Cybernetics and Systems Analysis, Vol. 48, No. 1, January, 2012

SOFTWARE–HARDWARE SYSTEMS

PARALLELIZATION OF SEQUENTIAL PROGRAMS: DISTRIBUTION OF ARRAYS AMONG PROCESSORS AND STRUCTURIZATION OF COMMUNICATIONS

E. V. Adutskevich,[a†] **N. A. Likhoded,**[a‡] **and A. O. Sikorsky**[b]

UDC 519.6+681.3.012

Abstract. *Data distribution functions are introduced. They are matched with scheduling functions. The processors and iterations are determined that use an array element at its fixed position in a statement. This makes it possible to obtain the initial data distribution and also information on the data volume for every processor and on the structure of required communications.*

Keywords: *parallel computing, parallelization of algorithms, affine loop nest, parallel computer with distributed memory, optimization of communications.*

INTRODUCTION

In adapting a sequential algorithm to an implementation on parallel computers with distributed memory, two major problems should be solved.

The first problem consists of the distribution of the operations of the algorithm among processors and establishment of a new order of their execution. The main issue arising in solving this problem consists of the preservation of the order of execution of informationally connected operations, i.e., the fulfillment of the conditions of preservation of dependencies of the algorithm. Many publications (in particular, [1–5]) are devoted to the investigation of this problem, and various methods of solving it are proposed in them.

The second problem lies in distributing data of the algorithm among processors and the establishing a data exchange scheme for executing the obtained parallel algorithm. The main issue that arises in solving this problem consists of the necessity to fix, at a definite instant of time, the location of a data item required for the execution of an operation and to supplement the computational algorithm with new operations of data transmission and reception. At the same time, one should take into account that the realization of communications on a parallel computer with distributed memory requires considerable expenditures. Since the objective of using parallel computers is the decrease in the time of solving problems, in parallelizing an algorithm one should strive to decrease communication expenditures for its implementation.

In contrast to the first problem, the second problem is less investigated. We will briefly review approaches to the solution of this problem.

An obvious approach to data distribution and minimization of communication expenditures lies in partitioning an algorithm into independently computed blocks [1, 6–9]. In this case, data are distributed according to the distribution of operations that use these data and, hence, all processors of a parallel computing system can work independently without the need for data exchange. As is obvious, this approach simplifies the problem of data distribution among processors and eliminates the problem of data exchange, but, in practice, algorithms seldom allows for a decomposition into independent fragments.

Another approach lies in obtaining block versions of an algorithm, i.e., partitioning the iteration space of loop nests in a special way [1, 10–13]. The objective of such a partitioning is the increase in the size of packages of transferred data and

[a]Institute of Mathematics, National Academy of Sciences of Belarus, Minsk, Republic of Belarus, [†]*zhenya@im.bas-net.by*; [‡]*likhoded@im.bas-net.by*. [b]LLC "Yandex Bel," Minsk, Republic of Belarus, *alex_s@tut.by*. Translated from Kibernetika i Sistemnyi Analiz, No. 1, pp. 144–163, January–February 2012. Original article submitted January 14, 2008. Updated article submitted March 19, 2010.

 1060-0396/12/4801-0122

decrease in the frequency of communications. The approach is directed towards the minimization of overheads for data exchanges irrespective of the problem of data array distribution.

There also is the approach lying in the establishment of a fixed (specified before the beginning of the execution of a program and invariable during its execution) distribution of data among processors of a computing system [4, 14–19]. The search for such a data distribution is carried under the condition of minimization of data exchanges between the processors whose local memories store data and the processors in which these data are computed or are used as arguments for computations during the execution of a parallel algorithm. Within the framework of this approach, algorithms of joint distribution of operations and data of an algorithm among processors are proposed in [4, 16, 17]. In [18, 19], conditions are considered whose fulfillment makes it possible to organize structured communications that are realized between a group of processors and can be performed more quickly than the corresponding collection of communications between pairs of processors. An advantage of this approach is a simplification of the data exchange scheme (an exchange always begins with the processor that uses a data item as a result of computations to the processor whose local memory stores this data item and vice versa, i.e., from the processor whose local memory stores a data item to another processor that uses this data item as an argument for computations). The possibility of joint determination of a distribution of operations and data that is directed towards the decrease in communications required for the realization of an algorithm is also among advantages of this approach.

However, in many computational algorithms, array elements are redefined over the entire period of their execution. In this case, this approach will lead to the organization of a large number of redundant communications between the processors that use an array element for computations and the processor that stores it rather than to directly transfer it in turn between the processors using this element. To reveal this situation and to change the data exchange scheme within the framework of the approach being considered, the corresponding algorithm should be additionally investigated as is shown in [18]. Other solutions to this problem are proposed in [20, 21].

In the present article, a new approach to the search for the distribution of data of an algorithm among processors and iterations of a parallel algorithm is considered. In this case, taking into account that the distribution of operations of the algorithm among processors and iterations of the parallel algorithm is already found, processors and iterations are determined that use an array element at its fixed position in an operator during program execution. This information makes it possible to obtain an initial distribution of arrays (including the distribution leading to the use of each array element by only one processor) and information on the size of the arrays used in each processor (which is necessary for the reservation of processor memory in writing a program), to establish the communications required for a given distribution of operations, and to determine the possibility of structurization of communications. The obtained results are suitable for automation, which makes it possible to use them for the development of software systems destined for parallelizing sequential algorithms.

BASIC DENOTATIONS

Let an algorithm be specified by an affine loop nest, i.e., the index expressions of variables and boundaries of changing the loop parameters of the algorithm are affine functions of loop parameters and external variables.

Let, in a loop nest, there be K operators S_β to be executed, and let L arrays a_l. be used. Simple variables are considered to be arrays of dimension 0; $V_\beta \subset \mathbf{Z}^{n_\beta}$ is the range of variation of the parameters of the loop nest for an operator S_β, where n_β is the number of loops enveloping the operator S_β; $W_l \subset \mathbf{Z}^{\nu_l}$ is the range of variation of indices of the lth array, where ν_l is the dimension of the lth array; $N \in \mathbf{Z}^e$ is the vector of external variables of the algorithm, where e is the number of external variables. We call a realization of the operator S_β for a concrete value of the vector of parameters of loops J an operation and denote it by $S_\beta(J)$. The execution of all operations dependent on J is called the Jth iteration.

An occurrence (l, β, q) is understood to be the qth occurrence of an array a_l in the operator S_β. In other words, the occurrence (l, β, q) is the qth access in a sequence of accesses to elements of the array a_l during the next execution of the operator S_β. For the sake of obviousness, along with (l, β, q), we will also use the denotation (a_l, S_β, q). We denote the variation range for indices of elements of an array $a_l(F)$ that are connected with the occurrence $(l, \beta\, q)$ by $W_{l,\beta,q}$. We denote by Q the set of occurrences (l, β, q) of all arrays a_l in all operators S_β (the union of occurrences (l, β, q) over all l, β, and q is taken).

For the lth array, the indices of its elements connected with an occurrence (l, β, q) are expressed by an affine function $\overline{F}_{l,\beta,q}: V_\beta \to W_l$ of the form

$$\overline{F}_{l,\beta,q}(J) = F_{l,\beta,q} J + G_{l,\beta,q} N + f^{(l,\beta,q)},$$
$$J \in V_\beta,\ N \in \mathbf{Z}^e,\ F_{l,\beta,q} \in \mathbf{Z}^{\nu_l \times n_\beta},\ G_{l,\beta,q} \in \mathbf{Z}^{\nu_l \times e},\ f^{(l,\beta,q)} \in \mathbf{Z}^{\nu_l},\ (l,\beta,q) \in Q. \quad (1)$$

Example 1. Let A be a left triangular matrix of order N with diagonal elements equal to 1. We consider the following algorithm for solving a system of linear algebraic equations $Ax = b$ by the inverse substitution method:

S_1: $x(1) = b(1)$
do $i = 2, N$
  S_2: $x(i) = b(i)$
  do $j = 1, i-1$
    S_3: $x(i) = x(i) - a(i, j)x(j)$
  enddo
enddo

The loop nest contains three operators S_1, S_2, and S_3 and elements of three arrays x, b, and x. In this case, $n_1 = 0$, $n_2 = 1$, $n_3 = 2$, $\nu_1 = 1$, $\nu_2 = 1$, $\nu_3 = 2$, $V_1 = \{(1)\}$, $V_2 = \{(i) \in \mathbf{Z} \mid 2 \le i \le N\}$, $V_3 = \{(i, j) \in \mathbf{Z}^2 \mid 2 \le i \le N, 1 \le j \le i-1\}$, $W_1 = W_2 = \{(i) \in \mathbf{Z} \mid 1 \le i \le N\}$, and $W_3 = V_3$.

Indices of array elements used by operators of the algorithm are expressed by functions $\overline{F}_{x,S_1,1}(1) = 1$, $\overline{F}_{x,S_2,1}(i) = E^{(1)}(i)$, $\overline{F}_{x,S_3,1}(i,j) = \overline{F}_{x,S_3,2}(i,j) = (1\ 0)(i\ j)^{\mathrm{T}}$, $\overline{F}_{x,S_3,3}(i,j) = (0\ 1)(i\ j)^{\mathrm{T}}$, $\overline{F}_{b,S_1,1}(1) = 1$, $\overline{F}_{b,S_2,1}(i) = E^{(1)}(i)$, and $\overline{F}_{a,S_3,1}(i,j) = E^{(2)}(i\ j)^{\mathrm{T}}$. ■

An operation $S_\beta(J), J \in V_\beta$, depends on an operation $S_\alpha(I), I \in V_\alpha$, if [22]

(1) $S_\alpha(I)$ is executed before $S_\beta(J)$;

(2) $S_\alpha(I)$ and $S_\beta(J)$ use the same array element (i.e., $\overline{F}_{l,\alpha,p}(I) = \overline{F}_{l,\beta,q}(J)$ for some l, p, and q) and at least one of them redefines (changes) the element;

(3) this element is not redefined between the operations $S_\alpha(I)$ and $S_\beta(J)$.

This dependence can be true dependence (if an array element is first defined and then its value is used as an argument), can be antidependence (if the value of an array element is first used as an argument and then is redefined), and can be dependence with respect to output (if an array element is defined and then is redefined).

We denote the dependence of the operation $S_\beta(J)$ on the operation $S_\alpha(I)$ by $S_\alpha(I) \to S_\beta(J)$. Let $P = \{(\alpha, \beta) \mid \exists\, I \in V_\alpha, J \in V_\beta, S_\alpha(I) \to S_\beta(J)\}$ be a set that specifies pairs of dependent operators. For each pair $(\alpha, \beta) \in P$, we denote $V_{\alpha,\beta} = \{J \in V_\beta \mid \exists\, S_\alpha(I) \to S_\beta(J)\}$. Note that there can be more than one dependence between the operations $S_\alpha(I)$ and $S_\beta(J)$ if the equality $\overline{F}_{l,\alpha,p}(I) = \overline{F}_{l,\beta,q}(J)$ is fulfilled for more than one collection of l, p, and q. In this case, the denotation $S_\alpha(I) \overset{l,p,q}{\to} S_\beta(J)$ can be used.

Let the dependencies between operations of an algorithm be specified by functions $\overline{\Phi}_{\alpha,\beta}: V_{\alpha,\beta} \to V_\alpha$ in such a manner that if $S_\alpha(I) \to S_\beta(J)$, $I \in V_\alpha$, $J \in V_{\alpha,\beta} \subseteq V_\beta$, then $I = \overline{\Phi}_{\alpha,\beta}(J)$. If a pair of operators S_α and S_β generates more than one dependence, then upper indices are also used for the denotation of the functions $\overline{\Phi}_{\alpha,\beta}$. We call the functions $\overline{\Phi}_{\alpha,\beta}$ dependence functions and consider that they are affine,

$$\overline{\Phi}_{\alpha,\beta}(J) = \Phi_{\alpha,\beta} J + \Psi_{\alpha,\beta} N - \varphi^{(\alpha,\beta)}, \quad (2)$$
$$J \in V_{\alpha,\beta},\ N \in \mathbf{Z}^e,\ \Phi_{\alpha,\beta} \in \mathbf{Z}^{n_\alpha \times n_\beta},\ \Psi_{\alpha,\beta} \in \mathbf{Z}^{n_\alpha \times e},\ \varphi^{(\alpha,\beta)} \in \mathbf{Z}^{n_\alpha},\ (\alpha,\beta) \in P,$$

Example 2 (a continuation of Example 1). The dependencies of a loop nest are described by the following functions [1, 6.8]: $\overline{\Phi}_{1,3}(i,1) = (0\ 0)(i\ 1)^{\mathrm{T}} + 1$, $(i,1) \in V_{1,3} = \{(i,1) \in \mathbf{Z}^2 \mid 2 \le i \le N\}$, $\overline{\Phi}_{2,3}(i,1) = (1\ 0)(i\ 1)^{\mathrm{T}}$, $(i,1) \in V_{2,3} = V_{1,3}$, $\overline{\Phi}^{(1)}_{3,3}(i,j) = \begin{pmatrix} 0 & 1 \\ 0 & 1 \end{pmatrix}(i\ j)^{\mathrm{T}} - (0\ 1)^{\mathrm{T}}$, $(i,j) \in V^{(1)}_{3,3} = \{(i,j) \in \mathbf{Z}^2 \mid 3 \le i \le N, 2 \le j \le i-1\}$, $\overline{\Phi}^{(2)}_{3,3}(i,j) = E^{(2)}(i\ j)^{\mathrm{T}} - (0\ 1)^{\mathrm{T}}$, $(i,j) \in V^{(2)}_{3,3} = V^{(1)}_{3,3}$. ■

Dependence functions form a convenient mathematical apparatus for the description of the fine information structure of an algorithm, i.e., for the description of information relations at the level of separate operations of the algorithm. Names of these functions can be different, for example, covering functions of graphs of algorithms [1] or *h*-transformations [23, 24]. It is proved (the V. V. Voevodin theorem on information covering [1, 25]) that all dependencies of algorithms presented by a sufficiently wide linear class of programs can be specified by functions linearly dependent on loop parameters and external variables. Each function is defined on a linear polyhedron; the number of functions is independent of external variables and is proportional to the number of assignment operators for real-life linear programs. The proportionality coefficient does not exceed several units.

We denote $n = \max\limits_{1 \le \beta \le K} n_\beta$. Let us consider vector functions $\bar{t}^{(\beta)} : V_\beta \to \mathbf{Z}^n$, $1 \le \beta \le K$. Let each function $\bar{t}^{(\beta)}$ associate with an operation $S_\beta(J)$ a vector $\bar{t}^{(\beta)}(J) = (t_1^{(\beta)}(J), \ldots, t_n^{(\beta)}(J))$ with integer-valued coordinates. We assume that functions $t_\xi^{(\beta)}$ are affine,

$$t_\xi^{(\beta)}(J) = \tau^{(\beta,\xi)} J + b^{(\beta,\xi)} N + a_{\beta,\xi},$$
$$J \in V_\beta,\ \tau^{(\beta,\xi)} \in \mathbf{Z}^{n_\beta},\ b^{(\beta,\xi)},\ N \in \mathbf{Z}^e,\ a_{\beta,\xi} \in \mathbf{Z},\ 1 \le \beta \le K,\ 1 \le \xi \le n. \tag{3}$$

Functions $\bar{t}^{(\beta)}$ are called vector timing functions if the following conditions are fulfilled:

$$\operatorname{rank} T^{(\beta)} = n_\beta,\ 1 \le \beta \le K, \tag{4}$$

$$\bar{t}^{(\beta)}(J) \ge_{lex} \bar{t}^{(\alpha)}(I),\ J \in V_\beta,\ I \in V_\alpha,\ S_\alpha(I) \to S_\beta(J), \tag{5}$$

where $T^{(\beta)}$ is a matrix whose rows are composed of vectors $\tau^{(\beta,1)}, \ldots, \tau^{(\beta,n)}$ and the notation $\ge_{lex}$ means "lexicographically greater or equal to." It may be noted that, in the terminology of [1], functions $t_\xi^{(\beta)}$ are called unfoldings of the graph of an algorithm.

Vector timing functions specify transformation of loop nests. This means that the operation $S_\beta(J)$ executed at the Jth iteration of the initial loop nest will be executed at the $\bar{t}^{(\beta)}(J)$th iteration of the transformed loop nest. Thus, the components of the vector $\bar{t}^{(\beta)}$ are interpreted as parameters of the transformed loop nest for the operator S_β, i.e., $t_1^{(\beta)}$ is the parameter of the outermost loop and $t_n^{(\beta)}$ is the parameter of the innermost loop. The fulfillment of condition (4) guarantees the nondegeneracy of the transformation, and the fulfillment of condition (5) guarantees the preservation of the order of execution of informationally related operations of the algorithm, which provides the correctness of the transformation. A collection of vector functions $\bar{t}^{(1)}, \ldots, \bar{t}^{(K)}$ is called multidimensional timing.

Example 3 (a continuation of Example 2). Let us consider the following two-dimensional timing: $\bar{t}^{(1)}(1) = (1,1)$; $\bar{t}^{(2)}(i) = (i,1),\ 2 \le i \le N$; $\bar{t}^{(3)}(i,j) = (i,j),\ 2 \le i \le N,\ 1 \le j \le i-1$. We transform the initial loop nest according to the following multidimensional timing:

```
do i = 1, N
  do j = 1, i - 1
    if (i = j = 1)      S1: x(1) = b(1)
    if (i ≥ 2, j = 1)   S2: x(i) = b(i)
    if (i ≥ 2)          S3: x(i) = x(i) - a(i, j)x(j)
  enddo
enddo
```

■

We denote $\rho_{l,\beta,q} = \operatorname{rank} F_{l,\beta,q}$. Let us consider the basis of the space $\mathbf{Z}^{n_\beta}$ with base vectors $u_i^\perp$, $1 \le i \le \rho_{l,\beta,q}$ and u_i, $1 \le i \le n_\beta - \rho_{l,\beta,q}$, where u_i are base vectors of the subspace $\ker F_{l,\beta,q}$; if $\rho_{l,\beta,q} = n_\beta$, then the vectors u_i are absent. We denote the matrix whose columns are composed of vectors $u_i^\perp$ by $U_{l,\beta,q}^\perp$.

Let F be an element of the set W_l. We denote by $V_{l,\beta,q}(F)$ the set of iterations of the initial loop nest whose occurrence (l,β,q) uses the same array element $a_l(F)$,

$$V_{l,\beta,q}(F)=\{J\in V_\beta \mid \overline{F}_{l,\beta,q}(J)=F\}.$$

We denote by $\Lambda_{l,\beta,q}(F)$ the set specifying the coordinates of the projection of the set $V_{l,\beta,q}(F)$ onto the linear envelope of vectors u_i;

$$\Lambda_{l,\beta,q}(F)=\left\{(\lambda_1,\ldots,\lambda_{n_\beta-\rho_{l,\beta,q}})\mid J=J^\perp+\sum_{i=1}^{n_\beta-\rho_{l,\beta,q}}\lambda_i u_i,\ J\in V_{l,\beta,q}(F)\right\};$$

in the case when $\rho_{l,\beta,q}=n_\beta$, the set $\Lambda_{l,\beta,q}(F)$ is empty.

FUNCTIONS USING ARRAYS AT ITERATIONS OF LOOP NESTS

We introduce functions $\overline{d}^{(l,\beta,q)}: W_{l,\beta,q}\to \mathbf{Z}^n$, $(l,\beta,q)\in Q$, with affine coordinate functions of the form

$$\begin{gathered} d_\xi^{(l,\beta,q)}(F)=\eta^{(l,\beta,q,\xi)}F+z^{(l,\beta,q,\xi)}N+y_{(l,\beta,q,\xi)}(F),\\ F\in W_{l,\beta,q},\ \eta^{(l,\beta,q,\xi)}\in\mathbf{Z}^{\nu_l},\ z^{(l,\beta,q,\xi)},\ N\in\mathbf{Z}^e,\\ y_{l,\beta,q,\xi}(F)\in\mathbf{Z},\ (l,\beta,q)\in Q,\ 1\le\xi\le n. \end{gathered} \tag{6}$$

For each fixed $F\in W_{l,\beta,q}$, we require the fulfillment of the condition

$$d_\xi^{(l,\beta,q)}(F)=t_\xi^{(\beta)}(J),\ J\in V_{l,\beta,q}(F). \tag{7}$$

Then each function $\overline{d}^{(l,\beta,q)}$ determines iterations of the transformed loop nest at which elements of the array a_l connected with the occurrence (l,β,q) are used. We call functions $\overline{d}^{(l,\beta,q)}$ functions using arrays at iterations of loop nests. We note that functions $d_\xi^{(l,\beta,q)}$ are multivalued in the general case.

THEOREM 1. The coordinates $d_\xi^{(l,\beta,q)}$ of functions using arrays at iterations of a loop nest are specified by formulas (6) in which $\eta^{(l,\beta,q,\xi)}$ is a solution to the following system of equations:

$$\eta^{(l,\beta,q,\xi)}F_{l,\beta,q}U^\perp_{l,\beta,q}=\tau^{(\beta,\xi)}U^\perp_{l,\beta,q}, \tag{8}$$

and vectors $z^{(l,\beta,q,\xi)}$ and quantities $y_{l,\beta,q,\xi}(F)$ are specified by the equalities

$$z^{(l,\beta,q,\xi)}=b^{(\beta,\xi)}-\eta^{(l,\beta,q,\xi)}G_{l,\beta,q}, \tag{9}$$

$$y_{l,\beta,q,\xi}(F)=a_{\beta,\xi}-\eta^{(l,\beta,q,\xi)}f^{(l,\beta,q)}+\sum_{i=1}^{n_\beta-\rho_{l,\beta,q}}\lambda_i\tau^{(\beta,\xi)}u_i,\ (\lambda_1,\ldots,\lambda_{n_\beta-\rho_{l,\beta,q}})\in\Lambda_{l,\beta,q}(F). \tag{10}$$

Proof. Let F be an arbitrary fixed element of the set $W_{l,\beta,q}$. Then $F=\overline{F}_{l,\beta,q}(J)$, $J\in V_{l,\beta,q}(F)$, and the fulfillment of condition (7) is equivalent to the fulfillment of the equality $t_\xi^{(\beta)}(J)-d_\xi^{(l,\beta,q)}(\overline{F}_{l,\beta,q}(J))=0$ for any $J\in V_{l,\beta,q}(F)$.

We have

$$t_\xi^{(\beta)}(J)-d_\xi^{(l,\beta,q)}(\overline{F}_{l,\beta,q}(J))=\tau^{(\beta,\xi)}J+b^{(\beta,\xi)}N+a_{\beta,\xi}-(\eta^{(l,\beta,q,\xi)}\overline{F}_{l,\beta,q}(J)$$

$$+z^{(l,\beta,q,\xi)}N+y_{l,\beta,q,\xi})=\tau^{(\beta,\xi)}J+b^{(\beta,\xi)}N+a_{\beta,\xi}-\eta^{(l,\beta,q,\xi)}(F_{l,\beta,q}J+G_{l,\beta,q}N+f^{(l,\beta,q)})-z^{(l,\beta,q,\xi)}N-y_{l,\beta,q,\xi}$$

$$=(\tau^{(\beta,\xi)}-\eta^{(l,\beta,q,\xi)}F_{l,\beta,q})J+(b^{(\beta,\xi)}-\eta^{(l,\beta,q,\xi)}G_{l,\beta,q}-z^{(l,\beta,q,\xi)})N+a_{\beta,\xi}-\eta^{(l,\beta,q,\xi)}f^{(l,\beta,q)}-y_{l,\beta,q,\xi}.$$

Equality (7) is fulfilled if the value of every quantity $(\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)} F_{l,\beta,q})J$, $(b^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)} G_{l,\beta,q} - z^{(l,\beta,q,\xi)})N$, and $a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)} f^{(l,\beta,q)} - y_{l,\beta,q,\xi}$ reaches zero.

The first of these quantities equals zero for any vector J if the vector $\eta^{(l,\beta,q,\xi)}$ is determined as a solution of the following system of equations: $\eta^{(l,\beta,q,\xi)} F_{l,\beta,q} = \tau^{(\beta,\xi)}$. However, this system does not necessarily has a solution since the number of unknowns (coordinates of the vector $\eta^{(l,\beta,q,\xi)}$) is smaller than or equal to the number of equations ($\nu_l \le n_\beta$). To this end, we represent the vector J in the form $J = J^{\perp} + J^{(0)}$, where $J^{\perp} = \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} u_i^{\perp}$ and $J^{(0)} = \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i u_i$. Then we obtain

$$(\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)} F_{l,\beta,q})J = (\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)} F_{l,\beta,q})(J^{\perp} + J^{(0)})$$

$$= (\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)} F_{l,\beta,q})J^{\perp} + \tau^{(\beta,\xi)} J^{(0)}$$

$$= (\tau^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)} F_{l,\beta,q}) + \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} u_i^{\perp} + \tau^{(\beta,\xi)} \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i u_i$$

$$= \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} (\tau^{(\beta,\xi)} u_i^{\perp} - \eta^{(l,\beta,q,\xi)} F_{l,\beta,q} u_i^{\perp}) + \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i \tau^{(\beta,\xi)} u_i .$$

The number of unknowns of the system $\eta^{(l,\beta,q,\xi)} F_{l,\beta,q} u_i^{\perp} = \tau^{(\beta,\xi)} u_i^{\perp}$ is greater than or equal to the number of equations ($\nu_l \ge \rho_{l,\beta,q}$). The system always has a solution since the rank of the extended matrix is equal to the rank of the matrix of the system.

Thus, we obtain

$$t_\xi^{(\beta)}(J) - d_\xi^{(l,\beta,q)}(\overline{F}_{l,\beta,q}(J)) = \sum_{i=1}^{\rho_{l,\beta,q}} \lambda_i^{\perp} (\tau^{(\beta,\xi)} u_i^{\perp} - \eta^{(l,\beta,q,\xi)} F_{l,\beta,q} u_i^{\perp})$$

$$+ \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}} \lambda_i \tau^{(\beta,\xi)} u_i + (b^{(\beta,\xi)} - \eta^{(l,\beta,q,\xi)} G_{l,\beta,q} - z^{(l,\beta,q,\xi)})N + a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)} f^{(l,\beta,q)} - y_{l,\beta,q,\xi} .$$

Equality (7) is fulfilled for any $J \in V_{l,\beta,q}(F)$ if components of each vector $\eta^{(l,\beta,q,\xi)}$ are determined as a solution of a system of $\rho_{l,\beta,q}$ equations with ν_l unknowns,

$$\eta^{(l,\beta,q,\xi)} F_{l,\beta,q} u_i^{\perp} = \tau^{(\beta,\xi)} u_i^{\perp}, \ 1 \le i \le \rho_{l,\beta,q}, \tag{11}$$

and the matrix $z^{(l,\beta,q,\xi)}$ and functions $y_{l,\beta,q,\xi}(F)$ are specified by equalities (9) and (10), respectively. It may be noted that the system of equations (11) can be written in the form (8) and, for each fixed F, vectors $(\lambda_1, \ldots, \lambda_{n_\beta - \rho_{l,\beta,q}})$ belong to the set $\Lambda_{l,\beta,q}(F)$. ■

In a special case (when the vector of external variables is not taken into account), Theorem 1 is proved in [26].

Example 4 (a continuation of Example 3). Let us find functions using arrays at iterations of the transformed loop nest. Note that the transformation has been carried out according to the following two-dimensional timing: $\bar{t}^{(1)}(1) = (1,1)$; $\bar{t}^{(2)}(i) = (i,1)$, $2 \le i \le N$; $\bar{t}^{(3)}(i,j) = (i,j)$, $2 \le i \le N$, $1 \le j \le i-1$.

For the occurrences $(x, S_1, 1)$ and $(x, S_2, 1)$, according to equality (7), we obtain $\bar{d}^{(x,S_1,1)}(1) = (1,1)$ and $\bar{d}^{(x,S_2,1)}(i) = (i,1)$.

Let us consider the occurrences $(x, S_3, 1)$ and $(x, S_3, 2)$. It suffices to take only one of these occurrences. We have $\tau^{(3,1)} = (1,0)$, $\tau^{(3,2)} = (0,1)$, $b^{(3,1)} = b^{(3,2)} = 0$, $a_{3,1} = a_{3,2} = 0$; $F_{x,S_3,1} = (1\ 0)$, $G_{x,S_3,1} = 0$, $f^{(x,S_3,1)} = 0$; $\rho_{x,S_3,1} = 1$, $n_3 = 2$; $u_1 = (0,1)$ and $u_1^{\perp} = (1,0)$. For $\xi = 1$, system (8) assumes the form $\eta^{(x,S_3,1,1)} (1\ 0) \begin{pmatrix} 1 \\ 0 \end{pmatrix} = (1\ 0) \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and, hence, $\eta^{(x,S_3,1,1)} = 1$.

From relationships (9) and (10) we obtain $z^{(x,S_3,1,1)} = 0-1\cdot 0 = 0$ and $y_{x,S_3,1,1}(i) = 0-1\cdot 0+\lambda_1(1,0)(0,1) = 0$. For $\xi = 2$, we obtain $\eta^{(x,S_3,1,2)}(1\ 0)\begin{pmatrix}1\\0\end{pmatrix} = (1\ 0)\begin{pmatrix}1\\0\end{pmatrix}$ and, hence, $\eta^{(x,S_3,1,2)} = 0$; $z^{(x,S_3,1,2)} = 0-0\cdot 0 = 0$ and $y_{x,S_3,1,2}(i) = 0-0\cdot 0+$ $\lambda_1(0,1)(0,1) = \lambda_1$, $\lambda_1 \in \Lambda_{x,S_3,1}(i)$. For each $i = 1,\ldots,N$ we define the set $V_{x,S_3,1}(i) = \{(i,j)\,|\,1\le j\le i-1\}$. Hence, $\Lambda_{x,S_3,1}(i) = \{j\,|\,1\le j\le i-1\}$ and $y_{x,S_3,1,2}(i) = j$, $1\le j\le i-1$. We obtain $\bar{d}^{(x,S_3,1)}(i) = \bar{d}^{(x,S_3,2)}(i) = (i,j)$, $1\le j\le i-1$.

Let us consider the occurrence $(x, S_3, 3)$. We have $F_{x,S_3,3} = (0\ 1)$, $G_{x,S_3,3} = 0$, and $f^{(x,S_3,3)} = 0$; $\rho_{x,S_3,3} = 1$; $u_i = (1,0)$ and $u_1^{\perp} = (0,1)$. For $\xi = 1$, we obtain $\eta^{(x,S_3,3,1)}(0\ 1)\begin{pmatrix}0\\1\end{pmatrix} = (1\ 0)\begin{pmatrix}0\\1\end{pmatrix}$, whence $\eta^{(x,S_3,3,1)} = 0$; $z^{(x,S_3,3,1)} = 0-0\cdot 0 = 0$ and $y_{x,S_3,3,1}(j) = 0-0+\lambda_1(1,0)(1,0) = \lambda_1$, $\lambda_1 \in \Lambda_{x,S_3,3}(j)$. For each $i = 1,\ldots,N-1$, we define the set $V_{x,S_3,3}(j) = \{(i,j)\,|$ $j+1\le i\le N\}$. Hence, $\Lambda_{x,S_3,3}(j) = \{i\,|\,j+1\le i\le N\}$ and $y_{x,S_3,1,1}(j) = i$, $j+1\le i\le N$. For $\xi = 2$, we obtain $\eta^{(x,S_3,3,2)}(0\ 1)\begin{pmatrix}0\\1\end{pmatrix} = (0\ 1)\begin{pmatrix}0\\1\end{pmatrix}$, whence $\eta^{(x,S_3,3,2)} = 1$; $z^{(x,S_3,3,2)} = 0-1\cdot 0 = 0$ and $y_{x,S_3,3,2}(j) = 0-0+\lambda_1(0,1)(1,0) = 0$. Thus, we have $\bar{d}^{(x,S_3,2)}(j) = (i,j)$, $j+1\le i\le N$.

We obtain functions using the array b. For the occurrences $(b, S_1, 1)$ and $(b, S_2, 1)$, according to equality (7), we find $\bar{d}^{(b,S_1,1)}(1) = (1,1)$ and $\bar{d}^{(b,S_2,1)}(i) = (i,1)$.

Let us determine the function using the array a (the occurrence $(a, S_3, 1)$). We have $\tau^{(3,1)} = (1,0)$, $\tau^{(3,2)} = (0,1)$, $b^{(3,1)} = b^{(3,2)} = 0$, and $a_{3,1} = a_{3,2} = 0$; $F_{a,S_3,1} = E^{(2)}$, $G_{a,S_3,1} = (0\ 0)^{\mathrm{T}}$, and $f^{(a,S_3,1)} = (0,0)$; $\rho_{a,S_3,1} = \operatorname{rank} E^{(2)} = 2$ and $n_3 = 2$; u_i are absent, $u_1^{\perp} = (1,0)$, and $u_2^{\perp} = (0,1)$. For $\xi = 1$, system (8) assumes the form $\eta^{(a,S_3,1,1)}E^{(2)}E^{(2)} = (1\ 0)E^{(2)}$ and, hence, $\eta^{(a,S_3,1,1)} = (1,0)$. From relationships (9) and (10) we obtain $z^{(a,S_3,1,1)} = 0-(1\ 0)(0\ 0)^{\mathrm{T}} = 0$ and $y_{a,S_3,1,1}(i,j) = 0-(1\ 0)(0\ 0)^{\mathrm{T}} = 0$. Similarly, for $\xi = 2$, we have $\eta^{(a,S_3,1,2)}E^{(2)}E^{(2)} = (0\ 1)E^{(2)}$, whence $\eta^{(a,S_3,1,2)} = (0,1)$, $z^{(a,S_3,1,2)} = 0-(0\ 1)(0\ 0)^{\mathrm{T}} = 0$, and $y_{a,S_3,1,2}(i,j) = 0-(0\ 1)(0\ 0)^{\mathrm{T}} = 0$. Accordingly, we obtain $\bar{d}^{(a,S_3,1)}(i,j) = (i,j)$. ■

THEOREM 2. If $\rho_{l,\beta,q} < n_\beta$, then each array element connected with the occurrence (l,β,q) is used at iterations $\bar{d}^{(l,\beta,q)}(F)$ differing from one another in linear combinations of vectors $T^{(\beta)}u_i$, $1\le i\le n_\beta-\rho_{l,\beta,q}$. If $\rho_{l,\beta,q} = n_\beta$, then the corresponding array element is used at a fixed iteration.

Proof. We first show that, for any value of $\eta^{(l,\beta,q,\xi)}$ that is a solution to system (8), for a fixed F, the quantity $\eta^{(l,\beta,q,\xi)}F-\eta^{(l,\beta,q,\xi)}G_{l,\beta,q}N-\eta^{(l,\beta,q,\xi)}f^{(l,\beta,q)}$ assumes the same value. Any vector J for which $F_{l,\beta,q}J+G_{l,\beta,q}N+f^{(l,\beta,q)} = F$ can be represented in the form $J = J^{\perp}+J^{(0)}$, where $J^{(0)}\in\ker F_{l,\beta,q}$ and $J^{\perp}$ is the normal solution of the system of equations $F_{l,\beta,q}J = F-G_{l,\beta,q}N-f^{(l,\beta,q)}$, $J^{\perp} = \sum_{i=1}^{\rho_{l,\beta,q}}\lambda_i^{\perp}u_i^{\perp}$. With allowance for the proof of Theorem 1, we obtain the value

$$\eta^{(l,\beta,q,\xi)}F-\eta^{(l,\beta,q,\xi)}G_{l,\beta,q}N-\eta^{(l,\beta,q,\xi)}f^{(l,\beta,q)}$$

$$=\eta^{(l,\beta,q,\xi)}(F_{l,\beta,q}J+G_{l,\beta,q}N+f^{(l,\beta,q)})-\eta^{(l,\beta,q,\xi)}G_{l,\beta,q}N-\eta^{(l,\beta,q,\xi)}f^{(l,\beta,q)}$$

$$=\eta^{(l,\beta,q,\xi)}F_{l,\beta,q}J=\eta^{(l,\beta,q,\xi)}F_{l,\beta,q}(J^{\perp}+J^{(0)})$$

$$=\eta^{(l,\beta,q,\xi)}F_{l,\beta,q}\sum_{i=1}^{\rho_{l,\beta,q}}\lambda_i^{\perp}u_i^{\perp}=\sum_{i=1}^{\rho_{l,\beta,q}}\lambda_i^{\perp}\eta^{(l,\beta,q,\xi)}F_{l,\beta,q}u_i^{\perp}=\sum_{i=1}^{\rho_{l,\beta,q}}\lambda_i^{\perp}\tau^{(\beta,\xi)}u_i^{\perp}=\tau^{(\beta,\xi)}J^{\perp}$$

that is a constant.

By Theorem 1, the coordinates of functions using arrays at iterations of loop nests are specified in the form

$$d_{\xi}^{(l,\beta,q)}(F)=\eta^{(l,\beta,q,\xi)}F+(b^{(\beta,\xi)}-\eta^{(l,\beta,q,\xi)}G_{l,\beta,q})N+a_{\beta,\xi}-\eta^{(l,\beta,q,\xi)}f^{(l,\beta,q)}$$
$$+\sum_{i=1}^{n_\beta-\rho_{l,\beta,q}}\lambda_i\tau^{(\beta,\xi)}u_i=J^{\perp}+b^{(\beta,\xi)}N+a_{\beta,\xi}+\sum_{i=1}^{n_\beta-\rho_{l,\beta,q}}\lambda_i\tau^{(\beta,\xi)}u_i.$$

Hence, if $\rho_{l,\beta,q}<n_\beta$, then the iterations at which the element $a_l(F)$ connected with the occurrence (l,β,q) is used can differ only in linear combinations of vectors $T^{(\beta)}u_i$, $1\le i\le n_\beta-\rho_{l,\beta,q}$.

If $\rho_{l,\beta,q}=n_\beta$, then the sum $\sum_{i=1}^{n_\beta-\rho_{l,\beta,q}}\lambda_i\tau^{(\beta,\xi)}$ is absent (the set $\Lambda_{l,\beta,q}(F)$ is empty) and the value of $d_{\xi}^{(l,\beta,q)}(F)$ becomes fixed. ■

Let us consider a set $\Xi_{fix}=\{\xi_1,\dots,\xi_\theta\}\subset\{1,\dots,n\}$ composed of θ arbitrary elements of a set $\{1,\dots,n\}$. We choose θ coordinates of multidimensional timing $t_{\xi_i}^{(\beta)}$, $\xi_i\in\Xi_{fix}$, and denote by $\bar{t}_{fix}^{(\beta)}$ a vector function composed of the chosen coordinates $t_{\xi_1}^{(\beta)},\dots,t_{\xi_\theta}^{(\beta)}$.

We denote by $T_{fix}^{(\beta)}$ a matrix whose rows are composed of vectors $\tau^{(\beta,\xi_i)}$, $\xi_i\in\Xi_{fix}$; $\rho_{l,\beta,q}^{fix}=\operatorname{rank}\begin{pmatrix}F_{l,\beta,q}\\T_{fix}^{(\beta)}\end{pmatrix}$. We denote by u_i^{fix}, $1\le i\le n_\beta-\rho_{l,\beta,q}^{fix}$, base vectors of the intersection of the subspace $\ker\begin{pmatrix}F_{l,\beta,q}\\T_{fix}^{(\beta)}\end{pmatrix}$ and $\mathbf{Z}^{n_\beta}$; vectors u_i^{fix} can be absent. Let us consider the basis of the space $\mathbf{Z}^{n_\beta}$ with base vectors $u_i^{\perp,fix}$, $1\le i\le\rho_{l,\beta,q}^{fix}$, and u_i^{fix}, $1\le i\le n_\beta-\rho_{l,\beta,q}^{fix}$. We denote the matrix whose columns are composed of vectors $u_i^{\perp,fix}$ by $U_{l,\beta,q}^{\perp,fix}$.

We assume that T_{fix} is an arbitrary fixed element of the set of values of the function $\bar{t}_{fix}^{(\beta)}$ and consider the set

$$V_\beta^{fix}(T_{fix})=\{J\in V_\beta\mid\bar{t}_{fix}^{(\beta)}(J)=T_{fix}\}.$$

Let us consider the set $V_{l,\beta,q}(F)\cap V_\beta^{fix}(T_{fix})$ of iterations of the initial loop nest whose occurrence (l,β,q) uses the same data item $a_l(F)$ and that are realized in the transformed loop nest at iterations with the fixed value of the function $\bar{t}_{fix}^{(\beta)}$ equal to T_{fix}. We denote by $\Lambda_{l,\beta,q}^{fix}(F,T_{fix})$ the set specifying the coordinates of the projection of the set $V_{l,\beta,q}(F)\cap V_\beta^{fix}(T_{fix})$ onto the linear envelope of vectors u_i^{fix},

$$\Lambda_{l,\beta,q}^{fix}(F,T_{fix})=\left\{(\lambda_1^{fix},\dots,\lambda_{n_\beta-\rho_{l,\beta,q}^{fix}}^{fix})\mid J=J^{\perp,fix}+\sum_{i=1}^{n_\beta-\rho_{l,\beta,q}^{fix}}\lambda_i^{fix}u_i^{fix},J\in V_{l,\beta,q}(F)\cap V_\beta^{fix}(T_{fix})\right\};$$

in the case when $\rho_{l,\beta,q}^{fix}=n_\beta$, the set $\Lambda_{l,\beta,q}^{fix}(F,T_{fix})$ is empty.

THEOREM 3. Iterations of the transformed loop nest that use an array element $a_l(F)$, $F\in W_{l,\beta,q}$, with the fixed value of the function $\bar{t}_{fix}^{(\beta)}$ equal to T_{fix} are determined by values of functions $d_{\xi}^{(l,\beta,q)}$ that are specified by formulas (6), where $\eta^{(l,\beta,q,\xi)}$ is a solution of the system of equations

$$\eta^{(l,\beta,q,\xi)}F_{l,\beta,q}U_{l,\beta,q}^{\perp,fix}=\tau^{(\beta,\xi)}U_{l,\beta,q}^{\perp,fix},\tag{12}$$

and the matrices $z^{(l,\beta,q,\xi)}$ and quantities $y_{l,\beta,q,\xi}(F)$ are specified, respectively, by equality (9) and the following equality:

$$y_{l,\beta,q,\xi}(F) = a_{\beta,\xi} - \eta^{(l,\beta,q,\xi)} f^{(l,\beta,q)} + \sum_{i=1}^{n_\beta - \rho_{l,\beta,q}^{fix}} \lambda_i^{fix} \tau^{(\beta,\xi)} u_i^{fix}, \tag{13}$$

$$(\lambda_1^{fix}, \ldots, \lambda_{n_\beta - \rho_{l,\beta,q}^{fix}}^{fix}) \in \Lambda_{l,\beta,q}^{fix}(F, T_{fix}).$$

THEOREM 4. If $\rho_{l,\beta,q}^{fix} < n_\beta$, then, in the transformed loop nest with a fixed value of the function $\bar{t}_{fix}^{(\beta)}$, each array element connected with the occurrence (l,β,q) is used at iterations differing from one another in linear combinations of vectors $T_{fix}^{(\beta)} u_i^{fix}$, $1 \le i \le n_\beta - \rho_{l,\beta,q}^{fix}$, where $T_{fix}^{(\beta)}$ is a matrix whose rows are composed of vectors $\tau^{(\beta,\xi_i)}$, $\xi_i \in \{1,\ldots,n\} \setminus \Xi_{fix}$. If $\rho_{l,\beta,q}^{fix} = n_\beta$, then, when the value of the function $\bar{t}_{fix}^{(\beta)}$ is fixed, an array element is used only once.

In essence, the proofs of Theorems 3 and 4 repeat the proofs of Theorems 1 and 2, respectively, if F and T_{fix} are fixed.

DISTRIBUTION OF OPERATIONS AND DATA AMONG PROCESSORS AND ITERATIONS

We introduce a set $\Xi_s = \{\xi_1, \ldots, \xi_r\} \subset \{1,\ldots,n\}$ composed of r arbitrary elements of a set $\{1,\ldots,n\}$ and a set $\Xi_t = \{\xi_{r+1}, \ldots, \xi_n\} \subset \{1,\ldots,n\} \setminus \Xi_s$. For convenience, we consider that the sets Ξ_s and Ξ_t are ordered as follows: $\xi_i < \xi_j$ if $i < j$ (for every $\xi_i, \xi_j \in \Xi_s$ and $\xi_i, \xi_j \in \Xi_t$).

We introduce the following vector functions: $\bar{t}_s^{(\beta)}$ is composed of functions $t_{\xi_i}^{(\beta)}$, $\xi_i \in \Xi_s$, and $\bar{t}_t^{(\beta)}$ is composed of functions $t_{\xi_i}^{(\beta)}$, $\xi_i \in \Xi_t$; $\bar{d}_s^{(l,\beta,q)}$ is composed of functions $d_{\xi_i}^{(l,\beta,q)}$, $\xi_i \in \Xi_s$, and $\bar{d}_t^{(l,\beta,q)}$ is composed of functions $d_{\xi_i}^{(l,\beta,q)}$, $\xi_i \in \Xi_t$.

To implement an algorithm on a parallel computer, we assume that functions $\bar{t}_s^{(\beta)}$ specify a mapping of operations of the algorithm into the r-dimensional space of virtual processors and functions $\bar{t}_t^{(\beta)}$ specify iterations executed by the processors. Functions $\bar{d}_s^{(l,\beta,q)}$ specify the coordinates of the processors in which the array elements connected with the occurrence (l,β,q) are used, and functions $\bar{d}_t^{(l,\beta,q)}$ specify the iterations at which these elements are used.

Using Theorem 1, the distribution of input and output data among processors can be found and information on sizes of arrays used in each processor can be obtained.

The distribution of input arrays among virtual processors can be specified with the help of the functions $\bar{d}_s^{(l,\beta,q)}$. To this end, an element $a_l(F)$ of the input array a_l, $F \in W_{l,\beta,q}$, should be allocated to the processor $P_{in}(\bar{d}_s^{(l,\beta,q)}(F))$ whose coordinates are values of functions $d_{\xi_i}^{(l,\beta,q)}(F)$, $\xi_i \in \Xi_s$, corresponding to a vector $(d_1^{(l,\beta,q)}(F), \ldots, d_{\xi_r}^{(l,\beta,q)}(F))$ with the lexicographically smallest value.

The results of computations, i.e., elements $a_l(F)$ of the output array, are stored in the processor $P_{out}(\bar{d}_s^{(l,\beta,q)}(F))$ whose coordinates are values of the functions $d_{\xi_i}^{(l,\beta,q)}(F)$, $\xi_i \in \Xi_s$, corresponding to the vector $(d_1^{(l,\beta,q)}(F), \ldots, d_{\xi_r}^{(l,\beta,q)}(F))$ with the lexicographically largest value.

The functions $\bar{d}_s^{(l,\beta,q)}$ also make it possible to obtain information on the sizes of the arrays used in each processor. To determine elements $a_l(F)$ of the output array a_l that are used in the processor with coordinates $(t_1,\ldots,t_r)$, all elements F of the set $W_{l,\beta,q}$ should be found for which $(d_{\xi_1}^{(l,\beta,q)}(F), \ldots, d_{\xi_r}^{(l,\beta,q)}(F)) = (t_1,\ldots,t_r)$.

We introduce the following denotations: $T_s^{(\beta)}$ is a matrix whose rows are composed of vectors $\tau^{(\beta,\xi_i)}$, $\xi_i \in \Xi_s$; $T_t^{(\beta)}$ is a matrix whose rows are composed of vectors $\tau^{(\beta,\xi_i)}$, $\xi_i \in \Xi_t$; $\rho_{l,\beta,q}^{s} = \operatorname{rank}\begin{pmatrix} F_{l,\beta,q} \\ T_s^{(\beta)} \end{pmatrix}$ and $\rho_{l,\beta,q}^{t} = \operatorname{rank}\begin{pmatrix} F_{l,\beta,q} \\ T_t^{(\beta)} \end{pmatrix}$.

Let us formulate the corollaries from Theorem 2 that allow one to find out whether it is required to transfer data between processors and iterations of a parallel algorithm or data are used by one processor or at one iteration and also to specify vectors determining the direction of transmission.

COROLLARY 1. If $\rho_{l,\beta,q} < \rho^s_{l,\beta,q}$, then each array element connected with the occurrence (l,β,q) is used by the virtual processors whose communications can be specified by vectors $T_s^{(\beta)} u_i$, $1 \le i \le n_\beta - \rho_{l,\beta,q}$; if $\rho_{l,\beta,q} = \rho^s_{l,\beta,q}$, then the array element is used only by one processor.

In fact, it follows from the proof of Theorem 2 that, for each array element connected with the occurrence (l,β,q), the values of $\bar{d}_s^{(l,\beta,q)}(F)$ differ in linear combinations of vectors $T_s^{(\beta)} u_i$, $1 \le i \le n_\beta - \rho^s_{l,\beta,q}$. If $\rho_{l,\beta,q} = \rho^s_{l,\beta,q}$, then $\ker \begin{pmatrix} F_{l,\beta,q} \\ T_s^{(\beta)} \end{pmatrix} = \ker F_{l,\beta,q}$ and $T_s^{(\beta)} u_i = 0$ for all u_i. Hence, in this case, $\bar{d}_s^{(l,\beta,q)}(F)$ assumes a fixed value, i.e., the array element is used by only one processor.

COROLLARY 2. If $\rho_{l,\beta,q} < \rho^t_{l,\beta,q}$, then each array element connected with the occurrence (l,β,q) is used at iterations differing in linear combinations of vectors $T_t^{(\beta)} u_i$, $1 \le i \le n_\beta - \rho_{l,\beta,q}$; if $\rho_{l,\beta,q} = \rho^t_{l,\beta,q}$, then the array element is used at only one iteration.

The proof of Corollary 2 is similar to the proof of Corollary 1.

We denote by u_i^s, $1 \le i \le n_\beta - \rho^s_{l,\beta,q}$, base vectors of the intersection of the subspace $\ker \begin{pmatrix} F_{l,\beta,q} \\ T_s^{(\beta)} \end{pmatrix}$ and $\mathbf{Z}^{n_\beta}$ and by u_i^t, $1 \le i \le n_\beta - \rho^t_{l,\beta,q}$, base vectors of the intersection of the subspace $\ker \begin{pmatrix} F_{l,\beta,q} \\ T_t^{(\beta)} \end{pmatrix}$ and $\mathbf{Z}^{n_\beta}$; the vectors u_i^s and u_i^t can be absent.

Let us formulate the corollaries from Theorem 4 that allow one to reveal the character of using data by a fixed processor or at a fixed iteration of the parallel algorithm.

COROLLARY 3. If $\rho^s_{l,\beta,q} < n_\beta$, then, in one virtual processor, each array element connected with the occurrence (l,β,q) is used at iterations differing in linear combinations of vectors $T_t^{(\beta)} u_i^s$, $1 \le i \le n_\beta - \rho^s_{l,\beta,q}$; if $\rho^s_{l,\beta,q} = n_\beta$, then the array element is used in one processor only at one iteration.

COROLLARY 4. If $\rho^t_{l,\beta,q} < n_\beta$, then, at one iteration, each array element connected with the occurrence (l,β,q) is used by virtual processors such that communications between them can be specified by vectors $T_s^{(\beta)} u_i^t$, $1 \le i \le n_\beta - \rho^t_{l,\beta,q}$; if $\rho^t_{l,\beta,q} = n_\beta$, then the array element is used at one iteration only in one processor.

Corollaries 3 and 4 can be obtained if, in the conditions of Theorem 4, we put $\Xi_{fix} = \Xi_s$ and $\Xi_{fix} = \Xi_t$, respectively.

DATA EXCHANGE ORGANIZATION

In practice, even an optimal initial distribution of data does not eliminate the necessity of data exchange between processors during the execution of an algorithm. As follows from Corollary 1, data exchange can be required for the elements of an array a_l that are connected with an occurrence (l,β,q) if the inequality $\rho^s_{l,\beta,q} > \rho_{l,\beta,q}$ is fulfilled.

As is well known, on parallel computers with distributed memory, such structured communications as broadcast, scatter, gather, reduction, and also data translation are executed much more quickly than the corresponding collection of point–to–point communications. Therefore, it is desirable to reveal the possibility of organization of such communications.

We will formulate and investigate the conditions that, in some cases, allow one to determine the possibility of organization of mostly used fast communications, namely, data broadcast and data translation. Broadcast (simultaneous propagation) is the transmission of a data item to a group of processors in which the data item is used as an argument at one iteration. Translation is the transmission of a data item from one processor to another in the case when an array element is

used in turn in different processors. Point-to-point communication is the transmission of a data item from one processor to another.

Organization of broadcast. Assume that we have functions of the form (6) that specify the use of data arrays by virtual processors and n-dimensional timing with coordinates of the form (3); r coordinates specify a spatial mapping of operations of an algorithm into the r-dimensional space of virtual processors and the other $n-r$ coordinates specify iterations executed by processors in lexicographic order.

Let a vector T be an element of the set of values of a function $\bar{t}_t^{(\beta)}$. We consider the set $V_\beta^t(T)=\{J\in V_\beta \mid \bar{t}_t^{(\beta)}(J)=T\}$ of iterations of the initial loop nest at which an operator S_β is executed and that are implemented in the space of virtual processors at the Tth iteration and the set $V_{l,\beta,q}(F)\cap V_\beta^t(T)$ of iterations of the initial loop nest whose occurrence (l,β,q) uses the same data item $a_l(F)$ and that are implemented in the space of virtual processors at the Tth iteration.

We denote by $U_{l,\beta,q}^t$ a matrix whose columns are composed of vectors u_i^t.

LEMMA 1 [18]. Let there be true dependencies generated by an occurrence (l,β,q), and let $\overline{\Phi}_{\alpha,\beta}$ be a dependence function. If the condition

$$\Phi_{\alpha,\beta}U_{l,\beta,q}^t=0 \tag{14}$$

is fulfilled, then the data item $a_l(F)$ is not redefined between iterations of the set $V_{l,\beta,q}(F)\cap V_\beta^t(T)$.

THEOREM 5. Let $\rho_{l,\beta,q}^t<n_\beta$. At the Tth iteration, the broadcast of the data item $a_l(F)$ to processors $P(\bar{d}_s^{(l,\beta,q)}(F))$ can be organized, where the parameters of functions $\bar{d}_s^{(l,\beta,q)}$ are specified according to formulas (12), (9), and (13) if we put $\Xi_{fix}=\Xi_t$ and $T_{fix}=T$ in one of the following cases:

(1) elements of the array a_l occur only in the right sides of operators of the algorithm;

(2) for a true dependence generated by the occurrence (l,β,q), condition (14) is fulfilled.

Proof. It follows from the condition of the theorem that the function $\overline{F}_{l,\beta,q}$ occurs in the right side of the operator S_β and, hence, elements of the array a_l are used as arguments at the qth occurrence in the operator S_β. Since $\rho_{l,\beta,q}^t<n_\beta$, according to Corollary 4, at one iteration, each array element connected with the occurrence (l,β,q) is used by virtual processors, and communications between them can be specified by vectors $T_s^{(\beta)}u_i^t$, $1\le i\le n_\beta-\rho_{l,\beta,q}^t$; by Theorem 3, when $T_{fix}=T$ and $\Xi_{fix}=\Xi_t$, the coordinates of such processors are specified by the function $\bar{d}_s^{(l,\beta,q)}(F)$ whose parameters are determined from system (12) and relationships (9) and (13). The processors use the same value of the data item $a_l(F)$ since this data item is not redefined between iterations of the set $V_{l,\beta,q}(F)\cap V_\beta^t(T)$; in case (1), this data item is not redefined at all, and, in case (2), the impossibility of redefinition is guaranteed by condition (14). ■

In case (1) of Theorem 5, the broadcast of the element $a_l(F)$ is carried out from the processor $P_{in}(\bar{d}_s^{(l,\beta,q)}(F))$ to which $a_l(F)$ is distributed if another scheme of distribution of input arrays (for example, array replication) has not been used. In case (2), the broadcast is carried out from the processor in which $a_l(F)$ has been computed.

It may be noted that, under the conditions of Theorem 5, broadcast can be degenerate if, by virtue of distinctive features of the set V_β, the set $V_{l,\beta,q}(F)\cap V_\beta^t(T)$ contains only one element.

Organization of data translation. We will formulate and prove theorems that allow one to reveal the possibility of organization of translation of a data item. In Theorem 6, the case will be investigated when a data item is used as an argument at different iterations by different processors and the data item is transferred at several iterations. In Theorem 7, the case will be investigated when a data item is used as an argument and is redefined in turn by different processors at one iteration with a shift in time in the course of program execution.

Let $\rho_{l,\beta,q}<n_\beta$. We denote by $U_{l,\beta,q}$ a matrix whose columns are composed of base vectors u_i, $1\le i\le n_\beta-\rho_{l,\beta,q}$.

LEMMA 2 [18]. Let there be true dependencies generated by an occurrence (l, β, q), and let $\overline{\Phi}_{\alpha,\beta}$ be a dependence function. If the condition

$$\Phi_{\alpha,\beta} U_{l,\beta,q} = 0 \tag{15}$$

is fulfilled, then, between iterations of the set $V_{l,\beta,q}(F)$, the data item $a_l(F)$ is not redefined.

THEOREM 6. Let $\rho^t_{l,\beta,q} > \rho_{l,\beta,q}$, let $\rho^s_{l,\beta,q} > \rho_{l,\beta,q}$, and let $\rho^t_{l,\beta,q} = n_\beta$. At iterations $\overline{d}_t^{(l,\beta,q)}(F)$, the translation of the data item $a_l(F)$ between processors $P(\overline{d}_s^{(l,\beta,q)}(F))$ can be organized, where the parameters of the functions $\overline{d}_s^{(l,\beta,q)}$ and $\overline{d}_t^{(l,\beta,q)}$ are specified according to formulas (8), (9), and (10) in one of the following cases:

(1) the array a_l occurs only in the right sides of operators of the algorithm;

(2) condition (15) is fulfilled for the true dependence generated by the occurrence (l, β, q).

Proof. The function $\overline{F}_{l,\beta,q}$ occurs in the right side of the operator S_β and, hence, elements of the array a_l are used at the qth occurrence in the operator S_β as arguments. The condition $\rho^t_{l,\beta,q} > \rho_{l,\beta,q}$ means that the data item $a_l(F)$ is used in the course of program execution at the qth occurrence of the array a_l in the operator S_β at several iterations (Corollary 2), and the condition $\rho^s_{l,\beta,q} > \rho_{l,\beta,q}$ means that $a_l(F)$ is used in several processors (Corollary 1); the condition $\rho^t_{l,\beta,q} = n_\beta$ implies (Corollary 4) that, at a fixed iteration, the array a_l is used by only one virtual processor.

Thus, the data item $a_l(F)$ is used at several iterations $\overline{d}_t^{(l,\beta,q)}(F)$ in several processors $P(\overline{d}_s^{(l,\beta,q)}(F))$, and this data item is used at each iteration by one processor. By Theorem 1, the parameters of the functions are determined from system (8) and relationships (9) and (10). Processors use the same data item since, between iterations of the set $V_{l,\beta,q}(F)$, the data item $a_l(F)$ is not redefined: in case (1), this data item is not redefined at all and, in the case (2), the impossibility of redefinition is guaranteed by condition (15). ■

Translation should be carried out according to the lexicographic ordering of vectors $(d_1^{(l,\beta,q)}(F), \ldots, d_{\xi_r}^{(l,\beta,q)}(F))$. In case (1) of Theorem 6, before the beginning of translation, the data item $a_l(F)$ is, according to the initial distribution, in the local memory of the processor $P_{in}(\overline{d}_s^{(l,\beta,q)}(F))$ that is the first to participate in translation. In case (2) of Theorems 6, before the beginning of translation, the data item $a_l(F)$ should be transferred from the processor in which $a_l(F)$ has been computed to the processor $P(\overline{d}_s^{(l,\beta,q)}(F))$ whose coordinates are values of functions $d_{\xi_i}^{(l,\beta,q)}(F)$, $\xi_i \in \Xi_s$, that correspond to the vector $(d_1^{(l,\beta,q)}(F), \ldots, d_{\xi_r}^{(l,\beta,q)}(F))$ with the lexicographically smallest value.

THEOREM 7. Let $\rho^t_{l,\beta,q} < n_\beta$, and let there be the true dependence generated by the occurrence (l, β, p) in the left side of an operator S_β and the occurrence (l, β, q) in the right side of the operator. Then the translation of the data item $a_l(F)$ between processors $P(\overline{d}_s^{(l,\beta,q)}(F))$ at the Tth iteration can be organized, where the parameters of the functions $\overline{d}_s^{(l,\beta,q)}$ are specified according to formulas (12), (9), and (13) under the assumption that $\Xi_{fix} = \Xi_t$ and $T_{fix} = T$.

Proof. The condition $\rho^t_{l,\beta,q} < n_\beta$ means that, generally speaking, the operator S_β is executed at the Tth iteration on more than one processor $P(\overline{d}_s^{(l,\beta,q)}(F))$ (Corollary 4). Since there is a true dependence generated by the occurrence p of the array a_l in the left side of the operator S_β and by the occurrence q in the right side of the operator, the data item $a_l(F)$ is used as an argument at the Tth iteration and is redefined. The aforesaid means that the translation of the data item $a_l(F)$ between processors $P(\overline{d}_s^{(l,\beta,q)}(F))$ can be organized at the Tth iteration. By Theorem 3, when $T_{fix} = T$ and $\Xi_{fix} = \Xi_t$, the coordinates of such processors are specified by the function $\overline{d}_s^{(l,\beta,q)}(F)$ whose parameters are determined from system (12) and relationships (9) and (13). ■

Translation should be carried out according to the lexicographic order of vectors $(d_1^{(l,\beta,q)}(F), \ldots, d_{\xi_r}^{(l,\beta,q)}(F))$. Before the beginning of the translation of the data item $a_l(F)$, it should be transferred from the processor in which it is stored to the processor $P(\overline{d}_s^{(l,\beta,q)}(F))$ whose coordinates are the values of functions $d_{\xi_i}^{(l,\beta,q)}(F)$, $\xi_i \in \Xi_s$, that correspond to the vector $(d_1^{(l,\beta,q)}(F), \ldots, d_{\xi_r}^{(l,\beta,q)}(F))$ with the lexicographically smallest value.

As in the case of broadcast, translation can be degenerate if the set $V_{l,\beta,q}(F) \cap V^t_{\beta}(T)$ consists of only one element.

Example 5 (a continuation of Example 4). According to the functions that use arrays and have been found above, we establish a distribution of array elements among processors and iterations of a parallel algorithm and determine a data exchange scheme.

Let us consider the case when $\Xi_s = \{1\}$ and $\Xi_t = \{2\}$. The first coordinate of the multidimensional timing specifies a mapping of operations of the algorithm onto a processor row, and the second coordinate specifies the order of execution of the operations by processors. We write the corresponding code (uniformly for each processor) destined for the execution of the algorithm on N processors. The symbol p denotes the number of a processor. The loop variable t corresponds to iterations of the algorithm

```
if (1 ≤ p ≤ N)
  do t = 1, max(1, p − 1)
    if (p = t = 1)      S1 : x(1) = b(1)
    if (p ≥ 2, t = 1)   S2 : x(p) = b(p)
    if (p ≥ 2)          S3 : x(p) = x(p) − a(p, t)x(t)
  enddo
endif
```

We have $T^{(1)}_s = (0)$, $T^{(2)}_s = (1)$, and $T^{(3)}_s = (1,0)$; $T^{(1)}_t = (0)$, $T^{(2)}_t = (0)$, and $T^{(3)}_t = (0,1)$. The equalities $\rho_{a_l,S_\beta,q} = \rho^s_{a_l,S_\beta,q}$ are fulfilled for all occurrences (a_l, S_β, q) except for $(x, S_3, 3)$. According to Corollary 1, each array element connected with any occurrence (a_l, S_β, q) except for $(x, S_3, 3)$ is used by only one processor. To use the third occurrence of the array x in the operator S_3, data should be transferred.

The true dependencies generated by the third occurrence of the array x in the operator S_3 are specified by the functions $\overline{\Phi}_{1,3}$ and $\overline{\Phi}^{(1)}_{3,3}$. For both functions, condition (14), namely, $\Phi_{1,3}U^t_{x,S_3,3} = (0\ 0)\begin{pmatrix}1\\0\end{pmatrix} = 0$ and $\Phi^{(1)}_{3,3}U^t_{x,S_3,3} = \begin{pmatrix}0 & 1\\0 & 1\end{pmatrix}\begin{pmatrix}1\\0\end{pmatrix} = 0$ are fulfilled. Hence, by Theorem 5 ($\rho^t_{x,S_3,3} = 1$, $n_3 = 2$, and $\rho^t_{x,S_3,3} < n_3$), at the Tth iteration, the broadcast of the data item $x(t)$ to the processors $P(\overline{d}^{(x,S_3,3)}_s(t)) = P(\eta^{(x,S_3,3,1)}t + z^{(x,S_3,3,1)}N + y_{x,S_3,3,1}(t))$ can be carried out, where $\eta^{(x,S_3,3,1)}$, $z^{(x,S_3,3,1)}$, and $y_{x,S_3,3,1}(t)$ are found by formulas (12), (9), and (13) under the assumption that $\Xi_{fix} = \Xi_t = \{2\}$ and $T_{fix} = T$. We obtain $\eta^{(x,S_3,3,1)}(0\ 1)\begin{pmatrix}0\\1\end{pmatrix} = (1\ 0)\begin{pmatrix}0\\1\end{pmatrix}$, whence $\eta^{(x,S_3,3,1)} = 0$; $z^{(x,S_3,3,1)} = 0 - 0 \cdot 0 = 0$, $y_{x,S_3,3,1}(t) = 0 - 0 + \lambda_1(1,0)(1,0) = \lambda_1$, $\lambda_1 \in \Lambda^t_{x,S_3,3}(t,T)$. Since $V_{x,S_3,3}(t) = \{(p,t) \mid t+1 \le p \le N\}$ for each $t = 1,\ldots,N-1$ and $V^t_3(T) = \{(p,T) \mid T+1 \le p \le N\}$, we have $V_{x,S_3,3}(t) \cap V^t_3(T) = \{(p,T) \mid T+1 \le p \le N\}$ for $t = T$ and $V_{x,S_3,3}(t) \cap V^t_3(T) = \varnothing$ for $t \ne T$. Then $\Lambda^t_{x,S_3,3}(T,T) = \{p \mid T+1 \le p \le N\}$ and $\Lambda^t_{x,S_3,3}(t,T) = \varnothing$, $t \ne T$. Accordingly, we have $y_{x,S_3,3,1}(T) = p$, where $T+1 \le p \le N$ and $y_{x,S_3,3,1}(t) = 0$, $t \ne T$.

Thus, at the Tth iteration, the broadcast of a data item $x(T)$ is carried out from the processor $P(T)$, in which $x(T)$ has been computed to processors $P(p)$, $T+1 \le p \le N$.

We can now write the following pseudocode of the algorithm (the rules of distribution of communication operations in a program code are described in [18]):

```
if (1 ≤ p ≤ N)
   do t = 1, max(1, p − 1)
      if (p = t = 1)     S1 : x(1) = b(1)
      if (p ≥ 2, t = 1)  S2 : x(p) = b(p)
      if (p ≥ 2)
         Broadcast: Receive x(t) from P(t)
          S3 : x(p) = x(p) − a(p, t)x(t)
      endif
   enddo
   if(p ≤ N − 1)
      Broadcast: Send x(p) to P(p + 1),..., P(N)
      endif
endif
```

Let us consider the case when $\Xi_s = \{2\}$ and $\Xi_t = \{1\}$ (the second coordinate of the multidimensional timing specifies a mapping of operations of the algorithm onto a processor row, and the first coordinate specifies the order of execution of the operations by processors).

We write the code destined for the execution of the algorithm on $N-1$ processors as follows:

```
do t = 1, N
   if (1 ≤ p ≤ t − 1)
      if (t = p = 1)     S1 : x(1) = b(1)
      if (t ≥ 2, p = 1)  S2 : x(p) = b(p)
      if (t ≥ 2)         S3 : x(p) = x(p) − a(p, t)x(t)
   endif
enddo
```

The equalities $\rho_{a_l,S_\beta,q} = \rho^s_{a_l,S_\beta,q}$ are fulfilled for all occurrences (a_l, S_β, q) except for $(x, S_3, 1)$ and $(x, S_3, 2)$ ($\rho_{x,S_3,1} = \rho_{x,S_3,2} = 1$ and $\rho^s_{x,S_3,1} = \rho^s_{x,S_3,2} = 2$). In executing the algorithm on a parallel computer, the use of elements of the first and second occurrences of the array x in the operator S_3 requires data transfer.

Since $\rho^t_{x,S_3,1} < n_3$ ($\rho^t_{x,S_3,1} = 1$ and $n_3 = 2$) and there is a true dependence generated by the occurrences $(x, S_3, 1)$ and $(x, S_3, 2)$, by Theorem 7, at the Tth iteration, the translation of the data item $x(t)$ between processors $P(\bar{d}^{(x,S_3,1)}_s(t)) = P(\eta^{(x,S_3,1,2)}t + z^{(x,S_3,1,2)}N + y_{x,S_3,1,2}(t))$ can be organized, where $\eta^{(x,S_3,1,2)}$, $z^{(x,S_3,1,2)}$, and $y_{x,S_3,1,2}(t)$ are found by formulas (12), (9), and (13) under the assumption that $\Xi_{fix} = \Xi_t = \{1\}$ and $T_{fix} = T$. We obtain $\eta^{(x,S_3,1,2)}(1\ 0)\begin{pmatrix}1\\0\end{pmatrix} = (0\ 1)\begin{pmatrix}1\\0\end{pmatrix}$, whence $\eta^{(x,S_3,1,2)} = 0$; $z^{(x,S_3,1,2)} = 0 - 0 \cdot 0 = 0$ and $y_{x,S_3,1,2}(t) = 0 - 0 \cdot 0 + \lambda_1(0,1)(0,1) = \lambda_1$, $\lambda_1 \in \Lambda^t_{x,S_3,1}(t,T)$. Since $V_{x,S_3,1}(t) = \{(t,p) \mid 1 \le p \le t-1\}$ for each $t = 1,\ldots,N$ and $V^t_3(T) = \{(T,p) \mid 1 \le p \le T-1\}$, we have $V_{x,S_3,1}(t) \cap V^t_3(T) = \{(T,p) \mid 1 \le p \le T-1\}$ for $t = T$ and $V_{x,S_3,1}(t) \cap V^t_3(T) = \varnothing$ for $t \ne T$. Then $\Lambda^t_{x,S_3,1}(T,T) = \{p \mid T+1 \le p \le N\}$ and $\Lambda^t_{x,S_3,1}(t,T) = \varnothing$, $t \ne T$. Accordingly, we have $y_{x,S_3,1,2}(T) = p$, where $1 \le p \le T-1$ and $y_{x,S_3,1,2}(t) = 0$, $t \ne T$.

Thus, at the Tth iteration, the translation of the data item $x(T)$ is carried out between processors $P(T)$, $1 \le p \le T-1$. We write the following pseudocode of the algorithm using the rules of organization of data transceiving that are described in [18]:

```
if (2 ≤ p ≤ N − 1) Receive x(p) from P(p − 1)
do t = 1, N
   if (1 ≤ p ≤ t − 1)
      if (t = p = 1)     S1 : x(1) = b(1)
      if (t ≥ 2, p = 1)  S2 : x(t) = b(t)
      if (t ≥ 2)
         if (p ≥ 2) Receive x(t) from P(p − 1)
         S3 : x(p) = x(p) − a(p, t)x(t)
         if (p ≤ N − 1) Send x(t) to P(p + 1)
      endif
   endif
enddo
```

Note that the method of organization of communications that is proposed in the present article is more easily used than the method from [18] and does not lead to redundant communications.

CONCLUSIONS

This article considers some aspects of mapping algorithms specified by sequential programs onto parallel computers with distributed memory. Problems are investigated that arise after the distribution of operations among processors and determination of the order of executing the operations by a parallel algorithm.

We summarize the obtained results as follows:

— for each element of data arrays, processors (and iterations) are specified on which (at which) it is used;

— the problem of distribution of initial data (input arrays) among processors and problem of determination of the sizes of the arrays used in each processor are investigated;

— new sufficient conditions convenient in practice are obtained for the possibility of organization of simultaneous data broadcast and translation.

In further investigations, the authors intend to apply the developed mathematical apparatus (functions using array elements) to the formalization of communication operations (only computing operations are formalized in the initial algorithm) and organization of dynamic distribution of array elements in realizing parallel algorithms.

REFERENCES

1. V. V. Voevodin and Vl. V. Voevodin, Parallel Computations [in Russian], BKhV-Peterburg, St. Petersburg (2002).
2. A. W. Lim and M. S. Lam, “An affine partitioning algorithm to maximize parallelism and minimize communication,” in: Proc. 1st ACM SIGARCH Intern. Conf. on Supercomputing (1999), pp. 228–237.
3. A. Darte and F. Vivien, “Automatic parallelization based on multi-dimensional scheduling,” Techn. Rep. 94-24, LIP, ENS-Lion (1994).
4. E. V. Adutskevich and N. A. Likhoded, “A consistent generation of pipeline parallelism and distribution of operations and data among processors,” Programmirovanie, **32**, No. 3, 54–65 (2006).
5. E. V. Adutskevich, S. V. Bakhanovich, and N. A. Likhoded, “Conditions of generation of consistent timing and distribution of operations and data among processors,” in: Proc. Intern. Sci. Conf. “Supercomputer systems and their application (SSA’2004),” Minsk (2004), pp. 160–164.
6. W. Shang and J. A. B. Fortes, “Independent partitioning of algorithms with uniform dependencies,” IEEE Trans. on Computers, **41**, No. 2, 190–206 (1992).

7. D. Bau, I. Kodukula, V. Kotluar, K. Pingali, and P. Stodghill, “Solving alignment using elementary linear algebra,” in: Proc. 7th Workshop on Languages and Compilers for Parallel Computing, Springer (1994), pp. 46–60.
8. A. W. Lim and M. S. Lam, “Communication-free parallelization via affine transformation,” in: Proc. 7th Workshop on Languages and Compilers for Parallel Computing, Springer (1994), pp. 92–106.
9. N. A. Likhoded, “Mapping of affine loop nests onto independent processors,” Cybernetics and Systems Analysis, Vol. 39, No. 3, 459–466 (2003).
10. R. Schreiber and J. J. Dongarra, “Automatic Blocking of Nested Loops,” Tech. Rep. 90-38, The University of Tennessee (1995).
11. M. Dion, T. Risset, and Y. Robert, “Resource-constrained scheduling of partitioned algorithms on processors arrays,” Integration, the VLSI Journal, **20**, No. 2, 139–159 (1996).
12. A. V. Frolov, “Determination and use of oriented sections of real-word graphs of algorithms,” Programmirovanie, No. 4, 71–80 (1997).
13. A. W. Lim, S.-W. Liao, and M. S. Lam, “Blocking and array contraction across arbitrary nested loops using affine partitioning,” in: Proc. ACM SIGPLAN Symposium on Principles and Practice of Programming Languages (2001), pp. 103–112.
14. A. Darte and Y. Robert, “On the alignment problem,” Parallel Processing Letters, **4**, No. 3, 259–270 (1994).
15. M. Dion and Y. Robert, “Mapping affine loop nests,” Parallel Computing, **22**, No. 10, 1373–1397 (1996).
16. E. V. Adutskevich and N. A. Likhoded, “Mapping affine loop nests: Solving of the alignment and scheduling problems,” in: Proc. 7th Int. Conf. on Parallel Computing Technologies (PaCT-2003), (Nizhni Novgorod, Russia, Sept. 15–19 (2003)), Springer, Berlin (2003), pp. 1–9.
17. E. V. Adutskevich and S. V. Bakhanovich, “Adaptation of algorithms for realization on distributed memory systems: Space-time localization and data distribution,” in: Proc. Intern. Sci. Conf. “Supercomputer systems and their application” (SSA’2004), Minsk, Republic of Belarus (2004), pp. 165–169.
18. E. V. Adutskevich and N. A. Likhoded, “Optimization of Data Exchange in Parallel Computers with Distributed Memory,” Cybernetics and Systems Analysis, Vol. 42, No. 2, 298–303 (2006).
19. M. Dion, C. Randriamaro, and Y. Robert, “Compiling affine nested loops: How to optimize the residual communications after the alignment phase?” J. Parallel and Distrib. Computing, **30**, No. 2, 176–187 (1996).
20. J. Garcia, E. Ayguade, and J. Labarta, “A framework for integrating data alignment, distribution, and redistribution in distributed memory multiprocessors,” in: IEEE Transactions on Parallel and Distributed Systems, **12**, No. 4, 416–430 (2001).
21. L. Pan, J. Xue, M. K. Lai, M. B. Dillencourt, and L. F. Bic, “Toward automatic data distribution for migrating computations,” in: Int. Conf. on Parallel Processing, Xian, China (2007).
22. U. Banerjee, “An introduction to a formal theory of dependence analysis,” J. Supercomput., No. 2, 133–149 (1988).
23. P. Feautrier, “Some efficient solutions to the affine scheduling problem. Part 1,” Intern. Journ. of Parallel Programming, **21**, No. 5, 313–348 (1992).
24. U. Bondhugula, M. Baskaran, S. Krishnamoorthy, J. Ramanujam, A. Rountev, and P. Sadayappan, “Automatic transformations for communication-minimized parallelization and locality optimization in the polyhedral model,” Lecture Notes in Computer Science, **4959**, 132–146 (2008).
25. V. V. Voevodin, “Information structure of sequential programs,” Russ. J. of Numerical Analysis and Math. Modeling, **10**, No. 3, 279–286 (1995).
26. N. A. Likhoded, “Functions for data distribution among processors and iterations of a parallel algorithm,” Dokl. NAN Belarusi, **51**, No. 4, 19–24 (2007).