CONCEPTS AND ALGORITHMS FOR PARALLEL SIMULATION OF DYNAMIC SYSTEMS

Svjatnyi V., Feldman L., Lapko V., Reuter A., Bräunl T.

Computer Science Department, Donetsk State Technical University
IPVR, Stuttgart University
svjatnyj@dstu.donetsk.ua

Abstract

This paper considers the parallel simulation of dynamic systems, 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 concepts of 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 are described.

1. Dynamic systems simulation facilities: main requirements

Simulation is the most effective and frequently used method of checking project solutions efficiency for systems and objects of real complexity. Classifying dynamic systems (DS) 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 main requirements to dynamic system simulation facilities are as follows:

1. User-friendliness at all stages of design and application of DS models.
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.
5. The presence of dialogue system supporting active user access to resources of the simulation facilities at all stages of design and use of models.
6. The ability of integration with computer-aided design systems (CAD, CASE).

7. Highly developed model testing system.

2. Massive parallel simulation environment

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 [5,6,7].

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 [1,4,10]. 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.

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

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

denotes the acceleration, and

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

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

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

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

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

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

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

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:

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

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

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

where

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

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 $N-1$ 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

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

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

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

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

$$\left.\begin{aligned} \frac{\tilde{v}_{m,n} - v_{m,n}^k}{\tau} &= a^2\,\frac{\tilde{v}_{m-1,n} - 2\tilde{v}_{m,n} + \tilde{v}_{m+1,n}}{h^2} + 0.5 f_{m,n}^k, \\ \frac{v_{m,n}^{k+1} - \tilde{v}_{m,n}}{\tau} &= a^2\,\frac{v_{m-1,n}^{k+1} - 2v_{m,n}^{k+1} + v_{m+1,n}^{k+1}}{h^2} + 0.5 f_{m,n}^{k+1}. \end{aligned}\right\} \qquad (10)$$

Here the solution at each time level consists of two stages: finding intermediate values $\tilde{v}_{m,n}$, and then calculating values $v_{m,n}^{k+1}$ 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

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

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

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

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

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

the corresponding difference equation of which is of the following form

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

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. Summary

Research and development of main algorithms for the simulation of is carried out in accordance with the stated concepts. The models are realised in the parallel programming language Parallaxis [6], which allows to conduct experimental researches of parallel algorithms both on the basis of a personal computer and on SIMD system MasPar. 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 cooperation between Donetsk State Technical University and University of Stuttgart [2, 3, 4, 10].

References

1. Abramov A., Feldman L., Svjatnyj V., *Simulation of Dynamic Processes of Mine Aerologie* (Russian), Kiev, Naukova Dumka, 1981.
2. Anoprienko A., Bräunl T., Eberhardt G., Feldmann L., Svjatnyj V. Aerodynamische Netze als parallele Modelle / *BI, Benutzerinformation des Rechenzentrums*, Rechenzentrum Universität Stuttgart. N1, 1995.
3. Anoprienko A., Feldmann L., Lapko V., Svjatnyi V., Bräunl T., Reuter A., Zeitz M. Massive parallel models of net dynamic objects. *Proceedings of the 1995 EUROSIM Conference, EUROSIM-95*, Vienna, Austria, 11-15 September 1995, ELSEVIER, 1995, p. 237 - 242.
4. Anoprienko A., Svjatnyi V., Bräunl T., Reuter A., Zeitz M., Massiv parallele Simulationsumgebung für dynamische Systeme mit konzentrierten und verteilten Parametern, *Simulationstechnik. 9 Simposium in Stuttgart, Oktober 1994, Tagungsband*, Vieweg (1994) 183-188.
5. Bräunl T. *Parallel Programming*. Prentice-Hall 1993.
6. Bräunl T., Parallaxis-III: A Language for Structured Data-Parallel Programming, *User Manual. Computer Science Report*, IPVR Univ. Stuttgart, Oktober 1994.
7. Bräunl T., Norz R., Modula-P, *User Manual. Computer Science Report*, IPVR Univ. Stuttgart, August 1992.
8. Feldmann L., Lapko V., Svjatnyi V., Trub I., Bräunl T., Reuter A., Zeitz M. Algorithmen einer massiv parallelen Simulationsumgebung Simulationsumgebung für dynamische Systeme mit verteilten Parametern, *Simulationstechnik. 10. Simposium in Dresden, September 1996*, Vieweg (1996) 519 - 524.
9. Reuter A., Grenzen der Parallälität, *Informationstechnik* (1992) 1, 62-74.
10. Svjatnyi V., Simulationsverfahren für aerogasdynamische Netzobjekte, *Jahrestagung der Gesellschaft für Informatik*, Stuttgart (1990) Band 1, 476-483.