The Strict Avalanche Criterion: Spectral Properties of Boolean Functions and an Extended Definition

Réjane Forré
Inst. for Communication Technology *

Abstract

A necessary and sufficient condition on the Walsh-spectrum of a boolean function is given, which implies that this function fulfills the Strict Avalanche Criterion. This condition is shown to be fulfilled for a class of functions exhibiting simple spectral symmetries. Finally, an extended definition of the Strict Avalanche Criterion is proposed and the corresponding spectral characterization is derived.

1 Introduction

The "Strict Avalanche Criterion" (SAC) was introduced by A.F. Webster and S.E. Tavares. They write [1]: "If a function is to satisfy the strict avalanche criterion, then each of its output bits should change with a probability of one half whenever a single input bit x is complemented to $\overline{x}$." The cryptographic significance of the SAC is highlighted by considering the situation where a cryptographer needs some "complex" mapping f of n bits onto one bit. Although the expression "complex" has no precise mathematical definition here, an information-theoretical approach can help assigning it an intuitively pleasant meaning. Maximizing the entropy $H(f([x_1, x_2, \ldots, x_n]))$ yields zero-one balanced functions, but this alone certainly does not ensure the "complexity" of a function. Maximizing the conditional entropy $H([f(x_1, \ldots, \overline{x_i}, \ldots, x_n]) \mid f([x_1, \ldots, x_i, \ldots, x_n]))$ for all i, $1 \leq i \leq n$, leads to SAC-fulfilling boolean functions, according to the definition in [1]. It is proposed here to go even further, by keeping one or more input bits of f constant, and making the obtained "subfunctions" complex as well. It is worthwhile pointing out the fact that any function f' of $n-1$ bits will be a relatively bad approximation of f if f fulfills the SAC. Indeed, the output of the best possible f' will differ from

*Sternwartstr. 7, ETH-Zentrum, 8092 Zürich, Switzerland

S. Goldwasser (Ed.): Advances in Cryptology - CRYPTO '88, LNCS 403, pp. 450-468, 1990.

the output of f with a probability of $\frac{1}{4}$. This lack of accuracy of lower-dimensional approximations is a wishable property of cryptosystems: the existence of some (relatively accurate) lower-dimensional approximation of an enciphering transformation could reduce the amount of work for an exhaustive search according to the dimension of the domain of the approximation. Functions for which flipping one input bit always flips the output of course are still more difficult to approximate (the best lower-dimensional approximation is inaccurate in 50% of the cases), but their conditional entropy $H([f(x_1,\ldots,\overline{x_i},\ldots,x_n]) \mid f([x_1,\ldots,x_i,\ldots,x_n]))$ is zero.

In the first part of this paper, Boolean functions $f(\underline{x})$ with n bits input and one bit output are considered. The Walsh-transform has shown to be very useful for the analysis of (statistical) properties of boolean functions. It is shown that a boolean function $f(\underline{x})$ fulfills the SAC if and only if, for all $i \in \{1,2,\ldots,n\}$, its Walsh transform $\hat{F}(\underline{w})$, $\underline{w} = [w_1, w_2, \ldots, w_n]$, fulfills

$$\sum_{\underline{w} \in Z_2^n} (-1)^{w_i} \cdot \hat{F}^2(\underline{w}) = 0,$$

where Z_2^n denotes the n-dimensional vector space over the finite field GF(2). This set of conditions is shown to be fulfilled for a class of functions $\hat{F}(\underline{w})$ that exhibits certain "visible symmetries" arising from equalities of the form $\hat{F}(\underline{w}) = \hat{F}(\underline{w} \oplus \underline{c})$.

In the second part of the paper, the requirements on a boolean function are made stronger, introducing the concept of "SAC of higher order". The corresponding spectral conditions are then established.

2 Walsh-Spectrum of SAC-fulfilling Functions

2.1 Spectral Characterization of Functions Fulfilling the SAC

First, a few basic definitions, lemmas and theorems are needed.

Definition 1 *[2,3,5] If $f(\underline{x})$ is any real-valued function whose domain is the vector space Z_2^n, the* **Walsh transform** *of $f(\underline{x})$ is defined as:*

$$F(\underline{w}) = \sum_{\underline{x} \in Z_2^n} f(\underline{x}) \cdot (-1)^{\underline{x} \cdot \underline{w}}, \tag{1}$$

where $\underline{w} \in Z_2^n$ and $\underline{x} \cdot \underline{w}$ denotes the dot-product of $\underline{x}$ and $\underline{w}$, defined as

$$\underline{x} \cdot \underline{w} = x_1 w_1 \oplus x_2 w_2 \oplus \ldots \oplus x_n w_n. \tag{2}$$

The function $f(\underline{x})$ can be recovered from $F(\underline{w})$ by the **inverse Walsh transform***:*

$$f(\underline{x}) = 2^{-n} \sum_{\underline{w} \in Z_2^n} F(\underline{w}) \cdot (-1)^{\underline{x} \cdot \underline{w}}. \tag{3}$$

The Walsh transform and its inverse (both defined for real-valued functions) may be applied to boolean functions if their values are viewed as the real values 0 and 1.

Very often, it is easier to work with boolean functions that take values in the range $\{1, -1\}$. The function $\hat{f}(\underline{x})$ is defined as

$$\hat{f}(\underline{x}) = (-1)^{f(\underline{x})} \quad \text{or} \quad \hat{f}(\underline{x}) = 1 - 2f(\underline{x}). \tag{4}$$

The relationship between the Walsh transforms of $f(\underline{x})$ and $\hat{f}(\underline{x})$ is stated in the following lemma [2,3].

Lemma 1 *If* $\hat{f}(\underline{x}) = (-1)^{f(\underline{x})}$, *then*

$$\hat{F}(\underline{w}) = -2F(\underline{w}) + 2^n\delta(\underline{w}), \tag{5}$$

which is equivalent to

$$F(\underline{w}) = 2^{n-1}\delta(\underline{w}) - \frac{1}{2}\hat{F}(\underline{w}), \tag{6}$$

where

$$\delta(\underline{w}) = \begin{cases} 1, & \text{for } \underline{w} = \underline{0} \\ 0, & \text{else.} \end{cases} \tag{7}$$

Let $\underline{x}$ and $\underline{x}_i$ denote two n-bit vectors, such that $\underline{x}$ and $\underline{x}_i$ differ only in bit i, $1 \leq i \leq n$. Z_2^n denotes the n-dimensional vector space over $\{0,1\}$. The function $f(\underline{x}) = z$, $z \in \{0,1\}$ fulfills the SAC if and only if

$$\sum_{\underline{x} \in Z_2^n} f(\underline{x}) \oplus f(\underline{x}_i) = 2^{n-1}, \quad \text{for all } i \text{ with } 1 \leq i \leq n. \tag{8}$$

If we denote by $\underline{c}_i$ the n-dimensional unit-vector with a one at the i-th place and zeroes elsewhere, condition (8) may be alternatively written as

$$\sum_{\underline{x} \in Z_2^n} f(\underline{x}) \oplus f(\underline{x} \oplus \underline{c}_i) = 2^{n-1}, \quad \text{for all } i \text{ with } 1 \leq i \leq n. \tag{9}$$

We now wish to express the SAC for the case of an $\hat{f}$-function (with range $\{1, -1\}$). The following Lemma yields an alternative definition of the SAC.

Lemma 2 $f(\underline{x})$ *fulfills the SAC if and only if the function* $\hat{f}(\underline{x}) = (-1)^{f(\underline{x})}$ *fulfills*

$$\sum_{\underline{x} \in Z_2^n} \hat{f}(\underline{x}) \cdot \hat{f}(\underline{x} \oplus \underline{c}_i) = 0, \tag{10}$$

for all $\underline{c}_i$ *with Hamming-weight one.*

This lemma is easily derived, considering that if a function $f(\underline{x})$ fulfills the SAC, exactly half the $\underline{x} \in Z_2^n$ satisfy $f(\underline{x}) \neq f(\underline{x} \oplus \underline{c}_i)$, for all $i \in 1, 2, \ldots, n$. This means that the function $\hat{f}(\underline{x}) = (-1)^{f(\underline{x})}$ satisfies

$$\hat{f}(\underline{x}) \cdot \hat{f}(\underline{x} \oplus \underline{c}_i) = -1 \quad \text{for half the } \underline{x} \in Z_2^n, \text{ and} \tag{11}$$

$$\hat{f}(\underline{x}) \cdot \hat{f}(\underline{x} \oplus \underline{c}_i) = 1 \quad \text{for the other half.} \tag{12}$$

Summing up over all the $\underline{i} \in Z_2^n$ thus yields (10). The term on the left-hand side of equation (10) can also be represented by the convolution of $\hat{f}(\underline{x})$ with itself:

$$\sum_{\underline{x} \in Z_2^n} \hat{f}(\underline{x}) \cdot \hat{f}(\underline{x} \oplus c) = [\hat{f} * \hat{f}](c). \quad (13)$$

From the well-known convolution theorem, which states that

$$Ms) = \sum_{\underline{y} \in Z_2^n} f(y) \cdot 9(V \odot \pounds) \Leftrightarrow H(w) = F(w) \cdot G(w), \quad (14)$$

we see that the left-hand side of (10) is also the inverse Walsh-transform of $\hat{F}(yL) \cdot \hat{F}\{\underline{w}) = \hat{F}^2\{\underline{w})$, and with (3) we get:

$$[/*/](\pounds,) = 2^m \sum_{\underline{w} \in Z_2^n} \hat{F}^2(\underline{w}) \cdot (-iy \cdot \underline{x}) \quad (15)$$

$$= 2^{-n} \sum_{\underline{w} \in Z_2^n} \hat{F}^2(\underline{w}) \cdot (-i)^{w_i}, \quad (16)$$

where we made use of the fact that $\underline{c}_i$ is of the form $[0,0,\ldots,0,Cj = 1,0,\ldots,0]$. This, together with (5), proves the following theorem.

Theorem 1 *A function $f(\underline{x}) : Z_2^n \longrightarrow \{1,-1\}$ fulfills the SAC if and only if its Walsh-transform $\hat{F}(\underline{xv})$ satisfies*

$$\sum_{\underline{w} \in Z_2^n} (-1)^{w_i} \cdot \hat{F}^2(\underline{w}) = 0 \quad (17)$$

for all $i \in \{1,2,\ldots,n\}$. Equivalently, the Walsh-transform $F(\underline{w})$ of $f(\underline{x}) = |(1 - \hat{f}(\underline{x}))$ has to fulfill

$$\sum_{\underline{w} \in Z_2^n} (-1)^{w_i} \cdot F^2(\underline{w}) = 2^n F([0,\ldots,0]) - 2^{2n-2} \quad (18)$$

for alii G {l,2,...,n}.

Note that $F([0,\ldots,0])$ equals the number of ones in the truth table of $/(\underline{x})$.

Example 1:
Consider the function $f(\underline{x}) : Z\backslash \longrightarrow \{0,1\}$, the corresponding $\hat{f}(\underline{x}) = (\_l)/(s)$ and their respective Walsh-transforms $F(\underline{xv})$ and $\hat{F}(\underline{x})$ given by the following table:

x_1 / w_1	x_2 / w_2	x_3 / w_3	/te); $\hat{i}^c(\underline{x})$		$F(\underline{w})$	$\hat{F}(\underline{w})$
0	0	0	0:	1	4	0
0	0	1	11	-1	0	0
0	1	0	I	-1	-11	-11
0	1	1	0	1	-1	-4
1	0	0	0	1	0	0
1	0	1	0;	1	0	0
1	1	0	1 :	-1	2	4
1	1	1	1 ;	-1	-2	24

It is easily checked that flipping the bit x_t flips the output $f(x\_)$ in 50% of the cases. That is true for x_3 too, but not for x_2: flipping x_2 always changes $f\{\underline{x}\}$. Therefore,

$$H(f([x_1,x_2,x_3])\backslash f([x_u x_I, x_s])) \quad = \quad 0$$

and this function does not fulfill the SAC. Indeed, when we compute Emez?!-$^{1})^{10}$' • $\wedge^2(\wedge)$ ^{for} $\ddot{\imath}$ = 1,2 and 3, we get zero for i = 1 and i = 3 and -64 for i = 2, which does not satisfy the requirements of theorem 1.

Example 2:

Next, we examine another function of three bits, $g(x)$.

x_1 / w_1	x_2 / w_2	x_3 / w_3	$g(3L)$	$\hat{g}(\underline{x})$	$G(m)$	$\hat{}$ <?(Ш)
0	0	0	0	1	4	0
0	0	1	0	1	-2	-4
0	1	0	0	1	-2	-4
0	1	1	1	-1	0	0
1	0	0	0	1	-2	-4
1	0	1	1	1	0	0
1	1	0	1	-1	0	0
1	1	1	1	-1	2	4

The reader can check that flipping any of the three input bits involves an output change in 50% of the cases. Therefore, this function fulfills the SAC and the requirements of theorem 1 can be checked to hold for i = 1,2 and 3.

It should be pointed out that if a function fulfills the SAC, it does not imply that it is zero/one balanced, as can be seen from the following example.

Example 3:

x_1 / w_1	x_7 /w_2	x_3 / w_3	$h(\underline{x})$	$\hat{k}(\underline{x})$	$H(\underline{w})$	$\hat{H}\{\underline{w}\}$
0	0	0	0	1	2	-4
0	0	1	0	1	0	0
0	1	0	0	1	0	0
0	1	1	1	1	2	4
1	0	0	1	-1	0	0
1	0	1	0	1	-2	-4
1	1	0	0	1	-2	-2
1	1	1	0	1	0	0

$h(\underline{x})$ takes on six times the value zero and only twice the value one, which doesn't prevent it from fulfilling the SAC.

2.2 Construction of SAC-Fulfilling Functions

A geometrical interpretation of theorem 1 can be introduced if we look at the $n-$tuples $[w_1, w_2, \ldots, w_n]$ as the corners of an $n-$dimensional cube with edges of length one. Let's attach to each corner $\underline{w} = [w_1, w_2, \ldots, w_n]$ a weight $m_{\underline{w}}$ equal to $\hat{F}^2(\underline{w})$. The center of gravity of this $n-$dimensional body has the coordinates $[\overline{w_1}, \overline{w_2}, \ldots, \overline{w_n}]$ with

$$\overline{w_i} = \frac{\sum_{\underline{w} \in Z_2^n} m_{\underline{w}} \cdot w_i}{\sum_{\underline{w} \in Z_2^n} m_{\underline{w}}} = \frac{\sum_{\underline{w} \,:\, w_i = 1} \hat{F}^2(\underline{w})}{\sum_{\underline{w} \in Z_2^n} \hat{F}^2(\underline{w})}, \tag{19}$$

for $1 \leq i \leq n$. If a function $\hat{f}(\underline{x}) : Z_2^n \longrightarrow \{1, -1\}$ fulfills the SAC, we know by theorem 1 that

$$\sum_{\underline{w} \,:\, w_i = 0} \hat{F}^2(\underline{w}) - \sum_{\underline{w} \,:\, w_i = 1} \hat{F}^2(\underline{w}) = 0 \tag{20}$$

$$\Longrightarrow \sum_{\underline{w} \,:\, w_i = 0} \hat{F}^2(\underline{w}) = \sum_{\underline{w} \,:\, w_i = 1} \hat{F}^2(\underline{w}). \tag{21}$$

And in that case we have

$$\overline{w_i} = \frac{\sum_{\underline{w} \,:\, w_i = 1} \hat{F}^2(\underline{w})}{\sum_{\underline{w} \in Z_2^n} \hat{F}^2(\underline{w})} = \frac{\sum_{\underline{w} \,:\, w_i = 0} \hat{F}^2(\underline{w})}{\sum_{\underline{w} \in Z_2^n} \hat{F}^2(\underline{w})}, \tag{22}$$

which shows that the coordinate $\overline{w_i}$ of the center of gravity of the considered cubic body remains unchanged if all the weights on one "face" of the cube (face with $w_i = 0$) are moved to the opposite "face" (face with $w_i = 1$) and conversely. Therefore, we can state that a function $\hat{f}(\underline{x})$ fulfills the SAC if and only if the $n-$cube with weights equal to $\hat{F}^2(\underline{w})$ attached to its corners has a center of gravity which is *equidistant from any two opposite "faces" of the cube, and thus from all the corners of the cube.* The center of gravity of the body associated to the Walsh-spectrum of an SAC-fulfilling function therefore has the coordinates $[\frac{1}{2}, \frac{1}{2}, \ldots, \frac{1}{2}]$.

Example 4:
The 3-dimensional cube associated to the function $g(\underline{x})$ of example 2 is represented on the right-hand side of Fig. 1. The dark circles designate weights of magnitude $\hat{F}^2(\underline{w}) = 16$. The exchange of "faces" may be performed in three ways:

$$\begin{aligned} \hat{G}_1^2(\underline{w}) &= \hat{F}^2(\underline{w} \oplus [1,0,0]), \\ \hat{G}_2^2(\underline{w}) &= \hat{F}^2(\underline{w} \oplus [0,1,0]), \\ \hat{G}_3^2(w) &= \hat{F}^2(\underline{w} \oplus [0,0,1]), \end{aligned}$$

all of them yielding the same body, namely the one represented on the left-hand side of Fig. 1.

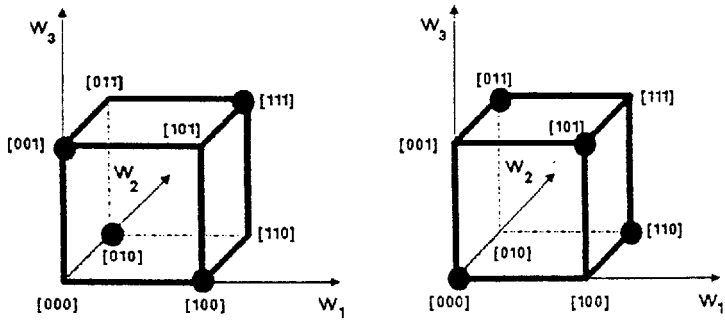

Figure 1: The 3-dimensional cubic body associated to the function $\hat{g}(\underline{x})$ of example 2 and its associated body obtained by exchanging "faces".

The idea that now naturally arises is to use this as a construction for new SAC-fulfilling functions from known ones. The pitfall is that $\hat{F}(\underline{w})$ might be taken as $\pm\sqrt{\hat{F}^2(\underline{w})}$ for each one of the 2^n $\underline{w}$'s. For the worst case where all 2^n $\underline{w}$'s are associated to nonzero values of $\hat{F}^2(\underline{w})$, this will yield 2^{2^n} possible choices for the mapping $\hat{F}(\underline{w})$, not all of them having valid boolean functions (i.e. 1/-1 valued) as inverse Walsh-transforms. In fact, a function $\hat{f}(\underline{x})$ is a boolean (1/-1 valued) function if and only if

$$\hat{f}^2(\underline{x}) = 1, \text{ for all } \underline{x} \in Z_2^n. \tag{23}$$

By the convolution theorem, we see that this is equivalent to

Theorem 2 *[2, p.167] $\hat{F}(\underline{w})$ is the Walsh-transform of a boolean function $\hat{f}(\underline{x})$: $Z_2^n \longrightarrow \{1,-1\}$ if and only if*

$$\sum_{\underline{w}\in Z_2^n} \hat{F}(\underline{w}) \cdot \hat{F}(\underline{w} \oplus \underline{s}) = 2^n \delta(\underline{s}) = \begin{cases} 2^n & \text{for } \underline{s} = [0,\ldots,0], \\ 0 & \text{otherwise.} \end{cases} \tag{24}$$

Let π be an operator on Z_2^n which, when applied to $\underline{x}$, permutes its indices [2, p.165]:

$$\underline{x} = [x_1, x_2, \ldots, x_n] \implies \pi\underline{x} = [x_{\pi_1}, x_{\pi_2}, \ldots, x_{\pi_n}]. \tag{25}$$

π^{-1} is the inverse operator such that

$$\pi^{-1}(\pi\underline{x}) = \underline{x}. \tag{26}$$

We write

$$\underline{y} = [y_1, y_2, \ldots, y_n] \implies \pi^{-1}\underline{y} = [y_{\pi'_1}, y_{\pi'_2}, \ldots, y_{\pi'_n}]. \tag{27}$$

Example 5:
If the permutation $\pi = [\pi_1, \pi_2, \pi_3] = [2, 3, 1]$ is applied to $\underline{x} = [x_1, x_2, x_3]$, one gets $\pi\underline{x} = [x_2, x_3, x_1]$. The inverse operator $\pi^{-1} = [\pi'_1, \pi'_2, \pi'_3]$ in this case equals $[3, 1, 2]$, since $\pi^{-1}(\pi\underline{x})$ must equal $\underline{x}$.

If a function $\hat{f}(\underline{x})$ fulfills the SAC, it is easy to see that this property is preserved under any permutation π of the input bits. Thus, $\hat{g}(\underline{x}) = \hat{f}(\pi\underline{x})$ fulfills the SAC too. Furthermore, $\hat{g}(\underline{x} \oplus \underline{c}) = \hat{h}(\underline{x})$ has $(-1)^{\underline{c}\cdot\underline{w}} \cdot \hat{G}(\underline{w}) = \hat{H}(\underline{w})$ as Walsh-transform (by the translate theorem), and this implies $\hat{H}^2(\underline{w}) = \hat{G}^2(\underline{w})$ for all $\underline{w} \in Z_2^n$. Consequently, $\hat{H}(\underline{w})$ satisfies equation (17) and the following theorem holds.

Theorem 3 *If $\hat{f}(\underline{x}) : Z_2^n \longrightarrow \{1, -1\}$ fulfills the SAC, then $\hat{g}(\underline{x}) = \hat{g}(\pi\underline{x} \oplus \underline{c})$ fulfills it too, for any permutation operator π and any constant $\underline{c} \in Z_2^n$.*

For symmetry reasons, the following lemma is easily seen to be true.

Lemma 3 *The function $\hat{g}(\underline{x}) = -\hat{f}(\underline{x})$ (resp. $g(\underline{x}) = \overline{f(\underline{x})}$) fulfills the SAC if and only if $\hat{f}(\underline{x})$ (resp. $f(\underline{x})$) fulfills the SAC.*

At this point, we already dispose of some tools to construct SAC-fulfilling boolean functions, and the question arises whether it is possible to construct all SAC-fulfilling functions with those tools. Computer experiments were carried out, in order to find such functions

(i) by exhaustive testing of all the 2^{2^n} existing boolean functions of n bits ($n = 3$ and $n = 4$),

(ii) by making use of Theorem 3 and Lemma 3 (but without trying out all possible assignations $\hat{G}(\underline{w}) = \pm\sqrt{\hat{F}^2(\underline{w})}$).

This established the fact that the above construction does not generate all the SAC-fulfilling functions, but only subclasses of them. We call the attention of the reader to the redundancy of the described synthesis rules: nothing ensures us that a newly obtained function will be different from the starting one or from a formerly constructed one.

Example 6:
Let $\hat{g}(\underline{x}) = \hat{f}(\underline{x} \oplus [1,0,1])$, where $\hat{f}(\underline{x})$ is defined through the following table.

x_1	x_2	x_3	$\hat{f}(\underline{x})$	$\hat{g}(\underline{x})$
0	0	0	1	1
0	0	1	1	1
0	1	0	1	1
0	1	1	-1	-1
1	0	0	.1	1
1	0	1	1	1
1	1	0	-1	-1
1	1	1	1	1

We notice that $\hat{g}(\underline{x}) = \hat{f}(\underline{x})$ for all $\underline{x} \in Z_2^3$. The reason is that $\hat{f}(\underline{x})$ is *partially symmetric* in x_1 and x_3 [4, p.123], that is $\hat{f}([x_1, x_2, x_3]) = \hat{f}([x_3, x_2, x_1])$ for all $[x_1, x_2, x_3] \in Z_2^3$.

2.3 Spectral Symmetries of SAC-Fulfilling Functions

We now introduce the definition of the 50%-dependence of boolean functions with respect to one of their input bits. The concept is not new: it was implicitely used in the definition of the SAC.

Definition 2 *A function $\hat{f} : Z_2^n \longrightarrow \{1,-1\}$ (resp. $f : Z_2^n \longrightarrow \{0,1\}$) is said to be* **50%-dependent of its $i-$th input bit** x_i *if and only if any two $n-$tuples $\underline{x}$ and $\underline{x_i}$ that differ only in bit i are mapped onto two different values with probability 1/2 and onto the same value with the same probability of 1/2. Or formally*

$$\sum_{\underline{x} \in Z_2^n} \hat{f}(\underline{x}) \cdot \hat{f}(\underline{x} \oplus \underline{c_i}) = 0, \tag{28}$$

for $\{1,-1\}-$valued functions, and

$$\sum_{\underline{x} \in Z_2^n} f(\underline{x}) \oplus f(\underline{x} \oplus \underline{c_i}) = 2^{n-1} \tag{29}$$

for $\{0,1\}-$valued functions.

We thus see that a boolean function fulfills the SAC if and only if it is 50%-dependent of each of its input bits.

The following theorem gives a *sufficient* condition for a function to be 50%-dependent of *one or more* of its input bits.

Theorem 4 *If for some nonzero $\underline{c} \in Z_2^n$ and for all $\underline{w} \in Z_2^n$*

$$\hat{F}^2(\underline{w}) = \hat{F}^2(\underline{w} \oplus \underline{c}) \tag{30}$$

holds, and if $\underline{c}$ has Hamming-weight m ($c_{i_1} = c_{i_2} = \ldots = c_{i_m} = 1$, $1 \leq m \leq n$), then $\hat{f}(\underline{x})$ is 50%-dependent of the input bits $x_{i_1}, x_{i_2}, \ldots, x_{i_m}$.

Proof:

According to the value of the subvector $\underline{w}' = [w_{i_1}, w_{i_2}, \ldots, w_{i_m}]$, the vector space Z_2^n can be divided into 2^m disjoint subsets $S_{\underline{w}'}$. To each of these subsets $S_{\underline{w}'}$ one can uniquely associate the subset $S_{\underline{v}'}$ where $\underline{v}' = [\overline{w_{i_1}}, \overline{w_{i_2}}, \ldots, \overline{w_{i_m}}]$, and because of (30) one can write

$$\sum_{\underline{w} \in S_{\underline{w}'}} \hat{F}^2(\underline{w}) = \sum_{\underline{w} \in S_{\underline{v}'}} \hat{F}^2(\underline{w}) \tag{31}$$

for each choice of $\underline{w}' \in Z_2^m$. Consequently, we have the following set of 2^{m-1} equations:

$$\begin{aligned}
\sum_{\underline{w} \in S_{[0,0,\ldots,0]}} \hat{F}^2(\underline{w}) &= \sum_{\underline{w} \in S_{[1,1,\ldots,1]}} \hat{F}^2(\underline{w}) \\
\sum_{\underline{w} \in S_{[0,\ldots,0,1]}} \hat{F}^2(\underline{w}) &= \sum_{\underline{w} \in S_{[1,\ldots,1,0]}} \hat{F}^2(\underline{w}) \\
&\vdots \qquad\qquad \vdots \\
\sum_{\underline{w} \in S_{[0,1,\ldots,1]}} \hat{F}^2(\underline{w}) &= \sum_{\underline{w} \in S_{[1,0,\ldots,0]}} \hat{F}^2(\underline{w}).
\end{aligned}$$

Summing up the left-hand side terms and the right-hand side terms respectively, we get

$$\sum_{\underline{w}\,:\,w_{i_1}=0} \hat{F}^2(\underline{w}) = \sum_{\underline{w}\,:\,w_{i_1}=1} \hat{F}^2(\underline{w}), \tag{32}$$

or equivalently

$$\sum_{\underline{w}\in Z_2^n} (-1)^{w_{i_1}} \cdot \hat{F}^2(\underline{w}) = 0, \tag{33}$$

which means that $\hat{f}(\underline{x})$ is 50%-dependent of x_{i_1}. For symmetry reasons, we get the same result for $x_{i_2}, \ldots, x_{i_m}$.

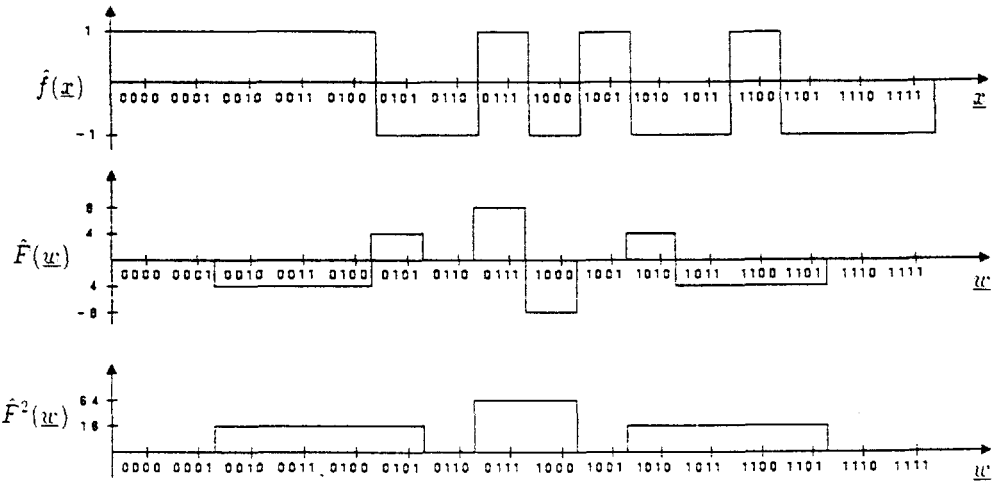

Figure 2: An SAC-fulfilling function $\hat{f}(\underline{x})$ of 4 bits whose squared Walsh-spectrum satisfies (34)

For the special case

$$\hat{F}^2(\underline{w}) = \hat{F}^2(\underline{w} \oplus [1, \ldots, 1]) = \hat{F}^2(\overline{\underline{w}}), \tag{34}$$

theorem 4 asserts that $\hat{f}(\underline{x})$ is 50%-dependent of all its input bits, or, in other words, that $\hat{f}(\underline{x})$ fulfills the SAC. This is interesting from a practical point of view, because the equality (34) is easily noticeable when looking at the squared Walsh-spectrum $\hat{F}^2(\underline{w})$.

Example 7:
The function $\hat{f}(\underline{x}) : Z_2^4 \longrightarrow \{1, -1\}$ takes on the following values (from the top to the bottom of the truth table): 1,1,1,1,1,-1,-1,1,-1,1,-1,-1,1,-1,1,-1. Fig. 2 shows this function, its Walsh-spectrum and its squared Walsh-spectrum. The discrete points where the functions are defined are connected by lines to make the diagrams more easily readable. We observe a symmetrical form of $\hat{F}^2(\underline{w})$ according to (34) and $\hat{f}(\underline{x})$ therefore fulfills the SAC.

But (34) is not a necessary condition for a function to fulfill the SAC. If, for example, $\hat{f}(\underline{x})$ is such that its squared Walsh-transform satisfies

$$\hat{F}^2(\underline{w}) = \hat{F}^2(\underline{w} \oplus [1,1,1,0,\ldots,0]) \tag{35}$$

$$\text{and } \hat{F}^2(\underline{w}) = \hat{F}^2(\underline{w} \oplus [1,1,0,1,\ldots,1]) \tag{36}$$

we know, by theorem 4 that $\hat{f}(\underline{x})$ fulfills the SAC (by (35), $\hat{f}(\underline{x})$ is 50%-dependent of the bits x_1, x_2 and x_3, by (36), $\hat{f}(\underline{x})$ is 50%-dependent of $x_1, x_2, x_4, \ldots, x_n$). The following example shows that a function $f(\underline{x})$ might be 50%-dependent of its input bit x_i even if there is no $\underline{c} \in Z_2^n$ such that $c_i = 1$ and (30) is satisfied for all $\underline{w} \in Z_2^n$. In other words, the condition of theorem 4 is sufficient but not necessary.

Example 8:
$\hat{F}^2(\underline{w})$ of Fig. 3 satisfies

$$\hat{F}^2(\underline{w}) = \hat{F}^2(\underline{w} \oplus [1,0,1,1]) \tag{37}$$

for all $\underline{w} \in Z_2^4$ but no other relation of the form (30). Equation (37) implies that $\hat{f}(\underline{x})$ is 50%-dependent of x_1, x_3 and x_4, but says nothing about x_2. Nonetheless, one can check that $\hat{f}(\underline{x})$ is 50%-dependent of x_2 as well.

3 Strict Avalanche Criterion of Higher Order

3.1 Definitions

As mentioned in the introduction, the SAC is cryptographically relevant because it maximizes the conditional entropy $H([f(x_1, \ldots, \overline{x_i}, \ldots, x_n]) \mid f([x_1, \ldots, x_i, \ldots, x_n]))$ and it assures that the best possible lower-dimensional space approximation of a mapping yields an erroneous result in 25% of the cases. We consider now a mapping of n bits onto one bit that fulfills the SAC. If one or more of its input bits are kept constant, the question arises whether it is possible to find some accurate approximation of this reduced mapping (reduced in the sense that it is defined only on a subspace of Z_2^n). If this is possible, the exhaustive search over the considered subspace can be reduced (compared with the exhaustive search over the full space

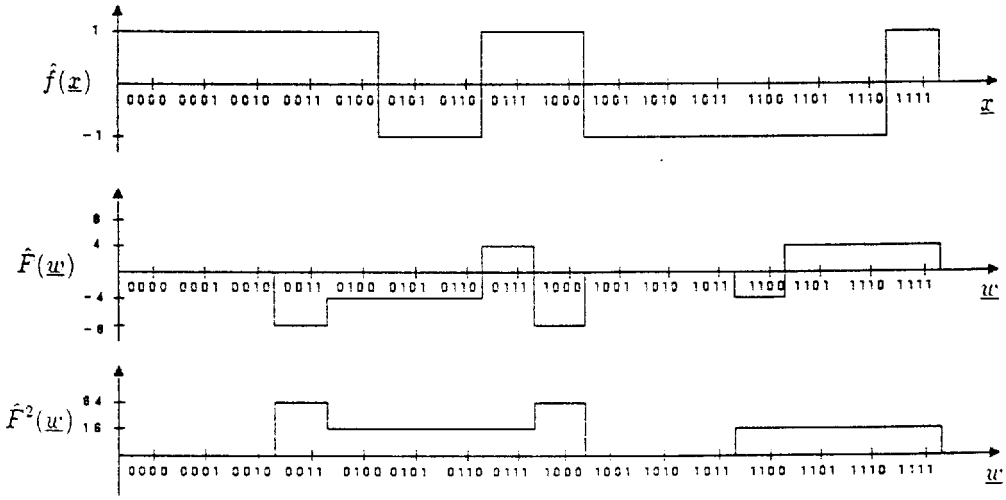

Figure 3: An SAC-fulfilling function that does not satisfy any equation of the form $\hat{F}^2(\underline{w}) = \hat{F}^2(\underline{w} \oplus [c_1, c_2 = 1, c_3, c_4])$ but nevertheless is 50%-dependent of the second input bit.

without approximation). In a chosen-plaintext attack, the opponent has the opportunity to perform such tests where one or more input bits are kept constant. For this reason, we now extend the definition of the SAC in order to cover situations like the one just described.

Let $f(\underline{x})$ be a function which maps Z_2^n onto $\{0,1\}$ and which fulfills the SAC. It is well-known that $f(\underline{x})$ can be written as

$$f(\underline{x}) = x_i \cdot f_{i,1}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) \oplus \overline{x_i} \cdot f_{i,0}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) \quad (38)$$

for every $i \in \{1, 2, \ldots, n\}$. The function $f_{i,1}$ (resp. $f_{i,0}$) is obtained from $f(x)$ by keeping the i-th bit of $\underline{x}$ constant and equal to 1 (resp. to 0). We now consider the 50%-dependence of the output of $f_{i,1}$ and $f_{i,0}$ with respect to each of their $n-1$ input bits.

Definition 3 *A function $f(\underline{x}) : Z_2^n \longrightarrow \{0,1\}$ is said to fulfill the* **Strict Avalanche Criterion of order 1** *if and only if*

- *$f(\underline{x})$ fulfills the SAC,*
- *and every function obtained from $f(\underline{x})$ by keeping the i-th input bit constant and equal to c fulfills the SAC as well (for every $i \in \{1, 2, \ldots, n\}$, and for $c = 0$ and $c = 1$).*

The definition can be extended to order m, where $1 \leq m \leq n-2$, if m input bits of $f(\underline{x})$ are kept constant.

Definition 4 *A function* $f(\underline{x}) : Z_2^n \longrightarrow \{0,1\}$ *is said to fulfill the* **Strict Avalanche Criterion of order** m *if and only if*

- $f(\underline{x})$ *fulfills the SAC of order* $m-1$,
- *and any function obtained from* $f(\underline{x})$ *by keeping* m *of its input bits constant fulfills the SAC as well (this must be true for any choice of the positions and of the values of the* m *constant bits).*

In what follows, the "classical" SAC will sometimes be called "SAC of order 0".

Example 9:
$f(\underline{x}) : Z_2^4 \longrightarrow \{0,1\}$ is defined through the following truth table.

x_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
x_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
x_3	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
x_4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
$f(\underline{x})$	0	0	0	1	0	1	1	1	1	0	0	0	0	0	0	1

Keeping the bit x_1 equal to 0, we get a function $f_{1,0} : Z_2^3 \longrightarrow \{0,1\}$ (left-hand half of truth table of $f(\underline{x})$) which can be checked to fulfill the SAC. To check whether $f(\underline{x})$ fulfills the SAC of order one, we must go further and control all eight functions of three bits obtained by keeping each input bit of $f(\underline{x})$ fix (equal to zero resp. to one); they are listed in the following table. All of them fulfill the SAC.

y_1	y_2	y_3	$f_{1,0}$	$f_{1,1}$	$f_{2,0}$	$f_{2,1}$	$f_{3,0}$	$f_{3,1}$	$f_{4,0}$	$f_{4,1}$
0	0	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	1	0	1	0	1
0	1	0	0	0	0	1	0	1	0	1
0	1	1	1	0	1	1	1	1	1	1
1	0	0	0	0	1	0	1	0	1	0
1	0	1	1	0	0	0	0	0	0	0
1	1	0	1	0	0	0	0	0	0	0
1	1	1	1	1	0	1	0	1	1	1

Therefore, $f(\underline{x})$ fulfills the SAC of order one. Keeping each pair (x_i, x_j) constant and equal to $(0,0)$, $(0,1)$, $(1,0)$ and $(1,1)$ respectively, one gets

$\binom{4}{2} \cdot 4 = 6 \cdot 4 = 24$ functions of 2 bits and each of them fulfills the SAC. $f(\underline{x})$ thus even satisfies the SAC of order 2. It makes of course no sense to consider the SAC of order 3 for this function, since keeping three input bits constant yields functions of one variable for which the SAC is not defined.

3.2 Spectral Characterization for SAC of Higher Order

From example 9, it is clear that a boolean function of n bits can fulfill the SAC of order *at most* $n-2$.

We are interested in a spectral characterization of boolean functions that fulfill some SAC of higher order. We again consider $\hat{f}(\underline{x}) = (-1)^{f(\underline{x})}$ rather than $f(\underline{x})$. The following equation is quite similar to (38).

$$\hat{f}(\underline{x}) = x_i \cdot \hat{f}_{i,1}([x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]) + \overline{x_i} \cdot \hat{f}_{i,0}([x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]) \quad (39)$$

and can be written for each $i \in \{1, 2, \ldots, n\}$. The "subfunctions" $\hat{f}_{i,1}$ and $\hat{f}_{i,0}$ map Z_2^{n-1} onto $\{1, -1\}$, and all $2n$ subfunctions $\hat{f}_{i,j}$ must fulfill the SAC of order zero if $\hat{f}(\underline{x})$ is to fulfill the SAC of order 1. We introduce

$$\hat{f}_{i,I}(\underline{x}) = x_i \cdot \hat{f}_{i,1}([x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]) \quad \text{and} \quad (40)$$

$$\hat{f}_{i,II}(\underline{x}) = \overline{x_i} \cdot \hat{f}_{i,0}([x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]) \quad (41)$$

and we compute their Walsh-transforms.

$$\hat{F}_{i,I}(\underline{w}) = \sum_{\underline{x} \in Z_2^n} x_i \cdot \hat{f}_{i,1}([x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]) \cdot (-1)^{\underline{x} \cdot \underline{w}} \quad (42)$$

$$= \sum_{\underline{x} \,:\, x_i = 1} \hat{f}_{i,1}([x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n]) \cdot (-1)^{w_i \oplus [x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n] \cdot [w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n]} \quad (43)$$

With the substitutions

$$\underline{x}' = [x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n], \ \underline{x}' \in Z_2^{n-1} \quad \text{and} \quad (44)$$

$$\underline{w}' = [w_1, \ldots, w_{i-1}, w_{i+1}, \ldots, w_n], \ \underline{w}' \in Z_2^{n-1} \quad (45)$$

we obtain

$$\hat{F}_{i,I}(\underline{w}) = \sum_{\underline{x}' \in Z_2^{n-1}} \hat{f}_{i,1}(\underline{x}') \cdot (-1)^{w_i} \cdot (-1)^{\underline{x}' \cdot \underline{w}'} \quad (46)$$

$$= (-1)^{w_i} \cdot \hat{F}_{i,1}(\underline{w}'), \quad (47)$$

where $\hat{F}_{i,1}(\underline{w}')$ designates the Walsh-transform of $\hat{f}_{i,1}(\underline{x}')$. Similarly, we get

$$\hat{F}_{i,II}(\underline{w}) = \sum_{\underline{x} \,:\, x_i = 0} \hat{f}_{i,0}(\underline{x}') \cdot (-1)^{\underline{x}' \cdot \underline{w}'} = \hat{F}_{i,0}(\underline{w}'). \quad (48)$$

Because of the linearity of the Walsh-transform and the fact that "+" in expression (39) can be considered as integer addition (because always one of both terms on the right-hand side of (39) equals zero) we get:

$$\hat{F}(\underline{w}) = (-1)^{w_i} \cdot \hat{F}_{i,1}(\underline{w}') + \hat{F}_{i,0}(\underline{w}'), \quad \text{for all } i \in \{1, 2, \ldots, n\} \tag{49}$$

or equivalently

$$\hat{F}(\underline{w}) = \begin{cases} \hat{F}_{i,1}(\underline{w}') + \hat{F}_{i,0}(\underline{w}') & \text{for } w_i = 0, \\ -\hat{F}_{i,1}(\underline{w}') + \hat{F}_{i,0}(\underline{w}') & \text{for } w_i = 1. \end{cases} \tag{50}$$

Adding, respectively subtracting both equations gives

$$\hat{F}_{i,0}(\underline{w}') = \frac{1}{2}[\hat{F}(\underline{w}) + \hat{F}(\underline{w} \oplus \underline{c}_i)] \tag{51}$$

$$\hat{F}_{i,1}(\underline{w}') = \frac{1}{2} \cdot (-1)^{w_i} \cdot [\hat{F}(\underline{w}) - \hat{F}(\underline{w} \oplus \underline{c}_i)] \tag{52}$$

where $\underline{c}_i = [0, 0, \ldots, 0, c_i = 1, 0, \ldots, 0]$. By theorem 1, $\hat{f}(\underline{x})$ will fulfill the SAC of order 1 if and only if

$$\sum_{\underline{w}' \in Z_2^{n-1}} (-1)^{w_j'} \cdot \hat{F}_{i,0}^2(\underline{w}') = 0 \quad \text{and} \tag{53}$$

$$\sum_{\underline{w}' \in Z_2^{n-1}} (-1)^{w_j'} \cdot \hat{F}_{i,1}^2(\underline{w}') = 0 \tag{54}$$

for all $i, j \in \{1, 2, \ldots, n\}$ with $i \neq j$. Replacing $\hat{F}_{i,0}$ in (53) by its equivalent form from (51) gives

$$\sum_{\underline{w} \,:\, w_i = 0} \frac{1}{4}[\hat{F}(\underline{w}) + \hat{F}(\underline{w} \oplus \underline{c}_i)]^2 \cdot (-1)^{w_j'} = 0, \quad j' \neq i \tag{55}$$

or

$$\frac{1}{4} \sum_{\underline{w} \,:\, w_i = 0} [\hat{F}^2(\underline{w}) + \hat{F}^2(\underline{w} \oplus \underline{c}_i)] \cdot (-1)^{w_j'} + \frac{1}{2} \sum_{\underline{w} \,:\, w_i = 0} \hat{F}(\underline{w}) \hat{F}(\underline{w} \oplus \underline{c}_i) \cdot (-1)^{w_j'} = 0. \tag{56}$$

The first sum in (56) can be written as $\sum_{\underline{w} \in Z_2^n} \hat{F}^2(\underline{w}) \cdot (-1)^{w_j'}$ and therefore equals zero since $\hat{f}(\underline{x})$ fulfills the SAC of order 0 (necessary condition for fulfilling the SAC of order one). Thus

$$\sum_{\underline{w} \,:\, w_i = 0} \hat{F}(\underline{w}) \cdot \hat{F}(\underline{w} \oplus \underline{c}_i) \cdot (-1)^{w_{j'}} = 0, \quad (j' \neq i) \tag{57}$$

which implies

$$\sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w}) \cdot \hat{F}(\underline{w} \oplus \underline{c}_i) \cdot (-1)^{w_{j'}} = 0, \quad (j' \neq i). \tag{58}$$

Inserting (52) in (54) also leads to (58). Theorem 5 follows.

Theorem 5 *A function* $\hat{f}(\underline{x}) : Z_2^n \longrightarrow \{1,-1\}$ *fulfills the SAC of order 1 if and only if it fulfills the SAC of order zero and*

$$\sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus \underline{c}_i) \cdot (-1)^{w_j} = 0 \tag{59}$$

for all $i,j \in \{1,2,\ldots,n\}$ *with* $i \neq j$.

To verify whether a function of n bits fulfills the SAC of order 1 or not, at most SAC order 0 $\overbrace{n}$ + SAC order 1 $\overbrace{n \cdot (n-1)}$ checks are therefore required. The spectral characterizations of the SAC of order 2 and of higher orders can be derived in a similar way and are given without proof in the following two theorems.

Theorem 6 *A function* $\hat{f}(\underline{x}) : Z_2^n \longrightarrow \{1,-1\}$ *fulfills the SAC of order 2 if and only if it fulfills the SAC of orders 0 and 1, and*

$$\sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus \underline{c}_{i,j}) \cdot (-1)^{w_k} = 0 \tag{60}$$

for all distinct $i,j,k \in \{1,2,\ldots,\}$, *and with* $\underline{c}_{i,j}$ *denoting the* $n-$*tuple with a one at the* $i-$*th and* $j-$*th place and zeroes elsewhere.*

Verifying whether the SAC of order 2 is fulfilled or not thus requires at most $n + n(n-1) + \binom{n}{2}(n-2)$ checks.

Theorem 7 *A function* $\hat{f}(\underline{x}) : Z_2^n \longrightarrow \{1,-1\}$ *fulfills the SAC of order* m, $0 \leq m \leq n-2$, *if and only if*

$$\sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus \underline{c}_s) \cdot (-1)^{w_k} = 0 \tag{61}$$

for all $\underline{c}_s \in Z_2^n$ *with Hamming-weights* $s = 0,1,2,\ldots,m$ *and for all* $k \in \{1,2,\ldots,n\}$ *such that the* $k-$*th bit of* $\underline{c}_s$ *is zero.*

Verifying whether the SAC of order m is fulfilled or not requires at most $n + n(n-1) + \binom{n}{2}(n-2) + \binom{n}{3}(n-3) + \ldots + \binom{n}{m}(n-m)$ checks.

Example 10:
If $\hat{f}(\underline{x})$ is a boolean function of five bits, the following sums have to be checked:

SAC order 0 $\left\{\sum_{\underline{w} \in Z_2^n} \hat{F}^2(\underline{w}) \cdot (-1)^{w_j},\ j \in \{1,2,\ldots,n\},\right.$

SAC order 1
$$\begin{cases} \sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus [00001]) \cdot (-1)^{w_j}, & j \in \{1,2,3,4\}, \\ \sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus [00010]) \cdot (-1)^{w_j}, & j \in \{1,2,3,5\}, \\ \quad\vdots \qquad\qquad\qquad\qquad\qquad \vdots & \\ \sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus [10000]) \cdot (-1)^{w_j}, & j \in \{2,3,4,5\}, \end{cases}$$

$n \longrightarrow$	2	3	4	5	6
no SAC	8	192	61408	?	?
SAC order 0	8	48	3808	?	?
SAC order 1	-	16	288	?	?
SAC order 2	-	-	32	?	?
SAC order 3	-	-	-	64	?
SAC order 4	-	-	-	-	128

Table 1: Number of functions that fulfill the SAC of some given order

$$\text{SAC order 2} \quad \begin{cases} \sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus [00011]) \cdot (-1)^{w_j}, & j \in \{1,2,3\}, \\ \sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus [00101]) \cdot (-1)^{w_j}, & j \in \{1,2,4\}, \\ \quad \vdots \qquad\qquad\qquad\qquad \vdots & \\ \sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus [11000]) \cdot (-1)^{w_j}, & j \in \{3,4,5\}, \end{cases}$$

$$\text{SAC order 3} \quad \begin{cases} \sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus [00111]) \cdot (-1)^{w_j}, & j \in \{1,2\}, \\ \sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus [01011]) \cdot (-1)^{w_j}, & j \in \{1,3\}, \\ \quad \vdots \qquad\qquad\qquad\qquad \vdots & \\ \sum_{\underline{w} \in Z_2^n} \hat{F}(\underline{w})\hat{F}(\underline{w} \oplus [11100]) \cdot (-1)^{w_j}, & j \in \{4,5\}. \end{cases}$$

Exhaustive computer search through functions of 2, 3 and 4 bits allowed to count how many boolean functions fulfill the SAC of a given order. The results are listed in table 1. One can check that the columns for $n = 2, 3$ and 4 sum up to 2^{2^n}. Notice that no function is counted twice, although in fact each function that fulfills the SAC of some order m by definition also fulfills the SAC of orders $m-1, m-2, \ldots, 1, 0$.

3.3 Construction of Functions Fulfilling the SAC of Maximum Order

The method used to count the SAC-fulfilling functions of maximum order $n-2$ for $n = 5$ and $n = 6$ is a constructive one. The definition of the SAC of order m implies the following lemma.

Lemma 4 *A boolean function $f(\underline{x})$ of n bits fulfills the SAC of order m if and only if*

- *$f(\underline{x})$ fulfills the SAC of order 0, and*
- *any function obtained from $f(\underline{x})$ by keeping one input bit constant (equal to 0 or to 1) fulfills the SAC of order $m-1$.*

This gives rise to the idea of using functions of $n-1$ bits that fulfill the SAC of order $n-3$ as basic elements for the synthesis of functions of n bits that fulfill the SAC of order $n-2$.

Example 11:
The eight functions of two bits that fulfill the SAC of order zero are listed below.

x_1	x_2	$f_1(\underline{x})$	$f_2(\underline{x})$	$f_3(\underline{x})$	$f_4(\underline{x})$	$f_5(\underline{x})$	$f_6(\underline{x})$	$f_7(\underline{x})$	$f_8(\underline{x})$
0	0	0	0	0	0	1	1	1	1
0	1	0	0	1	1	0	0	1	1
1	0	0	1	0	1	0	1	0	1
1	1	1	0	0	1	0	1	1	0

We can define $f(\underline{x}) : Z_2^3 \longrightarrow \{0,1\}$ as

$$f(\underline{x}) = f([x_1, x_2, x_3]) = x_1 \cdot f_i([x_2, x_3]) + \overline{x_1} \cdot f_j([x_2, x_3]) \qquad (62)$$

with $i, j \in \{1, 2, \ldots, 8\}$, $i \neq j$ and we get $\binom{8}{2} = 28$ functions $f(\underline{x})$; sixteen of them can be checked to fulfill the SAC of order 1. We can be sure that no other function of three bits satisfies the SAC of order 1, since any such function necessarily is decomposable according to (62) (by Lemma 4).

The procedure used in example 11 can be applied to the sixteen functions of three bits that fulfill the SAC of order 1, and it yields the 32 functions of four bits that fulfill the SAC of order 2, and so on.

4 Conclusion

The Strict Avalanche Criterion of order m has been introduced which corresponds to a generalized definition of the known SAC. It has been shown that the SAC of any order can be easily characterized in the Walsh-domain. This representation was used for the construction of further SAC-fulfilling boolean functions. The application of SAC-fulfilling functions for cryptosystem-design has still to be studied. An application would be, for instance, to use such functions for the synthesis of S-boxes in substitution/permutation (SP) block-ciphers. Since an S-box has many inputs and n outputs, n SAC-fulfilling functions should be chosen and combined in some adequate manner. For example, statistical dependencies between output bits should be avoided. Statistical independencies between input $m-$tuples and the output of boolean functions is known as $m-$th order correlation-immunity. It might be interesting to examine whether there are restrictions in the compatibility of correlation-immunity and SAC of order m. Any boolean function that is

$m-$th order correlation-immune [6] has vanishing values of $F(\underline{w})$ for all $\underline{w}$'s with Hamming-weigths between one and m [5]. Exhaustive search for functions of three and four bits showed that eight functions of three bits as well as ninety-six functions of four bits are first-order correlation-immune and fulfill the SAC of order 1 at the same time.

Acknowledgements

The author is grateful to Thomas Siegenthaler for many constructive discussions and for his suggestions to improve this paper. She also wishes to thank Othmar Staffelbach for his helpful comments.

References

[1] A.F. Webster and S.E. Tavares, *"On the Design of S-Boxes"*, Advances in Cryptology: Crypto'85 proceedings, Springer, 1986.

[2] R.C. Titsworth, *"Correlation Properties of Cyclic Sequences"*, Thesis, California Institute of Technology, Pasadena, California, 1962.

[3] Th. Siegenthaler, *"Methoden für den Entwurf von Stream Cipher-Systemen"*, Diss. ETH No. 8185, Dec. 1986.

[4] S.C. Lee, *"Modern Switching Theory and Digital Design"*, Prentice-Hall, 1978.

[5] G.Z. Xiao, J.L. Massey, *"A Spectral Characterization of Correlation-Immune Combining Functions"*, to be published in IEEE Tr. on Information Theory.

[6] Th. Siegenthaler, *"Correlation-immunity of Nonlinear Combining Functions for Cryptographic Applications"*, IEEE Tr. on Information Theory, vol. IT-30, pp. 776-780, Oct. 1984.