*Bull. Nov. Comp. Center*, Comp. Science, 19 (2003), 89–105
© 2003 NCC Publisher

# The computing system "Potok-3": The parallelization experience of a multi-program complex. Part 1. The parallelization strategy*

G.A. Tarnavskiy, V.D. Korneev

Technological aspects of parallelization of the computing system "Potok-3", intended for some numerical modeling of problems of aerodynamics and physical gas dynamics are considered. The methods and problems of global parallelization of a multi-program complex by major parameters, as well as C-, L- and V-types of the parallelization procedures of the basic iterative kernel of system for local acceleration of operations are proposed and investigated.

## 1. Introduction

In [1, 2], the general parallelization strategy of the large computer complex "Potok-3" intended for computer-aided modeling of tasks of aerodynamics and physical gas dynamics was developed. In [3,4], various parallelization schemes of multivariate scalar sweeping operations were considered, which constitute the basic algorithm of the numerical solution of a system of the Navier–Stokes differential equations of the dynamics of a viscous heat-conducting gas. Multi-purpose computing experiments on defining effective parallelization ways and their realization on various multiprocessor computers were carried out. In [5], some aspects of arranging calculations of the tasks in a subject domain, which are connected with providing a numerical method with the initial data were studied. The problems of outputting the obtained information and its layout on a long-term were discussed to some extent. The present work represents both the strategy and the experience gained in parallelization of the computing system "Potok-3" and the comprehensive analysis of various parallelization types applied.

## 2. Global parallelization

From the general parallelization types, considered in [1, 2], which are connected with the physical-mathematical, geometrical and technological de-

*Supported by the Russian Foundation for Basic Researches under Grants 00-07-90297, 01-01-00781, and 02-01-00097.

composition of a complete task in a number of parallel subtasks at the first stage of the reorganization of the uniprocessor multiple program complex "Potok-2" into the parallel computing system "Potok-3" oriented at the multiprocessor calculations with its possible wide usage, some ways of parallelization were applied. One of them is the global parallelization of the system according to the major input parameters.

Let us consider the physical essence of this method. The complex multiply program provides the solution of the task of flowing around an aircraft with a flow of gas within the wide range of determining parameters. Here we mean only the physical parameters, determining the basic modes of a flow As far as algorithmic parameters are concerned, they are not discussed here though they can be associated with the proposed strategy. The major among the physical parameters are: the shape of a surface of a body, thermal condition on it, height and speed of a flight, thermodynamic characteristics of gas. Let us consider two physical parameters, for example, the height $H$ and the speed $V$. As a rule, for studying the processes of flowing around it is necessary to carry out a significant number of calculations with a variation both of $H$ and $V$. Such cycles of calculations can be organized in simultaneous start up of the multiple complex program with parallelization "according to the global parameter", whose scheme is shown in Figure 1.

Here a number of parallelization levels, forming a certain one-coherent graph is shown: a tree-type branching non-crossed communications coming out from a uniform root, which is the input to the multiple program complex in question.

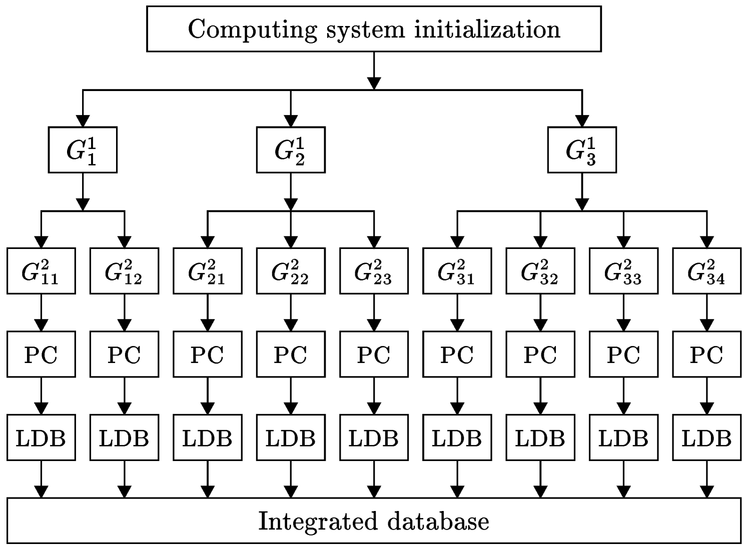

**Figure 1.** Global parallelization of the multiprogram complex "Potok-3"

Each level is represented by the symbol

$$G^m_{g_1,\dots,g_m}: \ m \in [1, M]; \ g_1 \in [1, g_{1F}], \ \dots, \ g_m \in [1, g_{mF}], \tag{1}$$

where the following designations (all indices are integer) are introduced:

$m$ is the number of the horizontal level graph (calculated from the root), or, in other words, a number of the type of the global parameter with which the parallelization is made, for example, the height of a flight, the speed of a flight, etc.;

$M$ is the parallelization depth (the number of the horizontal levels in the graph), i.e., the length of the list of physical-mathematical parameters of a task, with which the parallelization is made;

$g_{1F}$ is the length of the list (the number of variants) of parameter values of the first level;

$g_1$ is the current number in this list;

$g_{mF}$ is the length of the list (the number of variants) of parameter values in one subgroup of $m$-level;

$g_m$ is the current number in this list.

Let us note that in the general case, the length of lists in all subgroups of $m$-level can be various and defined by belonging to one or another subgroup of the previous level, that can be recurrently be presented by the formula:

$$g_{mF} = g_{mF}(g_{m-1}), \ m = \overline{2, M}. \tag{2}$$

Such a situation is shown in Figure 1, illustrating the two-level global parallelization. At the first $G^1$-level the length of the list of parameters $g_{1F} = 3$, i.e., the solution of the task with three values of the height of flight $H_1$, $H_2$, $H_3$ is performed. At the second $G^2$-level the lengths of the lists of parameters are defined by belonging to one or another subgroup (branches of the graph): for first subgroup – the length of the list (the number of variants) $g_{1,2F} = 2$, for the second – $g_{2,2F} = 3$, for the third – $g_{3,2F} = 4$. In the terms explaining Figure 1 this means the order for organization of calculation in the variant with the height of the flight $H = H_1$, two subvariants with values of the speed of flight $V = V_{11}$ and $V_{12}$, in the variant with $H = H_2$ – three subvariants with the values $V = V_{21}$, $V_{22}$, $V_{23}$, in the variant with $H = H_3$ – four subvariants with the values $V = V_{31}$, $V_{32}$, $V_{33}$, $V_{34}$.

After preparation of files of different variants of input parameters with "individual" values $G^1_{g_1}$, $G^2_{g_1,g_2}$ each file – the final the structure, most distant from the root of a graph – sends data to the multiple program complex PC and initiates its processor system. Let us emphasize once again that during the execution of a task as a whole there is no data exchange among the processor subgroups $G_2$, and, especially, between processor blocks $G_1$. This is the essence of the given parallelization of type – parallelization according to the global parameters for the multiple program complex, i.e., the major input parameters (scalars, as a rule), defining the physical-mathematical sense of a task.

Generally speaking, the processor time spent on the execution of operations of solution of a single sub-task (independent process) essentially depends on the used initial data: it can appear reasonable first to finish a process, for example, $G^n_{11}$, and then to accept its results as initial conditions for the process $G^m_{12}$ (for more detail see [5]). However, this question is beyond the present work, where complete independence of all the processes $G^m_{g_1,g_2}$ is assumed.

In this case the parallelization of the SIMD-type takes place: a single instruction (in Figure 1), the whole program complex is designated as PC; the multitude of data (the list $G^m_{g_1,g_2}$) and without data exchange among the processes.

The minimum number of processors $P_G$ required for such parallelization is defined by the sum of lengths of the lists of all subgroups of a level most distant from the root of a graph. Figure 1 presents a special case of two levels:

$$P_G = \sum_{g_1=1}^{3} g_{2F}(g_1). \tag{3}$$

In general case, the form of formula (3) becomes more complicated:

$$P_G = \sum_{g_{m-1}=1}^{g_{(m-1)F}} g_{MF}(g_{m-1}) \tag{4}$$

and it is required to use the recurrence formula (2).

From the point of view of the MPI-systems, the parallelization of the given type, i.e., the parallelization according to global parameters, has not caused essential difficulties, there also were no problems and conflicts in debugging. However, in the process of development a number of technically complex problems, connected with the unified input to the system, configuration of processor space as well as organization of the automated collection of all the information from all involved processors in the integrated databank of the computing system was revealed.

Briefly we will consider problems of organization of input to the system and its branching according to parameters. From more than 50 global input parameters of the computing system "Potok-3", defining the physical sense (height and speed of flight, thermal mode of a body and temperature of its surface, etc.), the mathematical sense ("geometrical" type of a task – flat, wasp symmetric, spatial, the type of a coordinate system –

Cartesian, spherical, etc.), the algorithmic sense (topology of a difference grid-dimension, factors of condensation of knots, the iterative parameters, etc.) and the operating conditions (input/output control the information) 12 "basic" parameters were selected most suitable for parallelization.

The maximum parallelization depth $M$, i.e., the number of parallelization levels (see Figure 1 – the levels $G^1$, $G^2$) for the first stage was chosen equal to 4 (debugging was carried out, basically for $M = 2$). There was constructed a parallelization graph, allowing one to place in arbitrary oder the basic parameters at the levels. For the above-mentioned examples ($G^1 \equiv H$), first the list of heights of flights was defined, for which it is necessary to obtain a solution, and then for each element of this list $H_{g1}$, the list of speeds of flights ($G^2 \equiv V$) was defined, in this case the lists $V_{g1,g2}$ can be different for each $H_{g1}$ as in their length (number of variants), and in their concrete values. The inverse parallelization ($G^1 \equiv V$, $G^2 \equiv H$) is also possible: the list of speeds of flights is defined, for which it is necessary to obtain a solution, and then for each element of this list $V_{g1}$ – the list of heights of flights $H_{g1,g2}$.

It is necessary to note that the steady functioning of this graph in the context of a reasonable statement of a task and starting the system by the experts in the field of computing aerodynamics, not involved in the development of the given multiple program complex, was provided only up to the level $M = 2$. This, apparently, should be realized within the system intended for a wide range of the users, as the use of deeper levels $M = 3$ or $M = 4$ causes essential difficulties.

However, the strategy and the experience gained in the cause of development and of operation of parallelization by global parameters employing a multilevel branching graph, can be useful when creating the parallel computing systems in other areas of knowledge.

Another problem of this type of parallelization was the problem of arranging the flows of the output information. Even a separate application task represents a large file of data which require structuring, for example "heading" (a list metadata, i.e., parameters of the resolved task, giving its complete description and allowing its unambiguous identification and distinguishing it from other ones. This is especially important with realization not of a single calculation, but for the multiple complex study of any scientific or application problem), "volume" (a diagram or a table), "chapter" (a table of various physical notions – density, pressure, the temperature in the field of currents, aerodynamic characteristics of an aircraft, etc.), "page" (a part of a "chapter", representing a physical function in any subdomain of parameters), etc., up to a "line" or a "column" (see any tables on gas dynamics, for example, those classical [6]). In the sequel such a structuring should provide rather a simple access to the obtained information and the possibility to compare data from various "volumes".

Basically any here can be used already fulfilled SUBD of a type Oracle, Access or Paradox, however this question still requires the decision. In represented development the special system of marks of the input information used with record of files in a temporary databank of carried out account, containing main and auxiliary keys was used.

The main key – individual (not repeating) identifier of carried out calculate was made of seven symbols: one letters and six figures. The letters H, B, N, F (Head, Body, Nearwake, Farwake) are the brief names of databases, in which the results of calculates in various geometrical segments of a task (more in detail see [1]) were placed, first three figures showed virtual number of the processor on which the calculate of variant was carried out, last three figures provided through numbering of variant in H-, B-, N- or F-database component an integrated databank. With such structure of names some restrictions take place: the maximum used number of processors should not exceed 999, and the number of records in one volume of a database should not exceed 999. Besides on the moment of execution of the task the additional channels of communications marked with the letters C, L, S, G, appropriate will be formed also: 1) to the data channel for exchange between segments "chemistry", "combustion", "turbulence" and "gas dynamics" of the computer complex; 2) to the data link from local database of current calculation in any module of a complex on its inquiry; 3) to S-channel of input of the standard data "a Near trace" (NearWake), G-channel of record of the data for the subsequent graphic processing.

With work with bases the names of output information files – channels were made of eight symbols, first of which – the letters W, R, U or P – meant a type of a flow of the data (Write – record, Read – reading, Universal – record and reading, Print – file of a print data), others seven coincided with symbols of a main key. So, for example, the symbol WH001004 means, that the channel is intended for record in a database of results of calculate of a flow of a head part of a body received on the processor with virtual number 001, and this record is placed in database under number 004.

Auxiliary keys – metadata of the carried, supplement the main key out calculate (main input parameters determining physical sense of a task, such as already indicated above height H and speed V flights etc.) is higher. The main key unequivocally identifies record in DB, as against auxiliary keys, as in DB, naturally, some records with the same number, for example, H can contain.

Generally speaking, this system (distant) far from end, has appeared bulky and not quite is convenient with operation, required attention of the manager of system and high discipline of calculators, as otherwise in DB there were records with identical names. It caused of creation temporary DB, their viewing by the manager and only then carry of records in DB of a long-term storage.

In the whole problem of fixation of the information got on multiprocessor computing systems, nowadays practically is not given of due attention, as use MVS by calculators – applauders is frequently carried out in a mode of a type "parallelization the program – has started a task – has looked results – and all (at the best has written down numbers on CD)". Similar the problems of visualization (graphic representation) solutions got with global parallelization on input parameters are also, however this question too great and lies beyond the framework of the present work.

## 3. Local parallelization

The local parallelization, according to classification described in [1], referred to the technological type of parallelization "of the lower level", is intended for acceleration of processes of calculation in subroutines which constitute the basic algorithmic kernel PC (see Figure 1) of the multiple complex program "Potok-3" [3]:

$$\{\mathrm{PC}\} = \sum_{n=1}^{N}\sum_{k=1}^{K}\sum_{i=1}^{I}\sum_{j=1}^{J} k\text{-OPERATIONS}(i,j;n), \tag{5}$$

$$k\text{-OPERATIONS}(i,j;n) \Rightarrow F_{ij}^{kn}. \tag{6}$$

The symbolical record (5), (6) means the following. From result of functioning of the $k$-th of a procedure of the algorithm, realized as separate subroutine

$$\text{SUBROUTINE}(k, \text{input parameters}, \text{output parameters}), \quad k \in [1, K]. \tag{7}$$

At the $n$-th iterative layer (the time step $n$)

$$t_n = \varphi_n(n), \;\; n \in [1, N]. \tag{8}$$

A two-dimensional array $F_{ij}^{kn}$ – the rectangular table of numbers (for brevity, we consider a two-dimensional problem of subject domain – a flat and an axially symmetric), where $i$ and $j$ denote, respectively, along the coordinates $(x, y)$ the nodes of a calculation grid, which discretizes the continuous differential problem:

$$y_i = \varphi_y(i), \;\; i \in [1, I], \tag{9}$$

$$x_j = \varphi_x(j), \;\; j \in [1, J]. \tag{10}$$

With the use of a uniform spatial-temporary pattern (8)–(10) we have $t_n = \tau n$, $y_i = i\Delta y$, $x_j = j\Delta x$, where $\tau$ is a time step, $\Delta y$ and $\Delta x$ are steps along the coordinates $x$ and $y$.

The subroutines $k$-OPERATIONS$(i, j; n)$ are repeatedly executed in the principal iterative process of the solution to a complex system of differential equations and contain the cyclic operators for sorting out indices inside themselves. Inside some cycles, a significant number of arithmetic operations is contained.

To speed-up the execution of all $k$-procedures of the principal iterative block of the multiple program complex "Potok-3" C-, L-, V-, W-types of parallelization, which correspond to the parallelization of operations "by columns", "by lines", "by a vector", "without parallelization" ("Column", "Line", "Vector", "Without"-parallelization) were developed and used.

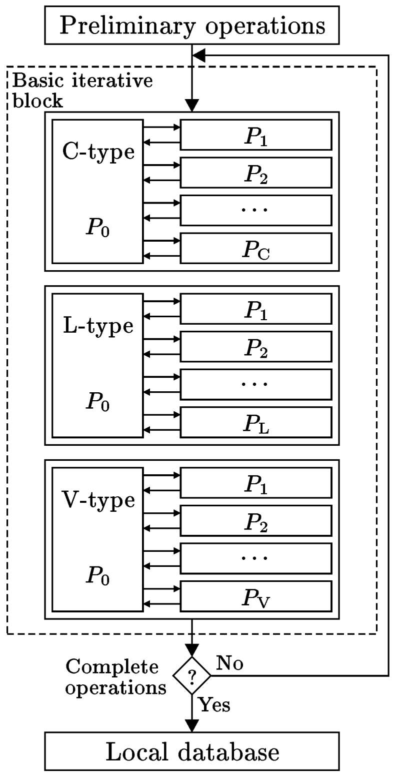

**Figure 2.** Local parallelization of the basic algorithmic kernel of the multiprogram complex "Potok-3"

The pioneering work on parallelization and, in particular, the types of parallelization considered above, can be found in [9].

A conditional scheme of parallelization of the basic computing kernel of the program complex "Potok-3" (designated in Figure 1 by the symbol PC) is presented in Figure 2. Each of the discussed type of parallelization is oriented to an individual algorithmic and program (realizing them) features of each $k$-procedure. The purpose of the above-said is to gain the maximum benefit in acceleration of computing process.

Let us note that the number and the order of execution of C-, L-, V-, and W-types of parallelization of procedures is defined by a certain algorithm and a concrete type of the subroutines, realizing it.

Let us dwell on the essence of each type of parallelization.

An implicit finite difference algorithm of a principle iterative kernel of "Potok-3" is based on splitting of a system of the Navier-Stokes differential equations, used for the modeling of non-stationary tasks of aerodynamics and physical gas dynamics

$$\frac{\partial F}{\partial t} = W\left(F, \frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}, \frac{\partial^2 F}{\partial x^2}, \frac{\partial^2 F}{\partial y^2}, \frac{\partial^2 F}{\partial x \partial y}\right), \tag{11}$$

$$F = F(t, x, y) \tag{12}$$

along the coordinate directions $x$ and $y$

$$W = W_1 + W_2 + W_3. \tag{13}$$

Splitting (13) is done so that to divide whenever possible the terms, containing the coordinate derivatives:

$$W_1 = W_1\left(F, \frac{\partial F}{\partial x}, \frac{\partial^2 F}{\partial x^2}\right), \tag{14}$$

$$W_2 = W_2\left(F, \frac{\partial F}{\partial y}, \frac{\partial^2 F}{\partial y^2}\right), \tag{15}$$

$$W_3 = W_3\left(F, \frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}, \frac{\partial^2 F}{\partial x^2}, \frac{\partial^2 F}{\partial y^2}, \frac{\partial^2 F}{\partial x \partial y}\right), \tag{16}$$

where $W_1$ contains a derivative only with respect to $x$, $W_2$ – only with respect to $y$, and $W_2$ is some "coordinate–non-splitting" part.

The blocks of system of the equations $W_1$ and $W_2$ are further divided according physical processes ("dynamics" and "dissipation"):

$$W_1 = W_{11} + W_{12} \tag{17}$$

$$W_2 = W_{21} + W_{22} \tag{18}$$

Thus, the dependencies $W_{11}, W_{12}, W_{21}, W_{22}$ from a vector of the desired functions $F$ written down in the matrix form,

$$W_{11} = A_{11}F, \tag{19}$$

$$W_{12} = A_{12}F, \tag{20}$$

$$W_{21} = A_{21}F, \tag{21}$$

$$W_{22} = A_{22}F, \tag{22}$$

are characterized by the following properties: the matrices $A_{11}$ and $A_{21}$ are diagonal, and the matrices $A_{12}$ and $A_{22}$ have an essential filling with nonzero entries (even without diagonal prevalence).

The concrete form (11)–(22) for further statement is inessential and not given here. (For more detail see [7, 8].)

The given structure of the splitting method (11)–(22) determines in the general strategy of parallelization of "Potok-3" tactics of local (technological according to terminology [1]) parallelization of the subroutines, realizing the stages of numerical algorithm (11)–(18) in view of their peculiarities (19)–(22).

## 4. C-type parallelization

This type of parallelization for a part of the computing algorithm is focused on acceleration of calculations in the subroutines realizing computation of the terms of equations of the form of (15), i.e., the equations containing a derivative, finite difference operators, to be exact only along $y$-coordinate direction. In the grid space of operations (6) the operands of the given a type, can be written down as for simplicity the index $k$ is removed

$$F_{ij}^{n+\nu} = \mathrm{OPERATIONS}(F_{ij}^n, F_{i\pm1,j}^n, F_{i\pm2,j}^n). \tag{23}$$

Formula (23) means that by the known of value $F$ at $n$-layer the values at $(n+\nu)$ layer are calculated. The symbol $\nu$ means some "fractional" (for example, $\nu = \frac{1}{6}, \frac{2}{6}$, etc.) stage of transition to $(n+1)$ iterative layer, however in the present work a concrete definition and specification of the algorithm are reasonable (for more detail see [7], as this is of minor importance in what follows.

In such operations as (23), of primary importance is the circumstance that in calculation of the array $F_{ij}^{n+v}$ the elements of the arrays $F_{ij}^n$ only with a variation of the first index $(i, i\pm1, i\pm2)$ are used: i.e., the coupling operations with respect to the first index $(i)$ and their independence of the second index $(j)$ take place.

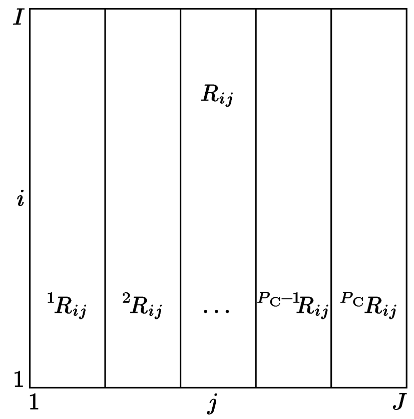

**Figure 3.** C-type parallelization of calculations

This means the possibility to realize operations simultaneously (in parallel) on all the columns of a computational grid (Figure 3), illustrating the essence of C-type ("Column") parallelization. The common grid space $R$

$$\{R_{ij}\} = i \otimes j \tag{24}$$

in the domain (9), (10) can be divided to $P_C$ subspaces ("vertical columns"), in each of which operations (23) are independent processes and can be carried out in parallel:

$$F_{ij}^{n+\nu} = \sum_{p=1}^{P_C} {}^pF_{ij}^{n+\nu}. \tag{25}$$

In this case, the number of independent processes $P_C$ should satisfy the condition

$$P_C \le J. \tag{26}$$

Thus, to speed-up the calculations, $P_C$ computers (processors) can be used in (23).

The sizes of columns (see Figure 3) can be different, since generally, at $P_C < J$ the ratio $J/P_C$ is not an integer, but, obviously, the splitting $R$ to almost identical columns is optimal, because the number of operations in columns is almost identical, and, respectively, the waiting time of simultaneous termination of all the processes, i.e., operations (23) and (25) in $R$ are minimized.

The realization of this type of parallelization in "Potok-3" was carried out as follows. To the user's order a special processor – dispatcher $P_0$ makes segmentation of $R$-space to $P_{\mathrm{C}}$ subspaces, according to the number of processors $P_{\mathrm{C}}$ i.e., divides all the arrays $F_{ij}$ involved in (23) to $P_{\mathrm{C}}$ subarrays ${}^{p}F_{ij}$ and sends them to $P_{\mathrm{C}}$ computers, which perform out operation (23) with them and then return them to the processor-dispatcher, which makes up the complete array $F_{ij}^{n+\nu}$. using formula (25).

## 5. L-type parallelization

This type of parallelization for a part of the computing algorithm, is focused on computation acceleration in the subroutines realizing the calculation of the terms of the form of (14), i.e., the equations containing a derivative (finite difference operators to be exact) along $x$-coordinate direction. In the grid space of operations (6) this type of operands can be written down as (the index $k$ for simplicity is removed)

$$F_{ij}^{n+\mu} = \mathrm{OPERATIONS}(F_{ij}^{n}, F_{i,j\pm1}^{n}, F_{i,j\pm2}^{n}). \tag{27}$$

As previously, formula (27) means, that by the known values of $F$ at $n$-layer the values at $(n+\mu)$ layer are calculated. So $\mu$ as previously, means some intermediate stage of transition at $(n+1)$ iterative layer. In operations (27) as opposed to operations (23), the coupling calculations with respect to the second ($j$) and their independence of the first index ($i$) take place. This means the possibility to realize operations simultaneously (in parallel) on all the lines (strings) of a computational grid, which is shown in Figure 4, illustrating the essence of L-type ("Line") parallelization. The common grid $R$-space (24) in the domain (9)–(10) can be divided to subspaces ("horizontal strips"), in each of which operations (27) are independent and can be executed in parallel:

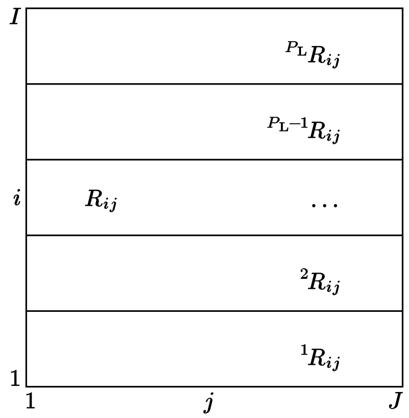

**Figure 4.** L-type parallelization of calculations

$$F_{ij}^{n+\mu} = \sum_{p=1}^{P_{\mathrm{L}}} {}^{p}F_{ij}^{n+\mu}. \tag{28}$$

The number of independent processes $P_{\mathrm{L}}$ should, obviously, satisfy the condition

$$P_{\mathrm{L}} \leq I \tag{29}$$

Thus, for acceleration of calculations in (27) $P_{\mathrm{L}}$ processors (computers) can be used.

The realization of this type of parallelization in "Potok-3" similar to realizations of C-type. To the user's order a special processor-dispatcher $P_0$ carries out distribution of $R$-space on $P_{\mathrm{L}}$ subspaces, according to the number of processors $P_{\mathrm{L}}$ i.e., divides all the arrays $F_{ij}$ involved in (27) to $P_{\mathrm{L}}$ subarrays ${}^{p}F_{ij}$ and sends them to $P_{\mathrm{L}}$ computers, which perform operations with them (27) and then return them to dispatcher, which makes up the complete array $F_{ij}^{n+\mu}$ using formula (28).

Despite the similarity ("vertical columns" and "horizontal strips"), C- and L-types of parallelization essentially differ in their computer realization on multiprocessor systems. Let us consider more specifically the functioning of the processor-dispatcher, (Figure 5), distributing elements $a_{ij}$ ($I = 4$, $J = 4$) of the complete array among the calculating processors ($P_{\mathrm{C}} = P_{\mathrm{L}} = 3$).

In Figure 5a, a standard mathematical representation of the array $a_{ij}$ with distribution of its elements along the lines (variation $j$) and along the columns (variation $i$) is shown. In the computer representation. a two-dimensional array $a_{ij}$ is mapped onto some address space, which in in terms of geometry can be treated as one-dimensional direct line (Figure 5b) with a consecutive arrangement of elements of the array $a_{ij}$. In the beginning of this line, the element $a_{11}$ is located. Further arrangement of elements depends on the programming language: in Fortran, the two-dimensional array $a_{ij}$ is represented as one-dimensional array with the help of a variation of the first index ($i$) and the fixed second index ($j$); then $j$ is changed for $j + 1$ and again, the variation $i$ takes place in all

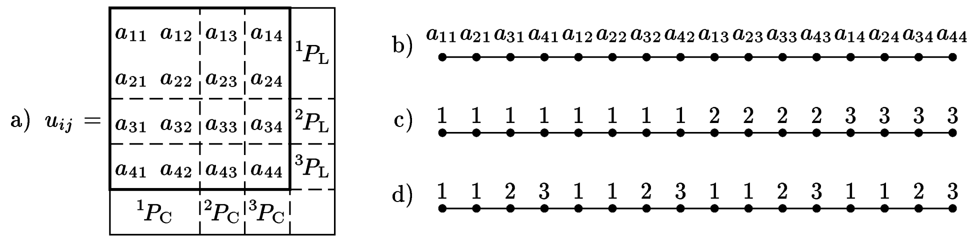
a) $u_{ij} =$
$a_{11}$ $a_{12}$ $a_{13}$ $a_{14}$
$a_{21}$ $a_{22}$ $a_{23}$ $a_{24}$
$a_{31}$ $a_{32}$ $a_{33}$ $a_{34}$
$a_{41}$ $a_{42}$ $a_{43}$ $a_{44}$
${}^{1}P_{\mathrm{L}}$
${}^{2}P_{\mathrm{L}}$
${}^{3}P_{\mathrm{L}}$
${}^{1}P_{\mathrm{C}}$ ${}^{2}P_{\mathrm{C}}$ ${}^{3}P_{\mathrm{C}}$
b) $a_{11}\,a_{21}\,a_{31}\,a_{41}\,a_{12}\,a_{22}\,a_{32}\,a_{42}\,a_{13}\,a_{23}\,a_{33}\,a_{43}\,a_{14}\,a_{24}\,a_{34}\,a_{44}$
c) 1 1 1 1 1 1 1 1 2 2 2 2 3 3 3 3
d) 1 1 2 3 1 1 2 3 1 1 2 3 1 1 2 3

**Figure 5.** Reconfiguration of processor space: a) two-dimensional representation of the array $a_{ij}$, $i \in [1,4]$; $j \in [1,4]$; b) one-dimensional Fortran-development two-dimensional of the array $a_{ij}$; c) distribution of elements of the array $a_{ij}$ with C-type parallelization on three processors; d) distribution of elements of the array $a_{ij}$ with L-type parallelization on three processors

the range of changes, etc.: $a_{11}$, $a_{21}$, $a_{31}$, ... (see Figure 5b). Thus, the mapping is carried out "in columns" (in C language – vice versa). With C-type parallelization ("in columns") the processor-dispatcher when dividing the array $a_{ij}$ to $P_{\mathrm{C}}$ subarrays (see Figure 5c, with $P_{\mathrm{C}} = 3$) defines the arrangement of the appropriate elements $a_{ij}$ in the address space (see Figure 5b).

In the common address space the addresses of elements of these subarrays are located "in dense groups", thus forming "simply-connected subsets", which essentially minimizes computer costs necessary for mapping the array $a_{ij}$ onto subarrays ${}^{p}a_{ij}$, $p \in [1, 3]$.

With L-type of parallelization ("in lines") the processor-dispatcher when dividing the array $a_{ij}$ to $P_{\mathrm{L}}$ subarrays (see Figure 5d, with $P_{\mathrm{L}} = 3$) defines the arrangement of the appropriate elements $a_{ij}$ in the address space (see Figure 5b). In the common address, space the addresses of the elements ${}^{p}a_{ij}$ are located in another way, namely, in isolated groups, forming some "multiply-connected subsets". The distances between the addresses of the first elements of these groups are equal in the index space to the values of height (width) of "a strip" (see Figure 4).

As is shown in Figure 4, we have "the width of a strip" 2, or 1, since the average value of width is specified from:

$$S = I/P_{\mathrm{L}} = 4/3. \tag{30}$$

Thus, elements 1, 2, and 3 of the subarrays will be located in groups consisting of 2 elements for the front strip and separately for the second and the third strips, the intervals between elements of the same group, for example, the first, being equal to

$$\frac{J(P_{\mathrm{L}} - 1)}{P_{\mathrm{L}}}. \tag{31}$$

Let us note that in the cases, when $I$ is not exactly divided on $P_{\mathrm{L}}$, and strips of a different width are used, formula (31) becomes essentially complicated.

In connection with aforesaid it is possible to state, that in practice L-type parallelization is much more troublesome, than C-type, because it requires the increased attention of the programmer and much higher computer costs.

## 6. V-type parallelization

This type of parallelization of a part of a computing algorithm is guided (focused) on acceleration of computation in the subroutines realizing the calculation of the terms of equations (19) and (21) so that it is possible to supplement C- and L- types of parallelization. V-type ("Vector") parallelization is determined by diagonal (property) of matrices $A_{11}$ and $A_{21}$ in (19) and (21), which results in independence in some subroutines of the complex "Potok-3" calculations (12) component of the vector $F = (F_1, F_2, F_3, F_4)$. In a three-dimensional problem, the vector $F$ consist of 5 components (see [7] for details).

This type of parallelization is rather simple in realization: the processor-dispatcher sends the necessary data to four processor-executors, which after realization of necessary operations return the new values of the arrays $F_{1_{ij}}^{n+n}$, $F_{2_{ij}}^{n+n}$, $F_{3_{ij}}^{n+n}$, $F_{4_{ij}}^{n+n}$ (here, as well as above, the symbol $u$ designates some "fractional" a step of the progress of the solution in the general algorithm from $F^n$ to $F^{n+1}$).

Now for the sake of generality we designate the number of processors needed for this type of parallelization, as $P_{\mathrm{V}}$ (plus the processor-dispatcher, though in this case the process of arranging data, their broadcasting and subsequent receiving is not very "expensive" speaking about computer costs, and the process-dispatcher can be given "to a charge" to operate with one of components of the vector $F$, reducing by one the number of processors-executors). Let us consider briefly some technical questions connected with the ratio of the values $P_{\mathrm{C}}$, $P_{\mathrm{L}}$ and $P_{\mathrm{V}}$. We can see that the first two can accept any positive values, but the value $P_{\mathrm{V}}$ is rigidly determined: $P_M = 4$ for a two-dimensional and $P_{\mathrm{V}} = 5$ for a three-dimensional problem.

If we choose $P_{\mathrm{C}}$ or $P_{\mathrm{L}}$ to be large enough (this choice is determined only by the access of the calculator to one or another multiprocessor system with various specifications), then at the moment of execution of the subroutines, according to V-type, the number of processors to be in the expectation mode, i.e., stand idle is $(\max(P_{\mathrm{C}}, P_{\mathrm{L}}) - P_{\mathrm{V}})$, which is not quite optimal.

Note that a similar problem also exists for the ratio of the values $P_{\mathrm{C}}$ and $P_{\mathrm{L}}$, which can be solved by a simple choice of $P_C = P_{\mathrm{L}}$. Generally speaking, this question is more complicated, than it seems at a glance, and is open to question, as there are strong contradictions between the requirements to efficiency of a problem and the optimality of using COMPUTER resources. No doubt that a scientific problem is to be effectively solved with minimum computer costs, but if there is no optimum between these poles, we should prefer the effective solution with really "unproductive" use of the MBC, than rather "efficient" use of the latter and obtaining the inefficient solution.

Coming back to the problem in question, it is possible to determine some way of optimization of the ratio $(P_{\mathrm{C}}, P_{\mathrm{L}})$ and $P_{\mathrm{V}}$. The procedures admitting V-type parallelization in algorithms of "Potok-3" also admit either C-type, or L-type parallelization. In this connection it is possible to do the following: either to refuse from V-type, or, in addition, to a V-type to apply C-type with the value $P \leq P_{\mathrm{C}}$, such that $P_{\mathrm{V}} P = P_{\mathrm{C}}$ be fulfilled.

## 7. W-type parallelization

In the principal computing kernel of the multi program complex "Potok-3", which is calculated in an iterative cycle, there exists a unique subroutine, carrying out operations (16), parallelization of calculations which in is inconvenient or inexpedient according to all the considered C-, L- and V-types. For generality of terminology we conditionally call this type as a W-type ("Without" – without parallelization).

As this subroutine is distinguished for the expenses of calculations (about 20% admit all the subroutines (5)), it is necessary to discuss theoretical and practical potentialities of its parallelization. It should be emphasized that in this section we speak about the ways of local acceleration of calculations for (technological parallelization, see [1]), i.e., operations in the subroutines. The wide spread large-scale geometrical parallelization, consisting in decomposition of complete computational domain to a series of subdomains, in which the same operations with subsequent sewing of the solutions are made, is beyond the scope of this paper (see [1, 2]).

The main obstacle complicating the use of C- and L-types parallelization, is employing in (16) of mixed derivatives $\frac{\partial^2 F}{\partial x \partial y}$, whose discrete analogue on a finite difference grid results in the coupling of values in all its nodes:

$$F_{ij}^{n+n} = \text{OPERATIONS}(F_{ij}^n, F_{i\pm1,j}^n, F_{i\pm2,j}^n, F_{i,j\pm1}^n, F_{i,j\pm2}^n). \tag{32}$$

Generally speaking, this difficulty can be overcome, applying in parallelization of (32) the segmentation of the computational domain:

$$\{R\}_{ij}, \;\; i \in [1, I], \;\; j \in [1, J] \tag{33}$$

to the subdomains $P$ "with overlapping"

$$\{R\}_{ij}^p : i \in [I_1^p - 2, I_2^p + 2], \;\; j \in [J_1^p - 2, J_2^p + 2], \;\; p \in [1, P]. \tag{34}$$

The calculations are carried out in their central part

$$\{R\}_{ij}^p : i \in [I_1^p, I_2^p], \;\; j \in [J_1^p, J_2^p], \tag{35}$$

and the value $F$ along the boundaries along the perimeters

$$\{DR\}_{ij}^p = \{R\}_{ij}^p - \{R_C\}_{ij}^p \tag{36}$$

is used for computing the finite difference analogues of the first and the second derivatives near to these boundaries. Clearly, the following should be valid:

$$\{R\}_{ij} = \sum_{p=1}^{p} \{R_C\}_{ij}^p \tag{37}$$

and

$$I_1^1 = 1, \;\; J_1^1 = 1, \;\; I_2^p = I, \;\; J_2^p = J. \tag{38}$$

The problem of arranging the calculation of (32) near to the boundary of the complete computational domain (32) and, respectively, arranging the calculation of near-to-boundary segments of (35) are not discussed here, since the are determined by a general solution of the differential problem and the statement of the boundary conditions.

The operations of (32) can be basically parallelized according to V-type, as (32) can be rewritten down in the vector form

$$F_{ij}^{n+n} = AF^n, \tag{39}$$

where $A$ is a differential matrix operator. However, the matrix $A$ in (39) is sufficiently filled with non-zero entries, so that calculation of one of components of the vector $F$ – according to the law of preservation of energy requires considerably more computing costs (about 85%) as compared to the calculations of other components. Therefore, acceleration of operations with V-type parallelization will not give a considerable gain in time no more than 15%, and in view of the communication losses even less.

Thus, C-, L-, and V-types parallelization, developed and applied for the local acceleration of operations in the subroutines of the principal computing kernel of "Potok-3", is not efficient for parallelization some of certain procedures.

To conclude, the authors consider it a pleasant debt to express their gratitude to Prof. V.E. Malyshkin, N.V. Kuchin, V.A. Vshyvkov, and G.S. Khakimzyanov for useful discussions and attention to this work.

## References

[1] Tarnavsky G.A., Shpak S.I. Decomposition of methods and parallelization of algorithms of solution of aerodynamics and physical gas dynamics problems: multiprogram complex "Potok-3" // Programmirovanie. – 2000. – № 6. – P. 45–57.

[2] Tarnavskiy G.A., Vshivkov V.A., Tarnavskiy A.G. Parallelization of algorithms and codes of the multiprogram system "Potok-3" // Programmirovanie. – 2003. – № 1. – P. 1–20.

[3] Tarnavskiy G.A., Shpak S.I. Parallelization schemes of operations of solution to systems of algebraic equations by the method of multi-dimensional scalar sweeping // Computing methods and programming. – 2000. – Vol. 1. – P.21–29 (http://www.srcc.msu.su/num-meth).

[4] Vshivkov V.A., Tarnavskiy G.A., Neupokoev E.V. Parallelization of algorithms of sweeping: multi-purpose computing experiments // Avtometriya. – 2002. – № 4. – P. 74–86.

[5] Tarnavskiy G.A., Tarnavskiy A.G. Modern computer technologies and non-uniquieness of solutions of gas dynamics problems // Symmetry and differential equations / Ed. V.K. Andreeva. – Krasnoyarsk: IVT, 2002. – P. 209–213.

[6] Lyubimov A.N., Rusanov V.V. Flows Around Blunt-Ended Bodies. – Moscow: Nauka, 1970.

[7] Kovenya V.M., Tarnavskiy G.A. Application of of the Splitting Method in Aerodynamics Problems. – Novosibirsk: Nauka, 1990.

[8] Tarnavskiy G.A., Shpak S.I. Problems of numerical modeling of supersonic laminar-turbulent flowing around of bodies of finite size // Matematicheskoe Modelirovanie. – 1998. – Vol. 10, № 6. – P. 53–74.

[9] Evreinov V.E., Kossarev Yu.G. High Efficiency Homogeneous Universal Computing Systems. – Novosibirsk: Nauka, 1966.