Coded Caching and Storage Planning in Heterogeneous Networks

Thang X. Vu, Symeon Chatzinotas, and Bjorn Ottersten

Interdisciplinary Centre for Security, Reliability and Trust (SnT)
University of Luxembourg, 4 rue Alphonse Weicker, L-2721 Luxembourg
Email: {thang.vu, symeon.chatzinotas, bjorn.ottersten}@uni.lu

***Abstract*—Content caching is an efficient technique to reduce delivery latency and system congestion during peak-traffic times by bringing data closer to end users. Existing works on caching usually assume symmetric networks with identical user requests distribution, which might be in contrast to practical scenarios where the number of users is usually arbitrary. In this paper, we investigate a cache-assisted heterogeneous network in which edge nodes or base stations (BSs) are capable of storing content data in their local cache. We consider general practical scenarios where each edge node is serving an arbitrary number of users. First, we derive an optimal storage allocation over the BSs to minimize the shared backhaul throughput for a uncoded caching policy. Second, a novel coded caching strategy is proposed to further reduce the shared backhaul's load. Finally, the effectiveness of our proposed caching strategy is demonstrated via numerical results.**

***Index terms*—** Content caching, rate-memory trade-off, heterogeneous networks.

I. Introduction

With the proliferation of mobile devices and rich-content applications, future wireless networks will have to address stringent requirements of delivering content at high speed and low latency. Various network architectures have been proposed in order to boost the network throughput and reduce transmission latency such as cloud radio access networks and heterogeneous networks (HetNets) [1]. Despite potential high rate in the new architectures, traffic congestion might occur during peak-traffic times. One approach to reduce peak traffic is to bring content closer to end users via distributed storage through out the network, which is referred to content placement or caching [2]. Caching usually consists of two phases: placement and delivery. The former is executed during off-peak periods when the network resources are redundant. In this phase, popular content is duplicated and stored in the distributed caches in the network. The delivery phase usually occurs during peak-traffic hours when the actual users' requests are revealed. If the requested content is available in the user's local storage, it can be served locally without being sent via the network. In this manner, caching allows significant throughput reduction during peak-traffic times and thus reducing network congestion [3].

Most research works investigate the caching problem via exploiting historic user requested data to optimize either placement or delivery phases [2], [4], [5]. For a fixed content delivery strategy, the placement phase is designed in order to maximize the local caching gain. This gain is proportional to parts of the files available in the local storage, which is related to the cache memory. Joint content caching and transmission design has recently been studied in heterogeneous [6], [7] and device-to-device networks [8], [9]. In [9], the authors study two caching policies which allow the storage of files at either small base stations or user terminals. Taking into account the wireless fading channel, a joint content replacement and delivering scheme is proposed to reduce the system energy consumption. The authors in [7] jointly optimize caching, routing and channel assignment via two sub-problems called restricted master and pricing. Cache-assisted multicast beam-forming design and power allocation are investigated in [6] to reduce transmitted power and fronthaul bandwidth. These caching methods store files independently and are known as *uncoded* caching. The caching gain can be further improved via multicasting a fixed combination of files during the delivery phase, which is known as *coded* caching [3]. By carefully placing the files in the caches and designing the coded data, all users can decode their desired content via a multicast stream. Rate-memory tradeoff is derived in [3] to achieve a global caching gain on top of the local caching gain. This gain is inversely proportional to the total cache memory. A similar rate-memory tradeoff is investigated in device-to-device (D2D) networks [10] and secrecy constraint [11]. In [12], the authors study the tradeoff between the cache memory at edge nodes and the transmission latency measured in normalized delivery time. The rate-memory tradeoff of multi-layer coded caching networks is studied in [13], [14].

The above-mentioned papers consider symmetric networks with identical user demands and cache capabilities, which might be in contrast to realistic scenarios, e.g., HetNets, in which the number of user requests can be arbitrary. It is worthy to highlight that the base stations (BSs) in HetNets (marco-, micro-, pico-) have various capabilities to serve the users with non-uniform distribution. For example, a BS serving an office building tends to have heavier traffic load than the one in a residential area. From the system perspective, it is demanding to plan more network resources (backhaul capacity, storage memory) to an area with higher statistic user demands. In this paper, we investigate cache-assisted HetNets in practical scenarios, in which each BS can serve an arbitrary number of

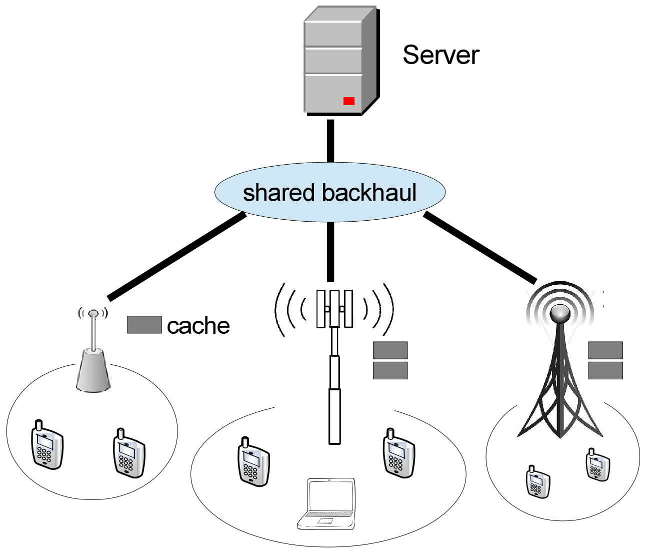

Fig. 1: Cache-assisted wireless networks with heterogeneous user requests and cache capabilities.

user requests. Our goal is to optimally allocate the storage memory across the BSs and design a caching strategy to minimize the shared backhaul's traffic. Our contributions are as follows:

- Firstly, we derive the required backhaul rate that each BS needs to serve its users for a given storage memory.
- Secondly, based on the derived rate, the storage allocation is optimized to minimize the aggregated throughput on the shared backhaul link under an uncoded caching approach. The optimal solution shows that it is more beneficial to allocate larger cache memory for BSs which are serving more users. Our method is fundamentally different from [15] because of following reasons. First, we investigate rate-memory tradeoff performance, while [15] considers failure probability. Second, we consider general wireless network architectures. On the other hand, [15] focuses on multiple-input single-output system.
- Thirdly, we propose a novel coded caching strategy and the corresponding storage allocation to further reduce the shared backhaul's traffic load. Numerical results demonstrate a significant gain of the proposed caching strategy.

The rest of this paper is organised as follows. Section II presents the proposed storage allocation with uncoded caching policy. Section III presents the proposed coded caching. Section IV shows numerical results. Finally, Section V concludes the paper.

II. Storage Allocation with Uncoded Caching

We consider a cache-assisted HetNet in which a data centre serves users via distributed BSs, as depicted in Figure 1. The BSs connect to the data centre via a error-free, bandwidth-limited shared link. Each BS serves a number of users via wireless channels simultaneously. It is assumed that adjacent cells operate in orthogonal frequencies, thus inter-cell interference is negligible. At every time instance, each user requests a file from a library with the size of F files. Each file has equal length of B bits. Denote by K the number of BSs and by $n_k \in \mathbb{N}^+$ the number of users served by BS k. Denote $\mathcal{N}_K = \{n_1, ..., n_K\}$ as the users profile. The total number of users is equal to $N = \sum_{k=1}^K n_k$. Without loss of generality, we assume that $n_i \le n_j$ for $i < j$. In order to leverage the shared link's load, each BS is equipped with a storage memory. Let M_k be the storage size of BS k. For convenience, the storage memory is normalized by the file length B. The total storage memory in all BSs is $M_\Sigma = \sum_{k=1}^K M_k$.

Our objective is to minimize the aggregated request rate on the shared backhaul link by optimally allocating the storage memory to the BSs. Let r_k denote the request rate on the shared backhaul from BS k. The optimization problem is formally stated as follows:

$$\underset{\{M_1,...,M_K\}}{\texttt{minimize}} \quad \sum_{k=1}^K r_k \tag{1}$$

$$\texttt{s.t.} \quad \sum_{k=1}^K M_k = M_\Sigma, \tag{1a}$$

$$0 \le M_k \le \min\{M_\Sigma, F\}, \ \forall k, \tag{1b}$$

where condition (1a) satisfies the total memory constraint and (1b) assures efficient use of the cache.

Proposition 1: Consider the k-th BS equipped with a cache of size M_k serving n_k users. The BS uses the uncoded caching strategy to cache the files. The backhaul rate that BS k requires to serve its users, as F is large, is calculated as follows:

$$r_k = n_k(1 - \frac{M_k}{F}). \tag{2}$$

Proof: See Appendix A. ■

By using Proposition 1, the aggregated backhaul load from all BSs is equal to

$$\sum_{k=1}^K r_k = N - \frac{1}{F}\sum_{k=1}^K n_k M_k. \tag{3}$$

Then the problem (1) is equivalent to the following problem:

$$\underset{\{M_1,...,M_K\}}{\texttt{minimize}} \quad N - \sum_{k=1}^K n_k M_k \tag{4}$$
$$\texttt{s.t.} \quad (1a) \text{ and } (1b).$$

Minimizing the objective in (4) is equivalent to maximizing $\sum_{k=1}^K n_k M_k$. Since $\sum_{k=1}^K n_k M_k$ is an increasing function with respect to $\{M_k\}_{k=1}^K$, it is straightforward to show that it achieves the maximum value at the corner point. By expressing $M_\Sigma = tF + M'$ with $M' < F$ and $t \in \mathbb{N}$, problem (1) achieves the maximum value at

$$M_k = \begin{cases} 0, & \text{if } k \le K - t - 1 \\ M', & \text{if } k = K - t \\ F, & \text{if } K - t + 1 \le k \le K \end{cases}. \tag{5}$$

Substituting (5) into (3) we obtain the minimum backhaul's load given as

$$R_{rnd}^{\texttt{opt}} = \sum_{k=1}^{K-t} n_k - n_{K-t} \frac{M_\Sigma - tF}{F}. \tag{6}$$

The optimal storage allocation (5) suggests to fill as large storage memory as the library size for the BSs with more user requests. This provides important guidelines for network resources planning and optimizing in HetNets.

Without storage allocation, every BS is allocated with a memory of equal size $\frac{M_\Sigma}{K}$. By using Proposition 1, the backhaul load in this case is calculated as follows:

$$R_{rnd} = \sum_{k=1}^{K} n_k \left(1 - \frac{M_\Sigma}{KF}\right). \tag{7}$$

The rate different between (7) and (1) is computed as

$$\begin{aligned} R_{rnd} - R_{rnd}^{\texttt{opt}} &= \sum_{k=1}^{K} n_k \left(1 - \frac{M_\Sigma}{KF}\right) \\ &\quad + n_{K-t} \frac{M_\Sigma - tF}{F} - \sum_{k=1}^{K-t} n_k \\ &= \left(1 - \frac{M_\Sigma}{KF}\right) \sum_{k=K-t+1}^{K} (n_k - n_{K-t}). \end{aligned} \tag{8}$$

It is observed that the right hand side of (8) is always ≥ 0 because $n_{k:k \geq K-t+1} \geq n_{K-t}$. This confirms the effectiveness of the proposed storage allocation. The equality in (8) holds when $n_1 = n_2 = \cdots = n_K$ or $K = 1$.

III. Proposed Coded-caching strategy

In this section, we propose a coded-caching strategy to further reduce the aggregated traffic on the shared backhaul. It is assumed that $M_\Sigma = tF$ bits[1] with $t \in \{0, 1, 2, \ldots, K\}$. This assumption is in line with practical scenarios because the storage memory usually takes a discrete value as a power of 2. Let $\mathcal{M}_K = \{M_k\}_{k=1}^K$ denote a storage allocation scheme with

$$M_k = \begin{cases} \frac{n_k F}{\sum_{k=1}^{K-t+1} n_k}, & \text{if } 1 \leq k \leq K - t + 1 \\ F, & \text{if } K - t + 2 \leq k \leq K \end{cases}. \tag{9}$$

It is straightforward to verify that $\mathcal{M}_K$ in (9) satisfies the storage constraint at every BS and the total memory constraint.

Theorem 1: Consider a cache-assisted HetNet with K BSs and the user profile $\mathcal{N}_K = \{n_1, ..., n_K\}$, e.g., BS k is serving n_k users. The aggregated backhaul load under $\mathcal{M}_K$ is given as

$$R_{coded}(\mathcal{M}_K) = \frac{\sum_{k=1}^{K-t} n_k \sum_{j=k+1}^{K-t+1} n_j}{\sum_{k=1}^{K-t+1} n_k}. \tag{10}$$

Proof: We first consider placement and delivery phases as follows:

[1]This is the total storage memory in the network. The local cache at each BS is usually less than F.

A) Placement phase: Following $\mathcal{M}_K$ in (9), all BSs $k \geq K - t + 2$ has a storage size of F and therefore can store all the files in their cache. The placement phase for BS k with $1 \leq k \leq K - t + 1$ is executed as following. Each file is divided into $N_t \triangleq \sum_{k=1}^{K-t+1} n_k$ subfiles of equal size as: $W_f = \{W_{f,ki}\}$ with $1 \leq k \leq K - t + 1, 1 \leq i \leq n_k$. Then each BS k stores $n_k F$ subfiles in its cache Z_k as following:

$$Z_k = \{W_{1,ki}, W_{2,ki}, \ldots, W_{F,ki} \mid \forall i \in \{1, \ldots, n_k\}\}.$$

The total volume of the subfiles cached in BS k is equal to $F n_k \times \frac{1}{N} = M_k$ which satisfies the memory constraint.

B) Delivery phase: Each user requests a file by sending a request to the data centre. We note that all user requests in BS $k \geq K - t + 2$ can be served directly from their cache without any cost for the shared backhaul. Therefore, it is sufficient to consider the BS k with $1 \leq k \leq K - t + 1$.

Let $d_{kj} \in \{1, 2, \ldots, F\}$ denote a request index from user j to BS k. Define vectors $\mathbf{V}_{kj} \triangleq \{W_{d_{kj},pq} \mid k < p \leq K - t + 1, 1 \leq q \leq n_p\}$ and $\mathbf{U}_{kj} \triangleq \{W_{d_{pq},kj} \mid k < p \leq K - t + 1, 1 \leq q \leq n_p\}$, both consist of $\sum_{i=k+1}^{K-t+1} n_i$ subfiles. Note that the cache Z_k already has n_k subfiles $\{W_{d_{kj},ki}\}_{i=1}^{n_k}$ of the requested file $W_{d_{kj}}$. Therefore, BS k needs to request $n_k(N_t - n_k)$ other subfiles from the data centre.

In order to serve the request d_{kj}, the data centre broadcasts $\mathbf{U}_{kj} \oplus \mathbf{V}_{kj}$ to the BSs, where $\oplus$ indicates element-wise XOR operation. This process is repeated until $k = K - t$. Since there are n_k user requests in BS k, the data centre needs to send $\sum_{k=1}^{K-t} n_k \sum_{j=k+1}^{K-t+1} n_j$ combinations of subfiles over the shared backhaul. Taking into account the fact that each subfile is of length $\frac{1}{N_t}$, the aggregated backhaul throughput is given as

$$R_{coded} = \frac{\sum_{k=1}^{K-t} n_k \sum_{j=k+1}^{K-t+1} n_j}{\sum_{k=1}^{K-t+1} n_k}.$$

Now we will show that all users can receive the requested file from the above delivery strategy. After the first $\sum_{k=2}^{K} n_k$ transmissions, all BSs received $\mathbf{Y}_{11} = \mathbf{U}_{11} \oplus \mathbf{V}_{11}$. Note that $\mathbf{U}_{11} = \{W_{d_{kj},11}\}, \forall 2 \leq k \leq K, 1 \leq j \leq n_k$, is already in the cache Z_1, the BS 1 can recover file $W_{d_{11}}$ by performing $\mathbf{U}_{11} \oplus \mathbf{Y}_{11}$. Meanwhile, consider user j at BS k with $k \geq 2$ and $1 \leq j \leq n_k$. Because the cache Z_k has $W_{d_{11},kj}$ and $\mathbf{Y}_{11}$ contains $W_{d_{kj},11} \oplus W_{d_{11},kj}$, the BS k can recover $W_{d_{kj},11}$ by performing $W_{d_{11},kj} \oplus W_{d_{kj},11} \oplus W_{d_{11},kj}$. Consequently, after the first $\sum_{k=2}^{K} n_k$ transmissions, BS 1 recovers file $W_{d_{11}}$ and all BSs $k \geq 2$ recover the first subfile for all their desired files. This process is repeated until the transmission of $\mathbf{U}_{(K-1)n_{K-1}} \oplus \mathbf{V}_{(K-1)n_{K-1}}$ after which all BSs recover their requested files. The detailed placement and delivery phases are given in Table I for BSs k with $1 \leq k \leq K - t + 1$. Note that BSs $k > K - t + 1$ has a storage memory of F, thus can serve its users locally without causing any backhaul traffic. ■

The example below will provide details on how the proposed caching strategy works.

Example 1: Consider $F = 4, K = 3$, user profile $\mathcal{N}_3 = \{1, 1, 2\}$, and storage allocation $\mathcal{M}_3 = \{1, 1, 2\}$. For simplicity, let us denote the files as $A, B,$ $C,$ and

TABLE I: CODED-CACHING ASSISTED

```
Inputs: $K, F, M_\Sigma = tF$
Placement phase
  for $f = 1$ to $F$
    Divide $W_f = \{W_{f,kj}\},\ \forall\ 1 \le k \le K, 1 \le j \le n_k$
  end
  for $k = 1$ to $K - t + 1$
    $Z_k = \{W_{f,kj}\},\ \forall\ 1 \le f \le F, 1 \le j \le n_k$
  end
Delivery phase
  for $k = 1$ to $K - t$
    for $j = 1$ to $n_k$
      $\mathbf{V}_{kj} = \{W_{d_{kj},pq}\},\ \forall\ 2 \le p \le K, 1 \le q \le n_p$,
      $\mathbf{U}_{kj} = \{W_{d_{pq},kj}\},\ \forall\ 2 \le p \le K, 1 \le q \le n_p$,
      broadcast $\mathbf{U}_{kj} \oplus \mathbf{V}_{kj}$ to the BSs,
    end
  end
```

D. Each file is divided into four subfiles as follows: $A = \{A_{11}, A_{21}, A_{31}, A_{32}\}, B = \{B_{11}, B_{21}, B_{31}, B_{32}\}, C = \{C_{11}, C_{21}, C_{31}, C_{32}\},\ D = \{D_{11}, D_{21}, D_{31}, D_{32}\}$. The subfiles are stored at three caches as:

$$Z_1 = \{A_{11}, B_{11}, C_{11}, D_{11}\}$$
$$Z_2 = \{A_{21}, B_{21}, C_{21}, D_{21}\}$$
$$Z_3 = \{A_{31}, B_{31}, C_{31}, D_{31}, A_{32}, B_{32}, C_{32}, D_{32}\}$$

In the delivery phase, the users send a request $\{A, B, C, D\}$ to the data centre. First, the data centre constructs vectors $\mathbf{V}_{11} = \{A_{21}, A_{31}, A_{32}\}, \mathbf{U}_{11} = \{B_{11}, C_{11}, D_{11}\}$. Then, it broadcasts $\mathbf{U}_{11} \oplus \mathbf{V}_{11} = \{A_{21} \oplus B_{11}, A_{31} \oplus C_{11}, A_{32} \oplus D_{11}\}$ to the BSs. Since the cache Z_1 contains B_{11}, C_{11} and D_{11}, BS 1 can subtract A_{21}, A_{31}, A_{32} from $\mathbf{U}_{11} \oplus \mathbf{V}_{11}$. The BS 2 can recover B_{11} because A_{21} is already in the cache Z_2. Similarly, BS 3 can subtract C_{11} and D_{11} since A_{31} and A_{32} are in its cache. In the next step, the data centre constructs $\mathbf{V}_{21} = \{B_{31}, B_{32}\}, \mathbf{U}_{21} = \{C_{21}, D_{21}\}$ and then sends $\mathbf{U}_{21} \oplus \mathbf{V}_{21} = \{B_{31} \oplus C_{21}, B_{32} \oplus D_{21}\}$ to the BSs. Upon received $B_{31} \oplus C_{21}, B_{32} \oplus D_{21}$, BS 2 subtracts for B_{31}, B_{32} since C_{21}, D_{21} are in the cache Z_2. At the same time, BS 3 can recover C_{21} and D_{21} because Z_3 contains B_{31} and B_{32}.

Corollary 1: Under the user profile $\mathcal{N}_K = \{n_k = k\}_{k=1}^K$, $M_\Sigma = F$, the aggregated shared backhaul's load is given as

$$R_{coded}(\mathcal{M}_K) = \frac{(K-1)(3K+2)}{12}. \quad (11)$$

Proof: See Appendix B ■

Corollary 2: The backhaul's load of the proposed caching strategy for the i.i.d user request $n_k = n, \forall k$ is given as

$$R_{coded}(\mathcal{M}_K) = \frac{n(K-t)}{K-t+2}. \quad (12)$$

The result of Corollary 2 is obtained directly from Theorem 1 by substituting $n_k = n, \forall k$. In the special case $n = 1$ and $M_\Sigma = F$ bits, the proposed scheme achieves the backhaul's load $\frac{K-1}{K+1}$, which is always less than or equal to backhaul's rate $\frac{K-1}{2}$ in [3] since $K \ge 1$.

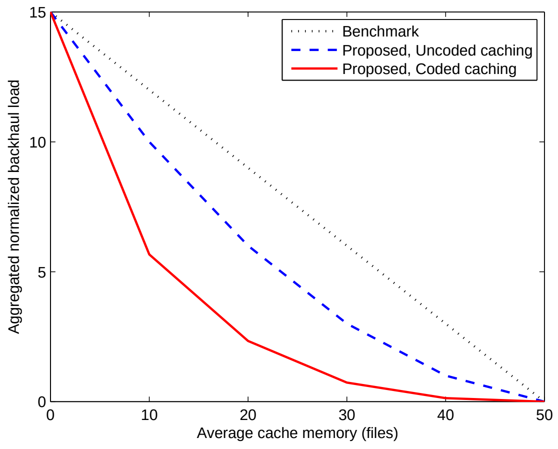

Fig. 2: Shared backhaul traffic as a function of average cache size M_Σ/K. The networks setting: $K = 5$ BSs, $F = 50$ files, and $\mathcal{N}_K = \{1, 2, 3, 4, 5\}$.

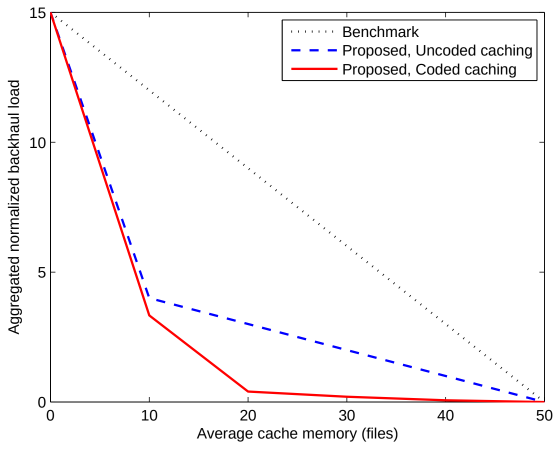

Fig. 3: Shared backhaul traffic as a function of average cache size M_Σ/K. The networks setting: $K = 5$ BSs, $F = 50$ files, and $\mathcal{N}_K = \{1, 1, 1, 1, 11\}$.

IV. Numerical Results and Discussions

In this section, we demonstrate the effectiveness of the proposed caching strategy via numerical results. For reference, we also present the benchmark, which employs uncoded caching policy and allocates equal storage memory $\frac{M_\Sigma}{K}$ to all BSs. The backhaul load of the benchmark scheme is given in (7). For ease of demonstration, we use *benchmark* and *proposed* to indicate the benchmark and proposed caching strategies in the figure, respectively. Note that the coded caching in [3] is not applicable in context of HetNets because it is designed for identical user requests.

Figure 2 shows the aggregated backhaul's load of the proposed caching schemes and the reference scheme for various

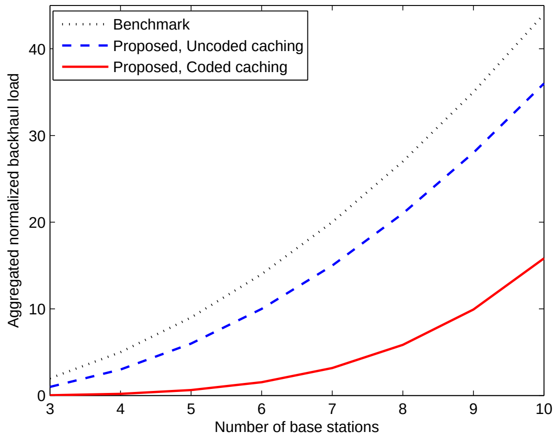

Fig. 4: Backhaul throughput as a function of number of base stations K. The library size $L = 100$ and the average cache size is equal to $\frac{2F}{K}$. The users profile $\mathcal{N}_K = \{n_k = k\}_{k=1}^{K}$.

cache sizes. The data centre with library of $F = 30$ files distribute content to $N = 15$ users via $K = 5$ BSs. It is shown that the proposed caching schemes significantly reduce the backhaul traffic. In particular, at an average cache size of 20 files, the proposed coded caching scheme reduces 75% the backhaul load compared with the benchmark scheme, while the proposed scheme with the uncoded caching degrades the backhaul load by 33%. When the average cache size is equal to 0 or F, the three schemes achieve the same backhaul load because the caches are either empty or full.

Figure 3 present results for a highly asymmetric user profile $\mathcal{N}_K = \{1, 1, 1, 1, 11\}$. In this case, the proposed schemes save 95% and 67% of backhaul load with coded and uncoded caching strategies, respectively. Figure 4 presents the shared backhaul load for different number of BSs. It is shown that the proposed coded caching schemes achieve more gain as the number of BSs increases.

V. Conclusions

We investigated cache-assisted heterogeneous networks with arbitrary number of user requests. We derived an optimal storage allocation under the uncoded caching policy to minimize the aggregated backhaul load. Furthermore, we proposed a novel coded caching scheme and the corresponding storage allocation to further reduce the shared backhaul traffic. Numerical results demonstrated the effectiveness of our proposed caching strategies.

VI. Acknowledgement

This work is financially supported in part by the H2020 SANSA project under the grant code I2R-SIG-PEU-15SNSA.

Appendix A
Proof of Proposition 1

By using the uncoded caching method, BS k stores M_k/F of every file in its cache. When a user requests a file, the BS can send directly parts of the file which is already. The subfiles which are not in the cache will be requested from the data centre via the shared backhaul link. Let $(d_1, d_2, ..., d_{n_k}), d_j \in \{1, ..., F\}$, be the requested file indexes. Since the users' requests are independent, the requested files can be either the same or different.

For any integer number $m, 1 \leq m \leq F$, there are F^m ways to choose m elements out of the set of size F, which can be further expressed as

$$F^m = \sum_{l=1}^{m} a_l^m \mathcal{C}_l^F,$$

where $\mathcal{C}_l^F \triangleq \frac{F!}{(F-l)!}$ and a_l^m is a constant. In the above equation, $a_l^m \mathcal{C}_l^F$ is the number of choices of m elements out of F which contains l different elements. By using the inductive method, we can obtain:

$$a_l^m = \begin{cases} 1, & \text{if } l = 1 \text{ or } m \\ m a_l^{m-1} + a_{l-1}^{m-1}, & \text{if } 1 < l < m \end{cases}$$

For a choice comprising of l different values, the BS needs to request $l(1 - M_k/F)$ subfiles from the data centre. Therefore, the average backhaul rate requested by BS k is calculated as

$$\begin{aligned} r_k &= \frac{1}{F^{n_k}} \sum_{l=1}^{n_k} l a_l^{n_k} \mathcal{C}_l^F \left(1 - \frac{M_k}{F}\right) \\ &= \sum_{l=1}^{n_k} \frac{l a_l^{n_k}}{F^{n_k - l}} \left(1 - \frac{M_k}{F}\right) \prod_{i=1}^{l} \frac{F - l + i}{F}. \end{aligned} \quad (13)$$

It is observed that the library size F is usually very large and n_k is small compared with F, thus $\frac{F-l+i}{F} \simeq 1, \forall 1 \leq i \leq l$ and

$$\frac{l a_l^{n_k}}{F^{n_k - l}} \simeq \begin{cases} 0, & \text{if } l < n_k \\ n_k, & \text{if } l = n_k \end{cases}. \quad (14)$$

From (13) and (14) we obtain:

$$r_k \simeq n_k \left(1 - \frac{M_k}{F}\right). \quad (15)$$

Appendix B
Proof of Corollary 1

From Theorem 1 we obtain the aggregated backhaul load in this case computed as

$$R = \frac{\sum_{k=1}^{K-1} k \sum_{j=k+1}^{K} j}{\sum_{k=1}^{K} k}. \quad (16)$$

The numerator of 16 is further expressed as follows:

$$\begin{aligned} \sum_{k=1}^{K-1} k \sum_{j=k+1}^{K} j &= \frac{1}{2}\left[\left(\sum_{k=1}^{K} k\right)^2 - \left(\sum_{k=1}^{K} k^2\right)\right] \\ &= \frac{1}{2}\left[\left(\frac{K(K+1)}{2}\right)^2 - \frac{K(K+1)(2K+1)}{6}\right] \\ &= \frac{1}{24} K(K^2 - 1)(3K + 2). \end{aligned} \quad (17)$$

Note that $\sum_{k=1}^{K} k = K(K+1)/2$. Substituting (17) into (16), we obtain (11).

References

[1] D. Lopez-Perez, I. Guvenc, G. de la Roche, M. Kountouris, T. Q. S. Quek, and J. Zhang, "Enhanced intercell interference coordination challenges in heterogeneous networks," *IEEE Wireless Commun.*, vol. 18, no. 3, pp. 22–30, Jun. 2011.

[2] S. Borst, V. Gupta, and A. Walid, "Distributed caching algorithms for content distribution networks," in *Proc. IEEE INFOCOM*, Mar. 2010, pp. 1–9.

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

[4] K. C. Almeroth and M. H. Ammar, "The use of multicast delivery to provide a scalable and interactive video-on-demand service," *IEEE J. Sel. Areas Commun.*, vol. 14, no. 6, pp. 1110–1122, Aug. 1996.

[5] D. Christopoulos, S. Chatzinotas, and B. Ottersten, "Cellular-broadcast service convergence through caching for comp cloud RAN," in *Proc. IEEE Symp. Commun. Veh. Tech. in the Benelux*, Luxembourg City, Nov. 2015, pp. 1-6.

[6] M. Tao, E. Chen, H. Zhou, and W. Yu, "Content-centric sparse multicast beamforming for cache-enabled cloud RAN," *IEEE Trans. Wireless Commun.*, vol. PP, no. 99, pp. 1–1, 2016.

[7] A. Khreishah, J. Chakareski, and A. Gharaibeh, "Joint caching, routing, and channel assignment for collaborative small-cell cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 8, pp. 2275–2284, Aug. 2016.

[8] L. Zhang, M. Xiao, G. Wu, and S. Li, "Efficient scheduling and power allocation for D2D-assisted wireless caching networks," *IEEE Trans. Commun.*, vol. 64, no. 6, pp. 2438–2452, Jun. 2016.

[9] M. Gregori, J. Gmez-Vilardeb, J. Matamoros, and D. Gndz, "Wireless content caching for small cell and D2D networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 5, pp. 1222–1234, May 2016.

[10] A. Sengupta, R. Tandon, and T. C. Clancy, "Fundamental limits of caching with secure delivery," *IEEE Trans. Inf. Forensics Security*, vol. 10, no. 2, pp. 355–370, Feb. 2015.

[11] M. Ji, G. Caire, and A. F. Molisch, "Fundamental limits of caching in wireless D2D networks," *IEEE Trans. Inf. Theory*, vol. 62, no. 2, pp. 849–869, Feb. 2016.

[12] A. Sengupta, R. Tandon, and O. Simeone, "Cache aided wireless networks: Tradeoffs between storage and latency," in *Proc. Conf. Inform. Sci. and Systems*, Mar. 2016, pp. 320–325.

[13] N. Karamchandani, U. Niesen, M. A. Maddah-Ali, and S. N. Diggavi, "Hierarchical coded caching," *IEEE Trans. Inf. Theory*, vol. 62, no. 6, pp. 3212–3229, Jun. 2016.

[14] L. Tang and A. Ramamoorthy, "Coded caching for networks with the resolvability property," in *Proc. IEEE Int. Symp. on Inf. Theory*, Jul. 2016, pp. 420–424.

[15] B. Hong and W. Choi, "Optimal storage allocation for wireless cloud caching systems with a limited sum storage capacity," *IEEE Trans. Wireless Commun.*, vol. PP, no. 99, pp. 1–1, 2016.