ISSN 0027-1322, Moscow University Mathematics Bulletin, 2019, Vol. 74, No. 1, pp. 5–13. © Allerton Press, Inc., 2019. Russian Text © S.B. Gashkov, I.B. Gashkov, and A.B. Frolov, 2019, published in Vestnik Moskovskogo Universiteta, Matematika. Mekhanika, 2019, Vol. 74, No. 1, pp. 7–15.

The Complexity of Solving Low Degree Equations over Ring of Integers and Residue Rings

S. B. Gashkov [a,*], **I. B. Gashkov** [b,**], **and A. B. Frolov** [c,***]

[a] *Moscow State University, Faculty of Mechanics and Mathematics, Leninskie Gory, Moscow, 119991 Russia,* [*] *e-mail: sbgashkov@gmail.com*
[b] *Karlstads Universitet, Universitetsgatan 2, Karlstad, 65188 Sweden,* [**] *e-mail: igor.gachkov@kau.se*
[c] *National Research University "Moscow Power Engineering Institute" Krasnokazarmennaya 14, Moscow, 111250 Russia,* [***] *e-mail: abfrolov@gmail.com.*

Received March 14, 2018

Abstract—It is proved that for an arbitrary polynomial $f(x) \in \mathbb{Z}_{p^n}[X]$ of degree d the Boolean complexity of calculation of one its root (if it exists) equals $O(dM(n\lambda(p)))$ for a fixed prime p and growing n, where $\lambda(p) = \lceil \log_2 p \rceil$, and $M(n)$ is the Boolean complexity of multiplication of two binary n-bit numbers. Given the known decomposition of this number into prime factors $n = m_1 \ldots m_k$, $m_i = p_i^{n_i}$, $i = 1, \ldots, k$, with fixed k and primes p_i, $i = 1, \ldots, k$, and growing n, the Boolean complexity of calculation of one of solutions to the comparison $f(x) = 0 \bmod n$ equals $O(dM(\lambda(n)))$. In particular, the same estimate is obtained for calculation of one root of any given degree in the residue ring $\mathbb{Z}_m$. As a corollary, it is proved that the Boolean complexity of calculation of integer roots of a polynomial $f(x)$ is equal to $O_d(M(n))$, where $f(x) = a_d x^d + a_{d-1}x^{d-1} + \ldots + a_0$, $a_i \in \mathbb{Z}$, $|a_i| < 2^n$, $i = 0, \ldots, d$.

DOI: 10.3103/S0027132219010029

INTRODUCTION

Algorithms for solving equations over rings of residues are applied in encoding and cryptography. At the end of XIX and beginning of XX century, A. Tonelli and M. Cipolla proposed algorithm for extracting roots over fields formed by residues modulo a prime number (see, e.g., [1–3]). Later on, these algorithms were repeatedly reopened. These algorithms are probabilistic, but under assumption of validity of some number-theoretic hypothesis the Tonelli algorithm has a deterministic version of polynomial complexity (see, e.g., [4]). If it is necessary to apply the algorithm many times over the same field (for example, for decoding schemes), then it is possible to neglect the complexity (which is uniquely determined by the field) of preliminary computations (whose results nevertheless can be used for the roots extracting scheme construction over this field), and the root extracting algorithms become deterministic.

For the Boolean complexity of Cipolla's algorithm there is the estimate $O(\log_2 p)M(\log_2 p)$, where $M(n)$ denotes the Boolean complexity of multiplication of binary n-bit numbers.

It is known, see [5, 6], that $M(n) = \psi(n)n \log n$, where $\psi(n)$ is a function growing slower than any iteration of the logarithm. For medium values of n Shenhage–Strassen's and Pollard's algorithms work better. For small n, Karatsuba's and Toom's methods were preferred (see, e.g., [1–3]).

COMPLEXITY OF SOLVING ALGEBRAIC EQUATIONS OVER RINGS OF RESIDUES

Solving equations over some rings $\mathbb{Z}_m$, we obtain the complexity estimate $O(M(\log_2 m))$. In particular, root extracting in such rings can be performed with the complexity $O(M(\log_2 m))$.

Consider the case $m = p^n$, where p is prime. The following result is useful for further considerations.

Lemma 1. *The multiplicative inverse can be calculated in the ring $\mathbb{Z}_{p^n}$ with the Boolean complexity $O(M(\lambda(p))\lambda(\lambda(p)) + M(n\lambda(p)))$, where $\lambda(p) = \lceil \log_2 p \rceil$.*

Proof. Apply an analogue of the well-known series inversion method (that can be also considered as using the Newton tangents method or the Karatsuba bisection method). Let we have to solve the equation $ax = 1$, $a, x \in \mathbb{Z}_{p^n}$, or (which is the same) to solve the comparison $ax = 1 \bmod p^n$, $a, x \in \{1, \ldots, p^n - 1\}$, where $(a, p) = 1$ (i.e., a is not divisible by p, otherwise there is no solution). As is known, for $a, x \in \{1, \ldots, p - 1\}$ the solution to the comparison $ax = 1 \bmod p$ can be obtained with the Boolean complexity $I(1) = O(M(\lambda(p))\lambda(\lambda(p)))$ by applying the fast extended Euclidean–Shenhage algorithm (see, e.g., [7]). Let

$n_1 = \lceil n/2 \rceil$. Assume $a_1 = a \bmod p^{n_1}$. As is known, the Boolean complexity of calculation of a_1 equals the complexity of the division with a remainder of a by p^{n_1}, i.e., to $O(M(n\lambda(p)))$. Suppose equaition $a_1 x = 1$, $a_1, x \in \mathbb{Z}_{p^{n_1}}$, can be solved with the complexity $I(n_1)$ and let $x_1 \in \{1, \ldots, p^{n_1} - 1\}$ be its solution. Search for the solution to the comparison $ax = 1 \bmod p^n$, $a, x \in \{1, \ldots, p^n - 1\}$, in the form $x = x_1 + p^{n_1} y$, where $ax_1 = a_1 x_1 = 1 \bmod p^{n_1}$. Since $(ax_1 - 1)^2 = (a_1 x_1 - 1)^2 = 0 \bmod p^{2n_1}$, then $-(ax_1 - 1)^2 = 0 \bmod p^n$, $a(2x_1 - ax_1^2) = 1 \bmod p^n$ and hence $x = 2x_1 - ax_1^2 \bmod p^n$. The computation of x_1^2 is executed with the complexity $M(n_1\lambda(p))$, the value $x_1^2 \bmod p^n$ can be found with the complexity $O(M(n\lambda(p)))$, and then $a(x_1^2 \bmod p^n)$ can be found with the complexity $M(n\lambda(p))$ (see, e.g., [8]). Therefore, $a(x_1^2 \bmod p^n) \bmod p^n$ is calculated with the complexity $O(M(n\lambda(p)))$ and $2x_1 - ax_1^2 \bmod p^n$ can be found with the complexity

$$O(M(n\lambda(p))) + O(n\lambda(p)) = O(M(n\lambda(p))).$$

Therefore, the complexity of solving the comparison $ax = 1 \bmod p^n$ can be estimated as

$$I(n) \le I(n_1) + O(M(n\lambda(p))) = I(\lceil n/2 \rceil) + O(M(n\lambda(p))).$$

Estimating $I(\lceil n/2 \rceil)$ from above in a similar way and using the inequalities $M(a+b) \le M(a) + O(ab)$, where $a > b$, we get

$$M((n+1)\lambda(p)) < M(n\lambda(p)) + O(n\lambda^2(p)), \quad M(n) \ge n,$$

and for fixed p and increasing n we obtain the inequality

$$I(n) \le I(\lceil n/4 \rceil) + O(M(\lceil n/2 \rceil \lambda(p)) + M(n\lambda(p))) = I(\lfloor n/4 \rfloor) + O(M(n\lambda(p))).$$

Proceeding in the same way, we get the estimate

$$I(n) \le I(1) + O(M(n\lambda(p)) + M(\lfloor n/2 \rfloor \lambda(p)) + M(\lfloor n/4 \rfloor \lambda(p)) + \ldots + M(\lambda(p))).$$

Applying the inequality $2M(n) \le M(2n)$ (which is always assumed to be valid under similar circumstances), we have

$$I(n) \le I(1) + O(M(n\lambda(p))) = O(M(\lambda(p))\lambda(\lambda(p)) + M(n\lambda(p))). \qquad \square$$

Example 1. Solve the comparison $4x = 1 \bmod 3^2$. First we solve the comparison $4x_1 = 1 \bmod 3$. Evidently, the solution is $x_1 = 1 \bmod 3$. Further, search for the solution on the form $x = x_1 + 3y = 1 + 3y$, where $y \in \{0, 1, 2\}$. Since $(4x_1 - 1)^2 = 0 \bmod 3^2$, then $4(2x_1 - 4x_1^2) = 1 \bmod 9$ and hence $x = 2x_1 - 4x_1^2 = 2 - 4 = -2 = 7 \bmod 9$.

Theorem 1. *For an arbitrary polynomial $f(x) \in \mathbb{Z}_{p^n}[X]$ of degree d the Boolean complexity of calculation of one its root (if it exists) is equal to $O(dM(n\lambda(p)) + M(\lambda(p))\lambda(\lambda(p)) \log_2 n + dpM(\lambda(p)))$. For a fixed p and an increasing n this estimate becomes $O(dM(n\lambda(p))$. In particular, the same estimate is valid for extracting the roots of any degree*[4].

Proof. Let $n_1 = \lceil n/2 \rceil$. Construct the polynomial $f_1(x) \in \mathbb{Z}_{p^{n_1}}[X]$ by replacing the coefficients of the polynomial f by their values modulo p^{n_1}. Consider the comparison $f(x) = 0 \bmod p^{n_1}$ equivalent to the comparison $f_1(x) = 0 \bmod p^{n_1}$. Any solution x of the comparison $f(x) = 0 \bmod p^n$ considered modulo p^{n_1} is a solution to the comparison $f(x) = 0 \bmod p^{n_1}$ and to the equivalent comparison $f_1(x) = 0 \bmod p^{n_1}$. By $F(n_1)$ we denote the complexity of calculation of one such solution $x_1 \in \mathbb{Z}_{p^{n_1}}$ of the comparison $f_1(x) = 0 \bmod p^{n_1}$. Since $x = x_1 \bmod p^{n_1}$, then $x = x_1 + p^{n_1} y$, $y \in \mathbb{Z}_{p^{n-n_1}}$. Therefore, to find a solution $x = x_1 + p^{n_1} y$, it suffices to find $y \in \mathbb{Z}_{p^{n-n_1}}$. As is known, Taylor's formula for calculation of $f(x) = f(x_1 + p^{n_1} y)$ can be written in the form

$$f(x_1 + p^{n_1} y) = f(x_1) + f^{[1]}(x_1) p^{n_1} y + f^{[2]}(x_1)(p^{n_1} y)^2 + \ldots + f^{[d]}(x_1)(p^{n_1} y)^d,$$

where $d = \deg f(x)$, the Hasse–Teichmuller derivative $f^{[k]}(x)$ is defined by the equality $k! f^{[k]}(x) = f^{(k)}(x)$, where $f^{(k)}(x)$ is the usual kth order derivative. For a monomial ax^m, $m \ge k$, the kth order Hasse–Teichmuller derivative is defined as $a\binom{m}{k} x^{m-k}$, and for $m < k$ it is equal to zero. For an arbitrary polynomial over the ring $\mathbb{Z}_{p^n}$ this derivative is evidently defined by linearity. Therefore, the comparison $f(x_1 + p^{n_1} y) = 0 \bmod p^n$ is equivalent to the comparison $f(x_1) + f'(x_1) p^{n_1} y = 0 \bmod p^n$. Since $f(x_1) \in \mathbb{Z}_{p^n}$, $f(x_1) = f_1(x_1) = 0 \bmod p^{n_1}$, then $f(x_1) = p^{n_1} z$, $z \in \mathbb{Z}_{p^{n-n_1}}$, $z = f(x_1)/p^{n_1}$. Then the comparison $f(x_1) + f'(x_1) p^{n_1} y = 0 \bmod p^n$ is equivalent to the comparison $z + f'(x_1) y = 0 \bmod p^{n-n_1}$, where $z = f(x_1)/p^{n_1}$, $f'(x_1) \in \mathbb{Z}_{p^{n_1}}$. Since $n - n_1 \le n_1$, then in this comparison one can take $f'(x_1) \bmod p^{n-n_1}$ instead of $f'(x_1)$. If $f'(x_1) \bmod p^{n-n_1} = 0$, then

[4]The number of roots of degree d over the rings $\mathbb{Z}_{p^n}$ could be more than d for $n > 1$ and $p \mid n$.

this comparison has no solutions for $z \neq 0 \bmod p^{n-n_1}$ and has p^{n-n_1} solutions for $z = 0 \bmod p^{n-n_1}$ (any $y \in \mathbb{Z}_{p^{n-n_1}}$ is a solution). Since the coefficients of the polynomials f and f_1 are the same modulo p^{n_1}, then the coefficients of their derivatives $f'(x)$ and $f_1'(x)$ are the same modulo p^{n_1} and hence they are the same modulo p^{n-n_1}. Therefore, $f'(x_1) \bmod p^{n-n_1}$ can be calculated by the formula $a = f_1'(x_1 \bmod p^{n-n_1}) \bmod p^{n-n_1}$. The latter implies that for $a \neq 0 \bmod p$ we have $y = -z/a \bmod p^{n-n_1}$ and so for $x = x_1 \bmod p^{n_1}$ there exists a unique solution $x = x_1 + p^{n_1}y \in \mathbb{Z}_{p^n}$. If

$$a = 0 \bmod p^k,\ a \neq 0 \bmod p^{k+1},\ z = 0 \bmod p^l,\ z \neq 0 \bmod p^{l+1},$$

then for $l \geq k$ we have $y = (-z/p^k)/(a/p^k) \bmod p^{n-n_1}$, and for $l < k$ there are no solutions (because $z + ya = 0 \bmod p^l$ and $z + ya \neq 0 \bmod p^{l+1}$). The Boolean complexity of the calculation of $z = f(x_1)/p^{n_1}$ is equal to $O(dM(n\lambda(p)))$ providing $f(x_1) \in \mathbb{Z}_{p^n}$ is calculated by Horner's method (the order of complexity of the division of $f(x_1) \bmod p^n$ by p^{n_1} is $O(M(n\lambda(p))))$. Similarly, the complexity of calculation of $a = f_1'(x_1 \bmod p^{n-n_1}) \bmod p^{n-n_1}$ is estimated as $O((d-1)M((n-n_1)\lambda(p)))$. The complexity of calculation of $1/(a/p^k) \bmod p^{n-n_1}$ is estimated as

$$\begin{aligned} I(n-n_1) + O(M((n-n_1)\lambda(p))) &= O(M(\lambda(p))\lambda(\lambda(p)) + M(n\lambda(p)) + M((n-n_1)\lambda(p))) \\ &= O(M(\lambda(p))\lambda(\lambda(p)) + M(n\lambda(p))). \end{aligned}$$

The complexity of calculation of $y = (-z/p^k)/(a/p^k) \bmod p^{n-n_1}$ equals $M((n-n_1)\lambda(p))$. At last, the complexity of calculation of the root $x = x_1 + p^{n_1}y$ is equal to $M((n-n_1)\lambda(p)) + O(n\lambda(p))$. The overall estimate of the complexity of all calculations is equal to

$$F(n) = F(n_1) + O(dM(n\lambda(p)) + n\lambda(p) + M(\lambda(p))\lambda(\lambda(p))).$$

Applying this recurrent estimate to estimate the sequence $F(n_1), F(n_2), \dots, F(n_k)$, where $n_1 = \lceil n/2 \rceil$, $n_2 = \lceil n/4 \rceil, \dots, n_k = \lceil n/2^k \rceil = 1$, $2^{k-1} < n \leq 2^k$, and using the equalities

$$M(\lceil n/2 \rceil) = M(\lfloor n/2 \rfloor) + O(n),\ F(1) = O(dpM(\lambda(p))),$$

we get

$$F(n) = O(dM(n\lambda(p)) + M(\lambda(p))\lambda(\lambda(p))\log_2 n + dpM(\lambda(p))).$$

(In the field $\mathbb{Z}_p$, the roots can be found by direct enumeration using Horner's method to calculate the values of the polynomial. For some polynomials the estimate can be improved, for example, for $f(x) = x^d$ in some cases the roots can be found with the complexity $O(\lambda(p)M(\lambda(p)))$, and one can use the Cipolla algorithm for square root extracting.) □

Example 2. Solve the comparison $f(x) = x^3 + x + 1 = 0 \bmod 3^4$. As in Example 1, for the sake of brevity, all calculations are performed not in the binary system, but in the decimal system (of course, in the ternary system the calculations would become simpler, but in this case one has to transform the final result into the binary system, but Example 2 illustrates the theorem estimating the Boolean complexity, i.e., it is supposed that just the binary system is used. We omit these calculations giving only the final results for short). The comparison $x^3 + x + 1 = 0 \bmod 3$ has the root $x_1 = 1 \bmod 3$. Search for solutions to the comparison $x^3 + x + 1 = 0 \bmod 3^2$ in the form $x_2 = x_1 + 3y = 1 + 3y \bmod 3^2$, $y \in \mathbb{Z}_3$. In this case,

$$f(x_2) = f(1+3y) = f(1) + f'(1)3y = 3 + (3+1)3y = 3 + 3y = 0 \bmod 3^2,\ y \in \mathbb{Z}_3,$$

so $y + 1 = 0 \bmod 3$, $y = 2$, $x_2 = 1 + 3 \cdot 2 = 7 \bmod 3^2$. Search for solutions to the comparison $f(x) = 0 \bmod 3^4$ in the form $x_3 = x_2 + 3^2y = 7 + 3^2y \bmod 3^4$, $y \in \mathbb{Z}_9$. In this case,

$$f(x_3) = f(7+9y) = f(7) + f'(7)3^2y = (7^2+1)7 + 1 + (3 \cdot 7^2 + 1)3^2y = 39 \cdot 9 + (4y) \cdot 9 = 0 \bmod 3^4,\ y \in \mathbb{Z}_9,$$

so $4y + 39 = 4y + 30 \bmod 3^2$. In Example 1 we had obtained that $4^{-1} \bmod 9 = 7$ and hence $y = -3 \cdot 7 = 3 \cdot 2 = 6 \bmod 3^2$, therefore, $x_3 = 7 + 3^2y = 7 + 9 \cdot 6 = 61 \bmod 3^4$.

Theorem 2. *Let the prime factorization $n = m_1 \dots m_k$, $m_i = p_i^{n_i}$, $i = 1, \dots, k$, of n be given. Then for fixed k, fixed prime p_i, $i = 1, \dots, k$, and increasing n the Boolean complexity of calculation of one root of the comparison $f(x) = 0 \bmod n$ equals*

$$O\left(M(\lambda(n))(d + \log_2 k) + \lambda(n)\sum_{i=1}^{k}\frac{M(\lambda(m_i))}{\lambda(m_i)}\right) = O(dM(\lambda(n))).$$

Proof. As is known, due to Chinese remainder theorem, solution of the comparison $f(x) = 0 \bmod n$ can be reduced to the solution of the comparisons $f(x) = 0 \bmod m_i$, $i = 1, \dots, k$. Let $x_i \in \mathbb{Z}_{m_i}$ be one of the roots of the comparison $f(x) = 0 \bmod m_i$, $i = 1, \dots, k$. In accordance with Theorem 1, such solution can be found with the complexity $O(dM(n_i\lambda(p_i))$ providing p_i is fixed and $n_i \to \infty$. If x_i, $i = 1, \dots, k$, are known, then the corresponding solution $x \in \mathbb{Z}_n$, $x \bmod n_i = x_i$, can be found in the following well-known way.

For each $i = 1, \dots, k$, assume

$$M_i = n/m_i = \prod_{j, j \neq i} m_j, \ a_i = M_i \bmod m_i.$$

It is evident that $(a_i, m_i) = 1$ and, therefore, there exists $b_i = a_i^{-1} \bmod m_i$, and then $b_iM_i = 1 \bmod m_i$, $b_iM_i = 0 \bmod m_j$ for all $j \neq i$; and for

$$x = \sum_{i=1}^{k} x_i b_i M_i \bmod n$$

the equalities $x = x_i \bmod m_i$ are valid, which implies $f(x) = f(x_i) = 0 \bmod m_i$,$i = 1, \dots, k$, and hence $f(x) = 0 \bmod n$. Use the following well-known result (its polynomial version can be found, e,g,. in [9]).

Lemma 2. *The complexity of the inverse Chinese algorithm of calculation of* $x = x_i \bmod m_i$, $i = 1, \dots, k$, *equals*

$$O\left(M(\lambda(n)) \log_2 k + \lambda(n) \sum_{i=1}^{k} \frac{M(\lambda(m_i))}{\lambda(m_i)}\right).$$

Proof. Assume $A_i = n \bmod m_i^2$. It is evident that $A_i = m_i c_i \bmod m_i^2$, $c_i \in \mathbb{Z}_{m_i}$. Since

$$m_i c_i - n = m_i c_i - m_i M_i = m_i(c_i - M_i) = 0 \bmod m_i^2,$$

then $c_i = M_i \bmod m_i = a_i$, therefore, $a_i = A_i/m_i$.

The Boolean complexity of calculation of $A_i = n \bmod m_i^2$, $i = 1, \dots, k$, is equal to $O(M(\lambda(n)) \log_2 k)$. This can be proved by the bisection method (see, e.g., [7–9]): assume

$$N_0 = m_1^2 \dots m_{\lfloor k/2 \rfloor}^2, \ N_1 = m_{\lfloor k/2 \rfloor + 1}^2 \dots m_k^2, \ C_i = n \bmod N_i, \ i = 0, 1,$$

then

$$A_i = C_0 \bmod m_i^2, \ i \leq \lfloor k/2 \rfloor, \ A_i = C_1 \bmod m_i^2, \ i > \lfloor k/2 \rfloor.$$

The complexity of calculation of N_0, N_1 by the bisection method is equal to $O(M(\lambda(n)) \log_2 k)$ (and intermediate results obtained during this calculation can be also used in recursion), the complexity of calculation of $C_i = n \bmod N_i$ equals $O(M(\lambda(n)))$ (see [8]), after that the problem is recursively reduced to two subproblems whose complexities are estimated as $O(M(\lambda(n))(\log_2 k - 1))$ by the inductive hypothesis.

The complexity of calculation of $a_i = A_i/m_i$, $i = 1, \dots, k$, is equal to

$$O\left(\sum_{i=1}^{k} M(\lambda(m_i))\right) = O(M(\lambda(n))).$$

In accordance to Lemma 1, the complexity of calculation of $b_i = a_i^{-1} \bmod m_i$ $i = 1, \dots, k$, is also equal to

$$O\left(\sum_{i=1}^{k} M(\lambda(m_i))\right) = O(M(\lambda(n)))$$

for fixed p_i and for $n_i \to \infty$. The complexity of multiplication of an a-place number by a b-place number for $b \geq a$ and also the complexity of the division of an $(a + b)$-place number by a b-place number with a remainder can be estimated as $O(bM(a)/a)$ (see, e.g., [8]). Using this inequality, we obtain that the complexity of calculation of the numbers $M_i = n/m_i$, $i = 1, \dots, k$, equals

$$O\left(\lambda(n) \sum_{i=1}^{k} \frac{M(\lambda(m_i))}{\lambda(m_i)}\right).$$

We can similarly get that the order of complexity of calculation of the remainders $b_iM_i \bmod n$, $i = 1, \dots, k$, is also equal to

$$\lambda(n) \sum_{i=1}^{k} \frac{M(\lambda(m_i))}{\lambda(m_i)}.$$

Further, calculate $x_ib_iM_i \bmod n$, $i = 1, \dots, k$, with the same order of complexity and then find

$$x = \sum_{i=1}^{k} x_ib_iM_i \bmod n$$

with the complexity $O(k\lambda(n))$. $\square$

THE COMPLEXITY OF INTEGER SOLUTION OF DIOPHANTINE EQUATIONS WITH A SINGLE VARIABLE

An algorithm solving equations in the rings $\mathbb{Z}_{p^n}$ can be applied to solve equations in the ring $\mathbb{Z}$. This idea had been proposed by Legendre and was implemented by Zassenhaus in [10] to obtain factorization in the polynomial ring $\mathbb{Z}[X]$. Let $f(x) = a_dx^d + a_{d-1}x^{d-1} + \ldots + a_0$, $a_i \in \mathbb{Z}$, $|a_i| < 2^n$, $i = 0, \dots, d$, $a_0 \geq 0$. An algorithm of complexity $d^{O(1)}n^3$ factorizing a polynomial f into irreducible factors with integer coefficients was presented in [11]. Using Theorem 1, one can propose an elementary algorithm calculating integer roots of low degree equations with large coefficients with the order complexity equal to that of multiplication of n-bits numbers.

Theorem 3. *The Boolean complexity of calculation of integer roots of the polynomial $f(x)$ is equal to $O_d(M(n))$.*

Proof. One can represent $f(x)$ in the form

$$f_1(x)f_2(x)^2 \ldots f_d(x)^d,$$

where $f_i(x) \in \mathbb{Z}[X]$ are polynomials without multiple roots, with the complexity $O_d(M(n))$ (more accurate estimates can be obtained using [9]). Therefore, below we can assume that $f(x)$ has no multiple roots and $a_0 > 0$.

In the case $d = 1$ the complexity of calculation of the unique root evidently equals $O(M(n))$. Choose prime p so that $p^m > 2a_0 > p^{m-1}$. Integer roots of the equation $f(x) = 0$ evidently divide a_0 and hence belong to the segment $[-a_0, a_0]$ (by means of known methods of determination of boundaries for roots or by Mignotte's inequality this segment can be shortened), but determination of roots by exhaustive search can be difficult (an evident approach uses the prime factorization of a_0). However, searching for such roots can be easily reduced to solving the comparisons $f(x) = 0 \bmod p^m$ and rejecting irrelevant roots (the verification by Horner's method whether $x \in [-a_0, a_0]$ is a root can be performed with the complexity $O(d^2M(n))$, and the bisection method has the complexity estimate $O(M(dn)\log_2 d)$ for the complexity of this computation).

To solve the comparison $f(x) = 0 \bmod p^m$, we can use Theorem 1. In the case when all integer roots $x_1, \dots, x_k$, $k \leq d$, of the equation $f(x)$ are pairwise distinct modulo p, it is possible to find each of them with the complexity $O(dM(n))$ for fixed d and for $n \to \infty$. Indeed, in this case the comparison $f(x) = 0 \bmod p$ has the roots $x_i \bmod p$, $i = 1, \dots, k$ (among others). Compute the derivative $f'(x) \in \mathbb{Z}[X]$. Apply the Euclidean algorithm to the pair of polynomials (f, f'). To perform calculations over the ring $\mathbb{Z}$, at each step of the algorithm we pass from a pair of polynomials (g, h), where $g = a_sx^s + \ldots$, $h = b_lx^l + \ldots$, $s \geq l$, to the pair $(h, b_lg - ha_sx^{s-l})$ such that the sum of the degrees of the polynomials is less than $s + l$. It is evident that any common divisor of the first pair of polynomials is also a common divisor of the second pair. The algorithm stops either if a pair of polynomials $(h, 0)$ appears (then the polynomial h is the common divisor of f and f'), or if a pair (h, c) appears, where $c \in \mathbb{Z}, c \neq 0$ (then the polynomials f and f' have no common roots over the field $\mathbb{Q}$ because they are relatively prime). In the first case we obtain a decomposition of f into two factors, and the problem of root searching is reduced to the same one, but for polynomials of lesser degrees. In the second case we choose prime p with the additional condition $c \neq 0 \mod p$. Then (performing all the calculations in the Euclidean algorithm modulo p) we obtain that the polynomials $f(x) \bmod p$ and $f'(x) \bmod p$ have no common roots over the field $\mathbb{Z}_p$ and hence $f'(x_i) \neq 0 \bmod p^m$ for $m = 1$, and so for any m. Hensel's lifting of the root $x_i \in \mathbb{Z}_p$ to a unique root over $\mathbb{Z}_{p^m}$, which is equal to x_i modulo p, can be implemented by methods of Theorem 1 with the complexity $O(dM(m\lambda(p)))$ for fixed d and $m \to \infty$ providing $f'(x_i) \neq 0 \bmod p^m$. If the comparison $f(x) = 0 \bmod p$ has roots distinct from $x_i \bmod p$, $i = 1, \dots, k$, for example, a root $x_{k+1} \in \mathbb{Z}_p$, then it also satisfies the condition $f'(x_{k+1}) \bmod p^m \neq 0$ and can be lifted to a root of $f(x) \bmod p^m$ for any m. However, if

$$p^m > |a_d a_0^d| + |a_{d-1} a_0^{d-1}| + \ldots + |a_0|,$$

then any integer root of the polynomial $f(x) = a_d x^d + a_{d-1} x^{d-1} + \ldots + a_0$ is evidently a root of the comparison $f(x) \bmod p^m$ satisfying one of the inequalities $0 \leq x \leq a_0$, $p^m - a_0 \leq x < p^m$, and vice versa. Therefore, to find all integer roots of $f(x)$, it suffices to find all roots of $f(x) \bmod p$, lift them to the roots of the comparison $f(x) = 0 \bmod p^m$, and throw away those not satisfying any of the inequalities $0 \leq x \leq a_0$, $p^m - a_0 \leq x < p^m$. All the roots x_i, $i = 1, \ldots, k$, of the polynomial $f(x)$ are contained among the solutions of the comparison $f(x) = 0 \bmod p^m$ already for m satisfying the inequalities $p^m > 2a_0 > p^{m-1}$, and each root of the comparison $f(x) = 0 \bmod p^m$ can be obtained by lifting of the corresponding root of the comparison $f(x) = 0 \bmod p$.

In accordance to Theorem 1, the complexity of calculation of a root of the comparison $f(x) = 0 \bmod p^m$ equals

$$O(dM(m\lambda(p)) + M(\lambda(p))\lambda(\lambda(p)) \log_2 m + dpM(\lambda(p))).$$

The latter term $dpM(\lambda(p))$ estimates the complexity of calculation of all roots of the comparison $f(x) = 0 \bmod p$ by full search of all elements of the field $\mathbb{Z}_p$. Since the number of roots of the polynomial $f(x)$ does not exceed $\min(d, p)$, then the complexity of determination of all integer roots of $f(x)$ can be estimated as

$$O(\min(d, p)(dM(m\lambda(p)) + M(\lambda(p))\lambda(\lambda(p)) \log_2 m) + dpM(\lambda(p))),$$

where $p^m > |a_d a_0^d| + |a_{d-1} a_0^{d-1}| + \ldots + |a_0| \geq p^{m-1}$, i.e., $m\lambda(p) = O(nd)$. Therefore, for $p^m > 2a_0 > p^{m-1}$, i.e., for $m\lambda(p) = O(n)$ one can find all roots of the comparison $f(x) = 0 \bmod p^m$ with the complexity

$$O(\min(d, p)(dM(m\lambda(p)) + M(\lambda(p))\lambda(\lambda(p)) \log_2 m) + dpM(\lambda(p)))$$

and then throw away extraneous roots with the additional complexity $O(\min(d, p)d^2 M(n))$, find all integer roots of the polynomial $f(x)$ with the coefficients whose absolute values are less than 2^n with the complexity

$$O(\min(d, p)(d^2 M(n) + M(\lambda(p))\lambda(\lambda(p)) \log_2 n) + dpM(\lambda(p))).$$

To calculate the number p, it is necessary to calculate the pair $(h(x), c)$, $c \in \mathbb{Z}$, at the last step of the Euclidean algorithm and find the minimal prime p not dividing the number c. The number of steps of the algorithm does not exceed $2d - 1$, the coefficients of polynomials calculated at the ith step do not exceed $2^{2^i(n+1)-1}$ (this estimate can be proved by induction) and hence $\log_2 |c| < 2^{2d}(n+1)$, and the complexity of calculation of c equals $O(M(2^{2d}n))$. To find the minimal prime p not dividing the number c, we consequently generate prime numbers $p_1, p_2, \ldots$ by means of the sieve of Eratosthenes and divide the number c by p_i. Estimate from above the minimal value of such p. Let the number c be divisible by $p_1 \ldots p_k$ but not divisible by p_{k+1}. Since

$$|c| \geq p_1 \ldots p_k > a^{p_k},\ a > 1,$$

in accordance with the well-known number-theoretic Mertens inequality, then $p_k < \log_2 |c| = 2^{2d}(n+1)$, and due to Bertrand's postulate, $p = p_{k+1} < 2p_k < 2^{2d+1}(n+1)$, therefore, $\lambda(p) < 2d + \log_2 2(n+1)$ (one can obtain slightly more accurate estimates). The complexity of generation of first $k+1$ primes by means of the sieve of Eratosthenes can be estimated (using Mertens and Chebyshev inequalities) as

$$O(\lambda(p)) \sum_{i=1}^{k} \frac{p}{p_i} = O(p\lambda(p)\lambda(\lambda(k))) = O(k\lambda^2(k)\lambda(\lambda(k))) = O(2^{2d}n(d + \lambda(n))^2 \lambda(d + \lambda(n))).$$

The complexity of the test divisions of the number c by the primes $p_1, \ldots, p_{k+1}$ can be estimated as

$$O(\lambda(c)) \sum_{i=1}^{k+1} M(\lambda(p_i))/\lambda(p_i) = o(2^{4d}n^2).$$

In the worst case the latter implies the estimate

$$O(d^3 M(n) + d2^{2d}nM(d + \lambda(n)) + o(2^{4d}n^2) = O_d(n^2)$$

for fixed d and $n \to \infty$.

This estimate can be improved by rearranging the algorithm as follows. Instead of calculation of the constant c by means of the Euclidean algorithm over the ring $\mathbb{Z}$ and verification whether the next prime p_i divides the number c, it suffices to find the greatest common divisor of the polynomials $f \bmod p_i$ and $f' \bmod p_i$ by means of the Euclidean algorithm. The complexity of this calculation equals $O(d^2M(\lambda(p_i)))$ (if one uses Strassen's fast version of the Euclidean algorithm, then the estimate of the complexity decreases to $O(M(d)M(\lambda(p_i))\log_2 d)$). If $c = 0 \bmod p_i$, then the degree of this greatest common divisor is greater than 0, and if $c \neq 0 \bmod p_i$, then the polynomials $f \bmod p_i$ and $f' \bmod p_i$ are relatively prime and hence they have no common roots over the field $\mathbb{Z}_p$ (in the case $c = 0 \bmod p_i$ the degree of the greatest common divisor can be greater than 1, and then the polynomials $f(x) \bmod p_i$ and $f'(x) \bmod p_i$ could have no common roots, and hence calculations modulo p_i^m can be used to find integer roots of $f(x)$). Therefore, one can find the roots of the comparison $f(x) \bmod {p_i}^m$ as above, where $p_i^m > 2a_0 > p_i^{m-1}$, $p_1 \dots p_{i-1} \mid c$ and, throwing away extraneous roots, find all integer roots of $f(x)$. The estimate of the complexity of all GCD of $(f \bmod p_j, f' \bmod p_j)$, $j = 1, \dots, i$, is equal to

$$O(d^2)\sum_{j=1}^{i} M(\lambda(p_j)) = O(d^2)(M(\lambda(p_1) + \ldots + \lambda(p_{i-1})) + M(\lambda(p_i)))$$

$$= O(d^2)(M(\lambda(p_1 \dots p_{i-1})) + M(\lambda(p_i)))$$

$$= O(d^2)(M(\lambda(c)) + M(\lambda(p_i))) = O(d^2)(M(2^{2d}n) + M(2d + \log_2(n+1))) = O(d^2M(2^{2d}n)).$$

Therefore, the final complexity estimate of the calculation of all integer roots of $f(x)$ equals

$$O(d^3M(n) + d2^{2d}nM(d + \lambda(n)) + d^2M(2^{2d}n)) = O_d(M(n)). \qquad \square$$

EXAMPLE OF FINDING INTEGER ROOTS OF THE POLYNOMIAL $f(x) = x^3 + 22551x^2 - 408321x - 109039822871$

Calculate $f'(x) = 3x^2 + 45102x - 408321$, then apply the Euclidean algorithm, i.e.,

$$3(x^3 + 22551x^2 - 408321x - 109039822871) - x(3x^2 + 45102x - 408321)$$

$$= 22551x^2 - 816642x - 327119468613,$$

$$3(22551x^2 - 816642x - 327119468613) - 22551(3x^2 + 45102x - 408321)$$

$$= -1019545128x - 972150358968,$$

$$(-1019545128x - 972150358968)14355691095384$$

$$+1019545128(14355691095384x - 138767228736696) = 1409736970359583672942080 0.$$

Further, notice that 14097369703595836729420800 is divisible by 2, 3, 5 and is not divisible by $p = 7$. Solving consequently the comparisons $f(x) = 0 \bmod 7$, $f(x) = 0 \bmod 7^2$, $f(x) = 0 \bmod 7^4$, $f(x) = 0 \bmod 7^8$, $f(x) = 0 \bmod 7^{16}$, lift the roots $0, 4, 6 \in \mathbb{Z}_7$ of the first one to the roots $-22351, 2111, -2311 \in \mathbb{Z}_{7^{16}}$. Since $7^{14} > 2 \cdot 109039822871 > 7^{13}$, then it is possible to choose the following sequence of calculations: $f(x) = 0 \bmod 7$, $f(x) = 0 \bmod 7^2$, $f(x) = 0 \bmod 7^4$, $f(x) = 0 \bmod 7^7$, $f(x) = 0 \bmod 7^{14}$. If one manages to obtain an estimate for the roots which is more accurate than the trivial one used above, then the power 14 could be decreased to 6, i.e., it would be sufficient to proceed the following chain of "lifts": $f(x) = 0 \bmod 7$, $f(x) = 0 \bmod 7^2$, $f(x) = 0 \bmod 7^3$, $f(x) = 0 \bmod 7^6$, and at the end of the chain the roots $-22351, 2111, -2311 \in \mathbb{Z}_{7^6}$ already appear. Using the second version of the algorithm it suffices to calculate the following GCDs: $(x^3+x^2+x+1, x^2+1) = x+1 \bmod 2$, $(x^3+1, 0) = x^3+1 \bmod 3$, $(x^3+x^2-x-1, 3x^2+2x-1) = x+1 \bmod 5$, $(x^3 + 4x^2 + 3x, 3x^2 + x + 3) = 1 \bmod 7$, and then solve the comparisons $f(x) \bmod 7^m$ as in the first version.

REMARK ON OTHER ELEMENTARY WAYS OF SOLVING COMPARISONS

Calculation of $a^{-1} \bmod m$ for $(a,m)=1$ is equivalent to solving the comparison $ax = 1 \bmod m$, or to solving of the equation $ax + my = 1$ under the conditions $0 < x < m$, $-a < y < 0$. This solution is unique, $x = a^{-1} \bmod m$. It can be found by the extended Euclidean algorithm. The estimate of the complexity of this algorithm is $O(\lambda(am))^2$, but the fast Shenhage modification of this algorithm is known (see, e.g., [7]). Its complexity is $\lambda(\lambda(am))M(\lambda(am))$. However, it is hard to implement it as a Boolean circuit. Such a circuit easily realizes calculations by the formula $a^{-1} \bmod m = a^{\phi(m)-1} \bmod m$ providing the values of the Euler function $\phi(m)$ are known. The complexity of this circuit is equal to $O(\lambda(m)M(\lambda(m)))$. If the canonical factorization $m = m_1 \dots m_k$, $m_i = p_i^{n_i}$, $i = 1, \dots, k$, into prime factors are known, then $\phi(m)$ can be calculated with the complexity $O(\lambda(k)M(\lambda(m))) = O(\lambda(\lambda(m))M(\lambda(m)))$.

Consider the dual problem of calculation of $m^{-1} \bmod a$. Its solution can be obtained by simultaneous calculation of $a^{-1} \bmod m$ by means of the extended Euclidean algorithm as a solution to the equation $ax+my=1$ under the conditions $0<x<m$, $-a<y<0$ because of $m^{-1} \bmod a = -y \bmod a$. If we find the number $x = a^{-1} \bmod m$ by some other algorithm, then $m^{-1} \bmod a = -y = \lfloor (ax-1)/m \rfloor$ can be calculated with the complexity $O(M(\lambda(am)))$.

In [12], the following formula was given in one of problems:

$$a^{-1} \bmod p = (-1)^{a-1}\frac{(p-1)\dots(p-a+1)}{a!} = (-1)^{a-1}\frac{\binom{p}{a}}{p} \bmod p.$$

Performing calculations with the use of Pascal triangle modulo p^2, one can find $a^{-1} \bmod p$ with the Boolean complexity $O(a^2\lambda(p) + M(\lambda(p))$, which could be faster than the calculation of the $(p-2)$th power for small a.

A method to solve the comparison $p^k x = b \bmod m$, where p is a prime, was presented in [12] in the form of a problem. This method is based on the following recurrent algorithm. Let $b + mt{=}0 \bmod p$, i.e., $t{=}{-}b \cdot m^{-1} \bmod p$. Find the maximal number δ such that $p^\delta \mid (p^k, b+mt)$. The problem can be reduced to solution of the comparison $p^{k-\delta}x = \frac{b+mt}{p^\delta} \bmod m$. If these calculations are executed in the base p numeral system, then the complexity of the algorithm can be estimated as $O(k^2(M(\lambda(p))))$. To estimate the Boolean complexity, it is necessary to add the complexity of transformation from the binary system to the base p system and backward. The latter can be estimated as $O(\lambda(\lambda(m))M(\lambda(m)))$ by means of the Shenhage algorithm. Therefore, the complexity of calculation of $(p^k)^{-1} \bmod m$ can be estimated as $O(k^2M(\lambda(p))) + O(\lambda(\lambda(m))M(\lambda(m)))$. Therefore, the complexity of calculation of $m^{-1} \bmod p^k$ equals

$$O(k^2M(\lambda(p)) + \lambda(\lambda(m))M(\lambda(m)) + M(\lambda(m) + k\lambda(p)) + \lambda(p)M(\lambda(p))).$$

For fixed p, $m < p^k$, and $k \to \infty$ this estimate equals $O(k^2M(\lambda(p))$.

Algorithms solving the comparisons $x^2 = a \bmod 2^k$ and $x^2 = a \bmod p^k$ were also presented in [12] in the form of problems. The Boolean complexity of the first one is equal to $O(k^2)$ and for the second one it is equal to $O(k^2\lambda^2(p))$. Both the algorithms are versions of Hensel's lifting.

Another algorithm solving the comparison $x^2 = a \bmod p^k$ was given in [11] in the form of problem and representing the solution by the explicit formula $x = \pm PQ' \bmod p^k$, where

$$P = 2^{k-1}\sqrt{a}^k, Q = 2^{k-1}\sqrt{a}^{k-1}, \sqrt{a} = b \in \mathbb{Z}_p, b^2 = a, QQ' = 1 \bmod p^k.$$

Since the Boolean complexity of calculations of Q' and of multiplication $PQ' \bmod p^k$ equals $O(M(k\lambda(p)))$, the complexity of calculation of $\sqrt{a}$ over the field $GF(p)$ equals $O(\lambda(p)M(\lambda(p)))$, and the complexity of calculation of the remainders $P, Q \bmod p^k$ equals $O(\lambda(k)(M(k\lambda(p)))$, then the complexity of square root extracting modulo p^k by this algorithm equals $O(\lambda(k)(M(k\lambda(p)))$.

ACKNOWLEDGMENTS

The work was supported by the Russian Foundation for Basic Research (projects nos. 19–01–00294 and 18–01–00337).

REFERENCES

1. A. A. Bolotov, S. B. Gashkov, A. B. Frolov, and A. A. Chasovskih, *Elementary Introduction to Elliptic Cryptography. Algebraic and Algorithmic Foundations* (URSS Lenand, Moscow, 2018) [in Russian].
2. O. N. Vasilenko, *Number-Theoretic Algorithms in Cryptography* (MTsNMO, Moscow, 2003; AMS, Providence, RI, 2006).
3. M. M. Gluhov, I. A. Krugov, A. B. Pichkur, and A. V. Cheremushkin, *Introduction to Number-Theoretic Methods of Cryptography* (Lan', St. Petersburg, 2011) [in Russian].
4. E. Bach, "Explicit Bounds for Primality Testing and Related Problems," Math. Comput. **22**, 355 (1989).
5. M. Fuerer, "Faster Integer Multiplication," SIAM J. Comput. **39** (3), 979 (2009).
6. D. Harvey, J. van der Hoeven, and G. Lecerf, "Faster Polynomial Multiplication Over Finite Fields," ArXive.org `arXive: 1407.3361` (2014).
7. A. Aho, J. Hopcroft, and J. Ullman, *The Design and Analysis of Computer Algorithms* (Addison-Wesley, Boston, 1974; Mir, Moscow, 1979).
8. S. B. Gashkov and V. N. Chubarikov, *Arithmetics, Algorithms, Complexity of Calculations* (Nauka, Moscow, 1996) [in Russian].
9. S. B. Gashkov, "On the Complexity of Integration of Rational Fractions," Trudy Matem. Inst. RAN, **218**, 122 (1997) [Proc. of the Steklov Institute of Math. **218**, 117 (1997)].
10. H. Zassenhaus, "A Remark on the Hensel Factorization Method," Math. Comput. **32**, 141, 287 (1978).
11. A. Lenstra, H. Lenstra, and L. Lovasz, "Factoring Polynomials with Rational Coefficients," Math. Ann. **261**, 515 (1982).
12. I. M. Vinogradov, *Elements of Number Theory* (GITTL, Moscow, 1952; Dover Publ., NY, 1954).

Translated by A. Ivanov