Smart Hill Climbing Finds Better Boolean Functions

William Millan, Andrew Clark and Ed Dawson
Information Security Research Centre
Queensland University of Technology
GPO Box 2434, Brisbane, Queensland, Australia 4001
Facsimile: 61-7-3221 2384
Email: {millan,aclark,dawson}@fit.qut.edu.au

Abstract

Block and stream ciphers are made from Boolean functions that usually require a compromise between several conflicting cryptographic criteria. Although some constructions exist to generate Boolean functions satisfying one or more criteria, such as balance and high nonlinearity, there are often drawbacks to them such as low nonlinear order. In this paper we present a new algorithm for simple modification of a Boolean function truth table to improve both nonlinearity and balance. We also show how to modify a balanced function in two truth table positions so that the nonlinearity is increased and the balance is maintained. When the algorithm fails to find an improvement, one does not exist, and we have then identified a locally maximum function. We present results comparing the probability distributions of random functions with that of locally maximum functions found by our algorithms, and also comment on how the number of steps required to find a local maximum is affected by increasing the number of variables.

1 About Boolean Functions

Let $f(x)$ denote the binary truth table ($f(x) \in \{0,1\}$) and $\hat{f}(x)$ the corresponding *polarity truth table*, $\hat{f}(x) \in \{1,-1\}$. We have $\hat{f}(x) = (-1)^{f(x)} = 1 - 2f(x)$. The Hamming weight of a Boolean function is the number of ones in the binary truth table, or equivalently the number of -1s in the polarity truth table. A balanced function has the same number of zeroes and ones in the truth table. Balance is a primary cryptographic criterion: an imbalanced function has suboptimum unconditional entropy (ie. it is correlated to a constant function). We define the imbalance of a Boolean function as $I_f = \sum_x \hat{f}(x)$. The correlation between a function and the constant zero function is simply $\frac{I_f}{2^n}$, which is a value between -1 and 1. A function with zero imbalance is balanced and has no correlation to the constant functions.

Every function has a unique representation in the Algebraic Normal Form (ANF) as the binary coefficient vector of a fixed (positive) polarity Reed-Muller expansion (for example, see [5]). The ANF describes a two level circuit: an XOR sum of AND products. The *nonlinear order* or just *order* of a Boolean function is the size of the largest product term in the ANF. Order zero functions are constant, affine functions have order 1, and linear functions are those affine functions with a zero constant term in their ANF. The exclusive-or operation is linear; a linear function is an XOR sum of variables. We may specify a linear function by an n-bit vector ω that selects the variables in this sum: $L_\omega(x) = \omega_1 x_1 \oplus \cdots \oplus \omega_n x_n$.

The Hamming distance to linear functions is an important cryptographic property, since ciphers that employ nearly linear functions can be broken easily by a variety of methods (for example see [7, 4]). In particular, both differential and linear cryptanalysis techniques [2, 8] are resisted by highly nonlinear functions. Thus the minimum distance to any affine function is an important indicator of the cryptographic strength of a Boolean function. The *nonlinearity* of a Boolean function is this minimum distance, or the distance to the set of affine functions. We note that complementing the output will not change the nonlinearity of any Boolean function, so we need to consider the magnitude of the correlation to all linear functions, of which there are 2^n.

The Hamming distance between a pair of functions can be determined by evaluating both functions for all inputs and counting the disagreements. This process has complexity $O(2^n)$. It follows that determining the nonlinearity in this naive fashion will require $O(2^{2n})$ function evaluations, which is infeasible even for small n. However, a tool exists that enables the calculation of all linear correlation coefficients in $O(n2^n)$ operations. This is the fast Walsh-Hadamard Transform, and its uses in cryptography and elsewhere are well known [1, 13].

Let $\hat{F}(\omega)$ denote the Walsh-Hadamard Transform (WHT) of a Boolean function. Its calculation is defined as $\hat{F}(\omega) = \sum_x \hat{f}(x)\hat{L}_\omega(x)$. It is clear from this definition that the value of $\hat{F}(\omega)$ is closely related to the Hamming distance between $f(x)$ and the linear function $L_\omega(x)$. In fact the correlation to the linear function is given by $c(f, L_\omega) = \frac{\hat{F}(\omega)}{2^n}$. The nonlinearity N_f of $f(x)$ is related to the maximum magnitude of WHT values WH_{max}, by $N_f = \frac{1}{2} * (2^n - WH_{max})$. Clearly in order to increase the nonlinearity, we must decrease WH_{max}. A function is uncorrelated with linear function $L_\omega(x)$ when $\hat{F}(\omega) = 0$. We would like to find a Boolean function that has all WHT values equal to zero, since such a function has no correlation to any affine function. However, it is known [9] that such functions do not exist. A well known theorem, widely attributed to Parseval [6], states that the sum of the squares of the WHT values is the same constant for every Boolean function: $\sum_\omega \hat{F}^2(\omega) = 2^{2n}$. Thus a tradeoff exists in minimising affine correlation. When we alter a function so that its correlation to some affine function is reduced, the correlation to some other affine function is increased.

It is known that the Bent functions [12] satisfy the property that $|\hat{F}(\omega)| = 2^{\frac{n}{2}}$ for all ω. Bent functions exist only for even n, and they attain the maximum possible nonlinearity of $N_{bent} = 2^{n-1} - 2^{\frac{n}{2}-1}$. It is an open problem to determine an expression for the maximum nonlinearity of functions with an odd number of inputs. It is known that, for n odd, it is possible to construct a function with nonlinearity $2^{n-1} - 2^{\frac{n-1}{2}}$ by concatenating Bent functions. It is known that for $n = 3, 5, 7$ that this is in fact the upper bound of nonlinearity. The only value of n for which it is known that this value is not the upper bound is $n = 15$ [10, 11]. We note that determining the covering radius of a Reed-Muller code is the same problem as finding an upper bound on low order approximation. It is a well known open problem to find the covering radius of Reed-Muller codes, so it is not known to what extent functions may resist low order approximations.

There seem to be no formal tools for low order nonlinear approximation, so we leave this difficult area, and instead concentrate on improving the nonlinearity of Boolean functions in a systematic way.

In this paper, we present algorithms that provide a list of truth table positions that, if complemented, will result in a Boolean function with higher nonlinearity. The approach is based on the observation that small changes to a truth table result in small magnitude changes to the WHT values. In particular, a single truth table complementation will cause every $\hat{F}(\omega)$ to alter by ± 2. Two truth table changes will cause $\Delta\hat{F}(\omega) \in \{-4, 0, 4\}$. We use these facts in the next section to prove conditions required for small changes to increase nonlinearity. When two changes are made, the Hamming weight can be maintained while nonlinearity is increased.

These techniques provide a fast way of hill-climbing the Boolean function terrain to locate highly nonlinear Boolean functions that would be difficult to obtain by a purely random search or exhaustive hill climbing.

2 Improving Nonlinearity

Consider altering a function $f(x)$ by complementing the output for a single input x_1, with the nonlinearity increasing. We define the 1-Improvement Set of $f(x)$, 1-IS_f , as the set of all inputs such that complementing the corresponding output of any one of them will increase the nonlinearity of the function.

Definition 1 *Let* $g(x) = f(x) \oplus 1$ *for* $x = x_1$ *and* $g(x) = f(x)$ *for all other* x. *If* $N_g > N_f$ *then* $x_1 \in$ *1-*IS_f. □

If 1-IS_f is empty, the function is a 1-local maximum for nonlinearity. Of course all Bent functions are global maxima, so their 1-Improvement Sets are empty. There also exist sub-optimum local maxima that will be found by hill climbing algorithms. It is computationally intensive to exhaustively alter truth table positions, find new WHTs and so determine the set 1-IS_f, so we seek a fast, systematic way to determine the 1-Improvement Set of a given Boolean function from its truth table and Walsh-Hadamard transform. In this section we present easily checked conditions for an input x to be in the 1-Improvement Set.

Definition 2 *Let $f(x)$ be a Boolean function with Walsh-Hadamard Transform $\hat{F}(\omega)$. Let WH_{max} denote the maximum absolute value of $\hat{F}(\omega)$. There will exist one or more linear functions $L_\omega(x)$ that have minimum distance to $f(x)$, and $|\hat{F}(\omega)| = WH_{max}$ for these ω. Let us define the following sets:*

$$\begin{aligned} W_1^+ &= \{\omega : \hat{F}(\omega) = WH_{max}\} \textit{ and} \\ W_1^- &= \{\omega : \hat{F}(\omega) = -WH_{max}\}. \end{aligned}$$

We also need to define sets of ω for which the WHT magnitude is close to the maximum.

$$\begin{aligned} W_2^+ &= \{\omega : \hat{F}(\omega) = WH_{max} - 2\}, \\ W_2^- &= \{\omega : \hat{F}(\omega) = -(WH_{max} - 2)\}, \\ W_3^+ &= \{\omega : \hat{F}(\omega) = WH_{max} - 4\}, \textit{ and} \\ W_3^- &= \{\omega : \hat{F}(\omega) = -(WH_{max} - 4)\}. \end{aligned}$$

□

When a truth table is changed in exactly one place, all WHT values are changed by +2 or -2. It follows that in order to increase the nonlinearity we need to make the WHT values in set W_1^+ change by -2, the WHT values in set W_1^- change by +2, and also make the WHT values in set W_2^+ change by -2 and the WHT values in set W_2^- change by +2. The first two conditions are obvious, and the second two conditions are required so that all other $|\hat{F}(\omega)|$ remain less than WH_{max}. These conditions can be translated into simple tests.

Theorem 1 *Given a Boolean function $f(x)$ with WHT $\hat{F}(\omega)$, we define sets $W^+ = W_1^+ \cup W_2^+$ and $W^- = W_1^- \cup W_2^-$. For an input x to be an element of the Improvement Set, all of the following conditions must be satisfied.*

(i) $f(x) = L_\omega(x)$ for all $\omega \in W^+$

and

(ii) $f(x) \neq L_\omega(x)$ *for all* $\omega \in W^-$.

If the function $f(x)$ *is not balanced, and we wish to reduce the imbalance, we impose the additional restriction that*

(iii) when $\hat{F}(0) > 0$, $f(x) = 0$, *else* $f(x) = 1$. □

Proof: We start by considering the conditions to make WHT values change by a desired amount. When $\hat{F}(\omega)$ is positive, there are more 1 than -1 in the polarity truth table, and more 0 than 1 in the binary truth table of $f(x) \oplus L_\omega(x)$. It follows that to make $\Delta\hat{F}(\omega) = -2$, we must change any single 0 to 1 in the truth table of $f(x) \oplus L_\omega(x)$. This means that we select an x to change such that $f(x) = L_\omega(x)$. We desire a -2 change for all WHT values with $\omega \in W^+$, so this proves condition (i). A similar argument proves condition (ii). A function is balanced when $\hat{F}(0) = 0$, so to reduce the imbalance we must select x according to condition (iii). □

We often seek to improve the nonlinearity of balanced Boolean functions, while retaining balance. Clearly this requires an even number of truth table changes. We now present the conditions on a pair of inputs x_1, x_2 so that complementing both their function values causes an increase in nonlinearity, without changing the Hamming weight. We define the 2-Improvement Set, 2-IS_f, as the set of all such input pairs. A function for which no pair satisfies these conditions is said to be a 2-local maximum.

Theorem 2 *Given a Boolean function* $f(x)$ *with WHT* $\hat{F}(\omega)$, *we define sets* $W_1 = W_1^+ \cup W_1^-$, $W_{2,3}^+ = W_2^+ \cup W_3^+$ *and* $W_{2,3}^- = W_2^- \cup W_3^-$. *A pair of inputs* (x_1, x_2) *is in the 2-Improvement Set of* $f(x)$ *if and only if all of the following conditions are satisfied:*

(i) $f(x_1) \neq f(x_2)$

(ii) $L_\omega(x_1) \neq L_\omega(x_2)$ *for all* $\omega \in W_1$

(iii) $f(x_i) = L_\omega(x_i)$, $i \in \{1,2\}$, *for all* $\omega \in W_1^+$

(iv) $f(x_i) \neq L_\omega(x_i)$, $i \in \{1,2\}$, *for all* $\omega \in W_1^-$

(v) for all $\omega \in W_{2,3}^+$, *if* $L_\omega(x_1) \neq L_\omega(x_2)$ *then* $f(x_i) = L_\omega(x_i)$, $i \in \{1,2\}$

(vi) for all $\omega \in W^{-}_{2,3}$, if $L_\omega(x_1) \neq L_\omega(x_2)$ then $f(x_i) \neq L_\omega(x_i)$, $i \in \{1,2\}$ □

Proof: Condition (i) is required to maintain the Hamming weight. Conditions (ii),(iii) and (iv) are proven similarly to theorem 1. In order to stop the correlation to other linear functions increasing too much, we require that $\Delta\hat{F}(\omega) \neq +4$, for all $\omega \in W^{+}_{2,3}$, and it follows that not both of $f(x_i) \oplus L_\omega(x_i) = 1$, or equivalently that at least one of $f(x_i) \oplus L_\omega(x_i) = 0$. Consequently,

$$[f(x_1) \oplus L_\omega(x_1)][f(x_2) \oplus L_\omega(x_2)] = 0,$$

and expanding this, noting from (i) that $f(x_1)f(x_2) = 0$, we have

$$f(x_1)L_\omega(x_2) \oplus f(x_2)L_\omega(x_1) \oplus L_\omega(x_1)L_\omega(x_2) = 0.$$

We need to consider four cases to find the exact conditions for this expression to be satisfied:

(a) When $L_\omega(x_1) = L_\omega(x_2) = 0$ the expression is satisfied and no further conditions on (x_1, x_2) are required.

(b) When $L_\omega(x_1) = L_\omega(x_2) = 1$ the expression becomes $f(x_1) \oplus f(x_2) = 1$ which is equivalent to condition (i).

(c) When $L_\omega(x_1) = 0$ and $L_\omega(x_2) = 1$ the expression becomes $f(x_1) = 0$.

(d) When $L_\omega(x_1) = 1$ and $L_\omega(x_2) = 0$ the expression becomes $f(x_2) = 0$.

Combining (a)-(d) we see that when $L_\omega(x_1) \neq L_\omega(x_2)$ for $\omega \in W^{+}_{2,3}$ we require that $f(x_i) = L_\omega(x_i)$, $i = 1, 2$, thus proving condition (v). The Proof of (vi) is similar. □

The following theorem shows how to modify the WHT of a Boolean function that has been altered in a single truth table position, with complexity $O(2^n)$. We note that the algorithm for incremental improvement of Boolean functions suggested in [3] recomputes the WHT after every single bit change regardless of whether that change improves the nonlinearity. Our algorithms are superior on two counts - every change is an improvement and the new WHT is found n times faster.

Theorem 3 *Let $g(x)$ be obtained from $f(x)$ by complementing the output for a single input, x_1. Then each component of the WHT of $g(x)$, $\hat{G}(\omega) = \hat{F}(\omega) +$*

$\Delta(\omega)$, can be obtained as follows: If $f(x_1) = L_\omega(x_1)$, then $\Delta(\omega) = -2$, else $\Delta(\omega) = +2$

Proof: When $f(x_1) = L_\omega(x)$, we have $(-1)^{f(x_1) \oplus L_\omega(x_1)} = 1$, which contributes to the sum in $\hat{F}(x_1)$. Changing the value of $f(x_1)$ changes this contribution to -1, so $\Delta\hat{F}(\omega) = -2$. Similarly when $f(x_1) \neq L_\omega(x)$, $\Delta\hat{F}(\omega) = +2$. □

3 Implementation and Results

In this section the implementation details for the one step improvement and two step improvement algorithms are given - **HillClimb** and **HillClimb2**. We note that condition (ii) of Theorem 2 is redundant, and is not referred to in the implementation of that algorithm.

```
• HillClimb(BF, WHT)
    1. Determine maximum value of the Walsh-Hadamard transform $\text{WH}_{\text{max}}$.
    2. By parsing the WHT find the values of $\omega$ which belong to the sets
       $W_1^+$, $W_1^-$, $W_2^+$ and $W_2^-$. At the completion of this step there should
       be two lists: $W^+ = W_1^+ \cup W_2^+$ and $W^- = W_1^- \cup W_2^-$. NB. Either
       (but not both) of $W^+$ and $W^-$ may be empty.
    3. For $i$ in $0 \ldots 2^n - 1$, do
        (a) Let $b_i$ denote the $i^{\text{th}}$ bit in the truth table of BF.
        (b) Parse the sets $W^+$ and $W^-$ ensuring that conditions (iii) and
            (iv) in Theorem 2 are satisfied - if not skip to Step 3e.
        (c) We have a candidate for improvement. Complement $b_i$ in the
            truth table of BF (denote the resulting boolean function BF'),
            update the WHT (becoming WHT') by using Theorem 3, and
            call HillClimb(BF', WHT').
        (d) Skip to Step 4.
        (e) Increment $i$ ($i = i + 1$).
    4. BF represents a 1-local maximum - terminate processing.
```

```
• HillClimb2(BF, WHT)
  1. Determine maximum value of the WHT $\mathrm{WH}_{\mathrm{max}}$.
  2. By parsing the WHT obtain the sets $W_1^+$, $W_1^-$, $W_{2,3}^+$ and $W_{2,3}^-$.
  3. For $i$ in $0 \ldots 2^n - 1$, do
     (a) Let $b_i$ denote the $i^{\mathrm{th}}$ bit in the truth table of BF.
     (b) Parse the sets $W^+$ and $W^-$ ensuring that conditions (iii) and (iv) in Theorem 2 are satisfied - if they are add $i$ to $c_{b_i}$.
  4. For each element of $c_0$, do
     (a) For each element of $c_1$, do
         i. Check conditions (v) and (vi) of Theorem 2 and if they are satisfied complement the corresponding bits in the truth table (call the resulting truth table BF'), find the adjusted WHT (becoming WHT') by applying Theorem 3 twice and call HillClimb2(BF', WHT'). Skip to Step 5.
  5. BF represents a 2-local maximum - terminate processing.
```

We now present examples of the distribution of nonlinearity for random functions and random balanced functions, compared with the nonlinearity of locally maximum functions obtained by our two algorithms. In Figures 1 and 2 we compare random functions with the maxima found by **HillClimb**, for 8 and 12 input variables respectively. From these graphs it is clear that random functions have a smooth, bell-shaped distribution, whereas 1-local maxima are much more likely to have an even value for nonlinearity. It is also clear that hill climbing will find highly nonlinear functions much more easily than random search.

Figures 3 and 4 illustrate the performance of random generation versus hill climbing, when confined to balanced functions only, for functions with 8 and 12 input variables respectively. Note that **HillClimb2** conserves Hamming weight. For our tests we started with balanced functions so that the 2-local maxima were also balanced. This allows a direct comparison with randomly generated balanced functions. It is easy to show that the nonlinearity of a

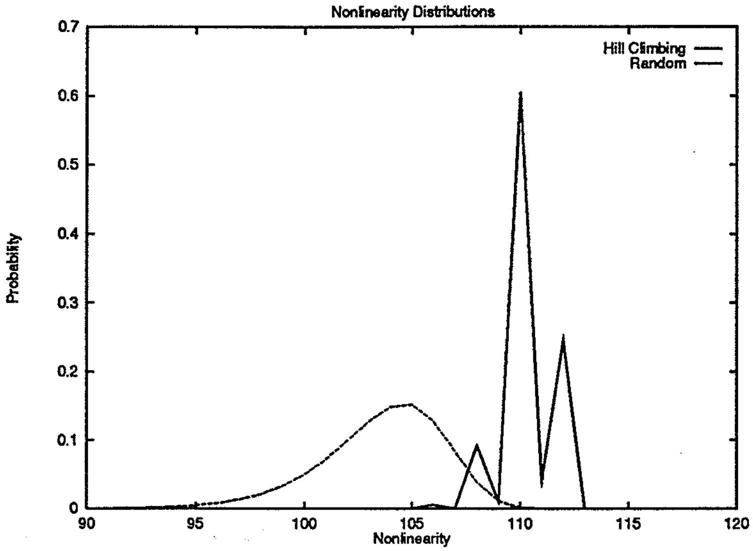

Figure 1: A Comparison of Hill Climbing with Random Generation, n=8.

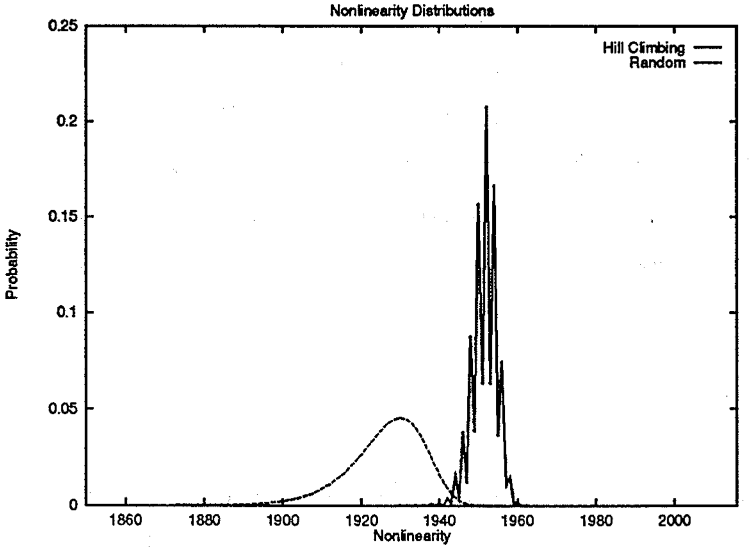

Figure 2: A Comparison of Hill Climbing with Random Generation, n=12.

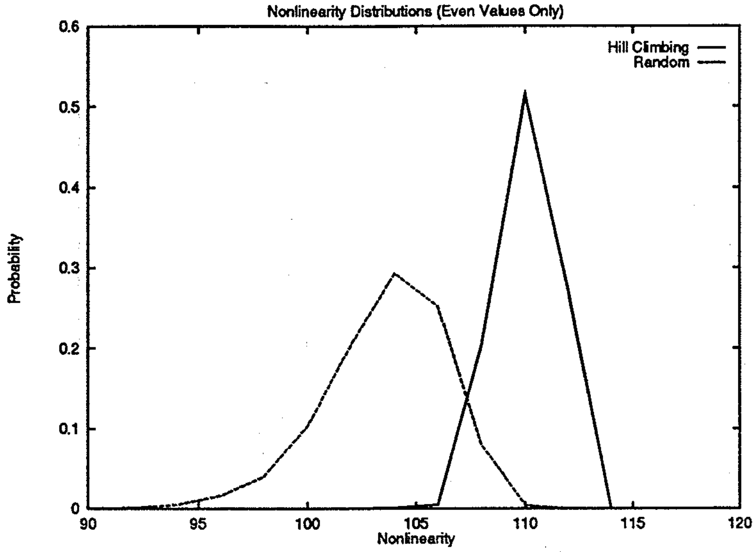

Figure 3: A Comparison of Hill Climbing with Random Balanced Generation, n=8.

balanced function is always even. For simplicity we only show results for even values of the nonlinearity in these graphs.

Figure 5 shows how the average number of steps to find a local maximum is changing with the number of variables. These results suggest that as n increases the distance from a random function to a local maximum is increasing in an exponential-like manner. The implication of this is that these hill climbing algorithms will be more effective for large n. It follows from Theorem 3 that making n hill climbing steps can be done in approximately the same time as a single random generation and complete fast WHT. The relative efficiency of the hill climbing algorithm improves as n increases.

4 Conclusion

We have presented two useful algorithms for the improvement of Boolean functions. With these tools it is now feasible to perform hill climbing to obtain locally maximum functions. In conjunction with heuristic search methods, these tools provide a means to find strong Boolean functions for cryptographic applications.

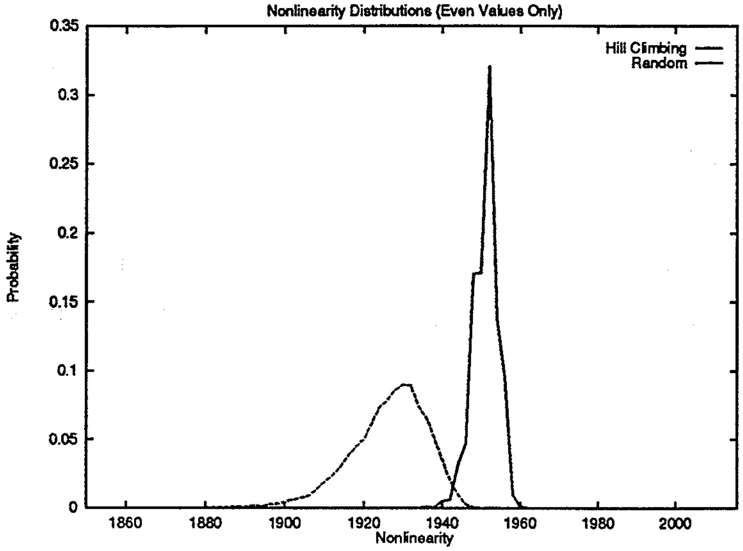

Figure 4: A Comparison of Hill Climbing with Random Balanced Generation, n=12.

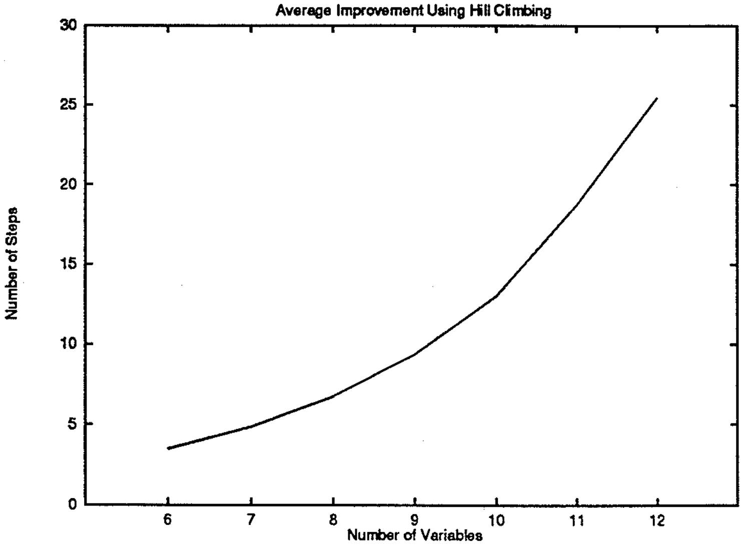

Figure 5: Average Distance To Local Maxima For Various n.

Several open problems remain. One is to determine the general relationship between the nonlinearity of a Boolean function and the size of its Improvement Set. Clearly local maxima have empty sets and local minima have full sets, but for arbitrary functions the relationship is not clear. Initial experiments have shown that two functions with the same nonlinearity can have different sized Improvement Sets, and the results suggest that the function closer to a local maximum has a smaller Improvement Set, so that in a hill climbing algorithm it may be of benefit to maintain as large a set as possible, thus avoiding grossly sub-optimum local maxima.

Smart hill climbing may be adapted to generate Boolean functions satisfying other cryptographic criteria, and to improve the nonlinearity of bijective S-boxes. These topics are the subject on ongoing research.

References

[1] K.G. Beauchamp. *Applications of Walsh and Related Functions.* Academic Press, 1984.

[2] E. Biham and A. Shamir. Differential cryptanalysis of DES-like cryptosystems. In *Advances in Cryptology - Crypto '90, Proceedings, LNCS*, volume 537, pages 2–21. Springer-Verlag, 1991.

[3] R. Forre. Methods and Instruments for Designing S-Boxes. *Journal of Cryptology*, 2(3):115–130, 1990.

[4] J.Dj. Golic. Linear Cryptanalysis of Stream Ciphers. In *Fast Software Encryption, 1994 Leuven Workshop, LNCS*, volume 1008, pages 154–169, December 1994.

[5] D.H. Green and G.A. Khuwaja. Simplification of switching functions expressed in Reed-Muller algebraic form. *IEE Proceedings, Pt E.*, 139(6):511–518, November 1992.

[6] F.J. MacWilliams and N.J.A. Sloane. *The Theory of Error Correcting Codes.* North-Holland Publishing Company, Amsterdam, 1978.

[7] J.L. Massey. Shift-Register Synthesis and BCH Decoding. *IEEE Transactions on Information Theory*, 15(1):122–127, January 1969.

[8] M. Matsui. Linear Cryptanalysis Method for DES Cipher. In *Advances in Cryptology - Eurocrypt '93, Proceedings, LNCS*, volume 765, pages 386–397. Springer-Verlag, 1994.

[9] W. Meier and O. Staffelbach. Nonlinearity Criteria for Cryptographic Functions. In *Advances in Cryptology - Eurocrypt '89, Proceedings, LNCS*, volume 434, pages 549–562. Springer-Verlag, 1990.

[10] N.J. Patterson and D.H. Wiedemann. The Covering Radius of the $(2^{15}, 16)$ Reed-Muller Code is at least 16276. *IEEE Transactions on Information Theory*, 29(3):354–356, May 1983.

[11] N.J. Patterson and D.H. Wiedemann. Correction to 'the covering radius of the $(2^{15}, 16)$ Reed-Muller code is at least 16276'. *IEEE Transactions on Information Theory*, 36(2):443, March 1990.

[12] O.S. Rothaus. On Bent Functions. *Journal of Combinatorial Theory (A)*, 20:300–305, 1976.

[13] G-Z. Xiao and J.L. Massey. A Spectral Characterization of Correlation-Immune Combining Functions. *IEEE Transactions on Information Theory*, 34(3):569–571, May 1988.

Balanced Boolean functions satisfying PC(2) and very large degree

Tomoyoshi Honda Takashi Satoh Tetsu Iwata
Kaoru Kurosawa

Dept. of Electrical and Electronic Engineering,
Faculty of Engineering,
Tokyo Institute of Technology

2–12–1 O-okayama, Meguro-ku, Tokyo 152, Japan
{tsato, tez, kurosawa}@ss.titech.ac.jp

Abstract: We show a Boolean function $f(x_1, \ldots, x_n)$ satisfying PC(2) such that $\deg(f)$ is almost $n - \log_2 n$. This degree is much larger than the best currently known degree $n/2$. We also show a balanced PC(2) function such that $\deg(f)$ is almost $n - \log_2 n$.

Key words: Boolean function, propagation criterion, large degree, balance.

1 Introduction

The security of DES-like block ciphers are often discussed by viewing their F functions (or S-boxes) as a set of Boolean functions. PC(l) [5] is an important cryptographic criterion of such Boolean functions. A Boolean function $f(x_1, \ldots, x_n)$ satisfies PC(l) if complementing any t $(1 \leq t \leq l)$ input bits results in changing the output bit with the probability one half. f satisfies SAC if and only if f satisfies PC(1). f is a bent function if and only if f satisfies PC(n).

On the other hand, balancedness and algebraic degree are other important cryptographic criteria. Let $\deg(f)$ denote the degree of the highest degree term in the algebraic normal form of f. Then $\deg(f)$ must be large. Actually, Jakobsen and Knudsen [2] showed an attack on block ciphers with small $\deg(f)$ recently.

f satisfies SAC(k) if any function obtained from f by keeping any k input bits constant satisfies SAC. Preneel et al. [5] showed that

$$\deg(f) \leq n - k - 1 \tag{1}$$

if f satisfies SAC(k). Recently, Kurosawa and Satoh [3] showed that the equality of this bound can be satisfied.

Now suppose that $f(x_1, \ldots, x_n)$ satisfies PC(2). Then since f satisfies SAC, we obtain a trivial upper bound on $\deg(f)$ such that

$$\deg(f) \leq n - 1$$

from eq.(1). No other better bound is known. On the other hand, a bent function satisfies PC(2) because it satisfies PC(n). Further, there exists a Maiorana type bent function $f(x_1, \ldots, x_n)$ such that $\deg(f) = n/2$. (This is the maximum degree of bent functions.) Therefore, there exists a PC(2) function such that $\deg(f) = n/2$. However, no PC(2) function is known such that $\deg(f) > n/2$.

Thus, there is a big gap between the current realization and an upper bound on $\deg(f)$ of PC(2) functions.

In this paper, we show a PC(2) function $f(x_1, \ldots, x_n)$ such that $\deg(f)$ is almost $n - \log_2 n$. This degree is much larger than the best degree so far. We also show a balanced PC(2) function $f(x_1, \ldots, x_n)$ such that $\deg(f)$ is almost $n - \log_2 n$.

2 Preliminaries

We use square brackets to denote vectors like $[a_1, \ldots, a_n]$ and round brackets to denote functions like $f(x_1, \ldots, x_n)$.

2.1 Algebraic degree

Definition 2.1 *The following form is called the algebraic normal form of f.*

$$f(x) = a_0 \oplus \bigoplus_{i=1}^{n} a_i x_i \oplus \bigoplus_{1 \leq i < j \leq n} a_{ij} x_i x_j \oplus \cdots \oplus a_{12\cdots n} x_1 x_2 \cdots x_n.$$

$\deg(f)$ *denotes the degree of the highest degree term in the algebraic normal form of f.*

2.2 Balance

Definition 2.2 *For any $\alpha \in \{0,1\}^n$, $W(\alpha)$ denotes the Hamming weight of α.*

Definition 2.3 *A Boolean function $f(x)$ is balanced if $|\{x \mid f(x) = 0\}| = |\{x \mid f(x) = 1\}| = 2^{n-1}$, where $x = [x_1, \ldots, x_n]$.*

Definition 2.4 *We call $f(x) = a_0 \oplus a_1x_1 \oplus \cdots \oplus a_nx_n$ an affine function.*

Proposition 2.1 *A non-constant affine function f is balanced.*

Proposition 2.2 *[7] A Boolean function $f(x_1, \ldots, x_n) \oplus g(y_1, \ldots, y_k)$ is balanced if f is balanced or g is balanced.*

2.3 PC(l), SAC and Bent function

Definition 2.5 *[5] $f(x_1, \ldots, x_n)$ satisfies PC(l) if $f(x) \oplus f(x \oplus \alpha)$ is balanced for any $\alpha \in \{0,1\}^n$ such that $1 \leq W(\alpha) \leq l$. We say that f is a PC(l) function if f satisfies PC(l).*

Definition 2.6 *[1, 8]*

1. *f satisfies SAC if and only if f satisfies PC(1). We say that f is an SAC function if f satisfies SAC.*

2. *f satisfies SAC(k) if any function obtained from f by keeping any k input bits constant satisfies SAC. We say that f is an SAC(k) function if f satisfies SAC(k).*

Proposition 2.3 *[5] If $f(x_1, \ldots, x_n)$ satisfies SAC(k), then*

$$\deg(f) \leq n - k - 1 \ .$$

Definition 2.7 *[4] $f(x_1, \ldots, x_n)$ is a bent function if and only if f satisfies PC(n).*

Proposition 2.4 *[6] If $f(x_1, \ldots, x_n)$ is a bent function, then*

$$\deg(f) \leq n/2 \ .$$

3 PC(2) with very large degree

Suppose that $f(x_1, \ldots, x_n)$ satisfies PC(2). Then from Proposition 2.3,

$$\deg(f) \leq n - 1 \ .$$

On the other hand, there exists a PC(2) function such that $\deg(f) = n/2$. (For example, a Maiorana type bent function.) However, no PC(2) function is known such that $\deg(f) > n/2$.

In this section, we show a PC(2) function $f(x_1, \ldots, x_n)$ such that $\deg(f)$ is almost $n - \log_2 n$. This degree is much larger than the best degree so far.

Definition 3.1 *For a Boolean function $f(x_1, \ldots, x_n)$, define*

$$\frac{df}{dx_i} \triangleq f(x_1, \ldots, x_i, \ldots, x_n) \oplus f(x_1, \ldots, x_i \oplus 1, \ldots, x_n) \ ,$$
$$\frac{df}{dx_i x_j} \triangleq f(x_1, \ldots, x_i, \ldots, x_j, \ldots, x_n) \oplus f(x_1, \ldots, x_i \oplus 1, \ldots, x_j \oplus 1, \ldots, x_n) \ .$$

Theorem 3.1 *Let $x = [x_1, \ldots, x_l]$ and $y = [y_1, \ldots, y_k]$. For a $l \times (k+1)$ matrix*

$$H = \begin{pmatrix} a_{11} & \cdots & a_{1k} & 1 \\ \vdots & & \vdots & \vdots \\ a_{l1} & \cdots & a_{lk} & 1 \end{pmatrix}$$

and any three Boolean functions $f_1(x), f_2(y), f_3(z)$, define

$$g(x, y, z) \triangleq f_1(x) \oplus f_2(y) \oplus f_3(z) \oplus [x_1, \ldots, x_l] H [y_1, \ldots, y_k, z]^T,$$

where g has $l + k + 1$ input bits. Then g satisfies PC(2) if H satisfies the following four conditions.

1. $[a_{1i}, \ldots, a_{li}]^T \neq [0, \ldots, 0]^T$ *for any i.*
2. $[a_{1i}, \ldots, a_{li}]^T \neq [1, \ldots, 1]^T$ *for any i.*
3. $[a_{1i}, \ldots, a_{li}]^T \neq [a_{1j}, \ldots, a_{lj}]^T$ *for any i and j such that $i \neq j$.*
4. $[a_{i1}, \ldots, a_{ik}] \neq [a_{j1}, \ldots, a_{jk}]$ *for any i and j such that $i \neq j$.*

Proof. g satisfies PC(2) if and only if

$$\frac{dg}{dx_i}, \frac{dg}{dy_i}, \frac{dg}{dz}, \frac{dg}{dx_ix_j}, \frac{dg}{dy_iy_j}, \frac{dg}{dx_iy_j}, \frac{dg}{dzx_i}, \frac{dg}{dzy_i}$$

are all balanced. First, for any i,

$$\frac{dg}{dx_i} = \frac{df_1}{dx_i} \oplus a_{i1}y_1 \oplus \cdots \oplus a_{ik}y_k \oplus z$$

is balanced because $a_{i1}y_1 \oplus \cdots \oplus a_{ik}y_k \oplus z$ is a non-constant affine function. Similarly, $\frac{dg}{dy_i}, \frac{dg}{dz}$ are all balanced. Next, for $i \neq j$,

$$\begin{aligned}\frac{dg}{dx_ix_j} &= \frac{df_1}{dx_ix_j} \oplus (a_{i1}y_1 \oplus \cdots \oplus a_{ik}y_k) \oplus (a_{j1}y_1 \oplus \cdots \oplus a_{jk}y_k) \\ &= \frac{df_1}{dx_ix_j} \oplus (a_{i1} \oplus a_{j1})y_1 \oplus \cdots \oplus (a_{ik} \oplus a_{jk})y_k \ .\end{aligned}$$

From the fourth condition, $[a_{i1}, \ldots, a_{ik}] \neq [a_{j1}, \ldots, a_{jk}]$ and hence $(a_{i1} \oplus a_{j1})y_1 \oplus \cdots \oplus (a_{ik} \oplus a_{jk})y_k$ is a non-constant affine function. Therefore, $\frac{dg}{dx_ix_j}$ is balanced. Similarly, the other cases are all balanced. □

Corollary 3.1 *There exists a Boolean function g satisfying PC(2) such that*

1. *g has $n \triangleq l + 2^l - 1$ input bits.*
2. $\deg(g) = n - l - 1 \approx n - \log_2 n$.

Proof. In Theorem 3.1, let

$$H = \begin{pmatrix} & & & 1 \\ e_1 & \cdots & e_{2^l-2} & \vdots \\ & & & 1 \end{pmatrix},$$

where e_i is the column vector of size l representing the value of i in binary. Note that H is a parity check matrix of the Hamming $[2^l - 1, 2^l - 1 - l, 3]$ code. Then it is easy to see that the four conditions on H are satisfied. Hence, we can choose k as $k = 2^l - 2$. (Actually, the maximum value of k is $2^l - 2$ because e_0 and e_{2^l-1} can not be used from the conditions *1* and *2*.) Further, let

$$f_2(y) = y_1y_2 \cdots y_k \ .$$

Then

$$\begin{aligned} n &= l + k + 1 = l + 2^l - 1 \\ \deg(g) &= \deg(f_2) = k = 2^l - 2 \\ &= n - l - 1 \\ &\approx n - \log_2 n. \end{aligned}$$

□

4 Balanced PC(2) with very large degree

In this section, we show a balanced PC(2) function such that $\deg(f)$ is almost $n - \log_2 n$.

Definition 4.1 *We say that f is balanced for a matrix H if*

$$|\{x \mid f(x) = 0, Hx^T = 0\}| = |\{x \mid f(x) = 1, Hx^T = 0\}| \ .$$

Theorem 4.1 *Let $x = [x_1, \ldots, x_l]$ and $y = [y_1, \ldots, y_k]$. For a $l \times (k+1)$ matrix*

$$H = \begin{pmatrix} a_{11} & \cdots & a_{1k} & 1 \\ \vdots & & \vdots & \vdots \\ a_{l1} & \cdots & a_{lk} & 1 \end{pmatrix}$$

and any two Boolean functions $f_2(y), f_3(z)$, define

$$g(x, y, z) \triangleq f_2(y) \oplus f_3(z) \oplus [x_1, \ldots, x_l] H [y_1, \ldots, y_k, z]^T,$$

where g has $l + k + 1$ input bits. Then g is balanced and satisfies PC(2) if the following five conditions are satisfied.

1. *$[a_{1i}, \ldots, a_{li}]^T \neq [0, \ldots, 0]^T$ for any i.*
2. *$[a_{1i}, \ldots, a_{li}]^T \neq [1, \ldots, 1]^T$ for any i.*
3. *$[a_{1i}, \ldots, a_{li}]^T \neq [a_{1j}, \ldots, a_{lj}]^T$ for any i and j such that $i \neq j$.*
4. *$[a_{i1}, \ldots, a_{ik}] \neq [a_{j1}, \ldots, a_{jk}]$ for any i and j such that $i \neq j$.*
5. *$f_2(y) \oplus f_3(z)$ is balanced for H.*

Proof. From Theorem 3.1, it is clear that g satisfies PC(2). We prove that g is balanced. Let

$$[s_1, \ldots, s_l]^T \triangleq H[y_1, \ldots, y_k, z]^T \ .$$

Then g is written as

$$\begin{aligned} g &= f_2(y) \oplus f_3(z) \oplus [x_1, \ldots, x_l] H[y_1, \ldots, y_k, z]^T \\ &= f_2(y) \oplus f_3(z) \oplus x_1 s_1 \oplus \cdots \oplus x_l s_l \ . \end{aligned}$$

If $[s_1, \ldots, s_l] \neq [0, \ldots, 0]$, then $x_1 s_1 \oplus \cdots \oplus x_l s_l$ is a non-constant affine function. In this case, from Propositions 2.1 and 2.2, g is balanced. If $[s_1, \ldots, s_l] = [0, \ldots, 0]$, then

$$g = f_2(y) \oplus f_3(z) \ .$$

From the fifth condition, $f_2(y) \oplus f_3(z)$ is balanced and g is balanced. □

Corollary 4.1 *There exists a balanced Boolean function g satisfying PC(2) such that*

1. *g has $n \triangleq l + 2^l - 1$ input bits.*

2. $\deg(g) = n - l - 1 \approx n - \log_2 n$.

Proof. As in the proof of Corollary 3.1, let $k = 2^l - 2$ and let

$$H = \begin{pmatrix} & & & 1 \\ e_1 & \cdots & e_{2^l-2} & \vdots \\ & & & 1 \end{pmatrix} ,$$

where e_i is the column vector of size l representing the value of i in binary. The first four conditions of Theorem 4.1 are satisfied. Let $f_3(z) = 0$. We show that there exists $f_2(y)$ such that

1. the fifth condition is satisfied.

2. $f_2(y)$ includes a term $y_1 y_2 \ldots y_k$.

Let

$$\begin{aligned} A_1 &\triangleq \{y \mid H[y, 0]^T = 0\} \ , \\ A_2 &\triangleq \{y \mid H[y, 1]^T = 0\} \ , \\ A_0 &\triangleq A_1 \cup A_2 \ . \end{aligned}$$

It is easy to see that $A_1 \cap A_2 = \emptyset$. Then it is also easy to see that $f_2(y) \oplus f_3(z)$ is balanced for H if and only if

$$|\{y \mid f_2(y) = 0, y \in A_0\}| = |\{y \mid f_2(y) = 1, y \in A_0\}| \ . \tag{2}$$

Further, $f_2(y)$ includes a term $y_1 y_2 \ldots y_k$ if and only if

$$|\{y \mid f_2(y) = 1\}| = odd \ . \tag{3}$$

Now write the truth table of $f_2(y)$ such that eq.(2) and eq.(3) are satisfied. Then a desirable $f_2(y)$ is obtained. Hence, as in the proof of Corollary 3.1,

$$n = l + k + 1 = l + 2^l - 1$$
$$\deg(g) = n - l - 1 \approx n - \log_2 n.$$

□

References

[1] R. Forré. The strict avalanche criterion : spectral properties of Boolean functions and an extend definition. In *Advances in Cryptology — CRYPTO '88 Proceedings, Lecture Notes in Computer Science* 403, pages 450–468. Springer-Verlag, 1990.

[2] T. Jakobsen and L.R. Knudsen. The interpolation attack on block ciphers. In *Preproc. of Fast Software Encryption*, pages 28–40. January, 1997.

[3] K. Kurosawa and T. Satoh. Design of SAC/PC(l) of order k Boolean functions and three other cryptographic criteria. In *Advances in Cryptology — EUROCRYPT '97 Proceedings, Lecture Notes in Computer Science* 1233, pages 434–449. Springer-Verlag, 1997.

[4] W. Meier and O. Staffelbach. Nonlinearity criteria for cryptographic functions. In *Advances in Cryptology — EUROCRYPT '89 Proceedings, Lecture Notes in Computer Science* 434, pages 549–562. Springer-Verlag, 1990.

[5] B. Preneel, W. Van Leekwijck, L. Van Linden, R. Govaerts, and J. Vandewalle. Propagation characteristics of Boolean functions. In *Advances in Cryptology — EUROCRYPT '90 Proceedings, Lecture Notes in Computer Science* 473, pages 161–173. Springer-Verlag, 1991.

[6] O. S. Rothaus. On bent functions. *Journal of Combinatorial Theory (A)*, 20:300–305, 1976.

[7] J. Seberry and X.M. Zhang. Highly nonlinear 0-1 balanced Boolean functions satisfying strict avalanche criterion. In *Advances in Cryptology — AUSCRYPT '92 Proceedings, Lecture Notes in Computer Science* 718. Springer-Verlag, 1993.

[8] A. F. Webster and S. E. Tavares. On the design of S-boxes. In *Advances in Cryptology — CRYPTO '85 Proceedings, Lecture Notes in Computer Science* 218, pages 523–534. Springer-Verlag, 1986.