Parallel Constrained Coding with Application to Two-Dimensional Constraints*

Shirley Halevy Ron M. Roth

Computer Science Department
Technion, Haifa 32000, Israel

e-mail: { shirleyh, ronny }@cs.technion.ac.il

April 9, 2002

Abstract

A parallel constrained coding scheme is considered where p-blocks of raw data are encoded simultaneously into q tracks such that the contents of each track belong to a given constraint $\mathcal{S}$. It is shown that as q increases, there are parallel block decodable encoders for $\mathcal{S}$ whose coding ratio, p/q, converges to the capacity of $\mathcal{S}$. Examples are provided where parallel coding allows block decodable encoders, while conventional coding, at the same rate, does not. Parallel encoders are then applied as building blocks in the construction of block decodable encoders for certain families of two-dimensional constraints.

Keywords: Block decodable encoders; Kronecker product; Parallel encoding; Runlength-limited constraints; Two-dimensional constraint.

1 Introduction

When sequences are recorded on a mass storage device, they typically need to belong to a certain *constrained system*. A (one-dimensional) constrained system (in short, a constraint) is defined by means of a labeled finite directed graph $G = (V, E, L)$, with a set of states

*This work was supported by grant No. 98-199 from the United-States–Israel Binational Science Foundation (BSF), Jerusalem, Israel.

V, a set of edges E, and a labeling $L : E \to \Sigma$ of the edges, where Σ is a finite alphabet. The constraint which is generated by G is the set $\mathcal{S} = \mathcal{S}(G)$ of all finite words that can be obtained by reading the labels of the edges along the finite paths in G.

An alternative definition assumes a labeling of the states rather than the edges. The definitions can be shown to be equivalent through the Moore form of G; see [18, p. 1653].

Examples of constrained systems include runlength constraints: the (d, k)-*runlength-limited (RLL) constraint* consists of all binary words in which each runlength of 0's between consecutive 1's is at least d, and no runlength of 0's exceeds k. In a *symmetric* runlength (SRLL) constraint, the runlengths of 0's, as well as the runlengths of 1's, are between d and k, except that the first and last runlengths may be shorter than d.

The study of constrained systems is mainly aimed at designing coding schemes that map arbitrary input binary sequences into words that belong to a given constraint $\mathcal{S}$. A commonly-used encoding model is that of a finite-state encoder at a fixed rate $p : q$, where the input binary sequence is divided into blocks of length p, and each such p-block is mapped, in a state-dependent manner, into a codeword of length q. The sequence of generated codewords forms a word that belongs to $\mathcal{S}$. A primary requirement from encoders is that we should be able to decode (reconstruct) the input binary sequence from the output constrained sequence. It follows from Shannon's converse-to-coding theorem [18, Theorem 3.21] that the coding ratio, p/q, is bounded from above by the capacity of $\mathcal{S}$, which is given by the limit

$$\mathsf{cap}(\mathcal{S}) = \lim_{\ell \to \infty} (1/\ell) \cdot \log_2 |\mathcal{S} \cap \Sigma^\ell| \, .$$

Of particular interest are *block decodable encoders*. Such encoders can be decoded by a *block decoder*, which maps every codeword of length q into the respective p-block, independently of the context of that codeword within the sequence of generated output codewords. Whether a block decodable encoder exists depends on the constraint $\mathcal{S}$ and on the parameters p and q. Block decodable encoders are preferable due to their simple decoding structure and their immunity against error propagation.

In this work, we explore the possibility of obtaining simple coding schemes by encoding several input streams simultaneously, i.e., *in parallel*. The coding model will still be a finite-state encoder at a fixed rate $p : q$. Yet, an input p-block will be mapped into q output symbols, each belonging to a different *track*. The sequence generated along each track will satisfy a given constraint $\mathcal{S}$. (The constraints in different tracks do not necessarily have to be the same, but we will be mainly interested here in the case where they do.) Decoding will then be carried out by reading q tracks at a time and reconstructing the respective input p-block. If the resulting encoder is block decodable, such a reconstruction requires only the knowledge of the current symbol in each track.

As we show in Section 3, there are cases where the parallel approach can allow having block decodable encoders, while such encoders, at the same rate, are prohibited in the conventional scheme where only one track is encoded (see Example 3.1). Then, in Section 4, we show that when q becomes large, the capacity of the constraint can be approached by

block decodable encoders.

Observe that in our model, the only dependency that exists between tracks is introduced by the coding itself, and not by the constraint; so, in a recording application, we do not assume any change in the channel description of each track, yet we allow to record more than one track at a time. In that regard, our setting differs from the model studied by Marcellin and Weber in [16] (see also Orcutt and Marcellin [19],[20]), where—based on physical features of the recording medium—the authors proposed relaxing the constraint along tracks by introducing dependency between them. This in effect transformed the specification of the recording channel into a *two-dimensional constraint*. Other attempts to increase the recording density have been made recently by exploiting the fact that the recording device is typically a *surface*: the recorded data is regarded as two-dimensional, as opposed to the track-oriented one-dimensional recording model. This approach dictates new types of constraints, which are two-dimensional rather than one-dimensional. For example, in optical recording [21, Ch. 3], a two-dimensional SRLL constraint with a prescribed parameter d can guarantee that any 'pit' or 'land' on the recording surface is large enough so that it can be detected from the reflection beam. See also Psaltis et al. [23] and Weeks and Blahut [25]. Two-dimensional constraints are found also in holographic memories; see Brady and Psaltis [1], Heanue, Bashaw, and Hesselink [9],[10], and Psaltis and Mok [22].

In general, a *two-dimensional constrained system* is defined by two *state*-labeled finite directed graphs, G and H, with the same set of states V and the same labeling $L : V \to \Sigma$ of the states. The two-dimensional constraint which is generated by G and H is the set $\mathcal{S} = \mathcal{S}(G, H)$ of all finite rectangular arrays $X = [x_{i,j}]$ over Σ, each of which can be associated with an array $U = U(X) = [u_{i,j}]$ over V such that the following three conditions hold: (a) $L(u_{i,j}) = x_{i,j}$ for all i and j; (b) each row in U is a path in G; and (c) each column in U is a path in H. Letting $\mathcal{S}[\ell, m]$ stand for the set of $\ell \times m$ arrays in $\mathcal{S}$, the capacity of a two-dimensional constraint is defined by

$$\mathsf{cap}(\mathcal{S}) = \lim_{\ell,m\to\infty} (1/(\ell m)) \cdot \log_2 |\mathcal{S}[\ell, m]| \; . \tag{1}$$

As in the one-dimensional case, the limit indeed exists by sub-additivity (see Burton and Steif [2], and Kato and Zeger [12]; the result in [12] is stated for the special case of runlength constraints, but the proof actually applies to all two-dimensional constraints).

In Section 5, we apply parallel encoding to show that for two-dimensional constraints that satisfy certain properties, capacity can be approached by fixed-rate block decodable encoders; that is, the decoding of a row in an $\ell \times m$ array requires only the knowledge of the current row. Our result applies in particular to the family of two-dimensional SRLL constraints.

Section 6 summarizes some properties of Kronecker powers of graphs, which are relevant to the design of parallel encoders. A short conclusion of this paper is then given in Section 7.

The next section contains a short summary of some background material from Sections 2 and 3 in [18].

2 Background

Hereafter, the term 'graph' means a finite edge-labeled directed graph and 'constraint' means a one-dimensional constraint, unless we explicitly say that the constraint is two-dimensional. Let $G = (V, E, L)$ be a graph with edge labeling $L : E \to \Sigma$. For each edge $e \in E$, we denote by $\tau(e)$ the terminal state of e in G. For a state $u \in V$, we denote by $E(u)$ the set of outgoing edges from u in G. We will sometime use the notation $u \xrightarrow{a} v$ to stand for an edge $e \in E(u)$ with $\tau(e) = v$ and $L(e) = a$. The constraint presented by G is denoted by $\mathcal{S}(G)$.

We say that a graph G is *irreducible* if for every pair of states $(u, v) \in V \times V$ there is a path from u to v in G.

A graph G is *deterministic* if for every $u \in V$, the edges in $E(u)$ are labeled distinctly.

Given nonnegative integers m and a, we say that a graph G is (m, a)-*definite* if all paths in G that generate a word

$$\boldsymbol{x} = x_{\mathsf{m}} x_{\mathsf{m}-1} \dots x_{-1} x_0 x_1 \dots x_{\mathsf{a}} \in \mathcal{S}(G) \cap \Sigma^{\mathsf{m}+\mathsf{a}+1}$$

coincide on their edge that generates x_0.

The *qth power graph* G^q is the graph with the same set of states as G, but one edge for each path of length q in G, labeled by the word (of length q) that is generated by that path.

The adjacency matrix of G is denoted by A_G: the latter is a $|V| \times |V|$ matrix whose rows and columns are indexed by V, and the (u, v)-entry, $(A_G)_{u,v}$, in A_G is the number of edges from u to v in G. By Perron-Frobenius theory, the spectral radius of A_G, denoted $\lambda(A_G)$, is an eigenvalue of A_G. As for the qth power graph, we have $A_{G^q} = (A_G)^q$ and, so, $\lambda(A_{G^q}) = (\lambda(A_G))^q$.

Let A be a nonnegative integer square matrix (such as an adjacency matrix of a graph) and n be a positive integer. An (A, n)-*approximate eigenvector* is a nonzero nonnegative integer vector $\boldsymbol{\xi}$ such that $A\boldsymbol{\xi} \geq n\boldsymbol{\xi}$, where the inequality holds component-by-component. The set of all (A, n)-approximate eigenvectors whose components are bounded from above by β is denoted by $\mathcal{X}(A, n, \beta)$. It is known that $\mathcal{X}(A, n, \infty) \neq \emptyset$ if and only if $n \leq \lambda(A)$. There is an algorithm due to Franaszek to compute an element (which is maximal in some sense) of $\mathcal{X}(A, n, \beta)$, whenever this set is nonempty [18, Section 3.1.4].

Let $G = (V, E, L)$ be an irreducible graph. A *stationary Markov chain on* G is a mapping $\mathcal{P} : E \to (0, 1]$ such that $\sum_{e \in E(u)} \mathcal{P}(e) = 1$ for every $u \in V$. The value $\mathcal{P}(e)$ is viewed as the probability of traversing an edge $e \in E(u)$ given that it is outgoing from state u. Every stationary Markov chain $\mathcal{P}$ on G has a unique *stationary probability vector* $\boldsymbol{\pi} = (\pi_u)_{u \in V}$, which satisfies

$$\sum_{u \in V} \pi_u \sum_{\substack{e \in E(u) : \\ \tau(e) = v}} \mathcal{P}(e) = \pi_v \qquad \text{for every } v \in V \tag{2}$$

(see [18, Section 3.2.3]). Note that if $\mathcal{P}$ takes rational values, then so do the components of $\boldsymbol{\pi}$.

The *entropy* $\mathsf{H}(\mathcal{P})$ of $\mathcal{P}$ is defined by

$$\mathsf{H}(\mathcal{P}) = -\sum_{u \in V} \pi_u \sum_{e \in E(u)} \mathcal{P}(e) \log_2 \mathcal{P}(e) \ .$$

The entropy satisfies $\mathsf{H}(\mathcal{P}) \leq \log_2 \lambda(A_G)$, with equality attained by the *maxentropic* stationary Markov chain on G, which is denoted by $\mathcal{P}_G$. There is a simple method for computing $\mathcal{P}_G$ and the associated stationary probability vector [18, Section 3.2.3].

Let $\mathcal{S} = \mathcal{S}(G)$ be a constraint over an alphabet Σ. A constraint $\mathcal{S}$ is irreducible if it has an irreducible graph presentation. Every (irreducible) constraint $\mathcal{S}$ has an (irreducible) deterministic graph presentation. If G is a deterministic presentation of $\mathcal{S}$, then $\mathsf{cap}(\mathcal{S}(G)) = \log_2 \lambda(A_G)$.

If $\mathcal{S}$ is a constraint presented by G, then $\mathcal{S}^q$ is the constraint presented by G^q and $\mathsf{cap}(\mathcal{S}^q) = q \cdot \mathsf{cap}(\mathcal{S})$.

Let $\mathcal{S}$ be a constraint over an alphabet Σ and n be a positive integer. An $(\mathcal{S}, n)$-*encoder* is a graph $\mathcal{E}$ such that the following three conditions hold: (i) there are n outgoing edges from each state in $\mathcal{E}$; (ii) $\mathcal{S}(\mathcal{E}) \subseteq \mathcal{S}$; and (iii) any two distinct paths with the same initial state and terminal state generate different words.

An $(\mathcal{S}, n)$-encoder exists if and only if $(\log_2 n) \leq \mathsf{cap}(\mathcal{S})$ [18, Theorems 3.20 and 3.21]. $(\mathcal{S}, n)$-encoders can be constructed by the state-splitting algorithm from a deterministic presentation G of $\mathcal{S}$ and an (A_G, n)-approximate eigenvector [18, Chapter 4].

A *tagged* $(\mathcal{S}, n)$-encoder is an encoder $\mathcal{E}$ where the outgoing edges from each state in $\mathcal{E}$ are assigned distinct *input tags* from $\Upsilon = \{0, 1, \ldots, n-1\}$. The reader is referred to [18, Section 3.3] for a description of the encoding process using tagged encoders. An encoder $\mathcal{E}$ is deterministic or (m, a)-definite if the respective property holds for the untagged graph $\mathcal{E}$.

A tagged $(\mathcal{S}, n)$-encoder $\mathcal{E}$ is (m, a)-*sliding block decodable* if there is a *decoding function* $\mathcal{D} : \Sigma^{\mathsf{m}+\mathsf{a}+1} \to \Upsilon$ such that all paths in $\mathcal{E}$ that generate a word

$$\boldsymbol{x} = x_{\mathsf{m}} x_{\mathsf{m}-1} \ldots x_{-1} x_0 x_1 \ldots x_{\mathsf{a}} \in \mathcal{S} \cap \Sigma^{\mathsf{m}+\mathsf{a}+1}$$

carry the same input tag, $\mathcal{D}(\boldsymbol{x})$, on the edge that generates x_0. An (m, a)-definite encoder is (m, a)-sliding-block decodable with respect to any tagging.

A *block decodable encoder* is a $(0, 0)$-sliding-block decodable encoder. A block decodable encoder is necessarily deterministic.

An *encoder for a constraint* $\mathcal{S}$ *at rate* $p : q$ is an $(\mathcal{S}^q, 2^p)$-encoder. Such an encoder exists if and only if its *coding ratio*, p/q, does not exceed $\mathsf{cap}(\mathcal{S})$. The *efficiency* of a rate $p : q$ encoder for $\mathcal{S}$ is defined by $(p/q)/\mathsf{cap}(\mathcal{S})$.

3 Kronecker product of constraints and graphs

Let $\mathcal{S}_1$ and $\mathcal{S}_2$ be two constraints over alphabets Σ_1 and Σ_2, respectively. The *Kronecker product* of $\mathcal{S}_1$ and $\mathcal{S}_2$, denoted $\mathcal{S}_1 \otimes \mathcal{S}_2$, is the set of all column words

$$\begin{matrix} x_{1,1} & x_{1,2} \\ x_{2,1} & x_{2,2} \\ \vdots & \vdots \\ x_{\ell,1} & x_{\ell,2} \end{matrix}$$

over the alphabet $\Sigma_1 \times \Sigma_2$ such that $x_{1,i}x_{2,i}\ldots x_{\ell,i} \in \mathcal{S}_i$ for $i = 1, 2$. (While words in a constraint are usually written as row words, it will be more convenient to regard the elements in a Kronecker product as column words.) The notation $\mathcal{S}^{\otimes q}$ will stand for $\mathcal{S} \otimes \mathcal{S} \otimes \cdots \otimes \mathcal{S}$ (q times).

Let $G_1 = (V_1, E_1, L_1)$ and $G_2 = (V_2, E_2, L_2)$ be two graphs. The *Kronecker product* of G_1 and G_2, denoted $G_1 \otimes G_2$, is a graph defined over the set of states

$$V_1 \times V_2 = \left\{ \langle y_1\ y_2 \rangle \ : \ y_1 \in V_1, y_2 \in V_2 \right\},$$

and

$$\langle y_1\ y_2 \rangle \xrightarrow{a_1 a_2} \langle z_1\ z_2 \rangle$$

is an edge in $G_1 \otimes G_2$ whenever $y_1 \xrightarrow{a_1} z_1$ and $y_2 \xrightarrow{a_2} z_2$ are edges in G_1 and G_2, respectively. The adjacency matrix of $G_1 \otimes G_2$ is the Kronecker product $A_{G_1} \otimes A_{G_2}$ and, therefore,

$$\lambda(A_{G_1 \otimes G_2}) = \lambda(A_{G_1}) \cdot \lambda(A_{G_2})$$

(see [14, p. 84]). One can easily verify that $\mathcal{S}(G_1 \otimes G_2) = \mathcal{S}(G_1) \otimes \mathcal{S}(G_2)$.

Given a graph $G = (V, E, L)$ with labeling $L : E \to \Sigma$, we will use the notation $G^{\otimes q}$ for $G \otimes G \otimes \cdots \otimes G$ (q times); the set of states of $G^{\otimes q}$ is given by $V^q = V \times V \times \cdots \times V$, and each edge of $G^{\otimes q}$ is labeled by an element of $\Sigma^q = \Sigma \times \Sigma \times \cdots \times \Sigma$. Clearly, $(\mathcal{S}(G))^{\otimes q} = \mathcal{S}(G^{\otimes q})$ and $A_{G^{\otimes q}} = A_G^{\otimes q} = A_G \otimes A_G \otimes \cdots \otimes A_G$. One can view the graph $G^{\otimes q}$ as generating q (column-)tracks, in parallel, of the constraint $\mathcal{S}(G)$.

The *profile* of an element $\langle y_i \rangle_{i=1}^q$ in V^q is defined as the integer vector $(r_u)_{u \in V}$, where for every $u \in V$,

$$|\{i \ : \ y_i = u\}| = r_u \ .$$

That is, every state $u \in V$ appears exactly r_u times among the components of $\langle y_i \rangle_{i=1}^q$. Clearly, $\sum_{u \in V} r_u = q$.

Techniques—such as the state-splitting algorithm— for constructing encoders from graph presentations can be applied to $G^{\otimes q}$ to obtain q-track parallel encoders for $\mathcal{S}(G)$.

Example 3.1 Let $\mathcal{S}$ be the constraint presented by a graph G with distinctly-labeled edges and an adjacency matrix

$$A_G = \begin{pmatrix} 9 & 3 & 0 \\ 0 & 9 & 8 \\ 5 & 0 & 6 \end{pmatrix} .$$

In this case $(\lambda(A_G))^2 > 2^7$ and, so, there is an $(\mathcal{S}^2, 2^7)$-encoder (i.e., a rate $7:2$ finite-state encoder for $\mathcal{S}$), and there is also an $(\mathcal{S}^{\otimes 2}, 2^7)$-encoder. The efficiency of each encoder is $(7/2)/\log_2 \lambda(A_G) \approx 0.943$.

Next we check whether those encoders, when tagged, can be made block decodable. Recall that every block decodable encoder must be deterministic; namely, edges outgoing from the same state are distinctly labeled. Conversely, every deterministic encoder whose edges are distinctly labeled can be tagged so that it is block decodable.

Given a deterministic graph H, there exists a deterministic $(\mathcal{S}(H), n)$-encoder if and only if $\mathcal{X}(A_H, n, 1) \neq \emptyset$ [18, Theorem 6.17]. Taking $H = G^2$, we find by Franaszek's algorithm that $\mathcal{X}(A_G^2, n, 1) \neq \emptyset$ if and only if $n \leq 126$. Therefore, there is no deterministic $(\mathcal{S}^2, 2^7)$-encoder. On the other hand, the vector

$$(1\ 1\ 1\ 1\ 0\ 1\ 1\ 1\ 0)^\top$$

is an $(A_G^{\otimes 2}, n)$-approximate eigenvector whenever $n \leq 132$. Since the edges of $G^{\otimes 2}$ are distinctly labeled, it follows that there is a block decodable $(\mathcal{S}^{\otimes 2}, 2^7)$-encoder. □

4 Parallel block decodable encoders

Let $G = (V, E, L)$ be an irreducible deterministic graph and let $\mathcal{P} : E \to (0, 1]$ be a rational stationary Markov chain on G with an associated rational stationary probability vector $\boldsymbol{\pi} = (\pi_u)_{u \in V}$. Denote by ν the smallest positive integer such that the values $\nu \pi_u \mathcal{P}(e)$ are integers for all $u \in V$ and $e \in E(u)$. Let q be a multiple of ν and define the integers $q_u = q\pi_u$ and $q_e = q_u \mathcal{P}(e)$ for every $u \in V$ and $e \in E(u)$.

Next, consider the following subgraph, $G_q = G_q(\mathcal{P})$, of $G^{\otimes q}$: the states of G_q are all the elements in V^q whose profile is $(q_u)_{u \in V}$, and

$$\langle y_1\ y_2\ \ldots\ y_q \rangle \overset{a_1 a_2 \ldots a_q}{\longrightarrow} \langle z_1\ z_2\ \ldots\ z_q \rangle$$

is an edge in G_q if and only if

$$|\{i\ :\ (y_i \overset{a_i}{\to} z_i) = e\}| = q_e \quad \text{for every edge } e \in E\ . \tag{3}$$

The requirement $\sum_{e \in E(u)} \mathcal{P}(e) = 1$ implies $\sum_{e \in E(u)} q_e = q_u$. Furthermore, from (2) we have for every $v \in V$,

$$\sum_{\substack{e \in E\ : \\ \tau(e) = v}} q_e \ = \ \sum_{u \in V} \sum_{\substack{e \in E(u)\ : \\ \tau(e) = v}} q\pi_u \mathcal{P}(e) \ = \ q\pi_v \ = \ q_v\ ;$$

therefore, if $\langle y_i \rangle_{i=1}^q$ is a state in G_q and $\langle z_i \rangle_{i=1}^q$ is an element of V^q that satisfies (3), then the profile of that element must be $(q_u)_{u \in V}$ and, as such, it must be a state in G_q. It follows that all states in G_q have the same out-degree, $n_q = n_q(\mathcal{P})$, which is given by the following product of multinomial coefficients

$$n_q = \prod_{u \in V} \frac{q_u!}{\prod_{e \in E(u)} (q_e!)} = \frac{\prod_{u \in V} (q_u!)}{\prod_{e \in E} (q_e!)} \ . \tag{4}$$

The next lemma follows from known approximations for multinomial coefficients [15, p. 309]. We include a proof for the sake of completeness.

Lemma 4.1

$$\frac{\log_2 n_q}{q} = \mathsf{H}(\mathcal{P}) + O\Big((|E| \log q)/q \Big) \ .$$

Proof. By the Stirling formula we have

$$\log_2(t!) = t \log_2(t/e) + O(\log t) \ ,$$

where e is the base of natural logarithms. Hence,

$$\log_2 n_q = \sum_u \Big(q_u \log_2 q_u - \sum_{e \in E(u)} q_e \log_2 q_e \Big) + O(|E| \log q) \ .$$

Finally, substitute $q_u = q\pi_u$ and $q_e = q_u \mathcal{P}(e)$. □

Lemma 4.2 *Let $\mathcal{S}$ be a constraint and $\mathcal{E} = (V, E, L)$ be a deterministic $(\mathcal{S}, n)$-encoder where $|V|n > 2$. Then there exists a block decodable $(\mathcal{S}, n')$-encoder where*

$$n' \geq \left\lfloor \frac{n}{\log_e (|V|n)} \right\rfloor \ .$$

Proof. Write $n' = \lfloor n/t \rfloor$ where $t = \log_e (|V|n)$ and denote by Υ' the set $\{0, 1, \ldots, n'-1\}$. Let Σ be the alphabet of $\mathcal{S}$ and assume a uniform probability distribution over the ensemble of all the functions of the form $\mathcal{D} : \Sigma \to \Upsilon'$. Select randomly such a function and assign accordingly input tags from Υ' to the edges of $\mathcal{E}$. In fact, the term 'input tag' is unjustified at this point, since from each state there will be outgoing edges that carry the same input tag. Hence, we delete a minimal number of edges from $\mathcal{E}$ to form a graph $\mathcal{E}'$ where no two outgoing edges from the same state are tagged the same. Yet, there may still be states in $\mathcal{E}'$ whose out-degree is less than n'. We next bound from above the probability of this event, assuming the uniform probability distribution over the ensemble of $\mathcal{D}$.

The probability that a given input tag of Υ' will not be assigned to any of the outgoing edges of a given state in $\mathcal{E}$ is $(1 - (1/n'))^n$. Hence, the probability of having at least one

input tag of Υ' missing from the outgoing edges of at least one state in $\mathcal{E}$ is bounded from above by

$$|V| \cdot n' \cdot (1 - (1/n'))^n < |V| \cdot n \cdot (1 - (1/n'))^{n't} < |V| \cdot n \cdot e^{-t} = 1 \ .$$

It follows that with strictly positive probability, the selected function $\mathcal{D}$ is such that $\mathcal{E}'$ is an $(\mathcal{S}, n')$-encoder. Such an encoder is necessarily block decodable. □

The following is the main result of this section.

Theorem 4.3 *Let $\mathcal{S}$ be a constraint. There is an infinite sequence of tagged encoders $\{\mathcal{E}(i)\}_{i=1}^{\infty}$ such that each $\mathcal{E}(i)$ is a block decodable $(\mathcal{S}^{\otimes q(i)}, n(i))$-encoder and*

$$\lim_{i \to \infty} \frac{\log_2 n(i)}{q(i)} = \mathsf{cap}(\mathcal{S}) \ .$$

Proof. Let $G = (V, E, L)$ be an irreducible deterministic presentation of a constraint $\mathcal{S}(G) \subseteq \mathcal{S}$ such that $\mathsf{cap}(\mathcal{S}(G)) = \mathsf{cap}(\mathcal{S}) = \log_2 \lambda(A_G)$; such a graph G always exists [18, p. 1674]. Let $\mathcal{P}_G$ be the maxentropic stationary Markov chain on G and, for a given $\epsilon > 0$, let $\mathcal{P}$ be a rational stationary Markov chain on G such that $\mathsf{H}(\mathcal{P}) > \mathsf{H}(\mathcal{P}_G) - \epsilon$. We construct a deterministic graph $G_q(\mathcal{P})$ with out-degree $n_q = n_q(\mathcal{P})$ in each state where, using Lemma 4.1, we select q to be large enough so that

$$\frac{\log_2 n_q}{q} > \mathsf{H}(\mathcal{P}) - \epsilon > \mathsf{H}(\mathcal{P}_G) - 2\epsilon \ .$$

On the other hand, $n_q \leq |\Sigma|^q$, where Σ is the alphabet of $\mathcal{S}$. Recalling that the number of states of $G_q(\mathcal{P})$ is bounded from above by $|V|^q$, it follows from Lemma 4.2 that there exists a block decodable $(\mathcal{S}^{\otimes q}, n'_q)$-encoder where

$$\frac{\log_2 n'_q}{q} \geq \frac{\log_2 n_q}{q} - \frac{\log_2 \log_{\mathrm{e}} \left((|V| \cdot |\Sigma|)^q + O(1) \right)}{q} \geq \frac{\log_2 n_q}{q} - \epsilon \ ,$$

where the last inequality holds for sufficiently large q. The result follows by letting ϵ go to zero and recalling that $\mathsf{H}(\mathcal{P}_G) = \log_2 \lambda(A_G) = \mathsf{cap}(\mathcal{S})$. □

We mention that there is a known counterpart of Theorem 4.3 for $(\mathcal{S}^q, n(i))$-encoders (see Theorem 3.22 in [18]); in fact, the latter encoders are *block encoders*, i.e., they have only one state.

5 Application to two-dimensional constraints

Next we turn to an application of parallel encoding to two-dimensional constraints.

Let $\mathcal{S} = \mathcal{S}(G, H)$ be a two-dimensional constraint, where G and H are finite state-labeled directed graphs with labeling over an alphabet Σ. We denote by $\mathcal{S}[\ell, m]$ the set of all $\ell \times m$ arrays in $\mathcal{S}$, where the case $\ell = 0$ (respectively, $m = 0$) corresponds to the set which consists of one 'empty' $0 \times m$ (respectively, $\ell \times 0$) array. We will also use the notations

$$\mathcal{S}[\ell, \cdot] = \bigcup_{m=0}^{\infty} \mathcal{S}[\ell, m]$$

and

$$\mathcal{S}[\cdot, m] = \bigcup_{\ell=0}^{\infty} \mathcal{S}[\ell, m] \ .$$

So, in the latter case, $\mathcal{S}[\cdot, m]$ consists of all arrays in $\mathcal{S}$ of width m. We can regard each row in such an array as an element of the super-alphabet Σ^m, in which case $\mathcal{S}[\cdot, m]$ becomes in effect a one-dimensional constraint, and we will treat it as such (e.g., we will speak about $(\mathcal{S}[\cdot, m], n)$-encoders). By the definition (1) of capacity it follows that

$$\lim_{m \to \infty} \frac{\mathsf{cap}(\mathcal{S}[\cdot, m])}{m} = \mathsf{cap}(\mathcal{S}) \ .$$

In a two-dimensional setting, we will be interested in investigating encoders that encode into elements of $\mathcal{S}[\cdot, m]$, row-by-row, at coding ratios that approach $\mathsf{cap}(\mathcal{S})$ as m goes to infinity. Specifically, we will consider an infinite sequence of graphs $\{\mathcal{E}_j\}_{j=1}^{\infty}$, where each $\mathcal{E}_j$ is an $(\mathcal{S}[\cdot, m_j], n_j)$-encoder and the coding ratio, $(\log_2 n_j)/m_j$, satisfies

$$\lim_{j \to \infty} \frac{\log_2 n_j}{m_j} = \mathsf{cap}(\mathcal{S}) \ .$$

The additional sought property from each $\mathcal{E}_j$ is that it be block decodable: given an array $X \in \mathcal{S}[\cdot, m_j]$ that was generated by the encoder, the recovery of the ith input tag (over an alphabet of size n_j) requires only the knowledge of row i of X.

A two-dimensional constraint $\mathcal{S} = \mathcal{S}(G, H)$ is called *horizontally primitive* if there exist a nonnegative integer B and a *merging function*

$$f : \bigcup_{\ell=0}^{\infty} \Big(\mathcal{S}[\ell, \cdot] \times \mathcal{S}[\ell, \cdot] \Big) \to \mathcal{S}[\cdot, B] \ ,$$

such that for every two arrays $W_1, W_2 \in \mathcal{S}[\ell, \cdot]$, the image $X = f(W_1, W_2)$ is in $\mathcal{S}[\ell, B]$ and

$$W_1 X W_2 \in \mathcal{S}[\ell, \cdot] \ .$$

The $\ell \times B$ array X is called a *merging array* of W_1 and W_2, and the parameter B will be referred to as the *merging width* of f.

For an array $W \in \mathcal{S}$ with at least one row, denote by W^* the array obtained by deleting the last row in W.

Let f be a merging function of a horizontally-primitive two-dimensional constraint $\mathcal{S}$. We say that f is *causal* if

$$(f(W_1, W_2))^* = f(W_1^*, W_2^*)$$

for every $(W_1, W_2) \in \bigcup_{\ell=1}^{\infty} \Big(\mathcal{S}[\ell, \cdot] \times \mathcal{S}[\ell, \cdot] \Big)$. That is, for every two given arrays W_1 and W_2 with ℓ rows and for every $i \leq \ell$, the ith row of the merging array $f(W_1, W_2)$ depends only on the first i rows of W_1 and W_2. A horizontally-primitive two-dimensional constraint is called causal if it has a causal merging function.

We present below several examples of causal horizontally-primitive two-dimensional constraints. (In Example 5.1, we also identify two state-labeled graphs G and H that define the constraint. Such a description can easily be obtained for each of the other examples.)

Example 5.1 Fix $k_{\rm r}$ and $k_{\rm c}$ to be positive integers and consider the set $\mathcal{S}$ of binary arrays in which each row satisfies the $(0, k_{\rm r})$-RLL constraint and each column satisfies the $(0, k_{\rm c})$-RLL constraint.

(The set $\mathcal{S}$ is a two-dimensional constraint that can be defined through state-labeled graphs G and H as follows. The states of both graphs are all the binary $k_{\rm c} \times k_{\rm r}$ arrays, and the label of each state is the lower-right (say) bit in that array. For every $(k_{\rm c}{+}1) \times k_{\rm r}$ array Y in $\mathcal{S}$, we endow H with an edge $Y^* \to Z$, where Z consists of the last $k_{\rm c}$ rows of Y. The construction of G is similar.)

For the two-dimensional constraint $\mathcal{S}$, we can take $B = 1$ with $f(W_1, W_2)$ being an all-one column. □

Example 5.2 Fix $d_{\rm r}$ and $d_{\rm c}$ to be positive integers and consider the set of binary arrays where each row satisfies the $(d_{\rm r}, \infty)$-RLL constraint and each column satisfies the $(d_{\rm c}, \infty)$-RLL constraint. Here we can select $B = d_{\rm r}$ and $f(W_1, W_2)$ can be taken as the all-zero array. □

Example 5.3 Let $d_{\rm r}$, $k_{\rm r}$, $d_{\rm c}$, and $k_{\rm c}$ be nonnegative integers such that $k_{\rm r} > d_{\rm r}$ and $k_{\rm c} > d_{\rm c}$, and consider the set $\mathcal{S}$ of binary arrays where each row satisfies the $(d_{\rm r}, k_{\rm r})$-SRLL constraint and each column satisfies the $(d_{\rm c}, k_{\rm c})$-SRLL constraint.

Next, we modify the proof of Etzion in [6] (see also [7]) and show that $\mathcal{S}$ is causal horizontally-primitive by exhibiting a causal merging function f with a merging width $B = (3d_{\rm r} - 2)(d_{\rm r} + 1)$. The values $f(W_1, W_2)$ for pairs $(W_1, W_2) \in \mathcal{S}[\ell, \cdot] \times \mathcal{S}[\ell, \cdot]$ will be defined recursively over ℓ. The recursion base corresponds to empty arrays and is therefore trivial.

Let W_1 and W_2 be two arrays in $\mathcal{S}[\ell, \cdot]$ where $\ell \geq 1$, and assume that f has already been defined for (W_1^*, W_2^*). The first $\ell{-}1$ rows of $f(W_1, W_2)$ are given by

$$(f(W_1, W_2))^* = f(W_1^*, W_2^*) \; .$$

It remains to specify the ℓth row of $f(W_1, W_2)$.

Let $\overline{b}$ be the inverse of the binary value b (i.e., $\overline{0}=1$ and $\overline{1}=0$). Denote by $\boldsymbol{x}_0$ the word that is formed by the last $d_{\rm r}+1$ entries in the ℓth row in W_1, and by $\boldsymbol{x}_{3d_{\rm r}-1}$ the word formed by the first $d_{\rm r}+1$ entries in the ℓth row in W_2 (if the ℓth row of W_1—or W_2—is shorter than $d_{\rm r}+1$, then just extend it to that length by adding preceding—or trailing—bits while maintaining the constraint along that row). Write the ℓth row of $f(W_1,W_2)$ in the form $\boldsymbol{x}_1\boldsymbol{x}_2\ldots\boldsymbol{x}_{3d_{\rm r}-2}$, where each block $\boldsymbol{x}_i$ has length $d_{\rm r}+1$. Let $x_{i,1}x_{i,2}\ldots x_{i,d_{\rm r}+1}$ be the components of $\boldsymbol{x}_i$. For $1\le i<2d_{\rm r}$, the blocks $\boldsymbol{x}_i$ are defined recursively by

$$\boldsymbol{x}_i=\begin{cases}\overline{x}_{i-1,2}\overline{x}_{i-1,3}\ldots\overline{x}_{i-1,d_{\rm r}}\overline{x}_{i-1,d_{\rm r}+1}\;\overline{x}_{i-1,d_{\rm r}+1} & \text{if } i\neq d_{\rm r}\\ \overline{x}_{i-1,2}\overline{x}_{i-1,3}\ldots\overline{x}_{i-1,d_{\rm r}}\overline{x}_{i-1,d_{\rm r}+1}\;\overline{x}_{3d_{\rm r}-1,1} & \text{if } i=d_{\rm r}\end{cases},$$

and for $2d_{\rm r}\le i<3d_{\rm r}-1$, the blocks $\boldsymbol{x}_i$ are defined by the backward recursion

$$\boldsymbol{x}_i=\overline{x}_{i+1,1}\;\overline{x}_{i+1,1}\overline{x}_{i+1,2}\overline{x}_{i+1,3}\ldots\overline{x}_{i+1,d_{\rm r}}\;.$$

We next show that the word $\boldsymbol{x}=\boldsymbol{x}_0\boldsymbol{x}_1\ldots\boldsymbol{x}_{3d_{\rm r}-1}$ belongs to the one-dimensional $(d_{\rm r},d_{\rm r}{+}1)$-SRLL constraint.

Consider first the blocks $\boldsymbol{x}_i$ for $i=0,1,\ldots,d_{\rm r}{-}1$. The block $\boldsymbol{x}_0$ belongs to the $(d_{\rm r},d_{\rm r}{+}1)$-SRLL constraint. Also, for $i>0$, the first $d_{\rm r}$ bits in $\boldsymbol{x}_i$ are the inverse of the last $d_{\rm r}$ bits in $\boldsymbol{x}_{i-1}$, while the last two bits in $\boldsymbol{x}_i$ are identical. It follows that for $i=0,1,\ldots,d_{\rm r}{-}1$, each block $\boldsymbol{x}_i$ has a 'switch-point' index $j_i\in\{1,2,\ldots,d_{\rm r}{+}1\}$ such that $x_{i,j}=\overline{x}_{i,j_i}$ when $j<j_i$ and $x_{i,j}=x_{i,j_i}$ when $j\ge j_i$. Furthermore, the index j_i satisfies the recursion

$$j_i=\begin{cases}j_{i-1}-1 & \text{if } j_{i-1}>1\\ 1 & \text{otherwise}\end{cases},\qquad i=1,2,\ldots,d_{\rm r}{-}1\;.$$

We thus conclude that the word $\boldsymbol{x}_0\boldsymbol{x}_1\ldots\boldsymbol{x}_{d_{\rm r}-1}$ belongs to the $(d_{\rm r},d_{\rm r}{+}1)$-SRLL constraint and that all the bits in $\boldsymbol{x}_{d_{\rm r}-1}$ are equal. A similar argument implies that $\boldsymbol{x}_{d_{\rm r}}\boldsymbol{x}_{d_{\rm r}+1}\ldots\boldsymbol{x}_{2d_{\rm r}-1}$ belongs to this constraint and that all the bits in $\boldsymbol{x}_{2d_{\rm r}-1}$ are equal. And the same applies—now by backward induction—to $\boldsymbol{x}_{2d_{\rm r}}\boldsymbol{x}_{2d_{\rm r}+1}\ldots\boldsymbol{x}_{3d_{\rm r}-1}$, except that here the first block, $\boldsymbol{x}_{2d_{\rm r}}$, is the one in which the bits are equal.

Hence, in order to establish that $\boldsymbol{x}$ belongs to the $(d_{\rm r},d_{\rm r}{+}1)$-SRLL constraint, it remains to show that this constraint contains the words $\boldsymbol{x}_{d_{\rm r}-1}\boldsymbol{x}_{d_{\rm r}}$ and $\boldsymbol{x}_{2d_{\rm r}-1}\boldsymbol{x}_{2d_{\rm r}}$. Now, this definitely holds for the former word since, by construction, the first $d_{\rm r}$ bits in $\boldsymbol{x}_{d_{\rm r}}$ are the inverse of the last $d_{\rm r}$ bits in $\boldsymbol{x}_{d_{\rm r}-1}$. As for the word $\boldsymbol{x}_{2d_{\rm r}-1}\boldsymbol{x}_{2d_{\rm r}}$, recall that each of its sub-blocks, $\boldsymbol{x}_{2d_{\rm r}-1}$ and $\boldsymbol{x}_{2d_{\rm r}}$, is either all-zero or all-one. On the other hand, by induction on i we have

$$\overline{x}_{d_{\rm r}+i,d_{\rm r}+1}=x_{3d_{\rm r}-1-i,1}=\begin{cases}x_{3d_{\rm r}-1,1} & \text{if } i \text{ is even}\\ \overline{x}_{3d_{\rm r}-1,1} & \text{if } i \text{ is odd}\end{cases},\qquad i=0,1,\ldots,d_{\rm r}{-}1\;.$$

Therefore, $\boldsymbol{x}_{2d_{\rm r}}=\overline{\boldsymbol{x}}_{2d_{\rm r}-1}$ and, so, $\boldsymbol{x}$ belongs to the $(d_{\rm r},d_{\rm r}{+}1)$-SRLL constraint.

Writing $X=f(W_1,W_2)$, we conclude that the ℓth row in W_1XW_2 belongs to the $(d_{\rm r},k_{\rm r})$-SRLL constraint. Turning to the columns of X, it follows by induction on ℓ that each column in X is either a column in W_1W_2 or its inverse; as such, it belongs to the $(d_{\rm c},k_{\rm c})$-SRLL constraint. □

Example 5.4 Let $\mathcal{S}$ be the set of all binary arrays $X = [x_{i,j}]$ in which

$$x_{i,j} = 1 \quad \Longrightarrow \quad x_{i,j+1} = x_{i+1,j-1} = x_{i+1,j+1} = 0$$

(whenever the indexes do not exceed the array boundaries); that is, no two 1's are adjacent either horizontally or along any of the diagonals. We can select here $B = 1$, with $f(W_1, W_2)$ taken as an all-zero column.

The two-dimensional constraint $\mathcal{S}$ was studied by Cohn in [4], motivated by the following recording application. Each row in an array $X \subseteq \mathcal{S}$ represents a valid recording of a (one-dimensional) track, while the different rows indicate the contents of the *same* track after every rewrite on that track; that is, the row index is in fact a *time* index. It is required that the contents of the track satisfy the (one-dimensional) $(1, \infty)$-RLL constraint and, in addition, no two adjacent bits will be altered during one rewrite phase.

The capacity of $\mathcal{S}$ was shown by Golin et al. in [8] to be at least 0.535 and, using a modification of the Engel-Calkin-Wilf technique [3],[5], it can be shown that this bound is tight up to (at least) the third decimal place.

For the mentioned application, the recorded data needs to be encoded so that each row can be decoded without the knowledge of any previous or subsequent rows: such rows are simply unavailable when data is read from the track (this is the case of 'encoder informed, decoder uninformed' as defined by Wolf et al. in [26]). The row-by-row tagged encoder must therefore be block decodable.

It will follow from the next result that as the row (i.e., track) length increases, there exist block decodable encoders for $\mathcal{S}$ with coding ratios approaching capacity. As pointed out in [4], there is a very simple block decodable encoder at rate $1 : 2$ for $\mathcal{S}$: when writing data on the track, put a 0 after each input bit. □

The following is the main result of this section.

Theorem 5.1 *Let $\mathcal{S}$ be a causal horizontally-primitive two-dimensional constraint. There is an infinite sequence of tagged encoders $\{\mathcal{E}_j\}_{j=1}^{\infty}$ such that each $\mathcal{E}_j$ is a block decodable $(\mathcal{S}[\cdot, m_j], n_j)$-encoder and*

$$\lim_{j\to\infty} \frac{\log_2 n_j}{m_j} = \mathsf{cap}(\mathcal{S}) \ .$$

Proof. Fix r to be a positive integer. It follows from Theorem 4.3 that there exists an infinite sequence of graphs $\{\mathcal{E}(i,r)\}_{i=1}^{\infty}$ such that each $\mathcal{E}(i,r)$ is a block decodable $((\mathcal{S}[\cdot, r])^{\otimes q(i,r)}, n(i,r))$-encoder and

$$\lim_{i\to\infty} \frac{\log_2 n(i,r)}{q(i,r)} = \mathsf{cap}(\mathcal{S}[\cdot, r]) \ .$$

Fix now also i and let $\mathcal{E} = \mathcal{E}(i,r)$, $q = q(i,r)$, and $n = n(i,r)$. Let B be the merging width of a causal merging function f of $\mathcal{S}$, and define the constant $m = m(i,r) = (B+r)q - B$. Also, let $G[m]$ be a graph presentation of $\mathcal{S}[\cdot, m]$. For a nonempty subset Γ of states of $G[m]$, let $\mathcal{A}(\Gamma)$ denote the set of all arrays

$$Y = W_1 X_1 W_2 X_2 \dots X_{q-1} W_q$$

in $\mathcal{S}[\cdot, m]$ that satisfy the following conditions:

1. $W_j \in \mathcal{S}[\cdot, r]$ and $W_1 W_2 \dots W_q \in \mathcal{S}(\mathcal{E})$;
2. $X_j = f(W_1 X_1 W_2 X_2 \dots X_{j-1} W_j, W_{j+1})$ for $j = 1, 2, \dots, q-1$; and—
3. Γ is the set of all terminal states of paths in $G[m]$ that generate Y.

We next construct a graph $\mathcal{E}'$ as follows. The states of $\mathcal{E}'$ are all the subsets Γ of states of $G[m]$ such that $\mathcal{A}(\Gamma) \neq \emptyset$; for each state Γ, we designate a particular element $Y = Y(\Gamma)$ in $\mathcal{A}(\Gamma)$. The edges of $\mathcal{E}'$ are labeled by elements of $\mathcal{S}[1, m]$, and we endow $\mathcal{E}'$ with an edge

$$\Gamma \xrightarrow{\boldsymbol{x}} \Gamma' \tag{5}$$

if and only if there is $Z \in \mathcal{A}(\Gamma')$ whose last row is $\boldsymbol{x}$ and $Z^* = Y(\Gamma)$; so, if

$$Z = W_1 X_1 W_2 X_2 \dots X_{q-1} W_q$$

then

$$Y(\Gamma) = W_1^* X_1^* W_2^* X_2^* \dots X_{q-1}^* W_q^* \ ,$$

where $X_j^* = f(W_1^* X_1^* W_2^* X_2^* \dots X_{j-1}^* W_j^* \ , \ W_{j+1}^*)$ for $j = 1, 2, \dots, q-1$.

The graph $\mathcal{E}'$ is a subgraph of the 'subset construction' of a deterministic presentation of $\mathcal{S}[\cdot, m]$ obtained from $G[m]$ [13, Theorem 3.3.2]. Hence, $\mathcal{E}'$ is deterministic and $\mathcal{S}(\mathcal{E}') \subseteq \mathcal{S}[\cdot, m]$. Also, since f is a causal merging function of $\mathcal{S}$, the out-degree of each state in $\mathcal{E}'$ is the out-degree, n, of each state in $\mathcal{E}$. Hence, $\mathcal{E}'$ is a deterministic $(\mathcal{S}[\cdot, m], n)$-encoder.

Let $\mathcal{D} : (\mathcal{S}[1, r])^{\otimes q} \to \{0, 1, \dots, n-1\}$ be a decoding function for $\mathcal{E}$, and write the label $\boldsymbol{x}$ of each edge (5) as

$$\boldsymbol{x} = \boldsymbol{w}_1 \boldsymbol{x}_1 \boldsymbol{w}_2 \boldsymbol{x}_2 \dots \boldsymbol{x}_{q-1} \boldsymbol{w}_q \ ,$$

where $\boldsymbol{w}_j \in \mathcal{S}[1, r]$ and $\boldsymbol{x}_j \in \mathcal{S}[1, B]$. We now define a function $\mathcal{D}'$ over the set of edge labels of $\mathcal{E}'$ by

$$\mathcal{D}'(\boldsymbol{x}) = \mathcal{D}(\boldsymbol{w}_1 \boldsymbol{w}_2 \dots \boldsymbol{w}_q) \ .$$

One can readily verify that $\mathcal{D}'$ is a decoding function for $\mathcal{E}'$. Hence, $\mathcal{E}'$ is a block decodable $(\mathcal{S}[\cdot, m(i,r)], n(i,r))$-encoder.

Taking now the limit over i, we obtain

$$\lim_{i \to \infty} \frac{\log_2 n(i,r)}{m(i,r)} = \lim_{i \to \infty} \frac{\log_2 n(i,r)}{(B+r)q(i,r)} = \frac{\mathsf{cap}(\mathcal{S}[\cdot, r])}{B+r} \ ,$$

and taking the limit also over r yields

$$\lim_{r\to\infty}\lim_{i\to\infty}\frac{\log_2 n(i,r)}{m(i,r)} = \lim_{r\to\infty}\frac{\mathsf{cap}(\mathcal{S}[\cdot,r])}{B+r} = \lim_{r\to\infty}\frac{r}{B+r}\cdot\frac{\mathsf{cap}(\mathcal{S}[\cdot,r])}{r} = \mathsf{cap}(\mathcal{S})\ .$$

This completes the proof. □

Example 5.5 Let $\mathcal{S}$ be the two-dimensional constraint in Example 5.4, The (one-dimensional) constraint $\mathcal{S}[\cdot,r]$ for $r=3$ is presented by the deterministic graph $G=(V,E,L)$ in Figure 1.

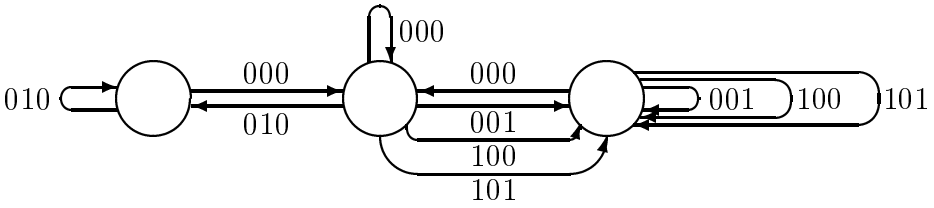

Figure 1: Graph G for Example 5.5.

The adjacency matrix of G is given by

$$A_G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 3 \\ 0 & 1 & 3 \end{pmatrix}\ ,$$

and $\mathsf{cap}(\mathcal{S}[\cdot,3]) = \log_2\lambda(A_G) \approx 2.0307$.

Let Q be the stochastic matrix

$$Q = \frac{1}{17}\begin{pmatrix} 5 & 12 & 0 \\ 1 & 4 & 12 \\ 0 & 5 & 12 \end{pmatrix}\ ,$$

and define the stationary Markov chain $\mathcal{P}: E\to(0,1]$ on G by

$$\mathcal{P}(e) = \frac{(Q)_{u,\tau(e)}}{(A_G)_{u,\tau(e)}} \qquad \text{for every } u\in V \text{ and } e\in E(u)\ .$$

The respective stationary probability vector is given by

$$\boldsymbol{\pi} = \tfrac{1}{209}\cdot(5\ \ 60\ \ 144)$$

(which is a left eigenvector of Q that is associated with the eigenvalue 1). The entropy of $\mathcal{P}$ is approximately 2.0270 (which is within 0.2% of the capacity of $\mathcal{S}[\cdot,3]$).

Next, we construct the graph $G_q = G_q(\mathcal{P})$ for $q = 17 \cdot 209 = 3,553$, as in Section 4. The states of G_q are all the elements in V^q whose profile is

$$q \cdot \boldsymbol{\pi} = 17 \cdot (5 \;\; 60 \;\; 144) = (85 \;\; 1,020 \;\; 2,448) \, ,$$

and the out-degree of each state in G_q is given by

$$n_q = \frac{85! \cdot 1,020! \cdot 2,448!}{25! \cdot (60!)^2 \cdot (240!)^4 \cdot 720! \cdot (576!)^3} > 2^{7,162}$$

(see (4)). The graph G_q is a deterministic $((\mathcal{S}[\cdot, r])^{\otimes q}, n_q)$-encoder. We regard each output of G_q as an array of width $rq = 3 \cdot 3,553 = 10,659$ over the alphabet $\{0, 1\}$.

(If we followed the proofs of Theorems 4.3 and 5.1, we would transform G_q at this stage into a block decodable encoder by random tagging and deleting excess edges. Here we defer the tagging until after the next step.)

Recall that the merging width of $\mathcal{S}$ is $B = 1$ with a merging function that maps to the all-zero column. We select $m = (B + r)q - B = 14,211$ and obtain a deterministic $(\mathcal{S}[\cdot, m], n_q)$-encoder $\mathcal{E}$ by putting a 0 after every $r = 3$ bits in each row of any array that is generated by G_q. In fact, since G is $(1, 0)$-definite, then so are G_q and $\mathcal{E}$; therefore, $\mathcal{E}$ is $(1, 0)$-sliding-block decodable with respect to every tagging of its edges.

Next, we apply Lemma 4.2 to $\mathcal{E}$ and conclude that there is a block decodable $(\mathcal{S}[\cdot, m], n'_q)$-encoder $\mathcal{E}'$ whose coding ratio satisfies

$$\frac{\log_2 n'_q}{m} \geq \frac{\log_2 \lfloor n_q / \log_e (|V|^q n_q) \rfloor}{m} \geq \frac{\log_2 \lfloor 2^{7,162} / \log_e (3^{3,553} \cdot 2^{7,162}) \rfloor}{14,211} \geq \frac{7,148}{14,211} \approx 0.5030 \, .$$

This is higher than the coding ratio of the simple encoder that just puts a 0 between any two adjacent input bits in each row. Yet, we did need to encode into rather wide arrays in order to achieve this improvement in the coding ratio. While enumerative coding techniques [11, Chapter 6] could yield a tractable implementation of the $(1, 0)$-definite encoder $\mathcal{E}$, it still remains open how one could implement efficiently the block decodable encoder $\mathcal{E}'$, given the unstructured tagging of the latter.

Indeed, the convergence to capacity in Theorem 5.1 can be rather slow: using Franaszek's algorithm, we have verified that there exist deterministic $(\mathcal{S}[\cdot, m], n)$-encoders for $m \leq 21$ if and only if $n \leq 2^{\lceil m/2 \rceil}$ (and the maximum degree is attained by putting a 0 between any two adjacent input bits). □

6 Properties of Kronecker powers of graphs

Let $\mathcal{S}$ be a one-dimensional constraint that is presented by a deterministic graph $G = (V, E, L)$. The design of $(\mathcal{S}^{(\otimes q)}, n)$-encoders through the state-splitting algorithm makes use of an $(A_G^{\otimes q}, n)$-approximate eigenvector. As $A_G^{\otimes q}$ has order $|V|^q \times |V|^q$, the design can be made simpler if smaller matrices or graphs are used instead. In Section 6.1 and 6.2 we show that sometimes the building blocks of the code design can indeed be made smaller.

6.1 Irreducible components of $G^{\otimes q}$

It follows from [18, Lemma 2.8] that when designing an $(\mathcal{S}^{\otimes q}, n)$-encoder for a constraint $\mathcal{S} = \mathcal{S}(G)$, it suffices to consider only the constraints that are generated by irreducible components of $G^{\otimes q}$ (see definition in [18, Section 2.5.1]). Indeed, as we show next, the graph $G^{\otimes q}$ can be reducible even when G is irreducible.

Let G be an irreducible graph with at least one edge (such a graph must contain a cycle). The *period* of G is the greatest common divisor, p, of all the lengths of the cycles in G. The graph G is called *primitive* if $\mathsf{p} = 1$. Two states u and v in G are *congruent* if there is a path from u to v whose length is divisible by p. Congruence is an equivalence relation that induces p equivalence classes, $C_0, C_1, \ldots C_{\mathsf{p}-1}$, on the set of states of G, and the length of each path from any state in C_j to any state in C_m is congruent to $m-j$ modulo p. Furthermore, there is a positive integer ℓ_0 such that for every $u \in C_j$ and $v \in C_m$ and for every $\ell \geq \ell_0$ there is a path from u to v of length $\ell\mathsf{p} + m - j$ [18, Section 3.1.2].

For a state v in G, let $C_{\jmath(v)}$ be the equivalence class that contains v. It is easy to verify by induction on the length of paths in $G^{\otimes q}$ that for all states $\langle y_i \rangle_{i=1}^{q}$ along a given path in $G^{\otimes q}$, the $q-1$ values

$$\jmath(y_i) - \jmath(y_1) \ , \quad i = 2, 3, \ldots, q \ ,$$

when taken modulo p, are invariant. This, in turn, implies that $G^{\otimes q}$ consists of p^{q-1} isolated subgraphs: each subgraph corresponds to a list $(m_2, m_3, \ldots, m_q) \in \{0, 1, \ldots, \mathsf{p}-1\}^{q-1}$ and consists of all states

$$\langle y_i \rangle_{i=1}^{q} \in \bigcup_{j=0}^{\mathsf{p}-1} \left(C_j \times C_{j+m_2} \times C_{j+m_3} \times \cdots \times C_{j+m_q} \right)$$

with their incident edges in $G^{\otimes q}$ (the indexes $j + m_i$ are taken modulo p). Furthermore, each such subgraph is irreducible with period p. It thus follows that $G^{\otimes q}$ is irreducible if and only if $q = 1$ or $\mathsf{p} = 1$.

6.2 $(A_G^{\otimes q}, n)$-approximate eigenvectors

Let G be a graph and A be the adjacency matrix of G. In what follows, we relate the elements of $\mathcal{X}(A^{\otimes q}, n, \beta)$ to those of $\mathcal{X}(A^{[q]}, n, \beta)$, where $A^{[q]}$ is a matrix of order $\binom{q+|V|-1}{q} \times \binom{q+|V|-1}{q}$. Note that this order is smaller than that of $A^{\otimes q}$ whenever $q > 1$ and $|V| > 1$; for instance, when $|V| = 2$, the matrix $A^{[q]}$ has order $(q+1) \times (q+1)$, while $A^{\otimes q}$ has order $2^q \times 2^q$.

Let $\mathcal{L}(V, q)$ denote the set of all lists $r = (r_u)_{u \in V}$ of nonnegative integers such that $\sum_{u \in V} r_u = q$; note that $\mathcal{L}(V, q)$ consists of all possible profiles of elements of V^q and that $|\mathcal{L}(V, q)| = \binom{q+|V|-1}{q}$. For $r \in \mathcal{L}(V, q)$, we let $\Phi(r)$ be the set of elements of V^q whose profile is r.

Let $\boldsymbol{x} = (x_u)_{u \in V}$ be a list of $|V|$ indeterminates and for every $r \in \mathcal{L}(V, q)$, define the

multivariate polynomial

$$g_r(\boldsymbol{x}) = \prod_{u \in V} \Big(\sum_{v \in V} (A)_{u,v} x_v \Big)^{r_u} \ . \tag{6}$$

The polynomial g_r is homogeneous with total degree q and can be written in the form

$$g_r(\boldsymbol{x}) = \sum_{s \in \mathcal{L}(V,q)} g_{r,s} \prod_{u \in V} x_u^{s_u} \ , \tag{7}$$

where $g_{r,s}$ are nonnegative integer coefficients and s_u is the component of s that is indexed by u.

It follows from (6) and (7) that if y is an arbitrary element of $\Phi(r)$, then for every $r, s \in \mathcal{L}(V,q)$,

$$g_{r,s} = \sum_{z \in \Phi(s)} \prod_{i=1}^{q} (A)_{y_i,z_i} \ ,$$

where y_i and z_i stand for the ith component in y and z, respectively. Observe that when the rows and columns of $A^{\otimes q}$ are indexed by elements of V^q, we have

$$g_{r,s} = \sum_{z \in \Phi(s)} (A^{\otimes q})_{y,z} \tag{8}$$

for every $y \in \Phi(r)$.

Next, we define the $\binom{q+|V|-1}{q} \times \binom{q+|V|-1}{q}$ matrix $A^{[q]}$ by

$$A^{[q]} = \Big(g_{r,s} \Big)_{r,s \in \mathcal{L}(V,q)} \ .$$

This matrix is called in [14, pp. 85–86] the *qth power-matrix of* A, but we will refrain from using this term here to avoid confusion with the (ordinary) qth power of A, which is A^q.

Example 6.1 Let

$$A = \begin{pmatrix} 0 & 8 \\ 13 & 1 \end{pmatrix}$$

and $q = 3$. Assuming in this case that $V = \{1, 2\}$ and $\boldsymbol{x} = (x_1, x_2)$, we have $\mathcal{L}(V,3) = \{(3,0), (2,1), (1,2), (0,3)\}$,

$$g_{(3-i,i)}(x_1, x_2) = (8x_2)^{3-i}(13x_1 + x_2)^i \ , \quad i = 0, 1, 2, 3 \ ,$$

and

$$A^{[3]} = \begin{pmatrix} 0 & 0 & 0 & 512 \\ 0 & 0 & 832 & 64 \\ 0 & 1352 & 208 & 8 \\ 2197 & 507 & 39 & 1 \end{pmatrix} \ .$$

Note that $A^{\otimes 3}$ is an 8×8 matrix in this case. □

Proposition 6.1 *Let A be a nonnegative integer square matrix (i.e., an adjacency matrix of a graph) and let q, n, and β be positive integers. Then $\mathcal{X}(A^{\otimes q}, n, \beta) \neq \emptyset$ if and only if $\mathcal{X}(A^{[q]}, n, \beta) \neq \emptyset$.*

Proof. Suppose that $\boldsymbol{\xi} = (\xi_y)_{y \in V^q}$ is an element in the (nonempty) set $\mathcal{X}(A^{\otimes q}, n, \beta)$ and define the vector $\boldsymbol{\mu} = (\mu_r)_{r \in \mathcal{L}(V,q)}$ by

$$\mu_r = \max_{y \in \Phi(r)} \xi_y \ , \quad r \in \mathcal{L}(V, q) \ .$$

We show that $\boldsymbol{\mu} \in \mathcal{X}(A^{[q]}, n, \beta)$.

Given $r \in \mathcal{L}(V, q)$, let $y \in \Phi(r)$ be such that $\mu_r = \xi_y$. Then,

$$(A^{[q]}\boldsymbol{\mu})_r = \sum_{s \in \mathcal{L}(V,q)} (A^{[q]})_{r,s} \mu_s = \sum_{s \in \mathcal{L}(V,q)} \mu_s \sum_{z \in \Phi(s)} (A^{\otimes q})_{y,z} \ ,$$

where the second equality follows from (8). Recalling that $\mu_s \geq \xi_z$ for every $z \in \Phi(s)$, it follows that

$$(A^{[q]}\boldsymbol{\mu})_r \geq \sum_{s \in \mathcal{L}(V,q)} \sum_{z \in \Phi(s)} (A^{\otimes q})_{y,z} \xi_z = \sum_{z \in V^q} (A^{\otimes q})_{y,z} \xi_z = (A^{\otimes q}\boldsymbol{\xi})_y \geq n\xi_y = n\mu_r \ .$$

Hence, $\boldsymbol{\mu} \in \mathcal{X}(A^{[q]}, n, \beta)$.

Conversely, suppose that $\boldsymbol{\mu} = (\mu_r)_{r \in \mathcal{L}(V,q)}$ is in $\mathcal{X}(A^{[q]}, n, \beta)$ and define the vector $\boldsymbol{\xi} = (\xi_y)_{y \in V^q}$ by

$$\xi_y = \mu_r \ , \quad y \in V^q \ ,$$

where $r = r(y)$ is the profile of y. For every $y \in V^q$ with profile r we have,

$$(A^{\otimes q}\boldsymbol{\xi})_y = \sum_{z \in V^q} (A^{\otimes q})_{y,z} \xi_z = \sum_{s \in \mathcal{L}(V,q)} \mu_s \sum_{z \in \Phi(s)} (A^{\otimes q})_{y,z} = (A^{[q]}\boldsymbol{\mu})_r \geq n\mu_r = n\xi_y \ .$$

Therefore, $\boldsymbol{\xi} \in \mathcal{X}(A^{\otimes q}, n, \beta)$. □

The proof of Proposition 6.1 in fact shows that whenever the set $\mathcal{X}(A_G^{\otimes q}, n, \beta)$ is nonempty, it contains an $(A_G^{\otimes q}, n)$-approximate eigenvector whose components are identical at locations that are indexed by states of $G^{\otimes q}$ with the same profile. Recall that outgoing edges in $G^{\otimes q}$ from states with the same profile terminate in states with the same profile. It thus follows that when using such an approximate eigenvector in the state-splitting algorithm, states with the same profile in $G^{\otimes q}$ can be split the same way, thereby resulting in an encoder with a 'uniform' structure. (On the other hand, such a restriction on the splitting can potentially result in an encoder that is inferior in other respects, e.g., possibly having a larger decoding delay [24].)

Example 6.2 Let the constraint $\mathcal{S}$ be presented by a graph G with distinctly-labeled edges and let the adjacency matrix of G be given by the matrix A in Example 6.1. One can verify that

$$\mathsf{cap}(\mathcal{S}) = \log_2 \lambda(A_G) = \log_2\bigl((1+\sqrt{417})/2\bigr) \approx 3.421 \ .$$

So, there is an $(\mathcal{S}^3, 2^{10})$-encoder and also an $(\mathcal{S}^{\otimes 3}, 2^{10})$-encoder. The efficiency of such encoders is $(10/3)/\log_2 \lambda(A_G) \approx 0.974$.

Using Franaszek's algorithm, we find that $\mathcal{X}(A_G^{[3]}, 2^{10}, 1)$—and,therefore, $\mathcal{X}(A_G^{\otimes 3}, 2^{10}, 1)$—is empty. This, in turn, implies that there is no deterministic $(\mathcal{S}^{\otimes 3}, 2^{10})$-encoder [18, Theorem 6.17]. On the other hand, the set $\mathcal{X}(A_G^{[3]}, 2^{10}, 2)$ contains the vector $(1\ 0\ 0\ 2)^\top$ and, so, $\mathcal{X}(A_G^{\otimes 3}, 2^{10}, 2)$ contains the vector

$$(1\ 0\ 0\ 0\ 0\ 0\ 0\ 2)^\top$$

(that is, the only nonzero components of this vector are indexed by states $\langle 1\ 1\ 1\rangle$ and $\langle 2\ 2\ 2\rangle$ of $G^{\otimes 3}$, and the respective values are 1 and 2). Hence, an $(\mathcal{S}^{\otimes 3}, 2^{10})$-encoder with three states can be obtained by applying one round of state splitting to $G^{\otimes 3}$. Such an encoder is $(0,1)$-definite and, therefore, it is $(0,1)$-sliding-block decodable for every tagging of its edges.

We mention that in this example, the sum of the (two) components of every $(A_G^3, 2^{10})$-approximate eigenvector is at least 5, and this sum is attained by the approximate eigenvector $(2\ 3)^\top$. Using this vector, a $(0,1)$-definite $(\mathcal{S}^3, 2^{10})$-encoder with five states can be obtained by applying one round of state splitting to G^3. It follows from Corollary 2 in [17] that no $(\mathcal{S}^3, 2^{10})$-encoder can have less than five states. □

6.3 Limitations

Examples 3.1 and 6.2 demonstrate that there are cases where $\mathcal{X}(A_G^q, n, \beta)$ is empty while $\mathcal{X}(A_G^{\otimes q}, n, \beta)$ is not. We do not have yet a characterization of all cases where this occurs. In the next proposition we show that this cannot happen if G has two states and $\beta = 1$.

Proposition 6.2 *Let A be a nonnegative integer 2×2 matrix. Then, for every positive integers n and q,*

$$\mathcal{X}(A^q, n, 1) = \emptyset \quad \Longrightarrow \quad \mathcal{X}(A^{[q]}, n, 1) = \emptyset \ .$$

Proof. Write

$$A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$$

where, without loss of generality, we assume that $a + b \le c + d$; otherwise, reverse the order of the rows and columns in A. To maintain simple notations, we index the rows and columns of $A^{[q]}$ by the integers $0, 1, \ldots, q$, where—using the notations in (6)–(7)—

$$(A^{[q]})_{i,j} = g_{(q-i,i),(q-j,j)} \quad \text{for every } 0 \le i, j \le q \ . \tag{9}$$

We extend the definition of $A^{[q]}$ in (6)–(7) also to $q = 0$ so that $A^{[0]}$ is the 1×1 matrix $\begin{pmatrix} 1 \end{pmatrix}$. It follows from (6), (7), and (9) that for every $0 \leq i \leq q$ and every integer j,

$$\begin{pmatrix} (A^{[q+1]})_{i,j} \\ (A^{[q+1]})_{i+1,j} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} (A^{[q]})_{i,j} \\ (A^{[q]})_{i,j-1} \end{pmatrix} , \tag{10}$$

where we assume that $(A^{[q]})_{i,j} = 0$ when $j < 0$ or $j > q$.

For every $0 \leq i \leq q$ and j define

$$t^{[q]}_{i,j} = \sum_{\ell \geq j} (A^{[q]})_{i,\ell} .$$

By (10) we obtain that the values $t^{[q]}_{i,j}$ satisfy for $q \geq 0$ the recurrence

$$\begin{pmatrix} t^{[q+1]}_{i,j} \\ t^{[q+1]}_{i+1,j} \end{pmatrix} = A \begin{pmatrix} t^{[q]}_{i,j} \\ t^{[q]}_{i,j-1} \end{pmatrix} \quad \text{for every } 0 \leq i \leq q \text{ and } j . \tag{11}$$

Suppose that $\mathcal{X}(A^{[q]}, n, 1)$ is nonempty and let $\boldsymbol{\mu} = (\mu_u)_{u=0}^{q}$ be a vector in $\mathcal{X}(A^{[q]}, n, 1)$. We show that $\mathcal{X}(A^q, n, 1)$ is nonempty by distinguishing between two cases.

Case 1: $a + b \geq d$. We show that the vector $\mathbf{1} = (1\ 1)^\top$ is in $\mathcal{X}(A^q, n, 1)$ by first proving, through induction on q, that

$$A^q \mathbf{1} \geq \begin{pmatrix} t^{[q]}_{i,i} \\ t^{[q]}_{i,i-1} \end{pmatrix} \quad \text{for every } 0 \leq i \leq q . \tag{12}$$

Starting with $q = 0$, the matrix $A^q = A^0$ is the identity matrix and, so,

$$A^0 \mathbf{1} = \mathbf{1} = \begin{pmatrix} t^{[0]}_{0,0} \\ t^{[0]}_{0,-1} \end{pmatrix} .$$

Turning to the induction step, we apply the induction hypothesis and (11) to obtain

$$A^{q+1} \mathbf{1} = A(A^q \mathbf{1}) \geq A \begin{pmatrix} t^{[q]}_{i,i} \\ t^{[q]}_{i,i-1} \end{pmatrix} = \begin{pmatrix} t^{[q+1]}_{i,i} \\ t^{[q+1]}_{i+1,i} \end{pmatrix} \quad \text{for every } 0 \leq i \leq q . \tag{13}$$

Now, since $c + d \geq a + b$ we have

$$A\,\mathbf{1} \geq (a + b)\,\mathbf{1} ,$$

which, in turn, implies that

$$A^{q+1} \mathbf{1} \geq (a + b)^{q+1} \mathbf{1} .$$

Recalling that $a + b \geq d$, we obtain

$$A^{q+1} \mathbf{1} \geq \begin{pmatrix} d^{q+1} \\ (a + b)^{q+1} \end{pmatrix} = \begin{pmatrix} t^{[q+1]}_{q+1,q+1} \\ t^{[q+1]}_{0,-1} \end{pmatrix} .$$

Combining the latter inequality with (13), we conclude that the two components of $A^{q+1}\mathbf{1}$ satisfy

$$(A^{q+1}\mathbf{1})_1 \geq t_{i,i}^{[q+1]} \quad \text{and} \quad (A^{q+1}\mathbf{1})_2 \geq t_{i,i-1}^{[q+1]} \quad \text{for every } 0 \leq i \leq q+1 \ ,$$

thereby proving (12).

Having established (12), we let i be the smallest index u for which $\mu_u \neq 0$. From the inequality $A^{[q]}\boldsymbol{\mu} \geq n\boldsymbol{\mu}$ we obtain

$$t_{i,i}^{[q]} \geq (A^{[q]}\boldsymbol{\mu})_i \geq n\mu_i = n \ , \tag{14}$$

and noting that $t_{i,i-1}^{[q]} \geq t_{i,i}^{[q]}$, it follow from (12) that

$$A^q\mathbf{1} \geq t_{i,i}^{[q]}\,\mathbf{1} \geq n\,\mathbf{1} \ ,$$

thus completing the proof of Case 1.

Case 2: $a + b < d$. We show that $\boldsymbol{e} = (0\ 1)^\top$ is in $\mathcal{X}(A^q, n, 1)$ by proving first that

$$A^q\boldsymbol{e} \geq \begin{pmatrix} t_{i,i+1}^{[q]} \\ t_{i,i}^{[q]} \end{pmatrix} \quad \text{for every } 0 \leq i \leq q \ . \tag{15}$$

The proof is, again, by induction on q, with the induction base

$$A^0\boldsymbol{e} = \boldsymbol{e} = \begin{pmatrix} t_{0,1}^{[0]} \\ t_{0,0}^{[0]} \end{pmatrix} \ .$$

As for the induction step, from (11) we have

$$A^{q+1}\boldsymbol{e} = A(A^q\boldsymbol{e}) \geq A\begin{pmatrix} t_{i,i+1}^{[q]} \\ t_{i,i}^{[q]} \end{pmatrix} = \begin{pmatrix} t_{i,i+1}^{[q+1]} \\ t_{i+1,i+1}^{[q+1]} \end{pmatrix} \quad \text{for every } 0 \leq i \leq q \ . \tag{16}$$

Furthermore, since $A\,\boldsymbol{e} \geq d\,\boldsymbol{e}$, it follows that $A^{q+1}\boldsymbol{e} \geq d^{q+1}\boldsymbol{e}$, which, in turn, implies that

$$A^{q+1}\boldsymbol{e} \geq \begin{pmatrix} 0 \\ (a+b)^{q+1} \end{pmatrix} = \begin{pmatrix} t_{q+1,q+2}^{[q+1]} \\ t_{0,0}^{[q+1]} \end{pmatrix} \ .$$

The latter inequality, when combined with (16), completes the induction proof of (15).

Finally, we let i be the smallest index u for which $\mu_u \neq 0$. From (14) and (15) we obtain

$$A^q\boldsymbol{e} \geq t_{i,i}^{[q]}\,\boldsymbol{e} \geq n\,\boldsymbol{e} \ ,$$

as claimed. □

Example 3.1 shows that Proposition 6.2 is false when A has order 3×3, and by padding all-zero rows and columns one can obtain a counter-example for any larger matrix order. Proposition 6.2 becomes false also when we attempt to generalize it to sets $\mathcal{X}(A^q, n, \beta)$ and $\mathcal{X}(A^{[q]}, n, \beta)$ where $\beta \geq 2$ and q is an odd integer greater than 1. We show this in our next result.

Proposition 6.3 *For every odd integer $q > 1$ and every positive integer β there is an all-positive integer 2×2 matrix A and a positive integer n such that $\mathcal{X}(A^{[q]}, n, 2) \neq \emptyset$ while $\mathcal{X}(A^q, n, \beta) = \emptyset$.*

Proof. Let b and c be integers greater than 1 and let the matrix A and the integer n be given by

$$A = A(b,c) = \begin{pmatrix} 1 & b \\ c & 1 \end{pmatrix}$$

and

$$n = n(b,c) = \min\{2b^q, \lfloor c^q/2 \rfloor\} \ .$$

The upper-right entry in $A^{[q]}$ equals b^q and the lower-left entry is c^q. Hence, the vector

$$(1\ 0\ 0\ \ldots\ 0\ 2)^\top$$

belongs to $\mathcal{X}(A^{[q]}, n, 2)$.

We now turn to the matrix A^q. Letting $\alpha = \sqrt{bc}$, we observe that the eigenvalues of A are $1 \pm \alpha$ and accordingly write

$$A^q = \frac{1}{2\alpha} \cdot \begin{pmatrix} \sqrt{b} & -\sqrt{b} \\ \sqrt{c} & \sqrt{c} \end{pmatrix} \begin{pmatrix} (1+\alpha)^q & 0 \\ 0 & (1-\alpha)^q \end{pmatrix} \begin{pmatrix} \sqrt{c} & \sqrt{b} \\ -\sqrt{c} & \sqrt{b} \end{pmatrix} \ .$$

Recalling that q is odd, we obtain

$$A^q = \alpha^q \begin{pmatrix} \epsilon & \sqrt{b/c} \\ \sqrt{c/b} & \epsilon \end{pmatrix} \ ,$$

where ϵ goes to zero as α goes to infinity. For every (A^q, n)-approximate eigenvector $(\xi_1\ \xi_2)^\top$ we thus have

$$\begin{pmatrix} \epsilon - (n/\alpha^q) & \sqrt{b/c} \\ \sqrt{c/b} & \epsilon - (n/\alpha^q) \end{pmatrix} \begin{pmatrix} \xi_1 \\ \xi_2 \end{pmatrix} \geq \mathbf{0} \ ,$$

which can be written when $n/\alpha^q > \epsilon$ as

$$\sqrt{c/b} \cdot \left((n/\alpha^q) - \epsilon \right) \leq \frac{\xi_2}{\xi_1} \leq \frac{\sqrt{c/b}}{(n/\alpha^q) - \epsilon} \ .$$

Now, let b and c grow so that the ratio c/b approaches the (irrational) value $\sqrt[q]{4}$. The ratio n/α^q then approaches 1 and ϵ approaches zero. Hence,

$$\lim_{c/b \to \sqrt[q]{4}} (\xi_2/\xi_1) = \sqrt{c/b} = \sqrt[q]{2} \ ,$$

thereby implying that in the limit, the components of $(\xi_1\ \xi_2)^\top$ grow without bound. We conclude that for every $\beta \geq 1$ there is a matrix $A = A(b,c)$ and an integer $n = n(b,c)$ such that $\mathcal{X}(A^q, n, \beta)$ is empty while $\mathcal{X}(A^{[q]}, n, 2)$ is not. $\square$

We conjecture that if q is even, then for every nonnegative integer 2×2 matrix A and every positive integers n and β,

$$\mathcal{X}(A^q, n, \beta) = \emptyset \quad \Longrightarrow \quad \mathcal{X}(A^{[q]}, n, \beta) = \emptyset \ .$$

The next example presents another case where the set $\mathcal{X}(A_G^{\otimes q}, n, \beta)$ is empty whenever $\mathcal{X}(A_G^q, n, \beta)$ is.

Example 6.3 Let G be a primitive graph and q a positive integer such that $(\lambda(A_G))^q$ is an integer. Suppose that n is selected to be $(\lambda(A_G))^q$. Recall that here both G^q and $G^{\otimes q}$ are primitive.

It follows from Theorems 3.1 and 3.6 in [18] that all (A_G^q, n)-approximate eigenvectors are scalar multiples of the unique (up to scaling) right eigenvector of A_G^q that is associated with the eigenvalue n. Denote by $\boldsymbol{\xi} = (\xi_u)_{u \in V}$ the (A_G^q, n)-approximate eigenvector whose largest component,

$$\xi_{\max} = \max_{u \in V} \xi_u \ ,$$

is the smallest among all (A_G^q, n)-approximate eigenvectors. Then,

$$\mathcal{X}(A_G^q, n, \beta) = \emptyset \quad \Longleftrightarrow \quad \beta < \xi_{\max} \ . \tag{17}$$

On the other hand, $\boldsymbol{\xi}$ is also the unique (up to scaling) right eigenvector of A_G that is associated with the eigenvalue $\lambda(A_G)$. It follows from Theorem 43.3 in [14] that

$$\boldsymbol{\xi}^{\otimes q} = \boldsymbol{\xi} \otimes \boldsymbol{\xi} \otimes \cdots \otimes \boldsymbol{\xi} \quad (q \text{ times})$$

is a right eigenvector of $A^{\otimes q}$ that is associated with the eigenvalue n. Furthermore, by Theorem 3.6 in [18], every $(A_G^{\otimes q}, n)$-approximate eigenvector $\boldsymbol{\mu}$ is a scalar multiple of $\boldsymbol{\xi}^{\otimes q}$. Observing that for all $u \in V$, the integers ξ_u^q—which are all components of $\boldsymbol{\xi}^{\otimes q}$—do not have a common integer factor, it follows that $\boldsymbol{\mu}$ is an *integer* multiple of $\boldsymbol{\xi}^{\otimes q}$. Hence,

$$\mathcal{X}(A_G^{\otimes q}, n, \beta) = \emptyset \quad \Longleftrightarrow \quad \beta < \xi_{\max}^q \ . \tag{18}$$

By (17) and (18) we get that if $\mathcal{X}(A_G^q, n, \beta)$ is empty, then so is $\mathcal{X}(A_G^{\otimes q}, n, \beta)$. □

7 Conclusion

In this paper, we introduced the concept of parallel encoding of (one-dimensional) constraints. We showed by example that there are instances of constraints and rates for which the parallel approach allows having block decodable encoders, while the conventional encoding model does not. We then applied parallel encoding to show that for certain families of two-dimensional constraints—including the family of two-dimensional SRLL constraints—one can approach capacity by encoders whose decoders can reconstruct any row within an encoded array without the knowledge of previous or subsequent rows. We ended by a discussion

on some of the tools that were used in the encoder constructions: we presented properties of Kronecker powers of graphs and derived conditions on the existence of $(A^{\otimes q}, n)$-approximate eigenvectors for nonnegative integer square matrices A.

References

[1] D. Brady, D. Psaltis, *Control of volume holograms, J. Opt. Soc. Am. A,* 9 (1992), 1167–1182.

[2] R. Burton, J.E. Steif, *Non-uniqueness of measures of maximal entropy for subshifts of finite type, Ergod. Th. Dynam. Sys.,* 14 (1994), 213–235.

[3] N. Calkin, H.S. Wilf, *The number of independent sets in a grid graph, SIAM J. Discrete Math.,* 11 (1997), 54–60.

[4] M. Cohn, *On the channel capacity of read/write isolated memory, Discrete Applied Math.,* 56 (1995), 1–8.

[5] K. Engel, *On the Fibonacci number of an $m \times n$ lattice, Fibonacci Quarterly,* 28 (1990), 72–78.

[6] T. Etzion, *Cascading methods for runlength-limited arrays, IEEE Trans. Inform. Theory,* 43 (1997), 319–324

[7] T. Etzion, V.K. Wei, *On two-dimensional runlength-limited codes, IEEE Int'l Workshop on Inform. Theory,* Salvador, Brazil (1991).

[8] M. Golin, X.R. Yong, Y.P. Zhang, L. Sheng, *New upper and lower bounds on the channel capacity of read/write isolated memory, Proc. IEEE Symp. Inform. Theory,* Sorrento, Italy (2000), p. 280.

[9] J.F. Heanue, M.C. Bashaw, L. Hesselink, *Volume holographic storage and retrieval of digital data, Science,* 265 (1994), 749–752.

[10] J.F. Heanue, M.C. Bashaw, L. Hesselink, *Channel codes for digital holographic data storage, J. Opt. Soc. Am. A,* 12 (1995).

[11] K.A.S. Immink, *Codes for Mass Data Storage Systems,* Shannon Foundation Publishers, The Netherlands, 1999.

[12] A. Kato, K. Zeger, *On the capacity of two-dimensional run length constrained channels, IEEE Trans. Inform. Theory,* 45 (1999), 1527–1540.

[13] D. Lind, B. Marcus, *An Introduction to Symbolic Dynamics and Coding,* Cambridge University Press, Cambridge, UK, 1995.

[14] C.C. MacDuffee, *The Theory of Matrices,* Chelsea, New York, 1946.

[15] F.J. MacWilliams, N.J.A. Sloane, *The Theory of Error-Correcting Codes,* North-Holland, Amsterdam, 1977.

[16] M.W. Marcellin, H.J. Weber, *Two-dimensional modulation codes, IEEE J. Select. Areas Commun.,* 10 (1992), 254–266.

[17] B.H. Marcus, R.M. Roth, *Bounds on the number of states in encoders graphs for input-constrained channels, IEEE Trans. Inform. Theory,* 37 (1991), 742–758.

[18] B.H. Marcus, R.M. Roth, P.H. Siegel, *Constrained systems and coding for recording channels,* in *Handbook of Coding Theory,* V.S. Pless and W.C. Huffman (Editors), Elsevier, Amsterdam, 1998, pp. 1635–1764.

[19] E.K. Orcutt, M.W. Marcellin, *Enumerable multi-track (d, k) block codes, IEEE Trans. Inform. Theory,* 39 (1993), 1738–1744.

[20] E.K. Orcutt, M.W. Marcellin, *Redundant multi-track (d, k) codes, IEEE Trans. Inform. Theory,* 39 (1993), 1744–1750.

[21] K.C. Pohlmann, *The Compact Disc Handbook,* Second Edition, A–R Editions, Madison, Wisconsin, 1992.

[22] D. Psaltis, F. Mok, *Holographic Memories, Scientific American,* 273, No. 5 (1995), 70–76.

[23] D. Psaltis, M.A. Neifeld, A. Yamamura, S. Kobayashi, *Optical memory disks in optical information processing, Applied Optics,* 29 (1990), 2038–2057.

[24] G. Ruckenstein, R.M. Roth, *Lower bounds on the anticipation of encoders for input-constrained channels, IEEE Trans. Inform. Theory,* 47 (2001), 1796–1812.

[25] W. Weeks IV, R.E. Blahut, *The capacity and coding gain of certain checkerboard codes, IEEE Trans. Inform. Theory,* 44 (1998), 1193–1203.

[26] J.K. Wolf, A.D. Wyner, J. Ziv, J. Körner, *Coding for a write-once memory, AT&T Bell Laboratories Tech. J.,* 63 (1984), 1089–1112.