MORE ABOUT THRESHOLD LOGIC

R. O. Winder

RCA Laboratories

Princeton, New Jersey

ABSTRACT

We pursue in this paper some of the ideas discussed a year ago at the First Annual Symposium on Switching Theory and Logical Design. For a general discussion of threshold logic, and for definitions and motivations of the terms used below, the reader is referred to "Single Stage Threshold Logic" also published in this volume* [13] . Also, a general survey of recent papers in the subject has been published elsewhere [14] .

The main subject treated below is compound synthesis. The importance of such a study was shown last year: The family of functions of n arguments realizable in a single stage becomes a vanishing fraction of all switching functions of n arguments as n grows (for n = 7 the ratio is about $10^{28\ 1/2}$). We provide an algorithm for determining "2-realizability" — realizability with two threshold elements. The general approach produces a good solution in any case, but one guaranteed optimal only for 2-realizable functions. We use here a geometric terminology; this new language is also used in the second section, where "higher" necessary conditions for realizability are discussed. A conjecture that certain of these conditions might be sufficient is disproved; three related conditions are treated in a common language. The final section considers optimal integral single-stage realizations, and disproves a conjecture made last year: That such a realization gives equal arguments equal weights.

I. COMPOUND SYNTHESIS

INTRODUCTION

Various problems of compound synthesis can be distinguished, involving various restrictions and optimality criteria:

1. The general problem. Realize a given function with a network of fewest threshold gates.

2. The timing problem. As (1), but restrict the network to the fewest levels (to minimize time delay).

3. The practical problem. As (1), but restrict the network by requiring $\sum |a_i| \leq M$ a fixed maximum, for each gate (to prevent component and signal variations from causing errors).

Other criteria enter with perhaps less weight: The loading of the input signals should be even, delays used to bring input signals to the lower gates in the network should be considered, the number of gates a given gate feeds may need to be restricted, networks with regular patterns are easier to manufacture and service. These and the two restrictions above can be combined in various ways to give a variety of problems.

Stating criteria and restrictions implicitly is very important; see for instance the paper of R. C. Minnick [9] , where improvements are claimed over results of S. Muroga [10] and are actually possible because of a relaxation of conditions Muroga failed to make explicit. (Namely, that delays be counted, too.)

Problem (3) has received some attention. M. Cohn and R. Lindaman [3,6,7] treat the case of M = 3, where the gates then must all be simple majority gates. The author also has some results which will be reported elsewhere. Here we wish to discuss a first step in the directions of problems (1) and (2): We give an explicit algorithm for determining whether two threshold gates suffice to realize a given function:

$$f(\vec{x}) = g(\vec{x}, h(\vec{x})) \qquad (1)$$

where g and h are threshold functions (f and h have n arguments, g has n+1). Algorithms for the case of one-gate realizability (or simply 1-realizability) are well known. The most relevant of these, reported by the author last year [13] , provides the basis for the 2-realizability algorithm to be presented below. The philosophy of this approach is:

See pages 321-333

In general, we want to realize a function as simply as possible. So first we try to realize it with one gate (we check 1-realizability). If this attempt fails, we use the data so obtained to try with two gates (we check 2-realizability). Supposing this attempt also fails, then we continue to add gates, using the data so far collected, until the function is finally realized.

THE GEOMETRIC LANGUAGE

The following geometric interpretation of algebraic ideas will be useful here and in later reports:

Definition: Given n fixed, and X a valuation, then [X] is defined as a corresponding family of vectors* (in n-space) as follows: Let Y be any valuation of all the arguments not involved in X (so XY gives values to all arguments, and thus defines a point on the n-cube). Let [X] be the family of all vectors parallel to and of the same length as the vector from $\bar{X}$Y to XY. (So the family is independent of the choice of Y.)

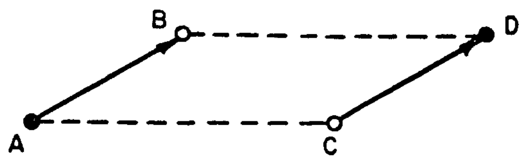
Definition: A vector, running from a point A to a point B in the n-cube, is f-negative when A ε f and B ∉ f (i.e., A belongs f and B doesn't), f-positive when A ∉ f and B ε f; all other vectors are f-neutral.

Definition: A valuation X is f-positive when [X] contains no f-negative vectors, f-negative when [X] contains no f-positive vectors, f-proper when it is either f-positive or f-negative, and otherwise f-improper. Any family of vectors is similarly, f-positive, -negative, -proper, and -improper according as to the vectors it contains.

These definitions connect back to earlier algebraic results [13] as follows:

Theorem: f is positive in X if and only if [X] is f-positive. f is completely monotonic if and only if every valuation is f-proper.

To complete the cycle: This last condition requires simply that there be no pair of equal-length, common-direction vectors, A ⟶ B and C ⟶ D, such that A and D belong to f while B and C do not. The argument that complete monotonicity is a condition necessary for realizability should now be clear: f is realizable in this language when there is a hyperplane separating its points from its non-points. If ABCD form a parellelogram as above, clearly no hyperplane can put A and D on one side, B and C on the other.



(Because the hyperplane must intersect the 2-plane defined by ABDC in a 1-plane which separates B and C from A and D — clearly impossible.)

This new geometric language will facilitate discussion of some interesting properties of switching functions in the next section. More important, it allows a geometric approach to compound synthesis, and provides a connection with previous (algebraic) results, which in turn make it possible to establish geometrically reasonable arguments rigorously. The author feels that the complexity of compound synthesis problems require that geometric intuition be used as a tool. No mechanical algebraic procedure is likely to be feasible in cases of many arguments and complicated functions.

Now, a geometric picture of the 2-realization algorithm: Suppose f is displayed in the usual fashion on an n-cube. Then equation (1) can be pictured as follows: $h(\bar{x})$ divides the n-cube into two sections. In the two sections, f is realized by g_1 and g_2, which are two versions of g obtained by parallel translation (adding the weight of g's $(n+1)^{st}$ argument or not, in case $h(\bar{x})$ is T or F) (see Fig. 1). In other words, g, displaced along a fault line h, separates vertices belonging to f from those not belonging.

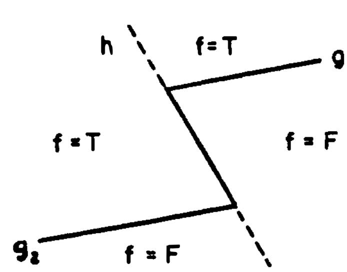


FIG.1

---

*In this section we shall consider only Boolean vectors, vectors whose two end points are vertices of the unit cube.

As an example consider the following realization of $x \oplus y$:

$$x \oplus y = x\bar{y} + \bar{x}y = S(-2 + 4.\sigma(x) + 3.\sigma(y) - 6.\sigma S(-4 + 2.\sigma(x) + 3.\sigma(y))).$$

Geometrically:

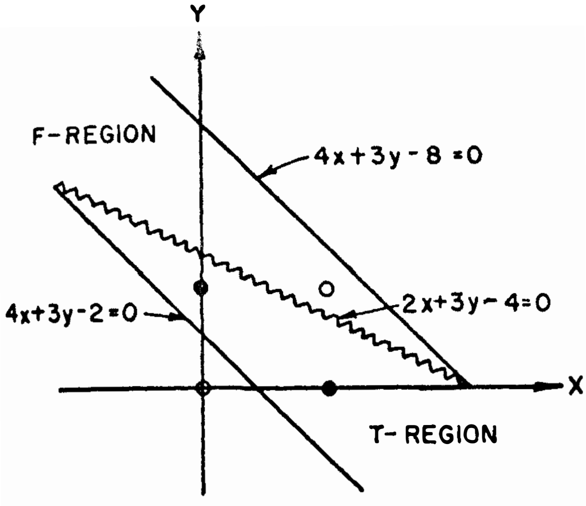


FIG 2

Often g and h will run parallel, dividing the n-space into four regions, alternating T, F, T, F.

TWO-ELEMENT SYNTHESIS

The main point of the procedure is contained in the following results:

Definition: Given f and any valuation X, define $[X]^o_f$, $[X]^+_f$, $[X]^-_f$ as those subsets of $[X]$ containing the f-neutral, f-positive, and f-negative vectors respectively.

Theorem: If f, g, and h are as in (1), then for every valuation X, $[X]^o_h$ is f-proper. In fact, one of the following four pairs of equations holds*:

$$[X]^+_h \supseteq [X]^+_f \text{ and } [X]^o_h \supseteq [X]^-_f,$$

or

$$[X]^-_h \supseteq [X]^+_f \text{ and } [X]^o_h \supseteq [X]^-_f,$$

or

$$[X]^+_h \supseteq [X]^-_f \text{ and } [X]^o_h \supseteq [X]^+_f,$$

or

$$[X]^-_h \supseteq [X]^-_f \text{ and } [X]^o_h \supseteq [X]^+_f.$$

In less precise terms, this says nothing more than that the hyperplane defined by h splits certain offending vectors apart — if X is f-improper, then h either splits all of its positive vectors apart, or splits all of its negative vectors apart, so that the resulting set of unsplit vectors, $[X]^o_h$, is f-proper. Furthermore, having split one set apart, it then splits none of the other set. Now the two versions of g have a chance of dividing f's T-points from its F-points. (Of course, we know that even a completely monotonic function may not be realizable, so the converse of the theorem doesn't hold.)

This result applies to the function of Fig. 2 as follows: Algebraically, we know that $x\bar{y} + \bar{x}y$ is not 1-realizable because it is not unate in x (for instance). But this translates to not being positive in $X = \{x = T\}$ or in $\bar{X}$. And geometrically this means that the vector-set consisting of

V_1: (0,0) → (0,1) and

V_2: (1,0) → (1,1)

is improper — V_1 is positive and V_2 is negative. This means that whatever h we use must either split V_1 or split V_2; our particular h split V_2, and saved V_1 intact.

The synthesis method makes use of the above necessary condition on h to specify enough of its properties that h can itself be determined: For various appropriate* valuations X, we first determine whether X is f-proper or not. If it is, no information is obtained. If not, then we know that one of the four equations (2) hold for X. If we take enough X's, this information, taken together with the fact that h must be completely monotonic, will usually determine h. A reasonable way to handle the whole procedure is to build up h as follows:

1. Make a list corresponding to the improper X_i. Each entry consists of $[X_i]^+_f$ and $[X_i]^-_f$.

2. Set i = 1, and initialize and execute the following process:

3. Determine whether both points of any of the vectors of $[X_i]^+_f + [X_i]^-_f$ have

*The choice of sign on $[X]^{+}_{h}$ / $[X]^{-}_{h}$ corresponds to which side of the h-plane we take as the T-side. The other choice corresponds to which of $[X]^+_f$ or $[X]^-_f$ we split apart (see discussion following theorem). Note that either $[X]^-_h$ or $[X]^+_h$ must be empty, since $[X]$ must be h-proper.

*The choice of an appropriate set of such X is facilitated by previous theory [13]. We usually begin with one-argument X's, then use those 2-argument X's appropriate, and so on, stopping with $[n/2]$-argument X's.

yet been determined in h.

a. If they have, then it is determined which of the vector sets must be split and which kept intact. Put them properly into two corresponding lists: a "split-list" and a "pair-list".

b. If they haven't, then store the present lists, together with the present value of i, in a pile* memory; put $[X_1]^{+}_{f}$ into the split-list, $[X_1]^{-}_{f}$ into the pair-list. (We come back to the other alternative later.)

4. Now go through three different processes, attempting to determine additional points of h, until none of them augment h further:

a. Look at each split-list entry**. If any of the points in any of the vectors in any entry have been determined for h, then all the points appearing in vectors of that entry are determined. In this case specify these values for h and erase this entry from the split-list.

b. Look at each pair in any entry of the pair-list. If either point has been determined for h, then the other of the pair is determined too, so specify it for h, and erase the pair (not the entry).

c. Apply the "don't-care" test synthesis augmentation described elsewhere [13] to h, specifying further points***.

In any of these processes it may happen that a contradiction arises, that no h satisfies the choices made so far. In this case go to step 6. Otherwise,

5. If all X_i have been considered, a consistent h satisfying all conditions has been found. In this case go to step 7. Otherwise, increment i by one and return to step 3.

6. Fetch the last entry in the pile; we now want to pu sue another branch. Reset i as it was at this last branch point, and this time put $[X_i]^{-}_{f}$ into the split list, $[X_i]^{+}_{f}$ into the pair list. (If we've backed all the way up past the last branch, then the pile will be empty, and we've shown that all possibilities are inconsistent, i.e., that f is not 2-realizable, and we exit.) Go to step 4.

7. Using the standard 1-realization algorithm, realize h. (If it is not realizable, which is possible since we only know that h is completely monotonic, then we must start over again at step 6.) And now 2^n of the 2^{n+1} vertices of g are determined (one of g's a guments is h). Use the don't-care test synthesis procedure to realize g. (Again — return to step 6 if g isn't 1-realizable.)

In hand computation with colored pencils, a pile of mimeographed cube-arrays, and geometric intuition, a function of four a guments can be handled in a few minutes. The procedure was used to demonstrate that the 38 symmetry types which Minnick, Glaser, and Moore [9] were unable to reduce to 2 gates a e, in fact, not so realizable. When the algorithm is programmed, this result will be machine-checked. It will then be feasible, also, to obtain integral-minimal* realizations for all of the four-argument symmetry types simply by returning from step 7 to step 6, and comparing the various realizations obtained (there won't be many, usually).

AN EXAMPLE

Just to sketch an example, consider Minnick's function number 132 [9] : (0, 1, 2, 4, 7, 9, 14) or

$$f(w,x,y,z) = wxy\bar{z} + \bar{x}\bar{y}z + \bar{w}(xyz + \bar{x}\bar{y} + \bar{x}\bar{z} + \bar{y}\bar{z}),$$

which is mapped in 4-space in Fig. 3.

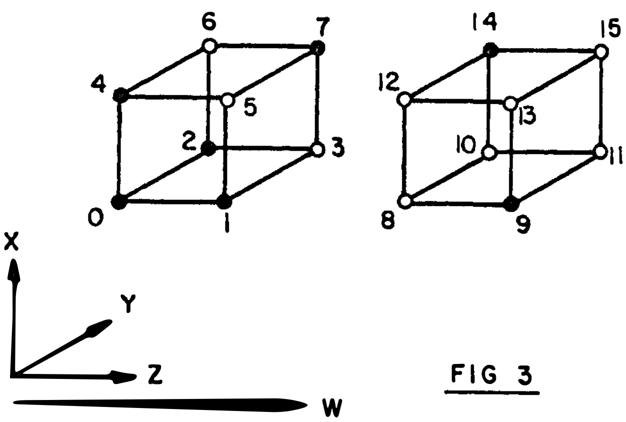


FIG 3

*A pile memory stores data serially, and when interrogated, produces the last item stored but not erased, and then erases it. E.g., a magnetic tape station which can be read from backwards could be used effectively.

**When i = 1, we arbitrarily put the single entry in the list ($[X_1]^{+}_{f}$ or $[X_1]^{-}_{f}$) into $[X]_h$ or $[X]_{\bar{h}}$, to get started. I.e., we pick a point arbitrarily in h so that all the points of this entry can be established in h. This arbitrary choice corresponds to settling which side of the h-plane will be T.

***As a simple example of how this works, suppose (see Fig. 2) that (0,0) has been determined F, and that (1,0) and (0,1) have been determined T. Then step 4.c would determine (1,1) as T so that the result could be unate.

*I.e., a realization in integers wherein the sum of absolute values of all weights involved is least.

Step 1 produces the following list of $[X]^{+}_{f}$, $[X]^{-}_{f}$:

X_1:	w-direction:	(6,14)	or (0,8), (2,10), (4,12), (7,15)
X_2:	x-direction:	(3,7), (10,14)	or (1,5), (2,6), (9,13)
X_3:	y-direction:	(5,7), (12,14)	or (1,3), (4,6), (9,11)
X_4:	z-direction:	(6,7), (8,9)	or (2,3), (4,5), (14,15)
X_5:	wx-direction:	(5,9)	or (4,8), (7,11)
X_6:	wy-direction:	(3,9)	or (2,8), (7,13)
X_7:	$\bar{x}$y-direction:	(8,14)	or (0,6), (9,15)
X_8:	xz-direction:	(9,12)	or (11,14)
X_9:	yz-direction:	(9,10)	or (13,14)

Now, when the second groups of X_1 and X_2 are chosen to be split, a contradiction will be obtained the first time step 4 actually gets going. The second group of X_1 together with the first group of X_2, however, forces us to split the first of X_1. At this point h is completely specified (by step 4) and the other X_i all work out without a hitch.

The function so specified may have turned out to be

$$\bar{w}\bar{z} + (\bar{w}+\bar{z})\,xy$$

(Fig. 4) or its complement, depending on how we arbitrarily began.

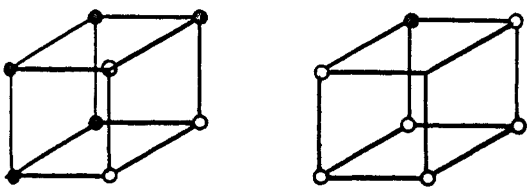


FIG 4

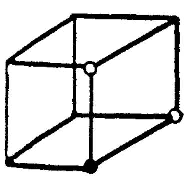
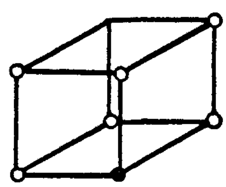


FIG 5



In Figure 5 the partially specified g is shown. It has one completely monotonic completion:

$$(\bar{x}+\bar{y}+z+w)h + \bar{x}\bar{y}z.$$

Combining these two, and writing down the obvious realizations, we get

$$f(w,x,y,z) = S(-2 + w - 2x - 2y + 2z + 5\cdot S(-2w + x + y - 2z)),$$

(writing "w" for "σ(w)", etc., which should produce no confusion here) or in Minnick's notation (which maps 0 → F)

$$2x,\ 2y,\ 1^{***}\ w,\ 2z,\ 5(2w, 2z^{*}\ x, y, 1).$$

If we decide to search for the minimal solution and continue considering possibilities, then by now we've considered all possibilities where the second group of X_1 is split. Taking then the first group to be split, it turns out that each of the four choices for X_2 and X_3 result in a different solution (one of them Minnick's). Of course, each of the five solutions we end up with has two forms, depending on whether we use h or $\bar{h}$ in each. And the solution given above is the integral-minimum solution.

GENERALIZATION

But suppose all alternatives result in contradiction , — that f is not 2-realizable. (With more than four arguments involved, this becomes the most likely case.) We can get a multi-element realization by iterating the procedure just described; an optimal realization is not guaranteed, but a good one will be obtained.

Suppose that in testing f for 2-realizability, h_1 was the h that in some sense came closest to working (perhaps it split the most improper valuations). Then in the second step of the overall process, we try to 2-realize the function g_1 which is implicitly and incompletely specified by the equation

$$f(\vec{x}) = g_1(\vec{x}, h_1(\vec{x})).$$

Since f wasn't 2-realizable, we know that g_1 won't be 1-realizable, so we try now to 2-realize it by repeating the procedure above. If we succeed, we have

$$f(\vec{x}) = g_2(\vec{x}, h_1(\vec{x}), h_2(\vec{x}, h_1(\vec{x}))),$$

a 3-realization. If we don't, we try again with g_2, and so on. The procedure terminates (and fast) because each new h makes more valuations proper.

The result of this process is

$$f(\vec{x}) = g_k(\vec{x}, h_1(\vec{x}), h_2(\vec{x},h_1(\vec{x})), \ldots$$
$$h_k(\vec{x}, h_1(\vec{x}), \ldots, h_{k-1}(\vec{x}, \ldots)))$$

which represents a structure as in Fig. 6.

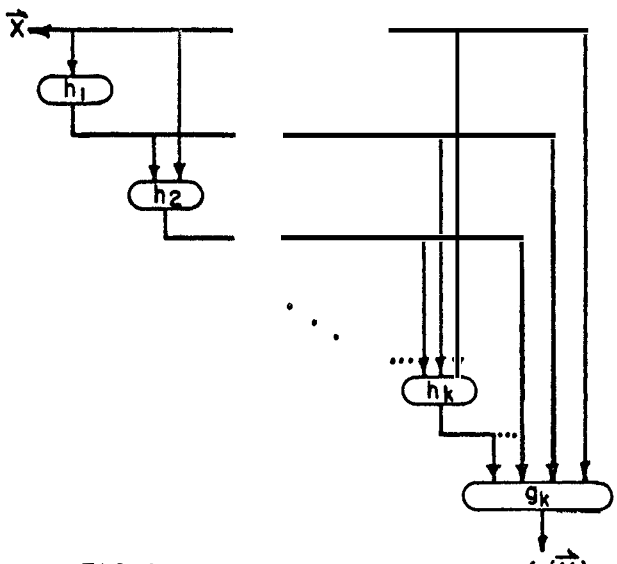


FIG. 6

When 2-level synthesis is desired, then in the process of finding the successive h_i we don't use the earlier h_i as arguments, and so get as a realization

$$f(\vec{x}) = g_k(\vec{x},h_1(\vec{x}), h_2(\vec{x}),\ldots, h_k(\vec{x})).$$

This special case allows a simple geometric interpretation generalizing from Fig. 1: The k h's divide the n-cube into many regions, and within each region a parallel version of g_k realizes the local restriction of f (see Fig. 7). If it is desired to eliminate the inputs $\vec{x}$ altogether from the lower stage(s) (as Muroga requires) then we add enough h-planes so that each region contains all T-vertices or all F-vertices.

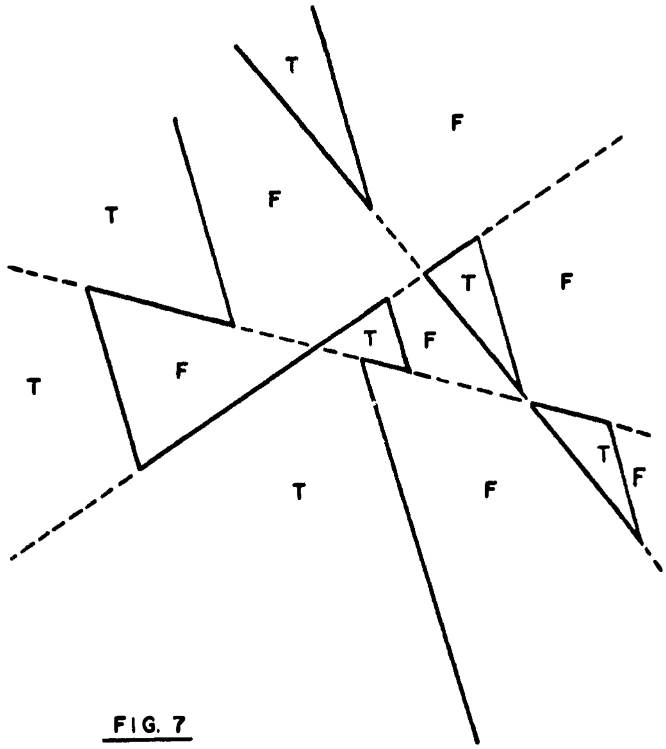


FIG. 7

Many questions about what can be expected in general and what biases are likely to lead to better solutions remain to be answered; our purpose here was simply to outline the application of k-monotonicity ideas to compound synthesis.

The interested reader can compare his results on the following example with those of the author:

$$f(v,w,x,y,z) = \bar{w}xz + \bar{v}x\bar{y} + v\bar{w}\bar{x}\bar{y} + x\bar{y}z$$
$$+ w\bar{v}x\bar{z} + w\bar{v}\bar{y}\bar{z} + \bar{w}\bar{v}\bar{x}y\bar{z} + wv\bar{x}yz$$

or

(2,4,5,7,8,12,13,14,16,17,21,23,27,29).

Trying to make proper the y- and z-directions, f was shown non 2-realizable very quickly: Only one out of four choices failed to produce a contradiction with just these two variations considered! And this one, although it managed to split the x- and w-directions, too, could not split the v-direction (and left two higher improper valuations unsplit: xz and $\bar{y}$z). The h which managed this is the term with coefficient 12 in the final result below. The next step was to split the remaining three improper valuations — this was quite simple, and partially specified h_2, which has coefficient -7 below. To realize g_2, which had only 32 out of its 128 vertices specified, we used the don't-care synthesis procedure, and ended up with the 3-element result

$f(v,w,x,y,z) = S(-2-3v-8w+10x-3y+8z$
$-7.S(-v-2w+x+2y)$
$+12.S(-1+v+2w-3x+2y-4z))$.

II. HIGHER CONDITIONS

THE CLASSICAL CONDITION AND SUMMABILITY

We recall the basic set of 2^n inequalities, whose solvability is equivalent to a function's realizability [13]. If $\vec{v}^{(j)}$ are such that $f(\vec{v}^{(j)}) = T$ while $\vec{w}^{(k)}$ are such that $f(\vec{w}^{(k)}) = F$, then the system of inequalities with variables $\vec{a}$, is

$$\begin{aligned} \vec{a} \cdot \vec{v}^{(j)} &\geq 0 \quad \text{all } j \\ \vec{a} \cdot \vec{w}^{(k)} &< 0 \quad \text{all } k \end{aligned} \qquad (2)$$

To make the type of sign uniform we write down the equivalent* system

$$\vec{a} \cdot (\vec{v}^{(j)} - \vec{w}^{(k)}) > 0 \qquad \text{all } j,k,$$

and for convenience let $\vec{u}^{(m)}$ run through all $\vec{v}^{(j)} - \vec{w}^{(k)}$, so that we get

$$\vec{a} \cdot \vec{u}^{(m)} > 0 \qquad \text{all } m.$$

(Note that the dummy argument $v_o^{(i)} = w_o^{(j)} = T$ for all i, j, so in effect we've eliminated the threshold a_o from the system.) There may be up to 4^{n-1} of these inequalities, but as shown last year [13], most of these are redundant and can be ignored.

Both C. C. Elgot [4] and C. K. Chow [2] have pointed out an equivalent condition for realizability obtained by applying a well known** result in the theory of linear inequalities. It is the following:

That there exist no $\alpha_m \geq 0$, $\Sigma \alpha_m > 0$, such that

$$\sum_m \alpha_m \cdot u_i^{(m)} = 0 \qquad \text{for all } i. \quad (3)$$

This new condition is nothing more than the dual of the old, in linear programming terms: Let A be the matrix whose columns are the $\vec{v}^{(m)}$, let U be (the column vector) $\vec{a}$, X be $\vec{\alpha}$. Then originally we had (T is the transpose)

$$A^T U > 0$$

and we got the equivalent conditions

$$AX = 0$$
$$X \geq 0, \ X \neq 0.$$

---

*Equivalent because this amounts to eliminating a_o.

**See references 1 and 5.

In A. W. Tucker's terminology [12], the systems

$$A^T U \geq 0 \text{ and } \begin{matrix} AX = 0 \\ X \geq 0 \end{matrix}$$

are dual. And his theorem 6 has as a corollary that requiring strict inequalities on the left is equivalent to requiring non-trivial solutions on the right, so far as consistency goes.

We next restate, and give a name to, a condition discussed by Elgot and Chow:

> <u>Definition</u>: A switching function is <u>k-summable</u> when there exist j f-positive vectors X_i $(2 \leq j \leq k)$ (not necessarily distinct) such that $\Sigma X_i = 0$. f is <u>k-asummable</u> when it is not k-summable. f is <u>summable</u> when it is k-summable for some k, otherwise <u>asummable</u>.

As shown by Elgot:

> <u>Theorem</u>: f is realizable if and only <u>if f is</u> asummable.

This restatement should be obvious: The positive vectors have the components of $\vec{u}^{(m)} = \vec{v}^{(j)} - \vec{w}^{(k)}$, and the theorem simply says that when f is unrealizable we can find integral α_m to satisfy (3). Since the coefficients are all integral, this is no restriction, and the theorem simply restates condition (3).

CONNECTIONS WITH OTHER PROPERTIES

As Elgot pointed out, 2-asummability is equivalent to complete monotonicity, and 3-asummability is equivalent to a condition of Muroga, et al [11] (page 387). 2-asummability is known to be not sufficient for realizability (E. F. Moore's counterexample, see reference 13), so the question arose, is k-asummability, for some fixed k (≥ 3), sufficient? It is not, showing that as k increases, the condition becomes in fact stricter. We demonstrate this by constructing a set of functions $\{f_k\}$, where f_k is k-asummable, but not realizable:

Fix k. Choose $(a_1, a_2, \ldots, a_{n-1})$ relatively prime in pairs, such that

$$a_i > 2(k+1) \quad \text{for all } i = 1, 2, \ldots n-1,$$

and such that

$$\frac{1}{2} < \sum_1^{n-1} \frac{1}{a_i} < 1.$$

Solve the following Diophantine equation for (a_n, p_n):

$$a_n\left[\prod_1^{n-1} a_i\left(\sum_1^{n-1}\frac{1}{a_i}-1\right)\right]+p_n\left[\prod_1^{n-1} a_i\right]$$
$= -1.$

Choose a solution such that

$$0 < 2p_n < a_n.$$

Define f_k as follows: f_k has $(\sum_1^n a_n)$ arguments, and is symmetric within groups of a_i arguments. We define the value of f_k in terms of how many arguments within each of the n groups have value T. This quantity (how many arguments in the j^{th} group have value T) we call y_j. (So $0 \leq y_j \leq a_j$.) Now f_k is almost separated by the plane

$$\sum_j \frac{y_j}{a_j} > 1.$$

For all argument (Σa_i)-tuples satisfying this inequality we assign f_k value T. For those tuples not satisfying it we assign value F, with one exception: The tuple with y-values

$$\vec{p} = (1, 1, \ldots, 1, p_n)$$

we assign value T, instead.

That the steps can be made as required above, and that the result is in fact k-asummable but not realizable, requires more proof than we have space for here.

FURTHER PROPERTIES, AND SOME OPEN QUESTIONS

During last year's conference interest was shown in the following property of realizable functions (we use Elgot's notation):

> Definition: Given a switching function f, then a relation on the points of the n-cube is defined as follows: $P \xrightarrow{f} Q$ means that there exists an f-positive vector parallel to and of the same length as that from P to Q. (We drop the "f" when no confusion can result.) f is transitive when f is 2-asummable (i.e., completely monotonic) and this relation is transitive, i.e., when $P \to Q \to R$ implies $P \to R$.

Unfortunately, transitivity falls far short of being necessary for realizability: The (realizable) function expressed by

AB (C + D)

is not transitive. Whether transitivity is sufficient for realizability remains an open question.

Another property of interest, first discussed by Elgot, is the following:

> Definition: A switching function f is k-cyclic when for some $j \leq k$ there exist points $p_1, p_2, \ldots, p_j$ in the n-cube such that $p_1 \xrightarrow{2} p_2 \to \ldots \to p_j \to p_1$. If not k-cyclic, f is k-acyclic; if k-cyclic for some k, f is cyclic, and otherwise acyclic.

Clearly a k-cyclic function is k-summable and a cyclic function summable, so k-acyclicity and acyclicity are necessary conditions for realizability. Also, a transitive function is acyclic. Elgot showed that for k = 2 and 3, k-acyclicity and k-asummability are equivalent. It remains an open question whether acyclicity is equivalent to realizability (and asummability). (Solved July 25: negative.)

III. INTEGRAL-MINIMAL REALIZATION

> Definition: a realization $\bar{a}$ of a function f is integral-minimal (or simply minimal) when the a_i are integers, and among such realizations of f, the weight
>
> $$W = \sum_0^n |a_i|$$
>
> is minimal.

The problem of finding such realizations efficiently has received a large amount of (unpublished) attention. It remains unsolved; our intention here is to point out the applicability of k-monotonicity ideas to the obvious direct approach, and to apply these ideas in disproving a conjecture made last year.

In the process of determining a function's realizability, an iterative solution for the a_i is found. Suppose we consider the g_2 of the last example of the section on compound synthesis (we drop the negation on t, w, v, and y):

$$\begin{aligned} f(t,u,v,w,x,y) = u\,[&x(w+z+t+v+y) \\ &+ wz + (w+z)(t+v+y) + tvy] \\ &+x\,[wz + (w+z)(t+vy) + tvy] \\ &+ wzt. \end{aligned}$$

Using the methods of reference 13, we get

$u > x > w = z > t > v = y > 0$, and by forming C(P) and reducing (in particular, assuming the coefficients of equal arguments are equal), we get as the essential terms (and now we use the same letter to represent the coefficient as the variable):

$$\left.\begin{array}{c} u + w + v \\ x + t + 2v \\ 2w + t \end{array}\right\} > \left\{\begin{array}{l} u + x \\ u + t + v \\ x + w + v \\ 2w + 2v \end{array}\right. \qquad (4)$$

The (iterative) solution of this system is obtained in the usual fashion; the tables below indicate a convenient way to arrange the calculation.

crossterms from (4):

	u	x	w	t	v
**		-1	1		1
**			1	-1	
**	1	-1			
	1		-1		-1
	-1			1	2
	-1	1			1
**			-1	1	1
**		1	-2	1	
	-1	-1	2	1	
	-1		2		-1
**		-1	1	1	-1
**				1	-2

u eliminated:

	x	w	t	v
	-1	1		1
**		1	-1	
**		-1	1	1
	1	-2	1	
**	-1	1	1	-1
**			1	-2
**	-1		1	2
**				1
**	-2	2	1	
**	-1	2		-1
*		-1	1	1
**	1	-1		
*	-1	1	1	-1
**		1		-2

x eliminated:

	w	t	v
	1	-1	
	-1	1	1
**		1	-2
**			1
**	1		-2
*	-1	1	1
*			1
**	-1	2	-1
*	-2	2	2
**	-2	3	
**		1	-1
*	-1	2	-1
*		1	-1
**	-1	1	2
**		1	
**	1		-1
*		1	-1

w eliminated:

	t	v
	1	-2
**		1
**	1	-1
**	1	
*		1
*	1	-1
*	1	
*		2
*	1	-1
*	1	
**	2	-3
**	3	-4
*	1	
*	2	-2
**	3	-2
**	1	1

t eliminated:

v
1

The first table contains all possible combinations from (4): $-x+w+v > 0$, etc. Subsequent tables list first the inequalities not involving the eliminated variable, and then list all sums of two inequalities which eliminate the variable. The * indicates duplicate equations, the ** indicates equations which were later found to be redundant. (The work would have been much shortened, of course, if we had looked for redundant equations immediately.) The solution is read off from the un-starred inequalities:

$$v > 0$$
$$t > 2v$$
$$t + v > w > t$$
$$w + v > x > 2w - t$$
$$t + 2v,\ x + v,\ 2w + t - x,\ 2w - v > u > w + v$$

The direct approach is to try integral solutionslexicographically as follows*:

v	t	w	x	u	
1	3	-			
1	4	-			making no progress. Note $v \geq 2$ required.
2	5	6	-		
2	6	7	-		ditto. Note that v must increase again
3	7	8	10	12	and the bias turns out to be -23**.

To assure ourselves that this is minimal, we use the following sort of argument:

$1 \leq w - t < x - w < v$, so $v \geq 3$

$t > 2v$, so $t \geq 7$

$w > t$, so $w \geq 8$

$1 \leq w - t < x - w$, i.e. $x > w + 1$ so $x \geq 10$

$u > w + v$ so $u \geq 12$

u + x does not attain threshold

so the bias ≤ -23.

Several drawbacks to this approach should be mentioned. The process of determining how to get out of a hopeless iteration needs clarification (a simple task). More important, it would be a great improvement if we knew in general that our result, a "lexicographic minimum", is the actual minimum wanted. (Conjecture: It always will be.) In such a case, the last (seat-of-the-pants) step is not needed. An important defect is the fact that equal arguments were assigned equal coefficients. Without this simplification, the number of inequalities gets out of hand (for large n). And with it, a minimum solution is not assured, contrary to the conjecture made last year (in private discussions). The following example demonstrates this.

---

*E.g., having assigned v = 1, we know t > 2, so we try t = 3. Then 4 > w > 3, impossible. So we try another assignment.

**To realize the original function, where t, w, v, and y were negated, we use instead -3,-3,-7,-8,8,10,12; -23+3+3+7+8 = -2.

F(A, B, C, D, E, G, H, I) =

A [(B + C) + (D + E) + G(H + I) + H I]
+ BC +(B+C) [DE + (D+E) [G + (H+I)] + GHI]
+ DE [G + HI]

Here we have

$$A > B = C > D = E > G > H = I.$$

If, as with the previous example, we assume b = c, d = e, and h = i, then the procedure (eventually) gives the realization

9,7,7,5,5,3,2,2; -13.

But the realization 9,7,6,5,5,3,2,2; -13, with W = 52, is minimal, as the following argument shows: We know a > b, c > d, e > g > h, i and we can check that d, e > h + i > g (by 3-monotonicity).

1. If h = i = 1, then 2 > g > 1; impossible.

2. If h = 1, i = 2, then 2 > g > 1; impossible.

3. If h ≥ 2, i ≥ 3, then the best possible realization would be 2,3,4,6,6,7, 7,8; -16 (since DGHI does not attain threshold) with W = 59.

4. So try h = i = 2. Then 4 > g > 2, i.e., g = 3.

A. If e = 6, the best possible realization would be 2,2,3,5,6,7,7,8; -14, with W = 54.

B. So try d = e = 5 (since d, e > 4). Now, since DEG attains threshold while DEH does not, the bias = -13. Since BC and AHI attain threshold, b + c ≥ 13 and a ≥ 9. So W ≥ 2 + 2 + 3 + 5 + 5 + 13 + 9 + 13 = 52, and the given solution is indeed minimal.

This function should be an interesting one for testing newly invented synthesis procedures. In particular, it would not be surprising if the linear programming methods (e.g., Muroga [11] and Minnick [9]) produced fractions with it.

"This paper, although based on work sponsored by the U. S. Air Force, has not been approved or disapproved by that Agency."

The research reported in this paper has been partially sponsored by the Electronics Research Directorate of the Air Force Office of Aerospace Research, contract AF19(604)-8423.

REFERENCES

1. W. B. Carver, "Systems of Linear Inequalities," Annals of Math. 23 (1921-22), pp. 212-220.

2. C. K. Chow, "Boolean Functions Realizable with Single Threshold Devices," Proc. IRE, Vol. 39, pp. 270-371, Jan. 1961.

3. M. Cohn and R. Lindaman, "Axiomatic Majority-Decision Logic," IRE Trans. on Electronic Computers, Vol. EC-10, pp. 17-21, March 1961.

4. C. C. Elgot, "Truth Functions Realizable by Single Threshold Organs," AIEE Conference Paper 60-1311, Oct. 1960, Revised Nov. 1960.

5. Ky Fan, "On Systems of Linear Inequalities," Linear Inequalities and Related Systems, Annals of Mathematics Studies, No. 8, Princeton University Press, 1956.

6. R. Lindaman, "A Theorem for Deriving Majority-Logic Networks Within an Augmented Boolean Algebra," IRE Trans. on Electronic Computers, Vol. EC-9, pp. 338-342, Sept. 1960.

7. R. Lindaman, "A New Concept in Computing," Proc. IRE, Vol. 48, p. 257, Feb. 1960.

8. R. McNaughton, "Unate Truth Functions," IRE Trans. on Electronic Computers, Vol. EC-10, pp. 1-6, March 1961.

9. R. C. Minnick, "Linear-Input Logic," IRE Trans. on Electronic Computers, Vol. EC-10, pp. 6-16, March 1961.

10. S. Muroga, "Logical Elements on Majority Decision Principle and Complexity of Their Circuits," Delivered at the International Conference on Information Processing -- UNESCO/NS/ICIP/G.2.10; June 1959.

11. S. Muroga, I. Ioda, S. Takasu, "Theory of Majority Decision Elements," Journal of the Franklin Institute, Vol. 271, pp. 376-418, May 1961.

12. A. W. Tucker, "Dual Systems of Homogeneous Linear Relations," Linear Inequalities and Related Systems, Annals of Mathematics Studies, No. 8, Prin. Univ. Press, 1956.

13. R. O. Winder, "Single Stage Threshold Logic," AIEE Conference Paper 60-1261, Oct. 1960.

14. R. O. Winder, "Some Recent Papers in Threshold Logic," Proc. IRE, Vol. 49, p. 1100, June 1961.