ハッシュ関数 *Luffa* の非線形置換における高階差分特性

山田 剛 † 渡辺 大 ‡ 秦野 康生 ‡ 金子 敏信 †

† 東京理科大学大学院 理工学研究科 電気工学専攻
278-8510 千葉県野田市山崎 2641

j7308677@ed.noda.tus.ac.jp kaneko@ee.noda.tus.ac.jp
‡(株) 日立製作所システム開発研究所
244-0817 神奈川県横浜市戸塚区吉田町 292

dai.watanabe.td@hitachi.com yasuo.hatano.bn@hitachi.com

あらまし *Luffa* は日立製作所によって提案されたハッシュ関数であり、NIST による SHA-3 Competition の Second Round Candidates のうちの一つである。*Luffa* の構成要素である非線形置換 Q_j の、代数的攻撃に対する安全性は現在詳細には検討されていない。本稿では、非線形置換 Q_j における高階差分特性を計算機実験により観測し、提案者らの見積もりよりも代数次数の増加傾向が小さいことを示す。また、この代数次数の増加傾向が小さい理由として、S-box のブール代数式表現の高次項が一致していることが一因となっている可能性を指摘する。

A Higher Order Differential Property of the Non-Linear Permutation in Hash Function *Luffa*

Tsuyoshi Yamada† Dai Watanabe‡ Yasuo Hatano‡ Toshinobu Kaneko†

†Department of Electrical Engineering,Faculty of Science and Technology,
Tokyo University of Science
2641 Yamazaki,Noda-shi,Chiba-ken,278-8510,Japan

j7308677@ed.noda.tus.ac.jp kaneko@ee.noda.tus.ac.jp
‡Systems Development Laboratory,Hitachi,Ltd.
292 Yoshida-cho,Totsuka-ku,Yokohama-shi,Kanagawa-ken,244-0817,Japan

dai.watanabe.td@hitachi.com yasuo.hatano.bn@hitachi.com

Abstract *Luffa* is a hash function proposed by Hitachi, Ltd., which is one of the second round candidates of SHA-3 Competition by NIST. The security against an algebraic attack of the non-linear permutation Q_j, which is a component of *Luffa*, has not been investigated in detail. In this paper, we observe a higher order differential property of the non-linear permutation Q_j by experiments on computing and show the degree increases more slowly than the estimate of proposers. In addition, we point out that a reason of this slow increase of the algebraic degrees is caused by the terms of highest degree of some polynomial expressions of the S-box being identical.

1 はじめに

Luffa は日立製作所によって提案されたハッシュ関数であり、その連鎖法はスポンジ関数の変形である。また NIST によって公募されている次世代ハッシュ関数アルゴリズム (SHA-3) の Second Round Candidates のうちの一つである。

提案者らは、*Luffa* の非線形置換 Q_j に対する有効な代数的攻撃は見つかっていないとしているが、詳細にこの安全性は検討されていない。そこで本稿では、非線形置換における高階差分特性を計算機実験により調査し、提案者らの見積もりよりも代数次数の増加傾向が小さいことを示す。また、この代数次数の増加傾向が小さい理由として、S-box のブール代数式表現の高次項が一致していることが一因となっている可能性を指摘する。

本稿では、2 節で *Luffa* の構造の一部を紹介し、3 節では高階差分の定義と性質を説明する。4 節で今回行った実験方法とその結果を述べ、5 節でこの結果について考察する。

2 *Luffa* の構造

Luffa の連鎖法はスポンジ関数の変形である。図 1 に連鎖法の基本的な構造を示す。ハッシュ関数の連鎖法はラウンド関数 C' と、終了部 C'' によって構成されている。

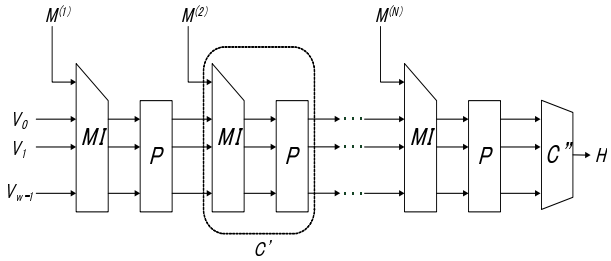

図 1: *Luffa* の連鎖法

以下、*Luffa* の構造について、本稿に関連する部分を紹介する (詳細については [1] 参照)。

2.1 ラウンド関数 C'

ラウンド関数 C' はメッセージ入力関数 MI と非線形関数 P によって構成されている。この非線形関数 P は 256 ビット入出力の非線形置換 Q_j に分けられる (図 2 参照)。

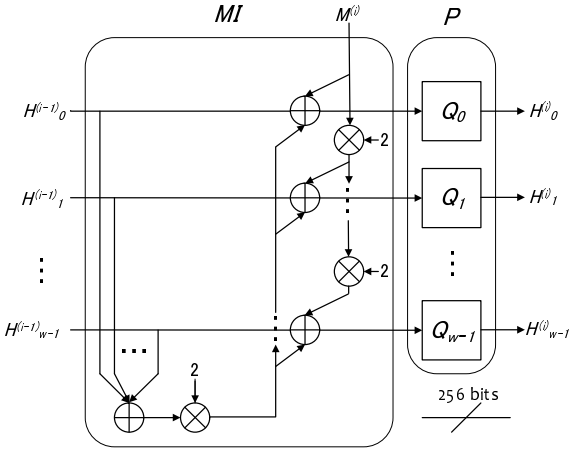

図 2: ラウンド関数 C'

2.2 非線形置換 Q_j

この節では、非線形置換 Q_j の詳細を示す。非線形置換 Q_j は、入力時の tweak と、ステップ関数 `Step` の繰り返しで構成されている。ステップ関数は 8 回適用され、tweak は一つの置換につき 1 回のみ適用される。

ステップ関数の開始時に、256 ビットデータは $a_k^{(r)}(0 \le k < 8)$ で示される 8 つの 32 ビットワードに分割される。ステップ関数は 3 つの関数 `SubCrumb`、`MixWord`、`AddConstant` で構成されている。図 3 にステップ関数の概要を示す。

2.2.1 `SubCrumb`

`SubCrumb` は a_0, a_1, a_2, a_3(または a_4, a_5, a_6, a_7) の l ビット目を、4 ビット S-Box において置換するものである。この 4 ビット S-box のブール代数次数は最大で 3 次である。以下に、入力ビットを x_0, x_1, x_2, x_3、出力ビットを y_0, y_1, y_2, y_3 とした場合の S-box のブール代数式表現を示す。

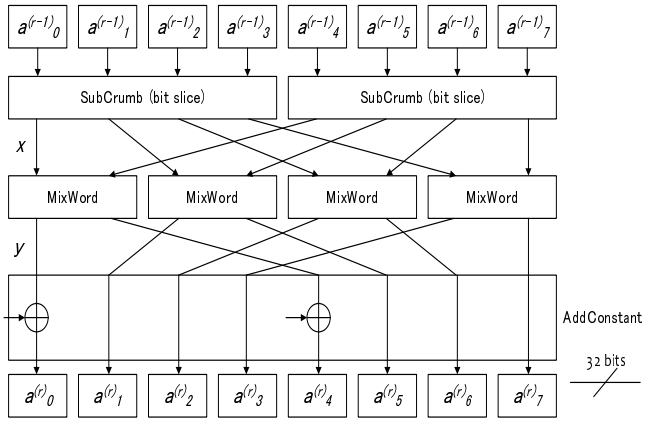

図 3: ステップ関数 Step

$$
\begin{aligned}
y_0 &= 1 + x_2 + x_0x_1 + x_1x_3 + x_2x_3 + x_0x_1x_3 \\
y_1 &= 1 + x_0 + x_2 + x_3 + x_0x_1 + x_0x_2 + x_1x_3 \\
&\quad + x_2x_3 + x_0x_1x_3 \\
y_2 &= 1 + x_1 + x_1x_3 + x_2x_3 + x_0x_1x_3 \\
y_3 &= x_0 + x_1 + x_2 + x_0x_1 + x_1x_2 + x_1x_3 \\
&\quad + x_0x_1x_2
\end{aligned}
$$

SubCrumb の出力を x_0, x_1, x_2, x_3(または x_4, x_5, x_6, x_7) とすれば、SubCrumb の置換は次式で与えられる。

$$
\begin{aligned}
&x_{3,l} \parallel x_{2,l} \parallel x_{1,l} \parallel x_{0,l} = S[a_{3,l} \parallel a_{2,l} \parallel a_{1,l} \parallel a_{0,l}] \\
&x_{7,l} \parallel x_{6,l} \parallel x_{5,l} \parallel x_{4,l} = S[a_{7,l} \parallel a_{6,l} \parallel a_{5,l} \parallel a_{4,l}] \\
&0 \le l < 32
\end{aligned}
$$

図 4 に、SubCrumb の概要を示す。

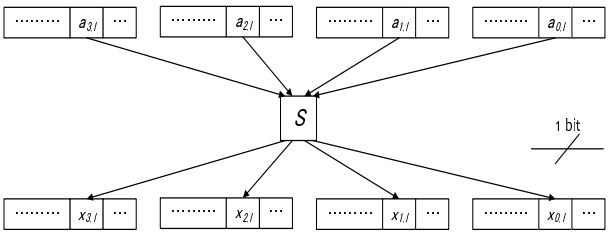

図 4: SubCrumb

2.2.2 MixWord

MixWord は、2 ワード入出力の線形置換である。図 5 に MixWord の概要を示す。

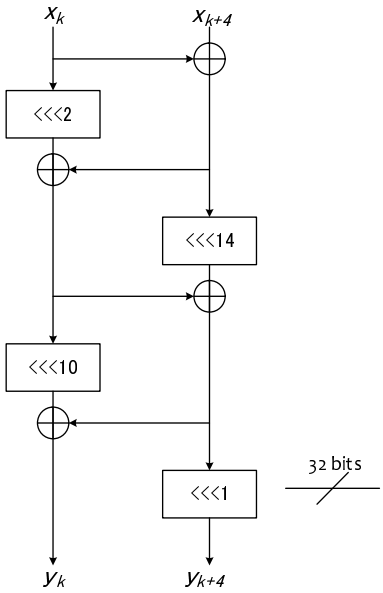

図 5: MixWord

3 高階差分

高階差分は、1994 年に Lai によって暗号解析への応用が示唆された [4]。またこの性質を利用した暗号の攻撃を高階差分攻撃と呼び、これは共通鍵暗号への汎用的な攻撃方法の一つである [3]。以下、[3] に従って高階差分の定義と性質を紹介する。

定義. $E(X;K)$ を、入力 $X \in GF(2)^n$ と鍵 $K \in GF(2)^s$ から、$Y \in GF(2)^m$ を出力する暗号化関数とする。

$\{a_1, a_2, \cdots, a_i\}$ を $GF(2)^n$ 上で *1* 次独立な i 個のベクトル、これらによって張られる i 次元部分空間を $V^{(i)}$ とする。関数 $E(X;K)$ の $V^{(i)}$ に関する i 階差分 $\Delta^{(i)}E(X;K)$ は、次式で定義される。

$$
\Delta^{(i)}_{V^{(i)}} E(X;K) = \bigoplus_{A \in V^{(i)}} E(X \oplus A; K)
$$

左辺の下付添え字 $V^{(i)}$ は、誤解のおそれが無い限り省略する。なお本稿では、$a_1, a_2, ..., a_i$ を変数ビットと呼ぶ。

高階差分は、次の性質を持つ。

表 1: 32 階差分特性の調査結果

Step	$\Delta^{(32)}a_0$	$\Delta^{(32)}a_1$	$\Delta^{(32)}a_2$	$\Delta^{(32)}a_3$	$\Delta^{(32)}a_4$	$\Delta^{(32)}a_5$	$\Delta^{(32)}a_6$	$\Delta^{(32)}a_7$
…	…	…	…	…	…	…	…	…
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	∗	∗	∗	-	#	#	#	-
6	-	-	-	-	-	-	-	-
…	…	…	…	…	…	…	…	…

'0' は 32 階差分値が常に 0 となることを、'-' は 32 階差分値が不定になることを示す。
' ∗'、'# ' は初期値に依存した値となるが、' ∗'(または'# ') 同士は 32 階差分値が等しくなることを示す。
また、パターン 1 の a_2 もしくは a_6 に変数を与えた場合、この現象が見受けられない場合もあった (a_2 に関しては 10 回中 6 回、a_6 に関しては 10 回中 2 回)。

性質. 関数 $E(X;K)$ の X に関するブール代数次数が N ならば、X と K に依存せず、次式が成立する。

$$\Delta^{(N+1)}E(X;K) = 0$$

この性質を用いることで、高階差分値が 0 となる値 N から代数次数を実験的に求めることができる。

4 計算機実験による高階差分特性

提案者らは、*Luffa* に対する有効な代数的攻撃は見つかっていないとしているが、詳細にこの安全性は検討されていない。また、非線形置換の中の非線形要素は S-box のみであるので、単純に見積もれば r ステップ目の代数次数は 3^r 次となる。

この節では、*Luffa* の非線形置換における高階差分特性を、計算機実験により観測した結果を述べる。

4.1 32 階差分特性の調査

4.1.1 変数ビット位置

32 階差分特性を観測するにあたり、変数ビットを与えるビット位置について考慮する必要がある。そこで S-box への入力を考慮し、以下の 4 つの変数ビット位置パターンを定める。

1. ワード $a_k(0 \leq k < 8)$ の 32 ビット
2. ワード $a_k, a_{k+4}(0 \leq k < 4)$ の各下位 16 ビット
3. ワード $a_k, a_{k+1}, a_{k+2}, a_{k+3}(k = 0, 4)$ の各下位 8 ビット
4. ワード $a_0, a_1, ..., a_7$ の各下位 4 ビット

ここでパターン 1 及び 2 では、1 ステップ目の 32 個の S-box への入力のうち 1 ビットのみを変数としている。S-box のブール代数式表現の最大次数は 3 次であるが、変数は 1 つのみであるので、出力は 1 次式とみなすことができる。これにより、1 ステップ目での `SubCrumb` による代数次数への影響を無視することができる。

またパターン 3 及び 4 は、8 個の S-box の入力のうち全てのビットを変数としている。S-box は全単射の置換であるので、入力を全通り取れば、出力も全通り出現する。これにより、1 ステップ目での `SubCrumb` による代数次数への影響を無視することができる。

4.1.2 実験結果

ここでは、前節の 4 パターンにおいて各ステップ各ワードの 32 階差分値を観測した結果を述べる。ここで、各パターンにおけるワード $a_0, ..., a_7$ の初期値はランダムに 10 個ずつ選び実験を行った。この実験結果を表 1 に示す。

結果、以下の 2 つの性質が得られた。

1. 全てのパターンにおいて 4 ステップ目までの 32 階差分値が、初期値の値に関わらず 0 となった。

2. 全てのパターンにおいて、5 ステップ目のワード a_0, a_1, a_2 及び a_4, a_5, a_6 における 32 階差分値が同じ値をとる現象が見られた。
言いかえれば、$(\Delta^{(32)}a_0, \Delta^{(32)}a_1, ..., \Delta^{(32)}a_7) = (\alpha, \alpha, \alpha, \beta, \gamma, \gamma, \gamma, \delta)$ となる現象である。($\alpha, \beta, \gamma, \delta$ は 32 ビットの値)
しかし、パターン 1 のワード a_2 もしくは a_6 に変数を与えた場合、初期値によってはこの現象が見受けられない場合もあった (a_2 に関しては 10 回中 6 回、a_6 に関しては 10 回中 2 回)。

4.2 ステップ毎の次数評価

この節では 1 階から 32 階差分特性を観測し、より詳細に次数の増加傾向を調査する。ここでは変数を与えるビットをワード a_0 の下位 n ビットとし n 階差分を各ステップ各ワードを観測する。ワード a_0 の初期値はランダムに 3 個ずつ選び実験を行った。

S-box の最大次数は 3 次であり、今回の実験では変数ビットの選び方によって 1 ステップ目での SubCrumb による代数次数への影響を無視しているので、r ステップ目の代数次数は 3^{r-1} 次と見積もることが出来る。

表 2 にステップ数と形式的な次数、そして高階差分特性から得られた実際の次数を示す。

表 2: ステップ毎の次数評価

	次数	
Step	形式評価	実験結果
1	1	1
2	3	1
3	9	2
4	27	12
5	81	-
…	…	…

5 考察

5.1 次数の増加傾向について

非線形置換の中の非線形要素は S-box のみであるので、単純に見積もれば r ステップ目の代数次数は 3^r 次となる。

今回の実験では、変数ビットの選び方によって 1 ステップ目での SubCrumb による代数次数への影響を無視している。したがって次数が増加し始めるのは 2 ステップ以降であるが、この増加傾向は見積もりより小さいことが表 1 より分かる。この理由を以下に考察する。

MixWord の入力は $x_k, x_{k+4}(0 \leq k < 3)$ となっている。これは MixWord が、S-box の出力の同じビット位置から成るワード同士を攪拌していることを示す。このことから MixWord の出力において、後述する S-box のブール代数式表現における 3 次項の性質は保たれる。同様に AddConstant は定数加算であるので、S-box の出力のブール代数式表現における高次項に影響を与えない。

これらの性質から、代数次数の増加は S-box の性質を考察すれば十分である。

S-box のブール代数式表現を見ると、出力 y_k の 3 次項に x_0x_1 が共通していることが分かる。ここで、3 次項を $x_0x_1x_k$、2 次以下の項を ξ_k とおくと、

$$\begin{aligned} y_k \cdot y_{k'} &= (\xi_k + x_0x_1x_k)(\xi_{k'} + x_0x_1x_{k'}) \\ &= \xi_k\xi_{k'} + (\xi_k x_{k'} + \xi_{k'}x_k + x_kx_{k'})x_0x_1 \end{aligned}$$

となることが分かる。したがって、$\deg\ y_k \cdot y_{k'} < \deg\ y_k + \deg\ y_{k'}$ となり、代数次数が 3 倍に増加しないことが分かる。

しかし、表 2 の 2、3 ステップ目における次数増加の傾向は明らかにこれよりも小さい。可能性として高次項が打ち消しあっていることや、構造上の関係で高階差分値が 0 となっていることが考えられるが、これに関しては更なる検証が必要である。

5.2　5 ステップ目における 32 階差分値

この節では、4.1 節の性質 2、つまり 5 ステップ目において $(\Delta^{(32)}a_0, \Delta^{(32)}a_1, ..., \Delta^{(32)}a_7) = (\alpha, \alpha, \alpha, \beta, \gamma, \gamma, \gamma, \delta)$ となる現象について考察する。

この現象が見られる一因として、S-box の出力 y_0, y_1, y_2 の 3 次項 $x_0x_1x_3$ が共通していることが考えられる。この根拠として、`MixWord`、`AddConstant` の変換後、すなわち次のステップにおける S-box の入力においても 3 次項 $x_0x_1x_3$ が一致しているという性質が成り立つことが挙げられる。この性質により、5 ステップ目の 32 階差分をとった多項式が y_0, y_1, y_2 で一致しているのではないかと考えられる。

またワード a_2 及び a_6 に変数を与えた場合、この現象が見受けられない場合がある。a_2 及び a_6 は 1 ステップ目の S-box の入力 x_2 に対応していることから、5 ステップ目において 32 階差分値が一致する現象は 1 ステップ目の 3 次項 $x_0x_1x_3$ が関連していることが考えられる。しかしこの現象が起こる頻度や、初期値の影響などを考察するために、より詳細な検討が必要である。

6　結論

本稿では、ハッシュ関数 *Luffa* の非線形置換における高階差分特性を観測し、見積もりよりも代数次数の増加傾向が小さいことを示した。また、この代数次数の増加傾向が小さい理由として、S-box のブール代数式表現の高次項が一致していることが一因となっている可能性を指摘した。しかし、我々が指摘した原因だけでは、実験結果と形式評価の違いは説明できない。この問題に関しては更なる検討が必要である。

また、5 ステップ目において $(\Delta^{(32)}a_0, \Delta^{(32)}a_1, ..., \Delta^{(32)}a_7) = (\alpha, \alpha, \alpha, \beta, \gamma, \gamma, \gamma, \delta)$ となる現象を発見し、S-box のブール代数式表現における高次の共通項が影響を与えていることを考察した。しかしワード a_2 及び a_6 に変数を与えた場合、この現象が見られないことがあり、これに関してはより詳細な検討が必要である。

参考文献

[1] C. D. Cannière, H. Sato and D. Watanabe "Hash Function *Luffa*, Specification," NIST SHA-3 competition, 2008.

[2] C. D. Cannière, H. Sato and D. Watanabe "Hash Function *Luffa*, Supporting Document," NIST SHA-3 competition, 2008.

[3] L. Knudsen, "Truncated and Higher Order Differentials", FSE2nd International Workshop, LNCS.1008.

[4] X. Lai, "Higher Order Derivatives and Differential Cryptanalysis", Communications and Cryptography.