Massive Parallel Simulation of Dynamic Systems

Vladimir Svjatnyi, Leo Feldman, Vladimir Lapko, Alexander Anopriyenko

Computer Science Department, Donetsk State Technical University, Artyoma Str. 58, 340000 Donetsk, Ukraine

Andreas Reuter, Tomas Bräunl

Computer Science Department, IPVR, Stuttgart University, Breitwiesenstraße 20-22, 70565 Stuttgart, Germany

Michael Zeitz

Department of Automatic Control, ISR, Stuttgart University, Pfaffenwaldring 9, 70569 Stuttgart, Germany

---

Abstract

This paper considers the parallel simulation of dynamic systems with lumped parameters, where continuous controlled processes are described by ordinary differential equations, and dynamic systems with distributed parameters, where processes are described by partial differential equations. The concept of massive parallel simulation environment with integration of hardware, system software and simulation software which supports all stages of model constructing and simulation is presented. Some parallel algorithms and examples of simulation are described.

Keywords: Continuous systems simulation; massive parallel computing; ordinary differential equations; partial deferential equations; lumped parameters; distributed parameters.

---

1. Characteristic of dynamic systems and requirements to their simulation facilities

Dynamic systems (DS) are a wide class of objects of technics and technology where continuous controlled processes exist. The process dynamics and frequency characteristics exert the determinative influence upon the design of control systems of dynamic objects. For objects of real complexity simulation is the most effective and frequently used method of checking project solutions efficiency. Classifying dynamic systems as simulated object two most important types of systems can be distinguished: these are dynamic systems with lumped parameters (DSLP) and dynamic systems with distributed parameters (DSDP).

DSLP are systems described by ordinary differential equations (ODE), algebraic equations (AE) and logical functions (LF). ODE characterise the progress of the processes, AE express the physical interconnections between their parameters and LF can reflect structure variables and interrelations between parameters which are essential for the control of DSLP. Real DSLP are multidimensional (hundreds or thousands of dynamic parameters) and characterised by the following parameters: non-linear static characteristics, hierarchy of controlling influences and sources of energy, strong interdependence of the process parameters. In some domains these systems have variable of structure dependant on time or process parameters.

DSDP are systems described by partial differential equations (PDE) with corresponding boundary conditions, as well as by ODE, AE and LF. DSLP and DSDP function like complete

systems in many domains. In practical researches and design of dynamic systems objects with distributed parameters can be approximated by the equivalent objects with lumped parameters.

Formal description of real complexity dynamic systems consist of a topology part and a system of equations. DSLP topology is depicted by graphs (dynamic network objects of different physical nature), technological schemes, block diagrams (control systems of dynamic objects). DSDP topology can be similar to DSLP topology and some of its components (graph branches, blocks of technological schemes and block diagrams) are objects with distributed parameters of a simple geometrical form, e.g. one-dimensional (pipelines, long lines), two-dimensional (filter surface), three-dimensional (reactors, power facilities). The approximation of continuous environments and their discretization by equivalent analogies generate DSDP topology.

The requirements to dynamic system simulation facilities are as follows [2, 3, 4]:

1. User-friendliness at all stages of design and application of DS models so that user can concentrate upon a simulation problem.
2. The ability to simulate DSLP and DSDP of real complexity with maximum regard to real properties of simulated processes.
3. The ability to solve simulation problems of DS in real or accelerated time rate.
4. The presence of high level simulation language. It is very important to have the possibility of model description using minimum amount of information. The presence of dialogue system supporting active user access to resources of the simulation facilities at all stages of design and use of models.
5. The ability to simulate DS with continuous structure.
6. Modern system organisation: simulation facilities should be accessible for many users via network. Therefore users of different domains should have ability to realise model properties in the best possible way.
7. Using modern facilities of storage, visualisation, documenting, archiving and reuse of models.
8. The ability of integration with computer-aided design systems (CAD, CASE).
9. Highly developed model testing system.
10. The presence of interactive facilities of user training, which allows to study simulation methods of complex systems in a short space time.

2. Massive parallel simulation environment: definition and structure organisation

Massive parallel simulation environment (MPSE) for dynamic systems is a combination of hardware, system software and simulation software, which supports all stages of model building and simulation and meets the above requirements.

The hardware includes parallel systems of SIMD and MIMD architectures accessed by users via network. The system software consists of operating systems, parallel program languages and their translators, network software intended to support remote users, organising input-output and solving real time problems. These components are available in the existing SIMD and MIMD systems. The parallel program languages Parallaxis and Modula-P were designed at IPVR [5,6].

The simulation software included in the MPSE should be developed by means of the existing software and based on the experience of creation of simulation environments [2,4]. It should include the DSLP and DSDP parallel simulation software, the library of parallel algorithms and the programs of numerical methods, software for supporting the flexible access to all the facilities mentioned above and visualisation of simulation results.

The simulation software structure for DSLP and DSDP is shown in Fig. 1. The main part consists of programs realising parallel numerical algorithms of solving ordinary and partial deferential equation systems, which require a great number calculations. During DSDP design it is expedient to base on the generally used classification of PDE.

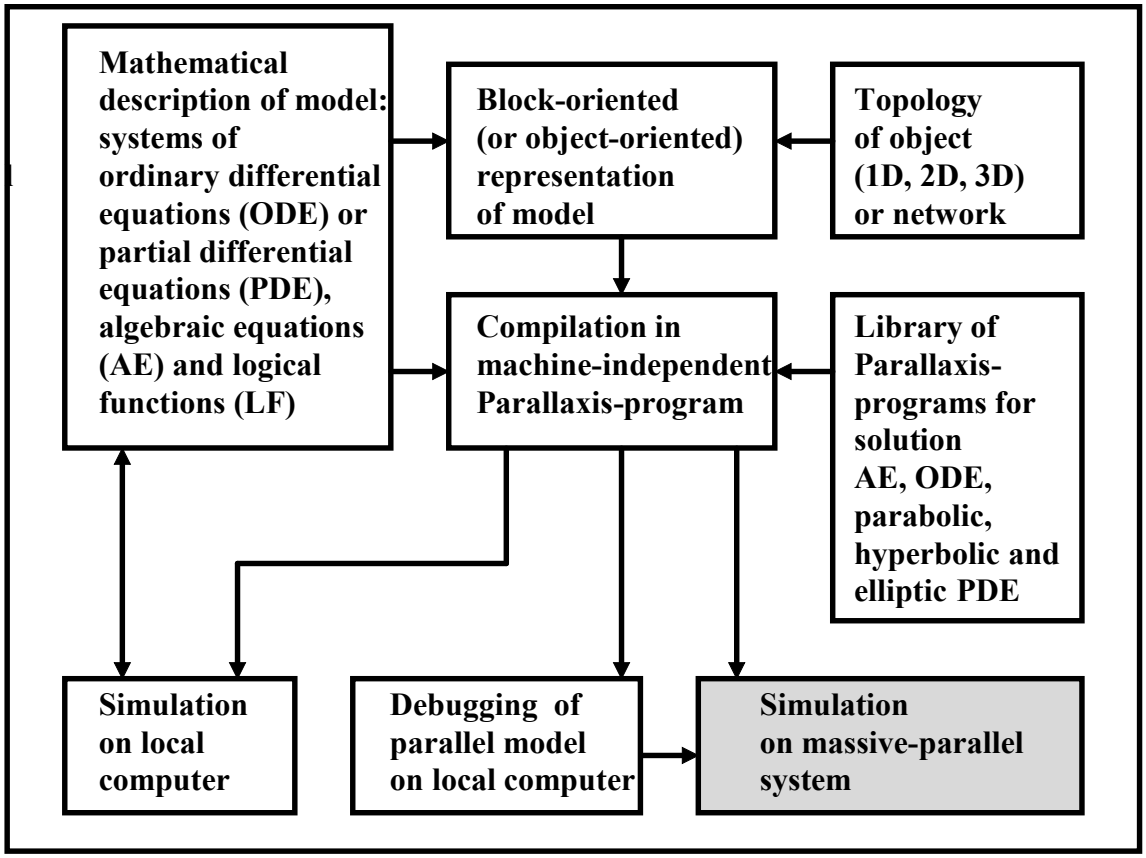

Fig. 1. Structure of simulation software for DSCP and DSDP

3. Multiprocessor systems and parallel algorithms

The difficulty of using multiprocessor systems consists not only in the limitation of processor interconnections. High performance can be achieved only when all processors or most of them are continuously loaded. But the algorithm of problem solving due to its own structure can support but frequently not the continuous load of large amount of processors despite of interconnection network structure. There are many calculation methods, which can not be effectively realised for whatever multiprocessor computer system. Hence, the structure of calculation methods should correspond to the computer system architecture. Otherwise, the required performance might not be achieved.

Below the SIMD-oriented algorithms, which can use more completely the potential parallelism difference schemes posses, are described which approximate boundary problems for partial differential equation. The highest performance of a SIMD parallel processor is achieved when it is used in matrix calculations. As is known, many algorithms require the same operations which are often repeated. If it is possible to assign one calculation node to one processor then all nodes can be calculated concurrently.

The efficiency of a parallel algorithm can be estimated by means of many criteria. The most popular criteria are acceleration and performance. Let n be the quantity of the problem parameters and $T_p(n)$ be the calculation time of the parallel algorithm using a parallel computer, which consists of $p>1$ processors and $T_1(n)$ be the calculation time of "the best" sequential algorithm. Then

$$S_p(n) = \frac{T_1(n)}{T_p(n)} \le p \tag{1}$$

denotes the acceleration, and

$$E_p = \frac{S_p(n)}{p} \le 1 \tag{2}$$

denotes the performance of a parallel algorithm. One of the aims of the parallel algorithm development is to achieve the maximum acceleration ($S_p=p$). However, the maximum acceleration can be achieved only for simple problems. The main factors decreasing the acceleration are the absence of maximum parallelism in the algorithm and time wasting for data exchange between processors. Beginning to discuss numerical algorithms for solving the problems of mathematical physics they will be estimated taking into account the two factors mentioned above. Firstly, the algorithms with maximum parallelism are discussed, then time wasting for data exchange is estimated for SIMD realisation on MasPar-1216 system.

4. Algorithms for solving boundary problems of parabolic equations

4.1. Algorithms for numerical solving of one-dimensional boundary problems

The numerical solution of a one-dimensional parabolic problem

$$\left.\begin{aligned} &\frac{\partial}{\partial}\frac{u}{t} = a^2\frac{\partial^2 u}{\partial x^2} + f(x,t),\ t>0;\ \ 0<x<l\ ; \\ &u(x,0)=\varphi(x)\ ;\ u(0,t)=\psi(t)\ ;\ u(l,t)=\xi(t) \end{aligned}\right\} \tag{3}$$

using the explicit scheme is expressed by the following equations which allow to get the solution sequentially for the next time level

$$\left.\begin{aligned} v_n^0 &= \varphi_n\ ;\ n=\overline{0,\ N+1\ ;} \\ v_n^{k+1} &= \sigma^2 v_{n-1}^k + (1-2\sigma^2)v_n^k + \sigma^2 v_{n+1}^k + \tau f_n^k\ ,\ n=\overline{1,\ ,N\ ,} \end{aligned}\right\} \tag{4}$$

where $v_0^k=\psi^k\ , v_{N+1}^k=\xi^k\ , \sigma^2=a^2\tau/h^2;\ \tau, h$ are steps of the grid. Obviously, this algorithm has maximum parallelism at each time level: $S_N(N) = N$. And it is easy to realise on SIMD processors because the algorithm requires only two data exchanges, which are accomplished between two neighbour processor elements, at each time level,. To realise this algorithm N processors are required. Its performance takes into account the dependence upon the calculation of complexity of functions f, ψ, ξ and equals $E_N(N)=0.75$ at worst for homogeneous equations and homogeneous conditions. However, the application of this

algorithm is restricted by the fact that explicit difference scheme is unalterable only when $\tau \le a^2\tau / h^2$.

Approximation of problem (3) by the implicit scheme is expressed in the following equation system

$$\left.\begin{aligned} v_n^0 &= \varphi_n; \\ v_0^{k+1} &= \psi^{k+1}, \\ \sigma^2 v_{n-1}^{k+1} - (1+2\sigma^2)v_n^{k+1} + \sigma^2 v_{n+1}^{k+1} &= -v_n^k - \tau f_n^k, \quad n = \overline{1, N}, \\ v_N^{k+1} &= \xi^{k+1}. \end{aligned}\right\} \tag{5}$$

Using sequential computers this problem is solved by Progonka method which is not parallel, thus it is not efficient to realise on parallel computers. There are some effective parallel algorithms of solving three-diagonal linear equation systems. One of them is briefly discussed below.

Omitting for short the superscript indexes in (5), assuming for convenience $N\text{-}1=2^q$ and denoting the right part with $\alpha_n, \beta_n, \gamma_n, g_n$, the following formulae can be written:

$$\left.\begin{aligned} \beta_{n-1}v_{n-2} - \alpha_{n-1}v_{n-1} + \gamma_{n-1}v_n &= g_{n-1} \\ \beta_n v_{n-1} - \alpha_n v_n + \gamma_n v_{n+1} &= g_n \\ \beta_{n+1}v_n - \alpha_{n+1}v_{n+1} + \gamma_{n+1}v_{n+2} &= g_{n+1}, \quad n = \overline{2, N-1}. \end{aligned}\right\} \tag{6}$$

If unknowns v_{n-1} and v_{n+1} are excluded and three neighbours are grouped again then we obtain

$$\left.\begin{aligned} \beta_{n-2}^{(1)}v_{n-4} - \alpha_{n-2}^{(1)}v_{n-2} + \gamma_{n-2}^{(1)}v_n &= g_{n-2}^{(1)}, \\ \beta_n^{(1)}v_{n-2} - \alpha_n^{(1)}v_n + \gamma_n^{(1)}v_{n+2} &= g_n^{(1)}, \\ \beta_{n+2}^{(1)}v_n - \alpha_{n+2}^{(1)}v_{n+2} + \gamma_{n+2}^{(1)}v_{n+4} &= g_{n+2}^{(1)}, \quad n = \overline{2, N-1}, \end{aligned}\right\} \tag{7}$$

where

$$\beta_n^{(1)} = \frac{\beta_{n-1}\beta_n}{\alpha_{n-1}}, \quad \alpha_n^{(1)} = \alpha_n + \frac{\beta_n\gamma_{n-1}}{\alpha_{n-1}} + \frac{\beta_{n+1}\gamma_n}{\alpha_{n+1}}, \quad \gamma_n^{(1)} = \frac{\gamma_n\gamma_{n+1}}{\alpha_{n+1}}, \quad g_n^{(1)} = \frac{\beta_n g_{n-1}}{\alpha_{n-1}} + g_n + \frac{\gamma_n g_{n+1}}{\alpha_{n+1}}.$$

In (7) it should be assumed, that $v_j=0$ for all $j<0$ and $j>N+1$ in order to be able to exclude the unknown variables simultaneously in each group. It is obvious, that the process can go on recursively until each equation consists of one variable only after $q = \log_2(N-1)$ steps. At the last $(q+1)$-th step all unknown variables can be found concurrently too.

The described method of cyclic reduction was previously used on sequential computers, since it has less amount of scalar arithmetical operations and, hence, it is the best sequential algorithm. If the factors of equations (7) are calculated at each step concurrently at $N\text{-}1$ processors of a SIMD-system, and then their values are sent to the appropriate processors, then the solution of the equation system (6) will require $q + 1$ step. At each step, except for the last one, each processor should send four values. We shall designate as t_f the quantity of operations of factors required for the calculation of the system (7) on a sequential computer, as t_u the time of the parallel exchange in a SIMD-structure, as t_g the time of the calculation of unknown x_i from the equation similar to (7), which is obtained at the first step in the method of sequential cyclic reduction (SERICR), then the acceleration of the parallel cyclic reduction algorithm (PARACR) will be equal to

$$S_N(N) \approx \frac{N(t_f + t_g) - \log_2 2N}{(t_f + t_u)\log_2 N} = O(\frac{N}{\log_2 N}). \tag{8}$$

If t_u denotes the average time necessary for one arithmetic operation, then the algorithm efficiency equals $E_N(N) \approx (\log_2 N)^{-1}$ for the problem under consideration (6).

4.2. Concurrent solution algorithms of multidimensional parabolic boundary problems

Explicit and implicit methods are usually used for solving multidimensional parabolic boundary problems. Explicit difference schemes for multidimensional parabolic problems, as in the case of one-dimensional problem, have maximum possible natural parallelism. However, because of conditional stability of such difference schemes, their use is not always possible. At present, there are no efficient parallel direct methods of solving implicit difference equation systems.

Difference schemes of splitting offer a means of parallelizing. For example, for the two-dimensional parabolic problem

$$\frac{\partial u}{\partial t} = a^2\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + f(x,y,t) \tag{9}$$

with appropriate initial and boundary conditions local one-dimensional scheme of splitting is of the following form

$$\left.\begin{aligned} \frac{\tilde{v}_{m,n} - v^k_{m,n}}{\tau} &= a^2\,\frac{\tilde{v}_{m-1,n} - 2\tilde{v}_{m,n} + \tilde{v}_{m+1,n}}{h^2} + 0.5 f^k_{m,n}, \\ \frac{v^{k+1}_{m,n} - \tilde{v}_{m,n}}{\tau} &= a^2\,\frac{v^{k+1}_{m-1,n} - 2v^{k+1}_{m,n} + v^{k+1}_{m+1,n}}{h^2} + 0.5 f^{k+1}_{m,n}. \end{aligned}\right\} \tag{10}$$

Here the solution at each time level consists of two stages: finding intermediate values $\tilde{v}_{m,n}$, and then calculating values $v^{k+1}_{m,n}$ at the next time level. At each of these stages difference equations are three diagonal systems of equations, which, as in the case of one-dimensional problem, can be solved by the cyclic reduction method. If for simplification of estimations it is assumed, that x, y area for the problem (9) is a square, then the parallelism degree of this algorithm being equal to N^2 the PARACR method is the most efficient. The acceleration of this method is

$$S_{N^2}(N^2) = \frac{N\left\{t_f\left[N - \log_2 N\right] + t_g N\right\}}{\left(t_f + t_u\right)\log_2 N} = O\left(N^2 / \log_2 N\right). \tag{11}$$

If only N processors are used, then each equation system can be solved by the PARACR method, using N processors, and then N such systems can be solved sequentially, or each system is solved on one processor by the SERICR method, and all N systems are solved concurrently, then the acceleration estimation is as follows

$$S_N(N^2)_{paracr} = O\left(N / \log_2 N\right); \quad S_N(N^2)_{sericr} = O(N), \tag{12}$$

These estimations testify to advantages of the SERICR method. Similar evaluations are also obtained for other difference schemes of splitting. In the same way it is possible to consider and get estimations for the space parabolic equation.

4.3. Concurrent solution algorithms of elliptic boundary problems

The parallelism estimations of algorithms are done for a model problem, approximating Poisson's equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x,y); \quad 0 < x < 1, \quad 0 < y < 1, \tag{13}$$

the corresponding difference equation of which is of the following form

$$v_{m-1,n} + v_{m+1,n} - 4v_{m,n} + v_{m,n-1} + v_{m,n+1} = h^2 f_{m,n}, \; m,n = \overline{0,N}. \tag{14}$$

If the first boundary problem is solved, then function u and, hence, grid function v are defined at the boundary. Thus, a linear system of N^2 equations is obtained, the matrix of which is diagonally rarefied and contains only five non-zero diagonals. At present, there are no efficient direct methods of solving similar systems of equations. For large values of N such systems are solved by iterative methods, as a rule. The analysis of the solution algorithms of difference equations, allows to answer the following questions:

- performance and possibilities of parallelizing;
- estimation of parallel algorithms given the characteristic correlation between the number of unknowns and the number of processor elements (PE).

Proceeding from the obtained estimations of efficiency of parallel algorithms we have chosen the following parallel algorithms for realisation on SIMD system MasPar: parallel method of upper relaxation, method of variable directions and multigrid Fedorenko method. The first two methods are well known in contrast to the third one, therefore we shall present a brief description of this method. To reduce the norm of the initial error lnN times this method needs $O(N^2)$ arithmetical operations. It is known, that the fastest method of Duglas-Rechford requires $O((lnN)N^2)$ operations. Limits of applicability of multigrid method are almost the same, as of the elementary method of establishment. The multigrid method was intended for application at a one-processor computer, where nodes of the grid area are processed sequentially. At the same time it is easy to parallelize and it is effective to realise on SIMD systems. Its implementation is especially simple for the case, when the number of PE is sufficient for processing of all nodes of a small-sized grid, i.e. $p=N^2$. When $p < N^2$ it is not difficult to construct a block sequential-parallel algorithm on a small-sized grid and a parallel algorithm on a large grid.

The multigrid method can be modified for the realisation on SIMD systems. For simplification we shall restrict ourselves to the case, when two grids are used – a small one and a large one. When calculating corrections in nodes of the large grid, it is also possible to calculate corrections in other internal nodes of the small grid, using the pattern of the large grid except for boundary nodes. The values of corrections in boundary nodes can be determined by interpolation. Besides, more accurate initial approximation for the small grid can be obtained,

using approximate solutions on a sequence of condensing grids. These changes of the algorithm allow to reduce the total number of parallel iterations, which are necessary for solving a problem with given accuracy, as well as to increase the degree of parallelism.

4.4. Library of programs for parallel simulation of DSDP

The library includes procedures of all the above considered numerical methods, as well as various auxiliary programs. For example, problem correctness checking, scanning the area, where the solution is being found, input/output procedures, interpreter of symbolic expressions for input of functions and others. The library allows to carry out the numerical solution of two-dimensional parabolic and elliptic boundary problems in the area, representing closed multiangles, parts of border of which being segments of straight lines and arches of circles. The boundary conditions in each section of the border are set in the form of conditions of the third kind. Both rather simple methods (such as Jacoby's method, Seidel's method, method of upper relaxation) and complex methods (multigrid, extrapolation, as well as Schwarz' method and area partition. The programs are made in parallel programming language Parallaxis-3 for operating system Linux, some auxiliary procedures are written in C ++.

5. Massive parallel models of dynamic net objects

Dynamic net objects (DNO) are widely known in different fields of engineering [1]. When designing control systems of network objects of real complexity there are problems of checking the correctness of design solutions, which can be resolved only by methods of mathematical simulation. DNO models present a number of rigid requirements to means of computer facilities, which can be satisfied only by parallel computer systems. The report considers the algorithms of massive parallel models of DNO construction and the results of their realisation in the structure of simulation facilities.

5.1. DNO topology and algorithm of topological part of models definition

The DNO topology is characterised by a strongly connected graph $G(u,v)$ where $u=n$ is a number of nodes and $v=m$ is a number of branches. In real DNO m exceeds 200 and n exceeds 100. Therefore, the task of formal definition of DNO consists in the development of specification language, which contains minimum initial information in terms and means of representation, and which is clear to the user of a subject area. Graph $G(u,v)$ is specified by the table of the following type

$$A_j,\ E_k,\ Q_i,\ (par\text{-}i),\ (kom\text{-}i), \tag{15}$$

where Q_i is a physical flow (current in the electrical net, gas or air expenditure in aerogasdynamic nets, water expenditure in hydraulic nets, etc.) in the *i-th* branch of the graph; A_j, E_k are numbers of the first and the last node of the branch.

Indexes $j,\ k \in (1,2,\ldots,n)$; *(par-i)* is an array of i-branch parameters; *(kom-i)* is verbal definition of the comment for i-branch.

The table (15) is prepared by the user of a subject area in a dialogue mode, corresponding to the format of a given simulation environment. Columns A_j, E_k, Q_i (table 15) are used for construction of topological incidence matrixes

$$A=F_a(A_j, E_k, Q_i), \tag{16}$$

and contour matrixes

$$S=F_s(A_j, E_k, Q_i), \tag{17}$$

required for the automatic formation of DNO equations. Here F_a, F_s are algorithms of matrix formation. Let us introduce the following operations for variables A_j, E_k.

Assignment

$$A_j:=E_k\ (j=k), \tag{18}$$

corresponds to the transition from the column of end nodes to the column of beginning nodes, realised during the analysis of rows of the table (15), i.e. interrelation of branches of incidental nodes $j=k$.

Logical operation

$$L[E_p\ ^a(A_j, E_{p-1}, E_p, E_{p+1})], \tag{19}$$

determines, whether node E_p belongs to the set of beginning and end nodes.

A table algorithm of constructing graph G(u,v) tree (antitree), being the modification of known consecutive algorithms, is proposed. The idea of the algorithm is that the initial table of the graph

$$TABUR=(A_j, E_k, Q_i), \tag{20}$$

is transformed at some stages to the final tables of the tree BAUMTAB and antitree ANTIBAUTAB:

1) Analysing rows of table TABUR, corresponding to parallel branches:

$$\begin{gathered} Aj(Qi)=Aj(Qp); \\ Ek(Qi)=Ek(Qp); \\ Qp<>Qi \end{gathered} \tag{21}$$

According to the results of analysis the TABOP table is formed. The table differs from TABUR in the absence of parallel branches, having the same end nodes and beginning nodes. From the rows "reseted" during the transformation of TABUR to TABOP, ANTIBAUTAB is formed to be appended at the further stages.

2) From TABOP table the current table of a tree BAUTAB-d and current array of nodes KNOAR-d are obtained (d is an index, taking on the values of *1,2,... D,* where *D* is the number of steps of the transformation of TABOP table to the final table BAUMTAB). The end nodes and beginning nodes of the tree are placed into the table KNOAR-d. The beginning nodes,

which are entered as end nodes in BUAMTAB-d, are determined by the operation (18). Operation (19) checks which end nodes E_k are already entered in the array KNOAR-d.

If $L[E_k{}^a(KNOAR\text{-}d)]=TRUE$,
then ANTIBAUTAB is extended by adding a row, containing E_k.
Array KNOAR-d and table BAUMTAB-d remain unchanged.

If $L[E_k{}^a(KNOAR\text{-}d)]=FALSE$,
then KNOAR-d and table BAUMTAB-d are extended.

3) Checking the condition of finishing the process of a tree construction by the formula

$$BF=(BZ=n-1)\ AND\ (ABZ=m-n+1)\ AND\ (MOD\ KNOAR\text{-}d=n) \qquad (22)$$

where *BF* is a logical variable "tree is complete";
BZ is a number of rows (tree branches) in BAUMTAB-d table;
ABZ is a number of rows (antitree branches) in ANTIBAUTAB-d table;
MOD KNOAR-d is a number of nodes in KNOAR-d array.

By condition BF=TRUE the search cycle is finished and the visual table construction is performed. To generate an incidence matrix, table TABXY is formed from the tables of tree and antitree. Components of the table TABXY are ordered in conformity with the identifiers of branches of a tree X_i and antitree Y_j $(i=1,2,...,n-1;\ j=1,2,...m-n+1)$. The following algorithm of generating a matrix table A=(AxAy),(TABA) is proposed.

Identifier KNO(j) with $j=1,2,...n$, corresponding to the row number in TABA and in matrix A, is compared by columns to the parameter of beginning nodes and end nodes $A(k)$, $E(k)$, $k=1,2,...m$.

By the results of comparison an array of rows Za(j) is formed in the following form:

$$Z_A(j)=(x_1\text{-}x_{20}...x_{n-2}x_{n-1}\text{-}y_{10}...y_{s-1},y_s) \qquad (23)$$

Rows of submatrixes AX, AY and matrix A (AX, AY) are formed from $Z_A(i)$. From the table TABXY a matrix of contours is formed in the form of S= (SX, SY) by the following algorithm:

1) From TABY, ABZ=m-n+1 of the contours' headers is formed in the form of

$$M(j):=(A_k(Yj)E_k(Yj))Yj \qquad (24)$$

2) Using operation (18) the base of contour $M(j)$ searches for branches, which belong to this contour. As the result there is a table of j-contour, which is transformed into the intermediate vector MS(j,k), the elements of which are the components of vectors X,Y and zeros.

3) From the vector MS(j, k) elements of submatrixes SX, SY and matrix S(SX, SY) are determined.

5.2. Generation and solution of equations of DNO with lumped parameters

Dynamic processes in the branches of net objects with lumped parameters (DNO-LP) are defined by ordinary differential-integral equations concerning material flows (current in the electrical net, gas or air expenditure in aerogasdynamic nets, water expenditure in hydraulic nets, etc.). The generalised system of equations for objects includes equations of continuity in nodes

$$AQ=0, \tag{25}$$

and equation of movement in closed contours

$$SK\frac{dQ}{dt} + SR(t)f(Q) + SB(t)\int \varphi(Q)dt = SH(Q) \tag{26}$$

where Q is a vector of current of some physical nature;

$K, R(t), B(t)$ are diagonal matrixes of parameters, characterising branch sluggishness, resistance to movement of a flow, capacitor properties;

$f(Q), \varphi(Q)$ are vectors with non-linear elements, characterising physical properties of flows in branches;

$H(Q)$ is a vector of characteristics of active DNO elements (generators, turbo-compressors, etc.).

Using the above obtained A(AX,AY), S(SX,SY), Q(XY), K(KX,KY), R(RX,RY), B(BX,BY), H(HX,HY) let us present the system in the form convenient for numerical solution:

$$X = -WY \tag{27}$$

$$\frac{dY}{dt} = S_n H(HX, HY) - S_n R(t)f(x, y) - S_n B(t)\int \varphi(x, y)dt \tag{28}$$

where W, S_n - matrixes, calculated in the topological part of parallel DNO models [2].

The system (27), (28) is solved by the numerical method, algorithm of which is realised in the language of parallel programming.

Languages of parallel DNO simulation of block-oriented, equation-oriented and problem-oriented types are developed on the basis of algorithms of construction of A, S and formation of equations in the form (27), (28).

5.3. Generation and solution of equations of DNO with distributed parameters

The dynamic processes in branches of network objects with distributed parameters are described by partial differential equations of various types. We shall consider a class of network objects, to which electrical, hydraulic and other networks belong. The branch of DNO can be assumed to be one-dimensional by the co-ordinate x, counted along branches from beginning and end nodes. Any of X or Y branches are described by the equation of movement

$$\frac{\partial P}{\partial x} = k\frac{\partial x}{\partial t} + rf(x) \tag{29}$$

and the equation of continuity

$$-\frac{\partial P}{\partial t} = b\frac{\partial x}{\partial x}, \tag{30}$$

where *k, r, b* are specific parameters, characterising the sluggishness, resistance and capacitor properties of the process;

x (or *y*) are currents in branches of tree and antitree of DNO graph;

P (*PX* and *PY* respectively) is a parameter which characterises the distribution of driving forces (voltage, pressure, etc.).

The boundary conditions for equations (29), (30) are the values of functions *P(x,t)* in the beginning and end nodes of the branches. These functions are the required variables in the equation system of the net, which consists of *m* pairs of equations in the form of (29), (30).

Approximating the system (29), (30) by the method of straight lines, receiving *M* pairs of ordinal differential equations for each branch in a kind

$$\frac{dX_{ik}}{dt} = \alpha_{xi}(P_{ik-1} - P_{ik}) - \beta_{xi} f(X_{ik})$$

$$\frac{dP_{ik}}{dt} = \gamma_{xi}(X_{ik} - X_{ik+1}) \tag{31}$$

where

i is a number of X-branch;

k is a number of elementary site of branch with length Dx;

α_{xi}, β_{xi}, γ_{xi} - factors.

With K conterminous with boundary node the equation of continuity acquires the following form

$$\frac{dP_{uj}}{dt} = \gamma(\sum X_{uj} - \sum Y_{uj}) \tag{32}$$

In the stabled mode equations (32) for all nodes of the net *(j=1,2,...,n-1)* are transformed into the system (27). Entering the vector of boundary (node) values for the whole net

$$P_u=(P_{u1},P_{u2},...,P_{un}),$$

diagonal parameter matrix $G(\gamma_{u1}, \gamma_{u2},...,\gamma_{un})$, vectors of currents X_u, Y_u incidental to the nodes, we obtain a matrix system of equations:

$$\frac{dP_u}{dt} = G(A_x XU + A_y YU), \tag{33}$$

determining boundary conditions for branches. The system (31) for X- and Y- branches is easily represented in matrix form.

When developing a parallel SIMD-model, there are two methods of solving the equation system (31), (33):

- co-ordinate method, in which a net with discrete branches is imposed on a grid of processors of type (grid [L_M], [L_K], where L_M is a total quantity of approximating elements by maximum contour; L_K is a number of processor branches corresponding to the number of base branches of antitree, graph, net.

- the method of approximated branches, in which each approximated branch is assigned to a pair chain of processor elements, and connections between them are constructed according to the equation (33).

In both cases the systems (31), (33) are solved numerically by a parallel algorithm.

6. Parallel simulation of optimum systems expressed by equations of network objects

The complex topological network object under consideration is the mine ventilation system, which distributes air among consumers with the purpose of decreasing to a safe level the concentration of harmful industrial gases in the atmosphere.

To establish a safe concentration of harmful gases, when their emission has boundary frequency of the order of $0.05\ s^{-1}$, the optimum regulation of air distribution should provide a law of establishment of the required flow rates both in character, and in duration of transition. For technological reasons it is rather important, that the transition has non-periodic character, even in the presence of harmonic containing finite number of fluctuations.

The transport movement in an uncontrolled network causes essential high-frequency fluctuations of resistance of the network branches and according to the flow rate with boundary circular frequency $\omega = 0.2 - 0.3\ s^{-1}$. Effective indemnification of these fast changes is the major dynamic problem of a system, regulating the air distribution in a network object.

At the preliminary stage of the development of a network object control system we carry out the synthesis of a control system, which is optimum in quality for a separate branch of the network. Qualitative characteristics of a network object control system as a whole are analysed with the application of a parallel mathematical model.

In general, aerodynamic processes in a separate branch of a network object are described by non-linear telegraphic equations:

$$-\frac{\partial H}{\partial x} = \frac{\rho_b}{F_b}\frac{\partial G}{\partial t} + RG^2; \tag{34}$$

$$-\frac{\partial G}{\partial x} = \frac{F_b}{\rho_b a^2}\frac{\partial H}{\partial t}. \tag{35}$$

Here H is a pressure along the axis of a branch; G - air flow rate; ρ_b - air density; a - speed of a sound in the air; F_b - section of a branch; R - specific resistance of a branch; x - distance of the given point from the beginning of a branch.

The equations (34) and (35) describe dynamics of a branch in an unlimited range of frequencies up to $\omega = \infty$. In the limited interval under consideration of the working frequencies of the order 0.5 s^{-1} more simple models of dynamics of a branch as a controlled object can be used for the control system synthesis. Linear system of telegraphic equations is considered for the synthesis of an approximate model of aerodynamic processes in a network branch:

$$-\frac{\partial P}{\partial x} = \frac{\rho_b}{F_b}\frac{\partial Q}{\partial t} + rQ; \tag{36}$$

$$-\frac{\partial Q}{\partial x} = \frac{F_b}{\rho_b a^2}\frac{\partial P}{\partial t}, \tag{37}$$

where $r = 2RG_H$ - differential specific resistance of a branch in the vicinity of a stationary nominal mode $G = G_H$;

$Q(x,t) = [G(x,t) - G_H]$ - deviation of the air flow rate from nominal mode;

$P(x,t) = [H(x,t) - H_H(x)]$ - deviation of pressure from nominal pressure $H_H(x)$ in stationary mode;

$H_H(x) = [H_K^H + RG_H^2(l_b - x)]$ - pressure distribution along a branch at nominal flow rate;

l_b - length of a branch.

According to (36) and (37) the resistance of a branch in the final section (at the end of a branch) is defined by the expression

$$W(s) = P_K(s) / Q(s) = -\bar{\rho}\, th(\gamma\tau / 2), \tag{38}$$

where $\tau = 2\, l_b/a$; $\gamma^2 = s(s + \delta)$; $\delta = \frac{rF_b}{\rho_b}$; $\bar{\rho} = \frac{\rho\gamma}{s}$; $\rho = \frac{\rho_b a}{F_b}$.

Approximating (34) and (35) by the method of straight lines and expanding function (38) in a Taylor series for branches of real networks in the essential range of frequencies in the presence of square-law resistance the following models of a branch as an object under control can be obtained:

$$K_i(p)G' + R_s G^2 = \Delta H_0. \tag{39}$$

Here $R_s = (R_b + R_c)$ is a total resistance of a controlled branch; $R_b = Rl_b$ is a resistance of a branch; R_c is a controlled resistance of a pressure-differential device; ΔH_0 is a general depression of an autocontrolled branch; $K_i(p) = K_1(p), K_2(p), K_3(p)$ are factors of inertia for models of aerodynamics in a branch of the third, second and first orders, respectively;

$$K_3(p) = (a_3 p^2 + a_2 p + a_1); \quad a_3 = \frac{\tau^2}{24} M_b; \quad M_b = \frac{\rho_b \ell_b}{F_b}; \quad a_2 = \frac{\tau^2}{12} rl_b + b_2 r_g^H;$$

$$a_1 = \frac{\tau^2 r\ell_b}{24T_b} + M_b; \quad b_1 = \frac{\tau^2}{8T_b}; \quad b_2 = \frac{\tau^2}{8}; \quad T_b = \frac{M_b}{rl_b}; \quad r_g^H = R_g^H G_H;$$

R_g^H - nominal resistance of a pressure-differential device;

$$K_2(p) = (e_2 p + e_1); \quad e_2 = \frac{\tau^2}{4} r_g^H; \quad e_1 = \frac{\tau^2}{4T_b} r_g^H + M_b;$$

$$K_1(p) = M_b;$$

It is shown, that the equations of the third, second and first orders describe adequately the dynamics of branches with the length of 2000, 600 and 300 meters respectively in the range under consideration of essential control frequencies of the order of 0.5 s^{-1} of. The adequacy of the models is justified by the high degree of convergence of the frequency characteristics of simplified models and of the corresponding gear function (38). In the time interval a chained line of elements with four poles was used as a reference model for the accuracy evaluation of approximate models. Control processes optimum in quality in autocontrolled branches were designed with the help of equation (38)

$$A_0^i(p)G = G^0(t), \tag{40}$$

where $G^0(t)$ is a given debit of air in a branch under control; G is a current flow rate of a branch;

$A_0^i(p) = A_0^3(p), A_0^2(p), A_0^1(p)$ are the optimum operator of the characteristic equation of transitions in branches of the third, second and first orders, respectively;

$A_0^3(p) = (a_3^0 p^3 + a_2^0 p^2 + a_1^0 p + 1);$

$A_0^2(p) = (e_2^0 p^2 + e_1^0 p + 1);$

$A_0^1(p) = T_b^0 + 1;$

$a_3^0, a_2^0, a_1^0, e_2^0, e_1^0, T_b^0$ are optimum constant factors of the characteristic equation for branches of appropriate length.

For the maintenance of natural adaptation regulation in a system is performed according to the law

$$u = K (G_T^{(n)} - G^{(n)}), \tag{41}$$

where $G_T^{(n)} = A_0^{(n)} G$ is the desirable value of the senior derivative of equation (40) in an autocontrolled branch; $G^{(n)}$ is the current value of the senior derivative, K is a parameter of a regulator set-up.

The resistance of a pressure-differential device of a branch is defined, in general, by a non-linear function

$$R_p(u) = f_d(u), \tag{42}$$

where $f_d(u)$ is an aerodynamic characteristic of the pressure-differential device. The formation of values of current phase co-ordinates of a branch and of the senior derivative is carried out by a real differentiating filter:

$$N_f^i(p)G_f = G(t), \tag{43}$$

with $N_f^i = N_f^3, N_f^2, N_f^1$ is a characteristic equation of the filter of the third, second and first orders, respectively.

The order of the filter inertia (43) is assumed to be less than minimum time constant of the equation of a branch (39). According to this

$$G^{(i)} = G_f^{(i)}, \tag{44}$$

where $G^{(i)}$ is the evaluation by derivative of an air flow rate.

The system of equations (39) - (44) defines the required amplification factor of a regulator and the border of the control system stability area in a separate network branch as a function of parameters of the filter and regulator. Aerodynamics of controlled network object is described as a whole by the planimetric equation

$$S_x K_x(p) G'_x + S_y K_y(p) G'_y + S R_s Z = SH. \quad (45)$$

Here $K_x(p)$, $K_y(p)$ are matrixes of inertia factors, corresponding to branches of a tree and antitree of a network; H is a vector of depression of draft sources; R_s is a matrix of total aerodynamic resistance of all branches of a network; S_x, S_y, S are matrixes of planimetric factors of a network; Z is a vector of square-law flow rates of a network. The flows of the tree and antitree of a network object are connected by the equation of a continuity

$$G_x = - W G_y, \quad (46)$$

where $W = A_x^{-1} A_y$ are matrixes of factors corresponding to the tree and antitree of a network. Taking into account (46) we shall present an equation (12) in the form of

$$W_y G_y = SH - S R_s Z, \quad (47)$$

with $W_y = S_y K_y(p) - S_x K_x(p) W$.

The analysis of the qualitative characteristics of the network object control system of the air distribution was carried out with the use of massive-parallel model of the network object. In the parallel model each vector element of the equation systems (40) - (44), (47) is assigned to a processor in the processor grid of MasPar system. The software of the massive-parallel model is realised in Parallaxis language.

During simulation the following issues are considered:

- transitions of establishment of a natural flow distribution in non-controllable network;
- improvement of the required flow distribution in various points of the areas of stability and non-periodicity of control processes;
- qualitative characteristics of static and dynamic errors when improving optimum trajectories in various modes of the system operation;
- invariance of system to the change of fast non-stationary parameters of a network object;
- border of the areas of stability and non-periodicity of the control processes depending on the amplification factor a regulator and on the parameters of set-up of differentiating filters of local control systems.

Simulation has confirmed, that the law of control by the senior derivative, accepted in regulators, has properties of natural adaptation and independence of functioning of regulators under the conditions of fast changes of network parameters and provides accuracy acceptable in practice. The researches confirm, that the border of stability area of a network and periodicity of the control processes practically coincide with theoretical calculations in the vicinity of typical nominal mode of the network operation.

7. Summary

Research and development of main algorithms for the simulation of DNO with lumped and distributed parameters is carried out in accordance with the stated concept. The models are realised in the parallel programming language Parallaxis [5], which allows to conduct experimental researches of parallel algorithms both on the basis of a personal computer and on SIMD system MasPar. Based on the facilities of X Windows the user interface is realised, which integrates all developed algorithms into the uniform, convenient for user simulation system. The parallel models created are used for the development of ventilation control systems for coal mines. The library, which is the main part of the parallel models subsystem of the DNO control system, is created. The results stated above are obtained by the authors within the framework of co-operation between Donetsk State Technical University and University of Stuttgart [2].

References

[1] Svjatnyi, Simulationsverfahren für aerogasdynamische Netzobjekte, *Jahrestagung der Gesellschaft für Informatik*, Stuttgart (1990) Band 1, 476-483.

[2] Anoprienko, V. Svjatnyi, T. Bräunl, A. Reuter, M. Zeitz, Massiv parallele Simulationsumgebung fuer dynamische Systeme mit konzentrierten und verteilten Parametern, *Simulationstechnik. 9 Simposium in Stuttgart, Oktober 1994, Tagungsband*, Vieweg (1994) 183-188.

[3] Abramov, L. Feldman, V. Svjatnyj, *Simulation of Dynamic Processes of Mine Aerologie* (Russian), Kiev, Naukova Dumka, 1981.

[4] Zeitz, Simulationstechnik, *Chem.-Ing.-Tech*. 59 (1987), Nr. 6, 464-469.

[5] T.Bräunl, Parallaxis-III: A Language for Structured Data-Parallel Programming, *User Manual. Computer Science Report*, IPVR Univ. Stuttgart, Oktober 1994.

[6] T.Bräunl, R. Norz, Modula-P, *User Manual. Computer Science Report*, IPVR Univ. Stuttgart, August 1992.

[7] Reuter, Grenzen der Parallälität, *Informationstechnik* (1992) 1, 62-74.

Citation:

Svjatnyi V., Feldmann L., Lapko V., Anopriyenko A., Reuter A., Bräunl T., Zeitz M. Massive parallel simulation of dynamic systems // Zeszyty naukowe. – 1997. – №1. – P. 207-229.