Proceedings of the Steklov Institute of Mathematics, Vol. 218, 1997, pp. 15–22.
Translated from Trudy Matematicheskogo Instituta imeni V. A. Steklova, Vol. 218, 1997, pp. 20–27.

From the Karatsuba Method for Fast Multiplication of Numbers to Fast Algorithms for Discrete Functions

V. B. Alekseev

Received February, 1997

In 1961, the Karatsuba result was published [1] which states that the multiplication of two n-bit binary numbers can be carried out in $O(n^{\log_2 3})$ bit operations. Thus the new direction in constructing fast algorithms was opened. In this work we show how the development of the employed idea leads to methods of constructing fast algorithms for the recognition of properties of discrete functions.

The main Karatsuba idea lies in the recursive reduction of the problem of multiplication of n-bit numbers to the same problems for $(n/2)$-bit numbers (for even n). Let A and B be two n-bit binary numbers. Partitioning their representations into two parts of length $n/2$, we obtain

$$A \times B = (A_1 2^{n/2} + A_0)(B_1 2^{n/2} + B_0) = A_1 B_1 2^n + (A_1 B_0 + A_0 B_1) 2^{n/2} + A_0 B_0.$$

Since the multiplication by powers of two and addition are fast operations, the problem reduces to the fast calculation of three bilinear forms $C_1 = A_1 B_1$, $C_2 = A_1 B_0 + A_0 B_1$, and $C_3 = A_0 B_0$. Karatsuba noted that it is sufficient for this purpose to calculate the bilinear forms C_1, C_3, and $D = (A_1 + A_0)(B_1 + B_0)$. In this case $C_2 = D - C_1 - C_3$. Thus the problem of the multiplication of n-bit binary numbers reduces to three problems of the multiplication of $(n/2)$-bit binary numbers (a superfluous bit remains to be taken away from D) and several additions (or subtractions) of no more than n-bit numbers. For the multiplication of n-bit binary numbers of maximal complexity, we have the recursive inequality, for even n,

$$L(n) \le 3L\Big(\frac{n}{2}\Big) + O(n),$$

from which it follows that $L(n) = O(n^{\log_2 3})$.

This idea of the recursive reduction of a problem to the same problems having sizes, which are smaller by a constant factor, turns out to be very fruitful for constructing fast algorithms. In [2] this idea is called the "divide and conquer" method. The complexity estimate of algorithms of this kind is based on the recursive inequality

$$L(n) \le aL\Big(\frac{n}{b}\Big) + cn^{\alpha}. \tag{1}$$

If L is a monotonically nondecreasing function and inequality (1) holds for all natural n that are divisible by b, then from (1) it follows that [2]

$$L(n) = \begin{cases} O(n^{\log_a b}) & \text{for } \alpha < \log_a b \\ O(n^{\alpha}) & \text{for } \alpha > \log_a b \\ O(n^{\alpha}\log n) & \text{for } \alpha = \log_a b. \end{cases} \tag{2}$$

Thus if a problem with parameter n (for n that is divisible by b) can be reduced to the same a problems with parameter n/b, then estimates (1) and (2) are valid for the complexity of its solving. If, in this case, the complexity of the reduction itself is characterized by the small quantity α in (1), then, by (2), the complexity estimate is defined by the parameters a and b.

If the representations of the numbers A and B are partitioned into k parts of length n/k (for n that is divisible by k) $A_{k-1}, A_{k-2}, \dots, A_0$ and $B_{k-1}, B_{k-2}, \dots, B_0$, then the problem of the multiplication of A by B reduces to the calculation of $2k-1$ bilinear forms $C_1 = A_{k-1}B_{k-1}$, $C_2 = A_{k-1}B_{k-2} + A_{k-2}B_{k-1}$, $\dots$, $C_{2k-1} = A_0 B_0$. By analogy with the Karatsuba construction, it is sufficient to find d bilinear forms of the type

$$D_r = \Bigl(\sum_i \alpha_i^{(r)} A_i\Bigr)\Bigl(\sum_j \beta_j^{(r)} B_j\Bigr)$$

so that all C_s may be obtained as linear combinations $C_s = \sum_s \gamma_s^{(r)} D_r$ of the forms D_r $(r = 1, \dots, d)$. In this case, from (1) and (2) we have the estimate for the complexity of multiplication of n-bit numbers

$$L(n) = O\bigl(n^{\log_k d}\bigr).$$

Thus the problem reduces to suitable choosing the coefficients $\alpha_i^{(r)}$, $\beta_j^{(r)}$, and $\gamma_s^{(r)}$ for a sufficiently small d. It turns out that this problem is not simple and, even for $k = 3$, we cannot choose coefficients for $d = 5$ (since $\log_3 5 < \log_2 3 < \log_3 6$) without invoking additional reasonings. In [3], A. L. Toom actually chose desirable coefficients for any k and $d = 2k-1$ using "the model widening" technique, namely, considering a wider problem of multiplication of polynomials. For the complexity of the multiplication of n-bit numbers, he obtained, as a result, the estimate

$$L(n) = O\bigl(n^{\log_k (2k-1)}\bigr)$$

for any k, i.e.,

$$L(n) = O\bigl(n^{1+\varepsilon}\bigr)$$

for any $\varepsilon > 0$. (In fact, Toom's result has a stronger form.) The best known upper estimate for the complexity of the multiplication of n-bit numbers is presented in [4].

In 1969 Strassen published his result [5] which stated that the multiplication of two quadratic matrices of order n can be performed in $O(n^{\log_2 7})$ arithmetic operations. Here the same Karatsuba idea was used. Every n by n matrix is partitioned into four $(n/2)$ by $(n/2)$ blocks. Since operations on block matrices of this kind can be carried out in the same manner as on matrices of order 2, a problem reduces to the calculation of four bilinear forms:

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}, \qquad C_{12} = A_{11}B_{12} + A_{12}B_{22},$$
$$C_{21} = A_{21}B_{11} + A_{22}B_{21}, \qquad C_{22} = A_{21}B_{12} + A_{22}B_{22}.$$

In this case, it is sufficient to find d bilinear forms of the special type

$$D_r = \Bigl(\sum \alpha_{ij}^{(r)} A_{ij}\Bigr)\Bigl(\sum \beta_{kl}^{(r)} B_{kl}\Bigr)$$

so that all C_{pq} may be obtained as linear combinations of the forms D_r. Strassen succeeded in choosing the coefficients $\alpha_{ij}^{(r)}$, $\beta_{kl}^{(r)}$, and $\gamma_{pq}^{(r)}$ for $d = 7$. Hence using (1) and (2) we get

$$L(n) = O\bigl(n^{\log_2 7}\bigr).$$

Partitioning the matrices into (n/k) by (n/k) blocks for other k, we can try to choose the coefficients so that $\log_k d$ should be as few as possible. Unfortunately, nobody succeeded in doing this for $d = 21$ ($\log_3 21 < \log_2 7 < \log_3 22$) even for $k = 3$. Probably, in this case, a wider model should be used, however, nobody did it in the past almost 30 years. Only a number of results were obtained that make it possible to construct fast algorithms for the multiplication of quadratic matrices by choosing coefficients for problems from a wider class (the multiplication of rectangular matrices, the approximate multiplication of matrices, the multiplication of matrices having zero elements, and others). These techniques make it possible to obtain the algorithm of the multiplication of n by n matrices with the number of arithmetic operations equal to $L(n) = O(n^{2.38})$. However, eventually, each of these techniques comes up against choosing coefficients [6].

It turned out that, in the theory of discrete functions, there also arose problems similar to the problem of multiplication of numbers or matrices. Namely, in some cases, the problem of constructing fast algorithm reduces simply to the problem of choosing suitable coefficients.

For this purpose we use the idea of going from the problem of recognition of properties of discrete objects to an algebraic computational problem. A technique of this kind was applied, for example, in [7] where the problem of constructing a transitive closure of a graph was reduced to the problem of multiplication of matrices over the ring of integers, and the algorithm with complexity $O(n^{\log_2 7}\log^3 n)$ bit operations was obtained by using the Strassen algorithm.

We suggest the general idea of using semirings with the object of searching for fast algorithms.

Recall that a commutative semiring is an arbitrary set with two binary operations of multiplication and addition that possess commutativity, associativity, and distributivity of multiplication with respect to addition.

We shall denote by S_2 the semiring with two elements 0 and 1 and with disjunction and conjunction as addition and multiplication.

Definition. A semiring A is said to be *logical* if there exists a homomorphism of this semiring onto the semiring S_2. We shall define a logical semiring as a pair (A_0, A_1), where A_0 is the preimage of 0 and A_1 is the preimage of 1 under the homomorphism of A onto S_2.

The examples of logical semirings:

(1) $(0, \mathbb{N})$, where $\mathbb{N}$ is the set of natural numbers;

(2) the set of polynomials in one variable with integer coefficients and a nonnegative free term, where A_0 is the set of polynomials whose free term is 0 and A_1 is the set of polynomials whose free term is positive;

(3) A_0 and A_1 are any subsets of a linearly ordered set, with any element from A_0 being less than any element from A_1 and max and min being considered as addition and multiplication.

The general idea of using logical semirings for the purpose of searching for fast algorithms is as follows. Many problems of the recognition of properties of discrete objects reduce to the verification of the truth of a sentence in which the existential quantifier and generality quantifier are taken over finite sets. Replacing the quantifiers by disjunction and conjunction, we can reduce the problem to the computation of the truth value of a certain logical expression T which is made up by predicates depending on an input with the aid of disjunction and conjunction. If the predicates can be calculated fast, then the problem comes up against the complexity of computing the logical expression T itself with respect to disjunction and conjunction. Then, instead of the calculation of T over S_2, we may compute its homomorphic preimage T' in a certain logical semiring A, and, finally, return homomorphically onto S_2. In this case the computations can be carried out even not in A, but in any its extension, which simplifies searching for the fast algorithm.

Let us present a simple example of applying this idea. Suppose that we have to find out whether there is a cycle of length 3 in a given graph with n vertices defined by an adjacency matrix $A = (a_{ij})_{n\times n}$. The exhaustive search algorithm for solving this problem needs the number of operations which is no less in order than n^3. At the same time the following statement is valid.

Statement. *For solving the problem given above there exists an algorithm with* $O(n^{\log_2 7}\log^2 n)$ *bit operations.*

Proof. The property of the graph, which we need, can be written as the logical formula

$$\exists i\ \exists j\ \exists k \quad (a_{ij}=1)\&(a_{jk}=1)\&(a_{ki}=1). \tag{3}$$

In order to pass to algebra we use the logical semiring from example (1). In this case, 0 must be assigned to the false values of the predicates and any natural numbers must be assigned to the truth values of the predicates. To do this, it is sufficient to assign the number a_{ij} to the sentence $a_{ij}=1$. Thereafter in order to obtain the logical value of formula (3), it is sufficient to compute the value of the formula

$$\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{n} a_{ij}a_{jk}a_{ki}$$

on the set of natural numbers. It is easy to see that this expression is the trace of the matrix $B = A\times A\times A$. If we pass to the computations in the ring of integers, then we can apply the Strassen algorithm for computing this trace, which needs $O(n^{\log_2 7})$ arithmetic operations. It is easy to show that, in this case, integers with $O(\log n)$ bits will be involved in the algorithm. Using bit operations for computing the sum and the product of numbers of this kind, we obtain the algorithm with $O(n^{\log_2 7}\log^2 n)$ bit operations.

Remark. Applying faster algorithms for multiplying matrices, we can obtain an algorithm with bit complexity $O(n^{2.38})$ for the problem considered.

Below we shall consider the problem for which we succeeded in using the described idea of passing from the recognition of a property to an algebraic computational problem in combination with the "divide and conquer" method.

Let $E_k = \{0,1,\dots,k-1\}$. Let E_k^n be the set of all strings of length n with elements from E_k. We denote by P_k the set of all functions $f(x_1,\dots,x_n)\colon E_k^n\to E_k$ for any n. By P_k^* we denote the set of all functions $f(x_1,\dots,x_n)\colon E_k^n\to E_k\cup\{*\}$. Functions from P_k are called k-valued functions and functions from P_k^* are called partial k-valued functions ($*$ is regarded as an indetermination). In order to cover both cases we shall consider the functions $f\colon E_k^n\to D$, where D is an arbitrary finite set. Many important classes of discrete functions can be described by means of relations on a domain of definition and a range of values. Let $R(y_1,\dots,y_m)$ be a relation on E_k and let $R_1(y_1,\dots,y_m)$ be a relation on D. Let us define the relation $R^n(\widetilde{y}_1,\dots,\widetilde{y}_m)$, where $\widetilde{y}_1,\dots,\widetilde{y}_m$ are strings of length n with elements from E_k ($\widetilde{y}_i = (y_i^1, y_i^2,\dots,y_i^n)$), as follows:

$$R^n(\widetilde{y}_1,\dots,\widetilde{y}_m) \implies \forall j \quad R(y_1^j,\dots,y_m^j).$$

Let $U(R,R_1)$ be the class of all functions $f(x_1,\dots,x_n)$ for all n that map E_k^n into D and satisfy the condition

$$R^n(\widetilde{\alpha}_1,\dots,\widetilde{\alpha}_m) \implies R_1\big(f(\widetilde{\alpha}_1),\dots,f(\widetilde{\alpha}_m)\big) \tag{4}$$

for any strings $\widetilde{\alpha}_1,\dots,\widetilde{\alpha}_m$ of length n. The classes $U(R,R_1)$ play very important part in studies of the functional systems P_k and P_k^* with superposition.

Since the domain of definition of the functions $f(x_1,\dots,x_n)$, which we consider, is finite, it follows that, on ordering it by a certain standard way, we may define the function f simply by the vector of its values $\widetilde{\gamma}=(\gamma_1,\gamma_2,\dots,\gamma_{k^n})$ on all strings from the domain of definition.

Suppose that k, the domain D, the relation R on E_k, and R_1 on D are fixed. Suppose also that the vector of the function $f(x_1,\dots,x_n)$, which maps E_k^n into D, arrives at the input of an algorithm. We have to find out whether f belongs to the class $U(R,R_1)$. We show how this problem of recognition of a property may be reduced to an algebraic computational problem and how then the "divide and conquer" method may be applied. The quantity $N=k^n$ is treated as a length of an input. The method suggested consists of the following steps. The sentence "$f(x_1,\dots,x_n)\notin U(R,R_1)$" may be written as the logical formula

$$\bigvee_{(b_1,\dots,b_m)\notin R_1}\bigvee_{\alpha_1}\cdots\bigvee_{\alpha_m} R(\alpha_1^1,\dots,\alpha_1^m)\cdots R(\alpha_n^1,\dots,\alpha_n^m)\times\big(f(\alpha_1)=b_1\big)\cdots\big(f(\alpha_m)=b_m\big),$$

where $\alpha_j=(\alpha_1^j,\dots,\alpha_n^j)$ and α_i^j runs over all values $0,1,\dots,k-1$. The complexity of computing the logical value of this formula in order can be determined by the complexity of computing the logical value of the form

$$\varphi_n=\bigvee_{\alpha_1}\cdots\bigvee_{\alpha_m} R(\alpha_1^1,\dots,\alpha_1^m)\cdots R(\alpha_n^1,\dots,\alpha_n^m)\times w_{\alpha_1}^1\cdots w_{\alpha_m}^m$$

at given values of the logical variables w_α^j.

In accordance with the general idea of using logical semirings, it is sufficient to consider a certain logical semiring (A_0,A_1) and replace the relation R by a function $t(y_1,\dots,y_m)$ so that

$$t(\alpha_1,\dots,\alpha_m)\in\begin{cases}A_0 & \text{if } R(\alpha_1,\dots,\alpha_m) \text{ is false}\\ A_1 & \text{if } R(\alpha_1,\dots,\alpha_m) \text{ is true,}\end{cases}$$

then replace similarly the logical values of w_α^j by any corresponding values of z_α^j from the semiring (A_0,A_1) and compute the polylinear form

$$\psi_n=\sum_{\alpha_1}\cdots\sum_{\alpha_m} t(\alpha_1^1,\dots,\alpha_1^m)\cdots t(\alpha_n^1,\dots,\alpha_n^m)\times z_{\alpha_1}^1\cdots z_{\alpha_m}^m.$$

over the semiring (A_0,A_1). In doing this the calculations may be performed in any semiring A that is an extension of the semiring (A_0,A_1). The form ψ_n can be written as

$$\psi_n=\sum_{(\alpha_1^1,\dots,\alpha_1^m)} t(\alpha_1^1,\dots,\alpha_1^m)\cdots\sum_{(\alpha_n^1,\dots,\alpha_n^m)} t(\alpha_n^1,\dots,\alpha_n^m)\times z_{\alpha_1}^1\cdots z_{\alpha_m}^m.$$

Hence it can be seen that the recursion may be used here. Indeed, we can show [8] that the computation of ψ_n reduces to d problems of the computation of ψ_{n-1} if the form

$$\sum_{(\alpha_1,\dots,\alpha_m)} t(\alpha_1,\dots,\alpha_m)z_{\alpha_1}^1\cdots z_{\alpha_m}^m$$

can be represented as

$$\sum_{r=1}^{d} L_1^r\cdots L_m^r, \tag{5}$$

where L_i^r is a linear form of variables $z_0^i, \dots, z_{k-1}^i$ with coefficients from A. Thus the problem of searching for a fast algorithm reduces to suitable choosing coefficients of these linear forms. In this case we pass from $N = k^n$ to $k^{n-1} = N/k$, i.e., the size of the input of a problem decreases by a factor of k. Consequently, in order to estimate the complexity of the algorithm obtained we can use (1) and (2).

Let us demonstrate the given method with an example of two families of classes of discrete functions, which play an important part in the study of k-valued and partial k-valued logics.

Theorem 1. *Let $R(y_1, \dots, y_m)$ be a relation on E_k that holds exactly on strings $(\alpha_1, \dots, \alpha_m)$ such that $\alpha_1 + \dots + \alpha_m = 0 \pmod k$ and let $R_1(y_1, \dots, y_m)$ be an arbitrary relation on D. Then for the recognition whether the function $f(x_1, \dots, x_n)$ defined by a vector of values belongs to the class $U(R, R_1)$, there exists an algorithm with the bit complexity $O(N \log^3 N)$, where $N = k^n$ is the length of the vector of values of the function.*

Remark. The exhaustive search algorithm based on the complete verification of implication (4) for all strings has the complexity which is no less than $O(N^m)$.

Proof. Let λ be a complex kth root of 1 of the multiplicative order k. Let us consider the ring of numbers of the form $p + \lambda q$, where p and q are integers. Consider the equality

$$\begin{aligned}
\sum_{r=0}^{k-1} \prod_{j=1}^{m} \sum_{i=0}^{k-1} \lambda^{ri} z_i^{(j)} &= \sum_{r=0}^{k-1} \sum_{i_1=0}^{k-1} \cdots \sum_{i_m=0}^{k-1} \lambda^{ri_1} z_{i_1}^{(1)} \lambda^{ri_2} z_{i_2}^{(2)} \cdots \lambda^{ri_m} z_{i_m}^{(m)} \\
&= \sum_{r=0}^{k-1} \sum_{i_1=0}^{k-1} \cdots \sum_{i_m=0}^{k-1} \lambda^{r(i_1+i_2+\dots+i_m)} z_{i_1}^{(1)} z_{i_2}^{(2)} \cdots z_{i_m}^{(m)} \\
&= \sum_{i_1=0}^{k-1} \cdots \sum_{i_m=0}^{k-1} \left(\sum_{r=0}^{k-1} \lambda^{r(i_1+i_2+\dots+i_m)} \right) z_{i_1}^{(1)} z_{i_2}^{(2)} \cdots z_{i_m}^{(m)} \\
&= \sum_{i_1=0}^{k-1} \cdots \sum_{i_m=0}^{k-1} t_{i_1 i_2 \dots i_m} z_{i_1}^{(1)} z_{i_2}^{(2)} \cdots z_{i_m}^{(m)}.
\end{aligned}$$

Here

$$t_{i_1 i_2 \dots i_m} = \begin{cases} 0 & \text{if } i_1 + i_2 + \dots + i_m \neq 0 \pmod k \\ k & \text{if } i_1 + i_2 + \dots + i_m = 0 \pmod k), \end{cases}$$

i.e., t corresponds to the relation R over the logical semiring $(0, N)$. Thus we have the expression of form (5) with $d = k$ for R. Then, by Theorem 1 from [8], we obtain that, for the problem considered, there is an algorithm with $O(N \log N)$ arithmetic operations over elements from the ring of numbers of the form $p + \lambda q$, where p and q are integers. In this case, every operation of this kind needs no more than $O(n^2) = O(\log^2 N)$ bit operations. This completes the proof of the theorem.

Theorem 2. *Let $R(y_1, \dots, y_m)$ be a relation on E_k that holds exactly on those strings $(\alpha_1, \dots, \alpha_m)$ that have at least two equal elements and let $R_1(y_1, \dots, y_m)$ be an arbitrary relation on D. Then for the recognition of belonging the function $f(x_1, \dots, x_n)$ defined by a vector of values to the class $U(R, R_1)$, there exists an algorithm with the bit complexity $O(N^{\log_k (k+1)} \log^3 N)$, where $N = k^n$ is the length of values of the function.*

Proof. Let us consider the expression

$$\sum_{r=0}^{k-1}\prod_{j=1}^{m}\big(z_0^{(j)}+z_1^{(j)}+\cdots+z_{k-1}^{(j)}+\varepsilon z_r^{(j)}\big).$$

We expand it in a power series of ε. The free term is

$$k\prod_{j=1}^{m}\big(z_0^{(j)}+z_1^{(j)}+\cdots+z_{k-1}^{(j)}\big).$$

The coefficient of ε is equal to

$$\sum_{r=0}^{k-1}\sum_{t=1}^{m} z_r^{(t)}\prod_{\substack{j=1\\ j\neq t}}^{m}\big(z_0^{(j)}+z_1^{(j)}+\cdots+z_{k-1}^{(j)}\big)=\sum_{t=1}^{m}\sum_{r=0}^{k-1} z_r^{(t)}\prod_{\substack{j=1\\ j\neq t}}^{m}\big(z_0^{(j)}+z_1^{(j)}+\cdots+z_{k-1}^{(j)}\big)$$

$$=m\prod_{j=1}^{m}\big(z_0^{(j)}+z_1^{(j)}+\cdots+z_{k-1}^{(j)}\big).$$

The coefficient of ε^2 is equal to

$$\sum_{r=0}^{k-1}\sum_{1\le t<l\le m} z_r^{(t)}z_r^{(l)}\prod_{\substack{j=1\\ j\neq t\\ j\neq l}}^{m}\big(z_0^{(j)}+z_1^{(j)}+\cdots+z_{k-1}^{(j)}\big).$$

On removing the brackets, the resulting sum with positive integer coefficients has exactly all those summands $z_{i_1}^{(1)}z_{i_2}^{(2)}\cdots z_{i_m}^{(m)}$ in which there are at least two equal indices. Hence we obtain that the expression

$$\frac{1}{\varepsilon^2}\sum_{r=0}^{k-1}\prod_{j=1}^{m}\big(z_0^{(j)}+z_1^{(j)}+\cdots+z_{k-1}^{(j)}+\varepsilon z_r^{(j)}\big)-\frac{k+\varepsilon m}{\varepsilon^2}\prod_{j=1}^{m}\big(z_0^{(j)}+z_1^{(j)}+\cdots+z_{k-1}^{(j)}\big)$$

can be transformed to

$$\sum_{i_1=0}^{k-1}\cdots\sum_{i_m=0}^{k-1} t_{i_1i_2\ldots i_m}z_{i_1}^{(1)}z_{i_2}^{(2)}\cdots z_{i_m}^{(m)},$$

where $t_{i_1i_2\ldots i_m}$ is a polynomial in ε having free term equal to a natural number if there are equal indices among $i_1,i_2,\ldots,i_m$ and equal to 0 if all indices are different. Thus t corresponds to the relation R over the logical semiring from example (2) above and we have, for R, the expression of form (5) with $d=k+1$. The coefficients of this expression are taken from the ring of Laurent polynomials in ε with integer coefficients. Then, according to Theorem 1 from [8], we obtain that, for the problem under consideration, there is an algorithm with $O(N^{\log_k(k+1)})$ arithmetic operations on elements from this ring and in this case every operation of this kind needs no more than $O(n^3)=O(\log^3 N)$ bit operations. We have proved the theorem.

Other examples of applying the used method can be found in [8–10]. By and large we obtain the construction, which generalizes the Karatsuba construction. However, a problem comes up against choosing coefficients in each particular case. It is of interest to find a regular technique of choosing of this kind. It is also of interest to expand a variety of logical semirings to be applied useful in the construction described. Note that it is shown in [11] that, for applying recursion, constructions more general than expansion (5) may be used.

ACKNOWLEDGMENTS

This work was supported by the Russian Foundation for Basic Research (project no. 97-01-00989).

REFERENCES

1. Karatsuba, A. A. and Ofman, Yu. P., Multiplication of Multiplace Numbers with Automata, *Dokl. Akad. Nauk SSSR*, 1962, vol. 145, no. 2, pp. 293–294.
2. Aho, A. V., Hopcroft, J. E., and Ullman, J. D., *The Design and Analysis of Computer Algorithms*, Reading (Ma): Addison-Wesley, 1974.
3. Toom, A. L., On the Complexity of a Scheme of Functional Elements that Realizes Multiplication of Integers, *Dokl. Akad. Nauk SSSR*, 1963, vol. 150, no. 3, pp. 496–498.
4. Schönhage, A. and Strassen, V., Schnelle Multiplikation grosser Zahlen, *Computing*, 1971, vol. 7, no. 3–4, pp. 281–292.
5. Strassen, V., Gaussian Elimination is not Optimal, *Numer. Math.*, 1969, vol. 13, no. 4, pp. 354–356.
6. Alekseev, V. B., Complexity of Multiplication of Matrices. Review, in: *Kibern. Sbornik*, no. 25, Moscow: Mir, 1988, pp. 189–236.
7. Furman, M. E., On Applying the Method of Fast Multiplication of Matrices in the Problem of Finding a Transitive Closure of a Graph, *Dokl. Akad. Nauk SSSR*, 1970, vol. 194, no. 3, p. 524.
8. Alekseev, V. B., Logical Semirings and Their Use for Constructing Fast Algorithms, *Vestn. MGU, Ser. 1, Mat., Mekh.*, 1997, no. 1, pp. 22–29.
9. Alekseev, V. B., Recognition of Properties in k-Valued Logic and Approximate Algorithms, in: *Lect. Notes Comput. Sci.*, Springer, 1987, vol. 278, pp. 10–13.
10. Alekseev, V. B. and Krivenko, M. M., On the Complexity of Recognition of Completenes of Function Systems in the Class P_3^*, *Vestn. MGU, Ser. 1, Mat., Mekh.*, 1997, no. 3, pp. 6–9.
11. Alekseev, V. B., Stair Bilinear Algorithms and Recognition of Completeness in k-Valued Logics, *Izv. Vuzov, Mat.*, 1988, no. 7, pp. 19–27.

Translated by A. Gordienko