FACTA UNIVERSITATIS (NIŠ)
SER.: ELEC. ENERG. vol. 24, no. 3, December 2011, 341-356

# BDD Based Construction of Resilient Functions

**Stanislav Stanković and Jaakko Astola**

**Abstract:** The construction of modern cryptographic systems relies on the so-called resilient Boolean functions, a special class of Boolean functions that possesses a balance between a high level of nonlinearity and correlation immunity.

In this paper, we discuss the problem of the compact representation and efficient construction of resilient functions. Binary Decision Diagrams (BDDs) were extensively used as a method of compact representation of various classes of Boolean functions. Furthermore, BDDs offer an opportunity for the efficient implementation of different construction methods for resilient functions. In this paper, we make use of BDDs with attributed edges to provide an implementation of two construction methods proposed by Maitra and Sakar. In addition, we demonstrate that the size of BDDs of resilient functions obtained in this way grows linearly with the number of variables.

**Keywords:** Decision diagrams; BDD; resilient; Boolean; cryptography; bent functions.

## 1 Introduction

Modern cryptographic systems rely on a class of Boolean functions known as *resilient* functions [6], [7], [14]. These functions posses several important properties, which make them suitable for use in stream cyphers. The two most important properties are correlation immunity and high nonlinearity. Resilient functions were first studied by Siegenthaler in [21]. This class of functions is closely related to *bent* functions, another class of Boolean functions with applications in cryptography [9], [18]. Bent functions exhibit the highest nonlinearity and can be used as a starting point to generate resilient functions [6].

Manuscript received August, 2011. An earlier version of this paper was presented at the Reed-Muller 2011 Workshop, May 25-26, 2011, Gustavelund Conference Centre, Tuusula, Finland

The authors are with the Department of Signal Processing, Tampere University of Technology, Korkeakoulunkatu 1, 33720 Tampere, Finland (e-mails: [stanislav.stankovic, jaakko.astola]@tut.fi).

Digital Object Identifier: 10.2298/FUEE1103341S

Binary decision diagrams (BDDs) are a canonic representation of Boolean functions [3]. They are considered to be compact and efficient in terms of space and processing time requirements. Properties of BDDs representing *bent* functions were explored in [20]. The recursive nature of decision diagrams corresponds well to the recursive nature of some well-known methods for the construction of resilient functions, opening ways for more efficient implementations. These methods can be divided into two broad groups: primary methods in which new resilient functions are created directly, usually using Boolean functions from other classes as a starting point, and secondary methods where resilient functions of a desired size are generated from already known resilient functions of the smaller size.

In [14], Maitra and Sekar discuss the various methods of creation of resilient functions. Functions constructed using these methods constitute a subset of a larger set of all resilient functions. In this paper, we present a BDD based implementation of one primary and one secondary method for the construction of resilient functions as proposed in [14]. We make use of BDDs with attributed edges. Furthermore, we demonstrate that the size of BDDs with attributed edges of resilient functions obtained in this way grows linearly with the number of variables.

We begin our discussion by providing definitions of resilient functions and related terms in Section 2. A brief introduction to BDDs is given in Section 3. We discuss the relationship between BDDs and resilient functions in Section 4. A BDD based implementation of the primary construction method is presented in Section 5. In this method, a new resilient function is created by modifying a bent function of suitable properties. In Section 6, we present a BDD based implementation secondary method for the construction of resilient functions. Resilient functions of an arbitrary size are generated iteratively starting from already known resilient functions of a smaller size. Finally, concluding remarks are given in Section 7.

## 2 Resilient Functions

The stream cipher method is one of the most common approaches for encryption of digital information. A given message, represented as a stream of bits, is encrypted by the application of a bitwise EXOR operation with another sequence of bits called the key stream. This key stream can be represented by a Boolean function. In order to qualify for this task, a Boolean function needs to satisfy several important criteria. An improperly chosen Boolean function will render the system open to various kinds of attacks. Boolean functions that satisfy these criteria are known as resilient functions. In what follows, we give the definitions of relevant functional properties and related mathematical terms.

**Definition 1.** *The Hamming weight of a binary string S is defined as the sum of its*

*elements* $wt(S) = \sum_{i=1}^{\lambda} S(i)$.

**Definition 2.** *Let* $S_1$ *and* $S_2$ *be two binary strings of length* $\lambda$. *By* $\#(S_1 = S_2)$ *we denote the number of places where* $S_1$ *and* $S_2$ *have equal value, and by* $\#(S_1 \neq S_2)$ *the number of places where they differ. The Hamming distance between* $S_1$ *and* $S_2$ *is defined as* $D(S_1, S_2) = \#(S_1 \neq S_2)$.

**Definition 3.** *Let* $V_n$ *be the vector space of n-tuples of elements of* $GF(2)$. *A Boolean function* $f$ *is a mapping from* $V_n$ *to* $GF(2)$. *By* $\Omega_n$ *we denote the set of all possible mappings, i.e. the set of all Boolean functions defined on* $V_n$.

*The truth-vector* $F$ *of a Boolean function* $f \in \Omega_n$ *is a binary vector of length* $2^n$ *obtained by explicitly stating the output of the function* $f$ *for each member of* $V_n$.

**Definition 4.** *A Boolean function* $f$ *is balanced if its output is equally distributed, i.e., the number of* 0 *elements in its truth-vector* $F$ *is equal to the number of* 1 *elements.*

The second important property of resilient functions, high algebraic degree, is linked to polynomial function representation. Each Boolean function $f$ can be uniquely represented by a polynomial.

**Definition 5.** *Let* $f$ *be a Boolean function. The Algebraic Normal Form of* $f$ *is the polynomial representation of* $f$*:*

$$f(x_1, ..., x_n) = \bigoplus_{(a_1,...,a_n) \in V_n} f(a_1, ..., a_n) x_1^{a_1} ... x_n^{a_n}. \quad (1)$$

*In the exponents, the elements 0 and 1 of* $GF(2)$ *are interpreted as integers 0 and 1.*

*The* algebraic degree *of a function* $f$ *is defined as the maximum number of variables in terms* $x_1^{a_1} ... x_n^{a_n}$ *in its Algebraic Normal Form.*

The algebraic degree is an important measure of the linear complexity of Boolean functions.

**Definition 6.** *A Boolean function is linear if and only if it has the following form* $f(x_1, ..., x_n) = a_1 x_1 \oplus a_2 x_2 \oplus ... \oplus a_n x_n$ *where* $a_i \in GF(2)$.

*Furthermore, a Boolean function is known as the Affine function if it has the form* $f(x_1, ..., x_n) = a_0 \oplus a_1 x_1 \oplus a_2 x_2 \oplus ... \oplus a_n x_n$ *where* $a_i \in GF(2)$. *The set of all affine functions in* $V_n$ *is denoted as* $A_n$.

*The algebraic degree of affine and linear functions is* 1.

Another measure of the nonlinearity of a function is its minimal Hamming distance from the set of affine functions $A_n$ [17]. The class of Boolean functions with the highest possible Hamming distance from the set of affine functions $A_n$ are *bent* functions [18].

Finally, a function $f$ is said to be correlation-immune of the order $t$ if the output of the function is statistically independent of the combination of any $t$ of its inputs [6]. In [12], Xiao and Massey provide the definition of the $k$-order correlation immune Boolean function by using properties of its Walsh spectrum.

**Definition 7.** *A Boolean function $f(x_1,...,x_n)$ is a k-ordered correlation immune where, $k < n$, if and only if for any choice of $t \leq k$ variables $x_{i(j)}$, $1 \leq j \leq t$, the function*

$$g(x_1,...,x_n) = f(x_1,...,x_n) \oplus \bigoplus_{j=1}^{t} x_{i(j)}$$

*is balanced [8].*

These properties can be observed in many classes of Boolean functions. Their presence determines the applicability of Boolean functions in cryptographic systems. Each of these properties is directly related to a certain type of attack to which a system might be exposed. The absence of a certain property might leave the cryptographic system vulnerable. However, these properties impose contradictory constraints. Therefore, to be applicable in a cryptographic system, a Boolean function needs to balance a trade off between certain criteria.

High algebraic degree provides immunity against the Berlekamp-Massey shift register synthesis attack algorithm [15]. A large distance from the set of affine functions ensures protection from affine approximation attack [10]. However, bent functions, Boolean functions with the highest nonlinearity, are known not to be balanced [18], which leaves them susceptible to other forms of attacks. Furthermore, the maximal algebraic degree of an $n$-variable bent function is $n/2$ [18].

Correlation immunity of Boolean functions was first explored, in this context, by Siegenthaler in [21]. This property is linked with the so-called divide-and-conquer attacks [5]. However, correlation immunity is not a sufficient condition for a function to be applicable as a stream key generator. For example, $n$ variable linear functions can have a high order of correlation immunity of $n-1$. Furthermore, constant functions have an order of correlation immunity of $n$.

The contradictory requirements of nonlinearity, especially high algebraic degree, balancedness, and correlation immunity were first formulated by Siegenthaler [21] in the form of an inequality relating the number of logic variables $n$,

the order of correlation immunity $m$, and the algebraic degree $k$ of a function, as $m+k \leq n-1$. This fundamental inequality and above mentioned properties have great implications on the structure and construction of resilient functions.

**Definition 8.** *A Boolean function which is at the same time balanced and k-ordered correlation immune is a k-ordered* resilient *function [6].*

Resilient functions were first proposed by Siegenthaler in [21]. The classification of resilient functions according to the properties of their Walsh spectra was introduced in [12] by Zhen and Massey and further examined by Braeken et al., in [2] and [7].

Several methods were proposed for the construction of various families of functions which optimize the Siegenthaler inequality, for example see [4], [6], [11], [16] and [19]. These methods provide a way for constructing different subsets of a larger set of all resilient functions.

Two distinct strategies were applied.

The first approach involves the direct construction of new resilient functions, usually by the modification of a function from some other class, such as bent function, in order to change its properties to satisfy the Siegenthaler inequality. Methods that employ this strategy are known as primary constructions. We discuss a BDD implementation of a primary construction proposed by Maitra and Sekar in Section 5.

Another strategy involves the construction of resilient functions of a desired size from already known resilient functions of a smaller size. These methods are known as secondary constructions. In Section 6 we present in detail the BDD based implementation of a secondary construction proposed in [14].

These two methods can be combined in such a way that a function created by a primary construction serves as a building block for larger function generated by secondary constructions.

## 3 BDDs

Decision diagrams are a method of compact representation of discrete functions. They have found application in various fields, especially in logic design. The application of decision diagrams for this sort of problems was first established in [3] by Bryant, by showing that BDDs are canonic representations of Boolean functions.

BDDs are a class of decision diagrams for the representation of Boolean functions based on the Shannon decomposition.

**Definition 9.** *Let $f(x_1,...,x_n)$ be an n-variable Boolean function. The Shannon decomposition with respect to the variable $x_i$ is defined as $f = \bar{x_i} f_0 \oplus x_i f_1, i = 1,...,n$, where $f_0 = f(x_i = 0)$, and $f_1 = f(x_i = 1)$.*

**Definition 10.** *A BDD of the function $f(x_1,...,x_n)$ is a directed acyclic graph G consisting of a set of nodes B and a set of edges E. Nodes in decision diagrams can be either non-terminal $b_n \in B_n$ or terminal $b_t \in B_t$, where $B_n$ and $B_t$ are sets of all non-terminal and terminal nodes of the diagram respectively, and $B = B_n \cup B_t$. Each edge $e \in E$ is an ordered pair of $e = (b_p, b_c)$, where $b_p \in B_n$ is a parent node and $b_c \in B$ is a child node.*

*Each non-terminal node in $b_k \in B$ represents a Shannon decomposition of the function $f(x_1,...,x_n)$ with respect to $x_i$, $i = 0,...,n$. Every non-terminal node $b_k \in B_n$ has exactly two edge elements $e_0$, $e_1$ associated with it, where $e_0$ corresponds to $f_0 = f(x_i = 0)$ and $e_1$ to $f_1 = f(x_i = 1)$.*

*Terminal nodes correspond to constant values of the truth-vector of the given function $f$.*

*The function is determined from its BDD by traversing all the paths starting at the root node and ending in terminal nodes. The inverse Shannon decomposition is applied at each non-terminal node.*

**Definition 11.** *If the order of variables is identical for each path in the BDD, then such a diagram is an ordered BDD.*

**Definition 12.** *A BDD is considered* reduced *if it contains no isomorphic sub-diagrams.*

A reduced ordered BDD is a canonic representation of the Boolean function [3].

In this paper we focus on BDDs with attributed edges, an extension of the concept of reduced ordered BDDs. We introduce two new edge labels $\bar{T}$ and $\bar{E}$ which indicate changes in the reading rule of reduced ordered BDDs.

**Definition 13.** *The label $\bar{T}$ associated with an incoming edge of the sub-diagram indicates that the values of the terminal nodes of the sub-diagram should be complemented.*

In the recursive traversal of the BDD, this is accomplished by inverting the output from the previous step of the recursion. The existence of the label $\bar{T}$ adds one more `IF-THEN-ELSE` evaluation to the algorithm complexity per each step of the recursion.

**Example 1.** *Consider a sub-diagram for vector $F = [0001]^T$. The change of the reading rule indicated by the $\bar{T}$ label associated with an incoming edge to this sub-diagram results in the vector $F^c = [1110]^T$.*

**Definition 14.** *The label $\bar{E}$ associated with and incoming edge of the sub-diagram indicates that all the edge values in the sub-diagram should be permuted.*

In the recursive traversal of the BDD, this corresponds to the difference recursion flow, from the left branch first to right branch first, at each recursion step. The label $\bar{E}$ adds one more `IF-THEN-ELSE` evaluation to the algorithm complexity at each step.

**Example 2.** *Consider again the sub-diagram for vector $F = [0001]^T$. The change of the reading rule indicated by the $\bar{E}$ label associated with an incoming edge to this sub-diagram results in the vector $F^r = [1000]^T$.*

The inclusion of $\bar{T}$ and $\bar{E}$ edge labels requires additional two bits for each attributed edge in the BDD.

The number of non-terminal nodes is an important parameter of the complexity of BDDs. By $Size(f)$ we denote the number of non-terminal nodes in the BDD for the function $f$. The size of a BDD is determined by the properties of the underlying function. If no reductions are possible, a BDD is equivalent to a complete Binary Decision Tree and consist of $2^n - 1$ non-terminal nodes. Thus, in general cases, the size of a BDD grows exponentially with the number of function variables. This rapid increase in the size of BDD represents a significant problem for the applications of a BDDs in many cases. However, it can be shown that, for certain classes of Boolean functions, the corresponding BDDs grow linearly with the number of variables $n$. In Section 6 we demonstrate that resilient functions obtained by the constructions by Maitra and Sakar [13] are one such class.

For further details on decision diagrams in general, please refer, for example, to [1] or [22].

## 4 Resilient Functions and BDDs

Special properties of resilient functions have a direct influence on the structure of their BDDs. We exploit this in order to make efficient BDD based implementations of construction methods for resilient functions.

The following operations over the truth-vector are of special interest for the construction of resilient functions.

By $F^u$ and $F^l$ we denote the upper and lower part of the truth-vector of the function $f$, where $f^u = f(x_1 = 0)$, $f^l = f(x_1 = 1)$. If $f \in \Omega_n$ then $f^u, f^l \in \Omega_{n-1}$. Notice that this is equivalent to the Shannon decomposition of the function $f$ with respect to the variable $x_1$. In the BDD of $f$, $f^u$ and $f^l$ correspond to sub-diagrams at level 1, associated with 0 and 1 edge of the root node respectively.

Consider two Boolean functions $f,g \in \Omega_n$. A function $h$ whose truth-vector $H$ is obtained by the concatenation of truth-vectors of $F$ and $G$, and denoted as $H = FG$, is a Boolean function in $\Omega_{n+1}$.

Let $f,g \in \Omega_{n-1}$ be two resilient functions with correlation immunity at least $m$. The two following constructions of resilient functions $Q,R \in \Omega_n$ were proposed in literature [13], [14], [21]:

1. $Q(f,g) = F^u F^l G^u G^l$,
2. $R(f,g) = G^u G^u G^l G^l$.

We point out that, in this framework, BDDs of the functions $f$ and $g$ are sub-diagrams of the BDDs of functions $Q$ and $R$, as shown in Figure 1.

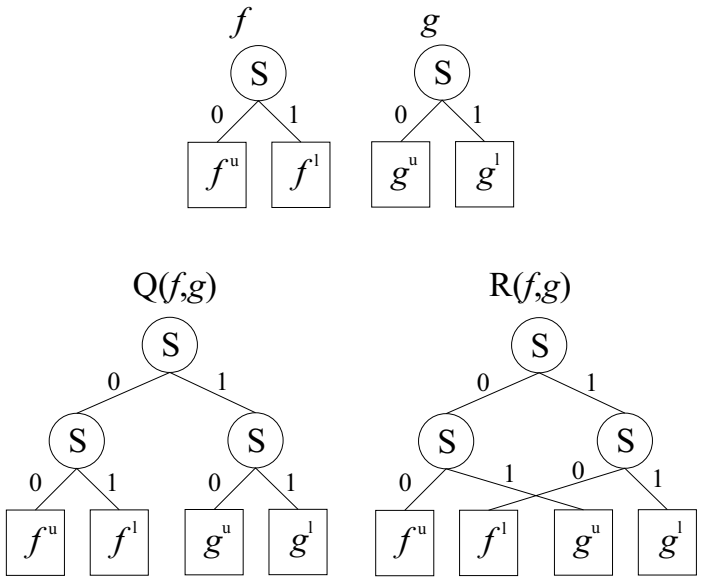


Fig. 1. BDD for $Q(f,g) = F^u F^l G^u G^l$ and $R(f,g) = F^u G^u F^l G^l$.

The total number of non-terminal nodes in BDDs, $Size(Q)$ and $Size(R)$ is in the worst case $Size(f) + Size(g) + 1$. This constitutes the upper bound on $Size(Q)$ and $Size(R)$. However, if the BDDs of functions $f$ and $g$ share any isomorphic sub-diagrams, $Size(Q)$ and $Size(R)$ are smaller, since only one copy of each sub-tree is retained in a BDD. In the case where $f = g$, i.e. the functions have identical BDDs, $Size(Q) = Size(R) = Size(f) + 1$, which is the lower bound on $Size(Q)$ and $Size(R)$. Depending on the choice of $f$ and $g$, $Size(f) + 1 \leq Size(Q)$, $Size(R) \leq Size(f) + Size(g) + 1$.

Furthermore, these constructions can be used in conjunction with two other important observations regarding resilient functions.

**Definition 15.** *Consider a n-variable function $f$ with a truth vector $F$. Function $f^r$ with a truth vector $F^r$ is obtained from $f$ by reversing the truth-vector by the following operation $F^r(X_1,...,X_n) = F(1 \oplus X_1,...,1 \oplus X_n)$.*

**Statement 1.** *Functions $f$ and $f^r$ have BDDs identical up to the labels on the edges.*

**Example 3.** *Consider the two-variable Boolean function $f = x_1x_2$ and its truth-vector $F = [0001]^T$. The truth-vector for $f^r$ is $F_r = [1000]^T$. As evident from Figure 2, BDDs for $f$ and $f^r$ have an identical distribution of non-terminal nodes and edges. They are identical up to the edge labels.*

The operation of reversing the function truth-vector results in permuting the left and right edge of every non-terminal node in the BDD. Functions $f$ and $f^r$ can be represented by a single BDD. Only the reading rule needs to be altered in the case of $f^r$. The information about the change in the reading rule needs to be explicitly stored along with the BDD. This observation holds for any function $f \in \Omega_n$.

**Definition 16.** *Let $f$ be an n-variable Boolean function with truth vector $F$. Function $f_c$ with a truth vector $F^c$ is obtained from $f$ as a bitwise complement of its truth-vector $F^c(X_1,...,X_n) = 1 \oplus F(X_1,...,X_n)$.*

**Statement 2.** *Functions $f$ and $f^c$ have BDDs identical up to the value of terminal nodes.*

**Example 4.** *Consider again the function $f = x_1x_2$ and its truth-vector $F = [0001]^T$. By bitwise complementing the truth-vector of $f$, we obtain $F_c = [1110]^T$. Figure 2 shows the BDD for $f^c$. These diagrams are identical up to values of terminal nodes.*

The operation of bitwise complementing of the truth-vector is equivalent to the complementing values of constants associated with terminal nodes in a BDD. Functions $f$ and $f^c$ can be represented by a single BDD. Once again, the reading rule needs to be altered in the case of $f^c$. This information needs to be specified explicitly.

Furthermore, if $f$ is a resilient function, then both $f^r$ and $f^c$ are also resilient functions [14].

These observations represent the foundation of the method for the construction of resilient functions for an arbitrary *n*.

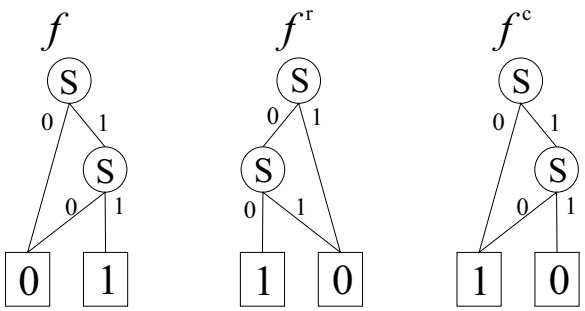


Fig. 2. BDDs for $F = [0001]^T$, $F^r = [1000]^T$ and $F^c = [1110]^T$

**Definition 17.** *Given a resilient function $f \in \Omega_{n-1}$, we can construct $f' \in \Omega_n$ as:*

$$f' = \Psi(f, f^\tau), \tag{2}$$

*where $\Psi \in \{Q, R\}$ and $\tau \in \{c, r, rc\}$.*

Due to the properties of $\{Q, R\}$ and $\{c, r, rc\}$, if $f$ is resilient, then $f'$ is also resilient. For a detailed discussion of the properties of resilient functions obtained in this way, please refer to [14].

We have demonstrated that the function $f$ and $f^\tau$, $\tau \in \{c, r, rc\}$, can be represented by a single BDD. Thus, $Size(f') = Size(f) + 1$. In addition, we need to explicitly represent the information about the reading rules associated with the $f^\tau$ sub-diagram. This can be done by associating additional edge labels with incoming edges of the $f^\tau$ sub-diagram. Since there are three possible reading rule choices associated with $\{c, r, rc\}$, we need the total of two bits per each incoming edge to represent this.

## 5 BDD Implementation of the Primary Construction of Resilient Functions

In Section 2, we have indicated that a resilient function optimizes the Siegenthaler inequality [21], the trade-off between the degree of correlation immunity, the algebraic degree of a function, and its distance from a set of affine functions. By definition, bent functions have the highest possible Hamming distance from the set of affine functions. However, they have a limited maximal algebraic degree. The idea is to modify an existing bent function, sacrifice something in the way of its distance from the set of affine functions in order to increase its algebraic degree.

In [14], Maitra and Sekar propose the following construction method for the construction of an $n$-variable $m$-order correlation immune function $f$ with an algebraic degree $k$, where $k = n - m - 1$.

**Statement 3.** *Let $f \in \Omega_n$, where $n = m + k + 1$ and $m \geq 1$. The truth vector $F$ of $f$ is obtained by a series of constructions represented as $(H, S_1, ..., S_{m+1})$, where for $i > 0$, $S_i = (\Phi_i, \tau_i)$, $\Phi_i \in \{Q, R\}$. For even $i$, $\tau_i \in \{c, r\}$, and for odd $i$, $\tau_i \in \{c, rc\}$.*

*For $i = 0$, $H$ is a truth vector of $h \in \Omega_k$, is a $k$-variable Boolean function. For even $k$, $h$ is obtained from a bent function $g \in \Omega_k$ by adding the term $x_1x_2...x_k$. For odd $k$, $h$ is obtained by concatenating function $g \in \Omega_{k-1}$ with itself and the term $x_1x_2...x_k$, as $g \oplus g \oplus x_1x_2...x_k$.*

In [14], Maitra and Sekar provide proof of the optimality of resilient functions constructed in this way. The resilient function thus constructed can be used in conjunction with the secondary construction method presented in Section 6, i.e. new larger resilient functions are constructed by a subsequent iterative application of constructions $\{Q, R\} \times \{c, r, rc\}$.

Furthermore, as demonstrated in [20] and [23], bent functions can be efficiently represented using decision diagrams.

**Example 5.** *Let $k = 4$, $m = 3$. Consider a four variable quadratic disjoint bent function $g = x_1x_2 \oplus x_3x_4$. Figure 3 is a BDD of this function. We obtain $h$ by appending the term $x_1x_2x_3x_4$ to $g$, i.e. $h = g \oplus x_1x_2x_3x_4 = x_1x_2 \oplus x_3x_4 \otimes x_1x_2x_3x_4$. It is evident from Figure 3 that in this case $Size(h) < Size(g)$. Function $h$ can be used further to construct resilient functions. As demonstrated in the previous section, the size of the BDD in each next iteration will increase by 1.*

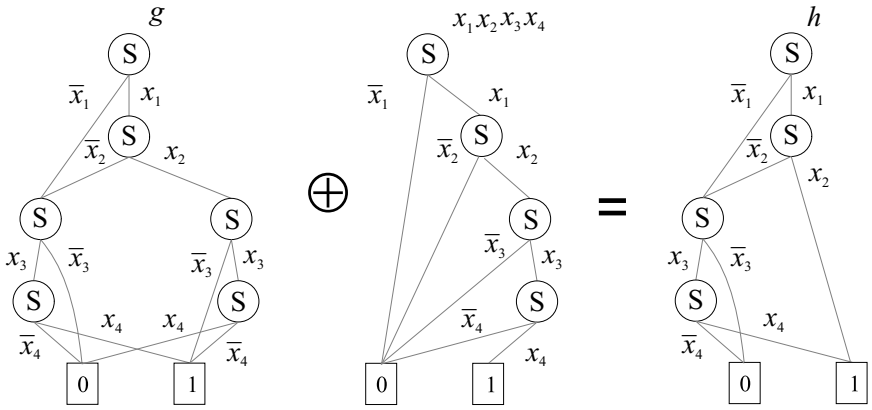


Fig. 3. BDDs of the functions $g = x_1x_2 \oplus x_3x_4$ and $h = g \oplus x_1x_2x_3x_4$.

## 6 BDD Implementation of the Secondary Construction of Resilient Functions

Exploiting the properties presented in Section 4, we can generate resilient functions of an arbitrary size.

By the recursive application of $\{Q,R\}\times\{c,r,rc\}$, we obtain the sequence $\{a, f_1, ..., f_m\}$, where $a$ is the starting resilient function and $f_j \in \{Q,R\}\times\{c,r,rc\}$. The truth-vector $F_j$ of $f_j$ consists of $2^j$ sub-vectors from the set of 8 possible vectors defined from $A$ of $a$.

These vectors are $A^u, A^l, A^{uc}, A^{lc}, A^{ur}, A^{lr}, A^{urc}, A^{lrc}$.

However, as demonstrated earlier, BDDs for $a^u$ and $a^l$ are sub-diagrams in BDD of $a$. BDDs of $a^{uc}, a^{ur}, a^{urc}$ are equivalent up to the reading rule to the BDD of $a^u$. Likewise, BDDs of $a^{lc}, a^{lr}, a^{lrc}$ are equivalent up to the reading rule to the BDD of $a^l$. Thus, $a^u, a^l, a^{uc}, a^{lc}, a^{ur}, a^{lr}, a^{urc}, a^{lrc}$ are represented by the BDD for $a$.

This observation has important consequences for the recursive application of the construction $\{Q,R\}\times\{c,r,rc\}$.

**Example 6.** *Consider the function $f_1 = R(a,a^c)$, and $f_2 = R(f_1, f_1^r)$. The vector $F_1 = A^u A^l A^{uc} A^{lc}$, and consequently $F_2 = F_1^u F_1^l F_1^{ur} F_1^{lr}$. Figure 4 shows BDDs for $f_1$ and $f_2$.*

It is evident from Figure 4 that a BDD consists of two sub-diagrams corresponding to $a$ and $a^c$. As shown earlier, these diagrams are identical up to the values of the terminal nodes. Thus, only one copy can be retained. The label $\overline{T}$ associated with the right edge of the diagram for $f_1$ indicates the change in the reading rule.

Likewise, the BDD for $f_2$ consists of two sub-diagrams corresponding to $f_1$ and $f_1^r$. Again, these two diagrams are identical up to the value of edge labels. Label $E$ associated with the right edge of the root node indicates the necessary change in the reading rule.

As shown in Example 6, each application of the $\{Q,R\}\times\{c,r,rc\}$ construction results in addition of one new node in the existing BDD. Sub-diagrams $a^u$ and $a^l$ are the basic building blocks used in every step of the recursion.

**Theorem 1.** *The complexity of the BDD with attributed edges of the function $f$ constructed from a resilient function $a$ using the $\{Q,R\}\times\{c,r,rc\}$ construction method, grows linearly with the number of variables.*

*Proof.* Consider an $n$-variable resilient function $a$. Its BDD consists of two sub-diagrams for $a^u$ and $a^l$, where $Size(a) \leq Size(a^u) + Size(a^l)$, from the definition

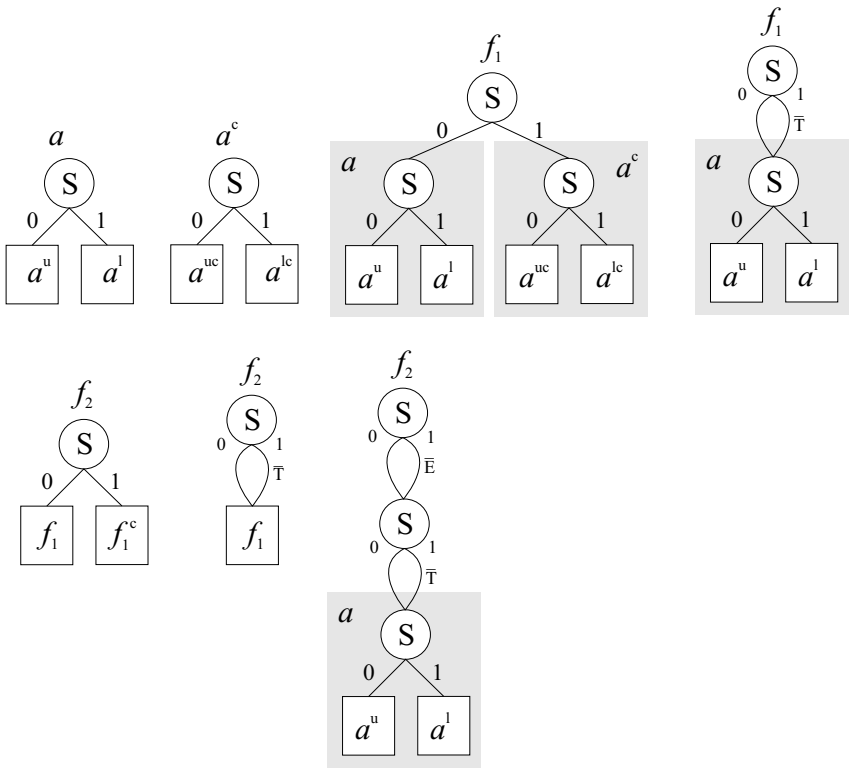


Fig. 4. BDDs for functions $f_1$ and $f_2$ in Example 6

of reduced ordered BDDs. From the Statement 1 and Statement 2, it follows that BDDs for $a^u, a^{uc}, a^{ur}, a^{urc}$ are identical up to the values of terminal nodes. The same applies to BDDs for $a^l, a^{lc}, a^{lr}, a^{lrc}$.

Consider an $(n+1)$-variable function $f_1$ constructed from $a$ by using the $\{Q,R\} \times \{c,r,rc\}$ construction method. From the properties of the Shannon decomposition, it follows that BDD of $f$ has one new node at the topmost level. BDD of $f$ consists of two sub-diagrams for $f^u$ and $f^l$. The sub-diagram for $f^u$ in turn consists of two sub-diagrams for $f^{uu}$ and $f^{ul}$, and $f^l$ consists of sub-diagrams for $f^{lu}$ and $f^{ll}$, where $f^{uu}, f^{ul}, f^{lu}, f^{ll} \in \{a^u, a^{uc}, a^{ur}, a^{urc}, a^l, a^{lc}, a^{lr}, a^{lrc}\}$. Due to the BDD reduction rules, and since sub-diagrams for $a^u, a^{uc}, a^{ur}, a^{urc}$ are isomorphic, and sub-diagrams $a^l, a^{lc}, a^{lr}, a^{lrc}$ are also isomorphic when using BDDs with attributed edges, only single copies of the sub-diagrams for $a^u$ and for $a^l$ are retained. Therefore, $Size(f_1) = Size(a) + 1$.

For any $n+k$ variable function $f_k$ constructed from $f_{k-1}$, $Size(f_k) = Size(f_{k-1}) + 1$, recursively $Size(f_k) = Size(a) + k$. □

This property of BDDs can be exploited for efficient software or hardware implementation of the proposed construction method. From the application point of view, a library of basic building blocks could be designed, each block corresponding to some simple resilient function represented by its BDD. Since the structure of the BDD remains the same, one single block could represent the original function $f$, $f_r$, $f_c$ and $f_{rc}$. The desired output could be selected as needed by an external signal. These building blocks could be then interconnected to generate different functions of the desired size. Furthermore, it is possible to combine this method with the primary method presented in Section 5. The starting function $f$ can be obtained by modifying a suitable bent function using the primary construction method.

The construction procedure for an $m$-variable resilient function starting from an $n$-variable resilient function could be formulated as follows:

1. From the library of functions, select an $n$-variable function $f$, where $n < m$.
2. Generate a BDD for $f$, with the possibility of choosing the reading rule. BDDs for $f^u$, $f^l$ are sub-diagrams of the BDD for $f$. Diagrams for $f^{uc}, f^{ur}, f^{ucr}$ and $f^{lc}, f^{lr}, f^{lcr}$ are covered by diagrams for $f^u$, $f^l$ respectively, up to the choice of the reading rule.
3. Choose one of the constructions from $\{Q,R\} \times \{c,r,rc\}$.
4. Add a new node to the BDD and create appropriate edges to generate the diagram for $f'$ with $n+1$ variables.
5. If $n+1 < m$, repeat the procedure with $f'$ as a starting function, using the diagram created in the previous step.

## 7 Conclusion

In this paper, we offered a reinterpretation of a method for two constructions for resilient functions of an arbitrary size, based on BDDs with attributed . One part of the proposed construction method is related to bent functions, a class of Boolean functions which can also be generated and represented BDDs in an efficient manner. The size of BDDs with attributed edges of the functions obtained in this way grows linearly with the size of the function, an important observation from the point of view of the efficient representation of large functions.

Furthermore, functions generated using the two constructions proposed my Maitra and Sekar are a subset of a larger set of resilient functions. BDDs with attributed edges of these functions have a regular structure. One open question arising from this conclusion is could this regularity in structure be exploited in an attack on a system based on these functions. If so, this would imply that this particular subset of resilient functions is unsuitable for cryptographic applications.

## 8 Acknowledgments

The authors would like to thank the reviewers for their valuable contribution to this paper. Their constructive comments and remarks were very useful in preparing the present version of the paper.

This work was supported by the Academy of Finland, Finnish Center of Excellence Program, Grant No. 5107491.

## References

[1] J. Astola, R. S. Stanković, *Fundamentals of Switching Theory and Logic Design*, Springer, 2006.

[2] A. Braeken, Y. Borissov, S. Nikova, and B. Preneel, "Classification of Cubic (n-4)-Resilient Boolean Functions", IEEE Trans. on Information Theory, Vol. 52, No. 4, pp. 1670-1676, April 2006.

[3] R. E. Bryant, "Graph-based Algorithms for Boolean Functions Manipulation", IEEE Trans. Comput., Vol. C-35, No. 8, 1986, 667 - 691.

[4] P. Camion, C. Claret, P. Charpin, and N. Sendrier, "On Correlation Immune Functions", Advances in Cryptography - CRYPTO' 91, pp. 86-100, Springer-Verlag, 1992.

[5] A. Canteaut, and M. Trabbia, "Improved Fast Correlation Attacks Using Parity-check Equations of Weight 4 and 5", In Advances in Cryptography - EUROCRYPT 2000, LNCS 1807, pp. 573-588, Springer-Verlag.

[6] C. Carlet, "More Correlation Immune and Resilient Functions over Galois Fields and Galois Rings", Advances in Cryptography - EUROCRYPT '97, pp 422-433, Springer-Verlag, May, 1997.

[7] C. Carlet, and P. Charpin, "Cubic Boolean Functions with Highest Resiliency", IEEE Trans. on Information Theory, Vol. 51, No. 2, pp. 562-571, February 2005.

[8] T. W. Cusick, "On Constructing Balanced Correlation Immune Functions, in Sequences and Their Applications", Proc. SETA '98, Springer Discrete Mathematics and Theoretical Computer Science, 1999, pp. 184-190.

[9] J. F. Dillon, *Elementary Hadamard Difference Sets*, Ph. D. Thesis, University of Maryland, 1974.

[10] C. Ding, G. Xiao, and W. Shan, "The Stability Theory of Stream Ciphers", Lecture Notes in Computer Science. Springer-Verlag, 1991.

[11] S. Gao, Y. Zhao, Z. Zhuo, "Walsh Spectrum of Cryptographically Concatenating Functions and Its Applications in Constructing Resilient Functions", J. of Computational Information Systems 7:4 (2011) pp. 1074-1081.

[12] X. Guo-Zhen and J. Massey, "A Spectral Characterization of Correlation-immune Combining Functions", IEEE Trans. Inf. Theory, vol. 34, no. 3, pp. 569-571, May 1988.

[13] S. Maitra, and P. Sakar, "Enumeration of Correlation Immune Boolean Functions", 4th Australian Conference on Information, Security and Privacy, Proc., Springer-Verlag, Lecture Notes in Comp. Sci., No 1587, 7-9 April, 1999.

[14] S. Maitra, and P. Sakar, "Highly Nonlinear Resilient Functions Optimizing Siegenthaler's Inequality", Advances in Cryptography - CRYPTO '99, pp. 198-215, Springer-Verlag, 1999.

[15] J. Massey, “Shift-Register Synthesis and BCH Decoding”, IEEE Transactions on Information Theory, IT-15, pp. 122-127, January. 1969.

[16] W. Millan, A. Clarke, and E. Dawson “Heuristic Design of Cryptographically Strong Balanced Boolean Functions”, Advances in Cryptography - EUROCRYPT’ 98, Springer-Verlag, 1998.

[17] K. Nyberg, “S-boxes and Found Functions with Controllable Linearity and Differential Uniformity” In Fast Software Encryption, FSE 2, volume 1008 of Lecture Notes in Computer Science. Springer-Verlag, 1995.

[18] O. S. Rothaus, “On ’Bent’ Functions”, Journal of Combinatorial Theory, Ser. A, Vol. 20, 300-305, 1976.

[19] J. Seberry, X. M. Zhang, and Y. Zheng, “On Constructions and Nonlinearity of Correlation Immune Boolean Functions”, Advances in Cryptography - EUROCRYPT ’93, pp. 181-199, Springer-Verlag, 1994.

[20] N. B. Schafer, Characteristics of Binary Decision Diagrams of Boolean Bent Functions, master thesis, Naval Postgraduate School, Monterey California, US, September 2009.

[21] T. Siegenthaler, “Correlation-immunity of Nonlinear Combining Functions for Cryptographic Applications”, IEEE Trans. on Information Theory, IT-30(5):776-780, September 1984.

[22] R. S. Stankovic, J. T. Astola, Spectral Interpretation of Decision Diagrams, Springer 2003.

[23] S. Stanković, M. Stanković, R. Stanković, J. Astola, “Representation of Bent Functions Using Walsh Decision Diagrams”, 9th International Workshop on Boolean Problems, pp. 179-186, September 16-17, 2010, Freiberg, Germany.