X. PROCESSING AND TRANSMISSION OF INFORMATION

Prof. P. Elias
Prof. R. M. Fano
Prof. D. A. Huffman
Prof. C. E. Shannon
Dr. M. V. Cerrillo
W. G. Breckenridge
E. Ferretti
K. Joannou
R. M. Lerner
J. B. O'Loughlin
L. S. Onyshkevych
A. J. Osborne
W. W. Peterson
E. T. Schoen
F. F. Tung
S. H. Unger
M. L. Wetherell
J. M. Wozencraft
W. A. Youngblood

A. MULTIPLE-GROUND CODING IN ITERATIVE SWITCHING CIRCUITS

One of the characteristic features of a combinational iterative switching circuit is that the conditions necessary for the transmission of a ground to the output terminal of the circuit do not, in general, depend upon the number of binary switching variables involved. Huffman (1) showed that the switching characteristics of an iterative circuit can be completely described by a flow table.

For example, the iterative circuit of Fig. X-1 transmits a ground to its output terminal only if no relays or if two or more relays are operated. An inspection of the relay contact network reveals the iterative or periodic nature of its structure. The repeated subnetwork is called a "typical cell."

The flow table for the typical cell is given in Fig. X-2. Each cell need only "know" if

1. no relays to the left are operated (designated state "1"), or
2. just one relay to the left is operated (state "2"), or
3. two or more relays to the left are operated (state "3").

The way the typical cell "knows" which of these three mutually-exclusive states exists for the network to the left is by means of a binary code that is presented to the input leads of the cell as the presence (or absence) of a set of grounds on these leads.

In our example above a "single-ground code" was assigned to each state of the flow table. With an assignment like this, a ground may appear on one and only one of the leads between each cell. When more than one lead can be grounded at a time, the code is called a "multiple-ground code." The use of these codes will be discussed later.

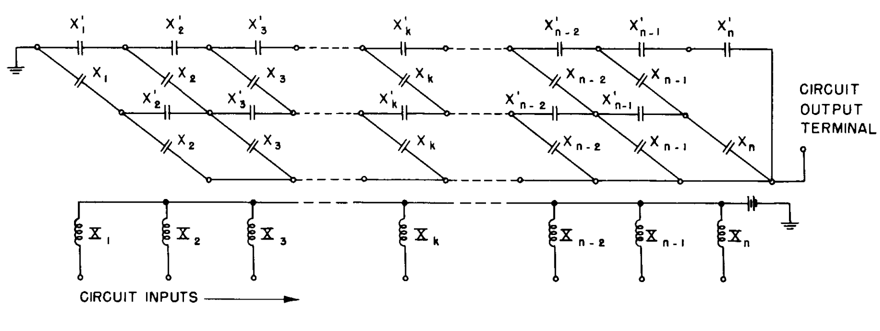

Fig. X-1. An illustrative iterative switching circuit.

FLOW TABLE / ASSIGNED CODE

STATE	X_k = 0	X_k = 1	OUTPUT	SINGLE GROUND CODE
1	1	2	1	1 0 0
2	2	3	0	0 1 0
3	3	3	1	0 0 1

Fig. X-2. Flow table and assigned code for the illustrative iterative circuit.

FLOW TABLE / CODE ASSIGNMENTS

STATE	X_k = 0	X_k = 1	OUTPUT	SINGLE GROUND CODE	MULTIPLE GROUND CODE
1	1	2	0	1 0 0 0 0 0	1 1 0 0
2	3	6	1	0 1 0 0 0 0	1 0 0 1
3	2	4	1	0 0 1 0 0 0	0 1 0 1
4	5	3	0	0 0 0 1 0 0	1 0 1 0
5	4	1	0	0 0 0 0 1 0	0 1 1 0
6	6	5	1	0 0 0 0 0 1	0 0 1 1

Fig. X-3. Flow table with single-ground and multiple-ground code assignments.

A typical cell can be synthesized from any flow table if a single-ground code assignment is used. A universal synthesis procedure of this sort, however, requires one lead (and its associated relay contacts) between typical cells for each row (state) in the flow table. The next example will illustrate how this restriction can be overcome with multiple-ground coding.

Let us assume that the switching requirements for an iterative circuit which we want to synthesize are given to us in the form of the flow table in Fig. X-3. First a check should be made in order to be certain that all equivalent (2) and impossible states have been removed from the table. (An impossible state is any state which, starting from state "1," can never be reached by any combination of the circuit input variables. Therefore, it should be removed from the flow table before we proceed.) Then a binary code is assigned to each state.

When the single-ground code assignment (column 4 of Fig. X-3) is made, the iterative circuit may be synthesized directly by standard techniques (1). The typical cell derived for this code assignment is shown in Fig. X-4a. The first input lead of the first cell is grounded so that the cell starts off in state "1." The second, third, and sixth output leads of the last cell (as specified by the "ones" in the output column of the flow table) are connected to the circuit output terminal. There are six leads between cells, and six normally closed and six normally open relay contacts (a total of 18 springs) are required per cell.

The author has found that several important features result when a "conservative multiple-ground code" assignment (column 5 of Fig. X-3) is made. (By "conservative" we mean that the binary code assigned to any one state in the flow table must be a permutation of the binary code assigned to any other state in the table.) An assignment like this implies that, if we initially ground "g" input leads of the iterative circuit, a ground must always appear on "g" of the leads between any two cells in the network, regardless of which relays or how many of them are operated.

The typical cell for this conservative code assignment is derived in the following manner. From the flow table in Fig. X-3, we see that when the input variable x_k has the value "0" (the k^{th} relay is not operated), and when a ground appears on the first and second input leads of the k^{th} cell (state "1"), a ground must appear on the first and second output leads of the same cell. This tells us

1. that the normally closed (primed) contacts within the typical cell must connect the first and second input leads either to the first and second or to the second and first output leads, respectively.

Likewise the flow table demands that state "2" (code 1001) be transformed into state "3" (code 0101). This implies

2. that input leads one and four must be connected either to the second and fourth or to the fourth and second output leads of the cell, respectively.

Clearly, requirements one and two can be met only when (see Fig. X-5a) input leads one, two, and four are connected by normally closed relay contacts to output leads two, one, and four, respectively. The third input lead must be connected to the third output lead by a normally closed relay contact because, by the definition of the conservative code assignment, we may never lose a ground that might appear on any of the input leads. Thus, we are given the derived contact structure of the typical cell when the input variable x_k has the value "0."

When x_k has the value "1" (the k^{th} relay is operated), reasoning similar to that presented above will show that the normally open relay contact structure in Fig. X-5b gives the required transformations for each state of the flow table. Any two

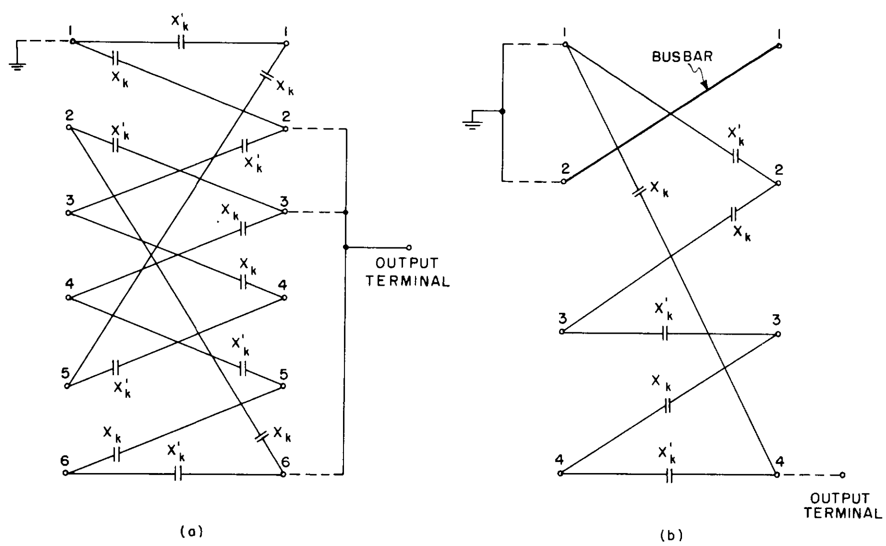

Fig. X-4. Derived typical cells for: (a) single-ground; and (b) multiple-ground code assignments.

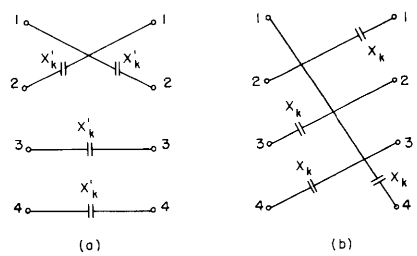

Fig. X-5. Relay contact subnetworks derived for the multiple-ground code assignment: (a) when $x_k = 0$; (b) when $x_k = 1$.

transformations in the "x_k equals one" column of Fig. X-3 can be used in the derivation.

The two relay contact structures are then combined, as in Fig. X-4b, to form the complete typical cell that was derived on the basis of the conservative code assignment. A bus bar is used to connect the second input lead to the first output lead of the typical cell because our synthesis procedure requires the paralleling of normally closed and normally open relay contacts between these two leads.

The end connections to the first and last cells in the chain must be made in such a way that the switching action at the output terminal of the circuit follows the requirements stated in the output column of the flow table in Fig. X-3. An inspection of these output requirements reveals that they correspond exactly to the last digit of each binary code in the multiple-ground assignment. Consequently, only the fourth output lead of the last cell is connected to the circuit output terminal. The first and second input leads of the first cell are grounded so that state "1" (code 1100) is presented to the input of the first cell. As it can be seen in Fig. X-4b, there are only four leads between cells, and three normally open and three normally closed relay contacts (nine springs) are required per cell.

Two iterative circuits, one made up entirely of typical cells in Fig. X-4a, the other from typical cells in Fig. X-4b, will have exactly the same end-to-end switching characteristics. Nevertheless, the iterative circuit derived on the basis of a conservative multiple-ground code requires two fewer leads between cells and only half the number of relay contacts per cell, as compared with the iterative circuit derived on the basis of the commonly used single-ground code. In this example, we clearly see a distinct advantage in using the multiple-ground code. The reduction in the number of leads that are required to connect typical cells might conceivably be more important than the saving of relay contacts, if our cells had to be located any great distance from one another.

Unfortunately, the conservative assignment is not applicable to every possible flow table, whereas the single-ground code assignment is. Exactly when and how the ground-conserving code assignment can be used in the synthesis of a general flow table will be the subject of a future report. Our purpose in the present paper is to inform the reader of its existence and to demonstrate some of its advantages.

To the best of his knowledge, the author is the first one to apply the conservative multiple-ground code to iterative switching circuits and he is presently engaged in the further study of conservative and other more general multiple-ground coding procedures.

J. B. O'Loughlin

References

1. D. A. Huffman, Quarterly Progress Report, Research Laboratory of Electronics, M.I.T., Jan. 15, 1955, p. 63.

2. D. A. Huffman, J. Franklin Inst. 257, 3 (March 1954), p. 184.

B. HAZARDS IN SEQUENTIAL SWITCHING CIRCUITS

This report is concerned with asynchronous sequential switching circuits of the type discussed in references 1 and 2. Systems of this kind can be represented by the block diagram of Fig. X-6, which corresponds to Fig. 4 in reference 2, page 2. Note that the term "state device" is used here in place of the word "delay" (2).

As a first approximation, it is generally assumed that the combinational network of Fig. X-6 is ideal, in that a change in its input is instantly followed by an appropriate output change. However, in any real situation, inevitable delays in the various components and leads prevent the output from responding immediately. Even more important is the fact that a lack of uniformity in the stray delays can result in the appearance of momentary false outputs at some of the terminals. Furthermore, the effects of the input change might not be felt simultaneously at all of the responding terminals.

Two types of malfunction can result from these departures from ideal behavior. The first, which we shall call a transient hazard, is present if, following certain input changes, temporary, false signals appear at the system output. The second, which we shall refer to as a steady-state hazard, is present, if there is a possibility that, on

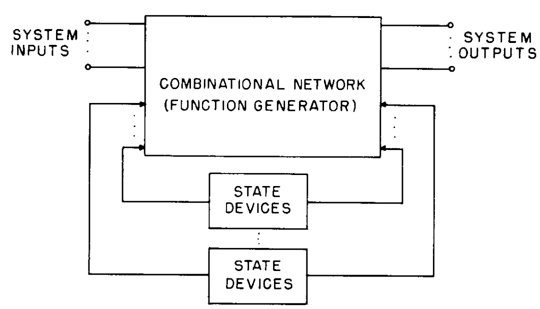

Fig. X-6. Block diagram of general sequential circuit.

INTERNAL STATE	INPUT STATE x_1	x_2
1	①	2
2	3	②
3	③	x

Fig. X-7. Flow table sector illustrating steady-state hazard.

INTERNAL STATE	INPUT STATE x_1	x_2
1	①	2*
2	②*	②*

(a)

INTERNAL STATE	INPUT STATE x_1	x_2
1	①	2*
2	3*	②*
3	③*	2*

(b)

Fig. X-8. Flow table sectors illustrating transient hazards.

account of stray delays, the internal state of the system is incorrect after some input change.

Smoothing, or inertial, devices (low-pass filters) connected at the inputs to each state device prevent the latter from hastily responding to transient, false signals from the combinational circuit, thereby eliminating steady-state hazards. Additional smoothers at the output terminals will eliminate transient hazards also. Devices of this sort always act as delays as well as smoothers. (Note that although smoothing implies delay, the converse is not true. A dead-time delay, which translates a signal f(t) into f(t-d), does not have any smoothing effect.)

It is possible to construct a smoothing element with a dead-time delay and combinational elements (3), in such a way that hazards can be circumvented without using low-pass filters. However, even this simulated smoother is unnecessary. It can be shown that a sequential switching circuit is safe (hazard-free) if dead-time delays are incorporated in each state device, and if all of its combinational circuits are hazard-free in the sense that false signals never appear at any of their output terminals. Reference 4 includes a detailed discussion of hazard-free combinational circuits and a proof that they can be constructed for any table of combinations.

Examples can be found of circuits that will function correctly even if delays of arbitrary value (including zero) are associated with each lead and element (including the state devices). In view of the obvious desirability of minimizing delays in high-speed logical systems, it is natural to inquire whether circuits cannot always be so synthesized that we can dispense with special delays at the state devices and still retain safe operation. Unfortunately the answer is no.

Any circuit described by a flow table that contains a section of the kind shown in Fig. X-7 will have a steady-state hazard unless delays are incorporated in some of the state devices. The input states corresponding to the columns of Fig. X-7 differ in one variable, and the x-entry represents either a stable state or a transition leading to a stable state other than 2. A flow table which does not have a section of this kind can always be realized by a circuit that will be inherently free of steady-state hazards regardless of the distribution of delays in the network.

The necessary and sufficient conditions for a flow table to be realizable by a circuit that is inherently free of transient, as well as steady-state, hazards consist of the preceding condition and the following constraints.

In any section of the kind shown in Fig. X-8a and b, the output states must be the same for all starred entries. (Again the columns of the tables differ in one input variable.)

Further discussion of the hazard problem and proofs of these statements will be presented in a later report.

S. H. Unger

References

1. D. A. Huffman, J. Franklin Inst. 257, Nos. 3 and 4 (March, April, 1954).
2. D. A. Huffman, Technical Report 293, Research Laboratory of Electronics, M.I.T., March 14, 1955.
3. D. A. Huffman, Quarterly Progress Report, Research Laboratory of Electronics, M.I.T., Oct. 15, 1955, p. 60.
4. D. A. Huffman, Design of hazard-free switching networks, Association for Computing Machinery Conference, 1955.

C. SYMMETRIC SWITCHING FUNCTION IDENTIFICATION BY MEANS OF DECIMAL PROCEDURES

Shannon's (1) general definition of a symmetric function is: "A function of the n variables $X_1, X_2, \ldots, X_n$ is said to be symmetric in these variables if any interchange of the variables leaves the function identically the same." Another way of specifying a symmetric function is to use the a-number theorem originally stated by Shannon (1, 2):

"A necessary and sufficient condition that a function be symmetric is that it may be specified by stating a set of numbers, $a_1, a_2, \ldots, a_k$ such that if exactly $a_j (j = 1, 2, 3, \ldots, k)$ of the variables have the value one, then the function has the value one, and not otherwise .... The set of numbers $a_1, a_2, \ldots, a_k$ may be any set of numbers selected from the numbers 0 to n, inclusive, where n is the number of variables in the symmetric function. For convenience, they will be called the a-numbers of the function."

For example, the function

$$X_3'X_2X_1X_o + X_3X_2'X_1X_o + X_3X_2X_1X_o' + X_3X_2X_1'X_o$$

has the value one when any <u>three</u> of the variables X_3, X_2, X_1, X_o have the value one. It has, therefore, the a-number 3 and is written as

$$S_3(X_3, X_2, X_1, X_o)$$

It is not necessary for the variables of symmetry to be all primed or all unprimed. Symmetric functions do exist in which some of the variables of symmetry are primed and some unprimed; these functions are said to be symmetric in mixed variables (2). For example, the function

$$X_3'X_2'X_1X_o + X_3'X_2X_1'X_o + X_3'X_2X_1X_o' + X_3X_2X_1X_o$$

has the value one when any three of the variables X_3', X_2, X_1, X_o have the value one. Accordingly, it is written as

$$S_3(X_3', X_2, X_1, X_o)$$

Symmetric switching functions lead to networks that are very economical in terms of the number of switching elements used. It is important, therefore, to be able to recognize and identify symmetric functions in the synthesis of switching networks. The problem in the study of symmetric functions is: Given a switching function, $T(X_{n-1}, X_{n-2}, \ldots, X_o)$ in n variables, determine whether or not the function is symmetric and, if so, what the associated variables of symmetry and the a-numbers are.

A switching function may be expressed in a number of ways. One way is the writing of the Boolean algebra expression as sums of products. Another way is the listing of the binary numbers that are formed when each of the products has its unprimed literals replaced by a one and its primed literals replaced by a zero. A third method is the listing of the decimal number equivalents of these binary numbers and prefixing the list with a summation sign in order to indicate that the numbers represent sums of products in a Boolean expression.

$$T(X_3, X_2, X_1, X_o) = X_3'X_2'X_1'X_o' + X_3'X_2X_1'X_o + X_3X_2'X_1X_o' + X_3X_2X_1X_o$$

can also be expressed as shown in Fig. X-9.

X_3	X_2	X_1	X_0
0	0	0	0
0	1	0	1
1	0	1	0
1	1	1	1

(a)

$$T(X_3, X_2, X_1, X_0) = \sum (0, 5, 10, 15)$$

(b)

Fig. X-9. Switching function expressions.

$$X_{n-1} \cdots X_j \cdots X_k \cdots X_1 X_0$$

$$\begin{array}{|l|} \hline \cdots \\ C_{n-1} \cdots C_j \cdots C_k \cdots C_1 C_0 \\ \\ \\ \hline \end{array}$$

$$K_{n-1} \cdots K_j \cdots K_k \cdots K_1 K_0$$

Fig. X-10. Tabular form for $T(X_{n-1}, X_{n-2}, \ldots, X_o)$.

The purpose of this discussion is to establish the symmetry of the function and then to determine the variables of symmetry and the a-numbers, assuming an expression of the type shown in Fig. X-9b. The methods will only involve operations with the decimal numbers. First of all, to motivate the decimal procedures we must consider the properties of symmetric functions when they are expressed as in Fig. X-9a (3). The function $T(X_{n-1}, X_{n-2}, \ldots, X_o)$, of n variables, will be expressed as shown in Fig. X-10. If T is the sum of p product terms, then Fig. X-10 will be a p × n matrix. $C_{n-1} \ldots C_j \ldots C_k \ldots C_1 C_o$ is a typical row, where the C's are either one or zero. Associated with each of the columns is a fraction K_j, called the coindex (3), which is the ratio of the number of ones to the number of zeros in the j^{th} column.

If we use the general definition of a symmetric function, a necessary condition for a function to be symmetric can be established. Assume that T is symmetric. Then any interchange of variables should produce the same function. If we interchange X_j and X_k, then K_j and K_k are also interchanged and the typical row becomes

$$C_{n-1} \ldots C_k \ldots C_j \ldots C_1 C_o$$

Since the function is unchanged, this new row must have been present in the matrix before the interchange of X_j and X_k. Hence the interchange must make that row become $C_{n-1} \ldots C_j \ldots C_k \ldots C_1 C_o$. More simply, if T is symmetric, an interchange of variables will merely interchange rows of the matrix. Since an interchange of X_j and X_k causes K_j and K_k to be interchanged, and since it has been shown that any coindex must remain the same if T is symmetric, then all coindices are equal. A necessary condition for a function to be symmetric is that all of its coindices are equal. It should be stated at this point that if X_j is primed in the set of symmetric variables, then this will cause K_j to be inverted. Hence, if some of the coindices must be inverted before all are equal, then the function may be symmetric in both primed and unprimed variables.

If T is symmetric with a-number a_j, then there must be $n!/\left[(n - a_j)!\, a_j!\right]$ rows in the matrix, each containing a_j ones. Since T must remain the same for all possible interchanges of variables, $n!/[(n - a_j)!\,a_j!]$ rows, each of which contains a_j ones, will form a "complete group of rows," sufficient to guarantee it. The equality of coindices and the fact that there are ${}_nCa_j$ rows, each of which has a_j ones, are the necessary and sufficient conditions for T to be symmetric with a-number a_j.

To detect symmetry by using the decimal numbers that specify the function in $T = \sum (d_i)$, it is only necessary to notice a property of positional number systems and then apply it to the previously mentioned conditions. If the decimal integer d is expressed in the base R, then

$$d = \sum_{i=0}^{n} C_i R^i$$

If we then define $p_i = [d/R^i]$, it is easy to show that $p_i \equiv C_i \bmod R$. In particular, if d is expressed in binary form, then C_i will be a one or a zero, depending upon whether or not p_i is odd or even, respectively.

The decimal method involves the use of an $n \times p$ matrix, where n is the number of variables in T and p is the number of decimal indicators in standard sum form. The first row of the $(n \times p)$ matrix contains the decimal indicators present in the standard sum form. (Ordering is of no importance here.) The rest of the matrix is then formed in accordance with the following rule: If $d_{i+1,j}$ is an element of the (i+1) row and j^{th} column of the matrix, then

$$d_{i+1,j} = \left[\frac{d_{i,j}}{2}\right]$$

Next, associate with each of the n rows a coindex K_i, where the coindex of the i^{th} row is equal to the ratio of odd to even numbers that appear in that row. If the coindices are all the same or can be made all the same by inverting some of them, then the function may be symmetric. If certain of the coindices must be inverted in order to make all the coindices equal, then it indicates a possibility of mixed-variable symmetry. Now assoiciate with the j^{th} column a number a_j which is equal to the number of odd entries of the j^{th} column that are not in primed rows plus the number of even entries in the j^{th} column that are in primed rows. If the number a_j appears ${}_nCa_j$ times, then T is symmetric, with a-number a_j. The variables of symmetry can be obtained for $T(X_{n-1}, X_{n-2}, \ldots, X_i, \ldots, X_o)$ by associating X_i with the (i+1) row and priming X_i if the (i+1) row has an inverted coindex.

The following examples will illustrate the method that has been discussed.

1. $T(X_4, X_3, X_2, X_1, X_o) = \sum (0, 1, 4, 8, 10, 12, 22)$

0	1	4	8	10	12	22	1/6	X_o
0	0	2	4	5	6	11	2/5	X_1
0	0	1	2	2	3	5	3/4	X_2
0	0	0	1	1	1	2	3/4	X_3
0	0	0	0	0	0	1	1/6	X_4

Since the coindices are not equal, this function cannot be symmetric.

2. $T(X_3, X_2, X_1, X_o) = \sum (3, 5, 6, 9, 10, 12, 15)$

3	5	6	9	10	12	15	4/3	X_o
1	2	3	4	5	6	7	4/3	X_1
0	1	1	2	2	3	3	4/3	X_2
0	0	0	1	1	1	1	4/3	X_3
2	2	2	2	2	2	4		

Since ${}_4C_2 = 6$ and ${}_4C_4 = 1$, we can say that $T(X_3, X_2, X_1, X_o) = \sum (3, 5, 6, 9, 10, 12, 15) = S_{2,4}(X_3, X_2, X_1, X_o)$.

3. $T(X_4, X_3, X_2, X_1, X_o) = \sum (1, 4, 7, 8, 11, 13, 14, 18, 21, 25, 28, 31)$

1	4	7	8	11	13	14	18	21	25	28	31	7/5	X_o
0	2	3	4	5	6	7	9	11	12	14	15	5/7	X_1
0	1	1	2	2	3	3	4	5	6	7	7	7/5	X_2
0	0	0	1	1	1	1	2	2	3	3	3	7/5	X_3
0	0	0	0	0	0	0	1	1	1	1	1	5/7	X_4
3	3	3	3	3	5	3	0	3	3	3	3		

This function is seen to be $S_{o,3,5}(X_4', X_3, X_2, X_1', X_o)$ when the coindices of the second and fifth rows are inverted.

4. A seven-variable problem is included to show that although the matrices become larger as the number of variables increases, no conceptual difficulties are involved. See Fig. X-11.

$T = \sum(11, 19, 25, 26, 27, 31, 36, 59, 67, 73, 74, 75, 79, 81, 82, 83, 87, 88, 89, 90, 93, 94, 95, 107, 115, 121, 122, 123, 127)$

11	19	25	26	27	31	36	59	67	73	74	75	79	81	82	83	87	88	89	90	93	94	95	107	115	121	122	123	127	21/8	X_o
5	9	12	13	13	15	18	29	33	36	37	37	39	40	41	41	43	44	44	45	46	47	47	53	57	60	61	61	63	21/8	X_1
2	4	6	6	6	7	9	14	16	18	18	18	19	20	20	20	21	22	22	22	23	23	23	26	28	30	30	30	31	8/21	X_2
1	2	3	3	3	3	4	7	8	9	9	9	9	10	10	10	10	11	11	11	11	11	11	13	14	15	15	15	15	21/8	X_3
0	1	1	1	1	1	2	3	4	4	4	4	4	5	5	5	5	5	5	5	5	5	5	6	7	7	7	7	7	21/8	X_4
0	0	0	0	0	0	1	1	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3	8/21	X_5
0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	21/8	X_6
5	5	5	5	6	5	0	5	5	5	5	6	5	5	5	6	5	5	6	6	5	5	6	5	5	5	5	6	5		

Coindices indicate possibility of total symmetry, suggest inverting, hence priming, X_2 and X_5.

$$\left.\begin{array}{l} \text{1 zero} \\ \text{21 fives} \\ \text{7 sixes} \end{array} \left\{\begin{array}{l} {}_7C_o = 1 \\ {}_7C_5 = 21 \\ {}_7C_6 = 7 \end{array}\right\}\right. \text{ a-numbers are } 0, 5, 6$$

Therefore, $T = S_{o,5,6}(X_6, X_5', X_4, X_3, X_2', X_1, X_o)$

Fig. X-11. A function of seven variables.

5. Situations may arise in examining $T = \sum (d_i)$ for symmetry in which the coindices all have equal numerators and denominators. Now, if the function is symmetric in mixed variables, there is no apparent way of telling which of the coindices to invert. For example, consider $T(X_3, X_2, X_1, X_o) = \sum (1, 2, 4, 11, 13, 14)$, which is known to be symmetric.

1	2	4	11	13	14	3/3	X_o
0	1	2	5	6	7	3/3	X_1
0	0	1	2	3	3	3/3	X_2
0	0	0	1	1	1	3/3	X_3
1	1	1	3	3	3		

The ${}_nCa_j$ criterion is not satisfied, since some of the coindices must be inverted to make the symmetry evident. The difficulty arises because p is even. Therefore, select some variable, say X_1, about which an expansion can be made by forming a new column of coindices. Then delete row X_i and form new coindices for the remaining rows by counting odd and even entries only in those columns for which X_i has an odd entry. The resulting matrix is:

1	2	4	11	13	14	3/3	1/2	X_o
~~0~~	~~1~~	~~2~~	~~5~~	~~6~~	~~7~~	~~3/3~~		~~X_1~~
0	0	1	2	3	3	3/3	1/2	X_2
0	0	0	1	1	1	3/3	2/1	X_3

↑ new coindices (column 1/2, 1/2, 2/1)

The new coindices suggest priming X_3; if this is done, the function will be $S_2(X_3', X_2, X_1, X_o)$.

W. G. Kellner

References

1. C. E. Shannon, Trans. AIEE 57, 713 (1938).
2. S. H. Caldwell, Trans. AIEE, Part I. Communication and Electronics, 73, 142 (May 1954).
3. E. J. McCluskey, Jr., Summer Session Course 6.567 Lecture, M.I.T., 1955.