Incremental coding for extractable compression in the context of Massive Random Access

Thomas Maugey[1], Aline Roumy[1], Elsa Dupraz[2], Michel Kieffer[3]

***Abstract*—In this paper, we study the problem of source coding with Massive Random Access (MRA). A set of correlated sources is encoded once for all and stored on a server while a large number of clients accesses various subsets of these sources. Due to the number of concurrent requests, the server is only able to extract a bitstream from the stored data: no re-encoding can be performed before the transmission of the data requested by the clients.**

First, we formally define the MRA framework and propose to model the constraints on the way subsets of sources may be accessed by a navigation graph. We introduce both storage and transmission costs to characterize the performance of MRA. We then propose an *Incremental coding Based Extractable Compression* (IBEC) scheme. We first show that this scheme is optimal in terms of achievable storage and transmission costs. Second, we propose a practical implementation of our IBEC scheme based on rate-compatible LDPC codes. Experimental results show that our IBEC scheme can almost reach the same transmission costs as in traditional point-to-point source coding schemes, while having a reasonable overhead in terms of storage cost.

***Index Terms*—Data compression, Source coding, Random Access, Channel coding.**

I. Introduction

In source coding for Massive Random Access (MRA), the data generated by several correlated sources are compressed, stored on a server and later *partly* requested by *many* clients. We propose to model the potential client's requests by a navigation graph that represents the application constraints. For instance, the sources may only be requested in a sequential order. In this setup, the large data volume and the high number of clients prevent re-encoding at the server. This implies that the whole data must be coded *before* clients can make requests and that only simple data extraction operations are allowed *after* receiving the requests. We consider the MRA problem from a pure source coding perspective, which differs from problems of massive multiple access to some communication resources [1].

Examples of MRA applications are free-viewpoint television (FTV) [2], access to data collected by a network of sensors [3] or a crowd-sensing platform [4]. In these applications, the items of the database (*e.g.*, the frames of a video, or the frames of the different views in multiview video, or the signals captured by different sensors) show strong correlations with each other. This suggests to jointly compress the sources [5, Th. 2.6.6] to exploit the statistical dependencies, which is unfortunately not compatible with the requirement of having random access. Tile-based approaches [6], [7] divide the sources into subsets, called tiles. Sources within a tile are jointly compressed, such that the correlation between sources can be partly exploited while, at the same time, enabling an access to a subset of sources. However, the random access granularity remains at the tile level, which implies that a lot of unrequested sources are usually transmitted.

Instead, several solutions have been proposed when random access is a key constraint. For instance, in video compression, picture-wise access is necessary, either for editing [8] or for interactive communication [9]. In the schemes of [8], [9], all frames are coded independently (All intra mode). When higher compression performance is targeted (for broadcasting or streaming applications), the sequence is encoded jointly and an access frame (*intra* coded, *i.e.,* independently of the others) is introduced to enter the decoding of the sequence roughly every second [10]. To allow to resynchronize faster than every second, some frames are encoded and stored twice (once as intra, and once as a complement to the previously coded frames) [11]. Multiple storage of the same frame has also been proposed in the context of interactive communication of multiview videos in [12], [13], [14]. This solution provides an increased accessibility to the video with almost the same transmission rate, but at the expense of an increased storage cost. A tradeoff between storage and transmission costs has been proposed in the context of multiview videos [15][16], where a frame is no more encoded several times. Instead it is stored and sent in a single worst case version (called a Merge frame). Similar techniques have been proposed to allow random access to compressed sensor data. For instance, separated compression of blocks of data, similar to the All Intra video compression, has been proposed for volumetric data [17] and digital power meter [18]. The idea to encode some data points as a reference, as in [11][19], has been proposed for Internet of Things databases [20] and for the genomes [21]. However, these methods generally come with a significant transmission or storage rate penalty.

The goal of this paper is to propose a novel coding architecture that is competitive in terms of storage and transmission costs while considering the application constraints incorporated into the navigation graph. To do so, we first formally introduce the MRA problem in Sec. II, as a multi-source coding problem where clients are allowed to access successively to some of the sources. Then, we propose to model the possible successive requests by a *navigation graph*, which structure is imposed by application constraints. In the

[1] are with Inria Rennes Bretagne-Atlantique, France, e-mail: {thomas.maugey,aline.roumy}@inria.fr

[2] is with IMT Atlantique, Lab-STICC, UBL, France France, e-mail: elsa.dupraz@imt-atlantique.fr

[3] is with L2S, UMR CNRS 8506, CentraleSupelec, Univ. Paris-Sud, France, e-mail: michel.kieffer@l2s.centralesupelec.fr

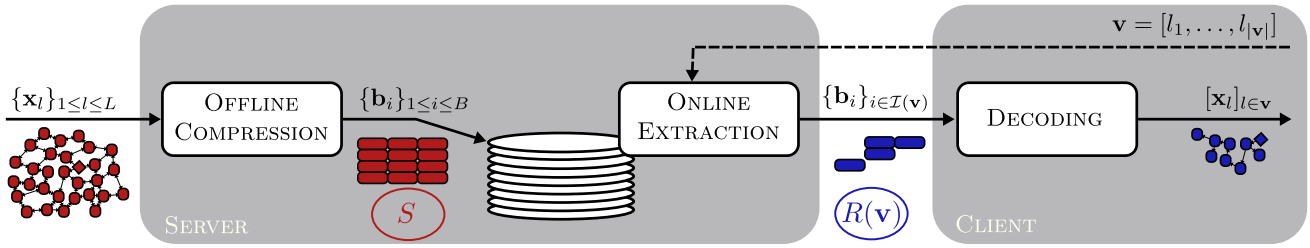


Fig. 1. The Massive Random Access system: L signals are jointly compressed and stored on a server, a client requests a sequel of source indices $\mathbf{v}$, the server extracts part of the stored bitstream and sends it to the client for decoding.

navigation graph, a directed edge exists from node j to node ℓ, if it is possible, for a client, to request source X_ℓ after source X_j. Hence, when coding a source X_ℓ, its neighboring sources in the navigation graph might be used to achieve lower compression cost, because one of them is present at the client's side, *i.e.,* obtained from previous requests. However, exploiting the correlation is a difficult task in the MRA context, because, when coding a given source X_ℓ, the encoder does not know which sources will be available at the decoder.

Then, we propose to evaluate the compression efficiency of MRA schemes in terms of two costs: the storage cost and the transmission cost, the latter being request-dependent. In Sec. III, we use these two costs to evaluate the coding performance of state-of-the-art methods, when these methods must satisfy the constraints defined by the navigation graph.

In Sec. IV, we propose a novel *Incremental coding Based Extractable Compression (IBEC)* scheme. For each source X_ℓ, incremental coding builds a bitstream that enables to decode X_ℓ based on any possible source potentially available at the decoder. More precisely, the stored bitstream is composed of different parts that are potentially transmitted depending on the level of correlation between X_ℓ and on the previously requested sources present at the client's side. For a storage cost that is as small as the most performing existing scheme, we show that the transmission cost achieved by the proposed IBEC scheme is the same as the cost that is obtained in the absence of random access. In Sec. IV-B, a practical implementation of the IBEC scheme is proposed based on rate-adaptive Low-Density Parity-Check (LDPC) codes.

Finally, Sec. V describes experimental comparisons of the proposed IBEC scheme with conventional coding architectures.

II. Massive Random Access (MRA)

The source coding problem with MRA is shown in Fig. 1. Consider a database containing the signals $\mathbf{x}_\ell$ generated by L correlated sources denoted $\{X_\ell\}_{1\leq\ell\leq L}$. Access to this database is made randomly, according to the client's choice, but also according to some restrictions imposed by the application. In Sec. II-A, we propose to model these restrictions with a *navigation graph*. The database is first compressed offline into a set of separable bitstreams denoted $\{\mathbf{b}_i\}_{1\leq i\leq B}$. Then, a client requests some of the signals $\mathbf{x}_\ell$ in the form of a vector $\mathbf{v}$ of source indices. Here, the signal $\mathbf{x}_\ell$ corresponds to the smallest entity that can be accessed, such that when the client requests the source of index ℓ, information about all

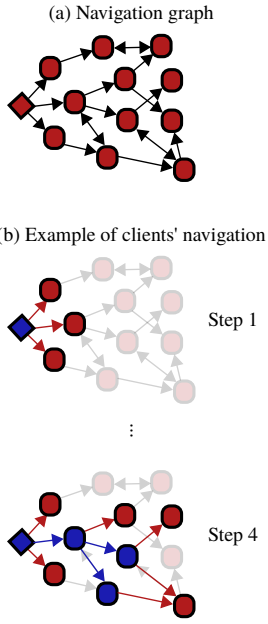


Fig. 2. A navigation graph and an example of client's navigation. (a) The nodes of the graph depicted by a circle represent the sources. The square node represents a dummy source used for the initialization of the navigation. A directed edge exists between node i and j if the source X_j can be requested once the source X_i has been previously requested and stored in the clients' memory. (b) In the example of navigation, the blue nodes correspond to the requested sources, the red ones to the sources that are allowed to be requested, and the faded red ones represent sources that can not be requested.

the samples of this source are sent. Note that a real source of information might be split into several sources, to offer more flexibility in the access to the database. Upon reception of a request, the server extracts and sends a subset of the stored bitstreams. The received bitsteams allow the client to reconstruct the requested sources.

In Sec. II-B, we formally define two criteria to characterize the coding performance of a solution to the MRA problem, namely the storage and rate costs S and R. For ease of presentation, we consider in this work a lossless compression scheme. A lossy extension may be obtained, *e.g.,* with a quantization of the input sources [22].

A. Modeling Random Access by means of a navigation graph

Access to a database is usually proposed with some restrictions. For instance, in FTV, the client observes a scene by navigating from one viewpoint to another. But, to offer a smooth client experience, the navigation might be limited to

neighboring viewpoints only. In the sensor network application, a client might be allowed to only request signals that have been acquired in some spatio-temporal windows.

Before showing how to integrate these restrictions, we first model a request as a vector of source indices. Indeed, in FTV, a frame of one view is modeled as the data of a source, and the navigation of the client is therefore equivalent to a request of ordered source indices. In the case of sensor networks, a source corresponds to the data of one sensor in a temporal window. Again, the request of a spatio-temporal window can be seen as a sequence of source indices, up to an ordering of the requested sources. This ordering can either be predefined (*e.g.*, navigation to the east and the south only) or optimized. In the latter case, the ordering needs to be sent with the data.

Now, to describe the set of allowed requests to the database that may be performed by a client, we introduce the oriented navigation graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. $\mathcal{N}$ is a set of $L+1$ nodes and $\mathcal{E}$ is a set of directed edges between these nodes. The nodes represent the L sources plus a (dummy) source X_0 used to initiate the navigation. A directed edge $e_{j,i}$ from node j to node i indicates that the source X_i can be accessed by a client, once X_j has been previously requested and stored in the clients' memory. The only source that may be directly accessed, without having previously requested another source, is the (dummy) source X_0 which corresponds to node 0. To summarize, the graph $\mathcal{G}$ introduces the constraints on the way sources may be accessed. The set of all possible requests consistent with $\mathcal{G}$ is denoted by $\mathcal{V}$.

B. Rate and storage costs

To reduce as much as possible the amount of data required to serve a client's request, a wise coding scheme must exploit the correlation between the requested sources. Based on the navigation constraints introduced above, we know that for a given requested source of index i, at least one of its neighbors in the graph $\mathcal{G}$ has been already accessed and can serve as a helper for decoding. We can thus model the request $\mathbf{v} = [v_1, v_2, \ldots]$ as a tree whose root is the node 0, and the edge indicates that the preceding source helps the decoding of the successive source (see Fig. 2). In general, there are several ways to extract and decode the data for a given request. For example, when v_j has two parents v_{i_1} and v_{i_2}, both previously requested, the server can either extract the compressed bitstream of the source X_{v_j} with either $X_{v_{i_1}}$ or $X_{v_{i_2}}$ taken as a reference. To avoid any ambiguity, the neighbor used for the extraction is sent by the client together with the request. In the rest of the paper, $\pi_{\mathbf{v}}(v_i)$ denotes the parent of the source of index v_i used in the extraction.

As described in Fig. 1, the aim of the offline compression is to compute a set of bitstreams representing the data generated by the L sources. Let $\mathcal{B} = \{\mathbf{b}_i\}_{1 \le i \le B}$ be the resulting set of B independently extractable bitstreams. A subset of necessary bitstreams will be transmitted as a response to each client's request $\mathbf{v}$. Several compression algorithms exist that satisfy these constraints (see Sec. III for a detailed presentation). To compare these schemes, we first introduce the *storage cost*, which represents the size of the bitstreams stored on the server:

$$S = \frac{1}{L} \sum_{i=1}^{B} |\mathbf{b}_i|, \tag{1}$$

where $|.|$ denotes the sub-stream size expressed in number of bits, and where the normalization factor L is in order to have a cost per source.

The second criterion used in order to compare compression schemes is related to the efficiency of serving requests. Indeed, at the other side of the chain, the client sends its request $\mathbf{v} = [v_1, \ldots, v_{|\mathbf{v}|}]$ to the server. The server then extracts from $\mathcal{B}$, the bitstreams $\{\mathbf{b}_i\}_{i \in \mathcal{I}(\mathbf{v})}$ necessary to recover the signals requested in $\mathbf{v}$. The set $\mathcal{I}(\mathbf{v})$ contains the indices of the $\mathbf{b}_i$ that are necessary to recover the sources whose index is in $\mathbf{v}$. The *per request transmission cost* is defined as the cumulated size of the bitstreams sent to a client for a given request:

$$R(\mathbf{v}) = \frac{1}{|\mathbf{v}|} \sum_{i \in \mathcal{I}(\mathbf{v})} |\mathbf{b}_i|, \tag{2}$$

where the normalization leads to a per source criterion. Finally, to obtain a criterion that does not depend on the client's request, we assume that a probability distribution p over the clients' requests is available. This leads to the expected *transmission cost*:

$$R = \mathbb{E}_{\mathbf{v}}(R(\mathbf{v})) = \sum_{\mathbf{v} \in \mathcal{V}} p(\mathbf{v}) R(\mathbf{v}). \tag{3}$$

III. Conventional architecture limitations

Building a coding scheme that solves the MRA problem consists in finding an off-line encoding function able to generate the bitstreams $\{\mathbf{b}_i\}_{1 \le i \le B}$ associated to the source realizations $\{\mathbf{x}_\ell\}_{1 \le \ell \le L}$, and an online extraction function, able to select sub-streams $\{\mathbf{b}_i\}_{i \in \mathcal{I}(\mathbf{v})}$ that satisfy any request $\mathbf{v} \in \mathcal{V}$.

This section provides the costs achieved by conventional coding and extraction schemes in the context of MRA. For that purpose, we introduce the following notations $\forall i \in [1, L], \forall j \in [0, L]$:

$$h_{i|j} = H(X_i^{n_i} | X_j^{n_j}) \text{ and } h_i = H(X_i^{n_i}), \tag{4}$$

where n_i and n_j stand for the signal length of the sources X_i and X_j respectively, $X_i^{n_i}$ denotes the random vector $(X_{i,1}, X_{i,2}, \ldots, X_{i,n_i})$, and $H(X)$ and $H(X|Y)$ denote the entropy and the conditional entropy respectively. We also recall that $\forall\ i,\ H(X_i|X_0) = H(X_i)$, since X_0 is a dummy source used for initialization only.

A first solution to compress the database while allowing various access to subsets of sources is to code each source *independently*. In this case, there is one $\mathbf{b}_i$ per source, and $B = L$. This scheme is called *All Intra (AI)*. It has been proposed to facilitate either picture-wise editing of compressed videos [8], or picture-wise access in the context of interactive video communication [9]. The interest of this method resides in the fact that it is totally flexible, in the sense that it can straightforwardly satisfy any client request. The drawback of this solution is that the correlation between the sources is not

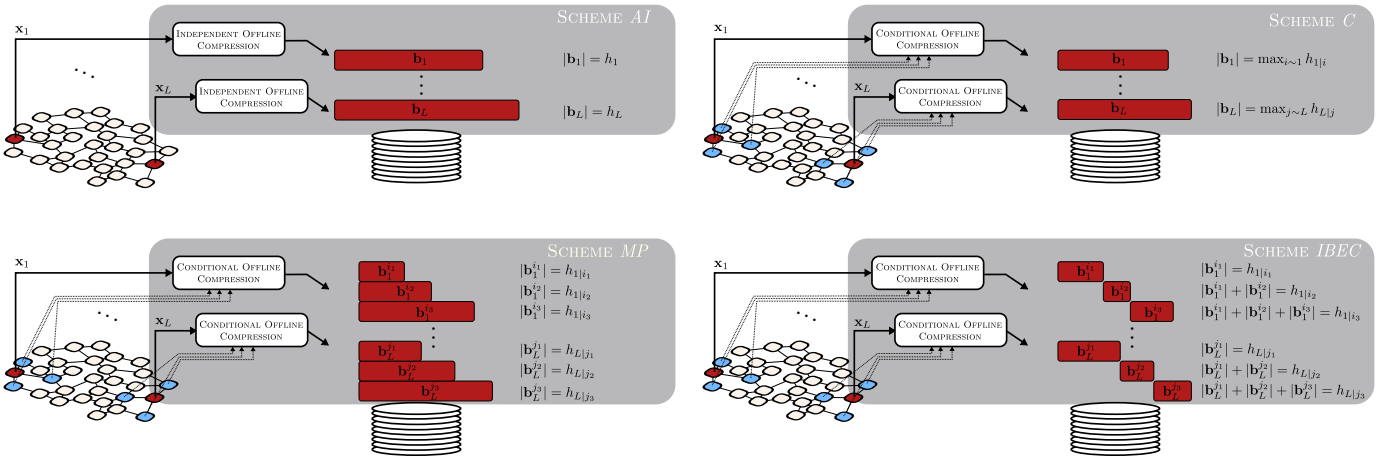


Fig. 3. Different architectures for individual offline compression.

exploited. More precisely, the storage cost achieved by the *AI* scheme is:

$$S_{\mathrm{AI}} = \frac{1}{L}\sum_{i=1}^{L} h_i \tag{5}$$

and the transmission cost is:

$$R_{\mathrm{AI}} = \sum_{\mathbf{v}\in\mathcal{V}} p(\mathbf{v}) \frac{1}{|\mathbf{v}|} \sum_{i\in\mathbf{v}} h_i. \tag{6}$$

Therefore, the *AI* scheme achieves low compression performance, both in terms of storage and transmission, especially when the sources are strongly correlated.

Alternatively, one can exploit the correlation between a source and the previously requested ones *i.e.,* the parents in the navigation graph $\mathcal{G}$, at the coding stage. This is for example the case of the predictive coding scheme: if $\mathbf{x}_j$ is in memory when decoding a vector $\mathbf{x}_i$ (because it has been already requested), the predictive encoder first builds a prediction $\hat{\mathbf{x}}_{i|j}$ of the signal $\mathbf{x}_i$, computes the residue $\mathbf{x}_i - \hat{\mathbf{x}}_{i|j}$ and stores the coded residue as a bitstream $\mathbf{b}_i^j$. If the prediction is good, the compression performance of such a scheme is high. However, the computed residue depends explicitly on the reference source used to build the prediction (X_j in the previous example), and does not allow random access. Thus, to use predictive coding in the MRA context, one needs to store on the server, for each source X_i, one residue per possible parent j of node i in the graph $\mathcal{G}$. This scheme is thus called the *Multiple Prediction (MP)* scheme [11], [12], [13], [14], and the storage cost is:

$$S_{\mathrm{MP}} = \frac{1}{L}\sum_{i=1}^{L} \sum_{j:e_{j,i}\in\mathcal{E}} h_{i|j}. \tag{7}$$

When serving a request $\mathbf{v}$, only the useful residues are sent to the client. The transmission cost is thus:

$$R_{\mathrm{MP}} = \sum_{\mathbf{v}\in\mathcal{V}} p(\mathbf{v}) \frac{1}{|\mathbf{v}|} \sum_{i\in\mathbf{v}} h_{i|\pi_{\mathbf{v}}(i)}. \tag{8}$$

Since $h_{i|j} \leq h_i$, the transmission cost is reduced with respect to the *AI* scheme, and the cost reduction increases with the correlation of the requested sources. On the other hand, the storage cost increases significantly with the averaged degree of the graph, *i.e.,* with the flexibility offered to the client to navigate within the database.

In the *Compound (C)* scheme proposed in [15], [23], for each source $\mathbf{x}_i$, all the possible predictions $\hat{\mathbf{x}}_{i|j}$ are built (with $e_{j,i} \in \mathcal{E}$). Then, considering $\hat{\mathbf{x}}_{i|j}$ as noisy versions of $\mathbf{x}_i$, a bitstream $\mathbf{b}_i$ able to correct the errors in all the predictions is generated (for example based on channel coders). In order to tackle all the predictions possibly generated by the clients, the stored bitstream corresponds to the worst one, *i.e.,* to the largest $h_{i|j}$. The storage cost is thus:

$$S_{\mathrm{C}} = \frac{1}{L}\sum_{i=1}^{L} \max_{e_{j,i}\in\mathcal{E}} h_{i|j}. \tag{9}$$

Similarly to the *AI* scheme, the extraction is simple since there is only one bitstream per source. The transmission cost is thus:

$$R_{\mathrm{C}} = \sum_{\mathbf{v}\in\mathcal{V}} p(\mathbf{v}) \frac{1}{|\mathbf{v}|} \sum_{i\in\mathbf{v}} \max_{e_{j,i}\in\mathcal{E}} h_{i|j}. \tag{10}$$

The *C* scheme is thus a good way of achieving reasonable transmission cost (between R_{AI} and R_{MP}) while having a smaller storage cost than those of *MP* and *AI* schemes. However, while it is reasonable to consider that one needs to anticipate the worst scenario at the server side, this is unfortunate to transmit this bitstream in all the cases, even if the prediction is of a better quality.

The storage and transmission costs are summarized in Fig. 4. In the next section, we introduce the proposed coder that enables to reach the transmission cost of the *MP* scheme and storage cost of the *C* scheme.

IV. Incremental coding based extractable compression (IBEC)

A. Proposed coding scheme

In [24], [25], we have shown that when compressing one single source with several potential side informations available at the decoder, it is possible to use incremental coding in order to only send the necessary amount of bits. Based on the graph-based client's navigation formalism and the transmission/storage costs proposed in Sec. II, we now generalize this

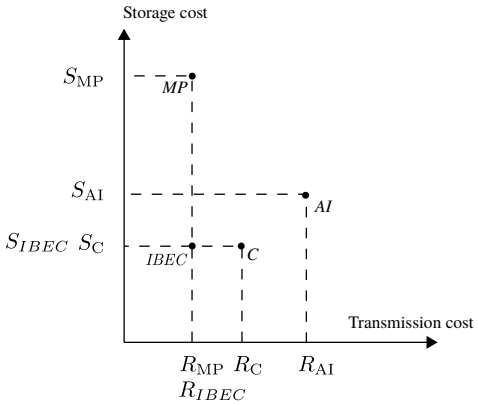


Fig. 4. Storage and rate transmission costs summary. Our proposed *IBEC* scheme, obtains the best theoretical storage and rate performance.

result to the multi-source scenario. We call this new coding scheme *Incremental coding Based Extractable Compression (IBEC)*.

For each signal $\mathbf{x}_i$ to be compressed, we first identify the parents of the source of index i in the navigation graph $\mathcal{G}$. These neighbors are used to build potential predictions $\hat{\mathbf{x}}_{i|j}$. They are then sorted from the best to the worst (*i.e.,* from the smallest $h_{i|j}$ to the largest). Then, we build a first bitstream $\mathbf{b}_i^{j_1}$ able to decode the best prediction assuming that $\mathbf{x}_{j_1}$ is already decoded at the receiver. For the second bitstream, the coding scheme is able to use $\mathbf{b}_i^{j_1}$ plus an additional bitstream $\mathbf{b}_i^{j_2}$ of size $h_{i|j_2} - h_{i|j_1}$. This incremental construction is applied in the same way to all predictions. As a result, the stored bitstream, for each source, has the same size than the *C* scheme, *i.e.,* the one corresponding to the highest $h_{i|j}$, but is split into several sub-streams so that only the necessary information can be extracted. All schemes are illustrated in Fig. 3.

The global performance of the *IBEC* scheme is:

$$S_{\mathrm{IBEC}} = \frac{1}{L}\sum_{i=1}^{L} \max_{e_{j,i}\in\mathcal{E}} h_{i|j}. \tag{11}$$

and

$$R_{\mathrm{IBEC}} = \sum_{\mathbf{v}\in\mathcal{V}} p(\mathbf{v}) \frac{1}{|\mathbf{v}|} \sum_{i\in\mathbf{v}} h_{i|\pi_{\mathbf{v}}(i)} \tag{12}$$

As it can be seen from Fig. 4, based on the costs, derived in Sec. III and IV, the proposed *IBEC* scheme obtains the smallest storage and transmission costs of all the conventional architectures, *i.e.*,

$$S_{\mathrm{IBEC}} = S_{\mathrm{C}} \leq S_{\mathrm{AI}} \leq S_{\mathrm{MP}} \tag{13}$$

and

$$R_{\mathrm{IBEC}} = R_{\mathrm{MP}} \leq R_{\mathrm{C}} \leq R_{\mathrm{AI}}. \tag{14}$$

In the next section, we explain how the incremental coding is built in practice.

B. Practical incremental coder

Incremental coding principle: The source coding problem with one source and one side information at the decoder can be solved in practice by channel codes [26], [27], [28]. Similarly, to solve the MRA problem, we propose to construct a coding scheme based on channel codes. However, the channel code needs to tolerate variable rate to adapt to all the potential side informations. In practice, rate adaptation is achieved by choosing a rate among a finite set of predefined source coding rates: $R \in \{\frac{1}{M}, \ldots, \frac{m}{M}, \ldots, \frac{M}{M}\}$.

Assume that the decoder requests the source X_i, and has previously requested the source X_j, with $j \in \{j_1, ..., j_J\}$, see the navigation graph of Fig. 5(a). Note that the size J of the neighborhood depends on the node i, but for ease of presentation, we remove the dependence with respect to i in the notation J. Let us further assume that the sources X_j, with $j \in \{j_1, ..., j_J\}$ are sorted in increasing order of conditional entropy, *i.e.,* from the most to the least correlated source X_j,

$$h_{i|j_1} \leq h_{i|j_2} \leq \ldots \leq h_{i|j_J}. \tag{15}$$

We now explain how to encode the source vector $\mathbf{x}_i$ into an extractable bitstream $\mathbf{b}_i = (\mathbf{b}_i^{j_1}, \mathbf{b}_i^{j_2}, \ldots, \mathbf{b}_i^{j_J})$.

Data encoding: let us consider that all correlated sources are marginally i.i.d., binary with uniform distribution, and that each source X_i generates a vector of length n. We model the pairwise correlation between the correlated sources by a channel with transition probability $p(\mathbf{x}_i|\mathbf{x}_j)$, see Fig. 5(b). Futher assume that the correlation channel is a binary symmetric channel. We use the rate-adaptive code called Low Density Parity Check Accumulate (LDPCA) Code introduced in [29]. Given a set of predefined target rates$\{\frac{1}{M}, \ldots, \frac{m}{M}, \ldots, \frac{M}{M}\}$ and a source vector length n, the LDPCA construction provides M parity check matrices denoted $(\mathbf{K}_1, \ldots, \mathbf{K}_m, \ldots, \mathbf{K}_M)$, where $\mathbf{K}_m$ is of size $n\frac{m}{M} \times n$ and where

$$\forall \mathbf{x}, \mathbf{K}_1\mathbf{x} \subseteq \mathbf{K}_2\mathbf{x} \subseteq \ldots \subseteq \mathbf{K}_M\mathbf{x} \tag{16}$$

meaning that $\mathbf{K}_1\mathbf{x}$ is a subvector of the vector $\mathbf{K}_2\mathbf{x}$. In our simulations, we considered the *6336_irregDeg2to21* LDPCA code, whose parameters are available at [30].

We now explain how to encode the source vector $\mathbf{x}_i$. First, the so-called accumulated syndromes are computed as

$$\forall m,\ \mathbf{a}_{i,m} = \mathbf{K}_m\mathbf{x}_i, \tag{17}$$

where $\mathbf{a}_{i,m}$ is of length $n\frac{m}{M}$. Then, for each possible side information $\mathbf{x}_j$, and for each accumulated syndrome $\mathbf{a}_{i,m}$, a reconstruction is performed according to the maximum *a posteriori* criterion, *i.e.*, $\forall j \in \{j_1, ..., j_J\}, \forall m \in \{1, \ldots, M\}$

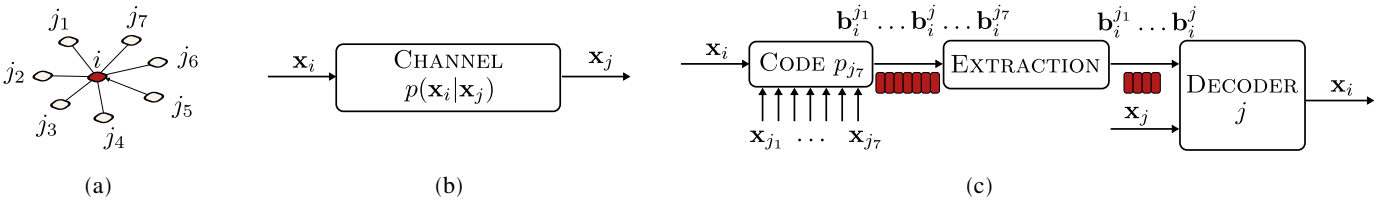
$$\hat{\mathbf{x}}_{i,j,m} = \arg \max_{\mathbf{x}_i : \mathbf{a}_{i,m} = \mathbf{K}_m\mathbf{x}_i} p(\mathbf{x}_i|\mathbf{x}_j). \tag{18}$$

Note that this defines a modified channel decoder, since the search space is the coset of syndrom $\mathbf{a}_{i,m}$ and not the coset of syndrom $\mathbf{0}$, as in classical channel coding. When a LDPC code is used, decoding is performed with the modified belief propagation (BP) algorithm proposed in [31].

Then, for each possible side information $\mathbf{x}_j$, we select the shortest accumulated syndrom $\mathbf{a}_i^j$ such that the BP decoder recovers $\mathbf{x}_i$ perfectly, *i.e.*, $\forall j \in \{j_1, ..., j_J\}, \forall m \in \{1, \ldots, M\}$

$$m^*(j) = \arg\min_m \{|\mathbf{a}_{i,m}| = n\frac{m}{M} \ s.t.\ \hat{\mathbf{x}}_{i,j,m} = \mathbf{x}_i\} \tag{19a}$$

$$\mathbf{a}_i^j = \mathbf{a}_{i,m^*(j)}. \tag{19b}$$



(a) (b) (c)

Fig. 5. (a) Navigation graph: source X_i can be requested after one of the source X_j, with $j \in \{j_1, ..., j_7\}$. (b) Correlation channel: the correlation between the sources X_i and X_j is modeled by a channel with transition probability $p(\mathbf{x}_i|\mathbf{x}_j)$. (c) MRA compression scheme based on channel codes: encoding is performed by first computing the bitstreams $(\mathbf{b}_i^{j_1} \dots \mathbf{b}_i^{j_7})$ needed for the less correlated possible side information X_{j_7}. These bits are stored at the server. Then, upon request of the source with index i, and knowing that the source with index j has been previously requested, a subset of the bitstreams is extracted and used with the side information $\mathbf{x}_j$ to reconstruct the source vector $\mathbf{x}_i$.

From the inclusion property of the accumulated syndromes (16), and from the ordering of the side information vectors (15), the optimal accumulated syndromes satisfy $\mathbf{a}_i^{j_1} \subseteq \mathbf{a}_i^{j_2} \subseteq \cdots \subseteq \mathbf{a}_i^{J}$.

Finally, for the source X_i, the stored sequence of bitstreams $\mathbf{b}_i = (\mathbf{b}_i^{j_1}, \mathbf{b}_i^{j_2}, \dots, \mathbf{b}_i^{j_J})$ is constructed from the $\mathbf{a}_i^j$ as follows. First, $\mathbf{b}_i^{j_1} = \mathbf{a}_i^{j_1}$. Then, the second bitstream $\mathbf{b}_i^{j_2}$ is obtained by retaining the bits in $\mathbf{a}_i^{j_2}$ that are not in $\mathbf{a}_i^{j_1}$, *i.e.,* $\mathbf{b}_i^{j_2} = \mathbf{a}_i^{j_2} \setminus \mathbf{a}_i^{j_1}$. More generally, we have $\mathbf{b}_i^{j_k} = \mathbf{a}_i^{j_k} \setminus \mathbf{a}_i^{j_{k-1}}$.

The resulting storage cost for the source X_i is $S_i = |\mathbf{a}_i^J|/n$, where the overall storage cost is $S = \frac{1}{L}\sum_{i=1}^{L} S_i$.

Data extraction: Upon request of the source X_i, and knowing that the source X_j is available at the decoder, the server extracts from $\mathbf{b}_i$ the subsequence $(\mathbf{b}_i^{j_1}, \mathbf{b}_i^{j_2}, \dots, \mathbf{b}_i^{j}) = \mathbf{a}_i^j$, and sends it to the decoder. This leads to a transmission cost $R_i^j = |\mathbf{a}_i^j|/n$. Then, the transmission cost of a request, $\mathbf{v} = (\ell_1, \dots, \ell_{|\mathbf{v}|})$ is

$$R(\mathbf{v}) = \frac{1}{|\mathbf{v}|} \sum_{i \in \mathbf{v}} R_i^{\pi_{\mathbf{v}}(i)} \tag{20}$$

Data decoding: Upon request of the source X_i, the decoder receives $(\mathbf{b}_i^{j_1}, \mathbf{b}_i^{j_2}, \dots, \mathbf{b}_i^{j}) = \mathbf{a}_i^j$. The decoder then performs BP decoding taking into the previously received side information $\mathbf{x}_j$. From the rate adaptation performed at the encoder (19), the reconstruction is performed without any error.

IBEC vs C scheme: The C scheme shares similarities with the proposed IBEC scheme since, in both cases, a channel code is used to perform data encoding. In the IBEC scheme, the index of the previous request j is used to adapt the transmission and send the complement information only, as shown in (19). In the C scheme, this knowledge is not used. The sent accumulated syndrome is the one that allows perfect reconstruction for any possible side information (21a), in particular for the worst one (21b).

$$m^* = \arg\min_m \{|\mathbf{a}_{i,m}| = n\frac{m}{M} \ s.t. \ \forall j \ \hat{\mathbf{x}}_{i,j,m} = \mathbf{x}_i\} \tag{21a}$$

$$= \arg\min_m \{|\mathbf{a}_{i,m}| = n\frac{m}{M} \ s.t. \ \hat{\mathbf{x}}_{i,j_J,m} = \mathbf{x}_i\} \tag{21b}$$

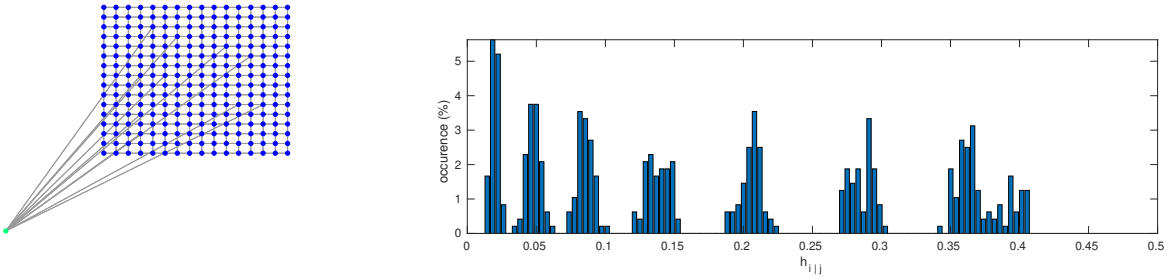
$$\forall j, \ \mathbf{a}_i^j = \mathbf{a}_{i,m^*}. \tag{21c}$$

IBEC and C schemes vs MP: The IBEC and C schemes use channel coding to perform data encoding. By contrast, in the MP scheme all possible residues $\hat{\mathbf{x}}_i - \mathbf{x}_{i|j}, \forall (i,j)$ are encoded with a variable length source code and then stored. Upon request of the source of index i, after having requested the source of index j, only the compressed bitstream of $\hat{\mathbf{x}}_i - \mathbf{x}_{i|j}$ is sent.

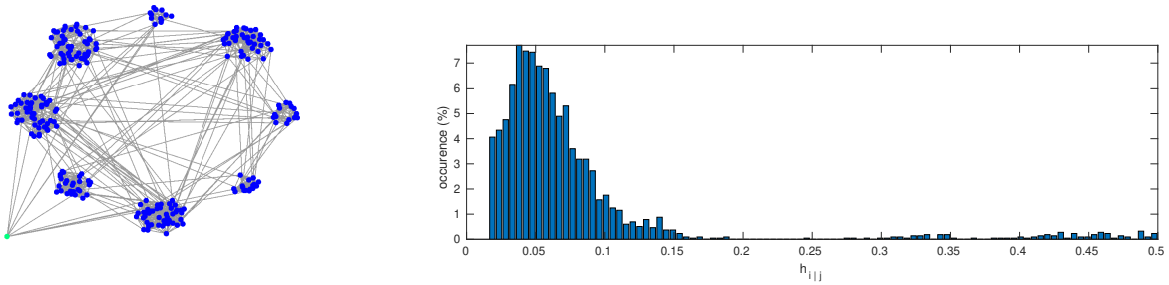
V. PERFORMANCE COMPARISON

Datasets: In order to evaluate the performance of the proposed incremental scheme compared to the baseline methods reviewed in Section III, we generate five different datasets according to Algorithm 1. We pick one graph structure described by an adjacency matrix $\mathbf{A}$ (available at [33]). Depending on the cases, this graph structure is randomly generated or corresponds to a predefined structure. To generate the sources X_ℓ, L independent random vectors $\mathbf{z}_\ell$ of size $n = 12672$ are first generated, forming a matrix $\mathbf{Z}$ of size $L \times n$. Then, in order to generate correlated sources, we iteratively perform a filtering of $\mathbf{Z}$ with the following filter $\mathbf{M} = \mathbf{D}^{-1/2}(\mathbf{A} + \mathbf{I}_L)\mathbf{D}^{-1/2}$, as in the distributed averaging method [32]. After a large number of iterations, the sources will converge to a consensus, where all sources have the same value equal to the data average. Since it is not our goal here, we intentionally perform less iterations in order to bring some correlation between the sources, without reaching complete consensus though. The variable $\mathbf{Z}$ is finally binarized in order to build the data $\mathbf{X}$ to encode.

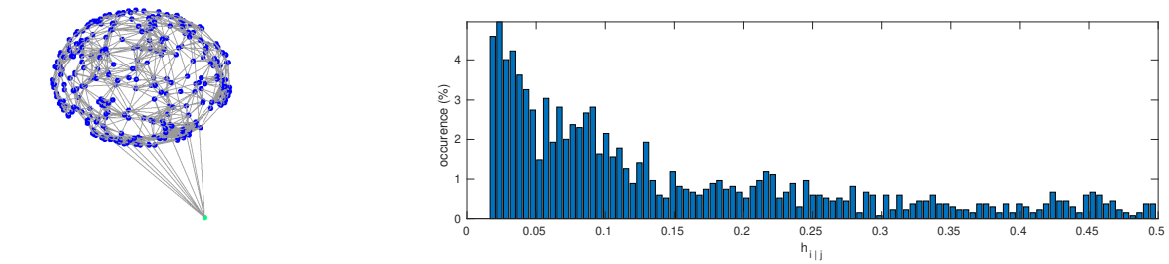
We considered various number of iterations N_{it} to generate sources with different characteristics, which correspond to a variety of scenarios that are encountered when compressing correlated data (*e.g.,* averaged connectivity degree and the heterogeneity of the correlations). They are depicted in Figure 6. The *2D grid* graph represents data acquired by sensors arranged in a regular 2D grid, where each source is connected to its direct neighbors. This graph models a navigation encountered in Free Viewpoint Television, where the navigation is performed among multiview videos. Here one node corresponds to one frame and the sources placed on the same line correspond to the frames of the same view. Moreover, the coding rates of the sources are heterogeneous as classically observed in video compression. The graph *Community Network* has 8 clusters, in which nodes are highly connected and correlated between each other. Some sources are also connected to sources in other clusters. The correlation is lower. The graph *Sphere* is similar to *2D grid*, with heterogeneous levels of correlation. However, the averaged connectivity degree is higher. The graph *Spiral* has a very small averaged connectivity degree. In other words, the randomness of the navigation is reduced. The graph *Torus* has homogeneous level of correlation and averaged connectivity degree.



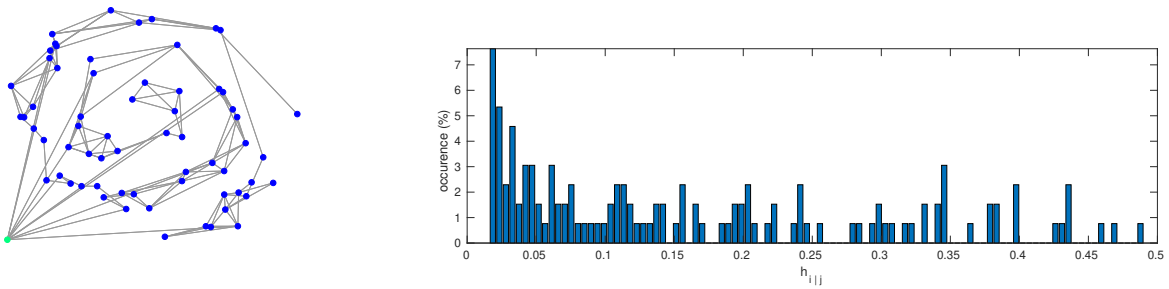
(a) *2D grid*, $L = 256$, $N_{\text{it}} = 80$



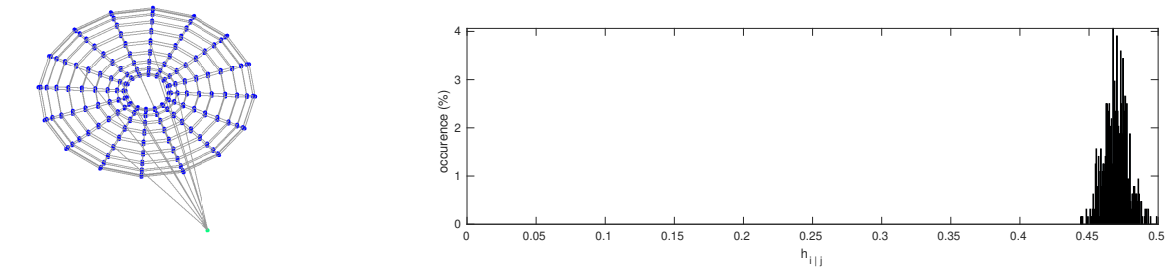
(b) *Community network*, $L = 256$, $N_{\text{it}} = 50$



(c) *Sphere*, $L = 300$, $N_{\text{it}} = 50$



(d) *Spiral*, $L = 64$, $N_{\text{it}} = 40$



(e) *Torus*, $L = 320$, $N_{\text{it}} = 10$

Fig. 6. Different graphs $\mathcal{G}$ used for experiments. The blue nodes are the sources, and the green node corresponds to X_0. The adjacency matrix $\mathbf{A}$ of each graph is available in [33].

Data: n, N_{it}, $\mathbf{A}$
Result: $\mathbf{x}_\ell$, $\mathbf{P}$, $\mathbf{H}$
Initialization:
Generate $\mathbf{Z} = [z_{\ell,i}]_{1\leq\ell\leq L, 1\leq i\leq n}$: $L \times n$ matrix whose rows are L independent random sources with the distribution $\mathcal{N}(0,1)$ of size n;
Build the filter matrix $\mathbf{M} = \mathbf{D}^{-1/2}(\mathbf{A}+\mathbf{I}_L)\mathbf{D}^{-1/2}$, where $\mathbf{D}$ is the degree matrix of $\mathbf{A}+\mathbf{I}_L$;
for $i \in [1, N_{\text{it}}]$ **do**
  $\mathbf{Z} = \mathbf{M}\mathbf{Z}$
end
Build a binary variable:
$\mathbf{X} = (\mathbf{Z} > 0)$;
Estimate the entropy rates $h_{\ell_1|\ell_2}$ between each source and its neighbors in the graph:
for $\ell_1 \in [1, L]$ **do**
  $h_{\ell_1} = 1$;
  for $\ell_2 \in [1, L]$ *s.t.* $e_{\ell_2,\ell_1} \in \mathcal{E}$ **do**
    Estimate the error probability:
    $p_{\ell_1|\ell_2} = p_{err}(\mathbf{x}_{\ell_1}|\mathbf{x}_{\ell_2})$;
    Calculate the entropy:
    $h_{\ell_1|\ell_2} =$
    $-p_{\ell_1|\ell_2}\log_2 p_{\ell_1|\ell_2} + (1-p_{\ell_1|\ell_2})\log_2(1-p_{\ell_1|\ell_2})$;
  end
end

Algorithm 1: Data generation

Client's request generation: In the following, we present storage and transmission costs evaluated on the five graphs generated as explained above. In order to evaluate the transmission cost, that is by nature, client-dependent, one need to simulate different client's requests, evaluate their transmission cost for the different schemes and average them in order to obtain the expected transmission cost in eq. (3). As explained in Section II, a client's request is a tree $\subset \mathcal{G}$. We thus generate random navigation trees of different lengths. For that purpose, we assume that all the transitions between sources are equally probable.

Results: We have implemented the *AI*, *MP*, *C* schemes and our proposed *IBEC*. For *AI*, *MP* schemes the coding of the source or the residue after prediction is done using an arithmetic coder. For *C* scheme, the encoding is done with the *6336_irregDeg2to21* LDPCA code available at [30] (as in (19), among the set of codes, we choose the LDPC code with minimum rate that allows perfect reconstruction of the vector). For each graph, we have encoded the corresponding data $\mathbf{X}$, and simulated 100 client's navigation, recording, each time, the transmission cost. We have also calculated, for each case, the theoretical expected performance, based on the entropy calculation, see eq. (5,6) for *AI*, eq. (7,8) for *MP*, eq. (9,10) for *C* and eq. (11,12) for *IBEC*. The results are shown in Figure. 7. In this figure, the storage cost is on the left while transmission costs are depicted on the right. For each method, the empty bars correspond to the theoretical performances while the filled ones represent the experimental ones.

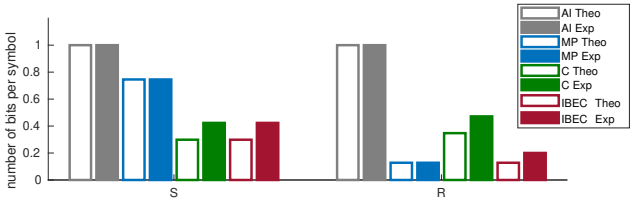
The *IBEC* scheme theoretically achieves the smallest storage (13) and transmission (14) costs. Said differently, our *IBEC* scheme achieves the smallest transmission rate as *MP* scheme while reaching also the smallest storage cost as the *C* scheme, which validates its potential advantage. Let us now discuss the practical performance.

The difference between theoretical and practical costs is small for the *AI* and *MP* schemes since they use an arithmetic coder whose performance is not far from the Shannon bounds. On the contrary, the channel codes used in the *C* scheme and by the incremental coders the *IBEC* scheme have a more significant gap between theory and practice. Nevertheless, despite this disadvantage, the practical performances comparison in Fig. 7 still demonstrate the benefits of our scheme. This is indeed visible from the following observations.

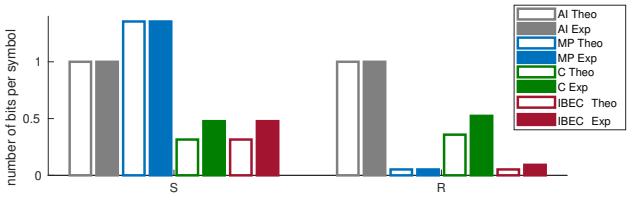
- Compared to the *AI* scheme, both storage and transmission costs are always smaller with the *IBEC* scheme. For the dataset *Torus*, the gap is lower since most $h_{i|j}$ are around 0.5 bits per symbol as shown in Fig. 6 (e), which results in a practical bitrate closer to 1, that is the cost of an intra source coding. In all the other datasets, the correlation between the sources is much higher, and the results demonstrate the ability of the *IBEC* scheme to take into account this correlation at the storage and transmission stages.
- Given the fact that the *MP* scheme reaches the best transmission rate possible (thanks to en extensive storage cost), we observe that our *IBEC* scheme = is really efficient. Indeed, the transmission rate achieved by the *IBEC* scheme is almost the same (or slightly higher) than the *MP* scheme, for a storage cost that is much lower. This is even more visible when the navigation graph $\mathcal{G}$ is highly connected (as the dataset *Community Network*). To be able to reach the best transmission cost, the *MP* scheme has to store many residues, exploding the storage cost, while this storage cost remains small with our scheme.
- Instead of storing any possible navigation transitions, the *C* scheme stores for each source, the *worst one*, as the *IBEC* scheme does. This is the reason why the storage costs of the *C* and *IBEC* schemes are the same (and the minimum ones) in the theoretical and practical performances. However, instead of transmitting the whole codeword for every request as the *C* scheme, our *IBEC* scheme only transmits the necessary subpart. This leads to reduced transmission costs as can be seen in Fig. 7. The difference is not large for the datasets where the correlation between the sources is homogeneous as the *Torus* one (see Fig. 6 (e)). Indeed, the gap between the "worst case" and the general case is small. On the contrary, when the correlation levels are heterogeneous, as in *Community Network* or *2D grid* (see Fig. 6 (a) and (b)), the transmission cost of our method is much smaller than that of *C* scheme.

As a conclusion, in both theoretical and practical aspects, one observes that the *IBEC* scheme reaches or is very close to the best expected transmission costs, with a minimal storage cost.

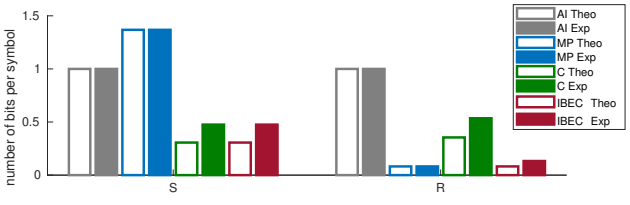
Comparison against tile-based approaches: as stated in the introduction, some methods, namely the tile-based ap-



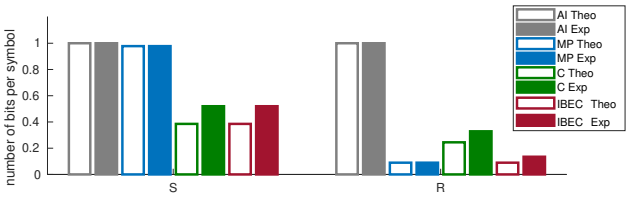
(a) *2D grid*



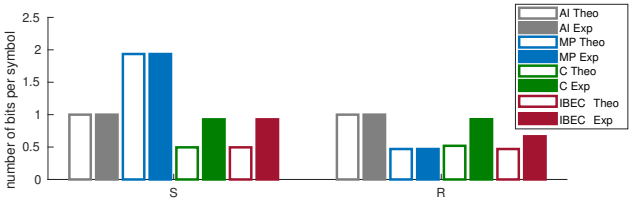
(b) *Community network*



(c) *Sphere*



(d) *Spiral*



(e) *Torus*

Fig. 7. Storage (left) and transmission (right) results for the five different datasets.

proaches, partition the source set in T tiles, encode each tile independently in one single bitstream $\mathbf{b}_i$ (one per tile), and transmit the whole tile information as soon as at least one source requested belongs to the tile. In other words, these tile-based approaches proceed at another granularity of request, *i.e.,* the smallest entity that can be transmitted is a tile. This decreases the storage cost, at the price of an increased transmission cost. We compare our *IBEC* approach against tile-based approaches for the graph *2D grid* (Fig. 6(a)), as it is developed in [6], [7]. Storage and transmission costs are shown in Tab. I. As expected, the storage cost is reduced compared to our *IBEC* method. However, at the same time, the transmission costs explode whatever the size of the tile. It confirms the interest of our *IBEC* approach, since it enables an ideal transmission cost, perfectly aligned with what the client requests, for a reasonable storage overhead.

	Tile-based				IBEC
	$T=1$	$T=4$	$T=16$	$T=64$	
S (bits per symbol)	0.18	0.17	0.22	0.37	0.42
R (bits per symbol)	0.46	0.45	0.49	0.57	0.20

TABLE I
EXPERIMENTAL STORAGE AND TRANSMISSION COSTS OBTAINED WITH OUR IBEC SOLUTION AND THE TILE-BASED APPROACH FOR DIFFERENT T (NUMBERS OF TILES).

VI. CONCLUSION

This paper proposes a formulation of the Massive Random Access problem in which the client's constraints are modeled by a navigation graph. This enables an original analysis of the existing coding scheme performance. In a second time, a new coding scheme based on incremental coding is proposed. The experimental results demonstrate that our method outperforms those of the literature by minimizing storage and transmission costs at the same time.

ACKNOWLEDGMENT

This work has received a French government support granted to the Cominlabs excellence laboratory and managed by the National Research Agency in the "Investing for the Future" program under reference ANR-10-LABX-07-01.

REFERENCES

[1] Y. Polyanskiy, "A perspective on massive random-access," in *2017 IEEE International Symposium on Information Theory (ISIT)*, June 2017, pp. 2523–2527.
[2] M. Tanimoto, "FTV: Free-viewpoint television," *Signal Processing: Image Communication*, vol. 27, no. 6, pp. 555–570, Jul. 2012.
[3] Z. Xiong, A. Liveris, and S. Cheng, "Distributed source coding for sensor networks," *IEEE Signal Processing Magazine*, vol. 21, no. 5, pp. 80–94, Sep 2004.
[4] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowdsensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, November 2011.
[5] T. Cover and J. Thomas, *Elements of information theory, second Edition*. Wiley, 2006.
[6] M. Hosseini and V. Swaminathan, "Adaptive 360 VR video streaming: Divide and conquer," in *IEEE International Symposium on Multimedia (ISM)*, 2016, pp. 107–110.
[7] M. Zink, R. Sitaraman, and K. Nahrstedt, "Scalable 360 video stream delivery: Challenges, solutions, and opportunities," *Proceedings of the IEEE*, vol. 107, no. 4, pp. 639–650, 2019.
[8] M. Wien, *High Efficiency Video Coding: Coding Tools and Specification*. Springer, 2015.
[9] J. Lou, H. Cai, and J. Li, "A real-time interactive multi-view video system," in *Proc. ACM Int. Conf. on Multimedia*, Singapore, 2005, pp. 161–170.
[10] M. Wien, R. Cazoulat, A. Graffunder, A. Hutter, and P. Amon, "Real-Time System for Adaptive Video Streaming Based on SVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 9, pp. 1227–1237, Sept 2007.
[11] M. Karczewicz and R. Kurceren, "The SP- and SI-frames design for H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, no. 7, pp. 637–644, July 2003.
[12] H. Kimata, M. Kitahara, K. Kamikura, and Y. Yashima, "Free-viewpoint video communication using multi-view video coding," *NTT Technical Review*, vol. 2, no. 8, pp. 21–26, Aug. 2004.
[13] S. Shimizu, M. Kitahara, H. Kimata, K. Kamikura, and Y. Yashima, "View scalable multiview video coding using 3-d warping with depth map," *IEEE Trans. on Circ. and Syst. for Video Technology*, vol. 17, no. 11, pp. 1485–1495, Nov. 2007.

[14] Y. Liu, Q. Huang, S. Ma, D. Zhao, and W. Gao, "Rd-optimized interactive streaming of multiview video with multiple encodings," *Journal of Visual Communication and Image Representation*, vol. 21, no. 5-6, pp. 523–532, 2010.

[15] G. Cheung, A. Ortega, and N. Cheung, "Interactive streaming of stored multiview video using redundant frame structures," *IEEE Transactions on Image Processing*, vol. 3, no. 3, pp. 744–761, Mar. 2011.

[16] D. Ren, S. . Gary Chan, G. Cheung, and P. Frossard, "Coding structure and replication optimization for interactive multiview video streaming," *IEEE Transactions on Multimedia*, vol. 16, no. 7, pp. 1874–1887, Nov 2014.

[17] F. F. Rodler, "Wavelet based 3d compression with fast random access for very large volume data," in *Proceedings. Seventh Pacific Conference on Computer Graphics and Applications (Cat. No.PR00293)*, Oct 1999, pp. 108–117.

[18] P. Suei, C. Kuo, R. Luoh, T. Kuo, C. Shih, and M. Liang, "Data compression and query for large scale sensor data on cots dbms," in *2010 IEEE 15th Conference on Emerging Technologies Factory Automation (ETFA 2010)*, Sep. 2010, pp. 1–8.

[19] S. Pratapa and D. Manocha, "Rlfc: Random access light field compression using key views and bounded integer sequence encoding," in *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, ser. I3D '19. New York, NY, USA: Association for Computing Machinery, 2019. [Online]. Available: https://doi.org/10.1145/3306131.3317018

[20] A. Ukil, S. Bandyopadhyay, and A. Pal, "IoT data compression: Sensor-agnostic approach," in *Data Compression Conference*, March 2015.

[21] S. Deorowicz and S. Grabowski, "Robust relative compression of genomes with random access," *Bioinformatics*, vol. 27, no. 21, pp. 2979–2986, 2011.

[22] E. Dupraz, T. Maugey, A. Roumy, and M. Kieffer, "Rate-distortion performance of sequential massive random access to gaussian sources with memory," in *2018 Data Compression Conference*, March 2018, pp. 406–406.

[23] S. C. Draper and E. Martinian, "Compound conditional source coding, Slepian-Wolf list decoding, and applications to media coding," in *IEEE International Symposium on Information Theory*, 2007.

[24] A. Roumy and T. Maugey, "Universal lossless coding with random user access: the cost of interactivity," in *Proceedings IEEE International Conference on Image Processing*, Quebec, Canada, Sep. 2015.

[25] E. Dupraz, A. Roumy, T. Maugey, and M. Kieffer, "Rate-storage regions for extractable sourcecoding with side information," *Physical Communication*, 2019 (accepted).

[26] A. Wyner, "Recent Result in the Shannon theory," *IEEE Transactions on Information Theory*, vol. 20, no. 1, pp. 2–10, 1974.

[27] V. Stankovic, A.D.Liveris, Z. Xiong, and C. Georghiades, "On code design for the Slepian-Wolf problem and lossless multiterminal networks," *IEEE Transactions on Information Theory*, vol. 52, no. 4, pp. 1495 – 1507, april 2006.

[28] C. Guillemot and A. Roumy, "Chapter 6 - Toward constructive Slepian-Wolf coding schemes," in *Distributed Source Coding: theory, algorithms and applications*, P. L. Dragotti and M. Gastpar, Eds. Boston: Academic Press, 2009, pp. 131 – 156. [Online]. Available: http://www.sciencedirect.com/science/article/pii/B9780123744852000111

[29] D. Varodayan, A. Aaron, and B. Girod, "Rate-adaptive codes for distributed source coding," *EURASIP Signal Processing*, vol. 86, no. 11, pp. 3123–3130, 2006.

[30] D. Varodayan, "Implementation of Rate-Adaptive LDPC Accumulate Codes for Distributed Source Coding." [Online]. Available: http://ivms.stanford.edu/~varodayan/ldpca.html

[31] A. Liveris, Z. Xiong, and C. Georghiades, "Compression of binary sources with side information at the decoder using LDPC codes," *IEEE Communications Letters*, vol. 6, pp. 440–442, 2002.

[32] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Systems & Control Letters*, vol. 53, no. 1, pp. 65 – 78, 2004. [Online]. Available: https://doi.org/10.1016/j.sysconle.2004.02.022

[33] N. Perraudin, J. Paratte, D. Shuman, L. Martin, V. Kalofolias, P. Vandergheynst, and D. K. Hammond, "GSPBOX: A toolbox for signal processing on graphs," *ArXiv e-prints*, Aug. 2014. [Online]. Available: https://epfl-lts2.github.io/gspbox-html/

Thomas Maugey graduated from Ecole Supérieure d'Electricité, Supélec, Gif-sur-Yvette, France in 2007. He received the M.Sc. degree in fundamental and applied mathematics from Supélec and Université Paul Verlaine, Metz, France, in 2007. He received hi Ph.D. degree in Image and Signal Processing at TELECOM ParisTech, Paris, France in 2010. From October 2010 to October 2014, he was a postdoctoral researcher at the Signal Processing Laboratory (LTS4) of Swiss Federal Institute of Technology (EPFL), Lausanne, Switzerland. Since November 2014, he is a Research Scientist at Inria Rennes-Bretagne-Atlantique. He serves as an Assiociate Editor for EURASIP Journal on advances in signal processing. His research deals with monoview, multiview and 3D video processing and compression.

Aline Roumy received the Engineering degree from Ecole Nationale Superieure de l'Electronique et de ses Applications (ENSEA), France in 1996, the Master degree in 1997 and the Ph.D. degree in 2000 from the University of Cergy-Pontoise, France. During 2000-2001, she was a research associate at Princeton University, Princeton, NJ. On November 2001, she joined INRIA, Rennes, France as a research scientist. She has held visiting positions at Eurecom and Berkeley University. She received the 2011 "Francesco Carassa" Best paper award. She serves as an Associate Editor for the Annals of telecommunications (2016–present), and for IEEE Transactions on Image Processing (2018–present). Her current research and study interests include the area of signal and image processing, coding theory and information theory.

Elsa Dupraz Elsa Dupraz was born in Paris (France). She earned her Master of Science (M.Sc) in Advanced Systems of Radiocommunications (SAR) in 2010 and gratuated from ENS de Cachan and University Paris Sud. In 2013, she got her Ph.D in physics from University Paris-Sud at Laboratoire des Signaux et Systèmes (LSS). From January 2014 to September 2015 she held a post-doctoral position at ETIS (ENSEA, University Cergy-Pointoise, CNRS, France) and ECE department of the University of Arizona (United States). Since October 2015, she is an Assistant Professor at IMT Atlantique. Her research interests lie in the area of coding and information theory, with a special interest on distributed source coding, LDPC codes, and energy-efficient channel codes.

Michel Kieffer is a full professor in signal processing for communications at the Paris-Sud University and a researcher at the Laboratoire des Signaux et Systemes, Gif-sur-Yvette. Since 2009, he is also part-time invited professor at the Laboratoire Traitement et Communication de l'Information, Telecom ParisTech, Paris. His research interests are in signal processing for multimedia, communications, and networking, distributed source coding, network coding, joint source-channel coding and decoding, joint source-network coding. Michel Kieffer is co-author of more than 150 contributions in journals, conference proceedings, or books. He is one of the co-authors of the book Applied Interval Analysis published by Springer-Verlag in 2001, and of the book Joint source-channel decoding: A crosslayer perspective with applications in video broadcasting published by Academic Press in 2009. He serves as associate editor of Signal Processing since 2008 and of the IEEE Transactions on Communications from 2012 to 2016. From 2011 to 2016, Michel Kieffer was junior member of the Institut Universitaire de France.