Super Symmetry

Peter M. Maurer
Dept. of Computer Science
Baylor University
Waco, Texas 76798-7356
Waco, Texas 76798

Abstract – Super symmetry is a type of matrix-based symmetry that extends the concept of total symmetry. Super symmetric functions are "even more symmetric" than totally symmetric functions. Even if a function is not super symmetric, the super symmetric transpose matrices can be used to detect partial super symmetries. These partial symmetries can be mixed arbitrarily with ordinary symmetric variable pairs to create large sets of mutually symmetric variables. In addition, one can detect subsets of super symmetric inputs, which are distinct from partial super symmetries. Super symmetry allows many new types of Boolean function symmetry to be detected and exploited.

1 Introduction

A symmetric Boolean function is a function of n variables, whose input variables can be rearranged in some fashion without changing the output of the function. An example is $abc+d$, (multiplication is AND, and addition is OR) in which the variables a, b and c can be rearranged arbitrarily.

This concept can be made more precise using permutations [1, 2]. Let f be an n-input Boolean function and $X=\{x_1,x_2,...,x_n\}$ be its set of input variables. If p is a permutation on the set X that leaves f unchanged, then f is symmetric and is said to be *invariant* with respect to p. Also, f and p are said to be *compatible*. The set of all permutations of X is called the *symmetric group* of X, and is designated S_X. The *symmetry group,* G_f, of an n-input Boolean function, f, is the set of all permutations $p\in S_X$ that are compatible with f. Because the identity permutation, which leaves X unchanged, is compatible with every function, G_f is always non-empty. A function, f, is said to be *symmetric* if G_f contains more than one element.

Symmetric Boolean functions were first studied by Shannon [3], who gave us Shannon's theorem, the basis of most symmetry detection algorithms. Shannon's theorem is based on the cofactors of a function. The cofactors of a Boolean function, f, are functions that are obtained by setting one or more input variables of f to constant values. For example, $bc+d$ is the cofactor obtained by setting a to 1 in the function $abc+d$.

Cofactors can be designated in several different ways. One can specify the variable and the value in a subscript, as in $f_{a=1}$. If there is a natural ordering to the variables, one can specify a list of variable values such as f_{xx10x}, where the x represents a variable that has not been replaced. Most often, when the variables in question are understood, we simply use lists of values as in f_0, f_1 or f_{101}.

Shannon's theorem states that two input variables, a and b, of a function f are symmetric variable pairs if and only if $f_{01}=f_{10}$, where the cofactors are taken with respect to a and b. A symmetric variable pair is a pair of variables that can be exchanged in arbitrary fashion without altering the output of the function. Symmetric variable pairs are transitive in the sense that if (a,b) is a symmetric variable pair, and (b,c), is a symmetric variable pair, then so is (a,c).

Since [3], there have been much more work on detecting and exploiting symmetric functions.[4-24]. Symmetries can be broken into three broad categories, total symmetry which allows the inputs of a function to be permuted arbitrarily, partial symmetry, which allows one or more subsets of inputs to be permuted arbitrarily, and strong symmetry, which includes everything else. Some subclasses of strong symmetry, such as hierarchical symmetry [16], and rotational symmetry [17] have been identified and studied. An algorithm for identifying any type of strong symmetry has been described in [25]

2 Super Symmetry

As pointed out in [26], permutation-based symmetry can be recast in terms of matrices over GF(2). If one views an n-input function as a function of a single n-element vector, then traditional symmetry can be defined in terms of permutation matrices on these vectors. Instead of permutations, one uses permutation matrices, which are matrices that have a single 1 in each row and in each column. A permutation matrix permutes the elements of a vector without changing them. Every permutation is a row-permutation of the identity matrix. That is, one can obtain any permutation matrix p by permuting the rows of the identity matrix, I.

There is a one-to-one correspondence between permutations and permutations, in fact, the set of all permutations on a set of n elements, S_n, and the set of all $n\times n$ permutation matrices, SR_n, are mathematical groups that are isomorphic to one another. Since the class of $n\times n$ non-singular matrices is much larger than the class of permutations on n input variables, matrices can be used to define a much larger class of symmetries than permutations.

In particular, matrices can be used to define conjugate symmetry. Let SR_n be the set of all $n\times n$ permutation matrices, (those matrices that have a single 1 in each row and each column) and let M be an arbitrary non-singular $n\times n$ matrix. Then the matrices in the set $G=\left\{M^{-1}NM \mid M\in SR_n\right\}$ define a new type of symmetry called *conjugate symmetry*. Conjugate symmetry cannot be defined directly in terms of permutations and is a type of matrix-based symmetry.

Super symmetry is another type of matrix-based symmetry that extends the concept of total symmetry. We start with SR_n, the $n\times n$ permutation matrices. Every matrix $M\in SR_n$ is both a row-permutation and a column-permutation of the identity matrix. For example, if $n=4$, then every element of SR_4 can be constructed by arranging the rows (or columns) 0001, 0010, 0100, and 1000 in some order. We can expand SR_n by adding an $n+1^{st}$ row containing all ones to the existing set of n rows. Let HR_n be the set of all matrices

that can be formed from these $n+1$ rows, without choosing duplicates. HR_n is closed under matrix multiplication, and is isomorphic to the symmetric group S_{n+1}. Figure 1 shows an example with $n=3$. By the same token, we can start with the columns containing a single 1, and add a column of all 1's. The set of all matrices that can be formed from these columns, without choosing duplicate columns, is VR_n. VR_n is also closed under matrix multiplication, and is isomorphic to S_{n+1}. If $n>2$ then $HR_n \neq VR_n$. We call HR_n and VR_n the *super symmetric groups* of degree n.

$$\begin{pmatrix}1&0&0\\0&1&0\\0&0&1\end{pmatrix}\begin{pmatrix}0&1&0\\1&0&0\\0&0&1\end{pmatrix}\begin{pmatrix}0&0&1\\0&1&0\\1&0&0\end{pmatrix}\begin{pmatrix}1&0&0\\0&0&1\\0&1&0\end{pmatrix}\begin{pmatrix}0&1&0\\0&0&1\\1&0&0\end{pmatrix}\begin{pmatrix}0&0&1\\1&0&0\\0&1&0\end{pmatrix}$$
$$\begin{pmatrix}1&1&1\\0&1&0\\0&0&1\end{pmatrix}\begin{pmatrix}1&1&1\\1&0&0\\0&0&1\end{pmatrix}\begin{pmatrix}1&1&1\\0&1&0\\1&0&0\end{pmatrix}\begin{pmatrix}1&1&1\\0&0&1\\0&1&0\end{pmatrix}\begin{pmatrix}1&1&1\\0&0&1\\1&0&0\end{pmatrix}\begin{pmatrix}1&1&1\\1&0&0\\0&1&0\end{pmatrix}$$
$$\begin{pmatrix}1&0&0\\1&1&1\\0&0&1\end{pmatrix}\begin{pmatrix}0&1&0\\1&1&1\\0&0&1\end{pmatrix}\begin{pmatrix}0&0&1\\1&1&1\\1&0&0\end{pmatrix}\begin{pmatrix}1&0&0\\1&1&1\\0&1&0\end{pmatrix}\begin{pmatrix}0&1&0\\1&1&1\\1&0&0\end{pmatrix}\begin{pmatrix}0&0&1\\1&1&1\\0&1&0\end{pmatrix}$$
$$\begin{pmatrix}1&0&0\\0&1&0\\1&1&1\end{pmatrix}\begin{pmatrix}0&1&0\\1&0&0\\1&1&1\end{pmatrix}\begin{pmatrix}0&0&1\\0&1&0\\1&1&1\end{pmatrix}\begin{pmatrix}1&0&0\\0&0&1\\1&1&1\end{pmatrix}\begin{pmatrix}0&1&0\\0&0&1\\1&1&1\end{pmatrix}\begin{pmatrix}0&0&1\\1&0&0\\1&1&1\end{pmatrix}$$

Figure 1. The Super Symmetric Group HR_3.

Before we can establish the claims made above, we must first prove that the matrices of HR_n and VR_n are non-singular. If this is not true, then the preceding claims cannot be established.

Theorem 1. *Every element of HR_n and VR_n is non-singular.*

Proof. Let $M \in HR_n$. If M is singular then some subset of the rows of M must sum to zero. If there is no row of all ones, then is a permutation and non-singular. Let us assume that row i of M is all ones. Other than row i, there are $n-1$ rows of M, each containing a single 1. These rows are part of some permutation matrix, and therefore, no subset of them can sum to zero. Other than the 1's in row i, M contains exactly $n-1$ 1's. Therefore, there must be one column of M that contains a single 1, that in row i. No sum of rows that includes row i can have a zero in column i, because every other row has a zero in this column. Therefore no subset of the rows of M sums to zero, and M is non-singular. Now consider $N \in VR_n$. By a similar argument, we can show that no subset of the columns of N can sum to zero, therefore N is nonsingular.■

Now we need to prove our claims.

Theorem 2: *HR_n and VR_n are closed under matrix multiplication, and are isomorphic to S_{n+1}.*

Proof: Let $M, N \in HR_n$ and consider the form of $K = M \times N$. Because M and N are nonsingular, K must be nonsingular. If no row of M is all ones, then M is a permutation matrix. In this case, K is a row-permutation of N, and $K \in HR_n$. So let us assume that row i of M is all 1's. Now, suppose N is a permutation matrix. Because every row of N has a single 1, every row, except row i, of K has a single 1. Row i of K is the sum of all rows of N, which is a row of all 1's. Therefore $K \in HR_n$. If N is not a permutation matrix, then it must have a row, j of all 1's. In this case, the rows of K, except for row i must be a permutation of the rows of N, not including row i. Row i of K must be the sum of the rows of N. Every column of N, except one, must have exactly 2 ones. The remaining column must have a single one. Therefore the sum of the rows of N must contain a single one in some position, and zeros elsewhere. Thus $K \in HR_n$, and HR_n is closed under multiplication. A similar argument shows that VR_n is also closed under multiplication. To show that HR_n is isomorphic to S_{n+1}, it suffices to show that HR_n is the set of permutations of a set of size $n+1$. This follows from the fact that every matrix in HR_n is a permutation of the $n+1$ rows used to form the elements of HR_n, each element, M, of HR_n has n rows from the set of $n+1$ rows. The missing row is always unique, and we can imagine it as being appended as the $n+1^{st}$ row of M. Thus HR_n is isomorphic to S_{n+1}. A similar argument on the columns of the elements of VR_n shows that VR_n is also isomorphic to S_{n+1}.■

Any finite set of matrices that is closed under multiplication is a group. Because HR_n and VR_n are groups, they can serve as the symmetry group of certain functions. We say that a function f is *super symmetric* if either HR_n or VR_n leaves f invariant. If we wish to be more specific, we will call f *H-super symmetric* or *V-super symmetric*.

3 Boolean Orbits

Let G be a group of $n \times n$ matrices. Two *n*-element vectors v and w are said to be in the same Boolean orbit of G if there is a matrix $M \in G$ such that $v \times M = w$. Being in the same Boolean orbit is an equivalence relation that breaks the set of all *n*-element vectors into a collection of disjoint subsets. The Boolean orbits of a group can be used to determine whether a group G is the compatible with a function f, because f is compatible with G if and only if f maps every element of each Boolean orbit of G to the same value. For example, the symmetric group S_3 has the Boolean orbits $\{(0,0,0)\}$, $\{(0,0,1),(0,1,0),(1,0,0)\}$, $\{(0,1,1),(1,0,1),(1,1,0)\}$ and $\{(1,1,1)\}$. A 3-input function f is totally symmetric if and only if f maps the three vectors $\{(0,0,1),(0,1,0),(1,0,0)\}$ to the same value, and the three vectors $\{(0,1,1),(1,0,1),(1,1,0)\}$ to the same value.

The Universal Symmetry Detection algorithm can detect any type of symmetry as long as the Boolean orbits of that symmetry are known. The Boolean orbits of V and H super symmetry are relatively easy to compute. Since every super symmetric function is also totally symmetric, all vectors of the same weight must be contained in a single orbit. The Boolean orbits of V and H symmetry can be obtained by combining the Boolean orbits of total symmetry.

Let us first derive the Boolean orbits of H super symmetry. We will designate the set of all vectors of weight k as W_k. The sets W_k are just the Boolean orbits of total symmetry. We will designate the Boolean orbits of H super symmetry as O_k, where k is the weight

of the lightest vector in O_k. Note that if an orbit O_i contains any vector of weight k, then $W_k \subseteq O_i$. In particular, $W_k \subseteq O_k$. Consider the orbit O_0. This orbit must contain a single vector, since every linear transformation maps the zero vector onto itself. The orbit O_1, contains all vectors of weight 1 and must also contain the vector of all 1's. Let v be an n-element vector of weight 1, and let $M \in HR_n$. The vector $v \times M$ must be equal to some row of M, and must either be a vector of weight 1 or a vector of all 1's. If v is a vector of all 1's, and M is a permutation matrix, then $v \times M$ is a vector of all 1's. If M contains a row of all 1's then $v \times M$ is the sum of the rows of M. Every column except one of M contains exactly two ones. The other column contains exactly one 1. Thus the sum of the rows of M is a vector of weight 1, and $O_1 = W_1 \bigcup W_n$. Now consider the orbit O_2 containing all vectors of weight 2. Let M be any element of HR_n. Any vector that can be formed by adding two rows i and j of M must be an element of O_2. If rows i and j of M are both of weight 1, then their sum is of weight 2 and is already contained in O_2. Let us assume that one of the rows is all ones. Then the sum of rows i and j is of weight $n-1$ and $W_{n-1} \subseteq O_2$. Now suppose that v is of weight $n-1$. If M is a permutation matrix or if M contains a row of all 1's and this row corresponds to the zero position of v, then $v \times M$ is of weight $n-1$. If M contains a row of all 1's and this row *does not* correspond to the zero element of v, then $v \times M$ is the sum of a vector of all 1's and $n-2$ distinct vectors of weight 1. Thus $v \times M$ is a vector of weight 2, and $O_2 = W_2 \bigcup W_{n-1}$. Continuing in this vein, we can show that any H super symmetry Boolean orbit, O_k, is equal to $W_k \bigcup W_{n-k+1}$, where k runs from 1 through $\left\lceil \frac{n}{2} \right\rceil$.

Now let us derive the Boolean orbits of V super symmetry. We will designate each orbit as Q_i, where i is the smallest weight of any element of Q_i. Note that if $W_j \bigcap Q_i \neq \phi$ then $W_j \subseteq Q_i$. In particular, $W_i \subseteq Q_i$. As before, Q_0 contains only the zero vector.

When a vector $v = (a_1, ..., a_n)$ is multiplied by a matrix $V_i \in VR_n$, the result is $v' = (a_1, ...a_{i-1}, p, a_{i+1} ..., a_n)$, where p is the parity of v. (i.e., p is 1 if the number of bits in v is odd.) If $a_i = 0$ and $p = 0$, or if $a_i = 1$ and $p = 1$, then $v = v'$. If $a_i = 0$ and $p = 1$ then the weight of v' is one larger than that of v. If $a_i = 1$ and $p = 0$ then the weight of v' is one smaller than that of v. Note that the weight of v can increase only if it is odd, and can decrease only if it is even. Thus $Q_i = W_i \bigcup W_{i+1}$, where i is odd, i running from 1 to m where m is the largest odd number less than or equal to n. The other matrices of VR_n will not affect these orbits because they are either permutation matrices that do not change the weight of a vector, or they are permutation matrices with a single column set to ones. Such matrices combine a permutation of v with parity insertion, and do not change the orbits described above.

We have created a super symmetry detection module to the universal symmetry detector using the Boolean orbits describe above.

4 Symmetric Variable Pairs

Although the universal symmetry detection algorithm can detect super symmetry, super symmetric functions are comparatively rare. The same is true, of course, for totally symmetric functions. However, when a function is not totally symmetric, it may be partially symmetric, and using symmetric variable pairs, we can detect such partial symmetries. By the same token, we can detect super symmetric variable pairs and partial super symmetries. The super symmetric variable pairs can be mixed arbitrarily with ordinary symmetric variable pairs.

Ordinary symmetric variable pairs correspond to a type of a permutation called a *transposition*. A transposition of a set, X, is a permutation that swaps two elements of X, leaving everything else fixed. In the matrix domain, a transposition corresponds to a *transpose matrix*. A transpose matrix swaps two elements of an input vector, leaving all other elements fixed. We designate a transpose matrix that swaps elements i and j of a vector as $T_{i,j}$. Every row, k, of $T_{i,j}$ except rows i and j, is identical to row k of the identity matrix. Row i of $T_{i,j}$ has a 1 in column j and zeros elsewhere. Row j has a 1 in column i and zeros elsewhere. Figure 2 has several examples of transpose matrices.

$$\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

Figure 2. Some transpose matrices.

Super symmetry introduces $2n$ new transpose matrices known as the *super symmetric transpose matrices*. Half of these matrices are taken from HR_n and half are taken from VR_n.

In permutation matrices, we consider a row containing a 1 in position i and zeros elsewhere to represent the i^{th} input variable. Alternatively, we could consider a column containing a 1 in the i^{th} position to represent the i^{th} input variable. In the super symmetric matrices, we consider the row of all 1's or a column of all 1's to represent an $n+1^{st}$ "invisible" variable. In HR_n a super symmetric transpose matrix is a matrix that is identical to the identity matrix except for row i, which is a row of all 1's. In VR_n a super symmetric transpose matrix is identical to the identity matrix except for column i which is a column of all 1's. We designate these matrices as V_i and H_i respectively. Figure 3 gives some examples of such matrices.

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$$

Figure 3. Super Symmetric Transpose Matrices.

For any ordinary transpose matrix $T_{i,j}$, the matrix is self-inverting. That is, $T_{i,j} T_{i,j} = I$. As the Theorem 3 shows, the same is true for the matrices V_i and H_i.

Theorem 3. $H_i H_i = I$ *and* $V_i V_i = I$ *for all* $1 \le i \le n$.

Proof: Since H_i is identical to the identity matrix, except for row i, every row of $H_i H_i$ is identical to the identity matrix, except for row i. Because row i of H_i is all ones, row i of $H_i H_i$ is the

sum of the rows of H_i. Every column of H_i contains exactly two 1's, except for column i which contains exactly one 1. Thus the sum of the rows of H_i has a 1 in column i and zeros elsewhere, and is equal to row i of the identity matrix.

Similarly, since every column k of V_i, except for column i, is identical to column k of the identity matrix, every column k of V_iV_i is identical to column k of the identity matrix. Because column i of V_i is all ones, column i of V_iV_i is the sum of the columns of V_i. Every row of V_i, except row i has exactly two 1's. Row i has exactly one 1. Therefore the sum of the columns of V_i has a 1 in row i and zeros elsewhere, and column i of V_iV_i is identical to column i of the identity matrix.■

It is convenient to think of the matrices H_i and V_i as being transpose matrices between x_i and the "invisible" $n+1^{st}$ variable, x_{n+1}. This makes the transitivity of the new matrices more obvious. For example, because of transitivity, a function is H super symmetric if it is compatible with $T_{1,2}$, $T_{1,3}$, …, $T_{1,n}$ and H_1. For V super symmetry, we substitute V_1 for H_1.

Another important and useful property of the super symmetric transpose matrices is that the conjugate of any matrix H_i with another matrix H_j $(i \neq j)$ is an ordinary transpose matrix. The same is true for matrices V_i and V_j, as the following theorem shows.

Theorem 4. *Suppose* $i \neq j$. *Then* $H_j^{-1}H_iH_j = T_{i,j}$ *and* $V_j^{-1}V_iV_j = T_{i,j}$.

Proof: By Theorem 3, $H_j^{-1} = H_j$ and $V_j^{-1} = V_j$, so $H_j^{-1}H_iH_j = H_jH_iH_j$ and $V_j^{-1}V_iV_j = V_jV_iV_j$. H_jH_i has the following form. Since every row of H_j, except row, is identical to the corresponding row of the identity matrix, every row, except row j of H_jH_i is identical to the corresponding row of H_i. Because row j of H_j is all ones, row j of H_jH_i is the sum of the rows of H_i. Every column of H_i has exactly two 1's, except for column i, which has exactly one 1. Thus row j of H_jH_i has a one in column i and zeros elsewhere. Row i of H_jH_i contains all 1's. We can use the structure of H_jH_i to deduce the structure of $H_jH_iH_j$. Because every row of H_jH_i except rows i and j is identical to the corresponding row of the identity matrix, every row of $H_jH_iH_j$, except rows i and j is identical to the corresponding row of the identity matrix. Because row j has a 1 in column i and zeros elsewhere, row j of $H_jH_iH_j$ is identical to row i of H_j, and has a 1 in column i and zeros elsewhere. Because row i of H_jH_i contains all 1's, row i of $H_jH_iH_j$ is the sum of the rows of H_j. Every column of H_j has exactly two ones, except for column j which has exactly one 1. Therefore row i of $H_jH_iH_j$ has a 1 in column j and zeros elsewhere. Therefore $H_jH_iH_j$ is the transpose matrix $T_{i,j}$.

Now consider the structure of V_iV_j. Because every column of V_j is identical to the corresponding column of the identity matrix, except for column j, every column of V_iV_j is identical to the corresponding column of V_i, except for column j. Because column j of V_i is all ones, column j of V_iV_j is equal to the sum of the columns of V_i. Every column of V_i contains exactly two 1's, except for column i which has exactly one 1. Thus column j of V_iV_j has a 1 in row i and zeros elsewhere. Column i of V_iV_j contains all ones. We can now deduce the structure of $V_jV_iV_j$. Since every column of V_iV_j except for columns i and j, is identical to the corresponding column of the identity matrix, every column of $V_jV_iV_j$, except for columns i and j, is identical to the corresponding column of V_j. But these columns are identical to the corresponding columns of the identity matrix, so every column of $V_jV_iV_j$, except for columns i and j, is identical to the corresponding column of the identity matrix. Column j of $V_jV_iV_j$ is equal to column i of V_j, which has a 1 in row i and zeros elsewhere. Because column i of V_iV_j is all ones, column i of $V_jV_iV_j$ is the sum of the columns of V_j. Every row of V_j has exactly two 1's, except for row j, which has exactly one 1. Thus the sum of the columns of V_j has a 1 in row j and zeros elsewhere. Thus $V_jV_iV_j$ is the transpose matrix $T_{i,j}$.■

Let f be an n-input function with input variables $\{x_1, x_2, ..., x_n\}$. To determine whether f is compatible with H_i, we select some variable other than x_i, say x_j with $i \neq j$, and conditionally invert every variable except x_j itself with respect to x_j. These conditional inversions can be done simultaneously using the matrix H_j. We compute $f'(v) = f(H_j(v))$. The function f is compatible with H_i if and only if (x_i, x_j) is a symmetric variable pair of f'. The correctness of this procedure stems from the fact that if $i \neq j$ then $H_j^{-1}H_iH_j = T_{i,j}$. Super symmetry can be viewed as a type of conjugate symmetry requiring multiple simultaneous conditional inversions.

Given the same function, f, we can determine whether f is compatible with V_i by selecting any input variable other than x_i, say x_j with $i \neq j$, and conditionally invert x_j with respect to every other variable other than x_j itself. This gives us the new function, $f''(v) = f(V_i(v))$. The function f is compatible with V_i if and only if (x_i, x_j) is a symmetric variable pair of f''. Again, the correctness of this procedure depends on the fact that $V_j^{-1}V_iV_j = T_{i,j}$.

Because super symmetric transpose matrices can be equated with a type of conjugate symmetry, they can be detected and utilized by the hyperlinear algorithm for digital simulation [26, 27], and by other algorithms that detect symmetry using symmetric variable pairs.

5 Sub-Symmetries

For a Boolean function f to possess X symmetry in variables $\{x_1, x_2, ..., x_k\}$ every cofactor of the form $f_{xx...xa_{k+1}...a_n}$ must possess X

symmetry. We usually do this by ensuring the symmetry relations exist between cofactors of the form $f_{a_1 \ldots a_k xx \ldots x}$. It is possible for an *n*-input function to be super symmetric in any proper subset of its input variables, and it is possible for a function to have several subsets of variables in which it is super symmetric.

This is not the same as partial symmetry, because the super symmetric variable pairs involve all inputs of a function, while sub-super symmetries involve only a subset of variables. It is possible to test a subset of variables for super symmetry, and to test the same subset for compatibility with the super symmetric transpose matrices of the sub-symmetry. This gives us many more opportunities to detect symmetries in a Boolean function, because there are $2^n - 2$ proper subsets of variables, and $\sum_{i=2}^{n-1} 2i = 2\left(\frac{n(n-1)}{2} - 1\right) = n^2 - n - 2$ additional super symmetric transpose matrices.

6 Experimental Data

To determine the prevalence of super symmetry in real circuits, we tested the ISCAS 85 benchmarks for the presence of super symmetries. We tested for total super symmetry, for super symmetric variable pairs, and for sub symmetries. The results of our tests are given in Figure 4. These results show that super symmetries do indeed exist in real circuits, and are, in fact, quite numerous. The results for super symmetric variable pairs and for sub symmetries are especially encouraging. Because, in several cases, the number of symmetries exceeds the number of functions, it is clear that there are many functions that exhibit multiple sub-super symmetries and that there are functions that are compatible with many super symmetric variable pairs.

Circuit	Super Sym.	Var. Pairs	Sub-Sym.
c432	78	213	1097
c499	0	56	728
c880	122	33	902
c1355	288	44	704
c1908	158	59	5326
c2670	276	90	3145
c3540	710	1310	2093
c5315	830	2313	6206
c6288	512	528	928
c7552	582	1660	10093

Figure 4. Experimental Results.

7 Conclusion

The various aspects of super symmetry allow many different types of Boolean function symmetry to be detected and exploited. In addition to super symmetry itself we have partial super symmetries which are generated by the super symmetric transposition matrices. These partial symmetries can be mixed and matched in an arbitrary fashion with ordinary symmetric variable pairs. In addition, there are sub-super symmetries and partial sub-super symmetries which greatly expand the opportunity for detecting and exploiting symmetries in a Boolean function.

What is even more exciting, super symmetry allows us to exploit more of the full power of matrix-based symmetry. For example, for 4-input functions, there are 24 permutations of the inputs, but 20160 non-singular 4×4 matrices. There are obviously many more kinds of matrix-based symmetry than permutation-based symmetry, and super symmetry is only one of these.

We expect this work to be the basis of much more extended work in the future.

8 References

[1] D. S. Passman, *Permutation Groups.* New York: W. A. Benjamin, 1968.

[2] D. Robinson, *A Course in the Theory of Groups.* New York: Springer, 1995.

[3] C. E. Shannon, "The synthesis of two-terminal switching circuits," *Bell System Technical Journal,* vol. 28, pp. 59-98, 1949.

[4] A. Abdollahi and M. Pedram, "Symmetry detection and Boolean matching utilizing a signature-based canonical form of Boolean functions," *IEEE Trans. on Computer-Aided Design,* vol. 27, pp. 1128-1137, June, 2008.

[5] N. N. Biswas, "On Identification of Totally Symmetric Boolean Functions," *Computers, IEEE Transactions On,* vol. 19, pp. 645-648, 1970.

[6] R. C. Born and A. K. Scidmore, "Transformation of switching functions to completely symmetric switching functions," *IEEE Transactions on Computers,* vol. 17, pp. 596-599, 1968.

[7] J. T. Butler, G. W. Dueck, V. P. Shmerko and S. Yanuskevich, "Comments on "Sympathy: fast exact minimization of fixed polarity Reed-Muller expansion for symmetric functions"," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 19, pp. 1386-1388, 2000.

[8] M. Chrzanowska-Jeske, "Generalized symmetric variables," in *The 8th IEEE International Conference on Electronics, Circuits and Systems,* 2001, pp. 1147-1150.

[9] K. S. Chung and C. L. Liu, "Local transformation techniques for multi-level logic circuits utilizing circuit symmetries for power reduction," in *Proceedings of the 1998 International Symposium on Low Power Electronics and Design,* 1998, pp. 215-220.

[10] P. T. Darga, K. A. Sakallah and I. L. Markov, "Faster symmetry discovery using sparsity of symmetries," in *Proceedings of the 45th Annual Design Automation Conference,* 2008, pp. 149-154.

[11] R. Drechsler and B. Becker, "Sympathy: Fast exact minimization of fixed polarity reed-muller expressions for symmetric functions," in *European Design and Test Conference,* 1995, pp. 91-97.

[12] Y. Hu, V. Shih, R. Majumdar and L. He, "Exploiting Symmetries to Speed Up SAT-Based Boolean Matching for Logic Synthesis of FPGAs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 27, pp. 1751-1760, 2008.

[13] B. Hu and M. Marek-sadowska, "In-place delay constrained power optimization using functional symmetries," in *Design Automation and Test in Europe,* 2001, pp. 377-382.

[14] W. Ke and P. R. Menon, "Delay-testable implementations of symmetric functions," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 14, pp. 772-775, 1995.

[15] N. Kettle and A. King, "An anytime algorithm for generalized symmetry detection in ROBDDs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 27, pp. 764-777, 2008.

[16] V. N. Kravets and K. A. Sakallah, "Generalized symmetries in boolean functions," Advanced Computer Architecture Laboratory Department of Electrical Engineering and Computer Science, The University of Michigan, Ann Arbor, MI 48109, 2002.

[17] J. Mohnke, P. Molitor and S. Malik, "Limits of using signatures for permutation independent Boolean comparison," *Formal Methods Syst. Des.,* vol. 21, pp. 167-191, 2002.

[18] D. Moller, J. Mohnke and M. Weber, "Detection of symmetry of boolean functions represented by ROBDDs," in *IEEE International Conference on Computer-Aided Design,* 1993, pp. 680-684.

[19] J. C. Muzio, D. M. Miller and S. L. Hurst, "Multivariable symmetries and their detection," *IEE Proceedings on Computers and Digital Techniques,* vol. 130, pp. 141-148, 2008.

[20] S. Panda, F. Somenzi and B. F. Plessier, "Symmetry detection and dynamic variable ordering of decision diagrams," in *IEEE International Conference on Computer-Aided Design,* 1994, pp. 628-631.

[21] J. Rice and J. Muzio, "Antisymmetries in the realization of boolean functions," in *IEEE International Symposium on Circuits and Systems,* 2002, pp. 69-72.

[22] C. Scholl, D. Moller, P. Molitor and R. Drechsler, "BDD minimization using symmetries," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 18, pp. 81-100, 1999.

[23] C. C. Tsai and M. Marek-Sadowska, "Generalized Reed-Muller forms as a tool to detect symmetries," *IEEE Transactions on Computers,* vol. 45, pp. 33-40, 1996.

[24] K. H. Wang and J. H. Chen, "Symmetry detection for incompletely specified functions," in *Proceedings of the 41st Annual Design Automation Conference,* 2004, pp. 434-437.

[25] P. Maurer, "A universal symmetry detection algorithm," Baylor University, 2013.

[26] P. M. Maurer, "Conjugate Symmetry," *Formal Methods Syst. Des.,* vol. 38, pp. 263-288, 2011.

[27] P. M. Maurer. Efficient event-driven simulation by exploiting the output observability of gate clusters. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions On 22(11),* pp. 1471-1486. 2003.