right)^K\right). \tag{21}$$

Since (21) grows exponentially with K, the proposed scheme reduces the number of required files exponentially. By a similar analysis, we can show that the proposed scheme also allows an exponential reduction in the number of Reduce functions.

Next, we compare the communication load of the proposed CDC scheme with that of [3]. While the proposed design applies to heterogeneous networks, the design in [3] only applies to homogeneous networks. Hence, to facilitate fair comparisons, we compare with an equivalent homogeneous network of [3] with the same r, N, Q, for appropriate choices of η_1 and η_2. The scheme of [3] achieves the communication load as a function of K, r and s as

$$L_1(r,s) = \sum_{\gamma=\max\{r+1,s\}}^{\min\{r+s,K\}} \frac{\gamma\binom{K}{\gamma}\binom{\gamma-2}{r-1}\binom{r}{\gamma-s}}{r\binom{K}{r}\binom{K}{s}}. \tag{22}$$

Corollary 2: Let $L_c(r)$ be the resulting communication load from using the NG File Mapping, Cascaded Function Assignment and MR Shuffle Method, and $L_1(r,r)$ given by (22) for an equivalent computation load r and number of nodes K and $r = s$.

(a) When $r = s = 2$, for both homogeneous and heterogeneous hypercuboid designs, we have $L_c(2) < L_1(2,2)$.
(b) When $r = s \geq 6$ and $K > r - 1 + 4r^3$, there exists a heterogeneous hypercuboid design where $L_c(r) < L_1(r,r)$.
(c) In the limiting regime, when $r = s = o(K)$,[5] we have $\lim_{K\to\infty} \frac{L_c(r)}{L_1(r,r)} \leq 1$.

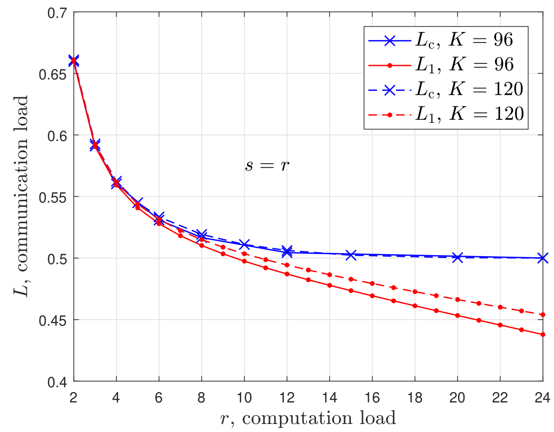
Proof: Corollary 2 is proven in Appendix E. ∎

A. Homogeneous Cascaded CDC

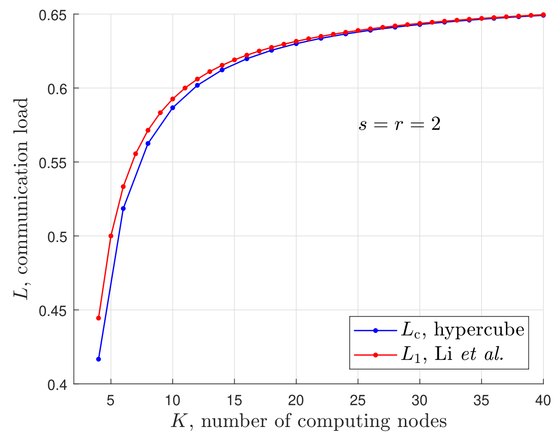
In this section, we provide numerical results to confirm the findings in Corollary 2 for homogeneous cascaded CDC.

In Fig. 5a, we compare $L_c(r)$ with $L_c(r,r)$ for large homogeneous networks ($K = 96, 120$) as r increases. For $s = r$, we observe that $L_c(r)$ with $L_c(r,r)$ are close when $r \ll K$, verifying Corollary 2 (c), but begin to deviate when $r = \Theta(K)$. We see that for most (but not all) values of K and r that $L_c = \frac{1}{2} + \epsilon$ where $\epsilon > 0$. The intuition behind this is for most of the Shuffle phase, IVs are included in coded pairs.

[5]We will use the following standard "order" notation: given two functions f and g, we say that: 1) $f(K) = O(g(K))$ if there exists a constant c and integer N such that $f(K) \leq cg(K)$ for $n > N$. 2) $f(K) = o(g(K))$ if $\lim_{K\to\infty} \frac{f(K)}{g(K)} = 0$. 3) $f(K) = \Omega(g(K))$ if $g(K) = O(f(K))$. 4) $f(K) = \omega(g(K))$ if $g(K) = o(f(K))$. 5) $f(K) = \Theta(g(K))$ if $f(K) = O(g(K))$ and $g(K) = O(f(K))$.



(a) Increase $r = s$ for fixed $K = 96, 100$.



(b) Increase K for fixed $r = s = 2$.

Fig. 5. Comparisons of communication load L_c of the proposed design and L_1 of [3] for homogeneous networks.

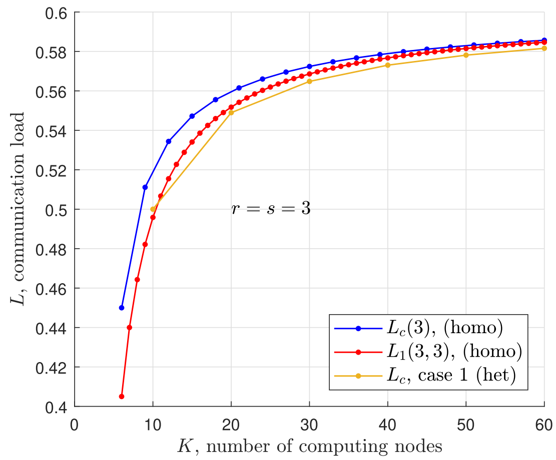
Meanwhile, from (22) and Fig. 5a, we see that $L_1(r,r)$ can have a communication load less than $\frac{1}{2}$.

Fig. 5 compares $L_c(r)$ and $L_1(r,r)$ as a function of K for fixed $r = s = 2$. This corresponds to the limiting regime of $r = o(K)$. Moreover, consistent with Corollary 2 (a), Fig. 5 shows the proposed design achieves a lower communication load than that of [3]. This is because while both the proposed scheme and that of [3] handle IVs that are requested by 1 or 2 nodes with the same efficiency, the former has a greater fraction of IVs which are requested by 0 nodes. The optimality of the scheme in [3] is proved under the key assumption on function assignment that every s nodes have at least 1 function in common. In contrast, we do not make such an assumption in the proposed design. This allows greater flexibility in the design of function assignment and enables a lower communication load than that of [3].

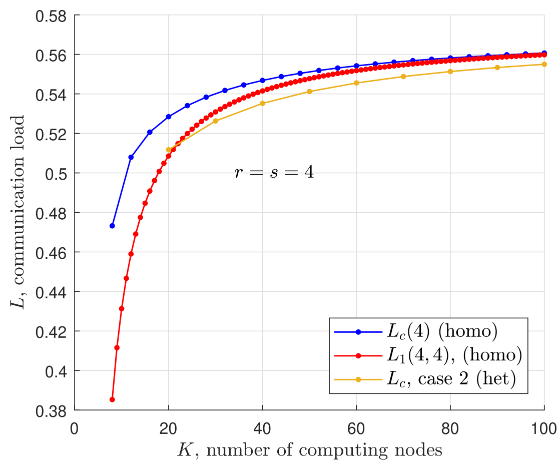
By the proposed NG File Mapping and Cascaded Function Assignment, the minimum requirement of N and Q is $\left(\frac{K}{r}\right)^r$ where $\eta_1 = \eta_2 = 1$. While the minimum requirements of N and Q in [3] are $\binom{K}{r}$ and $\binom{K}{s}$. Hence, it can be observed that the proposed approach reduces the required numbers of both N and Q exponentially as a function of r and s.

B. Heterogeneous Cascaded CDC

We consider the following two cases of heterogeneous network.



(a) Case 1



(b) Case 2

Fig. 6. Comparisons of the communication load achieved by the proposed heterogeneous design to equivalent homogeneous designs including the proposed design and the design from [3].

- *Case 1*: Assume $\frac{2}{5}$ of the nodes have 3 times as much storage capacity and computing power compared to the other $\frac{3}{5}$ of the nodes. Here, we set $P = 2, r_1 = 2, m_1 = 0.2\ K, r_2 = 1, m_2 = 0.6K$. Note that $m_2 = 3\ m_1$.
- *Case 2*: Assume $\frac{1}{5}$ of the nodes have 4 times as much storage capacity and computing power compared to the other $\frac{4}{5}$ of the nodes. Here, we set $P = 2, r_1 = 2, m_1 = 0.1\ K, r_2 = 2, m_2 = 0.4K$. Note that $m_2 = 4m_1$.

We compare these two cases to equivalent homogeneous schemes including the homogeneous scheme described in this paper and the scheme of [3]. Here, equivalent means the schemes are compared with the same r, s and K. Fig. 6 confirms Corollary 2 (b) that for fixed r and large K, there exists a proposed heterogeneous design with $L_c(r) < L_c(r, r)$. There appears to be an advantage of having a set of nodes with both more locally available files and assigned functions. In this way, less IV shuffling is required to satisfy the requests of these nodes. As discussed before, an extreme case of this can be observed where a subset of nodes each have all files locally available and compute all assigned functions. Furthermore, for the given simulations, the communication load of the heterogeneous designs approaches the communication load of the homogeneous designs as shown in Corollary 2 (c).

VI. Conclusion

In this work, we introduced a novel combinatorial hypercuboid approach for cascaded CDC frameworks with both homogeneous and heterogeneous network scenarios. The proposed low complexity combinatorial structure can determine both input file and output function assignments, requires significantly less number of input files and output functions, and operates on large heterogeneous networks where nodes have varying storage space and computing resources. Surprisingly, due to a different output function assignment, the proposed scheme can outperform the optimal state-of-the-art scheme with a different output function assignment. Moreover, we also show that the heterogeneous storage and computing resource can reduce the communication load compared to its homogeneous counterpart. Finally, the proposed scheme can be shown to be optimal within a constant factor of the information theoretic converse bound while fixing the input file and the output function assignments.

Appendix A
Proof of Theorem 1

Let $x_i = |\mathcal{K}_i|$ be the size of the i-th dimension of the hypercuboid. Note that if $\mathcal{K}_i \in \mathcal{C}_p$, then $x_i = m_p$, as defined in General Node Grouping of Section III-B. The communication load can be calculated by considering all r rounds of the Shuffle phase. For $\gamma \in \{1, \ldots, r-1\}$, in the γ-th round we use the IG Shuffle Method. We consider a node group $\mathcal{S}$ of 2γ nodes where there are 2 nodes from $\mathcal{K}_i$ for all $i \in \mathcal{A} \subseteq [r]$ such that $|\mathcal{A}| = \gamma$. Given $\mathcal{A}$ and $\mathcal{S}$ we identify all node sets $\mathcal{Y}$ which contain $r - \gamma$ nodes, 1 node from each set $\mathcal{K}_i$ for all $i \in [r] \setminus \mathcal{A}$. Given $\mathcal{A}$, there are $\prod_{i \in [r] \setminus \mathcal{A}} x_i$ possibilities for $\mathcal{Y}$. Furthermore, there are 2^γ possibilities for choosing a subset $\mathcal{S}' \subset \mathcal{S}$ such that $|\mathcal{S}'| = \gamma$. Therefore, there are

$$2^\gamma \prod_{i \in \mathcal{A}} \binom{x_i}{2} \prod_{i \notin \mathcal{A}} x_i = 2^\gamma \prod_{i \in \mathcal{A}} \frac{x_i(x_i - 1)}{2} \prod_{i \notin \mathcal{A}} x_i$$
$$= X \prod_{i \in \mathcal{A}} (x_i - 1) \quad (23)$$

unique pairs of $\mathcal{Y}$ and $\mathcal{S}'$ given $\mathcal{A}$. For each unique pair of $\mathcal{Y}$ and $\mathcal{S}'$, we define a set of IVs $\mathcal{V}^{\mathcal{S} \setminus \mathcal{S}'}_{\mathcal{S}' \cup \mathcal{Y}}$ which only contains IVs $v_{i,j}$ such that $i \in \mathcal{D}_\alpha$ and $w_j \in \mathcal{B}_\ell$ where $\{\{\mathcal{S} \setminus \mathcal{S}'\} \cup \mathcal{Y}\} = \mathcal{T}_\alpha$ and $\mathcal{S}' \cup \mathcal{Y} = \mathcal{T}_\ell$. Since $|\mathcal{B}_\ell| = \eta_1$ and $|\mathcal{D}_\alpha| = \eta_2$, we see that $|\mathcal{V}^{\mathcal{S} \setminus \mathcal{S}'}_{\mathcal{S}' \cup \mathcal{Y}}| = \eta_1 \eta_2$. All of the IV sets are transmitted in coded pairs, effectively reducing the contribution to the communication load by half. Therefore, given $\mathcal{A}$, there are $\frac{\eta_1 \eta_2 X}{2} \prod_{i \in \mathcal{A}} (x_i - 1)$ transmissions of size T bits, the number of bits in a single IV. $\mathcal{A}$ can range in size from 1 to $\gamma - 1$. Accounting for all possibilities of $\mathcal{A}$, we obtain the total number of bits transmitted as

$$\frac{\eta_1 \eta_2 X T}{2} \sum_{\gamma=1}^{r-1} \left(\sum_{\{\mathcal{A} : \mathcal{A} \subset [r], |\mathcal{A}| = \gamma\}} \left(\prod_{i \in \mathcal{A}} (x_i - 1) \right) \right). \quad (24)$$

Finally, in the r-th round, we use the LC Shuffle Method. We consider all node groups of $2r$ nodes, $\mathcal{S}$, such that $|\mathcal{S} \cap \mathcal{K}_i| = 2$ for all $i \in [r]$. There are $\prod_{i=1}^{r} \binom{x_i}{2}$ possibilities for a node group $\mathcal{S}$. Furthermore, given $\mathcal{S}$, there are 2^r possibilities for a node group $S' \subset \mathcal{S}$ such that $|\mathcal{S}' \cap \mathcal{K}_i| = 1$ for all $i \in [r]$. We see that $\mathcal{S}' = \mathcal{T}_\ell$ and $\{\mathcal{S} \setminus \mathcal{S}'\} = \mathcal{T}_{\alpha_\ell}$ for some ℓ which determines α_ℓ. Therefore, $|\mathcal{V}_{\mathcal{S}'}^{\mathcal{S} \setminus \mathcal{S}'}| = |\mathcal{B}_\ell| \cdot |\mathcal{D}_{\alpha_\ell}| = \eta_1\eta_2$. Each node of $\mathcal{S}$ transmits 2^{r-1} linear combinations of size $\frac{\eta_1\eta_2 T}{2r-1}$ bits and the total number of bits transmitted in the r-th round is

$$\frac{2r\eta_1\eta_2 2^{r-1}T}{2r-1} \prod_{i=1}^{r} \binom{x_i}{2} = \frac{r\eta_1\eta_2 TX}{2r-1} \prod_{i=1}^{r} (x_i - 1). \quad (25)$$

Next, we need to add (24), (25), and normalize by $QNT = \eta_1\eta_2 X^2 T$ to get L_c. The summation can be simplified using Lemma 1 below.

Lemma 1: Given a set of numbers $a_1, a_2, \ldots, a_c \in \mathbb{R}$, the sum of the product of all subsets, including the empty set, of this set of numbers is

$$\sum_{\mathcal{C} \subseteq [c]} \prod_{i \in \mathcal{C}} a_i = (a_1 + 1) \times (a_2 + 1) \times \cdots \times (a_c + 1). \quad (26)$$

Lemma 1 easily follows by considering the expansion of the right side of (26). Using Lemma 1, the communication load (12) is given by

$$\begin{aligned} L_{\mathrm{c}} &= \frac{1}{2X} \left(\prod_{i=1}^{r} |\mathcal{K}_i| - 1 - \prod_{i=1}^{r} (|\mathcal{K}_i| - 1) \right) \\ &\quad + \frac{r}{X(2r-1)} \prod_{i=1}^{r} (|\mathcal{K}_i| - 1) \\ &= \frac{X-1}{2X} + \frac{1}{X(4r-2)} \prod_{i=1}^{r} (|\mathcal{K}_i| - 1) \\ &< \frac{1}{2} + \frac{1}{4r-2}. \end{aligned} \quad (27)$$

Appendix B
Correctness of Heterogeneous CDC Scheme

Consider $r = s$ sets of IVs, where the γ-th set includes IVs requested by γ nodes. For each set, we prove that Shuffle Methods from Section III-B satisfy the following: 1) all IVs from that set are included in a coded transmission, 2) nodes can decode IVs they request from that set and 3) nodes only transmit IVs from that set which are computed from locally available files. Then by using the specified Shuffle Method for each $\gamma \in \{1, \ldots, s\}$, each node will receive all its requested IVs and be able to compute all assigned functions in the Reduce phase.

We first prove criterion 1) for the IG and LC Shuffle Methods. For $\gamma \in \{1, \ldots, r-1\}$, in the γ-th round we see $|\mathcal{T}_\alpha \cap \mathcal{T}_\ell| = |\mathcal{Y}| = r - \gamma$. Also, any $\mathcal{T}_\ell$ is possible and given $\mathcal{T}_\ell$ any $\mathcal{T}_\alpha$ is possible given that $|\mathcal{T}_\alpha \cap \mathcal{T}_\ell| = r - \gamma$. Therefore, the set of IVs transmitted is

$$\{v_{i,j} : i \in \mathcal{D}_\alpha, w_j \in \mathcal{B}_\ell, |\mathcal{T}_\ell \cap \mathcal{T}_\alpha| = r - \gamma\}. \quad (28)$$

This is the set of all IVs requested by γ nodes and this proves 1) for the IG Shuffle Method. Similarly, for the r-th round, in the LC Shuffle Method, we consider all possible pairs $\mathcal{T}_\ell$ and $\mathcal{T}_{\alpha\ell}$ such that $|\mathcal{T}_\alpha \cap \mathcal{T}_\ell| = 0$ and the sets have no nodes in common. The IVs included in the linear combinations in the r-th are then $\{v_{i,j} : i \in \mathcal{D}_{\alpha_\ell}, w_j \in \mathcal{B}_\ell, |\mathcal{T}_\ell \cap \mathcal{T}_{\alpha_\ell}| = 0\}$. which represents all IVs requested by r nodes and this proves 1) for the LC Shuffle Method.

Next, for the IG Shuffle Method, consider an arbitrary node $z \in \mathcal{S}$ that receives a multicast message from node $y \in \mathcal{Y}$ where $z \notin \mathcal{Y}$. The message is of the form $\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{S} \setminus \mathcal{S}'} \oplus \mathcal{V}_{\mathcal{T}_\alpha}^{\mathcal{S}'}$, given in (9), where $\mathcal{T}_\alpha = \{\mathcal{S} \setminus \mathcal{S}'\} \cup \mathcal{Y}$ and $\mathcal{T}_\ell = \mathcal{S}' \cup \mathcal{Y}$. Note that z is either in $\mathcal{S}'$ or $\mathcal{S} \setminus \mathcal{S}'$. If $z \in \mathcal{S}'$, then since $z \in \mathcal{T}_\ell$, it has access to $\mathcal{B}_\ell$ and thus can compute all IVs in $\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{S} \setminus \mathcal{S}'}$ and then subtract these off from the coded message to recover its desired IVs in $\mathcal{V}_{\mathcal{T}_\alpha}^{\mathcal{S}'}$. The same reasoning applies to the case when $z \in \mathcal{S} \setminus \mathcal{S}'$. This confirms 2). To confirm 3), we see that for any node $y \in \mathcal{Y}$, since y is in both $\mathcal{T}_\ell$ and $\mathcal{T}_\alpha$, by the NG File Mapping node y has access to both $\mathcal{B}_\ell$ and $\mathcal{B}_\alpha$ and thus can compute IVs in both $\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{S} \setminus \mathcal{S}'}$ and $\mathcal{V}_{\mathcal{T}_\alpha}^{\mathcal{S}'}$.

For the LC Shuffle Method in the r-th round, for a given $\mathcal{S}$, there are 2^r choices of $\mathcal{T}_\ell$ in (10) which determines the node group $\mathcal{T}_{\alpha_\ell}$. Fix a node $z \in \mathcal{S}$. Since half of these $\mathcal{T}_\ell$ include node z, we see that z can compute exactly half of these IVs, and requests the other half of them. These leads to 2^{r-1} unknown IV sets $\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{T}_{\alpha_\ell}}$ requested by node z. Since these IVs are further divided into $2r-1$ disjoint subsets, node z will request a total of $(2r-1)2^{r-1}$ unknown packets. Node z will receive transmissions from the other $2r-1$ nodes in $\mathcal{S}$ in which each node transmits 2^{r-1} random linear combinations of its known IV sets of interest. Therefore, node z can recover the $(2r-1)2^{r-1}$ unknown packets since it receives $(2r-1)2^{r-1}$ linearly independent combinations with high probability as the field size goes to infinity, which is shown below.

For the LC Shuffle method, we consider a $(2r-1)2^{r-1}$ square coefficient matrix $\mathbf{A} = (a_{i,j})$ for the requested packets of a node $z \in \{\mathcal{S} \setminus \mathcal{S}'\} \cap \mathcal{K}_i$. The first 2^{r-1} rows contain only non-zero elements since z receives linear combinations of all requested packets from the node $k \in \mathcal{S}' \cap \mathcal{K}_i$. We consider the other $2r-2$ nodes in pairs as $\{k_1, k_2\} = \mathcal{S} \cap \mathcal{K}_j$ for $j \neq i$. Nodes k_1 and k_2 send LCs with half of the requested packets by node z, and they complement each other such that k_1 and k_2 do not have any transmitted packets in common. Therefore, $\mathbf{A}$ contains $(2r-2)2^{r-1}$ rows with half non-zero elements and each half-zero row has a compliment row with the position of the zero and non-zero elements swapped. If we randomly generate the non-zero elements, we can show that at least one term of the determinant computation is non-zero. Therefore, by the Schwartz-Zippel Lemma [12], the probability of this matrix being invertible is high as the field size goes to infinity. We create a matrix $\mathbf{A}' = (a'_{n,m})$ by re-arranging the rows of $\mathbf{A}$ such that $a'_{n,n} > 0$. Consider the half-zero rows of $\mathbf{A}$ which we will place as the first and last $(2r-2)2^{r-2}$ rows of $\mathbf{A}'$. For each $n \in [(2r-2)2^{r-2}]$, consider some unplaced half-zero row of $\mathbf{A}$ which we label as row i and whose unplaced compliment row is row i'. If $a_{i,n} > 0$, then we let $a'_{n,m} = a_{i,m}$ and $a'_{(2r-1)2^{r-1}-n+1,m} = a_{i',m}$, where m spans the length of each row. Otherwise, if $a_{i,n} = 0$, we place the rows in the opposite places. Note that, for $n \in [(2r-2)2^{r-2}]$, there will always be an unplaced compliment row pair $\{i, i'\}$ such that the

$a_{i,n} > 0$ and $a'_{i',(2r-1)2^{r-1}-n+1} > 0$. After this, the 2^{r-1} entirely non-zero rows of $\mathbf{A}$ are placed in the remaining rows of $\mathbf{A}'$. This matrix has no non-zero elements along the diagonal and at least one term of the determinant computation is non-zero.

This proves criterion 2) for the LC Shuffle Method. To confirm 3) for LC Shuffle Method, we see that since node $k \in \mathcal{T}_\ell$, it has access to $\mathcal{B}_\ell$, and thus can compute all IVs in $\mathcal{V}_{\mathcal{T}_\ell}^{\mathcal{T}_{\alpha_\ell}}$.

Appendix C
Proof of Theorem 2

The proof of Theorem 2 utilizes Lemma 2 in [9] which is based on the approaches in [3], [13] and provides a lower bound on the entropy of all transmissions in the Shuffle phase given a specific function and file placement and a permutation of the computing nodes.

Lemma 2: Given a particular file placement and function assignment $\{\mathcal{M}_k, \mathcal{W}_k, \ \forall k \in [K]\}$, in order for every node $k \in [K]$ to have access to all IVs necessary to compute functions of $\mathcal{W}_k$, the optimal communication load over all achievable shuffle schemes, L^*, is bounded by

$$L^* \geq \frac{1}{TQN} \sum_{i=1}^{K} H\left(V_{\mathcal{W}_{k_i},:} | V_{:,\mathcal{M}_{k_i}}, Y_{\{k_1,\ldots,k_{i-1}\}}\right), \quad (29)$$

where $k_1, \ldots, k_K$ is some permutation of $[K]$, $V_{\mathcal{W}_{k_i},:}$ is the set of IVs necessary to compute the functions of $\mathcal{W}_{k_i}$. Here, the notation ":" is used to denote all possible indices. $V_{:,\mathcal{M}_{k_i}}$ is set of IVs which can be computed from the file set $\mathcal{M}_{k_i}$ and $Y_{\{k_1,\ldots,k_{i-1}\}}$ is the union of the set of IVs necessary to compute the functions of $\bigcup_{j=1}^{i-1} \mathcal{W}_{k_j}$ and the set of IVs which can be computed from files of $\bigcup_{j=1}^{i-1} \mathcal{M}_{k_j}$. □

Proof of Theorem 2: We pick a permutation of nodes by first dividing the K nodes into $m = \frac{K}{r}$ disjoint $([r], 1)$ node groups $\{\mathcal{G}_1, \ldots, \mathcal{G}_m\}$, each containing a node from $\{\mathcal{K}_i, i \in [r]\}$. Note that each $\mathcal{K}_i$ contains m nodes aligned along i-th dimension of the hypercube, and each $\mathcal{G}_i$ has size r. In particular, each $\mathcal{G}_i, i \in [m]$ is an element in the set of all possible $([r], 1)$ node groups $\{\mathcal{T}_1, \cdots, \mathcal{T}_X\}$ as defined in Single Node Grouping of Section III-B with $P = 1$. Then the permutation is defined such that $\mathcal{G}_1$ contains the first r nodes, $\mathcal{G}_2$ contains the next r nodes and this pattern continues such that $\mathcal{G}_m$ contains the last r nodes of the permutation. In other words, $\{k_{(j-1)r+1}, \ldots, k_{jr}\} = \mathcal{G}_j$ for all $j \in \{1, \ldots, m\}$. Given this permutation, to compute the i-th term of (29), we will show

$$H\left(V_{\mathcal{W}_{k_i},:} | V_{:,\mathcal{M}_{k_i}}, Y_{\{k_1,\ldots,k_{i-1}\}}\right) = \eta_1 \eta_2 T \hat{m}_i^{(2r-2\ell_i)} (\hat{m}_i - 1)^{(2\ell_i - 1)}, \quad (30)$$

where $\hat{m}_i = m - \lfloor \frac{i-1}{r} \rfloor$ and $\ell_i = i - r\lfloor \frac{i-1}{r} \rfloor$. Note that nodes $\{k_1, k_2, \cdots, k_{i-1}\}$ consists of all nodes in $\{\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_{\lfloor \frac{i-1}{r} \rfloor}\}$, $\ell_i - 1$ nodes in $\mathcal{G}_{\lfloor \frac{i-1}{r} \rfloor + 1}$, and no nodes in any of the $\hat{m}_i - 1$ node groups in $\{\mathcal{G}_{\lfloor \frac{i-1}{r} \rfloor + 2}, \cdots, \mathcal{G}_m\}$. In particular, k_i is the ℓ_i-th node in $\mathcal{G}_{n_i}$ where $n_i = \lfloor \frac{i-1}{r} \rfloor + 1$.

Since the IVs are assumed to be independent, we will take two steps to count the number of terms in (30). In Step 1, we count the number of functions that are in $\mathcal{W}_{k_i}$, but not in $\{\mathcal{W}_{k_1}, \cdots, \mathcal{W}_{k_{i-1}}\}$. These are referred to as functions of interests. By the definition of cascaded function assignment, this is equivalent to counting the number of $([r], 1)$ node groups $\mathcal{T}_l$ such that $\mathcal{T}_l$ includes k_i, but none of nodes in $\{k_1, k_2, \cdots, k_{i-1}\}$. Now, consider the first ℓ_i nodes in $\mathcal{G}_{n_i}$. Without loss of generality, assume that these nodes are taken from $\mathcal{K}_j, j = 1, \cdots, \ell_i$, respectively. Then, for any dimension $r_0 \in \{\ell_i + 1, \cdots, r\}$, $\mathcal{T}_{l,r_0}$ can be any of the $\hat{m}_i$ elements from $\{\mathcal{G}_{j,r_0}, n_i \leq j \leq m\}$. Here, $\mathcal{T}_{l,r_0}$ (or $\mathcal{G}_{j,r_0}$) denotes the element in $\mathcal{T}_l$ (or $\mathcal{G}_j$) that is chosen from $\mathcal{K}_{r_0}$. Similarly, for any $r_0 \in \{1, \cdots, \ell_i - 1\}$, $\mathcal{T}_{l,r_0}$ can be any of the $\hat{m}_i - 1$ elements from $\{\mathcal{G}_{j,r_0}, n_i + 1 \leq j \leq m\}$. When $r_0 = \ell_i$, we must have $\mathcal{T}_{l,r_0} = k_i$. This gives a total of $\hat{m}_i^{r-\ell_i}(\hat{m}_i - 1)^{\ell_i - 1}$ choices of such $\mathcal{T}_l$. In Step 2, we count the number of files that are not in $\{\mathcal{M}_{k_1}, \cdots, \mathcal{M}_{k_i}\}$. These are referred to as files of interests. This step is equivalent to counting the number of $([r], 1)$ node groups $\mathcal{T}_l$ that do not include any of the nodes $\{k_1, k_2, \cdots, k_i\}$. By replacing the case of $r_0 \in \{1, 2, \cdots \ell_i - 1\}$ in Step 1 by $r_0 \in \{1, 2, \cdots \ell_i\}$, we obtain a total of $\hat{m}_i^{r-\ell_i}(\hat{m}_i - 1)^{\ell_i}$ choices of $\mathcal{T}_l$. By taking the product of the results as in (19) from both steps and accounting for the number of files, η_1, and functions, η_2, assigned to a node group $\mathcal{T}_l$, we obtain (30). The counting principle described above can be visualized in Example 4. For instance, in Step 2, when considering node k_i after some "layers have been peeled off" (previous nodes were considered), the hypercuboid has ℓ_i dimensions of size $\hat{m}_i - 1$ and $r - \ell_i$ dimensions of size $\hat{m}_i$.

It follows from (29) that we can sum (30) over all nodes $\{k_i, i = 1, \cdots, mr\}$ to calculate the lower bound corresponding to this permutation. Note that summing over the right side of (30) from $i = 1$ to $i = mr$ is the same as summing over all possible mr pairs of $(\hat{m}_i, \ell_i)$, where $\hat{m}_i$ goes from 1 to m, and ℓ_i goes from 1 to r. For instance, nodes in $\mathcal{G}_j$ all have the same $\hat{m}_i = m - j + 1$ but different ℓ_i that goes from 1 to r. In the following, for brevity, we drop the subscript i in the double summation over $\{(\hat{m}_i, l_i)\}$, with the understanding that the first summation goes through all node groups $\mathcal{G}_1, \cdots, \mathcal{G}_m$, and the second summation goes through each of the r nodes in a given node group.

$$\begin{aligned}
&L^* QN \\
&\geq \eta_1 \eta_2 \sum_{\hat{m}=1}^{m} \sum_{\ell=1}^{r} \hat{m}^{(2r-2\ell)} (\hat{m} - 1)^{(2\ell - 1)} \\
&= \eta_1 \eta_2 \sum_{\hat{m}=1}^{m} \hat{m}^{2r-2} (\hat{m} - 1) \sum_{\ell=0}^{r-1} \left(\frac{\hat{m} - 1}{\hat{m}}\right)^{2\ell} \\
&= \eta_1 \eta_2 \sum_{\hat{m}=1}^{m} (\hat{m} - 1) \frac{\hat{m}^{2r} - (\hat{m} - 1)^{2r}}{\hat{m}^2 - (\hat{m} - 1)^2} \\
&= \eta_1 \eta_2 \left(\sum_{\hat{m}=1}^{m} \frac{\hat{m}^{2r} - (\hat{m} - 1)^{2r}}{2} - \sum_{\hat{m}=1}^{m} \frac{\hat{m}^{2r} - (\hat{m} - 1)^{2r}}{4\hat{m} - 2}\right) \\
&= \eta_1 \eta_2 \left(\frac{m^{2r}}{2} - \sum_{\hat{m}=1}^{m} \frac{\hat{m}^{2r}}{4\hat{m} - 2} + \sum_{\hat{m}=0}^{m-1} \frac{\hat{m}^{2r}}{4\hat{m} + 2}\right) \\
&= \eta_1 \eta_2 \left(\frac{m^{2r}}{2} - \frac{m^{2r}}{4m - 2} - \sum_{\hat{m}=1}^{m-1} \frac{\hat{m}^{2r}}{4\hat{m}^2 - 1}\right). \quad (31)
\end{aligned}$$

By normalizing (31) by $QN = \eta_1 \eta_2 m^{2r}$, we obtain (14).

Moreover, we can loosen the bound of (14) to find

$$\begin{aligned} L^* &\geq \frac{1}{2} - \frac{1}{4m-2} - \frac{1}{4}\left(\sum_{\hat{m}=1}^{m-1} \hat{m}^{2r-2}\right) m^{-2r} \\ &\geq \frac{1}{2} - \frac{1}{4m-2} - \frac{1}{8m} + \frac{1}{8m^2} \\ &\geq \frac{29}{96}. \end{aligned} \tag{32}$$

The last inequality in (32) follows from the left side of being an increasing function of m when $m \geq 2$, and the minimum is achieved at $m = 2$. Combining (32) with (13), we obtain (15).

Appendix D
Proof of Theorem 3

In the following, let $x_i = |\mathcal{K}_i|$ be the size of the i-th dimension of the hypercuboid. WLOG, assume $x_1 \geq x_2 \geq \ldots \geq x_{s-1} \geq x_s$. First, we take a similar approach to Example 4 and the proof of Theorem 2 to derive L_{P2}. With each node of the permutation we remove a layer of the hypercuboid. Through this process, the hypercuboid reduces in size as we disregard files available and functions assigned to nodes of the previous nodes of the permutation. In particular, we design the permutation such that the next node is aligned along the dimension with the largest remaining size (accounting for layers previously removed). For example, if after accounting for some nodes the remaining sizes of the dimensions are $\hat{x}_1, \ldots, \hat{x}_r$, we pick the next node from the set $\mathcal{K}_n$ such that $\hat{x}_n$ is the largest dimension. Then, we count the number of files of interests which is $\eta_1(\hat{x}_n - 1)\prod_{j\neq n}\hat{x}_j$ and number of functions of interests $\eta_2\prod_{j\neq n}\hat{x}_j$.

$$\begin{aligned} &L^*QN \\ &\geq \eta_1\eta_2\sum_{i=1}^{r} \\ &\quad\times\left(\left(\prod_{j=1}^{i-1} x_j^2\right)\sum_{j=1}^{x_i-x_{i+1}}\sum_{k=1}^{i}(x_i-j)^{2k-1}(x_i-j+1)^{2(i-k)}\right) \\ &= \eta_1\eta_2\sum_{i=1}^{r}\left(\left(\prod_{j=1}^{i-1} x_j^2\right)\sum_{\hat{m}=x_{i+1}+1}^{x_i}\sum_{\ell=1}^{i}(\hat{m}-1)^{2\ell-1}\hat{m}^{2(i-\ell)}\right). \end{aligned} \tag{33}$$

After scaling (33) by $QN = \eta_1\eta_2\, X^2$, we obtain the desired expression for L_{P2}.

Next, we derive L_{P1} using a different permutation that includes only the x_1 nodes aligned along the largest dimension. Note that since nodes aligned along the same dimension collectively compute all functions, the remaining nodes of the permutation are irrelevant. Each of the x_1 nodes computes $\eta_2\frac{X}{x_1}$ functions and there are $\eta_1 X\frac{x_1-1}{x_1}$ files which are not available to it. For the first node of the permutation there are $\eta_1\eta_2 X^2\left(\frac{x_1-1}{x_1^2}\right)$ IVs of interest using the bound of Lemma 2. Since nodes aligned along the same dimension do not have any assigned functions in common, the number functions of interest remains the same for the following nodes. However, the number of files of interest decreases by $\eta_1\frac{X}{x_1}$ for each following node of the permutation. Since nodes aligned along the same dimension do not have any available files in common, the number of files of interest decreases by the same amount with each node in the permutation. Thus,

$$\begin{aligned} L^*QN &\geq \sum_{i=1}^{x_1}\eta_1\eta_2\left(\frac{X}{x_1}\right)\left(X\frac{x_1-1}{x_1} - (i-1)\cdot\frac{X}{x_1}\right) \\ &= \eta_1\eta_2\frac{X^2(x_1-1)}{2x_1}. \end{aligned} \tag{34}$$

By combining (34) and (13), we obtain (18).

Appendix E
Proof of Corollary 2

For (a), given that $r = s = 2$, we obtain $L_1 = \frac{2(K-2)}{3(K-1)}$ from (22), and $L_c = \frac{2}{3} - \frac{1}{2X}\frac{K+2}{3}$ from (12) using $|\mathcal{K}_1| + |\mathcal{K}_2| = K$ and $|\mathcal{K}_1|\cdot|\mathcal{K}_2| = X$. Since L_c is the largest when X is maximized to be $X = (\frac{K}{2})^2$ (corresponding to the homogeneous network), we have $L_c \leq \frac{2(K+1)(K-2)}{3K^2} < L_1$.

Next, for (b), when $r = s \leq \frac{K}{2}$ such that there exists an achievable hypercuboid design, then $\min\{r+s, K\} = r + s = 2r$. By only considering the last term of $L_1(r,r)$ in (22) we derive the following lower bound.

$$\begin{aligned} L_1(r,r) &> \frac{2r\binom{K}{2r}\binom{2r-2}{r-1}\binom{r}{r}}{r\binom{K}{r}^2} \\ &= \frac{r}{2r-1}\cdot\frac{(K-r)(K-r-1)\cdots(K-2r+1)}{K(K-1)\cdots(K-r+1)} \\ &> \frac{r}{2r-1}\left(1-\frac{r}{K-r+1}\right)^r = \frac{r}{2r-1}\cdot(1+o(1)). \end{aligned} \tag{35}$$

Next, we derive an upper bound on L_c. For a given r and K, let $|\mathcal{K}_1| = \cdots = |\mathcal{K}_{r-1}| = 2$ and $|\mathcal{K}_r| = K - 2(r-1)$. Then by (12)

$$\begin{aligned} L_{\mathrm{c}} &= \frac{X-1}{2X} + \frac{K-2r+1}{2^{r-1}(K-2r+2)(4r-2)} \\ &< \frac{r}{2r-1}\left(1-\frac{1}{2r}+\frac{1}{r2^r}\right). \end{aligned} \tag{36}$$

Then, combining (35) and (36) we find $L_{\mathrm{c}} < L_1$ if

$$K > r - 1 + \frac{r}{1-\left(1-\frac{1}{2r}+\frac{1}{r2^r}\right)^{1/r}}. \tag{37}$$

We now aim to find an upper bound on the RHS of (37). It can be shown that if $r \geq 6$ then $\left(1-\frac{1}{2r}+\frac{1}{r2^r}\right)^{1/r} \leq \exp(-\frac{1}{2r^2})$ and $1 - \exp(-\frac{1}{2r^2}) > \frac{1}{4r^2}$. Substituting this into (37), we find $L_1 < L_{\mathrm{c}}$ if $r \geq 6$ and $K > r - 1 + 4r^3$ which proves (b).

From (35), if $r = \Theta(1)$ then $L_1(r,r) \geq \frac{r}{2r-1} + o(1)$, and alternatively, if $r = \Omega(1)$ and $r = o(K)$ then $L_1(r,r) \geq \frac{1}{2} + o(1)$. From (13), $L_{\mathrm{c}} < \frac{r}{2r-1}$. Therefore, with the given assumptions that $r \geq 1$ and $r = o(K)$, we find that $\frac{L_{\mathrm{c}}(r)}{L_1(r,r)} \leq 1 + o(1)$ which proves (c).

REFERENCES

[1] N. Woolsey, R.-R. Chen, and M. Ji, “A new combinatorial design of coded distributed computing,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jun. 2018, pp. 726–730.

[2] N. Woolsey, R.-R. Chen, and M. Ji, “Cascaded coded distributed computing on heterogeneous networks,” in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2019, pp. 2644–2648.

[3] S. Li, M. A. Maddah-Ali, Q. Yu, and A. S. Avestimehr, “A fundamental tradeoff between computation and communication in distributed computing,” *IEEE Trans. Inf. Theory*, vol. 64, no. 1, pp. 109–128, Jan. 2018.

[4] J. Dean and S. Ghemawat, “MapReduce: Simplified data processing on large clusters,” *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[5] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, “Spark: Cluster computing with working sets,” *HotCloud*, vol. 10, nos. 10–10, p. 95, 2010.

[6] K. Konstantinidis and A. Ramamoorthy, “Resolvable designs for speeding up distributed computing,” *IEEE/ACM Trans. Netw.*, vol. 28, no. 4, pp. 1657–1670, Aug. 2020.

[7] M. Kiamari, C. Wang, and A. S. Avestimehr, “On heterogeneous coded distributed computing,” in *Proc. GLOBECOM IEEE Global Commun. Conf.*, Dec. 2017, pp. 1–7.

[8] N. Shakya, F. Li, and J. Chen, “Distributed computing with heterogeneous communication constraints: The worst-case computation load and proof by contradiction,” 2018, *arXiv:1802.00413*. [Online]. Available: http://arxiv.org/abs/1802.00413

[9] N. Woolsey, R.-R. Chen, and M. Ji, “A new combinatorial coded design for heterogeneous distributed computing,” 2020, *arXiv:2007.11116*. [Online]. Available: http://arxiv.org/abs/2007.11116

[10] F. Xu and M. Tao, “Heterogeneous coded distributed computing: Joint design of file allocation and function assignment,” in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2019, pp. 1–6.

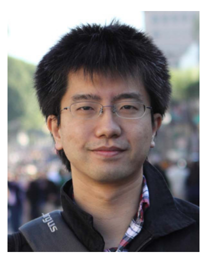
[11] N. Woolsey, R.-R. Chen, and M. Ji, “Coded distributed computing with heterogeneous function assignments,” in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2020, pp. 1–6.

[12] R. A. Demillo and R. J. Lipton, “A probabilistic remark on algebraic program testing,” *Inf. Process. Lett.*, vol. 7, no. 4, pp. 193–195, Jun. 1978.

[13] K. Wan, D. Tuninetti, and P. Piantanida, “An index coding approach to caching with uncoded cache placement,” *IEEE Trans. Inf. Theory*, vol. 66, no. 3, pp. 1318–1332, Mar. 2020.

Nicholas Woolsey (Student Member, IEEE) received the B.S. degree in biomedical engineering from the University of Connecticut in 2012 and the M.Eng. degree in bioengineering from the University of Maryland, College Park, MD, USA, in 2015, with a focus on signal processing, imaging and optics. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, The University of Utah. From 2014 to 2017, he was an Electrical Engineer with Northrop Grumman Corporation (NGC), Ogden, UT, USA, developing test and evaluation methods, modernization solutions and signal processing algorithms for the sustainment of aging aircraft and ground communication systems. He is also a Signal Processing Engineer with Trabus Technologies, San Diego, CA, USA, developing decentralized wireless network technologies. His research interests include combinatorial designs and algorithms for resource allocation, coding and efficient communications in distributed computing, and private and caching networks. While at NGC, he received the “Outside the Box” Grant to investigate the design of a modern receiver that interfaces aging technology and the 2016 Brent Scowcroft Team Award for performing exceptional systems engineering work. He defended his Ph.D. degree receiving the 2020 Best ECE Dissertation Award in December 2020.

Rong-Rong Chen (Member, IEEE) received the B.S. degree in applied mathematics from Tsinghua University, China, in 1993, and the M.S. degree in mathematics and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana–Champaign in 1995 and 2003, respectively. From 2003 to 2011, she was an Assistant Professor with The University of Utah, where she has been an Associate Professor since 2011. Her main research interests include communication systems and networks, with current emphasis on distributed computing, machine learning, caching networks, statistical signal processing, image reconstructions, and channel coding. She was a recipient of the M. E. Van Valkenburg Graduate Research Award for excellence in doctoral research in the ECE Department, University of Illinois at Urbana–Champaign in 2003. She was a recipient of the prestigious National Science Foundation Faculty Early Career Development (CAREER) Award in 2006. She was rated among the Top 15% Instructors of College of Engineering at The University of Utah in 2017 and 2018. She has served as an Associate Editor for IEEE TRANSACTIONS ON SIGNAL PROCESSING and as a Guest Editor of IEEE JOURNAL ON SELECTED TOPICS IN SIGNAL PROCESSING. She has served on the technical program committees of leading international conferences in wireless communication and networks.



Mingyue Ji (Member, IEEE) received the B.E. degree in communication engineering from the Beijing University of Posts and Telecommunications, China, in 2006, the M.Sc. degree in electrical engineering from the Royal Institute of Technology, Sweden, in 2008, the M.Sc. degree in electrical engineering from the University of California at Santa Cruz, Santa Cruz, CA, USA, in 2010, and the Ph.D. degree from the Ming Hsieh Department of Electrical Engineering, University of Southern California in 2015. From 2015 to 2016, he subsequently was a Staff II System Design Scientist with Broadcom Corporation (Broadcom Ltd.). He is currently an Assistant Professor with the Electrical and Computer Engineering Department and an Adjunct Assistant Professor with the School of Computing, The University of Utah. He is interested the broad area of information theory, coding theory, concentration of measure and statistics with the applications of caching networks, wireless communications, distributed storage and computing systems, distributed machine learning, and (statistical) signal processing. He received the IEEE Communications Society Leonard G. Abraham Prize for the Best IEEE JSAC Paper in 2019, the Best Paper Award in IEEE ICC 2015 Conference, the Best Student Paper Award in IEEE European Wireless 2010 Conference, and USC Annenberg Fellowship from 2010 to 2014. Since 2020, he has been serving as an Associate Editor for IEEE TRANSACTIONS ON COMMUNICATIONS.