2022 IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communication (SPAWC) | 978-1-6654-9455-7/22/$31.00 ©2022 IEEE | DOI: 10.1109/SPAWC51304.2022.9834012

Finite-length Analysis of D2D Coded Caching via Exploiting Asymmetry in Delivery

Xiang Zhang and Mingyue Ji
Department of Electrical and Computer Engineering, University of Utah
Email: {xiang.zhang, mingyue.ji}@utah.edu

***Abstract*—We present a novel Packet Type (PT)-based design framework for the finite-length analysis of Device-to-Device (D2D) coded caching. By the exploitation of the asymmetry in the coded delivery phase, two fundamental forms of subpacketization reduction gain for D2D coded caching, i.e., the subfile saving gain and the further splitting saving gain, are identified in the PT framework. The proposed framework features a streamlined design process which uses several key concepts including user grouping, subfile and packet types, multicast group types, transmitter selection, local/global further splitting factor, and PT design as an integer optimization. In particular, based on a predefined user grouping, the subfile and multicast group types can be determined and the cache placement of the users can be correspondingly determined. In this stage, subfiles of certain types can be potentially excluded without being used in the designed caching scheme, which we refer to as subfile saving gain. In the delivery phase, by a careful selection of the transmitters within each type of multicast groups, a smaller number of packets that each subfile needs to be further split into can be achieved, leading to the further splitting saving gain. The joint effect of these two gains results in an overall subpacketization reduction compared to the Ji-Caire-Molisch (JCM) scheme [1]. Using the PT framework, a new class of D2D caching schemes is constructed with order reduction on subpacketization but the same rate when compared to the JCM scheme.**

***Index Terms*—D2D, coded caching, packet type, finite-length analysis**

I. Introduction

D2D coded caching is an important variant of coded caching which extends the shared-link coded caching [2] to the case of D2D networks where no central server is presented but the D2D users take turns to transmit to their peers. Let K, N, M denote the total number of users, files and the per-user cache memory size. It is assumed that $t \triangleq \frac{KM}{N} \in \mathbb{N}$. The JCM scheme [1] uses the same combinatorial cache placement method as the shared-link scheme [2] but a different delivery scheme that achieves the worst-case communication rate $R_{\rm JCM} = \frac{N}{M} - 1$ which is shown to be optimal when $K \leq N$ under the assumption of uncoded cache placement and one-shot delivery [3]. To fully exploit the cache-induced DoF gain, the delivery phase is made symmetric by simply dividing each subfile $W_{n,\mathcal{T}}$ into t smaller packets, i.e., $W_{n,\mathcal{T}} = \left\{W^{(i)}_{n,\mathcal{T}}, i \in \{1, 2, \cdots, t\}\right\}$. This results in a total of $F_{\rm JCM} = t\binom{K}{t}$ packets per file which is shown to have exponential scaling in terms of the number of users [4], limiting its applicability to practical networks [5]. Yapar *et al.* [3] characterized the exact memory-rate trade-off of D2D coded caching by removing the redundancy in the delivery when $N < K$. However, the same file splitting as JCM was used, not addressing the subpacketization issue. The fundamental limits of D2D coded caching with distinct cache sizes, heterogeneous file popularity, secure delivery and private user demands were also investigated in [6]–[9].

There has been limited work on the finite-length analysis of D2D coded caching [10]–[14]. Woolsey *et al.* [10] proposed a hypercube scheme by modeling the cache placement as a multi-dimensional geometric hypercube. The hypercube scheme requires a subpacketization of $\left(\frac{N}{M}\right)^t$ which is smaller than $F_{\rm JCM}$. However, this scheme achieves a higher rate $R = \frac{t}{t-1}\left(\frac{N}{M} - 1\right) > R_{\rm JCM}$. Konstantinidis *et al.* [11] proposed an approach called resolvable design to reduce the number of subtasks in Coded Distributed Computing (CDC) [15]. Due to the close connection between CDC and D2D coded caching, this approach can also be used to obtain low-subpacketization D2D caching schemes but with a higher rate than $R_{\rm JCM}$. Placement Delivery Array (PDA) [16] is an approach to the finite-length analysis of shared-link coded caching which provides a flexible and reduced subpacketization at the cost of a higher rate than the scheme of [2]. Wang *et al.* [12] proposed an approach called D2D PDA (DPDA) under which lower bounds on both the rate and subpacketization were derived. It was shown that the JCM scheme meets the subpacketization lower bound when $t \in \{1, K-1\}$ but does not meet the bound when $t \in \{2, K-2\}$ although it always meets the rate lower bound. An extension of this work can be found in [13] where a direct translation of the shared-link PDA to the D2D case was used and new caching schemes were obtained based on existing shared-link designs. However, this approach cannot achieve the optimal rate.

As mentioned above, existing literature usually considers subpacketization reduction for D2D coded caching at the cost of a compromised rate, i.e., a rate higher than $R_{\rm JCM}$. One important question to ask is that, *Does the optimal rate have to be compromised if a lower subpacketization is to be pursued ?* It turns out that this is not true. The Packet Type (PT)-based framework [14] provides a new design paradigm to constructing low-subpacketization D2D caching schemes with optimal rate. This revealed that, unlike its shared-link counterpart, the JCM scheme is only optimal in terms of rate but not subpacketization in general. It was shown [14] that the JCM subpacketization $F_{\rm JCM} = t\binom{K}{t}$ can be improved without hurting the optimal rate $R_{\rm JCM} = \frac{N}{M} - 1$ for a large

range of caching parameters t, going beyond the two points $t \in \{2, K-2\}$ discovered in [12]. The distinguishing feature of the PT approach that separates it from shared-link approaches or DPDA is the exploitation of the *asymmetry* in the coded delivery phase which is specific to D2D coded caching. In particular, based on the idea of user grouping, subfiles/packets and multicast groups can be classified into multiple types. The cache placement can be optimized based on the subfile types and the coded delivery can also be optimized by using asymmetric transmitter selection across different multicast group types. The proposed vector Least Common Multiple (LCM) operation coordinates these different types of delivery which then produces the final caching scheme. Two fundamental subpacketization reduction gains, i.e., the *subfile saving gain* and the *further splitting saving gain*, were identified and formalized in the PT framework.

In this paper, we propose a new class of rate-optimal D2D caching schemes that achieves an order reduction on subpacketization compared to the JCM scheme. In particular, the proposed scheme uses an equal grouping where the users are divided into groups of two, based on which the subfile and multicast group types are determined. By a careful selection of the transmitters within each type of multicast groups, a new coded delivery scheme can be obtained which has an overall subpacketization that is significantly lower than the JCM scheme. Moreover, we show that the proposed user grouping achieves the minimum subpacketization among a class of equal-grouping PT designs.

Notation: $[m:n] \triangleq \{m, m+1, \cdots, n\}$, $(m:n) \triangleq (m, m+1, \cdots, n)$, $\underline{m}_n \triangleq (m, \cdots, m)$ where $|\underline{m}_n| = n$. We write $[1:n]$ as $[n]$ for short.

II. Problem Description

The (K, N, M) D2D coded caching problem consists of N files $W_1, \cdots, W_N$ each containing L bits, and K users each equipped with a cache memory that can store up to M files. Let $Z_1, \cdots, Z_K$ denote the user cache such that $|Z_k| = ML, \forall k$ bits. Our goal is to design both the cache placement and the delivery phase such that the rate and subpacketization can be minimized. Let $\boldsymbol{d} \triangleq (d_1, \cdots, d_K)$ be the demand vector where file W_{d_k} is requested by user k. In the following, we present a brief description of the PT design framework by decomposing it into multiple components including user grouping, subfile/packet type, multicast group type, further splitting factor, vector LCM operator, and PT as an integer optimization. Due to space limit, the details are omitted and can be found in [14].

User Grouping. The users are divided into m disjoint groups $\{\mathcal{Q}_i\}_{i=1}^m$ satisfying $\cup_{i=1}^m \mathcal{Q}_i = [K]$. A *user grouping* is a partition of K into m parts which can be represented by $\boldsymbol{q} \triangleq (q_1, q_2, \cdots, q_m)$ where $q_i \triangleq |\mathcal{Q}_{p_i}|$, $q_1 \geq \cdots \geq q_m > 0$ for some permutation $(p_1, \cdots, p_m)$ of $(1:m)$. Note that $\boldsymbol{q}$ only specifies the number of groups m and the size of each group $q_i, \forall i$ but not the specific assignments of the users to these groups. In the PT framework, it is not the specific assignments $\{\mathcal{Q}_i\}_{i=1}^m$ but the user grouping $\boldsymbol{q}$ that determines the caching scheme design. $\boldsymbol{q}$ is called an *equal grouping* if $q_i = K/m, \forall i$. For example, for $K = 4$, a possible $(2,1,1)$-grouping assignment is $\mathcal{Q}_1 = \{1,2\}, \mathcal{Q}_2 = \{3\}, \mathcal{Q}_3 = \{4\}$, and a possible $(2,2)$-grouping assignment is $\mathcal{Q}_1 = \{1,2\}, \mathcal{Q}_2 = \{3,4\}$. A *unique set* is defined as the union of all non-empty user groups containing the same number of users. Let N_d denote the total number of different unique sets in $\{\mathcal{Q}_i\}_{i=1}^m$ and let $\mathcal{U}_i$ denote the i^{th} unique set. For example, given $K = 7$ and the user grouping $\mathcal{Q}_1 = \{1,2,3\}$, $\mathcal{Q}_2 = \{4,5\}$, $\mathcal{Q}_3 = \{6\}$ and $\mathcal{Q}_4 = \{7\}$, there are $N_d = 3$ unique sets which are $\mathcal{U}_1 = \mathcal{Q}_1, \mathcal{U}_2 = \mathcal{Q}_2$ and $\mathcal{U}_3 = \mathcal{Q}_3 \cup \mathcal{Q}_4$.

Subfile/Packet Type. Given an equal grouping $\boldsymbol{q} = \left(\frac{K}{m}, \cdots, \frac{K}{m}\right)$ and a subset $\mathcal{B} \subseteq [K]$, we say $\mathcal{B}$ has *type* $\boldsymbol{v} \triangleq (v_1, \cdots, v_m)$ if $(v_1, \cdots, v_m) = (|\mathcal{B} \cap \mathcal{Q}_{p_1}|, \cdots, |\mathcal{B} \cap \mathcal{Q}_{p_m}|)$ where $|\mathcal{B} \cap \mathcal{Q}_{p_1}| \geq \cdots \geq |\mathcal{B} \cap \mathcal{Q}_{p_m}|$ and $(p_1, \cdots, p_m)$ is some permutation of $(1:m)$. For example, for a $(2,2)$-grouping $\mathcal{Q}_1 = \{1,2\}, \mathcal{Q}_2 = \{3,4\}$ of $K = 4$ users, type-$(1,0)$ subsets include all 1-subset of $[K]$; Type-$(1,1)$ subsets include $\{1,3\}, \{1,4\}, \{2,3\}$ and $\{2,4\}$; Type-$(2,0)$ subsets include $\{1,2\}, \{3,4\}$; Type-$(2,1)$ subsets include all 3-subset of $[K]$; The only type-$(2,2)$ subset is $[K]$. In D2D coded caching, a subfile $W_{n,\mathcal{T}}$ is stored by a set of users in $\mathcal{T}$ where $|\mathcal{T}| = t$. We say a subfile $W_{n,\mathcal{T}}$, or a packet $W_{n,\mathcal{T}}^{(i)}$ has type-$\boldsymbol{v}$ if $\mathcal{T}$ is a type-$\boldsymbol{v}$ subset over $\boldsymbol{q}$.[1] For example, given $[K] = \{1,2\} \cup \{3,4\}$ and $t = 2$, $W_{n,\{1,3\}}$ and $W_{n,\{2,4\}}$ are both type-$(1,1)$ subfiles.

Multicast Group Type. A multicast group $\mathcal{S}$ is a set of $t+1$ users among which the coded delivery is carried out. In particular, a subset or all of the $t+1$ users will be chosen as transmitters (TXs) and each of them will send a coded message in the form of XOR sums to the remaining t users in $\mathcal{S}$. Given an user grouping $\{\mathcal{Q}_i\}_{i=1}^m$, the type of a multicast group $\mathcal{S}$, denoted by $\boldsymbol{s} = (s_1, s_2, \cdots, s_m)$, is defined as the type of the subset $\mathcal{S}$ over $\boldsymbol{q}$. For example, for $[K] = \{1,2\} \cup \{3,4\}$ and $t = 2$, there is only one multicast group type $\boldsymbol{s} = (2,1)$ which includes the multicast groups $\{1,2,3\}$, $\{1,2,4\}$, $\{1,3,4\}$ and $\{2,3,4\}$. The *involved subfile type set* of multicast type $\boldsymbol{s}$, denoted by $\mathcal{I}$, is defined as the set of the subfile types that can appear in the coded delivery of the multicast groups of type $\boldsymbol{s}$. For example, for $K = 6, t = 2$ with $\mathcal{Q}_1 = \{1,2,3\}$, $\mathcal{Q}_2 = \{4,5,6\}$, there are two subfile types $\boldsymbol{v}_1 = (2,0), \boldsymbol{v}_2 = (1,1)$ and two multicast group types $\boldsymbol{s}_1 = (3,0), \boldsymbol{s}_2 = (2,1)$. The involved subfile type set related to $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ are $\mathcal{I}_1 = \{v_1\}$ and $\mathcal{I}_2 = \{\boldsymbol{v}_1, \boldsymbol{v}_2\}$ respectively.

Further Splitting (FS) Factor. Considering the coded delivery within a multicast group $\mathcal{S}$ with N_d unique sets $\mathcal{U}_1 \cdots, \mathcal{U}_{N_d}$ (i.e., $\mathcal{S} = \cup_{i=1}^{N_d} \mathcal{U}_i$), the users that are chosen as transmitters must be the users of one entire unique set or the union of multiple unique sets. Let $\mathcal{D}_{\text{TX}} \subseteq [N_d]$ be the indices of the unique sets that are chosen as transmitters. The involved subfile type set of $\mathcal{S}$ can be written as $\mathcal{I} = \{\boldsymbol{v}_i, i \in [N_d]\}$

[1] In D2D coded caching, there are two layers of subpacketization. The first layer is that the files are split into *subfiles* to fulfill the combinatorial cache placement; In the delivery phase, each subfile may need to be further split into multiple *packets* which is the second layer of subpacketization.

where $\boldsymbol{v}_i$ is the type of the subfiles $\left\{W_{d_{k_i},\mathcal{S}\backslash\{k_i\}}, k_i \in \mathcal{U}_i\right\}$. To ensure that each user $k_i \in \mathcal{U}_i$ can decode a desired packet from the coded message sent by each transmitter, the subfile $W_{d_{k_i},\mathcal{S}\backslash\{k_i\}}$ needs to be split into $\alpha(\boldsymbol{v}_i)$ (called *FS factors*) equal-sized packets such that each of them can be assigned to a different transmitter. $\alpha(\boldsymbol{v}_i)$ is given by

$$\alpha(\boldsymbol{v}_i) = \begin{cases} \sum\limits_{j\in\mathcal{D}_{\rm TX}} |\mathcal{U}_j| - 1, & \text{if } i \in \mathcal{D}_{\rm TX} \\ \sum\limits_{j\in\mathcal{D}_{\rm TX}} |\mathcal{U}_j|, & \text{if } i \notin \mathcal{D}_{\rm TX} \end{cases} \tag{1}$$

FS factors determined by (1) for multicast group type $\boldsymbol{s}$ are referred to as *local* FS factors and are represented by a local FS vector $\boldsymbol{\alpha} \triangleq (\alpha(\boldsymbol{v}_i))_{i=1}^{N_d}$.

Vector Least Common Multiple (LCM). Since there are usually multiple multicast group types and the same subfile type can be involved in more than one of them, the local FS vectors have to be coordinated. This can be achieved by using the *vector LCM* operator to produce a *global* FS vector $\boldsymbol{\alpha}^{\rm LCM} \triangleq (\alpha(\boldsymbol{v}_i))_{i=1}^{V}$ (V is the total number of subfile types) that determines the overall subpacketization. Denote $\mathbf{F} \triangleq [F(\boldsymbol{v}_i), \cdots, F(\boldsymbol{v}_V)]$ where $F(\boldsymbol{v}_i)$ is the total number of type-$\boldsymbol{v}_i$ subfiles under the user grouping $\boldsymbol{q}$. Then the total number of packets per file, i.e., subpacketization, is equal to $F_{\rm PT} = \boldsymbol{\alpha}^{\rm LCM}\mathbf{F}^{\rm T} = \sum_{i=1}^{V} \alpha(\boldsymbol{v}_i)F(\boldsymbol{v}_i)$. The definition of the vector LCM operator can be found in Definition 1 of [14].

PT Design as an Integer Optimization. With the above definitions, the subpacketization reduction problem can be formulated as an integer optimization with the objective of minimizing $\boldsymbol{\alpha}^{\rm LCM}\mathbf{F}^{\rm T}$ over all possible user groupings and transmitter selection for each multicast group type, subject to the cache memory constraint of the users.

III. Main Result

Theorem 1: Suppose $K(K \geq 4)$ and $\bar{t} \triangleq K - t$ are both even integers. When the memory size $M/N \geq 1/2$, the rate $R = N/M - 1$ of D2D coded caching is achievable with the supacketization $F_{\rm PT}$ satisfying

$$\frac{F_{\rm PT}}{F_{\rm JCM}} < \min\left\{\frac{\prod_{i=1}^{\bar{t}/2}(2i-1)}{K-\bar{t}}, 1\right\}. \tag{2}$$

Moreover, the upper bound (2) vanishes as K goes to infinity if $\frac{M}{N} = 1 - O\big(\frac{\log_2\log_2 K}{K}\big)$.

We present a new construction of D2D coded caching schemes using the PT design framework in Section V which achieves identical rate to the JCM scheme but a lower subpacketization by using a simple user grouping $\boldsymbol{q} = \left(\underline{2}_{K/2}\right)$ with $K/2$ groups each containing two users.

We highlight the implications of Theorem 1 as follows. Firstly, Theorem 1 reveals that existing D2D coded caching schemes are usually suboptimal in terms of subpacketization which can be potentially improved without compromising the rate. From (2), it can be seen that $F_{\rm PT}/F_{\rm JCM} = \Theta(1/K)$ if $\bar{t} = O(\log_2\log_2 K)$, implying an order reduction on subpacketization. The condition $M/N \geq 1/2$ is necessary to preserve the order reduction in the large memory regime. Since the JCM scheme can be viewed as a special class of PT design where all users within each multicast group are chosen as the transmitters which results in a global FS vector of $\boldsymbol{\alpha}_{\rm JCM} = (\underline{t}_V)$, FS factor reduction and subfile exclusion of the PT framework ensures $F_{\rm PT} \leq F_{\rm JCM}$ under any circumstance. Secondly, the corresponding PT design demonstrates that *asymmetric* multicast delivery plays a fundamental role for the general purpose of subpacketization reduction which has not been noticed and formalized by any existing work. Under the PT framework, two different reduction gains, including the subfile saving gain and further splitting saving gain, are conceptualized. The interplay between these two gains provides a new perspective to look at the finite-length analysis of D2D coded caching.

IV. Examples

Example 1: Consider $(K, N, M) = (10, 5, 3)$, $t = \frac{KM}{N} = 6$ and $\bar{t} = 4$. Using the user grouping $\boldsymbol{q} = (\underline{2}_5)$ with $\mathcal{Q}_i = \{2i-1, 2i\}, i \in [5]$, there are two multicast group types $\boldsymbol{s}_1 = (2, 2, 2, 1^\dagger, 0)$, $\boldsymbol{s}_2 = (2, 2, 1^\dagger, 1^\dagger, 1^\dagger)$ where the transmitters are marked by †, and three subfile/packet types which are $\boldsymbol{v}_1 = (2, 2, 2, 0, 0)$, $\boldsymbol{v}_2 = (2, 2, 1, 1, 0)$ and $\boldsymbol{v}_3 = (2, 1, 1, 1, 1)$ as shown in Fig. 1. The global FS vector is $\boldsymbol{\alpha}^{\rm LCM} = (0, 2, 3)$, implying that type-$\boldsymbol{v}_1$ subfiles are excluded while each type-$\boldsymbol{v}_2$ and $\boldsymbol{v}_3$ subfile needs to be split into two and three smaller packets respectively in the delivery phase. The placement and

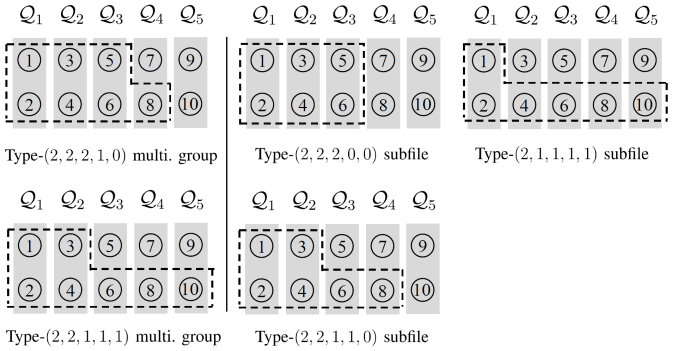

Fig. 1. Illustration of subfile and multicast group types.

delivery phases are described as follows.

1) Placement phase: Each file $W_n, n \in [5]$ is split into $F(\boldsymbol{v}_2) + F(\boldsymbol{v}_3) = \binom{5}{1}\binom{2}{2}\binom{4}{2}\binom{2}{1}^2 + \binom{5}{1}\binom{2}{2}\binom{4}{4}\binom{2}{1}^4 = 200$ subfiles, i.e., W_n can be written as

$$\begin{aligned} W_n = &\left\{W_{n,\mathcal{T}} : \mathcal{T} \subseteq [10], \texttt{type}(\mathcal{T}) = \boldsymbol{v}_2\right\} \\ &\cup \left\{W_{n,\mathcal{T}} : \mathcal{T} \subseteq [10], \texttt{type}(\mathcal{T}) = \boldsymbol{v}_3\right\}. \end{aligned} \tag{3}$$

All type-$\boldsymbol{v}_2$ subfiles have an identical size of $L/240$ bits and all type-$\boldsymbol{v}_3$ subfiles have an identical size of $L/160$ bits. The reason of using the above unequal subfile sizes is explained as follows. In the PT design, it is required that all packets have the same size (ℓ bits each) regardless of their types. Due to the further splitting, each type-$\boldsymbol{v}_i$ subfile will have $\alpha(\boldsymbol{v}_i)\ell$ bits where $\alpha(\boldsymbol{v}_i), i = 1, 2$ is the global FS factor. Since each file are finally split into $\alpha(\boldsymbol{v}_2)F(\boldsymbol{v}_2)+\alpha(\boldsymbol{v}_3)F(\boldsymbol{v}_3)$ equal-sized packets, the packet size can be calculated as $\ell =$

$L/\left(\alpha(\boldsymbol{v}_2)F(\boldsymbol{v}_2)+\alpha(\boldsymbol{v}_3)F(\boldsymbol{v}_3)\right) = L/480$ bits. Therefore, type-$\boldsymbol{v}_2$ and type-$\boldsymbol{v}_3$ subfiles will have sizes of $L/240$ and $L/160$ bits respectively. The cache of the users are

$$Z_k = \left\{W_{n,\mathcal{T}} : \forall \mathcal{T}, k \in \mathcal{T}, \forall n \in [5]\right\}, \quad \forall k \in [10] \tag{4}$$

It can be verified that each user stores 120 subfiles (72 type-$\boldsymbol{v}_2$ and 48 type-$\boldsymbol{v}_3$ subfiles) of each file, so the memory constraint $M = \left(5 \times \left(72 \times \frac{L}{240} + 48 \times \frac{L}{160}\right)\right)/L = 3$ is satisfied.

2) Delivery phase: The delivery phase is based on the PT design. In particular, to accommodate the coded delivery, the global FS vector $\boldsymbol{\alpha}^{\mathrm{LCM}} = (0, 2, 3)$ needs to be applied. Therefore, each type-$\boldsymbol{v}_2$ subfile is further split into 2 packets, i.e., $W_{n,\mathcal{T}} = \left\{W^{(i)}_{n,\mathcal{T}}, i = 1, 2\right\}$ if $\mathtt{type}(\mathcal{T}) = \boldsymbol{v}_2$, and each type-$\boldsymbol{v}_3$ subfile is split into 3 packets, i.e., $W_{n,\mathcal{T}} = \left\{W^{(i)}_{n,\mathcal{T}}, i = 1, 2, 3\right\}$ if $\mathtt{type}(\mathcal{T}) = \boldsymbol{v}_3$. This results in $F_{\mathrm{PT}} = 2F(\boldsymbol{v}_2) + 3F(\boldsymbol{v}_3) = 480$ packets per file which is smaller than $F_{\mathrm{JCM}} = 1260$.

First, consider the delivery within a type-$\boldsymbol{s}_1$ multicast group $\mathcal{S}_1 = \{1, 2, 3, 4, 5, 6, 8\}$ as shown in Fig. 1. User 8 is the only TX within this group and sends

$$\bigoplus_{k\in[6]} \left(W^{(1)}_{d_k,\mathcal{S}_1\setminus\{k\}}, W^{(2)}_{d_k,\mathcal{S}_1\setminus\{k\}}\right) \tag{5}$$

to all other users. From this delivery, each user $k \in [6]$ can decode two desired type-$\boldsymbol{v}_2$ packets. Next we consider the delivery within a type-$\boldsymbol{s}_2$ multicast group $\mathcal{S}_2 = \{1, 2, 3, 4, 6, 8, 10\}$ where users $6, 8, 10$ are the TXs. let (π_1, π_2, π_3) be a random permutation of $(1, 2, 3)$ and (π'_1, π'_2) be a permutation of $(1, 2)$. The coded message sent by each TX is shown in the following table. It can be seen that each user $k \in [4]$ can

TX	Coded message
6	$\left(\bigoplus_{k\in[4]} W^{(\pi_1)}_{d_k,\mathcal{S}_2\setminus\{k\}}\right) \oplus W^{(\pi'_1)}_{d_8,\mathcal{S}_2\setminus\{8\}} \oplus W^{(\pi'_1)}_{d_{10},\mathcal{S}_2\setminus\{10\}}$
8	$\left(\bigoplus_{k\in[4]} W^{(\pi_2)}_{d_k,\mathcal{S}_2\setminus\{k\}}\right) \oplus W^{(\pi'_1)}_{d_6,\mathcal{S}_2\setminus\{6\}} \oplus W^{(\pi'_2)}_{d_{10},\mathcal{S}_2\setminus\{10\}}$
10	$\left(\bigoplus_{k\in[4]} W^{(\pi_3)}_{d_k,\mathcal{S}_2\setminus\{k\}}\right) \oplus W^{(\pi'_2)}_{d_6,\mathcal{S}_2\setminus\{6\}} \oplus W^{(\pi'_2)}_{d_8,\mathcal{S}_2\setminus\{8\}}$

decode three type-$\boldsymbol{v}_3$ packets $\left\{W^{(i)}_{d_k,\mathcal{S}_2\setminus\{k\}}, i = 1, 2, 3\right\}$, and each user $k \in \{6, 8, 10\}$ can decode two type-$\boldsymbol{v}_2$ packets $\left\{W^{(i)}_{d_k,\mathcal{S}_2\setminus\{k\}}, i = 1, 2\right\}$. By looping through all type-$\boldsymbol{s}_1$ and type-$\boldsymbol{s}_2$ multicast groups, the delivery phase can be completed.

The correctness of the above delivery scheme can be verified as follows. Note that from each $\mathcal{S}$, each user therein can decode all the packets corresponding to a specific subfile. Therefore, we only need to make sure that each user can receive all the desired subfiles during the delivery phase. Recall that each file consists of $F(\boldsymbol{v}_2) = 120$ type-$\boldsymbol{v}_2$ subfiles and $F(\boldsymbol{v}_3) = 80$ type-$\boldsymbol{v}_3$ subfiles. In the placement phase, each user has already stored 72 type-$\boldsymbol{v}_2$ and 48 type-$\boldsymbol{v}_3$ subfiles. Therefore, it still needs all the $2 \times (120 - 72) = 96$ packets corresponding to the 48 type-$\boldsymbol{v}_2$ subfiles, and the $3 \times (80 - 48) = 96$ packets corresponding to the 32 type-$\boldsymbol{v}_3$ subfiles. We can write each type-$\boldsymbol{s}_1$ multicast group as $\mathcal{S} = \mathcal{Q}_{p_1} \cup \mathcal{Q}_{p_2} \cup \mathcal{Q}_{p_3} \cup \{u\}$, $u \in \mathcal{Q}_{p_4} \cup \mathcal{Q}_{p_5}$ where $(p_1, \cdots, p_5)$ is a random permutation of $(1 : 5)$. If $k = u$, user k does not receive any subfile from $\mathcal{S}$. Otherwise, there are $\binom{4}{2}\binom{2}{2}^2\binom{2}{1}\binom{2}{1} = 24$ multicast groups such that $k \in \mathcal{Q}_{p_1}$ and from each of them, user k can decode two type-$\boldsymbol{v}_2$ packets $\left\{W^{(i)}_{d_k,\mathcal{S}\setminus\{k\}}, i \in [2]\right\}$. Similarly, any type-$\boldsymbol{s}_2$ multicast group can be written as $\mathcal{S} = \mathcal{Q}_{p'_1} \cup \mathcal{Q}_{p'_2} \cup \{u_1, u_2, u_3\}$ where $u_1 \in \mathcal{Q}_{p'_3}$, $u_2 \in \mathcal{Q}_{p'_4}$, $u_3 \in \mathcal{Q}_{p'_5}$ and $(p'_1, \cdots, p'_5)$ is another random permutation. There are $\binom{4}{2}\binom{2}{2}^2\binom{2}{2}\binom{2}{1}^2 = 24$ multicast groups such that $k \in \{u_1, u_2, u_3\}$, from each of which user k can decode a different type-$\boldsymbol{v}_2$ subfile; There are $\binom{4}{1}\binom{2}{2}\binom{3}{3}\binom{2}{1}^3 = 32$ multicast groups such that $k \in \mathcal{Q}_{p'_1} \cup \mathcal{Q}_{p'_2}$, from each of which user k can decode a different type-$\boldsymbol{v}_3$ subfile (the three packets of that subfile). As a result, user k can in total decode $24+24 = 48$ type-$\boldsymbol{v}_2$ subfiles and 32 type-$\boldsymbol{v}_3$ subfiles. Therefore, each user can recover all its desired packets, proving the correctness of the above scheme.

Moreover, there are in total $F(\boldsymbol{s}_1) = \binom{5}{2}\binom{2}{2}^3\binom{2}{1}^2 = 40$ type-$\boldsymbol{s}_1$ and $F(\boldsymbol{s}_2) = \binom{5}{2}\binom{2}{2}^2\binom{3}{3}\binom{2}{1}^3 = 80$ type-$\boldsymbol{s}_2$ multicast groups. Since each type-$\boldsymbol{s}_1$ and type-$\boldsymbol{s}_2$ delivery contains 2ℓ and 3ℓ bits respectively, the achieved rate is $\left((2 \times 40 + 3 \times 80)\ell\right)/L = 2/3$ which is equal to R_{JCM}. The resulting subpacketization is $F_{\mathrm{PT}} = 480 < F_{\mathrm{JCM}} = 1260$. $\Diamond$

Remark 1: In Example 1, the subpacketization reduction of PT over the JCM scheme is a joint effect of *both* reduction gains. In particular, in terms of the subfile saving gain, the exclusion of type-$\boldsymbol{v}_1$ subfiles contributes to a reduction of $tF(\boldsymbol{v}_1) = 6 \times \binom{5}{2} = 60$ packets per file. In addition, the smaller FS factors for type-$\boldsymbol{v}_2$ and type-$\boldsymbol{v}_3$ subfiles contributes to a reduction of $(t-2)F(\boldsymbol{v}_2) + (t-3)F(\boldsymbol{v}_3) = 720$ packets per file, which reflects the further splitting saving gain.

V. General Achievable Scheme

We present the general PT design corresponding to Theorem 1 in this section. We show that order reduction on subpacketization can be achieved using the PT framework while preserving the same optimal rate as the JCM scheme. In particular, as long as $\bar{t} \triangleq K - t$ is a constant or upper bounded by $\bar{t} = O(\log_2 \log_2 K)$, the order gain can be preserved. The general achievable scheme is described as follows.

For $(K, \bar{t}) = (2m, 2r)$ with $m \geq \bar{t} + 1, r \geq 1$, consider the equal grouping $\boldsymbol{q} = (\underline{2}_m)$ with $m = K/2$ groups each containing two users. In this case, there are r different multicast group types and $r + 1$ subfile types. In particular, the i^{th} multicast group type and the j^{th} subfile type are

$$\boldsymbol{s}_i = \left(\underline{2}_{m-(r+i)+1}, \underline{1}^{\dagger}_{2i-1}, \underline{0}_{r-i}\right), \quad \forall i \in [r] \tag{6}$$

$$\boldsymbol{v}_j = \left(\underline{2}_{m-(r+j)+1}, \underline{1}_{2(j-1)}, \underline{0}_{r-j+1}\right), \quad \forall j \in [r+1] \tag{7}$$

where in $\boldsymbol{s}_i$, the transmitters are marked by $\dagger$. The i^{th} involved subfile type set is $\mathcal{I}_i = \{\boldsymbol{v}_i, \boldsymbol{v}_{i+1}\}$ and the corresponding local FS vector can be derived as $\boldsymbol{\alpha}_i = (\alpha(\boldsymbol{v}_i), \alpha(\boldsymbol{v}_{i+1})) = (2(i-1), 2i-1)$ according to (1). As a result, the global FS vector $\boldsymbol{\alpha}^{\mathrm{LCM}} = (\alpha_1, \alpha_2, \cdots, \alpha_{r+1})$ can be calculated using the vector LCM operator as follows: $\alpha_1 = 0, \alpha_2 = 2^{r-1}\prod_{k=1}^{r-1} k, \alpha_i = \prod_{k=1}^{i-1}(2k-1)\prod_{k=i-1}^{r-1} 2k, \forall i \in [3 : r-1], \alpha_r = 2(r-1)\prod_{k=1}^{r-1}(2k-1)$ and $\alpha_{r+1} = \prod_{k=1}^{r}(2k-1)$. $\alpha_1 = 0$ implies that type-$\boldsymbol{v}_1$ subfiles are excluded. Observing that the sequence $\{\alpha_i\}_{i=1}^{r+1}$ is strictly increasing and

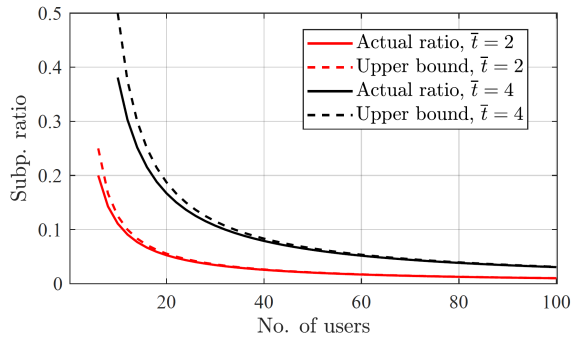

Fig. 2. Actual subpacketization ratio vs. upper bound.

$\binom{K}{t} = \sum_{i=1}^{r+1} F(\boldsymbol{v}_i)$, the subpacketization ratio $F_{\rm PT}/F_{\rm JCM}$ can be upper bounded by $\frac{F_{\rm PT}}{F_{\rm JCM}} = \frac{\sum_{i=1}^{r+1}\alpha_i F(\boldsymbol{v}_i)}{t\sum_{i=1}^{r+1}F(\boldsymbol{v}_i)} < \frac{\alpha_{r+1}}{t} = \frac{\prod_{i=1}^{\bar{t}/2}(2i-1)}{K-\bar{t}} = \Theta\left(\frac{1}{K}\right)$, implying an order reduction for any fixed $\bar{t}$. It can be shown that for $\bar{t} = O(\log_2\log_2 K)$, the order reduction can also be achieved. In addition, $F_{\rm PT}/F_{\rm JCM} < 1$ can always be guaranteed due to the exclusion of type-$\boldsymbol{v}_1$ subfiles. Therefore, we conclude $\frac{F_{\rm PT}}{F_{\rm JCM}} < \left\{\frac{\prod_{i=1}^{\bar{t}/2}(2i-1)}{K-\bar{t}}, 1\right\}$, which completes the proof of Theorem 1. A comparison of the actual ratio $F_{\rm PT}/F_{\rm JCM}$ and the upper bound in (2) for $\bar{t}=2,4$ is shown in Fig. 2. Due to space limit, the description of the cache placement and delivery phases is omitted. The above achievable scheme used an equal grouping with $K/2$ groups each contains two users. An interesting fact is that this user grouping actually achieves the minimum subpacketization among a class of equal grouping PT designs. A case for $\bar{t}=2$ is given in Lemma 1.

Lemma 1: For $\left(K,\bar{t}\right) = (mq,2)$*, the user grouping* $\boldsymbol{q} = (\underline{2}_m)$ *achieves the minimum subpacketization among the set of equal groupings* $\left\{\boldsymbol{q} = \left(\underline{q}_m\right) : mq = K, q \geq 2, m \geq 2\right\}$.

Proof: For the user grouping $\boldsymbol{q} = \left(\underline{q}_m\right)$ where $m, q \geq 2$, there is only one multicast group type $\boldsymbol{s} = \left(\underline{q}_{m-1}, (q-1)^{\dagger}\right)$. There are two subfile types which are $\boldsymbol{v}_1 = \left(\underline{q}_{m-1}, q-2\right)$, $\boldsymbol{v}_2 = \left(\underline{q}_{m-2}, \underline{(q-1)}_2\right)$. The involved subfile type set is $\mathcal{I} = \{\boldsymbol{v}_1, \boldsymbol{v}_2\}$. Since there is only one multicast group type, the global FS vector is the same as the local FS vector which is $\boldsymbol{\alpha}^{\rm LCM} = (q-2, q-1)$. Hence, the total number of packets is

$$\begin{aligned} F(q) &= \boldsymbol{\alpha}^{\rm LCM}\left[F(\boldsymbol{v}_1), F(\boldsymbol{v}_2)\right]^{\rm T} \\ &= (q-2, q-1)\left[K(q-1)/2, K(K-q)/2\right]^{\rm T} \\ &= (q-1)K(K-2)/2. \end{aligned} \tag{8}$$

Since $2 \leq q \leq K/2$, $F(q)$ is minimized by $q^* = 2$ and $F(q^*) = \frac{K(K-2)}{2}$. As a result, $\boldsymbol{q} = \left(\underline{2}_m\right)$ achieves the minimum subpacketization. ■

VI. Conclusion

Using the packet type-based design framework, we proposed a new class of rate-optimal D2D caching schemes with significantly reduced subpacketization than any known scheme. The proposed scheme employs an equal user grouping where the users are divided into groups of two. By carefully determining the transmitters within each type of multicast groups, the asymmetry in the coded delivery stage was exploited and two fundamental reduction gains, i.e., the subfile saving gain and further splitting saving gain can be achieved simultaneously whose combined effect leads to an order reduction on subpacketization over the JCM scheme. Several future directions can be investigated. First, it is not clear whether the order reduction can be achieved in the small memory regime ($M/N < 1/2$) which is of practical interest, or when either K or $\bar{t}$ is not even. Second, user groupings other than $\boldsymbol{q} = (\underline{2}_{K/2})$ can be explored to see if order or constant reduction can be achieved.

Acknowledgement

The work of X. Zhang and M. Ji was supported through the National Science Foundation grants CCF- 1817154 and SpecEES-1824558.

References

[1] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless d2d networks," *IEEE Transactions on Information Theory*, vol. 62, no. 2, pp. 849–869, Feb 2016.

[2] M. A. Maddah-Ali and U. Niesen, "Fundamental limits of caching," *Information Theory, IEEE Trans. on*, vol. 60, no. 5, pp. 2856–2867, 2014.

[3] Ç. Yapar, K. Wan, R. F. Schaefer, and G. Caire, "On the optimality of d2d coded caching with uncoded cache placement and one-shot delivery," *IEEE Transactions on Communications*, vol. 67, no. 12, pp. 8179–8192, 2019.

[4] X. Zhang, N. Woolsey, and M. Ji, "Cache-aided interference management using hypercube combinatorial design with reduced subpacketizations and order optimal sum-degrees of freedom," *IEEE Transactions on Wireless Communications*, vol. 20, no. 8, pp. 4797–4810, 2021.

[5] E. Lampiris and P. Elia, "Adding transmitters dramatically boosts coded-caching gains for finite file sizes," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 6, pp. 1176–1188, 2018.

[6] A. M. Ibrahim, A. A. Zewail, and A. Yener, "Device-to-device coded-caching with distinct cache sizes," *IEEE Transactions on Communications*, vol. 68, no. 5, pp. 2748–2762, 2020.

[7] M.-C. Lee, M. Ji, A. F. Molisch, and N. Sastry, "Throughput-outage analysis and evaluation of cache-aided d2d networks with measured popularity distributions," *IEEE Transactions on Wireless Communications*, vol. 18, no. 11, pp. 5316–5332, 2019.

[8] A. A. Zewail and A. Yener, "Device-to-device secure coded caching," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 1513–1524, 2020.

[9] K. Wan and G. Caire, "On coded caching with private demands," *IEEE Transactions on Information Theory*, vol. 67, no. 1, pp. 358–372, 2021.

[10] N. Woolsey, R.-R. Chen, and M. Ji, "Towards finite file packetizations in wireless device-to-device caching networks," *IEEE Transactions on Communications*, vol. 68, no. 9, pp. 5283–5298, 2020.

[11] K. Konstantinidis and A. Ramamoorthy, "Resolvable designs for speeding up distributed computing," *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1657–1670, 2020.

[12] J. Wang, M. Cheng, Q. Yan, and X. Tang, "On the placement delivery array design for coded caching scheme in d2d networks," *arXiv:1712.06212*, 2017.

[13] ——, "Placement delivery array design for coded caching scheme in d2d networks," *IEEE Transactions on Communications*, vol. 67, no. 5, pp. 3388–3395, 2019.

[14] X. Zhang, X. T. Yang, and M. Ji, "A new design framework on D2D coded caching with optimal rate and less subpacketizations," in *2020 IEEE International Symposium on Information Theory (ISIT)*, 2020, pp. 1699–1704.

[15] S. Li, M. A. Maddah-Ali, Q. Yu, and A. S. Avestimehr, "A fundamental tradeoff between computation and communication in distributed computing," *IEEE Transactions on Information Theory*, vol. 64, no. 1, pp. 109–128, 2017.

[16] Q. Yan, M. Cheng, X. Tang, and Q. Chen, "On the placement delivery array design for centralized coded caching scheme," *IEEE Transactions on Information Theory*, vol. 63, no. 9, pp. 5821–5833, 2017.