When Data Must Satisfy Constraints Upon Writing

Erik Ordentlich
Hewlett–Packard Laboratories
Palo Alto, CA 94304
erik.ordentlich@hp.com

Ron M. Roth
Computer Science Department
Technion, Haifa, Israel
ronny@cs.technion.ac.il

***Abstract*—We initiate a study of constrained codes in which any codeword can be transformed into any other codeword by a sequence of single symbol changes such that the intermediate words (after each symbol change) all satisfy the underlying constraint. We shall refer to a set of constrained words with this property as being Hamming connected. Hamming connected constrained codes might be useful for encoding data in storage media when a constraint must be met upon writing data, as might be the case in some emerging storage technologies. The stated property would permit overwriting encoded data without violating the constraint during intermediate writes. We study the Hamming connectedness of (d, k)-run-length limited constraints and a few other special cases. We also consider the decidability of Hamming connectedness for finite memory constraints.**

I. Introduction

The traditional role of constrained coding in storage systems has been to facilitate the *reading* of data by controlling inter-symbol interference and easing symbol timing recovery. In contrast, emerging data storage technologies such as memristor crossbars may benefit if the stored data satisfies certain constraints prior to the *(over-)writing* of each new bit [1]. In both cases, information must be coded into constrained words or patterns for storage onto the physical medium. In the case of write enabling constraints, however, an additional property is required to allow overwriting of stored data. Namely, it must be possible to meet the constraint during each intermediate step of the overwriting process which ultimately transforms the state of the physical medium from one encoded pattern to another.

This motivates the definition and study of Hamming connected constrained codes. Given a symbol alphabet Σ, for each positive integer n let $\mathcal{S}^{(n)}$ denote the constraint satisfying words of length n over Σ and let $\mathcal{S} = \bigcup_{n=1}^{\infty} \mathcal{S}^{(n)}$. A *Hamming connected constrained code* of block length n with respect to the constraint $\mathcal{S}$ is a subset $\mathcal{C} \subseteq \mathcal{S}^{(n)}$ with the property that for all pairs of elements (e.g., codewords) $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ of $\mathcal{C}$ there exists a sequence of words $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_m$ satisfying $\boldsymbol{w}_1 = \boldsymbol{c}_1$, $\boldsymbol{w}_m = \boldsymbol{c}_2$, $\boldsymbol{w}_i \in \mathcal{S}^{(n)}$ for each $i \in \{2, 3, \ldots, m-1\}$, and $\mathsf{d}_{\mathrm{H}}(\boldsymbol{w}_i, \boldsymbol{w}_{i+1}) = 1$ for each $i \in \{1, 2, \ldots, m-1\}$, where $\mathsf{d}_{\mathrm{H}}(\cdot, \cdot)$ denotes Hamming distance. Any two words in $\mathcal{S}^{(n)}$ with this property will be said to be *Hamming connected* within $\mathcal{S}$. Notice that this relation is an equivalence relation (in particular, it is symmetric).

This work was supported in part by Grant No. 2008058 from the United-States–Israel Binational Science Foundation. The work of R.M. Roth was done in part while visiting Hewlett–Packard Laboratories, Palo Alto, CA.

The customary definitions of encoder, decoder, and rate apply directly for such a code. The Hamming connectedness property allows for transforming a codeword into any other codeword by changing one symbol at a time such that all intermediate words are constraint satisfying. This permits the above noted constraint satisfying overwrite process required for write enabling constrained coded systems.

Given a constraint $\mathcal{S}$, we can define the *Hamming connectedness capacity* $\mathsf{c}_{\mathrm{H}}(\mathcal{S})$ as

$$\mathsf{c}_{\mathrm{H}}(\mathcal{S}) = \limsup_{n\to\infty} \frac{1}{n} \log_2 \left(\max_{\substack{\mathcal{C} \subseteq \mathcal{S}^{(n)}: \\ \mathcal{C} \text{ is Hamming connected}}} |\mathcal{C}| \right).$$

If the codes corresponding to the sets $\mathcal{S}^{(n)}$ are Hamming connected for each n, we shall say that the constraint is Hamming connected. In this paper, we determine $\mathsf{c}_{\mathrm{H}}(\mathcal{S})$ for several illustrative classical constraints $\mathcal{S}$. For the special case of finite memory constraints, we give some partial results on the problem of deciding whether a given such constraint is Hamming connected, which we have not been able to resolve. With the obvious extension of the Hamming connectedness concept to two-dimensional arrays in mind, we also analyze the Hamming connectedness properties of constraints that may be relevant to the aforementioned memristor crossbar application.

In the sequel we use the following notation. For a word $\boldsymbol{x} = x_1 x_2 \ldots x_n$ we denote by $\boldsymbol{x}_{i:j}$ the subword $x_i x_{i+1} \ldots x_j$. The subset of all words of a given length n in a set of words (in particular, a constraint) $\mathcal{S}$ will be denoted by $\mathcal{S}^{(n)}$.

A set of words $\mathcal{S}$ over an alphabet Σ is called a *finite memory* (or *finite type*) constraint [2] if there is a finite list L of words, known as a *forbidden list*, such that a word $\boldsymbol{x}$ over Σ is in $\mathcal{S}$ if and only if none of the subwords $\boldsymbol{x}_{i:j}$ are in L. A (finite memory) constraint $\mathcal{S}$ is *irreducible* if for any pair of words $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}$ there is a word $\boldsymbol{w} \in \mathcal{S}$ such that the concatenation $\boldsymbol{xwy}$ is also in $\mathcal{S}$. An irreducible constraint $\mathcal{S}$ is *primitive* if the length of such a $\boldsymbol{w}$ can be set to some fixed ℓ, independent of $\boldsymbol{x}$ and $\boldsymbol{y}$. The (ordinary) Shannon capacity of a constraint $\mathcal{S}$ will be denoted by $\mathsf{c}(\mathcal{S})$.

II. Examples

A. No-isolated-bits (NIB) constraint

This constraint, denoted $\mathcal{S}_{\mathrm{NIB}}$, is defined as the finite memory constraint with forbidden list $\{101, 010\}$. Notice that given a word in $\mathcal{S}_{\mathrm{NIB}}$, inserting a 1 in the middle of a run of

consecutive 0's or a 0 in the middle of a run of consecutive 1's would yield words not satisfying the constraint. Thus, it would seem to be impossible to break or merge runs via single bit changes. Nevertheless, $\mathcal{S}_{\mathrm{NIB}}$ turns out to be Hamming connected. The trick is to utilize either boundary of the word, noting that, say for a word $x_1x_2\ldots x_n$, it may be allowed to set $(x_1,x_2)\in\{(0,1),(1,0)\}$. This means that a word of the form, e.g., $0^{r_1}1^{r_2}0^{r_3}\ldots$ (denoting r_1 0's, followed by r_2 1's, and so on) can be transformed into $0^{r_1-1}1^{r_2+1}0^{r_3}\ldots$ and then into $0^{r_1-2}1^{r_2+2}0^{r_3}\ldots$, and ultimately into $1^{r_2+r_1}0^{r_3}\ldots$. This can be continued with a new phase to eliminate the starting run of 1's, with the first word being $1^{r_2+r_1-1}0^{r_3+1}\ldots$, until the all–0 word is obtained. Thus, all words are Hamming connected to the all–0 word and therefore, by the symmetry of Hamming connectedness, $\mathcal{S}_{\mathrm{NIB}}$ is fully Hamming connected.

Notice that the Hamming connectedness of $\mathcal{S}_{\mathrm{NIB}}$ makes critical use of the boundary, relying, in particular, on the fact that the boundary values are "less" constrained. In fact, a simple modification to a constrained word, such as inserting 00 into the middle of a run of 1's in the middle of the word requires completely rewriting, possibly several times, all values out to the boundary. In some scenarios, this may be impractical, such as in a large storage medium which is partitioned into blocks that need to be updated independently and are separated by "guard symbols" to enforce the constraint throughout the medium. Such scenarios can be modeled by fixing one or more boundary values and not allowing them to be changed. We can then consider a boundary restricted Hamming connectedness capacity wherein for a given k we further constrain the above definitions to words $\boldsymbol{x}$ which have the same values for their k-prefix (i.e., $\boldsymbol{x}_{1:k}$) and k-suffix ($\boldsymbol{x}_{n-k+1:n}$). In particular, this means in the definition of connectedness that the intervening words all share these values at the boundary. The restricted connectedness capacity may then depend on these values.

The NIB constraint is no longer Hamming connected under any fixed-boundary condition, since there is no way to break or merge runs. However, it turns out that the capacity (i.e., the growth rate of number of Hamming connected words) still equals the ordinary Shannon capacity, $\mathsf{c}(\mathcal{S}_{\mathrm{NIB}})$, of $\mathcal{S}_{\mathrm{NIB}}$, for all fixed k and corresponding boundary values. To show this, we note that since $\mathcal{S}_{\mathrm{NIB}}$ is irreducible and since the number of distinct runs of consecutive symbols grows at most linearly in n, there exists an $\ell>k$, a sequence of positive integers $(R_n)_n$, and binary words $\boldsymbol{u}$ and $\boldsymbol{v}$ of length ℓ, such that $\boldsymbol{u}_{1:k}$ and $\boldsymbol{v}_{\ell-k+1:\ell}$ are equal to the fixed left and right boundary values, and such that the set of words

$$\mathcal{D}_n=\{\boldsymbol{x}\in\mathcal{S}_{\mathrm{NIB}}^{(n)}\,:\,\boldsymbol{x}_{1:\ell}=\boldsymbol{u},\boldsymbol{x}_{n-\ell+1:n}=\boldsymbol{v},x_{\ell+1}=0,\\ \boldsymbol{x}_{\ell+1:n-\ell}\text{ has }R_n\text{ distinct runs}\}$$

satisfies $\lim_{n\to\infty}(1/n)\log_2|\mathcal{D}_n|=\mathsf{c}(\mathcal{S}_{\mathrm{NIB}})$. It is then easy to see that the runs comprising the inner portions of the words belonging to $\mathcal{D}_n$ can be expanded and reduced via single symbol switches to obtain any other word in $\mathcal{D}_n$. The set $\mathcal{D}_n$ is thus Hamming connected within $\mathcal{S}_{\mathrm{NIB}}$.

B. (d,k)-run-length limited (RLL) constraints

In the case of (d,k)-RLL constraints [2, p. 6], the Hamming connectedness of the constraint depends on the relative values of d and k. Specifically, we prove the following; hereafter, $\mathcal{S}_{d,k}$ denotes the set of all binary (d,k)-RLL constrained words, for given d and k.

Proposition 2.1: The constraint $\mathcal{S}_{d,k}$ is Hamming connected (and, therefore, $\mathsf{c}_{\mathrm{H}}(\mathcal{S}_{d,k})=\mathsf{c}(\mathcal{S}_{d,k})$) if and only $k\geq 3d+1$.

Proposition 2.2:

$$\mathsf{c}_{\mathrm{H}}(\mathcal{S}_{d,k})=\begin{cases}0 & \text{if } k\leq 2d\\ \mathsf{c}(\mathcal{T}_{d,k}) & \text{if } 2d<k<3d+1\end{cases},$$

where $\mathcal{T}_{d,k}$ is the finite memory constraint defined by the following forbidden list:

$$\{0^{k+1}\}\cup\{10^r1:r\in[0,d-1]\cup[k-d,2d]\}\,.$$

Proof of Proposition 2.1: The case of $d=0$ is almost immediate, since, starting with a constraint satisfying word, any 0 can be switched to a 1 without violating the constraint, leading ultimately to the all–1 word. Thus, all words are Hamming connected to the all–1 word which implies that the constraint is Hamming connected. In the rest of the proof we will assume $d>0$.

Suppose $k<3d+1$ and consider a word containing a subword 10^r1 positioned at least d away from either boundary, with $r=\min(k,2d)$. We claim that no symbol of this subword can be legally switched, irrespective of the rest of the word. Clearly, it is not possible to switch any 0, since this would result in at least one of the two resulting runs being shorter than d. Moreover, neither the starting nor ending 1 can be switched since this would merge the run with a neighboring run of length at least d resulting in a total run-length at least $\min(k+d+1,3d+1)>k$. Thus, no word containing such a subword can be converted into a word with say, only runs of length d, and it follows that for $k<3d+1$, the (d,k)-RLL constraint is not Hamming connected.

Suppose that $k\geq 3d+1$. We show how any word in $\mathcal{S}_{d,k}$ can be converted into a word with all runs equal to at most d (the only runs possibly shorter than d would be those at the boundary). First, it should be easy to see that any run longer than $2d$ can be broken up into shorter runs of length at most $2d$ (e.g., by switching every $(d{+}1)$st 0 from the start of the run until within $2d$ of the end of the run). We can thus assume that we are starting with a word $\boldsymbol{x}=x_1x_2\ldots x_n$ in $\mathcal{S}_{d,k}^{(n)}$ having only runs of length $2d$ or shorter. Moreover, we can further assume that the initial (left-most) run of 0's is of length d or shorter, for if it were otherwise, we could change the $(d{+}1)$st 0 from the end of this run to a 1. Now, suppose the ith run of 0's begins at position t_i (i.e., $x_{t_i-1}=1$) and that, inductively, $\boldsymbol{x}$ is Hamming connected to a word of the form $0^{r_1}10^d10^d\ldots10^d1x_{t_i:n}=0^{r_1}10^d10^d\ldots10^d10^{r_i}1x_{t_{i+1}:n}$ where $r_i\leq 2d$. The following chain of transformations can

then be carried out via legal single symbol changes:

$$0^{r_1}10^d10^d\ldots10^d10^{r_i}1x_{t_{i+1}:n} \to 0^{r_1}10^d10^d\ldots10^{d+r_i+1}1x_{t_{i+1}:n} \to$$
$$0^{r_1}10^d10^d\ldots10^{r_i}10^d1x_{t_{i+1}:n} \to \qquad \cdots \qquad \to$$
$$0^{r_1}10^{r_i}10^d\ldots10^d10^d1x_{t_{i+1}:n} \to 0^{r_1+r_i+1}10^d\ldots10^d10^d1x_{t_{i+1}:n} \to$$
$$0^{r_1+r_i-d}10^d\ldots10^d10^d1x_{t_{i+1}:n} \to 0^{r_1'}10^d\ldots10^d10^d1x_{t_{i+1}:n}$$

with $r_1' \leq d$. In the above chain of transformations, the first transformation therein is legal since $d+r_i+1 \leq 3d+1 \leq k$, and the first two transformations are repeatedly applied moving backwards along the word to obtain the final word. The entire process can then be repeated for the run starting at t_{i+1}, and so on, until a word of the form $0^r10^d10^d\ldots10^d10^s$, with $r, s \leq d$ has been reached. Moreover, words of this form are readily seen to be Hamming connected via transformations of the form

$$0^r10^d10^d10^d\ldots10^d10^s \to 0^{r+d+1}10^d10^d\ldots10^d10^s \to$$
$$0^{r-1}10^{d+1}10^d10^d\ldots10^d10^s \to 0^{r-1}10^{2d+2}10^d\ldots10^d10^s \to$$
$$0^{r-1}10^d10^{d+1}10^d\ldots10^d10^s \to \qquad \cdots \qquad \to$$
$$0^{r-1}10^d10^d\ldots10^d10^{s+1}$$

which is valid since for $d \geq 1$, $2d+2 \leq 3d+1 \leq k$. □

Proof of Proposition 2.2: For $k \leq 2d$ it follows from the observations in the previous proof that for any constraint satisfying word the symbols comprising non-boundary runs cannot be legally switched, irrespective of the rest of the word. Thus, $\mathsf{c}_{\mathrm{H}}(\mathcal{S}_{d,k}) = 0$ in these cases.

Suppose that $2d < k < 3d+1$. We can transform any word in $\mathcal{T}_{d,k}$ having any run in the range $[2d+1, k]$ into a word having runs only in the range $[d, k-d-1]$ as in the above proof, by switching the $(d+1)$st 0 of the longer runs (in the range $[2d+1, k]$). Each such longer run is broken into two runs of lengths d and at most $k-d-1$, respectively. The above iterative procedure can then be used to transform the resulting word into one consisting entirely of runs of length d (or shorter at the boundaries).

This establishes that $\mathsf{c}_{\mathrm{H}}(\mathcal{S}_{d,k}) \geq \mathsf{c}(\mathcal{T}_{d,k})$. To show the inequality in the other direction, consider the constraint $\mathcal{L}$ with forbidden list $\{0^{k+1}\} \cup \{10^r1 : r \in [0, d-1]\} \cup \{0^d10^r10^d : r \in [k-d, 2d]\}$. Notice that $\mathcal{L} \subseteq \mathcal{S}_{d,k}$. Moreover, since each word in the forbidden list of $\mathcal{T}_{d,k}$ is a subword of some word in the forbidden list of $\mathcal{L}$, it follows that $\mathcal{T}_{d,k} \subseteq \mathcal{L}$. It is also not hard to see that the distinction between the forbidden lists of $\mathcal{T}_{d,k}$ and $\mathcal{L}$ is only relevant for symbols near the start and end of a word, and, more specifically, that for $n > 2d$ it holds that $\{\boldsymbol{x}_{d+1:n-d} : \boldsymbol{x} \in \mathcal{L}^{(n)}\} = \mathcal{T}_{d,k}^{(n-2d)}$. It follows from the theory of finite memory constraints that the boundary conditions do not impact the exponential growth rate of $|\mathcal{L}^{(n)}|$ so that $\lim_{n\to\infty}(1/n)\log_2|\mathcal{L}^{(n)}| = \mathsf{c}(\mathcal{T}_{d,k})$, and, moreover, that there exists a constant α such that

$$|\mathcal{L}^{(n)}| \leq \alpha \cdot 2^{n\,\mathsf{c}(\mathcal{T}_{d,k})}. \tag{1}$$

We will now prove that the cardinality of *any* Hamming connected component of $\mathcal{S}_{d,k}^{(n)}$ is no larger than the right-hand side of (1), for each n, where a Hamming connected component is defined as an equivalence class of the Hamming connectedness equivalence relation on words within $\mathcal{S}_{d,k}^{(n)}$.

We proceed by induction. The base case is immediate since for sufficiently small n (e.g., $n \leq d$), $\mathcal{L}^{(n)}$ coincides with $\mathcal{S}_{d,k}^{(n)}$. Now fix n, and assume that the cardinality of any Hamming connected component is bounded from above by $\alpha \cdot 2^{n'\mathsf{c}(\mathcal{T}_{d,k})}$ for all $n' < n$. By definition, each word $\boldsymbol{x} \in \mathcal{S}_{d,k}^{(n)}$ belongs to one and only one Hamming connected component. Suppose, $\boldsymbol{x} \in \mathcal{S}_{d,k}^{(n)} \setminus \mathcal{L}^{(n)}$ so that $\boldsymbol{x}$ contains a subword $0^d10^r10^d$ for some r in the stated forbidden range. It is easy to see that irrespective of the rest of the word, no single symbol of this subword can be legally switched. Suppose that this subword starts at index t within $\boldsymbol{x}$. Thus, all other words in the connected component containing $\boldsymbol{x}$ also contain this subword at index t. Let $\mathcal{M}$ denote this connected component. Note that the set of words of length $n-r-1$ given by $\mathcal{M}' = \{\boldsymbol{y}_{1:t-1}0^d10^d\boldsymbol{y}_{t+r+2d+2:n} : \boldsymbol{y} \in \mathcal{M}\}$ is then contained in a Hamming connected component in $\mathcal{S}_{d,k}^{(n-r-1)}$ and that $|\mathcal{M}'| = |\mathcal{M}|$. It follows by the induction hypothesis that

$$|\mathcal{M}| \leq \alpha \cdot 2^{(n-r-1)\mathsf{c}(\mathcal{T}_{d,k})} \leq \alpha \cdot 2^{n\,\mathsf{c}(\mathcal{T}_{d,k})}$$

and that, therefore, $|\mathcal{M}| \leq \alpha \cdot 2^{n\,\mathsf{c}(\mathcal{T}_{d,k})}$, establishing the induction step for this case. Suppose, on the other hand, that $\boldsymbol{x} \in \mathcal{L}^{(n)}$. The above considerations for words not belonging to $\mathcal{L}^{(n)}$ imply that the Hamming connected component containing $\boldsymbol{x}$ must be a subset of $\mathcal{L}^{(n)}$. The induction step for this case then follows from (1). The proposition is thereby established since $\mathsf{c}_{\mathrm{H}}(\mathcal{S}_{d,k})$ coincides with the growth rate of the largest Hamming connected component. □

C. Row–column Hamming weight constrained arrays

Consider the set $\mathcal{A}_{n\times n}$ of all $n \times n$ arrays $A = (A_{i,j})_{i,j}$ over $\{0,1\}$ such that the Hamming weight of each row and column is at most u and at least ℓ namely, for every $i, j \in \{1, 2, \ldots, n\}$,

$$\ell \leq \sum_{h=1}^{n} A_{i,h} \leq u \quad \text{and} \quad \ell \leq \sum_{k=1}^{n} A_{k,j} \leq u\,. \tag{2}$$

The case of $\ell = 0$ and $u = n/2$ is the constraint alluded to in Section I as having been proposed (e.g., [1]) to facilitate the writing process in a next generation memory technology based on crossbars of memristive devices. This case is readily seen to be Hamming connected (assuming the obvious generalization thereof to this setting) since changing any 1 to a 0 in a legal array results in another legal array, thereby implying that all legal arrays are Hamming connected to the all–0 array.

Another potentially useful case for the crossbar application turns out to be $\ell = n/2-\delta$ and $u = n/2+\delta$ for some positive $\delta \ll n$ (e.g., $\delta = o(n)$). This case may also be useful in the memristor crossbar application, especially in conjunction with alternative write voltage biasing schemes differing from the one associated with the baseline case above. Here, we omit further details of this potential application and focus on the Hamming connectedness property for such u and ℓ. This case is more challenging than the $\ell = 0$ case, since there is not necessarily a "minimal" or "maximal" array. Nevertheless, we have the following.

Proposition 2.3: For all $0 \leq \ell < u \leq n$, the set $\mathcal{A}_{n\times n}$ defined by (2) is Hamming connected.

Proof: We shall say that a row or column is minimal (resp., maximal) if its Hamming weight is ℓ (resp., u). Given two arrays X and Y in $\mathcal{A}_{n\times n}$, we will demonstrate how to change X into Y in accordance with the Hamming connectedness property. We can assume that all 0's in X which must be changed to 1 in Y belong to a maximal row or column. Otherwise, any 0 to 1 change not satisfying this condition can be made legally, one at a time, until the stated assumption holds. Similarly, we can assume that all 1's to be changed to a 0 belong to a minimal row or column. Let (i_1, j_1) be the row–column indices of a 0 to 1 change. Let's assume that, say, row i_1 is maximal. Since Y is in the constraint there must exist $j_2 \neq j_1$ such that (i_1, j_2) corresponds to a 1 to 0 change. Moreover, by the above assumptions, column j_2 is minimal, and thus there must exist an $i_2 \neq i_1$ such that (i_2, j_2) corresponds to a 0 to 1 change. It follows, by assumption, that i_2 is maximal, and that there is $j_3 \neq j_2$ such (i_2, j_3) is a 1 to 0 transition. Note that j_3 might equal j_1. If not, we can continue this process until either the row index or column index, after each respectively changes, coincides with a previously encountered row or column index. Suppose (w.l.o.g.) it is the row index which is the first to repeat. In this case, by dropping all indices encountered prior to and including the *first* time the repeated row is encountered, we will have found a "cyclic" sequence of row–column index pairs $(i'_1, j'_1), (i'_2, j'_1), (i'_2, j'_2), \ldots, (i'_{t-1}, j'_t), (i'_t, j'_t)$, with $i'_t = i'_1$ and i'_s all distinct for $s < t$ and j'_s all distinct for $s \leq t$, and such that the index pairs, starting with the first, alternatingly correspond to 1 to 0 and 0 to 1 changes, and, finally, such that each index pair is in a maximal row *and* a minimal column. Notice that each row and column that contains an element of this sequence, contains precisely two elements.

We now claim that there must either be a 1 in row i'_1 that is not in a minimal column *or* a 0 in column j'_1 that is not in maximal row. Suppose neither of these were the case. Since row i'_1 is maximal, it would then follow from the first part not holding that the total number of 1's in the array is *at most* $u\ell + (n-u)u$ while it would follow from the second part not holding that the total number of 1's is *at least* $(n-\ell)u + \ell^2$. However, subtracting the latter from the former, yields

$$u\ell + (n-u)u - (n-\ell)u - \ell^2 = -u^2 + 2u\ell - \ell^2 = -(u-\ell)^2 \, ,$$

which is negative for $u > \ell$, leading to a contradiction. Suppose (w.l.o.g.) the claim holds with a 1 in row i'_1 and column j'', where the latter is not minimal. Thus, $j'' \notin \{j'_1, j'_2, \ldots, j'_t\}$. We can then temporarily change this 1 to a 0, to "unlock" row i'_1, that is, to render the row non-maximal. This, in turn, allows us to make the 0 to 1 change at location $(i'_t, j'_t) = (i'_1, j'_t)$, which in turn renders column j'_t non-minimal. And this in turn, allows us to make the 1 to 0 change at location (i'_{t-1}, j'_t), in the cyclic sequence above. We can continue this process, working backwards in the sequence, until we have made all of the changes in the sequence, noticing that each successive change renders the next row or column non-maximal or non-minimal, thereby allowing the next change to be made. Since, as noted above, each row and column undergoes both a 0 to 1 and a 1 to 0 change in this sequence of changes, the Hamming weights of all rows and columns, after the changes, are as they were just after the "unlocking" 1 to 0 change at (i'_1, j''). We are thus free to change the 0 back to a 1 at this location.

It should be clear that repeated applications of the above procedure will transform X into Y, via the requisite, constraint satisfying, single symbol changes. □

III. Decidability of Hamming connectedness

A natural question concerning the Hamming connectedness of finite memory constraints, as defined in Section I, is whether it is decidable in the classical sense of Turing decidability. Specifically, given a finite alphabet Σ (e.g., the binary alphabet with $\Sigma = \{0, 1\}$), we ask if there is an algorithm (i.e., Turing machine) that takes as input the forbidden list L defining a finite memory constraint and halts with an indication as to whether the corresponding constraint is Hamming connected. An equivalent question is whether there is a computable function $f : \mathcal{P} \to \mathbb{Z}$, with domain $\mathcal{P}$ being the set of all finite subsets of words over the alphabet Σ, with the property that the corresponding finite memory constraint $\mathcal{S}$ with forbidden list L is Hamming connected if and only if $\mathcal{S}^{(f(L))}$ is Hamming connected. So far we have been unable to obtain such a computable $f(L)$. Notice that the trivial algorithm which simply tests Hamming connectedness of $\mathcal{S}^{(n)}$ for $n = 1, 2, \ldots$ and returns the first n for which Hamming connectedness does not hold would not generate an output $f(L)$ for a constraint that *is* Hamming connected. We next present some partial results of relevance to this problem.

A. Role of the memory of the constraint

It is natural to conjecture that there is some connection between a computable $f(L)$ with the above properties and the length of the longest word in L. One less than this coincides with the memory of the constraint [2].

Figure 1, depicting a family of constraints over the alphabet $\{0, 1, \ldots, N\}$ for some $N \in \mathbb{Z}^+$, shows that any such connection is weak at best. For each N, the corresponding constraint is comprised of the vertex labels of all finite walks on the depicted graph. The corresponding forbidden list consists of those words of length 2 for which no directed edge connects the vertex labeled with the first symbol to the vertex labeled with the second. It is not hard to see that for each N, the corresponding constraint is Hamming connected for block length $N-2$ (or smaller), but not for block length $N-1$: for the latter, notice that no single symbol of the word $2, 3, 4, \ldots, N$ can be changed. Thus, for arbitrarily large block lengths n, there exist constraints $\mathcal{S}$ with memory 1 that are not Hamming connected, yet $\mathcal{S}^{(n)}$ is Hamming connected.

B. Connection to two-dimensional constraints of finite type

Let L be the forbidden list of a given finite memory constraint $\mathcal{S}$ with ℓ denoting the maximum length of words in L. Consider the forbidden list of two-dimensional $2 \times \ell$

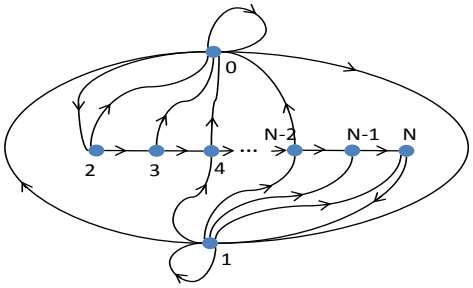

Fig. 1. Graph representation of a finite memory constraint over the alphabet $\{0, 1, \ldots, N\}$ with memory 1 which is not Hamming connected, but which is block Hamming connected for block lengths up to length $N-2$.

patterns:

$$L' = \{X \in \{0,1\}^{2\times \ell} : X_{1,i:j} \in L \text{ for some } i \leq j, \text{ or } X_{2,i:j} \in L \text{ for some } i \leq j, \text{ or } \mathsf{d_H}(X_{1,:}, X_{2,:}) > 1\} ,$$

where $X_{k,i:j}$ denotes the word $X_{k,i}, X_{k,i+1}, \ldots, X_{k,j}$ and $X_{k,:}$ denotes the kth row of X. We shall say that a two-dimensional constraint (corresponding to a set of valid $m \times n$ arrays over some alphabet with $m, n \in \mathbb{Z}^+$) is irreducible if the one-dimensional constraints corresponding to arrays with a fixed number of columns (or vertical stripes) are all irreducible. For a given stripe-width w, let $\mathcal{B}_w$ denote the one-dimensional vertical stripe constraint with forbidden list L' given above. Notice that $\mathcal{B}_w$ has memory 1 and irreducibility is thus equivalent to the property that for any pair of words $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}^{(w)}$, there exists an $m \times w$ array X in $\mathcal{B}_w$ with $X_{1,:} = \boldsymbol{x}$ and $X_{m,:} = \boldsymbol{y}$.

Proposition 3.1: The constraint $\mathcal{S}$ (corresponding to L) is Hamming connected if and only if the two-dimensional constraint corresponding to L' is irreducible.

Proof: The Hamming connectedness of $\mathcal{S}$ implies the irreducibility of each $\mathcal{B}_w$, since for any $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}^{(w)}$, the array formed by stacking the sequences corresponding to the transformation of $\boldsymbol{x}$ into $\boldsymbol{y}$ via single symbol changes, is readily seen to belong to $\mathcal{B}_w$. Conversely, for $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{S}^{(w)}$, the rows of the array X guaranteed by the irreducibility of $\mathcal{B}_w$ can be modified to obtain the required transformation for Hamming connectedness by introducing the symbol changes in each successive row one symbol change at a time. This can be done, since the forbidden list defining $\mathcal{B}_w$ forbids pairs of consecutive rows in X from differing in more than one symbol within corresponding words not longer than the longest word in the forbidden list of $\mathcal{S}$. Thus, undoing any subset of these changes (e.g., to implement them one at a time) will not introduce a violation of the forbidden list of $\mathcal{S}$. □

With Proposition 3.1, we have reduced the decidability of Hamming connectedness for finite memory constraints to the decidability of the irreducibility of the above family of two-dimensional constraints. This problem, in turn, is reminiscent of the famous undecidable tiling problem [3] which roughly asks if there exists an infinite array not containing any pattern in a finite forbidden list. Our two-dimensional problem, however, does not appear to have the generality of the tiling problem, as the constraint in the vertical direction is a specific one.

If we require a uniform bound h on the merging word length in the irreducibility for all stripe widths w, then, under this restriction, decidability follows from the decidability of the equivalence of a pair of one-dimensional sofic shifts [2, Theorem 3.4.13], where a sofic shift is defined simply as a symbol-wise mapping of a finite memory shift (of finite type) and is representable as the sequences of labels of walks on an edge-labeled graph. The two sofic shifts to be compared, in this case, would be the product shift $\{\mathcal{S}^{(n)} \times \mathcal{S}^{(n)}\}_{n=1}^{\infty}$, comprised of all pairs of equal length words belonging to $\mathcal{S}$, and the sofic shift obtained by applying the symbol-wise mapping that returns the top and bottom symbols of an $h \times 1$ column word to the columns of the one-dimensional shift of finite type comprised of the horizontal $h \times n$, $n \in \mathbb{Z}^+$, stripes (arrays) not containing the forbidden subarrays in L'.

IV. Constant prefix–suffix Hamming connectedness

Rather than allowing arbitrary single symbol changes, one may consider constant prefix–suffix restricted Hamming connectedness, similar to what was considered in the context of the NIB constraint in Section I. Under this restriction, if the starting and ending words have a common prefix of m symbols or longer, where m is equal to the memory of the constraint, there is a legal transformation in which the first m symbols remain *unchanged* in all intermediate words and if there is a common suffix of m symbols or longer, there is a legal transformation in which the last m symbols are unchanged. Additionally, if there is both a common prefix and suffix of length m, then there is a legal transformation preserving both the first m and last m symbols.

Proposition 4.1: Constant prefix–suffix Hamming connectedness is decidable for primitive finite memory constraints.

Proof sketch: It suffices to check constant prefix–suffix Hamming connectedness up to a block length n where n is at least $3m+2k$, with k being such that for all pairs of constraint satisfying words of length m there is a constraint satisfying word of length $2m + k$ having the respective m length words as a prefix and suffix. The existence of such a k follows from the primitivity assumption. Assuming constant prefix–suffix Hamming connectedness for all such block lengths, we can use irreducibility and induction to show constant prefix–suffix Hamming connectedness with an intermediate word sharing a sufficiently long prefix and suffix with the starting word and a sufficiently long middle portion with the ending word. □.

References

[1] E. Ordentlich and R.M. Roth, *Low complexity two-dimensional weight-constraint codes*, *IEEE Trans. Inf. Theory,* 58 (2012), 3892–3899.

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

[3] R. Berger, *The undecidability of the domino problem*, *Mem. Amer. Math. Soc.,* 66, 1966.