Construction and Security of a Non-Algebraic Tiny and Extensible Hash Function

Hirotake YAGUCHI (Mie University, JAPAN)

1. β-transformation on [1,2)

$\beta > 1$

$M_\beta : [1,2) \to [1,2)$, (We call the modified beta transformation)

$$M_\beta(t) = \beta t \mod [1,2) = \beta t - \lfloor \beta t \rfloor + 1$$

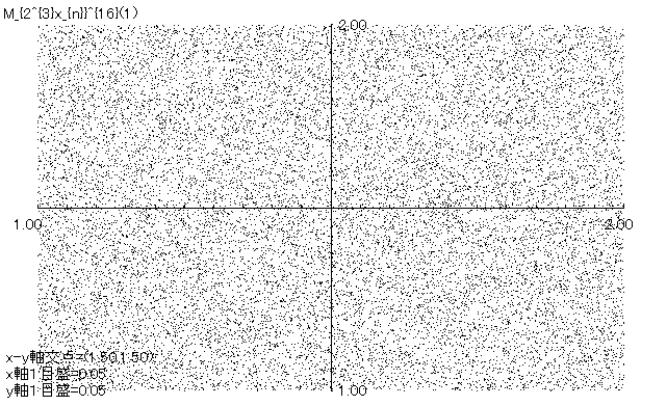
$$x_n = 1 + \frac{n}{20000}, \quad n = 0,1,\cdots,19999$$

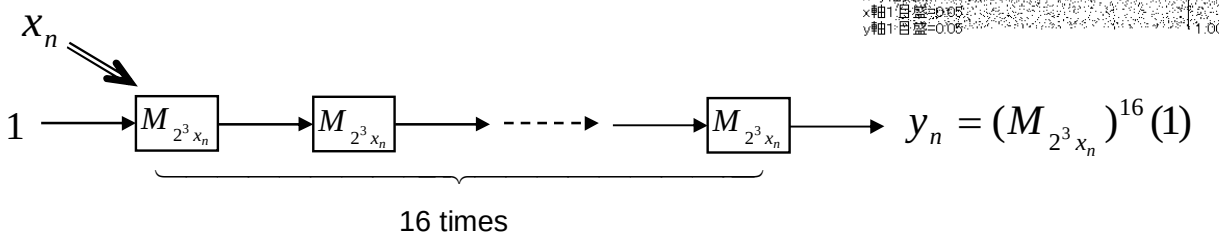
$$\beta = 2^3 x_n$$

$$\Rightarrow$$

The graph of $y_n = (M_{2^3 x_n})^{16}(1)$

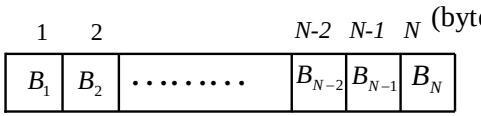
(double precision computation)





2. Diagram of a tiny hash function MB32hash

Input stream of bytes B =



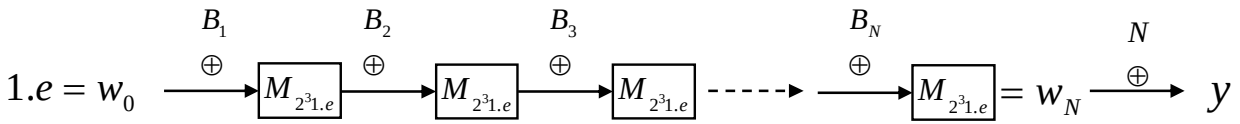
out put → ζ : random number

MB32hash

$$1.\ e \equiv 1 + \frac{1}{10}\left(1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \cdots\right) = 1.27181\cdots \quad (e = 2.7181\cdots : \text{Euler's constant})$$

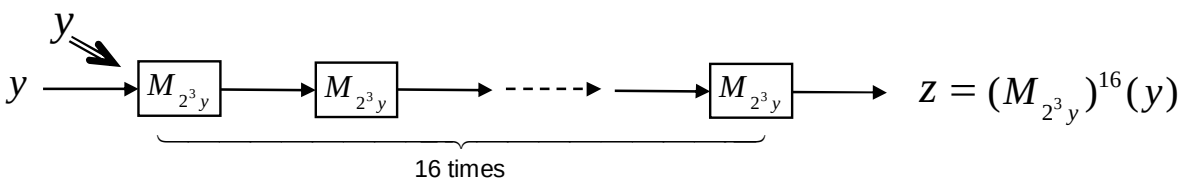
MB32hash

Stage 1. Compression of B



$$w_k = M_{2^3 1.e}(w_{k-1} \oplus B_k)$$

Stage 2. Scrambling of y

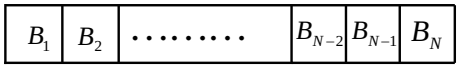


→ Taking out a 32-bit hash value ζ from z .

3. Precise algorithm of MB32hash

(We only state the algorithm here, and do not care how it will be implemented. The implementation will be stated in the next sheet.)

Stage 1. Compression of B

$B =$ 

$$w_0 = 1.e$$

$$w_k = M_{2^3 1.e}(w_{k-1} \oplus B_k)$$
$$= (2^3 1.e)(w_{k-1} \oplus B_k) - \lfloor (2^3 1.e)(w_{k-1} \oplus B_k) \rfloor + 1,$$
$$k = 1, \cdots, N \ (< 2^{31}),$$

$$w_{k-1} \oplus B_k = 1.b_1 \cdots b_7 \underbrace{(b_8 \oplus c_1) \cdots (b_{15} \oplus c_8)} b_{16} b_{17} \cdots\cdots$$

for $w_{k-1} = 1.b_1 \cdots b_7 b_8 \cdots b_{15} b_{16} b_{17} \cdots\cdots$ (binary),
$B_k = c_1 c_2 \cdots c_8$ (binary)

$$y \equiv w_N \oplus N \quad \text{(embedding the length of } B\text{)}$$
$$= 1.\ (b_1 \oplus \nu_1)(b_2 \oplus \nu_2) \cdots (b_{31} \oplus \nu_{31})\ b_{32} b_{33} \cdots\cdots \text{ for } N = \nu_1 \nu_2 \cdots \nu_{31} \text{ (binary)}$$

Stage 2. Scrambling of y

$$z = (M_{2^3 y})^{16}(y)$$

Taking out a hash value ζ from z

$$\zeta = \breve{b}_{16} \breve{b}_{17} \cdots \breve{b}_{47} \quad \text{for} \quad z = 1.\breve{b}_5 \breve{b}_6 \cdots \breve{b}_{15} \underbrace{\breve{b}_{16} \cdots \breve{b}_{47}}_{\zeta} \breve{b}_{48} \cdots\cdots$$

i.e., $\zeta = \left\lfloor 2^{32} \left(2^{11} z - \lfloor 2^{11} z \rfloor \right) \right\rfloor$

4. Implementation of MB32hash

We implement MB32hash on the following <u>32-bit number system</u> :

1) Every number in [1,2) is represented by using 32 bits such as $1.b_1b_2\cdots b_{31}$.
2) Result of multiplication of two numbers $x=1.x_1x_2\cdots x_{31}$ and $t=1.t_1t_2\cdots t_{31}$ is represented by 64-bit number $\breve{b}_0\breve{b}_1.\breve{b}_2\breve{b}_3\cdots\breve{b}_{63}$ with $\breve{b}_0\breve{b}_1=01$ or $\breve{b}_0=1$.

Under the 32-bit system above

$$\begin{aligned} M_{2^3x}(t) &= (2^3x)t \mod [1,2) = 2^3(xt) \mod [1,2) \\ &= 2^3\times\breve{b}_0\breve{b}_1.\breve{b}_2\breve{b}_3\breve{b}_4\breve{b}_5\breve{b}_6\cdots\breve{b}_{63} \mod [1,2) \\ &= \breve{b}_0\breve{b}_1\breve{b}_2\breve{b}_3\breve{b}_4.\breve{b}_5\breve{b}_6\cdots\breve{b}_{63} \mod [1,2) \\ &= 1.\breve{b}_5\breve{b}_6\cdots\breve{b}_{35}\cdots\breve{b}_{63} \xrightarrow{to\ 32-bit} 1.\breve{b}_5\breve{b}_6\cdots\breve{b}_{35}. \end{aligned}$$

We <u>identify</u>

$1.b_1b_2\cdots b_{31}$ with $1b_1b_2\cdots b_{31}$ (32-bit integer), and

$\breve{b}_0\breve{b}_1.\breve{b}_2\breve{b}_3\cdots\breve{b}_{63}$ with $\breve{b}_0\breve{b}_1\breve{b}_2\breve{b}_3\cdots\breve{b}_{63}$ (64-bit integer).

Then the implementaion of $M_{2^3x}(t)$ is realized by 64-bit integer-operations:

$$\begin{matrix}1x_1x_2\cdots x_{31}\\ 1t_1t_2\cdots t_{31}\end{matrix} \xrightarrow{multiplication} \breve{b}_0\breve{b}_1\breve{b}_2\breve{b}_3\breve{b}_4\breve{b}_5\cdots\breve{b}_{35}\cdots\breve{b}_{63} \xrightarrow{shift\ 4} \breve{b}_4\breve{b}_5\cdots\breve{b}_{35}\cdots\breve{b}_{63} \xrightarrow{OR\ with\ \breve{b}_4} 1\breve{b}_5\cdots\breve{b}_{35}\cdots\breve{b}_{63} \xrightarrow{cut} 1\breve{b}_5\cdots\breve{b}_{35}.$$

5. Scrambling is a random number generator

The scrambling of MB32hash is essentially a non-recursive pseudorandom number generator.

$$y_n = 1.e \oplus n \ = 1.\ (b_1 \oplus \nu_1)(b_2 \oplus \nu_2)\cdots(b_{31} \oplus \nu_{31})\ b_{32}b_{33}\cdots\cdots, \quad n = 0,1,\cdots,2^{31}-1$$

$$z_n = (M_{2^3 y_n})^{16}(y_n) \quad \text{(scrambling of } y_n\text{)}$$

$$\zeta_n = \left\lfloor 2^{32}\left(2^{11} z_n - \left\lfloor 2^{11} z_n \right\rfloor\right)\right\rfloor \quad \text{(taking out 32-bit hash value)}$$

Implementation : Our 32-bit number system

The (hash) values $\{\zeta_n\}_{n=0,1,\cdots}$ is a sequence of random numbers, and passes the NIST statistical test suite sts-2.1.1.

[RESULT of NIST's test] (We repeated NIST's suite 10 times, and took the average for each test.)

MB32 $\{\zeta_n\}$

```
P-VALUE     AvMax= 0.737001 at NonOverlappingTemplate [i=83]
P-VALUE     AvMin= 0.285480 at RandomExcursionsVariant [i=181]
PROPORTION  AvMax = 0.992503 at RandomExcursionsVariant [i=170]
PROPORTION  AvMin = 0.986800 at FFT [i=7]
```

Mersenne Twister

```
P-VALUE     AvMax= 0.759461 at NonOverlappingTemplate [i=49]
P-VALUE     AvMin= 0.282535 at NonOverlappingTemplate [i=92]
PROPORTION  AvMax = 0.992500 at NonOverlappingTemplate [i=133]
PROPORTION  AvMin = 0.986400 at FFT [i=7]
```

6. Security of the compression from a point of view of algorithm

(We only consider the security of algorithm here.)

$B =$ | B_1 | B_2 | ..... | B_{N-2} | B_{N-1} | B_N | (input stream of bytes)

$w_0 = 1.e, \quad w_k = M_{2^3 1.e}(w_{k-1} \oplus B_k) = (2^3 1.e)(w_{k-1} \oplus B_k) - \lfloor (2^3 1.e)(w_{k-1} \oplus B_k) \rfloor + 1$ (compression)

We consider *when* $w_N = \hat{w}_{\hat{N}}$ for B and $B\hat{}$.

Put $\gamma \equiv 2^3 1.e$. Then for B

$$\begin{aligned} w_k &= \gamma(w_{k-1} \oplus B_k) - \lfloor \gamma(w_{k-1} \oplus B_k) \rfloor + 1 \\ &\equiv \gamma(w_{k-1} \oplus B_k) + \tilde{p}_k \quad (\ \tilde{p}_k \text{ is in } \{-9,-10,\cdots,-19\}\ ) \\ &= \gamma(w_{k-1} + t_k) + \tilde{p}_k \quad (\ t_k \text{ is of the form } \pm 0.0000000 b_8 b_9 \cdots b_{15} 000\cdots\cdots) \end{aligned}$$

$$\therefore w_N = \gamma^N(1.e + t_1) + \gamma^{N-1}(\tilde{p}_1 + t_2) + \cdots + \gamma(\tilde{p}_{N-1} + t_N) + \tilde{p}_N$$

$w_N = \hat{w}_{\hat{N}}$ requires that γ should be included in the solutions of the following algebraic equations of $\tilde{\gamma}$:

$$\begin{aligned} &\tilde{\gamma}^N(1.e + t_1) + \tilde{\gamma}^{N-1}(\tilde{p}_1 + t_2) + \cdots + \tilde{\gamma}(\tilde{p}_{N-1} + t_N) + \tilde{p}_N \\ &= \tilde{\gamma}^{\hat{N}}(1.e + \hat{t}_1) + \tilde{\gamma}^{\hat{N}-1}(\tilde{q}_1 + \hat{t}_2) + \cdots + \tilde{\gamma}(\tilde{q}_{\hat{N}-1} + \hat{t}_{\hat{N}}) + \tilde{q}_{\hat{N}} \end{aligned}$$

(t_i and $\hat{t}_j$ are of the form $\pm 0.0000000 b_8 b_9 \cdots b_{15} 000\cdots$,
$\tilde{p}_i$ and $\tilde{q}_j$ are in $\{-9,-10,\cdots,-19\}$.
($\rightarrow$ totally, $(2^8 \times 2)^{N+\hat{N}} \times 11^{N+\hat{N}}$ equations))

However if we observe the form of $\gamma = 2^3\left\{1 + \frac{1}{10}\left(1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \cdots\right)\right\}$, we know that the requirement above is not satisfied except that $B = B\hat{}$.

7. Security of the scrambling from a point of view of algorithm

(We only consider the security of algorithm here.)

$z = (M_{2^3 y})^{16}(y)$ (scrambling of y) $\qquad \zeta = \left\lfloor 2^{32}\left(2^{11} z - \left\lfloor 2^{11} z \right\rfloor\right)\right\rfloor$ (taking out 32-bit hash value)

We consider *when* $\zeta = \hat{\zeta}$ for y and $\hat{y}$.

$$u_k = M_{2^3 y}(u_{k-1}) = (2^3 y)u_{k-1} - \left\lfloor (2^3 y)u_{k-1} \right\rfloor + 1$$

$$\equiv (2^3 y)u_{k-1} + \tilde{p}_k \quad (\ \tilde{p}_k \text{ is in } \{-7,-8,\cdots,-30\}\ )$$

$$\therefore\ z(y) = u_{16} = (2^3 y)^{16} y + (2^3 y)^{15}\tilde{p}_1 + \cdots + (2^3 y)\tilde{p}_{15} + \tilde{p}_{16}$$

$$\hat{z}(\hat{y}) = \hat{u}_{16} = (2^3 \hat{y})^{16}\hat{y} + (2^3 \hat{y})^{15}\tilde{q}_1 + \cdots + (2^3 \hat{y})\tilde{q}_{15} + \tilde{q}_{16}$$

$\zeta = \hat{\zeta}$ requires $\quad \left| \left(z - 2^{-11}\left\lfloor 2^{11} z \right\rfloor\right) - \left(\hat{z} - 2^{-11}\left\lfloor 2^{11}\hat{z} \right\rfloor\right) \right| < 2^{-43}.$ (#)

We have to solve the algebraic inequality (#) of y and $\hat{y}$ if we want to find y and $\hat{y}$ with $|y - \hat{y}| >\approx 2^{-91}$. (we see that $z'(y) >\approx 2^{48}$ in the sheet 9)

But to solve (#) is very difficult

(because we have no general algorithm of solving algebraic equations whose degree is higher than 4).

- Even if (#) is solved, it is difficult to find *B* and *B*^ which yield the compressed-values y and $\hat{y}$.
- Numerical solution of (#) is treated in the sheet 9.

8. Security of the compression from a point of view of implementation

Under our 32-bit number system, If $w_{k-1} \oplus B_k$ is taken with the last 8 bits of w_{k-1} such as

$$1.b_1 \cdots b_{23}(b_{24} \oplus c_1) \cdots (b_{31} \oplus c_8),$$

(MB32hash)

$$1.b_1 \cdots b_7 (b_8 \oplus c_1) \cdots (b_{15} \oplus c_8) b_{16} b_{17} \cdots b_{31}$$

not touched by $\oplus$

then we can have B and $B\hat{}$ satisfying $W_2 = \hat{W}_2$.

[Example]

B=0x0000, $B\hat{}$=0x0136 (hexadecimal)

$x = w_0 =$ 0xa2cb4411 $(1.e)$

compression of B

$$w_0 = a2cb4411 \xrightarrow{\oplus B_1} a2cb44(1\oplus 0)(1\oplus 0) = a2cb4411 \xrightarrow{mul.\ x} 6785e38a890f0921$$

$$\xrightarrow{shift\ 4} 785e38a890f0921 \xrightarrow{OR} f85e38\boxed{a890f0921} \xrightarrow{cut} f85e38\boxed{a8} = w_1$$

$$\xrightarrow{\oplus B_2} f85e38(a\oplus 0)(8\oplus 0) = f85e38a8 \xrightarrow{mul.\ x} \cdots\cdots \xrightarrow{cut} df0d49ac = w_2$$

compression of $B\hat{}$

$$w_0 = a2cb4411 \xrightarrow{\oplus \hat{B}_1} a2cb44(1\oplus 0)(1\oplus 1) = a2cb4410 \xrightarrow{mul.\ x} 6785e389e643c510$$

$$\xrightarrow{shift\ 4} 785e389e643c510 \xrightarrow{OR} f85e38\boxed{9e643c510} \xrightarrow{cut} f85e38\boxed{9e} = \hat{w}_1$$

$$\xrightarrow{\oplus \hat{B}_2} f85e38(9\oplus 3)(e\oplus 6) = f85e38a8(\text{the same as } B) \xrightarrow{mul.\ x} \cdots$$

$$\xrightarrow{cut} df0d49ac = \hat{w}_2\ (= w_2)$$

We avoid this problem by reserving bits $b_{16}b_{17}\cdots b_{31}$ in w_{k-1} which are not touched by $\oplus$.

9. Security of the scrambling from a point of view of implementation

$z = (M_{2^3 y})^{16}(y)$ (scrambling of y) $\quad \zeta = \left\lfloor 2^{32}\left(2^{11} z - \left\lfloor 2^{11} z \right\rfloor\right)\right\rfloor$ (taking out 32-bit hash value)

$$u_0 = y$$

$$u_k = M_{2^3 y}(u_{k-1}) = (2^3 y)u_{k-1} - \left\lfloor (2^3 y)u_{k-1} \right\rfloor + 1$$

$$\frac{d}{dy}u_k(y) = 2^3 u_{k-1}(y) + (2^3 y)\frac{d}{dy}u_{k-1}(y) \quad \text{a. e. } y$$

$$\therefore\ z'(y) = u'_{16}(y) = \sum_{k=0}^{15} 2^3 (2^3 y)^k u_{15-k}(y) + 2^3 (2^3 y)^{16} y \quad \text{a.e. } y$$

Since $1 \le y, u_k < 2$, it holds $z'(y) \ge \dfrac{2^{48}-1}{1-2^{-3}}$.

Then we know that

a variation η from y to $y+\eta$ with $\lvert\eta\rvert > 2^{-91}$ influences the hash value ζ.

This implies (#) $\left\lvert \left(z - 2^{-11}\left\lfloor 2^{11} z \right\rfloor\right) - \left(\hat{z} - 2^{-11}\left\lfloor 2^{11} \hat{z} \right\rfloor\right) \right\rvert < 2^{-43}$

- numerical solution of (#) is useless when it is rounded to 32-bit,
- rounding errors caused by truncation to 32-bit

  $$M_{2^3 y}(u_{k-1}) = 1.\breve{b}_5\breve{b}_6\cdots\breve{b}_{63} \xrightarrow{\text{to 32-bit}} 1.\breve{b}_5\breve{b}_6\cdots\breve{b}_{35} = u_k\ ,\quad k = 1,2,\cdots,16$$

  influence the hash value ζ, and increases the security of MB32hash
  (because rounding error is uncertain and difficult to control).

10. Extensibility of MB32hash

The algorithm of MB32hash is extensible to any size of hash value.

⟶ SSI*n*hash, n=160, 192, 256, 384, 512, 1024, 2048, in [4][5].

Conclusion

1) We have constructed a tiny hash function MB32hash based on modified β-transformations.
2) The algorithm of MB32hash is simple and extensible.
3) The security of the algorithm of MB32hash is sufficient and easy to formulate.
4) The security of the implementation of MB32hash is also considered.

References

[1] Parry, W., Representations for real numbers, Acta Math. Acad. Sci. Hungar. 15, 95-105 (1964).

[2] Renyi, A., Representations for real numbers and their ergodic properties, Acta Math. Acad. Sci. Hungar. 8, 477-493 (1957).

[3] Yaguchi, H. and Kubo, I., A new nonrecursive pseudorandom number generator based on chaos mappings, Monte Carlo Methods Appl., Vol. 14 No. 1, 85-98 (2008).

[4] Yaguchi, H., Takashima, K. and Ueda, S., Research of new nonrecursive pseudorandom number generator and its applications, The Institute of Statistical Mathematics Cooperative Research Report 235 (2010) (in Japanese).

[5] Yaguchi, H. and Ueda, S., Construction, randomness and security of new hash functions derived from chaos mappings. (to appear)