Research Article

Filipe Araujo and Samuel Neves*

The circulant hash revisited

https://doi.org/10.1515/jmc-2018-0054
Received Dec 02, 2018; accepted Jun 10, 2020

Abstract: At ProvSec 2013, Minematsu presented the circulant hash, an almost-xor universal hash using only the xor and rotation operations. The circulant hash is a variant of Carter and Wegman's H_3 hash as well as Krawczyk's Toeplitz hash, both of which are hashes based on matrix-vector multiplication over $\mathbb{F}_2$.
In this paper we revisit the circulant hash and reinterpret it as a multiplication in the polynomial ring $\mathbb{F}_2[x]/(x^n + 1)$. This leads to simpler proofs, faster implementations in modern computer chips, and newer variants with practical implementation advantages.

Keywords: circulant hash, almost universal hash, data-dependent rotation

2020 Mathematics Subject Classification: 94A60, 94A62, 11T71

1 Introduction

Universal hashing was formally introduced by Carter and Wegman [1–3] and has numerous applications in data structures, authentication, and many other areas. In cryptography, universal hashes are often found in Wegman-Carter-Shoup [3, 4] authenticators and their variants. Lucks [5] and later Naor-Reingold [6] showed that the first and last rounds of the 4-round Luby-Rackoff construction can be universal. Universal hash functions are also very useful in randomness extraction, via the leftover-hash lemma [7].

The usage of universal hashing for authentication does, in fact, predate Carter and Wegman, and is often attributed to Gilbert, MacWilliams, and Sloane's multilinear hash [8], as well as Zobrist [9]. Many practical universal hash functions have been since proposed, including integer multiply and shift [10], polynomial evaluation [11–13], polynomial or integer remainder [14, 15], or Toeplitz matrices [16]. Achieving the best possible speeds continues to be a popular research problem, with new proposals being regularly published.

One such almost-xor universal hash function, the *circulant hash*, was recently proposed by Minematsu [17]. It is described in Definition 1.1.

Definition 1.1. Let n be a positive integer. The circulant hash (CLH) is the keyed function $\{0,1\}^n \times \{0,1\}^{n-1} \mapsto \{0,1\}^n$ defined as

$$\mathrm{CLH}_n(k,a) = \bigoplus_{i=0}^{n-1} (k \lll i) \text{ if } a_i = 1\,. \tag{1}$$

In other words, the hash consists of the xor of the key k rotated by the amounts defined by the positions of the nonzero bits of the input a. Alternatively, CLH is a matrix-vector multiplication in which the matrix is a circulant matrix (hence the name). Minematsu proved this function is $2/2^n$-almost xor universal.

Theorem 1.2 ([17, Lemma 1])**.** *Let k be a key uniformly sampled from $\{0,1\}^n$. We have*

$$\mathbf{Pr}_k\left[\mathrm{CLH}_n(k,a) \oplus \mathrm{CLH}_n(k,b) = c\right] \le 2/2^n \text{ and } \mathbf{Pr}_k\left[\mathrm{CLH}_n(k,a) = c\right] \le 2/2^n\,,$$

***Corresponding Author: Samuel Neves:** CISUC, Dept. of Informatics Engineering, University of Coimbra, Portugal; Email: sneves@dei.uc.pt
Filipe Araujo: CISUC, Dept. of Informatics Engineering, University of Coimbra, Portugal; Email: filipius@uc.pt

for n prime and 2 *a primitive root modulo n.*

Minematsu's proof of Theorem 1.2 is fairly involved, and relies on nontrivial results in linear algebra, e.g., [18].

CLH is quite similar to the "shift register hash" first described by Vazirani in the context of entropy extraction [19]. Indeed, Vazirani's proof sketch [19, Lemma 2] is similar to the one presented below in Section 3.

1.1 Contribution

Our contribution is threefold. First, we reinterpret the circulant hash in terms of polynomial arithmetic modulo $x^n + 1$, which results in a much simpler proof (Section 3), and immediate implementation possibilities. Secondly, we present several variants of CLH that have speed or flexibility advantages (Sections 4 and 5). Lastly, we (re-)obtain the differential probability of data-dependent rotation and the stretch-then-shift hash using largely the same techniques (Section 6).

2 Notation and Definitions

Unless otherwise specified, a polynomial $p(x)$ of degree d is written as $x^d + \cdots + 1$ and has coefficients in $\mathbb{F}_2$. w_i denotes the ith bit of an n-bit word w. $s \xleftarrow{\$} S$ means that s is an element of S sampled uniformly at random. We denote concatenation of two bit strings a and b by $a \| b$.

We define a family of hash functions as a finite multiset H of 2^k functions, for some k, with each $h \in H$ having domain $\{0,1\}^n$ and range $\{0,1\}^m$ for some constant n and m.

The original definition of universal hashing is due to Carter and Wegman [1], and is presented below.

Definition 2.1. A hash function family $H : \{0,1\}^n \mapsto \{0,1\}^m$ of size 2^k is ϵ-almost universal if for every distinct $a, b \in \{0,1\}^n$ there are at most $\epsilon \cdot 2^k$ functions $h \in H$ such that $h(a) = h(b)$. In other words,

$$\mathbf{Pr}_{h \in H}\left[h(a) = h(b)\right] \leq \epsilon\,.$$

If $\epsilon = 2^{-n}$, H is simply called universal.

A stronger notion is presented next. It was stated independently by Krawczyk [16], Rogaway [20], and Lucks [5] for xor differences, and generalized to arbitrary additive groups by Stinson [21].

Definition 2.2. A hash function family $H : \{0,1\}^n \mapsto \{0,1\}^m$ of size 2^k is ϵ-almost xor universal if for every distinct $a, b \in \{0,1\}^n$ there are at most $\epsilon 2^k$ functions $h \in H$ such that $h(a) \oplus h(b) = c$, for any $c \in \{0,1\}^m$. In other words,

$$\mathbf{Pr}_{h \in H}\left[h(a) \oplus h(b) = c\right] \leq \epsilon\,.$$

If $\epsilon = 2^{-n}$, H is simply called xor universal.

A xor universal hash function is necessarily universal; this is the special case $c = 0$.

3 Simpler Proof of Theorem 1.2

Our proof approach follows the lead of Rivest [22] and observes the natural embedding of bit rotation by i as multiplication by x^i in the ring $\mathbb{F}_2[x]/(x^n + 1)$. That is, a word w of n bits is interpreted as the polynomial

$$w_0 + w_1 x + \cdots + w_{n-1} x^{n-1}\,.$$

Additionally, word rotation by i bits can be interpreted as multiplication by x^i modulo $x^n + 1$[1], and addition can be accomplished by element-wise addition modulo 2, i.e., xor. Coupled with the identities 0 and 1, this yields the ring $\mathbb{F}_2[x]/(x^n + 1)$. This interpretation of circulant matrices is often found in error correcting code literature [23], cellular automata [24, 25], as well as cipher design [26].

In light of this understanding, we can restate Definition 1.1 as follows.

Definition 3.1. Let n be a positive integer. The circulant hash (CLH) is the keyed function $\{0,1\}^n \times \{0,1\}^{n-1} \mapsto \{0,1\}^n$ defined as

$$\mathrm{CLH}_n(k, a) = k \cdot a \bmod (x^n + 1)\,. \tag{2}$$

Not only does this definition have more explanatory power than Minematsu's, it also makes it clear that it does not matter whether we are rotating the key depending on the data, or vice-versa. This allows variable-time implementations without the risk of timing side-channel attacks. It also makes it obvious how to implement this function in terms of carryless multiplication, in the common case where hardware supports it. Now we are ready to restate and prove Theorem 1.2.

Theorem 3.2. *Let $k \xleftarrow{\$} \{0,1\}^n$. For any distinct $a, b \in \{0,1\}^{n-1}$ and $c \in \{0,1\}^n$, we have*

$$\mathbf{Pr}_k\left[\mathrm{CLH}_n(k, a) \oplus \mathrm{CLH}_n(k, b) = c\right] \le 2/2^n \text{ and } \mathbf{Pr}_k\left[\mathrm{CLH}_n(k, a) = c\right] \le 2/2^n,$$

for n prime and 2 a primitive root modulo n.

Proof. Since $\mathrm{CLH}_n(k, a)$ is clearly linear, i.e., $k \cdot a + k \cdot b \equiv k \cdot (a + b) \pmod{x^n + 1}$, proving uniformity for nonzero input is sufficient [16, Theorem 6].

If n is an odd prime, $x^n + 1$ factors as $(x + 1)\Phi_n(x)$, $\Phi_n(x)$ being the nth cyclotomic polynomial $x^{n-1} + x^{n-2} + \cdots + x + 1$. Suppose that $\Phi_n(x)$ is irreducible modulo 2. Because the degree of a is at most $n - 2$, $a \bmod \Phi_n(x) = a$. We can analyze the behavior of CLH modulo $\Phi_n(x)$ and $x + 1$ independently:

- Modulo $\Phi_n(x)$. This is a finite field multiplication. Therefore any equation $k \cdot a \equiv c \pmod{\Phi_n(x)}$ has a unique solution $k \equiv ca^{-1} \pmod{\Phi_n(x)}$.
- Modulo $x + 1$. By a counting argument there can be at most 2 keys satisfying $k \cdot a \equiv c \pmod{(x+1)}$ for any choice of parameters.

Combining the two cases with the Chinese remainder theorem, we obtain at most 2 distinct solutions.

It remains to be determined that $\Phi_n(x)$ is irreducible. This follows directly from the fact that n is prime, $\Phi_n(x)$ is the nth cyclotomic polynomial, and 2 is a primitive root modulo n [27, §1.6]. Thus, the differential probability is at most $2/2^n$. □

Remark 3.3. The conditions imposed by Minematsu on n are precisely the necessary conditions for there being a type-I optimal normal basis for the field $\mathbb{F}_{2^{n-1}}$. Optimal normal bases are exceptionally fast representations for polynomials over binary fields, to the point that field sizes for binary elliptic curves are often chosen such that such bases are known to exist [28].

The circulant hash is, as a matter of fact, a multiplication in the "ghost bit" redundant representation of a type-I optimal normal basis [29, 30], with one exception—the end result remains in the ring $\mathbb{F}_2[x]/(x^n + 1)$ instead of being reduced modulo $\Phi_n(x)$. Reduction by "all-one" polynomials is exceptionally simple, and would enable a straightforward $\{0,1\}^{n-1} \times \{0,1\}^{n-1} \mapsto \{0,1\}^{n-1}$ finite-field multiplicative hash—simply add the coefficient of x^{n-1} to every coefficient in the polynomial, i.e.,

$$(w_0 + w_{n-1}) + (w_1 + w_{n-1})x + \cdots + (w_{n-1} + w_{n-1})x^{n-1}\,,$$

and output the first $n - 1$ coefficients. This can be achieved on a computer with a single xor and arithmetic shift.

1 In $\mathbb{F}_2$, 1 and −1 are interchangeable. As such, we use positive signs whenever possible for simplicity.

4 Polynomial Evaluation

A very successful approach to designing universal hashes that accept an arbitrary-sized input is polynomial evaluation [31, 32]. Our interpretation of the circulant hash in the previous section lends itself to a straightforward polynomial evaluation variant, which allows for arbitrary-sized inputs.

In this setting, we split an $m(n-1)$-bit input message $\mathbf{a}$ into m blocks $(a_0, a_1, \ldots, a_{m-1})$ of $n-1$ bits each, padding as needed[2]. We interpret the message $\mathbf{a}$ as the degree m polynomial with coefficients in $\mathbb{F}_2[x]/(x^n+1)$:

$$\mathbf{a}(X) = X^m \cdot a_{m-1} + X^{m-1} \cdot a_{m-2} + \cdots + X \cdot a_0 + 0\,. \tag{3}$$

We can now define the hash function $\mathrm{PCLH}(k, \mathbf{a})$ as the evaluation of this polynomial at k.

Definition 4.1. Let n be a positive integer. The polynomial evaluation circulant hash (PCLH) is a keyed function $\{0,1\}^n \times \{0,1\}^{(n-1)m} \mapsto \{0,1\}^n$ defined as

$$\mathrm{PCLH}_n(k, \mathbf{a}) = \sum_{i=1}^{m} k^i \cdot a_i \bmod (x^n + 1) = \mathbf{a}(k)\,. \tag{4}$$

Theorem 4.2. *Let n be a prime such that 2 is a primitive root modulo n. For messages of at most m blocks, the polynomial hash PCLH_n is $2m/2^n$-almost xor universal.*

Proof. By linearity, $\mathrm{PCLH}_n(k, \mathbf{a}) + \mathrm{PCLH}_n(k, \mathbf{b}) = \mathrm{PCLH}_n(k, \mathbf{a}+\mathbf{b})$, where $\mathbf{a}+\mathbf{b}$ indicates addition of polynomials of the form (3). Furthermore, $\mathrm{PCLH}_n(k, \mathbf{a}+\mathbf{b}) = (\mathbf{a}+\mathbf{b})(k)$. Thus we have $(\mathbf{a}+\mathbf{b})(k) = c$, from which follows that $(\mathbf{a}+\mathbf{b}+c)(k) = 0$, which means the number of keys coincides with the number of roots of the polynomial $\mathbf{a}+\mathbf{b}+c$, which has degree at most m.

As in Theorem 1.2, given the constraints on n we can consider the behavior of the hash modulo $x+1$ and $\Phi_n(x)$ independently:

- Modulo $\Phi_n(x)$. This is a finite field polynomial evaluation and the fundamental theorem of algebra applies—there are at most m roots of a degree m polynomial.
- Modulo $x+1$. Here, by a simple counting argument, there cannot be more than 2 roots for any polynomial.

This leads to a probability of at most $2m/2^n$. The case $m = 1$ is exactly Theorem 1.2. □

5 A Variant for Powers of 2

Operating on block sizes of prime size, as required by the original CLH function, is not very convenient. Ideally, one would work instead on "natural" power of two blocks, such as $n = 128$. To this end, we define a new variant of the circulant hash that works in this setting.

Definition 5.1. Let n be a power of 2. The modified circulant hash (MCLH) is a keyed function $\{0,1\}^n \times \{0,1\}^{n-1} \mapsto \{0,1\}^n$ defined as

$$\mathrm{MCLH}_n(k, a) = k \cdot \left(a + x^{n-1}(a + 1 \bmod (x+1))\right) \bmod (x^n + 1)\,. \tag{5}$$

In this variant, instead of keeping the $(n-1)$th coefficient empty we use it to ensure that the input is always invertible modulo $x^n + 1$ with the injective transformation $\left(a + x^{n-1}(a + 1 \bmod (x+1))\right)$. The input to MCLH

2 Here we simply consider messages of fixed length and multiple of $n-1$ bits. Standard padding methods [31, 32] apply to make PCLH suitable for variable-length messages.

can equivalently be defined as every element of $\{0,1\}^n$ with an odd number of bits. Lemma 5.2 proves this is the case.

Lemma 5.2. For any $a \in \{0,1\}^{n-1}$, $\left(a + x^{n-1}(a + 1 \bmod (x+1))\right) \bmod (x+1) = 1$.

Proof. $x^{n-1} \bmod (x+1) = (x^{n-1})(1) = 1$ by the polynomial remainder theorem. Thus,

$$\left(a + x^{n-1}(a + 1 \bmod (x+1))\right) \bmod (x+1) = a \bmod (x+1) + (a+1) \bmod (x+1) = 1.$$

□

We now prove MCLH is an almost-xor universal hash.

Theorem 5.3. *Let n be a power of 2. Let $k \xleftarrow{\$} \{0,1\}^n$. For any distinct $a, b \in \{0,1\}^{n-1}$ and $c \in \{0,1\}^n$, we have*

$$\mathbf{Pr}_k\left[\mathrm{MCLH}_n(k,a) \oplus \mathrm{MCLH}_n(k,b) = c\right] \le 1/2^n \text{ and } \mathbf{Pr}_k\left[\mathrm{MCLH}_n(k,a) = c\right] \le 1/2^n.$$

Proof. Again, by linearity it suffices to prove $\mathbf{Pr}_k\left[\mathrm{MCLH}_n(k,a) = c\right] \le 1/2^n$. Since we are working in characteristic 2 and n is a power of 2, $x^n + 1 = (x+1)^n$. As such, if $\gcd(a, x+1) = 1$, then $\gcd(a, (x+1)^n) = 1$ for any positive n.

Given an equation $k \cdot a \equiv c \pmod{x^n+1}$, there is a unique k such that $k \equiv ca^{-1} \pmod{x^n+1}$. a is always invertible modulo $x^n + 1$ by construction, as shown in Lemma 5.2. □

Remark 5.4. The value $a \bmod (x+1)$ can be efficiently computed with, e.g., the Intel instruction `popcnt`. It can also be computed by keeping track of the parity of the input.

6 Related Functions

6.1 Data-dependent Rotation

The same mathematical framework used in the previous sections may also be used to show that data-dependent rotation has low differential probability when the difference is in the rotation amounts. In Theorem 6.1 we re-derive the differential probability result of [33] in terms of multiplication in $\mathbb{F}_2[x]/(x^n+1)$.

Theorem 6.1 ([33])**.** *Let n be a power of 2, and $k \xleftarrow{\$} \{0,1\}^n$, and distinct $r_1, r_2 \in \{0,1,\ldots,n-1\}$ be inputs. Then*

$$\mathbf{Pr}_k\left[k \lll r_1 \oplus k \lll r_2 = c\right] \le 2^{\gcd(r_2-r_1,n)-n}.$$

Proof. As above, $k \lll r_1 \oplus k \lll r_2$ is equivalent to $k \cdot (x^{r_1} + x^{r_2}) \bmod (x^n+1)$. Thus, we want to bound the probability that $k \cdot (x^{r_1} + x^{r_2}) \bmod (x^n+1) = c$. We begin by rewriting it as $k \cdot \left(x^{r_2-r_1} + 1\right) = c \cdot x^{-r_1}$.

We may factor any exponent $r_2 - r_1 \bmod n$ as $2^p \cdot q$, for odd q. Then $x^{r_2-r_1} + 1 = (x^q+1)^{2^p} = (x+1)^{2^p}(x^{q-1} + x^{q-2} + \cdots + 1)^{2^p}$. We can now rewrite the above equality further as

$$k \cdot (x+1)^{2^p} = c \cdot x^{-r_1} \cdot (x^{q-1} + x^{q-2} + \cdots + 1)^{-2^p}, \tag{6}$$

as the right hand side multipliers are all units in this ring. The remaining factor $(x+1)^{2^p}$ remains to be handled. Since 2^p divides n, multiplication by $(x+1)^{2^p}$ is a surjective group homomorphism sending $\mathbb{F}_2[x]/(x^n+1)$ to the unique subgroup of $\mathbb{F}_2[x]/(x^n+1)$ of order 2^{n-2^p}. As such, each key is one of 2^{2^p} equivalent representations in this subgroup, $k + t \cdot (x+1)^{n-2^p}$ for t of degree less than 2^p. If the right hand side of (6) belongs to the subgroup, i.e., it is congruent to 0 modulo $(x+1)^{2^p}$, there is a unique solution in the subgroup with corresponding 2^{2^p} equivalents in the main group; otherwise there are no solutions.

Finally, since n is a power of 2, we have $2^p = \gcd(r_2 - r_1, n)$. Putting it all together, we have at most $2^{\gcd(r_2-r_1,n)}$ possible keys for any given r_1, r_2, c, leading to a maximum probability of $2^{\gcd(r_2-r_1,n)-n}$. □

6.2 Stretch-then-shift

The OCB3 authenticated encryption mode [34] introduced a special-purpose almost xor universal function—stretch-then-shift—to hash the 6 least significant bits of a nonce. This function, $H^c(k, a)$, takes a 128-bit key k, a 6-bit input a, and outputs the first 128 bits of the result:

$$H^c(k, a) = (\text{stretch}(k) \ll a)[0 \dots 127]\,,$$

where stretch(k) is defined as $k \| k \oplus (k \ll c)$, for some constant $c < 128$.

As with the case of the circulant hash, the authors of stretch-then-shift offer only a linear-algebraic rationale for the almost xor universal property of their function. Yet, we can also offer a polynomial interpretation that, once again, makes things simpler. We can understand this function as equivalent to a multiplication in the ring $\mathbb{F}_2[x]/(x^{128} + x^c + 1)$. Namely,

$$H^c(k, a) = k \cdot x^a \bmod (x^{128} + x^c + 1)\,.$$

This comes directly from the fact that shift left by a is equivalent to polynomial multiplication by x^a, and reduction of a polynomial f of degree $< 256 - c$ modulo $x^{128} + x^c + 1$ can be written as

$$f \bmod (x^{128} + x^c + 1) = f \bmod x^{128} + \lfloor f / x^{128} \rfloor \cdot (x^c + 1)\,,$$

since $x^{128} \bmod x^{128} + x^c + 1 = x^c + 1$. Therefore, the stretch-then-shift hash is nothing more than an optimized polynomial multiplication modulo a trinomial. Since there are no irreducible trinomials of degree 128, the analysis proceeds similarly to the case of rotation.

We now prove in Theorem 6.2 that the concrete choice of shift used in OCB3, H^8, makes for a xor universal hash.

Theorem 6.2. *Let $k \xleftarrow{\$} \{0,1\}^{128}$, and distinct $a, b \in \{0, 1, \dots, 63\}$ be inputs. Then, for any $c \in \{0,1\}^{128}$,*

$$\mathbf{Pr}_k \left[H^8(k, a) \oplus H^8(k, b) = c \right] \le 2^{-128}\,.$$

Proof. We use the equivalence of H^8 to $k \cdot x^a \bmod (x^{128} + x^8 + 1)$. We have, as before, $k \cdot x^a + k \cdot x^b = k \cdot (x^a + x^b) = c$ can be written as $k \cdot (x^{a-b} + 1) = c \cdot x^{-b}$. As long as both x^{a-b} and x^{-b} are unique (i.e., invertible modulo $x^{128} + x^8 + 1$), there is a unique k satisfying the equation.

The modulus $x^{128} + x^8 + 1$ factors as $(x^8 + x^6 + x^5 + x^3 + 1)^8 (x^8 + x^6 + x^5 + x^4 + x^3 + x + 1)^8$. As x^{-b} shares no factors with it, we need only concern with $x^{a-b} + 1$. Without loss of generality, we consider positive differences $a > b$ only. As long as $x^{a-b} + 1$ shares no factor with the modulus, this multiplication is invertible. We rewrite $x^{a-b} + 1$ as $x^{a-b} = 1$ modulo any of the factors of the modulus, i.e., the order of x. The order of x modulo $x^8 + x^6 + x^5 + x^3 + 1$ is 255; the order of x modulo $x^8 + x^6 + x^5 + x^4 + x^3 + x + 1$ is 85. Therefore, as long as $a - b \in [-84, \dots, 84]$, H^8 is injective and thus for any c at most one choice of k exists. □

This interpretation also gives us efficient ways to find suitable constants c. Given the factorization of the trinomial $x^{128} + x^c + 1$, the function is xor universal as long as x has sufficiently large order modulo every factor. Alternatively, $\gcd(x^a + 1, x^{128} + x^c + 1) = 1$ for every admissible value of a.

References

[1] Mark N. Wegman and Larry Carter, New Classes and Applications of Hash Functions, in: *20th Annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, 29-31 October 1979*, pp. 175–182, IEEE Computer Society, 1979.

[2] Larry Carter and Mark N. Wegman, Universal Classes of Hash Functions, *J. Comput. Syst. Sci.* **18** (1979), 143–154.
[3] Mark N. Wegman and Larry Carter, New Hash Functions and Their Use in Authentication and Set Equality, *J. Comput. Syst. Sci.* **22** (1981), 265–279.
[4] Victor Shoup, On Fast and Provably Secure Message Authentication Based on Universal Hashing, in: *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings* (Neal Koblitz, ed.), Lecture Notes in Computer Science 1109, pp. 313–328, Springer, 1996.
[5] Stefan Lucks, Faster Luby-Rackoff Ciphers, in: *Fast Software Encryption, Third International Workshop, Cambridge, UK, February 21-23, 1996, Proceedings* (Dieter Gollmann, ed.), Lecture Notes in Computer Science 1039, pp. 189–203, Springer, 1996.
[6] Moni Naor and Omer Reingold, On the Construction of Pseudorandom Permutations: Luby-Rackoff Revisited, *J. Cryptology* **12** (1999), 29–66.
[7] Russell Impagliazzo, Leonid A. Levin and Michael Luby, Pseudo-random Generation from one-way functions (Extended Abstracts), in: *Proceedings of the 21st Annual ACM Symposium on Theory of Computing, May 14-17, 1989, Seattle, Washigton, USA* (David S. Johnson, ed.), pp. 12–24, ACM, 1989.
[8] Edgar N. Gilbert, F. Jessie MacWilliams and Neil J. A. Sloane, Codes Which Detect Deception, *Bell System Technical Journal* **53** (1974), 405–424.
[9] Albert L. Zobrist, *A New Hashing Method With Application for Game Playing*, University of Wisconsin—Madison Department of Computer Sciences, Report no. #88, April 1970.
[10] Martin Dietzfelbinger, Universal Hashing and k-Wise Independent Random Variables via Integer Arithmetic without Primes, in: *STACS 96, 13th Annual Symposium on Theoretical Aspects of Computer Science, Grenoble, France, February 22-24, 1996, Proceedings* (Claude Puech and Rüdiger Reischuk, eds.), Lecture Notes in Computer Science 1046, pp. 569–580, Springer, 1996.
[11] Bert den Boer, A Simple and Key-Economical Unconditional Authentication Scheme, *Journal of Computer Security* **2** (1993), 65–72.
[12] Jürgen Bierbrauer, Thomas Johansson, Gregory Kabatianskii and Ben J. M. Smeets, On Families of Hash Functions via Geometric Codes and Concatenation, in: *Advances in Cryptology - CRYPTO '93, 13th Annual International Cryptology Conference, Santa Barbara, California, USA, August 22-26, 1993, Proceedings* (Douglas R. Stinson, ed.), Lecture Notes in Computer Science 773, pp. 331–342, Springer, 1993.
[13] Richard Taylor, Near Optimal Unconditionally Secure Authentication, in: *Advances in Cryptology - EUROCRYPT '94, Workshop on the Theory and Application of Cryptographic Techniques, Perugia, Italy, May 9-12, 1994, Proceedings* (Alfredo De Santis, ed.), Lecture Notes in Computer Science 950, pp. 244–253, Springer, 1994.
[14] Michael O. Rabin, *Fingerprinting by Random Polynomials*, Center for Research in Computing Technology, Harvard University, Report no. TR-CSE-03-01, 1981.
[15] Richard M. Karp and Michael O. Rabin, Efficient Randomized Pattern-Matching Algorithms, *IBM Journal of Research and Development* **31** (1987), 249–260.
[16] Hugo Krawczyk, LFSR-based Hashing and Authentication, in: *Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings* (Yvo Desmedt, ed.), Lecture Notes in Computer Science 839, pp. 129–139, Springer, 1994.
[17] Kazuhiko Minematsu, A Short Universal Hash Function from Bit Rotation, and Applications to Blockcipher Modes, in: *Provable Security - 7th International Conference, ProvSec 2013, Melaka, Malaysia, October 23-25, 2013. Proceedings* (Willy Susilo and Reza Reyhanitabar, eds.), Lecture Notes in Computer Science 8209, pp. 221–238, Springer, 2013.
[18] D. E. Daykin, On the Rank of the Matrix $f(A)$ and the Enumeration of Certain Matrices over a Finite Field, *Journal of the London Mathematical Society* **s1-35** (1960), 36–42.
[19] Umesh V. Vazirani, Efficiency Considerations in Using Semi-random Sources (Extended Abstract), in: *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA* (Alfred V. Aho, ed.), pp. 160–168, ACM, 1987.
[20] Phillip Rogaway, Bucket Hashing and Its Application to Fast Message Authentication, *J. Cryptology* **12** (1999), 91–115.
[21] Douglas R. Stinson, On the Connections Between Universal Hashing, Combinatorial Designs and Error-Correcting Codes, in: *Congressus Numerantium 114*, pp. 7–27, 1996.
[22] Ronald L. Rivest, The invertibility of the XOR of rotations of a binary word, *Int. J. Comput. Math.* **88** (2011), 281–284.
[23] Cunsheng Ding and Zhengchun Zhou, Binary cyclic codes from explicit polynomials over GF(2^m), *Discrete Mathematics* **321** (2014), 76–89.
[24] Olivier Martin, Andrew M. Odlyzko and Stephen Wolfram, Algebraic properties of cellular automata, *Comm. Math. Phys.* **93** (1984), 219–258.
[25] Dario Bini, Gianna M. Del Corso, Giovanni Manzini and Luciano Margara, Inversion of circulant matrices over $\mathbb{Z}_m$, *Math. Comput.* **70** (2001), 1169–1182.
[26] Joan Daemen, *Cipher and hash function design, strategies based on linear and differential cryptanalysis, PhD Thesis*, K.U.Leuven, 1995.
[27] Ian F. Blake and Ronald C. Mullin, *The Mathematical Theory of Coding*, Academic Press, April 1975.
[28] Christophe Doche, *Finite Field Arithmetic*, Handbook of Elliptic and Hyperelliptic Curve Cryptography. (Henri Cohen, Gerhard Frey, Roberto Avanzi, Christophe Doche, Tanja Lange, Kim Nguyen and Frederik Vercauteren, eds.), Chapman and Hall/CRC,

2005, pp. 201–237.
[29] Toshiya Itoh and Shigeo Tsujii, Structure of Parallel Multipliers for a Class of Fields GF(2^m), *Inf. Comput.* **83** (1989), 21–40.
[30] Joseph H. Silverman, Fast Multiplication in Finite Fields GF(2^N), in: *Cryptographic Hardware and Embedded Systems, First International Workshop, CHES'99, Worcester, MA, USA, August 12-13, 1999, Proceedings* (Çetin Kaya Koç and Christof Paar, eds.), Lecture Notes in Computer Science 1717, pp. 122–134, Springer, 1999.
[31] Ted Krovetz and Phillip Rogaway, Fast Universal Hashing with Small Keys and No Preprocessing: The PolyR Construction, in: *Information Security and Cryptology - ICISC 2000, Third International Conference, Seoul, Korea, December 8-9, 2000, Proceedings* (Dongho Won, ed.), Lecture Notes in Computer Science 2015, pp. 73–89, Springer, 2000.
[32] Daniel J. Bernstein, The Poly1305-AES Message-Authentication Code, in: *Fast Software Encryption: 12th International Workshop, FSE 2005, Paris, France, February 21-23, 2005, Revised Selected Papers* (Henri Gilbert and Helena Handschuh, eds.), Lecture Notes in Computer Science 3557, pp. 32–49, Springer, 2005.
[33] Scott Contini and Yiqun Lisa Yin, On differential properties of data-dependent rotations and their use in MARS and RC6, in: *The 2nd Conference on Advanced Encryption Standard (AES), Rome, Italy, March 22-23, 1999*, February 1999.
[34] Ted Krovetz and Phillip Rogaway, The Software Performance of Authenticated-Encryption Modes, in: *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers* (Antoine Joux, ed.), Lecture Notes in Computer Science 6733, pp. 306–327, Springer, 2011.