Compressing Sparse Sequences under Local Decodability Constraints

Ashwin Pananjady and Thomas A. Courtade
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Email: {ashwinpm, courtade}@eecs.berkeley.edu

***Abstract*—We consider a variable-length source coding problem subject to local decodability constraints. In particular, we investigate the blocklength scaling behavior attainable by encodings of r-sparse binary sequences, under the constraint that any source bit can be correctly decoded upon probing at most d codeword bits. We consider both adaptive and non-adaptive access models, and derive upper and lower bounds that often coincide up to constant factors. Notably, such a characterization for the fixed-blocklength analog of our problem remains unknown, despite considerable research efforts. Connections to communication complexity are also briefly discussed.**

I. Introduction

Efficient representation of a sequence of *source bits* by a significantly shorter sequence of *encoded bits* (i.e., a *codeword*) is the classical problem of lossless source coding, proposed by Shannon in his seminal 1948 paper [1]. It is widely known that optimal compression performance can be achieved with schemes such as Huffman codes [2] or the Lempel-Ziv universal compression algorithm [3]. However, these compression schemes suffer from the drawback that they do not support *local decodability*. Specifically, retrieving a single bit of the source sequence generally requires a decoder to access *all* of the encoded bits.

This is clearly undesirable in applications that favor retrieving selected pieces of information, rather than the entire source sequence. One such application is in bioinformatics [4], [5], in which a DNA sequence is stored as a binary string with relation to a *reference* sequence, with 1s representing single nucleotide polymorphisms (SNPs) at those positions. In SNP calling, we are interested in learning whether there is a SNP at position i. Since we are not interested in any other information about the sequence, we would ideally like to accomplish this by accessing few bits in the compressed representation of the DNA sequence. In this specific instance, local decodability is strongly motivated, since decompressing the whole genome can be prohibitively expensive from a memory standpoint.

Another example presents itself in the efficient storage of relationships among objects (e.g., relational databases [6]). Given a collection of n objects, the relationships among these objects can be represented by an undirected graph on n vertices, with the presence or absence of an edge (i, j) signifying that objects i and j are related or unrelated, respectively (e.g., friendships in social networks). One can think of representing all graphs with n vertices by sequences of $\binom{n}{2}$ bits representing all possible edges. A '1' at a given position indicates the presence of that edge, and a '0' indicates that it is absent. Thus, testing relationships between objects is accomplished by querying the value of the corresponding bit. As in SNP calling, it would be ideal to have a compressed representation of the graph which permits such queries upon accessing a small number of encoded bits.

We remark that both applications referred to above involve a source that is inherently sparse - both SNPs and the number of relationships are small compared to the total length of the sequence. Motivated in part by this, our objective in this paper is to analyze the fundamental tradeoffs between access constraints and compressibility of sparse sequences, in the context of locally decodable compression. We consider a variable blocklength model, in which source sequences can be mapped to codewords of varying lengths, and that the decoder is informed of the codeword length at the start of the decoding process.

Prior work on the problem of locally decodable source coding includes results on succinct data structures in the *bit probe* [7], [8] and *cell probe* [9], [10] complexity models. A widely studied problem in the bit-probe model is the static membership problem, which is closely related to the problem we consider. The bit-probe model for the static membership problem encodes subsets $S \subseteq \{1, 2, \ldots, n\}$ of size at most r into a data structure of *fixed* length ℓ, such that queries of the form "Is $i \in S$" for $i \in \{1, 2, \ldots, n\}$ can be determined by probing (i.e., accessing) at most d bits in the data structure, either adaptively or non-adaptively. Buhrman et al. [11] provided the lower bound $\ell = \Omega(dr^{1-1/d}n^{1/d})$, which remains the best lower bound for a general n, r, d. They also showed a scheme that achieves a blocklength $\ell = O(rd'n^{1/d'})$, with $d' = d - \Theta(\log r + \log\log n)$ and $d > \Theta(\log r + \log\log n)$. The interested reader is referred to [12] for a comprehensive survey on several improvements to these bounds (for specific regimes of r and d) that have been proposed in the literature [13]–[16]. Notably, [13] considered $d \leq 4$ probes and showed that, for $r = o(n)$, a blocklength $\ell = o(n)$ can be achieved by schemes using 3 adaptive probes or 4 non-adaptive probes, thereby settling a question posed in [11]. By letting S denote the set of indices where a binary sequence has ones, the static membership problem considered by the bit probe model yields a fixed-blocklength, locally decodable representation of sparse sequences, and can therefore be viewed as a fixed-blocklength

This work was supported in part by the NSF Center for Science of Information CCF-0939370.

counterpart to the problem that we consider.

Largely independent from the prior work on the bit probe model, locally decodable source coding has also received recent attention from the information theory community [17]–[19]. Closely related to the present paper is the recent work by Makhdoumi et al. [18], which considers the design of locally decodable source codes under the bit probe model for i.i.d. Bernoulli sources, with vanishing block-error probability.

Our model differs fundamentally from those appearing in both [11] and [18] since we consider a variable-blocklength setting (to be defined precisely in Section II), which has not been previously studied. This is practically motivated because compressed file size is rarely fixed a-priori by the compression scheme, and file length is often recorded in metadata available to a decompressor. As we show in the sequel, our analysis of the variable blocklength case allows us to provide tight order-wise bounds on the average blocklength of the code in many cases — a problem that has remained open for the fixed-blocklength bit-probe problem for over 3 decades. Also, in contrast to [18], we restrict ourselves exclusively to the *lossless* setting, in which the decoder must have zero (and not vanishing) probability of error, which is motivated by high-fidelity applications such as SNP calling.

Our Contributions: In this paper, we give upper and lower bounds on the average blocklength attainable by variable-blocklength compression schemes under local decodability constraints. These bounds are non-asymptotic in nature, and coincide (up to constant factors) in many cases. As a corollary, we give necessary and sufficient conditions on the number of bit-probes required to achieve competitively optimal compression performance, and briefly comment on connections to communication complexity.

II. Notation and Problem Setting

For an integer $k \geq 1$, we employ the shorthand notation $[k] \triangleq \{1, 2, \dots, k\}$. We make frequent use of the conventional notations $O(\cdot), o(\cdot), \Omega(\cdot), \omega(\cdot), \Theta(\cdot)$.

Throughout, we consider encodings of r-sparse binary vectors, which are simply sequences $x^n = (x_1, x_2, \dots, x_n) \in \{0,1\}^n$ having Hamming weight precisely r (we may assume without loss of generality that $r \leq n/2$). Our restriction to sequences of weight precisely r is primarily for convenience, since our arguments readily generalize to vectors having weight at most r. In some cases, we allow the sparsity parameter r to scale with n, in which case we write r_n.

The *support* of a source sequence x^n is defined to be the set of nonzero coordinates, i.e. $\mathsf{supp}(x^n) = \{i \in [n] : x_i = 1\}$. When referring to multiple distinct sequences, we will use the bracket subscript notation, i.e. $x^n_{(1)}, x^n_{(2)}, \cdots$.

Letting $\binom{[n]}{r} \subset \{0,1\}^n$ denote the set of r-sparse binary vectors, we assume random vectors $X^n \in \binom{[n]}{r}$ are drawn uniformly from all $\binom{n}{r}$ possibilities. A source code (i.e., compressor) c for r-sparse vectors is an invertible mapping $\mathsf{c}\colon \binom{[n]}{r} \to \{0,1\}^*$, where $\{0,1\}^* = \{0, 1, 00, 01, 10, \dots\}$ denotes the set of all binary strings. For a codeword $\mathsf{c}(x^n) = (c_1, c_2, \dots, c_\ell)$ and a set of integers $S \subset \{1, 2, \dots\}$, we let $\mathsf{c}_S(x^n) = \{c_i : i \in S\}$ denote the coordinates of the codeword indexed by entries in S. Letting $\ell(b)$ denote the length of $b \in \{0,1\}^*$, we remark that there are source codes for which the average codeword length is roughly[1]

$$\mathbb{E}\left[\ell(\mathsf{c}(X^n))\right] \approx \log \binom{n}{r} \ \mathsf{bits}, \tag{1}$$

and this is best-possible, since the entropy of the source $H(X^n) = \log \binom{n}{r}$. Indeed, the naïve scheme which lists the positions of each nonzero entry (requiring approximately $\log n$ bits each) is essentially optimal when $r \ll n$. However, it is not clear whether such a source code admits a decoding algorithm that, for any specified index $j \in [n]$, can recover bit x_j by probing a *bounded* number of bits in $\mathsf{c}(x^n)$. Thus, in the spirit of locally-decodable error-correcting codes [20] and the data structure counterparts in [11], [18], we define a *variable-length* (r, d, n)*-locally decodable source code*:

Definition 1. *A (r, d, n)-locally decodable source code, or simply, an (r, d, n) code, consists of a mapping*

$$\mathsf{c} : \binom{[n]}{r} \to \{0,1\}^* \tag{2}$$

with the property that, for each x^n and $j \in [n]$, there exists a set $S \subseteq \{1, 2, \dots\}$ of size $|S| \leq d$ for which x_j is a function of $\ell(\mathsf{c}(x^n))$ and $\mathsf{c}_S(x^n)$.

In other words, we can say c is a (r, d, n)-locally decodable source code only if there exists a corresponding '(r, d, n)-local decompressor' — i.e., an algorithm that takes as input a *query index* $j \in [n]$ and the codeword length $\ell(\mathsf{c}(x^n))$, and returns the data bit x_j after accessing at most d bits of $\mathsf{c}(x^n)$. In light of this, we refer to the number d as an *access constraint* (or, decoding depth), since it bounds the number of encoded bits that the decoder probes before making a determination. In contrast to the fixed-blocklength settings that have been considered previously (cf. [11], [18], [20]), Definition 1 does not preclude variable-length encoding schemes. As mentioned above, this is motivated by practice, where data structures are usually of variable length and any access protocol is cognizant of the encoded data's length so that segmentation faults are avoided. Indeed, in computer file systems, a file is typically accessed after first reading metadata that describes the location and length of the file.

Note that our definition of an (r, d, n)-local decompressor does not distinguish between adaptive or non-adaptive bit probes. That is, a decompressor can probe entries of $\mathsf{c}(x^n)$ in an adaptive manner (where codeword locations are accessed sequentially, and the positions accessed can depend on the bit values observed during previous probes), or in a non-adaptive manner (where codeword locations accessed are determined only by the query index $j \in [n]$ and the codeword length $\ell(\mathsf{c}(x^n))$). When such a distinction is necessary, we will explicitly refer to adaptive and non-adaptive (r, d, n) codes.

[1]Here and throughout, $\log(x)$ denotes the base-2 logarithm of x.

III. Main Results

A. Bounds on expected blocklength

In this section, we present lower and upper bounds on the expected blocklength achievable by variable-length source codes obeying a local decodability constraint, and give sufficient conditions for them to coincide. Proof sketches can be found in Section IV.

Theorem 1. *The expected codeword length of any (r,d,n)-locally decodable code with adaptive bit-probes satisfies*

$$\mathbb{E}\left[\ell(\mathsf{c}(X^n))\right]+1 \geq \left(\frac{rd+1}{4e}\right)\left(\binom{n}{r}^{1/(rd+1)}-1\right). \quad (3)$$

Recalling the identity $\lim_{m\to\infty} m(x^{1/m}-1)=\ln x$, we note that for fixed n,r the lower bound (3) becomes

$$\lim_{d\to\infty}\left(\binom{n}{r}^{1/(rd+1)}-1\right)\left(\frac{rd+1}{4e}\right)=\frac{1}{4e}\ln\binom{n}{r}. \quad (4)$$

Hence, we recover the information-theoretic lower bound (1) (up to constant factors) in the absence of a local decodability constraint. On this note, an important consequence of Theorem 1 is that it dictates how quickly d must scale with respect to n,r in order to accommodate encoding schemes that are near-optimal in the information-theoretic sense. In the next section we quantify this tension more precisely, and establish how large d must be in order to ensure competitive optimality. Before doing this, we discuss the tightness of (3).

Theorem 2. *For any choice of r,d,n, there exists a non-adaptive (r,d,n)-locally decodable code c with average codeword length*

$$\mathbb{E}\left[\ell(\mathsf{c}(X^n))\right] \leq 30(rd+1)\left((r+1)^{(r+1)}\binom{n}{r}\right)^{1/(rd+1)}. \quad (5)$$

Two remarks are in order. First, we emphasize that Theorem 1 is a converse result for adaptive schemes, while Theorem 2 is an achievability result for non-adaptive schemes. We will see shortly that these bounds coincide (up to constant factors) in many cases, showing that adaptivity provides at most constant-factor improvement in these settings. Second, we note that both Theorem 1 and Theorem 2 are non-asymptotic in nature. That is, they hold for any choice of parameters r,d,n. However, results become most crisp when $n\to\infty$, and r,d are functions of n. As a first example, we take $n\to\infty$ and r,d fixed (i.e., not depending on n). In this case, we find that the blocklength of an optimal sequence $\{\mathsf{c}_n^\star\}$ of (r,d,n) codes scales as $\mathbb{E}\left[\ell(\mathsf{c}_n^\star(X^n))\right]=\Theta(n^{r/(rd+1)})$. Hence, when r,d are fixed, performance scales poorly relative to the information-theoretic lower bound of $\Theta(\log n)$.

As a second example, consider the setting where $r_n=n^\epsilon$ and $d_n=\delta\log n$. Then it is a straightforward calculation using (3) and (5) to see that any optimal sequence $\{\mathsf{c}_n^\star\}$ of (r_n,d_n,n)-locally decodable codes will satisfy

$$C_1(2^{(1-\epsilon)/\delta}-1) \leq \frac{\mathbb{E}\left[\ell(\mathsf{c}_n^\star(X^n))\right]}{\delta n^\epsilon\log n} \leq C_2 2^{1/\delta} \quad \text{as } n\to\infty,$$

where C_1 and C_2 are absolute constants. Thus, up to constant factors, the blocklength scaling behavior of optimal codes in this regime is $\delta n^\epsilon\log n$, and the decoder will probe a fraction of the codeword proportional to $1/r_n$ in worst case. Contrast this with the trivial encoding scheme that simply stores the position of each '1'; the natural decoder based on binary search would require roughly $\log(r_n)\cdot\log(n)$ probes in worst case.

Similarly, if we parameterize $n_m=\binom{m}{2}$, $r_m=(1+\epsilon)\frac{\ln m}{m}\binom{m}{2}$ and $d_m=\delta\log m$, then as $m\to\infty$ any optimal sequence $\{\mathsf{c}_m^\star\}$ of (r_m,d_m,n_m)-locally decodable codes will satisfy

$$C_1(2^{1/\delta}-1) \leq \frac{\mathbb{E}\left[\ell(\mathsf{c}_m^\star(X^{n_m}))\right]}{r_m d_m} \leq C_2 2^{2/\delta}. \quad (6)$$

This particular choice of parameters can be interpreted as encoding a random graph on m vertices with $(1+\epsilon)\frac{\ln m}{m}\binom{m}{2}$ edges. Since $\frac{\ln m}{m}$ is the threshold for connectivity, this graph is connected with high probability for $\epsilon>0$. Now, querying whether two vertices are connected in this graph corresponds to querying a bit of X^{n_m}. In order to accomplish this in time that grows logarithmically in the number of vertices requires average blocklength of order $rd=\Theta\left(m\log^2(m)\right)$.

In the latter two examples, average blocklength scales $r_nd_n=\Theta(\log\binom{n}{r_n})$, which is within constant factors of the information-theoretic lower bound (i.e., competitively optimal). In both cases, we chose $r_nd_n=\Omega(\log\binom{n}{r_n})$ in order to achieve this scaling. Thus, it is natural to ask: *do there exist competitively optimal schemes with $r_nd_n=o(\log\binom{n}{r_n})$?* The answer to this question is negative, and is the focus of the next section. However, before we proceed, we unify the above examples under the following straightforward corollary of Theorems 1 and 2:

Corollary 1. *If $\log\binom{n}{r_n}=\Omega(r_nd_n)$ and $d_n=\Omega(\log r_n)$, then any optimal sequence of (r_n,d_n,n)-locally decodable codes $\{\mathsf{c}_n^\star\}$ satisfies*

$$\mathbb{E}\left[\ell(\mathsf{c}_n^\star(X^n))\right]=\Theta\left(r_nd_n\binom{n}{r_n}^{1/(r_nd_n+1)}\right). \quad (7)$$

B. Local Decodability and Competitive Optimality

We now focus on the question raised at the end of the previous section, and give necessary conditions for competitive optimality (proofs can be found in the complete manuscript [21]). To this end, we define:

Definition 2. *For a sequence of integers $\{r_n\}_{n\geq1}$ a sequence of encoders*

$$\mathsf{c}_n:\binom{[n]}{r_n}\to\{0,1\}^* \qquad n\geq1 \quad (8)$$

is said to be competitively optimal *if*

$$\limsup_{n\to\infty}\frac{\mathbb{E}[\ell(\mathsf{c}_n(X^n))]}{\log\binom{n}{r_n}}=O(1). \quad (9)$$

In other words, competitively optimal schemes attain compression rates within a constant factor of the information theoretic lower bound $\log\binom{n}{r_n}$ for large enough n.

From Theorem 1, it is possible to deduce the following necessary condition for competitive optimality:

Theorem 3. *If $\{\mathsf{c}_n\}$ is a competitively optimal sequence of (r_n, d_n, n)-locally decodable codes, then $r_n d_n = \Omega\left(\log \binom{n}{r_n}\right)$.*

In other words, we cannot expect to attain competitive optimality when r_n and d_n are simultaneously small relative to the source entropy (note the contrast to the sufficient conditions in Corollary 1). This relationship can be somewhat complicated since the source entropy generally depends on both n and r_n. However, when the source sequence is modestly sparse (i.e., $r_n = O(n^{1-\epsilon})$ for some $\epsilon > 0$), then the explicit dependence on r_n in Theorem 3 can be eliminated to obtain the following condition:

Corollary 2. *If $r_n = O(n^{1-\epsilon})$ for some $\epsilon > 0$, then there exists a competitively optimal sequence of (r_n, d_n, n)-locally decodable codes if and only if $d_n = \Omega(\log n)$.*

In contrast to Corollary 2, if $r_n = \Theta(n)$, the information theoretic lower bound is $\log \binom{n}{r_n} = \Theta(n)$, and the identity encoding $\mathsf{c}_n(x^n) = x^n$ is competitively optimal, with all source bits being decodable with $d_n = 1$ probes.

IV. Proof Sketches for Main Results

Due to space constraints, we only sketch the proofs of Theorems 1 and 2. Details and proofs for Section III-B can be found in the complete manuscript [21].

Proof Sketch of Theorem 1: Our proof is inspired by that of [11, Theorem 6]. Let c be a (r, d, n)-locally decodable code. For a source sequence x^n, let $\mathsf{c}_q(x^n)$ denote the qth coordinate of the codeword $\mathsf{c}(x^n)$, and define the set

$$T_i^k \triangleq \big\{\big(q, \mathsf{c}_q(x^n_{(i)})\big) : \ell(\mathsf{c}(x^n_{(i)})) = k, \text{and location } q \text{ of } \mathsf{c}(x^n_{(i)}) \text{ is accessed to determine } x_j \text{ for some } j \in \mathsf{supp}(x^n_{(i)})\big\},$$

where we have abused notation slightly by letting x_j denote the jth coordinate of sequence $x^n_{(i)}$. Note that each T_i^k is a subset of $[k]\times\{0,1\}$ of size at most rd, since $|\mathsf{supp}(x^n_{(i)})| = r$ and the decoder makes at most d probes in response to a query. Also note that for $i \neq i'$, $T_i^k \not\subseteq T_{i'}^k$. To see this, assume the contrary, that $T_i^k \subseteq T_{i'}^k$ for some $i \neq i'$. Let the encoded source word be $x^n_{(i')}$. If we now query the value of x_j for $j \in \mathsf{supp}(x^n_{(i)}) \setminus \mathsf{supp}(x^n_{(i')})$, we see that the decoder will make an error, establishing the contradiction.

Since for fixed k, the T_i^ks are not subsets of one another, an application of the LYM inequality [23] yields

$$\#\{i : \ell(\mathsf{c}(x^n_{(i)})) = k\} \leq \max_{v \leq rd} \binom{2k}{v} \quad \text{for each } k. \tag{10}$$

In light of (10), the average codeword length must satisfy

$$\mathbb{E}\left[\ell(\mathsf{c}(X^n))\right] \geq \sum_{k=1}^{M(n,r,d)} k \frac{\max_{v\leq rd}\binom{2k}{v}}{\binom{n}{r}}, \tag{11}$$

where $M(n, r, d)$ is the largest integer satisfying

$$\sum_{k=1}^{M(n,r,d)+1} \max_{v\leq rd}\binom{2k}{v} > \binom{n}{r} \geq \sum_{k=1}^{M(n,r,d)} \max_{v\leq rd}\binom{2k}{v}. \tag{12}$$

Now define the probability distribution

$$Q(k) = \frac{\max_{v\leq rd}\binom{2k}{v}}{\binom{n}{r}} \quad \text{for } 1 \leq k \leq M(n,r,d) \tag{13}$$

and $Q(M(n,r,d)+1) = 1 - \sum_{k=1}^{M(n,r,d)} Q(k)$. Since $Q(k) \leq Q(k+1)$ for $k < M(n,r,d)$ by definition, we can conclude

$$\mathbb{E}\left[\ell(\mathsf{c}(X^n))\right] \geq \sum_{k=1}^{M(n,r,d)+1} k \cdot Q(k) \geq \frac{M(n,r,d)+1}{2}. \tag{14}$$

Toward evaluating (14), we need the following technical estimate, which is proved in the complete manuscript [21].

Lemma 1. *For all $M, v \geq 1$,*

$$\sum_{k=1}^{M} \max_{i\leq v}\binom{2k}{i} \leq 2^v \frac{\left(M + 2 + \frac{v+1}{2e}\right)^{v+1}}{(v+1)!}. \tag{15}$$

Identifying $M \leftarrow M(n,r,d)+1$ and $v \leftarrow rd$ in (15), the first inequality in (12) can be rearranged to conclude

$$M(n,r,d) + 3 \geq \left(\binom{n}{r}^{1/(rd+1)} - 1\right)\left(\frac{rd+1}{2e}\right). \tag{16}$$

Recalling (14) proves the desired inequality. ∎

Proof Sketch of Theorem 2: The proof is by a random coding argument, but it is important to note that standard typicality arguments are not applicable here since they do not support local decodability. Briefly, the idea behind our encoding scheme is to first encode some information about $\mathsf{supp}(\mathsf{c}(x^n))$ into the codeword length, and then carefully encode the remaining information so that bit x_j can be recovered by computing the binary AND of d encoded bits. A description of the codebook generation and decoding procedure is given below. Performance analysis and further details can be found in [21].

Codebook Construction: For $k = rd+1, rd+2, \ldots$ choose a subset $S_k \subseteq [n]$ of size $\frac{r}{r+1}\frac{\binom{k}{d}}{\binom{rd}{d}}$ uniformly at random from all such subsets[2]. For each $j \in S_k$, choose a subset $T_{j,k} \subseteq [k]$ of size d independently and uniformly from all such subsets. All subsets are made available to both encoder and decoder.

For a sequence $x^n \in \binom{[n]}{r}$, let $k(x^n)$ denote the smallest integer k such that the following two conditions hold:

(C1) $\mathsf{supp}(x^n) \subseteq S_k$; and

(C2) $T_{j,k} \not\subseteq \cup_{i\in\mathsf{supp}(x^n)} T_{i,k}$ for all $j \in S_k \setminus \mathsf{supp}(x^n)$.

Encoding procedure: A sequence $x^n \in \binom{[n]}{r}$ is encoded to a codeword $\mathsf{c}(x^n)$ of length $k(x^n)$ satisfying

$$\mathsf{supp}(\mathsf{c}(x^n)) = \cup_{i\in\mathsf{supp}(x^n)} T_{i,k(x^n)}. \tag{17}$$

In other words, x^n is encoded to a vector of length $k(x^n)$, which has 1's in all positions $j \in T_{i,k(x^n)}$ if and only if $x_i = 1$.

Decoding procedure: On observing the length of codeword $\mathsf{c}(x^n) = (c_1, c_2, \ldots, c_{\ell(\mathsf{c}(x^n))})$, determine bit x_j as follows:

(1) If $j \notin S_{\ell(\mathsf{c}(x^n))}$, declare $x_j = 0$; else

(2) If $j \in S_{\ell(\mathsf{c}(x^n))}$, declare $x_j = \wedge_{i\in T_{j,k(x^n)}} c_i$, where '$\wedge$' denotes binary AND.

[2]Floor and ceiling operators are omitted for clarity of presentation.

By the nature of the codebook construction, it is clear that the decoder (i) will never make an error; and (ii) satisfies the non-adaptive d-local decodability constraint. The analysis of the average codeword length is omitted due to space constraints, and can be found in [21]. ∎

V. Concluding remarks

We provided bounds for the blocklength scaling behaviour of (r, d, n) locally-decodable codes that are order-wise tight for many regimes of r, d, and n, although determining the tight constant in these bounds is still an open problem. We also showed that in contrast to the fixed blocklength setting (cf. [13]), adaptivity of probes provides no essential advantage in our regime of variable length source coding. In conclusion, we mention two variations on our main results:

A. Compression with block errors

In [18], the authors allow for vanishing block-error probability in decoding. Although we only considered error-free encodings, the proof of Theorem 1 readily extends to incorporate block-error probability as follows: Letting $\hat{x}^n$ denote the decoder's estimate of the sequence x^n given codeword $\mathsf{c}(x^n)$, the block error rate is defined to be $\Pr\{X^n \neq \hat{X}^n\}$. Now, Theorem 1 continues to hold for any (r, d, n) code with block-error rate ε by simply replacing the quantity $\binom{n}{r}$ with $(1-\varepsilon)\binom{n}{r}$. Indeed, this follows by considering only those sequences that are correctly decoded and making the same substitution in (12) in the proof of Theorem 1.

B. Connection to communication complexity

It is known that the bit-probe model has applications to asymmetric communication complexity [22]. To draw an analogous connection to our setting, consider an asymmetric communication complexity model [22] in which Alice (the user) has $i \in [n]$, Bob (the server) has $S \subset [n]$ of size r, and they wish to compute the membership function

$$f(i, S) = \begin{cases} 1 \text{ if } i \in S \\ 0 \text{ otherwise.} \end{cases} \tag{18}$$

We now enforce that the function f must be computed under a SpeedLimit paradigm, which proceeds as follows. Communication starts with Bob sending a *speed limit* message to Alice consisting of some z bits, which limits the length of any of her messages to z bits. Bob's subsequent messages consist of 1 bit. After the initial round, Alice and Bob communicate over d rounds[3] to evaluate f. The setting arises in practice where a server imposes upload bandwidth limits on users it serves (e.g., to maintain quality or fairness of service).

Note that our scheme in Theorem 2 provides a communication protocol to compute f under the SpeedLimit paradigm. Bob is essentially given a source sequence x^n, which he stores as $c(x^n)$. Alice is given the index i of the source bit that must be decoded, and must do so by making queries to Bob. Bob begins by sending $\ell(\mathsf{c}(x^n))$ to Alice, using $\log \ell(\mathsf{c}(x^n))$ bits. Alice then sends messages m_j, $j \in [d]$ of $\log \ell(c(x^n))$ bits each. In response to message m_j, Bob sends back $c_{m_j}(x^n)$. Alice then announces f to be the AND of the d bits she has received from Bob. Therefore, from Theorem 2:

Corollary 3. *There exists a deterministic communication protocol for computing the function f as in* (18) *under the* SpeedLimit *paradigm for which the speed limit z and number of communication rounds d satisfy*

$$\mathbb{E}\left[2^z\right] \leq 30(rd+1)\left((r+1)^{(r+1)}\binom{n}{r}\right)^{1/(rd+1)}. \tag{19}$$

[3] To be consistent with the rest of the paper, a communication round consists of one message by Alice and a response by Bob.

References

[1] C. E. Shannon, "A mathematical theory of communication," *Bell Sys. Tech. J.*, vol. 27, pp. 379–423, 623–656, 1948.

[2] D. A. Huffman, "A method for the construction of minimum redundancy codes," *proc. IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[3] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Inf. Thy.*, vol. 23, no. 3, pp. 337–343, 1977.

[4] D. S. Pavlichin, T. Weissman, and G. Yona, "The human genome contracts again," *Bioinformatics*, vol. 29, no. 17, pp. 2199–2202, 2013.

[5] S. Deorowicz, A. Danek, and S. Grabowski, "Genome compression: a novel approach for large collections," *Bioinformatics*, vol. 29, no. 20, pp. 2572–2578, 2013.

[6] E. F. Codd, "A relational model of data for large shared data banks," *Communications of the ACM*, vol. 13, no. 6, pp. 377–387, 1970.

[7] P. Elias and R. A. Flower, "The complexity of some simple retrieval problems," *Journal of the ACM*, vol. 22, no. 3, pp. 367–379, 1975.

[8] P. Miltersen, "The bit probe complexity measure revisited," *Proc. of the Annual Symposium on Theoretical Aspects of Computer Science*, LNCS 665, pp. 662–671, Springer-Verlag, 1993.

[9] ——, "Cell probe complexity-a survey," in *19th Conference on the Found. of Software Tech. and Theoretical Computer Science*, 1999.

[10] A. C.-C. Yao, "Should tables be sorted?" *Journal of the ACM (JACM)*, vol. 28, no. 3, pp. 615–628, 1981.

[11] H. Buhrman, P. B. Miltersen, J. Radhakrishnan, and S. Venkatesh, "Are bitvectors optimal?" *SIAM Journal on Computing*, vol. 31, no. 6, pp. 1723–1744, 2002.

[12] P. K. Nicholson, V. Raman, and S. S. Rao, "A survey of data structures in the bitprobe model," in *Space-Efficient Data Structures, Streams, and Algorithms*. Springer, 2013, pp. 303–318.

[13] N. Alon and U. Feige, "On the power of two, three and four probes," in *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 2009, pp. 346–354.

[14] J. Radhakrishnan, V. Raman, and S. Srinivasa Rao, "Explicit deterministic constructions for membership in the bitprobe model," *Lecture Notes in Computer Science*, vol. 2161, pp. 290–299, 2001.

[15] E. Viola, "Bit-probe lower bounds for succinct data structures," *SIAM Journal on Computing*, vol. 41, no. 6, pp. 1593–1604, 2012.

[16] M. Lewenstein, J. I. Munro, P. K. Nicholson, and V. Raman, "Improved explicit data structures in the bitprobe model," in *Algorithms-ESA 2014*. Springer, 2014, pp. 630–641.

[17] V. Chandar, D. Shah, and G. W. Wornell, "A locally encodable and decodable compressed data structure," in *Communication, Control, and Computing, 2009. Allerton 2009. 47th Annual Allerton Conference on*. IEEE, 2009, pp. 613–619.

[18] A. Makhdoumi, S.-L. Huang, Y. Polyanskiy, and M. Medard, "On locally decodable source coding," *arXiv preprint arXiv:1308.5239*, 2013.

[19] H. Zhou, D. Wang, and G. Wornell, "A simple class of efficient compression schemes supporting local access and editing," in *Information Theory (ISIT), 2014 IEEE International Symposium on*. IEEE, 2014, pp. 2489–2493.

[20] J. Katz and L. Trevisan, "On the efficiency of local decoding procedures for error-correcting codes," in *Proceedings of the thirty-second annual ACM symposium on Theory of computing*. ACM, 2000, pp. 80–86.

[21] A. Pananjady and T. Courtade, "Compressing sparse sequences under local decodability constraints," *arXiv preprint arXiv:1504.02063*, 2015.

[22] P. B. Miltersen, N. Nisan, S. Safra, and A. Wigderson, "On data structures and asymmetric communication complexity," in *Proceedings of the twenty-seventh annual ACM symposium on Theory of computing*. ACM, 1995, pp. 103–111.

[23] K. Yamamoto, "Logarithmic order of free distributive lattice," *Jnl. of the Mathematical Soc. of Japan*, vol. 6, no. 3-4, pp. 343–353, 1954.