

the partition

$$\left\{ D_1, \dots, D_M, Y^N \setminus \bigcup_{i=1}^M D_i \right\}$$

and $\epsilon/(2M)$ in place of ϵ , there exists a partition

$$\left\{ D'_1, \dots, D'_M, Y^N \setminus \bigcup_{i=1}^M D'_i \right\}$$

such that $D'_i \in \mathcal{F}$ for $i = 1, \dots, M$ and

$$P^{c_i}(D_j \Delta D'_j) < \epsilon/2 \quad \text{for } i = 1, \dots, M \quad \text{and } j = 1, \dots, M.$$

This implies

$$P^{c_i}(D'_i)^c < \delta + \epsilon/2.$$

In other words, the code $\{(c_1, D'_1), \dots, (c_M, D'_M)\}$ has maximum probability less than $\delta + \epsilon/2$ when used for the limiting channel.

Denote by X' the set of all $x \in X$ that occur in at least one of the $c_i, i = 1, \dots, M$, and let Γ' be the set of all channels with input X' and output \mathcal{Y} with topology $\tau_{X'}$. By definition of the product topology, the projection from Γ to Γ' with respective topologies τ_X and $\tau_{X'}$ is a continuous function. Furthermore, if $(C'_n)_{n=1}^\infty$ is a sequence of channels with input X' $\tau_{X'}$ -converging to the channel $C' = (P^x)_{x \in X'}$, the maximum probability of error δ_n of the code $\{(c_1, D'_1), \dots, (c_M, D'_M)\}$ converges to that of the same code for the channel C . This holds because the decision regions D'_1, \dots, D'_M are P^{c_i} -continuous by construction and the product measures $P_n^{c_i}$ corresponding to channel n converge to $P^{c_i}, i = 1, \dots, M$ [2, Theorem 3.2]. Thus the maximum probability of error of the code $\{(c_1, D'_1), \dots, (c_M, D'_M)\}$ is a continuous function from Γ' to the real numbers at $C' \in \Gamma'$ and the result follows. \square

One notes that only the decision sets are modified, not the code symbols. Hence if the original code satisfies a constraint so does the modified one. This theorem combined with the coding theorem yields

Corollary 4: Suppose C is a channel with capacity C . Then for any rate $R < C$ and $\epsilon > 0$ there exists an N and a code $\{(c_1, D_1), \dots, (c_M, D_M)\}$ of length N with maximum probability of error less than ϵ in a sufficiently small neighborhood of C where $R \leq (1/N) \log M$. The code satisfies the constraint.

Although this corollary guarantees the existence of good codes in a sufficiently small neighborhood of the nominal channel C it does not directly contain a proof of Theorem 1 because the neighborhoods in the corollary are allowed to shrink with the error probability ϵ .

IV. REMARKS

Corollary 4 can be seen as a justification of the folk rule "if the channel is approximately Gaussian, take the Gaussian capacity formula." However, the example in the Introduction and Theorem 1 show that the reasoning "because channel capacity is continuous" does not hold.

The approach adopted here is infinitesimal, i.e., for any rate less than capacity of a nominal channel, there are codes which perform well for all channels sufficiently close to the nominal channel. It complements the minimax approach implicit in the definition of capacity for arbitrarily varying or compound channels (for any rate less than capacity there are codes which perform well even in the worst case) [3, Def. 5.9, Def. 6.2], [4].

The idea of modifying the decision regions of a given code in order to make it well-behaved in a specified sense, which is the key to the

proof of Theorem 3, also appears in the context of the coding theorem for \bar{d} -continuous channels [6]. There, a given δ -robust block code is modified by slightly enlarging the decision regions in Hamming space in such a manner that the resulting regions are still disjoint.

Although it is possible to define the convergence of channels with memory by weak convergence of the associated channel probability measures on sequence space, only memoryless channels are considered in this correspondence. A direct extension of the presented results to general channels with memory does not appear feasible.

ACKNOWLEDGMENT

The author wishes to thank H. Vinck and J. Sadowsky for fruitful discussions.

REFERENCES

- [1] P. Billingsley, *Ergodic Theory and Information*. New York: Wiley, 1965.
- [2] —, *Convergence of Probability Measures*. New York: Wiley, 1968.
- [3] I. Csiszár and J. Körner, *Information Theory: Coding Theorems for Discrete Memoryless Systems*. Orlando, FL: Academic Press, 1981.
- [4] I. Csiszár, "Arbitrarily varying channels with general alphabets and states," *IEEE Trans. Inform. Theory*, vol. 38, no. 6, pp. 1725–1742, Nov. 1992.
- [5] R. G. Gallager, *Information Theory and Reliable Communication*. New York: Wiley, 1968.
- [6] R. M. Gray and D. S. Ornstein, "Block coding for discrete stationary \bar{d} -continuous noisy channels," *IEEE Trans. Inform. Theory*, vol. IT-25, no. 3, pp. 639–667, May 1979.

A Rate 4/6 ($d = 1, k = 11$) Block-Decodable Runlength-Limited Code

Kees A. Schouhamer Immink, *Fellow, IEEE*

Abstract—A new rate 4/6 ($d = 1, k = 11$) runlength-limited code is presented. The new code has the virtue that it can be decoded on a block basis, i.e., without knowledge of previous or next codewords, and, therefore, it does not suffer from error propagation.

Index Terms—Runlength-limited code, (d, k) sequence, recording code.

I. INTRODUCTION

By far the most frequently reported coding schemes applied in recording products have been those constituted by runlength-limited sequences [1]. The length of time usually expressed in channel bits between consecutive transitions is known as the *runlength*. Runlength-limited (RLL) sequences are characterized by two parameters, $(d + 1)$ and $(k + 1)$, which stipulate the minimum and maximum runlength, respectively, that may occur in the sequence. Popular (d, k) codes incorporated in storage products are the $(2, 7)$ and the $(1, 7)$ codes of rate 1/2 and rate 2/3, respectively [2]. These codes are decoded by a sliding-block decoder, and as a result, show some error propagation. For example, a single bit error in a received sequence encoded by the $(2, 7)$ sliding-block code propagates at most

Manuscript received August 31, 1995; revised April 24, 1996.

The author is with Philips Research Laboratories, Eindhoven, The Netherlands.

Publisher Item Identifier S 0018-9448(96)06100-7.

over four decoded bits, while a bit error in the (1, 7) code propagates at most over five decoded bits.

Recently, an alternative to the above sliding-block codes was proposed by Immink [3] and Hollmann [4]. The RLL codes presented have the virtue that they can be decoded by observing single codewords. Clearly, these *block-decodable* codes, as they are called, confine error propagation to one decoded symbol, and they are therefore highly suitable in conjunction with symbol-correcting Reed-Solomon codes.

We will present a new rate 4/6 ($d = 1, k = 11$) RLL block-decodable code which is well adapted to byte-oriented storage systems. This code is particularly attractive as many commercially available Reed-Solomon codes operate in $GF(2^8)$. We will specifically be concerned with the implementation aspects of the new RLL encoder. As the new code takes full advantage of the difference between RLL and (d, k) sequences, we will commence with a brief section explaining these differences.

II. RLL VERSUS (d, k) SEQUENCES

A (d, k) sequence is by definition a binary sequence having at least d and at most k "zeros" between consecutive "ones." In general, the (d, k) sequence is not employed in optical or magnetic recording without a simple coding step. Prior to recording, a (d, k) sequence is converted to a runlength-limited (RLL) channel sequence in the following way. Let the channel signals be represented by a bipolar sequence $\{y_i\}, y_i \in \{-1, 1\}$. The channel signals represent the positive or negative magnetization of the recording medium, or pits or lands when dealing with optical recording. The logical "ones" in the (d, k) sequence indicate the positions of a transition $1 \rightarrow -1$ or $-1 \rightarrow 1$ of the corresponding runlength-limited sequence. The (d, k) sequence

0 1 0 0 0 1 0 0 1 0 0 0 1 1 0 1...

would be converted to the RLL channel sequence

1 -1 -1 -1 -1 1 1 1 -1 -1 -1 -1 1
-1 -1 1...

The mapping of the waveform by this coding step is known as *precoding*. A confusing term since it is in fact a "postcoding process." Sequences that are assumed to be recorded with such a precoding step (for example, (d, k) sequences) are said to be given in nonreturn-to-zero-inverse (NRZI) notation. Waveforms that are transmitted without such an intermediate coding step are referred to as nonreturn-to-zero (NRZ). These nebulous terms stem from telegraphy and have no meaning in relation to recording channels, but are still in common use today. Coding techniques using the NRZI format are generally accepted in digital optical and magnetic recording practice. In fact, to the author's best knowledge, all RLL codes used in recording practice, whether optical or magnetic, are essentially (d, k) codes followed by a precoder step. In everyday speech, the terms (d, k) code and RLL code are usually used as synonyms. The NRZI format offers some advantage in magnetic recording since differentiation occurs as part of the physical process in the read heads. The original (d, k) signal is restored in a quite natural fashion by observing the peaks in the retrieved signal. These peaks coincide with the "ones" of the stored sequence in NRZI notation. The use of the NRZI format in nondifferentiating channels, such as, e.g., the Compact Disc [1], is far from obvious. Code designers generally believed that an RLL code operating without the ubiquitous precoder step would not offer advantages over the state-of-the-art (d, k) codes in conjunction with precoding. The exciting result of the codes presented in [3] and the code presented below is that they show this belief is false. A

TABLE I
CODEBOOK OF A $d = 1, k = \infty$ RLL BLOCK CODE

Source	$b_1 = 0, x_6 = 0$	$b_1 = 1, x_6 = 0$	$b_1 = 0, x_6 = 1$	$b_1 = 1, x_6 = 1$
0	000110	000001	000110	000001
1	001110	011001	001110	100001
2	000000	000000	000000	000000
3	011100	011100	100011	100011
4	000011	000011	000011	000011
5	000111	000111	000111	000111
6	001100	001100	001100	001100
7	001111	001111	001111	001111
8	011111	011111	100000	100000
9	110011	110011	110011	110011
a	110000	110000	110000	110000
b	111000	111000	111000	111000
c	111100	111100	111100	111100
d	011000	011000	100111	100111
e	011110	111001	100110	111001
f	111110	110001	111110	110001

consequence of our finding is that the terms (d, k) code and RLL code cannot be used anymore as synonyms.

III. DESCRIPTION OF THE NEW CODE

Consider the construction of an encoder that converts source (or input) sequences $\{b_i\}$ into constrained (or output) channel sequences $\{x_i\}$, and a decoder which reconstitutes $\{x_i\}$ into the original $\{b_i\}$. The source sequence is partitioned into m -tuples $\{b_i\}$, called *source words* and the output sequence is partitioned into $\{x_i\}$, called *codewords*. The new one-symbol look-ahead RLL code can be encoded by an encoder structured as a standard finite-state automaton. The automaton is defined by the following two characteristic functions:

$$x_t = h(b_t, b_{t+1}, s_t)$$

$$s_{t+1} = g(b_t, b_{t+1}, s_t)$$

where s_t is a member of a finite set S , called the *state set* of the encoder. A block decoder is given by the function

$$b_t = f(x_t).$$

A straightforward implementation of the encoder automaton would require logic arrays of $|S| \times 2^{2m}$ terms.

The code, shown in Table I, exemplifies that much less hardware is required. The code has been modified with respect to the one presented in [3] to simplify the implementation. The sixteen source words are labeled by $\{0, \dots, f\}$. Nine source words, namely those with the labels $\{2, 4, 5, 6, 7, 9, a, b, c\}$ are uniquely represented by a single codeword. It can easily be verified that these codewords can indeed be concatenated without offending the prescribed runlength constraints, that is, solitary "ones" or "zeros" will never occur. The remaining source words are represented by two or three codewords. The choice depends on the last channel bit sent, x_6 and the most significant bit (MSB) of the upcoming source word, b_1 .

The assignment between source words and codewords shown in Table I is the outcome of a judicious process, and it allows a very efficient implementation of the encoder. The encoder function can be succinctly written by the Boolean function

$$x_t = h(b_t, b_{t+1,1}, x_{t-1,6})$$

i.e., the encoding is a function of a) the current 4-bit source word b_t , b) the most significant bit (MSB) of the next source word $b_{t+1,1}$, and c) the last channel bit transmitted $x_{t-1,6}$. Clearly, the encoder function can be readily implemented with a $6 \rightarrow 6$ ROM.

The decoding function can be accomplished with a simple logic array. Note that the code above can easily be transformed into a ($d = 1, k = 11$) RLL code by representing the source word "2" by "000000" or "111111." If $x_5x_6 = 00$, then represent the source word "2" by "000000." It should be appreciated that the smallest block-decodable conventional rate $2/3$ ($d = 1, k = 11$) code requires a codeword length of $n = 18$. This clearly shows that a design of an RLL code which is not a (d, k) code plus precoder can be quite advantageous.

IV. CONCLUSIONS

We have presented a new rate $4/6$ ($d = 1, k = 11$) runlength-limited code. The code is block-decodable, and it is particularly attractive as many commercially available Reed-Solomon codes operate in $GF(2^8)$. The encoder can be implemented with a simple 6-bit ROM, and decoding can be accomplished with a logic array.

REFERENCES

[1] K. A. S. Immink, *Coding Techniques for Digital Recorders*. Englewood Cliffs, NJ: Prentice-Hall Int. (UK) Ltd., 1991.
 [2] T. D. Howell, "Statistical properties of selected recording codes," *IBM J. Res. Develop.*, vol. 33, no. 1, pp. 60-73, Jan. 1989.
 [3] K. A. S. Immink, "Block-decodable runlength-limited codes via look-ahead technique," *Philips J. Res.*, vol. 46, no. 6, pp. 293-310, 1991.
 [4] H. Hollmann, "A block-decodable (d, k) = (1, 8) runlength-limited rate $8/12$ code," *IEEE Trans. Inform. Theory*, vol. 40, no. 4, pp. 1292-1295, July 1994.

Encoding of *dklr*-Sequences Using One Weight Set

Levente Pátrovics and Kees A. Schouhamer Immink, *Fellow, IEEE*

Abstract—Traditional schemes for encoding and decoding runlength-constrained sequences using the enumeration principle require two sets of weighting coefficients. A new enumeration is presented requiring only one set of coefficients.

Index Terms—Enumerative coding, runlength-limited sequences, *dk*-constrained codes.

I. INTRODUCTION

Runlength-limited codes have been applied in magnetic and optical recording as well as optical data transmission. A *dk*-limited (binary) sequence, in short, *dk*-sequence, satisfies simultaneously the following two conditions: 1) *d* constraint—two logical "ones" are separated by a run of consecutive "zeros" of length at least *d*, and 2) *k* constraint—any run of consecutive "zeros" is of length at most *k*. If only Condition 1 is satisfied, the sequence is said to be *d*-limited (with $k = \infty$), and will be termed a *d*-sequence. Similarly, if only Condition 2 is satisfied, the sequence is said to be *k*-limited.

In 1965, Kautz [1] described a method of encoding and decoding *k*-limited codewords by an algebraic approach termed *enumeration*.

Manuscript received November 22, 1995.

The authors are with Philips Research Laboratories, 5656 AA Eindhoven, The Netherlands.

Publisher Item Identifier S 0018-9448(96)05853-1.

Enumerative decoding is done by forming the weighted sum of the symbols of the codeword received. The integer-valued weights used in forming this sum are a function of the the channel constraints in force. Encoding is done by a method which is similar to decimal-to-binary conversion where, instead of the usual powers of two, the weights are used. The coding and decoding using enumeration has the virtue that its complexity grows polynomially with the codeword length contrasting the complexity of direct look-up which grows exponentially. Five years later, in 1970, Tang and Bahl [2] presented a classical method for enumerating *dk*-sequences. Their method requires only one set of weight coefficients for the encoding and decoding of *d*-limited sequences while the encoding operation of the general case of *dk*-sequences requires two sets of coefficients. Tang and Bahl also showed how *dk*-sequences can be cascaded without violating, at the boundaries, the prescribed constraints. A more efficient method for cascading *dk*-sequences was described by Beenker and Immink [5] using *dklr*-sequences. A *dklr*-limited sequence is a *dk*-sequence that starts with at most *l* "zeros" and ends with at most *r* "zeros." Also for enumerating *dklr*-sequences the prior art requires two sets of coefficients.

We will present a new method for enumerating *dklr*-sequences requiring only one set of coefficients. The major part of the electronics that implements the enumeration technique is taken by the storage of the coefficients. As a result, the new method presented will almost halve the electronics needed.

II. CODING AND DECODING USING ENUMERATION

Tang and Bahl [2] developed a general algebraic technique for encoding and decoding *dk*-sequences. They defined a 1-1 mapping from the set *S* of all *dk*-sequences of length *n* onto the set of integers $\{0, 1, \dots, |S| - 1\}$ and they presented an algorithm for converting *dk*-sequences to integers and *vice versa*. Tang and Bahl's method is briefly described below. Let $\{0, 1\}^n$ denote the set of binary sequences of length *n* and let *S* be any subset of $\{0, 1\}^n$. The set *S* can be ordered lexicographically as follows: if $\mathbf{x} = (x_1, \dots, x_n) \in S$ and $\mathbf{y} = (y_1, \dots, y_n) \in S$, then \mathbf{y} is called less than \mathbf{x} , in short, $\mathbf{y} < \mathbf{x}$, if there exists an *i*, $1 \leq i \leq n$, such that $y_i < x_i$ and $x_j = y_j, 1 \leq j < i$. For example, "00101" < "01010." The position of \mathbf{x} in the lexicographical ordering of *S* is defined to be the *rank* of \mathbf{x} denoted by $i_S(\mathbf{x})$, i.e., $i_S(\mathbf{x})$ is the number of all \mathbf{y} in *S* with $\mathbf{y} < \mathbf{x}$. Tang and Bahl showed that in the specific case of *dk*-sequences the rank can be computed by

$$i_S(\mathbf{x}) = \sum_{j=1}^n x_j N(n-j) \tag{1}$$

where $N(n)$ is the number of *dk*-sequences of length *n*, $n > 0$. By definition we have $N(0) = 1$. The above algorithm is usually called the *decoding* operation. In an implementation of this algorithm, the weight coefficients $N(n)$ can be precalculated and stored in memory or they can be calculated "on the fly." In the more specific case of *dklr*-sequences, a similar relationship can be written down. Translation of an integer *I*, $0 \leq I \leq N(n) - 1$, into the corresponding \mathbf{x} , the *encoding operation*, is done by the following algorithm:

```

 $\hat{I} := a + I;$ 
for  $j = 1$  to  $n$  do
  if  $\hat{I} \geq T(n-j)$  then
     $x_j := 1, \hat{I} := \hat{I} - N(n-j)$  else  $x_j := 0$ .The
    
```