Permutation Decoding of Systematic Codes

By JESSIE MACWILLIAMS

(Manuscript received September 4, 1963)

A symmetry of a systematic code is a permutation of bit positions in each code word (the same permutation is applied to all code words) which preserves the code as a whole. Permutation decoding makes use of these symmetries to build up a decoding algorithm for the code.

It is difficult to find an appropriate set of symmetries for a code picked at random. For cyclic codes the problem is somewhat easier, and for some special cyclic codes it is solved completely in this paper. For these codes, at least, it is evident that permutation decoding is easy to implement and inexpensive compared with other decoding schemes.

Permutation decoding as a means of error control is evaluated for the binary symmetric channel and for the switched telephone network as represented by experimental data. It is found to be extremely effective on the binary symmetric channel and of very doubtful value on the present telephone network.

INTRODUCTION

A systematic code of block length n is a subspace of the vector space of all possible rows of n symbols chosen from a finite field. In this paper such a code will be called an alphabet,[1] and the sequences belonging to the alphabet will be called letters.

The parameters used to describe an alphabet are block length, n, number of information places, k, and error correcting capability, e: n is the number of symbols in each letter, k is the dimension of the alphabet as a vector space, and e is defined by the property that the minimum Hamming distance between two letters is either $2e + 1$ or $2e + 2$.

It is well known[1] that an alphabet with parameters n,k,e is theoretically capable of correcting all occurrences of $\leqq e$ errors in a block of length n. However for $e > 1$, the process of error correction by decoding is complicated, and likely to require expensive equipment. In this paper we describe a new decoding scheme, permutation decoding, which is conceptually simple and quite easy to implement.

The decoding procedure consists of a sequence of permutations of the received block of symbols, each of which is followed by a parity check calculation. We can thus make a rough comparison between the complexity of the equipment required for encoding and decoding. The encoder uses one parity check register, and the decoder uses r (or uses one r times), where r is the number of permutations in the decoding sequence. Real time operation with a constant time delay is possible and perhaps not too expensive.

Permutation decoding owes much to the previous work of Peter Neumann[2] and Eugene Prange.[3] It depends essentially on the symmetries of the alphabet. A symmetry of an alphabet means a permutation of digit positions which preserves the alphabet as a whole. The same permutation is applied to the digits of every letter, and each letter is changed, if at all, into another letter of the same alphabet. Very little is known about symmetries of alphabets in general, but it will be shown that even this little is enough to enable us to apply the decoding scheme to a large class of alphabets.

Permutation decoding differs from previous schemes in two important ways. First, it becomes easier as the redundancy of the alphabet increases; it is most useful for alphabets with high error correcting capabilities. Secondly, it cannot correct more than e errors in n places. A received sequence containing more than e errors either will be "corrected" wrongly or will emerge unchanged from the decoder.

It will become apparent in Section III that permutation decoding produces many more undetected errors than does error control by detection and retransmission. The simpler scheme of detection and retransmission should be used when it is at all feasible.

The plan of this paper is as follows: Section I contains a description of permutation decoding in general, without reference to the particular alphabet we wish to decode. Section II is an example; it contains a detailed account of a particular permutation group which will suffice to decode many binary cyclic alphabets. Section III describes how the probability of improper correction and of detection without correction may be estimated.

I. ERROR CONTROL AND PERMUTATION DECODING

Let V^n be the set of all possible binary sequences of length n.* The distance between two sequences is the number of places in which they

* The method will work equally well for multilevel codes. All that is needed is to find a euphonious substitute for the term "binary sequence."

differ; the distance between v_1 and v_2 is the minimum number of bits we must change in v_1 in order to convert it into v_2.

For purposes of error control, some sequences of V^n are designated as the sequences which will be transmitted. This subset of V^n is called a code. Error detection consists in finding out whether a received sequence belongs to the code. Every method of error correction consists in mapping a received sequence onto the nearest member of the code, where nearness is defined in terms of the distance function defined above. If there are several nearest members, the correction procedure chooses one in some arbitrary fashion or indicates that an uncorrectable error has been found.

The strategy for choosing a code is usually to place its members as far apart as possible in V^n. It will then take a relatively large number of errors to cause a transmitted code sequence to be received as a different code sequence. If the distance between any two code sequences is $\geqq 2e + 1$, it is theoretically possible to correct all single, double, $\cdots$ e-fold errors. This may be restated as follows: If v is a received sequence in which $\leqq e$ errors have occurred, there is a unique code sequence α at distance $\leqq e$ from v. Every other member of the code is at distance $> e$ from v.

The business of decoding is to find α, given v. To do this expeditiously we need some additional structure in the code, and from now on we restrict our choice of codes to the kind described in the next paragraph.

An alphabet* (systematic code, group code) is one in which a fixed number of fixed bit positions are designated as information places, and the other bit positions contain parity checks, which are linear combinations of the contents of the information places. For convenience, the first k bit positions are taken to be the information places. From any k-place binary sequence h we obtain a unique letter of the alphabet by adding $n - k$ parity checks. This letter will be denoted by $m(h)$.

Let $\bar{\alpha}$ stand for the first k coordinates of the n-place sequence α. α is a letter of the alphabet if and only if $\alpha = m(\bar{\alpha})$. Let π be a permutation of bit positions in V^n which preserves the alphabet; if α is a letter, so is $\alpha\pi$. The first k positions of $\alpha\pi$ are information places, and $\alpha\pi = m(\overline{\alpha\pi})$.

Let v be a received sequence containing $\leqq e$ errors. If no errors have occurred in the first k places of v, $\alpha_0 = m(\bar{v})$ is the unique letter of the alphabet at distance $\leqq e$ from v, and is the corrected version of v. On the other hand, if one or more errors have occurred in the first k places of v, the letter $m(\bar{v})$ is not the corrected version of v, since it is the same as v in the first k places. In this case $m(\bar{v})$ is at distance $> e$ from v.

* It has been shown[1] that every systematic code is a group code and that every group code is a systematic code.

The first step in the decoding procedure is to form $\alpha_0 = m(\bar{v})$, find the distance between v and α_0, and take α_0 as the corrected version of v if this distance is $\leqq e$.

Let α denote the unique letter of the alphabet at distance $\leqq e$ from v. Let π be, as before, a permutation of bit positions which preserves the alphabet. Clearly the distance between $\alpha\pi$ and $v\pi$ is the same as that between α and v.

Suppose that we can find a permutation π_i which preserves the alphabet and which moves the errors in v out of the first k positions. Then $\alpha_i = m(\overline{v\pi_i})$ is at distance $\leqq e$ from $v\pi_i$, and is the unique letter of the alphabet with this property. Consequently $\alpha = \alpha_i\pi_i^{-1}$ is the corrected version of v.

This suggests the following decoding procedure: Let I (the identity), $\pi_1, \pi_2, \cdots$ be a sequence of permutations which preserve the alphabet. Form the letters

$$\alpha_0 = m(\overline{vI}), \qquad \alpha_1 = m(\overline{v\pi_1}), \qquad \alpha_2 = m(\overline{v\pi_2}), \cdots$$

and at each step find the distance between α_i and $v\pi_i$. Continue until a letter α_i is found which is at distance $\leqq e$ from $v\pi_i$. Then $\alpha_i\pi_i^{-1}$ is the corrected version of v.

A received vector which is at distance $>e$ from all letters of the alphabet will be detected as an error but not corrected by this procedure. Some provision must be made for this eventuality. This is discussed in Section III.

It is also possible for the decoder to make an incorrect "correction." This will happen if an error pattern (of more than e errors) causes the transmitted letter α to be received as a sequence v which is at distance $\leqq e$ from a different letter α'. The probability of this occurrence is calculated in Section III.

In order for permutation decoding to work, we must be sure that one of the sequences $v\pi_i$ is correct in the first k places. If $v = \alpha + f$, f being the error sequence, the permutation π_i must move all nonzero coordinates of f out of the first k places. In order for the procedure to be practical, it must be possible to move all sets of $\leqq e$ errors out of the first k places with a fairly short sequence of permutations.

To correct all sets of $\leqq e$ errors in a block of length n, we need the following: (i) an alphabet of block length n, dimension k, and minimum distance $\geqq 2e + 1$; and (ii) a set of permutations, $\pi_1, \pi_2, \cdots$ which preserve the alphabet and at the same time move any set of $\leqq e$ errors out of the first k places.

We emphasize that the reason for insisting that the permutations π_i shall preserve the alphabet is to keep the parity check calculation always the same. The encoder (the parity check calculator) is a complicated and expensive piece of equipment; it is desirable to use only one encoder in the decoding scheme. If for any reason (such as real time operation) it is necessary to have more than one encoder, we can, to a certain extent, relax the restriction on the permutations π_i.

It may seem to be quite a trick to find at the same time both a suitable alphabet and a suitable set of permutations; really the chief difficulty is that neither alphabets nor permutation groups have been studied from this point of view. It is shown in Section II that a very simple permutation group will do for many cyclic alphabets.

We conclude this section with an example of permutation decoding applied to the Hamming alphabet with $n = 7, k = 4, e = 1$.* The alphabet is written out in Table I; it is seen to be invariant under cyclic permutation.

TABLE I — A CYCLIC ALPHABET WITH $n = 7, k = 4, e = 1$

0	0	0	0	0	0	0
0	1	1	0	1	0	0
0	0	1	1	0	1	0
0	0	0	1	1	0	1
1	0	0	0	1	1	0
0	1	0	0	0	1	1
1	0	1	0	0	0	1
1	1	0	1	0	0	0
1	1	1	1	1	1	1
1	0	0	1	0	1	1
1	1	0	0	1	0	1
1	1	1	0	0	1	0
0	1	1	1	0	0	1
1	0	1	1	1	0	0
0	1	0	1	1	1	0
0	0	1	0	1	1	1

Let T denote the cyclic permutation. Clearly at most four applications of T will move any single error out of the first four places. The decoding sequence consists of the permutations I, T, T^2, T^3, T^4.†

Let the received vector be 1110100 (the first nonzero vector of Table I with an error in the first place). The successive stages of the decoding process are shown in Table II.

* The example is chosen for simplicity. Permutation decoding is *not* the most efficient way of correcting single errors.

† E. R. Berlekamp has pointed out that the shorter decoding sequence I, T^3, T^6 is sufficient.

TABLE II — DECODING PROCEDURE FOR THE ALPHABET OF TABLE I

$V = 1110100$	
$m(\bar{V}) = 1110010$	distance = 2
$VT = 0111010$	
$m(\overline{VT}) = 0111001$	distance = 2
$VT^2 = 0011101$	
$m(\overline{VT^2}) = 0011010$	distance = 3
$VT^3 = 1001110$	
$m(\overline{VT^3}) = 1001011$	distance = 2
$VT^4 = 0100111$	
$m(\overline{VT^4}) = 0100011$	distance = 1

Thus $\alpha = 0100011$ is the unique letter at distance $\leqq 1$ from VT^4, and the corrected version of V is $\alpha T^{-4} = \alpha T^3 = 0110100$.

II. PERMUTATION DECODING OF CYCLIC ALPHABETS*

The coordinate places in V^n are labeled by the numbers $0, 1, 2, \cdots, n - 1$. This notation is convenient for describing permutations. If ω stands for one of these numbers, the cyclic permutation is

$$T: \omega \to \omega + 1 \text{ (addition mod } n).$$

The powers of the cyclic permutation are

$$T^2: \omega \to \omega + 2; \qquad T^3: \omega \to \omega + 3, \cdots, \qquad T^n: \omega \to \omega + n = \omega.$$

A cyclic alphabet in V^n is an alphabet which is invariant under T, hence also invariant under T^2, T^3, etc. We assume that we wish to decode a cyclic alphabet with parameters n,k,e.

Successive cyclic shifts will eventually bring any k consecutive bits to the first k positions, and hence will move out of the first k positions any error pattern in which there is a gap of length $\geqq k$. In particular, the sequence $I, T, \cdots, T^{n-1}$ will always correct all single errors.†

This sequence will not correct an error pattern in which there is no gap of length $\geqq k$. Suppose, for example, that $n = 23$, $k = 12$. The error pattern shown below cannot be corrected by cyclic shifts alone.

X 1 2 3 4 5 6 7 8 X 10 11 12 13 14 15 16 17 18 X 20 21 22

To deal with such cases we introduce another permutation $U: \omega \to 2\omega$ (multiplication mod n), and its powers, $U^2: \omega \to 4\omega$, $U^3: \omega \to 8\omega$, etc. If n is odd, there exists a least integer t such that $2^t \equiv 1 \bmod n$; and $U^t = I$. The choice of U is motivated by the following theorem.

* It is to be emphasized that this section is only an example. The permutations described here will not suffice to decode all cyclic alphabets of odd block length. A method of finding other permutations is given in Appendix A.

† It will also correct all double errors if $k < n/2$, and so on.

Theorem 1: Every binary cyclic alphabet of odd block length is invariant under U and the powers of U.

The proof of this theorem is given in Appendix A.

The error pattern 0, 9, 19 above is changed by U into 0, 18, 15. This pattern is moved out of the first twelve places by 21 cyclic shifts.

The permutation group on $0, 1, \cdots, n-1$ generated by T and U will be called G_n. It is easy to check that $TU = UT^2$; hence we may represent every permutation in G_n in the form U^iT^j, with $0 \leqq i \leqq t-1$, $0 \leqq j \leqq n-1$. Now every power of U leaves 0 fixed, and no power of T (except the identity) leaves 0 fixed; thus $U^iT^j = U^hT^k$ if and only if $i = h \bmod t$ and $j = k \bmod n$. It follows that the group G_n is of order nt and consists of the permutations:

$$\begin{array}{ccccc}
I, & T, & T^2, & \cdots, & T^{n-1} \\
U, & UT, & UT^2, & \cdots, & UT^{n-1} \\
U^2, & U^2T, & U^2T^2, & \cdots, & U^2T^{n-1} \\
\cdot & \cdot & \cdot & \cdots, & \cdot \\
U^{t-1}, & U^{t-1}T, & U^{t-1}T^2, & \cdots, & U^{t-1}T^{n-1}.
\end{array}$$

Let $0 \leqq u_1 < u_2 < \cdots < u_s \leqq n-1$ (n odd), be a set of integers; we suppose that errors occur in places $u_1, \cdots, u_s$. Let $g(u_1, \cdots, u_s, n)$ be the length of the maximum gap which can be inserted in this sequence by repeated multiplication by 2 mod n.

If $u_{\nu k}$ denotes that integer less than n which is congruent to $2^k u_\nu$ mod n, then

$$g(u_1, \cdots, u_s, n) + 1 = \operatorname*{Max}_{i,j,k} |u_{ik} - u_{jk}|,$$

under the condition that the interval $[u_{ik}, u_{jk}]$ contain no other u_{vj}. Let $g(s,n)$ be the minimum value of this maximum gap for all possible choices of the s values $u_1, \cdots, u_s$.

$$g(s,n) = \operatorname*{Min}_{u_1,\cdots,u_s} g(u_1, \cdots, u_s, n).$$

The group G_n then contains a permutation which moves any set of s errors out of the first $g(s,n)$ places. Clearly $s' < s$ implies $g(s',n) \geqq g(s,n)$. Hence a binary cyclic n,k,e alphabet with n odd may be decoded by G_n if and only if $k \leqq g(e,n)$. The quantity $g(e,n)$ has a few obvious properties.

The numbers $0, 1, \cdots, n-1$ can be partitioned into subsets which are invariant under U.* For example, for $n = 15$, these subsets are

* The number of cyclic alphabets of block length n is determined by the number of these subsets; the dimensions of the cyclic alphabets are determined by the sizes of the invariant subsets; see, for example, Ref. 4.

$$(0),\ (1,2,4,8),\ (3,6,12,9),\ (5,10),\ (7,14,13,11).$$

The union of any number of invariant subsets is also invariant under U; from these subsets we may obtain upper bounds on $g(e,n)$ by inspection. In the example above we obtain:

$g(2,15) \leqq 9$, since the invariant set $(5,10)$ gives us a maximum gap 11,12,13,14,0,1,2,3,4 of length 9.

$g(3,15) \leqq 4$, since the invariant set $(0,5,10)$ gives us a maximum gap 1,2,3,4 of length 4.

$g(5,15) \leqq 2$, since the invariant set $(0,3,6,9,12)$ gives us a maximum gap of length 2.

These upper bounds limit the usefulness of the group G_n. However it is still sufficiently useful to be of interest.

The value of $g(e,n)$ for various choices of e,n have been computed on the IBM 7090. These are tabulated in Table III together with the parameters n,k,e for several cyclic alphabets.

TABLE III — EFFECT OF PERMUTATION U FOR DIFFERENT BLOCK LENGTHS

Code			Gap Length			
n	k	e	$g(2)$	$g(3)$	$g(4)$	$g(5)$
47	24	5				26
31	21	2	25			
31	16	3		19		
31	11	4			12	
23	12	3		17		
21	12	2	13			
21	9	3		6		
21	5	4			6	
17	9	2	13			
15	7	2	9			
15	5	3		4		

The gap length $g(s)$ is the maximum number of consecutive error-free positions which can be inserted into an arbitrary pattern of s errors in n places by successive applications of the permutation $w \to 2w \bmod n$. If $k \leqq g(e)$, the group generated by T and U contains a sequence of permutations which will suffice to decode an n, k, e alphabet.

It is desirable, if possible, to use only part of G_n in the decoding sequence. As an example we consider the alphabet with $n = 23$, $k = 12$, $e = 3$.* In this case one of the permutations, U, U^2, U^{11} will always create a gap of length at least 12 in any set of 3 errors. The decoding sequence is:†

* This alphabet is described in detail in Table V.

† It has been shown by E. R. Berlekamp that a subsequence of length 40 is all that is really necessary.

$$\begin{array}{ccccc} I, & T, & T^2, & \cdots, & T^{22} \\ U, & UT, & UT^2, & \cdots, & UT^{22} \\ U^2, & U^2T, & U^2T^2, & \cdots, & U^2T^{22} \\ U^{11}, & U^{11}T, & U^{11}T^2, & \cdots, & U^{11}T^{22}. \end{array}$$

The decoding procedure for this particular alphabet has been simulated on the IBM 7090. A diagram of the logical program is given in Fig. 1, and Table IV traces a particular sequence through the decoder. In practice it is convenient to add one more permutation (in this case TU) to the decoding permutations, so that a sequence passing through the entire

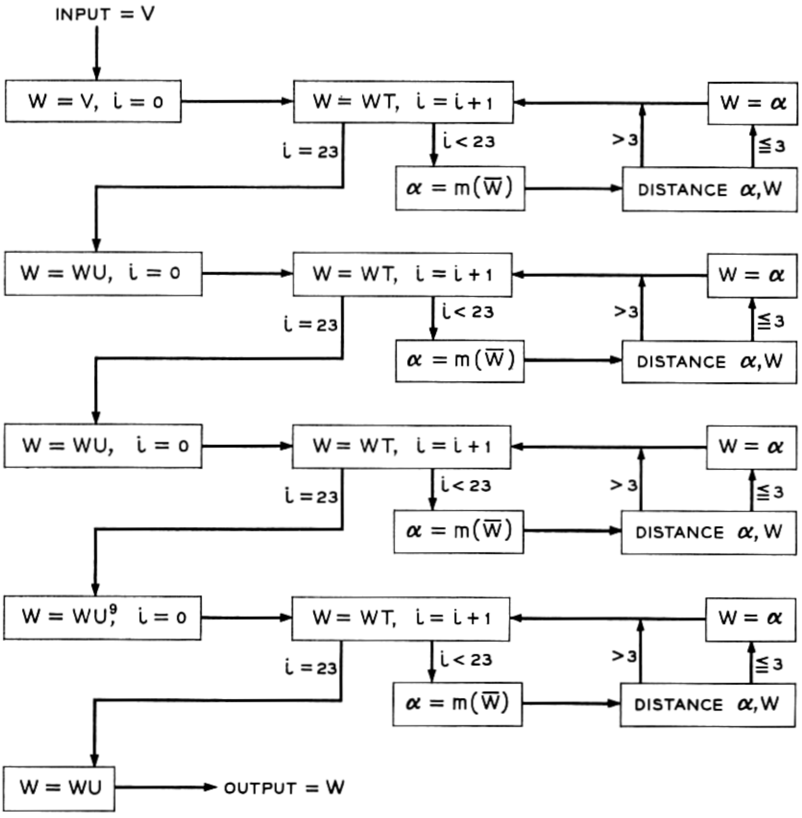

Fig. 1 — Logic of decoder for (23,12,3) alphabet. The contents of the boxes are Fortran-type instructions; for example, $i = i + 1$ is to be interpreted as "replace i by $i + 1$."

TABLE IV — DECODING PROCEDURE FOR THE (23,12,3) ALPHABET

													Parity Check												Distance
v	3	5	7	11	13	14	15	18	19	20	21	22	3	5	7	11	13	16	18	21					6
vT	0	4	6	8	12	14	15	16	19	20	21	22	0	4	6	8	16	18	19						7
vT^{22}	2	4	6	10	12	13	14	17	18	19	20	21	2	4	6	10	12	15	17	20					6
vU	3	5	6	7	10	13	14	15	17	19	21	22	3	5	6	7	10	12	15	16	17	19	20		7
vUT	0	4	6	7	8	11	14	15	16	18	20	22	0	4	6	7	8	11	12	13	14	20	21	22	6
vUT^{22}	2	4	5	6	9	12	13	14	16	18	20	21	2	4	5	6	9	13	14	17	18	19	21	22	6
vU^2	3	5	6	7	10	11	12	14	15	19	20	21	3	5	6	7	10	11	12	13	19	20	21	22	4
vU^2T	4	6	7	8	11	12	13	15	16	20	21	22	4	6	7	9	11	13	16	17	18	20	21		5
vU^2T^{12}	0	1	3	4	8	9	10	15	17	18	19	22	0	1	3	4	8	9	10	12	15	18	19		3
αU^2T^{12}	0	1	3	4	8	9	10	12	15	18	19														
αU^2T^{22}	2	5	6	9	12	14	16	18	20	21	22														
αU^{10}	0	9	10	11	13	14	17	18	19	21	22														
$\alpha U^{10}T$	0	1	10	11	12	14	15	18	19	20	21														
$\alpha U^{10}T^{22}$	8	9	10	12	13	16	17	18	20	21	22														
α	0	3	5	11	13	15	18	19	20	21	22														

The numbers indicate the positions of the ones in the 23-bit sequence. For example, the first sequence v is 00010101000101110011111.

set emerges in its original form. The final output of the permuting register is then the corrected form of the received sequence.

The operation of the 7090 program is of course sequential—it employs one subroutine to simulate the parity check calculation. It is clear from the logic diagram that it is quite convenient to split the encoder into four parallel sections, each of which contains a register capable of making a cyclic permutation and an encoder to calculate parity checks. This idea can be applied to speed up the decoding of any cyclic alphabet.

III. EVALUATION OF PERMUTATION DECODING AS A MEANS OF ERROR CONTROL

Permutation decoding of an n,k,e alphabet $\mathcal{A}$ will map a received sequence v onto the nearest letter of $\mathcal{A}$ provided that this letter is unique. This is the case if v lies at distance $\leqq e$ from some letter of $\mathcal{A}$. If v is at distance $>e$ from every letter of $\mathcal{A}$ the decoder will detect an error but will be unable to correct it.*

The decoder will also make mistakes. If f is an error sequence of more than e errors, and α the transmitted letter, the received sequence $\alpha + f$ may lie at distance $\leqq e$ from some other letter α' of $\mathcal{A}$. The decoder will then interpret $\alpha + f$ as α'. The error sequences which cause such incorrect decoding are characterized by the following theorem.

Theorem 2: The error sequences which cause the decoder to "correct" incorrectly are exactly those sequences of weight $>e$ which lie at distance $\leqq e$ from some letter of $\mathcal{A}$.

Proof: Let f be a sequence of weight $>e$ such that $f = \beta + f'$, $\beta \in \mathcal{A}$, f' of weight $\leqq e$.

For any transmitted letter α

$$\alpha + f = \alpha + \beta + f',$$

and the decoder will interpret $\alpha + f$ as $\alpha + \beta$.

Conversely, suppose that f is an error sequence such that $\alpha + f$ is decoded as $\alpha' \neq \alpha$. Then

$$\alpha + f = \alpha' + f' \qquad \alpha' \in \mathcal{A}, f' \text{ of weight } \leqq e,$$

and

$$f = \alpha' - \alpha + f'.$$

Hence f is at distance $\leqq e$ from the letter $\alpha' - \alpha$ of $\mathcal{A}$.

* One is tempted to suggest that the decoder ask for retransmission of such detected errors. This idea is of dubious value; error correction by decoding should not be used at all if error correction by detection and retransmission is a possible alternative. We must assume that retransmission is extremely awkward, if not completely infeasible.

Let $A(s)$, $s = 0, 1, \cdots, n$ be the number of letters of $\mathcal{A}$ of weight s. Let $C(s)$, $s = 0, 1, \cdots, n$ be the number of sequences of V^n of weight s which lie at distance $\leqq e$ from letters of $\mathcal{A}$. The $C(s)$ are uniquely determined by the $A(s)$, and may be obtained from them by a simple calculation; the exact formula is given in Appendix B. The values of $A(s)$ for a number of binary cyclic alphabets are tabulated in Table V, and the values of $C(s)$ for these alphabets are given in Table VI.

For $s \leqq e$, $C(s) = \binom{n}{s}$ and is the number of sequences of weight s which are properly corrected by the decoder. For $s > e$, $C(s)$ is the number of sequences of weight s which are improperly corrected by the decoder.

Let $D(s)$, $s = e + 1, \cdots, n$ be the number of error sequences of weight s which cause the decoder to detect an error that it cannot correct. Clearly $D(s) = 0$ for $s \leqq e$, and $D(s) = \binom{n}{s} - C(s)$ for $s > e$.

P_E denotes the probability that a received sequence will be "corrected" incorrectly by the decoder; P_D denotes the probability that a received sequence will be detected as an error but not corrected. We consider first a binary symmetric memoryless channel with bit error probability p. The probability of s specific errors in a block of length n is then $p^s(1 - p)^{n-s}$, and this probability is independent of the location of the errors. Hence

$$P_E(\text{B.S.}) = \sum_{s=e+1}^{n} C(s)p^s(1 - p)^{n-s},$$

$$P_D(\text{B.S.}) = \sum_{s=e+1}^{n} D(s)p^s(1 - p)^{n-s}.$$

It is to be noted that if error correction by detection and retransmission is used, the probability of an undetected error is

$$\sum_{s=d}^{n} A(s)p^s(1 - p)^{n-s}; \qquad d = 2e + 1 \quad \text{or} \quad 2e + 2.$$

This sum starts with the first nonzero value of $A(s)$ (for $s > 0$), i.e. with $s = 2e + 1$ or $s = 2e + 2$. It is obvious that for the values of p,n currently in use in the Bell System, error correction by detection and retransmission is the preferable scheme.

The values of $P_E(\text{B.S.})$ and $P_D(\text{B.S.})$ for a number of alphabets are tabulated in Table VII; p is taken to be 3.22×10^{-5}, the over-all bit error rate on the telephone network obtained from the Alexander, Gryb and Nast task force data.[5]

Except for the first and last example, the alphabets in Tables V, VI and VII occur in pairs. The second alphabet in each pair consists of the letters of even weight in the first. Since the minimum distance of the second alphabet is $2e + 2$, its value of $C(e + 1)$ is zero. In other words, every error of weight $e + 1$ will be detected. If this is important, it is advantageous to use the second alphabet.

The error rates of Table VI are fantastically low; unfortunately the fantasy resides in the binary symmetric channel. The situation is very different on the real telephone channel.

Let $P(s,n)$ be the probability of s errors in n consecutive bits. [For the binary symmetric channel $P(s,n) = \binom{n}{s} p^s(1 - p)^{n-s}$.] Tables of $P(s,n)$ for the telephone network have been calculated from the Alexander, Gryb and Nast task force data.

The decoder will either detect or correct wrongly every error of weight $>e$. Hence

$$P_E + P_D = \sum_{s=e+1}^{n} P(s,n).$$

It is impossible to obtain exact formulae for P_E and P_D separately. Using the methods of Ref. 5 we obtain the approximate formulae

$$P_E(\text{T.N.}) = \sum_{s=e+1}^{n} C(s) \frac{P(s,n)}{\binom{n}{s}}$$

$$P_D(\text{T.N.}) = \sum_{s=e+1}^{n} D(s) \frac{P(s,n)}{\binom{n}{s}}.$$

These numbers have been computed and are tabulated in Table VIII. This shows rather clearly that on the channel described by the Alexander, Gryb and Nast data the word error rate with error correction is greater than the bit error rate with no encoding.

Of course the average error rates of Table VIII conceal something. Examination of the $P(s,n)$ tables for the individual calls shows that about half of the calls in which errors occurred would be handled successfully by permutation decoding.

TABLE V — VALUES OF $A(s)$

n,k,e	47,24,5		31,21,2	31,20,2		31,16,3	31,15,3
s	$A(s)/47$	s	$A(s)/31$	$A(s)/31$	s	$A(s)/31$	$A(s)/31$
0	1/47	0	1/31	1/31	0	1/31	1/31
11	92	5	7	0	7	5	0
12	276	6	27	27	8	15	15
15	3795	7	75	0	11	168	0
16	7590	8	245	245	12	280	280
19	35420	9	655	0	15	589	0
20	49588	10	1387	1387	16	589	589
23	81720	11	2640	0	19	280	0
24	81720	12	4480	4480	20	168	168
27	49588	13	6510	0	23	15	0
28	35420	14	8310	8310	24	5	5
31	7590	15	9489	0	31	1/31	0
32	3795	16	9489	9489			
35	276	17	8310	0			
36	92	18	6510	6510			
47	1/47	19	4480	0			
		20	2640	2640			
		21	1387	0			
		22	655	655			
		23	245	0			
		24	75	75			
		25	27	0			
		26	7	7			
		31	1/31	0			

$n,k,e =$	23,12,3	23,11,3		(21,12,2)	(21,11,2)	(17,9,2)	(17,8,2)			(15,7,2)
s	$A(s)/23$	$A(s)/23$	s	$A(s)$	$A(s)$	s	$A(s)$	$A(s)$	s	$A(s)$
0	1/23	1/23	0	1	1	0	1	1	0	1
7	11	0	5	21	0	5	34	0	5	18
8	22	22	6	168	168	6	68	68	6	30
11	56	0	7	360	0	7	68	0	7	150
12	56	56	8	210	210	8	85	85	8	150
15	22	0	9	280	0	9	85	0	9	30
16	11	11	10	1008	1008	10	68	68	10	18
23	1/23	0	11	1008	0	11	68	0	15	1
			12	280	280	12	34	34		
			13	210	0	17	1	0		
			14	360	360					
			15	168	0					
			16	21	21					
			21	1	0					

Table VI — Values of $C(s)$

n,k,e =	47,24,5*	31,21,2*	31,20,2	31,16,3	31,15,3
s = 1	$C(s)$ = 47	31	31	31	31
2	1081	465	465	465	465
3	16215	2170	0	4495	4495
4	178365	13640	12555	5425	0
5	1533939	82274	5022	29295	26040
6	1997688	360964	339047	92225	13020
7	11700743	1276115	81685	329375	303180
8	58503719	3829585	3591040	1248525	86025
9	253516120	9788250	604655	3190675	2861455
10	1094459500	21506932	20159981	6790333	690525
11	3681363800	41087771	2569497	12963363	11812395
12	10764415000	68535730	64275400	21284445	1987720
13	28981118000	100106900	6245260	31108034	28201320
14	70307802000	128661310	120616350	40561485	3675360
15	154677160000	145890120	9085914	45969682	41569263
16	310193350000	145890120	136804210	45969682	4400419
17	565646700000	128661310	8044965	40561485	36886124
18	942103590000	100106900	93861645	31108034	2906715
19	1437947500000	68535730	4260330	21284445	19296724
20	2012287600000	41087771	38518275	12963363	1150968
21	2587226900000	21506932	1346950	6790333	6099808
22	3059047600000	9788250	9183595	3190675	329220
23	3325051800000	3829585	238545	1248525	1162500
24		1276115	1194430	329375	26195
25		360964	21917	92225	79205
26		82274	77252	29295	33255
27	+ symmetric	13640	1085	5425	5425
28	terms	2170	2170	4495	0
29		465	0	465	0
30		31	0	31	0
31		1	0	1	0

n,k,e =	23,12,3	23,11,3	21,12,2	21,11,2	17,9,2	17,8,2	15,7,2
s = 1	$C(s)$ = 23	23	21	21	17	17	15
2	253	253	210	210	136	136	105
3	1771	1771	210	0	340	0	180
4	8855	0	2625	2520	1190	1020	540
5	33649	28336	10269	1008	3910	408	1413
6	100947	14168	24024	21168	7820	6936	2355
7	245157	216568	52440	4200	11560	1428	3135
8	490314	61226	92610	85050	14450	13005	3135
9	817190	715990	131530	12810	14450	1445	2355
10	1144066	138138	161196	146748	11560	10132	1413
11	1352078	1180774	161196	14448	7820	884	540
12	1352078	171304	131530	118720	3910	3502	180
13	1144066	1005928	92610	7560	1190	170	105
14	817190	101200	52440	48240	340	340	15
15	490314	429088	24024	2856	136	0	1
16	245157	28589	10269	9261	17	0	
17	100947	86779	2625	105	1	0	
18	33649	5313	210	210			
19	8855	8855	210	0			
20	1771	0	21	0			
21	253	0	1				
22	23	0					
23	1	0					

* This table is correct to eight significant figures.

TABLE VII — ERROR RATES FOR THE BINARY SYMMETRIC CHANNEL

n,k,e	P_E	P_D	$P_E + P_D$
47,24,5	2.23×10^{-21}	4.56×10^{-20}	4.78×10^{-20}
31,21,2	7.23×10^{-11}	0.77×10^{-10}	1.50×10^{-10}
31,20,2	1.342×10^{-12}	1.50×10^{-10}	1.50×10^{-10}
31,16,3	5.83×10^{-15}	2.80×10^{-14}	3.38×10^{-14}
31,15,3	9.01×10^{-19}	3.38×10^{-14}	3.38×10^{-14}
23,12,3	9.59×10^{-15}	0	9.59×10^{-15}*
23,11,3	9.81×10^{-19}	9.59×10^{-15}	9.59×10^{-15}
21,12,2	7.00×10^{-12}	3.72×10^{-11}	4.42×10^{-11}
21,11,2	2.70×10^{-15}	4.42×10^{-11}	4.42×10^{-11}
17,9,2	4.54×10^{-12}	1.80×10^{-11}	2.27×10^{-11}
17,8,2	1.11×10^{-13}	2.27×10^{-11}	2.27×10^{-11}
15,7,2	6.092×10^{-12}	1.078×10^{-10}	1.078×10^{-10}

* This is a close-packed alphabet; every sequence of 23 binary bits is at distance $\leqq 3$ from some letter of the alphabet.

CONCLUSION

Permutation decoding is a simple and feasible scheme for error correction without retransmission. It is particularly suitable for use with a highly redundant alphabet. Like any such scheme it produces many more undetected errors than error correction by detection and retransmission,

TABLE VIII — ERROR RATES FOR THE TELEPHONE NETWORK*

n,k,e	P_E	P_D	$P_E + P_D$
47,24,5	4.65×10^{-6}	3.51×10^{-5}	4.08×10^{-5}
31,21,2	3.17×10^{-5}	3.73×10^{-5}	6.9×10^{-5}
31,20,2	1.23×10^{-5}	5.67×10^{-5}	6.9×10^{-5}
31,16,3	7.34×10^{-6}	3.99×10^{-5}	4.72×10^{-5}
31,15,3	3.06×10^{-6}	4.41×10^{-5}	4.72×10^{-5}
23,12,3	3.69×10^{-5}	0	3.69×10^{-5}
23,11,3	1.52×10^{-5}	2.17×10^{-5}	3.69×10^{-5}
21,12,2	1.82×10^{-5}	3.13×10^{-5}	5.05×10^{-5}
21,11,2	8.72×10^{-6}	4.18×10^{-5}	5.05×10^{-5}
17,9,2	2.36×10^{-5}	1.89×10^{-5}	4.25×10^{-5}
17,8,2	0.98×10^{-5}	3.27×10^{-5}	4.25×10^{-5}
15,7,2	1.83×10^{-5}	2.0×10^{-5}	3.83×10^{-5}

* P_E and P_D are approximate values. $P_E + P_D$ is exact.

but it is quite adequate for a channel in which $P(s,n)$ decreases rapidly as s increases. It is of very doubtful value on the telephone network as described by the Alexander, Gryb and Nast data.

ACKNOWLEDGMENTS

The author would like to express her gratitude to Miss Arline Mills, who wrote most of the computer programs and performed most of the calculations used in this paper.

APPENDIX A

Idempotents and Automorphisms of Cyclic Codes

We give first a short summary of the properties of cyclic alphabets. Let V be a finite field of characteristic q. Let V^n denote the direct sum of n copies of V.

Denote by $V[y]$ the ring of polynomials in y over the field V. Let $V[x] = V[y]/(y^n - 1)$ be the residue class ring of $V[y]$ mod $y^n - 1$. $V[x]$ consists of all polynomials of degree $\leqq n - 1$ with coefficients in V. Addition of polynomials is done as usual; to multiply two polynomials we multiply in the usual way and then reduce exponents mod n.

A subset $\mathcal{A}$ of polynomials of $V[x]$ is called an ideal if

$$\begin{aligned} &(i) \quad g_1, g_2 \in \mathcal{A} \Rightarrow \alpha_1 g_1 + \alpha_2 g_2 \in \mathcal{A}, \qquad \alpha_1, \alpha_2 \in V \\ &(ii) \quad g \in \mathcal{A} \Rightarrow xg \in \mathcal{A}. \end{aligned}$$

A polynomial is completely determined by its coefficients; it is possible, in fact, to identify $V[x]$ and V^n. However, it is convenient to regard them as separate entities, related by the (1-1) mapping

$$\alpha_0 + \alpha_1 x + \cdots + \alpha_{n-1} x^{n-1} \rightleftarrows \alpha_0, \alpha_1, \cdots, \alpha_{n-1}.$$

An ideal in $V[x]$ is, by property (i), a linear subspace of V^n. By property (ii) it is invariant under a cyclic permutation of coordinates; hence it is a cyclic alphabet in V^n. Conversely, a cyclic alphabet in V^n is an ideal in $V[x]$. We represent the ideal and the alphabet by the same letter, $\mathcal{A}$.

The ring $V[x]$ may be regarded as the group algebra of the cyclic group $1, x, \cdots, x^{n-1}$ over V. The group algebra is semi-simple provided that q does not divide n (Ref. 6, Section 10.8). In this case it is known that every ideal contains a polynomial e with the following properties

(*i*)	e is idempotent.	$(e^2 = e)$
(*ii*)	e is a unit for $\mathfrak{A}$.	$(a \in \mathfrak{A} \Rightarrow ae = a)$
(*iii*)	e generates $\mathfrak{A}$.	($\mathfrak{A}$ consists of all polynomials $fe, f \in V[x]$).

e will be called the generating idempotent of $\mathfrak{A}$.

An automorphism σ of $V[x]$ is a (1-1) mapping of $V[x]$ onto itself which respects both addition and multiplication. If $v_1, v_2 \in V[x]$, then

$$(v_1 + v_2)\sigma = v_1\sigma + v_2\sigma$$

$$(v_1v_2)\sigma = (v_1\sigma)(v_2\sigma).$$

Lemma 1.1: An automorphism σ of $V[x]$ preserves an ideal $\mathfrak{A}$ if and only if σ preserves the generating idempotent e of $\mathfrak{A}$.

Proof: Suppose that σ preserves e. Then $\mathfrak{A}\sigma = V[x]\cdot e\sigma = V[x]\cdot e = \mathfrak{A}$.

Suppose that σ preserves $\mathfrak{A}$. Then $e\sigma \in \mathfrak{A}$, and $e\sigma e = e\sigma$ by property (*ii*).

Let b be the element of $\mathfrak{A}$ such that $b\sigma = e$. Then

$$e\sigma e = e\sigma b\sigma = (eb)\sigma = b\sigma = e.$$

Hence $e = e\sigma e = e\sigma$, and σ preserves e.

Lemma 1.2: If q (the characteristic of V) is relatively prime to n (the block length of $\mathfrak{A}$), then the mapping $\sigma: x^i \rightarrow x^{iq}$ is an automorphism of $V[x]$.

Proof: Clearly this mapping respects addition and multiplication in $V[x]$. We have only to show that it is 1-1.

If $x^{iq} = x^{jq}$, then $(i - j)q \equiv 0 \bmod n$. Since q is prime to n, this implies that $i - j \equiv 0 \bmod n$ or $x^i = x^j$.

This proves the lemma.

Theorem 1.3: If q is prime to n, every ideal in $V[x]$ is preserved by the mapping $\sigma: x^i \rightarrow x^{iq}$.

Proof: Let $\mathfrak{A}$ be an ideal in $V[x]$ and $e = \sum_{i=0}^{n} \alpha_i x^i$, the generating idempotent of $\mathfrak{A}$.

$$\sigma e = \sum_{i=0}^{n} \alpha_i x^{iq} = \left(\sum_{i=0}^{n} \alpha_i x^i\right)^q = e^q,$$

since q is the characteristic of V.

$e = e^2 = e^3 = \cdots = e^q$; hence σ preserves e, and by Lemmas 1.1 and 1.2, σ preserves $\mathfrak{A}$.

Let the coordinate places of V^n be labeled $0, 1, \cdots, n - 1$; the mapping $\sigma: x^i \rightarrow x^{iq}$ in $V[x]$ corresponds to the permutation $U_q: \omega \rightarrow q\omega$ of coordinate places in V^n.

Corollary: Every binary cyclic alphabet of odd block length is preserved by the permutation $U: \omega \rightarrow 2\omega$.

This is Theorem 1 of Section II. The proof given here contains more machinery than is necessary to prove Theorem 1. This is done on purpose, in order to get Lemma 1.1, which suggests a method of finding other automorphisms of $V[x]$ which preserve a particular cyclic alphabet.

APPENDIX B

Distribution of Weights in the Cosets of a Group Code

Let $\mathcal{A}$ be an (n,k,e) binary alphabet, and let $\mathcal{B}$ be the orthogonal complement (dual alphabet) of $\mathcal{A}$ in V^n. Let $A(i)$, $B(i)$ denote the number of letters of weight i in $\mathcal{A}$, $\mathcal{B}$ respectively. The quantities $A(i)$, $B(i)$ are connected by the generating function.[7]

$$\sum_{i=0}^{n} A(i)(1+z)^{n-i}(1-z)^i = 2^k \sum_{i=0}^{n} B(i)z^i.$$

Since the $A(i)$ are known, the $B(i)$ may be calculated from this relationship.

Set

$$(1+z)^{n-i}(1-z)^i = \sum_{j=0}^{n} \psi(i,j)z^j.$$

Let $C(s,j)$ denote the number of sequences of weight s in V^n which are at distance j from some letter of $\mathcal{A}$. Then if $j \leqq e$, $C(s,j)$ and $B(i)$ are related by the generating function.[7]

$$\sum_{i=0}^{n} B(i)\psi(i,j)(1+x)^{n-i}(1-x)^i = 2^{n-k} \sum_{s=0}^{n} C(s,j)x^s.$$

Since $B(i)$ and $\psi(i,j)$ are known, $C(s,j)$ may be calculated from this relation.

Clearly

$$C(s) = \sum_{j=0}^{e} C(s,j).$$

REFERENCES

1. Slepian, D., A Class of Binary Signaling Alphabets, B.S.T.J., **35**, January, 1956, p. 634.

2. Neumann, P. G., A Note on Cyclic Permutation Error-Correcting Codes, Information and Control, **5**, March, 1962, p. 72.
3. Prange, E., The Use of Information Sets in Decoding Cyclic Codes, IRE Trans., IT-**8**, September, 1962, p. S-5–9.
4. Prange, E., An Algorithm for Factoring $x^n - 1$ over a Finite Field, Air Force Cambridge Research Center, AFCRC-TN-59-775.
5. Elliott, E. O., Estimates of Error Rates for Codes on Burst-Noise Channels, B.S.T.J., **42**, September, 1963, p. 1977.
6. Curtis, C. W., and Reiner, I., *Representation Theory of Finite Groups and Associated Algebras*, Interscience Publishers, New York, 1962.
7. MacWilliams, J., A Theorem on the Distribution of Weights in a Systematic Code, B.S.T.J., **42**, January, 1963, p. 79.