[5] J. Spragins, "Learning without a teacher," *IEEE Trans. Information Theory*, vol. IT-12, pp. 223–230, April 1966.
[6] E. Patrick and J. Hancock, "Nonsupervised sequential classification and recognition of patterns," *IEEE Trans. Information Theory*, vol. IT-12, pp. 362–372, July 1966.
[7] D. Stanat, "Unsupervised learning of mixtures of probability functions," in *Pattern Recognition*, L. Kanal, Ed. Washington, D. C.: Thompson, 1966, pp. 357–390.
[8] D. Lainiotis, "A nonlinear adaptive estimation algorithm," *IEEE Trans. Automatic Control* (Correspondence), vol. AC-13, p. 197, April 1968.
[9] D. Cooper and P. Cooper, "Non-supervised adaptive signal detection and pattern recognition," *Inform. and Control*, vol. 7, pp. 416–444, September 1964.
[10] J. Sammon, Jr., "An adaptive technique for multiple signal detection and identification," in *Pattern Recognition*, L. Kanal, Ed. Washington, D. C.: Thompson, 1968, pp. 409–439.
[11] K. S. Fu, "Learning Techniques in pattern recognition systems," in *Pattern Recognition*, L. Kanal, Ed. Washington, D. C.: Thompson, 1968, pp. 399–407.
[12] S. Schwartz, "An example of nonsupervised adaptive pattern classification," *IEEE Trans. Automatic Control*, vol. AC-13, pp. 107–108, February 1968.
[13] G. Saridis, Z. Nikolic, and K. S. Fu, "Stochastic approximation algorithms for system identification, estimation, and decomposition of mixtures," *IEEE Trans. System Science and Cybernetics*, vol. SSC-5, pp. 8–15, February 1969.
[14] E. Patrick, "On a class of unsupervised estimation systems," *IEEE Trans. Information Theory*, vol. IT-14, pp. 407–415, May 1968.
[15] H. Robbins, "The empirical Bayes approach to statistical decisions problems," *Ann. Math. Stat.*, vol. 35, pp. 1–20, February 1964.
[16] H. Teicher, "Identifiability of finite mixtures," *Ann. Math. Stat.*, vol. 34, pp. 1265–1269, December 1963.
[17] ——, "Identifiability of mixtures," *Ann. Math. Stat.*, vol. 32, pp. 244–248, March 1961.
[18] ——, "Identifiability of mixtures of product measures," *Ann. Math. Stat.*, vol. 38, pp. 1300–1302, August 1967.
[19] S. Yakowitz and J. Spragins, "On the identifiability of finite mixtures," *Ann. Math. Stat.*, vol. 39, pp. 209–214, February 1968.
[20] J. Kiefer and J. Wolfowitz, "On the deviations of the empirical distribution function of vector chance variables," *Trans. Am. Math. Soc.*, vol. 87, pp. 173–186, January 1958.
[21] R. Massey, "A note on the estimation of a distribution function by confidence limits," *Ann. Math. Stat.*, vol. 21, pp. 116–119, March 1950.
[22] A. Taylor, *Introduction to Functional Analysis*. New York: Wiley, 1958.
[23] B. Gnedenko and A. Kolmogorov, *Limit Distributions for Sums of Independent Random Variables*. Reading, Mass.: Addison-Wesley, 1954.
[24] H. Rogers, *Theory of Recursive Functions and Effective Computability*. New York: McGraw-Hill, 1967.
[25] E. Patrick and J. Costello, "On unsupervised estimation algorithms," Purdue University, Lafayette, Ind.
[26] K. Abend, "Compound decision procedures for unknown distributions and for dependent states of nature," in *Pattern Recognition*, L. Kanal, Ed. Washington, D. C.: Thompson, 1968.
[27] D. Cooper, "On the existence of nonsupervised adaptive signal detectors," Ph.D. dissertation, Columbia University, New York, N. Y., 1966.
[28] N. Alēns, "Compound Bayes learning without a teacher," Tech. Rept. 6151-2, Stanford University, Stanford, Calif., 1967.

Realization of Optimum Interleavers

JOHN L. RAMSEY, MEMBER, IEEE

Abstract—**Four realizations of interleavers that reorder a sequence of symbols so that no contiguous sequence of n_2 symbols in the reordered sequence contains any pair of symbols that were separated by fewer than n_1 symbols in the original ordering are introduced. For any n_1 and n_2 that satisfy an appropriate relative primeness condition, these interleavers are optimum in the sense that of all possible interleavers providing the indicated symbol separation, one of these four realizations achieves both the minimum possible encoding delay and the minimum possible combined storage capacity for the interleaver and its unscrambler.**

I. Introduction

An interleaver is a device that rearranges the ordering of a sequence of symbols in some one-to-one deterministic manner. Associated with any interleaver is an unscrambler, which is the device that restores the reordered sequence to its original ordering. Interleavers and unscramblers have a variety of applications in cryptography and in communication technology.

In many of the applications to communication technology, interleaving is used as an adjunct to coding for error correction. One technique, which is useful for some types of burst-error channels, is to insert an interleaver between the channel encoder and the channel. The interleaver redistributes the channel symbols so that the symbols from a codeword are mutually separated by somewhat more than the length of a "typical" burst of errors. Thus, interleaving effectively makes the channel appear like a random-error channel to the decoder. For some HF channels, this technique can improve the performance by one to three orders of magnitude [1].

Another technique requires interleavers for the generation of product codes [2]. Long product codes can be easily decoded in stages [2], [3], and they also provide a substantial improvement in performance when compared with practical one-stage codes [4], [5].

Many of the previous interleaver applications have been associated with block codes, so it is understandable that block interleavers have frequently been assumed for these applications. A common example of a block

Manuscript received July 8, 1969; revised November 20, 1969. This work was supported in part by the National Aeronautics and Space Administration under Grant NGL 22-009-013.

The author was previously with Stanford Research Institute, Menlo Park, Calif. He is now with Research Laboratory of Electronics, Massachusetts Institute of Technology, Cambridge, Mass. 02139.

interleaving function is to divide symbol sequences into blocks corresponding to a two-dimensional array, and to conceptually read symbols in by rows and out by columns [1].

In some applications such as those dealing with convolutional codes, it is more natural to consider synchronous interleavers, in which a symbol is read out each time a symbol is read in. Synchronous interleavers are a more general class of interleavers than block interleavers, since any block interleaving function can be realized by a synchronous interleaver.

Interleavers and unscramblers are characterized by their *encoding delay* D, which is the maximum delay encountered by any symbol before it is inserted into the output sequence, and by the *storage capacities* S and S_u, which are the number of symbols stored by the interleaver and by the unscrambler, respectively.

We shall introduce the class of (n_2, n_1) *interleavers*, which are those interleavers that reorder a sequence so that no contiguous sequence of n_2 symbols in the reordered sequence contains any symbols that were separated by fewer than n_1 symbols in the original ordering. Lower bounds are derived for the encoding delay and for the combined storage capacity $S + S_u$ achievable by any (n_2, n_1) interleaver. An (n_2, n_1) interleaver is *optimum* if it achieves both the minimum possible encoding delay and the minimum possible combined storage capacity. Section III presents four simple but similar techniques for realizing synchronous (n_2, n_1) interleavers. For any n_1 and n_2 satisfying certain relative primeness conditions, one of these techniques achieves the minimum possible encoding delay. Section V describes reduced-storage versions of these interleavers that also achieve the minimum possible combined storage capacity, and which therefore are optimum. Our results are similar to earlier unpublished work by Forney.[1]

II. Preliminaries

Let $\cdots, a_{z_1}, a_{z_2}, \cdots$ be the sequence of symbols in the output sequence, where $\cdots, z_1, z_2, \cdots$ are the positions of these symbols in the original ordering. For an (n_2, n_1) interleaver, therefore,

$$|z_i - z_j| \geq n_1 \tag{1a}$$

whenever

$$|i - j| \leq n_2 - 1. \tag{1b}$$

In the rest of this paper it will be useful to recognize the fact that the unscrambler for an (n_2, n_1) interleaver is itself an (n_1, n_2) interleaver. This assertion can be verified through the following arguments. Again, let $\cdots, a_{z_1}, a_{z_2}, \cdots$ be the sequence of symbols in the output of the interleaver, which is also the input to the unscrambler, where $\cdots, z_1, z_2, \cdots$ are the positions of these symbols in the original ordering. Let $\cdots, z_1', z_2', \cdots$ be the positions of these symbols in the output of the unscrambler. Since the unscrambler restores the sequence to its original ordering,

$$z_i' = z_i + D' \qquad \text{all } i,$$

where D' is a fixed delay introduced by the interleaving–unscrambling process. Thus (1a) and (1b) continue to apply to z_i' and z_j'; that is,

$$|z_i' - z_j'| \geq n_1 \tag{2a}$$

whenever

$$|i - j| \leq n_2 - 1. \tag{2b}$$

But (2a) and (2b) imply that if

$$|z_i' - z_j'| \leq n_1 - 1, \tag{3a}$$

then

$$|i - j| \geq n_2. \tag{3b}$$

This completes the verification, since (3a) and (3b) define an (n_1, n_2) interleaver.

The encoding delay is defined as

$$D = \sup_i (j - z_i),$$

where $j \geq z_i$, because the interleaver is assumed to be physically realizable. It is assumed that

$$d = \inf_i (j - z_i) = 0,$$

since D could be reduced by d if $d > 0$. It follows that the delay introduced by the combined interleaving and unscrambling operations is also D, which can be seen if

$$z_i' = z_i + D.$$

If D_u is the encoding delay of the unscrambler, then

$$D_u = \sup_i (z_i' - j) = D.$$

Thus, the encoding delays of the interleaver and the unscrambler are both equal to the delay introduced by the overall interleaving–unscrambling operation.

An (n_2, n_1) interleaver is said to be *uniform* if the members of every set of n_2 contiguous symbols in the output sequence are mutually separated by at least n_1 symbols in the input sequence, but there is no set of $n_2 + 1$ or more contiguous symbols in the output sequence in which the members are mutually separated by at least n_1 symbols in the input sequence. Clearly, an (n_2, n_1) interleaver is either uniform or nonuniform; if it is nonuniform, then the members of some set of $n_2 + 1$ or more contiguous symbols in the output sequence are mutually separated by at least n_1 symbols in the input sequence.

III. Basic Interleaving Techniques

We now describe four basic methods of using a commutator and a tapped shift register to realize an (n_2, n_1) interleaver. Subsequently, we shall show that at each point in the (n_2, n_1) plane satisfying a relative primeness condition, a modification of one of these methods realizes an optimum (n_2, n_1) interleaver.

[1] G. D. Forney, Jr., private communication.

Type I (n_2, n_1) Interleaver

Whenever n_1 and $n_2 + 1$ are relatively prime and $n_1 > n_2 + 1$, the device shown in Fig. 1 is a nonuniform (n_2, n_1) interleaver. That device consists of an $[n_2(n_1 - 1) + 1]$-stage shift register with taps at the outermost stages and at every $(n_1 - 1)$th intermediate stage, and an $(n_2 + 1)$-position commutator that cyclically samples the $n_2 + 1$ taps in reverse order of their distances from the input of the shift register. Observe that the encoding delay of this device is $n_2(n_1 - 1)$.

Two assertions must be verified to prove that the device is an (n_2, n_1) interleaver: 1) no contiguous sequence of n_2 output symbols contains any symbols that were separated by fewer than n_1 symbols in the input sequence; and 2) each symbol in the input sequence eventually appears in the output sequence. Assertion 1) ensures that the device performs the required symbol separation, while assertion 2) is required to show that the device provides a one-to-one mapping of the input sequence into the output sequence.

Assertion 1): Suppose that symbols a_k through $a_{k+n_2(n_1-1)}$ are stored in order in shift-register stages 0 through $n_2(n_1 - 1)$ when the commutator is at position 0. The device proceeds as follows. Symbol a_k is read out, a new symbol is shifted into the shift register, the commutator is advanced to position 1, symbol a_{k+n_1} is read out, and so on. The ordering of symbols in the output sequence is, therefore,

$$a_k, a_{k+n_1}, a_{k+2n_1}, \cdots, a_{k+jn_1}, \cdots, a_{k+n_2n_1},$$

$$a_{k+n_2+1}, a_{k+n_2+n_1+1}, \cdots, a_{k+n_2+jn_1+1}, \cdots.$$

We must show that each set of n_2 contiguous output symbols has the required separation. Certainly each set starting with symbols a_k or a_{k+n_1} has the required separation. Consider now the set of n_2 contiguous symbols in the output sequence starting with a_{k+jn_1} and ending with $a_{k+n_2+(j-2)n_1+1}$, $2 \leq j \leq n_2$. This set can be divided into two subsets, one of which contains symbols a_{k+jn_1} through $a_{k+n_2n_1}$, and the other symbols a_{k+n_2+1} through $a_{k+n_2+(j-2)n_1+1}$. Each subset obviously has the required separation. The lowest index in the first subset is $k + jn_1$, while the highest index in the second subset is $k + n_2 + (j - 2)n_1 + 1$. If

$$(k + jn_1) - [k + n_2 + (j - 2)n_1 + 1] = 2n_1 - n_2 - 1 \geq n_1,$$

or equivalently if $n_1 \geq n_2 + 1$, then the entire set has the required separation, since no symbol from one subset was within n_1 symbols of any symbol from the other subset in the original ordering. If $n_1 \leq n_2$, however, the entire set does not have the required separation because the symbol with index $k + n_2 + (j - 2)n_1 + 1$ must have been within n_1 symbols of some symbol from the first subset in the original ordering. This completes the proof of assertion 1).

Assertion 2): We must show that each input symbol appears somewhere in the output sequence whenever n_1 and $n_2 + 1$ are relatively prime. Let the commutator be at an arbitrary position j when the symbol a_0 is first shifted into the shift register. The symbol a_0 appears at tap k after $(n_2 - k)(n_1 - 1)$ shifts, and it is read out then if and only if the position of the commutator is also k. But the position of the commutator after $(n_2 - k)(n_1 - 1)$ shifts is

$$[j + (n_2 - k)(n_1 - 1)] \bmod (n_2 + 1) = [j - (k + 1)(n_1 - 1)] \bmod (n_2 + 1).$$

Therefore the required condition for a_0 to be read out at tap k is that

$$[j - (k + 1)(n_1 - 1) - k] \bmod (n_2 + 1) = [j - (k + 1)n_1 + 1] \bmod (n_2 + 1) = 0,$$

or equivalently that

$$(kn_1) \bmod (n_2 + 1) = \alpha, \tag{4}$$

where $\alpha = (j - n_1 + 1) \bmod (n_2 + 1)$.

If n_1 and $n_2 + 1$ are relatively prime, then (4) is satisfied for one and only one value of k in the range $0 \leq k \leq n_2$, so that an arbitrary symbol a_0 appears once and only once in the output sequence. This establishes assertion 2) and verifies that under the given conditions the device shown in Fig. 1 is indeed an (n_2, n_1) interleaver.

The device shown in Fig. 2 is a simple realization of an unscrambler for the interleaver shown in Fig. 1. By comparing Figs. 1 and 2, the reader should be able to verify that this device restores the original ordering of the sequence of symbols.

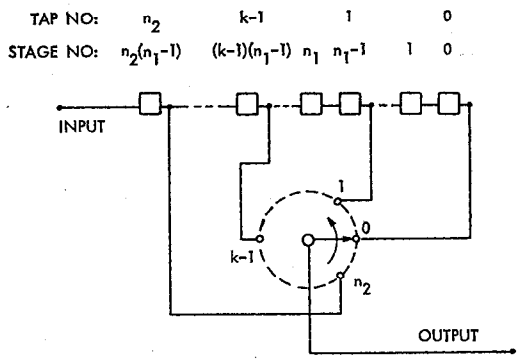

Fig. 1. Type I (n_2, n_1) interleaver; $n_1 > n_2 + 1$.

Type II (n_2, n_1) Interleaver

Recall that the unscrambling device for an (n_2, n_1) interleaver is an (n_1, n_2) interleaver. Using this fact, we see that whenever n_2 and $n_1 + 1$ are relatively prime, and $n_2 > n_1 + 1$, an (n_2, n_1) interleaver can be realized by a device consisting of an $[n_1(n_2 - 1) + 1]$-stage shift register with taps at the outermost stages and at every $(n_2 - 1)$th intermediate stage, and an $(n_1 + 1)$-position commutator that cyclically inserts input symbols into the $n_1 + 1$ taps in reverse order of their distances from the input of the shift register. The configuration for

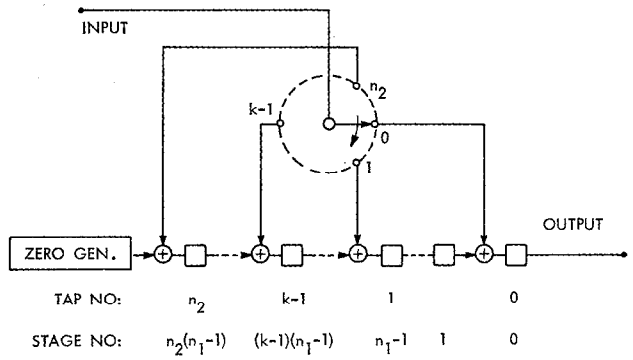

Fig. 2. Unscrambler for the interleaver of Fig. 1.

this device is shown in Fig. 2, with the parameters n_1 and n_2 interchanged. The corresponding unscrambler can be realized by the device shown in Fig. 1, again with the parameters n_1 and n_2 interchanged.

Type III (n_1, n_2) *Interleaver*

Whenever n_1 and n_2 are relatively prime, the device shown in Fig. 3 is an (n_1, n_2) interleaver. That device consists of an $[(n_2 - 1)(n_1 + 1)+1]$-stage shift register with taps at the outermost stages and at every $(n_1 + 1)$th intermediate stage, and an n_2-position commutator that cyclically samples the n_2 taps in the same order as their distances from the input of the shift register. The encoding delay of this device is therefore $(n_2 - 1)(n_1 + 1)$.

A verification that the device shown in Fig. 3 is indeed an (n_2, n_1) interleaver whenever n_1 and n_2 are relatively prime can be given in a manner similar to that given for the Type I interleaver. We shall omit doing so here. Observe that the ordering of symbols in the output sequence is $\cdots$, a_k, a_{k-n_1}, a_{k-2n_1}, $\cdots$, $a_{k-(n_2-1)n_1}$, a_{k+n_2}, $a_{k+n_2-n_1}$, $\cdots$, so that whenever $n_1 > n_2$, the device shown in Fig. 3 is a uniform (n_2, n_1) interleaver.

A simple realization of an unscrambler for the interleaver of Fig. 3 is given by the device shown in Fig. 4.

Type IV (n_1, n_2) *Interleaver*

Whenever n_1 and n_2 are relatively prime, an (n_2, n_1) interleaver can be realized by a device consisting of an $[(n_1 - 1)(n_2 + 1)+1]$-stage shift register with taps at the outermost stages and at every $(n_2 + 1)$th intermediate stage, and an n_1-position commutator that cyclically inserts input symbols into the n_1 taps in the same order as their distances from the input of the shift register. The configuration for this device is shown in Fig. 4, with the parameters n_1 and n_2 interchanged. The corresponding unscrambler can be realized by the device shown in Fig. 3, with the parameters n_1 and n_2 again interchanged.

IV. Optimality of Encoding Delay

We shall now show that one of the interleavers of Types I–IV achieves the minimum possible encoding delay for any (n_2, n_1) interleaver, provided that the appropriate relative primeness conditions between n_1 and n_2 are satisfied. We first demonstrate that whenever

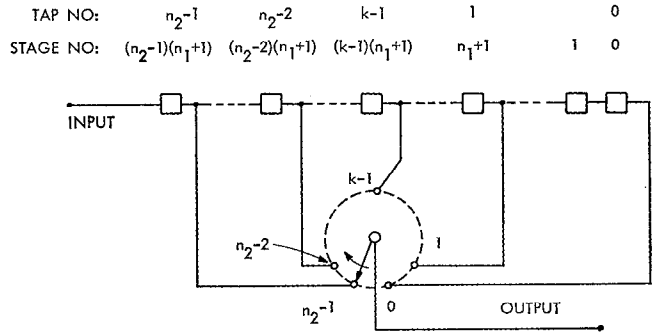

Fig. 3. Type III (n_2, n_1) interleaver.

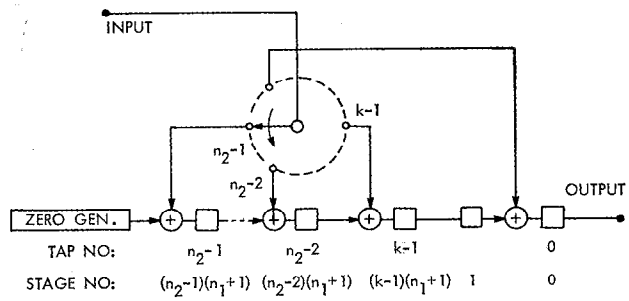

Fig. 4. Unscrambler for the interleaver of Fig. 3.

$n_1 > n_2$, either a Type I interleaver or a Type III interleaver achieves the minimum possible encoding delay.

Theorems I and II are proved in the Appendix.

Theorem 1

The encoding delay for a nonuniform (n_2, n_1) interleaver is at least $n_2(n_1 - 1)$.

Theorem 2

The encoding delay for a uniform (n_2, n_1) interleaver is at least $(n_2 - 1)(n_1 + 1)$.

Theorems 1 and 2 are precise statements of the facts that whenever $n_1 > n_2$, Type I and Type III interleavers achieve the minimum possible encoding delay for nonuniform and uniform (n_2, n_1) interleavers, respectively. Observe that a Type I interleaver provides the minimum possible encoding delay for any (n_2, n_1) interleaver for which $n_2 < n_1 < 2n_2$, and a Type III interleaver provides the minimum possible encoding delay for any (n_2, n_1) interleaver for which $n_1 \geq 2n_2$, provided the appropriate relative primeness conditions on n_1 and n_2 are met.

We shall now demonstrate that whenever $n_1 < n_2$, either a Type II interleaver or a Type IV interleaver achieves the minimum possible encoding delay.

Theorem 3

If an interleaver achieves the minimum possible encoding delay for any (n_2, n_1) interleaver, its unscrambler is an (n_1, n_2) interleaver that achieves the minimum possible encoding delay for any (n_1, n_2) interleaver.

Proof: Recall that an interleaver and its unscrambler both have the same encoding delay, and that the unscrambler for an (n_2, n_1) interleaver is itself an (n_1, n_2)

interleaver. Suppose the theorem were not true. Then the unscrambler for the minimum-delay (n_1, n_2) interleaver would be an (n_2, n_1) interleaver whose delay is less than that of the minimum-delay (n_2, n_1) interleaver. But this is a contradiction, so the theorem must be true.

In conjunction with Theorems 1 and 2, Theorem 3 asserts that a Type II interleaver provides the minimum possible encoding delay for any (n_2, n_1) interleaver for which $n_1 < n_2 < 2n_1$, and a Type IV interleaver provides the minimum possible encoding delay for any (n_2, n_1) interleaver for which $n_2 \geq 2n_1$, provided the appropriate relative primeness conditions on n_1 and n_2 are met. Thus, for any values of n_1 and n_2 that satisfy the appropriate relative primeness conditions, one of the interleavers of Types I–IV achieves the minimum possible encoding delay.

V. Reduction and Optimality of Storage

Although the basic interleaving techniques achieve the minimum possible encoding delay for any (n_2, n_1) interleaver, they are somewhat wasteful of storage. For example, many of the symbols stored in the later shift-register stages of the Type I or the Type III interleaver have already been read into the output sequence. This fact suggests that it might be possible to reduce the storage capacity of these interleavers without changing their interleaving functions. We now describe a technique for efficiently reducing the storage capacity of the basic interleavers, and then we demonstrate that these reduced-storage interleavers require the minimum possible combined storage capacity for any (n_2, n_1) interleaver.

We shall examine in some detail techniques for reducing the storage capacity of a Type I interleaver. Consider the symbols that must be stored by the interleaver at any given time. Recall that the ordering of the input symbols in the output sequence is $a_k, a_{k+n_1}, a_{k+2n_1}, \cdots, a_{k+n_2n_1}, a_{k+n_2+1}, a_{k+n_2+1+n_1}, \cdots$, for some $n_1 > n_2$. We now describe the symbols that must be stored in the interleaver from the time symbol a_k is read out until symbol $a_{k+n_2n_1}$ is read out. From the time symbol a_k is read out of tap 0, the ordering of input symbols read out of tap j is $a_{k+jn_1}, a_{k+jn_1+n_2+1}, a_{k+jn_1+2(n_2+1)}, \cdots$, $0 \leq j \leq n_2$. Thus symbol a_{k+n_1} is the first input symbol that will be read out of tap 1. Let us list all of the input symbols received before symbol a_{k+n_1} that must still be stored by the interleaver. These are the $\lfloor n_1/(n_2+1) \rfloor + 1$ symbols, $a_k, a_{k+n_2+1}, \cdots, a_{k+l(n_2+1)}$, for all $l(n_2+1) < n_1$, where "$\lfloor x \rfloor$" means "the greatest integer contained in x." All of these symbols will be read out of tap 0. Similarly, symbol a_{k+2n_1} is the first input symbol that will be read out of tap 2. The input symbols that were received before symbol a_{k+2n_1} that must still be stored by the interleaver are the $\lfloor n_1/(n_2+1) \rfloor + 1$ symbols $a_{k+n_1}, \cdots, a_{k+n_1+l(n_2+1)}$, for all $l(n_2+1) < n_1$, and the $\lfloor 2n_1/(n_2+1) \rfloor + 1$ symbols $a_k, \cdots, a_{k+l(n_2+1)}$, for all $l(n_2+1) < 2n_1$. The first $\lfloor n_1/(n_2+1) \rfloor + 1$ of these symbols will be read out of tap 0, as we have just seen, and then the remainder of these symbols will alternately be read out of taps 1 and 0. This listing can be continued in an obvious manner. We find that the input symbols that were received before symbol a_{k+jn_1} that must still be stored by the interleaver include $\lfloor n_1/(n_2+1) \rfloor + 1$ symbols to be read out of tap $j-1$, $\lfloor 2n_1/(n_2+1) \rfloor + 1$ symbols to be read out of tap $j-2$, and so on down to $\lfloor jn_1/(n_2+1) \rfloor + 1$ symbols to be read out of tap 0, $1 \leq j \leq n_2$. The total amount of storage capacity required before symbol a_{k+jn_1} is read out is not quite the sum of these quantities, however, since symbol a_{k+ln_1} may be discarded from storage after it has been read out. The total required storage capacity for the interleaver is therefore

$$S = 1 + \sum_{k=1}^{n_2} \left\lfloor \frac{kn_1}{n_2+1} \right\rfloor. \tag{5}$$

But

$$\sum_{k=1}^{n_2} \left\lfloor \frac{kn_1}{n_2+1} \right\rfloor = \sum_{k=1}^{n_2} \left\lfloor \frac{(n_2+1-k)n_1}{n_2+1} \right\rfloor = \sum_{k=1}^{n_2} \left\lfloor n_1 - \frac{kn_1}{n_2+1} \right\rfloor$$
$$= n_2(n_1-1) - \sum_{k=1}^{n_2} \left\lfloor \frac{kn_1}{n_2+1} \right\rfloor.$$

The last equality follows from the fact that n_1 and n_2+1 are relatively prime. Therefore (5) becomes $S = \frac{1}{2}n_2(n_1-1)+1$. This represents almost a 50 percent reduction in storage capacity from that used by the Type I interleaver.

The preceding discussion suggests an algorithm for constructing and using an interleaver with minimum storage capacity whose operation is identical to that of the Type I interleaver. We first describe the algorithm and then provide a simple example to illustrate its use.

The interleaver consists of an $[\frac{1}{2}n_2(n_1-1)+1]$-stage shift register with taps at positions

$$0, \left\lfloor \frac{n_1}{n_2+1} \right\rfloor, \cdots, \sum_{j=1}^{k} \left\lfloor \frac{jn_1}{n_2+1} \right\rfloor, \cdots, \tfrac{1}{2}n_2(n_1-1),$$

where the shift-register stages and the tap positions are labeled in reverse order of their distance from the input. For notational purposes, define $\beta = \lfloor n_1/(n_2+1) \rfloor$. Assume that symbols $a_k, a_{k+n_2+1}, \cdots, a_{k+\beta(n_2+1)}, a_{k+n_1}, a_{k+(\beta+1)(n_2+1)}, a_{k+n_1+n_2+1}, \cdots$, are stored in order in the shift-register stages. The algorithm proceeds as follows.

1) Read out symbol a_k from tap 0; then shift in a new input symbol.
2) Read out symbol a_{k+n_1} from tap 1; then shift in a new input symbol, but shift *only* the shift-register stages from the input through tap 1.
3) For each j, $0 \leq j \leq n_2$, continue the process in the obvious manner. Read out symbol a_{k+jn_1} from tap j; then shift in a new input symbol, but shift *only* the shift-register stages from the input through tap j.
4) After symbol $a_{k+n_2n_1}$ has been read out and a new input symbol has been shifted into the last shift-register stage, go back and keep repeating steps 1)–4).

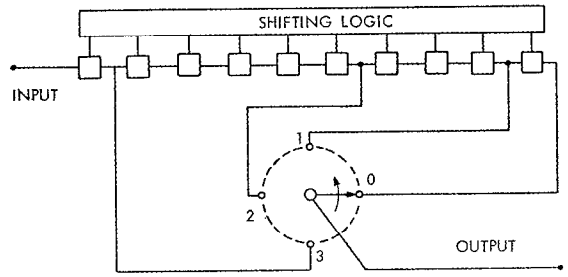

Fig. 5. Interleaver for Example 1.

Example 1

This algorithm can be more easily understood by means of a simple example. Consider a Type I interleaver for which $n_1 = 7$, $n_2 = 3$. It is evident that the ordering of input symbols in the output sequence is 0, 7, 14, 21, 4, 11, 18, 25, 8, 15, 22, 29, $\cdots$. Now consider the operation of the interleaver shown in Fig. 5, which was designed according to the construction procedure just described. It is a 10-stage shift register with taps at stages 0, 1, 4, and 9. Table I lists the symbols stored in the shift-register stages and the symbols that are read out from the initial conditions through the first 5 shifts. The operation of this interleaver is identical to that of the corresponding Type I interleaver, but it requires only 10 shift-register stages instead of 19.

Similar storage-reducing techniques can be applied to the Type II, III, and IV interleavers. The details will not be given here, since they closely follow the methods just described to reduce the storage capacity of Type I interleavers. It turns out that the reduced-storage version of each type of interleaver requires $\frac{1}{2}D + 1$ storage elements, where D is the encoding delay of the basic interleaver. We shall now demonstrate that this realization achieves the minimum possible combined storage capacity for any (n_2, n_1) interleaver and its unscrambler.

Theorem 4

$$S + S_u \geq D.$$

Proof: The effect of the combined interleaver–unscrambler operation is to delay the original sequence by D symbols. At the very least, therefore, symbols $a_{i+1}, a_{i+2}, \cdots, a_{i+D}$ must be stored in either the interleaver or the unscrambler when symbol a_i is read out of the unscrambler.

We have shown that at every point in the (n_1, n_2) plane satisfying the appropriate relative primeness conditions one of the four basic interleaver realizations achieves the minimum possible encoding delay. We have then shown that for the reduced-storage versions of these interleavers and unscramblers,

$$S + S_u = D + 2.$$

Both the interleaver and its unscrambler contain one stage of storage more than is absolutely necessary (consider Example 1 with shift-register stage 9 removed),

TABLE I
STEPS OF INTERLEAVER OPERATION

	Symbols Stored in Stages										Symbol Read Out
	9	8	7	6	5	4	3	2	1	0	
Initial contents	18	16	15	14	12	11	8	7	4	0	0
Shift 1	19	18	16	15	14	12	11	8	7	4	7
Shift 2	20	19	18	16	15	14	12	11	8	4	14
Shift 3	21	20	19	18	16	15	12	11	8	4	21
Shift 4	22	20	19	18	16	15	12	11	8	4	4
Shift 5	23	22	20	19	18	16	15	12	11	8	11

TABLE II
SUMMARY OF OPTIMUM INTERLEAVER PARAMETERS

Type	Encoding Delay; also Combined Storage Capacity	Range of Optimality	Restrictions
I	$n_2(n_1 - 1)$	$n_2 < n_1 < 2n_2$	$n_1, n_2 + 1$ relatively prime
II	$n_1(n_2 - 1)$	$n_1 < n_2 < 2n_1$	$n_2, n_1 + 1$ relatively prime
III	$(n_2 - 1)(n_1 + 1)$	$n_1 \geq 2n_2$	n_1, n_2 relatively prime
IV	$(n_1 - 1)(n_2 + 1)$	$n_2 \geq 2n_1$	n_1, n_2 relatively prime

but this initial stage is generally desirable in practical realizations. Except for this technicality, the reduced-storage interleavers achieve the minimum possible combined storage requirements for any (n_2, n_1) interleaver and its unscrambler.

We summarize the properties of the optimal interleavers that we have found in Table II.

APPENDIX

PROOF OF THEOREMS 1 AND 2

To prove Theorem 1 we use the following lemma.

Lemma 1

Let $a_{z_1}, a_{z_2}, \cdots, a_{z_n}$ be a set of n contiguous symbols in the output sequence of an interleaver, and let $z_1, z_2, \cdots, z_n$ be the positions of these symbols in the input sequence. Let $z_i = \max\{z_1, \cdots, z_n\}$ and $z_j = \min\{z_1, \cdots, z_n\}$, $i \neq j$. Then the encoding delay for the interleaver is at least $(z_i - z_j) - (i - j)$.

Proof: From the introduction, since $D = \sup_j (j - z_j)$ and $0 = \inf_j (j - z_j)$, then

$$D \geq (z_i - z_j) - (i - j) = (j - z_j) - (i - z_i) \geq -D.$$

Proof of Theorem 1

Let $a_{z_1}, a_{z_2}, \cdots, a_{z_{n_2+\alpha}}$ be a set of $n_2 + \alpha$ contiguous symbols in the output sequence, and let $z_1, z_2, \cdots, z_{n_2+\alpha}$ be the positions of these symbols in the input sequence, where $\{z_1, z_2, \cdots, z_{n_2+\alpha}\}$ are mutually separated by at least n_1. Let $z_i = \max\{z_1, z_2, \cdots, z_{n_2+\alpha}\}$, and $z_j = \min\{z_1, z_2, \cdots, z_{n_2+\alpha}\}$. Since the positions are mutually separated by at least n_1,

$$z_i - z_j \geq (n_2 + \alpha - 1)n_1.$$

On the other hand,

$$1 \leq i, j \leq n_2 + \alpha \qquad i \neq j,$$

so that

$$i - j \leq n_2 + \alpha - 1.$$

Applying Lemma 1, we obtain

$$\begin{aligned} D &\geq (z_i - z_j) - (i - j) \\ &\geq (n_2 + \alpha - 1)n_1 - (n_2 + \alpha - 1) \\ &= (n_2 + \alpha - 1)(n_1 - 1). \end{aligned}$$

For a nonuniform (n_2, n_1) interleaver, $\alpha \geq 1$, and Theorem 1 is proved.

Before proving Theorem 2, we first establish some intermediate results. We define a subblock to be the set $\{a_{z_1}, a_{z_2}, \cdots, a_{z_{n_2}}\}$ of n_2 contiguous symbols in the output sequence, where $z_1, z_2, \cdots, z_{n_2}$ denote the positions of these symbols in the input sequence. The kth adjacent subblock is the set

$$\{a_{z_{(1+kn_2)}}, a_{z_{(2+kn_2)}}, \cdots, a_{z_{(n_2+kn_2)}}\}$$

of n_2 contiguous symbols in the output sequence. The relative ordering of a subblock is defined to be the ordering of the input positions $z_1, z_2, \cdots, z_{n_2}$.

Lemma 2

For a uniform (n_2, n_1) interleaver, the relative ordering of contiguous subblocks is the same.

Proof: From the definition of a uniform (n_2, n_1) interleaver,

$$|z_{k+n_2} - z_k| < n_1. \tag{6}$$

Suppose the relative ordering of two contiguous subblocks differs. Then, for some i and j in the range $1 \leq i \neq j \leq n_2$,

$$z_i \geq z_j + n_1, \tag{7}$$

while

$$z_{j+n_2} \geq z_{i+n_2} + n_1. \tag{8}$$

Suppose $i > j$. Using (8), we obtain

$$\begin{aligned} z_{i+n_2} - z_i &= z_{i+n_2} - z_{j+n_2} + z_{j+n_2} - z_i \\ &\leq -n_1 + (z_{j+n_2} - z_i). \end{aligned} \tag{9}$$

Since $|z_{i+n_2} - z_i| < n_1$ from (6), (9) implies

$$z_{j+n_2} - z_i > 0. \tag{10}$$

Using (7), however, we have

$$\begin{aligned} z_{j+n_2} - z_j &= z_{j+n_2} - z_i + z_i - z_j \\ &\geq n_1 + (z_{j+n_2} - z_i). \end{aligned} \tag{11}$$

Since $|z_{j+n_2} - z_j| < n_1$ from (6), (11) implies

$$z_{j+n_2} - z_i < 0. \tag{12}$$

But (10) and (12) are contradictory. A similar contradiction exists when $i < j$. Thus the relative ordering of two contiguous subblocks cannot differ, and the lemma is proved.

Lemma 3

Let the boundaries of a subblock of output symbols from a uniform (n_2, n_1) interleaver be chosen so that $z_1 = \min\{z_1, z_2, \cdots, z_{n_2}\}$. Furthermore, let $\alpha = (z_{n_2+1} - z_1) < n_1$. Then $\max\{z_2, z_3, \cdots, z_{n_2}\} \geq z_1 + (n_2 - 1)n_1 + \alpha$.

Proof: Let $z_i = \min\{z_2, z_3, \cdots, z_{n_2}\}$. Since the members of all sets of n_2 contiguous symbols in the output sequence were mutually separated by at least n_1 symbols in the input sequence, $z_i \geq z_1 + \alpha + n_1$, and thus

$$\max\{z_2, z_3, \cdots, z_{n_2}\} \geq z_1 + (n_2 - 1)n_1 + \alpha.$$

Lemma 4

Let the boundaries of a subblock of output symbols from a uniform (n_2, n_1) interleaver be chosen so that $z_1 = \min\{z_1, z_2, \cdots, z_{n_2}\}$. Furthermore, let $1 \leq \alpha_1 = (z_{n_2+1} - z_1) < n_1$, and let $1 \leq \alpha_2 = (z_{2n_2+1} - z_{n_2+1}) < n_1$. Then, unless $z_2 > z_3 > \cdots > z_{n_2}$, $\max\{z_2, z_3, \cdots, z_{n_2}\} \geq z_1 + (n_2 - 1)n_1 + \alpha_1 + \alpha_2$.

Proof: Suppose that the condition $z_2 > z_3 > \cdots > z_{n_2}$ is not satisfied. Then there are two indices j and k such that $j < k$ and $z_j < z_k$. Without loss of generality, suppose that z_j is the lth lowest member of the set $\{z_1, z_2, \cdots, z_{n_2}\}$, and z_k is the $(l+1)$th lowest member of the set. A simple extension of the proof of Lemma 3 establishes that $z_{n_2+j} \geq z_1 + (l-1)n_1 + \alpha_1 + \alpha_2$. Then, since $j < k$, $z_k \geq z_{n_2+j} + n_1 = z_1 + ln_1 + \alpha_1 + \alpha_2$, so that $\max\{z_2, z_3, \cdots, z_{n_2}\} \geq z_1 + (n_2 - 1)n_1 + \alpha_1 + \alpha_2$.

Lemma 5

For any subblock of output symbols from a uniform (n_2, n_1) interleaver with finite storage, let $z_i = \min\{z_1, z_2, \cdots, z_{n_2}\}$. Define $\alpha_k = z_{kn_2+i} - z_{(k-1)n_2+i}$. Then

$$\bar{\alpha} = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \alpha_k = n_2.$$

Proof: Let $z_i = \max\{z_1, z_2, \cdots, z_{n_2}\}$, and consider the set of input symbols that appears in the first k subblocks of the output sequence. As k becomes arbitrarily large, this set will include all of the symbols a_1 through a_{kn_2}, except for a few symbols near symbols a_1 or a_{kn_2}, under the assumption that the interleaver has finite storage. This assumption also implies that the range $R_k = z_{kn_2+i} - z_{kn_2+i}$ of the kth subblock is also finite. For $0 \leq l \leq k-1$, let $z_i^{(L)} = \min_l z_{ln_2+i}$, and let $z_i^{(H)} = \max_l z_{ln_2+i}$. Then

$$\begin{aligned} = \lim_{N\to\infty} \frac{1}{N} \sum_{k=1}^{N} \alpha_k &= \lim_{k\to\infty} \frac{1}{k} (z_i^{(H)} - z_i^{(L)}) \\ &= \lim_{k\to\infty} \frac{1}{k} (kn_2 - R_k) = n_2, \end{aligned}$$

and Lemma 5 is established.

We have now accumulated enough results to prove Theorem 2.

Proof of Theorem 2

Consider the symbols appearing in a subblock of the output sequence. Without loss of generality, assume that $z_1 = \min \{z_1, z_2, \cdots, z_{n_2}\}$. We consider two cases.

Case I:

$$z_2 > z_3 > \cdots > z_{n_2}.$$

From Lemma 5, $\bar{\alpha} = n_2$, and thus for some k

$$\alpha_k = z_{kn_2+1} - z_{(k-1)n_2+1} \geq n_2.$$

Applying Lemma 3, we obtain (assuming $k = 1$)

$$z_2 = \max \{z_1, z_2, \cdots, z_{n_2}\} \geq z_1 + (n_2 - 1)n_1 + (n_2 - 1) + 1.$$

Applying Lemma 1, we establish that

$$D \geq (z_2 - z_1) - (2 - 1) \geq (n_2 - 1)(n_1 + 1),$$

thereby proving Theorem 2 for Case I.

Case II: The condition $z_2 > z_3 > \cdots > z_{n_2}$ is not satisfied.

From Lemma 5, $\bar{\alpha} = n_2$, so that for some k such that $\alpha_k \geq \alpha_{k-1}$,

$$\alpha_{k-1} + \alpha_k \geq 2n_2.$$

Suppose that $\alpha_{k-1} < 0$. Then $\alpha_k > 2n_2$, and we can apply Lemma 3 (assuming $k = 1$) to obtain

$$\max \{z_1, z_2, \cdots, z_{n_2}\} \geq z_1 + (n_2 - 1)n_1 + 2n_2 + 1.$$

Applying Lemma 1, and observing that $i - j \leq n_2 - 1$, we obtain

$$D \geq (n_2 - 1)n_1 + 2n_2 - (n_2 - 1) + 1 = [(n_2 - 1)(n_1 + 1) + 3],$$

thereby proving Theorem 2 for Case II when $\alpha_{k-1} < 0$.

Finally, suppose that $\alpha_{k-1} > 0$. Then we can apply Lemma 4 (assuming $k = 1$) to obtain

$$\max \{z_1, z_2, \cdots, z_{n_2}\} \geq z_1 + (n_2 - 1)n_1 + 2n_2.$$

Applying Lemma 1 as before, we obtain

$$D \geq [(n_2 - 1)(n_1 + 1) + 2],$$

thereby completing the proof of Theorem 2.

Acknowledgment

The author wishes to thank Prof. R. G. Gallager for his suggestions and for his helpful comments during the preparation of this paper. It was he who pointed out the useful fact that an (n_2, n_1) unscrambler is also an (n_1, n_2) interleaver.

References

[1] K. Brayer and O. Cardinale, "Evaluation of error correction block encoding for high-speed HF data," *IEEE Trans. Communication Technology*, vol. COM-15, pp. 371–382, June 1967.
[2] P. Elias, "Error-free coding," *IRE Trans. Information Theory*, vol. IT-4, pp. 29–37, September 1954.
[3] N. Abramson, "Cascade decoding of cyclic product codes," *IEEE Trans. Communication Technology*, vol. COM-16, pp. 398–402, June 1968.
[4] W. W. Peterson, *Error-Correcting Codes*. Cambridge, Mass.: M.I.T. Press, 1961, pp. 81–85.
[5] K. Brayer, "The improvement of digital HF communication through coding: II—Tandem interleaved cyclic coding," *IEEE Trans. Communication Technology*, vol. COM-16, pp. 779–786, December 1968.