Discrete Mathematics and Theoretical Computer Science DMTCS vol. **21**:4, 2019, #19

Symmetry Properties of Nested Canalyzing Functions

Daniel J. Rosenkrantz[1] Madhav V. Marathe[2] S. S. Ravi[1] Richard E. Stearns[1]

[1] *Biocomplexity Institute and Initiative, University of Virginia and Department of Computer Science, University at Albany – State University of New York, USA*
[2] *Biocomplexity Institute and Initiative & Department of Computer Science, University of Virginia, USA*

received 27th June 2019, revised 2nd Oct. 2019, accepted 29th Oct. 2019.

Many researchers have studied symmetry properties of various Boolean functions. A class of Boolean functions, called **nested canalyzing functions** (NCFs), has been used to model certain biological phenomena. We identify some interesting relationships between NCFs, symmetric Boolean functions and a generalization of symmetric Boolean functions, which we call r-symmetric functions (where r is the symmetry level). Using a normalized representation for NCFs, we develop a characterization of when two variables of an NCF are symmetric. Using this characterization, we show that the symmetry level of an NCF f can be easily computed given a standard representation of f. We also present an algorithm for testing whether a given r-symmetric function is an NCF. Further, we show that for any NCF f with n variables, the notion of strong asymmetry considered in the literature is equivalent to the property that f is n-symmetric. We use this result to derive a closed form expression for the number of n-variable Boolean functions that are NCFs and strongly asymmetric. We also identify all the Boolean functions that are NCFs and symmetric.

Keywords: Boolean functions, Nested canalyzing functions, Symmetry, Algorithms.

1 Introduction

1.1 Canalyzing and Nested Canalyzing Functions

Research on the symmetry properties of Boolean functions has received a lot of attention from the logic design and automated synthesis communities. (Additional discussion on this is provided in Section 1.4.) In this paper, we focus on the symmetry properties of a class of Boolean functions (called **nested canalyzing functions**) which are used to model biological phenomena. To define this class, we start with a simpler class of functions, called **canalyzing** Boolean functions. This class, introduced by Kauffman (1969), is defined as follows.

Definition 1.1 *Given a set $X = \{x_1, x_2, \ldots, x_n\}$ of n Boolean variables, a Boolean function $f(x_1, x_2, \ldots, x_n)$ over X is **canalyzing** if there is a variable $x_i \in X$ and values a, $b \in \{0, 1\}$ such that*

$$f(x_1, \ldots, x_{i-1}, a, x_{i+1}, \ldots, x_n) \;=\; b,$$

for all combinations of values assigned to the variables in $X - \{x_i\}$.

ISSN 1365–8050

Example 1: Consider the function $f(x_1, x_2, x_3) = \overline{x_1} \wedge (x_2 \vee \overline{x_3})$. This function is canalyzing since if we set to $x_1 = 1$, $f(1, x_2, x_3) = 0$, for all combinations of values for x_2 and x_3.

Another class of Boolean functions, called **nested canalyzing functions** (NCFs), was introduced later by Kauffman et al. (2003) to carry out a detailed analysis of the behavior of certain biological systems. We follow the presentation in Layne (2011) in defining NCFs. (For a Boolean value b, the complement is denoted by $\overline{b}$.)

Definition 1.2 *Let $X = \{x_1, x_2, \ldots, x_n\}$ denote a set of n Boolean variables. Let π be a permutation of $\{1, 2, \ldots, n\}$. A Boolean function $f(x_1, x_2, \ldots, x_n)$ over X is **nested canalyzing** in the variable order $x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)}$ with **canalyzing values** $a_1, a_2, \ldots, a_n$ and **canalyzed values** $b_1, b_2, \ldots, b_n$ if f can be expressed in the following form:*

$$f(x_1, x_2, \ldots, x_n) = \begin{cases} b_1 & \text{if } x_{\pi(1)} = a_1 \\ b_2 & \text{if } x_{\pi(1)} \neq a_1 \text{ and} \\ & x_{\pi(2)} = a_2 \\ \vdots & \vdots \\ b_n & \text{if } x_{\pi(1)} \neq a_1 \text{ and } \ldots \\ & x_{\pi(n-1)} \neq a_{n-1} \text{ and} \\ & x_{\pi(n)} = a_n \\ \overline{b_n} & \text{if } x_{\pi(1)} \neq a_1 \text{ and } \ldots \\ & x_{\pi(n)} \neq a_n. \end{cases}$$

For convenience, we will use a notation introduced in Stearns et al. (2018) to represent NCFs. For $1 \leq i \leq n$, line i of this representation has the form

$$x_{\pi(i)} : a_i \longrightarrow b_i$$

where $x_{\pi(i)}$ is the **canalyzing variable** that is **tested** in line i, and a_i and b_i are respectively the *canalyzing* and *canalyzed* values in line i, $1 \leq i \leq n$. Each such line is called a **rule**. When none of the conditions "$x_{\pi(i)} = a_i$" is satisfied, we have line $n+1$ with the "Default" rule for which the canalyzed value is $\overline{b_n}$:

Default: $\overline{b_n}$

As in Stearns et al. (2018), we will refer to the above specification of an NCF as the **simplified representation** and assume (without loss of generality) that each NCF is specified in this manner. The simplified representation provides the following computational view of an NCF. The lines that define an NCF are considered sequentially in a top-down manner. The computation stops at the first line where the specified condition is satisfied, and the value of the function is the canalyzed value on that line. We now present an example of an NCF using the two representations mentioned above.

Example 2: Consider the function $f(x_1, x_2, x_3) = \overline{x_1} \wedge (x_2 \vee \overline{x_3})$ used in Example 1. This function is nested canalyzing using the identity permutation π on $\{1, 2, 3\}$ with canalyzing values $1, 1, 0$ and canalyzed values $0, 1, 1$. We first show how this function can be expressed using the syntax of Definition 1.2.

$$f(x_1, x_2, x_3) = \begin{cases} 0 & \text{if } x_1 = 1 \\ 1 & \text{if } x_1 \neq 1 \text{ and } x_2 = 1 \\ 1 & \text{if } x_1 \neq 1 \text{ and } x_2 \neq 1 \text{ and} \\ & x_3 = 0 \\ 0 & \text{if } x_1 \neq 1 \text{ and } x_2 \neq 1 \text{ and} \\ & x_3 \neq 0 \end{cases}$$

A simplified representation of the same function is as follows.

$x_1: \ 1 \longrightarrow 0$
$x_2: \ 1 \longrightarrow 1$
$x_3: \ 0 \longrightarrow 1$
Default: 0

1.2 Symmetric Boolean Functions

A pair of variables of a Boolean function $f(x_1, x_2, \ldots, x_n)$ is said to be **symmetric** if their values can be interchanged without affecting the value of the function. As a simple example, each pair of variables in the function $f_2(x_1, x_2, x_3) = x_1 \oplus x_2 \oplus x_3$ is symmetric, where '$\oplus$' represents the Exclusive-Or operator. This form of interchange symmetry partitions the set of variables into a set of **symmetry groups**, where the members of each group are pairwise symmetric. A Boolean function f is r-**symmetric** if it has at most r symmetry groups. In this case, the value of f depends only on how many of the variables in each symmetry group have the value 1. We say that f is **properly** r-**symmetric** if it is r-symmetric, but not $(r-1)$-symmetric. For example, the function $f_3(x_1, x_2, x_3) = (x_1 \wedge x_2) \vee \overline{x_3}$ is not 1-symmetric since $f_3(1, 0, 1) \neq f_3(1, 1, 0)$; however, it is 2-symmetric with the symmetric groups being $\{x_1, x_2\}$ and $\{x_3\}$. For a Boolean function f, the integer r such that f is properly r-symmetric is referred to as the **symmetry level** of f. Thus, the symmetry level of f is the smallest integer r such that f is r-symmetric.

In the literature (see e.g., Crama and Hammer (2011); Haviarova and Toman (2016); Ecsedi-Tóth et al. (1977)), a 1-symmetric function f is referred to simply as a **symmetric** function or as a **totally symmetric function** (Biswas, 1970; Born and Scidmore, 1968). Thus, in any symmetric function, each pair of variables is symmetric. As a simple example, the function $f_2(x_1, x_2, x_3) = x_1 \oplus x_2 \oplus x_3$ (defined above) is symmetric. If f is a symmetric function, then for any input $(a_1, a_2, \ldots, a_n)$ to f, where $a_i \in \{0, 1\}$ for $1 \leq i \leq n$, and any permutation π of $\{1, 2, \ldots, n\}$, $f(a_1, a_2, \ldots, a_n) = f(a_{\pi(1)}, a_{\pi(2)}, \ldots, a_{\pi(n)})$. For fixed $r \geq 1$, the class of r-symmetric functions has been studied in the literature on discrete dynamical systems (see e.g., Barrett et al. (2007); Rosenkrantz et al. (2015); Mortveit and Reidys (2007)).

Other forms of symmetry which can be more general than the permutations corresponding to symmetry groups have also been considered (see e.g., Maurer (2015); Kravets and Sakallah (2000)). A Boolean function $f(x_1, x_2, \ldots, x_n)$ is **strongly asymmetric** if for any permutation π of $\{1, 2, \ldots, n\}$ *except* the identity permutation, there exists an input $(a_1, a_2, \ldots, a_n)$ to f such that $f(a_1, a_2, \ldots, a_n) \neq f(a_{\pi(1)}, a_{\pi(2)}, \ldots, a_{\pi(n)})$. In general, there are Boolean functions with n variables that are properly n-symmetric, but not strongly asymmetric (Kravets and Sakallah, 2000). However, in the case of NCFs with n variables, we will show (see Section 3) that the notion of strong asymmetry coincides with that of being properly n-symmetric.

1.3 Summary of Results

Our focus is on the relationships between NCFs and symmetric Boolean functions. Using a normalized representation for NCFs (defined in Section 2), we develop a characterization of when two variables of an NCF are symmetric. Using this characterization, we show that the symmetry level of an NCF f can be easily computed given a normalized representation of f. (In contrast, we show that one cannot even efficiently approximate the symmetry level of a general Boolean function to within any factor ≥ 1, unless **P = NP**.) We also present an algorithm to test whether a given r-symmetric function f is an NCF. Further, we show that for any NCF f with n variables, the notion of strong asymmetry considered in the literature is equivalent to the property that f is n-symmetric. We use this result to derive a closed form expression for the number of n-variable Boolean functions that are NCFs and strongly asymmetric. In addition, we identify all the Boolean functions that are canalyzing and symmetric as well as those that are NCFs and symmetric.

1.4 Related Work

The usefulness of symmetry properties in synthesizing combinational logic functions was first noted by Shannon (1938). Over the years, a considerable amount of research on detecting and exploiting symmetry properties of Boolean functions has been reported in the literature. For example, Biswas (1970) and Tsai and Marek-Sadowska (1996) present methods to determine whether a given Boolean function is symmetric. Born and Scidmore (1968) show how certain Boolean functions can be converted into symmetric functions by introducing additional variables. Several researchers have developed techniques for exploiting symmetries for automated logic synthesis (e.g., Abdollahi and Pedram (2008); Hu et al. (2008); Kravets and Sakallah (2000); Darga et al. (2008)). Maurer (2015) provides a thorough discussion of known symmetry detection algorithms and presents a new universal algorithm for detecting any form of permutation-based symmetry in Boolean functions.

The term **canalization**, coined by Waddington (1942), is generally used to describe the stability of a biological system with changes in external conditions. Kauffman (1969) introduced canalyzing Boolean functions to explain the stability of gene regulatory networks. The subclass of NCFs was introduced later by Kauffman et al. (2003) to facilitate a rigorous analysis of the Boolean network model for gene regulatory networks. It is known that the class of NCFs coincides with that of unate cascade Boolean functions (Jarrah et al., 2007). In the literature on computational learning theory, NCFs are referred to as 1-**decision lists** (Kearns and Vazirani, 1994). Many researchers have pointed out the usefulness of NCFs in modeling biological phenomena (e.g., Layne (2011); Layne et al. (2012); Li et al. (2011); Li and Adeyeye (2012); Li et al. (2013)). Properties of NCFs such as sensitivity and stability have also been studied in the literature (e.g., Kauffman et al. (2004); Layne (2011); Layne et al. (2012); Li et al. (2011, 2013); Klotz et al. (2013); Stearns et al. (2018); Paul et al. (2019); Kadelka et al. (2017a)). Generalized versions of nested canalyzing functions where the variables and function values can be from a domain of size three or more have also been studied (e.g., Kadelka et al. (2017b)).

To our knowledge, relationships between NCFs and symmetric Boolean functions have not been addressed in the literature.

2 Other Definitions and Preliminary Results

2.1 Overview

We present some definitions and recall a known result regarding NCFs which allow us to define a normalized representation for NCFs. This representation plays an important role in several problems considered in later sections. We also show (Theorem 2.5) that, in general, the symmetry level of a Boolean function cannot be efficiently approximated to within any factor $\rho \geq 1$, unless **P = NP**.

2.2 Layers and Normalized Representation

Recall that any NCF f with n variables, denoted by $x_1, x_2, \ldots, x_n$, is specified using a variable ordering $x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)}$, where π is a permutation of $\{1, 2, \ldots, n\}$. Throughout this paper, we will assume without loss of generality that the variable ordering uses the identity permutation so that x_i is the variable tested in line i, $1 \leq i \leq n$. We assume that an NCF f with n variables is specified using the simplified representation with n lines. Unless otherwise mentioned, the "Default" line (which is assigned the number $n+1$) in the simplified representation of an NCF is *not* considered in the results stated in this section. We will use the following result from Stearns et al. (2018).

Observation 2.1 *Let f be an NCF with n variables specified using n lines in the simplified representation.*

1. *For any $q \geq 2$ and for any i, $1 \leq i \leq n-q+1$, if the q consecutive lines i, $i+1$, ..., $i+q-1$ have the same canalyzed value, then the function remains unchanged if these q lines are permuted in any order without changing the other lines.*

2. *Suppose lines $n-1$ and n in the specification of f have complementary canalyzed values. Then, the function remains unchanged if the canalyzing value and canalyzed value in line n are both complemented. (Here, the value on the "Default" line is also complemented.)* ■

Li et al. (2013) defined the concept of a **layer** of an NCF in terms of an algebraic representation of the NCF as an extended monomial. For our purposes, we use the following definition based on the simplified representation of NCFs.

Definition 2.2 *A* **layer** *of an NCF representation is a maximal length sequence of lines with the same canalyzed value.*

Example 4: Consider the following function f of six variables $x_1, x_2, \ldots, x_6$.

$x_1: \ 1 \longrightarrow 0$
$x_2: \ 0 \longrightarrow 0$
$x_3: \ 0 \longrightarrow 0$
$x_4: \ 1 \longrightarrow 1$
$x_5: \ 1 \longrightarrow 1$
$x_6: \ 1 \longrightarrow 0$
Default: 1

This function has 3 layers, with Layer 1 consisting of the lines with x_1, x_2 and x_3, Layer 2 consisting of the lines with x_4 and x_5, and Layer 3 consisting of the line with x_6.

From Part 1 of Observation 2.1, it follows that the lines within the same layer of an NCF representation can be permuted without changing the function. Part 2 of Observation 2.1 points out that for any NCF f with $n \geq 2$ variables, there is a simplified representation in which lines $n-1$ and n are in the same layer. We can now define the notion of a **default-normalized** representation for NCFs.

Definition 2.3 *Let f be an NCF with $n \geq 2$ variables specified using the simplified representation with the variable ordering $\langle x_1, x_2, \ldots, x_n \rangle$. We say that the representation is* **default-normalized** *if lines $n-1$ and n have the same canalyzed value.*

Example 5: Consider the NCF f shown in Example 4. This representation is *not* default-normalized since lines 5 and 6 have different canalyzed values. If we change line 6 to "x_6: $0 \longrightarrow 1$" and the value on the "Default" line to 0, then we obtain a default-normalized representation.

From Observation 2.1, we note that in polynomial time, for any NCF representation with more than one line, we can first modify the last line if necessary so that it has the same canalyzed value as the preceding line, thereby obtaining an equivalent default-normalized NCF representation. We state this formally below.

Observation 2.4 *Given an NCF representation for a Boolean function f, a default-normalized representation for f can be obtained in polynomial time.* ■

In view of Observation 2.4, we assume henceforth that any NCF is specified in default-normalized form.

2.3 Complexity of Approximating the Symmetry Level of a General Boolean Function

Recall that the symmetry level of a Boolean function f is the smallest integer r such that the following condition holds: the inputs to f can be partitioned into r subsets (symmetry groups) so that the value of f depends only on how many of the inputs in each group have the value 1. We now show that given a Boolean function f in the form of a Boolean expression, it is **NP**-hard to approximate[(i)] the symmetry level of f to within any factor $\rho \geq 1$. This result holds even when the function is given as an expression in conjunctive normal form (CNF), that is, in the form of a conjunction of clauses where each clause is a disjunction of literals.

Theorem 2.5 *Unless **P = NP**, for any $\rho \geq 1$, there is no polynomial time algorithm for approximating the symmetry level of a Boolean function f specified as a Boolean expression to within the factor ρ. This result holds even when f is a CNF expression.*

Proof: For some $\rho \geq 1$, suppose there is an efficient algorithm $\mathcal{A}$ that approximates the symmetry level of a Boolean function specified as a Boolean expression to within the factor ρ. Without loss of generality, we can assume that ρ is an integer. We will show that $\mathcal{A}$ can be used to efficiently solve the CNF Satisfiability problem (SAT) which is known to be **NP**-hard (Garey and Johnson, 1979).

Let g be a CNF formula representing an instance of SAT. Let $X = \{x_1, x_2, \ldots, x_n\}$ denote the set of Boolean variables used in g. We create another CNF formula f as follows. Let $Y = \{y_1, y_2, \ldots, y_{\rho+1}\}$ and $Z = \{z_1, z_2, \ldots, z_{\rho+1}\}$ be two new sets of variables, with each set containing $\rho + 1$ variables.

[(i)] An algorithm approximates the symmetry level of a Boolean function f within the factor ρ if it finds a partition of the inputs to f into at most $\rho \times r^*$ symmetry groups, where r^* is the symmetry level of f.

The expression for f, which is a function of $n + 2\rho + 2$ variables, namely $x_1, \ldots, x_n$, $y_1, \ldots, y_{\rho+1}$, $z_1, \ldots, z_{\rho+1}$, is the following:

$$g(x_1, \ldots x_n) \wedge (y_1 \vee \overline{z_1}) \wedge (y_2 \vee \overline{z_2}) \wedge \ldots \wedge (y_{\rho+1} \vee \overline{z_{\rho+1}}).$$

Since g is a CNF formula, so is f. We have the following claims.

Claim 1: If g is *not* satisfiable, the symmetry level of f is 1.

Proof of Claim 1: If g is not satisfiable, f is also not satisfiable; that is, for all inputs, the value of f is 0. Thus, all the variables in $X \cup Y \cup Z$ can be included in one symmetry group. The value of f is 0 regardless of how many variables in the group have the value 1. □

Claim 2: If g *is* satisfiable, the symmetry level of f is at least $\rho + 1$.

Proof of Claim 2: Suppose g is satisfiable. We argue that if $i \neq j$, variables y_i and y_j cannot be in the same symmetry group for the function f. To see this, consider any two variables y_i and y_j with $i \neq j$, and construct an assignment α to the variables of f in the following manner.

1. For the variables in X, choose any assignment that satisfies g.
2. Let $y_i = 0$, $z_i = 1$, $y_j = 1$ and $z_j = 0$.
3. For $1 \leq p \leq \rho + 1$, $p \neq i$ and $p \neq j$, let $y_p = 1$ and z_p = 0.

Since this assignment α sets the clause $(y_i \vee \overline{z_i})$ to 0, the value of f under the assignment α is 0. Now, consider the assignment α' that is obtained by interchanging the values of y_i and y_j in α without changing the values assigned to the other variables. It can be seen that under assignment α', the value of f is 1. In both α and α', the number of 1-valued variables in the set $\{y_i, y_j\}$ is 1; however, the two assignments lead to different values for f. Therefore, if g is satisfiable, for $i \neq j$, y_i and y_j cannot be in the same symmetry group for f. Since there are $\rho + 1$ variables in Y, the number of symmetry groups for f must be at least $\rho + 1$, and this completes our proof of Claim 2. □

We now continue with the proof of Theorem 2.5. Suppose we execute the approximation algorithm $\mathcal{A}$ on the function f defined above. If g is *not* satisfiable, then from Claim 1, the symmetry level of f is 1. Since $\mathcal{A}$ is a ρ-approximation algorithm, it should produce at most ρ symmetry groups. On the other hand, if g is satisfiable, by Claim 2, the symmetry level of f is at least $\rho + 1$; so, $\mathcal{A}$ will produce at least $\rho + 1$ groups. In other words, g is not satisfiable iff the number of symmetry groups produced by $\mathcal{A}$ is at most ρ. Since $\mathcal{A}$ runs in polynomial time, we have an efficient algorithm for SAT, contradicting the assumption that **P** $\neq$ **NP**. ■

In contrast to the above result, we will show in the next section that the symmetry level of an NCF can be computed efficiently.

3 Symmetry of Nested Canalyzing Functions

3.1 Overview

Our first result (Theorem 3.1) provides a characterization of pairs of variables of an NCF that are part of the same symmetry group. This characterization allows us to give a simple closed form expression

(Theorem 3.2) for the symmetry level of an NCF specified using its default-normalized representation. We also present an algorithm (Theorem 3.6) for the converse problem, that is, testing whether a given r-symmetric function f is an NCF; if so, our algorithm constructs a default-normalized representation for f. Next, we show that for any NCF f with n variables, the two statements "f is strongly asymmetric" and "the symmetry level of f is n" are equivalent (Theorem 3.7). We use this result to derive a closed form expression for the number of n-variable NCFs that are also strongly asymmetric (Theorem 3.8). Our final result (Proposition 3.9) identifies all the functions that are symmetric and canalyzing as well as those that are symmetric and NCFs.

3.2 Symmetric Pairs of Variables in an NCF: A Characterization and its Applications

Theorem 3.1 *Two variables of an NCF f are symmetric iff in the default-normalized representation of f, they occur in the same layer and have the same canalyzing value.*

Proof: Suppose that two variables of an NCF f occur in the same layer of a default-normalized representation, and the lines containing these variables have the same canalyzing value. Then in any input assignment, the values of these two variables can be interchanged, and the evaluation of the lines in that layer will have the same effect. Thus, the variables are symmetric.

For the converse, consider the default-normalized NCF representation of f. As mentioned earlier, we assume without loss of generality that the canalyzing variable in line i of f is x_i, $1 \leq i \leq n$. Let x_j and x_k be two variables that occur in different layers of f, or occur in the same layer, but with different canalyzing values. Without loss of generality, assume that $j < k$, so that the line for x_j occurs above the line for x_k. Suppose that line i of f, $1 \leq i \leq n$, is

$$x_i : a_i \longrightarrow b_i.$$

Consider the following assignment $\alpha = (c_1, c_2, \ldots, c_n)$ to the variables x_1, x_2, ..., x_n of f.

1. For $1 \leq i < j$, set $c_i = \overline{a_i}$.
2. For $i = j$, set $c_j = a_j$.
3. For $i = k$, set $c_k = \overline{a_j}$.
4. For $j < i < k$, and for $k < i \leq n$, if $b_i = b_j$, then set $c_i = \overline{a_i}$, else set $c_i = a_i$.

Note that $f(\alpha) = b_j$, since the variables in all lines above line j have the complement of their canalyzing value, and variable x_j has its canalyzing value.

Now, let α' be the assignment that is obtained from α by interchanging the values of variables x_j and x_k, so that after the interchange, variable x_j has value $\overline{a_j}$ and variable x_k has value a_j. We will show that $f(\alpha') = \overline{b_j}$, so variables x_j and x_k are not symmetric.

In line k of f, canalyzing value a_k is either the same or the complement of canalyzing value a_j, and canalyzed value b_k is either the same or the complement of canalyzed value b_j, Thus, there are four possible cases for the form of line k. We now consider each of the four cases.

Case 1: Line k has the form $x_k : a_j \longrightarrow b_j$.

Since line k has the same canalyzing value and canalyzed value as line j, line k must occur in a lower layer than that of line j. Thus, there is at least one line between line j and line k for which the canalyzed

value is $\overline{b_j}$. Let line q be the first such line. Note that for all i such that $1 \leq i < q$, the value of variable x_i in assignment α' does not match the canalyzing value of line i, but the value of variable x_q does match the canalyzing value of line q. Since canalyzed value b_q equals $\overline{b_j}$, we have that $f(\alpha') = \overline{b_j}$.

Case 2: Line k has the form $x_k : a_j \longrightarrow \overline{b_j}$.

Let line q be the first line such that $j < q \leq k$ and the canalyzed value of line q is $\overline{b_j}$. Note that q might possibly equal k. For all i such that $1 \leq i < q$, the value of variable x_i in assignment α' does not match the canalyzing value of line i, but the value of variable x_q does match the canalyzing value of line q. Since canalyzed value b_q equals $\overline{b_j}$, we have that $f(\alpha') = \overline{b_j}$.

Case 3: Line k has the form $x_k : \overline{a_j} \longrightarrow b_j$.

Suppose there is a line below line j for which the canalyzed value is $\overline{b_j}$. Let q be the first such line. Note that for all i such that $1 \leq i < q$, the value of variable x_i in assignment α' does not match the canalyzing value of line i, but the value of variable x_q does match the canalyzing value of line q. Since canalyzed value b_q equals $\overline{b_j}$, we have that $f(\alpha') = \overline{b_j}$.

Now suppose there is no line below line j for which the canalyzed value is $\overline{b_j}$. Then lines j and k both occur in the last layer of f. Thus, for every line i, $1 \leq i \leq n$, the value of variable x_i in assignment α' does not match the canalyzing value of line i. Consequently, $f(\alpha') = \overline{b_j}$, the complement of the canalyzed value in the last layer of f.

Case 4: Line k has the form $x_k : \overline{a_j} \longrightarrow \overline{b_j}$.

Suppose there is a line, other than line k, below line j for which the canalyzed value is $\overline{b_j}$. Let q be the first such line. Note that for all i such that $1 \leq i < q$, the value of variable x_i in assignment α' does not match the canalyzing value of line i, but the value of variable x_q does match the canalyzing value of line q. Since canalyzed value b_q equals $\overline{b_j}$, we have that $f(\alpha') = \overline{b_j}$.

Now suppose that line k is the only line below line j for which the canalyzed value is $\overline{b_j}$. Since line j has canalyzed value b_j, line k is the only line in its layer. Since f is default-normalized, the last layer of f contains at least two lines. Thus, there is a last layer following the layer containing line k. The canalyzed value of all the lines in this last layer is b_j. Thus, for every line i, $1 \leq i \leq n$, the value of variable x_i in assignment α' does not match the canalyzing value of line i. Consequently, $f(\alpha') = \overline{b_j}$, the complement of the canalyzed value in the last layer of f. ■

Theorem 3.2 *Suppose the default-normalized representation of a given NCF contains r_1 layers with only one distinct canalyzing value, and r_2 layers with two distinct canalyzing values. Then the function is properly $(r_1 + 2r_2)$-symmetric.*

Proof: From Theorem 3.1, any pair of variables occurring in the same layer, with the same canalyzing value, are symmetric. Thus the function is at most $(r_1 + 2r_2)$-symmetric. Moreover, any pair of variables from different layers, or with different canalyzing values are not symmetric, so there are at least $(r_1 + 2r_2)$ symmetry groups. ■

Corollary 3.3 *For every $n \geq 2$, there is an n-variable NCF that is not $n-1$ symmetric.* ■

Corollary 3.4 *(i) An NCF with a default-normalized nested canalyzing representation consisting of q layers is $2q$-symmetric, and is not $(q-1)$-symmetric. (ii) An r-symmetric NCF has a default-normalized nested canalyzing representation with at most r layers.* ■

3.3 Testing Whether an r-Symmetric Function is an NCF

Theorem 3.2 shows that given an NCF f, the problem of finding the smallest integer r such that f is r-symmetric can be solved efficiently. We now consider the converse problem, that is, testing whether a given r-symmetric Boolean function is an NCF. For this problem, we present an algorithm whose running time is linear in the size of the representation of the r-symmetric function f. When the answer is "yes", the algorithm also constructs a default-normalized representation (defined in Section 2) of f.

Let $f(x_1, x_2, \ldots, x_n)$ be an r-symmetric Boolean function of n variables. We assume that the symmetry groups of f, denoted by g_1, g_2, ..., g_r, are given. Let $m_i = |g_i|$, $1 \leq i \leq r$. Thus, in group g_i, the number of variables which can take on the value 1 varies from 0 to m_i, $1 \leq i \leq r$. We also assume that f is given as a table T of the following form: each row of T specifies an r-tuple $(c_1, c_2, \ldots, c_r)$, where c_i is the number of variables of group g_i which have the value 1, along with the $\{0,1\}$ value of f for that r-tuple. Thus, μ, the number of rows in T, is given by $\mu = \prod_{i=1}^{r}(m_i + 1)$. Since there are μ rows and each row is an r-tuple, the size of the table T is $O(r\mu)$. As will be seen, our algorithm for determining whether f is an NCF runs in $O(r\mu)$ time. The algorithm relies on the following observation.

Observation 3.5 *Suppose $f(x_1, x_2, \ldots, x_n)$ is an r-symmetric Boolean function. Consider any variable x_i and suppose the number of variables in the group containing x_i is ν_i, $1 \leq i \leq n$.*

1. *Function f is canalyzing in variable x_i with canalyzing value 1 iff all the table entries where the number of 1's in the group containing x_i is nonzero have the same value of f. (This value of f is the canalyzed value when $x_i = 1$.)*
2. *Function f is canalyzing in variable x_i with canalyzing value 0 iff all the table entries where the number of 1's in the group containing x_i is less than ν_i have the same value of f. (This value of f is the canalyzed value when $x_i = 0$.)* ■

We now explain how Observation 3.5 can be used to develop an iterative algorithm for determining whether f is an NCF; if so, the algorithm also constructs a default-normalized representation for f. We use the following notation. At beginning of iteration j, r_j denotes the number of remaining groups, X_j denotes a set of variables with exactly one variable from each remaining group, T_j denotes the table which provides values for the function and μ_j denotes the number of rows of T_j. Initially (i.e., $j = 1$), $r_1 = r$ (the number of symmetry groups of f), X_1 is constructed by choosing one variable (arbitrarily) from each of the r groups, $T_1 = T$ (the given table T for f) and $\mu_1 = \mu$ (the number of rows of T); further, the NCF representation for f is empty. The algorithm carries out iteration j if $r_j \geq 1$; in that iteration, the algorithm performs Steps I, II and III as described below.

I. Use Observation 3.5 to determine whether there is a canalyzing variable $x_i \in X_j$.

II. If yes, perform the following steps.

1. Let α and β be the respective canalyzing and canalyzed values for x_i found in Step I.
2. For each variable x_p in the group containing x_i, append the rule "$x_p : \alpha \longrightarrow \beta$" to the NCF representation of f.
3. Set X_{j+1} = $X_j - \{x_i\}$.
4. Obtain T_{j+1} by retaining only those rows of T_j where all the variables in the group containing x_i are set to $\overline{\alpha}$.

5. Set $r_{j+1} = r_j - 1$.

6. If $r_{j+1} \geq 1$, start the next iteration; otherwise, **stop**.

III. (Here, Step I didn't find a canalyzing variable.) Output "f is not an NCF" and **stop**.

The correctness of the algorithm is a direct consequence of Observation 3.5. If the algorithm is successful, it produces a simplified NCF representation of f; this can then be efficiently converted into the default-normalized representation as discussed in Section 2.2.

To estimate the running time, we note that in iteration j, we can determine whether there is a canalyzing variable in X_j in $O(r_j\mu_j) = O(r\mu_j)$ time using Observation 3.5. (This is done by considering each variable in X_j.) Once a canalyzing variable is identified, the other steps in that iteration can be completed in $O(\mu_j)$ time. Thus, the time for iteration j is $O(r\mu_j)$. We claim that in each iteration, the number of rows in the table is reduced by a factor of at least 2; that is, $\mu_{j+1} \leq \mu_j/2$. To see why, suppose the canalyzing variable x_i found in iteration j is in group g_p with m_p variables. Thus, in T_j, there are $m_p + 1 \geq 2$ possibilities for the number of 1's in group g_p. As indicated in Substep 4 of Step II above, T_{j+1} contains only those rows of T_j where all the variables in g_p have the same value. In other words, there is only one possibility for the number of 1's in group g_p. Thus, μ_{j+1}, the number of rows in T_{j+1}, is at most $\mu_j/(m_p + 1) \leq \mu_j/2$. It follows by simple induction that $\mu_j \leq \mu/2^{j-1}$. Thus, the running time over all the r iterations is given by $O(r[\sum_{j=1}^{r}(\mu/2^{j-1})]) = O(r\mu)$, since the geometric sum is bounded by 2μ. Recall that the size of input table T (which specifies the r-symmetric function f) is $O(r\mu)$. Thus, the running time of the algorithm is linear in the size of the input. The following theorem summarizes this result.

Theorem 3.6 *Suppose an r-symmetric function f is given as a table where each row has an r-tuple of the form $(c_1, c_2, \ldots, c_r)$, with c_i being the number of variables in group g_i that have value 1, along with the {0,1} value of the function for that row. The problem of testing whether f is an NCF can be solved in time that is linear in the size of the input.* ■

When an r-symmetric function f with n variables is specified by giving the table representation described in the statement of Theorem 3.6, the size of the given table is $O(n^r)$. When r is *fixed*, the size of this representation and the running time of the above algorithm are both polynomials functions of n. When r is not fixed, while the size of the input and the running time of the algorithm are not necessarily polynomial functions of n, the running time remains linear in the input size.

3.4 Strong Asymmetry and NCFs

The notion of strong asymmetry of Boolean functions was defined in Section 2. For general Boolean functions, the notions of "n-symmetry" and "strong asymmetry" are not equivalent. We illustrate this by presenting an example of a Boolean function which is properly n-symmetric but not strongly asymmetric.

Example 6: Consider the Boolean function f of four variables, namely a, b, c and d, whose truth table is shown in Table 1. Function f is symmetric under the permutation $cdab$. For convenience, we give the truth table for f in a form that makes its symmetry under the permutation $cdab$ clear. So, the function f is not strongly asymmetric. We now demonstrate that the function is properly 4-symmetric by observing that it is not symmetric with respect to any pair of variables.

$a\ b$	$c\ d$	Value of f	$a\ b$	$c\ d$	Value of f
0 0	0 1	0	0 1	1 1	1
0 1	0 0	0	1 1	0 1	1
0 0	1 0	1	1 0	1 1	1
1 0	0 0	1	1 1	1 0	1
0 0	1 1	0	0 0	0 0	0
1 1	0 0	0	0 1	0 1	0
0 1	1 0	1	1 0	1 0	0
1 0	0 1	1	1 1	1 1	0

Tab. 1: An example of a Boolean function with 4 variables which is properly 4-symmetric but not strongly asymmetric.

(i) $f(0100) \neq f(1000)$, so a and b are not symmetric.

(ii) $f(0011) \neq f(1001)$, so a and c are not symmetric.

(iii) $f(0001) \neq f(1000)$, so a and d are not symmetric.

(iv) $f(0100) \neq f(0010)$, so b and c are not symmetric.

(v) $f(0011) \neq f(0110)$, so b and d are not symmetric.

(vi) $f(0001) \neq f(0010)$, so c and d are not symmetric.

We now show that any n variable NCF that is properly n-symmetric is also strongly asymmetric. As a consequence, we observe that there are NCFs with n variables that are properly n-symmetric.

Theorem 3.7 *An NCF with n variables is strongly asymmetric iff it is properly n-symmetric.*

Proof: If an NCF is $(n-1)$-symmetric, then consider a permutation that interchanges two variables from a symmetry group with at least two members. The value of the function is invariant under any such permutation, so the function is not strongly asymmetric.

For the converse, consider an n variable NCF that is properly n-symmetric. Let f be the default-normalized NCF representation for the function. For purposes of notational simplicity, without loss of generality, we assume that the canalyzing variable in line i of f is x_i, $1 \leq i \leq n$. Let π be any permutation of $\{1, 2, \ldots, n\}$ *except* the identity permutation. We will construct an assignment $(c_1, c_2, \ldots, c_n)$ to the variables of f such that $f(c_1, c_2, \ldots, c_n) \neq f(c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)})$.

Let k be the smallest index such that $k \neq \pi(k)$. Since π is a permutation, $k < n$. Overall, for $1 \leq i < k$, $\pi(i) = i,\ \pi(k) > k,$ and for $k < i \leq n,\ \pi(i) \geq k$.

Suppose that line i of f, $1 \leq i \leq n$, is

$$x_i : a_i \longrightarrow b_i.$$

Assignment $(c_1, c_2, \ldots, c_n)$ is constructed as follows.

1. For $1 \leq i < k$, set $c_i = \overline{a_i}$.
2. For $i = k$, set $c_k = a_k$.

3. For $i > k$, let $i' = \pi^{-1}(i)$, so that $\pi(i') = i$. Thus, after the permutation of values, variable $x_{i'}$ will have value c_i. If $b_{i'} = b_k$, then set $c_i = \overline{a_{i'}}$, else set $c_i = a_{i'}$.

Since c_k matches the canalyzing value of line k of f, and c_i does not match the canalyzing value of any other earlier line i, $1 \leq i < k$, we have that $f(c_1, c_2, \ldots, c_n) = b_k$.

Now consider $f(c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)})$. We first note that for $1 \leq i < k$, $c_{\pi(i)}$ does not match the canalyzing value of line i.

Let $k' = \pi^{-1}(k)$, so that $\pi(k') = k$. Canalyzing value $a_{k'}$ is either the same or the complement of a_k, and canalyzed value $b_{k'}$ is either the same or the complement of b_k, Thus, there are four possible cases for the form of line k' of f. We will show that in each of the four cases, $f(c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)}) = \overline{b_k}$.

Case 1: Line k' has the form $x_{k'} : a_k \longrightarrow b_k$.

Since line k' has the same canalyzing value and canalyzed value as line k, and f is properly n-symmetric, line k' must occur in a lower layer than that of line k. Thus, there is at least one line between line k and line k' for which the canalyzed value is $\overline{b_k}$. Let line q be the first such line. Note that for all i such that $1 \leq i < q$, $c_{\pi(i)}$ does not match the canalyzing value of line i, but $c_{\pi(q)}$ does match the canalyzing value of line q. Since canalyzed value b_q equals $\overline{b_k}$, we have that $f(c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)}) = \overline{b_k}$.

Case 2: Line k' has the form $x_{k'} : a_k \longrightarrow \overline{b_k}$.

Let line q be the first line such that $k < q \leq k'$ and the canalyzed value of line q is $\overline{b_k}$. Note that for all i such that $1 \leq i < q$, $c_{\pi(i)}$ does not match the canalyzing value of line i, so $f(c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)}) = \overline{b_k}$.

Case 3: Line k' has the form $x_{k'} : \overline{a_k} \longrightarrow b_k$.

Suppose there is a line below line k for which the canalyzed value is $\overline{b_k}$. Let q be the first such line. Note that for all i such that $1 \leq i < q$, $c_{\pi(i)}$ does not match the canalyzing value of line i, but $c_{\pi(q)}$ does match the canalyzing value of line q. Thus, $f(c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)}) = \overline{b_k}$.

Now suppose there is no line below line k for which the canalyzed value is $\overline{b_k}$. Then line k occurs in the last layer of f. Since f is both default-normalized and properly n-symmetric, this last layer contains exactly two lines. Thus $k = n - 1$ and $k' = n$. Note that for all i, $c_{\pi(i)}$ does not match the canalyzing value of line i. Thus, $f(c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)}) = \overline{b_k}$.

Case 4: Line k' has the form $x_{k'} : \overline{a_k} \longrightarrow \overline{b_k}$.

Suppose there is a line, other than line k', below line k for which the canalyzed value is $\overline{b_k}$. Let q be the first such line. Note that for all i such that $1 \leq i < q$, $c_{\pi(i)}$ does not match the canalyzing value of line i, but $c_{\pi(q)}$ does match the canalyzing value of line q. Thus, $f(c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)}) = \overline{b_k}$.

Now suppose that line k' is the only line below line k for which the canalyzed value is $\overline{b_k}$. Since line k has canalyzed value b_k, line k' is the only line in its layer. Since f is default-normalized, there is a last layer following the layer containing line k'. Thus, for all i, $c_{\pi(i)}$ does not match the canalyzing value of line i; therefore, $f(c_{\pi(1)}, c_{\pi(2)}, \ldots, c_{\pi(n)}) = \overline{b_k}$. ■

We now provide examples of properly n-symmetric n-variable NCFs satisfying the conditions of Theorem 3.2. For each $n > 1$, let f_n be the function of n variables, namely $x_1, \ldots, x_n$, defined by the following formula:

$$x_1 \vee (\overline{x}_2 \wedge (x_3 \vee (\overline{x}_4 \wedge (\cdots))))$$

For instance,

$$f_6 = x_1 \vee (\overline{x}_2 \wedge (x_3 \vee (\overline{x}_4 \wedge (x_5 \vee \overline{x}_6)))) \quad \text{and}$$
$$f_7 = x_1 \vee (\overline{x}_2 \wedge (x_3 \vee (\overline{x}_4 \wedge (x_5 \vee (\overline{x}_6 \wedge x_7))))).$$

Function $f_n(x_1, x_2, \ldots, x_n)$ is the NCF corresponding to the following NCF representation:

$$\begin{array}{l} x_1 : 1 \longrightarrow 1 \\ x_2 : 1 \longrightarrow 0 \\ x_3 : 1 \longrightarrow 1 \\ \qquad \vdots \end{array}$$

If n is odd, the last line is

$$x_n : 1 \longrightarrow 1$$

and if n is even, the last line is

$$x_n : 1 \longrightarrow 0.$$

Note that if $x_i = 0$ for all i, $1 \leq i \leq n$, then the value of f_n is 0 if n is odd and 1 if n is even. Also, note that the above representation is not default-normalized. To obtain a default-normalized representation, the last line would be changed to have canalyzing value 0, and the same canalyzed value as the preceding line.

3.5 Number of Strongly Asymmetric NCFs

Theorem 3.8 *For any $n \geq 2$, the number of Boolean functions with n variables that are both strongly asymmetric and NCFs is $n!\,2^{n-1}$.*

Proof: Consider a default-normalized representation of a strongly asymmetric NCF f. The representation has $n-1$ layers. The first $n-2$ layers each contain one line, and the last layer contains two lines.

Consider the canalyzed values in f. The canalyzed value in the first layer can be either 0 or 1. The canalyzed value in every other layer is the complement of the canalyzed value in the preceding layer. Thus, there are 2 possibilities for the sequence of canalyzed values in f.

The canalyzing variable in each of the first $n-2$ lines (one per layer) can be any variable that has not yet occurred in a preceding line. The last layer contains the remaining two variables. Thus, there are $n!/[n-(n-2)]! = n!/2$ possibilities for the pattern of canalyzing variable occurrences in f.

For each of the first $n-2$ variables, the canalyzing value for the line containing that variable can be either 0 or 1. For the last two lines, the canalyzing values for the variables in those lines must be complements of each other; otherwise, by Theorem 3.1, those two variables will be symmetric. Thus, the number of possible canalyzing values over all the n lines is $2^{n-2} \times 2 = 2^{n-1}$.

Hence, the number of Boolean functions with n variables that are both strongly asymmetric and NCFs is equal to $2 \times (n!/2) \times 2^{n-1} = n!\,2^{n-1}$. ■

3.6 Symmetric Canalyzing and Nested Canalyzing Functions

Proposition 3.9 *(i) The only symmetric canalyzing functions are OR, AND, NOR, NAND, the constant function 0 and the constant function 1. (ii) The only symmetric NCFs are OR, AND, NOR and NAND.*

Proof:

Part (i): Suppose that symmetric function f is also a canalyzing function. Then there is a variable x, and values a and b such that whenever $x = a$, function f has value b. Since f is symmetric, f has value b whenever any of its variables has value a. Thus, if at least one of its variables has value a, then f has value b. If f has value $\overline{b}$ when none of its variables has value a, the four possible combinations of values for a and b correspond to the four functions OR, AND, NOR, and NAND. If f has value b when none of its variables has value a, then f is the constant function b.

Part (ii): Suppose that symmetric function f is also an NCF. Since constant functions are not NCFs, f has value $\overline{b}$ when none of its variables has value a. The four possible combinations of values for a and b correspond to the four functions OR, AND, NOR and NAND, each of which is an NCF. ■

4 Summary

We presented a characterization of when two variables of an NCF are symmetric and used that characterization to show that the symmetry level of an NCF can be easily computed. We also showed that an n-variable NCF is strongly asymmetric iff its symmetry level is n. We presented several corollaries of this result, including a closed form expression for the number of NCFs which are also strongly asymmetric. Thus, our results bring out several interesting relationships between NCFs and symmetric Boolean functions.

Acknowledgments

We sincerely thank the reviewer for carefully reading the manuscript and providing very helpful comments. This work has been partially supported by DTRA CNIMS (Contract HDTRA1-11-D-0016-0001), NSF Grant IIS-1908530, NSF Grant OAC-1916805, NSF DIBBS Grant ACI-1443054, NSF BIG DATA Grant IIS-1633028 and NSF EAGER Grant CMMI-1745207. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government.

References

A. Abdollahi and M. Pedram. Symmetry detection and Boolean matching utilizing a signature-based canonical form of Boolean functions. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 27(6): 1128–1137, 2008.

C. L. Barrett, H. B. Hunt III, M. V. Marathe, S. S. Ravi, D. J. Rosenkrantz, R. E. Stearns, and M. Thakur. Predecessor existence problems for finite discrete dynamical systems. *Theor. Comput. Sci.*, 386(1-2): 3–37, 2007.

N. N. Biswas. On identification of totally symmetric Boolean functions. *IEEE Trans. Computers*, 19(7): 645–648, 1970.

R. C. Born and A. K. Scidmore. Transformation of switching functions to completely symmetric switching functions. *IEEE Trans. Computers*, 17(6):596–599, 1968.

Y. Crama and P. Hammer. *Boolean Functions: Theory, Algorithms, and Applications*. Cambridge University Press, New York, NY, 2011.

P. T. Darga, K. A. Sakallah, and I. L. Markov. Faster symmetry discovery using sparsity of symmetries. In *Proceedings of the 45th Design Automation Conference, DAC 2008, Anaheim, CA, USA, June 8-13, 2008*, pages 149–154, 2008.

P. Ecsedi-Tóth, F. Móricz, and A. Varga. A note on symmetric Boolean functions. *Acta Cybernetica*, 3 (4):321–326, 1977.

M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-completeness*. W. H. Freeman & Co., San Francisco, CA, 1979.

L. Haviarova and E. Toman. Properties of symmetric Boolean functions. *J. Applied Mathematics, Statistics and Informatics (JAMSI)*, 12(1):5–24, 2016.

Y. Hu, V. Shih, R. Majumdar, and L. He. Exploiting symmetries to speed up SAT-based Boolean matching for logic synthesis of FPGAs. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 27(10):1751–1760, 2008.

A. S. Jarrah, B. Raposa, and R. Laubenbacher. Nested canalyzing, unate cascade and polynomial functions. *Physica D*, 233(2):167–174, Sept. 2007.

C. Kadelka, J. Kuipers, and R. Laubenbacher. The influence of canalization on the robustness of Boolean networks. *Physica D*, 353-354:39–47, 2017a.

C. Kadelka, Y. Li, J. Kuipers, J. O. Adeyeye, and R. Laubenbacher. Multistate nested canalizing functions and their networks. *Theoretical Computer Science*, 675:1–14, 2017b.

S. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theoretical Biology*, 22(3):437–467, 1969.

S. Kauffman, C. Peterson, B. Samuelsson, and C. Troein. Random Boolean network models and the yeast transcriptional network. *Proc. National Academy of Sciences (PNAS)*, 100(25):14796–14799, Dec. 2003.

S. Kauffman, C. Peterson, B. Samuelsson, and C. Troein. Genetic networks with canalyzing Boolean rules are always stable. *Proc. National Academy of Sciences (PNAS)*, 101(49):17102–17107, Dec. 2004.

M. J. Kearns and U. V. Vazirani. *An Introduction to Computational Learning Theory*. MIT Press, Cambridge, MA, 1994.

J. G. Klotz, R. Heckel, and S. Schober. Bounds on the average sensitivity of nested canalyzing functions. *PLOS One*, 8:8 pages, May 2013.

V. N. Kravets and K. A. Sakallah. Generalized symmetries in Boolean functions. In *Proc. IEEE/ACM Intl. Conf. Computer Aided Design 2000 (ICCAD-2000)*, pages 526–532, Nov. 2000.

L. Layne. *Biologically Relevant Classes of Boolean Functions*. PhD thesis, Department of Mathematics, Clemson University, 2011.

L. Layne, E. Dimitrova, and M. Macauley. Nested canalyzing depth and network stability. *Bulletin of Mathematical Biology*, 74(2):422–433, 2012.

Y. Li and J. O. Adeyeye. Sensitivity and block sensitivity of nested canalyzing functions. arXiv:1209.1597v1 [cs.DM], Sept. 2012.

Y. Li, J. O. Adeyeye, and R. C. Laubenbacher. Nested canalyzing functions and their average sensitivities. arXiv:1111.7217v1 [cs.DM], Nov. 2011.

Y. Li, J. O. Adeyeye, D. Murrugarra, B. Aguilar, and R. C. Laubenbacher. Boolean nested canalizing functions: A comprehensive analysis. *Theoretical Computer Science*, 481:24–36, 2013.

P. M. Maurer. A universal symmetry detection algorithm. *SpringerPlus*, 4, 2015. 30 pages.

H. Mortveit and C. Reidys. *An Introduction to Sequential Dynamical Systems*. Springer Science & Business Media, New York, NY, 2007.

E. Paul, G. Pogudin, W. Qin, and R. Laubenbacher. The dynamics of canalizing Boolean networks. arXiv:1902.00056v1 [q-bio.MN], Jan. 2019.

D. J. Rosenkrantz, M. V. Marathe, H. B. Hunt III, S. S. Ravi, and R. E. Stearns. Analysis problems for graphical dynamical systems: A unified approach through graph predicates. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2015, Istanbul, Turkey, May 4-8, 2015*, pages 1501–1509, 2015.

C. E. Shannon. A symbolic analysis of relay and switching circuits. *AIEE Transactions*, 57:713–723, 1938.

R. E. Stearns, D. J. Rosenkrantz, S. S. Ravi, and M. V. Marathe. A characterization of nested canalyzing functions with maximum average sensitivity. *Discrete Applied Mathematics*, 251:5–14, 2018.

C. Tsai and M. Marek-Sadowska. Generalized Reed-Muller forms as a tool to detect symmetries. *IEEE Trans. Computers*, 45(1):33–40, 1996.

C. H. Waddington. Canalyzation of development and the inheritance of acquired characters. *Nature*, 150 (14):563–565, 1942.