BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 63, No. 4, 2015
DOI: 10.1515/bpasts-2015-0112

The average time complexity of probabilistic algorithms for finding generators in finite cyclic groups

T. ADAMSKI[1] and W. NOWAKOWSKI[2]*

[1] Institute of Electronic Systems, Warsaw University of Technology, 15/19 Nowowiejska St., 00-662 Warsaw, Poland
[2] Institute of Mathematical Machines, 34 Krzywickiego St., 02-078 Warsaw, Poland

Abstract. Generators of finite cyclic groups play important role in many cryptographic algorithms like public key ciphers, digital signatures, entity identification and key agreement algorithms. The above kinds of cryptographic algorithms are crucial for all secure communication in computer networks and secure information processing (in particular in mobile services, banking and electronic administration). In the paper, proofs of correctness of two probabilistic algorithms (for finding generators of finite cyclic groups and primitive roots) are given along with assessment of their average time computational complexity.

Key words: probabilistic algorithm, average time complexity, group generator, cyclic groups, primitive roots, primitive polynomials.

List of Symbols

N	–	set of natural numbers,
Z	–	set of integers,
$\langle n, m\rangle$	–	subset of the set of integers $\langle n, m\rangle \overset{df}{=} \{k \in Z : n \le k \le m\}$,
F_q	–	finite field with q elements,
$GF(p^n)$	–	finite field with p^n elements,
Z_n	–	ring of integers modulo n,
Z_p	–	field of integers modulo p, where p is a prime,
Z_n^*	–	multiplicative group of the ring Z_n,
F_q^*	–	multiplicative group of the finite field F_q^*,
C_n	–	cyclic group of the order n,
$G_1 \times G_2$	–	direct product of groups G_1 and G_2,
$(\Omega, \mathcal{M}, P)$	–	probabilistic space,
φ	–	the Euler's function,
$E(X)$	–	expected value of the random variable X,
$D^2(X)$	–	variance of the random variable X,
$GCD(n, m)$	–	greatest common divisor of n and m,
$\#G$	–	number of elements of a finite set G, order of the group G,
$o(a)$	–	order of the group element a,
$\langle A\rangle$	–	subgroup of a group G generated by a subset $A \subseteq G$,
$\langle a\rangle$	–	subgroup of a group G generated by an element $a \in G$,
$d\vert\, n$	–	d divides n,
ker f	–	kernel of a homomorphism f,
$Z_p[x]$	–	ring of all polynomials with coefficients in the field Z_p,
$Z_p[x]/(f(x))$	–	quotient ring of polynomials modulo $f(x)$.

1. Introduction

If G is a group and there is an element $g \in G$ that $G = \{g^k; k \in Z\}$ then we say that g is a generator of the group G and G is called a cyclic group. Cyclic groups can be finite or infinite.

Example 1.1. A natural number 3 is a generator of the multiplicative group Z_7^*. It is easy to verify because we have for consecutive powers of 3: $3^1\,(\mathrm{mod}7) = 3$, $3^2\,(\mathrm{mod}\ 7) = 2$, $3^3\,(\mathrm{mod}7) = 6$, $3^4\,(\mathrm{mod}7) = 4$, $3^5\,(\mathrm{mod}7) = 5$, $3^6\,(\mathrm{mod}\ 7) = 1$.

Computation of generators in cyclic groups is important for pseudorandom number generators, error correcting codes, and many cryptosystems like:

- ElGamal and Massey-Omura public key ciphers,
- Diffi-Hellmann key agreement protocol,
- DSA, ElGamal and Nyberg-Rueppel digital signatures.

There is no efficient algorithm known for the problem of finding generators for a cyclic group G unless the prime factorization of $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot ... q_r^{k_r}$ is given. Even in this special case we must resort to the use of probabilistic algorithms. On the other hand, factoring an integer in general is believed to be a very difficult problem. So it may not be easy to compute the prime factorization of $\#G$.

In the sequel we describe and analyze two algorithms which compute from the input data (i.e. factorization of the number $\#G$ and definition of group multiplication) generators of a finite cyclic group. In particular (in the case of the multiplicative group Z_p^*) the input data are: a prime p and the factorization of the number $p-1$.

The main aim of the paper is assessment of the average computational complexity of two analyzed algorithms.

2. Probabilistic algorithms for finding a cyclic group generator

A simple probabilistic algorithm (version #1) for finding generators of the finite cyclic group G is the following (see Fig. 1).

*e-mail: t.adamski@ise.pw.edu.pl

```
Algorithm: Probabilistic algorithm for finding generators of the finite cyclic group $G$ (version #1).
Input Data:
1.Definitions of the set $G$ and group multiplication in $G$ and the order $\#G$ of the group $G$
2. The canonical factorization of the number $\#G$ i.e. pairwise different primes $q_1, q_2, .., q_r$, $r$ number of primes and natural numbers $k_1, k_2, ..., k_r \in N$ that $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$
Output Data: A generator $g$ of the cyclic group $G$
1. Choose at random (with the uniform probability distribution on $G$) an element $g \in G$
2. for $i := 1$ to $r$ do; In the loop we verify if for every $i \in \langle 1, r \rangle$, we have $g^{\#G/q_i} \neq 1$.
                          If that condition is true then $g \in G$ is a generator of the group $G$
                          and we finish the algorithm.
        begin
        $a := g^{\#G/q_i}$;
        if $a = 1$ then goto; 1. If the condition: for every $i \in \langle 1, r \rangle$, $g^{\#G/q_i} \neq 1$ is
                              not true then we go to the point 1.
        end
3. write ($a$ is a group generator)
```

Fig. 1. Algorithm for finding a generator g of the finite cyclic group G (version #1)

```
Algorithm: Probabilistic algorithm for finding generators of the finite cyclic group $G$ (version #2)
Input data:
1.Definitions of the set $G$ and group multiplication in $G$ and the order $\#G$ of the group $G$
2. The canonical factorization of the number $\#G$ i.e. pairwise different primes $q_1, q_2, ..., q_r$, $r$ number of primes and natural numbers $k_1$, $k_2, ..., k_r \in N$ that $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$
Output data: $g \in G$ generator of the cyclic group $G$
$g := 1$;
for $i := 1$ to $r$ do begin
        repeat
        Choose at random an element $a \in G$ (with uniform probability distribution); we are seeking an element of the order $q_i^{k_i}$ in the cyclic group $G$.
                $b := a^{(p-1)/(q_i)}(\mathrm{mod} p)$;
        until ($b \neq 1$);
    $c := a^{(p-1)/(q_i^{k_i})}(\mathrm{mod} p)$; we compute an element of the order $q_i^{k_i}$
    $g := g * c(\mathrm{mod} p)$; we compute a product of elements of orders: $q_1^{k_1}, q_2^{k_2}, \ldots, q_r^{k_r}$
    end;
write ("generator =", $g$);
```

Fig. 2. Probabilistic algorithm for finding generators of a cyclic group G (version #2)

A probabilistic algorithm for efficient finding generators of the finite cyclic group G is shown in Fig. 2. The algorithm computes a generator from the input data i.e. a number $\#G$ and the factorization of the number $\#G$.

The simplest algorithm for finding generators in a cyclic group G is the following:

1. Choose at random an element $a \in G$.
2. Compute consecutive powers $a^1, a^2, ..., a^{\#G}$.
3. Verify if $\{a^1, a^2, ..., a^{\#G}\} = G$
4. If this equation is fulfilled the element a is a generator of the cyclic group G, if not go to the point 1.

The above algorithm is simple but the point 2 has the exponential complexity and for large group orders is intractable.

3. Correctness of the algorithm for finding generators in version #1

Correctness of the probabilistic algorithm (in version #1) for finding generators follows immediately from the Theorem 3.1.

Theorem 3.1. Assume G is a finite cyclic group, $r \in N$ and $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$, where $q_1, q_2, ..., q_r$ are primes and $q_1 < q_2 < ... < q_r$, $k_1, k_2, ..., k_r \in N$. If $g \in G$ then g is a generator of the group G, if and only if, for every $i \in \langle 1, r \rangle$ we have $g^{\#G/q_i} \neq 1$.

Proof. 1. Implication $\Rightarrow$. If an element g is a generator of the multiplicative group G then $o(g) = \#G$. Hence for every $i \in \langle 1, r \rangle$ we have $g^{p\#G/p_i} \neq 1$.

2. Implication $\Leftarrow$. If g is not a generator of the group G then there is the smallest exponent $s \in \langle 1, p-2 \rangle$, that $g^s = 1$ (then s is the order of the element g). It follows from the Lagrange theorem that the order of an element of a finite group is always divisor of the group order. Then we have $s | \#G$ i.e. $s | q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$ and there are exponents $k_1', k_2', ..., k_r' \in N \cup \{0\}$ that for every $i \in \langle 1, r \rangle$, $k_i' \leq k_i$ and we have:

$$s = q_1^{k_1'} \cdot q_2^{k_2'} \cdot ... \cdot q_r^{k_r'}.$$

Because $s < \#G$ then there is such $j \in \langle 1, r \rangle$, that $k_j' < k_j$. It follows from the equality $g^s = 1$, that $g^{\#G/p_j} = 1$. Hence it is not true, that for every $i \in \langle 1, r \rangle$ we have $g^{\#G/p_i} \neq 1$, which proves implication left.

4. Correctness of the algorithm for finding generators in version #2

Crucial for the correctness of the algorithm (in version #2, see Fig. 2) described in the Sec. 2 are the following theorems from the group theory.

Theorem 4.1. Every subgroup H of the cyclic group G is a cyclic group.

Proof. Assume $g \in G$ is a generator of the group G. Every element of the group G can be written as g^k, where $k \in Z$. If a subgroup $H = \{1\}$ or $H = G$ then of course, the subgroup H is cyclic. Assume, that the subgroup H is not equal to $H = \{1\}$ then there is in the subgroup H an element different from 1. Denote by n such a smallest natural number, that $g^n \in H$. We prove in the sequel, that g^n is a generator of the group H. Indeed, for an arbitrary $h \in H$, $h \neq 1$ then there is an integer $m \in Z$, $m \neq 0$ that we have $h = g^m$. There are integers k and $r \in \langle 0, n-1 \rangle$ that $m = k \cdot n + r$. Then we have $h = g^m = g^{k \cdot r + r} = (g^n)^k \cdot g^r$ and multiplying both sides of this equality by $(g^n)^{-k}$ we obtain $g^r = (g^n)^{-k} \cdot h$. Because g^n, $h \in H$ then $g^r \in H$. The number r as a remainder from division by r then $0 \leq r < n$ and we have to have $r = 0$. Finally, we can write $h = g^m = g^{n \cdot k} = (g^n)^k$, then g^n is a generator of the group H and H is a cyclic group.

Theorem 4.2. If G is a finite cyclic group and a natural number d is a divisor of the G order then there is exactly one subgroup of the order d of the group G.

Proof. 1. Existence. Denote by $g \in G$ a generator of the group G, then $G = \{g, g^2, ..., g^n\}$, (where $g^n = 1$). If $n = d \cdot m$, then the element g^m is of the order d, then a subgroup generated by the element g^m i.e. $\langle g^m \rangle$ is also a group of the order d.

2. Uniqueness. Assume A and B are different subgroups of the group G with the same order d. Every subgroup of a cyclic group is cyclic then there are elements $a, b \in G$, that $A = \langle a \rangle$ and $B = \langle b \rangle$. The subgroup $\langle a, b \rangle$ of the group G has more than k elements because $\langle a, b \rangle$ contains as subgroups two groups $A = \langle a \rangle$ and $B = \langle b \rangle$, which are different. Every cyclic group is abelian. Hence for every element of the subgroup $\langle a, b \rangle$ there are integers $i, j \in Z$ that this element can be written as $a^i b^j$. Every element $a^i b^j$ raised to the power d gives as a result 1 then there is no element in the subgroup $\langle a, b \rangle$ with the order equal to $\# \langle a, b \rangle$. Then we come to the conclusion that the group $\langle a, b \rangle$ is not cyclic which is not true (see Theorem 4.1). Assumption, that there are two different subgroups of the order d contradicts with the Theorem 4.1.

From the above theorem we obtain immediately the following corollary.

Corollary 4.3. If G is a finite cyclic group, $d \in N$ and $d | \#G$ then there is an element of the order d in the group G.

Theorem 4.4. Assume G is a group and $a \in G$. If for a prime p and a number $e \in N$ we have

$$a^{p^e} = 1, \tag{1}$$

$$a^{p^{e-1}} \neq 1, \tag{2}$$

then the element a has the order p^e.

Proof. If m is the order of the element $a \in G$ then because $a^{p^e} = 1$, we have $m \,| p^e$. Then there is $f \in \langle 0, e \rangle$ that $m = p^f$. If $f < e$ then $a^{p^f} = 1$ and also $a^{p^{e-1}} = 1$ which contradicts with the assumption that $a^{p^{e-1}} \neq 1$. Then $f = e$ and the element a is of the order p^e.

The second crucial fact for the algorithm correctness is given by the following Theorem 4.5.

Theorem 4.5. Assume G is an Abelian group. If $g_1, g_2, ..., g_n \in G$ and s_i is an order of the element g_i for $s_1, s_2, ..., s_n$ are relatively prime in pairs) then the order of the product $g_1 \cdot g_2 \cdot ... \cdot g_n$ is equal to $s_1 \cdot s_2 \cdot ... \cdot s_n$.

Proof. It is sufficient to prove the theorem for $n = 2$ and then apply the finite induction method. Assume 1 is a unit of the group G. If there is an integer k that $(g_1 \cdot g_2)^k = 1$ then $(g_1 \cdot g_2)^{k \cdot s_i} = 1$ for $i = 1, 2$. Therefore we have $g_1^{k \cdot s_2} = 1$ and $g_2^{k \cdot s_1} = 1$. As a result we obtain $s_1 | \, k \cdot s_2$ and $s_2 | \, k \cdot s_1$. Because $GCD(s_1, s_2) = 1$ then we have $s_1 | \, k$, $s_2 | \, k$ and $s_1 s_2 | \, k$. Hence the order of the product $g_1 \cdot g_2$ is also divisible by $s_1 \cdot s_2$. But we have $(g_1 \cdot g_2)^{s_1 \cdot s_2} = 1$ then the order of the product $g_1 \cdot g_2$ is equal to $s_1 \cdot s_2$.

From the Theorems 4.1 and 4.2 it follows, that for the cyclic group G and for every divisor d of the number $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$ there is an element of the order d in particular an element of the order $q_i^{k_i}$.

In the loop **repeat**... **until** for every $k = 1, 2, ..., r$ we are looking for an element of the order $q_i^{k_i}$ (using the method proposed in the Theorem 4.5). Such an element exists in the cyclic group G for every $i = 1, 2, ..., r$ because $q_i^{k_i}$ is a divisor of the number $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot ... q_r^{k_r}$ ($\#G$ is the order of the group G). The loop **repeat**... **until** which detects an element of the order $q_i^{k_i}$ is then finishing with probability 1.

From the Theorem 4.4, we obtain that: if $b_i = a^{(p-1)/(q_i)} \neq 1$ then the element $b_i = a^{(p-1)/(q_i^{k_i})}$ of the group G has the order equal to $q_i^{k_i}$.

The integers $q_1^{k_1}, q_2^{k_2}, ..., q_r^{k_r}$ are relatively prime in pairs then from the Theorem 4.5 it follows, that the element: $\prod_{i=1}^{r} b_i^{(p-1)/(q_i^{k_i})}$ has the order equal to $q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$ which proves the correctness of the algorithm.

5. Number of different generators of the cyclic group

Theorem 5.1. If G is an arbitrary group and $x \in G$ is an element of the order $n \in N$ then for every $k \in Z$ we have: $x^k = 1$ if and only if $n | \, k$.

Proof. 1. For $n = 0$ the thesis is true. Assume then, that $n \geq 2$.

2. $\Leftarrow$ If $n | \, k$ then there is $r \in Z$, that $k = r \cdot n$. Hence $x^k = x^{r \cdot n} = (x^n)^r = 1$.

3. $\Rightarrow$ There are two numbers $q \in Z$ and $r \in \langle 0, n-1 \rangle$, that $k = q \cdot n + r$ (division with remainder). Then from properties of raising to a power in groups we have: $1 = x^k = x^{q \cdot n + r} = (x^n)^q \cdot x^r = x^r$. But r has to be equal to 0, because $r \in \langle 0, n-1 \rangle$ and n is order of x. Hence $k = q \cdot n$ and $n | \, k$.

Theorem 5.2. If G is a group and $x \in G$ is an arbitrary group element of the order $n \in N$ (and $o(x)$ denotes order of the element x i.e. $o(x) = n$) then for every $k \in Z$ we have: $o(x^k) = m$, where $m = \dfrac{n}{GCD(k, n)}$.

Proof. Denote $d = GCD(k, n)$, then there is $r \in Z$, that we have $n = d \cdot m$ and $k = d \cdot r$. Then we have:

$$(x^k)^m = (x^{d \cdot r})^m = x^{d \cdot r \cdot m} = (x^{d \cdot m})^r = (x^n)^r = 1.$$

It follows from the Theorem 5.1 that $x^k = 1$ iff $n | \, k$. We have that $o(x^k) \,| m$. If we denote $o(x^k)$ by s then $1 = (x^k)^s = x^{k \cdot s}$ and we obtain $n \,| (k \cdot s)$ or equivalently $(d \cdot m) \,| (d \cdot r \cdot s)$.

Then $m\,|(r \cdot s)$ and $m\,|s$ because m and r are relatively prime. Finally, we have that $s = m$ i.e. $o(x^k) = m$.

Theorem 5.3. If G is a group and $x \in G$ is an arbitrary group element of the order $n \in N$ then for every $k \in Z$ we have:

$$\langle x \rangle = \langle x^k \rangle \quad \text{if and only if} \quad GCD(k,n) = 1,$$

where $\langle a \rangle$ denotes a subgroup of the group G generated by the element $a \in G$.

Proof. Thesis of the theorem is a direct conclusion from the Theorem 5.2, which says that $o(x^k) = \dfrac{n}{GCD(k,n)}$.

The number of different generators of a cyclic group can be counted with the following theorem.

Theorem 5.4. If G is a finite cyclic group of the order n and g is a generator of G then for every $k \in \langle 1, \#G \rangle$ g^k is a generator of the group G if and only if $GCD(k,n) = 1$.

Proof. Thesis of the theorem is an immediate consequence of the Theorem 5.3. If we assume that $x = g$ then from the Theorem 5.3 we obtain, that g^k is a generator of the group G if and only if $GCD(k,n) = 1$.

It follows from the Theorem 5.4 that the number of all generators of the finite cyclic group $G$is equal to $\varphi(\#G)$, where φ is the Euler function. The quotient $\dfrac{\varphi(\#G)}{\#G}$ allows to assess probability of finding a generator, when we choose elements from G at random with the uniform distribution on G. The following examples illustrate this result.

Example 5.1. Additive group Z_n is a finite cyclic group of the order n then it has $\varphi(n)$ different generators. For example additive group Z_{101} is a finite cyclic group of the order $n = 101$, then it has $\varphi(101) = 100$ different generators and the probability $\dfrac{\varphi(\#Z_n)}{\#Z_n} = \dfrac{100}{101}$.

Example 5.2. If p is a prime then the multiplicative group Z_p^* is a finite cyclic group of the order $\varphi(p) = p-1$ and has $\varphi(\varphi(p)) = \varphi(p-1)$ different generators. For example 101 is a prime and a multiplicative group Z_{101}^* is a finite cyclic group of the order $\varphi(101) = 100$. Then the multiplicative group Z_{101}^* has $\varphi(\varphi(101)) = 40$ different generators and the probability $\dfrac{\varphi(\#Z_{101}^*)}{\#Z_{101}^*} = \dfrac{40}{100}$.

Example 5.3. Assume $n \in N$ and $n \geq 2$. If the multiplicative group Z_n^* is cyclic then it has $\varphi(n)$ elements and $\varphi(\varphi(n))$ generators (called also primitive roots from n) and the probability $\dfrac{\varphi(\#Z_n^*)}{\#Z_n^*} = \dfrac{\varphi(\varphi(n))}{\varphi(n)}$. In particular case if p is an odd prime and k is a natural number then for $n = p^k$ primitive roots from n (i.e. generators of $Z_{p^k}^*$) always exist and $\#Z_{p^k}^* = \varphi(p^k)$. In this case number of primitive roots is even to $\varphi(\varphi(p^k)) = \varphi(p^{k-1} \cdot (p-1))$.

Example 5.4. The multiplicative group F_q^* of the finite field F_q is always a cyclic group of the order $q-1$ and has exactly $\varphi(q-1)$ generators and probability $\dfrac{\varphi(\#F_q^*)}{\#F_q^*} = \dfrac{\varphi(q-1)}{q-1}$.

It is easy to prove that the Euler's function φ is "irregular" i.e. $\limsup\limits_{n\to\infty} \dfrac{\varphi(n)}{n} = 1$ and $\liminf\limits_{n\to\infty} \dfrac{\varphi(n)}{n} = 0$. It means that for arbitrary $N_0 \in N$ and $\varepsilon > 0$ there are $n_1, n_2 \in N$, $n_1, n_2 > N_0$ that $\frac{\varphi(n_1)}{n_1} < \varepsilon$ and $\left|\dfrac{\varphi(n_1)}{n_1} - 1\right| < \varepsilon$. Then the quotient $\dfrac{\varphi(\#G)}{\#G}$ i.e. probability of choosing (at random with uniform probability) a generator from the cyclic group G can be very small or near to 1.

6. The average time complexity of the algorithm for finding group generators in version # 1

In this section we assess the average complexity of the probabilistic algorithm (in version #1) for finding group generators of the cyclic group G (see Fig. 1). We assume that G is a finite cyclic group, $r \in N$ and $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r}$, where $q_1, q_2, ..., q_r$ are primes, $q_1 < q_2 < \ldots < q_r$ and $k_1, k_2, ..., k_r \in N$. Then we know factorization of the order $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r}$ of the group G.

If we compute the power $g^{\#G/q_i}$ for every $i \in \langle 1, r \rangle$ using fast squaring algorithm (see for example Menezes [6]) then time T_{ver} devoted for verification, that for every $i \in \langle 1, r \rangle$, we have $g^{\#G/q_i} \neq 1$ can be assessed with the following inequality

$$T_{ver} \leq r \cdot (\lfloor \log_2 \#G \rfloor + 1)\, \Delta t,$$

where Δt is execution time of the group multiplication and r denotes number of different primes in $\#G$ factorization.

Analyzed algorithm from Fig. 1 realizes a sequence of Bernoulli trials with probability of success equal to $q = \dfrac{\varphi(n)}{n}$. The first success i.e. fulfilling the condition:

$$\text{for every} \quad i \in \langle 1, r \rangle, \quad \text{we have} \quad g^{\#G/q_i} \neq 1,$$

means that a generator g of the group G was found and the algorithm is finished.

In a sequence of Bernoulli trials with probability of success equal to $q \in (0,1)$, a waiting time for the first success is described by a random variable with geometric distribution. A discrete random variable $X : \Omega \to N$ defined on a probabilistic space $(\Omega, \mathcal{M}, P)$ and with values in the set of natural numbers has (from definition) geometric distribution if for every $k \in N$ we have:

$$P(X = k) = (1-q)^{k-1} q,$$

where a parameter $q \in (0,1)$. The mean value and variance of the random variable X are even to:

$$E(X) = \frac{1}{q}, \qquad D^2(X) = \frac{1-q}{q^2}.$$

Then in the case of our algorithm $E(X) = \dfrac{\#G}{\varphi(\#G)}$. Finally, the average time T_{ave} devoted for execution of the whole algorithm can be assessed by the following inequality:

$$T_{ave} \leq E(X) \cdot T_{wer} \leq \frac{\#G}{\varphi(\#G)} \cdot r \cdot (\lfloor \log_2 \#G \rfloor + 1) \cdot \Delta t. \quad (3)$$

As we mentioned the coefficient $E(X) = \dfrac{\#G}{\varphi(\#G)}$ is "irregular" as a function of the group order $\#G$ because we have: $\limsup\limits_{n\to+\infty} \dfrac{\varphi(n)}{n} = 1$ and $\liminf\limits_{n\to+\infty} \dfrac{\varphi(n)}{n} = 0$.

3. The number r of different primes in factorization of the number $\#G$ can be easily assessed by an inequality $r \le \log_2 \#G$ then from (3) we obtain:

$$\begin{aligned} T_{ave} &\le E(X) \cdot T_{wer} \\ &\le \frac{\#G}{\varphi(\#G)} \cdot (\log_2 \#G) \cdot (\lfloor \log_2 \#G \rfloor + 1) \cdot \Delta t. \end{aligned}$$

Hence we can say that the average time complexity of the algorithm is equal to $O(\log_2^2 \#G)$, when we assess number of squarings in the group G. Usually group multiplication has square time bit complexity i.e. $\Delta t = O(\log_2^2 \#G)$ then total average time bit complexity of the algorithm is even to $O(\log_2^4 \#G)$.

7. The average time complexity of the algorithm for finding group generators in version #2

The average computational time complexity is the most important parameter of any algorithm. In the sequel we assess the average time computational complexity of the algorithm from Fig. 2 assuming that the discrete random variable (which describes random choosing of elements from the cyclic group G inside of the loop **repeat**... **until**) has the uniform probability distribution on G.

Theorem 7.1. (on group homomorphism).

Assume we have two groups G_1 and G_2. If $H$is a normal divisor of the group G_1, $f : G_1 \to G_2$ is a homomorphism of the group G_1 onto the group G_2 and the inclusion $H \subseteq \ker f$ is fulfilled then there is exactly one homomorphism $f_0 : G_1/H \to G_2$, that $f_0\kappa = f$, where κ is a canonical homomorphism of the group G_1 onto the quotient group G_1/H. In other words there is exactly one homomorphism $f_0 : G_1/H \to G_2$, that the following diagram

$$\begin{tikzcd} G_1 \arrow[rr, "\kappa"] \arrow[dr, "f"'] & & G_1/H \arrow[dl, "f_0"] \\ & G_2 & \end{tikzcd}$$

is commutative. In case, when $H = \ker f$ the homomorphism f_0 is an isomorphism.

Proof. see. V. Shoup [1], Cz. Bagiński [3], A. Białynicki [10].

Corrollary 7.2. Assume G_1 is a finite group and $H \subseteq G_1$ is a normal divisor of the group G_1. If a random variable $X : \Omega \to G_1$ defined on the probabilistic space $(\Omega, \mathcal{M}, P)$ (with values in a measurable space $(G_1, 2^{G_1})$) has the uniform distribution on G_1 then

1. random variable $\kappa \circ X$ with values in the measurable space $(G_1/H, \mathcal{F})$, (where $\mathcal{F} = 2^{G_1/H}$ and $\kappa : G_1 \to G_1/H$ is the canonical homomorphism onto the quotient group G_1/H) has the uniform distribution on the quotient group G_1/H and for every $k = 1, 2, ..., \dfrac{\#G_1}{\#H}$ we have:
$$P(\kappa \circ X = H_k) = \frac{\#H}{\#G_1},$$
where H_k is k-th coset of the quotient group G_1/H.
2. If additionally we have a finite group G_2 and a homomorphism $f : G_1 \to G_2$ of the group G_1 onto G_2 and $\ker f = H$ then the random variable $f(X)$ has the uniform distribution on G_2.

Proof. The first part of the theorem thesis follows from the fact that every coset of the quotient group G_1/H has the same number of elements (i.e. exactly $\#H$ elements). The second part of the theorem thesis is an immediate consequence of the Theorem 7.1 on the group homomorphism.

The following theorem is useful in the proof of the Theorem 5.4.

Theorem 7.3. If G is a finite group and $a \in G$ then $a^{\#G} = 1$.

Proof. It follows immediately from the Lagrange theorem from group theory.

Theorem 7.4.

Assume G is a finite cyclic group and $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r}$. If we define a function $f_i : G \to G$ given for every $a \in G$ and for every $i \in \{1, 2, ..., r\}$ by the following formula:

$$f_i(a) = a^{\frac{\#G}{q_i}}$$

then f_i is a homomorphism of the group G into the group G and $f_i(G)$ is a cyclic subgroup of the order q_i of the cyclic group G.

Proof. 1. It is easy to verify, that f_i is a homomorphism of the group G into the group G.

Indeed, because G as a cyclic group, G is an abelian group and we have for every $a, b \in G$

$$f_i(a \cdot b) = (a \cdot b)^{\#G/q_i} = a^{\#G/q_i} \cdot b^{\#G/q_i} = f_i(a) \cdot f_i(b).$$

2. A homomorphic image of a group is a group then $f_i(G)$ is a subgroup of the group G. Of course $f_i(G)$ is a cyclic group as a subgroup of a cyclic group.

3. We can in a simple way assess the order of the group $f_i(G)$. If g is a generator of the group G then the following elements belong to the group $f_i(G)$:

$$f_i(g^1), f_i(g^2), ..., f_i(g^{\#G})$$

therefore for consecutive $1, 2, \ldots, q_i$ powers of the generator g we have the following q_i values:

$$g^{\#G/q_i \cdot 1}, g^{\#G/q_i \cdot\cdot 2}, ..., g^{\#G/q_i \cdot q_i},$$

which are pairwise different (because exponents are different and g is the generator of the group G). It is easy to compute powers of $g^{\#G/q_i}$ for consecutive integer exponents $s \in \{q_i + 1, q_i + 2, ..., \#G\}$. For every $s \in \{q_i + 1, q_i + 2, ..., \#G\}$

there are such $k \in N$, and $z \in \{0,1,...,q_i-1\rangle$, that we have $s = z + k \cdot q_i$.

Because $g^{\#G} = 1$ (see Theorem 7.3) then we have $g^{\#G/q_i \cdot s} = g^{\#G/q_i \cdot (k \cdot q_i + z)} = g^{\#G/q_i \cdot z}$ and all powers of g for exponents $s \in \{q_i+1, q_i+2, ..., \#G\}$ belong to the following q_i element set

$$\{g^{\#G/q_i \cdot 1}, g^{\#G/q_i \cdot \cdot 2}, ..., g^{\#G/q_i \cdot q_i}\}.$$

In short, $f_i(G)$ is a subgroup of the order q_i of the cyclic group G.

Assume now, that $G_1 = G$ and $G_2 = f_i(G)$, and take as a surjective homomorphism of the group G_1 onto G_2 the homomorphism f_i. If a discrete random variable X has the uniform probability distribution on G then from the Corollary 7.2 we obtain, that the random variable $f_i(X)$ has the uniform distribution on the subgroup $G_2 \subseteq G$ of the order q_i.

Then for every $i \in \langle 1, r\rangle$ probability that after the first pass of the internal loop **repeat ... until** we go out from the internal loop (probability of success) is equal to

$$P(f_i(X) \neq 1) = P(X^{\#G/q_i} \neq 1) = \frac{q_i - 1}{q_i}.$$

Assume $(X_j)_{j=1}^{\infty}$ is a sequence of independent discrete random variables with values in the cyclic group G and with uniform probability distribution on G (consecutive draws of the element $a \in G$ are independent). In other words $(X_j)_{j=1}^{\infty}$ is a discrete white noise with uniform probability distribution. It is a stochastic process describing draw of consecutive elements $a \in G$ inside of the loop **repeat** ... **until**.

Conditions of the going out from the i-th loop **repeat** ... **until** are described by the stochastic process:

$$(\operatorname{sgn}(f_i(X_j) - 1))_{j=1}^{\infty}. \tag{4}$$

If $\operatorname{sgn}(f_i(X_j) - 1) = 0$ then the condition of going out from the loop **repeat** ... **until** is not fulfilled (lack of success) and we have

$$P(\operatorname{sgn}(f_i(X_j) - 1) = 0) = \frac{1}{q_i}.$$

If $\operatorname{sgn}(f_i(X_j) - 1) = 1$ then the condition of going out from the loop **repeat** ... **until** (success) is fulfilled and we have

$$P(\operatorname{sgn}(f_i(X_j) - 1) = 1) = \frac{q_i - 1}{q_i}.$$

In the algotithm we go out from the loop **repeat** ... **until** immediately after the output condition is fullfilled (the first success).

The stochastic process (4) is an infinite sequence of Bernoulli trials with probability of success equal to $\frac{q_i - 1}{q_i}$.

Denote by Y_i a function defined on the probabilistic space $(\Omega, \mathcal{M}, P)$ and with values in the set $N \cup \{+\infty\}$ in the following way: for every $k \in N$ and every $\omega \in \Omega$ $Y_i(\omega) = k$ if and only if the trajectory $(\operatorname{sgn}(f_i(X_j(\omega)) - 1))_{j=1}^{\infty}$ of the process $(\operatorname{sgn}(f_i(X_j) - 1))_{j=1}^{\infty}$ is the following: $\operatorname{sgn}(f_i(X_1(\omega)) - 1) = 0$, $\operatorname{sgn}(f_i(X_2(\omega)) - 1) = 0, ..., \operatorname{sgn}(f_i(X_{k-1}(\omega) - 1) = 0$, $\operatorname{sgn}(f_i(X_k(\omega)) - 1) = 1$ i.e. the first success (the first 1 in the sequence) occurs exactly in the k-th trial and $Y_i(\omega) = +\infty$ if and only if the trajectory $(\operatorname{sgn}(f_i(X_j(\omega)) - 1))_{j=1}^{\infty}$ is a sequence of zeroes.

It can be easily proved that the defined on the probabilistic space $(\Omega, \mathcal{M}, P)$ as above function Y_i is a random variable. The random variable Y_i has the geometric distribution with the probability of success equal to $\frac{q_i - 1}{q_i}$ i.e. for every $k \in N$ we have

$$P(Y_i = k) = \left(\frac{1}{q_i}\right)^{k-1} \cdot \frac{q_i - 1}{q_i}.$$

The random variable $Y_i = k$, if and only if, from the i-th execution of the internal loop **repeat** ... **until** we go out exactly after the k-th draw of a inside of this loop.

The expected value of the random variable Y_i (i.e the average time till going out from the internal loop during the i-th pass of the external loop) is equal to $\frac{q_i}{q_i - 1}$ then we have:

$$E(Y_i) = \frac{q_i}{q_i - 1} \leq 2$$

and the variance:

$$D^2(Y_i) = \frac{q_i}{(q_i - 1)^2} \leq 2.$$

Then the average time of execution of r **repeat** ... **until** loops is described by a random variable

$$Z = Y_1 + Y_2 + ... + Y_r$$

and is even to

$$E(Z) = \sum_{i=1}^{r} E(Y_i) = \sum_{i=1}^{r} \frac{q_i}{q_i - 1} \leq 2 \cdot r. \tag{5}$$

Random variables $Y_1, Y_2, \ldots, Y_r$ are independent then the variance of the random variable $Z = Y_1 + Y_2 + ... + Y_r$ is even to:

$$D^2(Z) = D^2(Y_1 + Y_2 + ... + Y_r) = \sum_{i=1}^{r} D^2(Y_i)$$
$$= \sum_{i=1}^{r} \frac{q_i}{(q_i - 1)^2}.$$

If $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot ...q_r^{k_r}$ is the canonical factorization of $\#G$ then we have

$$\log_2 \#G \geq \log_2(q_1^{k_1} \cdot q_2^{k_2} \cdot ...q_r^{k_r})$$
$$= \sum_{i=1}^{r} k_i \log_2 q_i \geq r$$

and $r < \log_2 \#G$. Then because $r \leq \log_2 p$, we obtain from (5)

$$E(Z) = \sum_{i=1}^{r} E(Y_i) \leq 2 \cdot r \leq 2 \log_2(\#G).$$

Therefore, if we use inside of the loop the well-known fast algorithm for raising to a power in the group (having complexity $O(\log_2 \#G)$) then the average number of group multiplications inside of all r loops **repeat** ... **until** has complexity $O((\log_2 \#G)^2)$ (we assess number of group multiplications). We treat $\log_2 \#G$ as the dimension of the input data.

In similar way we can assess number of group multiplications on the outside of the loops **repeat** ... **until**. If we use the fast algorithm for raising to a power in the group, this number can be also assessed by $O((\log_2 \#G)^2)$. Finally, we obtain that the average computational complexity of the analyzed algorithm is even to $O((\log_2 \#G)^2)$.

In many particular cases the group multiplication has bit computational complexity even to $O((\log_2 \#G)^2)$ then the average bit complexity of the analyzed algorithm is even to $O((\log_2 \#G)^4)$.

8. Generators of the multiplicative groups of finite fields

In this chapter we consider two particular cases of cyclic groups frequently used in cryptography.

Assume p is a prime and $k \in N$ then from the following Theorem 8.1 we obtain that multiplicative groups Z_p^* (with multiplication modulo p as a group operation) and $GF^*(p^k)$ (with multiplication of polynomials modulo an irreducible polynomial of the degree k as a group operation) are cyclic groups.

Theorem 8.1. If p is a prime and $k \in N$ then every multiplicative group $GF^*(p^k)$ of the finite field $GF(p^k)$ is a cyclic group of the order $p^k - 1$.

Proof. The multiplicative group $GF^*(p^k) = GF(p^k)\backslash\{0\}$ then the order of the group $GF^*(p^k)$ is equal to $p^k - 1$. The fact that $GF^*(p^k)$ is cyclic is proved for example in N. Koblitz [2], C. Bagiński [3].

The described algorithms can be applied in both cases. For Z_p^* we have to know factorization $p - 1 = q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$ and assume to avoid triviality that p is an odd prime.

For $GF^*(p^k)$ we have to know factorization $p^k - 1 = q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$ and assume to avoid triviality that $p^k - 1 \geq 2$.

In cryptography our goal is to construct a large prime p together with a generator g for Z_p^*. The specific prime p is not so important. Then if we do not know the prime factorization of the number $p - 1$ we can:

1. generate a random factored number n in some range $n = q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$ and next;
2. test $n + 1$ for primality (using for example the well-known Miller-Rabin algorithm);
3. if $n + 1$ is not a prime we repeat the point 1, if $n + 1$ is a prime we have (denoting $n + 1$ by p) a prime p and the factorization of $p - 1$.

A probabilistic algorithm (in version # 2) for efficient finding generators of the multiplicative group Z_p^* is shown in the Fig. 3. The algorithm computes a primitive root from p (i.e. a generator of the group Z_p^*) from the input data i.e. a prime p and the factorization of the number $p - 1$.

```
Algorithm: Probabilistic algorithm for primitive root mod $p$ generation
Input data:
1. An odd prime $p$
2.The canonical factorization of the number $p - 1$ i.e. $r \in N$, different in pairs primes $q_1, q_2, .., q_r$ and natural numbers $k_1, k_2, ..., k_r \in N$ that $p - 1 = q_1^{k_1} \cdot q_2^{k_2} \cdot ... \cdot q_r^{k_r}$
Output data: $g \in Z_p^*$ generator of the multiplicative group $Z_p^*$

g := 1;
for $i := 1$ to $r$ do begin
    repeat
        Choose at random a number $a \in Z_p^*$ (with uniform probability distribution); we are seeking an element of the order $q_i^{k_i}$ in the group $Z_p^*$.
        $b := a^{p-1/q_i}(\mathrm{mod} p)$;
    until $(b \neq 1)$;
    $c := a^{p-1/q_i^{k_i}}(\mathrm{mod} p)$; we compute an element of the order $q_i^{k_i}$
    $g := g * c(\mathrm{mod} p)$; we compute a product of elements of orders: $q_1^{k_1}, q_2^{k_2}, \ldots, q_r^{k_r}$
end;
write ("generator =", $g$);
```

Fig. 3. Probabilistic algorithm for primitive root modulo p computation (version #2)

9. Primitive roots modulo *n*

The multiplicative group Z_n^* of the ring Z_n, where $n \in N$, $n \geq 2$ is defined as a set $Z_n^* \overset{df}{=} \{k \in N; k < n, GCD(n, m) = 1\}$ along with the multiplication modulo n as a group operation.

A primitive root g from a natural number n (or as we say primitive root modulo n) is (from definition) a generator g of the multiplicative group Z_n^* where $n \in N$, $n \geq 2$.

The multiplicative group Z_n^* (for $n \in N$, $n \geq 2$) can be cyclic or not. Then the primitive root from n exists or does not exist. The problem is explained in detail by the following three theorems. The first describes a structure of the multiplicative group Z_n^*, the second says when $Z_{p^k}^*$ is a cyclic group, the last answers the question when primitive roots exist.

Theorem 9.1.

If $n = p_1^{k_1} \cdot p_2^{k_2} \cdot ... p_r^{k_r}$, where $p_1, p_2, .., p_r$ are different in pairs primes and $k_1, k_2, .., k_r \in N$ (i.e. $n = p_1^{k_1} \cdot p_2^{k_2} \cdot ... p_r^{k_r}$ is a canonical factorization of n) then the multiplicative group Z_n^* is isomorphic with the direct product of groups $Z_{p_1^{k_1}}^* \times Z_{p_2^{k_2}}^* \times ... \times Z_{p_r^{k_r}}^*$ i.e.

$$Z_n^* \cong Z_{p_1^{k_1}}^* \times Z_{p_2^{k_2}}^* \times ... \times Z_{p_r^{k_r}}^*.$$

Proof. The theorem is a direct conclusion from the Chinese remainder theorem.

Then to know the structure of the multiplicative group Z_n^* it is sufficient to know the structure of of the multiplicative group $Z_{p^k}^*$.

Theorem 9.2.

1.If p is an odd prime then the multiplicative group $Z^*_{p^k}$ is cyclic for every $k \in N$.

2. Multiplicative groups Z^*_2 and Z^*_4 are cyclic and their order is equal appropriately to 1 and 2.

3. The multiplicative group $Z^*_{2^k}$ for $k \geq 3$ is isomorphic with a direct product of a cyclic group of the order 2 and the cyclic group of the order 2^{k-2} i.e. $Z^*_{2^k} \cong C^*_2 \times C^*_{2^{k-2}}$.

Proof. see. [1, 2].

From the Theorem 9.1 and 9.2. we obtain the following Theorem 9.3 on existence of primitive roots modulo n.

Theorem 9.3. (on existence of primitive roots modulo n)

The multiplicative group Z^*_n (where $n \in N$, $n \geq 2$) is a cyclic group if and only if $n = 2, 4, p^k, 2p^k$, where p is an odd prime jest and $k \in N$.

Proof. see for example W. Narkiewicz [8].

The analyzed algorithms for finding generators works correctly for every n for which primitive root exists.

So far it is not solved the problem of the least primitive root for a prime p. If we denote by $r(p)$ the least primitive root for a prime p then we can assess $r(p)$ with the following inequality: $r(p) < p^{\frac{1}{4}+\varepsilon}$ for arbitrary $\varepsilon > 0$. More details concerning assessment of the least primitive root for a natural number n can be found in monographs [4] (W. Narkiewicz) and [10] (A. Paszkiewicz).

10. Primitive polynomials

Described in the section 2 algorithms in a version for $G = GF(p^n)$ after small modifications can be applied to verify if a given polynomial $f(x) \in Z_p[x]$ is a primitive polynomial. The primitive polynomial $f(x) \in Z_p[x]$ it is (from definition) such a irreducible polynomial in the ring $Z_p[x]$ that a polynomial $g(x) = x$, where $(g(x) \in Z_p[x])$ is a generator of the multiplicative group of the field $Z_p[x]/(f(x))$ (or in other words an element of the order $p^m - 1$ in the multiplicative group of the field $Z_p[x]/(f(x))$). Primitive polynomials are used for example in LFSR (Linear Feedback Shift Register) to design LFSR with maximal number of possible states.

The following Theorem 10.1 is a polynomial version of the Theorem 3.1 proved in the Sec. 3.

Theorem 10.1.

Assume p is a prime and $p^m - 1 = q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r}$, where $m \in N$, $q_1 < q_2 < \ldots < q_r$ are primes and $k_1, k_2, \ldots, k_r \in N$. If $f(x) \in Z_p[x]$ is an irreducible polynomial of the degree equal to m then $f(x)$ is a primitive polynomial, if and only if, for every $i \in \langle 1, r\rangle$: $x^{p^m-1/q_i}(\bmod f(x)) \neq 1$.

Proof. The thesis of the theorem follows immediately from the Theorem 3.1 and the Theorem 8.1.

The algorithm which verifies if a polynomial $f(x) \in Z_p[x]$ is primitive is the following.

Algorithm: Verification if a polynomial $f(x) \in Z_p[x]$ is primitive

Input data: 1. A prime p, $m \in N$ and factorization $p^m - 1 = q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r}$, where $q_1 < q_2 < \ldots < q_r$ are primes and $k_1, k_2, \ldots, k_r \in N$

2. Irreducible polynomial $f(x) \in Z_p[x]$ of the degree equal to m

Output data: answer if $f(x) \in Z_p[x]$ is a primitive polynomial

```
for $i := 1$ to $r$ do begin
$l(x) := x^{p^m-1/q_i}(\bmod f(x))$;
if $l(x) := 1$ then begin write ('$f(x)$) is not a primitive polynomial'); goto end_label end
                 end
write ('$f(x)$) is a primitive polynomial');
end_label:
```

Fig. 4. Algorithm for verification if an irreducible polynomial $f(x) \in Z_p[x]$ is primitive

11. Conclusions and comments

In the paper correctness of two probabilistic algorithms for group generator finding were proved and their average computational complexity was assessed as equal to $O((\log_2 \#G)^2)$. It means that the average computational complexity of analyzed algorithms is in the polynomial class. Both considered in the paper algorithms can be also in natural way parallelized.

If we know only a partial prime factorization of the number $\#G$ i.e. we have $\#G = q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r \cdot} \cdot a$, where $q_1, q_2, \ldots, q_r$ are pairwise different primes and $k_1, k_2, \ldots, k_r$, $a \in N$ then both algorithms after small modifications can be used to find an element of the order $q_1^{k_1} \cdot q_2^{k_2} \cdot \ldots \cdot q_r^{k_r \cdot}$. Frequently for cryptographic algorithms we seek only elements of sufficiently large order and we do not need to have obligatory a generator. The modified algorithm #2 works correctly under assumption that the finite group G is abelian (see Theorem 4.5). The algorithm #1 works correctly also for noncommutative groups.

REFERENCES

[1] V. Shoup, *A Computational Introduction to Number Theory and Algebra*, University Press, Cambridge, 2008.

[2] N. Koblitz, *A Course in Number Theory and Cryptography*, Springer, New York, 1994.

[3] C. Bagiński, *Introduction to Group Theory*, SCRIPT, Warszawa, 2002, (in Polish).

[4] W. Narkiewicz, *Number Theory*, PWN, Warszawa, 1990, (in Polish).

[5] A. Menezes, P. Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press Inc., London, 1997, (http://cacr.math.uwaterloo.ca/hac).

[6] D. Hankerson, A. Menezes, and S. Vanstone, *Guide to Elliptic Curve Cryptography*, Springer, New York, 2004.

[7] S. Yan, *Number Theory for Computing*, Springer, Berlin, 2002.

[8] J. Pieprzyk, T. Hardjono, and J. Seberry, *Fundamentals of Computer Security*, Springer, Berlin, 2003.

[9] A. Białynicki-Birula, *Algebra*, PWN, Warszawa 2009, (in Polish).