AN APPLICATION OF GROUP THEORY TO THE ANALYSIS OF SYMMETRIC GATES

Peter M. Maurer

Department of Computer Science
Baylor University
Waco, TX 76798

ABSTRACT

A method for determining the symmetries of the inputs of a logic gate either from its truth table or from facts obtained by inspection of its circuit is presented. The symmetry rule of a gate with *n* inputs is defined in terms of a subgroup of the symmetric group of degree *n*. This technique leads to an expanded and more complete definition of partial symmetry than has previously appeared. This definition of symmetry is used to show that the set of boolean expressions that represent non-totally-symmetric functions is NP-complete. The group-theoretic concept of conjugate sets is used to identify symmetry rules that are fundamentally the same but applied to different sets of inputs. A complete analysis of all forms of symmetry for 2-input, 3-input and 4-input gates is provided. An example is given to show how this theory was applied to a problem in VLSI design verification.

Editor's addendum to the abstract:

This paper was originally written in 1985 while I was a member of technical staff at the now defunct Bell Laboratories. Although I had high hopes for a publication, I spent many years attempting to publish this work without success. The theorists considered it too trival, while the practitioners considered it too theoretical. I disagree with both, but I wasn't able to convince anyone. While it is less sophisticated than any paper I would write today, I believe that the material is worth knowing and not available in any other form. Thus I am releasing this paper as a Baylor Technical Report.

This copy of the paper was OCRed from a paper copy and extensively edited to remove typos and OCR errors. Any remaining errors are mine and mine alone.

AN APPLICATION OF GROUP THEORY TO THE ANALYSIS OF SYMMETRIC GATES

Peter M. Maurer

Department of Computer Science
Baylor University
Waco, TX 76798

1. Introduction.

The study of symmetric boolean functions in *n* variables has a long history. The original motivation for studying such functions was to simplify the analysis and design of relay-based switching networks [10] [11]. Shannon recognized two types of symmetry, total symmetry which allows arbitrary permutations of the inputs of a gate, and partial symmetry which allows arbitrary permutations of selected sets of inputs [11]. In this paper we extend Shannon's concept of partial symmetry to general sets of permutations of the inputs of a gate.

Over the years attempts have been made to find efficient algorithms for identifying symmetric functions [8] and for creating symmetric functions from general boolean formulas [6] [13] [2]. Neither effort has met with great success. Indeed, we show in Section 4 that the set of boolean formulas that represent non-totally-symmetric functions is NP-complete[1][5].

The motivation for revisiting this topic is the problem of verifying the correctness of hand-packed VLSI layouts. Tools exist for translating a set of VLSI masks into a network of transistors, and for translating a transistor network into a network of gates[12]. Recently a logic-to-logic comparator has been developed to compare the logic-designer's schematics with logic diagrams extracted from a chip's layout[7]. Because of symmetries at the geometric and transistor levels, the gates extracted from VLSI masks may differ from the gates of the original logic design, even though the layout is correct. Successful verification of a VLSI layout by this method demands an accurate knowledge of a gate's symmetry. Unfortunately, the symmetries of some common gates, such as AOI22, are not completely describable as total or partial symmetries in the sense of Shannon[11].

The techniques presented in this paper require the calculation of the entire truth table of a boolean function, and thus are exponentially bounded. This approach is justifiable both because of the NP-completeness result of Section 4, and because the intended application is to boolean functions with few inputs. The techniques presented here are not applicable to gates with more than ten inputs, but this poses no problem because most logic gates used in VLSI design have four or fewer inputs or have symmetries that are obvious by inspection.

In the following, those definitions and lemmas that are taken form elementary group theory are identified by references to Robinson[9] and Herstein[4]. Any standard text on group theory or abstract algebra may be used as a substitute for these references.

2. The Symmetry Rule of a Gate.

Intuitively, a symmetric gate is a logic gate whose inputs can be rearranged in some way without changing the function of the gate. The following definition from elementary group theory can be used to formalize this concept.

DEFINITION 1. Let X_n be the set of integers $\{1,2,...,n\}$. A *permutation* of X_n is defined to be a one-to-one function from X_n onto itself. The set of all permutations of X_n forms a group under composition of functions. It is called the *symmetric group of order n* and is written S_n. (See Robinson[9] P. 6 or Herstein[4] p. 27).

Given a boolean expression with variables $x_1, x_2, ..., x_n$, permutations can be used to formalize the concept of rearranging the variables of the expression. However, in this paper we are primarily concerned with the function represented by an expression rather than the expression itself. In the following we show how the theory of permutations can be applied to boolean *functions* of *n* variables.

Let *F* be the set {0, 1} and F^n be the set of all *n*-tuples of elements of *F*. (Strictly speaking *F* is the Galois field *GF(2),* and F^n is the *n*-dimensional vector space over *GF(2),* but we will be making no use of the algebraic properties in this paper.) A physical gate **G** implements a function *G* from F^n to *F*. Since this paper is not concerned with the physical properties of gates, the gate **G** will be considered identical to the function *G* it implements. A rearrangement of the inputs of **G** is equivalent to rearranging the elements of each member of F^n before applying the function *G*. Such a rearrangement represents a function from F^n to itself.

The mapping $p \in S_n$ induces a mapping T_p on F^n in the following way. If $a=(a_1,a_2,...,a_n)$ $a = (a_1, ..., a_n)$ is an element of F^n then $T_p(a) = (a_{p(1)}, ..., a_{p(n)})$. Many of the nice mathematical properties of S_n arise from the fact that all permutations are one-to-one functions. The following lemma shows that the induced transformations also have this property.

LEMMA 1. *For each* $p \in S_n$ *the induced map* T_p *is one-to-one and onto.*

(The proofs of this and all other lemmas and theorems will be found in Appendix A.)

(Editor's note. The function T_p is, in fact, a linear transformation over $GF(2)^n = F^n$.)

Since F^n is a finite set, the set of all one-to-one onto functions from F^n forms a group which is isomorphic to S_m where $m = 2^n$. As the following lemma shows, the set of induced transformations forms a subgroup of S_m which is isomorphic to S_n.

LEMMA 2. *The mapping* M *from* S_n *to functions on* F^n *defined by* $M(p) = T_p$ *is an isomorphism.*

Since S_n and $M(S_n)$ are isomorphic, there is no need to continually refer to elements of S_n, and induced transformations T_p. To simplify the discussion, $M(S_n)$ will be written S_n, and p will be used instead of T_p to denote members of $M(S_n)$ When there is no danger of ambiguity, the members of $M(S_n)$ will be referred to as permutations.

It is now possible to formalize the intuitive concept of rearranging the inputs of a gate.

DEFINITION 2. Let p be an element on S_n. A *rearrangement* of the gate G is the function Gp from F^n to F. An element p is said to be *compatible* with G if $G=Gp$, that is, $G(a)=G(p(a))$ for all $a \in F^n$.

Obviously the identity element I of S_n is compatible with all n-input gates. If the inputs of a gate G cannot be rearranged in any way then the set of all permutations compatible with G is $\{I\}$. If all of the inputs of a gate G are identical as for an n-input **AND** gate, then the set of all elements of S_n compatible with G is S_n itself. Both S_n and $\{I\}$ are subgroups of S_n. The following theorem shows that the subgroup property is true for all gates G.

THEOREM 1. *If H is the set of all elements of S_n that are compatible with the n-input gate G, then H is a subgroup of S_n.*

Theorem 1 implies that if p and q are compatible with G then pq is compatible with G. Thus if X is a set of elements of S_n that are compatible with a gate G, then every element of the subgroup H generated by X is also compatible with G. This fact can be useful in computing additional compatible elements of S_n from a known set of compatible elements.

Since compatible elements "come in groups," it is possible to extend the definition of compatibility to subgroups of S_n.

DEFINITION 3. The subgroup H of S_n is said to be *compatible* with a gate G if every element of H is compatible with G.

Now suppose a subgroup H of S_n has been determined to be compatible with G. The subgroup H may not contain all elements that are compatible with G, but it is a subgroup of the group containing all elements compatible with G as the following lemma shows.

LEMMA 3. *If H_1 an H_2 are subgroups of S_n that are compatible with a gate G, then there exists a subgroup H_3 of S_n such that H_1 and H_2 are subgroups of H_3.*

This lemma gives a technique for computing additional permutations compatible with a gate *G* from tow groups of such permutations (assuming, of course, that one is not a subgroup of the other).

It is obvious that the set of subgroups of S_n compatible with a gate *G* is partially ordered by set inclusion. It is also obvious that the maximal element of this set is the set of all permutations *p* compatible with *G*. The following lemma captures these two facts.

LEMMA 4. *The set of all subgroups of S_n compatible with a gate G has a maximal element with respect to set inclusion.*

The maximal subgroup completely determines the symmetry of *G*, which leads to the following definition.

DEFINITION 4. The *symmetry rule* of a gate *G* is defined to be the maximal subgroup of S_n that is compatible with *G*.

The remainder of this paper will be concerned with the discovery and analysis of a gate's symmetry rule.

3. Orbits in F^n.

Although it is helpful to know that the set of elements of S_n compatible with *G* is a subgroup of S_n, it is still difficult to determine whether a compatible subgroup is the symmetry rule of a gate. The tools developed in this section will simplify the process of discovering a gate's symmetry rule. The following definition is slightly modified version of a standard definition from elementary group theory.

DEFINITION 6. Let a be an element of F^n and *H* be a subgroup of S_n. The *orbit* of *a* under H, written $O(a,H)$, is the set $\{x \mid x = p(a) \text{ for some } p \in H\}$. (See Robinson[9] p. 31, or Herstein [4] p. 65)

The standard definition specifies orbits in X_n with respect to a single permutation $p \in S_n$. Because F^n has a richer structure than X_n the orbits of F^n have a correspondingly richer structure than those of X_n. One useful fact from the traditional theory of orbits is that "belonging to the same orbit" is an equivalence relation.

LEMMA 5. *Let H be a subgroup of S_n and a and b be elements of F^n. Then the relation $a\mathrm{R}b \equiv a \in O(b,H)$ is an equivalence relation.*

This lemma shows that every subgroup H of S_n partitions F^n into disjoint subsets. The following theorem shows that if *H* is compatible with a gate *G*, then *G* is, in effect, a mapping from the collection of disjoint subsets to *F*.

THEOREM 2. *A subgroup H of* S_n *is compatible with a gate G only if* $a \in O(b, H)$ *implies that G*(*a*)=*G*(*b*).

Theorem 2 provides a method for testing whether or not a subgroup *H* is compatible with a gate *G*. First, use *H* to partition F^n into disjoint subsets, then check to see whether any two elements of the same subset are mapped to different values.

It was stated above that the orbits of F^n were richer in structure than those of X_n. Lemma 6, along with Definition 7, shows why this is so.

DEFINITION 7. Let *a*=(a_1, ... ,a_n) be an element of F^n. The *weight* of *a* is defined to be the number of elements, a_i, that are equal to 1.

Intuitively, since a permutation does nothing but rearrange the elements of a member of F^n, the weight of any member of F^n should be preserved by any permutation in S_n. The following lemma shows that this intuition is correct.

LEMMA 6. *Let a be an element of* F^n *and p be an element of* S_n. *Then the weight of a is equal to the weight of* $p(a)$.

This lemma shows that every subgroup *H* of S_n, including S_n itself, will partition F^n under the above orbit relation into subsets of *n*-tuples that have the same weight. This shows that there is no subgroup *H* of S_n with respect to which the orbit of $a \in F^n$ is all of F^n. Lemma 6 will be useful in calculating the symmetry rule of a gate from its truth table. Lemma 6 allows F^n to be partitioned with no *a priori* knowledge of the structure of the subgroup *H*. It is meaningful to ask whether F^n can be partitioned any further without specific knowledge of *H*. Lemma 7 shows that the answer is no.

LEMMA 7. *If the weight of* $b \in F^n$ *is equal to the weight of* $a \in F^n$ *then there exists a* $q \in S_n$ *such that* $q(a) = b$.

Gates with symmetry rule S_n are important (they include the logical functions AND, OR, NAND, NOR, and XOR) and are easy to analyze. Theorem 3, which is due to Shannon, characterizes an *n*-input gate with symmetry rule S_n in terms of its values on an (*n*+1)-element subset of F^n.

DEFINITION 8. Let P^n be the set of all $(a_1, a_2, ..., a_n)$ such that there exists an i with $0 \le i \le n$ such that $a_j = 1$ whenever $j \le i$ and $a_j = 0$ otherwise. The set P^n is called the *prefix-set* of F^n.

From the conditions on *i* it is obvious that P^n has *n*+1 members. It is also obvious that no two elements of P^n have the same weight. The *n*-tuple (0,0, ... ,0) is a member of

P^n (for $i = 0$) as is (1,1, ... ,1) (for $i = n$). As Theorem 3 shows, totally symmetric gates are characterized by their values on P^n.

THEOREM 3. (Shannon)*The value of a totally symmetric gate G on the members of* F^n *is completely determined by the value of G on the members of the prefix-set* P^n. *Furthermore, there are* 2^{n+1} *totally symmetric n-input gates.*

Given a truth table, the procedure for determining the symmetry rule of a gate is to partition F^n into disjoint subsets, and then determine the largest subgroup of S_n that preserves the partitioning. (That is, the largest subgroup H of S_n such that $O(a,H)$ is contained in subset of the partition for all $a \in F^n$.) The better the partition, the easier it is to determine the symmetry rule. Lemma 6 gives one partitioning rule, and Theorem 2 gives another. A third partitioning rule is given by Definition 9 and Lemma 8.

DEFINITION 9. Let $a = (a_1,...,a_n)$ and c be the function from F to F defined by $c(0) = 1$ and $c(1) = 0$. The *ones complement* of a written $c(a)$ is the n-tuple $(c(a_1),...,c(a_n))$. If X is is a subset of F^n then the *ones complement* of X, written $c(X)$ is the set $\{c(a) \mid a \in X\}$.

The ones complement of $a \in F^n$ is the n-tuple with ones where a has zeros, and zeros where a has ones. Note that $c(c(a)) = a$ If the weight of a is k, then the weight of $c(a)$ is $n-k$. The following lemma shows that taking the ones complement of a commutes with taking the orbit of a under H.

LEMMA 8. *If* $p \in S_n$ *and* $a \in F^n$ $p(c(a)) = c(p(a))$. *Also, if* H *is a subgroup of* S_n *then* $oc(O(a,H))=O(oc(a),H)$ $c(O(a,H)) = O(c(a),H)$.

Given a gate G whose truth table is known, Lemmas 6 and 8, and Theorem 2 can be applied to partition F^n. First F^n is partitioned into n+1 disjoint subsets, such that a and b are in the same subset if they have the same weight, and in different subsets otherwise. Next, each subset X of the first partition is split into two subsets $X_1 = \{a \mid a \in X$ and $G(a) = 1\}$ and $X_0 = \{a \mid a \in X$ and $G(a) = 0\}$ one of which may be empty. Lastly for each subset X in the second partition, $c(X)$ is computed. If $c(X)$ is a proper subset of some set Y of the second partition, then Y is split into two sets $c(X)$ and $Y - c(X)$. This last process continues until no further partitioning is possible. The third partition is used to compute the symmetry rule of G.

4. The Computational Complexity of Detecting Total Symmetry

Let *SYM* be the set of boolean formulas that represent totally symmetric functions. The complement of this set, the set of boolean formulas that represent non-totally-symmetric functions, will be denoted by $\overline{SYM}$. The following theorem shows that the set

$\overline{SYM}$ is NP-complete. (The proof of this theorem given in Appendix A follows the method of Karp[5].)

THEOREM 4. *The set* $\overline{SYM}$ *of boolean expressions that represent non-totally-symmetric functions is NP-complete.*

This theorem shows that there exists an efficient algorithm for identifying totally-symmetric functions boolean functions if and *only if* the two well-known complexity classes P and NP are equal. (Note that the existence of a deterministic polynomial time algorithm for any set such as $\overline{SYM}$ trivially implies the existence of such an algorithm for the complement of the set, in this case SYM.) As of this writing, the problem of whether P is equal to NP is still open hand has resisted all attempts at a solution. (The reader should consult Garey and Johnson[3] for more information about this famous open problem.)

5. Rearrangements of Gates.

When computing the symmetries of different *n*-input gates, it quickly becomes obvious that some symmetry rules are fundamentally different from one another, and some are "the same, but applied to different inputs." This distinction is more than intuitive, and can be used to simplify the categorization of symmetry rules. Mathematically, the symmetry rules that are "the same but applied to different inputs," are conjugates of one another. The following definition of conjugacy is taken from elementary group theory.

DEFINITION 10. Let X be a subset of S_n and let p be an element of S_n. The *conjugate of* X *by* p, written X^p, is the set $\{p^{-1}ap \mid a \in X\}$ (See Robinson[9] p. 26, or Herstein[4] p. 70).

Taking conjugates takes subgroups into subgroups as the following lemma shows.

LEMMA 9. *If* H *is a subgroup of* S_n *then* H^p *is a subgroup of* S_n

Recall the definition of a rearrangement of a gate G. Up to this point it has been assumed that the transformation p was compatible with G. However, it is possible to expand the study of rearrangements of G to permutations that may or may not be compatible with G. The following lemma shows how rearranging the inputs of a gate (possibly in a non-compatible way) affects its symmetry rule.

LEMMA 10. *if a gate* G *has symmetry rule* H *then* Gp *has symmetry rule* H^p.

Note that if $p \in H$ then $H^p = H$. Therefore if G has symmetry rule *H*, and p is compatible with G, then $Gp = G$ and has symmetry rule H As the following lemma shows, “being conjugate to one another” is an equivalence relation.

LEMMA 11. *Let* H *and* K *be subgroups of* S_n. *The relation* HRK *defined by* $H = K^p$ *for some* $p \in S_n$ *is an equivalence relation.*

Since conjugacy is an equivalence relation, it partitions the set of subgroups of S_n into disjoint subsets called conjugacy classes. The symmetry rules that are "the same but applied to different inputs" are in the same conjugacy class, while the symmetry rules that are fundamentally different from one another are in different conjugacy classes. To completely understand a conjugacy class, it is sufficient to enumerate the members of the class and provide a complete analysis of one member. The following section will make use of this fact to analyze the symmetries of 2-input, 3-input, and 4-input gates.

6. An Analysis of 2-, 3-, and 4-Input Gates.

In this section, the cycle notation will be used to denote members of S_n, and I will be used to denote the identity element. A cycle is a parenthesized list of integers without duplicates taken from the set $\{1,...,n\}$. Let a be an element of F^n. The cycle $(i_1,...,i_k)$ denotes the permutation that moves the element in position i_1 of a to position i_2 and the element in position i_2 to position i_3, and so forth. The element is position i_k is moved to position i_1. The cycle (1,2) denotes the permutation that swaps the first two elements of a, while the cycle (1,2,3) denotes the permutation that rotates the first three elements of a. A permutation that cannot be written as a single cycle can always be written as the product of two or more cycles. For example, the permutation (1,2)(3,4) swaps the first and second elements of a and then swaps the third and fourth elements. The reader should consult a book on abstract algebra or group theory for more information[4][9].

Appendix B of this paper lists the members and subgroups of S_2, S_3, and S_4. In addition, there are tables of orbits for some subgroups. For the sake of brevity, this section will make references to these tables.

Gates with two inputs exhibit the simplest form of symmetry. The symmetric group S_2 contains only two members, I and (1,2), and has only two subgroups $\{I\}$ and S_2. Thus a 2-input gate is either totally symmetric or non-symmetric. Theorem 5, which is an obvious application of Theorem 3, gives a quick method for testing the symmetry of a 2-input gate.

THEOREM 5. *A 2-input gate* G *is totally symmetric if and only if* $G(0,1)=G(1,0)$.

By theorem 3 there are 8 totally symmetric 2-input gates. Since there are a total of 16 2-input gates, half of all 2-input gates are totally symmetric. An example of a totally symmetric 2-input gate is ab while an example of a non-symmetric 2-input gate is ab'.

Gates with three inputs are also simple to categorize. There are only three forms of symmetry, and these are fairly obvious.

THEOREM 6. *A 3-input gate is either non-symmetric, totally symmetric, or partially symmetric with a symmetry rule conjugate to the subgroup* B1. (See Appendix B, Table 5.)

There are 256 distinct 3-input gates. By Theorem 3, 16 of these are totally symmetric. Since B1 has six distinct orbits in F^3, there are 64 gates compatible with B1. However, 16 of these are totally symmetric, so there are 48 gates with symmetry rule B1. Similarly, there are 48 gates with symmetry rule B2, and 48 gates with symmetry rule B3. Therefore, there are 144 partially symmetric and 96 non-symmetric 3-input gates.

The symmetries of 4-input gates are considerably more interesting than those for 2- and 3-input gates, as the following theorem shows.

THEOREM 7. *A 4- input gate is either non-symmetric, totally symmetric, or partially symmetric with a symmetry rule conjugate to one of the five subgroups* C1, C7, C17, C21, or C24. (See Appendix B, Table 10.)

(Editor's Note: Strictly speaking the theorem is incorrect. Group C1 represents a partial symmetry in two inputs, and C17 represents a partial symmetry in 3 inputs. Groups C7, C21 and C24 represent *weak* symmetries.)

By Theorem 33 there are 32 totally symmetric 4-input gates. Since C24 has six distinct orbits, there are 64 functions compatible with C24, 332 of which are totally symmetric. Therefore there are 32 gates with symmetry rule C24. Similarly there are 32 gates with symmetry rule C25, and 32 gates with symmetry rule C26. Since C21 has nine distinct orbits, there are 512 gates compatible with C21, 32 of which are totally symmetric and 32 of which have symmetry rule C25 (since C21 is contained in C25). Therefore, there are 448 gates with symmetry rule C21, and the same number with symmetry rules C22 and C23. Since C17 has eight orbits there are 256 gates compatible with C17. Of these, 32 are totally symmetric, so there are 224 gates with symmetry rule C17 and a similar number with symmetry rules C18, C19, and C20. Of the 1024 gates compatible with C7, 32 are totally symmetric, 448 have symmetry rule C21, 32 have symmetry rule C24, 32 have symmetry rule C25, and 32 have symmetry rule C26. This leaves 448 gates with symmetry rule C7, and a similar number with symmetry rules C8 and C9. Of the 4096 gates compatible with C1, 32 are totally symmetric, 224 have symmetry rule C17, 224 have symmetry rule C18, 448 have symmetry rule C21, and 32 have symmetry rule C24. This leaves 3136 gates with with symmetry rule C1. There are an identical number of gates with each of the symmetry rules C2 through C6. All told there are 32 totally symmetric 22496 partially symmetric, and 43008 non-symmetric 4-input gates.

7. An Example.

This study of symmetric gates was motivated by the recently-developed logic-to-logic comparator for VLSI layout verification[7]. This tool is used to compare logic diagrams extracted from a VLSI layout with the diagrams produced by the logic designer. The comparator is capable of correctly comparing symmetric gates,.as long as it has been supplied with the correct symmetry rule. During the shakedown phase of this tool, a new 4-input gate named **FF1** was defined to represent a certain very common electrical

structure. Although the truth table of this gate was not known, a few facts were obvious by inspection of the circuit. First, **FF1**(*a,b,c,d*) was equal to **FF1**(*b,a,d,c*) and to **FF1**(*c,d,a,b*) for all *a, b, c,* and *d* in {0,1}. Furthermore there was at least one combination of a, b, c, and d for which **FF1**(*a,b,c,d*) was not equal to **FF1**(*d,b,c,a*) implying that **FF1** was not totally symmetric. By applying Theorem 1, it was easy to see that **FF1**(*a,b,c,d*) must be equal to **FF1**(*d,c,b,a*) for all *a, b, c,* and *d*. The group consisting of these three permutations (plus the identity) is C27, but by Theorem 6, there is no gate with symmetry rule C27. Therefore the true symmetry rule of **FF1** had to be either C24, C25 or C26. C26 was ruled out by the fact that **FF1**(*a,b,c,d*) did not always equal **FF1**(*d,b,c,a*). The circuit was tested using the two vectors (0,0,1,1) and (0,1,1,0). It turned out that **FF1**(0,0,1,1) was equal to **FF1**(0,1,1,0) implying that **FF1** had symmetry rule C25. (If it had turned out that **FF1**(0,0,1,1) was not equal to **FF1**(0,1,1,0), it would have implied that the symmetry rule was C24.) Note that the *value* of **FF1** was known only for the two vectors tested. It should also be noted that there were several attempts to deduce the symmetry rule of **FF1** by inspection, all of which produced incorrect results. Currently the techniques described in this paper are used to determine the symmetry rules of all new gates.

(Editor's note: The logic-to-logic comparator described here was used for many years by Bell Laboratories. With the demise of Bell Laboratories, the logic-to-logic comparator passed from existence.)

8. Conclusion.

The tools presented in this paper allow the symmetry rule of a gate to be determined from its truth table with very little effort. The method used here is considerably easier to apply than inspection, and generally produces more accurate results. Most of the calculations described in this paper were done by hand. For gates with five or more inputs, computer techniques must be used to identify subgroups, analyze sets of orbits, and so forth. Even so, it is unlikely that these techniques will ever be useful for gates with more than ten inputs. The reader should note that the isomorphic copy of Sn examined in this paper is only a small subset of the parent group S_{2^n}. Shannon used a larger group in his study of switching functions but did not consider the full group S_{2^n}. Since this group has 16!=20,922,789,888,00 members for n=4, a complete analysis is probably not feasible. It may be possible that there are other "interesting" undiscovered subgroups of S_{2^n}.

(Editor's note: These speculations turned out to be correct.)

Appendix A: Proofs of Lemmas and Theorems.

Proof of LEMMA 1.

Let $a=(a_1,...,a_n)$ and $b=(b_1,...,b_n)$ be elements of F^n, and let p be an element of S_n. Suppose $T_p(a)=T_p(b)$. Let i be an integer $1\le i\le n$. Because p is onto, there exists a j such that $1\le j\le n$ and $p(j)=i$. Then $a_i=a_{p(j)}=b_{p(j)}=b_i$. Thus $a=b$, and T_p is one-to-one. Now since S_n is a group there exists a $q\in S_n$ such that $pq=I$, the identity element of S_n. Let a' be the element of F^n defined by $(a_{q(1)},...,a_{q(n)})$. Clearly $T_p(a')=a$, and T_p is onto.

Proof of LEMMA 2.

Let p and q be elements of S_n, and let a be an element of F^n. Then $T_p(T_q(a))=T_p(a_{q(1)},...,a_{q(n)})=(a_{p(q(1))},...,a_{p(q(n))})=(a_{pq(1)},...,a_{pq(n)})=T_{pq}$. Thus M is a homomorphism. To establish that M is one-to-one, observe that the kernel of M is the identity element of S_n. Suppose $p\in S_n$ is not the identity element of S_n. Then there exists an i, $1\le i\le n$ such that $p(i)=j$, and $i\ne j$. Let $a=(a_1,...,a_n)$ be the element of F^n such that $a_k=1$ if $k=j$ and $a_k=0$ otherwise. Let $b=T_p(a)$. If $k=i$ then $b_k=b_i=a_{p(i)}=a_j=1$ $b_k=b_i=a_{p(i)}=a_j=1$. Since $i\ne j$, $a_i=0$ which implies that $a\ne T_p(a)$ which in turn implies that T_p is not the identity function. Therefore p is not in the kernel of M.

Proof of THEOREM 1.

It suffices to show that H is closed under composition. Let p and q be elements of S_n that are compatible with G. Let a be an element of F^n. Since $q(a)$ is an element of F^n and since p is compatible with G, $G(p(q(a)))=G(q(a))$. Since q is compatible with G, $G(q(a))=G(a)$. Thus $G(pq(a))=G(p(q(a)))=G(a)$ for all $a\in F^n$, and pq is compatible with G.

Proof of LEMMA 3.

Let Q be the set of all pq such that $p\in H_1$ and $q\in H_2$, and let H_3 be the subgroup of S_n generated by Q. Since the identity element of S_n is contained in both H_1 and H_2, $H_1\subseteq Q\subseteq H_3$ and $H_2\subseteq Q\subseteq H_3$. The compatibility of H_3 follows from the proof of Theorem 1.

Proof of LEMMA 5.

Let I be the identity element of H. Since $I(a) = a$, $a \in O(a, H)$. Suppose $a \in O(b, H)$. Then there exists a $p \in H$ such that $p(b) = a$. Since H is a group, there must be a $p^{-1} \in H$ such that $pp^{-1} = I$, and $p^{-1}(a) = b$. Thus $b \in O(a, H)$. Now suppose $a \in O(b, H)$ and $b \in O(c, H)$. Then there exists a $p \in H$ such that $p(b) = a$, and a $q \in H$ such that $q(c) = b$. Since H is a group, $pq \in H$ and $pq(c) = p(q(c)) = p(b) = a$. Therefore $a \in O(c, H)$.

Proof of THEOREM 2.

Suppose H is compatible with G and $a \in O(b, H)$. Then for some $p \in H$, $a = p(b)$. Since H is compatible with G, $G(a) = G(p(a)) = G(b)$. Now suppose that whenever $a \in O(b, H)$, $G(a) = G(b)$. Let a be an arbitrary element of F^n and p be an element of H. By definition $p(a) \in O(a, H)$ and by our assumption, $G(p(a)) = G(a)$. Thus H is compatible with G.

Proof of THEOREM 3.

Suppose G is a totally symmetric gate with a inputs. Let x_i be the unique member of P^n with weight i. By Lemma 7, if the weight of $a \in F^n$ is equal to i, then $a \in O(x_i, S_n)$. Thus by Theorem 2, if $a \in F^n$ has weight i, $G(a) = G(x_i)$. Thus G is completely characterized by its values on the set P^n. There are 2^{n+1} functions from the set P^n to the set $\{0,1\}$.

Proof of LEMMA 8.

Let $a = (a_1, ..., a_n)$, $c(p(a)) = (b_1, ..., b_n)$, $p(c(a)) = (c_1, ..., c_n)$, and $c(a) = (d_1, ..., d_n)$. Let p be an element of S_n and p be the permutation of $\{1, ..., n\}$ that induces T_p on F^n. Let c be the function from F to F defined by $c(0) = 1$ and $c(1) = 0$. Let i be an integer such that $1 \le i \le n$. Then $c_i = d_{p(i)}$. But $d_{p(i)} = c(a_{p(i)})$, and $c(a_{p(i)}) = b_i$. Thus $c(T_p(a)) = T_p(c(a))$.

Proof of THEOREM 4.

To prove NP-completeness we must show two things. We must show that $\overline{\mathbf{SYM}}$ is in NP, and for each $L \in \mathrm{NP}$, we must exhibit a function f, computable in polynomial time, such that $f(x) \in \overline{SYM}$ iff $x \in L$. For the second part it suffices to exhibit an appropriate function from one NP-complete set[5]. We begin by showing that SAT, the NP-complete set of satisfiable boolean formulas, can be mapped into $\overline{SYM}$ in polynomial time. For any boolean formula e, let x_1 and x_2 be two variables that do not appear in e. If n is the length of e, then the variables x_1 and x_2 can be found in time $O(n^2)$ by enumerating variables in canonical order and searching e for an occurence. Generate the expression $g = x_1(e) + x_2'(e)$. If e is not satisfiable then neither is g. All non-satisfiable formulas represent the totally symmetric zero function, so if e is not satisfiable, then g is symmetric. Now suppose $e \in SAT$. If m is the number of distinct variables in e, then there exists an $a \in F^m$ such that $e(a) = 1$. Let a_1 and a_2 be the elements of F^{m+2} obtained by appending 01 and 10 to a respectively. The appended characters represent the values of x_1 and x_2. The weight of a_1 is identical to the weight of a_2. now $g(a_1) = 0(1) + 1'(1) = 0$ and $g(a_2) = 1(1) + 0'(1) = 1$. By Theorem 3, g cannot represent a totally symmetric function.

To show that $\overline{SYM}$ is in NP, consider the following non-deterministic procedure.

```
begin
    Input e;
    n := the number of distinct variables in e;
    Select a member of F^n and assign to v1;
    Select a member of F^n and assign to v2;
    k1 := the weight of v1;
    k2 := the weight of v2;
    r1 := the result of evaluating e on v1;
    r2 := the result of evaluating e on v2;
    if (r1 ≠ r2) and (k1 = k2)
    then accept;
end
```

This procedure accepts e if and only if there exist two elements of F^n with the same weight that cause the expression e to take on two different values. By Theorem 3 this implies that the procedure accepts e if and only if the function represented by e is not totally symmetric. The procedure is obviously polynomially bounded.

Proof of LEMMA 9.

Suppose x and y are elements of H^p. Then there exist elements a and b of H such that $x = p^{-1}ap$ and $y = p^{-1}bp$. Then $xy = p^{-1}app^{-1}bp = p^{-1}abp$. Since $ab \in H$, $xy = p^{-1}abp \in H^p$.

Proof of LEMMA 10.

This is the same as saying $Gpq = Gp$ if and only if $q \in H^p$. Suppose $q \in H^p$ then for some $a \in H$, $Gpq = Gpp^{-1}ap = Gap$. Since $a \in H$ and H is the symmetry rule of G, $Ga = G$. Thus $Gap = Gp$. Now assume that q is not an element of H^p. Let $a = pqp^{-1}$. Obviously $p^{-1}ap = q$. Since q is not an element of H^p, a is not an element of H. Since a is not an element of H, $Ga \neq G$. By Lemma 1, p is one-to-one and onto, so $Gap \neq Gp$. But $Gap = Gpp^{-1}ap = Gpq$. So $Gpq \neq Gp$.

Proof of LEMMA 11.

Let H be a subgroup of S_n, and let *I* be the identity element of S_n. Then $I^{-1}hI = h$ for all $h \in H$, and $H^I = H$. Therefore $H\mathrm{R}H$ for all subgroups H of S_n. Suppose $H\mathrm{R}K$ then $K = H^p$ for some $p \in S_n$. Let $q = p^{-1}$. Let x be an element of K^q. Then there exists a $k \in K$ such that $x = q^{-1}kq$. Now since $k \in K$ there exists an $h \in H$ such that $k = p^{-1}hp$, and $x = q^{-1}p^{-1}hpq$. But $q = p^{-1}$ therefore $x = pp^{-1}hpp^{-1} = h$, which implies that $x \in H$. This in turn implies that $K^q = H$ so $K\mathrm{R}H$. Now suppose $H\mathrm{R}K$ and $K\mathrm{RL}$. Then there exists a $p \in S_n$ such that $K = H^p$, and a $q \in S_n$ such that $L = K^q$. Let x be an element of L. Then there exists a $k \in K$ such that $x = q^{-k}kq$. There also exists an $h \in H$ such that $k = p^{-1}hp$. Thus there exists an $h \in H$ such that $x = q^{-1}p^{-1}hpq$. Since $(pq)^{-1} = q^{-1}p^{-1}$, $x \in H^{pq}$. Thus $L = H^{pq}$ and $H\mathrm{R}L$.

Proof of THEOREM 6.

The gate abc is totally symmetric, and the gate $a'b + c'$ is non-symmetric. Furthermore, the gate $ab + c$ is partially symmetric with symmetry rule B1, so all three types of gates exist. The subgroups B2 and B3 are conjugate with B1. The only other subgroup of S_3 is B4, but as Tables 7 and 8 (in Appendix B) show, any gate compatible with B4 is totally symmetric by Theorem 2.□

Proof of THEOREM 7.

The gate $abcd$ is totally symmetric, and the gate $ab'+c+d'$ is non-symmetric, so both these types exist. The five partially (and weak ed.) symmetric types exist as well, since $ab+c+d'$ has symmetry rule C1, $ab'+cd'$ has symmetry rule C7, $abc+d$ has symmetry rule C17, $ab+c+d$ has symmetry rule C21, and $ab+cd$ has symmetry rule C24. The conjugacy classes of subgroups of S_4 are {C0}, {C1,C2,C3,C4,C5,C6}, {C7,C8,C9}, {C10,C11,C12,C13}, {C14,C15,C16}, {C17,C18,C19,C20}, {C21,C22,C23}, {C24,C25,C26}, {C27}, and $\{S_4\}$. Tables 13 and 15 (in Appendix B) show that any gate compatible with C10 is also compatible with C17, so neither C10 nor any of its conjugates can be the symmetry rule of a gate. Similarly, Tables 14 and 17 show that any gate compatible with C14 is also compatible with C24, and Tables 19 and 20 show that any gate compatible with C28 is totally symmetric. Therefore neither C14, C15, C16, nor C28 can be the symmetry rule of a gate. The only remaining conjugacy class is {C27}. Assume that *G* is compatible with C27. If *G* maps orbits 3 and 4 to the same value, then *G* is compatible with C26. If *G* maps orbits 3 and 5 to the same value, then *G* is compatible with C24. And if *G* maps orbits 4 and 5 to the same value, then *G* is compatible with C25. Since the range of *G* is {0,1}, at least two of the orbits 3, 4, and 5 must be mapped to the same value, and *G* must be compatible with a conjugate of C24. Therefore, C27 cannot be the symmetry rule of any gate.

Appendix B: Tables for S_2, S_3, and S_4.

Element	Permutation	Cycle
0	1 2	I
1	2 1	(1 2)

TABLE 1. The Elements of S_2.

Name	Elements
A0	I
A1	I, (1 2)

TABLE 2. The Subgroups of S_2.

Number	Elements
1	00
2	01, 10
3	11

TABLE 3. The Orbits of S_2.

Element	Permutation	Cycle
1	1 2 3	I
2	1 3 2	(2 3)
3	2 1 3	(1 2)
4	2 3 1	(1 3 2)
5	3 1 2	(1 2 3)
6	3 2 1	(1 3)

TABLE 4. The Elements of S_3.

Name	Elements
B0	I
B1	I, (1 2)
B2	I, (1 3)
B3	I, (2 3)
B4	I, (1 2 3), (1 3 2)
B5	S_3

TABLE 5. The Subgroups of S_3.

Number	Elements
1	000
2	001
3	010, 100
4	011, 101
5	110
6	111

TABLE 6. The Orbits of B1.

Number	Elements
1	000
2	001, 010, 100
3	011, 101, 110
4	111

TABLE 7. The Orbits of B4.

Number	Elements
1	000
2	001, 010, 100
3	011, 101, 110
4	111

TABLE 8. The Orbits of S_3.

Element	Permutation	Cycle
1	1 2 3 4	I
2	1 2 4 3	(3 4)
3	1 3 2 4	(2 3)
4	1 3 4 2	(2 4 3)
5	1 4 2 3	(2 3 4)
6	1 4 3 2	(2 4)
7	2 1 3 4	(1 2)
8	2 1 4 3	(1 2)(3 4)
9	2 3 1 4	(1 3 2)
10	2 3 4 1	(1 4 3 2)
11	2 4 1 3	(1 3 4 2)
12	2 4 3 1	(1 4 2)
13	3 1 2 4	(1 2 3)
14	3 1 4 2	(1 2 4 3)
15	3 2 1 4	(1 3)
16	3 2 4 1	(1 4 3)
17	3 4 1 2	(1 3)(2 4)
18	3 4 2 1	(1 4 2 3)
19	4 1 2 3	(1 2 3 4)
20	4 1 3 2	(1 2 4)
21	4 2 1 3	(1 3 4)
22	4 2 3 1	(1 4)
23	4 3 1 2	(1 3 2 4)
24	4 3 2 1	(1 4)(2 3)

TABLE 9. The Elements of S_4.

Name	Elements
C0	I
C1	I, (1 2)
C2	I, (1 3)
C3	I, (1 4)
C4	I, (2 3)
C5	I, (2 4)
C6	I, (3 4)
C7	I, (1 2)(3 4)
C8	I, (1 3)(2 4)
C9	I, (1 4)(2 3)
C10	I, (1 2 3), (1 3 2)
C11	I, (1 2 4), (1 4 2)
C12	I, (1 3 4), (1 4 3)
C13	I, (2 3 4), (2 4 3)
C14	I, (1 2 3 4), (1 3)(2 4), (1 4 3 2)
C15	I, (1 2 4 3), (1 4)(2 3), (1 3 4 2)
C16	I, (1 3 2 4), (1 2)(3 4), (1 4 2 3)
C17	I, (1 2), (1 3), (2 3), (1 2 3), (1 3 2)
C18	I, (1 2), (1 4), (2 4), (1 2 4), (1 4 2)
C19	I, (1 3), (1 4), (3 4), (1 3 4), (1 4 3)
C20	I, (2 3), (2 4), (3 4), (2 3 4), (2 4 3)
C21	I, (1 2), (3,4), (1 2)(3 4)
C22	I, (1 3), (2 4), (1 3)(2 4)
C23	I, (1 4), (2 3), (1 4)(2 3)
C24	I, (1 3), (2 4), (1 2)(3 4), (1 3)(2 4), (1 4)(2 3), (1 2 3 4), (1 4 2 3)
C25	I, (1 2), (3 4), (1 3)(3 4), (1 3)(2 4), (1 4)(2 3), (1 3 2 4), (1 4 2 3)
C26	I, (1 4), (2 3), (1 2)(3 4), (1 3)(2 4), (1 4)(2 3), (1 2 4 3), (1 3 2 4)
C27	I, (1 2)(3 4), (1 3)(2 4), (1 4)(2 3)
C28	I, (1 2)(3 4), (1 3)(2 4), (1 4)(2 3), (1 2 3), (1 3 2), (1 2 4), (1 4 2), (1 3 4), (1 4 3), (2 3 4), (2 4 3)
C29	S_4

TABLE 10. The Subgroups of S_4.

Number	Elements
1	0000
2	0001
3	0010
4	0100, 1000
5	0011
6	0101, 1001
7	1010, 0110
8	1100
9	1011, 0111
10	1101
11	1110
12	1111

TABLE 11. The Orbits of C1.

Number	Elements
1	0000
2	0001, 0010
3	0100, 1000
4	0011
5	0101, 1010
6	0110, 1001
7	1100
8	1110, 1101
9	1011, 0111
10	1111

TABLE 12. The Orbits of C7.

Number	Elements
1	0000
2	0001
3	0010, 0100, 1000
4	0011, 1001, 0101
5	1100, 0110, 1010
6	1101, 1011, 0111
7	1110
8	1111

TABLE 13. The Orbits of C10.

Number	Elements
1	0000

2	0001, 0010, 0100, 1000
3	0011, 1100, 1001, 0110
4	0101, 1010
5	1110, 1101, 1011, 0111
6	1111

TABLE 14. The Orbits of C14.

Number	Elements
1	0000
2	0001
3	0010, 0100, 1000
4	0011, 1001, 0101
5	1100, 0110, 1010
6	1101, 1011, 0111
7	1110
8	1111

TABLE 15. The Orbits of C17.

Number	Elements
1	0000
2	0001, 0010
3	0100, 1000
4	0011
5	0110, 1001, 0101, 1010
6	1100
7	1110, 1101
8	1011, 0111
9	1111

TABLE 16. The Orbits of C21.

Number	Elements
1	0000
2	0001, 0010, 0100, 1000
3	0011, 1100, 1001, 0110
4	0101, 1010
5	1110, 1101, 1011, 0111
6	1111

TABLE 17. The Orbits of C24.

Number	Elements

1	0000
2	0001, 0010, 0100, 1000
3	0011, 1100
4	0101, 1010
5	0110, 1001
6	1110, 1101, 1011, 0111
7	1111

TABLE 18. The Orbits of C27.

Number	Elements
1	0000
2	0001, 0010, 0100, 1000
3	0011, 1100, 0110, 1001, 0101, 1010
4	1110, 1101, 1011, 0111
5	1111

TABLE 19. The Orbits of C28.

Number	Elements
1	0000
2	0001, 0010, 0100, 1000
3	0011, 1100, 0110, 1001, 0101, 1010
4	1110, 1101, 1011, 0111
5	1111

TABLE 20. The Orbits of S_4.

REFERENCES

1. S. A. Cook, "The Complexity of Theorem Proving Procedures," in *Proc. Third Annual ACM Symp. on Theory of Computing*, 1971, pp. 151-158.

2. B. Dahlberg, "On Symmetric Functions With Redundant Variables - Weighted Functions," *IEEE Transactions on Computers*, Vol. C-22, pp. 450-458, May, 1983.

3. M. R. Garey, D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, New York: W. H. Freeman and Co., 1979.

4. I. N. Herstein, *Topics in Algebra*, Waltham, MA: Blaisdell, 1964, Chapter 2.

5. R. M. Karp, "Reducibility Among Combinatorial Problems," in *Complexity of Computer Computations*, R. E. Miller and J. W. Thatcher, Eds.,m New York: Plenum, 1972, pp. 85-103.

6. W. H. Kautz, "The Realization of Symmetric Switching Functions With Linear-Input Logical Elements," *IRE Transactions on Electronic Computers*, Vol. EC-10, pp. 371-378, Sept., 1961.

7. P. M. Maurer, "A Logic-to-Logic Comparator for VLSI Layout Verification," *IEEE Transactions on CAD*, to appear, Aug., 1988.

8. A. Mukhopadhyay, "Detection of total or Partial Symmetry of a Switching Function with the Use of Decomposition Charts," *IEEE Transactions on Electronic Computers*, Vol. EC-12, pp. 553-557, Oct., 1983.

9. D. J. S. Robinson, *A Course in the Theory of Groups*, New York: Springer-Verlag, 1982, Chapter 1.

10. C. E. Shannon, "A Symbolic Analysis of Relay and Switching Circuits," *AIEE Transactions*, Vol. 57, pp. 713-723, 1938.

11. C. E. Shannon, "The Synthesis of Two-Terminal Switching Circuits," *Bell Systems Technical Journal*, Vol. 28, pp. 59-98, Jan., 1948.

12. P. A. Swartz et al., "HCAP: A Topological Analysis Program for IC Mask Artwork", in *Proceedings ICCD*, Portchester, NY, Oct., 1983.

13. S. S. Yau and Y. S. Tang, "Transformation of an Arbitrary Switching Function to a Totally Symmetric Function," *IEEE Transactions on Computers*, Vol. C-20, pp. 1606-1609, Dec., 1971.

Editors addendum:

The work done to analyze the structure of the S_4 subgroups was unbelievably tedious. A similar analysis was done for S_5, but this analysis is lost to history. It was done on paper and over the years was lost. If this sort of thing is of interest to you I suggest that you start with the conjugate elements of S_n and build groups up by starting with a single element and adding generators. The analyses done here and for S_5 were done more or less by brute force with some computer help. The computers in question were old VAX machines, so brute force may not be quite as tedious as it was in 1985 when this paper was first written.

To find the conjugate elements of S_n, one must first generate all partitions of the integer n. A partition of n is a set of positive integers in non-ascending order that add up to n. The following gives the partitions of 5 and 6.

5
4,1
3,2
3,1,1
2,2,1
2,1,1,1
1,1,1,1,1

6
5,1
4,2
4,1,1
3,3
3,2,1
3,1,1,1
2,2,2
2,2,1,1
2,1,1,1,1
1,1,1,1,1,1

Each partition corresponds to a different cycle-structure. In S_n, two elements are conjugate if and only if they have the same cycle structure. We can create a sample element for each partition as follows. Remember that cycles of the form (1) are never written.

(1,2,3,4,5)
(1,2,3,4)
(1,2,3)(4,5)
(1,2,3)
(1,2)(3,4)
(1,2)

I

(1,2,3,4,5,6)
(1,2,3,4,5)
(1,2,3,4)(5,6)
(1,2,3,4)
(1,2,3)(4,5,6)
(1,2,3)(4,5)
(1,2,3)
(1,2)(3,4),(5,6)
(1,2)(3,4)
(1,2)
I

The order of an element in cycle form is the least common multiple of the cycle-lengths, so (1,2,3,4,5) is of order 5, and (1,2,3)(4,5) is of order 6. The simplest subgroups consist of a single generator, but may contain elements with a cycle structure differing from that of the generator. This occurs when the order of the generator is a composite number. For example, the subgroup generated by (1,2,3,4,5) is

I, (1,2,3,4,5), (1,3,5,2,4), (1,4,2,5,3), (1,5,4,3,2)

While the subgroup generated by (1,2,3,4) is:

I, (1,2,3,4), (1,3)(2,4), (1,4,3,2)

For a single-generator subgroup, the subgroup will have a number of elements equal to the order of the generator.

The orbit-analysis technique is probably the most sophisticated tool presented in this paper. It can be used to analyze groups of many different kinds.