SIAM J. DISCRETE MATH.
Vol. 19, No. 2, pp. 321–344

STABILIZATION OF BLOCK-TYPE-DECODABILITY PROPERTIES FOR CONSTRAINED SYSTEMS*

PANU CHAICHANAVONG† AND BRIAN H. MARCUS‡

Abstract. We consider a class of encoders for constrained systems, which we call block-type-decodable encoders. For a constrained system presented by a deterministic graph, we design a block-type-decodable encoder by selecting a subset of states of the graph to be used as encoder states. Such a subset is known as a set of principal states. Our goal is to find an optimal set of principal states, i.e., a set which yields the highest code rate. We study the relationship between optimal sets of principal states at finite block length and at asymptotically large block length. Specifically, we show that for a primitive constraint and a large enough block length, any optimal set of principal states is also asymptotically optimal. Moreover, we give bounds on the block length such that this relationship holds. We also characterize asymptotically optimal block-type-decodable encoders. Finally, we study the complexity of various problems related to block-type-decodable encoders.

Key words. constrained systems, block encoders, block-decodable encoders, deterministic encoders, sets of principal states, integer programming, NP-complete problems

AMS subject classifications. 94A99, 94B99, 68R10, 90C90, 68Q17

DOI. 10.1137/S0895480104443679

1. Introduction.

In most recording channels, arbitrary data is encoded into constrained sequences to improve the performance of storage systems. A constraint is presented by a labeled finite directed graph, and a constrained sequence is obtained by reading the labels of a path in the graph. The best known constraint is the runlength-limited (RLL(d,k)) constraint, which is the binary constraint that bounds the lengths of the runs of zeros to be at least d and at most k (see Figure 1.1). This constraint is used in magnetic tape drives and optical drives to suppress the interference between adjacent bits and improve the timing recovery system. The constraint that we will use as an example throughout this paper is the asymmetric-RLL(d_0,k_0,d_1,k_1) (see, e.g., Immink [7, section 4.5]), which requires that the lengths of the runs of zeros are between d_0 and k_0 and the lengths of the runs of ones are between d_1 and k_1.

For a given constraint and a given block length q, we consider fixed-rate encoders that encode arbitrary user data into constrained blocks of length q such that strings formed from concatenating consecutive encoded blocks satisfy the constraint. The precise definitions of the encoders that we consider in this paper are given in sections 2 and 3.

In order to avoid error propagation in the decoding process, many practical applications use block encoders. Although these encoders are conceptually simplest, we may be able to achieve higher rates using block-decodable encoders for which error propagation is still limited to one block. However, the optimal rate is difficult to compute, and an achieving block-decodable encoder is hard to design. Nevertheless, for some constraints—including the RLL(d,k) constraint—this problem has been shown

*Received by the editors May 12, 2004; accepted for publication (in revised form) November 29, 2004; published electronically October 20, 2005. Part of this work was presented at the 2003 IEEE International Symposium on Information Theory, Yokohama, Japan.
http://www.siam.org/journals/sidma/19-2/44367.html

†Center for Magnetic Recording Research, University of California-San Diego, 9500 Gilman Drive, La Jolla, CA 92093 (panu@ucsd.edu).

‡Department of Mathematics, University of British Columbia, 1984 Mathematics Road, Vancouver V6T 1Z2, BC, Canada (marcus@math.ubc.ca).

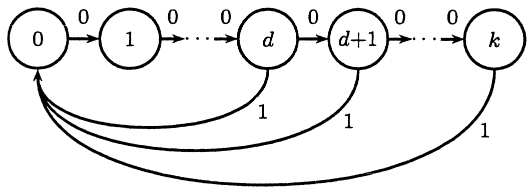

FIG. 1.1. *Presentation of RLL(d, k) constraint.*

to be equivalent to the problem of designing a deterministic encoder [4], which is much more tractable.

In this work, we are interested in these three classes of encoders, which we call block-type-decodable encoders: block, block-decodable, and deterministic encoders. It is known that the characterization of block-type-decodable encoders can be specified by subsets of states called the sets of principal states [10].

An optimal set of principal states for a deterministic encoder can be found using the Franaszek algorithm [4]. Algorithms for computing the optimal sets of principal states for a block encoder were presented by Freiman and Wyner [5] and Marcus, Siegel, and Wolf [11]; in this paper, we present a new framework for this problem. We also give candidates for optimal sets of principal states for a block-decodable encoder, together with upper and lower bounds on the optimal code rate; this is based on an integer programming interpretation.

Typically, high code rates require large block lengths. Thus, it is of interest to study the relationship between the optimal sets of principal states at a finite block length and those at asymptotically large block length. In [3], for deterministic encoders, we showed how to compute an asymptotically optimal set of principal states and observed that this is sometimes easier than the same problem at a finite block length. Empirically, this asymptotically optimal set of principal states is a good approximation to the finite case. In the present paper, we show how to compute an asymptotically optimal set of principal states for block and block-decodable encoders. We will establish the relationship between the finite case and the asymptotic case by showing that for a primitive constraint, there is a q_0 such that for any $q \geq q_0$, any optimal set of principal states at block length q is also asymptotically optimal. An upper bound on q_0 is given for each class of encoder; empirically, this bound appears to be small.

Finally, we consider the complexity of designing optimal block-type-decodable encoders. For deterministic encoders, this is known to be polynomial because the Franaszek algorithm is polynomial. Ashley, Karabed, and Siegel [1] showed that the problem of designing block-decodable encoders is NP-complete. In section 8, we show that the complexity of designing a block encoder is also NP-complete. We further show that if the number of states is fixed, all of these problems can be solved in polynomial time.

2. Background. Here we summarize basic definitions in constrained coding used in this paper. More detail can be found in [10, 7].

A *labeled directed graph* (or simply a *graph*) $G = (V, E, L)$ consists of a finite set of states $V = V_G$, a finite set of edges $E = E_G$ where each edge has an initial state and terminal state in V_G, and an edge labeling $L = L_G : E \to \Sigma$ where Σ is a finite alphabet. We will sometimes refer to a label or a sequence of labels of G as a *word*.

A *constrained system* or *constraint* $S = S(G)$ is the set of finite sequences obtained by reading the edge labels of a path in a labeled graph G. Such a graph is called a *presentation* of the constraint.

Two important properties of a graph are irreducibility and primitivity. A graph is *irreducible* if for any given pair u, v of states, there is a path from u to v and a path from v to u. A graph that is not irreducible is called *reducible*. Such a graph consists of nonoverlapping irreducible subgraphs, called *irreducible components*, and transitional edges between them. A graph is *primitive* if there exists a positive integer ℓ such that for all pairs u, v of states, there are paths from u to v and v to u of length ℓ. A constrained system is said to be irreducible if it has an irreducible presentation. Similarly, a constrained system is primitive if it has a primitive presentation. From the definitions, we can see that primitivity is stronger in the sense that every primitive graph (constrained system) is irreducible. Many practical constraints including RLL(d, k) are primitive.

Irreducibility and primitivity are properties of the topology of a graph alone but not its labeling. We now state the definitions of important properties of graph labeling that are used throughout the paper.

- A labeled graph is *deterministic* if at each state, all outgoing edges carry distinct labels. It is well known that every constraint has a deterministic presentation. Furthermore, for an irreducible constraint, there is a unique minimal (in terms of the number of states) deterministic presentation, called the *Shannon cover*. This presentation is often used as a starting point to construct a constrained encoder.
- A labeled graph has *finite memory* if there is an integer N such that all paths of length N with the same labeling terminate at the same state. The smallest N for which this holds is called the *memory* of the graph.
- A labeled graph is *lossless* if any two distinct paths with the same initial state and terminal state have different labels. This is the weakest property among all mentioned properties of labeling.

Let G be a labeled graph. The *adjacency matrix* $A = A_G$ is the $|V_G| \times |V_G|$ matrix whose entry $A_{u,v}$ is the number of edges from state u to state v in G. We say that a matrix is irreducible if it is the adjacency matrix of an irreducible graph. Similarly, a matrix is primitive if it is the adjacency matrix of a primitive graph.

Let G be a labeled graph. The *qth power of G*, denoted G^q, is the labeled graph with the same set of states as G, but with one edge for each path of length q in G, labeled by the word of length q generated by that path. For a constraint S presented by a labeled graph G, the *qth power of S*, denoted S^q, is the constraint presented by G^q. If A is the adjacency matrix of G, it can be shown that the adjacency matrix of G^q is A^q.

The *capacity* of a constraint S, denoted $\text{cap}(S)$, is defined to be

$$\text{cap}(S) = \lim_{q \to \infty} \frac{1}{q} \log N(q; S),$$

where $N(q; S)$ is the number of words of length q in S. (In this paper, the logarithmic function has base 2.) The capacity measures the growth rate of the number of words in S. It is known that $\text{cap}(S^q) = q\text{cap}(S)$.

To express the capacity in terms of the adjacency matrix of a lossless (in particular, deterministic) presentation G of S, we need the following notation. For a square matrix A, we denote by $\lambda(A)$ the *spectral radius* of A, that is, the largest of the absolute values of the eigenvalues of A. According to the Perron–Frobenius theory [12], $\lambda(A)$ is an eigenvalue of A. It is well known that

$$\mathrm{cap}(S) = \log \lambda(A_G).$$

Let S be a constrained system and let n be a positive integer. An (S, n) *encoder* is a labeled graph $\mathcal{E}$ such that

- each state of $\mathcal{E}$ has *out-degree* n, i.e., n outgoing edges,
- $S(\mathcal{E}) \subseteq S$,
- $\mathcal{E}$ is lossless.

The labels of the encoder are sometimes called *output labels.* A *tagged* (S, n) *encoder* is an (S, n) encoder whose outgoing edges from each state are assigned distinct *input tags* from an alphabet of size n, and this defines an encoding function. For an (S^q, n) encoder, we define the *block length* to be q and the *rate* to be $(\log n)/q$. It is known that $\mathrm{cap}(S)$ is an upper bound on the rate of any (S^q, n) encoder.

3. Block-type-decodable encoders. In this paper, we restrict our interest to block, block-decodable, and deterministic encoders. A *block encoder* (blk) is a finite-state encoder such that any two edges have the same input tag if and only if they have the same output label. A *block-decodable encoder* (blkdec) is a finite-state encoder such that any two edges with the same output label have the same input tag. A *deterministic encoder* (det) is a finite-state encoder with deterministic output labeling.

It is easy to see that a block encoder is block decodable, which in turn is deterministic. A block-decodable encoder can be viewed as a deterministic encoder with a consistent input tag assignment. In this paper, we focus on these three classes of encoders which we call *block-type-decodable encoders.*

For a constrained system S, a class of encoders $\mathcal{C} \in \{\mathrm{blk}, \mathrm{blkdec}, \mathrm{det}\}$, and a block length q, define $M_{\mathcal{C}}(q)$ to be the maximum n such that there exists an (S^q, n) encoder in class $\mathcal{C}$. Suppose that S is irreducible and let G be an irreducible deterministic presentation of S. For each class $\mathcal{C}$ of block-type-decodable encoders, it can be shown that there exists an (S, n) encoder in class $\mathcal{C}$ if and only if there exists such an encoder which is a subgraph of G. (For block encoder, see [5]. For block-decodable encoder, this is a special case of [2, Corollary 12.2]. For deterministic encoder, see [4]. For a unified treatment, see [10].) Thus the problem of designing block-type-decodable encoders can be solved by choosing a subgraph of G. This can be broken into two steps: First, choose a set of states, called a *set of principal states.* (A principal set of states may be a more appropriate term, but we will follow Franaszek [4] who defined it for deterministic encoders.) Then choose edges.

The reason for breaking this into two steps is that we often need to design an (S^q, n) encoder for various block lengths q. Since the graphs G^q have the same set of states for all q, we may need to solve the first step only once. In fact, the problem of determining whether a set of principal states is optimal for all large enough q is one of the main themes of the paper. That is, we study whether the optimal sets of principal states stabilize and, if so, at what value of q. (A set of principal states is optimal if it induces an encoder with the highest rate. For a more precise definition, see below.)

Let $M_{\mathcal{C}}(q, P)$ denote the maximum n such that there exists an (S^q, n) encoder in class $\mathcal{C}$ constructed from the set of principal states P. Therefore we can write $M_{\mathcal{C}}(q) = \max_{P \subseteq V_G} M_{\mathcal{C}}(q, P)$. Moreover, we say that P *achieves* $M_{\mathcal{C}}(q)$ if $M_{\mathcal{C}}(q, P) = M_{\mathcal{C}}(q)$. We shall later refer to such a set P as an *optimal set of principal states.*

In order to quantify the optimality of block-type-decodable encoders, we need the following notations. Let u and v be any states in a labeled graph G. The *follower set* of u in G, denoted $\mathcal{F}_G(u)$, is the set of all finite words that can be generated from u in G. We shall use $\mathcal{F}_G^q(u,v)$ to denote the set of all words of length q that can be generated in G by paths that start at u and terminate at v. Similarly, for a set of states P, $\mathcal{F}_G^q(u,P)$ denotes the set of all words of length q that can be generated in G by paths that start at u and terminate at a state in P, i.e., $\mathcal{F}_G^q(u,P) = \bigcup_{v\in P}\mathcal{F}_G^q(u,v)$.

The states of a labeled graph are naturally endowed with the *partial ordering* by inclusion of follower sets: $u \preceq v$ if $\mathcal{F}_G(u) \subseteq \mathcal{F}_G(v)$. We say that a set $P \subseteq V_G$ is *complete* if whenever u is in P and $u \preceq v$, then v is also in P.

Based on these notations, Freiman and Wyner [5] showed that

$$M_{\mathrm{blk}}(q) = \max_{P\subseteq V_G}\left|\bigcap_{u\in P}\mathcal{F}_G^q(u,P)\right|.$$

To simplify the search for an optimal P, they further proved that when G has finite memory less than or equal to q, it suffices to consider sets P which are complete. In fact, the following proposition shows that this is true for all classes of block-type-decodable encoders even when the condition that q is greater than the memory is removed.

PROPOSITION 3.1. *Let S be a constrained system with a deterministic presentation G. Let $P \subseteq V_G$ and let P' be the smallest complete set such that $P \subseteq P'$. Then for each class $\mathcal{C}$ of encoder and block length q, $M_{\mathcal{C}}(q,P) \le M_{\mathcal{C}}(q,P')$.*

Proof. Let $v \in P'$. It suffices to show that there is a state $u \in P$ such that $\mathcal{F}_G^q(u,P) \subseteq \mathcal{F}_G^q(v,P')$.

Since P' is the smallest complete set such that $P \subseteq P'$, there must be a state $u \in P$ such that $u \preceq v$. Let $w \in \mathcal{F}_G^q(u,P)$. Since $u \preceq v$, v can also generate w. Since G is deterministic, the outgoing edges from u and v labeled by w are unique. Denote the terminal states of these edges by $\bar{u}$ and $\bar{v}$, respectively. Then $\bar{u} \preceq \bar{v}$ because $u \preceq v$. Hence $\bar{v} \in P'$ because P' is complete and $\bar{u} \in P \subseteq P'$. This implies that $w \in \mathcal{F}_G^q(v,P')$. ∎

Similar expressions for $M_{\mathrm{det}}(q,P)$ and $M_{\mathrm{det}}(q)$ are due to Franaszek [4]:

$$M_{\mathrm{det}}(q,P) = \min_{u\in P}|\mathcal{F}_G^q(u,P)| = \min_{u\in P}\sum_{v\in P}(A_G^q)_{u,v}, \tag{3.1}$$

$$M_{\mathrm{det}}(q) = \max_{P\subseteq V_G}\min_{u\in P}\sum_{v\in P}(A_G^q)_{u,v}.$$

We do not know of a formula for $M_{\mathrm{blkdec}}(q)$ as simple as those above, but, as with $M_{\mathrm{blk}}(q)$ and $M_{\mathrm{det}}(q)$, it is a function of only an arbitrary irreducible deterministic presentation of the constraint, such as the Shannon cover.

4. Stabilization at large block length. We know from the previous section that to design a block-type-decodable encoder, we need to choose a set of principal states. Our goal is to find an optimal set of principal states that maximizes the code rate. In some cases, it is easier to find such an optimal set of principal states at asymptotically large block length. Thus it is desirable if we can relate the optimal sets of principal states at asymptotically large block length to the ones at finite block length. In this section, we study the relationship between the two.

Recall that for a constraint S with the Shannon cover G, $\mathrm{cap}(S) = \log\lambda(A_G)$. When it is clear from the context, we also denote $\lambda(A_G)$ by λ. From the expression for cap(S), we would expect $M_{\mathcal{C}}(q,P)$ to grow as λ^q. Thus it is natural to define $M^q_{\mathcal{C}}(P) = M_{\mathcal{C}}(q,P)/\lambda^q$. Let $M^\infty_{\mathcal{C}}(P) = \lim_{q\to\infty} M^q_{\mathcal{C}}(P)$. In [3, Proposition 3], we showed that $M^\infty_{\text{det}}(P)$ exists for primitive constraints. We shall prove that $M^\infty_{\text{blk}}(P)$ and $M^\infty_{\text{blkdec}}(P)$ exist for primitive constraints in sections 6 and 7, respectively. We define $M^\infty_{\mathcal{C}} = \max_{P\subseteq V_G} M^\infty_{\mathcal{C}}(P)$. We say that a set P is *asymptotically optimal* if $M^\infty_{\mathcal{C}}(P) = M^\infty_{\mathcal{C}}$. Furthermore, define $\mathcal{P}_{\mathcal{C}}(q)$ and $\mathcal{P}^\infty_{\mathcal{C}}$ to be the collection of optimal sets of principal states at block length q and the collection of asymptotically optimal sets of principal states, respectively. Lastly we define $M^*_{\mathcal{C}} = \lim_{q\to\infty} M_{\mathcal{C}}(q)/\lambda^q$.

PROPOSITION 4.1. *For any class $\mathcal{C}$, if $M^\infty_{\mathcal{C}}(P)$ exists for each $P \subseteq V_G$, then the following hold:*

(i) *$\mathcal{P}_{\mathcal{C}}(q) \subseteq \mathcal{P}^\infty_{\mathcal{C}}$ for sufficiently large q.*

(ii) *$M^*_{\mathcal{C}}$ exists and is equal to $M^\infty_{\mathcal{C}}$.*

A proof of Proposition 4.1 is given later in this section. A slightly different version of this proposition for deterministic encoders appears in [3].

Assuming that the condition in Proposition 4.1 is satisfied, it is natural to wonder when (i) holds. In later sections, we give bounds on q such that this holds for each class of encoder. In order to establish those bounds and to prove Proposition 4.1, we need the following lemma. First, define

$$\epsilon_{\mathcal{C}} = M^\infty_{\mathcal{C}} - \max_{P\notin\mathcal{P}^\infty_{\mathcal{C}}} M^\infty_{\mathcal{C}}(P).$$

LEMMA 4.2. *If q satisfies*

$$|M^q_{\mathcal{C}}(P) - M^\infty_{\mathcal{C}}(P)| < \frac{\epsilon_{\mathcal{C}}}{2} \tag{4.1}$$

for each $P \subseteq V_G$, then $\mathcal{P}_{\mathcal{C}}(q) \subseteq \mathcal{P}^\infty_{\mathcal{C}}$.

Proof. Let $P \in \mathcal{P}_{\mathcal{C}}(q)$ and $P^* \in \mathcal{P}^\infty_{\mathcal{C}}$. It follows from (4.1) that

$$M^\infty_{\mathcal{C}}(P) + \frac{\epsilon_{\mathcal{C}}}{2} > M^q_{\mathcal{C}}(P) \geq M^q_{\mathcal{C}}(P^*) > M^\infty_{\mathcal{C}}(P^*) - \frac{\epsilon_{\mathcal{C}}}{2} = M^\infty_{\mathcal{C}} - \frac{\epsilon_{\mathcal{C}}}{2}.$$

Therefore, $M^\infty_{\mathcal{C}} - M^\infty_{\mathcal{C}}(P) < \epsilon_{\mathcal{C}}$, and so $P \in \mathcal{P}^\infty_{\mathcal{C}}$ by the definition of $\epsilon_{\mathcal{C}}$. ∎

In the case that $\mathcal{P}^\infty_{\mathcal{C}}$ has only one element, the condition in the lemma implies that $\mathcal{P}_{\mathcal{C}}(q) = \mathcal{P}^\infty_{\mathcal{C}}$. This allows us to determine the optimal set of principal states at large block length, in particular the block length that satisfies the bounds given in later sections, from the asymptotically optimal set of principal states.

Proof of Proposition 4.1. Suppose that $M^\infty_{\mathcal{C}}(P)$ exists for each $P \subseteq V_G$. Then (4.1) holds for sufficiently large q, and (i) follows by Lemma 4.2.

Since $M^q_{\mathcal{C}}(P)$ is a convergent sequence for each $P \subseteq V_G$,

$$M^*_{\mathcal{C}} = \lim_{q\to\infty} \max_{P\subseteq V_G} M^q_{\mathcal{C}}(P) = \max_{P\subseteq V_G} \lim_{q\to\infty} M^q_{\mathcal{C}}(P) = M^\infty_{\mathcal{C}}.$$

This proves (ii). ∎

5. Stabilization for deterministic encoders. In this section, we study bounds on q such that $\mathcal{P}_{\text{det}}(q) \subseteq \mathcal{P}^\infty_{\text{det}}$ by utilizing the Perron–Frobenius theory [12]. From the Perron–Frobenius theory, an irreducible matrix A has a unique largest positive eigenvalue $\lambda = \lambda(A)$. Moreover, the corresponding right and left eigenvectors, $\mathbf{r}$ and $\mathbf{l}$, have all positive entries. In our context, $\mathbf{r}$ is a column vector and $\mathbf{l}$ is a row vector. Suppose $\mathbf{r}$ and $\mathbf{l}$ are normalized so that $\mathbf{lr} = 1$. Define $\Lambda = \mathbf{rl}$, a rank-one matrix. If A is primitive, then it follows from the Perron–Frobenius theory that

$$\lim_{q\to\infty} \frac{A^q}{\lambda^q} = \Lambda. \tag{5.1}$$

The following result, which gives a characterization of M^*_{det}, is a consequence of [3, Proposition 4] and Proposition 4.1 above.

PROPOSITION 5.1 (see [3]). *For each $P \subseteq V_G$, $M^\infty_{\text{det}}(P)$ exists. Moreover,*

(i) *$\mathcal{P}_{\text{det}}(q) \subseteq \mathcal{P}^\infty_{\text{det}}$ for sufficiently large q,*

(ii) *M^*_{det} exists and is equal to M^∞_{det}.*

Before stating the main result of this section, we provide the definition of the maximum row sum matrix norm. Let A be an $n \times n$ matrix over the complex numbers. Then $\|A\|_\infty$ is defined as

$$\|A\|_\infty = \max_{1 \le i \le n} \sum_{j=1}^{n} |A_{i,j}|.$$

It can also be written as

$$\|A\|_\infty = \max_{\|x\|_\infty = 1} \|Ax\|_\infty.$$

The following theorem provides a bound on block length q such that an optimal set of principal states for a deterministic encoder is also asymptotically optimal.

THEOREM 5.2. *If q satisfies*

$$\left\| \frac{A^q}{\lambda^q} - \Lambda \right\|_\infty < \frac{\epsilon_{\text{det}}}{2}, \tag{5.2}$$

then $\mathcal{P}_{\text{det}}(q) \subseteq \mathcal{P}^\infty_{\text{det}}$.

Proof. We shall show that if (5.2) holds, then for each P,

$$|M^q_{\text{det}}(P) - M^\infty_{\text{det}}(P)| < \frac{\epsilon_{\text{det}}}{2},$$

and the theorem follows from Lemma 4.2.

Let $\mathbf{x} = (x_u)$ be the *characteristic vector* of P, that is, a 0-1 vector of dimension $|V_G|$ such that $x_u = 1$ if $u \in P$ and $x_u = 0$ otherwise. Define

$$\mathbf{y} = \frac{A^q}{\lambda^q}\mathbf{x},$$
$$\mathbf{z} = \Lambda\mathbf{x}.$$

From (3.1) and (5.1), one can show that [3]

$$M^q_{\text{det}}(P) = \min_{u \in P} y_u \tag{5.3}$$

and

$$M^\infty_{\text{det}}(P) = \min_{u \in P} z_u. \tag{5.4}$$

Let u and v be states achieving the minimum in (5.3) and (5.4), respectively. Then $y_u = M^q_{\text{det}}(P)$ and $z_v = M^\infty_{\text{det}}(P)$. Furthermore, $y_u \le y_v$ and $z_v \le z_u$. Since $\mathbf{x}$ is a 0-1 vector, $\|\mathbf{x}\|_\infty = 1$. Then it follows from (5.2) that

$$\|\mathbf{y} - \mathbf{z}\|_\infty = \left\| \frac{A^q}{\lambda^q}\mathbf{x} - \Lambda\mathbf{x} \right\|_\infty < \frac{\epsilon_{\text{det}}}{2}.$$

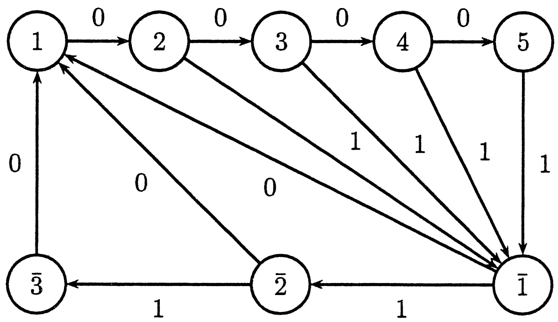

FIG. 5.1. *Shannon cover of the asymmetric-RLL*$(2,5,1,3)$.

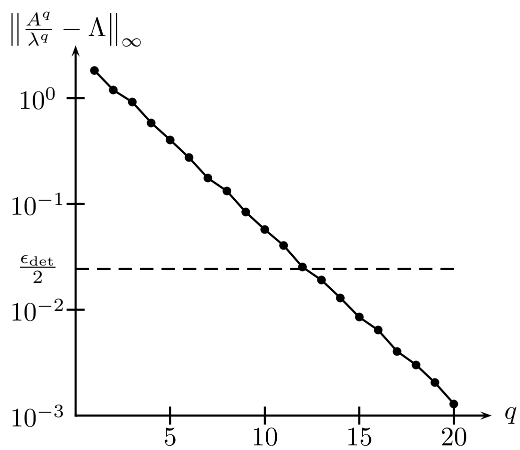

FIG. 5.2. $\|\frac{A^q}{\lambda^q} - \Lambda\|_\infty$.

This implies that $|y_u - z_u| < \epsilon_{\text{det}}/2$ and $|y_v - z_v| < \epsilon_{\text{det}}/2$. We want to show that $|y_u - z_v| < \epsilon_{\text{det}}/2$.

Case 1. Suppose $y_u - z_v \geq \epsilon_{\text{det}}/2$. Since $y_u \leq y_v$, we have $y_v - z_v \geq \epsilon_{\text{det}}/2$, a contradiction.

Case 2. Suppose $y_u - z_v \leq -\epsilon_{\text{det}}/2$. Since $z_v \leq z_u$, we have $y_u - z_u \leq -\epsilon_{\text{det}}/2$, a contradiction.

Thus we conclude that $|M^q_{\text{det}}(P) - M^\infty_{\text{det}}(P)| < \epsilon_{\text{det}}/2$. ∎

Example 5.3. The Shannon cover for the asymmetric-RLL$(2,5,1,3)$ constraint is shown in Figure 5.1.

In contrast to RLL constraints [8, 6], there is no known explicit characterization of the optimal sets of principal states for the asymmetric-RLL constraint. However, we can numerically compute $M_{\text{det}}(q)$, M^*_{det}, and the achieving set of principal states easily. We obtain $M^*_{\text{det}} = 0.7563$, $\epsilon_{\text{det}} = 0.0487$, and $P^*_{\text{det}} = \{1,2,3,4,\bar{1},\bar{2}\}$ is the only asymptotically optimal set of principal states.

We compute $\|\frac{A^q}{\lambda^q} - \Lambda\|_\infty$ explicitly for small values of q in Figure 5.2. The plot suggests that $\|\frac{A^q}{\lambda^q} - \Lambda\|_\infty < \epsilon_{\text{det}}/2$ holds for $q \geq 13$. Since we do not know whether $\|\frac{A^q}{\lambda^q} - \Lambda\|_\infty$ is decreasing with q, we will compute an upper bound for $\|\frac{A^q}{\lambda^q} - \Lambda\|_\infty$.

In this example, A is diagonalizable: $A = TDT^{-1}$, where $D = \operatorname{diag}[\lambda_i]$ is a diagonal matrix with $\lambda = |\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_8|$. Moreover, the first column of T and the first row of T^{-1} are, respectively, the right ($\mathbf{r}$) and left ($\mathbf{l}$) eigenvectors of A associated with the eigenvalue λ normalized so that $\mathbf{lr} = 1$. Then we have

$$\begin{aligned}
\left\| \frac{A^q}{\lambda^q} - \Lambda \right\|_\infty &= \left\| \frac{1}{\lambda^q} T D^q T^{-1} - \mathbf{rl} \right\|_\infty \\
&= \left\| \frac{1}{\lambda^q} T \begin{bmatrix} 0 & & & \\ & \lambda_2^q & & \\ & & \ddots & \\ & & & \lambda_8^q \end{bmatrix} T^{-1} \right\|_\infty \\
&\leq \frac{1}{\lambda^q} \|T\|_\infty \left\| \begin{bmatrix} 0 & & & \\ & \lambda_2^q & & \\ & & \ddots & \\ & & & \lambda_8^q \end{bmatrix} \right\|_\infty \|T^{-1}\|_\infty \\
&= \|T\|_\infty \|T^{-1}\|_\infty \left(\frac{|\lambda_2|}{\lambda} \right)^q \\
&= (2.8811)(3.8981) \left(\frac{1.1271}{1.6372} \right)^q .
\end{aligned}$$

If $q \geq 17$, then

$$\left\| \frac{A^q}{\lambda^q} - \Lambda \right\|_\infty \leq 11.2308(0.6884)^{17} = 0.0197 < 0.0243 = \frac{\epsilon_{\text{det}}}{2}.$$

Therefore P^*_{det} is the only optimal set of principal states for $q \geq 17$.

In fact, by computing $M_{\text{det}}(q, P)$ for $1 \leq q \leq 12$, one can show that P^*_{det} is optimal for all q and is the only optimal set of principal states precisely when $q = 5$ and $q \geq 7$.

With the motivation from the above example, we offer the following corollary.

COROLLARY 5.4. *Let λ_i be the distinct eigenvalues of A with $\lambda_1 = \lambda = \lambda(A)$. Let s_i be the multiplicity of λ_i. Let T be a transformation matrix which decomposes A into Jordan canonical form. If*

$$\frac{1}{\lambda^q} \|T\|_\infty \|T^{-1}\|_\infty \max_{i \geq 2} \sum_{k=0}^{s_i - 1} \binom{q}{k} |\lambda_i|^{q-k} < \frac{\epsilon_{\text{det}}}{2},$$

then $\mathcal{P}_{\text{det}}(q) \subseteq \mathcal{P}^\infty_{\text{det}}$.

Proof. Let J be the Jordan form of all eigenvalues of A other than λ; then

$$\begin{aligned}
\left\| \frac{A^q}{\lambda^q} - \Lambda \right\|_\infty &= \left\| \frac{1}{\lambda^q} T \begin{bmatrix} \lambda^q & 0 \\ 0 & J^q \end{bmatrix} T^{-1} - \Lambda \right\|_\infty \\
&= \left\| \frac{1}{\lambda^q} T \begin{bmatrix} 0 & 0 \\ 0 & J^q \end{bmatrix} T^{-1} \right\|_\infty \\
&\leq \frac{1}{\lambda^q} \|T\|_\infty \|T^{-1}\|_\infty \|J^q\|_\infty \\
&= \frac{1}{\lambda^q} \|T\|_\infty \|T^{-1}\|_\infty \max_{i \geq 2} \|J_i^q\|_\infty, \qquad (5.5)
\end{aligned}$$

where J_i is the Jordan (sub)matrix associated with λ_i.

The Jordan matrix J_i can have several forms. The one which yields the largest $\|J_i^q\|_\infty$ is the one with single block

$$J_i = \begin{bmatrix} \lambda_i & 1 & 0 & \cdots & 0 \\ 0 & \lambda_i & 1 & \cdots & 0 \\ 0 & 0 & \lambda_i & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_i \end{bmatrix}_{s_i \times s_i}.$$

One can show that

$$J_i^q = \begin{bmatrix} \lambda_i^q & \binom{q}{1}\lambda_i^{q-1} & \binom{q}{2}\lambda_i^{q-2} & \cdots & \binom{q}{s_i-1}\lambda_i^{q-s_i+1} \\ 0 & \lambda_i^q & \binom{q}{1}\lambda_i^{q-1} & \cdots & \binom{q}{s_i-2}\lambda_i^{q-s_i+2} \\ 0 & 0 & \lambda_i^q & \cdots & \binom{q}{s_i-3}\lambda_i^{q-s_i+3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \lambda_i^q \end{bmatrix}.$$

Therefore

$$\|J_i^q\|_\infty = \sum_{k=0}^{s_i-1} \binom{q}{k} |\lambda_i|^{q-k}.$$

Then it follows from (5.5) that

$$\left\| \frac{A^q}{\lambda^q} - \Lambda \right\|_\infty \le \frac{1}{\lambda^q} \|T\|_\infty \|T^{-1}\|_\infty \max_{i \ge 2} \sum_{k=0}^{s_i-1} \binom{q}{k} |\lambda_i|^{q-k},$$

and the corollary follows from Theorem 5.2. □

6. Stabilization for block encoders. In this section, we present an algorithm which computes $M_{\text{blk}}(q)$ and M^*_{blk} together with the achieving sets of principal states. Stabilization of block encoders is also studied.

Let G be a labeled graph. Define $T_G(w, v)$ to be the subset of states of G which are the terminal states of the paths labeled by w starting from state v. (Note that $T_G(w, v)$ has only one state if G is deterministic.)

DEFINITION 6.1. *Let G be a labeled graph. We define $\bar{G}$ to be the graph with $V_{\bar{G}}$ being the set of all nonempty subsets of V_G, with an edge from U to V labeled by w if*
1. *for each $u \in U$, there is an outgoing edge with label w,*
2. *$\bigcup_{u \in U} T_G(w, u) = V$.*

We denote by $\bar{A}$ the adjacency matrix of $\bar{G}$.

This graph $\bar{G}$ is typically reducible and is closely related to the subset construction in finite automata theory. Note that $S(\bar{G}) = S(G)$. Moreover, $\bar{G}$ is always deterministic.

Example 6.2. The Shannon cover G and the corresponding $\bar{G}$ of RLL(1,2) are shown in Figures 6.1 and 6.2. By viewing each state u as a singleton subset $\{u\}$, we see that G is a subgraph of $\bar{G}$. (This is true for any deterministic graph.)

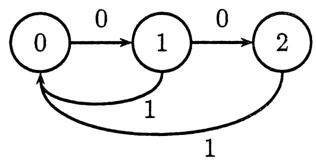

FIG. 6.1. *Shannon cover G of RLL*$(1,2)$.

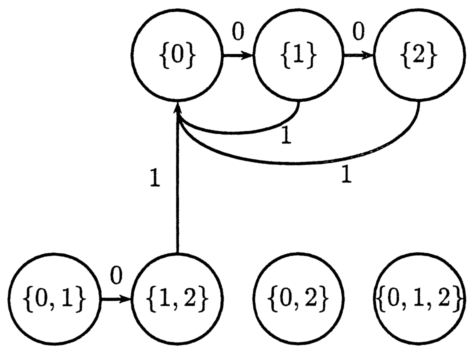

FIG. 6.2. $\bar{G}$ *for RLL*$(1,2)$.

LEMMA 6.3.

$$\overline{G^q} = (\bar{G})^q.$$

Proof. First observe that the sets of vertices of $\overline{G^q}$ and $(\bar{G})^q$ are the same. Next, we can see that there is an edge from U to V in $\overline{G^q}$ if and only if there is a sequence w and a path of length q labeled by w from every state $u \in U$ such that $\bigcup_{u\in U} T_G(w,u) = V$. The same is true for $(\bar{G})^q$. Because $\overline{G^q}$ and $(\bar{G})^q$ are deterministic by construction, the edge is unique and we can conclude that $\overline{G^q} = (\bar{G})^q$. ∎

The next theorem shows how to compute $M_{\mathrm{blk}}(q,P)$ from $\bar{A}$.

THEOREM 6.4. *Let S be a constrained system and let G be a deterministic presentation of S. Let $\bar{A}$ be the adjacency matrix of $\bar{G}$. Then*

$$M_{\mathrm{blk}}(q,P) = \sum_{U\subseteq P} \bar{A}^q_{P,U}.$$

Proof. From the definition of $M_{\mathrm{blk}}(q,P)$, a word that can be counted for $M_{\mathrm{blk}}(q,P)$ must be generated by an edge from every state in P and the terminal state for this edge must be in P. Hence,

$$\begin{aligned} M_{\mathrm{blk}}(q,P) &= \left|\bigcup_{U\subseteq P} \mathcal{F}^1_{\overline{G^q}}(P,U)\right| = \left|\bigcup_{U\subseteq P} \mathcal{F}^1_{(\bar{G})^q}(P,U)\right| \qquad \text{(by Lemma 6.3)} \\ &= \left|\bigcup_{U\subseteq P} \mathcal{F}^q_{(\bar{G})}(P,U)\right| = \sum_{U\subseteq P} \bar{A}^q_{P,U}, \end{aligned}$$

where the last equality follows from the fact that $\bar{G}$ is deterministic. ∎

LEMMA 6.5. *Let S be a primitive constrained system and let G be the Shannon cover of S with adjacency matrix A. Then the adjacency matrix $\bar{A}$ of $\bar{G}$ has the following properties:*

(i) *$\bar{A}$ has a unique largest eigenvalue $\lambda = \lambda(A)$.*

(ii) *The right ($\bar{\mathbf{r}}$) and left ($\bar{\mathbf{l}}$) eigenvectors associated with λ are nonnegative. Furthermore, if the states of $\bar{G}$ are ordered so that the first $|V_G|$ states are of the form $\{u\}$, where $u \in V_G$ (subset of size one), then $\bar{\mathbf{r}}$ and $\bar{\mathbf{l}}$ have the form*

$$\bar{\mathbf{r}} = \begin{bmatrix} \mathbf{r} \\ \mathbf{s} \end{bmatrix}, \quad \bar{\mathbf{l}} = \begin{bmatrix} \mathbf{l} & 0 \end{bmatrix},$$

where $\mathbf{r}$ and $\mathbf{l}$ are the right and left eigenvectors of A associated with λ.

(iii) *Suppose $\bar{\mathbf{r}}$ and $\bar{\mathbf{l}}$ are normalized so that $\bar{\mathbf{l}}\bar{\mathbf{r}} = 1$ (equivalently $\mathbf{l}\mathbf{r} = 1$), and define $\bar{\Lambda} = \bar{\mathbf{r}}\bar{\mathbf{l}}$. Then $\lim_{q\to\infty} \frac{\bar{A}^q}{\lambda^q} = \bar{\Lambda}$.*

Proof.

(i) Because G is deterministic, G is a subgraph of $\bar{G}$. In particular, G is an irreducible component of $\bar{G}$. Since G is the Shannon cover of S, there must be a *homing word* h for a state in G [10, Lemma 2.10] (i.e., all paths in G that generate h must terminate in the same state). Let H be another irreducible component of $\bar{G}$. Then H cannot generate h because any path with label h must end in G. Therefore $S(H)$ is a proper subset of S. Thus $\lambda(A_H) < \lambda$ by [9, Corollary 4.4.9]. Since the set of eigenvalues of $\bar{A}$ is the union of the sets of eigenvalues of the adjacency matrices of the irreducible components of $\bar{G}$, we conclude that λ is the unique largest eigenvalue of $\bar{A}$.

(ii) It is easy to see that $\bar{A}$ has the form

$$\bar{A} = \begin{bmatrix} A & 0 \\ C & D \end{bmatrix}.$$

Let $\bar{\mathbf{l}} = \begin{bmatrix} \bar{\mathbf{l}}_1 & \bar{\mathbf{l}}_2 \end{bmatrix}$. Then the left eigenvector equation is

$$\begin{bmatrix} \bar{\mathbf{l}}_1 A + \bar{\mathbf{l}}_2 C & \bar{\mathbf{l}}_2 D \end{bmatrix} = \lambda \begin{bmatrix} \bar{\mathbf{l}}_1 & \bar{\mathbf{l}}_2 \end{bmatrix}.$$

From (i), λ is larger than all eigenvalues of D. Thus $\bar{\mathbf{l}}_2 = 0$. Moreover, $\bar{\mathbf{l}}_1 = \mathbf{l}$ is the left eigenvector of A corresponding to λ.

On the other hand, let

$$\bar{\mathbf{r}} = \begin{bmatrix} \bar{\mathbf{r}}_1 \\ \bar{\mathbf{r}}_2 \end{bmatrix}.$$

Then the right eigenvector equation is

$$\begin{bmatrix} A\bar{\mathbf{r}}_1 \\ C\bar{\mathbf{r}}_1 + D\bar{\mathbf{r}}_2 \end{bmatrix} = \begin{bmatrix} \lambda\bar{\mathbf{r}}_1 \\ \lambda\bar{\mathbf{r}}_2 \end{bmatrix}.$$

This implies that $\bar{\mathbf{r}}_1 = \mathbf{r}$ is the right eigenvector of A associated with λ and

$$\begin{aligned} (\lambda I - D)\bar{\mathbf{r}}_2 &= C\mathbf{r}, \\ \bar{\mathbf{r}}_2 &= (\lambda I - D)^{-1} C\mathbf{r} \\ &= \lambda^{-1}\left(I + \frac{D}{\lambda} + \frac{D^2}{\lambda^2} + \cdots\right) C\mathbf{r} \\ &\geq 0. \end{aligned}$$

(iii) $\bar{A}$ can be transformed into Jordan canonical form as

$$\bar{A} = \begin{bmatrix} \bar{\mathbf{r}} & R \end{bmatrix} \begin{bmatrix} \lambda & 0 \\ 0 & J \end{bmatrix} \begin{bmatrix} \bar{\mathbf{l}} \\ L \end{bmatrix},$$

where J comprises eigenvalues of D and A not equal to λ. From (i), all eigenvalues of D have magnitude less than λ. Moreover, it can be shown that the Shannon cover of a primitive constraint is primitive. Thus all eigenvalues of A not equal to λ have magnitude less than λ. Therefore

$$\bar{A}^q = \bar{\mathbf{r}}\bar{\mathbf{l}}\lambda^q + o(\lambda^q),$$

where $\lim_{q\to\infty} o(\lambda^q)/\lambda^q = 0$. Then the result follows. □

The following gives a characterization of M^*_{blk}.

THEOREM 6.6. *Let* $\bar{\mathbf{r}}$ *and* $\bar{\mathbf{l}}$ *be as in* (iii) *of Lemma* 6.5. *For a primitive constrained system,*

$$M^\infty_{\mathrm{blk}}(P) = \bar{\mathbf{r}}_P \sum_{u\in P} \bar{\mathbf{l}}_{\{u\}}.$$

Moreover,

(i) $\mathcal{P}_{\mathrm{blk}}(q) \subseteq \mathcal{P}^\infty_{\mathrm{blk}}$ *for sufficiently large* q,

(ii) $M^*_{\mathrm{blk}} = \max_{P\subseteq V_G} \left(\bar{\mathbf{r}}_P \sum_{u\in P} \bar{\mathbf{l}}_{\{u\}}\right)$.

Proof. From Theorem 6.4, $M^q_{\mathrm{blk}}(P) = \frac{1}{\lambda^q} \sum_{U\subseteq P} \bar{A}^q_{P,U}$. Thus from (iii) of Lemma 6.5, we have

$$M^\infty_{\mathrm{blk}}(P) = \lim_{q\to\infty} M^q_{\mathrm{blk}}(P) = \sum_{U\subseteq P} \bar{\Lambda}_{P,U} = \bar{\mathbf{r}}_P \sum_{U\subseteq P} \bar{\mathbf{l}}_U.$$

Since $\bar{\mathbf{l}} = \begin{bmatrix} \mathbf{l} & 0 \end{bmatrix}$,

$$M^\infty_{\mathrm{blk}}(P) = \bar{\mathbf{r}}_P \sum_{u\in P} \bar{\mathbf{l}}_{\{u\}}.$$

Then (i) and (ii) follow from Proposition 4.1. □

THEOREM 6.7. *If* q *satisfies*

$$\left\| \frac{\bar{A}^q}{\lambda^q} - \bar{\Lambda} \right\|_\infty < \frac{\epsilon_{\mathrm{blk}}}{2},$$

then $\mathcal{P}_{\mathrm{blk}}(q) \subseteq \mathcal{P}^\infty_{\mathrm{blk}}$.

Proof. Let P be any set of principal states and let $\mathbf{x} = (x_U)$ be a 0-1 vector of dimension $|V_{\bar{G}}|$ such that $x_U = 1$ if $U \subseteq P$ and $x_U = 0$ otherwise. Then

$$\begin{aligned} |M^q_{\mathrm{blk}}(P) - M^\infty_{\mathrm{blk}}(P)| &= \left| \left(\frac{\bar{A}^q}{\lambda^q}\mathbf{x} \right)_P - (\bar{\Lambda}\mathbf{x})_P \right| \\ &\le \left\| \frac{\bar{A}^q}{\lambda^q} - \bar{\Lambda} \right\|_\infty < \frac{\epsilon_{\mathrm{blk}}}{2}. \end{aligned}$$

Then the theorem follows from Lemma 4.2. □

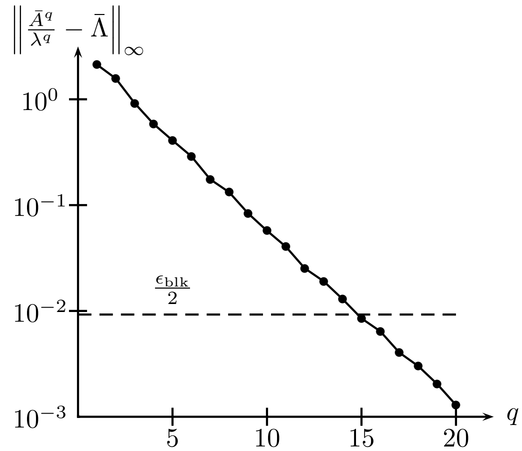

FIG. 6.3. $\|\frac{\bar{A}^q}{\lambda^q} - \bar{\Lambda}\|_\infty$.

Example 6.8. Consider the asymmetric-RLL$(2,5,1,3)$ described in Example 5.3. It is found that $M^*_{\rm blk} = 0.3445$ and the only asymptotically optimal sets of principal states are $P^*_{\rm blk1} = \{2,3,\bar{1}\}$ and $P^*_{\rm blk2} = \{2,\bar{1},\bar{2}\}$. The second largest $M^\infty_{\rm blk}(P)$ is 0.3260 when $P = \{2,\bar{1}\}$ and $\{2,3,\bar{1},\bar{2}\}$. Therefore $\epsilon_{\rm blk} = 0.3445 - 0.3260 = 0.0185$.

We plot $\|\frac{\bar{A}^q}{\lambda^q} - \bar{\Lambda}\|_\infty$ in Figure 6.3. The plot suggests that $\|\frac{\bar{A}^q}{\lambda^q} - \bar{\Lambda}\|_\infty < \epsilon_{\rm blk}/2$ for $q \geq 15$. This would imply that either $P^*_{\rm blk1}$ or $P^*_{\rm blk2}$ (or both) is an optimal set of principal states for $q \geq 15$.

The set of eigenvalues of $\bar{A}$ comprises the eight eigenvalues of A, all of which are nonzero and have multiplicity 1, and a zero eigenvalue with large multiplicity. Computing a transformation matrix for a matrix with an eigenvalue having such a large multiplicity is unstable; thus the idea in Corollary 5.4 cannot be directly applied. However, since the Shannon cover G has memory 5, all paths of length $q \geq 5$ in G that carry the same label must terminate at the same state. Therefore, assuming $q \geq 5$, every path of length q in $\bar{G}$ must terminate at a state of the form $\{u\}$ (a singleton state). Hence, $\bar{A}^q$ has only eight nonzero columns (that correspond to the singleton subsets). It follows that the Jordan blocks of $\bar{A}^q$ that correspond to the zero eigenvalue become zero. For this reason, when $q \geq 5$, we can write

$$\bar{A}^q = \bar{R}\bar{D}^q\bar{L} = \begin{bmatrix} R & 0 \end{bmatrix} \begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}^q \begin{bmatrix} L \\ 0 \end{bmatrix},$$

where D is the diagonal matrix containing the eigenvalues of A, and R and L contain the right and left eigenvectors of $\bar{A}$ corresponding to these eigenvalues, normalized so that LR is the identity matrix. Now we apply Theorem 6.7:

$$\begin{aligned} \left\| \frac{\bar{A}^q}{\lambda^q} - \bar{\Lambda} \right\|_\infty &\leq \|\bar{R}\|_\infty \|\bar{L}\|_\infty \left(\frac{|\lambda_2|}{\lambda} \right)^q \\ &= (5.9628)(2.7878) \left(\frac{1.1271}{1.6372} \right)^q. \end{aligned}$$

If $q \geq 21$, then

$$\left\| \frac{\bar{A}^q}{\lambda^q} - \bar{\Lambda} \right\|_\infty \leq (16.6233)(0.6884)^{21} = 0.0065 < 0.0092 = \frac{\epsilon_{\rm blk}}{2}.$$

(We remark that $\|\bar{R}\|_\infty \|\bar{L}\|_\infty$ is not unique; different normalization of L and R gives different $\|\bar{R}\|_\infty \|\bar{L}\|_\infty$.)

By explicitly computing $M_{\text{blk}}(q, P)$ for $1 \leq q \leq 30$, we find that P^*_{blk1} and P^*_{blk2} are optimal for all q in that range except $q = 5$. Moreover, they both are the only optimal sets of principal states when $8 \leq q \leq 30$.

We can further analyze the block codes $\mathcal{L}_1$ and $\mathcal{L}_2$ supported by P^*_{blk1} and P^*_{blk2}. One can show that $\mathcal{L}_1$ comprises words with prefix 001, 10, or 110, and suffix 100, 1000, or 01. Similarly, $\mathcal{L}_2$ comprises words with prefix 001, 0001, or 10, and suffix 100, 01, or 011. Thus, a word $\mathbf{w} = w_1 w_2 \cdots w_q$ is in $\mathcal{L}_1$ if and only if its reversal $w_q w_{q-1} \cdots w_1$ is in $\mathcal{L}_2$. Therefore $M_{\text{blk}}(q, P^*_{\text{blk1}}) = |\mathcal{L}_1| = |\mathcal{L}_2| = M_{\text{blk}}(q, P^*_{\text{blk2}})$. We can now conclude that P^*_{blk1} and P^*_{blk2} are optimal for all q except $q = 5$ and are the only optimal sets of principal states when $q \geq 8$.

7. Stabilization for block-decodable encoders. Among block-type-decodable encoders, we know the least about block-decodable encoders. In this section, we show that $M^\infty_{\text{blkdec}}(P)$ and M^*_{blkdec} exist for any primitive constraint. Computation of asymptotically optimal sets of principal states is described. We also give a bound on q such that $\mathcal{P}_{\text{blkdec}}(q) \subseteq \mathcal{P}^\infty_{\text{blkdec}}$.

First consider the following input tag assignment problem. For a given deterministic graph G, we wish to find a block-decodable encoder that is a subgraph of G and has the same set of states as G. We can proceed as follows.

```
INPUT TAG ASSIGNMENT
    Ψ ← set of all edge labels of G
    τ ← 1
    while (it is possible to choose a set of edge labels ψ = {w_1, ..., w_k} ⊆ Ψ
            such that each state of G can generate at least one w_i)
            do assign tag τ to each label in ψ
                τ ← τ + 1
                Ψ ← Ψ \ ψ
```

After the assignment, we obtain a desired encoder by keeping outgoing edges with distinct labels at each state and removing the other edges.

If we choose ψ wisely, the algorithm will give an optimal block-decodable encoder. Unfortunately, it is not clear how to choose ψ to maximize the number of tags; thus an algorithm to choose ψ is needed. We will use integer and linear programming to tackle this problem. Because the upcoming formulation of the integer programming problem involves many complex notations, we offer the following example to illustrate the idea.

Example 7.1. Let S be the constrained system presented by G in Figure 7.1. To simplify the figure, we draw only one edge for parallel edges. For example, state I has two edges to state J labeled by w_3 and w_4.

We wish to find an optimal block-decodable encoder for S. First we fix the set of principal states $P = \{I, J, K\}$ and compute $M_{\text{blkdec}}(1, P)$. Consider the subgraph of G with the set of states P. We divide the labels of this subgraph into groups so that labels are in the same group if the sets of states that can generate them are equal. The diagram in Figure 7.2 summarizes this.

From the diagram, only I can generate w_3, only I and J can generate u, w_4, w_5, and so on. We will denote each region in the diagram by a subset of P; for example, the region that contains w_2 and w_6 is denoted by $\{I, K\}$.

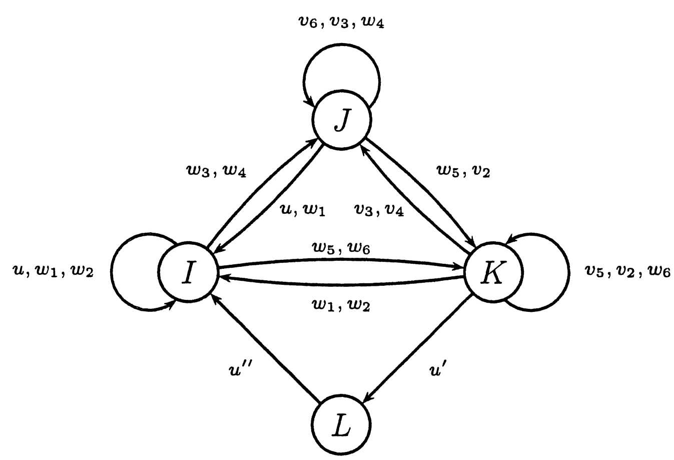

FIG. 7.1. *G in Example* 7.1.

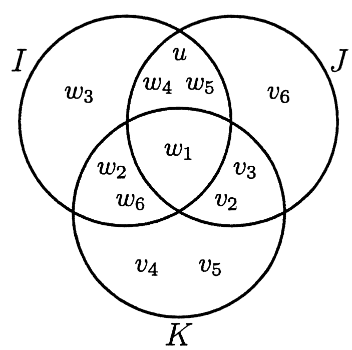

FIG. 7.2. *Partition of labels based on initial states for Example* 7.1.

From the input tag assignment algorithm, we choose a set of labels such that each state in P can generate at least one label. For instance, we can choose $\{w_1\}$ because every state in P can generate w_1. Then we assign tag 1 to all edges labeled by w_1. Also, we can choose $\{v_2, w_2\}$ because I and K can generate w_2 and J and K can generate v_2. So we assign tag 2 to all edges labeled by v_2 and w_2. Choosing a set of labels like this determines a cover of P. For example, choosing $\{w_1\}$ determines $\{\{I, J, K\}\}$. Also, choosing $\{v_2, w_2\}$ determines $\{\{J, K\}, \{I, K\}\}$. To obtain an optimal encoder, we only need to choose a set of labels that determines a minimal cover of P, that is, a cover for which removing a single member destroys the covering property [15].

For the design of codes, it can be seen that only the number of labels in each region is needed. For this reason, we further simplify the diagram to Figure 7.3.

It can be seen that there are eight minimal covers of P. We denote cover i by a 0-1 vector $\mathbf{z}_i = (z_U)$ of size $2^{|P|} - 1 = 7$ indexed by subsets of P such that $z_U = 1$ if U is in the cover and $z_U = 0$ otherwise. Let c_i denote the number of times that we choose cover i. Then the input tag assignment problem becomes an integer programming problem:

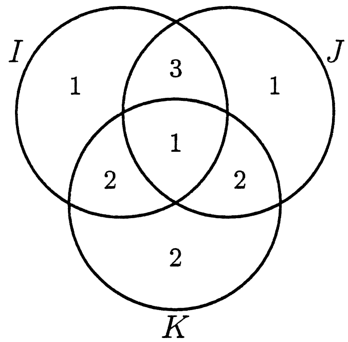

FIG. 7.3. *Number of labels based on initial states for Example* 7.1.

maximize $c_1 + c_2 + \cdots + c_8$
subject to $c_i \in \mathbb{Z}$,
$c_i \geq 0$,

$$\sum_{i=1}^{8} c_i \mathbf{z}_i = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ c_5 \\ c_6 \\ c_7 \\ c_8 \end{bmatrix} \leq \begin{bmatrix} 1 \\ 1 \\ 2 \\ 3 \\ 2 \\ 2 \\ 1 \end{bmatrix}. \tag{7.1}$$

A solution to this problem is $(c_i) = [\ 0\ \ 1\ \ 1\ \ 2\ \ 0\ \ 0\ \ 1\ \ 1\ ]^T$ which gives $M_{\text{blkdec}}(1, P) = \sum_i c_i = 6$. This solution can be achieved by assigning tag j to the edges labeled by v_j or w_j. Compare this to $M_{\det}(1, P) = 7$ (minimum of the sums of each circle in Figure 7.3) and $M_{\text{blk}}(1, P) = 1$ (the size of the region $\{I, J, K\}$).

It can be shown that no other sets of principal states can give $M_{\text{blkdec}}(1, P) = 6$. This can be easily checked by computing $M_{\det}(1, P)$. The maximum of $M_{\det}(1, P)$ for $P \neq \{I, J, K\}$ is 5; thus $M_{\text{blkdec}}(1, P) \leq 5$ for $P \neq \{I, J, K\}$.

Now we give a formal description of how to relate the input tag assignment problem to an integer programming problem. Let G be a deterministic presentation of a constrained system S. Suppose w is a word and $P \subseteq V_G$; we define

$$D_G(w, P) = \{u \in P \ :\ w \in \mathcal{F}_G(u, P)\}.$$

We say that two words w_1 and w_2 are equivalent with respect to P if $D_G(w_1, P) = D_G(w_2, P)$. Clearly this is an equivalence relation. Therefore all words can be grouped into classes; each class is identified with $D_G(w, P)$, a subset of P, where w belongs to that class. In Example 7.1, this is the same as arranging words in Figure 7.2.

Define $\mathbf{d}_G(q, P)$ to be the $2^{|P|} - 1$ tuple indexed by nonempty subsets of P: for each $U \subseteq P$, $d_G(q, P)_U = |\{w \ :\ D_G(w, P) = U \text{ and } |w| = q\}|$, i.e., the number of words of length q in class U. In Example 7.1, this $\mathbf{d}_G(q, P)$ represents the vector in the right-hand side of (7.1).

We claim that $\mathbf{d}_G(q, P)$ is determined by $\bar{A}^q$. To see this, let M be a $(2^{|V_G|} - 1) \times (2^{|V_G|} - 1)$ matrix indexed by the nonempty subsets of V_G. For each nonempty

$U \subseteq P$, let $\gamma_M(U,P) = \sum_{V \subseteq P} M_{U,V}$. In particular when $M = \bar{A}^q$, $\gamma_{\bar{A}^q}(U,P)$ is the number of words of length q that can be generated by every state in U with terminal state in P. Note that $\gamma_{\bar{A}^q}(U,P)$ overcounts $d_G(q,P)_U$ because it also counts words generated from proper supersets of U. To compute $\mathbf{d}_G(q,P)$, define

$$\Delta(M,U,P) = \gamma_M(U,P) - \sum_{\{v\} \subseteq P \setminus U} \gamma_M(U \cup \{v\}, P)$$

$$+ \sum_{\{v_1,v_2\} \subseteq P \setminus U} \gamma_M(U \cup \{v_1,v_2\}, P) - \cdots (-1)^{|P|-|U|} \gamma_M(P,P).$$

Then it follows from the principle of inclusion and exclusion that

$$d_G(q,P)_U = \Delta(\bar{A}^q, U, P).$$

Define $\mathbf{d}_G^\infty(P) = \lim_{q \to \infty} \mathbf{d}_G(q,P)/\lambda^q$. It follows from (iii) of Lemma 6.5 that if S is primitive and G is the Shannon cover of S, then $\mathbf{d}_G^\infty(P)$ exists and $d_G^\infty(P)_U = \Delta(\bar{\Lambda}, U, P)$.

By following the idea in Example 7.1, we view the classes of words as subsets of P. Then we choose a minimal cover of P which can be represented by the vector $\mathbf{z}$. Let $t = t(|P|)$ be the number of minimal covers of P. ($t = 8$ in Example 7.1.) Then the problem of finding an input tag assignment which achieves $M_{\mathrm{blkdec}}(q,P)$ becomes an integer programming problem:

maximize	$c_1 + c_2 + \cdots + c_t$,
subject to	$c_i \in \mathbb{Z}$,
	$c_i \geq 0$,
	$c_1\mathbf{z}_1 + c_2\mathbf{z}_2 + \cdots + c_t\mathbf{z}_t \leq \mathbf{d}_G(q,P)$.

If we delete the first condition, this becomes a linear programming problem. We view the maximum of the objective function of this relaxed problem as a function $\mu(\mathbf{x})$ whose argument $\mathbf{x}$ represents $\mathbf{d}_G(q,P)$ above. ($\mathbf{x}$ is allowed to be real.) So the value of $\mu(\mathbf{x})$ is $\sum_{i=1}^t c_i$, where (c_i) is a solution to the relaxed problem. This defines $\mu(\mathbf{x})$ for a vector $\mathbf{x}$ of fixed dimension. We can generalize the domain of μ to include all nonnegative real vectors with dimension of the form $2^n - 1$, $1 \leq n \leq |V_G|$. In this way, we define $\mu(\mathbf{d}_G(q,P))$ for any P. We can show the following properties of μ.

Proposition 7.2. *Let $\mathbb{R}_{\geq 0}$ denote the set of nonnegative reals.*

(i) *$\mu(a\mathbf{x}) = a\mu(\mathbf{x})$ for any $a \in \mathbb{R}_{\geq 0}$.*

(ii) *$|\mu(\mathbf{x}) - \mu(\mathbf{y})| \leq \|\mathbf{x} - \mathbf{y}\|_1$ for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}_{\geq 0}^{2^n-1}$.*

Proof.

(i) Since the case $a = 0$ is trivially true, we assume that $a > 0$. Suppose $\mathbf{c} = (c_i)$ is a solution to the linear programming problem with input $\mathbf{x}$. Then $a\mathbf{c}$ satisfies the condition of the problem when the input is $a\mathbf{x}$. Thus $\mu(a\mathbf{x}) \geq a\sum_i c_i = a\mu(\mathbf{x})$. Using the same argument with $\mathbf{x}$ replaced by $a\mathbf{x}$ and a replaced by $1/a$, we can show that $\mu(a\mathbf{x}) \leq a\mu(\mathbf{x})$. Therefore $\mu(a\mathbf{x}) = a\mu(\mathbf{x})$.

(ii) It is sufficient to show this when $\mathbf{x}$ and $\mathbf{y}$ differ at only one entry; then the proposition follows from the triangle inequality. Suppose $\mathbf{x}$ and $\mathbf{y}$ differ at only the jth entry. Without loss of generality, assume $x_j > y_j$. Suppose $\mathbf{c}$ is a solution to the problem when $\mathbf{x}$ is the input. Consider the sum $\sum_{i=1}^{t} c_i (z_i)_j$, which must be less than or equal to x_j. If it is less than or equal to y_j, then $\mathbf{c}$ is also a solution when $\mathbf{y}$ is the input and we have $\mu(\mathbf{y}) = \mu(\mathbf{x})$. If the sum is greater than y_j, we find a vector $\mathbf{c}'$ as follows. Let $I = \{i \ : \ (z_i)_j = 1\}$. Let $\mathbf{c}'$ be a vector such that $c'_i \geq 0$ for all $1 \leq i \leq t$, $c'_i = c_i$ for $i \notin I$, and $\sum_{i \in I} c'_i = y_j$. The vector $\mathbf{c}'$ satisfies the condition of the problem when $\mathbf{y}$ is the input. Therefore $\mu(\mathbf{x}) - \mu(\mathbf{y}) \leq \sum_{i=1}^{t} c_i - \sum_{i=1}^{t} c'_i = \sum_{i \in I} c_i - \sum_{i \in I} c'_i \leq x_j - y_j$, and the proposition is proved. □

We note that from (ii) above, μ is uniformly continuous.

PROPOSITION 7.3.

$$M_{\text{blkdec}}(q, P) \leq \mu(\mathbf{d}_G(q, P)) \leq M_{\text{blkdec}}(q, P) + t.$$

Proof. Recall that $M_{\text{blkdec}}(q, P)$ is the maximum of the objective function in the integer programming problem above. Since the domain of the variables is more restricted (to integers rather than reals), $M_{\text{blkdec}}(q, P) \leq \mu(\mathbf{d}_G(q, P))$.

Suppose that a vector $\mathbf{c} = (c_i)$ is a solution to the linear programming problem. Then $\mu(\mathbf{d}_G(q, P)) = \sum_{i=1}^{t} c_i$, and $\lfloor \mathbf{c} \rfloor = (\lfloor c_i \rfloor)$ satisfies the condition in the integer programming problem. Thus

$$M_{\text{blkdec}}(q, P) \geq \sum_{i=1}^{t} \lfloor c_i \rfloor \geq \sum_{i=1}^{t} c_i - t = \mu(\mathbf{d}_G(q, P)) - t. \qquad \square$$

THEOREM 7.4. *For a primitive constrained system,*

$$M_{\text{blkdec}}^{\infty}(P) = \mu(\mathbf{d}_G^{\infty}(P)).$$

Moreover,

(i) $\mathcal{P}_{\text{blkdec}}(q) \subseteq \mathcal{P}_{\text{blkdec}}^{\infty}$ *for sufficiently large* q,

(ii) $M_{\text{blkdec}}^{*} = \max_{P \subseteq V_G} \mu(\mathbf{d}_G^{\infty}(P))$.

Proof.

$$\begin{aligned} \lim_{q \to \infty} \frac{\mu(\mathbf{d}_G(q, P))}{\lambda^q} &= \lim_{q \to \infty} \mu\left(\frac{\mathbf{d}_G(q, P)}{\lambda^q}\right) \qquad \text{(by (i) of Proposition 7.2)} \\ &= \mu(\mathbf{d}_G^{\infty}(P)) \qquad \text{(since } \mu \text{ is continuous).} \end{aligned}$$

From Proposition 7.3, $M_{\text{blkdec}}(q, P)/\lambda^q$ also converges to the same limit. Then (i) and (ii) follow from Proposition 4.1. □

Next we give a bound on q similar to Theorems 5.2 and 6.7. Recall that $t(k)$ is the number of minimal covers of a set of size k. Define

$$\rho(G, q) = (2^{|V_G|} - 1) \sum_{U,V} \left| \left(\frac{\bar{A}^q}{\lambda^q}\right)_{U,V} - \bar{\Lambda}_{U,V} \right| + \frac{t(|V_G|)}{\lambda^q}.$$

Note that $\lim_{q \to \infty} \rho(G, q) = 0$ because $\frac{\bar{A}^q}{\lambda^q}$ converges to $\bar{\Lambda}$.

THEOREM 7.5. *If q satisfies $\rho(G,q) < \epsilon_{\text{blkdec}}/2$, then $\mathcal{P}_{\text{blkdec}}(q) \subseteq \mathcal{P}^{\infty}_{\text{blkdec}}$.*

Proof. First observe that $\Delta(\bar{A}^q, W, P)$ can be written as $\sum_{U,V} a_{U,V} \bar{A}^q_{U,V}$, where $a_{U,V} \in \{0, 1, -1\}$. Thus for any W,

$$\frac{d_G(q,P)_W}{\lambda^q} - d^{\infty}_G(P)_W = \sum_{U,V} a_{U,V} \left(\left(\frac{\bar{A}^q}{\lambda^q} \right)_{U,V} - \bar{\Lambda}_{U,V} \right)$$
$$\leq \sum_{U,V} \left| \left(\frac{\bar{A}^q}{\lambda^q} \right)_{U,V} - \bar{\Lambda}_{U,V} \right|.$$

Therefore

$$\begin{aligned}
&|M^q_{\text{blkdec}}(P) - M^{\infty}_{\text{blkdec}}(P)| \\
&\leq \left| \frac{\mu(\mathbf{d}_G(q,P))}{\lambda^q} - \mu(\mathbf{d}^{\infty}_G(P)) \right| + \left| \frac{M_{\text{blkdec}}(q,P)}{\lambda^q} - \frac{\mu(\mathbf{d}_G(q,P))}{\lambda^q} \right| \\
&\leq \left| \mu\left(\frac{\mathbf{d}_G(q,P)}{\lambda^q} \right) - \mu(\mathbf{d}^{\infty}_G(P)) \right| + \frac{t(|P|)}{\lambda^q} \\
&\leq \left\| \frac{\mathbf{d}_G(q,P)}{\lambda^q} - \mathbf{d}^{\infty}_G(P) \right\|_1 + \frac{t(|P|)}{\lambda^q} \\
&\leq (2^{|P|} - 1) \sum_{U,V} \left| \left(\frac{\bar{A}^q}{\lambda^q} \right)_{U,V} - \bar{\Lambda}_{U,V} \right| + \frac{t(|P|)}{\lambda^q} \\
&\leq (2^{|V_G|} - 1) \sum_{U,V} \left| \left(\frac{\bar{A}^q}{\lambda^q} \right)_{U,V} - \bar{\Lambda}_{U,V} \right| + \frac{t(|V_G|)}{\lambda^q} \\
&= \rho(G,q) < \frac{\epsilon_{\text{blkdec}}}{2}.
\end{aligned}$$

Then the theorem follows from Lemma 4.2. ∎

With the technique described in this section, we can compute upper and lower bounds for $M_{\text{blkdec}}(q,P)$. The upper bound comes from the relaxed linear programming problem. The lower bound is obtained by "rounding down" the solution of the linear programming problem. Thus by checking all sets of principal states, we can obtain upper and lower bounds for $M_{\text{blkdec}}(q)$. In fact, from Proposition 3.1, it is sufficient to check all complete sets. Given a deterministic graph G and an integer n, Marcus, Siegel, and Wolf [11] gave an algorithm to find all complete sets P such that $M_{\text{det}}(1,P) \geq n$. Therefore we can design a block-decodable encoder as follows.

Given a deterministic presentation G of the desired constraint S and a block length q, find an optimal set of principal states P_{det} for a deterministic encoder. Then compute the upper and lower bounds for $M_{\text{blkdec}}(q, P_{\text{det}})$; set n to be the lower bound. For each complete set P such that $M_{\text{det}}(q,P) \geq n$, compute the upper and lower bounds for $M_{\text{blkdec}}(q,P)$. Set the upper and lower bounds for $M_{\text{blkdec}}(q)$ to be the maximum of the upper bounds and the maximum of the lower bounds for $M_{\text{blkdec}}(q,P)$, respectively. In this way, we also have the candidates for the optimal sets of principal states.

Example 7.6. By solving the linear programming problem described in this section, we find the asymptotically optimal set of principal states for the asymmetric-RLL$(2,5,1,3)$ to be $P^*_{\text{blkdec}} = \{1,2,3,\bar{1},\bar{2}\}$ and $M^*_{\text{blkdec}} = 0.7076$. Moreover, $\epsilon_{\text{blkdec}} = 0.0146$.

Next we apply Theorem 7.5 to compute the bound on q such that P^*_{blkdec} achieves $M_{\text{blkdec}}(q)$. Let $\bar{\mathbf{r}}_i$ and $\bar{\mathbf{l}}_i$, $1 \le i \le 8$, be the right and left eigenvectors corresponding to λ_i. From [14, Sequence A046165], $t(8) = 3731508$. Thus

$$\rho(G,q) = (2^{|V_G|}-1)\sum_{U,V}\left|\left(\frac{\bar{A}^q}{\lambda^q}\right)_{U,V} - \bar{\Lambda}_{U,V}\right| + \frac{t(|V_G|)}{\lambda^q}$$

$$= (255)\frac{1}{\lambda^q}\sum_{U,V}\left|\sum_{i=2}^{8}\lambda_i^q(\bar{r}_i\bar{l}_i)_{U,V}\right| + \frac{3731508}{\lambda^q}$$

$$\le (255)\frac{1}{\lambda^q}\sum_{U,V}\sum_{i=2}^{8}|\lambda_i|^q|(\bar{r}_i\bar{l}_i)_{U,V}| + \frac{3731508}{\lambda^q}.$$

This expression is decreasing with q. If $q \ge 44$, then $\rho(G,q) \le 0.0050 < \epsilon_{\text{blkdec}}/2 = 0.0073$.

In our construction of the integer programming problem described in this section, the number of variables depends only on $|P|$. But $\mathbf{d}_G(q,P)$ usually contains many zeros and thus many minimal covers of P are not necessary. Thus we can formulate an equivalent problem with a much smaller number of variables, and so the bound on q given in Theorem 7.5 can be very weak. This is especially true when the constraint has a lot of structure (e.g., when many follower sets can be ordered by inclusion). For this example, after neglecting the unnecessary minimal covers, the maximum number of variables is 14 when $P = \{1,2,3,4,\bar{1},\bar{2}\}$ while $t(8) = 3731508$.

Finally we compute bounds on $M_{\text{blkdec}}(q)$, $1 \le q \le 43$, as well as candidates for the achieving P. We find that P^*_{blkdec} is the only optimal set of principal states for $12 \le q \le 43$. From this computation and the bound on q explained above, we conclude that P^*_{blkdec} is the only optimal set of principal states for $q \ge 12$.

8. Complexity of block-type-decodability. We have studied some algorithms to design block-type-decodable encoders, and the reader may have noticed that finding an optimal deterministic encoder is easier than finding an optimal block encoder, which in turn is easier than finding an optimal block-decodable encoder. In this section, we study the complexity of these problems and show that, in some aspects, this observation is indeed the case.

Let S be a constrained system with a deterministic presentation G and let n be a positive integer. For each class $\mathcal{C}$ of encoders, we consider three slightly different problems.

1. *Subgraph encoder:* We study the complexity of determining whether there exists an (S,n) encoder in class $\mathcal{C}$ which is a subgraph of G. In this case, we aim to answer whether $M_{\mathcal{C}}(1) \ge n$. This is the most general and possibly the most important problem.

2. *Fully supported subgraph encoder:* We consider the same problem but require that the set of principal states be V_G. This case can be viewed as a subproblem of the first problem: we fix a set of principal states P and wish to determine whether $M_{\mathcal{C}}(1,P) \ge n$. We will see that this case distinguishes the complexity of computing block and block-decodable encoders.

3. $|V_G|$ *fixed:* In a practical encoder design, we usually fix the constraint and let the block length q vary; thus we study the first problem but consider the number of states of G to be fixed.

We remark that our goal is to compute $M_{\mathcal{C}}(1)$, but the complexity of this problem is equivalent to the complexity of determining whether $M_{\mathcal{C}}(1) \geq n$. We consider the latter problem because it is a decision problem, and hence its complexity class is easier to determine.

We begin with the problem of determining whether there exists an (S, n) deterministic encoder which is a subgraph of G. This problem can be solved by applying the Franaszek algorithm [4] to the adjacency matrix A of G and the all-ones vector $\mathbf{x}$. The algorithm proceeds by iteratively computing $\mathbf{x} \leftarrow \min\left\{\left\lfloor \frac{1}{n} A\mathbf{x} \right\rfloor, \mathbf{x}\right\}$ (taken componentwise) until it converges. If $M_{\det}(1) < n$, the algorithm returns a zero vector; if $M_{\det}(1) \geq n$, the algorithm returns the characteristic vector of the largest set of principal states P such that $M_{\det}(1, P) \geq n$. It is easy to see that the algorithm terminates in at most $|V_G|$ iterations; in each iteration, the running time is polynomial. Thus, for the class of deterministic encoders, this problem is solvable in polynomial time.

From this result, it follows that the other two easier problems on deterministic encoders are also solvable in polynomial time. For the fully supported subgraph encoder problem, there exists an (S, n) deterministic encoder with the set of principal states V_G if and only if the Franaszek algorithm terminates in one iteration and returns the all-ones vector.

Next, we consider the class of block encoders. We will show that the problem of determining whether there exists an (S, n) block encoder is NP-complete by relating it to the well-known clique problem. We first describe the clique problem. A *k-clique* in an undirected graph is a subgraph with k nodes such that there is an edge between every two nodes in the clique. The clique problem is to determine whether a graph contains a clique of a specified size. This problem is known to be NP-complete [13].

THEOREM 8.1. *Given a labeled graph G and an integer n, the problem of determining whether there exists an $(S(G), n)$ block encoder which is a subgraph of G is NP-complete. However, the problem becomes polynomial for every fixed n.*

Proof. Given a graph G' with output labeling and input tagging, it can be verified in polynomial time whether (i) G' is a subgraph of G and (ii) G' is an (S, n) block encoder. Therefore this problem is in NP. What remains is to show that the clique problem is polynomial-time reducible to this problem. Given an undirected graph $H = (V_H, E_H)$, we construct a labeled directed graph G as follows. Let $V_G = V_H$ and assign an edge from state u to state v labeled by v if H has an edge between u and v. Moreover, assign a self-loop to every state v labeled by v. One can show that H has an n-clique if and only if there exists an (S, n) block encoder which is a subgraph of G. Hence we conclude that the block encoder problem is NP-complete.

Suppose that n is fixed; we will show that the problem becomes polynomial. We choose n words from the set of all words and determine whether they can be concatenated with each other. If so, we have an (S, n) block encoder. If not, choose another set of n words. Since there are polynomially many ways to choose n words, we conclude that the problem is polynomial. ∎

If we require that the set of principal states of our block encoder be V_G, this problem becomes polynomial. To see this, consider the following polynomial-time algorithm.

COMPUTATION OF $M_{\text{blk}}(1, V_G)$
Input: G with $V_G = \{v_1, \ldots, v_m\}$
$\mathcal{L} \leftarrow \mathcal{F}_G^1(v_1)$
for each $2 \le i \le m$
  for each $w \in \mathcal{L}$
    if $w \notin \mathcal{F}_G^1(v_i)$
      then $\mathcal{L} \leftarrow \mathcal{L} \setminus \{w\}$
Output: $|\mathcal{L}|$

For the third case where the number of states of G is fixed, the block encoder problem becomes polynomial because we can adapt the above algorithm for each set of principal states, and there is a fixed number of sets of principal states, namely, $2^{|V_G|} - 1$.

Finally, we turn to the block-decodability problem. The complexity of this problem has been studied by Ashley, Karabed, and Siegel [1]; the following theorem is a special case of [1, Theorem 5.4].

THEOREM 8.2 (see [1]). *Given a labeled graph G and an integer n, the problem of determining whether there exists an $(S(G), n)$ block-decodable encoder $\mathcal{E}$ which is a subgraph of G and $V_{\mathcal{E}} = V_G$ is NP-complete. It is also NP-complete for fixed $n \ge 2$.*

Thus the fully supported subgraph encoder problem for the block-decodable encoder is NP-complete. We will show that the subgraph encoder problem is also NP-complete by relating it to the fully supported subgraph encoder problem.

THEOREM 8.3. *Given a labeled graph G and an integer n, the problem of determining whether there exists an $(S(G), n)$ block-decodable encoder which is a subgraph of G is NP-complete. It is also NP-complete for fixed $n \ge 2$.*

Proof. This problem is easily seen to be in NP. We will show that the fully supported subgraph encoder problem is polynomial-time reducible to this more general problem. Given a graph H with $V_H = \{v_1, \ldots, v_m\}$, we construct another graph G as follows. Let $V_G = V_H$, and for each outgoing edge from v_i in H, we assign an edge in G from v_i to v_{i+1} with the same label. (If $i = m$, we assign an edge from v_m to v_1.) Clearly, if there is an $(S(H), n)$ block-decodable encoder $\mathcal{E}$ which is a subgraph of H and $V_{\mathcal{E}} = V_H$, then there is an $(S(G), n)$ block-decodable encoder which is a subgraph of G. On the other hand, if there is an $(S(G), n)$ block-decodable encoder which is a subgraph of G, this encoder must have the same set of states as G. By using the corresponding edges in H and the same tag assignment, we obtain an $(S(H), n)$ block-decodable encoder $\mathcal{E}$ which is a subgraph of H and $V_{\mathcal{E}} = V_H$. ∎

From Theorems 8.2 and 8.3, the block-decodability problem is generally intractable. However, if we fix the number of states but let only the number of edges and the size of the alphabet grow, then the problem becomes more tractable.

THEOREM 8.4. *Given a constrained system S with a deterministic presentation G and an integer n, the problem of determining whether there exists an (S, n) block-decodable encoder is solvable in polynomial time if we fix the number of states of G.*

Proof. First we fix a set of principal states P. It is sufficient to show that the problem of determining whether $M_{\text{blkdec}}(1, P) \ge n$ is solvable in polynomial time. This is because the number of sets of principal states is fixed ($= 2^{|V_G|} - 1$).

In section 7, we showed that the computation of $M_{\text{blkdec}}(1, P)$ is equivalent to an integer programming problem. The worst-case number of variables of this problem (the largest t) depends only on $|P|$. Hence, we can consider the number of variables to be fixed.

Case 1. $n > |E_G|$. Clearly, we can conclude that $M_{\text{blkdec}}(1, P) < n$.

Case 2. $n \le |E_G|$. If we check the feasibility of all $\mathbf{c}$ such that $0 \le c_i \le n$, we can determine whether $M_{\text{blkdec}}(1, P) \ge n$. Since there are $(n+1)^t$ such $\mathbf{c}$, it can be checked in polynomial time. This is because $(n+1)^t \le (|E_G|+1)^t$ and t is fixed. ∎

The complexity of each problem for each class of encoder is summarized in the following table.

TABLE 8.1
Complexity of block-type-decodability problems.

Encoder class	Subgraph encoder	Fully supported subgraph encoder	$\|V_G\|$ fixed
Deterministic	polynomial	polynomial	polynomial
Block	NP-complete (polynomial for any fixed n)	polynomial	polynomial
Block-decodable	NP-complete for fixed $n \ge 2$	NP-complete for fixed $n \ge 2$	polynomial

Acknowledgment. The authors are grateful to the anonymous referees for various comments and suggestions, particularly on the subject of minimal covers of finite sets which greatly improves section 7.

REFERENCES

[1] J. J. Ashley, R. Karabed, and P. H. Siegel, *Complexity and sliding-block decodability*, IEEE Trans. Inform. Theory, 42 (1996), pp. 1925–1947.

[2] J. Ashley and B. Marcus, *Canonical encoders for sliding block decoders*, SIAM J. Discrete Math., 8 (1995), pp. 555–605.

[3] P. Chaichanavong and B. H. Marcus, *Optimal block-type-decodable encoders for constrained systems*, IEEE Trans. Inform. Theory, 49 (2003), pp. 1231–1250.

[4] P. A. Franaszek, *Sequence-state coding for digital transmission*, Bell System Tech. J., 47 (1968), pp. 143–155.

[5] C. V. Freiman and A. D. Wyner, *Optimum block codes for noiseless input restricted channels*, Information and Control, 7 (1964), pp. 398–415.

[6] J. Gu and T. E. Fuja, *A new approach to constructing optimal block codes for runlength-limited channels*, IEEE Trans. Inform. Theory, 40 (1994), pp. 774–785.

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

[8] P. Lee and J. K. Wolf, *A general error-correcting code construction for runlength limited binary channels*, IEEE Trans. Inform. Theory, 35 (1989), pp. 1330–1335.

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

[10] B. H. Marcus, R. M. Roth, and P. H. Siegel, *Constrained systems and coding for recording channels*, in Handbook of Coding Theory, Vols. I and II, North–Holland, Amsterdam, 1998, pp. 1635–1764.

[11] B. H. Marcus, P. H. Siegel, and J. K. Wolf, *Finite-state modulation codes for data storage*, IEEE J. Select. Areas Commun., 10 (1992), pp. 5–37.

[12] E. Seneta, *Nonnegative Matrices and Markov Chains*, 2nd ed., Springer Ser. Statist., Springer-Verlag, New York, 1981.

[13] M. Sipser, *Introduction to the Theory of Computation*, PWS, Boston, 1997.

[14] N. J. A. Sloane, *The On-Line Encyclopedia of Integer Sequences*, http://www.research.att.com/~njas/sequences (22 April 2005).

[15] E. W. Weisstein, *Minimal Cover*, from MathWorld—A Wolfram Web Resource, http://mathworld.wolfram.com/MinimalCover.html (2005).