Demand-aware Cooperative Content Caching in 5G/6G Networks with MEC-enabled Edges

Tadege Mihretu Ayenew, Dionysis Xenakis, Luis Alonso, Nikos Passas, Lazaros Merakos

***Abstract*—Today, billions of smart devices are interconnected via wireless networks, leading to large volumes of video contents circulating through the bandwidth-limited backhaul. This causes network performance to deteriorate. As a mitigation mechanism, caching of highly popular contents to network edges is deployed. We propose a cooperative and demand-aware caching strategy, which is modelled using the Separable Assignment Problem, to maximize the cache hit ratio. This problem is solved with a recursive enumeration method, where dynamic programming is used to fill each edge. The extensive application-level evaluations show that the proposed strategy outperforms existing caching policies.**

***Index Terms*—Caching, separable assignment problem, MEC, optimization, cache hit ratio**

I. Introduction

In recent years, the traffic volume in cellular networks is exponentially increasing. This proliferation is triggered by the fast growth of technologies such as augmented reality, online gaming, Internet of things (IoT), and high subscription rate to high-bandwidth multimedia services in the heterogeneous cellular network (HCN).

With such an explosive traffic growth, avoiding congestion in a non-flexible and constrained backhaul network is challenging. In addition, *popular contents* are redundantly retransmitted, from the content server to the end user equipment (UE), which increases service cost. Besides, contents have to travel long distances to reach the UEs, which increases end-to-end latency. These challenges make it difficult to meet user expectations for seamless connectivity, high transfer rate, and ultrafast responses.

To address these challenges and meet the expected quality of experience (QoE), *content caching* is used in the emerging networks. Indeed, content caching is a powerful tool to offload backhaul congestion and reduce service latency and utility cost [1], [2]. On top of caching, multi-access edge computing (MEC) further enhances the caching strategy by offloading data analytic computation to network edges, known as mobile helpers (MH).

In [3], MEC has been integrated in the caching system to reduce the completion latency, where the system is modeled by mixed-integer non-linear programming (MINLP) and solved by block successive upper-bound minimization. In [4], authors have proposed a joint caching policy to a MEC-assisted network, by modelling it as a MINLP problem and solved by branch-and-bound. But the central system hardly gets information of the radio network. Similarly, the authors in [5] have used mixed integer programming and solved it by a greedy strategy. In [6], authors have used Multiple-choice Knapsack Problem to model cooperative content caching while the authors in [7] have modelled a joint caching strategy using a reward maximization problem. In [8], a stochastic geometry model is used and solved by an approximation method while greedy approach was used in [9]. However, greedy and heuristic algorithms can not deal with the spatial difference of content popularity and cache constraints. In [10], we proposed a cooperative caching using Multiple Knapsack Problem and solved it using bound-and-bound algorithm, to maximize the hit probability.

Although content caching is well investigated, almost all works assume a global popularity of contents by aggregating requests through all caching edges. This assumption underestimates the relative interests of MHs towards each content and increases the intra-cluster cost. The unique contribution of this work is that we developed a demand-aware caching strategy, which fully addresses the above critical issues, where content popularity varies at each MH. This novel work answered the two interlaced questions: 1) Which content should we prioritize while caching at a MH? 2) At which MH should a content be cached to maximize their impact? We modelled this very realistic problem by the Separable Assignment Problem (SAP) [12] and solved it using an iterative algorithm. In this way, contents are placed at the MH where they are most 'demanded', and maximize the *cache hit ratio* (CHR).

The rest of the paper is organized as follows. Section II presents our system model of cooperative content caching strategy, while Section III describes the proposed solution. Section IV discusses the obtained numerical results and Section V contains our conclusion.

II. System Model

A. System description

We focus on the downlink of a three-tier HCN, shown in Fig. 1. We consider a cluster of MEC-enabled content caching edges: one macro base station (MBS), serving as a central head, and a set $\mathcal{N}$ of MHs (it can be small base stations (SBSs), smart UEs, etc). Each n^{th} MH has a set $\mathcal{U}_n$ of active users associated to it ($n = 1, 2, ..., |\mathcal{N}|$). The MBS gets video contents from service providers and forms a huge library, $\mathcal{M} = \{f_m : 1 \leq m \leq |\mathcal{M}|\}$, where f_m is unique content identifier. The MBS passes popular contents to the MHs, which relay them to the UEs.

Manuscript received xxx xx, 2022; revised Xxxx xx, 2022; accepted Xxxx xx, 2022. This work was partially supported by the project PID2021-125799OA-I00, funded by the Spanish Government. The associate editor coordinating the review of this article and approving it for publication was XXX. (Corresponding author: Tadege M. Ayenew)

T. M. Ayenew and L. Zarate are with the Department of Signal Theories and Communications, Universitat Politècnica de Catalunya (UPC), Spain ({tadege.mihretu.ayenew, luis.g.alonso}@upc.edu)

D. Xenakis, N. Passas, and L. Merakos are with the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, 15784, Athens, Greece ({nio, passas, merakos}@di.uoa.gr)

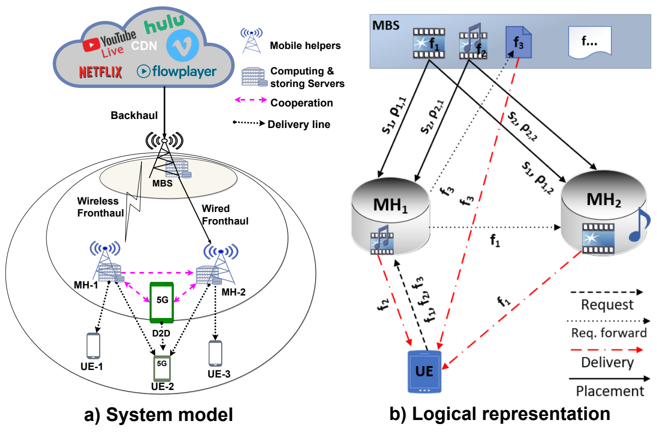

Fig. 1: Cooperative demand-aware caching system model

The MHs in a cluster cooperatively cache subsets of popular video contents, at off-peak times. However, their cache size L_n[bits] is limited. Each content's *popularity* ($\rho_{n,m}$) is different towards each MH while content size (s_m) [bits] does not differ. Here, popularity refers to the probability that a respective content is requested by UEs. For a given $\mathcal{M}$, we have an estimated popularity matrix ($\mathcal{P}$), where $\mathcal{P} = \{\rho_{n,m} : m = 1, ..., |\mathcal{M}|, \forall n \in \mathcal{N}\}$, and a vector of content sizes ($\mathcal{S}$), defined $\mathcal{S} = \left\{s_1, s_2, ...s_{|\mathcal{M}|}\right\}$.

Mainly, we focus on cache optimizing in a single cluster, within which a subset of contents is placed at each MH until its storage is full. The caching decision is centrally made by the cluster head (*i.e.*, MBS). The subset of contents eventually placed at n^{th} MH is denoted by $\mathcal{C}_n$, $\mathcal{C}_n = \{f_m : 1 \leq m \leq |\mathcal{C}_n|, f_m \in \mathcal{M}\}$. Ultimately, the superset of cached contents in the cluster is denoted as: $\mathcal{C} = \bigcup_{n=1}^{|\mathcal{N}|} \mathcal{C}_n$, where $\mathcal{C}_n \subseteq \mathcal{C} \subseteq \mathcal{M}$. To enhance caching cooperation, the MBS sends a cache-decision table $(x_{n,m})$ to each MH. During caching, we consider: i) contents are not partitioned, *i.e.*, either an entire part is cached or not selected at all, ii) there is no content overlap across all MHs (*i.e.*, $\mathcal{C}_n \cap \mathcal{C}_k = \emptyset, n \neq k, k \in \mathcal{N}$), iii) the popularity $\rho_{n,m}$ and impact of a content differs towards each MH.

We assume that any user u requests a content f_m through it's associated relay n, called *local* MH. Every MEC-enabled MH stores its local request rates ($\{\lambda_{n,m}\}$), perform data analytic tasks on radio resources, and passes computed results to the MBS. Mainly, the MBS collects all request rates, Λ=$\{\lambda_{n,m} : \forall f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$ and the following three important parameters are extracted for a given epoch.

1) Content popularity ($\rho_{n,m}$): shows the relative popularity of contents within a given MH. Let m_n denote the rank of a content in the request rate vector of n^{th} MH; it's popularity using Zipf distribution is given by:

$$\rho_{n,m} = \frac{m_n^{-\gamma}}{\sum_{j=1}^{|\mathcal{M}|} j_n^{-\gamma}}, \forall n \in \mathcal{N} \tag{1}$$

where $\gamma \geq 0$ is Zipf parameter (indicates popularity distribution skewness). Generally, $\sum_{m=1}^{|\mathcal{M}|} \rho_{n,m}$=1, $\forall n \in \mathcal{N}$.

2) Spatial request ratio ($\eta_{n,m}$): refers to the relative interest of each MH, towards a content. Using the request rate received by a content across all MHs in a cluster, the MBS estimates the $\eta_{n,m}$ value as:

$$\eta_{n,m} = \frac{\lambda_{n,m}}{\sum_{n=1}^{\mathcal{N}} \lambda_{n,m}}, \forall m \in \mathcal{M} \tag{2}$$

where it holds: $\sum_{n=1}^{|\mathcal{N}|} \eta_{n,m}$=1, $\forall f_m \in \mathcal{M}$. This parameter tells us which MH has higher request for a content.

3) Request probability ($\phi_{n,m}$): is the combined impact of the above two parameters on a content, to be requested by users. It shows the real demand level of a video content with respect to a specific helper n. The $\phi_{n,m}$ value is a linear effect, with their binary linear constraints, as:

$$\phi_{n,m} = \rho_{n,m} \cdot \eta_{n,m} \tag{3}$$

and forms matrix $\mathcal{R} = \{\phi_{n,m} : m = 1, ..., |\mathcal{M}|, \forall n \in \mathcal{N}\}$.

When a user u requests for f_m, through its associated local n, either: i) *cache hit* happens, meaning that f_m is found at n and immediately delivered to u, ii) *local cache miss* happens, meaning that f_m is not found in n but at any neighbouring MH, iii) *cache miss* happens, meaning that f_m is not found in any of MHs in the cluster. In the two last cases, helper n forwards the request either to neighbouring MH or to MBS, by checking from $x_{n,m}$. Then, f_m is directly served to u through a newly established downlink.

Meanwhile, the availability of video contents in the cluster is measured by *cache hit ratio* (ψ), which is the ratio of contents having successful cache hit event with the total received requests [13]. Given $|\mathcal{U}_n|$ active users and request probability ($\phi_{n,m}$) of contents, the weighted number of requests towards f_m is estimated by: $\lambda_{n,m} = |\mathcal{U}_n| * \phi_{n,m}$ [7]. Having this, the CHR in the MEC cluster, defined over $\mathcal{M}$, is estimated as:

$$\psi(\mathcal{M}) = \frac{\sum_{n=1}^{|\mathcal{N}|} \sum_{m=1}^{|\mathcal{M}|} x_{n,m} \cdot \lambda_{n,m}}{\sum_{n=1}^{|\mathcal{N}|} \sum_{m=1}^{|\mathcal{M}|} \lambda_{n,m}} \tag{4}$$

where $x_{n,m}$ is cache-decision indicator, *i.e.*, if f_m is placed at edge n, $x_{n,m}$=1 else, $x_{n,m}$=0.

B. Problem formulation

We aim at selecting non-overlapping subsets ($\mathcal{C}_n$) of popular contents, to cache at appropriate MH, and globally optimize the cluster hit ratio, as follows.

$$P1 : \quad \underset{\substack{\{x_{n,m}\} \\ \forall n \in \mathcal{N}, 1 \leq m \leq |\mathcal{M}|}}{\arg\max} \quad \psi(\mathcal{M}) \tag{5a}$$

$$\text{subject to: } \sum_{m=1}^{|\mathcal{M}|} x_{n,m} \cdot s_m \leq L_n, \forall n \in \mathcal{N} \tag{5b}$$

$$\sum_{n=1}^{|\mathcal{N}|} x_{n,m} \leq 1, \forall m : f_m \in \mathcal{M} \tag{5c}$$

$$\sum_{m=1}^{|\mathcal{M}|} x_{n,m} \cdot \rho_{n,m} \leq 1, \forall n \in \mathcal{N} \tag{5d}$$

$$\sum_{n=1}^{|\mathcal{N}|} x_{n,m} \cdot \eta_{n,m} = 1, \forall m : f_m \in \mathcal{M} \tag{5e}$$

$$x_{n,m} \in \{0, 1\}, \forall n \in \mathcal{N}, \forall m : f_m \in \mathcal{M} \tag{5f}$$

Here, constraint (5b) indicates that the sum of sizes of all contents in each MH should not exceed its cache size and (5c) limits that a content is cached at only one MH. The constraint in (5d) indicates that popularity of cached contents at each MH never exceeds 1 and it may have different distribution among MHs. Constraint (5e) shows the normalized demand rate and a cached content should be requested by at least one MH. Lastly, (5f) limits that contents should not be partitioned during caching process.

In (5a), the objective function $\psi(\mathcal{M})$ is globally maximized over an extremely huge number of combinatorial subsets, from which the final caching decision table $x_{n,m}$ is chosen. The most important aspect of this caching scheme is that both $\rho_{n,m}$ and $\eta_{n,m}$, collectively $\phi_{n,m}$, vary for different MHs. Using the $\phi_{n,m}$, to capture all impacts, we modelled it by *Separable Assignment Problem* (SAP).

III. Proposed Solution

Problem **P1** in (5a) is NP-hard and extremely difficult to solve optimally, at this form; meaning, to get an optimal content placement decision $x_{n,m}$. Instead, we used an iterative approach by partitioning the problem into $|\mathcal{N}|$ subproblems. Each subproblem is modelled by the 0/1-Knapsack Problem and solved using dynamic programming (DP) at pseudo-polynomial time of $O(|\mathcal{M}| \cdot L_n)$.

In the proposed selection procedure (see Algorithm-1) we first initialize variables such as cache-decision matrix ($x_{n,m}$) and subset of contents in the cluster ($\mathcal{C}$). Then, we calculate the available cache sizes and free contents, with respective popularity and sizes (Lines 6-11), that may trigger the selection process based on updated information.

In the Round step (Lines 14-16), every MH gets its candidate subset $\mathcal{C}_n$ and temporary decision matrix $\hat{x}_{n,m}$ using DP (space limit here, details in [10]). After initial selection, if any f_m exists at multiple subsets, it is assigned to subset where f_m has highest $\phi_{n,m}$ value (Lines 17-21) and the decision matrix is updated by XOR (Line 22).

All resulting subsets so far are feasible but not optimal, as there are free cache spaces. Rather, the expected value of rounded CHR is at least (1-(1-$\frac{1}{|\mathcal{N}|}$)$^{|\mathcal{N}|}$) times the optimal performance (interested readers referred to [12] for proofs). Therefore, available cache space and free contents are analysed after each step and the selection is repeated whenever there is space and unassigned content (Lines 23-29). The final caching decision of all iterations in an epoch is found when there is no space to cache any content. This strategy can also be used for cache *refilling* at after a profile update- an intermediary step before a full-scale caching. This strategy decides whether a free f_m is placed at its most 'demanding' MH. We call this policy as *zero/one separable assignment problem (ZoSAP)* caching strategy. It gives an optimal content selection at a pseudo-polynomial time complexity of *O*($|\bar{\mathcal{M}}| \cdot \sum_{n=1}^{|\mathcal{N}|} \bar{L}_n$) per iteration, which vanishes very fast due to small number of iterations.

Algorithm 1: Proposed solution to the SAP model

```
   Input: $\mathcal{M}, \mathcal{R}, \mathcal{S}, \mathcal{N}, \{L_n\}$
   Result: $x_{n,m}$
1  [Initialize]
2  $x_{n,m} = zeros(|\mathcal{N}|, |\mathcal{M}|)$;
3  $\mathcal{C} = \emptyset$;
4  Function: ZoSAP ($\mathcal{R}, \mathcal{S}, \mathcal{M}, \{L_n\}$)
5      [Availability]
6      for $n = 1$ to $|\mathcal{N}|$ do
7          $\bar{L} = \left\{\bar{L}_n : \bar{L}_n = L_n - \sum_{m=1}^{|\mathcal{C}_n|} s_m \cdot x_{n,m}\right\}$;
8          $\bar{\mathcal{M}} = \{f_m : x_{n,m} = 0, f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$;
9          $\bar{\mathcal{R}} = \{\phi_{n,m} : \phi_{n,m} \in \mathcal{R}, \forall f_m \in \bar{\mathcal{M}}\}$;
10         $\bar{\mathcal{S}} = \{s_m : s_m \in \mathcal{S}, \forall f_m \in \bar{\mathcal{M}}\}$;
11     end
12     [Round]
13     $\hat{x}_{n,m} = zeros(|\mathcal{N}|, |\mathcal{M}|)$;
14     for $n = 1$ to $|\mathcal{N}|, \forall m \in \mathcal{M}$ do
15         $\hat{x}_{n,m}$ = DP-ZOSKP($\bar{\mathcal{M}}, \bar{\mathcal{R}}(n, \forall m), \bar{\mathcal{S}}, \bar{L}_n$)
             (apply Algorithm-1 of [10]);
16     end
17     for $m = 1$ to $|\mathcal{M}|$ do
18         if ($\sum_{n=1}^{|\mathcal{N}|} \hat{x}_{n,m} > 1$) then
19             $\hat{x}_{n,m} = \begin{cases} 1, & n : \phi_{n,m} = \max_{\mathcal{N}}(\phi_{n,m}) \\ 0, & \text{else}; \end{cases}$
20         end
21     end
22     $x_{n,m} = x_{n,m} \oplus \hat{x}_{n,m}$;
23     Call Availability;
24     for $n = 1$ to $|\mathcal{N}|$ do
25         if ($\bar{L}_n \geq \min(s_m), \forall s_m \in \bar{\mathcal{S}}$) then
26             $\bar{L} = \bar{L} \cup \{\bar{L}_n\}$;
27         end
28     end
29     Go to Round;
30     return $x_{n,m}$;
31 end
```

IV. Numerical Results

In this section, we evaluate the performance of the proposed content caching strategy, formulated by the SAP, compared to two baseline caching strategies at multiple MHs. According to the state-of-the-art, the two baseline strategies assume that popularity of a content is equal towards all MHs, and they estimate a global popularity by taking the aggregated request rate. The baseline strategies are: i) *bound-and-bound zero/one-Multiple Knapsack Problem (BB-ZOMKP)* caching strategy, an exact strategy that models the content caching by MKP and iteratively solves each iteration using DP (details in [10]), ii) *Greedy-MKP* caching strategy that selects most popular contents after sorting them in a decreasing order of popularity, which is very often used [14]. It fills each MH with an iterative approach using free contents per iteration.

We evaluate the performance of the ZoSAP caching strategy in terms of CHR, which shows the percentage of content requests that are successfully fulfilled by the helpers. It is the expectation of user request to be a cache hit event; so the CHR is a more general performance measuring metric [15]. We did extensive system-level MATLAB simulations (on 20GB RAM, Intel i3-7100 CPU) on several scenarios.

We considered a 3-tier MEC-cluster which contains three types of computing and caching edges: one central MBS at first tier, 20 SBS ($\mathcal{N}_1$) and 100 femto-base stations (FBS) ($\mathcal{N}_2$) at second tier; *i.e.*, $|\mathcal{N}_1|$+$|\mathcal{N}_2|$=120 MHs. The cache size distributions, denoted by $\mathcal{L}_1$ for the SBS and $\mathcal{L}_2$

This article has been accepted for publication in IEEE Networking Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LNET.2022.3192173

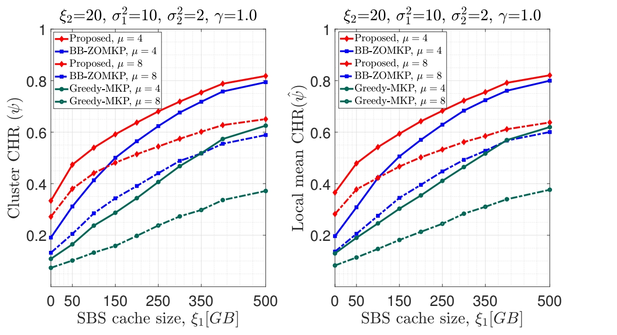

Fig. 2: For increasing cache size: a) CHR for entire cluster (left), b) Mean CHR for each helper (right).

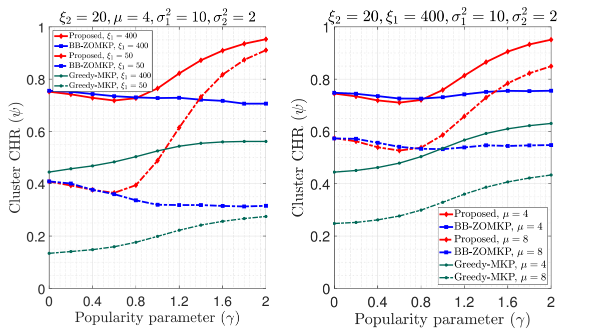

Fig. 4: Cache hit ratio versus popularity index for cluster of MHS: a) varying ξ_1 (left), b) varying μ (right)

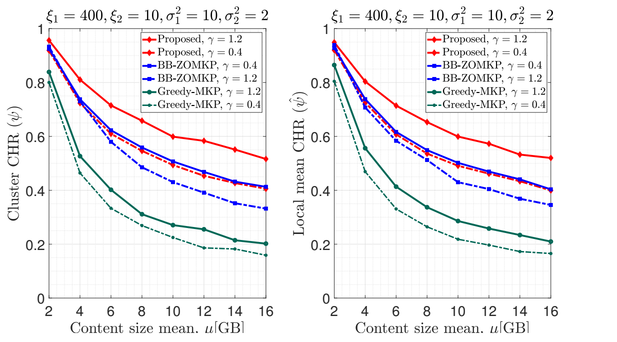

Fig. 3: For increasing content size: a) CHR for entire cluster (left), b) Mean CHR for each helper (right).

for the FBS, follow a normal distribution with mean and variance pairs of (ξ_1, σ_1^2) and (ξ_2, σ_2^2), respectively. That is, $\mathcal{L}_1 \sim \{norm(\xi_1, \sigma_1^2)\}$ and $\mathcal{L}_2 \sim \{norm(\xi_2, \sigma_2^2)\}$, while $\mathcal{L}^{1x120} = \{L_n : n = 1, ..., |\mathcal{N}|\} = \mathcal{L}_1 \cup \mathcal{L}_2$. For simplicity, we set $\xi_2 = 20GB$, $\sigma_1^2 = 10$, $\sigma_2^2 = 2$, and ξ_1 varies.

We consider $|\mathcal{M}|$=5,000 contents, in library $\mathcal{M}$, at the MBS where content sizes s_m follow an exponential distribution of mean μ; *i.e.*, $\mathcal{S}^{1x5000} = \{s_m : s_m \sim exp(\mu)\}$. Worth notes that $L \ll \sum_{m=1}^{|\mathcal{M}|} s_m$, where $L = \sum_{n=1}^{|\mathcal{N}|} L_n$. The content popularity, within each MH, is modelled by the Zipf distribution, where all demanded contents are ranked based on their number of requests received in that MH, at given time epoch. Here, the number of active UE associated to each MH is represented by exponential distribution ($|\mathcal{U}_n| \sim exp(\nu)$), with mean value ν=10.

Technically, we focus on analysing large videos which are creating high burden on the network performance. These are long lasting popular videos such as the 4K types, whose content size is adapted according to average bit rate values in Recommended Upload Encoding Settings for YouTube (Google, last retrieved March 2022) and Guidelines for Video Delivery Over a Mobile Network (NTT-DOCOMO, Jul. 2014). Based on these guidelines and survey on size specifications from few content delivering companies, we fixed μ=4GB, which corresponds to an HDR (4K) video of resolution 3840×2160 (2160p) at High Frame Rate, average length of 8 minutes, and bitrate 66Mbps. We further fix the Zipf parameter to γ=1.0, which shows that 10% of contents account for 75% of local popularity.

After extensive system-level simulation, we presented the CHR values: for the entire cluster (ψ) and *mean local CHR* ($\hat{\psi}$) for single MH, in side-by-side plots. The ψ indicates availability of contents in the system regardless of where they are placed, maybe in neighbouring MHs. The local ($\hat{\psi}$) shows the ratio of cache hit events within an MH, means the degree of demand-based content placement.

A. Impact of cache size on cache hit ratio

For this scenario, the ξ_1 value stepped from ξ_1=0GB (means no SBS) to ξ_1=500GB, while ξ_2=20GB for all FBSs. At every stage, the cluster cache size is estimated as: $L = |\mathcal{N}_1| \cdot norm(\xi_1, \sigma_1^2) + |\mathcal{N}_2| \cdot norm(20, \sigma_2^2)$ GB. We set γ=1.0 for contrasting μ= 4GB and μ= 8GB.

The two plots in Fig. 2 indicate the CHR of a cluster (left) and mean local CHR (right) per each MH. As can be seen, the $\hat{\psi}$ is as large as the ψ which indicates that almost all contents are placed at MHs through which they were highly requested. We also notice that, for all strategies, as the cache size increases, both cluster and local CHR increase. This is because, we get sufficient space to cache more content. However, as μ steps up from 4GB to 8GB, the gap between corresponding ψ and $\hat{\psi}$ values linearly widens. At equal level of CHR or content availability, doubling μ value needs about 2.5×ξ_1 space. This indicates that only increment of cache size does not bring the expected CHR but also depends on content sizes.

In all cases, the proposed ZoSAP caching strategy outperforms baselines strategies. Mainly, ZoSAP strategy is extremely useful when we have smaller cache sizes; *e.g.*, at ξ_1=50GB, it gives 150% hit ratio than the BB-ZOMKP and 260% than the Greedy-MKP strategies, both locally and globally. This is achieved because ZoSAP algorithm places every content at the MH that demands it the most while the baselines do not handle this spatial demand. The ZoSAP might not be required when the cluster cache size is very large. We separately proved that ZoSAP and BB-ZOMKP strategies equally perform for $L \geq 0.70 \cdot \sum_{m=1}^{|\mathcal{M}|} s_m$. Here, the BB-ZOMKP outperformed the Greedy-MKP strategy.

B. Impact of content size on cache hit ratio

The two plots in Fig. 3 give us more details on the relation of both ψ and $\hat{\psi}$ with μ, under two popularity skewness γ=0.4 and γ=1.2. The video content mean size (μ) of exponential distribution, increased μ=2GB to μ=16GB while the cache size of MHs is fixed with ξ_2=20GB and $\xi_1 = 20 \cdot \xi_2$. We observe that all strategies having small-sized (μ=2GB) video contents give very high hit ratio, from ψ=0.85 (Greedy-MKP strategy) to ψ=0.95 (ZoSAP strategy). When content size gets bigger to μ=16GB, this performance exponentially decays to ψ=0.15 and ψ=0.51, respectively. This is because, the cache size becomes severe constraint to cache larger contents.

In other comparison, when popularity index gets lower from γ=1.2 (highly skewed) to γ=0.4 (close to uniform), the CHR value generally drops. Specially, when proposed strategy is applied on nearly uniform popularity but large-sized contents (say μ=16GB), their availability decreases from 52% to 40% while BB-ZOMKP drops from 41% to 33% because the content size constraint becomes tougher.

In all cases of this scenario, the proposed ZoSAP strategy outperformed the baselines by far. Mainly, when we have highly skewed popularity (γ=1.2) and content size of μ=2GB, it outperforms by 102% than BB-ZOMKP strategy, and increases to 125% at μ=16GB. This is achieved due to the fact that ZoSAP strategy places contents at MHs where they are most 'demanded'. The BB-ZOMKP showed better performance than Greedy-MKP strategy since the algorithm can cache more contents and get an exact solution per multiple MHs, without being aware to spatial-demand. The Greedy-MKP strategy has the lowest performance since it can't deal on the size constraints of the problem.

C. Impact of popularity skewness on cache hit ratio

The impact of popularity index on CHR is shown in plots of Fig. 4 for the global CHR of the cluster under: a) two cache size values of ξ_1=50GB and ξ_1=400GB, at μ=4GB; b) two content sizes of μ=4GB and 8GB, at ξ_1=400GB. The popularity index γ ranges from 0 (uniform popularity) to 2.0 (only very few videos are very popular). In both cases, the FBS edges have cache mean size ξ_2=20GB. The result shows that for all strategies, larger video contents have less CHR in the cluster due to MHs cache size limitation. This is noticed either when ξ_1 is reduced from 400GB to 50GB for fixed μ (left plot) or μ is doubled from 4GB to 8GB for fixed ξ_1 (right plot).

Interestingly, the performance of the ZoSAP strategy varies over range of skewness indexes. Looking at the left plot of Fig. 4, the ZoSAP strategy performs almost equally with BB-ZOMKP strategy at ξ_1=400GB for $\gamma \leq$0.5. Even when we have sufficient cache size (e.g., $\xi_1 \geq$400GB) for $\gamma \leq$0.8, the BB-ZOMKP is preferred, however more complex and computationally costly. Based on mean cache size (μ), the ZoSAP gives an exponentially increasing CHR after some popularity index such as γ=0.5 for ξ_1=50GB or γ=0.8 for ξ_1=400GB. The CHR value for ξ_1=50GB case drastically increases close to the ξ_1=400GB case. From this plot, we understand that γ has significant relation with ξ_1, unlike for the case when cluster cache size is fixed; such as ξ_1=400GB (right plot). In general, for real cellular networks, with γ >0.8, the proposed strategy is quite useful. This high performance is achieved by the fact that ZoSAP caches contents at local MHs, through which they received the highest demand rate from active users.

Though the BB-ZOMKP strategy gives high cache hit probability [10] (only focuses on choosing very popular ones), regardless of ξ and μ values, it's cache hit ratio slightly decreases across γ values because the γ seen in each MH is counterbalanced at the MBS so the popularity is less affected. Apparently, the strategy disregards demand of individual MHs so that highly popular and impactful contents are displaced from their destined local MH. In contrast, but with lower performance, the Greedy-MKP strategy increases with skewness because it places free video contents to better-demanding MH and fulfils cache hit.

V. Conclusions

In this work, we have studied cooperative content caching in a cluster of MEC-enabled edges, which are densely deployed to offload the central MBS. We focused on how to assign contents to the MHs where they are most 'demanded'. Using combined impact of two local parameters, request probability, we have modelled the caching scheme using Separable Assignment Problem and proposed an iterative combinatorial content placement strategy, where each iteration is optimally done using dynamic programming. The obtained results from extensive simulations show that the proposed strategy profoundly outperforms for the demand-based content caching. Future work includes computational cost analysis of the proposed strategy itself and scaling up the caching scheme to a dual-nested cost optimization.

References

[1] T. X. Tran, A. Hajisami and D. Pompili, "Cooperative Hierarchical Caching in 5G Cloud Radio Access Networks," IEEE Network, vol. 31, no. 4, pp. 35-41, 2017.

[2] N. Giatsoglou, K. Ntontin, E. Kartsakli, A. Antonopoulos and C. Verikoukis, "D2D-Aware Device Caching in mmWave-Cellular Networks," IEEE JSAC, vol. 35, no. 9, pp. 2025-2037, 2017.

[3] L. N. T. Huynh, Q. -V. Pham, T. D. T. Nguyen, M. D. Hossain, Y. -R. Shin and E. -N. Huh, "Joint Computational Offloading and Data-Content Caching in NOMA-MEC Networks," IEEE Access, vol. 9, pp. 12943-12954, 2021.

[4] Y. M.Saputra, H.T.Dinh, D.Nguyen, E.Dutkiewicz, "A Novel Mobile Edge Network Architecture with Joint Caching-Delivering and Horizontal Cooperation," IEEE Trans. Mob. Comp., pp.1–1, 2019.

[5] R. Liu et al., "Cooperative caching scheme for content oriented networking," IEEE Comm. Let., vol. 17, no. 4, pp. 781–784, 2013.

[6] K. Poularakis, G. Iosifidis, A. Argyriou, I. Koutsopoulos and L. Tassiulas, "Caching and operator cooperation policies for layered video content delivery," IEEE INFOCOM, CA, pp. 1-9, 2016.

[7] A. Sengupta, S. Amuru, R. Tandom, R. M. Buehrer, and T. C. Clancy, "Learning Distributed Caching Strategies in Small Cell Cell Networks," IEEE ISWCS, pp. 917–921, 2014.

[8] J. Wen, K. Huang, S. Yang, V. O. K. Li, "Cache-Enabled Heterogeneous Cellular Networks: Optimal Tier-Level Content Placement," IEEE Tran. on Wirel. Com, vol. 16, no. 9, pp. 5939-5952, 2017.

[9] S. Zhang, P. He, K. Suto, P. Yang, L. Zhao, and X. Shen, "Cooperative Edge Caching in User-Centric Clustered Mobile Networks," IEEE Tran. Mob. Comput., vol. 17, no. 8, pp. 1791-1805, 2018.

[10] T. M. Ayenew, D. Xenakis, N. Passas and L. Merakos, "Cooperative Content Caching in MEC-Enabled Heterogeneous Cellular Networks," in IEEE Access, vol. 9, pp. 98883-98903, 2021.

[11] D. Xenakis, M. Kountouris, L. Merakos, N. Passas, C. Verikoukis, "Performance Analysis of Network-Assisted D2D Discovery in Random Spatial Networks," IEEE Tran. on Wirel. Comm., vol. 15, no. 8, pp. 5695-5707, 2016.

[12] L. Fleischer, M. X. Goemans, V. S. Mirrokni, M. Sviridenko, "Tight Approximation Algorithms for Maximum Separable Assignment Problems." Math. Oper. Res., vol. 36, no. 3, pp. 416–431, 2011.

[13] A. -T. Tran et al., "Hit Ratio and Latency Optimization for Caching Systems: A Survey," 2021 International Conference on Information Networking (ICOIN), pp. 577-581, 2021.

[14] D. T. Hoang, D. Niyato, D. N. Nguyen, E. Dutkiewicz, P. Wang, and Z. Han, "A Dynamic Edge Caching Framework for Mobile 5G Networks," IEEE Wirel. Commun., vol. PP, pp. 1–9, 2018.

[15] N. K. Panigrahy, J. Li, and D. Towsley, "Hit rate vs. Hit probability based cache utility maximization," SIGMETRICS Perform. Eval. Rev., ACM, vol. 45, no. 2, pp. 21–23, 2017