Private Information Retrieval From a Cellular Network With Caching at the Edge

Citation for the original published paper (version of record):
Kumar, S., Graell i Amat, A., Rosnes, E. et al (2019). Private Information Retrieval From a Cellular Network With Caching at the Edge. IEEE Transactions on Communications, 67(7): 4900-4912. http://dx.doi.org/10.1109/TCOMM.2019.2906229

(article starts on next page)

Private Information Retrieval From a Cellular Network With Caching at the Edge

Siddhartha Kumar, *Student Member, IEEE*, Alexandre Graell i Amat, *Senior Member, IEEE*, Eirik Rosnes, *Senior Member, IEEE*, and Linda Senigagliesi, *Student Member, IEEE*

***Abstract*—We consider the problem of downloading content from a cellular network that is cached at the wireless edge while achieving privacy. In particular, we consider private information retrieval (PIR) of content from a library of files, i.e., the user wishes to download a file and does not want the network to learn any information about which file she is interested in. To reduce the backhaul usage, content is cached at the wireless edge in a number of small-cell base stations (SBSs) using maximum distance separable codes. We propose a PIR scheme based on generalized Reed-Solomon codes for this scenario that achieves privacy against a number of spy SBSs that collaborate. The proposed PIR scheme is an extension of a scheme by Kumar *et al.* to the case of multiple code rates, suitable for the scenario where files have different popularities. We derive the backhaul rate and optimize the content placement to minimize it. We prove that uniform content placement is optimal, i.e., all files that are cached should be stored using the same code rate. This is in contrast to the case where no PIR is required. Furthermore, we show numerically that popular content placement is optimal for some scenarios.**

***Index Terms*—Colluding servers, distributed caching, generalized Reed-Solomon codes, private information retrieval, subcodes.**

I. Introduction

Bringing content closer to the end user in wireless networks, the so-called caching at the wireless edge, has emerged as a promising technique to reduce the backhaul usage. The literature on wireless caching is vast. Information-theoretic aspects of caching were studied in [1], [2]. To leverage the potential gains of caching, several papers proposed to cache files in densely deployed small-cell base stations (SBSs) with large storage capacity, see, e.g., [3]–[7]. In [5], content is cached in SBSs using maximum distance separable (MDS) codes to reduce the download delay. This scenario was further studied in [7], where the authors optimized the MDS-coded caching to minimize the backhaul rate. Caching content directly in the mobile devices and exploiting device-to-device communication has been considered in, e.g., [8]–[12]. In the literature, the concept of coded caching refers to both the caching of uncoded content to facilitate index-coded broadcasts, e.g., [1], [2], and the use of erasure correcting codes to cache content, e.g., [5], [7]. In both cases, the goal is to deliver content efficiently.

Recently, private information retrieval (PIR) has attracted a significant interest in the research community [13]–[23]. In PIR, a user would like to retrieve data from a distributed storage system (DSS) in the presence of spy nodes, without revealing any information about which piece of data she is interested in to the spy nodes. PIR was first studied by Chor *et al.* [24] for the case where a binary database is replicated among n servers (nodes) and the aim is to privately retrieve a single bit from the database in the presence of a single spy node (referred to as the noncolluding case), while minimizing the total communication cost, i.e., the sum of the upload and download cost. In the last few years, spurred by the rise of DSSs, research on PIR has been focusing on the more general case where data is stored using a storage code. Under the assumption that the data stored is big (e.g., big files) compared to the size of all queries sent to the storage nodes, the upload cost is negligible compared to the download cost, and the latter dominates the total communication cost. Hence, recent literature has focused on minimizing the download cost. The efficiency of a PIR protocol is then measured in terms of the so-called PIR rate, defined as the ratio between the size of the requested data and the amount of downloaded data, where a higher PIR rate means a higher efficiency.

The PIR capacity, i.e., the maximum achievable PIR rate, was studied in [18], [19], [21]–[23]. In [19], [23], the PIR capacity was derived for the scenario where data is stored in a DSS using a repetition code. In [22], for the noncolluding case, the authors derived the PIR capacity for the scenario where data is stored using an (single) MDS code, referred to as the MDS-PIR capacity. In [21], it was shown that for certain non-MDS storage codes the MDS-PIR capacity can be achieved. For the case where several spy nodes collaborate with each other, referred to as the colluding case, the MDS-PIR capacity is in general still unknown, except for some special cases [18] (and for repetition codes [23]). PIR protocols for DSSs have been proposed in [14], [16], [17], [20], [21]. In [16], a PIR protocol for MDS-coded DSSs was proposed and shown to achieve the MDS-PIR capacity for the case of noncolluding nodes when the number of files stored in the DSS goes to infinity. PIR protocols for the case where data is stored using non-MDS codes were proposed in [17], [20], [21]. There are also several works on PIR that have gone beyond the classical distributed storage model. In [25], the authors considered PIR

S. Kumar and E. Rosnes were supported by the Research Council of Norway (grant 240985/F20). A. Graell i Amat was supported by the Swedish Research Council under grant #2016-04253.

S. Kumar and E. Rosnes are with Simula UiB, N-5020 Bergen, Norway (e-mail: {kumarsi,eirikrosnes}@simula.no).

A. Graell i Amat is with the Department of Electrical Engineering, Chalmers University of Technology, SE-41296 Gothenburg, Sweden (e-mail: alexandre.graell@chalmers.se).

L. Senigagliesi is with Dipartimento di Ingegneria dell'Informazione, Università Politecnica delle Marche, Ancona, Italy (e-mail: l.senigagliesi@pm.univpm.it).

with side information. The works [26]–[32] further generalized the system model in [25] and presented appropriate PIR schemes. A common theme across these papers is that they consider multiple servers and replication. In [29], [30], the authors presented PIR schemes for DSSs where the servers are constrained in storage capacity.

In this paper, we consider PIR of content from a cellular network. In particular, we consider the private retrieval of content from a library of files that have different popularities. We consider a coded caching scenario (in the form of [5], [7]), where to reduce the backhaul usage erasure correcting codes are used to cache content. In particular, we consider a similar scenario as in [7] where content is cached in SBSs using MDS codes. We propose a PIR scheme based on generalized Reed-Solomon (GRS) codes for this scenario that achieves privacy against a number of spy SBSs that possibly collude. The proposed PIR scheme is an extension of Protocol 3 in [21] to the case of multiple code rates, suitable for the scenario where files have different popularities. We also propose a slightly different MDS-coded content placement than the one in [7] but that is more adapted to the PIR case. We show that, for the conventional content retrieval scenario with no privacy, the proposed content placement is equivalent to the one in [7], in the sense that it yields the same backhaul rate. We then derive the backhaul rate for the PIR case as a function of the content placement. We prove that uniform content placement, i.e., all files that are cached are encoded with the same code rate, is optimal. This is a somewhat surprising result, in contrast to the case where no PIR is considered, where optimal content placement is far from uniform [7]. We further consider the minimization of a weighted sum of the backhaul rate and the communication rate from the SBSs, relevant for the case where limiting the communication from the SBSs is also important. We finally report numerical results for both the scenario where SBSs are placed regularly in a grid and for a Poisson point process (PPP) deployment model where SBSs are distributed over the plane according to a PPP. We show numerically that popular content placement is optimal for some system parameters. To the best of our knowledge, PIR for the wireless caching scenario has not been considered before.

The work most closely related to our work is [33], where the authors considered a different scenario where data is stored in a data center and in a number of databases with limited capacity. Furthermore, differently from our work, in [33] the databases store uncoded uniformly distributed data and they do not collude.

Notation: We use lowercase bold letters to denote vectors, uppercase bold letters to denote matrices, and calligraphic uppercase letters to denote sets. For example, $\boldsymbol{x}$, $\boldsymbol{X}$, and $\mathcal{X}$ denote a vector, a matrix, and a set, respectively. We denote a submatrix of $\boldsymbol{X}$ that is restricted in columns by the set $\mathcal{I}$ by $\boldsymbol{X}|_{\mathcal{I}}$. $\mathcal{C}$ will denote a linear code over the finite field $\mathrm{GF}(q)$. The multiplicative subgroup of $\mathrm{GF}(q)$ (not containing the zero element) is denoted by $\mathrm{GF}(q)^{\times}$. We use the customary code parameters (n, k) to denote a code $\mathcal{C}$ of block length n and dimension k. A generator matrix for $\mathcal{C}$ will be denoted by $\boldsymbol{G}^{\mathcal{C}}$ and a parity-check matrix by $\boldsymbol{H}^{\mathcal{C}}$. A set of coordinates of $\mathcal{C}$, $\mathcal{I} \subseteq \{1, \ldots, n\}$, of size k is said to be an *information set* if and only if $\boldsymbol{G}^{\mathcal{C}}|_{\mathcal{I}}$ is invertible. The Hadamard product of two linear subspaces $\mathcal{C}$ and $\mathcal{C}'$, denoted by $\mathcal{C} \circ \mathcal{C}'$, is the space generated by the Hadamard products $\boldsymbol{c} \circ \boldsymbol{c}' \triangleq (c_1 c'_1, \ldots, c_n c'_n)$ for all pairs $\boldsymbol{c} \in \mathcal{C}$, $\boldsymbol{c}' \in \mathcal{C}'$. The inner product of two vectors $\boldsymbol{x}$ and $\boldsymbol{x}'$ is denoted by $\langle \boldsymbol{x}, \boldsymbol{x}' \rangle$, while $w_{\mathsf{H}}(\boldsymbol{x})$ denotes the Hamming weight of $\boldsymbol{x}$. $(\cdot)^{\mathsf{T}}$ represents the transpose of its argument, while $\mathsf{H}(\cdot)$ represents the entropy function. An erasure pattern is a binary vector where the ones represent erased positions, while the zeros represent nonerased positions. The weight of an erasure pattern is the number of erased positions, and an erasure pattern $\boldsymbol{x}$ is said to be correctable by a code $\mathcal{C}$ if $\boldsymbol{H}^{\mathcal{C}}|_{\chi(\boldsymbol{x})}$ has rank $|\chi(\boldsymbol{x})|$, where $\chi(\cdot)$ denotes the support (i.e., the set of nonzero entries) of its argument.

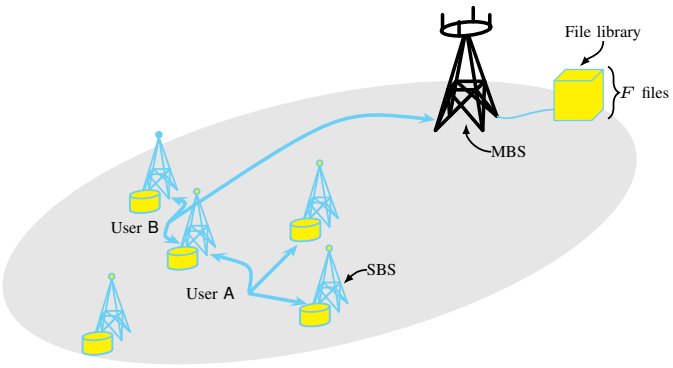

Fig. 1. A wireless network for content delivery consisting of an MBS and five SBSs. Users download files from a library of F files. The MBS has access to the library through a backhaul link. Some files are also cached at SBSs using a $(5, 3)$ MDS code. User A retrieves a cached file from the three SBSs within range. User B retrieves a fraction $2/3$ of a cached file from the two SBSs within range and the remaining fraction from the MBS.

II. System Model

We consider a cellular network where a macro-cell is served by a macro base station (MBS). Mobile users wish to download files from a library of F files that is always available at the MBS through a backhaul link. We assume all files of equal size, as content can always be divided into chunks of equal size. In particular, each file consists of βL bits and is represented by a $\beta \times L$ matrix $\boldsymbol{X}^{(i)}$,

$$\boldsymbol{X}^{(i)} = \begin{pmatrix} \tilde{\boldsymbol{x}}_1^{(i)} \\ \vdots \\ \tilde{\boldsymbol{x}}_\beta^{(i)} \end{pmatrix},$$

where superscript $i = 1, \ldots, F$ is the file index. Therefore, each file can be seen as divided into β stripes $\tilde{\boldsymbol{x}}_1^{(i)}, \ldots, \tilde{\boldsymbol{x}}_\beta^{(i)}$ of L bits each. The file library has popularity distribution $\boldsymbol{p} = (p_1, \ldots, p_F)$, where file $\boldsymbol{X}^{(i)}$ is requested with probability p_i. We also assume that N_{SBS} SBSs are deployed to serve requests and offload traffic from the MBS whenever possible. To this purpose, each SBS has a cache size equivalent to M files. The considered scenario is depicted in Fig. 1.

A. Content Placement

File $\boldsymbol{X}^{(i)}$ is partitioned into βk_i packets of size L/k_i bits and encoded before being cached in the SBSs. In particular,

each packet is mapped onto a symbol of the field $\mathrm{GF}(q^{\delta_i})$, with $\delta_i \geq \frac{L}{k_i \log_2 q}$. For simplicity, we assume that $\frac{L}{k_i \log_2 q}$ is integer and set $\delta_i = \frac{L}{k_i \log_2 q}$. Thus, stripe $\tilde{\boldsymbol{x}}_a^{(i)}$ can be equivalently represented by a stripe $\boldsymbol{x}_a^{(i)}$, $a = 1, \ldots, \beta$, of symbols over $\mathrm{GF}(q^{\delta_i})$. Each stripe $\boldsymbol{x}_a^{(i)}$ is then encoded using an (N_{SBS}, k_i) MDS code $\mathcal{C}_i$ over $\mathrm{GF}(q)$, $q > N_{\mathsf{SBS}}$, into a codeword $\boldsymbol{c}_a^{(i)} = (c_{a,1}^{(i)}, \ldots, c_{a,N_{\mathsf{SBS}}}^{(i)})$, where code symbols $c_{a,j}^{(i)}$, $j = 1, \ldots, N_{\mathsf{SBS}}$, are over $\mathrm{GF}(q^{\delta_i})$. For later use, we define $k_{\mathsf{min}} \triangleq \min\{k_i\}$, $k_{\mathsf{max}} \triangleq \max\{k_i\}$, and $\delta_{\mathsf{max}} \triangleq \frac{L}{k_{\mathsf{min}} \log_2 q}$.

The encoded file can be represented by a $\beta \times N_{\mathsf{SBS}}$ matrix $\boldsymbol{C}^{(i)} = (c_{a,j}^{(i)})$. Code symbols $c_{a,j}^{(i)}$ are then stored in the j-th SBS (the ordering is unimportant). Thus, for each file $\boldsymbol{X}^{(i)}$, each SBS caches one coded symbol of each stripe of the file. For analysis purposes, we define $\mu_i \triangleq 1/k_i$. As $k_i \in \{1, \ldots, N_{\mathsf{SBS}} - 1\}$,

$$\mu_i \in \mathcal{M} \triangleq \{0, 1/(N_{\mathsf{SBS}} - 1), \ldots, 1/2, 1\},$$

where $\mu_i = 0$ implies that file $\boldsymbol{X}^{(i)}$ is not cached. Note that, to achieve privacy in a nontrivial manner (i.e., without downloading everything), $k_i < N_{\mathsf{SBS}}$, i.e., files need to be cached with redundancy. As a result, $\mu_i = 1/N_{\mathsf{SBS}}$ is not allowed. This is in contrast to the case of no PIR, where $k_i = N_{\mathsf{SBS}}$ (and hence $\mu_i = 1/N_{\mathsf{SBS}}$) is possible.

Since each SBS can cache the equivalent of M files, the μ_i's must satisfy

$$\sum_{i=1}^{F} \mu_i \leq M.$$

We define the vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_F)$ and refer to it as the *content placement*. Also, we denote by $\mathcal{C}_{\mathsf{MDS}}^{\boldsymbol{\mu}}$ the caching scheme that uses MDS codes $\{\mathcal{C}_i\}$ according to the content placement $\boldsymbol{\mu}$. For later use, we define $\mu_{\mathsf{min}} \triangleq \min\{\mu_i | \mu_i \neq 0\}$ and $\mu_{\mathsf{max}} \triangleq \max\{\mu_i\}$.

We remark that the content placement above is slightly different than the content placement proposed in [7]. In particular, we assume fixed code length (equal to the number of SBSs, N_{SBS}) and variable k_i, such that, for each file cached, each SBS caches a single symbol from each stripe of the file. In [7], the content placement is done by first dividing each file into k symbols and encoding them using an $(\tilde{n}_i, k)$ MDS code, where $\tilde{n}_i = k + (N_{\mathsf{SBS}} - 1)m_i$, $m_i \leq k$. Then, m_i (different) symbols of the i-th file are stored in each SBS and the MBS stores $k - m_i$ symbols.[1] Our formulation is perhaps a bit simpler and more natural from a coding perspective. Furthermore, we will show in Section IV that the proposed content placement is equivalent to the one in [7], in the sense that it yields the same backhaul rate.

B. File Request

Mobile devices request files according to the popularity distribution $\boldsymbol{p} = (p_1, \ldots, p_F)$. Without loss of generality, we assume $p_1 \geq p_2 \geq \ldots \geq p_F$. The user request is initially served by the SBSs within communication range. We denote by γ_b the probability that the user is served by b SBSs and define $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_{N_{\mathsf{SBS}}})$. If the user is not able to completely retrieve $\boldsymbol{X}^{(i)}$ from the SBSs, the additional required symbols are fetched from the MBS. Using the terminology in [7], the average fraction of files that are downloaded from the MBS is referred to as the backhaul rate, denoted by R, and defined as

$$\mathsf{R} \triangleq \frac{\text{average no. of bits downloaded from the MBS}}{\beta L}.$$

Note that for the case of no caching $\mathsf{R} = 1$.

As in [7], we assume that the communication is error free.

C. Private Information Retrieval and Problem Formulation

We assume that some of the SBSs are spy nodes that (potentially) collaborate with each other. On the other hand, we assume that the MBS can be trusted. The users wish to retrieve files from the cellular network, but do not want the spy SBSs to learn any information about which file is requested by the user. The goal is to retrieve data from the network privately while minimizing the use of the backhaul link, i.e., while minimizing R. Thus, the goal is to optimize the content placement $\boldsymbol{\mu}$ to minimize R.

III. Private Information Retrieval Protocol

In this section, we present a PIR protocol for the caching scenario. The PIR protocol proposed here is an extension of Protocol 3 in [21] to the case of multiple code rates.[2] For the classical DSS setting and the case of no colluding servers and an MDS code as the underlying storage code, Protocol 3 achieves the PIR capacity (derived in [22]) as the number of files tends to infinity. Furthermore, for the case of colluding servers and replication, Protocol 3 also achieves the PIR capacity (derived in [23]) as the number of files tends to infinity.

Assume without loss of generality that the user wants to download file $\boldsymbol{X}^{(i)}$. To retrieve the file, the user generates $n \leq N_{\mathsf{SBS}}$ query matrices, $\boldsymbol{Q}^{(l)}$, $l = 1, \ldots, n$, where $\boldsymbol{Q}^{(1)}, \ldots, \boldsymbol{Q}^{(b)}$ are the queries sent to b SBSs within visibility and the remaining $n - b$ queries $\boldsymbol{Q}^{(b+1)}, \ldots, \boldsymbol{Q}^{(n)}$ are sent to the MBS, unless $b = 0$ (i.e., there are no SBSs within communication range) in which case the file is downloaded directly from the MBS since by assumption it can be trusted. Note that n is a parameter that needs to be optimized. Each query matrix is of size $d \times \beta F$ symbols (from $\mathrm{GF}(q)$) and has the following structure,

$$\boldsymbol{Q}^{(l)} = \begin{pmatrix} \boldsymbol{q}_1^{(l)} \\ \vdots \\ \boldsymbol{q}_d^{(l)} \end{pmatrix} = \begin{pmatrix} q_{1,1}^{(l)} & \cdots & q_{1,\beta F}^{(l)} \\ \vdots & \cdots & \vdots \\ q_{d,1}^{(l)} & \cdots & q_{d,\beta F}^{(l)} \end{pmatrix}.$$

The query matrix $\boldsymbol{Q}^{(l)}$ consists of d subqueries $\boldsymbol{q}_j^{(l)}$, $j = 1, \ldots, d$, of length βF symbols each. In response to query matrix $\boldsymbol{Q}^{(l)}$, a SBS (or the MBS) sends back to the user a

[1] This is because the model in [7] assumes that one SBS is always accessible to the user. If this is not the case, the MBS must store all k symbols of the file. Here, we consider the case where the MBS must store all k symbols because it is a bit more general.

[2] Protocol 3 in [21] is based on and improves the protocol in [20], in the sense that it achieves higher PIR rates.

response vector $\boldsymbol{r}^{(l)} = (r_1^{(l)}, \dots, r_d^{(l)})^{\mathsf{T}}$ of length d, computed as

$$\begin{aligned}\boldsymbol{r}^{(l)} &= (r_1^{(l)}, \dots, r_d^{(l)})^{\mathsf{T}} \\ &= \boldsymbol{Q}^{(l)}\big(c_{1,l}^{(1)}, \dots, c_{\beta,l}^{(1)}, \dots, c_{1,l}^{(F)}, \dots, c_{\beta,l}^{(F)}\big)^{\mathsf{T}}.\end{aligned} \tag{1}$$

We will denote the j-th entry of the response vector $\boldsymbol{r}^{(l)}$, i.e., $r_j^{(l)}$, as the j-th subresponse of $\boldsymbol{r}^{(l)}$. Each response vector consists of d subresponses, each being a linear combination of βF symbols. Note that the operations are performed over the largest extension field, i.e., $\mathrm{GF}(q^{\delta_{\mathsf{max}}})$, and the subresponses are also over this field, i.e., each subresponse is of size $L/k_{\mathsf{min}} = L\mu_{\mathsf{max}}$ bits and hence each response is of size $dL\mu_{\mathsf{max}}$ bits.

The queries and the responses must be such that privacy is ensured and the user is able to recover the requested file. More precisely, information-theoretic PIR in the context of wireless caching with spy SBSs is defined as follows.

Definition 1. *Consider a wireless caching scenario with N_{SBS} SBSs that cache parts of a library of F files and in which an arbitrary set $\mathcal{T}$ of T SBSs act as colluding spies. A user wishes to retrieve the i-th file and generates queries $\boldsymbol{Q}^{(l)}$, $l = 1, \dots, n$. In response to the queries the SBSs and (potentially) the MBS send back the responses $\boldsymbol{r}^{(l)}$. This scheme achieves perfect information-theoretic PIR if and only if*

$$\textit{Privacy:} \qquad \mathsf{H}\big(i|\{\boldsymbol{Q}^{(l)} : l \in \mathcal{T}\}\big) = \mathsf{H}(i); \tag{2a}$$

$$\textit{Recovery:} \qquad \mathsf{H}\big(\boldsymbol{X}^{(i)}|\boldsymbol{r}^{(1)}, \dots, \boldsymbol{r}^{(n)}\big) = 0. \tag{2b}$$

Condition (2a) means that the spy SBSs gain no additional information about which file is requested from the queries (i.e., the uncertainty about the file requested after observing the queries is identical to the a priori uncertainty determined by the popularity distribution), while Condition (2b) guarantees that the user is able to recover the file from the n response vectors.

We define the (n, k_i) code $\mathcal{C}'_i$, $i = 1, \dots, F$, as the code obtained by puncturing the underlying (N_{SBS}, k_i) storage code $\mathcal{C}_i$, and by $\mathcal{C}'_{\mathsf{max}}$ the code with parameters (n, k_{max}).[3] For the protocol to work, we require that k_{min} divides k_i for all i, i.e., $k_{\mathsf{min}} \mid k_i$. This ensures that $\mathrm{GF}(q^{\delta_i}) \subseteq \mathrm{GF}(q^{\delta_{\mathsf{max}}})$. Furthermore, we require the codes $\mathcal{C}'_i$ to be such that $\mathcal{C}'_i \subseteq \mathcal{C}'_{\mathsf{max}}$. The protocol is characterized by the codes $\{\mathcal{C}'_i\}$ and by two other codes, $\bar{\mathcal{C}}$ and $\tilde{\mathcal{C}}$. Code $\bar{\mathcal{C}}$ (over $\mathrm{GF}(q)$) has parameters $(n, \bar{k})$ and characterizes the queries sent to the SBSs and the MBS, while code $\tilde{\mathcal{C}}$ (defined below) defines the responses sent back to the user from the SBSs and the MBS. The designed protocol achieves PIR against a number of colluding SBSs $T \leq d_{\mathsf{min}}^{\bar{\mathcal{C}}^\perp} - 1$, where $d_{\mathsf{min}}^{\bar{\mathcal{C}}^\perp}$ is the minimum Hamming distance of the dual code of $\bar{\mathcal{C}}$.

A. Query Construction

The queries must be constructed such that privacy is preserved and the user can retrieve the requested file from the n response vectors $\boldsymbol{r}^{(l)}$, $l = 1, \dots, n$. In particular, the protocol is designed such that the subresponses $r_j^{(l)}$, $l = 1, \dots, n$, corresponding to the n subqueries $\boldsymbol{q}_j^{(1)}, \dots, \boldsymbol{q}_j^{(n)}$ recover Γ unique code symbols of the file $\boldsymbol{X}^{(i)}$.

The queries are constructed as follows. The user chooses βF codewords $\bar{\boldsymbol{c}}_m^{(i)} = (\bar{c}_{m,1}^{(i)}, \dots, \bar{c}_{m,n}^{(i)}) \in \bar{\mathcal{C}}$, $m = 1, \dots, \beta$, $i = 1, \dots, F$, independently and uniformly at random. Then, the user constructs n vectors,

$$\mathring{\boldsymbol{c}}_l = (\mathring{\boldsymbol{c}}_l^{(1)}, \dots, \mathring{\boldsymbol{c}}_l^{(F)}), \quad l = 1, \dots, n, \tag{3}$$

where $\mathring{\boldsymbol{c}}_l^{(i)}$ collects the l-th coordinates of the β codewords $\bar{\boldsymbol{c}}_m^{(i)}$, $m = 1, \dots, \beta$, i.e., $\mathring{\boldsymbol{c}}_l^{(i)} = (\bar{c}_{1,l}^{(i)}, \dots, \bar{c}_{\beta,l}^{(i)})$.

Assume that the user wants to retrieve file $\boldsymbol{X}^{(i)}$. Then, subquery $\boldsymbol{q}_j^{(l)}$ is constructed as

$$\boldsymbol{q}_j^{(l)} = \mathring{\boldsymbol{c}}_l + \boldsymbol{\delta}_j^{(l)}, \tag{4}$$

where

$$\boldsymbol{\delta}_j^{(l)} = \begin{cases} \boldsymbol{\omega}_{\beta(i-1)+s_j^{(l)}} & \text{if } l \in \mathcal{J}_j, \\ \boldsymbol{\omega}_0 & \text{otherwise,} \end{cases} \tag{5}$$

for some set $\mathcal{J}_j$ that will be defined below. Vector $\boldsymbol{\omega}_t$, $t = 1, \dots, \beta F$, denotes the t-th (βF)-dimensional unit vector, i.e., the length-βF vector with a one in the t-th coordinate and zeroes in all other coordinates, and $\boldsymbol{\omega}_0$ the all-zero vector. The meaning of index $s_j^{(l)}$ will become apparent later.

According to (4), each subquery vector is the sum of two vectors, $\mathring{\boldsymbol{c}}_l$ and $\boldsymbol{\delta}_j^{(l)}$. The purpose of $\mathring{\boldsymbol{c}}_l$ is to make the subquery appear random and thus ensure privacy (i.e., Condition (2a)). On the other hand, the vectors $\boldsymbol{\delta}_j^{(l)}$ are deterministic vectors which must be properly constructed such that the user is able to retrieve the requested file from the response vectors (i.e., Condition (2b)). Similar to Protocol 3 in [21], the vectors $\boldsymbol{\delta}_j^{(l)}$ are constructed from a $d \times n$ binary matrix $\hat{\boldsymbol{E}}$ where each row represents a weight-Γ erasure pattern that is correctable by $\tilde{\mathcal{C}}$ and where the weights of its columns are determined from β information sets $\mathcal{I}_m$, $m = 1, \dots, \beta$, of $\mathcal{C}'_{\mathsf{max}}$.

The construction of $\hat{\boldsymbol{E}}$ is addressed below. We define the set $\mathcal{F}_l$ as the index set of information sets $\mathcal{I}_m$ that contain the l-th coordinate of $\mathcal{C}'_{\mathsf{max}}$, i.e., $\mathcal{F}_l = \{m : l \in \mathcal{I}_m\}$. To allow the user to recover the requested file from the response vectors, $\hat{\boldsymbol{E}}$ is constructed such that it satisfies the following conditions.

- C1. The user should be able to recover Γ unique code symbols of the requested file $\boldsymbol{X}^{(i)}$ from the responses to each set of n subqueries $\boldsymbol{q}_j^{(l)}$, $l = 1, \dots, n$. This is to say that each row of $\hat{\boldsymbol{E}}$ should have exactly Γ ones. We denote by $\mathcal{J}_j$ the support of the j-th row of $\hat{\boldsymbol{E}}$.
- C2. The user should be able to recover $\Gamma d \geq \beta k_i$ unique code symbols of the requested file $\boldsymbol{X}^{(i)}$, at least k_i symbols from each stripe. This means that each row $\hat{\boldsymbol{e}}_j = (\hat{e}_{j,1}, \dots, \hat{e}_{j,n})$, $j = 1, \dots, d$, of $\hat{\boldsymbol{E}}$ should be an erasure pattern that is correctable by $\tilde{\mathcal{C}}$.
- C3. Let $\boldsymbol{t}_l$, $l = 1, \dots, n$, be the l-th column vector of $\hat{\boldsymbol{E}}$. The protocol should be able to recover $w_{\mathsf{H}}(\boldsymbol{t}_l)$ unique code symbols from the l-th response vector, which means that it is required that $w_{\mathsf{H}}(\boldsymbol{t}_l) = |\mathcal{F}_l|$. We call the vector $(w_{\mathsf{H}}(\boldsymbol{t}_1), \dots, w_{\mathsf{H}}(\boldsymbol{t}_n))$ the *column weight profile of* $\hat{\boldsymbol{E}}$.

[3] Without loss of generality, to simplify notation we assume that the rightmost coordinates of the code are punctured.

The existence of a scheme that satisfies conditions C1–C3 is shown in the appendix, as part of the proof of Theorem 1. Finally, from $\hat{\boldsymbol{E}}$ we construct the vectors $\boldsymbol{\delta}_j^{(l)}$ in (5). In particular, index $s_j^{(l)}$ in (5) is such that $s_j^{(l)} \in \mathcal{F}_l$ and $s_j^{(l)} \neq s_{j'}^{(l)}$ for $j \neq j'$, $j, j' = 1, \ldots, d$.

B. Response Vectors

The j-th subresponse corresponding to subquery $\boldsymbol{q}_j^{(l)}$, $j = 1, \ldots, d$, is (see (1))

$$r_j^{(l)} = \langle \boldsymbol{q}_j^{(l)}, (c_{1,l}^{(1)}, \ldots, c_{\beta,l}^{(F)}) \rangle.$$

The user collects the n subresponses $r_j^{(l)}$, $l = 1, \ldots n$, in the vector $\boldsymbol{\rho}_j$,

$$\boldsymbol{\rho}_j = \begin{pmatrix} r_j^{(1)} \\ r_j^{(2)} \\ \vdots \\ r_j^{(n)} \end{pmatrix} = \sum_{m=1}^{\beta} \underbrace{\begin{pmatrix} \bar{c}_{m,1}^{(1)} c_{m,1}^{(1)} \\ \bar{c}_{m,2}^{(1)} c_{m,2}^{(1)} \\ \vdots \\ \bar{c}_{m,n}^{(1)} c_{m,n}^{(1)} \end{pmatrix}}_{\in \left\{ \boldsymbol{x} \in (\mathrm{GF}(q^{\delta_{\max}}))^n : \boldsymbol{H}^{\mathcal{C}_1' \circ \bar{\mathcal{C}}} \boldsymbol{x} = \boldsymbol{0} \right\}} + \underbrace{\begin{pmatrix} \bar{c}_{m,1}^{(2)} c_{m,1}^{(2)} \\ \bar{c}_{m,2}^{(2)} c_{m,2}^{(2)} \\ \vdots \\ \bar{c}_{m,n}^{(2)} c_{m,n}^{(2)} \end{pmatrix}}_{\in \left\{ \boldsymbol{x} \in (\mathrm{GF}(q^{\delta_{\max}}))^n : \boldsymbol{H}^{\mathcal{C}_2' \circ \bar{\mathcal{C}}} \boldsymbol{x} = \boldsymbol{0} \right\}} + \cdots + \underbrace{\begin{pmatrix} \bar{c}_{m,1}^{(F)} c_{m,1}^{(F)} \\ \bar{c}_{m,2}^{(F)} c_{m,2}^{(F)} \\ \vdots \\ \bar{c}_{m,n}^{(F)} c_{m,n}^{(F)} \end{pmatrix}}_{\in \left\{ \boldsymbol{x} \in (\mathrm{GF}(q^{\delta_{\max}}))^n : \boldsymbol{H}^{\mathcal{C}_F' \circ \bar{\mathcal{C}}} \boldsymbol{x} = \boldsymbol{0} \right\}} + \begin{pmatrix} o_j^{(1)} \\ o_j^{(2)} \\ \vdots \\ o_j^{(n)} \end{pmatrix}, \tag{6}$$

where symbol $o_j^{(l)}$ represents the code symbol from file $\boldsymbol{X}^{(i)}$ downloaded in the j-th subresponse from the l-th response vector, or zero if no symbol is downloaded in the j-th subresponse from the l-th response vector. Due to the structure of the queries obtained from $\hat{\boldsymbol{E}}$, the user retrieves Γ code symbols from the set of n subresponses to the j-th subqueries. Consider a retrieval code $\tilde{\mathcal{C}}$ of the form

$$\tilde{\mathcal{C}} = \sum_{i=1}^{F} \mathcal{C}_i' \circ \bar{\mathcal{C}} \overset{(a)}{=} \left(\sum_{i=1}^{F} \mathcal{C}_i' \right) \circ \bar{\mathcal{C}}, \tag{7}$$

where $\mathcal{C}_i' + \mathcal{C}_j'$ denotes the sum of subspaces $\mathcal{C}_i'$ and $\mathcal{C}_j'$, resulting in the set consisting of all elements $\boldsymbol{c} + \boldsymbol{c}'$ for any $\boldsymbol{c} \in \mathcal{C}_i'$ and $\boldsymbol{c}' \in \mathcal{C}_j'$, and where (a) follows due to the fact that the Hadamard product is distributive over addition.

The symbols requested by the user are then obtained by solving the system of linear equations defined by

$$\boldsymbol{H}^{\tilde{\mathcal{C}}} \boldsymbol{\rho}_j = \boldsymbol{H}^{\tilde{\mathcal{C}}} \begin{pmatrix} o_j^{(1)} \\ o_j^{(2)} \\ \vdots \\ o_j^{(n)} \end{pmatrix}.$$

C. Privacy

For the retrieval, we require $\tilde{\mathcal{C}}$ to be a valid code, i.e., it must have a code rate strictly less than 1. For a given number of colluding SBSs T, the combination of conditions on $\bar{\mathcal{C}}$ and $\tilde{\mathcal{C}}$ restricts the choice for the underlying storage codes $\{\mathcal{C}_i\}$. In the following theorem, we present a family of MDS codes, namely GRS codes, that work with the protocol. A GRS code $\mathcal{C}$ over $\mathrm{GF}(q)$ of length n and dimension k is a weighted polynomial evaluation code of degree k defined by a weighting vector $\boldsymbol{v} = (v_1, \ldots, v_n) \in (\mathrm{GF}(q)^{\times})^n$ and an evaluation vector $\boldsymbol{\kappa} = (\kappa_1, \ldots, \kappa_n) \in (\mathrm{GF}(q)^{\times})^n$ satisfying $\kappa_i \neq \kappa_j$ for all $i \neq j$ [34, Ch. 5]. In the sequel, we refer to $(n, k, \boldsymbol{v}, \boldsymbol{\kappa})$ as the parameters of a GRS code $\mathcal{C}$.

Lemma 1. *Given an $(n, k_{\max}, \boldsymbol{v}, \boldsymbol{\kappa})$ GRS code $\mathcal{C}_{\max}$, for all $k < k_{\max}$, there exists an $(n, k, \boldsymbol{v}, \boldsymbol{\kappa})$ GRS code that is a subcode of $\mathcal{C}_{\max}$.*

Proof: The canonical generator matrix of an $(n, k_{\max}, \boldsymbol{v}, \boldsymbol{\kappa})$ GRS code $\mathcal{C}_{\max}$ is given by

$$\begin{pmatrix} 1 & 1 & \ldots & 1 \\ \kappa_1 & \kappa_2 & \ldots & \kappa_n \\ \vdots & \vdots & \ldots & \vdots \\ \kappa_1^{k_{\max}-1} & \kappa_2^{k_{\max}-1} & \ldots & \kappa_n^{k_{\max}-1} \end{pmatrix} \begin{pmatrix} v_1 & 0 & \ldots & 0 \\ 0 & v_2 & \ldots & 0 \\ \vdots & \vdots & \ldots & \vdots \\ 0 & 0 & \ldots & v_n \end{pmatrix}. \tag{8}$$

Clearly, taking the first k rows of the leftmost matrix of (8) and multiplying it with the rightmost diagonal matrix generates an (n, k) subcode of $\mathcal{C}_{\max}$ which by itself is an $(n, k, \boldsymbol{v}, \boldsymbol{\kappa})$ GRS code with the same weighting vector $\boldsymbol{v}$. Thus, GRS codes are naturally nested, and the result follows. ∎

Theorem 1. *Let $\mathcal{C}_{\mathsf{MDS}}^{\boldsymbol{\mu}}$ be a caching scheme with GRS codes $\{\mathcal{C}_i\}$ of parameters $(N_{\mathsf{SBS}}, k_i, \boldsymbol{v}, (\kappa_1, \ldots, \kappa_{N_{\mathsf{SBS}}}))$ and let $\mathcal{C}_i'$ be the (n, k_i) code obtained by puncturing $\mathcal{C}_i$. Also, let $\bar{\mathcal{C}}$ be an $(n, T, \bar{\boldsymbol{v}}, (\kappa_1, \ldots, \kappa_n))$ GRS code, and let $T \leq n - k_{\max}$. Then, for $\beta = \Gamma = n - (k_{\max} + T - 1)$ and $d = k_{\max}$, the protocol achieves PIR against up to T colluding SBSs.*

Proof: The proof is given in the appendix. ∎

Note that the query code $\bar{\mathcal{C}}$ depends on the b SBSs within visibility that are contacted by the user through its evaluation vector.

The proposed protocol achieves a PIR rate of

$$\frac{\beta k_i \frac{L}{k_i}}{n d \frac{L}{k_{\min}}} = \frac{n - (k_{\max} + T - 1)}{n} \cdot \frac{k_{\min}}{k_{\max}},$$

unless $b = 0$ (i.e., there are no SBSs within communication range) in which case the requested file is downloaded directly from the MBS since by assumption it can be trusted. Furthermore, with some slight modifications, the proposed protocol can be adapted to work with non-MDS codes.

Remark 1. *For the particular case of noncolluding SBSs ($T = 1$), one can alternatively have the code $\mathcal{C}_i$ to be an (N_{SBS}, k_i) code such that the punctured code $\mathcal{C}_i' \subseteq \mathcal{C}_{\max}'$, and have the code $\bar{\mathcal{C}}$ to be an $(n, 1)$ binary repetition code. Then, from* (7) *the resulting retrieval code is $\tilde{\mathcal{C}} = \mathcal{C}_{\max}'$. The privacy and*

retrievability can be guaranteed in a similar way as shown in the proof of Theorem 1.

D. Example

As an example, consider the case of $F = 2$ files, $\boldsymbol{X}^{(1)}$ and $\boldsymbol{X}^{(2)}$, both of size $\beta L = 5$ bits. The first file $\boldsymbol{X}^{(1)}$ is stored in the SBSs according to Fig. 2 using an $(N_{\mathsf{SBS}} = 6, k_1 = 1)$ binary repetition code $\mathcal{C}_1$. Similarly, the second file $\boldsymbol{X}^{(2)}$ is stored (again according to Fig. 2) using an $(N_{\mathsf{SBS}} = 6, k_2 = 5)$ binary single parity-check code $\mathcal{C}_2$. Assume $n = N_{\mathsf{SBS}} = 6$ (i.e., no puncturing) and that none of the SBSs collude, i.e., $T = 1$. Furthermore, we assume that the user wants to retrieve $\boldsymbol{X}^{(1)}$ and is able to contact $b = n = 6$ SBSs (i.e., we consider the extreme case where the user is not contacting the MBS). According to Remark 1 (and Theorem 1), we can choose $\beta = \Gamma = n-(k_{\mathsf{max}}+T-1) = 6-(5+1-1) = 1$ and $d = k_{\mathsf{max}} = 5$. Finally, we choose $\bar{\mathcal{C}}$ as an $(n = 6, 1)$ binary repetition code.

According to (7), the retrieval code $\tilde{\mathcal{C}} = (\mathcal{C}_1 + \mathcal{C}_2) \circ \bar{\mathcal{C}} = \mathcal{C}_1 + \mathcal{C}_2 = \mathcal{C}_2$ and can be generated by

$$\boldsymbol{G}^{\tilde{\mathcal{C}}} = \boldsymbol{G}^{\mathcal{C}_2} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Moreover, let

$$\hat{\boldsymbol{E}} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix} \quad \text{and} \quad \mathcal{I}_1 = \{1, 2, 3, 4, 5\},$$

where $\mathcal{I}_1$ is an information set of $\mathcal{C}_{\mathsf{max}} = \mathcal{C}_2$ (the submatrix $\boldsymbol{G}^{\mathcal{C}_2}_{\mathcal{I}_1}$ has rank $k_2 = 5$). Note that $\hat{\boldsymbol{E}}$ satisfies all three conditions C1–C3 and has column weight profile $(1, 1, 1, 1, 1, 0) = (|\mathcal{F}_1|, \ldots, |\mathcal{F}_6|)$.

Query Construction. The user generates $\beta F = 2$ codewords $\bar{\boldsymbol{c}}_1^{(1)}$ and $\bar{\boldsymbol{c}}_1^{(2)}$ independently and uniformly at random from $\bar{\mathcal{C}}$. Without loss of generality, let $\bar{\boldsymbol{c}}_1^{(1)} = \bar{\boldsymbol{c}}_1^{(2)} = (1, \ldots, 1)$. Next, the $n = 6$ subqueries $q_1^{(l)}$, $l = 1, \ldots, 6$, are constructed according to (4), (5) as

$$\boldsymbol{q}_1^{(l)} = \begin{cases} \mathring{\boldsymbol{c}}_l + (1, 0) & \text{if } l = 1, \\ \mathring{\boldsymbol{c}}_l + (0, 0) & \text{otherwise,} \end{cases}$$

where $\mathring{\boldsymbol{c}}_l$ is defined in (3).

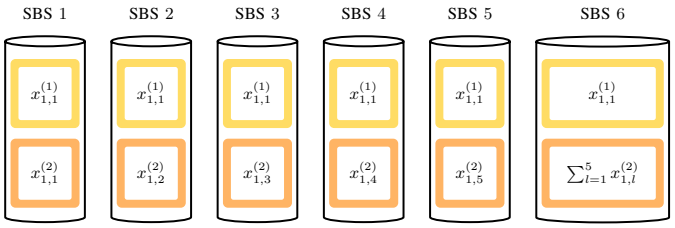

Fig. 2. Wireless caching scenario in which there are $N_{\mathsf{SBS}} = 6$ SBSs. The SBSs store $F = 2$ files, $\boldsymbol{X}^{(1)} = (x_{1,1}^{(1)}) \in \mathrm{GF}(2^5)^{1\times 1}$ and $\boldsymbol{X}^{(2)} = (x_{1,1}^{(2)}, x_{1,2}^{(2)}, x_{1,3}^{(2)}, x_{1,4}^{(2)}, x_{1,5}^{(2)}) \in \mathrm{GF}(2)^{1\times 5}$, of $\beta L = 5$ bits each. The first file $\boldsymbol{X}^{(1)}$ is encoded using an $(N_{\mathsf{SBS}} = 6, k_1 = 1)$ binary repetition code $\mathcal{C}_1$, while the second file $\boldsymbol{X}^{(2)}$ is encoded using an $(N_{\mathsf{SBS}} = 6, k_2 = 5)$ binary single parity-check code $\mathcal{C}_2$.

File Retrieval. Consider the $n = 6$ subresponses $r_1^{(l)}$, $l = 1, \ldots, 6$. Then, according to (6),

$$\boldsymbol{\rho}_1 = \begin{pmatrix} r_1^{(1)} \\ r_1^{(2)} \\ r_1^{(3)} \\ r_1^{(4)} \\ r_1^{(5)} \\ r_1^{(6)} \end{pmatrix} = \underbrace{\begin{pmatrix} \bar{c}_{1,1}^{(1)} c_{1,1}^{(1)} \\ \bar{c}_{1,2}^{(1)} c_{1,2}^{(1)} \\ \vdots \\ \bar{c}_{1,6}^{(1)} c_{1,6}^{(1)} \end{pmatrix}}_{\in \left\{\boldsymbol{x} \in (\mathrm{GF}(2^5))^n : \boldsymbol{H}^{\mathcal{C}_1' \circ \bar{\mathcal{C}}} \boldsymbol{x} = \boldsymbol{0}\right\}} + \underbrace{\begin{pmatrix} \bar{c}_{1,1}^{(2)} c_{1,1}^{(2)} \\ \bar{c}_{1,2}^{(2)} c_{1,2}^{(2)} \\ \vdots \\ \bar{c}_{1,6}^{(2)} c_{1,6}^{(2)} \end{pmatrix}}_{\in \left\{\boldsymbol{x} \in (\mathrm{GF}(2^5))^n : \boldsymbol{H}^{\mathcal{C}_2' \circ \bar{\mathcal{C}}} \boldsymbol{x} = \boldsymbol{0}\right\}} + \begin{pmatrix} o_1^{(1)} \\ o_1^{(2)} \\ \vdots \\ o_1^{(6)} \end{pmatrix}$$

$$= \begin{pmatrix} x_{1,1}^{(1)} \\ x_{1,1}^{(1)} \\ x_{1,1}^{(1)} \\ x_{1,1}^{(1)} \\ x_{1,1}^{(1)} \\ x_{1,1}^{(1)} \end{pmatrix} + \begin{pmatrix} x_{1,1}^{(2)} \\ x_{1,2}^{(2)} \\ x_{1,3}^{(2)} \\ x_{1,4}^{(2)} \\ x_{1,5}^{(2)} \\ \sum_{l=1}^{5} x_{1,l}^{(2)} \end{pmatrix} + \begin{pmatrix} x_{1,1}^{(1)} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

and the code symbol $x_{1,1}^{(1)}$ of the file $\boldsymbol{X}^{(1)}$ is recovered from

$$\boldsymbol{H}^{\tilde{\mathcal{C}}} \boldsymbol{\rho}_1 = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_{1,1}^{(1)} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = x_{1,1}^{(1)}.$$

Note that in order to retain privacy across the two files of the library, we need to send $d = k_{\mathsf{max}} = 5$ subqueries to each SBS, thus generating 5 subresponses from each SBS (even if the first file can be recovered from the $n = 6$ subresponses $r_1^{(l)}$, $l = 1, \ldots, 6$).

IV. Backhaul Rate Analysis: No PIR Case

In this section, we derive the backhaul rate for the proposed caching scheme for the case of no PIR, i.e., the conventional caching scenario where PIR is not required.

Proposition 1. *The backhaul rate for the caching scheme $\mathcal{C}^{\boldsymbol{\mu}}_{\mathsf{MDS}}$ in Section II for the case of no PIR is*

$$\begin{aligned}&\mathsf{R}_{\mathsf{noPIR}}\\&=\sum_{i=1}^{F} p_i\lceil\mu_i\rceil\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\max\left(0,1/\mu_i-b\right)\mu_i+\sum_{i=1}^{F}p_i\lfloor 1-\mu_i\rfloor.\end{aligned}\tag{9}$$

Proof: To download file $\boldsymbol{X}^{(i)}$, if the user is in communication range of a number of SBSs, b, larger than or equal to $1/\mu_i$, the user can retrieve the file from the SBSs and there is no contribution to the backhaul rate. Otherwise, if $b<1/\mu_i$, the user retrieves a fraction $L/k_i=L\mu_i$ of the file from each of the b SBSs, i.e., a total of $b\beta L\mu_i$ bits, and downloads the remaining $(1/\mu_i-b)\beta L\mu_i$ bits from the MBS. Averaging over $\boldsymbol{\gamma}$ and $\boldsymbol{p}$ (for the files cached) and normalizing by the file size βL, the contribution to the backhaul rate of the retrieval of files that are cached in the SBSs is

$$\sum_{i=1}^{F} p_i\lceil\mu_i\rceil\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\max\left(0,1/\mu_i-b\right)\mu_i.\tag{10}$$

On the other hand, the files that are not cached are retrieved completely from the MBS, and their contribution to the backhaul rate is

$$\sum_{i=1}^{F}p_i\lfloor 1-\mu_i\rfloor.\tag{11}$$

Combining (10) and (11) completes the proof. ∎

We denote by $\mathsf{R}^*_{\mathsf{noPIR}}$ the minimum backhaul rate resulting from the optimization of the content placement. $\mathsf{R}^*_{\mathsf{noPIR}}$ can be obtained by solving the following optimization problem,

$$\begin{aligned}\mathsf{R}^*_{\mathsf{noPIR}}=\min_{\mu_i\in\mathcal{M}'}&\sum_{i=1}^{F} p_i\lceil\mu_i\rceil\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\max\left(0,1/\mu_i-b\right)\mu_i\\&+\sum_{i=1}^{F}p_i\lfloor 1-\mu_i\rfloor\\\text{s.t.}&\sum_{i=1}^{F}\mu_i\le M,\end{aligned}$$

where $\mathcal{M}'=\mathcal{M}\cup\{1/N_{\mathsf{SBS}}\}$, as $\mu_i=1/N_{\mathsf{SBS}}$ is a valid value for the case where PIR is not required.

In the following lemma, we show that the proposed content placement is equivalent to the one in [7], in the sense that it yields the same backhaul rate.

Lemma 2. *The backhaul rate given by* (9) *for the caching scheme $\mathcal{C}^{\boldsymbol{\mu}}_{\mathsf{MDS}}$ in Section II is equal to the one given by the caching scheme in [7], i.e., the two content placements are equivalent.*

Proof: We can rewrite (9) as

$$\begin{aligned}&\mathsf{R}_{\mathsf{noPIR}}\\&=\sum_{i=1}^{F} p_i\lceil\mu_i\rceil\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\max\left(0,1/\mu_i-b\right)\mu_i+\sum_{i=1}^{F}p_i\lfloor 1-\mu_i\rfloor\\&=\sum_{i=1}^{F} p_i\lceil\mu_i\rceil\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\max\left(0,1-b\mu_i\right)+\sum_{i=1}^{F}p_i\lfloor 1-\mu_i\rfloor\\&=\sum_{i=1}^{F} p_i\lceil\mu_i\rceil\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\big(1-\min\left(1,b\mu_i\right)\big)+\sum_{i=1}^{F}p_i\lfloor 1-\mu_i\rfloor\\&\overset{(a)}{=}\sum_{i=1}^{F} p_i(\lceil\mu_i\rceil+\lfloor 1-\mu_i\rfloor)\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\big(1-\min\left(1,b\mu_i\right)\big)\\&=\sum_{i=1}^{F} p_i\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\big(1-\min\left(1,b\mu_i\right)\big),\end{aligned}$$

which is the expression in [7, eq. (1)]. (a) follows from the fact that we can write $p_i\lfloor 1-\mu_i\rfloor$ as $p_i\lfloor 1-\mu_i\rfloor\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\big(1-\min\left(1,b\mu_i\right)\big)$. For $0<\mu_i\le 1$ both expressions are zero, while for $\mu_i=0$ both expressions boil down to p_i as $p_i\lfloor 1-\mu_i\rfloor\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b\big(1-\min\left(1,b\mu_i\right)\big)=p_i\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b$ and $\sum_{b=0}^{N_{\mathsf{SBS}}}\gamma_b=1$. ∎

For popular content placement, i.e., the case where the M most popular files are cached in all SBSs (this corresponds to caching the M most popular files using an $(N_{\mathsf{SBS}},1)$ repetition code, i.e., $\mu_i=1$ for $i\le M$ and $\mu_i=0$ for $i>M$), the backhaul rate is given by

$$\mathsf{R}^{\mathsf{pop}}_{\mathsf{noPIR}}=\gamma_0\sum_{i=1}^{M}p_i+\sum_{i=M+1}^{F}p_i.\tag{12}$$

V. Backhaul Rate Analysis: PIR Case

In this section, we derive the backhaul rate for the case of PIR (i.e., when the user wishes to download content privately) and we prove that uniform content placement (under the PIR protocol in Section III with GRS codes) is optimal. The backhaul rate is given in the following proposition.

Proposition 2. *The backhaul rate for the caching scheme $\mathcal{C}^{\boldsymbol{\mu}}_{\mathsf{MDS}}$ in Section II (with GRS codes) for the PIR case is*

$$\begin{aligned}\mathsf{R}_{\mathsf{PIR}}=&\frac{\mu_{\mathsf{max}}}{\mu_{\min}(n-T+1)-1}\sum_{i=1}^{F}p_i\lceil\mu_i\rceil\sum_{b=1}^{n}\gamma_b(n-b)\\&+\gamma_0\sum_{i=1}^{F}p_i\lceil\mu_i\rceil+\sum_{i=1}^{F}p_i\lfloor 1-\mu_i\rfloor.\end{aligned}\tag{13}$$

Proof: For a file that is cached, when the user is not in communication range of any SBS (i.e., $b=0$), the user can trivially download the file from the MBS since by assumption it can be trusted. Averaging over $\boldsymbol{p}$ (for the files cached) and normalizing by the file size βL we get the second term in (13). On the other hand, if the user is in communication range of $b>0$ SBSs, to download file $\boldsymbol{X}^{(i)}$, it generates n query matrices. From the $b>0$ SBSs in communication range it receives b responses (one from each SBS). The responses to the remaining $n-b$ query matrices need to be downloaded

from the MBS. Since each response consists of d subresponses of size $L\mu_{\mathsf{max}}$ bits, the user downloads $(n-b)dL\mu_{\mathsf{max}}$ bits from the MBS. Averaging over $\boldsymbol{\gamma}$ and $\boldsymbol{p}$ (for the files cached) and normalizing by the file size βL, the contribution to the backhaul rate of the retrieval of files that are cached in the SBSs is

$$\frac{1}{\beta}\sum_{i=1}^{F} p_i \lceil \mu_i \rceil \sum_{b=1}^{n} \gamma_b (n-b) d\mu_{\mathsf{max}}. \tag{14}$$

Now, using the fact that $\beta = \Gamma = n - (k_{\mathsf{max}} + T - 1) = \frac{\mu_{\mathsf{min}}(n-T+1)-1}{\mu_{\mathsf{min}}}$ and $d = k_{\mathsf{max}} = 1/\mu_{\mathsf{min}}$ (see Theorem 1), we can rewrite (14) as

$$\frac{\mu_{\mathsf{max}}}{\mu_{\min}(n-T+1)-1} \sum_{i=1}^{F} p_i \lceil \mu_i \rceil \sum_{b=1}^{n} \gamma_b (n-b). \tag{15}$$

Finally, the files that are not cached are retrieved completely from the MBS, and their contribution to the backhaul rate is (as for the no PIR case)

$$\sum_{i=1}^{F} p_i \lfloor 1 - \mu_i \rfloor. \tag{16}$$

Combining (15) and (16) completes the proof. ∎

A. Optimal Content Placement

Let $\mathsf{R}^*_{\mathsf{PIR}}$ be the minimum backhaul rate resulting from the optimization of the content placement. $\mathsf{R}^*_{\mathsf{PIR}}$ can be obtained by solving the following optimization problem,

$$\begin{aligned}\mathsf{R}^*_{\mathsf{PIR}} = \min_{\substack{\mu_i \in \mathcal{M} \\ n \in \mathcal{A}}} \; & \frac{\mu_{\mathsf{max}}}{\mu_{\min}(n-T+1)-1} \sum_{i=1}^{F} p_i \lceil \mu_i \rceil \sum_{b=1}^{n} \gamma_b (n-b) \\ & + \gamma_0 \sum_{i=1}^{F} p_i \lceil \mu_i \rceil + \sum_{i=1}^{F} p_i \lfloor 1 - \mu_i \rfloor \qquad (17) \\ \text{s.t.} \; & \sum_{i=1}^{F} \mu_i \leq M \text{ and } k_{\mathsf{min}} \mid k_i,\end{aligned}$$

where $\mathcal{A} = \{1/\mu_{\mathsf{min}} + T, \ldots, N_{\mathsf{SBS}}\}$ and the minimum value that n can take on, i.e., $1/\mu_{\mathsf{min}} + T$, comes from the fact that $\mu_{\min}(n-T+1)-1$ has to be positive.

Note that the minimization of $\mathsf{R}_{\mathsf{PIR}}$ requires the optimization over the number of queries n sent to the SBSs and the MBS. The reason is the following. The PIR protocol retrieves $\Gamma = n - (k_{\mathsf{max}} + T - 1)$ desired symbols from each set of n subqueries. This reflects in the factor $\mu_{\mathsf{min}}(n-T+1)-1$ in the denominator of the fraction of (17). From this, increasing n decreases the backhaul rate. However, $n-b$ queries are also sent to the MBS, which contributes to the term $n-b$ in (17). In this respect, n should be small. As a result we need to optimize the value of n such that the backhaul rate is minimized.

Lemma 3. *Uniform content allocation, i.e., $\mu_i = \mu$ for all files that are cached, is optimal. Furthermore, the optimal number of files to cache is the maximum possible, i.e., $\mu_i = \mu$ for $i \leq \min(M/\mu, F)$.*

Proof: We first prove the first part of the lemma. We need to show that either the optimal solution to the optimization problem in (17) is the all-zero vector $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_F) = (0, \ldots, 0)$, or there exists a nonzero optimal solution $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_F)$ for which $\mu_{\mathsf{max}} = \mu_{\mathsf{min}}$. Consider the second case, and let $\boldsymbol{\mu}$ denote any nonzero feasible solution to (17), i.e., a nonzero solution that satisfies the cache size constraint. Furthermore, let $\boldsymbol{\mu}' = (\mu'_1, \ldots, \mu'_F)$ denote the length-F vector obtained from $\boldsymbol{\mu}$ as $\mu'_i = \mu_{\mathsf{min}}$ for $\mu_i \neq 0$ and $\mu'_i = 0$ otherwise. Clearly, $\boldsymbol{\mu}'$ satisfies the cache size constraint as well. Note that $\mu'_{\mathsf{max}} = \mu'_{\mathsf{min}} = \mu_{\mathsf{min}}$. Thus,

$$\begin{aligned}\frac{\mu'_{\mathsf{max}}}{\mu'_{\mathsf{min}}(n-T+1)-1} &= \frac{\mu_{\mathsf{min}}}{\mu_{\mathsf{min}}(n-T+1)-1} \\ &\leq \frac{\mu_{\mathsf{max}}}{\mu_{\mathsf{min}}(n-T+1)-1}.\end{aligned}$$

Furthermore, since both the double summation in the first term of the objective function in (17) and the second and third terms in (17) only depend on the support of $\boldsymbol{\mu}$, it follows that the value of the objective function for $\boldsymbol{\mu}'$ is smaller than or equal to the value of the objective function for $\boldsymbol{\mu}$. Thus, for any nonzero feasible solution $\boldsymbol{\mu}$ there exists another at least as good nonzero feasible solution $\boldsymbol{\mu}'$ for which all nonzero entries are the same (i.e., $\mu'_{\mathsf{min}} = \mu'_{\mathsf{max}} = \mu$), and the result follows by applying the above procedure to a (nonzero) optimal solution to (17).

We now prove the second part of the lemma. Caching a file helps in reducing the backhaul rate if

$$\frac{\mu}{\mu(n-T+1)-1} \sum_{b=1}^{n} \gamma_b (n-b) < 1, \tag{18}$$

for some $n \in \mathcal{A}$ and $\mu \in \mathcal{M}$. This is independent of the file index i. Thus, if the optimal solution is to cache at least one file ($\boldsymbol{\mu} \neq \mathbf{0}$), (18) is met for some $n \in \mathcal{A}$ and caching other files (as many files as permitted up to the cache size constraint, with decreasing order of popularity) is optimal as it further reduces the backhaul rate. ∎

Following Lemma 3, the optimization problem in (17) can be rewritten as

$$\begin{aligned}\mathsf{R}^*_{\mathsf{PIR}} = \min_{\substack{\mu \in \mathcal{M} \\ n \in \mathcal{A}}} \; & \frac{\mu}{\mu(n-T+1)-1} \sum_{i=1}^{\min(M/\mu, F)} p_i \sum_{b=1}^{n} \gamma_b (n-b) \\ & + \gamma_0 \sum_{i=1}^{\min(M/\mu, F)} p_i + \sum_{i=M/\mu+1}^{F} p_i. \qquad (19)\end{aligned}$$

B. Popular Content Placement

For popular content placement, the backhaul rate is given by

$$\begin{aligned}\mathsf{R}^{\mathsf{pop}}_{\mathsf{PIR}} = \min_{n \in \mathcal{A}} \; & \frac{1}{n-T} \sum_{i=1}^{M} p_i \sum_{b=1}^{n} \gamma_b (n-b) \\ & + \gamma_0 \sum_{i=1}^{M} p_i + \sum_{i=M+1}^{F} p_i. \qquad (20)\end{aligned}$$

Note that the optimization over n is still required.

VI. Weighted Communication Rate

So far, we have considered only the backhaul rate. However, it might also be desirable to limit the communication rate from SBSs to the user. We thus consider the weighted communication rate, $\mathsf{C}_{\mathsf{PIR}}$, defined as[4]

$$\mathsf{C}_{\mathsf{PIR}} = \mathsf{R}_{\mathsf{PIR}} + \theta \mathsf{D}_{\mathsf{PIR}},$$

where $\mathsf{D}_{\mathsf{PIR}}$ is the average communication rate (normalized by the file size βL) from the SBSs, and θ is a weighting parameter. We consider $\theta \leq 1$, stemming from the fact that the bottleneck is the backhaul. Note that minimizing the backhaul rate corresponds to $\theta = 0$.

Proposition 3. *The average communication rate from the SBSs for the caching scheme $\mathcal{C}^{\boldsymbol{\mu}}_{\mathsf{MDS}}$ in Section II (with GRS codes) for the PIR case is*

$$\mathsf{D}_{\mathsf{PIR}} = \frac{\mu_{\mathsf{max}}}{\mu_{\min}(n-T+1)-1} \sum_{b=1}^{n} \tilde{\gamma}_b b, \tag{21}$$

where $\tilde{\gamma}_b = \gamma_b$ for $b < n$ and $\tilde{\gamma}_n = \sum_{b=n}^{N_{\mathsf{SBS}}} \gamma_b$.

Proof: To ensure privacy, the user needs to download data from the $b > 0$ SBSs within visibility regardless whether the requested file is cached or not. This is in contrast to the case of no PIR. Note that, if the user queries the SBSs only in the case the requested file is cached, then the spy SBSs would infer that the user is interested in one of the files cached, thus gaining some information about the file requested. In other words, the user sends dummy queries and downloads data that is useless for the retrieval of the file but is necessary to achieve privacy. The user receives b responses from the b SBSs within communication range, each of size $dL\mu_{\mathsf{max}}$ bits. Let $\tilde{\gamma}_b$ denote the probability to receive responses from b SBSs. For $b < n$, $\tilde{\gamma}_b$ is equal to the probability that b SBSs are within communication range, i.e., $\tilde{\gamma}_b = \gamma_b$. On the other hand, the probability to receive responses from n SBSs, $\tilde{\gamma}_n$, is the probability that at least n SBSs are within communication range, i.e., $\tilde{\gamma}_n = \sum_{b=n}^{N_{\mathsf{SBS}}} \gamma_b$. Averaging over $\tilde{\boldsymbol{\gamma}}$ and $\boldsymbol{p}$ (for all files, cached and not cached) and normalizing by the file size βL, the contribution to the communication rate of the retrieval of a file from the SBSs is

$$\frac{1}{\beta} \sum_{i=1}^{F} p_i \sum_{b=1}^{n} \tilde{\gamma}_b b d \mu_{\mathsf{max}}. \tag{22}$$

Now, using the fact that $\beta = \Gamma = n - (k_{\mathsf{max}} + T - 1) = \frac{\mu_{\mathsf{min}}(n-T+1)-1}{\mu_{\mathsf{min}}}$ and $d = k_{\mathsf{max}} = 1/\mu_{\mathsf{min}}$ (see Theorem 1), we can rewrite (22) as (21). ∎

The corresponding optimization problem is

$$\mathsf{C}^*_{\mathsf{PIR}} = \min_{\substack{\mu_i \in \mathcal{M} \\ n \in \mathcal{A}}} \mathsf{R}_{\mathsf{PIR}} + \theta \mathsf{D}_{\mathsf{PIR}} \tag{23}$$
$$\text{s.t.} \sum_{i=1}^{F} \mu_i \leq M \text{ and } k_{\mathsf{min}} \mid k_i,$$

where $\mathsf{R}_{\mathsf{PIR}}$ is given in (13).

[4]For the case of no PIR, a linear scalarization of the MBS and SBS download delays was considered in [5]. The communication rate is directly related to the download delay.

Lemma 4. *Uniform content allocation, i.e., $\mu_i = \mu$ for all files that are cached, is optimal. Furthermore, the optimal number of files to cache is the maximum possible, i.e., $\mu_i = \mu$ for $i \leq \min(M/\mu, F)$.*

Proof: The proof of Lemma 3 applies to both terms in (23) and the result follows. ∎

Following Lemma 4, the optimization problem in (23) can be rewritten as

$$\begin{aligned}\mathsf{C}^*_{\mathsf{PIR}} = \min_{\substack{\mu \in \mathcal{M} \\ n \in \mathcal{A}}} \frac{\mu}{\mu(n-T+1)-1} & \sum_{i=1}^{\min(M/\mu,F)} p_i \sum_{b=1}^{n} \gamma_b (n-b) \\ & + \gamma_0 \sum_{i=1}^{\min(M/\mu,F)} p_i + \sum_{i=M/\mu+1}^{F} p_i \\ & + \theta \frac{\mu}{\mu(n-T+1)-1} \sum_{b=1}^{n} \tilde{\gamma}_b b. \end{aligned} \tag{24}$$

VII. Numerical Results

For the numerical results in this section, we assume that the files popularity distribution $\boldsymbol{p}$ follows the Zipf law [35], i.e., the popularity of file $\boldsymbol{X}^{(i)}$ is

$$p_i = \frac{1/i^{\alpha}}{\sum_{\ell} 1/\ell^{\alpha}},$$

where α, in the range from 0.5 to 1.5, is the skewness factor [7] and by definition $p_1 \geq p_2 \geq \ldots \geq p_F$. In Figs. 3, 4, and 5, we consider a network topology where SBSs are deployed over a macro-cell of radius D meters according to a regular grid with distance d meters between them [5], [7]. Each SBS has a communication radius of r meters. Let $\mathcal{R}_b$ be the area where a user can be served by b SBSs. Then, assuming that the users are uniformly distributed over the macro-cell area with density ϕ users per square meter, the probability that a user is in communication range of b SBSs can be calculated as [7]

$$\gamma_b = \frac{\phi \mathcal{R}_b}{\phi \sum_{a=1}^{N_{\mathsf{max}}} \mathcal{R}_a},$$

where the areas $\mathcal{R}_b$ can be easily obtained by simple geometrical evaluations, and N_{max} is the maximum number of SBSs within communication range of a user.

For the results in Figs. 3, 4, and 5 the system parameters (taken from [7]) are $d = 60$ meters and $D = 500$ meters, which results in $N_{\mathsf{SBS}} = 316$ over the macro-cell area, $F = 200$ files, $\alpha = 0.7$, and $r = 60$ meters. This results in $\boldsymbol{\gamma} = (0, 0, 0.1736, 0.5113, 0.3151, 0, \ldots, 0)$, i.e., the maximum number of SBSs in visibility of a user is $N_{\mathsf{max}} = 4$.

In Fig. 3, we plot the optimized backhaul rate $\mathsf{R}^*_{\mathsf{PIR}}$ (red, solid lines) according to (19) as a function of the cache size constraint M for the noncolluding case ($T = 1$) and $T = 2$ and $T = 3$ colluding SBSs.[5] The curves in Fig. 3 should be interpreted as the minimum backhaul rate that is necessary

[5]Note that a backhaul rate (and a weighted communication rate) of 1 can be trivially obtained by downloading the file directly from the MBS since by assumption it can be trusted. Thus, in all figures, if the PIR protocol gives a backhaul/weighted communication rate larger than 1, we plot 1.

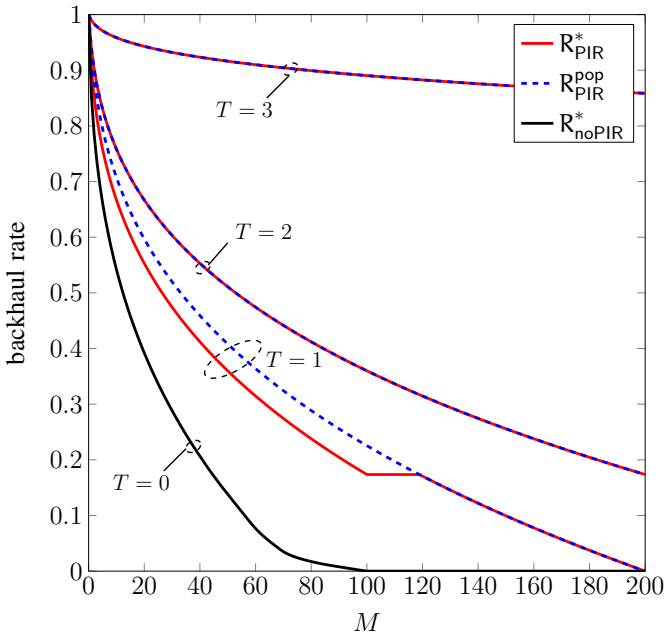

Fig. 3. Backhaul rate as a function of the cache size constraint M for a system with $F = 200$ files, $N_{\mathsf{SBS}} = 316$, and $\alpha = 0.7$.

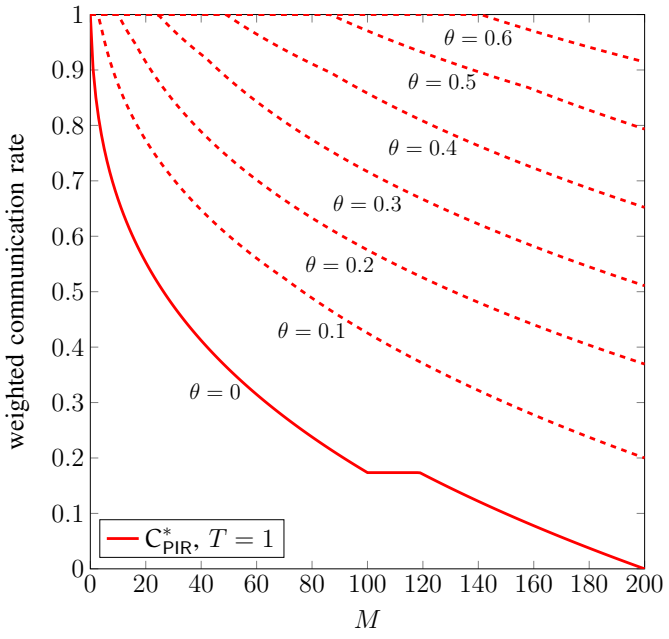

Fig. 4. Optimized weighted communication rate as a function of the cache size constraint M for a system with $T = 1$ spy SBS, $F = 200$ files, $N_{\mathsf{SBS}} = 316$, $\alpha = 0.7$, and several values of θ.

in order to achieve privacy against T spy SBSs out of the n SBSs that are contacted by the user. For the particular system parameters considered, the optimal value of n is 3 for $T = 1$ and $T = 2$, and all values of M, i.e., the scheme yields privacy against T spy SBSs out of the $n = 3$ SBSs contacted. For $T = 3$ the optimal value of n is 4 for all values of M, and thus the scheme yields privacy against 3 spy SBSs out of $n = 4$ SBSs contacted. We also plot the optimized backhaul rate $\mathsf{R}^*_{\mathsf{noPIR}}$ for the case of no PIR.[6] As can be seen in the figure, caching helps in significantly reducing the backhaul rate for $T = 1$ and $T = 2$. For $T = 3$ caching also helps in reducing the backhaul rate, but the reduction is smaller. Also, as expected, compared to the case of no PIR ($\mathsf{R}^*_{\mathsf{noPIR}}$, black, solid line) achieving privacy requires a higher backhaul rate. The required backhaul rate increases with the number of colluding SBSs T.

For $M \geq 100$ and no PIR, the backhaul rate is zero, as all files can be downloaded from the SBSs. Indeed, for $M = 100$, we can select $k_i = 2$ for all i and cache one coded symbol from each stripe of each file in each SBS (thus satisfying the constraint $\sum_{i=1}^{F} \mu_i \leq M$ as $\sum_{i=1}^{200} \mu_i = \sum_{i=1}^{200} 1/k_i = \sum_{i=1}^{200} 0.5 = 100$). Since for no PIR to retrieve each stripe of a file it is enough to download 2 symbols from each stripe of the file (due to the MDS property) and according to $\boldsymbol{\gamma}$ at least 2 SBSs are within range, for $M = 100$ (and hence for $M > 100$ as well) the user can always retrieve the file from the SBSs and the backhaul rate is zero. For the case of PIR and $T = 1$, on the other hand, the required backhaul rate is positive unless all complete files can be cached in all SBSs, i.e., $M = F$. For $T = 2$ and $T = 3$, even for $M = F$ the backhaul rate is not zero. This is because in this case the user needs to receive $n = 3$ and $n = 4$ responses $\boldsymbol{r}^{(l)}$, $l = 1, \ldots, n$, respectively (from the SBSs and/or the MBS). However, for the considered system parameters the probability that the user has $b \geq 3$ SBSs within range is not one, thus the user always needs to download data from the MBS to recover the file and the backhaul rate is positive.

For comparison purposes, in the figure we also plot the backhaul rate for the case of popular content placement $\mathsf{R}^{\mathsf{pop}}_{\mathsf{PIR}}$ in (20) (blue, dashed lines). In this case, the optimal value of n is 2, 3, and 4 for $T = 1$, $T = 2$, and $T = 3$, respectively. We remark that the curve $\mathsf{R}^{\mathsf{pop}}_{\mathsf{PIR}}$ for $T = 1$ overlaps with the curve $\mathsf{R}^{\mathsf{pop}}_{\mathsf{noPIR}}$. This is due to the fact that for $T = 1$, $n = 2$, and $\gamma_0 = \gamma_1 = 0$, $\mathsf{R}^{\mathsf{pop}}_{\mathsf{PIR}}$ in (20) boils down to $\sum_{M+1}^{F} p_i$, which is $\mathsf{R}^{\mathsf{pop}}_{\mathsf{noPIR}}$ in (12). However, for the general case, i.e., other $\boldsymbol{\gamma}$, $\mathsf{R}^{\mathsf{pop}}_{\mathsf{PIR}}$ and $\mathsf{R}^{\mathsf{pop}}_{\mathsf{noPIR}}$ may differ. As already shown in [7], for no PIR the optimized content placement yields significantly lower backhaul rate than popular content placement. For the PIR case and $T = 1$, up to $M = 118$ the optimized content placement also yields some performance gains with respect to popular content placement, albeit not as significant as for the case of no PIR. Interestingly, as shown in the figure, for $M \geq 119$, PIR popular content placement is optimal. Furthermore, as shown in the figure, for $T = 2$ and $T = 3$ popular content placement is optimal for all M.

In Fig. 4, we plot the optimized weighted communication rate $\mathsf{C}^*_{\mathsf{PIR}}$ in (24) for the noncolluding case ($T = 1$) as a function of the cache size constraint M and several values of θ. For the considered system parameters, caching is still useful for small values of θ if the cache size is big enough. For example, for $\theta = 0.5$ caching helps in reducing the weighted communication rate with respect to no caching for $M \geq 87$. For $\theta \geq 0.7$, caching does not bring any reduction of the

[6]The curve $\mathsf{R}^*_{\mathsf{noPIR}}$ in the figure is identical to that in [7, Fig. 4]. As proved in Lemma 2, while the proposed content placement is different from the one in [7], they are equivalent in terms of backhaul rate.

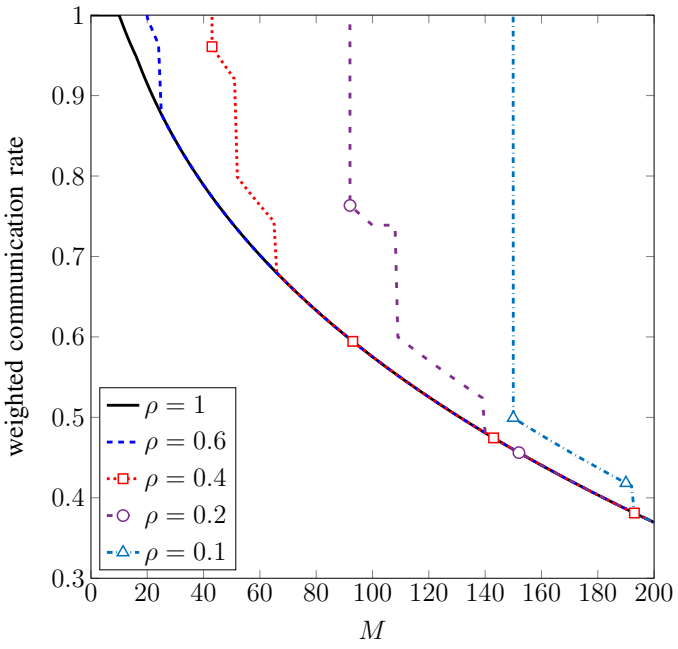

Fig. 5. Optimized weighted communication rate as a function of the cache size constraint M for different values of ρ, i.e., the maximum allowed backhaul rate, $\theta = 0.2$, $F = 200$, $N_{\mathsf{SBS}} = 316$, $\alpha = 0.7$, and $T = 1$.

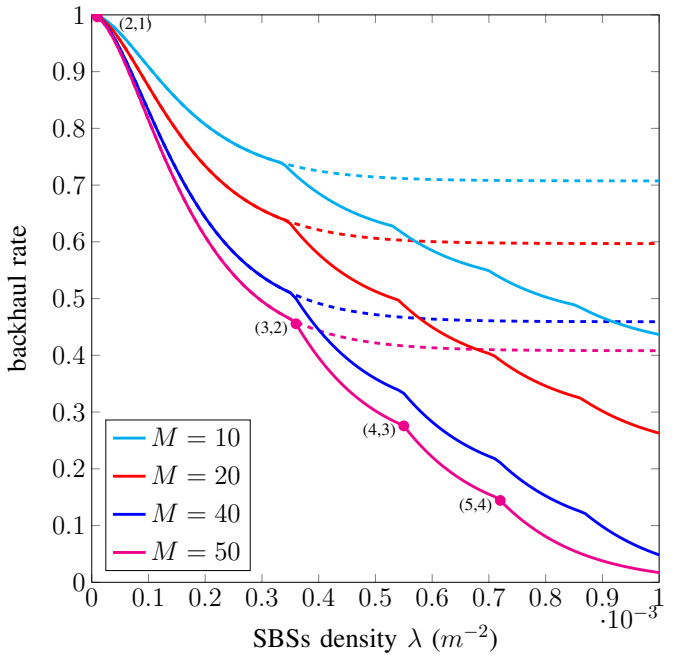

Fig. 6. Backhaul rate as a function of the density of SBSs λ and several values M for the scenario where SBSs are distributed according to a PPP and $T = 1$. $F = 200$ files and $\alpha = 0.7$. Solid lines correspond to optimal content placement ($\mathsf{R}^{*}_{\mathsf{PIR}}$ in (19)) and dashed lines to popular content placement ($\mathsf{R}^{\mathsf{pop}}_{\mathsf{PIR}}$ in (20)).

weighted communication rate.

In Fig. 5, we plot the optimized weighted communication rate $\mathsf{C}^{*}_{\mathsf{PIR}}$ for the noncolluding case ($T = 1$), $\theta = 0.2$, and the scenario where we set a limit on the backhaul rate. In other words, the weighted communication rate $\mathsf{C}_{\mathsf{PIR}}$ is minimized under the condition that the backhaul rate $\mathsf{R}_{\mathsf{PIR}}$ satisfies $\mathsf{R}_{\mathsf{PIR}} \leq \rho$, with $\rho \leq 1$. In the figure, we plot $\mathsf{C}^{*}_{\mathsf{PIR}}$ for different values of ρ. Note that $\rho = 1$ corresponds to the case where the backhaul rate is not constrained, and hence the weighted communication rate is the same as that in Fig. 4. For a given $\rho < 1$, the curves go down for a given value of the cache size constraint M. For smaller cache size constraint than this value, there is no feasible solution, i.e., PIR cannot be achieved with a backhaul rate smaller than ρ. This is expected, since for small values of M (fewer files stored in the SBSs) it is to be expected that more data needs to be downloaded from the MBS, and limiting the backhaul rate to ρ prevents that. For large enough M, a feasible solution exists, but it is worse than that of the case where the backhaul rate is not constrained ($\rho = 1$) up to a given value of M, where the curves eventually merge with those of $\rho = 1$.

In Figs. 6 and 7, we plot the backhaul rate for a PPP deployment model where SBSs are distributed over the plane according to a PPP and a user at an arbitrary location in the plane can connect to all SBSs that are within radius r_{u}. Let λ be the density of SBSs per square meter. For this scenario, the probability that a user is in communication range of b SBSs is given by [36]

$$\gamma_b = \mathrm{e}^{-\psi} \frac{\psi^b}{b!},$$

where $\psi = \lambda \pi r_{\mathsf{u}}^2$. In Fig. 6, we plot the optimized backhaul rate ($\mathsf{R}^{*}_{\mathsf{PIR}}$ in (19), solid lines) as a function of the density λ

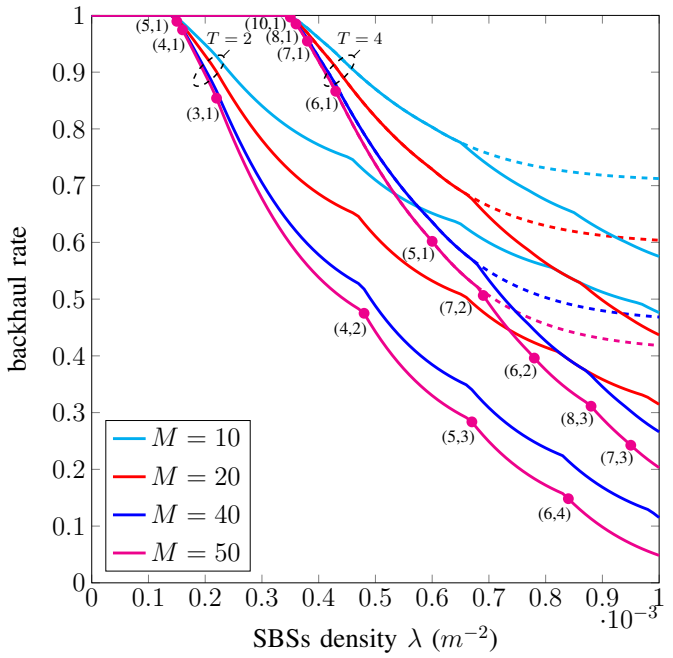

Fig. 7. Backhaul rate as a function of the density of SBSs λ and several values of M for the scenario where SBSs are distributed according to a PPP and $T = 2$ and $T = 4$. $F = 200$ files and $\alpha = 0.7$. Solid lines correspond to optimal content placement ($\mathsf{R}^{*}_{\mathsf{PIR}}$ in (19)) and dashed lines to popular content placement ($\mathsf{R}^{\mathsf{pop}}_{\mathsf{PIR}}$ in (20)).

for $F = 200$ files, $\alpha = 0.7$, $r_{\mathsf{u}} = 60$ meters, different cache size constraint M, and a single spy SBS, i.e., $T = 1$. As expected, the required backhaul rate diminishes by increasing the density of SBSs. For comparison purposes, we also plot

the backhaul rate for popular content placement ($\mathsf{R}_{\mathsf{PIR}}^{\mathsf{pop}}$ in (20), dashed lines). Interestingly, popular content placement is optimal up to a given density of SBSs, after which optimizing the content placement brings a significant reduction of the required backhaul rate. Similar results are observed for $T = 2$ and $T = 4$ colluding SBSs in Fig. 7 with the same system parameters as in Fig. 6. However, for $T = 2$ and $T = 4$ caching does not help in reducing the backhaul rate for small densities. In Figs. 6 and 7, for each M the optimal value of n and μ depends on the density of SBSs. Typically, a pair (n, μ) is optimal for a range of densities. In the figures, we give the optimal values of n and k for $M = 50$ (in particular we give the pair (n, k), with $k = 1/\mu$, which is also the code parameters of the punctured code $\mathcal{C}'$). For convenience, in the figures we only give the parameters for the densities where the optimal pair (n, k) changes. The values should be read as follows: In Fig. 7, walking the curve for $M = 50$ and $T = 2$ from top-left to bottom-right, no caching is optimal for densities up to $\lambda = 1.4 \cdot 10^{-4}$. For $\lambda = 1.5 \cdot 10^{-4}$, $(5, 1)$ is optimal. Then, $(4, 1)$ is optimal for densities $\lambda = 1.6 \cdot 10^{-4}$ to $\lambda = 2.1 \cdot 10^{-4}$. From $\lambda = 2.2 \cdot 10^{-4}$ to $\lambda = 4.7 \cdot 10^{-4}$ the optimal value is $(3, 1)$, and so on (the curves are plotted with steps of 10^{-5}).

VIII. Conclusion

We proposed a PIR scheme that allows to download files of different popularities from a cellular network, where to reduce the backhaul usage content is cached at the wireless edge in SBSs, while achieving privacy against a number of spy SBSs. We derived the backhaul rate for this scheme and formulated the content placement optimization. We showed that, as for the no PIR case, up to a number of spy SBSs caching helps in reducing the backhaul rate. Interestingly, contrary to the no PIR case, uniform content placement is optimal. Furthermore, popular content placement is optimal for some scenarios. Although uniform content placement is optimal, the proposed PIR scheme for multiple code rates may be useful in other scenarios, e.g., for distributed storage where data is stored using codes of different rates.

Appendix
Proof of Theorem 1

To prove that the protocol achieves PIR against T colluding SBSs, we need to prove that both the privacy condition in (2a) and the recovery condition in (2b) are satisfied. We first prove that the recovery condition in (2b) is satisfied.

According to Lemma 1, GRS codes with a fixed weighting vector $\boldsymbol{v}$ and evaluation vector $\boldsymbol{\kappa}$ are naturally nested. Furthermore, puncturing a GRS code results in another GRS code, since GRS codes are weighted evaluation codes. Indeed puncturing implies removing a coordinate of the original GRS code. Each coordinate of a GRS code is a weighted polynomial evaluation. Thus, after puncturing, each coordinate is still a weighted evaluation of the same original polynomial, hence the punctured code is still a GRS code [34, Ch. 5]. Thus, $\mathcal{C}'_i \subseteq \mathcal{C}'_{\mathsf{max}}$ for all i, and it follows from (7) that

$$\tilde{\mathcal{C}} = \left(\sum_{i=1}^{F} \mathcal{C}'_i \right) \circ \bar{\mathcal{C}} = \mathcal{C}'_{\mathsf{max}} \circ \bar{\mathcal{C}}.$$

Furthermore, it can easily be shown that the Hadamard product of two GRS codes with the same evaluation vector $(\kappa_1, \ldots, \kappa_n)$ is also a GRS code with dimension equal to the sum of the dimensions minus 1. Thus, $\tilde{\mathcal{C}}$ is a GRS code of dimension $k_{\mathsf{max}}+T-1$. As $\tilde{\mathcal{C}}$ is an $(n, k_{\mathsf{max}}+T-1)$ MDS code (GRS codes are MDS codes), it can correct arbitrary erasure patterns of up to $\Gamma = n-(k_{\mathsf{max}}+T-1)$ erasures. This implies that one can construct a valid $k_{\mathsf{max}} \times n$ ($d = k_{\mathsf{max}}$) matrix $\hat{\boldsymbol{E}}$ (satisfying conditions C1–C3) from $\beta = \Gamma$ information sets $\{\mathcal{I}_m\}$ of $\mathcal{C}'_{\mathsf{max}}$ as shown below.

Let $\mathcal{J}_j = \{j, \ldots, (j + \Gamma - 1) \bmod n\}$, $j = 1, \ldots, k_{\mathsf{max}}$. Construct $\hat{\boldsymbol{E}}$ in such a way that $\mathcal{J}_j$ is the support of the j-th row of $\hat{\boldsymbol{E}}$. Hence, C1 is satisfied. Furthermore, since $\tilde{\mathcal{C}}$ is an $(n, k_{\mathsf{max}} + T - 1)$ MDS code and $\Gamma = n - (k_{\mathsf{max}} + T - 1)$, all rows of $\hat{\boldsymbol{E}}$ are correctable by $\tilde{\mathcal{C}}$, and thus C2 is satisfied. Finally, run Algorithm 1, which constructs $\beta = \Gamma$ information sets $\{\mathcal{I}_m\}$ of $\mathcal{C}'_{\mathsf{max}}$ (and the corresponding sets $\{\mathcal{F}_l\}$) such that C3 is satisfied. Note that since $\mathcal{C}'_{\mathsf{max}}$ is an MDS code, all coordinate sets of size k_{max} are information sets of $\mathcal{C}'_{\mathsf{max}}$, and hence Algorithm 1 will always succeed in constructing a valid set of information sets of $\mathcal{C}'_{\mathsf{max}}$ (the inequalities in Lines 6 and 7 together with the fact that the overall weight of $\hat{\boldsymbol{E}}$ is Γk_{max} ensure that $\beta = \Gamma$ valid information sets for $\mathcal{C}'_{\mathsf{max}}$ are constructed). In particular, the while-loop in Line 6 will always terminate. For clarity purposes, in the following, we present an example of the construction of the β information sets.

Example 1. *Consider that $\mathcal{C}_{\mathsf{max}}$ is an $[n = 7, k_{\mathsf{max}} = 4]$ GRS code and $T = 2$. With $\beta = \Gamma = n - (k_{\mathsf{max}} + T - 1) = 2$,*

$$\hat{\boldsymbol{E}} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix},$$

which has column weight profile $w_{\mathsf{H}}(\boldsymbol{t}_1) = w_{\mathsf{H}}(\boldsymbol{t}_5) = 1$, $w_{\mathsf{H}}(\boldsymbol{t}_2) = w_{\mathsf{H}}(\boldsymbol{t}_3) = w_{\mathsf{H}}(\boldsymbol{t}_4) = 2$, and $w_{\mathsf{H}}(\boldsymbol{t}_6) = w_{\mathsf{H}}(\boldsymbol{t}_7) = 0$. Given these weights, one can construct the information sets by running Algorithm 1. Below we show the steps of the algorithm.

1) ***Iteration 1:*** $l = 1$.
   - $\mathcal{F}_1 = \{1\}$ *and* $\mathcal{I}_1 = \{1\}$.
2) ***Iteration 2:*** $l = 2$.
   - $\mathcal{F}_2 = \{1\}$ *and* $\mathcal{I}_1 = \{1, 2\}$.
   - $\mathcal{F}_2 = \{1, 2\}$ *and* $\mathcal{I}_2 = \{2\}$.
3) ***Iteration 3:*** $l = 3$.
   - $\mathcal{F}_3 = \{1\}$ *and* $\mathcal{I}_1 = \{1, 2, 3\}$.
   - $\mathcal{F}_3 = \{1, 2\}$ *and* $\mathcal{I}_2 = \{2, 3\}$.
4) ***Iteration 4:*** $l = 4$.
   - $\mathcal{F}_4 = \{1\}$ *and* $\mathcal{I}_1 = \{1, 2, 3, 4\}$.
   - $\mathcal{F}_4 = \{1, 2\}$ *and* $\mathcal{I}_2 = \{2, 3, 4\}$.

Algorithm 1: Construction of $\{\mathcal{I}_m\}$ for Theorem 1

Input : $\hat{\boldsymbol{E}}$, β, n, k_{max}
Output: $\{\mathcal{I}_m\}$, $\{\mathcal{F}_l\}$
for $m \in \{1, \ldots, \beta\}$ **do**
  $\mathcal{I}_m \leftarrow \emptyset$
end
for $l \in \{1, \ldots, n\}$ **do**
  $\mathcal{F}_l \leftarrow \emptyset$, $m \leftarrow 1$
  while $|\mathcal{F}_l| < w_{\mathsf{H}}(\boldsymbol{t}_l)$ **do**
    if $|\mathcal{I}_m| < k_{\mathsf{max}}$ **then**
      $\mathcal{F}_l \leftarrow \mathcal{F}_l \cup \{m\}$
      $\mathcal{I}_m \leftarrow \mathcal{I}_m \cup \{l\}$
    end
    $m \leftarrow m + 1$
  end
end

5) ***Iteration 5:*** $l = 5$.
   - $\mathcal{F}_5 = \{2\}$ *and* $\mathcal{I}_2 = \{2, 3, 4, 5\}$. *This is because* $|\mathcal{I}_{m=1}| = k_{\mathsf{max}} = 4$, *and the algorithm skips Steps 7-10 for* $m = 1$.
6) *Since* $w_{\mathsf{H}}(\boldsymbol{t}_6) = w_{\mathsf{H}}(\boldsymbol{t}_7) = 0$, *the algorithm terminates by printing* $\mathcal{I}_1 = \{1, 2, 3, 4\}$ *and* $\mathcal{I}_2 = \{2, 3, 4, 5\}$.

From the constructed matrix $\hat{\boldsymbol{E}}$, the user is able to recover $\Gamma d \geq \beta k_i$ unique code symbols of the requested file $\boldsymbol{X}^{(i)}$, at least k_i symbols from each stripe. Furthermore, a set of k_i recovered code symbols from each stripe corresponds to an information set of $\mathcal{C}'_i$ (any subset of size k_i of any information set of size k_{max} of $\mathcal{C}'_{\mathsf{max}}$ is an information set of $\mathcal{C}'_i$), and the requested file $\boldsymbol{X}^{(i)}$ can be recovered. This can be seen following a similar argument as in the proof of [21, Th. 6], and it follows that the recovery condition in (2b) is satisfied.

Secondly, we consider the privacy condition in (2a). A reasoning similar to the proof of [21, Lem. 6] shows that it is satisfied, and we refer the interested reader to this proof for further details. The fundamental reason is that addition of a deterministic vector in (4) does not change the joint probability distribution of $\{\boldsymbol{Q}^{(l)} : l \in \mathcal{T}\}$ for any set $\mathcal{T}$ of size T, and the proof follows the same lines as the proof of [20, Th. 8]. However, note that there is a subtle difference in the sense that independent instances of the protocol may query different sets of SBSs. However, since the set of SBSs that are queried is independent of the requested file and depends only on which SBSs that are within communication range, this fact does not leak any additional information on which file is requested by the user.

REFERENCES

[1] U. Niesen, D. Shah, and G. W. Wornell, "Caching in wireless networks," *IEEE Trans. Inf. Theory*, vol. 58, no. 10, pp. 6524–6540, Oct. 2012.

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

[3] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5G be?" *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[4] D. Liu, B. Chen, C. Yang, and A. F. Molisch, "Caching at the wireless edge: Design aspects, challenges, and future directions," *IEEE Commun. Mag.*, vol. 54, no. 9, pp. 22–28, Sep. 2016.

[5] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Trans. Inf. Theory*, vol. 59, no. 12, pp. 8402–8413, Dec. 2013.

[6] E. Baştuğ, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5G wireless networks," *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 82–89, Aug. 2014.

[7] V. Bioglio, F. Gabry, and I. Land, "Optimizing MDS codes for caching at the edge," in *Proc. Global Commun. Conf. (GLOBECOM)*, San Diego, CA, Dec. 2015.

[8] 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.

[9] N. Golrezaei, P. Mansourifard, A. F. Molisch, and A. G. Dimakis, "Base-station assisted device-to-device communications for high-throughput wireless video networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 7, pp. 3665–3676, Jul. 2014.

[10] J. Pedersen, A. Graell i Amat, I. Andriyanova, and F. Brännström, "Distributed storage in mobile wireless networks with device-to-device communication," *IEEE Trans. Commun.*, vol. 64, no. 11, pp. 4862–4878, Nov. 2016.

[11] A. Piemontese and A. Graell i Amat, "MDS-coded distributed storage for low delay wireless content delivery," in *Proc. 9th Int. Symp. Turbo Codes & Iterative Inf. Process. (ISTC)*, Brest, France, Sep. 2016, pp. 320–324.

[12] J. Pedersen, A. Graell i Amat, I. Andriyanova, and F. Brännström, "Optimizing MDS coded caching in wireless networks with device-to-device communication," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 286–295, Jan. 2019.

[13] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Batch codes and their applications," in *Proc. 36th Annual ACM Symp. Theory Comput. (STOC)*, Chicago, IL, Jun. 2004, pp. 262–271.

[14] N. B. Shah, K. V. Rashmi, and K. Ramchandran, "One extra bit of download ensures perfectly private information retrieval," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, HI, Jun./Jul. 2014, pp. 856–860.

[15] T. H. Chan, S.-W. Ho, and H. Yamamoto, "Private information retrieval for coded storage," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Hong Kong, China, Jun. 2015, pp. 2842–2846.

[16] R. Tajeddine, O. W. Gnilke, and S. El Rouayheb, "Private information retrieval from MDS coded data in distributed storage systems," *IEEE Trans. Inf. Theory*, vol. 64, no. 11, pp. 7081–7093, Nov. 2018.

[17] S. Kumar, E. Rosnes, and A. Graell i Amat, "Private information retrieval in distributed storage systems using an arbitrary linear code," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 1421–1425.

[18] H. Sun and S. A. Jafar, "Private information retrieval from MDS coded data with colluding servers: Settling a conjecture by Freij-Hollanti *et al*." *IEEE Trans. Inf. Theory*, vol. 64, no. 2, pp. 1000–1022, Feb. 2018.

[19] ——, "The capacity of private information retrieval," *IEEE Trans. Inf. Theory*, vol. 63, no. 7, pp. 4075–4088, Jul. 2017.

[20] R. Freij-Hollanti, O. W. Gnilke, C. Hollanti, and D. A. Karpuk, "Private information retrieval from coded databases with colluding servers," *SIAM J. Appl. Algebra Geom.*, vol. 1, no. 1, pp. 647–664, Nov. 2017.

[21] S. Kumar, H.-Y. Lin, E. Rosnes, and A. Graell i Amat, "Achieving maximum distance separable private information retrieval capacity with linear codes," *IEEE Trans. Inf. Theory*, to appear.

[22] K. Banawan and S. Ulukus, "The capacity of private information retrieval from coded databases," *IEEE Trans. Inf. Theory*, vol. 64, no. 3, pp. 1945–1956, Mar. 2018.

[23] H. Sun and S. A. Jafar, "The capacity of robust private information retrieval with colluding databases," *IEEE Trans. Inf. Theory*, vol. 64, no. 4, pp. 2361–2370, Apr. 2018.

[24] B. Chor, O. Goldreich, E. Kushilevitz, and M. Sudan, "Private information retrieval," in *Proc. 36th Annual IEEE Symp. Found. Comp. Sci. (FOCS)*, Milwaukee, WI, Oct. 1995, pp. 41–50.

[25] S. Kadhe, B. Garcia, A. Heidarzadeh, S. El Rouayheb, and A. Sprintson, "Private information retrieval with side information: the single server case," in *Proc. 55th Annual Allerton Conf. Commun., Control, Comput.*, Monticello, IL, Oct. 2017, pp. 1099–1106.

[26] Y.-P. Wei, K. Banawan, and S. Ulukus, "Fundamental limits of cache-aided private information retrieval with unknown and uncoded prefetching," *IEEE Trans. Inf. Theory*, to appear.

[27] Z. Chen, Z. Wang, and S. Jafar, "The capacity of private information retrieval with private side information," Sep. 2017, arXiv:1709.03022v1 [cs.IT]. [Online]. Available: https://arxiv.org/abs/1709.03022

[28] Y.-P. Wei, K. Banawan, and S. Ulukus, "Private information retrieval with partially known private side information," in *Proc. 52nd Annual Conf. Inf. Sci. Sys. (CISS)*, Princeton, NJ, Mar. 2018.

[29] R. Tandon, M. Abdul-Wahid, F. Almoualem, and D. Kumar, "PIR from storage constrained databases – coded caching *meets* PIR," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, May 2018.

[30] M. A. Attia, D. Kumar, and R. Tandon, "The capacity of uncoded storage constrained PIR," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Vail, CO, Jun. 2018, pp. 1959–1963.

[31] Y.-P. Wei, K. Banawan, and S. Ulukus, "Cache-aided private information retrieval with partially known uncoded prefetching: Fundamental limits," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1126–1139, Jun. 2018.

[32] S. Li and M. Gastpar, "Converse for multi-server single-message PIR with side information," Sep. 2018, arXiv:1809.09861v1 [cs.IT]. [Online]. Available: https://arxiv.org/abs/1809.09861

[33] Y.-P. Wei, B. Arasli, K. Banawan, and S. Ulukus, "The capacity of private information retrieval from decentralized uncoded caching databases," Nov. 2018, arXiv:1811.11160v1 [cs.IT]. [Online]. Available: https://arxiv.org/abs/1811.11160

[34] W. C. Huffman and V. Pless, Eds., *Fundamentals of Error-Correcting Codes*. Cambridge, UK: Cambridge University Press, 2010.

[35] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: Evidence and implications," in *Proc. IEEE Joint Conf. Comput. Commun. Soc. (INFOCOM)*, New York, NY, Mar. 1999, pp. 126–134.

[36] B. Serbetci and J. Goseling, "On optimal geographical caching in heterogeneous cellular networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, San Francisco, CA, Mar. 2017.