Programming and Computer Software, Vol. 29, No. 2, 2003, pp. 59–71. Translated from Programmirovanie, Vol. 29, No. 2, 2003.

Solution of Systems of Linear Equations by the *p*-Adic Method

G. I. Malaschonok

Derzhavin State University, Tambov, ul. Internatsional'naya 33, Tambov, 392622 Russia
e-mail: malashonok@math-iu.tambov.su

Received June 3, 2002

Abstract—A new algorithm for solving systems of linear equations $Ax = b$ in an Euclidean domain is suggested. In the case of the ring $\mathbb{Z}$ of integers, the complexity of this algorithm is $O^{\sim}(n^3 m \log^2 \|A\|)$, where $A \in \mathbb{Z}^{n \times m}$ $(m > n)$ is a matrix of rank n and $\|A\| = \max\limits_{i,j} |A_{i,j}|$, if standard algorithms for the multiplication of integers and matrices are used. Under the same conditions, the best algorithm of this kind among those published earlier, which was suggested by Labahn and Storjohann in [1], has complexity $O^{\sim}(n^4 m \log^2 \|A\|)$. True, when using fast algorithms for the multiplication of numbers and matrices, the theoretical complexity estimate for the latter algorithm is $O^{\sim}(n^{\beta} m \log^2 \|A\|)$, which is better than the similar estimate $O^{\sim}(n^3 m \log \|A\|)$ for the new algorithm.

1. INTRODUCTION

In this paper, we consider two Euclidean domains—the ring of integers and the ring of polynomials over a field—and two following problems: solution of a system of linear equations in a domain and that in the field of quotients of the domain.

Consider a system of linear equations with a square nonsingular matrix of coefficients. Such a system is referred to as a **determined system**; it has only one solution in the field of quotients of the domain.

Let all coefficients of the system be integer and the order of the system be n. Let the system be solved by an algorithm of complexity $\theta(n^3)$ operations on the coefficients.[1]

If the standard integer arithmetic is used, the complexity of the multiplication operation is proportional to the product of logarithms of the multipliers. The operands on the kth step, $k = 1, \ldots, n-1$, are minors of the kth order; hence, their logarithms grow as $O^{\sim}(k)$. The total complexity of such an algorithm grows as $O^{\sim}(n^5)$.

The modular approach based on the Chinese remainder theorem makes it possible to avoid the growth of the coefficients. The computations are performed in the residue ring modulo prime number, and the coefficients do not grow. In this case, the reconstruction of a solution requires $O^{\sim}(n)$ modules; therefore, it is required to solve the system $O^{\sim}(n)$ times. Hence, the complexity of such an algorithm grows as $O^{\sim}(n^4)$.

The use of the p-adic method with linear lifting makes it possible to avoid not only the growth of the coefficients but also the growth of the number of modules. First, the inverse matrix and solution vector modulo p are sought, which requires $\theta(n^3)$ operations modulo p. Next, by multiplying the inverse matrix by the reconstructed column of the free terms, the solution vector is lifted up to $p^{O^{\sim}(n)}$, which requires $O^{\sim}(n)$ steps of lifting. On each step, the multiplication of a matrix by a vector requires $\theta(n^2)$ operations modulo p. Then, n rational numbers that constitute the solution vector are reconstructed; each of them requires $\theta(n^2)$ operations. Thus, each stage requires $O^{\sim}(n^3)$ operations, and the total complexity of the algorithm is $O^{\sim}(n^3)$.

Let now a determined system be solved by an algorithm with complexity $O^{\sim}(n^{\beta})$ $(\beta \approx 2.376)$ operations on the coefficients, and let numbers be multiplied by means of the FFT algorithm [2]. The multiplication of n-bit numbers requires $O^{\sim}(n)$ bitwise operations. The calculation of the greatest common divisor and reconstruction of a number by mutually prime remainders requires the same number of operations. It should be noted that the last estimates can be viewed only as theoretical ones unlike those given above, which can easily be implemented in practice.

The first and second algorithms, i.e., the algorithms with standard and modular arithmetic, require in this case $O^{\sim}(n^{\beta+1})$ bitwise operations, whereas the p-adic

[1] If f and g are functions with the range in $\mathbb{R}$, we use the following notation: $g = O(f)$ if $|g/f|$ is bounded at $+\infty$, $g = O^{\sim}(f)$ if $g = O(f \log^c f)$ for a certain constant c, $g = \theta(f)$ if $|g/f|$ and $|f/g|$ are bounded at $+\infty$, and $g = o(f)$ if $\lim(g/f) = 0$.

0361-7688/03/2902-0059$25.00

algorithm requires, as before, $O^{\sim}(n^3)$ bitwise operations.

Hence, the asymptotically best method for determined systems is the *p*-adic method from both theoretical and practical standpoints. This method was developed by Dixon in 1982 [3]. Note that the best algorithms for small-dimension systems are those with standard arithmetic; for systems of intermediate dimension, modular algorithms; and, only for large-scale systems, the *p*-adic method is the best one. This can be explained, first of all, by that the reconstruction of a quotient is a more complicated procedure than that of the numerator and denominator separately.

In this work, we consider algorithms for solving an **undetermined system** of linear equations that find a general solution of such a system.

The traditional method for solving linear systems of the Diophantine equations is based on the calculation of the Smith form for the matrix of the coefficients of the system [4–7].

Another approach, which is currently being developed, consists in representing the general solution of a system as a sum of a particular solution and the general solution of the corresponding homogeneous system. The best results in the framework of this approach are obtained in [1, 8, 9]. Let $xA = 0$ be a homogeneous Diophantine system of linear equations, the matrix $A \in \mathbb{Z}^{m \times n}$ have rank n, $m > n$, $x \in \mathbb{Z}^{1 \times m}$, U be a unimodular matrix over $\mathbb{Z}$, such that $UA = H$ and H is an upper triangular matrix. Then, it is not difficult to see that the last $m - n$ rows of U constitute a basis in the set of all solutions of the homogeneous system $xA = 0$.

An algorithm of this kind is discussed in [8]. To find the general solution of a homogeneous Diophantine system $xA = 0$, an algorithm for the calculation of the upper triangular form for the matrix A is suggested, which has complexity $O^{\sim}(n^2 m \boldsymbol{M}(n \log \|A\|))$. In the work [1], this algorithm was improved, and the complexity of the improved algorithm, which calculates the Hermite upper triangular form, is $O^{\sim}(n^{\beta-1} m \boldsymbol{M}(n \log \|A\|))$. In [9], a probabilistic method for finding a particular solution of a Diophantine system of linear equations is suggested; its expected complexity is $O^{\sim}(n^3 \log^3 \|A\|)$ when using standard multiplication algorithms. Hence, the major part of the total complexity of this algorithm for solving Diophantine systems of linear equations is that associated with the calculation of the general solution of the homogeneous system. Here and in what follows, $\|A\| = \max_{i,j}(|A_{i,j}|)$, and β is the exponent of power in the complexity estimate of the matrix multiplication. It is assumed that the multiplication of two matrices of order n requires $\theta(n^\beta)$ operations and $\theta(\boldsymbol{M}(t))$ is the complexity of the multiplication of two t-bit numbers. For standard algorithms, $\beta = 3$, and $\boldsymbol{M}(t) = t^2$. The best currently known theoretical results give $\beta \approx 2.376$ and $\boldsymbol{M}(t) = t \log t \log\log t$ [2, 10].

The best theoretical results cannot be realized yet because of limited capabilities of modern computers. In connection with this, it is commonly accepted to distinguish between the best theoretical and best practical results. For standard multiplication algorithms, the complexity estimate for the algorithm [1] for solving Diophantine systems of linear equations is $O^{\sim}(n^4 m \log^2 \|A\|)$, whereas, for the best multiplication algorithms, it is equal to $O^{\sim}(n^\beta m \log \|A\|)$.

We consider a different approach to solving undetermined systems of linear equations; namely, we consider the deterministic and probabilistic *p*-adic methods, which were first suggested in [11] and are further developed in this work.

The basic idea of the approach is to find a basis set of points in the plane containing all solutions of the system in the space $\mathbb{Q}^n$ by applying the *p*-adic lifting. Then, in order to get a solution in $\mathbb{Z}$, integer points in this plane are calculated. Note that, in the deterministic method, the problem is reduced to solving a system in the quotient ring. In the probabilistic method, one integer solution is calculated if the ideal generated by the denominators of all rational solutions is unit, and, then, all other basis integer solutions are constructed.

The complexity estimate for the deterministic algorithm (see Proposition 3) is $O^{\sim}(n^3 m \log^2 \|A\|)$ when using standard algorithms for matrix and number multiplication and $O^{\sim}(n^3 m \log \|A\|)$ for theoretically best algorithms for matrix and number multiplication, where $A \in \mathbb{Z}^{n \times m}$ is a matrix of rank n. The comparison of these estimates with those for the algorithm from [1] shows that, for the fast multiplication algorithms, the complexity of the new algorithm is greater by $n^{3-\beta}$ times, whereas, for the standard multiplication algorithms, it is less by n times. This allows us to consider the algorithm from [1] as theoretically best and the new algorithm as practically best.

Another advantage of the new algorithm is that the coefficients in any solution vector being calculated do not exceed the Hadamard bound $v = (\sqrt{n}\,\|A\|)^n$ for the greatest minor of order n, whereas, in the algorithm from [1], the bound for the absolute value of the coefficients is nv^2 (in the algorithm from [8], it is even greater).

The new results obtained in this work can be summarized as follows.

• In the framework of the general method discussed in [11], the deterministic and probabilistic *p*-adic algorithms are suggested (Figs. 2–5).

• More exact complexity estimates for these algorithms are obtained for various system dimensions and various computational models.

• The complexity estimate in the case of using standard multiplication algorithms is reduced by n times as compared to the estimate obtained in [11]. This was

achieved through the use of the modular algorithm for computing matrix product in the quotient ring.

The paper is organized as follows. In Section 2, two computational models are presented, which are further used for getting complexity estimates for the algorithms. In Section 3, the *p*-adic algorithms (Figs. 2 and 3) for solving systems of linear equations in the field of quotients are discussed. The probabilistic method for solving systems in commutative domains (Fig. 4) is discussed in Section 4. In Section 5, the deterministic method for solving systems in Euclidean domains (Fig. 5) is discussed, and complexity estimates are presented.

2. COMPUTATIONAL MODEL

Let $F[x]$ be a ring of polynomials over a field F. The *norm* of an element p in $F[x]\backslash\{0\}$, denoted as $\mathrm{Nr}(p)$, is the degree of the polynomial, and the norm of an element in the ring of integers $\mathbb{Z}\backslash\{0\}$ is the logarithm of the absolute value of this element; i.e.,

$$\mathrm{Nr}(p) = \begin{cases} \deg p \ \text{ for } \ F[x], \\ \log_w |p| \ \text{ for } \ \mathbb{Z}. \end{cases} \tag{1}$$

The logarithmic base is convenient to take equal to the length of the machine word w, which is currently either 2^{32} or 2^{64}. If every coefficient of a polynomial is stored in one machine word, then the number of the words required for storing one element p is equal to $\lfloor \mathrm{Nr}(p) \rfloor + 1$ in both cases. It is evident that $0 \le \mathrm{Nr}(p)$ and $\mathrm{Nr}(pq) = \mathrm{Nr}(p) + \mathrm{Nr}(q)$. We define the norm of a nonzero matrix $A = (a_{i,j})$ as the greatest norm of its elements, $\|A\| = \max\limits_{i,j} \mathrm{Nr}(a_{i,j})$, and introduce the function ρ_A as

$$\rho_A = \begin{cases} \|A\| \ \text{ for } \ F[x], \\ \|A\| + \dfrac{1}{2}\log_w n \ \text{ for } \ \mathbb{Z}. \end{cases} \tag{2}$$

Here, n is the smaller dimension of A. The upper bound for the norm of the determinant of a matrix A of order n can be written in terms of ρ_A, such that

$$\mathrm{Nr}(\det A) \le n\rho_A. \tag{3}$$

For the ring $F[x]$, this inequality is evident; for the case of the ring $R = \mathbb{Z}$, it can be proved by means of Hadamard's inequality. Note also that the determinant of any submatrix A_0 of order k of matrix A is bounded from above: $\mathrm{Nr}(\det A_0) \le k\rho_A$. We will estimate algorithms by using two following computational models.

The best modern model (B-model). *The currently best algorithm for the multiplicative operations on integer N-bit numbers* [12] *has complexity*

$$\Lambda_B = O(N \log N \log\log N), \tag{4}$$

which is given in terms of multiplicative operations on machine words, and the complexity of the best algorithm [13] *for calculating the greatest common divisor (GCD) for N-bit numbers is*

$$\Phi_B = O(\log^2 N \log\log N). \tag{5}$$

The same estimates take place for the number of multiplicative operations in the field F for polynomials of degree N from F[*x*] (*see* [2, 10] *for more detail*).

To get complexity estimates for intermediate algorithms, including the classical one, we will use the following model.

The common computational model (C-model). *We assume that there exist algorithms for the multiplication and division of numbers and polynomials with the norm less than that of an element a that have complexity*

$$\Lambda_C \le \eta \mathrm{Nr}^\delta(a) + o(\mathrm{Nr}^\delta(a)), \tag{6}$$

which is expressed in terms of multiplicative operations on machine words for $\mathbb{Z}$ *and in terms of multiplicative operations in the field F for F*[*x*]. *Here,* η *and* δ *are constants, and* $1 < \delta \le 2$. *We assume that there exists an Euclidean algorithm for finding the greatest common divisor of two elements a and b* ($\mathrm{Nr}(b) \le \mathrm{Nr}(a)$) *with the complexity*

$$\Phi_C \le \mathrm{Nr}^2(a)\log_2 w + O(\mathrm{Nr}(a)), \tag{7}$$

where, for $\mathbb{Z}$, *w is equal to the length of the machine word and, for F*[*x*], $w = 2$.

For $\eta = 1$ and $\delta = 2$, (6) is the estimate for the standard multiplication algorithm. For $\eta = 1$ and $\delta = \log_2 3$, it coincides with the estimate for the Karatsuba multiplication algorithm when a is a power of number 2.

For $F[x]$, estimate (7) is evident; for $\mathbb{Z}$, it follows from the well-known estimate [14]: *the maximum number of divisions in the Euclidean algorithm for numbers a and b* ($|b| \le |a|$) *does not exceed*

$$\lfloor 2\log_2 a \rfloor + 1. \tag{8}$$

It is known also [15] that, for a fixed b and an arbitrary a, the maximum number of divisions in the Euclidean algorithm does not exceed

$$(\log_2(3 + \sqrt{5}) - 1)^{-1}\log_2 b \ \ (\approx 0.72\log_2 b).$$

Numerical experiments show that the average number of divisions in the Euclidean algorithm for integers is $\dfrac{2}{1 + \sqrt{5}}\log_2 b$, and the quotients usually occupy one machine word. Therefore, to find the quotient, one division operation is, as a rule, sufficient. The finding of a residual requires as many operations as there are words in the number designation. This is discussed in a more detail in [16, 17].

Complexity estimates for the C-model will be denoted by the subscript C. They are given in terms of the number of multiplicative operations on machine words for the ring $\mathbb{Z}$ and in terms of the number of oper-

ations in the field F for the ring $F[x]$. Estimates obtained for the B-model will be denoted by the subscript B.

3. SOLUTION OF SYSTEMS BY THE p-ADIC METHOD IN THE FIELD OF QUOTIENTS

The general scheme of solving a determined system by the p-adic method has been suggested by Dixon in 1982 [3] and is as follows.

An appropriate prime element p of the ring R is selected. It is convenient to take an element of size equal to the length of one machine word, such that arithmetic operations modulo p could be performed without going beyond the limits of the maximum integer determined by the computational system. The element p must not be a divisor of the determinant of the matrix of coefficients. A random selection of p may be inappropriate. Since the numbers multiple of a prime p occur in the ring $\mathbb{Z}$ with the probability $1/p$, the random selection of p in $\mathbb{Z}$ may turn out inappropriate with the probability $1/p$. If the solution check shows that the solution obtained is not correct, the next prime element p is selected. A solution of the system in the ring of residues modulo p can be sought by any fast method for solving systems in a field with the complexity of the matrix multiplication, for example, by the recursive method suggested in [18].

It follows from Hadamard's inequality that the determinant of a matrix A of order n satisfies the inequality $|\det A| \le (\sqrt{n}\,\|A\|)^n$. This makes it possible to find upper bounds for the numerator and denominator of the solution, which, in turn, allows us to find the estimate for the lifting boundary p^k. The lifting boundary must be not less than the doubled product of the greatest numerator and the greatest denominator of the solution vector in order that any fraction (with regard to its sign) could be reconstructed ($p^k \ge 2(\sqrt{n}\,\|A\|)^{2n}$). Then, the solution is lifted modulo p up to p^k, and the rational solution is reconstructed. A detailed discussion of the p-adic methods can be found in [16].

The algorithms discussed in this paper are based on Dixon's algorithm for solving a determined system by means of linear p-adic lifting, Wang's algorithm of a rational fraction reconstruction [19], and the general method of reduction of solution of an arbitrary system to solving determined systems, as well as on Theorems 1 and 2.

Algorithm RationalReconstruction
Input: Module μ and remainder $u \in \mathbb{Z}_\mu$.

Output: The pair of integers (a, b), $b > 0$, such that
$a \equiv bu \pmod{\mu}$, $|a|, |b| < \sqrt{\mu/2}$.
$(a_1, a_2) := (\mu, u)$; $(v_1, v_2) := (0, 1)$;
while *TRUE* **do**
  if $v_2 \ge \sqrt{\mu/2}$ **then return NIL**;
  if $a_2 < \sqrt{\mu/2}$ **then return** $(\mathrm{sign}(v_2)a_2, |v_2|)$;
  $q := \lfloor a_1/a_2 \rfloor$; $(a_1, v_1) := (a_1, v_1) - q(a_2, v_2)$;
  PERMUTATION$((a_1, v_1) \leftrightarrow (a_2, v_2))$;
end while

Fig. 1. Wang's algorithm for reconstruction of rational numbers.

3.1. Fraction Reconstruction

The reconstruction of a rational number a/b by a given module μ (b and μ are mutually prime) and a remainder u is the calculation of integers a and b such that their absolute values are not greater than $\sqrt{\mu/2}$ and $a \equiv bu \bmod \mu$.

An algorithm for the fraction reconstruction has been suggested by Wang [19]; later, it was improved by Collins and Encarnacion [20]. In Fig. 1, Wang's algorithm [19] is presented.

For numbers μ and u, Wang's algorithm follows the Euclidean algorithm and calculates a sequence of remainders (a_1, a_2). Simultaneously, an additional sequence (v_1, v_2) beginning with the pair (1, 0) that calculates one of two coefficients in the extended Euclidean algorithm is constructed. The computation is terminated when the magnitude of the next remainder is less than $\sqrt{\mu/2}$. In this case, a is the last remainder obtained, and b is the last term of the additional sequence. The complexity of Wang's algorithm is the same as that of the Euclidean algorithm.

3.2. The p-Adic Method for Solving Systems in the Field of Quotients

Let R be an Euclidean domain. Consider a system of linear equations over R,

$$Ax = c, \tag{9}$$

where the matrix of coefficients $A \in R^{n \times m}$ has rank r. Let S and T be permutation matrices. We represent matrices SAT and Sc in the block form as

$$SAT = \begin{pmatrix} A_0 & A_1 \\ A_2 & A_3 \end{pmatrix}, \quad Sc = \begin{pmatrix} c_0 \\ c_1 \end{pmatrix}, \quad c_0 \in R^r, \tag{10}$$

where A^0 is an $r \times r$ submatrix and $\det A_0 \ne 0$.

Consider first the *homogeneous system* ($c = 0$).

3.2.1. The p-adic method for solving homogeneous systems in the field of quotients.

Theorem 1. *Let system* (9) *be homogeneous and* $a_j \in R^r$ *be columns of the matrix* A_1; i.e., $A_1 = (a_1, a_2, \ldots, a_{m-r})$. *Then, the systems* $A_0 x_j = -a_j$, $j = 1, \ldots, m - r$, *are*

determined, and their solutions $x_j \in K^r$ form a basis set for solutions of system (9),

$$\left\{ T\begin{pmatrix} x_j \\ e_j \end{pmatrix}, \quad j = 1, \ldots, m-r \right\},$$

where e_j are columns of the identity matrix $I_{m-r} = (e_1, e_2, \ldots, e_{m-r})$.

Proof. Since $\det A_0 \neq 0$, all systems $A_0 x_j = -a_j$ are determined. Introduce the notation $y = T^{-1}x$. By the assumption of the theorem, $(A_0, A_1)y = 0$. If we seek the solution in the form $y_j = \begin{pmatrix} x_j \\ e_j \end{pmatrix}$, we arrive at the system $A_0 x_j = -a_j$. The linear independence of the vectors y_j follows from linear independence of the vectors e_j, $j = 1, \ldots, m-r$.

The corresponding algorithm is shown in Fig. 2.

In this algorithm, estimates for the boundary of the p-adic lifting $L_j^H[A]$ are calculated separately for each system, which requires estimates for the numerator and denominator of the solution of the system. For $\mathbb{Z}$, the latter are calculated by Hadamard's inequality.

This algorithm uses the following functions:

$D = \sum_{k=1}^{n} \mathrm{Nr}(a_k)$, the estimate for the denominator in $F[x]$;

$D = \frac{n}{2}\log_w n + \sum_{k=1}^{n} \mathrm{Nr}(a_k)$, the estimate for the denominator in $\mathbb{Z}$;

Algorithm RationalBasisH[**A**]
Input: $A \in R^{n \times m}$
Output: Solutions of system $Ax = 0$.
p := *FirstPrime*;
repeat
  Initialization of the algorithm:
  $\{\tilde{A}_0^{-1}, S, T, r\}$:= *DiagonalFormH*[$mod_p A$];
  if $m = r$ **then return**$\{\varnothing\}$;
  $\begin{pmatrix} A_0 & A_1 \\ A_2 & A_3 \end{pmatrix}$:= *SAT*; $(A_0', a_0) := A_0$; $(a_1, \ldots, a_{m-r}) := A_1$;
  $j := 1$; *Test* := *TRUE*;
  repeat
    p-adic lifting:
    $l := 1$; $x := 0$; $c^* := -a_j$;
    repeat
      $\tilde{x} := mod_p \tilde{A}_0^{-1} c^*$; $c^* := (c^* - A_0\tilde{x})/p$; $x := x + l\tilde{x}$; $l := lp$;
    until $\mathrm{Nr}(l) \geq L_j^H[A]$;
    $\begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix}$:= *RationalReconstruction*[x, N_j^H, D];
    Solution check:
    Comment: $(e_1, \ldots, e_{m-r}) = I_{m-r}$
    $x_j := T\begin{pmatrix} \bar{x}_j \\ \xi_j e_j \end{pmatrix}$;
    if $Ax_j \neq 0$ **then** *Test* := *FALSE*;
    $j := j + 1$;
  until $j > m - r$ **or** *Test* = *FALSE*;
  p := *NextPrime*[p];
until *Test* = *TRUE*;
return $\{x_1, \ldots, x_{m-r}\}$.

Fig. 2. The algorithm for computing rational basis of solutions of a homogeneous system of linear equations by the p-adic method.

$N_j^H = D + \mathrm{Nr}(a_j) - \min_{k=1,\ldots,n} \mathrm{Nr}(a_k)$, the estimate for the numerator in $F[x]$;

$N_j^H = \frac{n}{2}\log_w n + D + \mathrm{Nr}(a_j) - \min_{k=1,\ldots,n} \mathrm{Nr}(a_k)$, the estimate for the numerator in $\mathbb{Z}$;

$L_j^H[A] = N_j^H + D$, the estimate for the boundary of the p-adic lifting in $F[x]$;

$L_j^H[A] = N_j^H + D + \log_w 2$, the estimate for the boundary of the p-adic lifting in $\mathbb{Z}$ (the last addend is used to take into account the sign);

NextPrime[p], the generator of prime elements of the ring R, which returns the next prime element after p;

FirstPrime returns the first prime element;

$\tilde{R} = R/pR$ denotes the quotient ring;

DiagonalFormH[$mod_p A$], the diagonalization algorithm in the ring $\tilde{R}$ for the homogeneous systems, which calculates $\{\tilde{A}_0^{-1}, S, T, r\}$;

RationalReconstruction[$\tilde{x}, N, D$], reconstruction of the solution vector in the quotient field of the ring R by the solution $\tilde{x}$ in the ring R/p^kR when the numerator and denominator are bounded by the numbers N and D (Wang's algorithm or the improved algorithm from [20]).

3.2.2. The p-adic method for solving nonhomogeneous systems in the quotient field. Now, we turn to the case of a *nonhomogeneous system* ($c \neq 0$).

Theorem 2. *Let* (9) *be a nonhomogeneous system of linear equations and S, T, and P be permutation matrices such that S and T satisfy* (10). *Introduce the notation*

$$(B, b) = PA_0^{-1}(A_1, c_0),$$

$$B = \begin{pmatrix} \bar{b}_1 & \ldots & \bar{b}_{m-r} \\ \beta_1 & \ldots & \beta_{m-r} \end{pmatrix},$$

$$b = \begin{pmatrix} \bar{b} \\ \beta \end{pmatrix}, \beta, \beta_j \in R.$$

Let matrix P be selected from the condition that β is nonzero element of the free-term column. Let J denote the set of numbers of columns of matrix B with zero elements in the last row, $J = \{j | \beta_j = 0, j = 1, \ldots, m-r\}$. Introduce the notation

$$Q = \begin{pmatrix} P & 0 \\ 0 & I_{m-r} \end{pmatrix},$$

$$U = I_{m-r+1} + \sum_{j \in J} E_{1, j+1},$$

$$W = \begin{pmatrix} I_{r-1} & 0 \\ 0 & U \end{pmatrix},$$

$$V = QW, \quad (\bar{A}_0, a_0, a_1, \ldots, a_{m-r}) = P(A_0, A_1)V,$$

$$A_0 \in R^{r \times (r-1)}, \ a_j \in R^r.$$

Then, the systems

$$(\bar{A}_0, a_j)x_j = Pc_0, \quad j = 0, 1, \ldots, m-r, \tag{11}$$

are determined; their solutions are the vectors

$$x_j = \begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix} = \begin{cases} \begin{pmatrix} I_{r-1} & -\bar{b}_j \\ 0 & 1 \end{pmatrix} PA_0^{-1}c_0, \text{ for } j \in J; \\ \begin{pmatrix} I_{r-1} & -\bar{b}_j\beta_j^{-1} \\ 0 & \beta_j \end{pmatrix} PA_0^{-1}c_0, \text{ for } j \notin J; \end{cases} \tag{12}$$

and the vectors

$$TV\begin{pmatrix} \bar{x}_j \\ \xi_j \bar{e}_j \end{pmatrix}, \quad j = 0, 1, \ldots, m-r, \tag{13}$$

where $\bar{e}_j$ are columns of the identity matrix $I_{m-r+1} = (\bar{e}_0, \bar{e}_1, \ldots, \bar{e}_{m-r})$, form a basis set of solutions of the system $Ax = c$.

Proof. Introduce the notation $y = V^{-1}T^{-1}x$. Then, by the assumption, $P(A_0, A_1)Vy = Pc_0$ and $P(A_0, A_1)V = (\bar{A}_0, a_0, a_1, \ldots, a_{m-r})$. If we seek the solution in the form $y = \begin{pmatrix} \bar{x}_j \\ \xi_j \bar{e}_j \end{pmatrix}$, we arrive at systems (11).

Let us show that systems (11) are determined. To this end, we multiply them from the left by $PA_0^{-1}P$. Since $P = P^{-1}$, we have

$$PA_0^{-1}P(\bar{A}_0, a_j)x_j = PA_0^{-1}c_0, \quad j = 0, 1, \ldots, m-r.$$

Since $b = PA_0^{-1}c_0$ and $PA_0^{-1}P(\bar{A}_0, a_0) = I_r$, the first system in (11) takes the form $I_r x_0 = b$. Introduce the notation $I_r = (\bar{I}, e)$, $e \in R^r$. Then, $PA_0^{-1}P\bar{A}_0 = \bar{I}$, and $PA_0^{-1}Pa_0 = e$.

By the assumption,

$$(\bar{A}_0, a_0, a_1, a_2, \ldots, a_{m-r})$$

$$= P(A_0, A_1)\begin{pmatrix} P & 0 \\ 0 & I_{m-r} \end{pmatrix}\begin{pmatrix} I_{r-1} & 0 \\ 0 & U \end{pmatrix}.$$

Algorithm RationalBasis[**A**, **c**]
Input: $A \in R^{n \times m}$, $c \in R^n$
Output: Solutions of system $Ax = c$ or *NIL*
$p := FirstPrime$; $i := 1$;
repeat
Initialization of the algorithm:
$\{\tilde{A}_0^{-1}, \tilde{B}, \tilde{b}, S, T, P, r, r_c\} := DiagonalForm[mod_p(A, c)]$;
if $r < r_c$ **then return** *NIL*;
Comments: $\tilde{B} = (\tilde{b}_1, \tilde{b}_2, \ldots, \tilde{b}_{m-r})$, $I_{m-r+1} = (\bar{e}_0, \bar{e}_1, \ldots, \bar{e}_{m-r})$,
Comments: $\tilde{b}_j = \begin{pmatrix} \bar{b}_j \\ \tilde{\beta}_j \end{pmatrix}$, $Q = \begin{pmatrix} P & 0 \\ 0 & I_{m-r} \end{pmatrix}$.
$\begin{pmatrix} A_0 & A_1 \\ A_2 & A_3 \end{pmatrix} := SAT$; $\begin{pmatrix} c_0 \\ c_1 \end{pmatrix} := Sc$;
$\hat{c} := Pc_0$; $(\bar{A}_0, a_0) := PA_0P$; $(a_1, \ldots, a_{m-r}) := PA_1$;
$J := \varnothing$;
for $j := 1$ **to** $m - r$ **do if** $\beta_j = 0$ **then** $J := J \cup \{j\}$;
$j := 0$; $Test = TRUE$;
repeat
if $j = 0$ **then** $A_0 := (\bar{A}_0, a_0)$; $\tilde{A}_0^{-1} := P\tilde{A}_0^{-1}$
else if $j \in J$
then $A_j := (\bar{A}_0, a_j + a_0)$; $\tilde{A}_j^{-1} := \begin{pmatrix} I_{r-1} & -\bar{b}_j \\ 0 & 1 \end{pmatrix}\tilde{A}_0^{-1}$
else $A_j := (\bar{A}_0, a_j)$; $\tilde{A}_j^{-1} := \begin{pmatrix} I_{r-1} & -\bar{b}_j\tilde{\beta}_j^{-1} \\ 0 & \tilde{\beta}_j^{-1} \end{pmatrix}\tilde{A}_0^{-1}$
p-adic lifting:
$l := 1$; $x := 0$; $c^* := \hat{c}$;
repeat
$\tilde{x} := mod_p \tilde{A}_j^{-1} c^*$;
$c^* := (c^* - A_j\tilde{x})/p$;
$x := x + l\tilde{x}$; $l := lp$;
until $\mathrm{Nr}(l) \geq L_j[A, c]$
$\begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix} := RationalReconstruction[x, N_j, D_j]$;
Solution check:
if $j = 0$ **then** $x_0 := T\begin{pmatrix} P\begin{pmatrix} \bar{x}_0 \\ \xi_0 \end{pmatrix} \\ 0 \end{pmatrix}$
else if $j \in J$
then $x_j := T\begin{pmatrix} P\begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix} \\ \xi_j e_j \end{pmatrix}$ **else** $x_j := T\begin{pmatrix} P\begin{pmatrix} \bar{x}_j \\ 0 \end{pmatrix} \\ \xi_j e_j \end{pmatrix}$;
if $Ax_j \neq c$ **then** $Test := FALSE$;
$j := j + 1$;
until $j > m - r$ **or** $Test = FALSE$
$i := i + 1$; $p := NextPrime[p]$;
until $Test = TRUE$ **or** $i = NumberOfPrimes$
if $Test = FALSE$ **then return** *NIL* **else return** $\{x_0, x_1, \ldots, x_{m-r}\}$.

Fig. 3. The algorithm for computing rational basis of solutions of a nonhomogeneous system of linear equations by the *p*-adic method.

Hence, it follows that $PA_0^{-1}P(a_0, \ldots, a_{m-r}) = PA_0^{-1}P(a_0, PA_1)U = (e, B)U$. Denote by $d_j, j = 1, \ldots, m-r$, the columns of the matrix $(e, B)U$. Since $U = I_{m-r+1} + \sum_{j \in J} E_{1, j+1}$ and J is the set of numbers of columns in the matrix B with zero elements in the last row, the multiplication by the matrix U implies that the column e is added to each column of the matrix (e, B) with the number $j \in J$. Therefore, the last row of the matrix $(e, B)U$ does not contain zero elements:

$$d_j = \begin{cases} b_j & \text{for } j \notin J, \\ b_j + e & \text{for } j \in J. \end{cases}$$

Hence, after the multiplication by $PA_0^{-1}P$, system (11) takes the form

$$(\bar{I}, d_j)x_j = b, \quad j = 0, 1, \ldots, m-r, \qquad (14)$$

with $\det(\bar{I}, d_j) \neq 0$. Since the last element of the vector b is nonzero, solutions $x_j = \begin{pmatrix} \bar{x}_j \\ \xi_j \end{pmatrix}$ of the system satisfy the condition $\xi_j \neq 0$. Therefore, the vectors $\xi_j \bar{e}_j$, $j = 1, \ldots, m-r$, and, hence, vectors (13) are linearly independent. Multiplying them by the inverse of the matrix $(\bar{I}, d_j)$, we get solutions of the system in form (12).

Since $V = QW = \begin{pmatrix} P & 0 \\ 0 & I_{m-r} \end{pmatrix}\begin{pmatrix} I_{r-1} & 0 \\ 0 & U \end{pmatrix}$, calculating the product Vx_j, we get the basis set of solutions in the form

$$\left\{ T\begin{bmatrix} P\begin{bmatrix} \bar{x}_0 \\ \xi_0 \end{bmatrix} \\ 0 \end{bmatrix}, T\begin{bmatrix} P\begin{bmatrix} \bar{x}_j \\ \xi_j \end{bmatrix} \\ \xi_j e_j \end{bmatrix} \quad \forall j \in J, \right.$$

$$\left. T\begin{bmatrix} P\begin{bmatrix} \bar{x}_j \\ 0 \end{bmatrix} \\ \xi_j e_j \end{bmatrix} \quad \forall j \notin J \right\},$$

where e_j are columns of the identity matrix $I_{m-r} = (e_1, \ldots, e_{m-r})$.

The corresponding algorithm is presented in Fig. 3.

In this algorithm, estimates for the boundary of the p-adic lifting $L_j[A, c]$ are calculated separately for each system. For the ring $\mathbb{Z}$, they are obtained from Hadamard's inequality. In doing so, it is taken into account that the denominator is the minor of the matrix of coefficients, and the numerator is the minor of the augmented matrix. This allows us to get a finer estimate, which may be important when the free terms are small. In addition, it is taken into account that, when going from system j to system $j + 1$, only the last column (denoted as $\hat{a}_j$) of the matrix A is changed.

The algorithm shown in Fig. 3 uses additionally the following functions:

$D_j = \mathrm{Nr}(\hat{a}_j) + \sum_{k=1}^{n-1} \mathrm{Nr}(a_k)$, the estimate for the denominator in $F[x]$;

$D_j = \frac{n}{2}\log_w n + \mathrm{Nr}(\hat{a}_j) + \sum_{k=1}^{n-1} \mathrm{Nr}(a_k)$, the estimate for the denominator in $\mathbb{Z}$;

$N_j = D_j + \mathrm{Nr}(c) - \min_{k=1, \ldots, n-1, j} \mathrm{Nr}(a_k)$, the estimate for the numerator in $F[x]$;

$N_j = \frac{n}{2}\log_w n + D_j + \mathrm{Nr}(c) - \min_{k=1, \ldots, n-1, j} \mathrm{Nr}(a_k)$, the estimate for the numerator in $\mathbb{Z}$;

$L_j[A, c] = N_j + D_j$, the estimate for the boundary of the p-adic lifting in $F[x]$;

$L_j[A, c] = N_j + D_j + \log_w 2$, the estimate for the boundary of the p-adic lifting in $\mathbb{Z}$ (the last addend is used to take into account the sign);

MinPrime, the prime element with the least norm;

NumberOfPrimes = $\max(D_j)$, the number of prime elements in $F[x]$;

NumberOfPrimes = $\lfloor w^{\max(D_j)}/|\mathit{MinPrime}| \rfloor$, the number of prime elements in $\mathbb{Z}$;

DiagonalForm[$\mathit{mod}_p(A, c)$], the diagonalization algorithm in the ring $\tilde{R}$ for nonhomogeneous systems, which calculates $\{\tilde{A}_0^{-1}, \tilde{B}, \tilde{b}, S, T, P, r, r_c\}$; here, the tilde denotes matrices over the quotient ring $\tilde{R}$, and r and r_c are ranks of the matrices $\mathit{mod}_p A$ and $\mathit{mod}_p(A, c)$, respectively.

3.3. Complexity Estimate for the p-Adic Method for Solving Systems in the Quotient Field

We consider a system $Bx = c$ with n-by-m matrix of coefficients $A = (B, c)$ of rank n. The module p in the ring $\mathbb{Z}$ is selected from the condition that it is the largest number that can be stored in one machine word; in the ring $F[x]$, the module p is taken to be a polynomial of the first degree.

The complexity of multiplication of matrices of order n is denoted as $\gamma n^\beta + o(n^\beta)$, where γ and β are some constants, $3 \geq \beta > 2.376$. The constant γ may be viewed as a function of β. For example, $\gamma(3) = 1$ and $\gamma(\log_2 7) = 1$, when n is a power of 2.

The number of multiplicative operations on elements required to reduce an n-by-m matrix to the diagonal form by the recursive method is denoted as

$\gamma_\beta mn^{\beta-1} + o(mn^{\beta-1})$. The estimates for the function γ_β can be found in [18], e.g., $\gamma_3 = 1/3$ and $\gamma_{\log_2 7} = 7/15$.

Proposition 1. Solution in the quotient field. The complexity of the solution of a system of linear equations with an n-by-m matrix of coefficients A in the quotient fields $\mathbb{Z}$ and $F[x]$ by the p-adic method is estimated as follows:

$$C_{C_p} \le \gamma_\beta mn^{\beta-1} + 4n^3(m-n)\rho_A^2(\log_2 w + 1) + O(n^2(m-n)\rho_A) + o(mn^{\beta-1}), \quad (15)$$

$$C_{B_p} = O(mn^{\beta-1} + \rho_A^2 n^3(m-n) + \rho_A n^2(m-n)\log^2\rho_A n \log\log\rho_A n). \quad (16)$$

Proof. Consider the common computational model.

Stage 1. The matrix (A, c) is reduced to the diagonal form modulo p by the recursive method. The complexity of this operation is

$$C_1 = \gamma_\beta mn^{\beta-1} + o(mn^{\beta-1}).$$

Stage 2. The complexity of the computation of $m-n$ matrices A_j^{-1} modulo p is

$$C_2 = n^2(m-n).$$

Stage 3. The lifting up to p^k, which consists of $k = 2\rho_A n$ lifting steps. The computation of the product of a matrix and the vector $\tilde{x} = mod_p \tilde{A}_j^{-1} c^*$ requires n^2 operations, and the multiplication of the matrix A_j by the vector $\tilde{x}$, $n^2\|A\|$ operations. Here, for simplicity, the complexity of the multiplication algorithm (6) is estimated for $\delta = 2$ and $\eta = 1$. Thus, we have

$$C_3 = 2\rho_A n^3(\|A\| + 1)(m-n).$$

Stage 4. The complexity of reconstruction of $n(m-n)$ rational fractions by their remainders modulo $p^{2\rho_A n}$, with regard to estimate (7), is

$$C_4 \le n(m-n)(2\rho_A n)^2\log_2 w + O(n^2(m-n)\rho_A).$$

The total complexity of the algorithm is the sum of complexities of all stages, $C_1 + C_2 + C_3 + C_4$. Hence, we obtain

$$C_{C_p} \le \gamma_\beta mn^{\beta-1} + 2n^3(m-n)\rho_A(2\rho_A\log_2 w + \|A\| + 1) + o(mn^{\beta-1}) + O(n^2(m-n)\rho_A).$$

The desired estimate for the C-model follows from this inequality. The estimate for the B-model is obtained similarly with the use of (4) and (5).

These are estimates for both the ring of integers and for polynomials over a field. The difference is in the values of ρ_A and w. It should be noted that the solution obtained needs to be checked by substitution. The solution may occur incorrect with the probability $1/p$; i.e., the actual complexity may occur greater than C_p by k times with the probability $1/p^k$.

4. PROBABILISTIC METHOD FOR SOLVING SYSTEMS IN COMMUTATIVE DOMAINS

4.1. Commutative Domains with Identity

Let R be a commutative domain with identity, for which there is an algorithm for computing a basis of a finitely generated ideal, e.g., an algorithm of the computation of the Gröbner basis. For $x = (x_1, x_2, \ldots, x_s)$ and $y = (y_1, y_2, \ldots, y_s)$, we use the notation $\langle x, y\rangle = \sum_{i=1}^s x_i y_i$.

Let $Ax = c$ be a system of linear equations over R, and let

$$x_i = \chi_i^{-1}\boldsymbol{x}_i,\ \chi_i \in R\backslash 0,\ \boldsymbol{x} \in R^m,\ i = 1, 2, \ldots, h \quad (17)$$

be a rational basis of the set $\mathcal{M}$ of its solution. It is not difficult to show that the set $\mathcal{M}$ can be written as

$$\mathcal{M} = \left\{ \frac{\langle \hat{\boldsymbol{x}}, q\rangle}{\langle \chi, q\rangle} \middle| q \in R^h, \langle \chi, q\rangle \ne 0 \right\}. \quad (18)$$

Consider the probabilistic algorithm for finding a basis consisting of integer solutions of this system, i.e., solutions belonging to R^m.

Let an ideal generated by all denominators $\chi_1, \chi_2, \ldots, \chi_h$ of the rational basis (17) be a unit ideal, and let $(q_1, q_2, \ldots, q_h)$ be numbers such that $\sum_{i=1}^h q_i\chi_i = 1$. Then,

$$z = \sum_{i=1}^h q_i \boldsymbol{x}_i$$

is an integer solution of system $Ax = c$. If, in this case, $q_k \ne 0$, then

$$\{z, \boldsymbol{x}_i - z(\chi_i - 1), i = 1, 2, 3, \ldots, k-1, k+1, \ldots, h\}$$

are linearly independent integer solutions that form a basis set of solutions of the system $Ax = c$.

The algorithm shown in Fig. 4 uses the following functions.

The function *RationalBasis*$[A, c]$ computes a basis set of rational solutions $\{\boldsymbol{x}_1\chi_1^{-1}, \boldsymbol{x}_2\chi_2^{-1}, \ldots, \boldsymbol{x}_h\chi_h^{-1}\}$ for system $Ax = c$ and the number h of vectors in it.

The function $GI[(\chi_1, \chi_2, \ldots, \chi_h)]$ computes a generator of the ideal generated by the numbers $\chi_1, \chi_2, \ldots, \chi_h$. If this ideal is the principal ideal (*gcd*) generated by an invertible element, then multipliers $q_1, q_2, \ldots, q_h$ are also calculated such that $1 = \sum_{i=1}^h q_i\chi_i$, and the value

Algorithm IntegerBasis[**A**, **c**]
Input: $A \in R^{n\times m}$, $c \in R^n$
Output: Basis of integer solutions or *NIL*
$\{h, x_1\chi_1^{-1}, x_2\chi_2^{-1}, \ldots, x_h\chi_h^{-1}\} := RationalBasis[A, c]$;
if $h = 0$ **then return** NIL;
$(gcd, u_1, u_2, \ldots, u_h) := GI[(\chi_1, \chi_2, \ldots, \chi_h)]$;
if $gcd \neq 1$ **then return** NIL
else
$k := 1$; **while** $u_k = 0$ **do** $k := k + 1$;
$z_k := \sum_{i=1}^{h} x_i u_i$;
for $i = 1, 2, \ldots, k-1, k+1\ldots, h$ **do**
$z_i := x_i - x_k(\chi_i - 1)$;
return $\{z_1, z_2, \ldots, z_h\}$.

Fig. 4. The probabilistic algorithm for computing integer basis of solutions.

$gcd = 1$ is returned. If this ideal is not principal, then the value $gcd = 0$ is returned.

If this algorithm returns NIL, then the next iteration is to be executed.

For the next iteration, new permutation matrices S and T are randomly selected, and a new rational basis is constructed. The algorithm *GI* is to be applied to the denominators of all rational solutions obtained earlier.

4.2. Solution in Euclidean Domains by the p-Adic Probabilistic Method

If the domain is Euclidean, the algorithm *GI* is an extended Euclidean algorithm.

The idea to apply the probabilistic algorithm based on (18) for finding solutions of a Diophantine system of linear equations was suggested in [21] and applied to solving sparse systems; then, it was used in the work [9] for finding one solution of dense systems of Diophantine equations. Note that, for each new solution, a new inverse matrix was constructed.

In this paper, we apply the probabilistic method for finding all basis solutions, with the number of matrix inversions being reduced; namely, the finding of $m - n$ basis solutions requires one matrix inversion.

The estimate of the expected number of random rational solutions required for finding one integer solution in the ring of integers is $O(\rho_A)$ [9]. One iteration results in $m - n$ rational solutions; therefore, the expected number of iterations in the ring of integers is $O(\rho_A/(m - n))$. These estimates are related to matrices of coefficients of general form. In some cases, matrices of coefficients are inconvenient for the probabilistic method, which fails to find a solution for such systems. This is the case, for example, for systems with almost diagonal matrices: the denominators of rational solutions have common multipliers (diagonal elements).

The complexity of computing a basis set of solutions in the ring is determined by the number of operations in the algorithm for solving systems in the quotient field and by that in the algorithm for computing a basis of a finitely generated ideal.

The asymptotic complexity of the former is $O(mn^{\beta-1})$, and that of the latter is $O(m(m - n) + C_G)$ operations in the ring R. Here, C_G is the number of operations required for expanding the identity into $m - n$ generators of the unit ideal in the ring R. To expand the identity, one can take advantage of the algorithm for computation of Gröbner bases (see [16, 22–24]).

For the case of an *Euclidean domain R*, the complexity of computation of a rational basis is determined by (15) and (16). As to the complexity of finding the greatest common divisor for $m - n$ denominators whose norms do not exceed $n\rho_A$, it is, at least, n times less than the complexity of computation of a rational basis. Hence, it follows that the complexity of one iteration is determined by (15) and (16).

5. SOLUTION IN EUCLIDEAN DOMAINS BY THE *p*-ADIC DETERMINISTIC METHOD

Let a system $Bx = c$ satisfy the following conditions: $A = (B, c) \in \mathbf{R}^{n\times m}$, $c \in \mathbf{R}^n$, $n < m$, $\operatorname{rank} A = n$, and let $k = m - n$ be the number of linearly independent solutions. Let $A = (A_0, A_1)$, where A_0 is a square matrix, $\det A_0 \neq 0$, and the last row of the matrix $A_0^{-1}(A_1, c)$ does not contain zeros. The general case can be reduced to this one, like in Theorem 2.

Denote by $\boldsymbol{e}_j$ the columns of the identity matrix $I_k = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_k)$ of order k. If the solution of system $Ax = c$ is sought in the form $\begin{pmatrix} x_j \\ \xi_j \mathbf{e}_j \end{pmatrix}$, where $x_j \in \mathbf{K}^{n-1}$ and $\xi_j \in \mathbf{K}\backslash\{0\}$, $j = 1, \ldots, k$, we arrive at a determined system of order n. Each determined system is solved by the p-adic method. As a result, we get a basis set of solutions in the form

$$\left\{\begin{pmatrix}\mathbf{v}_1\\ \alpha\boldsymbol{e}_1\end{pmatrix}\delta_1^{-1}, \begin{pmatrix}\mathbf{v}_2\\ \alpha\boldsymbol{e}_2\end{pmatrix}\delta_2^{-1}, \ldots, \begin{pmatrix}\mathbf{v}_k\\ \alpha\boldsymbol{e}_k\end{pmatrix}\delta_k^{-1}\right\}.$$

Introduce the notation $\mathbf{u} = (\delta_1, \ldots, \delta_k)$, $\mathbf{u} \in \mathbf{R}^k$, and $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k)$, $V \in \mathbf{R}^{(n-1)k}$. Let us take advantage of the results obtained in the work [11].

Theorem 3. *A system $Ax = c$ is consistent if and only if the system*

$$\begin{cases} Vy = 0 \bmod \alpha, \\ \mathbf{u}y = \alpha \end{cases} \tag{19}$$

is consistent. If Y is a set of solutions of system (19), *then* $X = \left\{ \begin{pmatrix} y \\ Vy\alpha^{-1} \end{pmatrix} \middle| y \in \mathrm{Y} \right\}$ *is a set of solutions of system* $Ax = c$.

Thus, the problem is reduced to solving a homogeneous system of dimension $(n-1)$-by-$(m-n)$ in the quotient ring. The corresponding algorithm is presented in Fig. 5.

Elements of the quotient ring modulo α, where α is a minor of order n, are represented by elements with the norm $O(n\rho_A)$. Consider two algorithms of multiplication of matrices of order n in this quotient ring.

The first algorithm uses fast algorithms of matrix and number multiplication. Depending on the complexity of these algorithms, the complexity of the matrix multiplication is estimated between $O(n^5\rho_A^2)$ and $O^{\sim}(n^{\beta+1}\rho_A)$.

The second algorithm uses modular arithmetic and needs $O(n\rho_A)$ simple modules. The multiplication of matrices by each simple module requires n^3 operations. Finally, the reconstruction of the result requires reconstruction of n^2 numbers, each of which requires $O(n^2\rho_A^2)$ operations. Hence, the complexity of this algorithm can be estimated as $O(n^4\rho_A^2)$.

Both algorithms terminate by computing the remainders modulo α for all n^2 elements of the matrix. If the standard algorithm for multiplication and division of numbers is used, $O(n^2\rho_A^2)$ operations are required; the use of the fast algorithms for multiplication and division of numbers results in $O^{\sim}(n\rho_A)$ operations per element.

Complexity estimates for the algorithms for solving systems of linear equations in the quotient ring of the Euclidean ring are given in [11]. Based on them, we will give without proof estimates of complexity of solving systems of linear equations in Euclidean domains for two cases. In the first case, the standard arithmetic and fast multiplication algorithm are used. In the second case, matrix multiplication is implemented by means of the modular approach.

Proposition 2. The complexity of solving systems of linear equations with an n-by-m matrix of coefficients A in the rings $\mathbb{Z}$ and $F[x]$ by the p-adic method with the use of the fast multiplication algorithm is estimated as follows:

$$C_{C,s}^{m-n \ll n} \le \gamma_\beta n^{\beta-1} m = 7\rho_A^2 n^3 (m-n)(\log_2 w + 1) + 2\gamma\eta(\lambda_\beta + 4)\rho_A^\delta n^{\delta+1}(m-n)^{\beta-1} + o(n^{\beta-1}m + \rho_A n^\beta + \rho_A^\delta n^{\delta+1}(m-n)^{\beta-1});$$

Algorithm IntegerBasisDeterministic[**A**, **c**]
Input: $A \in R^{n\times m}$, $c \in R^n$
Output: Basis of integer solutions or $\varnothing$

$$\left\{ \begin{pmatrix} \mathbf{v}_1 \\ \alpha \boldsymbol{e}_1 \end{pmatrix} \delta_1^{-1}, \begin{pmatrix} \mathbf{v}_2 \\ \alpha \boldsymbol{e}_2 \end{pmatrix} \delta_2^{-1}, \ldots, \begin{pmatrix} \mathbf{v}_k \\ \alpha \boldsymbol{e}_k \end{pmatrix} \delta_k^{-1} \right\} := \mathit{RationalBasis}[A, c];$$

Comments: $\mathbf{u} = (\delta_1, \ldots, \delta_k)$, $V = (\mathbf{v}_1, \ldots, \mathbf{v}_k)$.
$Y := \mathit{SolutionOfSystem}(Vy = 0 \bmod \alpha, \mathbf{u}y = \alpha)$;
if $Y = \varnothing$ **then return** $\varnothing$

else return $\left\{ \begin{pmatrix} y \\ Vy\alpha^{-1} \end{pmatrix} \mid y \in Y \right\}$.

Fig. 5. The deterministic algorithm for computing integer basis of solutions.

$$C_{C,s}^{n\times 2n} \le 7n^4\rho_A^2(\log_2 w + 1) + \gamma\eta\psi_\beta\rho_A^\delta n^{\delta+\beta} + o(\rho_A^\delta n^{\delta+\beta});$$

$$C_{C,s}^{m \ll n} \le 7\rho_A^2 m n^3(\log_2 w + 1) + 2\gamma\eta(\lambda_\beta + 6)\rho_A^\delta n^{\beta+\delta-1} m + o(\rho_A^\delta n^{\beta+\delta-1} m);$$

$$C_{B,s} = O(\rho_A^2 n^3 m + \rho_A(n^\beta + n^2 \log \rho_A n) m \log \rho_A n \log\log \rho_A n).$$

Proposition 3. The complexity of solving systems of linear equations with an n-by-m matrix of coefficients A in the rings $\mathbb{Z}$ and $F[x]$ by the p-adic method with the use of the modular matrix multiplication algorithm is estimated as follows:

$$C_C^{m-n \ll n} \le \gamma_\beta n^{\beta-1} m + 4\rho_A^2 n^3 (m-n)(\log_2 w + 1) + 2\gamma(\lambda_\beta + 6)\rho_A n (m-n)^\beta + \rho_A^2 n (m-n)^3 (3\log_2 w + 49\log_2(m-n) + 28) + o(\rho_A^2 n^3 (m-n) + n^{\beta-1} m);$$

$$C_C^{n\times 2n} \le 2\gamma\psi_\beta\rho_A n^{\beta+1} + \rho_A^2 n^4 (7\log_2 w + 49(\log_2^2 n + 2\log_2 n) + 4) + o(\rho_A^2 n^4);$$

$$C_C^{m \gg n} \le 2\gamma(\lambda_\beta + 6)\rho_A n^\beta m + \rho_A^2 n^3 m(7\log_2 w + 49\log_2 n + 46) + o(\rho_A^2 n^3 m);$$

$$C_B = O(mn^{\beta-1} + \rho_A^2 n^3 (m-n) + \rho_A n^2 (m-n)\log^2 n \log^2 \rho_A n \log\log \rho_A n).$$

Here, the estimate for the B-model (4), (5) (which is denoted by the subscript B) is obtained for the general case of arbitrary n and m. The estimate for the common computational model (6), (7) (denoted by the subscript C) is obtained for three particular cases. The superscript

stands for three types of matrices of coefficients: a quasi-square matrix ($m - n \ll n$); an n-by-$2n$ matrix ($n \times 2n$); and a matrix the number of columns in which is much greater than the number of rows ($m \gg n$), when we can neglect n as compared to m. In the above estimates,

$$\lambda_\beta = \frac{7}{(2^{\beta-2}-1)}, \quad \psi_\beta = \frac{7 \times 2^{\beta-1}}{(2^\beta-2)(2^{\beta-1}-2)}.$$

6. CONCLUSION

The traditional method of solving systems of linear equations in Euclidean domains suggests the unimodular transformation of the augmented matrix to the Smith form or the transformation of the matrix of coefficients augmented by the unit vector to the Hermite form. In both cases, the basic difficulty is an uncontrolled growth of coefficients appearing in intermediate calculations when computing the unimodular multiplier.

The main distinctive feature of the suggested method is that coefficients in the intermediate calculations do not grow. Each coefficient appearing in the course of computation is either a minor of the original matrix of coefficients or a number in the ring of residues modulo this minor. In both cases, the estimate of its magnitude can easily be derived from Hadamard's inequality.

The p-adic methods work very well in many computer algebra problems, and the best method for solving determined systems in Euclidean domains is just the p-adic method.

Having this in mind, an attempt to use the p-adic method for finding a basis set of solutions in the quotient field seems quite natural. Each solution vector is a ratio of numbers, which are not greater than the corresponding minors of the original matrix.

The basis set in the domain can further be computed either by the probabilistic method by obtaining one integer solution and reconstructing the rational basis to an integer one or by the deterministic method by going to an equivalent system in the ring of residues modulo the leading minor. This, "natural," algebraic approach turned out quite justified.

Note also that, in this paper, we considered only dense systems, i.e., systems that are not assumed to have many zero elements or a matrix of coefficients of a special form. Methods for solving sparse systems are discussed in [25, 26], and relevant parallel algorithms, in [26, 27].

ACKNOWLEDGMENTS

The author thanks the reviewer for valuable comments on the original version of this manuscript, which made it possible to considerably improve the paper.

REFERENCES

1. Labahn, G. and Storjohann, A., Asymptotically Fast Computation of Hermite Normal Forms of Integer Matrices, *Proc. of ISSAC'96*, ACM, 1996, pp. 259–266.
2. Aho, A.V., Hopcroft, J.E., and Ullman, J.D., *The Design and Analysis of Computer Algorithms*, Reading: Addison–Wesley, 1975. Translated under the title *Postroenie i analiz vychislitel'nykh algoritmov*, Moscow: Mir, 1979.
3. Dixon, J., Exact Solution of Linear Equations Using p-Adic Expansions, *Numer. Math.*, 1982, vol. 40, pp. 137–141.
4. Smith, H.J.S., On System of Linear Indeterminate Equations and Congruences, *Phil. Trans. Royal Soc. (London)*, 1861, vol. 151, pp. 293–326.
5. Smith, H.J.S., On the Arithmetical Invariants of a Rectangular Matrix of Which the Constituents are Integral Numbers, *Proc. London Math. Soc.*, 1873, vol. 4, pp. 236–349.
6. Gregory, R.T. and Krishnamurthy, E.V., *Methods and Applications of Error-free Computation*, Berlin, 1984.
7. Hurt, M.F. and Waid, C.A., A Generalized Inverse Which Give All the Integral Solutions to a System of Linear Equations, *SIAM J. Appl. Math.*, 1970, vol. 10, pp. 547–550.
8. Hafner, J.L. and McCuley, K.S., Asymptotically Fast Triangularization of Matrix Over Rings, *SIAM J. Comput.*, 1991, vol. 20, no. 6, pp. 1068–1083.
9. Mulders, T. and Storjohann, A., Diophantine Linear System Solving, *Proc. of ISSAC'99*, Vancouver: ACM, 1999, pp. 181–188.
10. Von zur Gathen, J. and Gerhard, J., *Modern Computer Algebra*, Cambridge Univ. Press, 1999.
11. Malaschonok, G.I., Solution of Systems of Linear Diophantine Equations, *Proc. of Conf. on Computer Algebra in Scientific Computing, CASC 2001*, Springer, 2001, pp. 401–415.
12. Schönhage, A. and Strassen, V., Schnelle Multiplikation Grosser Zahlen, *Computing*, 1971, vol. 7, pp. 281–292.
13. Schönhage, A., Schnelle Berechnung von Kettenbruchentwicklungen, *Acta Informatica*, 1971, vol. 1, pp. 139–144.
14. Abramov, S.A., Some Estimates Related to the Euclidean Algorithm, *Zh. Vychisl. Mat. Mat. Fiz.*, 1979, vol. 19, pp. 756–760.
15. Abramov, S.A., A Note on the Number of Division Steps in the Euclidean Algorithm, *SIGSAM Bull.*, 2000, vol. 34 (Issue 134), pp. 1–2.
16. Buchberger, B., Collins, G.E., and Loos, R., *Computer Algebra. Symbolic and Algebraic Computation*, New York: Springer, 1982. Translated under the title *Kompyuternaya algebra. Simvolnye i algebraicheskie vychisleniya*, Moscow: Mir, 1986.
17. Knuth, D.E., *The Art of Computer Programming*, vol. 2: *Seminumerical Algorithms*, Reading: Addison-Wesley, 1969. Translated under the title *Iskusstvo programmirovaniya dlya EVM*, tom 2: *Poluchislennye algoritmy*, Moscow: Mir, 1977.
18. Malaschonok, G.I., Recursive Method for the Solution of Systems of Linear Equations Computational Mathematics, *Proc. of the 15th IMACS World Congress*, vol. I

(Berlin, 1997), Berlin: Wissenschaft and Technik, 1997, pp. 475–480.

19. Wang, P.S., A *p*-Adic Algorithm for Univariante Partial Factors, *Proc. of SYMSAC'81*, ACM, 1981, pp. 212–217.
20. Collins, G.E. and Encarnacion, M.J., Efficient Rational Number Reconstruction, *J. Symbolic Computation*, 1995, vol. 20, pp. 287–297.
21. Giesbrecht, M., Efficient Parallel Solution of Sparse System of Linear Diophantine Equations, *Proc. of the Second Int. Symp. on Parallel Symbolic Computation, PASCO'97*, ACM, 1997, pp. 1–10.
22. Latyshev, V.N., *Kombinatornaya teoriya kolets. Standartnye bazisy* (Combinatorial Theory of Rings: Standard Bases), Moscow: Mosk. Gos. Univ., 1988.
23. Mikhalev, A.V. and Pankrat'ev, E.V., *Komp'yuternaya algebra. Vychisleniya v differentsial'noi i raznostnoi algebre* (Computer Algebra: Computation in Differential and Difference Algebras), Moscow: Mosk. Gos. Univ., 1989.
24. Pankrat'ev, E.V., *Komp'yuternaya algebra. Faktorizatsiya mnogochlenov* (Computer Algebra: Factorization of Polynomials), Moscow: Mosk. Gos. Univ., 1988.
25. Wiedemann, D., Solving Sparse Linear Equations over Finite Fields, *IEEE Trans. Information Theory*, 1986, vol. 32, pp. 54–62.
26. Kaltofen, E. and Pan, V., Processor Efficient Parallel Solution of Linear Systems over an Abstract Field, *Proc. of the 3rd Annu. ACM Symp. on Parallel Algorithm Architecture, SPAA'91*, New York: ACM, 1991, pp. 180–191.
27. Kaltofen, E., Blocked Iterative Sparse Linear System Solvers for Finite Fields, *Proc. of Symp. on Parallel Comput. Solving Large Scale Irregular Applications, Stratagem'96*, France: Sophia Antipolis, 1996, pp. 91–95.
28. Huang, X. and Pan, V.Y., Fast Rectangular Matrix Multiplication and Improving Parallel Matrix Computations, *Proc. of PASCO'97*, ACM, 1997, pp. 11–23.
29. Malaschonok, G.I., Effective Matrix Methods in Commutative Domains, in *Formal Power Series and Algebraic Combinatorics*, Springer, 2000, pp. 506–517.
30. Storjohann, A., Algorithms for Matrix Canonical Forms, *Ph D Dissertation*, Zurich: Swiss Federal Inst. of Technology, 2000.