Random Linear Coded Distributed Caching for Video Streaming over D2D

Hasti A. Pedersen*, Lucile Sassatelli, Ramon Aparicio-Pardo †
*University of California at San Diego, San Diego, CA, USA
†Universite Cote d'Azur, CNRS, I3S, Sophia Antipolis, France
*hasti@eng.ucsd.edu, †{sassatelli, raparicio}@i3s.unice.fr

***Abstract*—In distributed file systems, Random-Linear-Coding (RLC) reduces scheduler complexity and download time by eliminating the coupon collector problem [1]. Similar benefits for video streaming in Distributed Video Caching and Sharing (DVCS) systems require segmentation of videos into codeblocks, over which RLC can be applied, to ensure videos' quality. This research proposes RLC-based DVCS (RLC-DVCS), and shows that RLC reduces complexity of the distributed video caching decision from *which* video chunks to *how many* RLC chunks to cache at each mobile device, and better facilitates distributability of content among devices (more mobile devices dedicating fewer resources each). All while increasing the number of video requests served on-time from DVCS, despite the challenges of video streaming. RLC-DVCS utilizes optimization and approximation methods for video segmentation, cache selection and allocation, to ensure videos' quality and cache more videos of highest request probability by reducing distributed size of each video. Using lagrangian relaxation allows for distributed RLC-DVCS where each mobile device decides how many RLC chunks to cache with limited coordination from other devices or a centralized scheduler. Monte Carlo evaluation of the proposed methods shows that RLC-DVCS achieves higher recovery probability than symmetric allocation of RLC chunks across mobile devices or un-coded DVCS.**

***Index Terms*—Random Linear Coding, Network Coding, Mobile Device Caching, Wireless Networks, Video Quality of Experience.**

I. Introduction

Over-the-top (OTT) video delivery for cellular networks is soaring in popularity but remains a challenge due to its high bandwidth demand, required Quality of Experience (QoE), and communication over wireless channel. Data offloading over alternative networks and content caching at cell edge and devices alleviate the challenges of OTT delivery [2]–[4]. In this paper, we propose RLC-based Distributed Video Caching and Sharing (RLC-DVCS) to merge the benefits of RLC (reduced scheduler complexity and download time [1]) and DVCS to improve QoE and capacity of the OTT video delivery systems.

D2D video sharing is similar to Peer-to-Peer (P2P) [5]–[7] except for the shorter communication range and smaller distributed cache size (combined size of all Mobile Device Caches (MDCs) a requester can reach). DVCS effectiveness relies on caching *Most Likely Requested* (MLR) videos within a D2D range, to serve video requests from MDCs despite their intermittent availability. Like P2P, D2D is susceptible to the free-riding problem [8], occuring when a requester (MDC) downloads content without adequate participation in assisting other MDCs' downloads [1]. Distributability – i.e., many MDCs contributing the least amount of resources – is a desired property of DVCS as it alleviates the free-riding problem and compensates for unreliability of MDCs due to mobility and D2D collaboration range. Such distributed design without RLC introduces the *Coupon Collector Problem*, where it becomes progressively harder to find data chunks in MDCs that have not already been downloaded. Therefore, the requester must query MDCs further away to find the missing chunks, leading to degraded performance due to increased path loss (in wireless networks) or throughput bottlenecks (in wired networks), and increased latency of finding MDCs with the missing chunks.

An RLC chunk is a random linear combination of the chunks (codeblocks) with coefficients chosen from a Galois Field (GF) with a size that strikes a balance between performance and complexity, e.g., GF(2^8). With RLC, any chunk found will, with high probability, provide innovative data [1] (analogous to finding a chunk not earlier downloaded); reducing the problem of finding a specific set of chunks to finding a certain number of chunks. RLC is more beneficial when applied to large segments, e.g., the entire video, as it avoids the coupon collector problem across segments of a video. However, video streaming requires the playback deadline of each video frame to be met, which forces RLC over segments of shorter duration than the entire video, for low decoding latency. In this paper, we propose a video segmentation that satisfies videos' QoE (initial delay and no stalling) while getting most of the benefits of RLC.

A segment is recovered if the required number of chunks for its decoding are downloaded before the playback deadline of its 1^{st} frame. A video's QoE is satisfied if all its segments are recovered. Due to MDCs' probabilistic availability, the recovery of a segment is probabilistic, which necessitates redundancy to ensure its target recovery probability is met. The distributed segment size is the total storage dedicated to that segment across all MDCs. Like uncoded-DVCS, RLC-DVCS effectiveness relies on caching MLR video segments with enough redundancy to ensure MLR videos are recovered with a target probability (*cache selection*), while optimally distributing chunks across MDCs to lower the distributed segment size (*cache allocation*), and therefore allow more MLR videos to be recovered locally.

The terms we use to explain our methods and related works are as follows: *homogeneous* and *heterogeneous* MDCs refer to MDCs with availability and preferences for videos that are independently and identically distributed (i.i.d) and independently but not identically distributed (i.n.d), respectively. *Symmetric maximal spreading allocation*, a term we borrowed from [9]–[11], divides the total storage budget of a video equally across all MDCs. *Recovery probability* is the probability that a segment is recovered locally. On-time recovery of all segments of a video ensures its QoE (initial delay and no stalling), therefore, we use recovery probability of a video segment as an indicator for capacity and QoE gain.

A. Related Work

Cache selection of videos in cellular networks is an active area of research [12]–[15]. Traditional processor caching policies, e.g., Least Recently Used (LRU) and Least Frequently Used (LFU) are studied for caching videos in cellular networks [12]. Recently, analytics on video servers have enabled caching policies based on videos' popularity rankings, Most Popular Video (MPV) caching policies [12]–[15]. User Preference Profile (UPP) based policies that cache MLR videos proactively or reactively using users' video preferences, achieve high cache hit ratio even for small-sized RAN or device caches [12]. The *cache allocation* optimizes distributed video size for a recovery probability to reduce storage required for single data object [9]–[11] or allow for more videos cached locally [13], [14]. The optimal *cache allocation* for maximum recovery probability from homogeneous caches, given a total cache budget and access pattern, is combinatorial even for one data object, but symmetric maximal spreading allocation is asymptotically optimal [9]. For heterogeneous MDCs an optimal solution based on large deviation inequalities and convex optimization asymptotically maximizes recovery probability for a single data object [11].

B. Our Contribution

The uncoded distributed MPV caching that minimizes redundancy and maximizes distributed cache size is NP-hard due to reduction from the set-cover problem, even without accounting for unreliability of MDCs [14]. We formulate the joint *cache selection and allocation*, to maximize the number of video requests served from RLC-DVCS, as a generalized network flow problem. The formulation reduces redundancy in a D2D range for the recovery probability that ensures most video requests are served from local MDCs – videos are cached at MDCs that are available with high probability at the time of requests. We show the formulation is NP-hard and subsequently propose a more tractable disjoint *cache selection* and *allocation* formulation, based on Central Limit Theorem (CLT) [16]. The proposed solution is evaluated across a video database rather than one data object and for both heterogeneous and homogeneous MDCs using Monte Carlo analysis.

The rest of the paper is organized as follows: Section II states the problem formulation and motivates the breakdown of the problem to tractable sub-problems. Section III proposes methods based on approximation for each sub-problem. A distributed alternative using Lagrangian relaxation is proposed in Section IV. Section V outlines a proactive caching policy to illustrate the development feasibility of our proposed video segmentation, cache selection and allocation methods. We present results and conclude the paper in Section VI and VII.

II. Problem Formulation and Overall Approach

This research aims to increase the number of video requests served from local MDCs with their required QoE while benefiting from RLC (reduced scheduler complexity and distributed video size). The objective is achieved by solving a joint video segmentation (to satisfy QoE), cache selection (to identify MLR videos and their target recovery probability) and allocation (to reduce distributed video size) formulation that maximizes the number of video requests served from local MDCs given the available resources and their constraints. If all segments of a video are equally likely to be requested, any video segmentation results in the same distributed video size. Therefore, separation of video segmentation from *cache selection and allocation* has no impact on the optimality of the solution. For clarity, we present a joint cache selection and allocation formulation for videos of one segment, and later explain how to extend it for segmented videos that ensure video streaming QoE.

We formulate the joint cache selection and allocation problem for RLC videos, as a *Stochastic Generalized Maximum Reward Multicommodity Flow Problem* [17] [18]. Fig. 1(a) shows the associated network flow graph. Figs. 1(b)-(e) detail each block of Fig. 1(a). The request (*demands*) of video requester nodes, R_k, for video (*commodities*), V_j, $d_{j,k}$, is a Bernoulli random variable (r.v.) with mean $P_{\text{Req}}^{(k)}(\mathrm{V}_j)$, the probability that R_k requests V_j. The connectivity between MDCs C_i and C_k, $X_{i,k}$, is a Bernoulli r.v. with mean $P_a^{(k)}(\mathrm{C}_i)$. If $i = k$, $P_a^{(k)}(\mathrm{C}_i) = 1$, indicating R_k can always access its own cache, C_k. If $i \neq k$, $P_a^{(k)}(\mathrm{C}_i)$ represents the probability that R_k can reach C_i, capturing the effect of D2D range, mobility, and bandwidth. M video requesters that also serve as caches may request any of the N available videos, V_j $j = 1..N$, each of size $|w_j|$, divided into w_j chunks, where $w_j = \frac{|w_j|}{s_c}$, and s_c is the chunk size.

Each MDC, C_i, caches video V_j ($f_{i,j} = 1$), given the demand distribution of those requester nodes, R_k, that can reach C_i ($\sum_k X_{i,k} d_{j,k}$), its cache capacity (W_i), and V_j's required cache size ($|w_j|$), Fig. 1(b). If $f_{i,j} = 1$ then $f_{i,j,k} = 1$ for all R_k that can reach C_i (flow gain at V_j to capture the caching impact), Fig. 1(c). RLC content and availability of other MDCs, C_i, to serve video requests, allow for fractional flows, where each C_i can cache multiple chunks from V_j, $f_{i,j} \times w_j$. However, as video chunks are not infinitesimally small, a fractional $f_{i,j}$ may result in fractional chunks, e.g., $f_{i,j}|w_j| = 1.6$ chunks, which is rounded to the boundaries of the chunk, e.g., 2. C_i can cache fewer chunks than required for R_k to recover V_j, but R_k can access other MDCs to download chunks of V_j from. $\mathrm{V}_j - \mathrm{R}_k$ is the aggregator node for each V_j and R_k pair, representing all MDCs that R_k can reach to download V_j from. Each ingress flow to $\mathrm{V}_j - \mathrm{R}_k$ is a random variable, representing the event that R_k can reach C_i with any chunks of V_j cached, $f_{i,j,k} X_{i,k}$, Fig. 1(d). The

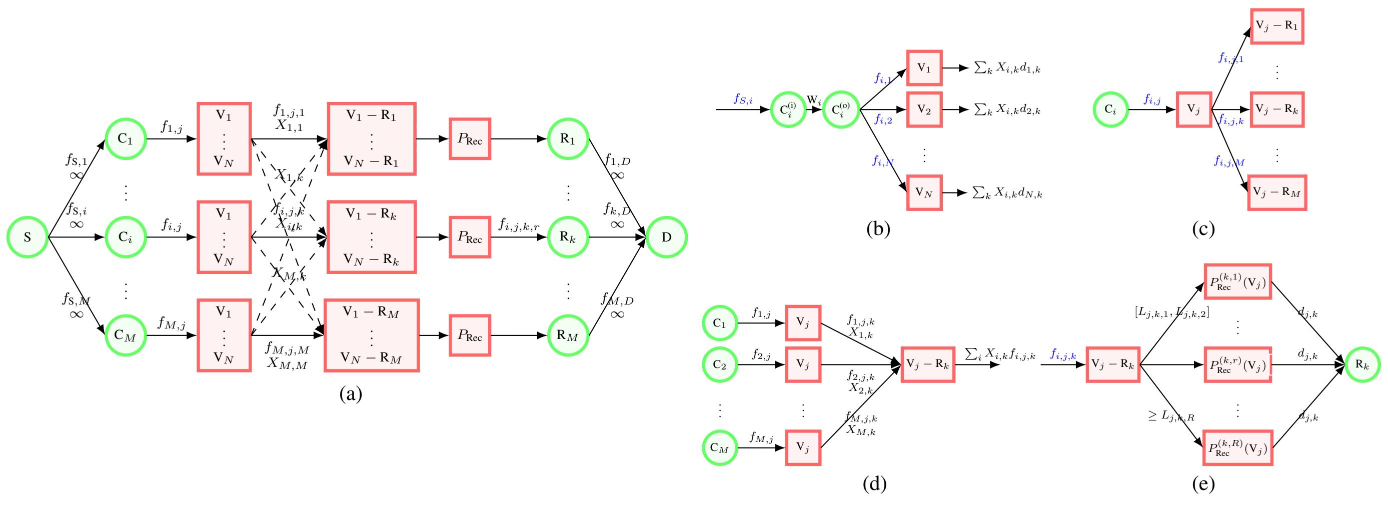


Fig. 1. (a) Maximum reward generalized network flow graph, (b) Edge capacity constraint at C_i, (c) Flow conservation with gain at V_j, (d) Flow conservation at $\mathrm{V}_j - \mathrm{R}_k$, (e) Edge capacity constraint of P_{Rec}.

egress flow from $\mathrm{V}_j - \mathrm{R}_k$ is the sum of ingress flows (flow conservation at $\mathrm{V}_j - \mathrm{R}_k$), representing the event that R_k can recover V_j from local MDCs ($\sum_i X_{i,k} f_{i,j,k}$). $\sum_i X_{i,k} f_{i,j,k}$ is the weighted sum of random variables and should satisfy the edge capacity of one of the $|R|$ available $P_{\text{Rec}}(r)$ nodes, with $r = 1..|R|$. $P_{\text{Rec}}(r)$ is greater than a threshold as a lower recovery probability may not warrant dedicating any cache resource to V_j. $P^{(k,r)}_{\text{Rec}}(\mathrm{V}_j)$ is the recovery probability of V_j by R_k with probability $P_{\text{Rec}}(r)$. Given the mean and variance of r.v. $X_{i,k}$ (edge capacity), there is a one to one mapping between $\sum X_{i,k} f_{j,k,r} = L_{j,k,r}$ and $P_{\text{Rec}}(r)$, Fig. 1(e). Although evaluation of the sum is not possible, as we explain later, it is possible to do worst-case analysis. If any of the P_{Rec} edge capacities is satisfied, the associated output flow is 1, and otherwise 0. The edges between hypothetical source, S, and C_i and R_k and D (hypothetical destination) have infinite capacities. The edges between P_{Rec} nodes and R_k have capacity 1 and gain $d_{j,k}$.

The network flow formulation that maximizes flows of highest gains from S to D, reduces the redundant copies of the MLR videos cached across all collaborting MDCs, given the existing constraints, is given as follows:

Objective:

$$\max \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{r=1}^{|R|} \sum_{k=1}^{M} P^{(k)}_{\text{Req}}(\mathrm{V}_j) P^{(k,r)}_{\text{Rec}}(\mathrm{V}_j) f_{i,j,k,r} \qquad (1)$$

Subject to:

1. $\sum_{j=1}^{N} f_{i,j} |w_j| \le \mathrm{W}_i \qquad \forall i$
2. $f_{i,j} = \max_k (f_{i,j,k}) \qquad \forall i, j$
3. $P\left(\sum_{i=1}^{M} X_{i,k} f_{i,j,k} \ge L_{j,k,r}\right) \ge P^{(k,r)}_{\text{Rec}}(\mathrm{V}_j) \qquad \forall j, k, r$
4. $0 \le f_{i,j,k,r} \le 1 \qquad \forall i, j, k, r$

The solutions, $f_{i,j,k,r}$ (fractional flows), are the optimal *cache selection and allocation* that specify the number of RLC chunks of V_j ($f_{i,j,k,r} \times w_j$) to cache at each C_i to achieve $P^{(k,r)}_{\text{Rec}}(\mathrm{V}_j)$ that maximizes the objective. The 1st constraint is C_i edge capacity constraint and ensures its cache capacity (W_i) is satisfied (edge capacity of C_i, Fig. 1(b)). The 2nd constraint is flow conservation with caching gain at V_j and is set to the maximum flow existing V_j (flow conservation at V_j, Fig. 1(c)). The 3rd constraint is the flow conservation constraint at $\mathrm{V}_j - \mathrm{R}_k$ and ensures the sum of flows received for V_j from all C_i at $\mathrm{V}_j - \mathrm{R}_k$ satisfies the upper/lower limits of the corresponding $P^{(k,r)}_{\text{Rec}}(\mathrm{V}_j)$ (Fig. 1(e)). The 4th constraint is the edge capacity constraint between nodes P_{Rec} and R_k and ensures the selected flow is between 0 and 1. We do not write the flow conservation for any node with ∞ ingress/egress flow.

Solving eq. 1 is hard due to the 3rd constraint, which involves the weighted sum of independent Bernoulli r.v.s with different means, resulting in a Poisson Binomial distribution. Futhermore, this problem is NP-hard as the generalized network flow problem [19] can be reduced to the proposed formulation of eq. 1. Thus, we further divide it into more tractable sub-problems of video segmentation, cache selection, and allocation. We explained separation of video segmentation from cache selection and allocation earlier. Separation of cache selection from allocation is justified as a video segment is recovered successfully if a sufficient number of RLC chunks is recovered before its 1st frame playback deadline – regardless of the cache allocation (whether one or multiple MDCs having supplied the RLC chunks). *Cache selection* formulation determines which videos to cache and their target recovery probability (P_{Rec}) given their request probabilities (P_{Req}) and distributed sizes. For *cache selection*, distributed size of a video segment is determined assuming maximal spreading symmetric allocation across MDCs, which can be sub-optimal. The *cache allocation* minimizes distributed video segment size for the selected P_{Rec} by caching RLC chunks at MDCs so as to reduce distributed video size, ensure their QoE, and potentially satisfy another criteria, e.g., distributability.

III. Solution for Each Sub-Problem

In this section, we detail our proposed algorithms for each of the sub-problems explained above.

A. Division of Videos into Segments for RLC

The partitioning of a video into segments for RLC depends on the application. For offline consumption, the optimal number of segments is one, as it eliminates the Coupon Collector Problem across segments. For video streaming, the segment size must be kept small enough to allow for the on-time decoding of the first video frame of each segment, necessitating division of videos into multiple segments. Segmentation of videos to reduce latency reintroduces a new version of the coupon collector problem since now the downloaded RLC chunks must belong to the desired segment. On the contrary, for a video of only one segment, with high probability, any RLC chunk will provide a degree of freedom (assuming large enough GF size).

As videos' QoE requirements (initial delay and stalling probability) determine the optimal segment size, we use Leaky Bucket Parameters (LBP) or Time Varying LBP (TV-LBP) [20] [12] for video segmentation. LBPs consist of N 3-tuples ($R_{\min}$,$F_{\min}$,$B_{\max}$) corresponding to N sets of transmission rates and buffer size parameters for a given video. Complying with any of the LBP 3-tuples ensures stall-free playback of the video if video playback does not start until after an initial delay of $T_{\text{init}} = \frac{F_{\min}}{R_{\min}}$, and the transmission rate does not fall below required $R_{\min}$ of each video segment. The video segmentation described in this section exploits the properties implied by the LBP to determine the segment sizes. If video playback does not start until downloading $F_{\min}$ bits, setting the first RLC segment size to $F_{\min}$ is intuitive. However, this entails selecting a specific 3-tuple associated with $F_{\min}$ – i.e., making a specific assumption about $R_{\min}$, which we refer to as R_{seg}. The initial playback delay T_{init} serves both to improve video QoE by absorbing the video frame size fluctuations and to allow for larger RLC segment sizes than if the video had been played back without delay. For RLC, R_{seg} must be a realistic representation of the expected transmission rate at the time of video request and one that results in a sufficiently large $F_{\min}$ to benefit RLC. The discussions on the selected $R_{\min}$ for a video and its impact on the initial delay can be found in Appendix. II of [12] and in [20]. $R_{\min}$ for each video can result in any of the two regions for T_{init}: $R_{\min} \geq R_{\text{Thresh}}$, when the transmission rate is high enough to not necessitate an initial playback buffer for absorbing video frame size fluctuations and only requires receiving of the first frame for playback. If $R_{\min} < R_{\text{Thresh}}$, as $R_{\min}$ increases, required $F_{\min}$ decreases and therefore T_{init} decreases both due to increased $R_{\min}$ and decreased $F_{\min}$. Therefore, we select an R_{seg} which results in an $F_{\min}$ greater than I-frame but one that can still be used by a range of achieved $R_{\min}$.

After determining the transmission rate R_{seg} for video segmentation, RLC-DVCS sets the first segment size of V_j, $|w_{j,1}|$, to $F_{\min}$ corresponding to $R_{\min} = R_{\text{seg}}$, as the first video frame is played back after receiving $F_{\min}$ video bits. Any subsequent video segment is set to the number of bits that can be downloaded during the playback time of the preceding segment. For example, $|w_{j,2}|$ is set to the number of bits that can be downloaded during the time it takes to play back $F_{\min}$, denoted by $T_p(F_{\min})$. The video segmentation can be expressed recursively as:

$$|w_{j,1}| = F_{\min}$$
$$|w_{j,l}| = T_p(\text{V}_{j,l-1}) \times R_{\text{seg}} \quad \forall l = 2, ..., L$$

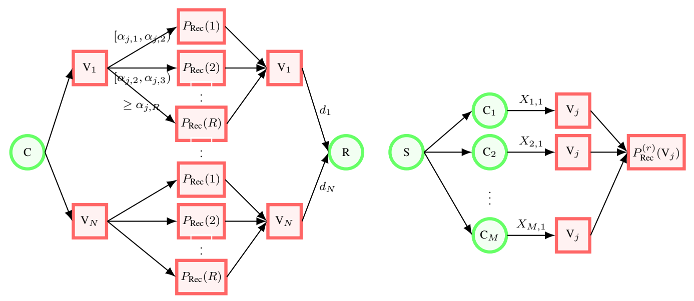
Note that other criteria such as energy efficiency can be incorporated when making video segmentation decisions similar to the methods proposed in [21]. After deciding on the segment sizes of a video, each segment is divided into codeblocks of equal size (s_c), where codeblock size is multiple of the GF symbol size. Therefore, V_j is divided into $L(\text{V}_j)$ segments and the number of codeblocks for segment l of V_j is $w_{j,l} = \lceil \frac{|w_{j,l}|}{s_c} \rceil$. The last codeblock, which is generally only partially filled, is zero-padded to reach the size s_c bits. We use the term chunk to refer to codeblocks for compatibility with [5], however chunks as intended in [5] have equal playback times while codeblocks have equal sizes measured in bits. Next, given the video segmentation, we detail the *cache selection* and *allocation* algorithms that achieve tractable solutions to the formulation of section II.

B. Cache Selection

In this subsection, we relax the problem formulation of section II to make it more tractable for *Cache Selection*. First, explicit video requesters ($\text{R}_k, k = 1, ..., M$, having video demand $P^k_{\text{Req}}(\text{V}_j)$ for V_j and cache connectivity $X_{i,k}$ ($P^k_a(\text{C}_i)$) are replaced with a generic requester (R, with demand $P^{(k)}_{\text{Req}}(\text{V}_j)$ and availability X_i ($p_i = P_a(\text{C}_i)$) – i.e., both demand and connectivity are aggregated across requesters (R_k). R represents any of the uniformly distributed requesters within a cell, with a UPP equal to the average UPP of the requesters with similar UPPs. Second, we assume a finite set of $P_{\text{Rec}} \geq 0.5$ (e.g., $P_{\text{Rec}} = \{0.65, 0.75, 0.85, 0.95\}$) to choose from, leading to caching only videos that result in non-negligible P_{Rec}. Finally, we assume maximum spreading symmetric allocation for *cache selection*, which simplifies the analysis of the 3rd constraint of eq. 1, from that of the Cumulative Distribution Function (CDF) of Poisson binomial distribution to binomial distribution. Therefore, individual C_i in Fig. 1, can be replaced with one common C that caches $\alpha_{j,l,r}$ RLC chunks for each video segment $\text{V}_{j,l}$. Fig. 2(a) shows the simplified network flow for *cache selection* given the stated assumptions. The upper and lower bound capacities of each $P_{\text{Rec}}(r)$ edge, are identified in terms of $\alpha_{j,l,r}$ and $\alpha_{j,l,r+1}$, for $P_{\text{Rec}}(r)$ and $P_{\text{Rec}}(r+1)$, respectively, assuming symmetric maximal spreading as:

$$P\left(\alpha_{j,l,r} \sum_{i=1}^{M} X_{i,j,l} \geq w_{j,l}\right) = P_{\text{Rec}}(r) \tag{2}$$

$\sum X_{i,j,l}$ is the sum of i.n.d Bernoulli r.v.s that can be approximated by a Gaussian distribution due to a variant of the Central



(a) (b)

Fig. 2. Network flow: (a) *Cache Selection* assuming maximal spreading symmetric allocation to find $P^{(r)}_{\text{Rec}}(\text{V}_j)$ for each V_j, (b) *Cache Allocation* across C_i for each V_j selected for caching to achieve $P^{(r)}_{\text{Rec}}(\text{V}_j)$.

Limit Theorem (CLT) [16] [22]. The approximation is valid as the 3rd moment of the Bernoulli r.v. satisfies the Lyapunov condition required for the validity of CLT for i.n.d. random variables [22]. For i.i.d. $X_{i,j,l}$s, the approximation is due to the standard CLT. Next, approximating the (Poisson) Binomial distribution of eq. 2 with a corrected Gaussian Distribution, the minimum $\alpha_{j,l,r}$ required for $P_{\text{Rec}}(r)$ is as follows:

$$\alpha_{j,l,r} = \frac{w_{j,l}}{\sigma_{j,l}\Phi^{-1}(1 - P_{\text{Rec}}(r)) + \mu_{j,l}} \tag{3}$$

$\mu_{j,l}$ and $\sigma_{j,l}$ are the mean and Standard Deviation (SD) of MDCs' availability, for each $\text{V}_{j,l}$, respectively. For heterogeneous MDCs, $\mu_{j,l}$ and $\sigma_{j,l}$ are the mean and SD of Poisson Binomial distribution: $\mu_{j,l} = \sum_{i=1}^{M} p_{i,j,l}$ and $\sigma^2_{j,l} = \sum_{i=1}^{M}(1 - p_{i,j,l})p_{i,j,l}$, where $p_{i,j,l} = P_a(\text{C}_i)$ for $\text{V}_{j,l}$. For the homogeneous MDCs, or heterogeneous MDCs with the assumption of symmetric availability ($p = \frac{\sum p_i}{M}$), $\mu_{j,l}$ and $\sigma_{j,l}$ are the mean and SD of a binomial distribution. The approximation is valid for $np \geq 5$ and $n = M$, p is C's availability. Gaussian approximation is not valid for $np < 5$, representing the case where insufficient number of neighbors makes replication benefit similar to RLC. In Fig. 2, the lower bound capacity is set to $\alpha_{j,l,r}$ (for edge $P_{\text{Rec}}(r)$) and upper bound to $\alpha_{j,l,r+1}$, (for edge $P_{\text{Rec}}(r + 1)$), where $\alpha_{j,l,r}$ is the minimum number of RLC chunks required for each video segment to achieve $P_{\text{Rec}}(r)$. The *cache selection* formulation that maximizes the weighted sum of the flows with the highest demands for each video segment given the resources and their constraints is:

Objective:

$$\max \sum_{j=1}^{N} \sum_{l=1}^{|L(v_j)|} \sum_{r=1}^{|R|} f_{j,l,r} P_{\text{Req}}(\text{V}_{j,l}) P_{\text{Rec}}(r) \tag{4}$$

Subject to:

1. $\sum_{j=1}^{N} \sum_{l=1}^{|L(\text{V}_j)|} \sum_{r=1}^{|R|} f_{j,l,r}|w_{j,l}| \leq \text{W}_i \quad \forall i$
2. $\alpha_{j,l,r} \leq f_{j,l,r} \times w_{j,l} < \alpha_{j,l,r+1} \quad \forall r$
3. $f_{j,l,r} \in [0, 1] \quad \forall j, l, r$

$f_{j,l,r}$ are the solutions, $0 \leq f_{j,l,r} \leq 1$, where $f_{j,l,r}w_{j,l}$ represents the number of RLC chunks of $\text{V}_{j,l}$ to cache at each MDC, C_i, to achieve the recovery probability $P_{\text{Rec}}(r)$ for $\text{V}_{j,l}$ (3rd constraint). The 1st constraint is the cache size constraint at C_i. The 2nd constraint ensures P_{Rec} that satisfies the corresponding α range is selected for caching, which is an edge capacity constraint of $P_{\text{Rec}}(r)$ nodes.

C. Cache Allocation

Cache allocation minimizes distributed video size by caching RLC chunks optimally at MDCs to achieve the target recovery probability (P_{Rec}). Fig. 2(b) shows the flow graph of the *cache allocation* problem. The *cache allocation* formulation that minimizes distributed video size given MDCs availability and the target P_{Rec}, derived by solving *cache selection* problem, is:

Objective:

$$\min_{\alpha_{i,j,l}} \sum_{i=1}^{M} \alpha_{i,j,l}$$

Subject to: (5)

1. $P\left(\sum_{i=1}^{M} \alpha_{i,j,l} X_{i,j,l} \geq w_{j,l}\right) \geq P^{(r)}_{\text{Rec}}(\text{V}_{j,l})$
2. $0 \leq \alpha_{i,j,l} \leq w_{j,l}$

The 1st constraint ensures the recovery probability of $\text{V}_{j,l}$ is at least $P_{\text{Rec}}(\text{V}_{j,l})$, set in the *cache selection* step. The 2nd contraint limits the number of RLC chunks cached at each C_i to the number of RLC chunks of video segment $\text{V}_{j,l}$ ($w_{j,l}$), as otherwise, it results in no additional degrees of freedom.

To solve this formulation, we re-write the 1st constraint as a Second Order Cone Programming (SOCP) constraint, similar to the method used in portfolio optimization and asset allocation problem [23]. Subtracting $\mu_{j,l} = \sum_{i=1}^{M} \alpha_{i,j,l}\mu_{i,j,l}$ from both side of the above inner inequality and dividing both by $\sigma_{j,l} = \sqrt{\sum_{i=1}^{M} \sigma^2_{i,j,l}\alpha^2_{i,j,l}}$ allows for applying CLT-like results similar to eq. 3. The 1st constraint of Eq. 5, using complementary CDF of the Gaussian distribution is as follows:

$$w_{j,l} - \mu_{j,l} \leq \sigma_{j,l}\Phi^{-1}(1 - P_{\text{Rec}}(\text{V}_{j,l})) \tag{6}$$

$\Phi^{-1}(1 - P_{\text{Rec}}(\text{V}_{j,l}))$ is negative for $P_{\text{Rec}}(\text{V}_{j,l})$ between 0.5 and 1, therefore the first constraint is convex. For simplicity of notation, we set: $\Phi_{\text{Rec}}(\text{V}_{j,l}) = \left|\Phi^{-1}(1 - P_{\text{Rec}}(\text{V}_{j,l}))\right|$. *Cache allocation* formulation, for each video segment $\text{V}_{j,l}$, is SOCP – with M variables ($\alpha_{i,j,l}$s) and one SOCP constraint – as it requires the affine function, $\alpha^T_{j,l}1$, to lie in the second-order cone.

IV. Distributed Allocation of RLC Chunks

In this section, using Lagrangian relaxation of eq. 5 each MDC minimizes $\alpha_{i,j,l}$ required to achieve $P_{\text{Rec}}(\text{V}_{j,l})$ with limited feedback from other MDCs or a centralized scheduler. To apply Lagrangian relaxation, we change the SOCP constraint of eq. 5 to the Quadratically Constrained Quadratic Program (QCQP). Raising both sides of eq. 6 to the power of two:

$$\sum_{i=1}^{M} \sigma_{i,j,l}^2 \alpha_{i,j,l}^2 \le \frac{(\sum_{i=1}^{M} \mu_{i,j,l}\alpha_{i,j,l})^2}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2} + \frac{w_{j,l}^2}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2} - \frac{2w_{j,l}\sum_{i=1}^{M} \mu_{i,j,l}\alpha_{i,j,l}}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2} \quad \forall j,l \tag{7}$$

Next, we replace the first term of the right-hand-side (RHS) inequality with its upper bound using Jensen's inequality [24]. Jensen's inequality is valid for any convex function φ with nonnegative real numbers, $\lambda_1,..,\lambda_n$, for which $\sum_{i=1}^{n}\lambda_i = 1$, and can be stated as: $\varphi\left(\sum_{i=1}^{M}\lambda_i x_i\right) \le \sum_{i=1}^{M}\lambda_i\varphi(x_i)$. To apply Jensen's inequality, we divide the first term of the RHS of eq. 7 by $\sum_{i=1}^{M}\mu_{i,j,l}$ and define $\mu'_{i,j,l} = \frac{\mu_{i,j,l}}{\sum_{i=1}^{M}\mu_{i,j,l}}$ and $\Phi'_{\text{Rec}}(\text{V}_{j,l}) = \frac{\Phi_{\text{Rec}}(\text{V}_{j,l})}{\sum_{i=1}^{M}\mu_{i,j,l}}$. Subsequently, we rewrite eq. 7 as:

$$\sum_{i=1}^{M}\left(\sigma_{i,j,l}^2 - \frac{\mu'_{i,j,l}}{\Phi'_{\text{Rec}}(\text{V}_{j,l})^2}\right)\alpha_{i,j,l}^2 + \frac{2w_{j,l}\sum_{i=1}^{M}\mu_{i,j,l}\alpha_{i,j,l}}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2} - \frac{w_{j,l}^2}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2} \le 0 \tag{8}$$

Lagrange dual of eq. 5, with its constraint replaced by eq. 8 and setting $\Lambda_{i,j,l} = 1 + \frac{2\lambda w_{j,l}\mu_{i,j,l}}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2}$ and $\Gamma_{i,j,l} = (\sigma_{i,j,l}^2 - \frac{\mu'_{i,j,l}}{\Phi'_{\text{Rec}}(\text{V}_{j,l})^2})$, can be written as:

$$\min_{\alpha_{i,j,l}}\max_{\lambda\ge 0}\sum_{i=1}^{M}\Lambda_{i,j,l}\alpha_{i,j,l} + \lambda\left(\sum_{i=1}^{M}\Gamma_{i,j,l}\alpha_{i,j,l}^2 - \frac{w_{j,l}^2}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2}\right) \tag{9}$$

λ is the multiplier of the recovery probability constraint, and can be interpreted as the cost of serving video segments that are not recovered from local MDCs, from cellular links. Given an initial assumption for λ and as eq. 9 is the sum of positive numbers, like in [25], we decompose eq. 9 into sub-problems (eq. 10), where each MDC C_i calculates $\alpha_{i,j,l}$ using:

$$\min_{\alpha_{i,j,l}} \Lambda_{i,j,l}\alpha_{i,j,l} + \lambda(\Gamma_{i,j,l}\alpha_{i,j,l}^2) \tag{10}$$

Differentiating eq. 10 with respect to $\alpha_{i,j,l}$ yields $\alpha_{i,j,l} = \frac{-\Lambda_{i,j,l}}{2\lambda\Gamma_{i,j,l}}$. Now, we write eq. 9, the original dual problem that updates the dual variable λ, as:

$$\begin{aligned} \text{minimize:} \quad & \textstyle\sum_i g_i(\lambda) - \lambda\frac{w_{j,l}^2}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2} \\ \text{subject to:} \quad & \lambda \ge 0 \end{aligned} \tag{11}$$

Where $g_i(\lambda)$ is the dual function and the maximum value of the Lagrangian solved for each sub-problem (eq. 10) for a given λ. Subgradient for each $g_i(\lambda)$ is:

$$s_i(\lambda) = -\Gamma_{i,j,l}{\alpha_{i,j,l}^*}^2 + \frac{2w_{j,l}\sum_i \mu_{i,j,l}\alpha_{i,j,l}^*}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2} \tag{12}$$

The global subgradient is $s(\lambda) = \sum_i s_i(\alpha_{i,j,l}^*(\lambda)) - \frac{w_{j,l}^2}{\Phi_{\text{Rec}}(\text{V}_{j,l})^2}$. Solving this formulation iteratively converges to an optimal solution for $\alpha_{i,j,l}$. This approach is only applicable if strong duality holds [25]. For the considered P_{Rec}s in this paper, the formulation is convex and the strong duality holds if D2D range is greater than a threshold, as there exists an $\alpha_{i,j,l}$ for which the constraint is strictly feasible ($\lambda > 0$). Each MDC solves the optimization of eq. 10 to determine the required $\alpha_{i,j,l}$ to contribute to selected $P_{\text{Rec}}(\text{V}_{j,l})$, given the knowledge of mean and variance of the MDCs' availability, which is captured using network analytics given MDCs availability pattern and communicated to them as sideband signaling.

```
RLC-based Proactive Caching Policy
Cache Population
1.  Use Cache Selection to find $P_{\text{Rec}}^{(r)}(\text{V}_{j,l})$ for each $\text{V}_{j,l}$;
2.  $L_{\text{cache}}$: Sorted list of videos in descending order of $P_{\text{Rec}}$ and $P_{\text{Req}}$
3.  For each $\text{V}_{j,l} \in L_{\text{cache}} \cap \text{W}_i > 0$
4.     Run cache allocation formulation and decide on $\alpha_{i,j,l}$
5.     $\text{W}_i = \text{W}_i - \alpha_{i,j,l} \quad \forall i$
6.  End For

Video Request
7.  For each video segment, $\text{V}_{j,l}$, requested by $\text{R}_k$
8.     $\mathcal{S}$=Find 1-hop neighbors and request $\alpha_{j,l}$
9.     If $\sum_{i=1}^{|\mathcal{S}|}\alpha_{i,j,l} \ge w_{j,l}$
10.       Download $\alpha_{j,l}$ from 1-hop neighbors closest to $\text{R}_k$
11.    Else If $\sum_{i=1}^{|\mathcal{S}|}\alpha_{i,j,l} < w_{j,l}$
12.         Download $\text{V}_{j,l}$ from cellular links
13. End For, End If
```

Fig. 3. RLC-based Proactive Caching Policy.

V. RLC-based Proactive Caching Policy

Fig. 3 details an RLC-based proactive mobile device caching policy (RLC-P-MDC), that caches video segments ($\text{V}_{j,l}$) proactively at MDCs. RLC-P-MDC identifies videos to cache and their target P_{Rec} using *cache selection* formulation of subsection III-B (line 1). In *distributed allocation* RLC-P-MDC transfers metadata containing video segment size, and mean and variance of MDCs' availability to each C_i for each $\text{V}_{j,l}$ selected for caching, sorted in descending order of $P_{\text{Req}}(\text{V}_{j,l})$ and $P_{\text{Rec}}(\text{V}_{j,l})$. Subsequently, MDCs derive the number of RLC chunks, of each $\text{V}_{j,l}$, to cache using distributed *cache allocation* (lines 2-6). MDCs update the available cache size and schedule RLC chunks via cellular or alternative networks (lines 4-5). In *centralized allocation*, RLC-P-UPP runs *cache allocation* algorithm to identify $\alpha_{i,j,l}$ that each MDC has to cache, and pushes content to MDCs. Note that the metadata can be multicasted to MDCs with limited communication overhead. In the event of video request, requester recovers as many RLC chunks required for the decoding of the video segments from its 1st-hop neighbors. If the number of recovered RLC chunks is not sufficient for decoding of the video segment, it downloads the entire segment from cellular links (lines 7-14).

VI. Simulation Environment and Results

Using Monte Carlo analysis, we benchmark performance of the centralized and distributed RLC-DVCS against symmetric maximum spreading and uncoded-DVCS. We assume 200 MDCs, with availability probability $P_a(\text{C}_i) = p_i \times \frac{r^2}{R^2}$, uniformly distributed within a cell with radius $R = 500$m. p_i captures the effect of availability due to mobility, with

$p_i = 1$ accounting for availability only due to the D2D range of r. MDCs make 1000 requests to a database containing 200 videos of one common video category with popularity ranking that follows a Zipf distribution with $\alpha = 0.8$ [12]. Video duration is exponentially distributed with mean of 8 min, truncated to maximum of 30 min, and minimum of 2 min, while video bitrate is uniformly distributed between 200 kbps (QVGA quality) and 2 Mbps (HD quality). Each video is divided into segments using the methodology of section III.A., with accepted initial delay of 10s.

We study the impact of the distributed cache size, which is proportional to the availability probability (p), D2D range (r), and per-MDC cache size (W), as well as the number of MDCs (M), on the recovery probability of the proposed methods without considering the effect of path-loss and interference from neighboring MDCs. Fig. 4 shows the average number of neighbors, derived from simulation for each considered combinations of p and r, and is highest for $p = 1$, r=300 m and M=200, and lowest for $p \in [0.4, 0.6]$, r=100 m and M=100.

Fig. 5(a) shows the impact of cache size and D2D range on the performance of the allocation methods while Fig. 5(b) shows the impact of availability probability and D2D range. Fig. 5(a) shows that RLC-DVCS with centralized allocation improves the recovery probability over the symmetric allocation and uncoded-DVCS, regardless of the cache size and D2D range. e.g., for W=50k and r=300m, centralized allocation improves the recovery probability of the symmetric allocation and uncoded-DVCS by 74% and 179%, respectively. The superior performance of the centralized RLC-DVCS can be attributed to the *cache allocation* optimization of subsection III, that further optimizes distributed video size for the target recovery probability.

Distributed allocation enables distributed caching decision per-MDC, while still improving the recovery probability over the symmetric allocation and uncoded-DVCS regardless of the cache size and D2D range. e.g., distributed allocation improves the recovery probability over symmetric and uncoded-DVCS by 38% and 122%, respectively, for W=50k and r=200m. This shows the effectiveness of the optimization relaxation proposed in Section IV.

Overall we observe that asymmetric allocation improves the recovery probability over the symmetric allocation even for homogeneous MDCs (comparing the centralized vs. symmetric allocation for $p = 1$). The effectiveness of the *cache allocation* optimization, for the considered settings, where the storage size of the MLR videos is larger than the total available distributed cache size, can be explained by the property of the sum and weighted sum of the independent Bernoulli r.v. For independent Bernoulli r.v., X_i, with $P(X_i = 1) = p_i, i = 1, 2, .., n$, $P(\sum X_i \geq t)$ is Schur-convex in p (order preserving function), when $0 \leq t \leq n\bar{p} - 2$ [26]. For such X_i, $\log(X_i)$ is log-convex and $\log(a) \prec \log(b)$ results in $P(\sum a_i X_i > t) \geq P(\sum b_i X_i > t)$ (Theorem 1 of [27]). Schur-convexity implies that if x majorizes y ($y \prec x$), then $f(x) \geq f(y)$; indicating that asymmetric allocation outperforms symmetric allocation. A real vector $b = (b_1, .., b_n)$ majorizes $a = (a_1, .., a_n)$, written as $a \prec b$, if (i) $\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} b_i$, (ii) $\sum_{i=k}^{n} a_{(i)} \leq \sum_{i=k}^{n} b_{(i)}, k = 2, .., n$, where $(a_1, .., a_n)$ and $(b_1, .., b_n)$ are arranged in increasing order, respectively. Had the total storage exceeded the required cache size of the MLR videos by $n\bar{p}+1 \leq t \leq n$, where $\bar{p} = \frac{\sum p_i}{n}$ [26], the symmetric allocation would have been optimal as $P(\sum X_i \geq t)$ is Schur-concave for the range.

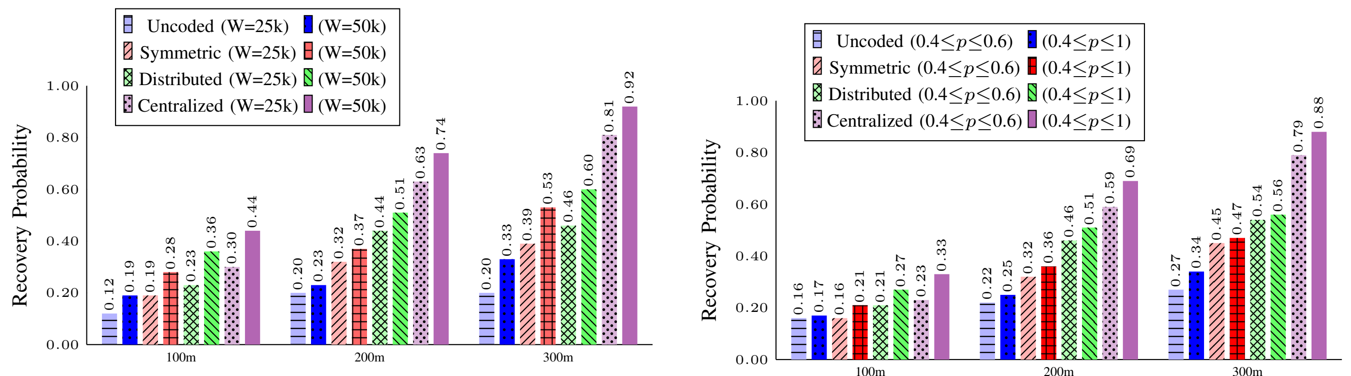
Fig. 5(b) shows the impact of the lower availability probabilities of $p \in [0.4, 0.6]$ and $p \in [0.4, 1]$ on the performance of the allocation methods for heterogeneous MDCs and their abilities to introduce redundancy to counter the effect of lower availability. We observe that lower availability probability, for otherwise similar configurations of Fig. 5(a), reduces the recovery probability regardless of the allocation method. Even in this case, the centralized and distributed allocations achieve higher recovery probability than the symmetric allocation and uncoded-DVCS, regardless of the D2D range. e.g., centralized allocation improves the recovery probability over the symmetric allocation and uncoded-DVCS by 87% and 159%, respectively, when r=300m, 0.4≤p≤1 and W=50k. The magnitude of improvements due to the centralized allocation is more pronounced for the larger distributed cache sizes. e.g., it increases the recovery probability of the uncoded-DVCS by 59, 54, and 52 percentage points, and symmetric allocation by 39, 41, and 34 percentage points for $p = 1$, 0.4≤p≤1 and 0.4≤p≤0.6, respectively, when W=50k, r=300m.

When p=1 (Fig. 5(a)), the magnitude of improvements due to increasing cache size from W=25k to W=50k is most pronounced for r=100m, due to the smaller distributed cache, regardless of the RLC allocation scheme. When reducing the availability probabilities (Fig. 5(b)), the larger cache size is used to compensate for the lower availability and reduces the impact of larger caches in improving recovery probability. Comparing r=100m for different availability probabilities and cache sizes, we see that uncoded-DVCS with W=25k and p=1 results in lower recovery probability than W=50k and 0.4≤p≤0.6, unlike RLC-DVCS; indicating the effectiveness of RLC in reducing the performance degradation due to the coupon collector problem associated with uncoded-DVCS.

Overall, we infer that for the considered settings in this paper: (a) RLC improves DVCS performance regardless of the allocation method (symmetric, distributed or centralized), (b) centralized RLC-DVCS improves the recovery probability of uncoded-DVCS and DVCS with symmetric allocation significantly by upto 159% and 87%, (c) distributed RLC-DVCS improves the recovery probability of uncoded-DVCS and symmetric DVCS by 122% and 38% while facilitating distributed caching decisions per-MDC, (d) Centralized and distributed allocations can effectively utilize the higher

(p=1, $\mathbf{p} \in [0.4, 1]$, $\mathbf{p} \in [0.4, 0.6]$)			
MDCs	**d2d=100 m**	**d2d=200 m**	**d2d=300 m**
100	**(3.8, 2.6, 2.0)**	**(15.7, 10.8, 7.6)**	**(35, 24, 17)**
200	**(7.5, 5.5, 3.7)**	**(30, 25, 15)**	**(70, 49, 35)**

Fig. 4. Average number of neighbors for considered combinations of availability probability, D2D range and number of MDCs.



(a) Availability probability $P_a(\mathrm{C}_i) = \frac{r^2}{R^2} \times p$, $p = 1$

(b) Per-MDC cache size W=50k

Fig. 5. Recovery probability of different allocation methods for 200 MDCs and D2D ranges of 100m, 200m, and 300m for, (a) $p = 1$ and W=25k and W=50k, (b) W=50k and $0.4 \leq p \leq 0.6$ and $0.4 \leq p \leq 1$.

distributed cache size to improve the recovery probability compared with symmetric and uncoded-DVCS.

VII. Conclusion

In this paper we proposed a random linear coded distributed video caching and sharing system that caches videos most likely to be requested accounting for MDCs' cache size and compensating for their probabilistic availability. The proposed methods ensure distributability and cache few coded chunks on many MDCs, thereby reducing the resources required from each MDC to assist subsequent requests. We formulated cache selection and allocation as a stochastic multicommodity information flow problem, showed its complexity and subsequently divided the problem into more tractable sub-problems which were solved using approximation methods. We proposed a segmentation that ensures QoE of random linear coded videos. Furthermore, we proposed a distributed approach where each MDC decides on the amount of storage to dedicate to each video with limited coordination among MDCs. Our simulation results showed the superiority of our centralized and distributed allocation methods relative to symmetric allocation and uncoded-DVCS.

Acknowledgements

This material is based upon research supported by the Chateaubriand Fellowship of the Office for Science & Technology of the Embassy of France in the United States.

References

[1] S. Acedanski, S. Deb, M. Medard, and R. Koetter, "How good is random linear coding based distributed networked storage?" *NetCod*, 2005.
[2] A. Le, H. Seferoglu, B. Cici, C. Fragouli, and A. Markopoulou, "Microcast: Cooperative video streaming using cellular and D2D connections," *NetCod*, 2014.
[3] J. Jiang, S. Zhang, B. Li, and B. Li, "Maximized cellular traffic offloading via device-to-device content sharing," *IEEE Journal on Selected Areas in Communications*, 2016.
[4] V. Sciancalepore, D. Giustiniano, A. Banchs, and A. Picu, "Offloading cellular traffic through opportunistic communicationsl analysis and optimization," *NetCod*, 2014.
[5] H. A. Pedersen and S. Dey, "Video and network aware mobile device caching and sharing in cellular networks," *Under Review*.
[6] A. Altieri, P. Piantanida, L. R. Vega, and C. G. Galarza, "On fundamental trade-offs of device-to-device communications in large wireless networks," *IEEE Transactions on Wireless Communications*, 2015.
[7] M. Gregori, J. Gomez-Vilardebo, J. Matamoros, and D. Gunduz, "Wireless content caching for small cell and d2d networks," *IEEE Transactions on Wireless Communications*, 2016.
[8] T. Locher, P. Moor, S. Schmid, and R. Wattenhofer, "Free riding in bittorrent is cheap," *HotNets*, 2006.
[9] D. Leong, A. G. Dimakis, and T. Ho, "Distributed storage allocations," *HotNets*, 2011.
[10] C. S. Chang, T. Ho, M. Effros, M. Medard, and B. Leong, "Issues in peer-to-peer networking: a coding optimization approach," *2010 IEEE International Symposium on Network Coding*, June, 2010.
[11] V. Ntranos, G. Caire, and A. G. Dimakis, "Allocations for heterogeneous distributed storag," *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, 2012.
[12] H. Ahlehagh and S. Dey, "Video aware scheduling and caching in the radio access network," *IEEE Transactions on Networking*, August, 2014.
[13] N. Golrezaei *et al.*, "Device-to-device collaboration through distributed storage," *Proceedings of IEEE GLOBECOM*, 2012.
[14] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *INFOCOM*, 2013.
[15] V. Poliakov, L. Sassatelli and D. Saucez, "Impact of caching on http adaptive streaming decisions: towards an optimal," *Computer Communication Workshops, IEEE Conference*, 2016.
[16] D. Hunter, "CLT, Part II: Independent but not identically distributed," *http://sites.stat.psu.edu/˜dhunter/asymp/fall2002/lectures/ln04.pdf*.
[17] R. K. Ahuja, J. B. Orlin, and T. L. Magnanti, "Network flows: Theory, algorithms, and applications," *Book*, 1994.
[18] J. Kleinberg and E. Tardos, "Algorithm design," *Pearson Education*, 2018.
[19] S. Sahni, "Computationally related problems," *SIAM Journal Computer*, December 1974.
[20] H. A. Pedersen and P. Cosman, "Ensuring over-the-top video QoE using video aware scheduling," *Under Review*.
[21] M. Siekkinen, M. A. Hoque, and J. K. Nurminen, "Using viewing statistics to control energy and traffic overhead in mobile video streaming," *IEEE Transactions on Networking*, 2015.
[22] J. Pitman, "Lyapunov condition," *http://www.stat.berkeley.edu/ pitman/s205f02/lecture10.pdf*.
[23] S. Boyd and L. Vandenberghe, "Convex optimization," *Cambridge University Press*.
[24] W. Feller, "An introduction to probability theory and its applications," *Wiley Series In Probability and Mathematical Statistics*, 1950.
[25] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition," *Proceedings of the IEEE*, 2007.
[26] M. Merkle and L. Petrovic, "Inequalities for sums of independent geometrical random variables," *Aequations Mathematicae*, 1997.
[27] Y. Yu, "Some stochastic inequalities for weighted sums," *Bernoulli 17(3)*, 2011.