Algorithms for Sliding Block Codes

An Application of Symbolic Dynamics to Information Theory

ROY L. ADLER, DON COPPERSMITH, AND MARTIN HASSNER, MEMBER, IEEE

***Abstract*—Ideas which have origins in Shannon's work in information theory have arisen independently in a mathematical discipline called symbolic dynamics. These ideas have been refined and developed in recent years to a point where they yield general algorithms for constructing practical coding schemes with engineering applications. In this work we prove an extension of a coding theorem of Marcus and trace a line of mathematics from abstract topological dynamics to concrete logic network diagrams.**

I. Introduction

A. The Problem

We address the problem of encoding and decoding digital data from one type of constraint to another by means of finite state automata. The data are long strings of symbols from a finite alphabet, usually zeros and ones or blocks of them. In this paper, we consider encoding arbitrary sequences of zeros and ones into a constrained format dictated by the data processor. The constraints may be due to physical limitations of a transmission or storage system or artificial limitations dictated by data processing procedures.

B. The Model

The appropriate mathematical models for dealing with the problem are *symbolic dynamical* systems, i.e., spaces, invariant under the shift transformation, of two-sided infinite sequences of symbols from finite alphabets. The term dynamical system is due to the fact that such spaces are composed of discrete time orbits, each orbit consisting of a succession of shifted sequences.

Practical encoders and decoders have short finite memories but strings they process are so long as to seem infinite. The proposed model is suitable to the problem because

i) the constraints are time independent (shift invariant);
ii) encoders and decoders can be constructed from mappings between systems which commute with some power of the shift (sliding block codes) [6], [25].

Manuscript received March 10, 1982; revised July 1, 1982. This work was supported in part by NSF Grant MCS81-07092. This work was presented in part at the IEEE International Symposium on Information Theory, Santa Monica, CA, February 1981.

The authors are with IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598.

Constraints encountered in practice are standard ones in symbolic dynamics. Of particular importance are those specified by a finite list of forbidden blocks of symbols, e.g., upper and lower bounds on run lengths of zeros and ones (Sec. VII), [14], [15], [16], [29], [33], [49]. In symbolic dynamics such systems are called *shifts of finite type or topological Markov shifts* [4], [51]. More complex constraints are also important such as ones involving the power spectrum of symbol sequences, e.g., no dc component in a signal representing the symbol sequence [14], [22], [30], [32], [34], [41], [46]. These can be described as outputs of a finite state automaton whose inputs are shifts of finite type. In symbolic dynamics such systems are called *sofic* (from the Hebrew word for finite) [50]. In engineering contexts, the above constraints have been described by the notion of a channel: namely, shifts of finite type are deterministic finite state channels with finite memory, and sofic systems are deterministic finite state channels with infinite memory. Some areas where these constraints are met are magnetic recording, fiber optics, and data protocols in communication networks.

C. Shannon Theory

Suppose we wish to encode in a decodable way every sequence $(\cdots x_{-1}, x_0, x_1 \cdots)$ of a system X satisfying one set of constraints into another system Y of sequences $(\cdots y_{-1}, y_0, y_1 \cdots)$ satisfying another. Each component x_n, y_n may itself consist of a finite block of symbols, say of length p and q, respectively, in which case we say that the *coding rate* $r = p/q$. The concept of topological entropy governs when this is possible. *Topological entropy* is defined as the exponential growth rate, as $n \to \infty$, of the number of different strings of length n appearing in the infinite sequences of a symbolic system. The term "topological" is used to distinguish this entropy from its probabilistic counterpart. It was defined in purely topological terms in [3].

In the present context where output symbols are of equal duration, Shannon's noiseless coding theorem [48, p. 28] amounts to the following obvious statement: coding of arbitrarily long finite strings is possible when the topological entropy of X is less than that of Y and impossible when the inequality is reversed, the case of equality being left unresolved. Shannon called the system Y, a *channel*, and its *topological entropy*, the *channel capacity*. He called the system X, the source, and endowed it with a probabilistic entropy (topological entropy maximizes probabilistic entropy supported by the source [12], [23]). The full content of Shannon's theorem applies to the situation where the topological entropy of the source is greater than that of the channel but its probabilistic entropy is less.

We sharpen Shannon's theorem for the special case where the source entropy is the topological entropy to show that coding is possible even in the case of equality. In addition, the method of proof provides efficient sequential encoding and decoding algorithms which do not depend on the length of strings processed, a feature absent from Shannon's original theorem. We treat the class of shifts of finite type (channels with finite memory) and leave the more general case of sofic systems (channels with infinite memory) to subsequent work. Actually, we deal with the case where the topological entropy of the source is the logarithm of an integer, the most common one in applications. The more general case where it is the logarithm of an algebraic integer can be handled by a slight extension of the "tableaux" method of [4], but then certain desirable error propagation properties usually must be forgone.

This paper is written for two worlds, engineering and mathematics, at the risk of satisfying neither. It runs the gamut from the sublimely abstract to the hard-nose concrete. We start with notions of sets, mappings, and topology in Section II, supplant these with combinatorial ideas by Section V, and finish at the end of Section IX with logic circuit diagrams. The complete trip is hardly needed for constructing codes, but it is useful in organizing the flow of ideas and bringing order to the subject. For the less mathematically minded, interested only in making codes for some practical purpose, we suggest concentrating on the description of the symbol splitting process of Section VI (not the proof), the example of Section VIII, and the implementation of it in Section IX. Following the pattern there one should be able to construct encoders and decoders for any shift of finite type constraint. The method can be extended to cover sofic systems arising from the aforementioned spectral constraints, but this has not yet appeared in print [38] and its applicability is not fully assessed. The excessive number of tables included in Sections VIII and IX are there to indicate the labor involved in constructing codes.

For the paper as a whole, we assume knowledge of the Perron-Frobenius theory of nonnegative matrices [21], [47] and some basic elements of symbolic dynamics which can be found in [4], [8], [11], [27], [28].

We present only one proof, that of the main theorem in Section VI. All others are standard and easy ones from symbolic dynamics. These are stated without proof. Where possible, references are cited in which proofs can be found; otherwise they should be treated as exercises. Actually Sections II–V is to be regarded as a survey.

Methods for doing noiseless coding have also been developed by P. Franaszek [14]–[19]. In [19], [20] he gives a general one which is different from ours yet intriguingly based on the same inequality (6.1) from the Perron–Frobenius theory. We hope someday to return to this topic and clarify the relationship between the two methods. Franaszek's ideas are very interesting and maybe lead to simpler implementations. From a mathematician's point of view these works [19], [20] leave something to be desired—namely, precise statements on the scope of the method along with complete proofs. A. Lempel and M. Cohn [35] work some examples by Franaszek's method, but leave the same mathematical questions unsettled.

The main results of our work were presented at the IEEE International Symposium on Information Theory, Santa Monica, CA (Feb. 1981) [2]. The theme, which is the application of recent developments in symbolic dynamics to coding problems in information theory, was suggested by M. Hassner in [26].

II. Abstract Dynamical Systems

Let X be a compact metric space and σ a homeomorphism—i.e., continuous one-to-one map—of X onto itself. We call the pair (X, σ) an *abstract dynamical system*. For a comprehensive treatment of such systems see [11]. If X' is a closed σ-invariant subset of X then the system (X', σ) is called a *subsystem* of (X, σ), and we write $(X', \sigma) \subset (X, \sigma)$. We define the *orbit, future orbit,* and *past orbit* of a point x by the respective sequences $\operatorname{orb} x \equiv \{\sigma^n x\}_{n\in Z}$, $\operatorname{orb}^+ x \equiv \{\sigma^n x\}_{n\geq 0}$, $\operatorname{orb}^- x \equiv \{\sigma^n x\}_{n<0}$. In order to economize on notation we shall always use the following convention for metrics. We denote the distance between two points x, y by $|x-y|$ even though subtraction and absolute value may not be defined.

Two orbits, orb x and orb y, are called *positively (negatively) asymptotic* if $|\sigma^n x - \sigma^n y| \to 0$, $n \to \infty$ $(n \to -\infty)$.

We have the following indecomposability conditions for a dynamical system and its higher iterates. A system is called *nonwandering transitive* if for every pair of neighborhoods there is a point in the first whose future orbit hits the second. A system is called *aperiodic* if (X, σ^n) is nonwandering transitive for all n.

Let $(X, \sigma), (Y, \tau)$ be two abstract dynamical systems. A continuous map φ of Y onto X such that $\varphi \circ \sigma = \tau \circ \varphi$ is called a *topological homomorphism*. If such a map exists we have the following *commutative diagram*: (Fig. 1).

We also refer to φ as a *factor map* and call (X, τ) a factor of (Y, σ) and (Y, σ) an *extension* of (X, τ). If, in addition, φ is one-to-one (hence invertible, φ^{-1} being continuous by compactness) we call it an *isomorphism* and say that (X, σ) is *topologically conjugate* to (Y, τ) and write $(X, \sigma) \approx (Y, \tau)$. Nonwandering transitivity and aperiodicity are preserved under isomorphism. Topological conjugacy is the strongest sense of equivalence of dynamical systems from the purely topological point of view and too strong for many practical applications (see Remark 8.1). Consequently, we introduce a weaker one.

Definition 2.1 (Parry [45]): We say (X, σ) and (Y, τ) are *finitely equivalent*, and write $(X, \sigma) \sim (Y, \tau)$, if there exists a common extension (Z, ρ) and boundedly finite-to-one factor maps φ, ψ of (Z, ρ) onto (X, σ) and (Y, τ), respectively. We represent this situation as in Fig. 2.

$$\begin{tikzcd} Y \arrow[r, "\sigma"] \arrow[d, "\varphi"'] & Y \arrow[d, "\varphi"] \\ X \arrow[r, "\tau"] & X \end{tikzcd}$$

Fig. 1. Commutative diagram of topological homomorphism.

$$\begin{tikzcd} & (Z,\rho) \arrow[dl, "\varphi"'] \arrow[dr, "\psi"] & \\ (Y,\tau) & & (X,\sigma) \end{tikzcd}$$

Fig. 2. Finite equivalence diagram.

Finite equivalence can be slightly strengthened, which is done in the next definition, and still be weaker than topological conjugacy.

Definition 2.2 (see [4], [24]): We say that (X, σ) and (Y, τ) are *almost one-to-one finitely equivalent* and write $(X, \sigma) \overset{\star}{\sim} (Y, \tau)$ if, in addition to being finitely equivalent, the factor maps are one-to-one except on nondoubly transitive points. A point x is said to be *nondoubly transitive* if either $\mathrm{orb}^+ x$ or $\mathrm{orb}^- x$ fails to be dense in its dynamical system, otherwise it is called *doubly transitive*.

Theorem 2.1 [4, p. 10]: $\approx$, $\sim$, and $\overset{\star}{\sim}$ are equivalence relations and

$$(X, \sigma) \approx (Y, \tau) \Rightarrow (X, \sigma) \overset{\star}{\sim} (Y, \tau) \Rightarrow (X, \sigma) \sim (Y, \tau).$$

We remark that the set of doubly transitive points is an invariant subset of a dynamical system and that factor maps in Definition 2.2 are isomorphisms between subsystems which, however, are not compact.

Definition 2.3 (Bowen–Dinaburg [7], [12]): The *topological entropy* $h(X, \sigma)$ for abstract dynamical systems is defined as the largest growth rate possible, as $n \to \infty$, of the number of ϵ-separated orbits of length n, i.e.,

$$h(X, \sigma) = \sup_{\epsilon > 0} \overline{\lim_{n \to \infty}} \frac{1}{n} \log N(\epsilon, n),$$

where $N(\epsilon, n)$ denotes the number of ϵ-separated orbits of length n. Two orbits of length $n \{\sigma^k x\}_{0 \le k \le n-1}$ and $\{\sigma^k y\}_{0 \le k \le n-1}$ are said to be *ϵ-separated* if $|\sigma^k x - \sigma^k y| \ge \epsilon > 0$ for some k, $0 \le k \le n-1$.

An easy consequence of the definition is that, if (X, σ) is a factor of (Y, τ), then $h(X, \sigma) \le h(Y, \tau)$. Also, if (X', σ) is a subsystem of (X, σ), then $h(X', \sigma) \le h(X, \sigma)$. We have Theorem 2.2.

Theorem 2.2 [4, p. 9]: If (X, σ) is a finite-to-one factor of (Y, τ) then $h(X, \sigma) = h(Y, \tau)$.

Corollary 2.3: If $(X, \sigma) \approx (Y, \tau)$ then $h(X, \sigma) = h(Y, \tau)$. Thus topological entropy is an invariant for all three equivalence relations. We also have the next theorem.

Theorem 2.4 [3, p. 311]: $h(X, \sigma^n) = |n| \cdot h(X, \sigma)$, $n \in Z$.

For coding applications we need a certain kind of invertibility condition for factor maps which for abstract systems is expressed by the following.

Definition 2.4 (Kitchens [31]): A factor map φ of (Y, σ) onto (X, τ) is called *right*, *left*, or *two-sided closing* if $\varphi x \ne \varphi y$ whenever $x \ne y$ and orb x, orb y are, respectively, negatively, positively or both negatively and positively asymptotic.

III. Symbolic Systems

Let A be an alphabet (sometimes called a *state space*), by which we mean a finite set of symbols (also called states) with an ordering. We denote the cardinality of a set A by $|A|$. Examples of alphabets are $\mathcal{A} = \{0, 1\}$, $\mathcal{A} = \{(0,0), (0,1), (1,0), (1,1)\}$, etc. We freely abuse notation by using indexing symbols to represent interchangeably both an element of an alphabet and its ordinal number. Which it should be clear from context. This sloppiness is often compounded by the fact that numbers also appear as alphabet symbols and that a numerical symbol may not coincide with its ordinal number. The advantage of inconsistency here is that it keeps notation to a minimum.

As is customary, $\mathcal{A}^Z$ denotes the set of two-sided infinite sequences of elements of $\mathcal{A}$. The space $\mathcal{A}^Z$ can be endowed with a metric, the distance between sequences $x = \{x_n\}_{n \in Z}$, $y = \{y_n\}_{n \in Z}$ being defined by $|x - y| \equiv \sum_{n=-\infty}^{\infty} |x_n - y_n| / 2^{|n|}$ where $|x_n - y_n|$ is defined to be one when $x_n \ne y_n$ and zero otherwise. In this metric the more two sequences consecutively agree the closer they are, and we have a neighborhood basis which consists of the family of sets, called *cylinder sets*, of the form $\{x = (\cdots x_{-1}, x_0, x_1, \cdots): (x_{n+1}, \cdots, x_{n+k}) = (a_1, \cdots, a_k)\}$ where $(a_1, \cdots, a_k)$ is some fixed k-tuple of symbols of $\mathcal{A}$. In this topology $\mathcal{A}^Z$ is compact.

We define the *shift transformation* σ of $\mathcal{A}^Z$ onto itself by $(\sigma x)_n = x_{n+1}$ for $x \in \mathcal{A}^Z$, $n \in Z$. In the above metric σ is a homeomorphism and we form the dynamical system $(\mathcal{A}^Z, \sigma)$ which is called the *full N-shift* where $N = |\mathcal{A}|$. Any subsystem $(X, \sigma) \subset (\mathcal{A}^Z, \sigma)$ is called a *subshift*. We use symbols $\mathcal{A}$, $\mathcal{B}$, $\mathcal{C}$ to denote alphabets. Occasionally we use $\mathcal{A}_X$ to denote the alphabet of a dynamical system (X, σ) which in the above is a subset of $\mathcal{A}$. Any finite n-tuple $(a_1, \cdots, a_n)$ which appears in any sequence of X is called an *admissible* n-block. The topological entropy of a subshift is given by

$$h(X, \sigma) = \overline{\lim_{n \to \infty}} \, 1/n \log N(n), \tag{3.1}$$

where $N(n)$ is the number of admissible n-blocks. We observe that

$$h(\mathcal{A}^Z, \sigma) = \log |\mathcal{A}| \tag{3.2}$$

and $h(X, \sigma) \le \log |\mathcal{A}|$ for $(X, \sigma) \subset (\mathcal{A}^Z, \sigma)$. The reader can regard (3.1) as a definition [44] although it is an easy exercise to derive it from Definition 2.3.

We introduce the notation $x|_m^n = (x_m, x_{m+1}, \cdots, x_n)$ for a sub-$(n - m + 1)$-tuple of an n-tuple or sequence x. Given a subshift (X, σ) we can form a subshift $(X^{[n]}, \sigma)$, called the *higher n-block system* of (X, σ), where $(X^{[n]}, \sigma)$ consists of sequences $(\cdots, x|_{-1}^{n-2}, x|_0^{n-1}, x|_1^n, \cdots)$, where $x \in X$. The higher n-block system $(X^{[n]}, \sigma)$ is *canonically isomorphic* to (X, σ) under the correspondence

$$\left(\cdots x|_{-1}^{n}, x|_0^{n-1}, x|_1^n\right) \longleftrightarrow (\cdots x_{-1}, x_0, x_1, \cdots)$$

$(X^{[n]}, \sigma)$ is a subshift of the full shift based on an alphabet of symbols consisting of all admissible n-blocks of X.

Definition 3.1: Let $(X, \sigma), (Y, \sigma)$ be subshifts of two full shifts $(\mathcal{A}^Z, \sigma), (\mathcal{B}^Z, \sigma)$, respectively. A mapping φ of (Y, σ) onto (X, σ) is called a *k-block map requiring memory l* and *anticipation m*, if there exists a function φ: $\mathcal{A}^k \to \mathcal{B}$ such that if $\{x_n\} = \varphi\{y_n\}$ then

$$x_n = \varphi(y_{n-l}, \cdots, y_{n+m}),$$

where $k = m + l + 1$, for $n \in Z$.

We use here the following abuse of notation. The same symbol φ is used to denote a mapping defined on sequences and the component function of several variables by which it is specified. What is meant will always be clear from context, hopefully.

In dynamical systems only the finiteness of k, not its size, is important. We can often take advantage of the conceptual simplification of regarding a k-block map as a 1-block map on a system with a larger alphabet. This is done by replacing a k-block map φ by the 1-block map $\varphi\vartheta^{-1}$ where ϑ is the canonical isomorphism between (Y, σ) and $(Y^{[k]}, \sigma)$. (See Fig. 3.)

However, for construction of encoders and decoders in engineering applications it is important to have k and the alphabet size as small as possible; so the above artifice is of no advantage from this point of view.

Theorem 3.1 [*11, p. 3*]*:* An onto mapping φ between subshifts is a homomorphism if and only if it is a k-block map.

If a k-block map φ between subshifts is invertible, then its inverse φ^{-1} is also a k-block map, perhaps with a different k. We define a weaker form of invertibility.

Definition 3.2: Let φ be as in Definition 3.1 with $\varphi(\{y_n\}) = \{x_n\}$. φ is said to be *right resolving with parameters p, q, r* of *memory* and *anticipation* if each y_n is uniquely determined from the upcoming $x_{n-p}, \cdots, x_{n+q}$ and preceding $y_{n-r}, \cdots, y_{n-1}$—in other words, there exists a function f: $\mathcal{B}^r \times \mathcal{A}^{p+q+1} \to \mathcal{B}$ such that

$$y_n = f(y_{n-r}, \cdots, y_{n-1}; x_{n-p}, \cdots, x_{n+q}).$$

A similar definition is given for *left resolving* by replacing $\{x_n\}$, $\{y_n\}$ by $\{x_{-n}\}$, $\{y_{-n}\}$ in the above definition.

We remark that Definition 3.2 is what Definition 2.4 becomes in the context of subshifts. Kitchens [31] used the term right (left) closing here. In [4] the definition of right and left resolving covered only the cases where $r = 1$, $p = q = 0$. The concept of right resolving is not new in information theory. It was called *unifilar* by McMillan [36, p. 216]. We could also give the definition of *two-sided resolving* [4] which is what two-sided closing becomes for symbolic systems, but that will not be needed in the present work. Suffice it to say that the importance of the concept of resolvability is put into evidence by the following theorem.

$$\begin{tikzcd} Y \arrow[r, "\vartheta"] \arrow[d, "\varphi"'] & Y^{[k]} \arrow[dl, "\varphi\vartheta^{-1}"] \\ X & \end{tikzcd}$$

Fig. 3. Equivalence of k-block to the 1-block map.

Theorem 3.2 [*4, p. 23*]*:* A homomorphism is finite-to-one if it is right or left resolving. (Right or left resolving imply two-sided resolving but not conversely. For subshifts of finite type which are defined in Section V finite-to-one implies two-sided resolving.)

Closely associated with the concept of right and left resolving is the notion of resolving block. Such blocks serve as a means for resetting an encoding automaton constructed from a right resolving map. Let φ in Definition 3.1 be a 1-block map, i.e., $x_n = \varphi(y_n)$.

Definition 3.3 [*1*], [*4*]*:* An X-admissible m-block $(a_1, \cdots, a_m)$ is called a *resolving block* if there exists an index i_0, $1 \le i_0 \le m$, such that if $(y_1, \cdots, y_m)$ and $(y_1', \cdots, y_m')$ are two Y-admissible m-blocks such that $\varphi(y_i) = \varphi(y_i') = a_i$, $1 \le i \le m$, then $y_{i_0} = y_{i_0}'$. In other words, the block $(a_1, \cdots, a_m)$ determines a unique preimage in the i_0th coordinate.

Remark 3.1: If φ is also right resolving with $r = 1$, $p = q = 0$ and $x|_1^m = (a_1, \cdots, a_m)$ then the sequence $y|_{i_0}^\infty$ can be uniquely determined from $x|_1^\infty$ whenever $x = \varphi(y)$. Furthermore, if there are so many resolving blocks that in every k-block there exists at least one, then φ is invertible, in fact, φ^{-1} can be seen to be a k-block map.

IV. Encoders and Decoders

Let (X, σ) and (W, σ) be two subshifts with alphabets $\mathcal{A}$ and $\mathcal{B}$, respectively. In the vocabulary of engineering let us call (X, σ) the *source* and (W, σ) the *channel*. Usually $\mathcal{A}$ and $\mathcal{B}$ consist, respectively, of admissible p-blocks and q-blocks of 0 and 1 for some fixed p and q. Furthermore, the source usually consists of unconstrained sequences and the channel of constrained ones. Thus $\mathcal{A}$ is the set of all 2^p p-blocks whereas $\mathcal{B}$ is some subset of q-blocks.

Our problem is to construct two finite state automata: an *encoder* which converts source sequences $\{x_n\} \in X$ to channel sequences $\{y_n\} \in W$; and a *decoder* which recovers $\{x_n\}$ from $\{y_n\}$. The coding rate is $r = p/q$ (p source symbols per q channel symbols) and we want this as large as feasible. At the same time we want q and p small to minimize complexity.

A *finite state automaton*, say the encoder, is given by two functions

$$\begin{aligned} y_n &= e(x_{n-l}, \cdots, x_{n+m}, z_n), \\ z_n &= f(x_{n-l}, \cdots, x_{n+m}, z_{n-1}), \end{aligned} \tag{4.1}$$

where z_n belongs to some finite alphabet $\mathcal{C}$, called the *internal states* of the automaton. The elements y_n are called the *output* and $x_{n-l}, \cdots, x_{n+m}$, the *inputs*, with l, m param-

$$\begin{tikzcd}
Y \arrow[r, "\sigma^{\nu}"] \arrow[d, "d"'] & Y \arrow[d, "d"] \\
X \arrow[r, "\sigma^{\nu}"] & X
\end{tikzcd}$$

Fig. 4. Commutative diagram of decoding map.

eters of *memory* and *anticipation*. The function e is called the *output function* and f the *next state function*. By substitution, y_n is a function of $x_{n-l},\cdots,x_{n+m}, z_{n-1}$. The numbering of variables in (4.1) may be slightly off from standard usage. We adopt the present one to conform to our notation of subsequent constructions.

The output sequences $\{y_n\}$ belong to a subsystem $(Y,\sigma) \subset (W,\sigma)$. By virtue of (4.1) this subsystem is a factor of a system of $((\mathcal{B}\times\mathcal{C})^Z,\sigma)$, which in turn is a factor of a subsystem of $((\mathcal{A}\times\mathcal{C})^Z,\sigma)$. A single error in the input will possibly propagate forever in the output. We take the point of view that the source is error free. However, errors may occur in the channel, and we want to limit the range of their propagation in the decoder. In order to do this, we should also make the decoder a finite state automaton, but one in which the internal state at a particular time does not depend on the input, but only on the previous state.

Thus the decoder is given by two functions:

$$x_n = d(y_{n-\tilde{l}},\cdots,y_{n+\tilde{m}};\tilde{z}_n),$$
$$\tilde{z}_n = \tilde{f}(\tilde{z}_{n-1}), \tag{4.2}$$

where $\tilde{z}$ belongs to some other finite alphabet $\tilde{\mathcal{C}}$. Since the set of states is finite, say $|\tilde{\mathcal{C}}| = \nu$, we can label them so that

$$\tilde{z}_n = n \pmod{\nu}.$$

From this we see that we require the decoder to be a finite-block map satisfying

$$d\,\sigma^{\nu} = \sigma^{\nu} d;$$

that is, we have the diagram of Fig. 4.

In designing a decoder, we try to make ν as small as possible, hopefully $\nu = 1$. This condition can always be trivially achieved at the expense of increasing p and q by a multiplicative factor ν, but this would not count as an improvement.

We would also like the encoder to be given by a finite block map of (X,σ) onto (Y,σ), which would mean $(X,\sigma) \approx (Y,\sigma)$. This is usually not possible, so we must be content with some weaker form of invertibility of the decoding map, like right resolvability, which is sufficient for constructing an encoding automaton.

V. Subshifts of Finite Type

We single out a special class of subshifts which go under a variety of names, two of which we shall use, the choice depending on the mode of description. The term *subshift of finite type* shall be used to designate a subshift (X,σ) when X is defined by specifying a finite list of forbidden finite blocks which do not occur anywhere in the sequences of X.

Let $T = (t_{ij})$ be an $N \times N$ matrix of zeros and ones, which we call a *transition matrix*. A k-tuple $(x_1,\cdots,x_k)$ of symbols $x_i \in \mathcal{A}$, is called a *T-admissible k-block* if $t_{x_i,x_{i+1}} = 1$ for $i = 1,\cdots,k-1$. A two-sided infinite sequence $\{x_n\}_{n\in Z}$ is called *T-admissible* if $t_{x_n,x_{n+1}} = 1$ for $n \in Z$. Let $\{T\}$ denote the set of T-admissible sequences. We use the term *topological Markov shift* to describe a subshift (X,σ) when $X = \{T\}$.

The first description tells what is forbidden, the second what is allowed. Both definitions describe the same class of dynamical systems: for one obtains a finite list of forbidden 2-blocks (i, j) from a transition matrix T whenever $t_{ij} = 0$. Conversely, if L is the length of the longest block in the forbidden list, then a new alphabet can be chosen to be the admissible $(L-1)$-blocks. Tacking on the right single symbols from the original alphabet in such a way as to get admissible L-blocks defines a transition matrix between $(L-1)$-blocks which overlap in $L-2$ places. The system that results is isomorphic to the original one. For this reason subshifts of finite type could also be called $(L-1)$*-step Markov* systems and topological Markov shifts, 1*-step Markov*.

A transition matrix T defines a *directed graph*, the symbols are *nodes* and the transitions *edges*. If we label edges with a new alphabet, then a new transition matrix $T^{[2]}$ is formed by specifying how the edges are connected. The topological Markov shift $(\{T^{[2]}\},\sigma)$ is merely the higher 2-block system of $(\{T\},\sigma)$. Similarly we can form still higher edge graphs to obtain all the higher block systems $(\{T^{[n]}\},\sigma)$.

Using the notion of directed graph we can also define a dynamical system (X,σ) for arbitrary nonnegative integral matrices $T = (t_{ij})$ in the following manner. From i to j draw t_{ij} directed edges and label each with a distinct symbol. Let us again use the notation $T^{[2]}$ to designate the directed edge graph. Then $T^{[2]}$ is a zero-one transition matrix, so we can form the dynamical system $(\{T^{[2]}\},\sigma)$ which serves as a definition of a topological Markov shift given by a matrix in which appear positive integers larger than one.

Sometimes we must deal with dynamical systems $(\{T\},\sigma^p)$ involving a higher power of the shift. In order to apply the results as they are expressed in Section VI we must represent it as a first power and to this end we have the following theorem.

Theorem 5.1: If the pth matrix power $T^{[k]^p}$ of the kth higher edge graph $T^{[k]}$ for a transition matrix T is zero-one (which is always the case for $k = p$), then $(\{T\},\sigma^p\} \approx (\{T^{[k]^p}\},\sigma)$ with the conjugacy given by a canonical map like in Section III. Alternatively if T^p is not zero-one, then its edge graph $T^{p[2]}$ defined above is zero-one and $(\{T\},\sigma^p) \approx (\{T^{p[2]}\},\sigma)$.

A subshift which is a finite-to-one homomorphic image of a topological Markov shift is called a *sofic system*. A sofic system need not be a subshift of finite type (these systems were studied in [9], [10], [50]). However, a subshift which is an isomorphic image of a subshift of finite type is again a subshift of finite type. Sometimes, when the transition matrix specifying a topological Markov shift is large, we can take advantage of the above fact by specifying the system by an isomorphism (invertible k-block map) from a system given by a much smaller transition matrix, thus reducing the overall complexity of the description. Symbols in sequences in the domain of the above isomorphisms are sometimes called *channel states* and symbols in sequences of the range, *channel symbols*. An isomorphism of a shift of finite type is sometimes called a *deterministic channel with finite memory* and a homomorphism with a sofic image which is not a shift of finite type, a *deterministic channel with infinite memory*. We remark that the relation of the above terminology to that in engineering literature is a bit blurred. For example, whether the word, channel, should refer to a mapping, its image, or both is nebulous. We shall not dwell further on these pedantic difficulties, except to say that (W, σ) was called a channel in Section IV because the constraints on W are typically specified by defining it as the homomorphic image of a shift of finite type.

We introduce some useful terminology with regard to topological Markov shifts. We say j is a (T-admissible) *successor* of i, or equivalently the transition i to j is *allowable* (under T), and write $i \to j$, if $t_{ij} = 1$. We also say in this case, i is a (T-admissible) *predecessor* of j. We denote the successors of i by the set $T(i) = \{j_1, \cdots, j_{|T(i)|}\}$. The transpose matrix T^* defines another transition matrix in which the roles of predecessor and successor have been interchanged. Observe that $(\{T^*\}, \sigma)$ is isomorphic with $(\{T\}, \sigma^{-1})$.

Definition 5.1: T is said to be *irreducible* if for every $i, j \in \mathcal{C}$ there exists a positive integer n (depending on i, j) such that $t_{ij}^{(n)} > 0$, i.e., there exists $i_1, \cdots, i_{n-1} \in \mathcal{C}$ such that $i = i_0 \to i_1 \to \cdots \to i_{n-1} \to i_n = j$.

Definition 5.2: We shall call T *aperiodic* if there exists $n > 0$ such that $T^n > 0$, i.e., $t_{ij}^{(n)} > 0$ for all i, j (n being independent of i, j).

Definition 5.3: The greatest common divisor (gcd) of the set $\{n\colon t_{ii}^{(n)} > 0,\ i \in \mathcal{C},\ n = 1, 2, \cdots\}$ of cycle lengths is called the *period* of T.

Theorem 5.2 [21, pp. 65], [97]: T is aperiodic if and only if T is irreducible and has period 1.

Theorem 5.3 [4, p. 19], [24, p. 15]: A topological Markov shift $(\{T\}, \sigma)$ is nonwandering transitive if and only if T is irreducible.

Theorem 5.4 [4, p. 19]: A topological Markov shift $(\{T\}, \sigma)$ is aperiodic if and only if T is aperiodic.

Definition 5.4: A subalphabet $\mathcal{C}' \subset \mathcal{C}$ under transitions T' inherited from T is called an *irreducible component* if

i) $i \in \mathcal{C} \Rightarrow T(i) \subset \mathcal{C}'$
ii) $i, j \in \mathcal{C}' \Rightarrow \exists i_1, \cdots, i_{n-1} \in \mathcal{C}'$ such that $i = i_0 \to i_1 \to \cdots i_{n-1} \to i_n = j$.

Theorem 5.5 [4, p. 21]: If $T(i) \neq \varnothing$ for every $i \in \mathcal{C}$, then there exists an irreducible component.

The number $N(n)$ of T-admissible n-blocks is given by

$$N(n) = \sum_{i,j=1}^{N} t_{i,j}^{(n-1)}, \qquad n \geq 2. \tag{5.1}$$

It follows from the Perron–Frobenius theory of nonnegative matrices that there exist positive constants a, b such that

$$a\lambda^n \leq N(n) \leq b\lambda^n, \tag{5.2}$$

where λ is the largest positive characteristic value (spectral radius) of T. Thus from (3.1)

$$h(\{T\}, \sigma) = \log \lambda. \tag{5.3}$$

Let us address the problem of topological conjugacy. Besides the topological entropy invariant some stronger ones are known which are contained in the following theorems.

Theorem 5.6 [31]: If $(\{T_1\}, \sigma)$ and $(\{T_2\}, \sigma)$ are two equal entropy topological Markov shifts with the first a factor of the second, then the block of the Jordan canonical form of T_1 with nonzero characteristic values is a principal submatrix of that of T_2.

Corollary 5.7 [40]: In Theorem 5.6 the characteristic polynomial of T_1 divides that of T_2 when the monomial factors are deleted.

We also have an algebraic characterization of topological conjugacy.

Theorem 5.8 [51]: $(\{T_1\}, \sigma) \approx (\{T_2\}, \sigma)$ if and only if there exists nonnegative integral rectangular matrices A_i, B_i, $i = 1, \cdots, n$, for some n such that $A_{i+1}B_{i+1} = B_iA_i$, $i = 1, \cdots, n-1$, $T_1 = A_1B_1$ and $T_2 = B_nA_n$.

Using Theorem 5.8 it is easy to construct matrices T_1, T_2 with the same Jordan canonical form such that $(\{T_1^{[2]}\}, \sigma) \not\approx (\{T_2^{[2]}\}, \sigma)$. For example $T_1 = \begin{pmatrix} 5 & 4 \\ 1 & 1 \end{pmatrix}$ and $T_2 = \begin{pmatrix} 5 & 2 \\ 2 & 1 \end{pmatrix}$. Assuming conjugacy it would follow from Theorem 5.8 that there exists an integral 2×2 matrix S such that $ST_1 = T_2S$ and $\det S = 1$. However, an easy computation shows that 2 divides $\det S$, a contradiction. Thus the invariants presented above are not complete ones for topological conjugacy. In fact, the major unsolved problem in symbolic dynamics is to give a finite procedure for determining when two shifts of finite type are topologically conjugate. Possibly there is none. Also unsolved is the following conjecture which is still a far cry from a finite procedure.

Conjecture 5.1 [51]: $(\{T_1\}, \sigma) \approx (\{T_2\}, \sigma)$ if and only if there exists a positive integer l and nonnegative integral matrices A, B such that $AT_1 = T_2A$, $T_1B = BT_2$, $T_1^l = AB$, and $T_2^l = BA$.

The situation for determining finite equivalence or almost one-to-one finite equivalence is just the opposite. We do have a finite procedure which comes down to checking whether transition matrices have the same largest characteristic value. The completeness of topological entropy for finite equivalence and almost one-to-one finite equivalence is revealed in the following theorems.

Theorem 5.9 [4], [45]: Let $(X, \sigma), (Y, \sigma)$ be two nonwandering transitive subshifts of finite type, i.e., their transition matrices are irreducible. Then $(X, \sigma) \sim (Y, \sigma)$ if and only if $h(X, \sigma) = h(Y, \sigma)$.

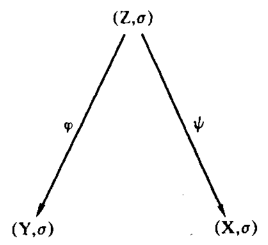


Fig. 5. Finite equivalence diagram (same as Fig. 2, reproduced of convenience).

Theorem 5.10 [4]: Let (X,σ) and (Y,σ) be two aperiodic subshifts of finite type. Then $(X,\sigma)\overset{\star}{\sim}(Y,\sigma)$ if and only if $h(X,\sigma)=h(Y,\sigma)$. Furthermore in [4], [45] methods are given for constructing the associated factor maps which are depicted in Fig. 5.

In the constructions one of the factor maps is right resolving and the other is left—one is free to choose which. We usually draw the right and left resolving maps on the corresponding side of the diagram.

The special case where (X,σ) is the full N-shift and $h(Y,\sigma)=h(X,\sigma)=\log N$, N an integer, was treated in [1]. Marcus [37] showed how to achieve an invertible map φ, which is not always possible if (X,σ) is not a full N-shift. If we select a right resolving ψ, then from Marcus' result we obtain a right resolving finite block map $\varphi^{-1}\psi$, the very thing needed to construct an encoder and decoder of Section IV.

In applications we generally have $h(W,\sigma)>h(X,\sigma)$, so we must find a subsystem $(Y,\sigma)\subset(W,\sigma)$ such that $h(Y,\sigma)=h(X,\sigma)$. This problem was not addressed by Marcus, but it can be done by strengthening his result, which is the main theorem of this work.

VI. Method of Symbol Splitting

Main Theorem 6.1: Let (X,σ) be the full N-shift for an integer $N\geq 2$, i.e., $X=\{S\}$ where S is an $N\times N$ matrix all of whose entries are one. If T is an $M\times M$ irreducible transition matrix such that $h(\{T\},\sigma)\geq h(\{S\},\sigma)=\log N$, then there exists by construction an irreducible transition matrix $\hat{T}$ with row sum N, an invertible left resolving 1-block factor map (isomorphism) φ of $(\{\hat{T}\},\sigma)$ onto a subshift of finite type $(Y,\sigma)\subset(\{T\},\sigma)$, and a right resolving 2-block factor map ψ of $(\{\hat{T}\},\sigma)$ onto $(\{S\},\sigma)$. The composition $\varphi^{-1}\psi$ is a right resolving factor map of (Y,σ) onto (X,σ).

Proof: The plan is to construct from T a matrix with row sum $\geq N$, then delete excess transitions, that is change some entries from 1 to 0, in order to get a matrix with row sum N.

We have by hypothesis that $h(\{T\},\sigma)\geq\log N$, so $\lambda\geq N$ where λ is the spectral radius of T. From the Perron–Frobenius theory of nonnegative matrices [21], [47] there exists a column vector $v=(v_i)_{1\leq i\leq M}$ which we call an *approximate characteristic vector*, satisfying

$$Tv\geq Nv,\qquad v>0, \tag{6.1}$$

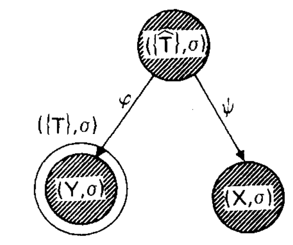


Fig. 6. Coding scheme.

where inequality here means componentwise inequality. Since T has integer entries, in fact zeros and ones, we can satisfy (6.1) with integers $v_i>0$ and furthermore with $\gcd(v_i)=1$. (See Appendix for a method of solving this integer programming problem.)

If all $v_i=1$, then T itself is the sought after matrix; so we assume $\max v_i>1$, which also implies $\min v_i<\max v_i$. Let us call v_i the *weight* of i.

Consider the set $T(i)=\{j_1,\cdots,j_{|T(i)|}\}$ of successors of i. For each i, $1\leq i\leq M$, choose a disjoint partition $\alpha=\alpha_i=\{A_1,\cdots,A_{|\alpha|}\}$ of $T(i)$ where the following conditions hold

$$\sum_{j\in A_k} v_j=0\ (\mathrm{mod}\ N),\qquad 1\leq k\leq|\alpha|-1, \tag{6.2}$$

$$v_i-\sum_{k=1}^{|\alpha|-1}\sum_{j\in A_k} v_j/N\geq 0. \tag{6.3}$$

We dispense with the subscript on α when it is clear from context. Usually we pick partitions of $T(i)$ with largest possible $|\alpha|$, but sometimes we must settle for $|\alpha|=1$. Nevertheless, we show there exists an index i_0 for which

$$|\alpha_{i_0}|>1, \tag{6.4}$$

namely, take i_0 for which $v_{i_0}=\max v_i$ and $T(i_0)$ contains an index, say j_1, such that $v_{j_1}<v_{i_0}$. Such an i_0 exists; for otherwise T would be reducible because the indices of maximum weight would "circulate" only among themselves. We are free to order symbols so that $i_0=1$. From (6.1) follows

$$v_1|T(1)|>\sum_{j\in T(1)} v_j\geq Nv_1$$

from which we conclude $|T(1)|>N$. Consider next the following sums modulo N:

$$\begin{gathered} v_{j_1}\\ v_{j_1}+v_{j_2}\\ v_{j_1}+\cdots+v_{j_N}. \end{gathered}$$

Either there are N distinct values and one of them is 0 (mod N), or two repeat, in which case their difference $v_{j_p+1}+\cdots+v_{j_q}=0$ (mod N) where $1\leq p<q\leq N$. Thus we can find a nonempty subset $A_1\not\subseteq T(1)$, $|A_1|\leq N<|T(1)|$, such that

$$\sum_{j\in A_1} v_j=0\ (\mathrm{mod}\ N)$$

and

$$\sum_{j\in A_1} v_j<Nv_1,$$

equivalently

$$v_1 > \sum_{j \in A_1} v_j/N.$$

This last inequality holds because state one is heaviest and either $|A_1| < N$, or $|A_1| = N$ and $j_1 \in A_1$, j_1 being lighter than state 1.

Next we "split" i into new symbols $i^1, \cdot, i^{|\alpha_i|}$, which we call *offspring* of i. With respect to the alphabet $\mathcal{C}^1 = \{i^k: 1 \le k \le |\alpha_i|, 1 \le i \le M\}$ of new symbols ordered in some fashion, we obtain a new transition matrix T' specified by the transitions

$$i^k \to j^l, \qquad j \in A_k \subset T(i),$$

where j^l are offspring of j, i.e., $T'(i^k) = \{j^l: j \in A_k\}$.

We define a 1-block map φ of $\{T'\}$ to $\{T\}$ by

$$\varphi i^k = i.$$

This map is obviously onto. Suppose i is a predecessor of j under transitions of T. Then, because no elements of α_i overlap, there is a unique offspring i^k of i such that $T'(i^k)$ contains the offspring of j. This fact establishes that φ is left resolving and that every 2-block (i, j) is a resolvable one. Consequently by Definition 3.2, φ is invertible, in other words, an isomorphism.

We form an approximate characteristic *subvector* v' with components v_{i^k}, $1 \le k \le |\alpha_i|$, $1 \le i \le M$, for T' as follows:

$$v'_{i^k} \equiv \sum_{j \in A_k} v_j/N, \qquad 1 \le k \le |\alpha_i| - 1, \tag{6.5}$$

$$v'_{i^{|\alpha|}} \equiv v_i - \sum_{k=1}^{|\alpha|-1} v'_{i^k}. \tag{6.6}$$

We use the term "sub" above because, as we shall show, $T'v' \ge Nv'$ but v' may not be strictly positive. Actually it is an approximate characteristic vector for a subsystem as we shall see. We observe that (6.6) expresses the fact that the weight of i equals the total weight of its offspring among whom one, $i^{|\alpha|}$, may be weightless. From (6.2)–(6.6) we conclude

$$v'_{i^k} \text{ are integers,} \tag{6.7}$$

$$v'_{i^k} \ge 0, \qquad i^k \in \mathcal{Q}', \tag{6.8}$$

$$v'_{i^k} > 0,\ 1 \le k \le |\alpha_i| - 1; \quad \text{in particular} \quad v'_{1^1}, v'_{1^2} > 0, \tag{6.9}$$

$$v'_{i^k} = \sum_{j^l \in T'(i^k)} v'_{j^l}/N, \qquad 1 \le k \le |\alpha_i| - 1, \tag{6.10}$$

and

$$v'_{i^{|\alpha|}} \le \sum_{j^l \in T'(i^{|\alpha|})} v'_{j^l}/N. \tag{6.11}$$

Equations (6.7)–(6.10) are immediate, and (6.11) is derived as follows:

$$v'_{i^{|\alpha|}} = v_i - \sum_{k=1}^{|\alpha|-1} v'_{i^k} = v_i - \sum_{k=1}^{|\alpha|-1} \sum_{j \in A_k} v_j/N$$

$$= \frac{Nv_i - \sum_{k=1}^{|\alpha|-1} \sum_{j \in A_k} v_j}{N} \le \frac{\sum_{j \in T(i)} v_j - \sum_{j \in T(i) - A_{|\alpha|}} v_j}{N}$$

$$= \sum_{j \in A_{|\alpha|}} v_j/N = \sum_{j^l \in T'(i^{|\alpha|})} v'_{j^l}/N.$$

Next we take the subalphabet $\mathcal{C}'' = \{i^k \in \mathcal{C}', v'_{i^k} > 0\}$ along with transitions which we denote by T'', that are inherited from T' by deleting transitions $i^k \to j^l$ whenever v'_{i^k} or $v'_{j^l} = 0$, in other words, by crossing out rows and columns of T' corresponding to components of v' that vanish. From (6.10), (6.11) we see that $T''(i^k) \ne \varnothing$, $i^k \in \mathcal{C}''$. So by Definition 5.4 there exists an irreducible component, i.e., a further subalphabet $\bar{\mathcal{C}}$ along with an irreducible transition matrix $\bar{T}$ inherited from T''. From (6.10), (6.11) we have

$$v'_{i^k} \le \sum_{j^l \in T'(i^k)} v'_{j^l}/N = \sum_{j^l \in T''(i^k)} v'_{j^l}/N$$

$$= \sum_{j^l \in \bar{T}(i^k)} v'_{j^l}/N, \qquad i^k \in \bar{\mathcal{C}}.$$

If $\bar{v}$ is the restriction of v' to the components with indices in $\bar{\mathcal{C}}$, then

$$\bar{T}\bar{v} \ge N\bar{v}, \qquad \bar{v} > 0.$$

The system $(\{\bar{T}\}, \sigma)$ is a topological Markov subshift of $(\{T'\}, \sigma)$. The restriction of φ to $\{\bar{T}\}$ is a left resolving invertible 1-block map of $(\{\bar{T}\}, \sigma)$ onto the subshift $(\varphi\{\bar{T}\}, \sigma) \subset (\{T\}, \sigma)$ of finite type, and its inverse φ^{-1} a 2-block map.

From (6.6) follows that no offspring is heavier than its parent, and offspring are actually lighter when there are more than one of nonvanishing weight. We have proved that at least one index of maximum weight has been split into lighter ones: namely, 1 into $1^1, \cdots, 1^{|\alpha|}$, $|\alpha| \ge 2$. Thus, compared to T, either the maximum weight of symbols of T' or the number of symbols of maximum weight has been reduced. The same is true for T'' and $\bar{T}$.

We repeat this splitting process with the role of the new T played by the previous $\bar{T}$, continuing until a vector $\bar{v}$ is reached having all components equal one, at which point we terminate.

At each step we have a 1-block left resolving isomorphism φ of $(\{\bar{T}\}, \sigma)$ into $(\{T\}, \sigma)$ whose inverse φ^{-1} is a 2-block map. If there are n steps, then the resulting composition of isomorphisms yields a 1-block left resolving isomorphism, which we denote again by φ, of the final $(\{\bar{T}\}, \sigma)$ into the original $(\{T\}, \sigma)$ whose inverse φ^{-1} is an $(n+1)$-block map.

The final transition matrix $\bar{T}$ satisfies

$$\bar{T}\bar{v} \ge N\bar{v}$$

with all components of $\bar{v} = 1$. Thus $\bar{T}$ has row sum $\ge N$.

From (6.6) we conclude that the weight of the progeny of i at the final step is $\leq v_i$, the ith component of the original v of (6.1) (it would be equal if at each step the inequality $v' > 0$ held or T' were irreducible). This means that i ultimately gets split into $\leq v_i$ symbols, and the final $\overline{T}$ is an $\overline{M} \times \overline{M}$ matrix, where

$$\overline{M} \leq \sum_{i=1}^{M} v_i. \tag{6.12}$$

To form $\hat{T}$ we delete some excess transitions in the final $\overline{T}$ so that every row sum equals N. The resulting matrix may not be irreducible, so we take $\hat{T}$ to be an irreducible component. The alphabet $\hat{\mathcal{C}}$ for $\hat{T}$ will be a subset of $\overline{\mathcal{C}}$. The composition of isomorphisms constructed on $\{\overline{T}\}$ when restricted to $\{\hat{T}\}$ is the desired isomorphism φ, and $(Y, \sigma) = (\varphi\{\hat{T}\}, \sigma)$ is a subshift of $(\{T\}, \sigma)$ of finite type.

Since $\hat{T}$ has row sum N we can write $\hat{T}(i) = \{j_1, \cdots, j_N\}$ for each $i \in \mathcal{Q}_{\{\hat{T}\}}$ and define a 2-block map ψ of $(\{\hat{T}\}, \sigma)$ onto the full N-shift $(\{S\}, \sigma)$ by

$$\psi(i, j_k) = k, \qquad 1 \leq k \leq N. \tag{6.13}$$

It is clear by its definition that ψ is right resolving. □

Remark 6.1: There are many choices of ψ depending on how $\hat{T}(i)$ is ordered. If each symbol has the same ordinal number, whenever it appears as a successor, then ψ reduces to a 1-block map. A particular ψ may or may not have resolving blocks. A necessary condition for existence of ψ with resolving blocks is aperiodicity[1] of $\hat{T}$ (see [1]). An unsolved conjecture, called the "road coloring problem," is that it is also sufficient. The most general result so far appears in [43]. Nevertheless is any specific examples resolving blocks have always been easily found merely by hunting for them. Although a complete solution to the problem has been elusive, in [1] a method is given for solving the road coloring problem for some higher n-block system of an arbitrary aperiodic transition matrix with integer spectral radius. We use this to prove the following result.

Theorem 6.2: If T is aperiodic, then $\hat{T}$ can be constructed so that ψ has resolving blocks in its image.

Proof: We repeat the construction of Theorem 6.1 except that in choosing partitions α we insist that inequality hold in (6.3). Although now more steps may be required to achieve an approximate characteristic vector of all 1's, at the end of each, $\overline{T} = T'' = T'$ and the accompanying φ is an isomorphism of $\{\overline{T}\}, \sigma)$ onto $(\{T\}, \sigma)$. Thus the final $(\{\overline{T}\}, \sigma)$ is topologically conjugate to the original $(\{T\}, \sigma)$ by means of a composition of isomorphisms from each stage of the splitting process. As before, the final $\overline{T}$ has row sum $\geq N$. By invariance of aperiodicity under isomorphism $\overline{T}$ is aperiodic. From considerations treated in [4, pp. 14, 15] there exists a higher block system $\overline{T}^{[r]}$ whose graph contains the following configuration: two disjoint cycles, C_1: $i_1 \to i_2 \to \cdots \to i_p \to i_1$ and C_2: $j_1 \to \cdots \to j_{p+1} \to j_1$, of lengths p and $p+1$, joined by two simple paths, P_1: $i_a = k_1 \to k_2 \to \cdots \to k_m = j_b$, connecting C_1 to C_2, and P_2: $j_c = l_1 \to l_2 \cdots \to l_q = i_d$, connecting C_2 to C_1. At each node in the configuration at most two edges (transitions) have been used from those specified by $\overline{T}^{[r]}$. Because $\overline{T}^{[r]}$ has row sum $N \geq 2$ just like $\overline{T}$, we can and shall retain these edges when deleting excess transitions from $\overline{T}^{[r]}$. Furthermore, because $\overline{T}^{[r]}$ is irreducible, every node outside the configuration has at least one edge directed along a path to the configuration. These we also retain. We then delete excess transitions from the remaining ones specified by $\overline{T}^{[r]}$ to form a transition matrix of row sum N. This matrix may be reducible, but there is one irreducible component, which we denote by $\tilde{T}$, and its graph contains the above configuration. $\tilde{T}$ has row sum N. Also $\tilde{T}$ is aperiodic because it is irreducible and the gcd of cycle lengths is one, (there are cycles of length p and $p+1$). We now apply the results of [1] to construct a still higher block system $\hat{T}$ of $\tilde{T}$ (also having row sum N) for which a 2-block map ψ can be defined as in (6.13) having resolving blocks. □

[1]A necessary condition for aperiodicity of $\hat{T}$ is that T also be aperiodic. This is easily proved by considerations in [4, defn. (2.8); (3.15)].

VII. Run-Length Constraints

A constraint frequently encountered in engineering applications is one modeled by a subshift of finite type (Y, σ) of the full 2-shift (X, σ) with $\mathcal{Q}_x = \mathcal{Q}_y = \{0, 1\}$, the forbidden blocks of which are specified by lower and upper bounds on the number of consecutive 0 separating the 1. Such a system is said to have a $[d, k]$ *run-length constraint*, where d is the lower and k the upper bound. An explanation for the prominence of this sort of constraint will be given later.

The finite list of blocks designated by the above constraint consists of blocks

$$(1,1), (1,0,1), \cdots, \underbrace{(1,0,\cdots,0,1)}_{d-1}$$

and

$$\underbrace{[0,\cdots,0]}_{k+1}.$$

If $d = 0$, blocks of the first type do not appear in the list; and if $k = \infty$, the second type block is absent. To represent this system as a topological Markov shift according to the recipe of Section V requires an $N \times N$ transition matrix where N is the cardinality of the set of admissible k-blocks. A more convenient method of representing the $[d, k]$ constraint is by means of a smaller, actually $(k+1) \times (k+1)$, transition matrix together with a 2-block isomorphism map. Let $\mathcal{Q}_{\{T\}} = \{0, 1, \cdots, k\}$ and T be given by

$$\begin{cases} i \to i+1, & 0 \leq i < d, \\ i \to i+1, 0, & d \leq i < k, \\ i \to 0, & i = k. \end{cases}$$

We define a 1-block map ϑ of $\{T\}$ onto a subshift of (X, σ) by

$$\vartheta(i) = \begin{cases} 0, & 1 \leq i \leq k, \\ 1, & i = 0. \end{cases}$$

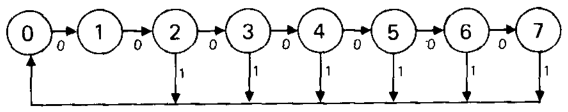


Fig. 7. State transition diagram of (2, 7) system.

TABLE I
RATIONAL APPROXIMATIONS TO $h = h[d, k]$

d, k			
0, 1	$2/3 <$	$9/13 < h < 7/10$	< 1
0, 2		$7/8 < h < 22/25$	$< 8/9$
0, 3		$17/18 < h < 18/19$	
0, 4		$39/40 < h < 79/81$	$< 40/41$
0, ∞		$h = 1$	
1, 2	$2/5 <$	$15/37 < h < 13/32$	$< 1/2$
1, 3	$1/2 <$	$11/20 < h < 5/9$	< 1
1, 4	$1/2 < 3/5 <$	$8/13 < h < 5/8$	$< 2/3 < 1$
1, 5	$1/2 <$	$13/20 < h < 28/43$	$< 2/3$
1, 6	$2/3 <$	$95/142 < h < 93/139$	< 1
1, 7	$2/3 <$	$36/53 < h < 17/25$	< 1
1, ∞		same as [0, 1]	
2, 6		$118/237 < h < 1/2$	
2, 7	$1/2 <$	$15/29 < h < 14/27$	< 1
2, ∞	$1/2 <$	$11/20 < h < 5/9$	< 1
3, 6		$1/3 < h < 3/8$	
3, 7	$2/5 <$	$15/37 < h < 13/32$	$< 1/2$
3, ∞	$6/13 <$	$13/28 < h < 7/15$	$< 1/2$
4, 8		$1/3 < h < 11/32$	$< 1/2$
4, 15		$1/3 < h < 2/5$	
4, 16		$2/5 < h < 99/247$	$< 1/2$
4, ∞		$2/5 < h < 13/32$	$< 1/2$
5, 12		$1/3 < h < 31/92$	$< 1/2$
5, ∞	$1/3 <$	$17/47 < h < 4/11$	$< 1/2$
9, ∞		$1/4 < h < 6/23$	$< 1/3$

In Fig. 7 we depict the directed graph of T for $[d, k] = [2, 7]$ with edges leading into nodes i labeled by the values of ϑ.

The following statements are easily established. ϑ is right resolving. The 1-block 1 is a resolving block. In the image of ϑ the symbol 1 appears at least once in every block of length $k + 1$. Thus ϑ is an isomorphism by Remark 3.1. The system $(\vartheta\{T\}, \sigma) \subset (X, \sigma)$ coincides with the subshift of finite type (Y, σ) specified by the $[d, k]$ constraint.

The entropy $h[d, k] \equiv h(Y, \sigma) = h(\{T\}, \sigma)$ of such $[d, k]$-constrained systems was computed in [49] and tabulated in [41]. It is given by $h[d, k] = \log_2 \lambda$, where λ is the largest positive root of the characteristic equation of T, which is

$$\begin{cases} z^{k+1} - \dfrac{z^{k+1-d} - 1}{z - 1} = 0 & \text{for } k < \infty, \\ z - \dfrac{z^{1-d}}{z - 1} = 0 & \text{for } k = \infty. \end{cases} \tag{7.1}$$

We cannot code between (X, σ) and any subshift of $(\{T\}, \sigma)$ because $\lambda < 2$, i.e., $h(\{T\}, \sigma) < h(X, \sigma)$. However, we can code between (X, σ^p) and some subshift of $(\{T\}, \sigma^q)$ (in other words, at a rate $r = p/q < 1$, of p source symbols to q channel symbols) whenever $h(X, \sigma^p) \leq h(\{T\}, \sigma^q)$, i.e.,

$$p/q \leq \log_2 \lambda, \tag{7.2}$$

which follows from Theorem 2.4 and (3.2), (5.2). Subject to the limitation imposed by (7.2) we want to select p/q as large as we are willing to pay for in terms of complexity. The well-known theory of continued fractions can be applied to find "best" rational approximations p/q to $\log_2 \lambda$. We present samples in Table I.

From Table I we observe that the [1, 7] constraint admits a code with rate 2 : 3 while [2, 7] admits one with rate 1 : 2. It would therefore seem that coding for the first constraint can be done more efficiently than for the second. However, physical constraints explained below make [2, 7] more attractive with regard to information density.

The importance of run-length constraints is due to a particular way information is ascribed to the behavior of certain physical devices having two states. In the writing (sending) mode we cause transition from one state to the other to occur at specified time intervals. In the reading (receiving) mode we measure the time intervals between transitions. This measurement is made in terms of integral multiples of a unit of time we call a *clock unit*. If there are $l + 1$ clock units between two transitions, then we assign the symbol 1 to the first one and 0 to the remaining l. The appearance of $[d, k]$ constraints with this scheme is natural. In order to detect transitions properly, they cannot occur too close together, so a minimum allowable time, say $t_{\min}$, between them is prescribed. If $t_{\min} = d + 1$ clock units, then we get the lower bound d for run-lengths of the 0. On the other hand, clocks are imperfect devices. They drift and lose power to discriminate the number of clock units between transitions which are far apart. Requiring that transitions be separated by no more than $k + 1$ clock units places the upper-bound k on runs of the 0.

The *information density* $E = E[d, k]$ is not the coding rate $r = p/q$ but the amount of information per unit time, and this is given by

$$E = p/q \frac{d + 1}{t_{\min}} \tag{7.3}$$

for the above scheme. Applying (7.3) we have $E[2, 7] = (3/2)(1/t_{\min})$ and $E[1, 7] = (4/3)(1/t_{\min})$ so that given the same $t_{\min}$ for each we have $E[2, 7] = (9/8)E[1, 7]$. In this case, the clock unit for the [2, 7] is 2/3 of that for [1, 7]. Consequently, in order to achieve the higher density a more accurate clock is required. In fact, with a perfect clock we can get an infinite[2] amount of information per unit time.

Substituting values from Table I into (7.3) and taking into account the low complexity indicated by a rate $r = 1/2$ singles out the [2, 7] constraint as a likely candidate for

[2]What we mean is that $E(d, k)$ can be made arbitrarily large by choosing d and k sufficiently large. This follows from

$$\lim_{k \to \infty} E(d, k) = E(d, \infty) = \frac{(d + 1) \log \lambda_d}{t_{\min}}$$

where λ_d is the largest root of $x^{d+1} - x^d - 1 = 0$, another form of (7.2). This root satisfies

$$\left(\frac{(1 - \epsilon)d}{\log d}\right)^{1/d} \leq \lambda_d \leq \left(\frac{(1 + \epsilon)d}{\log d}\right)^{1/d}$$

for large d. Thus $E(d, \infty)$ grows like constant times $\log d$.

applications. Indeed IBM Disk Files, models 3370, 3380, incorporate a code based on it. These products are magnetic recording devices. The transitions are magnetic ones occurring in circular tracks on disks. Time is equated to distance along a track by steady rotation of a disk. We can therefore treat information in a spatial context in exactly the same way as the temporal one described above. The quantity $t_{\min}$ can be equated to the reciprocal of maximum density of magnetic transitions (flux changes) on a track, and formula (7.3) can be turned into a measure of recorded information density.

The issues involved in selecting the most appropriate constraint are very complex indeed. After balancing factors such as complexity of available codes, accuracy of inexpensive clocking schemes, density of detectable flux changes, the [2,7] constraint was chosen for the above products. A code for it was available. According to methods in [15] a code between [2,7] constrained sequences and unconstrained sequences of the 0 and the 1 was gotten with rate 1 : 2 from a set of correspondences like Table II (see [42]).

TABLE II
CORRESPONDENCE BETWEEN PREFIX FREE LISTS

11	$\leftarrow\rightarrow$	0100
10	$\leftarrow\rightarrow$	1000
011	$\leftarrow\rightarrow$	000100
010	$\leftarrow\rightarrow$	001000
000	$\leftarrow\rightarrow$	100100
0011	$\leftarrow\rightarrow$	00100100
0010	$\leftarrow\rightarrow$	00001000

On the left side of Table II is a uniquely decipherable prefix free list of blocks from unconstrained sequences and on the right, one obeying the [2,7] constraint. A uniquely *decipherable prefix free list* is a collection of blocks, no one of which is the beginning of another; consequently every sequence can be partitioned into them in only one way. It is an interesting exercise to show that the system of all possible concatenations of blocks on the right side of Table II, which generates another kind of dynamical system called a renewal system, is also a subshift of finite type. There are 24 permutations of the above correspondences from which a code can be gotten. One of these is employed in the aforementioned magnetic storage products. We remark in passing that an encoder and decoder as in Section IV can be constructed from Table II. The one used in the IBM products [13] has a slightly simpler implementation than those described by Tables XI, XVII, and (9.5). Correspondence tables of uniquely decipherable lists, not necessarily prefix free, have been discovered for other constraints [29]. The question arises: how does one find such tables achieving a desired rate for arbitrary subshifts of finite type and from which simple encoder/decoder automata can be made? Is there a method? Our approach is an alternate solution to the problem. We remark that when one compares encoders and decoders produced by our method and the best of those constructed from any other method, the complexity of electronic design is about the same.

VIII. SYMBOL SPLITTING FOR THE [2,7] CONSTRAINT

We now carry out the method of symbol splitting for the [2,7] run-length constraint and derive an encoder and decoder from it.

Let the source (X, σ) be the full 2-shift and the channel (Y, σ) the subshift of finite type given by the [2,7] constraint with $\mathcal{Q}_X = \mathcal{Q}_Y = \{0,1\}$. We use the specification given in Section VII and depicted in Fig. 7, namely, $(Y, \sigma) = (\vartheta\{T\}, \sigma)$ where $\mathcal{Q}_{\{T\}} = \{0,1,2,3,4,5,6,7\}$,

$$T = \begin{array}{llll} 0 \rightarrow & & 1 \\ 1 \rightarrow & & 2 \\ 2 \rightarrow & 0 & 3 \\ 3 \rightarrow & 0 & 4 \\ 4 \rightarrow & 0 & 5 \\ 5 \rightarrow & 0 & 6 \\ 6 \rightarrow & 0 & 7 \\ 7 \rightarrow & 0 & \end{array},$$

and ϑ is the 1-block map

$$\vartheta(i) = \begin{cases} 0, & 1 \le i \le 7, \\ 1, & i = 0. \end{cases} \tag{8.1}$$

(Here i, j are symbols of $\mathcal{Q}_{\{T\}}$ which differ from their ordinal numbers by 1.)

From Table I we see that this constraint admits coding at rate 1 : 2. Thus we want to code between the full 2-shift (X, σ) and the system $(\{T\}, \sigma^2)$ involving the second power of σ. In order to apply Section VI we use Theorem 5.1 to represent $(\{T\}, \sigma^2)$ as a topological Markov shift involving the first power of σ. In this case it conveniently turns out that T^2 is a zero-one matrix so we get $(\{T\}, \sigma^2) \approx (\{T^2\}, \sigma)$. Specifying T^2 by its transitions we get

$$T^2 = \begin{cases} 0 \rightarrow & & & 2 \\ 1 \rightarrow & 0 & & 3 \\ 2 \rightarrow & 0 & 1 & 4 \\ 3 \rightarrow & 0 & 1 & 5 \\ 4 \rightarrow & 0 & 1 & 6 \\ 5 \rightarrow & 0 & 1 & 7 \\ 6 \rightarrow & 0 & 1 & \\ 7 \rightarrow & & 1 & \end{cases} \tag{8.2}$$

One can easily verify that $v = (2,3,4,4,3,3,1,1)$, read as a column vector, is an approximate characteristic vector for T^2, i.e., $T^2 v \ge 2v$ or equivalently

$$\sum_{j \in T^2(i)} v_j \ge 2v_i,$$

which is the more convenient form for use with (8.2).

In accordance with Section VI we construct $\hat{T}^2$ (not to be confused with $(\hat{T})^2$), with alphabet $\{0^1, 0^2, 1^1, 1^2, 1^3, 2^1, 2^2, 2^3, 2^4, 3^1, 3^2, 3^3, 3^4, 4^1, 4^2, 4^3, 5^1, 5^2, 5^3, 6^1, 7^1\}$, which is obtained by splitting the symbols of $\mathcal{Q}_{\{T^2\}} = \mathcal{Q}_{\{T\}}$. We introduce a notation which keeps track of the genealogy of symbols at each stage of the splitting process and at the same time indicates their weight. We replace a symbol $i \in \mathcal{Q}_{\{T^2\}}$ by i^{1, v_i}. At each step the parent symbols are of

TABLE III
SPLITTING STEP 0

$0 = 0^{1,2}$	$\rightarrow$			$2^{1,4}$
$1 = 1^{1,3}$	$\rightarrow$	$0^{1,2}$		$3^{1,4}$
$2 = 2^{1,4}$	$\rightarrow$	$0^{1,2}$	$1^{1,3}$	$4^{1,3}$
$3 = 3^{1,4}$	$\rightarrow$	$0^{1,2}$	$1^{1,3}$	$5^{1,3}$
$4 = 4^{1,3}$	$\rightarrow$	$0^{1,2}$	$1^{1,3}$	6^1
$5 = 5^{1,3}$	$\rightarrow$	$0^{1,2}$	$1^{1,3}$	7^1
$6 = 6^1$	$\rightarrow$	$0^{1,2}$	$1^{1,3}$	
$7 = 7^1$	$\rightarrow$		$1^{1,3}$	

TABLE IV
SPLITTING STEP 1

				$0^{1,2}$	$\rightarrow$	$2^{1,4}$		$=$	2^1	$2^{2,4}$		
5^1	4^1	3^1	2^1	1^1	$\rightarrow$	$0^{1,2}$		$=$	$0^{1,2}$			
				$1^{2,3}$	$\rightarrow$	$3^{1,4}$		$=$	3^1	$3^{2,4}$		
				$2^{2,4}$	$\rightarrow$	$1^{1,3}$	$4^{1,3}$	$=$	1^1	$1^{2,3}$	4^1	$4^{2,3}$
				$3^{2,4}$	$\rightarrow$	$1^{1,3}$	$5^{1,3}$	$=$	1^1	$1^{2,3}$	5^1	$5^{2,3}$
				$4^{2,3}$	$\rightarrow$	$1^{1,3}$	6^1	$=$	1^1	$1^{2,3}$	6^1	
				$5^{2,3}$	$\rightarrow$	$1^{1,3}$	7^1	$=$	1^1	$1^{2,3}$	7^1	
				6^1	$\rightarrow$	$0^{1,2}$	$1^{1,3}$	$=$	$0^{1,2}$	1^1	$1^{2,3}$	
				7^1	$\rightarrow$	$1^{1,3}$		$=$	1^1	$1^{2,3}$		

the form $i^{j,k}$ having weight $v_{i^{j,k}} = k - j + 1$. Their offspring become symbols of the form $i^{j',k'}$, where $j \leq j'$, $k' \leq k$, having weight $k' - j' + 1$. In addition we abbreviate $i^{j,j}$ by i^j. Thus we start with the assignment of Table III.

Observe that $2^{1,4}$ has successors $0^{1,2}, 1^{1,3}, 4^{1,3}$. This successor set is to be partitioned according to (6.2) and (6.3) into $\{0^{1,2}\}$ and $\{1^{1,3}, 4^{1,3}\}$. We then split $2^{1,4}$ into two offsprings 2^1 and $2^{2,4}$ and form transitions

$$\begin{array}{lll} 2^1 & \rightarrow & 0^{1,2}, \\ 2^{2,4} & \rightarrow & 1^{1,3} \quad 4^{1,3}. \end{array}$$

At the end of the first step $1^{1,3}$ will have been split into 1^1, $1^{2,3}$, and $4^{1,3}$ into 4^1, $4^{2,3}$, so that the above transitions become

$$\begin{array}{lll} 2^1 & \rightarrow & 0^{1,2}, \\ 2^{2,4} & \rightarrow & 1^1 \quad 1^{2,3} \quad 4^1 \quad 4^{2,3}. \end{array}$$

In the next stage the successors of $2^{2,4}$ will be partitioned into $\{1^1, 4^1\}$, $\{1^{2,3}\}$, $\{4^{2,3}\}$ and $2^{2,4}$ will be split into 2^2, 2^3, and 2^4.

From now on when presenting transition tables, we take advantage of the following economy. Whenever several states have the same successors, we write them together on the same line to the left of the arrow, e.g., line two in Table IV.

The first step of the splitting process yields Table IV. Continuing we have Table V and VI. We remark that there are other choices for successors of some of the states. For example, we could have had

$$\begin{array}{lll} 0^1 & \rightarrow & 2^1 \quad 2^2, \\ 0^2 & \rightarrow & 2^3 \quad 2^4, \end{array}$$

etc. However, those of Table VI have been carefully selected due to considerations taken up in Section IX.

In order to form $\hat{T}^2$ having row sum 2 we drop transitions $6^1 \rightarrow 0^1, 0^2, 1^1$, and $7^1 \rightarrow 1^1$ to get Table VII.

TABLE V
SPLITTING STEP 2

				$0^{1,2}$	$\rightarrow$	2^1	$2^{2,4}$		$=$	2^1	2^2	2^3	2^4
5^1	4^1	3^1	2^1	1^1	$\rightarrow$	$0^{1,2}$			$=$	$0^{1,2}$			
				$1^{2,3}$	$\rightarrow$	3^1	$3^{2,4}$		$=$	3^1	3^2	3^3	3^4
	5^2	4^2	3^2	2^2	$\rightarrow$	$1^{2,3}$			$=$	$1^{2,3}$			
				2^3	$\rightarrow$	1^1	4^1		$=$	1^1	4^1		
				2^4	$\rightarrow$	$4^{2,3}$			$=$	4^2	4^3		
				3^3	$\rightarrow$	1^1	5^1		$=$	1^1	5^1		
				3^4	$\rightarrow$	$5^{2,3}$			$=$	5^2	5^3		
				4^3	$\rightarrow$	1^1	6^1		$=$	1^1	6^1		
				5^3	$\rightarrow$	1^1	7^1		$=$	1^1	7^1		
				6^1	$\rightarrow$	$0^{1,2}$	1^1	$1^{2,3}$	$=$	$0^{1,2}$	1^1	$1^{2,3}$	
				7^1	$\rightarrow$	1^1	$1^{2,3}$		$=$	1^1	$1^{2,3}$		

TABLE VI
SPLITTING STEP 3

				0^1	$\rightarrow$	2^2	2^4		$=$	2^2	2^4			
				0^2	$\rightarrow$	2^1	2^3		$=$	2^1	2^3			
5^1	4^1	3^1	2^1	1^1	$\rightarrow$	$0^{1,2}$			$=$	0^1	0^2			
				1^2	$\rightarrow$	3^1	3^3		$=$	3^1	3^3			
				1^3	$\rightarrow$	3^2	3^4		$=$	3^2	3^4			
	5^2	4^2	3^2	2^2	$\rightarrow$	$1^{2,3}$			$=$	1^2	1^3			
				2^3	$\rightarrow$	1^1	4^1		$=$	1^1	4^1			
				2^4	$\rightarrow$	4^2	4^3		$=$	4^2	4^3			
				3^3	$\rightarrow$	1^1	5^1		$=$	1^1	5^1			
				3^4	$\rightarrow$	5^2	5^3		$=$	5^2	5^3			
				4^3	$\rightarrow$	1^1	6^1		$=$	1^1	6^1			
				5^3	$\rightarrow$	1^1	7^1		$=$	1^1	7^1			
				6^1	$\rightarrow$	$0^{1,2}$	1^1	$1^{2,3}$	$=$	0^1	0^2	1^1	1^2	1^3
				7^1	$\rightarrow$	1^1	$1^{2,3}$		$=$	1^1	1^2	1^3		

TABLE VII
ENCODING STATE TRANSITION TABLE

							$x_n = 0$	$x_n = 1$
					z_{n-1}		z_n	z_n
					0^1	$\rightarrow$	2^2	2^4
					0^2	$\rightarrow$	2^1	2^3
	5^1	4^1	3^1	2^1	1^1	$\rightarrow$	0^2	0^1
					1^2	$\rightarrow$	3^1	3^3
					1^3	$\rightarrow$	3^2	3^4
7^1	6^1	5^2	4^2	3^2	2^2	$\rightarrow$	1^2	1^3
					2^3	$\rightarrow$	4^1	1^1
					2^4	$\rightarrow$	4^2	4^3
					3^3	$\rightarrow$	5^1	1^1
					3^4	$\rightarrow$	5^2	5^3
					4^3	$\rightarrow$	6^1	1^1
					5^3	$\rightarrow$	7^1	1^1

The mapping φ: $\{\hat{T}^2\} \rightarrow \{T^2\}$ is defined by dropping superscripts. The mapping ψ according to (6.13) is a 2-block map with values 0 or 1 depending on transitions into the respective first or second column on the right of Table VII. For example $\psi(0^1, 2^2) = 0$ and $\psi(0^1, 2^4) = 1$, etc. The map ϑ extends to a one-block map of $\{T^2\} \rightarrow Y^{[2]}$, which means that ϑ maps $\mathcal{Q}_{\{T^2\}}$ onto $\mathcal{Q}_{Y^{[2]}}$ according to

$$\vartheta(i) = \begin{cases} 0 \quad 1 & i = 0 \\ 1 \quad 0 & i = 1 \\ 0 \quad 0 & 2 \leq i \leq 7 \end{cases}$$

We now have the diagram of Fig. 8.

We proceed to describe an encoder and decoder based on the mappings φ, ϑ, ψ. Let $\{x_n\}$, $\{z_n\}$, $\{u_n\}$, and $\{y_n\}$ be the sequences, respectively, in X, $\{\hat{T}^2\}$, $\{T^2\}$, and $Y^{[2]}$.

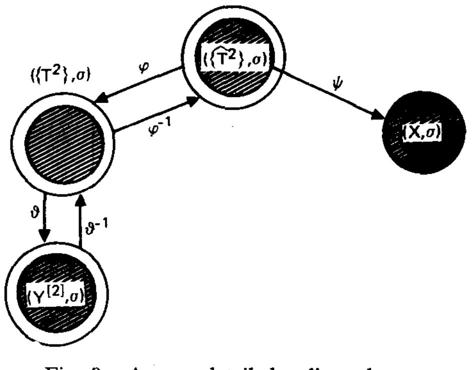


Fig. 8. A more detailed coding scheme.

TABLE VIII
PREDECESSOR TABLE

					1^1	$\to$	0^2	0^1
	5^1	4^1	3^1	2^1	2^3	$\to$	1^1	
		5^3	4^3	3^3	2^2	$\to$	1^2	1^3
7^1	6^1	5^2	4^2	3^2	0^2	$\to$	2^1	2^3
					0^1	$\to$	2^2	2^4
					1^2	$\to$	3^1	3^3
					1^3	$\to$	3^2	3^4
					2^3	$\to$	4^1	
					2^4	$\to$	4^2	4^3
					3^3	$\to$	5^1	
					3^4	$\to$	5^2	5^3
					4^3	$\to$	6^1	
					5^3	$\to$	7^1	

Thus $x_n \in \{0, 1\}$, z_n is of the form $i^j \in \mathcal{Q}_{\{\hat{T}^2\}}$, u_n is of the form $i \in \mathcal{Q}_{\{T^2\}}$ and $y_n \in \{00, 01, 10, 11\}$.

The encoder is the following finite state automaton as per (4.1)

$$\begin{aligned} y_n &= e(z_n) = \vartheta\varphi(z_n) \\ z_n &= f(x_n, z_{n-1}), \end{aligned} \tag{8.3}$$

where e does not depend directly on any input x_n but only on internal state z_n, e.g., $e(0^1) = 10$, $e(1^2) = 01$, $e(3^4) = 00$, etc., and the function f is read off Table VII according to the rule: z_n is the entry in column one on the right of the arrow from z_{n-1}, if $x_n = 0$; and the one in column two, if $x_n = 1$: e.g., $7^1 = f(0, 5^3)$. That f is well-defined follows from the right resolving property of ψ, which is verified by noting that each successor pair in Table VII consists of two distinct states.

The block 0 1 0 0 0 0 is resolving for ψ because upon following this input we can see how the initial ambiguity is resolved; namely,

$$\text{if } z_{n-1} \in \mathcal{Q}_{\{\hat{T}^2\}}$$

$$\begin{array}{ll} \text{and if} & x_n = 0, \\ \text{then} & z_n \in \{2^2, 2^1, 0^2, 3^1, 3^2, 1^2, 4^1, 4^2, 5^1, 5^2, 6^1, 7^1\}, \\ \text{and if} & x_{n+1} = 1, \\ \text{then} & z_{n+1} \in \{1^3, 0^1, 2^3, 3^3\}, \\ \text{and if} & x_{n+2} = 0, \\ \text{then} & z_{n+2} \in \{3^2, 2^2, 4^1, 5^1\}, \\ \text{and if} & x_{n+3} = 0, \\ \text{then} & z_{n+3} \in \{1^2, 0^2\}, \\ \text{and if} & x_{n+4} = 0, \\ \text{then} & z_{n+4} \in \{3^1, 2^1\}, \\ \text{and if} & x_{n+5} = 0, \\ \text{then} & z_{n+5} = 0^2. \end{array} \tag{8.4}$$

Thus in six steps the encoding automaton can be reset to 0^2 by the above sequence of inputs no matter what the initial state was.

The decoder is the map $d = \psi\varphi^{-1}\vartheta^{-1}$.

The map ϑ is a right resolving 1-block map. The blocks 01 and 10, which are 1-blocks in terms of the alphabet $\mathcal{Q}_{Y^{[2]}}$, are resolving, i.e., $\vartheta^{-1}(01)$ and $\vartheta^{-1}(10)$ have single preimages 0 and 1, respectively. One of these blocks occurs in every T^2-admissible 4-block. Thus ϑ^{-1} is a 4-block map, that is

$$u_n = \vartheta^{-1}(y_{n-3}, y_{n-2}, y_{n-1}, y_n),$$

e.g., if $(y_{n-3}, y_{n-2}, y_{n-1}, y_n) = (00, 01, 00, 00)$, then u_{n-3} is undetermined, $u_{n-2} = 0$, $u_{n-1} = 2$, $u_n = 4$.

The map φ is left resolving because in the predecessor Table VIII, which is another way of writing Table VII, no symbol when its superscript is removed appears twice in a row on the left of the arrow.

A resolving block occurs in every block of four symbols. This is because φ^{-1} is known to be a $(1 + 3)$-block map from the fact that there were three steps in the splitting process. So

$$z_n = \varphi^{-1}(u_n, u_{n+1}, u_{n+2}, u_{n+3}).$$

A sample computation of φ^{-1} is as follows. Suppose $(u_n, u_{n+1}, u_{n+2}, u_{n+3}) = (0, 2, 1, 3)$. Then

$$z_{n+3} \in \{3^1, 3^2, 3^3, 3^4\},$$

and we get in succession from Table VIII

$$\begin{aligned} z_{n+2} &\in \{1^2, 1^3\} \\ z_{n+1} &\in \{2^2\} \\ z_n &= 0^1. \end{aligned}$$

We can efficiently express this computation by

$$0^1, 2^2, 1^{2,3}, 3^{1,2,3,4}, \tag{8.5}$$

which contains the possibilities of z_n, z_{n+1}, z_{n+2}, z_{n+3} corresponding to a particular u_n, u_{n+1}, u_{n+2}, u_{n+3}. This is a useful way to carry out the computation necessary to tabulate the values of the four-block map φ^{-1}, which we do in Tables IX and X.

The map ψ is a 2-block map

$$x_n = \psi(z_{n-1}, z_n)$$

read off from Table VII.

Putting the three maps together we get

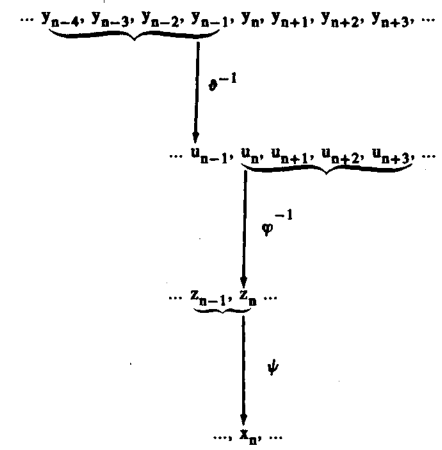
$$x_n = d(y_{n-4}, y_{n-3}, y_{n-2}, y_{n-1}, y_n, y_{n+1}, y_{n+2}, y_{n+3}), \tag{8.6}$$

TABLE IX
BLOCK LIST FOR DECODING MAP

x^n	z_n	z_{n+1}	z_{n+2}	z_{n+3}	y_n	y_{n+1}	y_{n+2}	y_{n+3}
	not in range of encoder				00	00	00	01
1	2^4	4^3	6^1	1^{1-3}	00	00	00	10
	3^4	5^3	7^1	1^{1-3}				
1	2^3	4^1	$0^{1,2}$	2^{1-4}	00	00	01	00
	3^3	5^1	$0^{1,2}$	2^{1-4}				
	2^4	4^2	$1^{2,3}$	3^{1-4}				
1	3^4	5^2	$1^{2,3}$	3^{1-4}	00	00	10	00
	4^3	6^1	$1^{2,3}$	3^{1-4}				
	5^3	7^1	$1^{2,3}$	3^{1-4}				
1	2^4	4^3	1^1	$0^{1,2}$	00	00	10	01
	3^4	5^3	1^1	$0^{1,2}$				
	2^1	$0^{1,2}$						
0	3^1	$0^{1,2}$	—	—	00	01	—	—
	4^1	$0^{1,2}$	—	—				
	5^1	$0^{1,2}$						
	2^2	$1^{2,3}$	3^{1-4}	—				
	3^2	$1^{2,3}$	3^{1-4}	—				
0	4^2	$1^{2,3}$	3^{1-4}	—	00	10	00	—
	5^2	$1^{2,3}$	3^{1-4}	—				
	6^1	$1^{2,3}$	1^{1-4}	—				
	7^1	$1^{2,3}$	1^{1-4}	—				
	2^3	1^1	$0^{1,2}$	—				
1	3^3	1^1	$0^{1,2}$	—	00	10	01	—
	4^3	1^1	$0^{1,2}$	—				
	5^3	1^1	$0^{1,2}$	—				
1	0^1	2^4	4^3	6^1	01	00	00	00
0	0^2	2^3	4^1	$0^{1,2}$	01	00	00	01
1	0^1	2^4	$4^{2,3}$	1^{1-3}	01	00	00	10
0	0^2	2^1	$0^{1,2}$	2^{1-4}	01	00	01	00
1	0^1	2^2	$1^{2,3}$	3^{1-4}	01	00	10	00
0	0^2	2^3	1^1	$0^{1,2}$	01	00	10	01
1	1^3	3^4	5^3	7^1	10	00	00	00
0	1^2	3^3	5^1	$0^{1,2}$	10	00	00	01
1	1^3	3^4	$5^{2,3}$	1^{1-3}	10	00	00	10
0	1^2	3^1	$0^{1,2}$	2^{1-4}	10	00	01	00
1	1^3	3^2	$1^{2,3}$	3^{1-4}	10	00	10	00
0	1^2	3^3	1^1	$0^{1,2}$	10	00	10	01
1	1^1	$0^{1,2}$	—	—	10	01	—	—

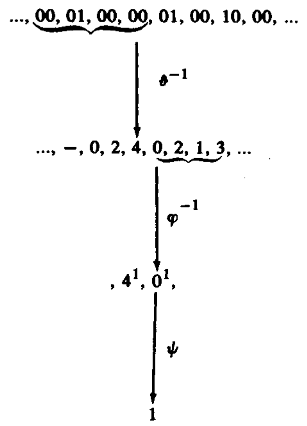
TABLE X
DECODER TRUTH TABLE

ξ	η_1	η_2	η_3	η_4	η_5	η_6	η_7	η_8
1	0	0	0	0	—	—	—	—
1	0	0	1	0	0	1	—	—
1	0	1	0	0	0	0	—	0
1	0	1	0	0	1	0	0	0
1	1	0	0	0	0	0	—	0
1	1	0	0	0	1	0	0	0
1	1	0	0	1	—	—	—	—
0	all others in range of e							
0/1	blocks not in range of e							



$\cdots y_{n-4}, y_{n-3}, y_{n-2}, y_{n-1}, y_n, y_{n+1}, y_{n+2}, y_{n+3}, \cdots$
ϕ^{-1}
$\cdots u_{n-1}, u_n, u_{n+1}, u_{n+2}, u_{n+3}, \cdots$
φ^{-1}
$\cdots z_{n-1}, z_n \cdots$
ψ
$\cdots, x_n, \cdots$

Fig. 9. Description of decoder as a composition of finite sliding block maps.



$\ldots, 00, 01, 00, 00, 01, 00, 10, 00, \ldots$
ϕ^{-1}
$\ldots, -, 0, 2, 4, 0, 2, 1, 3, \ldots$
φ^{-1}
$, 4^1, 0^1,$
ψ
1

Fig. 10. Sample of decoding computation.

which is diagrammed in Fig. 9. A sample computation is provided in Fig. 10.

For a standard implementation of the encoder and decoder we must give d, e, f in terms of Boolean functions of binary variables (bits). Since $|\mathcal{Q}_{\{\hat{T}^2\}}| = 24$, it takes 5 bits to label the states of z_n. We then have that d is a map of 16 bits to 1 bit, i.e., a Boolean function of 16 binary variables; e a map of 5 bits to 2 bits, i.e., two functions of possibly five variables each; and f a map of 6 bits to 5 bits. Furthermore, an error in the input of d ostensibly propagates 8 bits in the output. The above complexity and error propagation is what we might have to settle for in a general situation. However, by taking advantage of choices which we are allowed to make in the construction of Table VII and which are somehow inherent in the [2, 7] constraint we can reduce complexity and error propagation. Needless to say, this is highly desirable, if not essential, for practical implementation; and we deal with this subject in the next section.

Remark 8.1: We comment here on the observation made in Section II that a weaker notion than isomorphism is needed for practical applications. There is no subsystem of $(\{T^2\}, \sigma)$ isomorphic to the full 2-shift. This can be seen by the following argument. The number of fixed points is invariant under isomorphism. There are two fixed points in the full 2-shift—namely, the sequence of all ones and the sequence of all zeros. However, there are no fixed points in $\{(T^2\}, \sigma)$. Thus, if we want to code between arbitrary sequences and [2, 7] run-length constrained ones at the rate 1 : 2, then we must forego isomorphism in favor of right resolving homomorphism. Actually ismorphism is still possible at rates of $1m$: $2m$ for a large m.

IX. Implementation

Before describing hardware for the [2, 7] code we take up the question of reducing complexity and error propagation.

First, observe that we have been able to put each symbol of $\mathcal{Q}_{\{\hat{T}^2\}}$ into a unique column on the right side of Table VII. This means that ψ is really a 1-block map

$$x_n = \psi(z_n),$$

e.g., $\psi(1^1) = 0$, $\psi(4^1) = 1$. Therefore (8.6) can be expressed by

$$x_n = d(y_{n-3}, y_{n-2}, y_{n-1}, y_n, y_{n+1}, y_{n+2}, y_{n+3}) \quad (9.1)$$

which reduces d from a 16 bit function to a 14-bit one.

Second, we show that Table VII admits a way of filling in the columns such that x_n does not depend on $y_{n-3}, y_{n-2}, y_{n-1}$. Table IX is a tabulation of all possibilities of $x_n, z_n, z_{n+1}, z_{n+2}, z_{n+3}$ corresponding to each fixed $y_n, y_{n+1}, y_{n+2}, y_{n+3}$. It substantiates the above claim by virtue of the fact that all z_n's corresponding to each bit pattern of a 4-block $(y_n, y_{n+1}, y_{n+2}, y_{n+3})$ have been fit into the same column of Table VII and hence output the same x_n. We use the notation of (8.5) which shows how, from the predecessor Table VIII, the z_n are gotten, via the u_n, from the y_n. We conclude from Table IX, for example, that $2^2, 3^2, 4^2, 5^2, 6^1, 7^1$ should be put in the same column on Table VII.

From Table IX we tabulate in Table X the truth table for the decoder which is a Boolean function of 8 bits

$$\xi = d(\eta_1, \cdots, \eta_8),$$

where $\xi = x_n$, $y_n = (\eta_1, \eta_2)$, $y_{n+1} = (\eta_3, \eta_4)$, $y_{n+2} = (\eta_5, \eta_6)$, $y_{n+3} = (\eta_7, \eta_8)$.

The last line of Table X consists of what we call "*don't care*" conditions. It means that we are free to choose the output ξ for those 8-blocks $(\eta_1, \cdots, \eta_8)$ not in the range of the encoder, in particular those violating the [2, 7] constraint. The choice may be made with various ends in mind, for example, simplifying hardware, error correcting, etc. To convert a truth table such as Table X into a Boolean function we invoke the following.

Theorem 9.1 [39, *p.* 77]: Every Boolean function $f(x_1, \cdots, x_n)$ can be expressed in the disjunctive normal form

$$f(x_1, \cdots, x_n) = \bigvee_{(\epsilon_1, \cdots, \epsilon_n) \in \{0,1\}^n} f(\epsilon_1, \cdots, \epsilon_n) x_1^{\epsilon_1} \cdots x_n^{\epsilon_n}$$

where $x_i^0 = \bar{x}_i$ and $x_i^1 = x_i$.

If we set $\xi = 0$ in the last line of Table X, then combining Table X and Theorem 9.1 and simplifying we get

$$\begin{aligned} \xi = {} & \bar{\eta}_1\bar{\eta}_2\bar{\eta}_3\bar{\eta}_4 \vee \eta_1\bar{\eta}_2\bar{\eta}_3\eta_4 \\ & \vee \bar{\eta}_1\bar{\eta}_2\eta_3\bar{\eta}_4\bar{\eta}_5\bar{\eta}_6 \vee \bar{\eta}_1\eta_2\bar{\eta}_3\bar{\eta}_4\bar{\eta}_5\bar{\eta}_6\bar{\eta}_8 \\ & \vee \bar{\eta}_1\eta_2\bar{\eta}_3\bar{\eta}_4\eta_5\eta_6\eta_7\eta_8 \vee \eta_1\bar{\eta}_2\bar{\eta}_3\bar{\eta}_4\bar{\eta}_5\bar{\eta}_6\bar{\eta}_8 \\ & \vee \eta_1\bar{\eta}_2\bar{\eta}_3\bar{\eta}_4\eta_5\bar{\eta}_6\bar{\eta}_7\bar{\eta}_8. \end{aligned} \quad (9.2)$$

TABLE XI
Reduced Decoder Truth Table

ξ	η_1	η_2	η_3	η_4	η_5	η_6	η_7	η_8
1	0	0	0	0	—	—	—	—
1	—	—	1	—	—	1	—	—
1	—	1	—	—	—	0	—	0
1	1	—	—	—	—	0	—	0
1	1	—	—	1	—	—	—	—
0				all others				

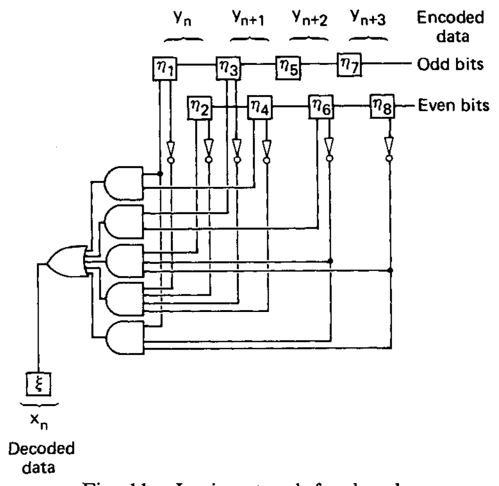


Fig. 11. Logic network for decoder.

However, by setting $\xi = 1$ for a judicious choice of $(\eta_1, \cdots, \eta_8)$ violating the [2, 7] constraint we have another equally valid truth table for the function d operating on the range of e: namely Table XI. Table XI minimizes hardware for d, and the Boolean function for it is

$$\xi = \eta_1\eta_4 \vee \eta_3\eta_6 \vee \eta_2\bar{\eta}_6\bar{\eta}_8 \vee \bar{\eta}_1\bar{\eta}_2\bar{\eta}_3\bar{\eta}_4 \vee \eta_1\bar{\eta}_6\bar{\eta}_8. \quad (9.3)$$

Equation (9.3) yields the logic network in Fig. 11 for the decoder.

We remark that due to the fact that φ is left resolving there is another way of constructing decoders by means of an automaton with a stack. We shall not go into this topic here, but it is a useful alternative when d is a function of very many variables.

Next we turn our attention to the encoder. We reproduce Table VII, to which we add the outputs that result upon the tabulated state transition. Table XII contains all necessary information for construction of the encoding automaton. Some of the states of this table can be amalgamated, which results in a smaller table, hence reduced encoder complexity. The rule for amalgamation is the following: state which have identical successor/output pairs are combined. For example, $5^1, 4^1, 3^1, 2^1, 1^1$ are amalgamated into one state which we can label 1^1. This results in a smaller table to which the rule is again applied. We present this series of reductions in Tables XIII through XVI.

After amalgamation, the encoder is the finite state automaton

$$\begin{aligned} y_n &= e(x_n, z_{n-1}) \\ z_n &= f(x_n, z_{n-1}) \end{aligned} \quad (9.4)$$

TABLE XII
ENCODING STATE TRANSITION AND OUTPUT TABLE

					0^1	$\rightarrow$	$2^2/00$	$2^4/00$
					0^2	$\rightarrow$	$2^1/00$	$2^3/00$
	5^1	4^1	3^1	2^1	1^1	$\rightarrow$	$0^2/01$	$0^1/01$
					1^2	$\rightarrow$	$3^1/00$	$3^3/00$
					1^3	$\rightarrow$	$3^2/00$	$3^4/00$
7^1	6^1	5^2	4^2	3^2	2^2	$\rightarrow$	$1^2/10$	$1^3/10$
					2^3	$\rightarrow$	$4^1/00$	$1^1/10$
					2^4	$\rightarrow$	$4^2/00$	$4^3/00$
					3^3	$\rightarrow$	$5^1/00$	$1^1/10$
					3^4	$\rightarrow$	$5^2/00$	$5^3/00$
					4^3	$\rightarrow$	$6^1/00$	$1^1/10$
					5^3	$\rightarrow$	$7^1/00$	$1^1/10$

TABLE XIII
FIRST REDUCTION

	0^1	$\rightarrow$	$2^2/00$	$2^4/00$
	0^2	$\rightarrow$	$1^1/00$	$2^3/00$
	1^1	$\rightarrow$	$0^2/01$	$0^1/01$
	1^2	$\rightarrow$	$1^1/00$	$3^3/00$
	1^3	$\rightarrow$	$2^2/00$	$3^4/00$
	2^2	$\rightarrow$	$1^2/10$	$1^3/10$
3^3	2^3	$\rightarrow$	$1^1/00$	$1^1/10$
	2^4	$\rightarrow$	$2^2/00$	$4^3/00$
	3^4	$\rightarrow$	$2^2/00$	$5^3/00$
5^3	4^3	$\rightarrow$	$2^2/00$	$1^1/10$

TABLE XIV
SECOND REDUCTION

	0^1	$\rightarrow$	$2^2/00$	$2^4/00$
1^2	0^2	$\rightarrow$	$1^1/00$	$2^3/00$
	1^1	$\rightarrow$	$0^2/01$	$0^1/01$
	1^3	$\rightarrow$	$2^2/00$	$3^4/00$
	2^2	$\rightarrow$	$1^2/10$	$1^3/10$
	2^3	$\rightarrow$	$1^1/00$	$1^1/10$
3^4	2^4	$\rightarrow$	$2^2/00$	$4^3/00$
	4^3	$\rightarrow$	$2^2/00$	$1^1/10$

TABLE XV
THIRD REDUCTION

1^3	0^1	$\rightarrow$	$2^2/00$	$2^4/00$
	0^2	$\rightarrow$	$1^1/00$	$2^3/00$
	1^1	$\rightarrow$	$0^2/01$	$0^1/01$
	2^2	$\rightarrow$	$0^2/10$	$1^3/10$
	2^3	$\rightarrow$	$1^1/00$	$1^1/10$
	2^4	$\rightarrow$	$2^2/00$	$4^3/00$
	4^3	$\rightarrow$	$2^2/00$	$1^1/10$

TABLE XVI
FINAL ENCODING TABLE

		$x_n = 0$	$x_n = 1$
z_{n-1}		z_n/y_n	z_n/y_n
0^1	$\rightarrow$	$2^2/00$	$2^4/00$
0^2	$\rightarrow$	$1^1/00$	$2^3/00$
1^1	$\rightarrow$	$0^2/01$	$0^1/01$
2^2	$\rightarrow$	$0^2/10$	$0^1/10$
2^3	$\rightarrow$	$1^1/00$	$1^1/10$
2^4	$\rightarrow$	$2^2/00$	$4^3/00$
4^3	$\rightarrow$	$2^2/00$	$1^1/10$

TABLE XVII
ENCODER TRUTH TABLE

				$\xi_1 = 0$	$\xi_1 = 1$
		$\zeta_2\ \zeta_3\ \zeta_4$		$\zeta_2'\ \zeta_3'\ \zeta_4'\ /\ \eta_1\ \eta_2$	$\zeta_2'\ \zeta_3'\ \zeta_4'\ /\ \eta_1\ \eta_2$
0^1	=	0 0 0	$\rightarrow$	0 1 1 / 0 0	1 0 1 / 0 0
0^2	=	0 0 1	$\rightarrow$	0 1 0 / 0 0	1 0 0 / 0 0
1^1	=	0 1 0	$\rightarrow$	0 0 1 / 0 1	0 0 0 / 0 1
2^2	=	0 1 1	$\rightarrow$	0 0 1 / 1 0	0 0 0 / 1 0
2^3	=	1 0 0	$\rightarrow$	0 1 0 / 0 0	0 1 0 / 1 0
2^4	=	1 0 1	$\rightarrow$	0 1 1 / 0 0	1 1 0 / 0 0
4^3	=	1 1 0	$\rightarrow$	0 1 1 / 0 0	0 1 0 / 1 0

where the next state function f is read off of Table XVI just like before. The state space for z_n is $\mathcal{C} = \{0^1, 0^2, 1^1, 2^2, 2^3, 2^4, 4^3\}$, having cardinality 7. When given by Boolean functions, e and f of (9.4) map 4 bits, respectively, to 2 and 3 bits, a substantial reduction in complexity compared to e and f of (8.3). Note the difference between the e of (8.3) and that of (9.4); namely, the former depends explicitly only on z_n whereas the later on both x_n and z_{n-1}. In this connection we make some observations apropos dynamical systems. In Section VIII (Z, σ) was an extension of (Y, σ^2), a fact which requires $h(Z, \sigma) = h(Y, \sigma^2)$. By means of amalgamation we have introduced in Section IX a new dynamical system (Z, σ) with smaller topological entropy, which means (Y, σ^2) cannot be a homomorphic image of it. However, (Y, σ^2) is the homomorphic image of a skew product of X and Z in which the entropy comes from its (X, σ) factor. This skew product system is the topological Markov shift defined by the transitions $(x_{n-1}, z_{n-1}) \rightarrow (x_n, f(x_n, z_{n-1}))$ gotten from Table XVI.

As in the decoder we express our variables in binary form. Thus, let x_n be given by ξ_1; and y_n by η_1, η_2. Let z_{n-1} be given by $\zeta_2, \zeta_3, \zeta_4$; and z_n by $\zeta_2', \zeta_3', \zeta_4'$. In terms of binary labels Table XVI becomes Table XVII.

Also, as in the decoder, we are going to take advantage of a "don't care" condition—namely, we are free to choose convenient values for ζ_2', ζ_3', ζ_4', η_1, and η_2, whenever $\zeta_2, \zeta_3, \zeta_4 = 1, 1, 1$. To minimize encoder logic we add the following transition/output entry to Table XVII:

$$1\ 1\ 1 \quad \rightarrow \quad 0\ 1\ 1\ /\ 1\ 0 \qquad 1\ 1\ 0\ /\ 1\ 0.$$

Applying Theorem 9.1 to the information contained in Table XVII along with the "don't care" condition and simplifying, we get

$$\begin{aligned} \zeta_2' &= \xi_1\bar{\zeta}_3 \vee \xi_1\zeta_2\zeta_4, \\ \zeta_3' &= \bar{\xi}_1\bar{\zeta}_3 \vee \zeta_2, \\ \zeta_4' &= \bar{\zeta}_2\bar{\zeta}_3\bar{\zeta}_4 \vee \bar{\xi}_1\zeta_3 \vee \bar{\xi}_1\zeta_2\zeta_4, \\ \eta_1 &= \zeta_3\zeta_4 \vee \xi_1\zeta_2\bar{\zeta}_4, \\ \eta_2 &= \bar{\zeta}_2\zeta_3\bar{\zeta}_4. \end{aligned} \tag{9.5}$$

From (9.5) we diagram the logic network for the encoder in Fig. 12.

Finally, we remark that the internal state z_n of the encoder can be reset to 001 by the input 0100. This follows from (8.4) and the fact that 1^2, 0^2 have been amalgamated for z_{n+3} and $0^2 = 001$. We note that the spurious state 111, which is transient, behaves nicely with respect to the initial-

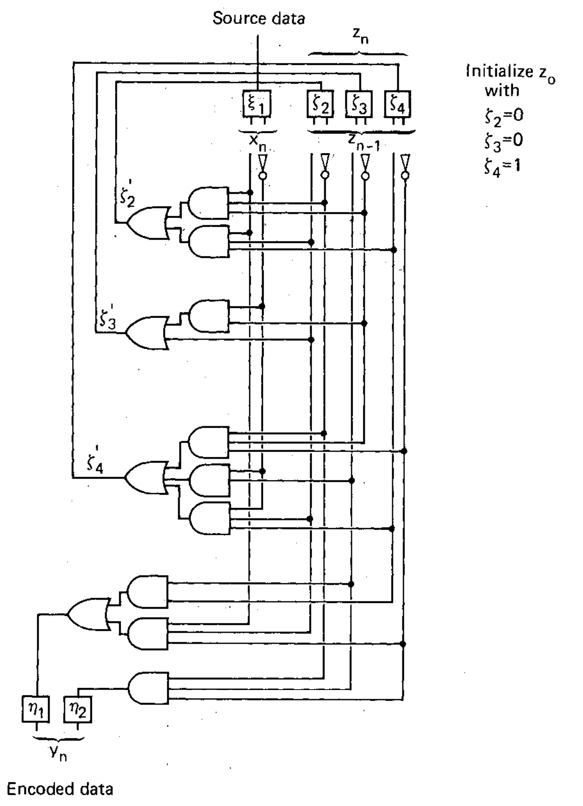


Fig. 12. Logic network for encoder.

izing sequence. By being able to reset the internal state of the encoder at will we can generate specific constraint outputs of our choice by means of inputs alone.

Appendix

The purpose of this section is to provide the reader with a method for finding an approximate characteristic vector which can easily be programmed for computer (particularly in APL), and, in fact, lends itself to hand computation for examples of moderate size. This method, for solving the following integer programming problem, was used in [19].

Suppose λ is the spectral radius (largest positive characteristic vector) of an $M \times M$ transition matrix T. If N is an integer $\leq \lambda$ then, by the Perron–Frobenius theory, there exists a vector v of integers satisfying

$$Tv \geq Nv, \qquad v > 0. \tag{A1}$$

We wish to find such a vector. In order to do so, choose an initial vector $v^{(0)}$, for instance whose components $v_i^{(0)} = L$ where L is an integer > 0. Define inductively

$$v_i^{(n+1)} \equiv \min\left(v_i^{(n)}, \left[\sum_{j=1}^{M} t_{ij} v_j^{(n)}/N \right] \right) \tag{A2}$$

where $[a]$ means the largest integer in a nonnegative number a. Let

$$v \equiv v^{(n)} \tag{A3}$$

where n is the first integer such that $v^{(n+1)} = v^{(n)}$. Such an n exists because $v_i^{(n)}$ are all nonnegative integers and $v_i^{(n+1)} \leq v_i^{(n)}$. In fact $n \leq L \cdot M$. From (A2), (A3) we have

$$v_i \leq \left[\sum_{j=1}^{M} t_{ij} v_j/N \right] \leq \sum_{j=1}^{M} t_{ij} v_j/N,$$

i.e.,

$$Tv \geq Nv.$$

Three cases arise:

1) $v > 0$;
2) $v \geq 0$, $v \neq 0$, and some $v_i = 0$;
3) $v = 0$.

Case 1 yields a solution to (A1). Case 3 means that there is no approximate characteristic vector v with $v_j \leq L$, which we shall prove below. So we must try again with larger L. Case 2 is interesting because v is then an approximate characteristic subvector. We can apply the same method as the one which gets $\bar{T}$ from T' in the steps of the splitting process of Theorem 6.1. We thereby obtain a new matrix $\bar{T}$ and an integral vector $\bar{v}$ such that

$$\bar{T}\bar{v} \geq N\bar{v},$$
$$\bar{v} > 0,$$

where $\bar{T}$ is an irreducible component of the matrix gotten from T by crossing out the ith row and column whenever $v_i = 0$, and $\bar{v}$ is the restriction of v to indices of this irreducible component. This new matrix is suitable, perhaps even preferrable to the original, as an initial one for the splitting process described in Theorem 6.1.

Remark A1: If u is a vector of integers satisfying $Tu \geq Nu$ and $u \leq v^{(0)}$ then by induction it is easy to see that $u \leq v^{(n)}$. Hence $u \leq v$. So if there are approximate characteristic vectors (or subvectors) with components $\leq L$, then (A2) will converge in a finite number of steps to the largest one; and we know by the Perron–Frobenius theory that if we choose L large enough there will be approximate characteristic vectors within its range.

There are various parameters of encoders and decoders we wish to optimize. A significant factor in the complexity of such devices based on mappings constructed in Theorem 6.1 is the size of $\hat{T}$. Therefore we wish to solve (A1) with Σv_i small. The complexity of the decoder as well as its error propagation properties is related to the block size of the mapping φ in Theorem 6.1, which is governed by the number of steps in the splitting process, which in turn is connected vaguely to the size of $\max v_i$. This means we also want that quantity small. So, after finding a solution to (A1), it may be desirable to find a "better" one. A reasonable procedure is to take a solution, reduce one of its components and apply (A2) again. If there is a better v to be found, then it will be found by doing this in a systematic fashion.

References

[1] R. L. Adler, L. W. Goodwyn, and B. Weiss, "Equivalence of topological Markov shifts," *Israel J. Math*, vol. 27, pp. 49–63, 1977.
[2] R. L. Adler and M. Hassner, "Algorithms for sliding block codes," in *Intern. Symp. Inform. Theory, Abstracts of Papers*, Santa Monica, CA, p. 50, Feb. 9–12, 1981.
[3] R. L. Adler, A. G. Konheim, and M. H. McAndrew, "Topological entropy," *Trans. Amer. Math. Soc.*, vol. 114, pp. 309–319, 1965.
[4] R. L. Adler and B. Marcus, "Topological entropy and equivalence of dynamical systems," *Mem. Amer. Math. Soc.*, vol. 219, 1979.
[5] R. L. Adler and B. Weiss, "Similarity of automorphisms of the torus," *Mem. Amer. Math. Soc.*, vol. 98, 1970.
[6] T. Berger and J. K. Y. Lau, "On binary sliding block codes," *IEEE Trans. Inform. Theory*, vol. IT-23, pp. 343–353, 1977.
[7] R. Bowen, "Entropy for group endomorphisms and homogeneous spaces," *Trans. Amer. Math. Soc.*, vol. 153, pp. 401–414, 1971.
[8] E. M. Coven and M. E. Paul, "Endomorphisms of irreducible subshifts of finite type," *Math. Syst. Theory*, vol. 8, pp. 167–175, 1974.
[9] E. M. Coven and M. E. Paul, "Sofic systems," *Israel J. Math.*, vol. 20, pp. 165–177, 1975.
[10] E. M. Coven and M. E. Paul, "Finite procedures for sofic systems,"

Monatsh. Math, vol. 83, pp. 265–278, 1977.

[11] M. Denker, C. Grillenberger, and K. Sigmund, *Ergodic Theory on Compact Spaces*, *Lecture Notes in Math. 527*. New York: Springer-Verlag, 1976.

[12] E. I. Dinaburg, "On the relations among various entropy characteristics of dynamical systems," *Math USSR Izvestija*, vol. 5, pp. 337–378, 1971.

[13] J. S. Eggenberger and P. Hodges, "Sequential encoding and decoding of variable word length, fixed rate data codes," U.S. Patent 4,115,768, 1978.

[14] P. A. Franaszek, "Sequence-state coding for digital transmission," *Bell Sys. Tech. J.*, pp. 113–157, 1968.

[15] ——, "On synchronous variable length coding for discrete noiseless channels," *Inform. Contr.*, vol. 1-J, pp. 155–164, 1969.

[16] ——, "Sequence-state methods for run-length-limited coding," *IBM J. Res. Dev.*, vol. 14, pp. 376–383, 1970.

[17] ——, "On future-dependent block coding for input restricted channels," *IBM J. Res. Dev.*, vol. 23, pp. 75–81, 1979.

[18] ——, "Synchronous bounded delay coding for input restricted channels," *IBM J. Res. Dev.*, vol. 24, pp. 43–48, 1980.

[19] ——, "A general method for channel coding," *IBM J. Res. Dev.*, vol. 24, pp. 638–641, 1980.

[20] ——, "Construction of bounded delay codes for discrete noiseless channels," *IBM Res. Dev.*, vol. 26, pp. 506–514, 1982.

[21] F. R. Gantmacher, *The Theory of Matrices*, *Vol. II*. Chelsea, NY, 1959.

[22] E. Gorog, "Redundant alphabets with desirable frequency spectrum properties," *IBM J. Res. Dev.*, vol. 12, pp. 234–240, 1968.

[23] T. N. T. Goodman, "Relating topological entropy and measure entropy," *Bull. London Math. Soc.*, vol. 3, pp. 176–180, 1971.

[24] R. L. Gray, "Generalizing period and topological entropy to transitive nonwandering systems," Masters thesis, Univ. of NC, Chapel Hill, 1978.

[25] R. M. Gray, "Sliding-block source coding," *IEEE Trans. Inform. Theory*, vol. 21, pp. 357–368, 1975.

[26] M. Hassner, "A nonprobabilistic source and channel coding theory," Ph.D. dissertation, UCLA, 1980.

[27] G. A. Hedlund, *Transformations Commuting with the Shift*, *Topological Dynamics (an International Symposium)*, Joseph Auslander and Walter Gottshalk, Eds. New York: W. A. Benjamin, 1968.

[28] ——, "Endomorphisms and automorphisms of the shift dynamical system," *Math Syst. Theory*, vol. 3, pp. 320–375, 1969.

[29] T. Horiguchi and K. Morita, "An optimization of modulation codes in digital recording," *IEEE Trans. Magn.*, vol. MAG-12, no. 6, pp. 740–742, 1976.

[30] S. S. Hong and D. L. Ostapko, "Codes for self-clocking, AC-coupled transmission: Aspects of synthesis and analysis," *IBM J. Res. Dev.*, vol. 19, pp. 358–365, 1975.

[31] B. Kitchens, "Continuity properties of factor maps in ergodic theory," Ph.D. dissertation, Univ. of NC, Chapel Hill, 1981.

[32] H. Kobayashi, "Coding schemes for reduction of intersymbol interference in data transmission systems," *IBM J. Res. Dev.*, vol. 14, pp. 343–353, 1970.

[33] ——, "A survey of coding schemes for transmission or recording of digital data," *IEEE Trans. Comm. Tech.*, vol. COM-19, pp. 1087–1100, 1971.

[34] D. A. Lindholm, "Power spectra of channel codes for digital magnetic recording," *IEEE Trans. Magn.*, vol. MAG-14, pp. 321–323, 1978.

[35] A. Lempel and M. Cohn, "Look ahead coding for input restricted channels," *IEEE Trans. Inform. Theory*, vol. IT-28, pp. 933–937, Nov. 1982.

[36] B. McMillan, "The basic theorems of information theory," *Ann. Math. Stat.*, vol. 24, pp. 196–219, 1953.

[37] B. Marcus, "Factors and extensions of full shifts," *Monatshefte für Math*, vol. 88, pp. 239–247, 1979.

[38] ——, "Sofic systems and encoding data on magnetic tape," preliminary report notices," *Amer. Math. Soc.*, vol. 29, p. 43, 1982.

[39] R. E. Miller, *Switching Theory*, *Vol. I*. New York: John Wiley, 1965.

[40] M. Nasu, "Uniformly finite-to-one and onto extensions of homomorphisms between strongly connected graphs," Preprint, Research Institute of Electrical Communication, Tohohu Univ., Sendai, Japan.

[41] K. Norris and D. S. Bloomberg, Channel capacity of charge constrained run-length limited codes," *IEEE Trans. Magn.*, vol. MAG-17, pp. 3452–3455, 1981.

[42] "Small Winchester drives move up to main frame encoding schemes," *Electron. Des.*, pp. 51–52, Oct. 15, 1981.

[43] G. L. O'Brien, "The road-colouring problem," *Israel J. Math.*, vol. 39, pp. 145–154, 1981.

[44] W. Parry, "Intrinsic Markov chains," *Trans. Amer. Math. Soc.*, vol. 112, pp. 55–66, 1964.

[45] ——, "A finitary classification of topological Markov chains and sofic systems," *Bull. London Math. Soc.*, vol. 9, pp. 86–92, 1977.

[46] A. M. Patel, "Zero modulation encoding in magnetic recording," *IBM J. Res. Dev.*, vol. 19, no. 4, pp. 366–378, 1975.

[47] E. Seneta, *Non Negative Matrices*. New York: John Wiley, 1973.

[48] C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication*. Urbana: Univ. IL, 1963.

[49] D. T. Tang and L. R. Bahl, "Block codes for a class of constrained noiseless channels," *Inform. Contr.*, vol. 17, pp. 436–461, 1970.

[50] B. Weiss, "Subshifts of finite type and sofic systems," *Monatsh. Math.*, vol. 77, pp. 462–474, 1973.

[51] R. F. Williams, "Classification of shifts of finite type," *Ann. Math.*, vol. 98, pp. 120–153, 1973; *Errata*, *Ann. Math.*, vol. 99, pp. 380–381, 1974.