Bounds on the Rate of 2-D Bit-Stuffing Encoders[*]

Ido Tal Ron M. Roth
Computer Science Department
Technion, Haifa 32000, Israel.
Email: {idotal, ronny}@cs.technion.ac.il

***Abstract*—A method for bounding the rate of bit-stuffing encoders for 2-D constraints is presented. Instead of considering the original encoder, we consider a related one which is quasi-stationary. We use the quasi-stationary property in order to formulate linear requirements that must hold on the probabilities of the constrained arrays that are generated by the encoder. These requirements are used as part of a linear program. The minimum and maximum of the linear program bound the rate of the encoder from below and from above, respectively.**

A lower bound on the rate of an encoder is also a lower bound on the capacity of the corresponding constraint. For some constraints, our results lead to tighter lower bounds than what was previously known.

I. Introduction

Two-dimensional (2-D) constraints are formally defined in [8]. Consider a 2-D constraint $\mathbb{S}$ defined over some finite alphabet Σ. Informally, a bit-stuffing encoder for $\mathbb{S}$ operates as follows. We encode information to an $M \times N$ rectangular array; namely, we produce an array $a \in \mathbb{S} \cap \Sigma^{M \times N}$. We first initialize the "boundaries" of the array (formally defined later) according to some fixed probability distribution. Then, we write to the "interior" of the array in raster fashion: row-by-row. The symbol currently written is the result of a coin toss. The probability distribution of the coin is a function of neighboring symbols, which have already been written. However, the "coins" used are in fact (invertible) probability transformers, the input of which is the information we wish to encode. Thus, information can be encoded, and decoded.

A bit-stuffing encoder is "variable-rate". The bit-stuffing technique was initially devised for encoding one-dimensional (1-D) constraints [2]. In [7] and [10], bit-stuffing encoders for specific 2-D constraints were presented and analyzed. In [6], a slightly different definition of bit-stuffing was used to give lower bounds on the capacity of specific 2-D constraints.

In this work, we derive upper and lower bounds on the rate of a general bit-stuffing encoder. A lower bound on the rate of an encoder is also a lower bound on the capacity of the corresponding constraint:

$$\mathsf{cap}(\mathbb{S}) = \lim_{M,N \to \infty} \frac{1}{M \cdot N} \cdot \log_2 \left| \mathbb{S} \cap \Sigma^{M \times N} \right| \ .$$

For some constraints, our results lead to tighter lower bounds on capacity than what was previously known.

[*] This work was supported by grant No. 2002197 from the United-States–Israel Binational Science Foundation (BSF), Jerusalem, Israel. The results of this work were presented at the IEEE International Symposium on Inforamtion Theory, Toronto, Ontario, Canada, July 2008.

$$\begin{array}{cccccccc}
1 & \mathbf{0} & 0 & 0 & \mathbf{0} & 1 & \mathbf{0} & 0 \\
\mathbf{0} & \mathbf{0} & 0 & 0 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
0 & 0 & 0 & 0 & 0 & \mathbf{0} & 1 & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & 0 & 0 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & 1 & \mathbf{0} & 0 & 0 & 0 & 0 & 0
\end{array}$$

Fig. 1. Binary array satisfying the square constraint. If we flip any one (or more) of the highlighted "0" bits to "1", then the resulting array will not satisfy the square constraint.

Fix some 2-D constraint $\mathbb{S}$ over an alphabet Σ. As a running example, consider the square constraint $\mathbb{S}_{\mathrm{sq}}$, defined over the binary alphabet $\Sigma_{\mathrm{sq}} = \{0, 1\}$ (see Figure 1). A binary array satisfies the square constraint if each entry set to "1" has all of its eight-neighbors set to "0". Namely, two entries equal to "1" may not appear consecutively along a row, column, or diagonal.

The rest of this paper is organized as follows. In Sections II and III, we define our notation and our model of a bit-stuffing encoder, respectively. In Section IV, we define the concept of quasi-stationarity. We also prove that, w.l.o.g., we may assume that our encoder is quasi-stationary. In Section V, we take advantage of the quasi-stationary property and define a linear program. The minimum (maximum) of the linear program bounds the rate of our encoder from below (above). Finally, section VI states a generic lower bound on capacity, and contains examples where this bound improves on previous results.

We note at this point that although this work deals with 2-D constraints, our method can be easily generalized to higher dimensions as well.

II. Notation

We first set up some notation.

Rectangle and parallelogram: For $M, N > 0$ and $t \geq 0$, denote

$$\mathsf{B}_{M,N} = \{(i,j) : 0 \leq i < M \ , \quad 0 \leq j < N\} \ ,$$

and

$$\Delta^{(t)}_{m,n} = \{(i,j) : 0 \leq i < M \ , \quad 0 \leq t \cdot i + j < N\} \ .$$

Configuration: Let $a = (a_{i,j})_{(i,j) \in \mathsf{U}}$ be a 2-D configuration over Σ. Namely, the index set satisfies $\mathsf{U} \subseteq \mathbb{Z}^2$, and for all $(i,j) \in \mathsf{U}$ we have that $a_{i,j} \in \Sigma$.

Shifts: For integers α, β we denote the shifted index set as

$$\sigma_{\alpha,\beta}(\mathsf{U}) = \{(i + \alpha, j + \beta) : (i,j) \in \mathsf{U}\} \ .$$

Also, by abuse of notation, let $\sigma_{\alpha,\beta}(a)$ be the shifted configuration (with index set $\sigma(\mathsf{U})$):

$$\sigma_{\alpha,\beta}(a)_{i+\alpha,j+\beta} = a_{i,j} \ .$$

Restriction of configuration: For an index set $\Psi \subseteq \mathsf{U}$, denote the restriction of a to Ψ by $a[\Psi] = (a[\Psi]_{i,j})_{(i,j)\in\Psi}$. Namely,

$$a[\Psi]_{i,j} = a_{i,j} \ , \quad \text{where} \quad (i,j) \in \Psi \ .$$

Shift and restrict: Let $\tau_{\alpha,\beta}(a, \Psi)$ be shorthand for

$$\tau_{\alpha,\beta}(a,\Psi) = (\sigma_{-\alpha,-\beta}(a))[\Psi] \ .$$

Namely, shift the configuration a such that index (α, β) is now index $(0, 0)$, and then restrict to Ψ.

Boundary: Denote by $\partial(\mathsf{U}, \Psi)$ the set of all the indexes $(\alpha, \beta) \in \mathsf{U}$ for which the "shift and restrict" operation is invalid.

$$\partial(\mathsf{U},\Psi) = \{(\alpha,\beta) \in \mathsf{U} : \sigma_{\alpha,\beta}(\Psi) \nsubseteq \mathsf{U}\} \ .$$

The index set $\partial(\mathsf{U}, \Psi)$ is termed the "boundary", and the "interior" is

$$\bar{\partial}(\mathsf{U},\Psi) = \mathsf{U} \setminus \partial(\mathsf{U},\Psi) \ .$$

When $\mathsf{U} = \mathsf{B}_{M,N}$ and Ψ is understood from the context, we abbreviate

$$\partial_{M,N} = \partial(\mathsf{B}_{M,N},\Psi) \ , \quad \bar{\partial}_{M,N} = \bar{\partial}(\mathsf{B}_{M,N},\Psi) \ .$$

Figure 2 shows an example of such sets, where

$$\Psi = \{(0,-2),(0,-1),(-1,-1),(-1,0),(-1,1)\} \ . \qquad (1)$$

Restriction of constraint: Denote the restriction of $\mathbb{S}$ to U by

$$\mathbb{S}[\mathsf{U}] = \{a : \text{there exists } a' \in \mathbb{S} \text{ such that } a'[\mathsf{U}] = a\} \ .$$

If $\mathsf{U} = \mathsf{B}_{M,N}$, then we abbreviate

$$\mathbb{S}_{M,N} = \mathbb{S}[\mathsf{B}_{M,N}] \ .$$

Lexicographic ordering: We define a lexicographic ordering $\prec$ on $\mathbb{Z}^2$ as

$$(i',j') \prec (i,j) \quad \Longleftrightarrow \quad (i' < i) \ \text{or} \ (i' = i \quad \text{and} \quad j' < j) \ .$$

Also, we define the index set

$$\mathsf{T}_{i,j} = \{(i',j') : (i',j') \prec (i,j)\} \ . \qquad (2)$$

III. Bit stuffer definitions

In this section, we present the formal definition of bit-stuffing encoders. A bit-stuffing encoder for $\mathbb{S}$ is defined through a triple

$$\mathcal{E} = (\Psi, \mu, \boldsymbol{\delta} = (\delta_{M,N})_{M,N>0}) \ .$$

The set

$$\Psi \subseteq \mathsf{T}_{0,0} \qquad (3)$$

is termed the *neighbor set*. The *conditional probability function* μ,

$$\mu(\cdot|\cdot) \ , \quad \mu : \Sigma \times \mathbb{S}[\Psi] \to [0,1] \ ,$$

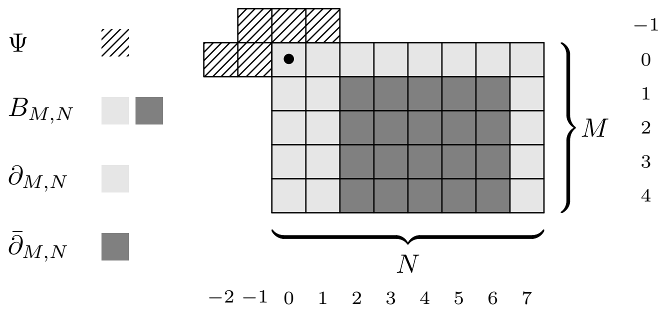

Fig. 2. The index $(0,0)$ is represented by $\bullet$. We take Ψ as in (1), and it is represented by the diagonally striped cells. We set $M = 5$ and $N = 8$. The index set $\mathsf{B}_{M,N}$ is represented by the shaded part (both light and dark). The boundary $\partial_{M,N}$ is represented by the lighter shaded part, while the interior $\bar{\partial}_{M,N}$ is represented by the darker shaded part.

$$\varphi^{(0)} = \begin{array}{cccc} & 0 & 0 & 0 \\ 0 & 0 & \bullet & \end{array} \qquad \varphi^{(1)} = \begin{array}{cccc} & 0 & 0 & 0 \\ 1 & 0 & \bullet & \end{array}$$

Fig. 3. The two non-trivial configurations for μ in our running example, where $\bullet$ designates coordinate $(0,0)$.

is a conditional probability distribution on Σ, given an element of $\mathbb{S}[\Psi]$. For $M, N > 0$, the *boundary probability function*

$$\delta_{M,N} : \mathbb{S}[\partial_{M,N}] \to [0,1]$$

is a probability distribution on $\mathbb{S}[\partial_{M,N}]$. From here onward, we fix $\mathcal{E}$.

For our running example, let the neighbor set $\Psi_{\text{sq}} = \Psi$ be as in (1), and define $\varphi^{(0)}, \varphi^{(1)} \in \mathbb{S}_{\text{sq}}[\Psi]$ as

$$\begin{array}{lllll} \varphi^{(0)}_{0,-2}{=}0 & \varphi^{(0)}_{0,-1}{=}0 & \varphi^{(0)}_{-1,-1}{=}0 & \varphi^{(0)}_{-1,0}{=}0 & \varphi^{(0)}_{-1,1}{=}0 \\ \varphi^{(1)}_{0,-2}{=}1 & \varphi^{(1)}_{0,-1}{=}0 & \varphi^{(1)}_{-1,-1}{=}0 & \varphi^{(1)}_{-1,0}{=}0 & \varphi^{(1)}_{-1,1}{=}0 \end{array}$$

(see Figure 3). Also, take the conditional probability function as

$$\mu_{\text{sq}}(1|\varphi) = 1 - \mu_{\text{sq}}(0|\varphi) = \begin{cases} 0.258132 & \varphi = \varphi^{(0)} \\ 0.312231 & \varphi = \varphi^{(1)} \\ 0 & \text{otherwise} \ . \end{cases} \qquad (4)$$

Thus, $\mu_{\text{sq}}(\cdot|\cdot)$ can be implemented using two coins (one for the context $\varphi^{(0)}$ and one for $\varphi^{(1)}$). For our running example, we take $\delta_{M,N}$ as the function equal to 1 for the all zero boundary $(0)_{(i,j)\in\partial_{M,N}}$, and 0 for all other members of $\mathbb{S}_{\text{sq}}[\partial_{M,N}]$.

Given integers $M, N > 0$, the bit-stuffing encoder $\mathcal{E}$ defines a probability measure on the elements $a = (a_{i,j})_{(i,j)\in\mathsf{B}_{M,N}}$ of $\mathsf{B}_{M,N}$, in the following manner. As a first step, we set the boundary $a[\partial_{M,N}]$, according to the probability distribution $\delta_{M,N}$. Next, we write the contents of the interior of a in raster fashion: row-by-row, from left to right. The probability of writing $w \in \Sigma$ in entry $(i,j) \in \bar{\partial}_{M,N}$ is given by

$$\text{Prob}(a_{i,j} = w) = \mu(w|(\tau_{i,j}(a,\Psi)) \ .$$

Specifically, note that when writing entry (i,j), we have by (3) that $\tau_{i,j}(a)$ is a function of entries of a which have already been written. A fundamental requirement for Ψ and μ is that for every M, N, and $\delta_{M,N}$, the support of the probability measure thus defined is contained in $\mathbb{S}_{M,N}$.

Let

$$A(\mathcal{E}, M, N) = A = (A_{i,j})_{(i,j)\in \mathsf{B}_{M,N}}$$

be a random variable taking values on $\mathbb{S}_{M,N}$ according to the measure we have just defined. Namely,

$$\mathrm{Prob}(A = a) = \delta_{M,N}(a[\partial_{M,N}])\cdot \prod_{(i,j)\in\bar{\partial}_{M,N}} \mu(a_{i,j}|\tau_{i,j}(a,\Psi)) \ . \quad (5)$$

We now explain how $\mathcal{E}$ is used to actually encode information. The "coin tosses" corresponding to the invocations of μ are, in effect, a function of the information we wish to encode. Specifically, the values of the tosses are the output of distribution transformers on the input stream (the mapping from the input stream to the sequence of coin tosses is one-to-one) [10]. Thus, we may encode information, and also decode it. So, we define the *rate* of our encoder as

$$R(\mathcal{E}) \triangleq \liminf_{M,N\to\infty} \frac{H(A[\bar{\partial}_{M,N}]|A[\partial_{M,N}])}{M\cdot N} \ ,$$

where

$$A = A(\mathcal{E}, M, N) \ .$$

Note that since

$$\liminf_{M,N\to\infty} \frac{|\bar{\partial}_{M,N}|}{M\cdot N} = 1 \ ,$$

we also have that

$$R(\mathcal{E}) = \liminf_{M,N\to\infty} \frac{H(A(\mathcal{E},M,N))}{M\cdot N} \ .$$

IV. Quasi-stationarity

Fix $k > 0$. Define the random variable

$$A^{(k)}(\mathcal{E}, M, N) = A^{(k)} = (A^{(k)}_{i,j})_{(i,j)\in\mathsf{B}_{M,N}}$$

taking values on $\mathbb{S}_{M,N}$ as follows. For $w \in \mathbb{S}_{M,N}$, we have

$$\mathrm{Prob}(A^{(k)}{=}w) = \frac{1}{k^2}\sum_{0\le i,j<k} \mathrm{Prob}(\sigma_{-i,-j}(A'[\mathsf{B}_{M,N}]){=}w) \ ,$$

where

$$A' = A(\mathcal{E}, M+k-1, N+k-1) \ .$$

Namely, given A', we randomly and uniformly pick an $M\times N$ sub-configuration of it, and shift accordingly. The usefulness of $A^{(k)}$ is that it is "quasi-stationary" [7, §6].

Lemma 1 ([7, Proposition 6.1]): Let $\mathcal{E}$, M, N, and k be given. Let $\mathsf{U} \subseteq \mathsf{B}_{M,N}$ be an index set, and let $w \in \mathbb{S}[\mathsf{U}]$ be given. Suppose that for given integers α, β we have that $\sigma_{\alpha,\beta}(\mathsf{U}) \subseteq \mathsf{B}_{M,N}$. Denote $A^{(k)} = A^{(k)}(\mathcal{E}, M, N)$. Then,

$$\left|\mathrm{Prob}(A^{(k)}[\mathsf{U}] = w) - \mathrm{Prob}(A^{(k)}[\sigma_{\alpha,\beta}(\mathsf{U})] = \sigma_{\alpha,\beta}(w))\right| \le \frac{|\alpha|+|\beta|}{k} \ .$$

Next, we show that $A^{(k)}$ is a random variable corresponding to an encoder very similar to $\mathcal{E}$. First, define $\boldsymbol{\delta}^{(k)} = (\delta^{(k)}_{M,N})_{M,N>0}$, where

$$\delta^{(k)}_{M,N} : \mathbb{S}[\partial_{M,N}] \to [0,1]$$

(that is, $\delta^{(k)}_{M,N}$ is a probability distribution on $\mathbb{S}[\partial_{M,N}]$), and for every $d \in \mathbb{S}[\partial_{M,N}]$,

$$\delta^{(k)}_{M,N}(d) = \mathrm{Prob}(A^{(k)}(\mathcal{E},M,N)[\partial_{M,N}] = d) \ .$$

Next, define the encoder $\mathcal{E}^{(k)}$ as

$$\mathcal{E}^{(k)} = (\Psi, \mu, \boldsymbol{\delta}^{(k)}) \ . \quad (6)$$

Lemma 2 ([7, Proposition 6.2]): The probability distributions of $A^{(k)}(\mathcal{E}, M, N)$ and $A(\mathcal{E}^{(k)}, M, N)$ are equal.

The next lemma essentially states that the normalized entropies of A and $A^{(k)}$ are asymptotically equal (for $M, N \to \infty$ and k fixed). The proof is straightforward.

Lemma 3: Fix an integer $k > 0$. Then,

$$R(\mathcal{E}) = R(\mathcal{E}^{(k)}) \ .$$

It follows from Lemma 3 that we can obtain bounds on $R(\mathcal{E})$ by bounding instead the rate of the quasi-stationary encoder $\mathcal{E}^{(k)}$. And, indeed, quasi-stationarity will turn out to be useful for this purpose.

V. Linear program

In this section, we present lower and upper bounds on $R(\mathcal{E})$. The bounds will be expressed as values of corresponding linear programs.

For $r, s > 0$ and $t \ge 0$, we say that the parallelogram $\Delta^{(t)}_{r,s}$ is valid with respect to the neighbor set Ψ if the set

$$\left\{(\alpha,\beta) : (\Psi \cup (0,0)) \subseteq \sigma_{\alpha,\beta}(\Delta^{(t)}_{r,s})\right\} \quad (7)$$

is non-empty. Namely, some shift of the parallelogram includes the neighbor set Ψ and $(0,0)$. From here onward, we fix r, s, and t so that $\Delta^{(t)}_{r,s}$ is valid. Also, we fix u and v, where (u,v) is the largest element of (7), with respect to the ordering $\prec$.

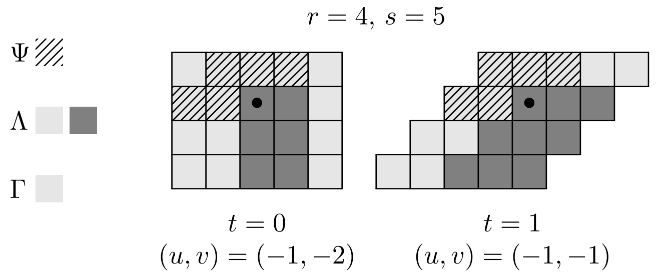

Fig. 4. The index sets Ψ, Λ and Γ. The index sets are shown for $r = 4$, $s = 5$, and for both $t = 0$ and $t = 1$. The index $(0,0)$ is represented by $\bullet$. We take Ψ as in (1), and it is represented by the diagonally striped cells. The index set Λ is represented by the shaded part (both light and dark). The boundary Γ is represented by the lighter shaded part. Note that $\Psi \subseteq \Gamma \subseteq \Lambda$.

Denote (see Figure 4)

$$\Lambda = \sigma_{u,v}(\Delta^{(t)}_{r,s}) \ , \quad \Gamma = \partial(\Lambda, \Psi) \ .$$

For an as yet unspecified probability distribution over $\mathbb{S}[\Gamma]$

$$\pi(z) , \quad z \in \mathbb{S}[\Gamma] ,$$

define the random variable Y taking values on $\mathbb{S}[\Lambda]$ as follows. For $y \in \mathbb{S}[\Lambda]$,

$$\mathrm{Prob}(Y = y) = \pi(y[\Gamma]) \prod_{(i,j)\in\Lambda\setminus\Gamma} \mu(y_{i,j}|\tau_{i,j}(y,\Psi)) \tag{8}$$

(compare to (5)). Note that $\mathrm{Prob}(Y = y)$ is a linear function of the various $\pi(z)$'s. Next, define

$$\Lambda' = \sigma_{u,v}(\Delta^{(t)}_{r-1,s}) , \quad \Lambda'' = \sigma_{u,v}(\Delta^{(t)}_{r,s-1}) ,$$

and

$$\Gamma' = \partial(\Lambda', \Psi) , \quad \Gamma'' = \partial(\Lambda'', \Psi) .$$

Consider the linear program in Figure 5. First, note that it is indeed a linear program. Namely, recall that by (8), the probability distribution of Y is a linear function of the $\pi(z)$'s. Thus, both sides of (9) and (10) are also linear functions of the $\pi(z)$'s. For example, the LHS of (9) equals

$$\sum_{y\in\mathbb{S}[\Lambda]:y[\Gamma']=z'} \pi(y[\Gamma]) \prod_{(i,j)\in\Lambda\setminus\Gamma} \mu(y_{i,j}|\tau_{i,j}(y,\Psi)) .$$

Denote the value of the linear program when minimizing by $\mathrm{lp}^*_{\min} = \mathrm{lp}^*_{\min}(\mathcal{E})$, and when maximizing by $\mathrm{lp}^*_{\max} = \mathrm{lp}^*_{\max}(\mathcal{E})$. Since (5) and (8) are very similar, we may intuitively say that $\mathcal{E}$ outputs Y. The optimization is over the probability distribution of the boundary $Y[\Gamma]$. The linear requirements (9) and (10) are added to force the distribution of Y to be stationary. The objective function is the rate at point $(0, 0)$.

The following theorem is our main result.

Theorem 4: For the linear program in Figure 5, we have that

$$\mathrm{lp}^*_{\min} \le R(\mathcal{E}) \le \mathrm{lp}^*_{\max} .$$

In order to prove the theorem, we first state and prove a lemma, on a slightly modified linear program.

Lemma 5: Fix $k > 0$, and replace (9) and (10) in Figure 5 by

$$\left|\mathrm{Prob}(Y[\Gamma'] = z') - \mathrm{Prob}(Y[\sigma_{0,1}(\Gamma')] = \sigma_{0,1}(z'))\right| \le \frac{1}{k}$$

and

$$\left|\mathrm{Prob}(Y[\Gamma''] = z'') - \mathrm{Prob}(Y[\sigma_{1,-t}(\Gamma'')] = \sigma_{1,-t}(z''))\right| \le \frac{t+1}{k} ,$$

respectively.

Denote the minimum and maximum of the resulting linear program as $\mathrm{lp}^{(k)}_{\min}$ and $\mathrm{lp}^{(k)}_{\max}$, respectively. Then,

$$\mathrm{lp}^{(k)}_{\min} \le R(\mathcal{E}) \le \mathrm{lp}^{(k)}_{\max} .$$

Proof: Consider $\mathcal{E}^{(k)}$ (as defined by (6)). For given M and N, define the index sets

$$\mathsf{B} = \partial(\mathsf{B}_{M,N}, \Lambda) , \quad \mathsf{U} = \bar{\partial}(\mathsf{B}_{M,N}, \Lambda) .$$

Obviously,

$$\lim_{M,N\to\infty} \frac{|\mathsf{U}|}{M \cdot N} = 1 . \tag{11}$$

Denote $A^{(k)} = A^{(k)}(\mathcal{E}, M, N)$. By (11) and Lemma 2,

$$R(\mathcal{E}^{(k)}) = \lim_{M,N\to\infty} \frac{H(A^{(k)}[\mathsf{U}]|A^{(k)}[\mathsf{B}])}{|\mathsf{U}|} .$$

Notice that $\Psi \subseteq \Lambda$. Thus, $\mathsf{U} \subseteq \bar{\partial}_{M,N}$, and we have

$$\begin{aligned} H(A^{(k)}[\mathsf{U}]|A^{(k)}[\mathsf{B}]) &= \sum_{(i,j)\in\mathsf{U}} H(A^{(k)}_{i,j}|A^{(k)}[\mathsf{T}_{i,j} \cap \mathsf{B}_{M,N}]) \\ &= \sum_{(i,j)\in\mathsf{U}} H(A^{(k)}_{i,j}|\tau_{i,j}(A^{(k)}, \Psi)) , \end{aligned}$$

where $\mathsf{T}_{i,j}$ is as defined in (2) and the last equality follows from (5).

We now prove the following claim: for all $(i, j) \in \mathsf{U}$, we have that

$$\mathrm{lp}^{(k)}_{\min} \le H(A^{(k)}_{i,j}|\tau_{i,j}(A^{(k)}, \Psi)) . \tag{12}$$

To see this, fix some $(i, j) \in \mathsf{U}$, and define for all $z \in \mathbb{S}[\Gamma]$,

$$p^{(k)}(z) = \mathrm{Prob}(\tau_{i,j}(A^{(k)}, \Gamma) = z) .$$

Substituting $\pi(z) = p^{(k)}(z)$, the objective function in Figure 5 is equal to $H(A^{(k)}_{i,j}|\tau_{i,j}(A^{(k)}, \Psi))$. Also, notice that the probability distribution of Y is equal to that of $\tau_{i,j}(A^{(k)}, \Lambda)$. By the fact that $A^{(k)}$ is quasi-stationary (and thus, so is every sub-configuration of it), all the linear requirements in the modified linear program are satisfied (i.e., the $p^{(k)}(z)$'s form a feasible solution). So, our claim (12) is proved.

We conclude that $\mathrm{lp}^{(k)}_{\min} \le R(\mathcal{E}^{(k)})$. Thus, by Lemma 3,

$$\mathrm{lp}^{(k)}_{\min} \le R(\mathcal{E}) .$$

A similar proof yields $R(\mathcal{E}) \le \mathrm{lp}^{(k)}_{\max}$. ■

Minimize (Maximize)

$$-\sum_{z\in\mathbb{S}[\Gamma]} \pi(z) \sum_{w\in\Sigma} \mu(w|z[\Psi]) \log_2 \mu(w|z[\Psi])$$

over the variables $(\pi(z) : z \in \mathbb{S}[\Gamma])$, subject to the following:

$$\sum_{z\in\mathbb{S}[\Gamma]} \pi(z) = 1 .$$

For all $z \in \mathbb{S}[\Gamma]$,

$$\pi(z) \ge 0 .$$

For all $z' \in \mathbb{S}[\Gamma']$,

$$\mathrm{Prob}(Y[\Gamma'] = z') = \mathrm{Prob}(Y[\sigma_{0,1}(\Gamma')] = \sigma_{0,1}(z')) . \tag{9}$$

For all $z'' \in \mathbb{S}[\Gamma'']$,

$$\mathrm{Prob}(Y[\Gamma''] = z'') = \mathrm{Prob}(Y[\sigma_{1,-t}(\Gamma'')] = \sigma_{1,-t}(z'')) . \tag{10}$$

Fig. 5. Linear program. The minimum (maximum) value is denoted $\mathrm{lp}^*_{\min}$ ($\mathrm{lp}^*_{\max}$) and is a lower (upper) bound on $R(\mathcal{E})$.

TABLE I
BOUNDS ON THE RATES OF ENCODERS USING A SMALL NUMBER OF COINS.

Constraint	Coins	$\mathrm{lp}^*_{\min}$	$\mathrm{lp}^*_{\max}$	[7]
$(2,\infty)$-RLL	1	0.440722	0.444679	0.4267
$(3,\infty)$-RLL	1	0.349086	0.386584	0.3402
n.i.b.	2	0.91773	0.919395	0.91276
$(1,\infty)$-RLL	3	0.587776	0.587785	—

Proof of Theorem 4: First, note that the modified linear program defined in Lemma 5 has at least one feasible solution, $p^{(k)}(z)$, whenever M and N are large enough so that U is non-empty.

For a given k, denote the minimizing variable values of the modified linear program by $\pi^{(k)}(z)$, $z \in \mathbb{S}[\Gamma]$. Think of these variable values as a vector

$$\boldsymbol{\pi}^{(k)} = (\pi^{(k)}(z))_{z\in\mathbb{S}[\Gamma]} \ .$$

By compactness, the series $\boldsymbol{\pi}^{(k)}$, $k = 1, 2, \ldots$, has a cluster point, which we denote by $\boldsymbol{\pi}^*$. Obviously, $\boldsymbol{\pi}^*$ implies a feasible solution for the linear program in Figure 5. More so, we must also have that the value of the objective function for this feasible solution is a lower bound on $R(\mathcal{E})$. So,

$$\mathrm{lp}^*_{\min} \le R(\mathcal{E}) \ .$$

Similarly, we deduce that

$$R(\mathcal{E}) \le \mathrm{lp}^*_{\max} \ .$$

■

Remark: While the definition of the encoder $\mathcal{E}$ includes (besides Ψ and μ) also the boundary distributions $\boldsymbol{\delta} = (\delta_{M,N})_{M,N>0}$, the bounds $\mathrm{lp}^*_{\min}$ and $\mathrm{lp}^*_{\max}$ do *not* depend on $\boldsymbol{\delta}$.

Applying Theorem 4 to our running example, with $r = 4$, $s = 5$, $t = 1$, gives

$$0.42430953 \le R(\mathcal{E}) \le 0.42442765 \ .$$

To the best of our knowledge, our running example is the highest rate bit-stuffing encoder known, given that we are allowed to use at most two coins (i.e., two probability transformers). For comparison, we have calculated by the method presented in [3] that

$$\mathsf{cap}(\mathbb{S}_{\mathrm{sq}}) \le 0.425078 \ .$$

Namely, with two coins we achieve a rate that is only 0.2% less than capacity.

Table I contains our results for a number of constraints. We abbreviate the "no isolated bits" constraints as "n.i.b.". In the first three rows, we compare ourselves to the results in [7] (Table 1 and Equation (12)). For the comparison to be fair, we restrict ourselves to the neighbor sets Ψ used in [7], and use the same number of coins.

TABLE II
BOUNDS ON THE RATES OF CERTAIN BIT-STUFFING ENCODERS.

Constraint	Coins	$\mathrm{lp}^*_{\min}$	$\mathrm{lp}^*_{\max}$	Others
$(2,\infty)$-RLL	5	**0.444202**	0.444997	0.4423
$(3,\infty)$-RLL	2	0.359735	0.368964	0.3641
$(0,2)$-RLL	66	**0.815497**	0.816821	0.7736
	18	0.815013	0.816176	
	9	0.810738	0.819660	
n.i.b.	56	**0.922640**	0.923748	0.9156

VI. A LOWER BOUND ON CAPACITY

The following is a straightforward corollary of Theorem 4.

Corollary 6: For every bit-stuffing encoder $\mathcal{E}$,

$$\mathrm{lp}^*_{\min}(\mathcal{E}) \le \mathsf{cap}(\mathbb{S}) \ .$$

Thus, we can use the minimizing linear program of Figure 5 to bound $\mathsf{cap}(\mathbb{S})$ from below.

To obtain better lower bounds on $\mathsf{cap}(\mathbb{S})$, we can search for good Ψ and μ. For instance, for the set $\Psi = \Psi_{\mathrm{sq}}$ in (1), the function μ_{sq} in (4) was obtained by maximizing $\mathrm{lp}^*_{\min}$ over all μ that form with Ψ_{sq} (and every $\boldsymbol{\delta}$) a bit-stuffing encoder for $\mathbb{S}_{\mathrm{sq}}$. Better lower bounds can be obtained by looking at larger sets Ψ (at the price of higher computational complexity).

Table II summarizes our results for certain constraints. The "Others" column contains previously published lower bounds on the capacity of the corresponding constraint. The bounds in that column are taken from [9], [6], [1], and [5], respectively. We have highlighted values of $\mathrm{lp}^*_{\min}$ which are an improvement of these previously known results.

We can use ideas somewhat reminiscent of the ones outlined in this paper in order to derive an *upper* bound on $\mathsf{cap}(\mathbb{S})$. The interested reader is referred to [11].

ACKNOWLEDGMENT

The first author wishes to thank Roee Engelberg for very stimulating discussions.

REFERENCES

[1] J. J. Ashley and B. H. Marcus, "Two-dimensional low-pass filtering codes," *IEEE Trans. Commmun.*, 46 (1998), 724–727.

[2] P. Bender and J. K. Wolf, "A universal algorithm for generating optimal and nearly optimal run-length-limited, charge constrained binary sequences," *Proc. 1993 IEEE Symp. Inform. Theory (ISIT'1993)*, San Antonio, Texas (1993), p.6.

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

[4] S. Forchhammer and J. Justesen, "Bounds on the capacity of constrained two-dimensional codes," *IEEE Trans. Inform. Theory*, 46 (2000), 2659–2666.

[5] S. Forchhammer and T. V. Laursen, "A model for the two-dimensional No Isolated Bits constraint," *Proc. 2006 IEEE Symp. Inform. Theory (ISIT'2006)*, Seattle, Washington, 2006, pp. 1189–1193.

[6] S. Forchhammer and T. V. Laursen, "Entropy of bit-stuffing-induced measures for two-dimensional checkerboard constraints," *IEEE Trans. Inform. Theory*, 53 (2007), 1537–1546.

[7] S. Halevy, J. Chen, R. M. Roth, P. H. Siegel, and J. K. Wolf, "Improved bit-stuffing bounds on two-dimensional constraints," *IEEE Trans. Inform. Theory*, 50 (2004), 824–838.

[8] S. Halevy and R. M. Roth, "Parallel constrained coding with application to two-dimensional constraints," *IEEE Trans. Inform. Theory*, 48 (2002), 1009–1020.

[9] E. Ordentlich and R.M. Roth, “Capacity lower bounds and approximate enumerative coding for 2-D constraints,” *Proc. 2007 IEEE Symp. Inform. Theory (ISIT’2007)*, Nice, France, 2007, pp. 1681–1685.

[10] R. M. Roth, P. H. Siegel, and J. K. Wolf, “Efficient coding schemes for the hard-square model,” *IEEE Trans. Inform. Theory*, 47 (2001), 1166-1176.

[11] I. Tal and R. M. Roth, “Concave programming upper bounds on the capacity of 2-D constraints,” `arXiv:YYMM.NNNN [category]`.