Modeling of Hash Functions on the Basis of Irreducible Polynomials in a Finite Fields

George Vostrov
Ph. D. of Technical Sciences
Odessa national polytechnic university
Odessa, Ukraine
vostrov@gmail.com

Olena Ponomarenko
Odessa national polytechnic university
Odessa, Ukraine
ponomarenkoelena1997@gmail.com

Abstract—**In this paper the method for constructing hash functions on the basis of irreducible polynomials in finite fields has been considered. The problem of searching for irreducible polynomials was considered. Computer modeling of hash functions using irreducible polynomials was performed. The results of the use of various irreducible polynomials and their analysis was presented.**

Index Terms—***H*ash function, finite field, irreducible polynomial, cyclic redundant code, polynomial arithmetic, collision.**

I. Introduction

Information technology systems require the availability of effective tools that would significantly reduce the amount of memory required to store and transfer large amounts of data available to a limited number of users, and to verify their integrity. This is also related to the fact that financial transactions with funds and the storage of personal data of users are carried out on the Internet. For such purposes hash functions are widely used. They convert the initial data of an arbitrary length into a fixed-length sequence, called a hash code, hash or message convolution.

Hash functions are advisable to apply to valuable confidential data, which can only be accessed by certain individuals. Such data is most often represented as text or a sequence of symbols. It should be noted that with minor changes in the input data, the hash function result must completely change, that is, have an avalanche effect in order to ensure that data can not be falsified by minor changes. This hash function also allows to use them in the following cases: to find duplicates in the data set; constructing associative arrays; calculation of checksums for the subsequent detection and correction of errors that occurred during the transfer or storage of data; development of an electronic digital signature; for saving passwords in databases.

Designing a quality hash function is a difficult task. When developing hashing algorithms, the vulnerability of hash functions should be taken into account. The powers of the set of input sequences and the set of all possible values of the hash function can be found in any relation. As a rule, the set of input data has a larger dimension than the number of all possible values of the function, which leads to the transformation of various messages into one hash. Such a case is called a "collision" and is one of the important factors that is taken into account when constructing hashing algorithms in cryptographic systems [1], as well as in many data structures, such as hash tables [2].

At this stage of development of the theory of hashing, there is still no clear definition of the concept of hash function and exact requirements for their construction. The general requirements that hash functions must meet are irreversibility (it is impossible to create an algorithm with polynomial computational complexity that restores the original data in real time), collision resistance, fast calculation speed and the presence of an avalanche effect (with a slight change in the input data, the result should vary essentially). Depending on the application, additional requirements are imposed on the hash function, such as calculation complexity, convolution length, and cryptographic stability.

The main problem of using hash functions is that the existence of irreversible functions that exclude the possibility of collisions is not proved. In addition, there are no universal methods of hashing and they should be selected based on the field of their application. In practice, functions are used for which the theoretical probability of collisions is close to zero, but with the advent of more powerful computational devices, the search for collisions may not be such a difficult task. For this reason, existing algorithms require constant improvement. A special role is played by theoretical-complexity problems, namely, algebraic number theory [3]. One such problem is the search for irreducible polynomials of a given degree over a field F_p or $GF(p)$ that can be used to find hash codes of messages. The problem of finding irreducible polynomials has been considered in the paper, as well as a hash method based on calculating the remainder from dividing by an irreducible polynomial.

II. Arithmetic of finite fields

Such sections of algebra as the theory of finite fields and the theory of polynomials over finite fields have increasingly influence the construction of various systems for protecting information, encoding and decoding information. In particular, there appeared algorithms for cyclic redundant codes(CRC)

[4], which use polynomials over a fields F_p. Cyclic redundancy codes can be used as hash functions to detect errors and verify data integrity.

Because the finite field is a set with a finite number of elements, the operations of addition, subtraction, multiplication and division can be performed in accordance with the axioms of the field [5]. Since the finite fields are closed with respect to the above operations, i.e. for any two elements of the field $a,b \in F_p$, then when any of the operations are performed, the result is an element belonging to this field $c \in F_p$. It should be borne in mind that all calculations in finite fields are made modulo p, which is a characteristic of a finite field and is a prime number.

The simplest example of a finite field is the ring of residue classes $Z/(p)$ modulo p a prime number, which can be identified with a Galois field $F_p = GF(p)$ of order p [5]. According to the theorem on the existence and uniqueness of finite fields, for every prime number p and natural number n, there exists a finite field of p^n elements. To construct a field F_{p^n}, it is necessary to find a polynomial $P(x)$ of degree n irreducible over a field F_p. Such a field is represented by polynomials over F_p a degree not higher $n-1$. An irreducible polynomial is a polynomial that is not decomposable into non-trivial polynomials and is an analog of prime numbers in the natural series. A peculiarity of irreducible polynomials is that, being irreducible in one field, a polynomial turns out to be reducible in another field, which has found application in the theory of coding and information protection systems.

The search for irreducible polynomials is a difficult-to-compute problem, especially over fields of large dimension. The procedure for finding irreducible polynomials requires efficient algorithms and large computational resources, as in the case of finding prime numbers, which is the main problem for constructing effective hashing algorithms based on them. At the moment there are no effective algorithms for searching for irreducible polynomials, there are only criteria for irreducibility and verification methods for irreducibility. The search is carried out by examining the multibodies and checking each individually for irreducibility. To check the polynomial $P(x)$ of degree $n \geq 2$ on irreducibility over a field of characteristic there exists the following algorithm [6]:

1) The initial value of a polynomial is initialized $G_0(x) = x$.

2) The following value is calculated $G_1(x) = G_0(x)^p \bmod P(x)$.

3) The greatest common divisor (GCD) between $P(x)$ and $(G_1(x) - x)$ is calculated. If the GCD is not equal to one, then this polynomial is reducible. Otherwise, the next value is calculated according to the recurrence formula $G_i(x) = G_{i-1}(x)^p \bmod P(x)$, where $i = \overline{1, \lfloor n/2 \rfloor}$, $\lfloor\ \rfloor$ is the operation of taking the whole part of the number.

4) If the GCD $P(x)$ and each one $(G_i(x) - x)$ is equal to one, then the polynomial $P(x)$ is irreducible.

The disadvantage of such an algorithm is the low computation speed for sufficiently large values, since at each step the operation of raising and finding the GCD is performed.

For computations in finite fields, polynomial arithmetic is used. The addition in the field F_{p^n} corresponds to the usual addition of polynomials modulo p. Multiplication is performed in two steps - first as a simple multiplication of polynomials, and then the remainder from division into an irreducible polynomial is calculated, with which the field F_{p^n} is constructed. For example, fields of the same dimension can be constructed in different ways, depending on the choice of an irreducible polynomial. They are of the same order and are isomorphic to each other. This follows from the fact that for the characteristic p field there are several irreducible polynomials of degree n. Examples of irreducible polynomials for a field F_2 are given in TABLE1.

TABLE I. IRREDUCIBLE POLYNOMIALS OVER A FIELD OF CHARACTERISTIC 2

Power	Irreducible polynomials
2	(x^2+x+1)
3	(x^3+x^2+1), (x^3+x+1)
4	$(x^4+x^3+x^2+x+1)$, (x^4+x^3+1), (x^4+x+1)
5	(x^5+x^2+1), $(x^5+x^3+x^2+x+1)$, $(x^5+x^4+x^3+x+1)$, $(x^5+x^4+x^3+x^2+1)$, $(x^5+x^4+x^2+x+1)$

III. Hashing with Irreducible Polynomials

A possible way to construct a hash function is to use modulo division of an irreducible polynomial [3]. For efficiency of computer implementation it is convenient to use calculations in fields F_{2^g}. This allows calculations on the data in the form of a sequence of bits. The search for the remainder of the division is realized using bitwise shifts and an exclusive disjunction (XOR).

For hashing, the data is encoded by some selected method in the sequence $a_1, a_2, ..., a_k$ of zeros and ones that corresponds to a certain polynomial $A(x)$, and the hash code

$h(a_1, a_2, ..., a_k)$ is the sequence of bits obtained by dividing by an irreducible polynomial $P(x)$ and is calculated by the formulas (1) and (2).

$$B(x) = A(x) \bmod P(x) \qquad (1)$$

$$h(a_1, a_2, ..., a_k) = b_{n-1}b_{n-2}..b_1b_0 \qquad (2)$$

In formula (2) $b_{n-1}b_{n-2}..b_1b_0$ are the coefficients of the polynomial $B(x)$ obtained as a remainder from dividing the polynomial $A(x) = a_1x^{k-1} + a_2x^{k-2} + ... + a_{k-1}x + a_k$ by an polynomial $P(x) = p_nx^n + p_{n-1}x^{n-1} + ... + p_1x + p_0$ of degree n.

Such a function is stable to the restoration of the initial data, since even knowing the dimensionality of the field and the irreducible polynomial used, it is difficult to decipher the data, especially for large powers of an irreducible polynomial. Irreducible polynomials should be selected based on the scope of the hash functions, since the convolution length is equal to the degree of the polynomial. So, for use in information security systems, at the moment the optimal length is not less than 128 bits and not more than 512 bits. The use of a polynomial of sufficient dimension plays a significant role. If a polynomial of degree is chosen l, then the set of all possible values that the convolution of a function can take is equal to. 2^l . For example, using an irreducible polynomial $P(x) = x^4 + x + 1$, the number of all possible bundles will be 16 and finding messages with identical convolutions is not difficult.

Computer simulation of hash functions was performed on the basis of irreducible polynomials of degree 32 with a different number of monomials As a result, an analysis of the effectiveness of hashing is performed using each of the polynomials. An important factor in choosing an irreducible polynomial is the number of monomials in it. For higher computing speed, it is desirable to find polynomials with a minimal number of monomials. To compare the results of hashing, polynomials with 5, 12, and 18 monomials were chosen. The computation rates for convolutions of input data of different lengths and the use of various irreducible polynomials of the same degree are given in TABLEII. Irreducible polynomials were written in the binary representation, which is a sequence of coefficients for monomials. The coefficient of the highest degree is not taken into account here. For example, for a polynomial $P(x) = x^4 + x + 1$ it will be true $x^4 = x + 1$, since the subtraction operation is the same as adding modulo 2. Write the polynomial in the form $x^4 = 0 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 1 \cdot x^0$, so in the binary representation the polynomial will be written as 0011 and the number of bits for its recording is equal to the degree of the polynomial.

TABLE II. THE SPEED OF CALCULATION OF CONVOLUTIONS (IN MS) USING IRREDUCIBLE POLYNOMIALS OF DEGREE 32

№	Irreducible polynomials	Sequence length, bit		
		64	256	512
1	00000000010000000000000000000111	1,1	1,8	2,2
2	10000001010000001010000011010101 1	1,9	2,1	2,5
3	01110100000110111000110011010111	2,1	3,6	4,9

At first glance, the difference in the speed of computation is small, but when processing data volumes from 1Mb or more, the difference between calculations can be one hour or more. This hashing method is effective for applying over small amounts of data within a few Kbytes. TABLEIII shows the results of hashing by irreducible polynomials from TABLEII for an arbitrary 64-bit string and for the same row with minor changes to check for avalanche effect and mixing property.

TABLE III. THE RESULTS OF APPLYING HASH FUNCTIONS ON THE BASIS OF IRREDUCIBLE POLYNOMIALS OF DEGREE 32

Original bit sequence	Irreducible polynomials		
	1	2	3
0110001100111001 0001110000011111 1110000011001000 0110110100011111	11010011 11101101 01000000 00001110	01000110 00001100 01101101 01100110	01010111 11011111 00101111 01001100
0110001100111001 0010110000011111 1110000011001000 0110110100011111	11010000 11101101 11010000 00101010	01010010 00000101 00100010 10110110	11110010 00010011 11101010 01001010

The functions realized on the basis of the polynomials under consideration have the mixing property. This means that there is no connection between the convolution and the original data. Since with a slight change in the source data, the hashing result should change significantly, in the original bit sequence, the 19th and 20th bits were changed to test for compliance with this property. Based on the results given in TABLEIII, the best avalanche effect is possessed by functions based on polynomials with a large number of monomials. It is worth noting that, despite the small degree of the above polynomials, hash functions based on them have some resistance to collisions. When sorting the bundles obtained by processing data in the volume of 1000, 5000, 10,000 and 30,000, no collisions were found, although this can not guarantee their absence on large volumes.

The hashing method considered is suitable for bit sequences that can be represented by a polynomial of higher degree than the degree of the selected irreducible polynomial. Lesser sequences must be supplemented with a function. In most existing hash algorithms, the addition to the required length is accomplished by appending to the sequence one single bit and bits of zeros. In addition, it is desirable to add to the sequence its original length, which will reduce the likelihood of collisions after the addition. The use of the remainder from division into an irreducible polynomial can serve as a separate hash function, and is used in conjunction with other algorithms to improve certain properties. Also, the hash found by this method can be used as a cryptographic salt.

CONCLUSION

The theory of finite fields can be used to construct hash functions, but along with its application, problems arise that require a separate investigation for further solutions. One of these problems is finding irreducible polynomials with certain properties. For fields of characteristic of prime numbers of high digit capacity, the problem of finding irreducible polynomials of certain degrees is considerably more complicated and requires large computational costs.

It was shown that it is expedient to use irreducible polynomials of sufficiently large powers. Polynomials consisting of a small number of monomials allow us to find convolutions for a smaller number of operations. However, polynomials with a large number of polynomials improve the avalanche effect of the hash function. Both have the same resistance to collisions. An irreducible polynomial must be chosen based on the desired properties of the hash function. To strengthen the cryptographic strength and improve the avalanche effect, it is necessary to choose irreducible polynomials of degree 128 and higher with the maximum possible number of monomials. In cases where the hash function is used in systems requiring high computational speed, it is advisable to use irreducible polynomials with a minimal number of monomials.

The main disadvantage of hash functions based on irreducible polynomials is the low computational speed for large amounts of data. In addition to the expansions of the field under consideration, in order to increase the resistance to collisions, it is necessary to consider fields of large characteristics, which is a task for further solution. This will allow the data to be hashed into elements from a larger field, but it should be taken into account that this will complicate computer operations on the computer.

REFERENCES

[1] B. Schneier, “Applied Cryptography, Second Edition: Protocols, Algorthms, and Source Code in C”, 1995,784 p.

[2] R. Sedgewick, “Algorithms in C++, Parts 1-4: Fundamentals, Data Structure, Sorting, Searching”, 3rd ed., 1988, 752 p.

[3] E. A. Khomich, "Irreducible polynomials over finite fields and connection with cryptography", Academic Publicistics, 2017, pp. 19-22.

[4] Henry S. Warren, Jr., “Hacker’s Delight”, 3nd ed.,2013, 816 p.

[5] R. Lidl, H. Niederreiter, “Finite Fields”, 1997, 768 p.

[6] R. Crandall, K. Pomerance, “Prime Numbers: A Computational Perspective”, 2011, 664 p.

[7] M. O. Rabin, R. M. Karp. “Efficient randomized pattern-matching algorithms”, IBM, 1987, pp. 249 – 260.