Array Codes for Functional PIR and Batch Codes

Mohammad Nassar
Technion — Israel Institute of Technology
Haifa 3200003, Israel
mohamadtn@cs.technion.ac.il

Eitan Yaakobi
Technion — Israel Institute of Technology
Haifa 3200003, Israel
yaakobi@cs.technion.ac.il

***Abstract*—A *functional PIR array code* is a coding scheme which encodes some s information bits into a $t \times m$ array such that every linear combination of the s information bits has k mutually disjoint *recovering sets*. Every recovering set consists of some of the array's columns while it is allowed to read at most ℓ encoded bits from every column in order to receive the requested linear combination of the information bits. *Functional batch array codes* impose a stronger property where every multiset request of k linear combinations has k mutually disjoint recovering sets. Given the values of s, k, t, ℓ, the goal of this paper is to study the optimal value of the number of columns m such that these codes exist. Several lower bounds are presented as well as explicit constructions for several of these parameters.**

I. Introduction

Private information retrieval (PIR) codes and batch codes are families of codes which have several applications such as PIR protocols [2], [7], [10], [11], [24], [27], erasure codes in distributed storage systems [17], [18], [21], one-step majority-logic decoding [13], [15], load balancing in storage, cryptographic protocols [12], switch codes [5], [8], [23], and more. They have been recently generalized to *functional PIR* and *functional batch* codes [30]. In this work we study these families of codes when they are used as array codes.

The setup of storing information in array codes works as follows. Assume s bits are encoded to be stored in a $t \times m$ array, where each column corresponds to a *server* such that the encoded bits are stored in the server. The encoded bits should satisfy several properties which depend upon whether the resulting code is a PIR, batch, functional PIR, or functional batch codes. Given a design parameter k of the code, it is required in PIR codes that every information bit has k mutually disjoint *recovering sets*. Here, a recovering set is a set of columns, i.e., servers, in which given the encoded bits in the columns of the recovering set it is possible to recover the information bit. In case it is possible to read only a portion of the encoded bits in every column, we denote this parameter by ℓ. An array code with these parameters and properties is defined as an (s, k, m, t, ℓ) *PIR array code*. Furthermore, it will be called an (s, k, m, t, ℓ) *batch array code* if every *multiset* request of the k information bits has k mutually disjoint recovering sets. In case the requests are not only of information bits but any linear combination of them, we receive an (s, k, m, t, ℓ) *functional PIR array code*, if the same linear combination is requested k times or (s, k, m, t, ℓ) *functional batch array code* for a multiset request of k linear combinations.

The main figure of merit when studying these families of codes is to optimize the number of columns, i.e., servers, given the values of s, k, t, ℓ. Thus, the smallest m such that an (s, k, m, t, ℓ) PIR, batch, functional PIR, functional batch code exists, is denoted by $P_{t,\ell}(s,k)$, $B_{t,\ell}(s,k)$, $FP_{t,\ell}(s,k)$, $FB_{t,\ell}(s,k)$, respectively. Studying the value of $P_{t,\ell}(s,k)$ has been initiated in [11] and since then several more results have appeared; see e.g. [3], [4], [6], [29]. Note that the first work [12] which studied batch codes defined them in their array codes setup and only later on they were studied in their one-dimensional case, also known as *primitive batch codes*; see e.g. [1], [14], [19], [22], [28]. Functional PIR and batch codes have been recently studied in [30] but only for vectors, that is, $t = \ell = 1$. Thus, this paper initiates the study of functional PIR and batch codes in the array setup.

The motivation to study functional PIR and batch codes originates from the observation that in many cases and protocols, such as PIR, the user is not necessarily interested in one of the information bits, but rather, some linear combination of them. Furthermore, functional batch codes are closely related to the family of *random I/O (RIO) codes*, introduced by Sharon and Alrod [20], which are used to improve the random input/output performance of flash memories. A variant of RIO codes, called *parallel RIO codes*, was introduced in [25], and linear codes of this family of codes have been studied in [26]. It was then shown in [30] that in fact linear parallel RIO codes are equivalent to functional batch codes.

The rest of the paper is organized as follows. In Section II, we formally define the codes studied in the paper, discuss some of the previous related work, and list several basic properties. In Section III, we show lower bounds on the number of servers for functional PIR and batch array codes. Section IV lists several code constructions which are based on the Gadget Lemma, covering codes, and several more results for $k = 1, 2$. Section V presents three constructions of array codes and in Section VI the rates of these codes are studied. Due to the lack of space, some of the proofs in the paper are omitted and can be found in the full version of this work in [16].

II. Definitions and Preliminaries

This work is focused on four families of codes, namely *private information retrieval* (*PIR*) codes that were defined recently in [11], *batch codes* that were first studied by Ishai et al. in [12], and their extension to *functional PIR codes* and *functional batch codes* that was recently investigated in [30]. In these four families of codes, s information bits are encoded to m bits. While for PIR codes it is required that every information bit has k mutually disjoint recovering sets, batch codes impose this property for every multiset request of k bits. Similarly, for functional PIR codes it is required that every linear combination of the information bits has k mutually disjoint recovering sets, and functional batch codes impose this property for every multiset request of k linear combination of the bits. While this description of the codes corresponds to the case of one-dimensional codewords, the goal of this work is to study their extension as *array codes*, which is defined as follows. The set $[n]$ denotes the set of integers $\{1, 2, \ldots, n\}$ and $\Sigma = \mathbb{F}_2$.

Definition 1.

1) *An (s, k, m, t, ℓ) **PIR array code** over Σ is defined by an encoding map $\mathcal{E} : \Sigma^s \to (\Sigma^t)^m$ that encodes s information bits $x_1, \ldots, x_s$ into a $t \times m$ array and a decoding function $\mathcal{D}$ that satisfies the following property. For any $i \in [s]$ there is a partition of the columns into k recovering sets $S_1, \ldots, S_k \subseteq [m]$ such that x_i can be recovered by reading at most ℓ bits from each column in S_j, $j \in [k]$.*
2) *An (s, k, m, t, ℓ) **batch array code** over Σ is defined by an encoding map $\mathcal{E} : \Sigma^s \to (\Sigma^t)^m$ that encodes s information bits $x_1, \ldots, x_s$ into a $t \times m$ array and a decoding function $\mathcal{D}$ that satisfies the following property. For any multiset request of k bits $i_1, \ldots, i_k \in [s]$ there is a partition of the*

columns into k recovering sets $S_1, \ldots, S_k \subseteq [m]$ such that x_{i_j}, $j \in [k]$ can be recovered by reading at most ℓ bits from each column in S_j.

3) *An (s,k,m,t,ℓ) **functional PIR array code** over Σ is defined by an encoding map $\mathcal{E}: \Sigma^s \to (\Sigma^t)^m$ that encodes s information bits $x_1, \ldots, x_s$ into a $t \times m$ array and a decoding function $\mathcal{D}$ that satisfies the following property. For any request of a linear combination $\boldsymbol{v}$ of the information bits, there is a partition of the columns into k recovering sets $S_1, \ldots, S_k \subseteq [m]$ such that $\boldsymbol{v}$ can be recovered by reading at most ℓ bits from each column in S_j, $j \in [k]$.*
4) *An (s,k,m,t,ℓ) **functional batch array code** over Σ is defined by an encoding map $\mathcal{E}: \Sigma^s \to (\Sigma^t)^m$ that encodes s information bits $x_1, \ldots, x_s$ into a $t \times m$ array and a decoding function $\mathcal{D}$ that satisfies the following property. For any multiset request of k linear combinations $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_k$ of the information bits, there is a partition of the columns into k recovering sets $S_1, \ldots, S_k \subseteq [m]$ such that $\boldsymbol{v}_j$, $j \in [k]$ can be recovered by reading at most ℓ bits from each column in S_j.*

We refer to each column as a *bucket* and to each entry in a bucket as a *cell*. Furthermore, it is said that a cell stores a *singleton* if one of the information bits is stored in the cell. In the rest of the paper we will refer to every linear combination of the information bits as a binary vector of length s, which indicates the information bits in this linear combination. Our goal is to fix the values of s,k,t and ℓ and then seek to optimize the value of m. In particular, we will have that t and ℓ are fixed, where $t \geqslant \ell$, and then study the growth of m as a function of s and k. Hence, we denote by $P_{t,\ell}(s,k), B_{t,\ell}(s,k), FP_{t,\ell}(s,k), FB_{t,\ell}(s,k)$ the smallest m such that an (s,k,m,t,ℓ) PIR, batch, functional PIR, functional batch code exists, respectively. In case $\ell = t = 1$ we will simply remove them from these notations.

The following upper and lower bounds on the number of buckets for PIR array codes have been shown in [4], [6], [29] and are stated in the following theorem.

Theorem 2.

1) $P_{t,t}(s,k) \geqslant \frac{2 \cdot k \cdot s}{s+t}$, *[4, Th. 3].*
2) *For any integer $t \geqslant 2$ and any integer $s > t$,* $P_{t,t}(s,k) \geqslant \frac{k \cdot s \cdot (2s-2t+1)}{(2s-2t+1)t+(s-t)^2}$, *[4, Th. 4].*
3) *For any integer $t \geqslant 2$ and any integer $s > 2t$,* $P_{t,t}(s,k) \geqslant \frac{2k \cdot s \cdot (s+1)}{(s-t)^2+3st-t^2+2t}$, *[29, Th. 16].*
4) *For any integer $t \geqslant 2$ and any integer $t < s \leqslant 2t$,* $P_{t,t}(s,k) \leqslant \frac{k \cdot s \cdot (2s-2t+1)}{(2s-2t+1)t+(s-t)^2}$, *[4, Th. 6].*
5) *For any integers p,t with $p \leqslant t+1$,* $P_{t,t}(pt,k) \leqslant m$, *where* $k = \binom{t}{t-p+1}\binom{s}{t}$ *and* $m = \binom{t}{t-p+1}\binom{s}{t} + \binom{s-p}{t-p+1}\binom{s-1}{p-1}$, *[6, Th. 10].*

Note that for any two integers $t \geqslant 2$ and $s > t$, the bound in Theorem 2(2) improves upon the bound in Theorem 2(1). This is verified by showing that $\frac{k \cdot s \cdot (2s-2t+1)}{(2s-2t+1)t+(s-t)^2} - \frac{2 \cdot k \cdot s}{s+t} \geqslant 0$ by basic algebraic manipulations. However the lower bound in Theorem 2(1) holds for all values of s, while the one in Theorem 2(2) only for $s > t$. Also, in [29] it was shown that for any two integers $t \geqslant 2$ and $s > 2t$, the bound in Theorem 2(3) is stronger than the bound in Theorem 2(2).

The result in Theorem 2(4) is achieved by Construction 1 in [4]. The authors of [4] presented another construction which is not reported here due to its length. For the exact details please refer to [4, Construction 4 and Th.8]. This construction was then improved in [29] and [6]. Several more constructions of PIR array codes have also been presented in [6], [29].

The following theorem summarizes some of the known basic previous results, as well as several new ones. The proofs are rather simple and are thus omitted.

Theorem 3. *For every s,k,t,ℓ,a positive integers:*

1) $P_{t,\ell}(s,1) = B_{t,\ell}(s,1) = \lceil s/t \rceil$.
2) $FP_{t,\ell}(s,k_1+k_2) \leqslant FP_{t,\ell}(s,k_1) + FP_{t,\ell}(s,k_2)$ *(also for P, B, and FB).*
3) $FP_{t,\ell}(s,a \cdot k) \leqslant a \cdot FP_{t,\ell}(s,k)$ *(also for P, B, and FB).*
4) $FP_{t,\ell}(s_1+s_2,k) \leqslant FP_{t,\ell}(s_1,k) + FP_{t,\ell}(s_2,k)$ *(also for P, B, and FB).*
5) $FP_{t,\ell}(a \cdot s,k) \leqslant a \cdot FP_{t,\ell}(s,k)$ *(also for P, B, and FB).*
6) $FP_{t,\ell}(s,k) \leqslant a \cdot FP_{a \cdot t,\ell}(s,k)$ *(also for P, B, and FB).*

One of the simplest ways to construct array PIR and batch codes uses the Gadget Lemma, which was first proved in [12].

Lemma 4. ***(The Gadget Lemma)*** *Let $\mathcal{C}$ be an $(s,k,m,1,1)$ batch code, then for any positive integer t there exists an $(ts,k,m,t,1)$ batch array code $\mathcal{C}'$ (denoted also by $t \cdot \mathcal{C}$).*

It is easily verified that the Gadget Lemma holds also for PIR codes and therefore $P_{t,\ell}(s,k) \leqslant P_{t,1}(s,k) \leqslant P(\lceil s/t \rceil, k)$ and $B_{t,\ell}(s,k) \leqslant B_{t,1}(s,k) \leqslant B(\lceil s/t \rceil, k)$. However, unfortunately, the Gadget Lemma does not hold in general for functional PIR and batch codes. Even a weaker variation of the Gadget Lemma, where $\ell = t$, does not hold in general for functional PIR and batch codes either. Assume by contradiction that if there is an $(s,k,m,1,1)$ functional PIR code $\mathcal{C}$, then for any positive integer t there exists a (ts,k,m,t,t) functional PIR array code. Then, this will imply that $FP_{t,t}(ts,k) \leqslant FP(s,k)$. However, it is known that $FP(2,2) = 3$ by the simple parity code. Thus, under this assumption it would hold that $FP_{2,2}(4,2) \leqslant FP(2,2) = 3$. But, according to a lower bound on functional PIR array codes, which will be shown in Theorem 6, it holds that $FP_{2,2}(4,2) \geqslant \frac{2 \cdot 2 \cdot 15}{15+3} > 3$, which is a contradiction.

III. Lower Bounds on Array Codes

In this section we present several lower bounds on functional PIR and batch array codes. Let $\left\{ {a \atop b} \right\}$ be the Stirling number of the second kind, which calculates the number of partitions of a set of a elements into b nonempty subsets. It is well known that $\left\{ {a \atop b} \right\} = \frac{1}{b!}\sum_{i=0}^{b}(-1)^{b-i}\binom{b}{i}i^a$.

Theorem 5. *Let s,k,t and ℓ be positive integers. Then,*

1) $FB_{t,\ell}(s,k) \geqslant m^*$, *where m^* is the smallest positive integer such that* $\sum_{i=k}^{m^*}\binom{m^*}{i} \cdot \left\{ {i \atop k} \right\} \cdot \left(\sum_{j=1}^{\ell}\binom{t}{j}\right)^i \geqslant \binom{2^s+k-2}{k}$.
2) $FP_{t,\ell}(s,k) \geqslant m^*$, *where m^* is the smallest positive integer such that* $\sum_{i=k}^{m^*}\binom{m^*}{i} \cdot \left\{ {i \atop k} \right\} \cdot \left(\sum_{j=1}^{\ell}\binom{t}{j}\right)^i \geqslant 2^s - 1$.
3) $FP_{t,\ell}(s,k) \geqslant m^*$, *where m^* is the smallest positive integer such that* $\sum_{i=1}^{m^*-k+1}\binom{m^*}{i} \cdot \left(\sum_{j=1}^{\ell}\binom{t}{j}\right)^i \geqslant k \cdot (2^s - 1)$.
4) $FP_{t,\ell}(s,k) \geqslant \left\lceil \frac{\log_2(k(2^s-1)+1)}{\log_2(\sum_{i=0}^{\ell}\binom{t}{i})} \right\rceil$.

Lastly in this section we show a different lower bound for functional PIR array codes, which is motivated by the corresponding lower bound for PIR array codes from [4, Th. 3].

Theorem 6. *For any s,k,t and ℓ positive integers,* $FP_{t,\ell}(s,k) \geqslant \frac{2 \cdot k \cdot (2^s-1)}{(2^s-1)+(\sum_{i=1}^{\ell}\binom{t}{i})}$.

Proof: Suppose there exists an (s,k,m,t,ℓ) functional PIR array code. There are 2^s-1 possible linear combination requests which are denoted by $\boldsymbol{u}_i$ for $1 \leqslant i \leqslant 2^s-1$. For each $i \in [2^s-1]$, let α_i be the number of recovering sets of size 1 of the i-th linear combination request $\boldsymbol{u}_i$.

Since it is possible to read at most ℓ bits from each bucket, every bucket can satisfy at most $\sum_{i=1}^{\ell}\binom{t}{i}$ linear combinations. Thus, the number of recovering sets of size 1 is $m\cdot\sum_{i=1}^{\ell}\binom{t}{i}$, and $\sum_{j=1}^{2^s-1}\alpha_j \leqslant m\cdot\sum_{i=1}^{\ell}\binom{t}{i}$. Hence, there exists $q\in[2^s-1]$ such that $\alpha_q \leqslant \frac{m\cdot\sum_{i=1}^{\ell}\binom{t}{i}}{2^s-1}$, so out of its k disjoint recovering sets of $\boldsymbol{u}_q$, at most α_q of them are of size 1, and the size of each of the remaining $k-\alpha_q$ subsets is at least 2. Hence,

$$m \geqslant \alpha_q + 2(k-\alpha_q) = 2k-\alpha_q \geqslant 2k - \frac{m\cdot\sum_{i=1}^{\ell}\binom{t}{i}}{2^s-1},$$

and therefore $m(1+\frac{\sum_{i=1}^{\ell}\binom{t}{i}}{(2^s-1)}) \geqslant 2k$, which implies that $FP_{t,\ell}(s,k) \geqslant \frac{2k(2^s-1)}{(2^s-1)+\sum_{i=1}^{\ell}\binom{t}{i}}$. ∎

IV. General Constructions of Array Codes

In this section we present several constructions of array codes for functional PIR and batch codes.

A. Basic Constructions

Even though the Gadget Lemma cannot be extended in general for functional PIR and batch codes, here we show a variation of it that will hold.

Lemma 7. *For any positive integer p, if there exists an $(s, p\cdot k, m, t, \ell)$ functional batch array code, then there exists an $(p\cdot s, k, m, p\cdot t, \ell)$ functional batch array code. Therefore,*

$$FP_{p\cdot t,\ell}(s,k) \leqslant FB_{p\cdot t,\ell}(p\cdot s,k) \leqslant FB_{t,\ell}(s,p\cdot k),$$

and in particular, $FP_{t,1}(s,k) \leqslant FB_{t,1}(s,k) \leqslant FB(\lceil\frac{s}{t}\rceil, t\cdot k)$.

Proof Outline: Let $\mathcal{C}$ be an $(s, p\cdot k, m, t, \ell)$ functional batch array code. We construct a $(p\cdot s, k, m, p\cdot t, \ell)$ functional batch array code $\mathcal{C}'$ by using the code $\mathcal{C}$. The $p\cdot s$ information bits are partitioned into p parts, each of size s, such that the i-th part is encoded to a $t\times m$ array A_i using $\mathcal{C}$. The code $\mathcal{C}'$ is presented by a $pt\times m$ array A that contains the p arrays $A_1,\ldots,A_p$.

Let $R=\{\boldsymbol{v}_1,\ldots,\boldsymbol{v}_k\}$ be a multiset request of size k of the $p\cdot s$ information bits, where $\boldsymbol{v}_i, i\in[k]$ is a binary vector of length ps that represents the i-th request. For each $i\in[k]$, denote $\boldsymbol{v}_i=(\boldsymbol{v}_i^1,\ldots,\boldsymbol{v}_i^p)$ where $\boldsymbol{v}_i^j, j\in[p]$ is a vector of length s, that represents the linear combination of the j-th part of the information bits. Let $R^*=\{\boldsymbol{v}_i^j : 1\leqslant i\leqslant k, 1\leqslant j\leqslant p\}$ be a multiset request of size pk consisting of pk vectors of length s each. By requesting R^* from the code $\mathcal{C}$ we get pk recovering sets. For each $i\in[k]$, the request $\boldsymbol{v}_i$ can be satisfied by the union of the recovering sets of each $\boldsymbol{v}_i^j, j\in[p]$, where for each $j\in[p]$, in the recovering sets of $\boldsymbol{v}_i^j$ we read the cells from the array A_j. It can be shown that each recovering set obtained from $\mathcal{C}$ is used only once in one of the recovering sets in the code $\mathcal{C}'$. Thus, the recovering sets are disjoint and from each bucket at most ℓ cells are read as in the code $\mathcal{C}$. ∎

Another general construction is stated in the next theorem.

Theorem 8. *For any positive integers, s,k,t,t_0, and ℓ, $FB_{t,\ell}(s,k) \leqslant m+m_0$, where $m = FB_{t+t_0,\ell}(s,k)$ and $m_0 = FB_{t,\ell}(m\cdot t_0,k)$.*

The idea behind the proof of Theorem 8 is explained for the specific parameters $t_0=m_0=k=1$. An $(s,1,m+1,t,\ell)$ functional batch array code $\mathcal{C}'$ can be constructed using an $(s,1,m,t+1,\ell)$, $(m,1,1,t,\ell)$ functional batch array codes $\mathcal{C}_1$, $\mathcal{C}_2$, respectively. The code $\mathcal{C}'$ will have the first t rows of the code $\mathcal{C}_1$ and another one bucket which is the encoding of all the linear combinations in the cells of the last row of $\mathcal{C}_1$ using the code $\mathcal{C}_2$. In the decoding, when a linear combination is requested, the recovering set of the code $\mathcal{C}_1$ are used. However, instead of reading the cells in the last row, we read them from the additional bucket in order to get a recovering set for the same request of the code $\mathcal{C}'$. Note that a similar statement can hold for functional PIR array code, where for any positive integers s,k,t,t_0, and ℓ, $FP_{t,\ell}(s,k) \leqslant m+m_0$, where $m=FP_{t+t_0,\ell}(s,k)$ and $m_0=FB_{t,\ell}(m\cdot t_0,k)$.

B. Constructions based upon Covering Codes

In this section it is shown how covering codes are used to construct array codes. Denote by $d_H(\boldsymbol{x},\boldsymbol{y})$ the Hamming distance between two vectors $\boldsymbol{x},\boldsymbol{y}$. Next we remind the definition of covering codes [9].

Definition 9. *Let $n\geqslant 1$, $R\geqslant 0$ be integers. A code $\mathcal{C}\subseteq\Sigma^n$ is called an R-**covering code** if for every word $\boldsymbol{y}\in\Sigma^n$ there is a codeword $\boldsymbol{x}\in\mathcal{C}$ such that $d_H(\boldsymbol{x},\boldsymbol{y})\leqslant R$. The notation $[n,k,R]$ denotes a linear code of length n, dimension k, and covering radius R. The value $g[n,R]$ denotes the smallest dimension of a linear code with length n and covering radius R.*

The following property is well known for linear covering codes; see e.g. [9, Th. 2.1.9].

Property 10. For an $[n,k,R]$ linear covering code with some parity check matrix H, every syndrome vector $s\in\Sigma^{n-k}$ can be represented as the sum of at most R columns of H.

The connection between linear codes and functional batch array codes is established in the next theorem.

Theorem 11. *Let $\mathcal{C}$ be a $[t,t-s,\ell]$ linear covering code, then there exists an $(s,1,1,t,\ell)$ functional batch array code. In particular, $FB_{t,\ell}(t-g[t,\ell],1)=1$.*

Theorem 11 holds also for functional PIR array code and thus the following results are derived.

Corollary 12. *Let s,k,t and ℓ be positive integers. Then,*

1) $FP_{t,\ell}(s,k)\leqslant FB_{t,\ell}(s,k)\leqslant k\cdot\left\lceil\frac{s}{t-g[t,\ell]}\right\rceil$.
2) $FP_{t+t_0,\ell}(s,k)\leqslant FP_{t,t}(s,k)$, *where* $t_0=g[t+t_0,\ell]$. *Also works for FB.*
3) $FP_{t,\ell}(s,k)\leqslant FB_{t,\ell}(s,k)\leqslant k\cdot\left(\lceil\frac{s}{\alpha}\rceil+1\right)$, *where* $\lceil\frac{s}{\alpha}\rceil\leqslant t-g[t,\ell]$, *and* $\alpha=(t+1)-g[(t+1),\ell]$.

The third claim of Corollary 12 is derived from Theorem 11 and Theorem 8.

C. The Cases of $k=1,2$

Even though the cases of $k=1,2$ are the most trivial ones when the codewords are vectors, they are apparently not easily solved for array codes. In this section we summarize some of our findings on these important and interesting cases.

Theorem 13. *For each s,t,ℓ positive integers:*

1) $FP_{t,\ell}(s,1)\geqslant\left\lceil\frac{s}{\log_2(\sum_{i=0}^{\ell}\binom{t}{i})}\right\rceil$.
2) $FP_{t,t}(s,1)=\lceil\frac{s}{t}\rceil$.
3) $\left\lceil\frac{s}{\log_2(t+1)}\right\rceil\leqslant FP_{t,1}(s,1)\leqslant\left\lceil\frac{s}{\lfloor\log_2(t+1)\rfloor}\right\rceil$.
4) $FP_{t,\alpha\cdot t}(s,1)\leqslant\left\lceil\frac{s}{t-g[t,\alpha\cdot t]}\right\rceil$, *where* $0<\alpha<1$.
5) $FP_{t,t/2}(s,1)=\frac{s}{t}+1$, *where t is even, $\frac{s}{t}$ is integer, and* $\frac{s}{t}\leqslant t-1$.

Claims 2, 3 and 4 are derived from Theorem 11. Claim 5 is derived from Corollary 12(2). An improvement for the case of $\ell=1$ is proved in the following theorem.

TABLE I
$(15,1,7,4,1)$ FUNCTIONAL PIR ARRAY CODE

1	2	3	4	5	6	7
x_1	x_3	x_5	x_7	x_9	x_{11}	$x_1x_3x_5x_7x_9x_{11}$
x_2	x_4	x_6	x_8	x_{10}	x_{12}	$x_2x_4x_6x_8x_{10}x_{12}$
x_1x_2	x_3x_4	x_5x_6	x_7x_8	x_9x_{10}	$x_{11}x_{12}$	$x_1\cdots x_{12}$
x_{13}	x_{14}	x_{15}	$x_{13}x_{14}$	$x_{13}x_{15}$	$x_{14}x_{15}$	$x_{13}x_{14}x_{15}$

Theorem 14. *For any positive integers s_1, s_2, and t,*

$$FP_{t,1}(s_1+s_2,1) \leqslant \left\lceil \frac{s_1}{\lfloor \log_2(t+1) \rfloor} \right\rceil + 1,$$

where $2^{s_2}-1 \leqslant \left(\left\lceil \frac{s_1}{\lfloor \log_2(t+1) \rfloor} \right\rceil + 1 \right) (t-(2^{\lfloor \log_2(t+1) \rfloor}-1))$.

Proof: We construct an $(s_1+s_2,1,m,t,1)$ functional PIR array code for $m = \left\lceil \frac{s_1}{\lfloor \log_2(t+1) \rfloor} \right\rceil + 1$. The first s_1 information bits are divided into $m-1$ parts, where $h_i, i \in [m-1]$ is the size of part i, and $h_i \leqslant \lfloor \log_2(t+1) \rfloor$. Then, all the linear combinations of part $i \in [m-1]$ are written in the i-th bucket, so in each of the first $m-1$ buckets there are at least $t-(2^{\lfloor \log_2(t+1) \rfloor}-1)$ empty cells. In the last bucket, the parity of each of the first $2^{\lfloor \log_2(t+1) \rfloor}-1$ rows is stored. Since $2^{s_2}-1 \leqslant m \cdot (t-(2^{\lfloor \log_2(t+1) \rfloor}-1))$, each of the $2^{s_2}-1$ linear combinations of the s_2 bits can be written in the empty cells of the m buckets.

Let $\boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m)$ be a request such that for any $i \in [m-1]$ the length of $\boldsymbol{v}_i$ is h_i, the length of $\boldsymbol{v}_m$ is s_2, and for simplicity assume that they are all nonzero. The linear combination $\boldsymbol{v}_m$ is satisfied by the cell where it is stored and assume it is in the j-th bucket, where $j < m$. Assume that the cell in the j-th bucket where the linear combination $\boldsymbol{v}_j$ is stored is in row r. We read from each bucket $b \in [m-1]$, where $b \neq j$ the cell with the linear combination represented by $\boldsymbol{v}_b + \boldsymbol{u}_b$, where $\boldsymbol{u}_b$ is the vector that represents the cell in bucket b in row r, while if $\boldsymbol{v}_b + \boldsymbol{u}_b = \boldsymbol{0}$ we don't read from bucket b. Also, we read the cell in row r from the last bucket. Then the linear combination we get is the combination that is represented by $(\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{m-1})$, because $\sum_{1 \leqslant b \leqslant m, b \neq j} \boldsymbol{u}_b = \boldsymbol{v}_j$ and for each $b \in [m-1]$ where $b \neq j$ we read the linear combination that is represented by $\boldsymbol{v}_b + \boldsymbol{u}_b$ from bucket b. ∎

For any t, s_1, s_2 where $s = s_1 + s_2$ and $s_2 \geqslant \lfloor \log_2(t+1) \rfloor$, the upper bound in Theorem 14 improves upon the one in Theorem 13(3) since $\left\lceil \frac{s}{\lfloor \log_2(t+1) \rfloor} \right\rceil \geqslant \left\lceil \frac{s_1}{\lfloor \log_2(t+1) \rfloor} \right\rceil + 1$.

Example 1. In this example the construction of a $(15,1,7,4,1)$ functional PIR array code is demonstrated based on Theorem 14. It can be verified that the parameters $t=4, s_1=12$ and $s_2=3$ satisfy the constraints of Theorem 14. The construction is given in Table I. The first $s_1=12$ information bits are partitioned into 6 parts, each part of size 2. All the nonzero linear combinations of part $i, i \in [6]$ are written in the i-th bucket with one cell remains empty. The sum of each of the first 3 rows is written. Now, there are still 7 empty cells, which are used to store all the nonzero linear combinations of the last $s_2=3$ bits in the empty cells. This example implies that $FP_{4,1}(15,1) \leqslant 7$, and from Theorem 13(3) it can be verified that $FP_{4,1}(15,1) \geqslant 7$. Thus, $FP_{4,1}(15,1)=7$. Note that in this example and in the rest of the paper the notation $x_{i_1}x_{i_2}\cdots x_{i_h}$ is a shorthand to the summation $x_{i_1}+x_{i_2}+\cdots+x_{i_h}$. □

Lastly, we report on several results for $k=2$.

Theorem 15. *Let s be positive integer. Then,*

1) $6 \leqslant FB_{2,2}(8,2) \leqslant 7$.
2) $6 \leqslant FB_{3,1}(8,2) \leqslant 7$.
3) $0.71s \lesssim \log_7(2^{s-1} \cdot (2^s-1)) \leqslant FB_{2,2}(s,2) \leqslant 7 \cdot \lceil \frac{s}{8} \rceil$.

TABLE II
$(8,2,7,2,2)$ FUNCTIONAL PIR ARRAY CODE

1	2	3	4	5	6	7
x_1	x_2	x_1x_2	x_5	x_6	x_5x_6	$x_1x_2x_5x_6$
x_3	x_4	x_3x_4	x_7	x_8	x_7x_8	$x_3x_4x_7x_8$

TABLE III
CONSTRUCTION A FOR $t=2$

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
x_1	x_1	x_1	x_1	x_1	x_2	x_2	x_2	x_2	x_3	x_3	x_3	x_4	x_4	x_5
x_2	x_3	x_4	x_5	x_6	x_3	x_4	x_5	x_6	x_4	x_5	x_6	x_5	x_6	x_6

16	17	18	19	20	21	22	23	24	25
$x_1x_2x_3$	$x_1x_2x_4$	$x_1x_2x_5$	$x_1x_2x_6$	$x_1x_3x_4$	$x_1x_3x_5$	$x_1x_3x_6$	$x_1x_4x_5$	$x_1x_4x_6$	$x_1x_5x_6$
$x_4x_5x_6$	$x_3x_5x_6$	$x_3x_4x_6$	$x_3x_4x_5$	$x_2x_5x_6$	$x_2x_4x_6$	$x_2x_4x_5$	$x_2x_3x_6$	$x_2x_3x_5$	$x_2x_3x_4$

The first claim of Theorem 15 can be verified by using Theorem 5(1) and the construction given in Table II. The second claim of Theorem 15 is derived from the first claim and Corollary 12(2). Lastly, the upper bound of the third claim is derived from the first claim and Theorem 3(5). The lower bound is derived from Theorem 5(1).

V. SPECIFIC CONSTRUCTIONS OF ARRAY CODES

In this section we discuss three constructions of array codes.

A. Construction A

The first construction was given in [11, Th.20], where it was proved in [6, Th.10] that this construction gives a PIR array code for any integer $t \geqslant 2$. We study how it can be used also as batch and functional PIR array codes for $t=2$. Denote by $\mathcal{C}_2^A$ the code that is obtained from this construction with $t=2$ which is demonstrated in Table III.

Now we want to show that the code $\mathcal{C}_2^A$ is a $(6,15,25,2,2)$ batch array code, by using several properties which are proved in the following three lemmas. For each $i \in [6]$ denote by $\mathcal{F}_i \subseteq [15]$ the subset of buckets from the first 15 buckets, that have a cell with the singleton x_i. Assume that every multiset request R of size $k=15$ is represented by a vector $(k_1, \ldots, k_6)$, where k_i indicates the number of times x_i appears in the multiset request and $k_1 \geqslant \cdots \geqslant k_6$.

Lemma 16. *For any multiset request $(k_1, \ldots, k_6)$ of size $k=15$, the code $\mathcal{C}_2^A$ can satisfy all the requests of bits x_3, x_4, x_5, x_6 by using only the first 15 buckets.*

Lemma 17. *In the code $\mathcal{C}_2^A$, for any information bit x_i and for any bucket $b_1 \in [15] \setminus \mathcal{F}_i$, there exists a bucket $b_2, 16 \leqslant b_2 \leqslant 25$ such that $\{b_1, b_2\}$ is a recovering set of x_i. In addition, the $|[15] \setminus \mathcal{F}_i|$ recovering sets we get are disjoint.*

For any information bit $x_i, i \in [6]$ denote by R_b^i the recovering set that uses bucket $b \in [15]$ and can satisfy x_i. For example, $R_1^1 = \{1\}$ and $R_{12}^1 = \{12, 22\}$.

Lemma 18. *For the two information bits x_1, x_2, the buckets $\{10, 11, \ldots, 15\}$ are divided into 3 pairs, $\{(10,15), (11,14), (12,13)\}$, such that for any pair (b_1, b_2), it holds that $\left|R_{b_1}^1 \cap R_{b_2}^2\right| > 0$ and $\left|R_{b_1}^2 \cap R_{b_2}^1\right| > 0$.*

Theorem 19. *The code $\mathcal{C}_2^A$ is a $(6,15,25,2,2)$ batch array code. In particular, $B_{2,2}(6,15) = 25$.*

The upper bound in Theorem 19 can be verified based on Lemmas 16, 17, and 18. The lower bound is derived from Theorem 2(3). In addition it is possible to show that the code $\mathcal{C}_2^A$ is a $(6,11,25,2,2)$ functional PIR array code. This is stated in the following theorem while the lower bound is obtained using Theorem 6.

Theorem 20. *The code $\mathcal{C}_2^A$ is a $(6,11,25,2,2)$ functional PIR array code. In particular, $21 \leqslant FP_{2,2}(6,11) \leqslant 25$.*

TABLE IV
CONSTRUCTION 21 FOR $r = 3$

1	2	3	4
$x_1x_2x_3$	x_1	x_2	x_1
x_4	x_2	x_3	x_3
x_6	$x_4x_5x_6$	x_4	x_5
x_7	x_7	x_5	x_6
x_8	x_9	$x_7x_8x_9$	x_8
x_{10}	x_{10}	x_{11}	x_9
x_{11}	x_{12}	x_{12}	$x_{10}x_{11}x_{12}$

B. Construction B

Next we generalize an example given in [11] of a PIR code and study how it can be used also as batch array codes. First the construction for the general case is presented.

Construction 21. *Let $r \geqslant 3$ be a fixed integer, the number of information bits is $s = r(r+1)$, the number of the buckets is $m = r+1$, and the number of the cells in each bucket is $t = (r-1)r+1$. The information bits are partitioned into $r+1$ parts each of size r, denote by $\mathcal{S}_i$ the part i of the bits. For each $i \in [r+1]$, write the linear combination $\sum_{j\in\mathcal{S}_i} x_j$ to bucket i. For each $i, i \in [r+1]$ write each one of the subsets of size $r-1$ of $\mathcal{S}_i$ as singletons in a different bucket other than bucket i.*

For $r \geqslant 3$ denote the code obtained from Construction 21 by $\mathcal{C}_r^B$. Construction 21 for the case of $r = 3$ is demonstrated in Table IV. It is possible to show that for any $r \geqslant 3$ the code $\mathcal{C}_r^B$ is an $(r^2+r, r, r+1, r^2-r+1, r-1)$ PIR array code.

Theorem 22. *For any integer $r \geqslant 3$ the code $\mathcal{C}_r^B$ from Construction 21 is an $(r^2+r, r, r+1, r^2-r+1, r-1)$ PIR array code. In particular,*

$$\frac{r\cdot(4r^2+3r-1)}{4r^2-r+1} \leqslant P_{r^2-r+1,r-1}(r^2+r, r) \leqslant r+1.$$

Next we want to show that for any integer $r \geqslant 3$ the code $\mathcal{C}_r^B$ is an $(r^2+r, r, r+1, r^2-r+1, r-1)$ batch array code, by using a property stated in the following lemma.

Lemma 23. *For any integer $r \geqslant 3$ it holds that every two buckets of the code $\mathcal{C}_r^B$ can form a recovering set of every bit x_i by reading at most $r-1$ cells from each bucket.*

Theorem 24. *For any integer $r \geqslant 3$ the code $\mathcal{C}_r^B$ from Construction 21 is an $(r^2+r, r, r+1, r^2-r+1, r-1)$ batch array code. In particular,*

$$\frac{r\cdot(4r^2+3r-1)}{4r^2-r+1} \leqslant B_{r^2-r+1,r-1}(r^2+r, r) \leqslant r+1.$$

The lower bound in Theorem 24 (and 22) is achieved by using Theorem 2(2). The upper bound is achieved by using Contruction 21, where it can be verified that for any multiset request of $r+1$ bits, the code $\mathcal{C}_r^B$ can satisfy the first $r-1$ bits of the request by using only $r-1$ buckets. Then by using the remaining two buckets the code $\mathcal{C}_r^B$ can satisfy the last bit according to Lemma 23. Furthermore, since for any $r \geqslant 3$, $r < \frac{r\cdot(4r^2+3r-1)}{4r^2-r+1} \leqslant P_{r^2-r+1,r-1}(r^2+r, r) \leqslant B_{r^2-r+1,r-1}(r^2+r, r) \leqslant r+1$, we conclude that Construction 21 gives optimal PIR and batch array codes.

C. Construction C

We present our third construction and study how it is used as PIR and functional PIR array codes for several parameters.

Construction 25. *Let $s \geqslant 2$ be a fixed integer. The number of information bits is s, the number of cells in each bucket (the number of the rows) is 2. We write each two nonzero disjoint linear combinations of total size at most s, thus we need $m = \sum_{i=2}^{s}(\binom{s}{i}\cdot\{{i \atop 2}\})$ buckets. Then,*

$$m = \sum_{i=2}^{s}\left(\binom{s}{i}\left\{{i \atop 2}\right\}\right) = \sum_{i=2}^{s}\binom{s}{i}(2^{i-1}-1) = \frac{3^s+1}{2} - 2^s.$$

TABLE V
CONSTRUCTION FOR $s = 4$

1	2	3	4	5	6	7	8	9	10	11	12	13	14
x_1	x_1	x_1	x_2	x_2	x_3	x_1	x_1	x_1	x_2	x_2	x_2	x_3	x_3
x_2	x_3	x_4	x_3	x_4	x_4	x_2x_3	x_2x_4	x_3x_4	x_1x_3	x_1x_4	x_3x_4	x_1x_2	x_1x_4

15	16	17	18	19	20	21	22	23	24	25
x_3	x_4	x_4	x_4	x_1	x_2	x_3	x_4	x_1x_2	x_1x_3	x_1x_4
x_2x_4	x_1x_2	x_1x_3	x_2x_3	$x_2x_3x_4$	$x_1x_3x_4$	$x_1x_2x_4$	$x_1x_2x_3$	x_3x_4	x_2x_4	x_2x_3

For any integer $s \geqslant 2$ denote the code that is obtained from Construction 25 by $\mathcal{C}_s^C$. Construction 25 for the case of $s = 4$ is demonstrated in Table V and provides the following results.

Theorem 26.

1) $23 \leqslant P_{2,1}(4, 16) \leqslant 25$.
2) $24 \leqslant FP_{2,2}(4, 14) \leqslant 25$.
3) $88 \leqslant FP_{2,2}(5, 48) \leqslant 90$.

The lower bound of the first claim is obtained using Theorem 2(2). The lower bounds of the last two claims are obtained using Theorem 6.

VI. ASYMPTOTIC ANALYSIS OF ARRAY CODES

The goal of this section is to provide a figure of merit in order to compare between the different constructions of array codes. For simplicity we consider the case where $\ell = t$, that is, it is possible to read all the bits in every bucket. Under this setup, it holds that $FP_{t,t}(s,k) \leqslant sk/t$ for all s, k, and t. This motivates us to define the following values

$$\mathcal{R}_X(t,k) = \limsup_{s\to\infty} \frac{X_{t,t}(s,k)}{sk/t},$$

where $X \in \{P, B, FP, FB\}$. The case where $t = 1$ has been studied in several previous works. For example, for functional PIR array codes we have $\mathcal{R}_{FP}(1,k) \geqslant \frac{1}{k\cdot H(1/k)}$ for any even integer $k \geqslant 4$ [30, Th. 13]. Also, for functional batch array codes it holds from [30, Th. 21] that $\mathcal{R}_{FB}(1,k) \leqslant \frac{1}{k\cdot H(c_k)}$, where $c_1 = \frac{1}{2}$ and c_{k+1} is the root of the polynomial $H(z) = H(c_k) - zH(c_k)$. For the case $k = 1$ we have $\mathcal{R}_{FB}(t,1) = \mathcal{R}_{FP}(t,1) = 1$ from Theorem 13(2). According to the bounds and constructions studied in the paper, we can already summarize several results in the following theorems for $t = 2$ and some other values.

Theorem 27.

1) $\mathcal{R}_{FP}(2,2) \leqslant \mathcal{R}_{FB}(2,2) \leqslant \frac{7}{8} = 0.875$, *and* $\mathcal{R}_{FB}(2,2) \geqslant 0.71$.
2) $\mathcal{R}_{FP}(2,11) \leqslant \frac{25}{33} = 0.758$. $\mathcal{R}_{FP}(2,14) \leqslant \frac{25}{28} = 0.893$. $\mathcal{R}_{FP}(2,48) \leqslant \frac{3}{4} = 0.75$.
3) $\mathcal{R}_P(2,16) \leqslant \frac{25}{32} = 0.78125$. $\mathcal{R}_B(2,15) \leqslant \frac{5}{9} = 0.556$.

Theorem 28.

1) *For any $r \geqslant 3$, $\mathcal{R}_P(r^2-r+1, r) \leqslant \frac{(r+1)(r^2-r+1)}{r(r^2+r)}$ (also for B).*
2) *For any $t \geqslant 2$, $\mathcal{R}_P(t,k) \leqslant \frac{m}{k(t+1)}$, where $k = \binom{t(t+1)}{t}$ and $m = k + \frac{\binom{t(t+1)}{t+1}}{t}$.*
3) *For any two integers t and k, $\mathcal{R}_{FB}(t,k) \leqslant \frac{1}{k\cdot H(c_{tk})}$, where $c_1 = \frac{1}{2}$ and c_{k+1} is the root of the polynomial $H(z) = H(c_k) - zH(c_k)$. (From Lemma 7).*
4) *For any positive integers t, k, and a, $\mathcal{R}_X(t, a\cdot k) \leqslant \mathcal{R}_X(t,k)$ and $\mathcal{R}_X(t,k) \leqslant \mathcal{R}_X(a\cdot t, k)$, where $X \in \{P, B, FP, FP\}$.*

ACKNOWLEDGMENT

This work was partially supported by the ISF grant 1817/18 and by the Technion Hiroshi Fujiwara cyber security research center and the Israel cyber directorate.

REFERENCES

[1] H. Asi and E. Yaakobi, "Nearly optimal constructions of PIR and batch codes," in *Proc. IEEE Int. Symp. Inf. Theory*, pp. 151–155, Aachen, Germany, Jun. 2017.

[2] A. Beimel, Y. Ishai, E. Kushilevitz, and J.F. Raymond, "Breaking the $O(n^{1/(2k-1)})$ barrier for information theoretic private information retrieval," *Proc. of the 43rd Symposium on Foundations of Computer Science*, Vancouver, B.C., IEEE Computer Society, pp. 261–270, 2002.

[3] S. Blackburn and T. Etzion, "PIR array codes with optimal PIR rates," in *Proc. IEEE Int. Symp. on Inf. Theory*, pp. 2658–2662, Aachen, Germany, Jun. 2017.

[4] S. Blackburn and T. Etzion, "PIR array codes with optimal virtual server rate," *IEEE Trans. on Inf. Theory*, vol. 65, no. 10, pp. 6136–6145, Oct. 2019.

[5] S. Buzaglo, Y. Cassuto, P. H. Siegel, and E. Yaakobi, "Consecutive switch codes," *IEEE Trans. Inform. Theory*, vol. 64, no. 4, pp. 2485–2498, Apr. 2018.

[6] Y.M. Chee, H.M. Kiah, E. Yaakobi, and H. Zhang, "A generalization of Blackburn-Etzion construction for PIR array codes," *Proc. IEEE Int. Symp. on Inf. Theory*, pp. 1062–1066, Paris, France, Jul. 2019.

[7] B. Chor, E. Kushilevitz, O. Goldreich, and M. Sudan, "Private information retrieval," *J. ACM*, 45, 1998. Earlier version in FOCS 95.

[8] Y. M. Chee, F. Gao, S. T. H. Teo, and H. Zhang, "Combinatorial systematic switch codes," in *Proc. IEEE Int. Symp. Inf. Theory*, pp. 241–245, Hong Kong, Jun. 2015.

[9] G. Cohen, I. Honkala, S. Litsyn, and A. Lobstein, *Covering Codes*, North-Holland, Amsterdam, 1997.

[10] Z. Dvir and S. Gopi, "2-server PIR with subpolynomial communication," *J. ACM*, vol. 63, no. 4, pp. 39:1–39:15, Nov. 2016.

[11] A. Fazeli, A. Vardy, and E. Yaakobi, "PIR with low storage overhead: coding instead of replication," arXiv:1505.06241, May 2015.

[12] Y. Ishai, E. Kushilevitz, R. Ostrovsky, and A. Sahai, "Batch codes and their applications," in *Proc. of the 36-sixth Annual ACM Symposium on Theory of Computing*, pp. 262–271, Chicago, ACM Press, 2004.

[13] S. Lin and D.J. Costello, *Error Control Coding*. Upper Saddle River, NJ, USA: Prentice-Hall, 2004.

[14] H. Lipmaa and V. Skachek, "Linear batch codes," *Coding Theory and Applications, CIM Series*, vol. 3. pp. 245–253, 2015.

[15] J.L. Massey, *Threshold Decoding*. Cambridge, MA, USA: MIT Press, 1963.

[16] M. Nassar and E. Yaakobi, "Array codes for functional PIR and batch codes," *arXiv*, Jan. 2020.

[17] L. Pamies-Juarez, H. D. Hollmann, and F. Oggier, "Locally repairable codes with multiple repair alternatives," in *Proc. IEEE Int. Symp. Inf. Theory*, pp. 892–896, Istanbul, Turkey, Jul. 2013.

[18] A. Rawat, D. Papailiopoulos, A. Dimakis, and S. Vishwanath, "Locality and availability in distributed storage," in *Proc. IEEE Int. Symp. Inf. Theory*, pp. 681–685, Honolulu, HI, Jun. 2014.

[19] A. S. Rawat, Z. Song, A. G. Dimakis, and A. Gál, "Batch codes through dense graphs without short cycles," *IEEE Trans. Inform. Theory*, vol. 62, no. 4, pp. 1592–1604, Apr. 2016.

[20] E. Sharon and I. Alrod, "Coding scheme for optimizing random I/O performance," *Non-Volatile Memories Workshop*, San Diego, Apr. 2013.

[21] I. Tamo and A. Barg, "A family of optimal locally recoverable codes," *IEEE Trans. on Inf. Theory*, vol. 60, no. 8, pp. 4661–4676, Aug. 2014.

[22] A. Vardy and E. Yaakobi, "Constructions of batch codes with near-optimal redundancy," in *Proc. IEEE Int. Symp. Inf. Theory*, pp. 1197–1201, Barcelona, Spain, Jul. 2016.

[23] Z. Wang, H.M. Kiah, Y. Cassuto, and J. Bruck, "Switch codes: Codes for fully parallel reconstruction," *IEEE Trans. on Infor. Theory*, vol. 63, no. 4, pp. 2061–2075, Apr. 2017.

[24] G. William, "A survey on private information retrieval," Bulletin of the EATCS. 2004.

[25] E. Yaakobi and R. Motwani, "Construction of random input-output codes with moderate block lengths," in *Proc. IEEE Trans. on Comm.*, vol. 64, no. 5, pp. 1819–1828, May 2016.

[26] A. Yamawaki, H. Kamabe, and S. Lu, "Construction of parallel RIO codes using coset coding with Hamming code," in *Proc. IEEE Inf. Theory Workshop (ITW)*, pp. 239–243, Kaohsiung, Taiwan, Nov. 2017.

[27] S. Yekhanin, "Private information retrieval," *Comm. of the ACM*, vol. 53, no. 4, pp. 68–73, 2010.

[28] H. Zhang and V. Skachek, "Bounds for batch codes with restricted query size," in *Proc. IEEE Int. Symp. Inf. Theory*, pp. 1192–1196, Barcelona, Spain, Jul. 2016.

[29] H. Zhang, X. Wang, H. Wei, and G. Ge, "On private information retrieval array codes," *IEEE Trans. on Inf. Theory*, vol. 65, no. 9, pp. 5565–5573, Sep. 2019.

[30] Y. Zhang, E. Yaakobi, and T. Etzion, "Bounds on the length of functional PIR and batch codes," arXiv:1901.01605, Jan. 2019.