The NORMA Language Application to Solution of Strong Nonequilibrium Transfer Processes Problem with Condensation of Mixtures on the Multiprocessor System

A.N. Andrianov[1], K.N. Efimkin[1], V.Yu. Levashov[2], and I.N. Shishkova[3]

[1] The Keldysh Institute of Applied Mathematics, Russia,
efi@a5.kiam.ru
[2] Institute for High Performance Computing and Data Bases, Russia
levashov@mcsa.ac.ru
[3] Moscow Power Engineering Institute (Technical University), Russia
irina@mcsa.ac.ru

Abstract. The application of NORMA language for problem of strong nonequilibrium transfer processes is discussed. The parallel algorithm for problem solution is created. This algorithm was realized on the multiprocessors system.The result of calculation and efficiency of program are presented.

1 Some Properties and Facilities of the NORMA Language

The declarative NORMA language [1]- [3] formalised the mathematical specification resulting from discretization of continuous differential equations. Thus this is the language of extremely high level and a specification of a computational task is turned into an executable program automatically by the translator-synthesiser for the NORMA language. This language was created to specify generic grid-based solutions to problems in applied mathematics but the area of its application turned out to be wider. Now the NORMA language is at the stage of practical use.

Note that the specification of a task solution in NORMA mentions only those rules (constraints) which must be met by the values of the variables and besides specification has no embedded memory representations and few of usual elements of programs (e.g., no control statements). Only the Norma translator needs to know about memory, processors, caches and the other hardware paraphernalia that make most programs in general purpose programming languages so hard to port to new computing environments.

It is important to note that there are no extra dependencies in the NORMA specification though they are usually imposed in programming especially at the stage of algorithm optimisation. These links often limit the possibilities of parallelising. For instance the construction COMMON in the Fortran language or indirect addressing usually limits automatic parallelization of the programs. From

V.N. Alexandrov et al. (Eds.): ICCS 2001, LNCS 2073, pp. 502–510, 2001.

this point of view the NORMA language has one more advantage: it is the language with single assignment. This fact is known to be very important for automatic parallelising.

The scheme for parallelising a solution used during a NORMA translation is briefly as follows. A graph of data dependencies is built after their analysis. A level-parallel graph of the algorithm is constructed to satisfy all dependencies with the maximum possible (ideal) parallelism. That graph is projected onto the architectural model of the target computer system. In constructing a projection, the memory model (distributed memory or shared memory), the number of the processors, and various component bandwidth factors are taken into consideration. Different optimizations are performed, of various kinds to attempt to solve performance problems related to the granularity of parallelism, load balancing on all available processors etc.

From our point of view, instead of changing a sequential version of a program to a parallel one or directly designing a parallel program, there is a more promising third way when the original statement of a problem may be realized as both parallel and sequential variants. This third approach is based on a key principle: *when formulating a new task do not impose extra constraints - later you may face an environment where they cause inconvenience and inefficiency.* Fortunately, mathematical specifications almost always adhere to this principle.

The NORMA language required new algorithms for translation and parallelising. The major results are given in [4]. Norma language compiler can generate target parallel programs in Fortran with PVM or MPI libraries or OpenMP Fortran or serial Fortran programs.

Some key features of the NORMA language are named below.

NORMA contains features for both *common* mathematical notions (e.g. integer, real numbers, vectors, matrices, functions etc.) and the notions *typical* for the given application domain (e.g. grid, index space, grid function, iteration on time and space).

In NORMA the notion *domain* represents an index space. It contains integer sets $i_1, \ldots, i_n, n > 0$ each of them is a co-ordinate of a point in n- dimensional index space. A unique index name is given to each coordinate axis of n-dimensional space. Domain may be *conditional* and *unconditional*. Conditional domain consists of the points from index space which number and coordinates may be changed depending on meeting the *conditions on domain*. An unconditional domain consists of a fixed number of points in an index space at coordinates that can be determined during translation.

A *one-dimensional domain* sets a range of points along some coordinate axis of index space, for example: **RegionK: (k=1..15)**. A *multidimensional domain* is a domain product built by operation "**;**". For example, two-dimensional domain **Oij** is a product of domains **Oi** and **Oj**: **Oij: (Oi;Oj)**, where domains **Oi** and **Oj** can be declared as **Oi: (i=1..N)** and **Oj: (j=2..M)**. Possible *modification* to a domain includes adding or deleting some points and changing the range. Domain may be *conditional* and *unconditional*. Conditional domain consists of the points from index space which number and coordinates may be changed

depending on meeting the *conditions on domain*. An unconditional domain consists of a fixed number of points in an index space at coordinates that can be determined during translation.

In NORMA *scalar variables* are simple variables but *variables defined on domain* are vectors, arrays and matrices. Declaration of the variable sets its type - **REAL**, **INTEGER**, or **DOUBLE** - and, if it is a variable on a domain, indicates the domain of points where variable values may be computed. For example, declaration **VARIABLE First**, **Last DEFINED ON Oij** defines variables **First**, **Last** on domain **Oij**; it means that the values may be assigned to these variables in every point of domain **Oij** for $i = 1, \ldots, N, j = 2, \ldots, M$.

Calculating formulae obtained by technical expert are usually written in the form of *relations*. For example, calculating formulae for the solution system of linear equations $Ax = B$ has the form:

$$
\begin{array}{ll}
m_{0,i,j} = a_{i,j}, & j = 1, \ldots, N, i = 1, \ldots, N; \\
r_{0,i} = b_i, i = 1, \ldots, N; & \\
m_{t,t,j} = m_{t-1,t,j}/m_{t-1,t,t}, & j = 1, \ldots, N, i = 1, \ldots, N; \\
r_{t,t} = r_{t-1,t}/m_{t-1,t,t}, & i = 1, \ldots, N; \\
m_{t,i,j} = m_{t-1,i,j} - m_{t-1,i,t} * m_{t,t,j}, & j = 1, \ldots, N, \\
 & i = 1, \ldots, t-1, t+1, \ldots, N; t = 1, \ldots, N; \\
r_{t,i} = r_{t-1,i} - m_{t-1,i,t} * r_{t,t}, & i = 1, \ldots, t-1, t+1, \ldots, N; t = 1, \ldots, N; \\
x_i = r_{N,i}, & i = 1, \ldots, N;
\end{array}
$$

Extract from the NORMA program is given below:

Example of a NORMA Program

```
Ot:(t=0..n) . Oi:(i=1..n) . Oj:(j=1..n).
Oij:(Oi;Oj) . Otij:(Ot;Oij) .
Oti:(Ot;Oi) . Otij1:Otij/t=1..n.  Oti1:Oti/t=1..n.
        DOMAIN PARAMETERS n=10.
VARIABLE a DEFINED ON Oij.   VARIABLE m DEFINED ON Otij.
VARIABLE b DEFINED ON Oi.   VARIABLE r DEFINED ON Oti.
        INPUT a ON Oij,  b ON Oi.
        OUTPUT x ON Oi.
FOR Otij/t=0 ASSUME m = a.
FOR Oti/t=0 ASSUME r=b.
Otij11, Otij12:Otij1/i=t.   Oti11, Oti12:Oti1/i=t.
FOR Otij11 ASSUME m = m[t-1,i=t]/m[t-1,i=t,j=t].
FOR Oti11 ASSUME r = r[t-1,i=t]/m[t-1,i=t,j=t].
FOR  Otij12 ASSUME m = m[t-1]-m[t-1,j=t]?m[i=t].
FOR Oti12 ASSUME r = r[t-1]-m[t-1,j=t]?r[i=t].
FOR Oi ASSUME x = r[t=n].
```

Necessary computations are specified by **ASSUME** operator:
FOR *domain* **ASSUME** *relation*.

This operator is a key-feature of the NORMA language. The relation gives the rule for computing the variable's value from the left part by the values of

the variable from the right part. It also gives the index dependencies between the variables. Values for the left variable must be computed at all points of the domain. The rule for each computation is defined very precisely but the computations do not need to occur where given. The program does not specify the mode (parallel or serial) or order for computations. It just tells what value relations must be preserved. Specific computations must be done only soon enough to determine values that depend upon them.

Indices with no offsets in the formulae notations may be omitted because they are automatically restored by the translator in analysis of the program.

The conditional domain definition, **Otij11**, **Otij12 : Otij1/i=t** determines two disjoint sub-domains **Otij11** and **Otij12**. The first consists of points from domain **Otij1** where condition **i=t** is true; the second, points where $\mathbf{i} \neq \mathbf{t}$. In general, a condition is a logical expression. The differences in two given above ways of calculating formulae representation are only in the form (index representation, linearity of the specification) but they are equivalent in their contents.

2 Description of the Application

Norma language and Norma system was used to create parallel program implementation for the problem of strong nonequilibrium transfer processes with condensation of different mixtures on the surfaces.

Continuum media methods give a good description of different phenomena in conditions which are characterized by small deviation from the state of thermodynamic equilibrium. By nonequilibrium growth the study of transfer processes should be made by the help of molecular kinetic theory basing on the Boltzmann equation.

Boltzmann kinetic equation (BKE) for two-dimensional non-steady statement for two- component mixture has form:

$$\begin{aligned} \frac{\partial f_A}{\partial t} + \xi_x \frac{\partial f_A}{\partial x} + \xi_y \frac{\partial f_A}{\partial y} &= J_{AA} + J_{AB} \\ \frac{\partial f_B}{\partial t} + \xi_x \frac{\partial f_B}{\partial x} + \xi_y \frac{\partial f_B}{\partial y} &= J_{BB} + J_{BA}, \end{aligned} \tag{1}$$

where $f = f(x, t, \xi)$ is velocity distribution function, $\xi(\xi_x, \xi_y, \xi_z)$ - molecular velocity, t - time, x, y - Cartesian co-ordinates, J_{AA} - collision integral describing interactions between molecules of component A, J_{BB} - collision integral describing interactions between molecules of component B, J_{AB} and J_{BA} - between A and B molecules.

In writing the expression for each collision integrals we have used the notations introduced by [5].

$$J = \iint\limits_{\Omega} \iiint (f' f_1' - f f_1) \, |\xi - \xi_1| b d\Omega, \quad d\Omega = db \, d\epsilon \, d\xi_1. \tag{2}$$

The method of direct numerical solving of Boltzmann equation [5] is used which, the authors believe, is one of the most correct methods in kinetic solving of the processes characterized by strong nonequilibrium. The Boltzmann kinetic equation, from the physical viespoint, adequately describes gas flows with high deviation from local thermodynamic equilibrium. No additional suppositions about the shape of its solution or simplifications of the equation itself, that may lower the accuracy of the physical model, are made. Current status of interphase energy, momentum, mass transfer in strong vaporation- condensation of pure vapor is sufficient for calculation of various one-dimensional problems. Two- and three-dimensional problems have been investigated not so comprehensively as for one-dimensional statement. However the investigation of transfer problems namely in many-dimensional statement can have the principal meaning. The paper assumes obtaining the macroparameters field and velocity distribution functions of molecules along the whole flow area. They present independent scientific interest and provide understanding of nonequilibrium gas flows features and revealing the role of molecules interaction in a rarefied environment. Certain attention will be paid to solving of one kinetic equation in the case of pure gas or the kinetic equation system written for different components in the case of gas mixture with calculating of collision integrals considering interaction of particles of different nature.

The method of direct numerical solving of the Boltzmann kinetic equation developed by F.G.Tcheremissine and V.V.Aristov [5]. The direct numerical solving of the Boltzmann equation presupposes introduction of a fixed grid in the velocity and physical space. Transition from constantly changed values to a set of discrete values leads to a system of a large number (of the order of some hundreds or thousands) integro-differential equations. Partial derivates are replaced by their finite difference analogues:

$$\begin{cases} \dfrac{\Delta f_A^k}{\Delta t} + \xi_x^k \dfrac{\Delta f_A^k}{\Delta x} + \xi_y^k \dfrac{\Delta f_A^k}{\Delta y} = J_{AA}^k + J_{AB}^k, \\ \dfrac{\Delta f_B^k}{\Delta t} + \xi_x^k \dfrac{\Delta f_B^k}{\Delta x} + \xi_y^k \dfrac{\Delta f_B^k}{\Delta y} = J_{BB}^k + J_{BA}^k, \end{cases} \tag{3}$$

were k - is number of point in velocity grid.

This system is resolved by an iterative procedure. Calculation of the multidimensional collision integral is made by the Monte-Carlo technique which can be improved by the use of randomized uniformly distributed sequences instead of usual random nodes. The used method is accepted worldwide. It allows one to obtain solving of steady and non-steady many-dimensional problems with complex boundary conditions with sufficient precision and to reveal delicate features of nonequilibrium flow by different potentials of molecules interaction.

3 Results of Computation

Solution results for above described problem were obtained due to the NORMA system application on the system with distributed memory multiprocessers (two Alpha 21264/667MHz in node, memory 1Gb in node, SAN Myrinet to communication, 64 nodes). Norma program was compiled in Fortran with MPI library. Problem size (dimension of grid) is 100 x 100 x 18 x 18 x 9.

For measurements purpose we have used four variants of processors configurations: 20 processors (20 x 1 in line and 5 x 4 in matrix) and 50 processors (50 x 1 in line and 10 x 5 in matrix). Table 1 shows times (in seconds) of computations that were obtained for 100 iterations during solution process.

Table 1. Times (in seconds) of computations for 100 iterations

Processors Time	20(20x1)	20(5x4)	50(50x1)	50(10x5)
Total time	265.61	253.66	142.81	119.88
Communication	30.83	19.19	17.19	10.20

Communication overhead are results of point-to-point communications and all-to-all communications. Volume of all-to-all communications is about 2,2 Mb in each iteration.

We hope, that the approach based on usage of Norma system, can appear useful for creation of mobile effective parallel programs for solution of a practical mathematical physics problems.

Solution results of the problem about gas mixture flow are presented in Fig. 1 - 4. Calculation domain is 50x50 mean free paths of nitrogen molecules at temperature $T_0 = 300$ K and numerical density $n_0 = 2.42 \cdot 10^{17}$. The gas mixture enter in investigated domain through opening and flow out through others boundary surfaces. Maximum value of nitrogen density and helium density in Fig. 1, 2 is $0.50 n_0$.

It should be noted that results were got as solutions of the Boltzmann equation for gas-gas mixture in nonequilibrium problem. The presented pictures are illustrating that flows of mixture components are different. This result is caused by nitrogen and helium molecule interactions.

4 Acknowledgment

The study is supported by Russian Foundation for Basic Researches, Grants 00-02-16273 and 01-01-00411.

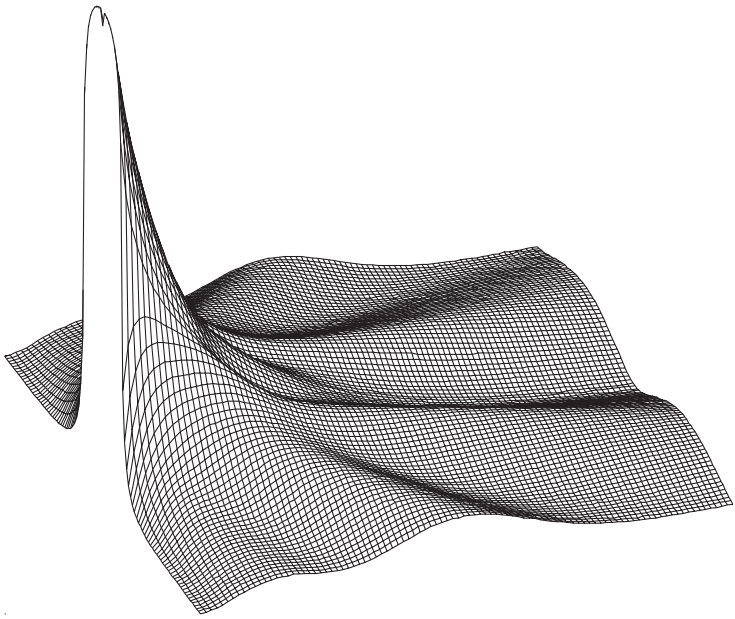

Fig. 1. Nitrogen density

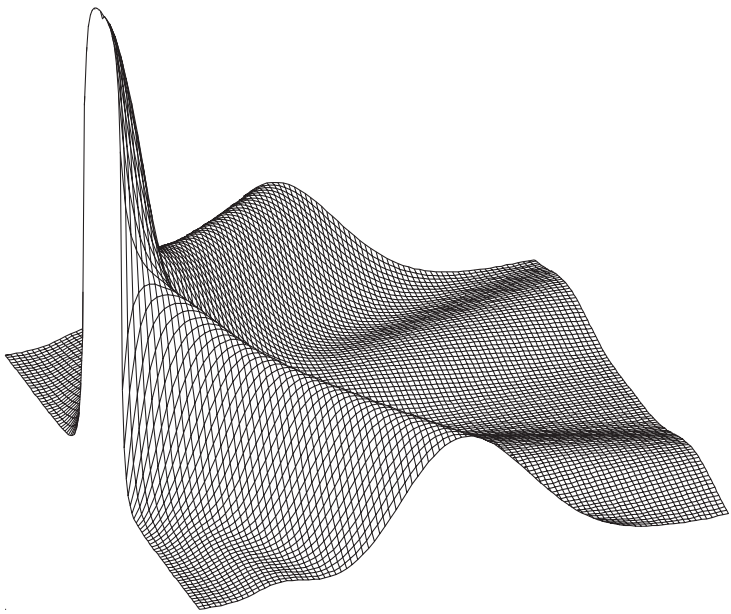

Fig. 2. Helium density

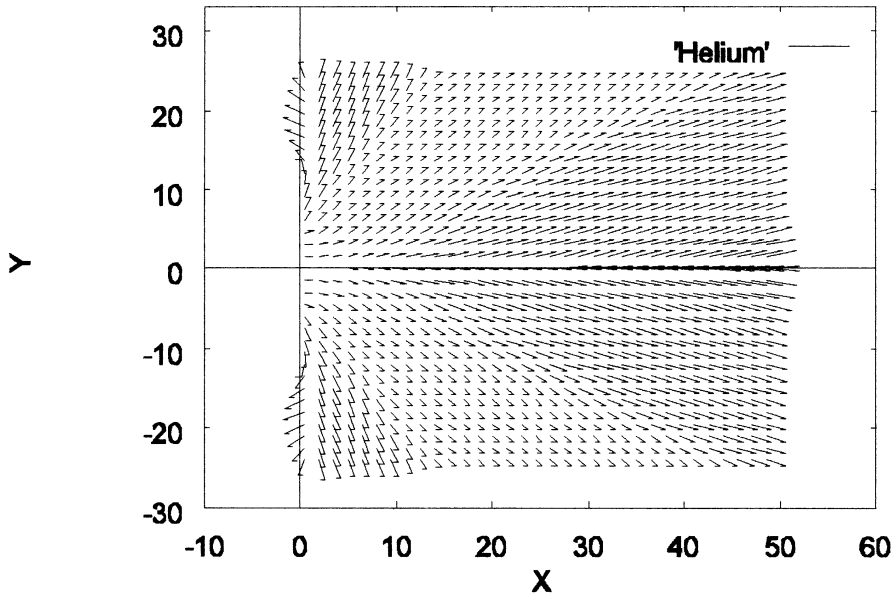

Fig. 3. Nitrogen velocity

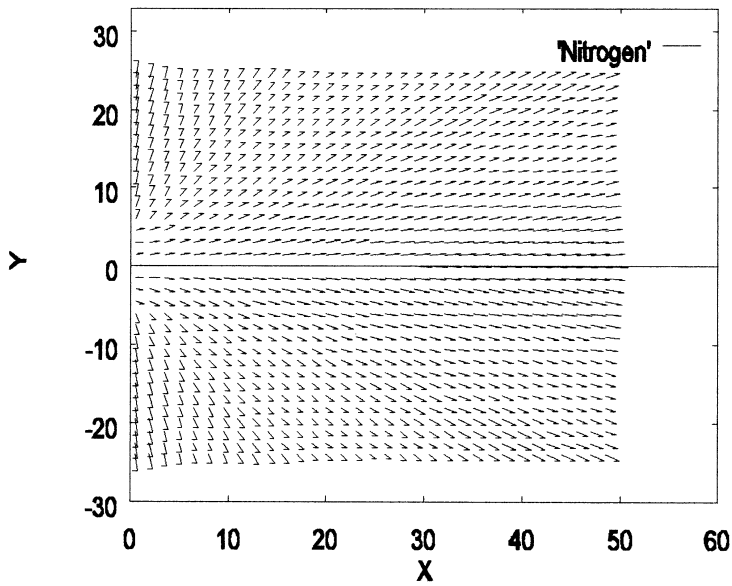

Fig. 4. Helium velocity

References

1. A.N. Andrianov, K.N. Efimkin, I.B. Zadykhailo. Nonprocedural language for mathematical physics. Programming and Computer Software, v17, 2(1991), pp. 121-133
2. A.N. Andrianov, A.B. Bugerja, K.N. Efimkin, I.B. Zadykhailo. The specification of the NORMA language. Preprint of Keldysh Ins. of Appl. Math., Russian Academy of Sc., 120(1995), pp.1-50. (in Russian).
3. I.B. Zadykhailo, K.N. Efimkin. Meaningful terms and new generation languages (the problem of stability, friendly interface and adaptation to execution environment). Information technologies and computational systems, 2(1996), pp. 46-58. (in Russian)
4. A.N.Andrianov. The synthesis of parallel and vector programs by the nonprocedural Norma specification. Ph.D., Moscow, 1990, 131 p. (In Russian).
5. Aristov V.V., Tcheremissine F.G., The direct numerical solving of the kinetic Boltzmann equation., Moscow: Computing Center of the Russian Academy of Sciense. 1992.