ISSN 0361-7688, Programming and Computer Software, 2011, Vol. 37, No. 2, pp. 78–86. © Pleiades Publishing, Ltd., 2011.
Original Russian Text © S.A. Abramov, A. Gheffar, D.E. Khmelnov, 2011, published in Programmirovanie, 2011, Vol. 37, No. 2.

Rational Solutions of Linear Difference Equations: Universal Denominators and Denominator Bounds

S. A. Abramov[a], A. Gheffar[b], and D. E. Khmelnov[a]

[a] *Dorodnicyn Computing Center, Russian Academy of Sciences, ul. Vavilova 40, Moscow, 119991 Russia*
[b] *XLIM, Universit@e de Limoges, CNRS, 123, av. Thoma, 87060 Limoges, France*
e-mail: sergeyabramov@mail.ru, f_gheffar@yahoo.fr, dennis_khmelnov@mail.ru

Received September 15, 2010

Abstract—Complexities of some well-known algorithms for finding rational solutions of linear difference equations with polynomial coefficients are studied.

DOI: 10.1134/S0361768811020022

1. INTRODUCTION

Finding rational solutions (i.e., solutions that have form of rational functions) of linear difference equations is a part of many computer algebra algorithms. Study of methods for constructing such solutions is of interest from the computer algebra standpoint.

Let k be a field of characteristic 0. We will consider equations of the form

$$y(x+n) + a_{n-1}(x)y(x+n-1) + \ldots + a_1(x)y(x+1) + a_0(x)y(x) = \varphi(x), \quad (1)$$

$\varphi(x), a_1(x), \ldots, a_{n-1}(x) \in k(x), a_0(x) \in k(x)/\{0\}$. Clearing the denominators, we obtain

$$b_n(x)y(x+n) + \ldots + b_1(x)y(x+1) + b_0(x)y(x) = \psi(x), \quad (2)$$

$\psi(x), b_1(x), \ldots b_{n-1}(x) \in k[x], b_0(x), b_n(x) \in k[x]\backslash\{0\}$. The last equation will be written in the operator form $L(y) = \psi(x)$ with the operator

$$L = b_n(x)\phi^n + \ldots + b_1(x)\phi + b_0(x), \quad \phi(y(x)) = y(x+1). \quad (3)$$

The notation $f(x) \perp g(x)$ will denote coprime polynomials $f(x), g(x) \in k[x]$; if $F(x) \in k(x)$, then $\operatorname{den}F(x)$ is a monic polynomial (with the leading coefficient equal to 1) such that $F(x) = \dfrac{f(x)}{\operatorname{den}F(x)}$ for some $f(x) \in k[x]$, $f(x) \perp \operatorname{den}F(x)$. The set of monic irreducible polynomials from $k[x]$ is denoted as $\operatorname{Irr}(k[x])$. If $p(x) \in \operatorname{Irr}(k[x])$, $f(x) \in k[x]$, then $\operatorname{val}_{p(x)}f(x)$ is defined to be the greatest $m \in \mathbb{N}$ such that $p^m(x)|f(x)$ ($\operatorname{val}_{p(x)}0 = \infty$) and $\operatorname{val}_{p(x)}F(x) = \operatorname{val}_{p(x)}f(x) - \operatorname{val}_{p(x)}g(x)$ for $F(x) = \dfrac{f(x)}{g(x)}$, $f(x), g(x) \in k[x]$.

The very first algorithm for constructing all rational solutions of equations of form (2) was proposed in [2]. It was succeeded by the algorithm proposed in [3] and algorithms from [5, 7, 8, 10, 11] (the latter algorithms are applicable both in the scalar case and in the case of systems of linear equations). All known algorithms for finding rational solutions can be described based on the following RS scheme:

RS1: Construct a rational function $S(x)$ over k such that any rational over k solution of the original equation can be represented in the form $S(x)f(x)$ with $f(x) \in k[x]$.

RS2: Transform the original equation into an equation (also with polynomial coefficients and right-hand side) for which $f(x) \in k[x]$ is a solution if and only if $S(x)f(x)$ is a solution of the original equation.

RS3: Construct all polynomial solutions of the transformed equation.

Each rational function $S(x)$ that possesses the property specified in RS1 is called a *denominator bound* for the original equation.

If we separate the problem of construction of the denominator bound from the more general problem of construction of rational solutions (i.e., consider step RS1 only), we should, first of all, mention that the algorithms from [2, 3, 5, 7, 8, 10] construct $S(x)$ in the form $\dfrac{1}{U(x)}$, where $U(x)$ is a polynomial over k called *a universal denominator* for Eq. (2). The algorithm from [11] constructs a rational function, which will be denoted as $R(x)$. The numerator of $R(x)$ may have a positive degree; in this case, the numerator of any rational solution of the original equation is divisible by the numerator of function $R(x)$.

The very first algorithm from [2] is the slowest one and will not be further considered. Algorithm $\mathbf{A}_U$ for constructing universal denominators [10] was improved in [7], where it was shown that the new version $\mathbf{A}'_U$ of this algorithm has lesser complexity than

the algorithms from [3, 5, 8]. Its complexity is also less than that of $\mathbf{A}_U$. Algorithms $\mathbf{A}_U$, $\mathbf{A}'_U$, and algorithms from [3, 5, 8] find one and the same universal denominator $U(x)$. The algorithm from [11] was presented in [10] in a more general form (instead of complex poles of rational functions over $\mathbb{C}$, irreducible divisors of denominators of rational functions over an arbitrary field k of characteristic 0 are considered).

Using one of the algorithms $\mathbf{A}_U$, $\mathbf{A}'_U$, or $\mathbf{A}_B$ on step RS1 and some (one and the same) algorithm for finding polynomial solutions (for example, one of the algorithms suggested in [1, 6, 5, 8]) on step RS3, we obtain algorithm $\langle \mathbf{A}_U \rangle$, $\langle \mathbf{A}'_U \rangle$, $\langle \mathbf{A}_B \rangle$ for constructing all rational solutions. In Section 3, we will prove that, under natural assumptions on the algorithm for finding polynomial solutions used in the overall algorithm, complexity of algorithm $\langle \mathbf{A}_B \rangle$ is greater than that of algorithm $\langle \mathbf{A}'_U \rangle$. This fact will be experimentally confirmed in Section 4. This will allow us to conclude that, among all considered algorithms for finding rational solutions, $\langle \mathbf{A}'_U \rangle$ is preferable.

The concluding Section 5 deals with systems of the form

$$Y(x+1) = A(x)Y(x), \tag{4}$$

where $Y(x) = (Y_1(x), Y_2(x), \ldots, Y_n(x))^T$ and $A(x) = (a_{ij}(x)) \in \mathrm{Mat}_n(k(x))$. The inverse matrix $A^{-1}(x) = (\tilde{a}_{ij}(x)) \in \mathrm{Mat}_n(k(x))$ is assumed to exist. Let system $Y(x+1) = A(z)Y(x) + G(x)$ be given, where matrix $A(x)$ is the same as that in (4) and $G(x) \in k(x)^n$. Then, supplementing $Y(x)$ with the $(n+1)$-th component equal to 1, we can transform the system to a homogeneous system with an invertible matrix belonging to $\mathrm{Mat}_{n+1}(k(x))$ (see, for example, [11]). Therefore, we confine our consideration to only homogeneous systems. We consider modified algorithms $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$ designed for finding rational solutions

$$\begin{gathered} Y(x) = (Y_1(x), Y_2(x), \ldots, Y_n(x))^T, \\ Y_i(x) \in k(x), \quad i = 1, 2, \ldots, n. \end{gathered} \tag{5}$$

Advantages of modification $\langle \mathbf{A}'_U \rangle$ compared to modification $\langle \mathbf{A}_B \rangle$ are noted.

2. PRELIMINARIES

2.1. Set M

In algorithms $\mathbf{A}_U$, $\mathbf{A}'_U$, and $\mathbf{A}_B$, step RS1 begins with the construction of a finite set M of irreducible polynomials satisfying the condition: if, for some rational solution $y(x)$ of the original equation and $p(x) \in \mathrm{Irr}(k[x])$, inequality $\mathrm{val}_{p(x)}y(x) < 0$ holds, then $p(x) \in M$ [10]. This set of candidates to the irreducible divisors of the denominators of rational solutions is constructed based on the polynomials

$$V(x) = b_n(x-n), \quad W(x) = b_0(x) \tag{6}$$

(see (2)). If $p(x) \in \mathrm{Irr}(k[x])$ and $f(x) \in k[x]\backslash\{0\}$, we may consider the finite set

$$\mathcal{N}_{p(x)}(f(x)) = \{m \in \mathbb{Z} : p(x+m) | f(x)\}. \tag{7}$$

In the case of $\mathcal{N}_{p(x)}(f(x)) = \varnothing$, we define $\max\mathcal{N}_{p(x)}(f(x)) = -\infty$ and $\min\mathcal{N}_{p(x)}(f(x)) = +\infty$. The set M is defined as

$$\begin{gathered} M = \{p(x) \in \mathrm{Irr}(k[x]) : \min\mathcal{N}_{p(x)}(W(x)) \le 0, \\ \max\mathcal{N}_{p(x)}(V(x)) \ge 0\}. \end{gathered}$$

When constructing M, we find, first, complete factorization of polynomials $V(x)$ and $W(x)$ and, then, a finite set $Q \subset \mathrm{Irr}(k[x])$ such that $q(x) \in Q$ if and only if

$$\min\mathcal{N}_{q(x)}(W(x)) = 0, \quad \max\mathcal{N}_{q(x)}(V(x)) \ge 0.$$

Let $Q \ne \varnothing$ and $Q = \{q_1(x), q_2(x), \ldots, q_s(x)\}$, $s \ge 1$. For each $1 \le i \le s$, we consider

$$M_{q_i(x)} = \{q_i(x), q_i(x+1), \ldots, q_i(x+d_i)\}, \tag{8}$$

where

$$d_i = \max\mathcal{N}_{q_i(x)}(V(x)). \tag{9}$$

Then,

$$M = \bigcup_{i=1}^{s} M_{q_i(x)}. \tag{10}$$

In [11], instead of M, a different set of candidates denoted as $\bar{S}$ is considered. It is shown in [10] that $M \subseteq \bar{S}$ for $k = \mathbb{C}$ and that it often happens that M is a proper subset of set $\bar{S}$.

The number

$$d = \max\{d_1, d_2, \ldots, d_s\} \tag{11}$$

coincides with the *dispersion* $\mathrm{dis}(V(x), W(x))$ of polynomials (6), i.e., with the greatest integer m for which $\deg\gcd(V(x), W(x+m)) > 0$. If such integers do not exist, then, by definition, $\mathrm{dis}(V(x), W(x)) = -\infty$.

2.2. Algorithms $\mathbf{A}_U$ and $\mathbf{A}'_U$

For each $q_t(x+j) \in M$, algorithm $\mathbf{A}_U$ calculates

$$\begin{aligned} \gamma_{j,t} = \min\Bigg\{ &\sum_{i \in \mathbb{N}} \mathrm{val}_{q_t(x+j+i)}V(x), \\ &\sum_{i \in \mathbb{N}} \mathrm{val}_{q_t(x+j-i)}W(x) \Bigg\} \end{aligned} \tag{12}$$

and, then, finds

$$U(x) = \prod_{\substack{1 \le t \le s \\ 0 \le j \le d_t}} q_t^{\gamma_{j,t}}(x+j). \tag{13}$$

Polynomial $U(x)$ is a universal denominator for Eq. (2): any solution $y(x) \in k(x)$ can be represented as $\frac{f(x)}{U(x)}$, $f(x) \in k[x]$; hence, rational function $\frac{1}{U(x)}$ is a denominator bound for this equation.

Algorithm $\mathbf{A}'_U$ differs from $\mathbf{A}_U$ in that it takes into account the possibility of equality of exponents $\gamma_{j,t}$ for different (sometimes, many) j for a fixed t; algorithm $\mathbf{A}'_U$ performs arithmetic calculations based on formula (12) only when $q_t(x+j)$ divides $V(x)$ or $W(x)$.

2.3. Algorithm $\mathbf{A}_B$

Algorithm $\mathbf{A}_B$ relies on the fact that all rational solutions of the original equation satisfy the equations

$$y(x) = v_{N,n-1}(x)y(x-N) + \ldots + v_{N,0}(x)y(x-N-n+1) + v_{N,-1}(x), \quad (14)$$

$$y(x) = w_{N,n-1}(x)y(x+N) + \ldots + w_{N,0}(x)y(x+N+n-1) + w_{N,-1}(x), \quad (15)$$

$v_{N,-1}(x)$, $v_{N,0}(x)$, ..., $v_{N,n-1}(x)$, $w_{N,-1}(x)$, $w_{N,0}(x)$, ..., $w_{N,n-1}(x) \in k(x)$, $N = 1, 2, \ldots, d+1$ (see (11)), which are constructed for positive N by means of shifts of Eq. (1) and Gaussian eliminations. For example, when going from N to $N+1$ in (14), elimination $y(x-N)$ is used with the help of the shifted Eq. (1) written in the form

$$y(x-N) + a_{n-1}(x-n-N)y(x-N-1) + \ldots + a_0(x-n-N)y(x-n-N) - \varphi(x-n-N) = 0.$$

Let $p(x) \in \mathrm{Irr}(k[n])$ and N be a positive integer. Let us define $B(p(x), N)$ as the minimum value of function $\mathrm{val}_{p(x)}$ for all coefficients $v_{N,-1}(x)$, $v_{N,0}(x)$, ..., $v_{N,n-1}(x)$ occurring in (14). Similarly, we define $B(p(x), -N)$ as the minimum value of function $\mathrm{val}_{p(x)}$ for all coefficients $w_{N,-1}(x)$, $w_{N,0}(x)$, ..., $w_{N,n-1}(x)$ occurring in (15). Algorithm $\mathbf{A}_B$ successively constructs Eqs. (14) for $N = 1, 2, \ldots, d+1$ and, for each t such that $1 \le t \le s$ and $d_t \ge N-1$, finds $B(q_t(x+d_t-N+1))$, which yields a *left* lower bound for each value $\mathrm{val}_{q_t(x+d_t-N+1)}y(x)$. In a similar way, the algorithm constructs Eqs. (15) for $N = 1, 2, \ldots, d+1$ and, for each t such that $1 \le t \le s$ and $d_t \ge N-1$, finds $B(q_t(x+N-1), -N)$, which yields a *right* lower bound for each value $\mathrm{val}_{q_t(x+N-1)}y(x)$. Out of the two lower bounds for $\mathrm{val}_{q_t(x+j)}y(x)$, $t = 1, 2, \ldots, s$, $j = 0, 1, \ldots, d_t$, we take the maximum one and denote it as $\beta_{j,t}$. The rational function

$$R(x) = \prod_{\substack{1 \le t \le s \\ 0 \le j \le d_t}} q_t^{\beta_{j,t}}(x+j) \quad (16)$$

yields a denominator bound.

2.4. Indicial Equation at ∞

Step RS2 consists in the construction of the operator product LS and subsequent clearing denominators in the equation $(LS)(y) = \psi(x)$. This results in the equation

$$K(y) = g(x) \quad (17)$$

with polynomial coefficients and polynomial right-hand side. Equality $L(S(x)f(x)) = \psi(x)$ holds if and only if $K(f(x)) = g(x)$.

The discussion of step RS3 needs some preliminary comments. Operator L is made to correspond to the algebraic equation $I(\lambda) = 0$, which is called the *indicial equation* at ∞. We will further refer to it as simply the indicial equation, which does not result in any confusion in our context. The basic feature of equation $I(\lambda) = 0$ is as follows: if equation $L(y) = 0$ has solution $S(x) = \frac{s_1(x)}{s_2(x)}$, $s_1(x), s_2(x) \in k[x]$, then the integer

$$\mathrm{val}_\infty S(x) = \deg s_1(x) - \deg s_2(x)$$

is a root of the indicial equation [9]. In particular, if $f(x)$ is a polynomial solution of equation $L(y) = 0$, then $\deg f(x)$ is a root of the indicial equation, since, in this case, $\mathrm{val}_\infty f(x) = \deg f(x)$.

To construct $I(\lambda)$, we first need to write L in terms of powers of $\Delta = \phi - 1$ (for example, by substituting $\phi = \Delta + 1$ into L). For the operator

$$L = b_n(x)\phi^n + \ldots + b_1(x)\phi + b_0(x) = c_n(x)\Delta^n + \ldots + c_1(x)\Delta + c_0(x),$$

we set

$$\omega = \max_{0 \le j \le n}(\deg c_j - j),$$

$$I(\lambda) = \sum_{\substack{0 \le j \le n \\ \deg c_j - j = \omega}} \mathrm{lc}(c_j)\lambda^{\underline{j}}, \quad (18)$$

where $\lambda^{\underline{j}} = \lambda(\lambda-1)\ldots(\lambda-j+1)$. The number ω is called an *increment* of operator L. Clearly,

$$c_j(x) = \sum_{i=j}^{n} \binom{i}{j} b_i(x). \quad (19)$$

It is known (see, for example, [13]) that, if equation $L(y) = \psi(x)$, $\psi(x) \in k[x]$, has a polynomial solution, then the degree of this solution does not exceed

$$h = \max\{\deg\psi - \omega, \tilde{\lambda}\}, \quad (20)$$

where $\tilde{\lambda} = \max(\{d \in \mathbb{N} : I(d) = 0\} \cup \{-\infty\})$. The number h is referred to as the *height* of equation $L(y) = \psi(x)$.

To find the height of Eq. (17) on step RS3, we need the indicial equation of operator K, and, of course, this equation can be constructed directly by K. Here, it should be noted that there exists a simple relationship between the indicial equation $I(\lambda) = 0$ of operator L

and the indicial equation $\bar{I}(\lambda) = 0$ of operator K: up to a nonzero constant factor, polynomial $\bar{I}(\lambda)$ coincides with $I(\lambda + \mathrm{val}_\infty S(x))$ [9]. We will take advantage of this fact in Section 3. Note also that the multiplication of the operator from the left by a nonzero polynomial $u(x)$ increases the increment of the operator by $\deg u(x)$ and results in the multiplication of polynomial $I(\lambda)$ corresponding to the operator by $\mathrm{lc}\, u(x)$. Then, it follows that the multiplication of the equation $L(y) = \psi(x)$, $\psi(x) \in k[x]$, by a nonzero polynomial does not change the height of the equation.

2.5. Algorithms $\langle \mathbf{A}_U \rangle$, $\langle \mathbf{A}'_U \rangle$, and $\langle \mathbf{A}_B \rangle$

If, in constructing Eq. (17), we use $S(x) = \dfrac{1}{U(x)}$, and the polynomial $U(x)$ is obtained by means of $\mathbf{A}_U$, then Eq. (17) is called a *U-image* of the original equation $L(y) = \psi$. If a rational function $R(x)$ obtained by means of $\mathbf{A}_B$ is used, then it is called a *B-image* of this equation.

We assume that the algorithm for finding all polynomial solutions of Eq. (17) used on step RS3, first, calculates the height of this equation and, then, looks for all polynomial solutions having known that their degrees do not exceed the height found. To the best authors' knowledge, this is the way all algorithms of this kind work (the simplest among them implements the method of undetermined coefficients). In the general case, if the equation height is nonnegative, then the time required to determine the absence of polynomial solutions is about the same as that required for constructing such solutions if they exist.

As has already been noted in Section 1, the notation $\langle \mathbf{A}_U \rangle$, $\langle \mathbf{A}'_U \rangle$, and $\langle \mathbf{A}_B \rangle$ is used for algorithms for constructing rational solutions employing algorithms $\mathbf{A}_U$, $\mathbf{A}'_U$, and $\mathbf{A}_B$ on step RS1.

Remark 1. *The majority of equations have no rational solutions. However, when using scheme RS, their absence is determined only on step RS3, when two previous steps were completed. It is shown in [9] that, for homogeneous equations (with zero polynomial $\psi(x)$ in (2)), the absence of rational solutions can often be predicted much earlier. We will not focus on it, since, in the framework of study of algorithm complexities, of interest are the worst cases.*

3. COMPARISON OF COMPLEXITIES OF ALGORITHMS FOR FINDING RATIONAL SOLUTIONS

In [10], a combined size of equation was introduced, and it was established that, for any fixed size s, it is possible to find an equation E_s such that its U-image is some polynomial $U(x)$ whose degree is greater than that of all U-images of equations of size s and the B-image of equation E_s has form $\dfrac{1}{U(x)}$. On the one hand, this rebuts an assumption that significant expenditures of algorithm $\mathbf{A}_B$ on step RS1 are always compensated by insignificant expenditures on step RS3, so that total expenditures of $\langle \mathbf{A}_B \rangle$ never exceed total expenditures of algorithm $\langle \mathbf{A}_U \rangle$. On the other hand, this fact does not mean that complexity of algorithm $\langle \mathbf{A}_U \rangle$ is less than that of algorithm $\langle \mathbf{A}_B \rangle$, since we have no grounds to assume that equation E_s is the worst case for $\langle \mathbf{A}_U \rangle$: it is a priori possible that, for another equation E'_s of the same size s, algorithm $\langle \mathbf{A}_U \rangle$ on step RS1 will construct a polynomial $U'(x)$ of possibly lesser degree than $U(x)$ such that finding polynomial solutions (step RS3) for the corresponding U-image will require greater expenditures (for example, because the height of the U-image will be too high).

Below in this paper, we show that, under some natural assumptions about the algorithm for finding polynomial solutions used on step RS3, complexity of algorithm $\langle \mathbf{A}'_U \rangle$, which is a modified version of algorithm $\langle \mathbf{A}_U \rangle$, is less than that of $\langle \mathbf{A}_B \rangle$ (the concept of size in this case is refined). We also give some lower bound for the difference of these complexities.

Let operator L in the equation $L(y) = \psi(x)$ have form (3). Let us set $l = \{\deg b_0(x), \deg b_1(x), \ldots, \deg b_n(x)\}$ and define d as the corresponding dispersion (see (11)). Let also n denote $\mathrm{ord} L$ and h denote height (20) of the equation. The quadruple (l, d, n, h) is called a *combined size,* or simply *size*, of equation $L(y) = \psi(x)$. Further in this section, we consider only the cases where $d \geq 0$. For the size components of a U-image, we introduce the notation l_U, d_U, n_U, and h_U (it can easily be seen that $n_U = n$). The additional notation r_U is used for the degree of the right-hand side of the U-image. Assuming that the U-image has form (17), we have $r_U = \deg g(x)$.

Let $\mathcal{S}_{l,d,n,h}$ denote the set of all equations of size (l, d, n, h). From the description of algorithms $\mathbf{A}_U$ and $\mathbf{A}_B$, it is not difficult to see that, if $U(x)$ and $R(x)$ are results of application of these algorithms to some equation from $\mathcal{S}_{l,d,n,h}$, then $\deg U(x) \leq l(d+1)$ and $\deg \mathrm{den}\, R(x) \leq l(d+1)$.

Lemma 1. *Let the size of equation $L(y) = \psi(x)$ be (l, d, n, h). Then,*

(i) *the set M for this equation (see Section 2.1) has not more than $l(d+1)$ elements, and*

$$h_U \leq hl + (d+1),$$
$$l_U \leq l(n-1), \quad r_U \leq h + l(d+n);$$

(ii) *the height of the equation*

$$f(x+n+d)y(x+n)+(x+1)^{l}y(x+n-1)+\ldots+(x+1)^{l}y(x+1)+f(x)y(x)=(x+1)^{h+l}, \quad (21)$$

where

$$f(x)=\prod_{t=1}^{l}\left(x+\frac{1}{t+1}\right),$$

is equal to h (*hence*, *the size of this equation is* (*l*, *d*, *n*, *h*)); *the set M for this equation has l*(*d* + 1) *elements*; *the universal denominator U*(*x*) *obtained by algorithm* $\mathbf{A}'_U$ *has degree l*(*d* + 1); *and*

$$h_U = h + l(d+1), \quad r_U = h + l(d+n),$$

with the degree of each coefficient on the left-hand side of the *U*-image being equal to *l*(*n* − 1).

Proof. (i) The fact that *M* has not more than *l*(*d* + 1) elements follows from the structure of set (10).

Let *G*(*x*) be the least common multiple of the coefficients of operator $L\frac{1}{U(x)}$, and let *L* have form (2). Let

$$s = \min\{\deg\gcd(U(x), b_0(x)), \deg\gcd(U(x), b_n(x-n))\}.$$

Then,

$$\deg\operatorname{den}\frac{b_0(x)}{U(x)} \le \deg U(x) - s,$$

$$\deg\operatorname{den}\frac{b_n(x)}{U(x+n)} \le \deg U(x) - s,$$

and $\deg U(x) \le s(d+1)$. Hence, $\deg G(x) \le \deg U(x) + ns - 2s \le s(d+n-1) \le l(d+n-1)$, and we obtain $l_U \le l + \deg G(x) - \deg U(x) \le l(n-1)$.

It can be derived from the definition of the height of equation $L(y) = \psi(x)$ that

$$\deg\psi(x) \le h + \omega \le h + l. \quad (22)$$

From this inequality and $\deg G(x) \le l(d+n-1)$, it follows that $r_U \le h + l(d+n)$.

When clearing the denominators in the equation $\left(L\frac{1}{U(x)}\right)(y) = \psi(x)$, we may start from the multiplication of both sides of this equation by *G*(*x*). This yields the equation $L'(y) = \psi'(x)$, $\deg\psi'(x) = \deg\psi(x) + \deg G(x)$. Increment ω' of operator *L*' is equal to $\omega + \deg G(x) - \deg U(x)$. As has already been noted in Section 2.4, polynomial *I*'(λ) constructed for the operator *L*' coincides up to a nonzero scalar factor with $I\left(\lambda + \operatorname{val}_\infty\frac{1}{U(x)}\right)$, i.e., with $I(\lambda - \deg U(x))$. Therefore,

$$\omega' = \omega + \deg G(x) - \deg U(x),$$

$$\deg\psi'(x) = \deg\psi(x) + \deg G(x),$$

$$\tilde{\lambda}' = \tilde{\lambda} + \deg U(x),$$

where $\tilde{\lambda}' = \max(\{d \in \mathbb{N} : I'(d) = 0\} \cup \{-\infty\})$. The height of equation $L'(y) = \psi'(x)$ is equal to

$$\max\{\deg\psi'(x) - \omega', \tilde{\lambda}'\} = h + \deg U(x)$$

and, hence, does not exceed $h + l(d+1)$. Note that equation $L'(y) = \psi'(x)$ may differ from the *U*-image of equation $L(y) = \psi(x)$ by only nonzero polynomial factors. As has already been noted in the end of Section 2.4, the multiplication by such a factor does not change the height of the equation. Therefore, $d_U \le h + l(d+1)$.

(ii) It is easy to see that, for Eq. (21), the set *M* has *l*(*d* + 1) elements, and the degree of the universal denominator *U*(*x*) obtained by algorithm $\mathbf{A}'_U$ is equal to *l*(*d* + 1). The remaining part of the assertion can be checked directly. □

Further, we consider complexities $T_B(l, d, n, h)$ and $T_U(l, d, n, h)$ of algorithms $\langle\mathbf{A}_B\rangle$ and $\langle\mathbf{A}'_U\rangle$ in terms of the numbers of operations in field *k* performed in the worst case. It is easy to see that, if we exclude *h* from the components of the combined size (leaving only *l*, *d*, and *n*), then complexity of each algorithm will be equal to ∞, since, for given *l*, *d*, and *n*, the equation may have an arbitrary large height, resulting thus in arbitrary large expenditures on step RS3. The presence of component *h* excludes this possibility. Note also that the right-hand side ψ(*x*) satisfies (22); therefore, the expenditures required for constructing each equation of form (14) and (15) are bounded if the size of the original equation is fixed.

The set of all equations of size (*l*, *d*, *n*, *h*) that, among all equations from $\mathcal{S}_{l,d,n,h}$, require the greatest expenditures on the construction of polynomial solutions (or determination of their absence) of the corresponding *U*-image is denoted as $\mathcal{U}_{l,d,n,h}$. This set may consist of more than one equation.

Theorem 1. *Let the algorithm for finding polynomial solutions be such that Eq.* (21) *belongs to* $\mathcal{U}_{l,d,n,h}$ *for all admissible values of l*, *d*, *n*, *and h. Then*, *complexities* $T_B(l, d, n, h)$ *and* $T_U(l, d, n, h)$ *of algorithms* $\langle\mathbf{A}_B\rangle$ *and* $\langle\mathbf{A}'_U\rangle$ *satisfy the inequality* $T_B(l, d, n, h) > T_U(l, d, n, h)$, *with* $T_B(l, d, n, h) - T_U(l, d, n, h) = \Omega(dln)$.[1]

Before proving the theorem, we note that the assumption that Eq. (21) belongs to set $\mathcal{U}_{l,d,n,h}$ (i.e., the assumption that the execution of step RS3 of algorithm $\langle\mathbf{A}'_U\rangle$ applied to this equation requires maximum expenditures) is quite natural. We do not make specific which algorithm is used for finding polynomial solutions, but, by the assumption in Section 2.5, this algorithm uses equation height as a bound of solution degrees. According to Lemma 1, the height of the

[1] We take advantage of the Ω-notation adopted in the complexity theory, which is used for description of asymptotic lower bounds (whereas the *O*-notation is used for description of asymptotic upper bounds), see, for example, [12] and [4, §2] for detail.

U-image of Eq. (21) is as large as possible, and the U-image itself is most "cumbersome" compared to the U-images of other equations from $\mathcal{S}_{l,d,n,h}$.

Proof of Theorem 1. The set M for Eq. (21) has the greatest possible number of elements. The number of the elements of the set M that are divisors of $V(x)$ and $W(x)$ is equal to $2l$; i.e., it also reaches the greatest possible value (see the last paragraph in Section 2.2). Therefore, the expenditures of algorithm $\langle \mathbf{A}'_U \rangle$ on step RS1, when applied to Eq. (21), reach the largest possible value. Based on this and on the assumption that Eq. (21) belongs to set $\mathcal{U}_{l,d,n,h}$, we conclude that Eq. (21) considered as the input of algorithm $\langle \mathbf{A}'_U \rangle$ corresponds to the worst case. The difference $T_B(l, d, n, h) - T_U(l, d, n, h)$ is not less than the difference of the expenditures required for constructing all rational solutions of Eq. (21) by means of $\langle \mathbf{A}_B \rangle$ and $\langle \mathbf{A}'_U \rangle$. The application of $\mathbf{A}_B$ to (21) yields $R(x) = \dfrac{1}{U(x)}$, where $U(x)$ is the universal denominator obtained by algorithm $\mathbf{A}'_U$. This is because

$$\nabla_{N,-1}(x) = \frac{(x-n-N+2)^{h+2}}{f(x+d-N+1)}$$

in (14) and relation

$$\mathrm{val}_{p(x+d-N+1)}\nabla_{N,-1}(x) = -1$$

holds for any $p(x) \in M$ in the given case. Therefore, the expenditures of both algorithms on step RS3 when applied to (21) are the same. Consider construction of polynomial $U(x)$ by algorithm $\langle \mathbf{A}'_U \rangle$ and rational function $\dfrac{1}{U(x)}$ by algorithm $\langle \mathbf{A}_B \rangle$. For simplicity, we assume that the expenditures required for calculation of each $\gamma_{j,t}$ in (13) coincide with the expenditures required for calculation of $\beta_{j,t}$ in (16), as long as Eqs. (14) and (15) required for algorithm $\langle \mathbf{A}_B \rangle$ have already been constructed (true, as shown in [7], expenditures of algorithm $\mathbf{A}'_U$ on calculation of the exponents are very small).

By virtue of the fact that, upon writing Eqs. (21) in form (1), all $a_i(x)$ and $\varphi(x)$ have numerators and denominators to the power l, we find that, if, for example, Eq. (14) is constructed for some $1 \le N \le d+1$, then, when constructing a similar equation for $N+1$, just the current shift of Eq. (1) will require $\Omega(nl)$ operations in field k. Algorithm $\mathbf{A}_B$ constructs such equations for $N = 1, 2, \ldots, d+1$, which proves the theorem. □

Remark 2. *The above-proved assertion on the difference $T_B(l, d, n, h) - T_U(l, d, n, h)$, most likely, can considerably be strengthen. When estimating $T_B(l, d, n, h) - T_U(l, d, n, h) = \Omega(ldn)$, we did not take into account "swelling" of the coefficients of Eqs.* (14) *and* (15) *as N increases, owing to which the difference $T_B(l, d, n, h) - T_U(l, d, n, h)$ for fixed n and l and increasing d grows faster than d. Our goal was only to show that the difference $T_B(l, d, n, h) - T_U(l, d, n, h)$ is positive and that it grows when each component n, l, or d of the equation size grows.*

4. EXPERIMENTAL COMPARISON

We carried out experimental comparison of algorithms $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$. To this end, they have been implemented in the computer algebra system Maple [14]. On step RS1, the implementation of algorithm $\langle \mathbf{A}'_U \rangle$ uses the implementation of algorithm $\mathbf{A}'_U$ described in [7], and the implementation of algorithm $\langle \mathbf{A}_B \rangle$ uses the implementation of algorithm $\mathbf{A}_B$. The latter was written specifically for this experiment in accordance with the description presented in Section 2.3. On step RS3, the implementations of both algorithms $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$ use procedure **polysols** from the standard package **LREtools** available in Maple. The procedure is designed for finding polynomial solutions of difference equations.

We carried out three experiments.

4.1. Experiment 1

In the first experiment, equations of form (21) from Lemma 1 were used. Altogether, 27 equations were used in the experiment. For all equations, $h = 6$, and the other parameters took three different values each: $n = 3, 6, 9$; $l = 2, 4, 6$; and $d = 5, 10, 15$. Each equation was solved by $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$. Results of this experiment are presented in Table 1.

Columns of the table correspond to different values of d, and each row corresponds to a pair of values of n and l. Each cell shows the difference of times spent for solving the corresponding equation by algorithms $\langle \mathbf{A}_B \rangle$ and $\langle \mathbf{A}'_U \rangle$ and (in parenthesis) the ratio of this difference to d.

The experiment illustrates the assertion proved in Theorem 1 and confirms the assumption made in Remark 2 regarding that, for fixed n and l, the difference $T_B(l, d, n, h) - T_U(l, d, n, h)$ grows faster than d.

Parameter h is not shown in the table, since its value was fixed for all equations and equal to 6. We carried out additional experiments where parameter h was taken to be 2 and 4. Note that this variation almost did not affect the results: for example, the increase of h by three times (from 2 to 6) resulted in variation of the time difference by less than 3% under fixed values of the other parameters, which also agrees with the assertion proved in Theorem 1.

4.2. Experiment 2

In the second experiment, three sets consisting of 20 equations of order $n = 2, 4, 6$ were used. Each equa-

Table 1. Results of experiment 1

n	l	$d = 5$	$d = 10$	$d = 15$
3	2	0.546 – 0.141 = 0.405 (0.081)	1.438 – 0.125 = 1.313 (0.131)	2.796 – 0.203 = 2.593 (0.173)
3	4	1.359 – 0.235 = 1.124 (0.225)	4.188 – 0.375 = 3.813 (0.381)	9.594 – 0.812 = 8.782 (0.585)
3	6	2.703 – 0.375 = 2.328 (0.466)	10.172 – 0.969 = 9.203 (0.920)	24.937 – 1.734 = 23.203 (1.547)
6	2	0.813 – 0.234 = 0.579 (0.116)	2.015 – 0.328 = 1.687 (0.169)	4.625 – 0.453 = 4.172 (0.278)
6	4	2.313 – 0.672 = 1.641 (0.328)	7.515 – 1.063 = 6.452 (0.645)	17.235 – 2.140 = 15.095 (1.006)
6	6	5.094 – 1.547 = 3.547 (0.709)	18.484 – 3.156 = 15.328 (1.533)	45.656 – 6.094 = 39.562 (2.637)
9	2	1.047 – 0.563 = 0.484 (0.097)	3.062 – 0.671 = 2.391 (0.239)	6.610 – 1.063 = 5.547 (0.370)
9	4	3.687 – 1.328 = 2.359 (0.472)	11.063 – 2.516 = 8.547 (0.855)	25.484 – 4.265 = 21.219 (1.415)
9	6	8.281 – 3.172 = 5.109 (1.022)	28.453 – 6.875 = 21.578 (2.158)	69.672 – 13.328 = 56.344 (3.756)

Table 2. Results of experiment 2

n	l	d	$\deg U(x)$ $\mathbf{A}'_U$	$\deg \operatorname{den} R(x)$ $\mathbf{A}_B$	Time $\langle \mathbf{A}'_U \rangle$	Time $\langle \mathbf{A}_B \rangle$
2	4–14	0–16	1–26	1–6	2.813	14.765
4	11–32	9–18	19–47	4–9	13.563	99.828
6	17–47	14–18	23–53	5–12	45.937	340.093

tion has fundamental system of solutions consisting of rational functions. The numerators of these rational functions are random polynomials of degree from one through three with integer coefficients varying from –9 to 9, and the denominators are polynomials of degree from one through three with random integer roots in the range from –9 to 9. This choice of equations ensures sharp calculation of denominator bounds (cannot be improved) obtained by algorithm $\mathbf{A}_B$ [11, Theorem 1] and places algorithm $\langle \mathbf{A}_B \rangle$ in a more advantageous position compared to $\langle \mathbf{A}'_U \rangle$.

The equations in each set were solved by means of $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$. Results of this experiment are presented in Table 2.

The first column shows the order n of the equations in the set. The other columns of the table specify ranges of parameters l and d for the equations in the corresponding set, ranges of the degrees of the denominators found on step RS1 by algorithms $\mathbf{A}'_U$ and $\mathbf{A}_B$ applied to the equations from the corresponding set, and the total times of solution of all equations of the corresponding set by algorithms $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$.

In this experiment, for all sets, the time of solution by $\langle \mathbf{A}'_U \rangle$ turns out less than that by $\langle \mathbf{A}_B \rangle$ in spite of the fact that the set of data was advantageous for $\langle \mathbf{A}_B \rangle$.

4.3. Experiment 3

In the third experiment, eight sets consisting of 20 randomly generated equations of orders $n = 3$ and 9 with parameters $l = 3$ and 9 and $d = 3$ and 9, respectively, were used. For each equation in a set, coefficients of the equation were polynomials, which were generated randomly. For the leading and trailing coefficients, a polynomial $p(x)$ of degree l with random integer roots in the range from –9 to 9 was constructed; after this, the leading and trailing coefficients of the equation were taken to be $p(x)$ and $p(x + r)$, respectively, where the value of r was determined by parameter d. The other coefficients were constructed as polynomials of degree from 1 through l with random roots in the range from –9 to 9. The equations constructed in this way have no nontrivial rational solutions but do have nontrivial universal denominators and denominator bounds calculated by $\mathbf{A}'_U$ and $\mathbf{A}_B$, respectively.

The equations in each set were solved by means of $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$. Results of this experiment are presented in Table 3.

Each row of the table corresponds to the set given by the parameters n, l, and d specified in the first three columns. The other columns of the table specify ranges of the degrees of the denominators found on step RS1 by algorithms $\mathbf{A}'_U$ and $\mathbf{A}_B$ applied to the equations from the corresponding set and the total times of solution of all equations of the corresponding set by algorithms $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$.

Table 3. Results of experiment 3

n	l	d	$\deg U(x)$ $\mathbf{A}'_U$	$\deg \operatorname{den} R(x)$ $\mathbf{A}_B$	Time $\langle \mathbf{A}'_U \rangle$	Time $\langle \mathbf{A}_B \rangle$
3	3	3	4–6	2–6	0.937	3.313
3	3	9	10–21	9–21	0.985	15.500
3	9	3	4–8	2–8	1.531	7.703
3	9	9	13–37	8–33	1.875	56.922
9	3	3	4–9	2–9	2.250	6.937
9	3	9	10–24	9–24	3.406	38.531
9	9	3	4–9	2–9	4.078	19.297
9	9	9	15–29	12–28	6.922	146.110

Like in the first two experiments, the time spent by $\langle \mathbf{A}'_U \rangle$ turned out less than that by $\langle \mathbf{A}_B \rangle$ for all sets of equations.

5. SYSTEMS OF EQUATIONS

Let system (4) be given, where $A(x) = (a_{ij}(x)) \in \mathrm{Mat}_n(k(x))$ and $A^{-1}(x) = (\tilde{a}_{ij}(x)) \in \mathrm{Mat}_n(k(x))$. Let us define

$$\operatorname{den} A(x) = \operatorname{lcm}_{i=1}^{n} \operatorname{lcm}_{j=1}^{n} \operatorname{den} a_{ij}(x),$$

$$\operatorname{den} A^{-1}(x) = \operatorname{lcm}_{i=1}^{n} \operatorname{lcm}_{j=1}^{n} \operatorname{den} \tilde{a}_{ij}(x)$$

and set

$$V(x) = u_1(x-1), \quad W(x) = u_0(x),$$

where $u_1(x) = \operatorname{den} A(x)$ and $u_0(x) = \operatorname{den} A^{-1}(x)$.

The set M is defined like in case (6).

Algorithms $\mathbf{A}_U$ and $\mathbf{A}'_U$ are executed in the same way as in the scalar case, i.e., in accordance with formula (12).

Below is the extension of the algorithm from [11] to the case of an arbitrary field k of characteristic 0 suggested in [10].

Following [11], we define $A_N(x) = A(x-1)A(x-2)\ldots A(x-N)$ for all $N \in \mathbb{N}$. Matrix A is invertible, and we can also define $A_{-N} = A^{-1}(x)A^{-1}(x+1)\ldots A^{-1}(x+N-1)$. Then, for each rational solution (5) of system (4), equalities $Y(x) = A_N(x)Y(x-N)$ and $Y(x) = A_{-N}(x)Y(x+N)$ hold.

Let $p(x) \in \mathrm{Irr}(k[x])$ and N be a positive integer, $1 \le i \le n$. Let us define $B(p(x), N, i)$ as the minimum value of function $\mathrm{val}_{p(x)}$ for all elements of the ith row of matrix $A_N(x)$. Let set $B(p(x), -N, i)$ be defined like in (8), $t = 1, 2, \ldots, s$, and let $d = \max\{d_1, d_2, \ldots, d_t\}$.

Algorithm $\mathbf{A}_B$ successively constructs matrices $A_N(x)$ for $N = 1, 2, \ldots, d+1$ and, for each t such that $1 \le t \le s$ and $d_t \ge N-1$, finds $B(q_t(x+d_t-N+1)), i = 1, 2, \ldots, n$. This yields the left lower bounds for $\mathrm{val}_{q_t(x+N-1)} Y_i(x)$, $i = 1, 2, \ldots, n$. In a similar way, the algorithm constructs matrices $A_{-N}(x)$ for $N = 1, 2, \ldots, d+1$ and, for each t such that $1 \le t \le s$ and $d_t \ge N-1$, finds $B(q_t(x+N-1), -N, i)$, $i = 1, 2, \ldots, n$, which yields the right lower bounds for $\mathrm{val}_{q_t(x+N-1)} Y_i(x)$, $i = 1, 2, \ldots, n$. Out of the two lower bounds for $\mathrm{val}_{q_t(x+j)} Y_i(x)$, $i = 1, 2, \ldots, n$, $t = 1, 2, \ldots, s$, $j = 0, 1, \ldots, d_t$, the maximum one is taken and denoted as $\alpha_{i,j,t}$. Rational functions

$$R_i(x) = \prod_{\substack{1 \le t \le s \\ 0 \le j \le d_t}} q_t^{\alpha_{i,j,t}}(x+j), \quad i = 1, 2, \ldots, n,$$

yield a denominator bound for the original system.

Algorithms $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$ use rational functions

$$S_1(x) = S_2(x) = \ldots = S_n(x) = \frac{1}{U(x)}$$

and, accordingly,

$$S_i(x) = R_i(x), \quad i = 1, 2, \ldots, n,$$

for transforming the original system to a system such that

$$(f_1(x), f_2(x), \ldots, f_n(x))^T,$$

$$f_i(x) \in k[x], \quad i = 1, 2, \ldots, n,$$

are solutions of the transformed system if and only if

$$(f_1(x)S_1(x), f_2(x)S_2(x), \ldots, f_n(x)S_n(x))^T$$

is a solution of the original system (analogue of step RS2), after which it requires to find polynomial solutions (analogue of step RS3) for the transformed system.

From general considerations, it is clear that, in the case of a system, algorithm $\langle \mathbf{A}_B \rangle$ requires construction of matrices A_N and A_{-N}, which is associated with even greater expenditures than those required for construction of Eqs. (14) and (15). This fact was confirmed experimentally.

For such an experiment, algorithms $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$ were extended to the case of systems of equations and implemented in Maple. Like in the scalar case, the implementation of algorithm $\langle \mathbf{A}'_U \rangle$ on step RS1 uses the implementation of algorithm $\mathbf{A}'_U$ described in [7], and the implementation of algorithm $\langle \mathbf{A}_B \rangle$ uses an implementation of algorithm $\mathbf{A}_B$, which was written specially for this experiment in accordance with the description given earlier in this section. On step RS3, the extensions of both algorithms $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$ to the case of systems use procedure **PolynomialSolution from the standard package LinearFunctionalSystems** available in Maple, which is designed for finding polynomial solutions of systems of ordinary equations (including difference ones).

For the experiment, we used three sets consisting of 20 randomly generated systems of order $n = 2$, 3, and 4. To generate each system in a set, first, a scalar equation of order n was randomly generated by a method used in the experiment from Section 4.2. Then, based on this equation, the corresponding accompanying system was constructed. This system was transformed by applying a change of variables specified by randomly generated nonsingular matrices. Zero entries of these matrices were generated with probability 1/2, and nonzero entries were rational functions with random numerators and denominators raised to powers from 1 through 3. Note that, like in the experiment from Section 4.2, such system construction ensures sharpness of bounds of the denominator found by algorithm $\mathbf{A}_B$.

Systems in each set were solved by $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$. Results of this experiment are presented in Table 4.

Each row of the table corresponds to the set determined by parameter n specified in the first column. The other columns of the table specify ranges of the degrees of the denominators found on step RS1 by algorithms $\mathbf{A}'_U$ and $\mathbf{A}_B$ applied to the equations from the corresponding set and the total times of solution of all equations of the corresponding set by algorithms $\langle \mathbf{A}'_U \rangle$ and $\langle \mathbf{A}_B \rangle$.

Like in the experiments for the scalar case discussed in Section 4, the time spent by $\langle \mathbf{A}'_U \rangle$ turned out less than that by $\langle \mathbf{A}_B \rangle$ for all sets of equations, in spite of sharpness (impossibility to improve) of results of $\mathbf{A}_B$ on systems in the sets. Note that results of $\mathbf{A}'_U$ on systems from these sets differ considerably from the sharp results, which can be seen from the ranges of degrees of the universal denominators found by this algorithm shown in Table 4.

Table 4. Results of the experiment with systems

n	$\deg U(x)$ $\mathbf{A}'_U$	$\deg \operatorname{den} R_i(x)$ $\mathbf{A}_B$	Time $\langle \mathbf{A}'_U \rangle$	Time $\langle \mathbf{A}_B \rangle$
2	7–39	2–8	7.216	22.922
3	18–49	3–21	38.859	169.906
4	36–74	5–28	176.829	836.172

ACKNOWLEDGMENTS

This work was supported in part by the Russian Foundation for Basic Research, project no. 10-01-00249-a, and by ECONET, project no. 21315ZF.

REFERENCES

1. Abramov, S.A., Computer Algebra Problems Associated with Searching Polynomial Solutions of Linear Differential and Difference Equations, Vestn. Mosk. Univ., Ser. 15 Vychisl. Mat. Kibernetika, 1989, no. 3, pp. 53–60 [*Moscow Univ. Comput. Math. Cybernet. (Engl. Transl.)*, 1989, no. 3, pp. 63–80].
2. Abramov, S.A., Rational Solutions of Linear Differential and Difference Equations with Polynomial Coefficients, Zh. Vychisl. Mat. Mat. Fiz., 1989, vol. 29, no. 11, pp. 1611–1620 [*Comput. Math. Math. Phys. (Engl. Transl.),* 1989, vol. 29, pp. 7–12].
3. Abramov, S.A., Rational Solutions of Linear Differential and Difference Equations with Polynomial Coefficients, Programmirovanie, 1995, no. 6, pp. 3–11 [*Program. Comp. Soft. (Engl. Transl.),* 1995, vol. 21, pp. 273–278].
4. Abramov, S.A., *Lectures on Algorithm Complexity,* Moscow: MTsNMO, 2009.
5. Abramov, S.A. and Barkatou, M., Rational Solutions of First Order Linear Difference Systems, *Proc. of ISSAC'98*, 1998, pp. 124–131.
6. Abramov, S.A., Bronstein, M., and Petkovšek, M., On Polynomial Solutions of Linear Operator Equations, *Proc. of ISSAC'95*, 1995, pp. 290–295.
7. Abramov, S.A., Gheffar, A., and Khmelnov, D.E., Factorization of Polynomials and GCD Computations for Finding Universal Denominators, *Proc. of CASC'2010*, 2010, pp. 4–18.
8. Barkatou, M., Rational Solutions of Matrix Difference Equations: Problem of Equivalence and Factorization, *Proc. of ISSAC'99,* 1999, pp. 277–282.
9. Gheffar, A., Linear Differential, Difference and q-Difference Homogeneous Equations Having no Rational Solutions, *ACM Commun. Comput. Algebra*, 2010, vol. 44, no. 3, pp. 78–83.
10. Gheffar, A. and Abramov, S., Valuations of Rational Solutions of Linear Difference Equations at Irreducible Polynomials, *Adv. Appl. Math.*, 2011 (in press).
11. Van Hoeij, M., Rational Solutions of Linear Difference Equations, *Proc. of ISSAC'98*, 1998, pp. 120–123.
12. Knuth, D.E., Big Omicron and Big Omega and Big Theta, *ACM SIGACT News*, 1976, vol. 8, no. 2, pp. 18–23.
13. Petkovšek, M., Wilf, H.S., and Zeilberger, D., $A = B$, Peters, 1996.
14. Maple online help: http://www.maplesoft.com/support/help/