SESSION E

Thursday, October 22, 1992

8:30 a.m.

High-Performance Computing

Papers:

1. Using High-Performance Distributed Computing
   - David Forslund (LANL)

2. Visualization Environment for Engineering Science
   - Larry Schoof (SNL)

3. Language Issues in Parallel Computing
   - Chris Hendrickson (LLNL)

4. MEDUZA: A Method for Numerical Gas-Dynamic Calculations on Irregular Meshes
   - Liudmila Nesterenko (Arzamas)

5. Computer Simulation at the VNIEF
   - Ivan Sofronov (Arzamas)

SESSION E

High-Performance Computing

Using High-Performance Distributed Computing - David Forslund (LANL)

Many of the projects in the ACL involve cooperative computing between several different systems. This might take the form of simply processing data generated by the Connection Machine to linking together over the HIPPI network two different codes running on more than one supercomputer. For example, for the Global Climate Grand Challange, we have implemented a distributed application where a general circulation model for the ocean is computed on the CM-2 while the atmospheric general circulation model is simultaneously run on the Cray. Information at the ocean surface involving wind velocities, salinity and temperatures are exchanged between the code every cycle or every few cycles to couple the applications together. While we await higher level protocols to function between these two disparate machines, we are using raw HIPPI to exchange the information in single block transfers. We have used PVM to control and synchronize the data communications between the two machines. Because the CM-2 HIPPI channel is actually managed by a separate computer on the CM-IO bus, the control can be somewhat complicated, and as a result, the latencies are quite high.

To start the process, we initialize the PVM daemons on the CM-2 frontend, on the Cray and on the CM-2 HIPPI computer. The CM-2 HIPPI computer is attached to the CM-2 I/O Bus and can send or receive data over that bus from either the CM-2 memory or the CM-2 Datavault. The distributed application is controlled from the CM-2 front end, which initializes tasks on the Cray and on the CM-2 HIPPI computer. We have used two different techniques to transfer the data to try to hide latency. In the first case, the CM-2 writes its ocean surface data to the CM-2 Datavault, asynchronously notifies the Cray and then the Cray reads the datavault through the CM-2 HIPPI computer when it is ready to do so. The process then reverses itself to send data back to the CM-2. The data vault is used to simulate asynchronous I/O on the CM-2 so that both computers don't have to synchronize in order to send the data. The price to be paid, however is the additional transfer of data over the CM-2 I/O bus. Although the two systems need to be synchronized, we see much faster overall performance when we send the data directly between memory on the two machines. A significant portion of the time, however, is spent in transposing the data on the CM-2 to make it understandable by the Cray. We see no degradation of performance over the fiber-extenders we have and see much enhanced performance when sending data between two Y/MP's than between the CM-2 and the Y/MP. Sometime in FY93 we hope to demonstrate this capability using the CNRI sponsored CASA Gigabit network between Los Alamos, San Diego Supercomputer Center, JPL, and CalTech.

We also are running applications with toolkits from Parasoft (EXPRESS) and from Isis Distributed Systems (ISIS). The former will be used to link together applications in the CASA Gigabit project, and the latter is especially useful in modeling event driven systems and has

important capability for high performance by pipelining messages (and thus hiding latency) and for programs that must run in the face of machine failures. Both are most useful for coarse-grained simulations for which the message passing overhead is not too severe.

Proper utilisation of the large massively parallel machines that are coming on line in the 90's will require robust distributed computing tools over high speed networks combined with advanced visualisation techniques. We anticipate continued development in this area as more applications and most of the Grand Challenges begin to use these tools over local and wide area gigabit networks.

SESSION E

High-Performance Computing

A Visualization Environment for Engineering Sciences

Larry A. Schoof
Applied Visualization Group, Org. 1425
Sandia National Laboratories
Albuquerque, NM

Abstract

This presentation describes our efforts to implement a production visualization environment for use by a large community of engineering scientists at Sandia. The proposed environment makes use of a visualization server concept to provide effective, interactive visualization to the user's desktop. Our prototype environment includes a visualization server machine connected via high speed communications to a central supercomputer and network storage facility. Experiences using this environment will be summarized. Benefits of using the visualization server approach will be discussed, and some thoughts regarding desirable features for visualization server architectures will be addressed. A brief discussion of the software environment, which is based on the Application Visualization System (AVS) software, will be included, along with a general status report on our progress towards completion of the production environment.

SESSION E

High-Performance Computing

Chris Hendrickson

Abstract for Russian Visit - "Language Issues in Parallel Computing"

Abstract

Functional languages hold the promise of allowing ease of programming massively parallel computers that imperative languages such as FORTRAN and C do not offer. At LLNL, we have initiated a project to write the physics of a major production code in SISAL, an applicative language developed at LLNL in collaboration with researchers throughout the world. We are investigating the expressibility of SISAL, as well as its performance on a shared-memory multiprocessor, the Y-MP. An important aspect of the project is that the SISAL modules call FORTRAN modules, and are callable by them. This obviates the rewriting of 70% of the production code that would not benefit from parallel execution. The presentation will include:

(1) A brief comparison of functional languages and imperative languages.

(2) A discussion of recent research that has overcome some major shortcomings of functional languages.

(3) A discussion of the experience in programming the production code in SISAL.

This work was performed under the auspices of the United States Department of Energy by Lawrence Livermore National Laboratory under contract #W-7405-ENG-48.

SESSION E

High-Performance Computing

MEDUZA: A Method for Numerical Gasdynamic Calculations on irregular meshes

Volkov S.G., Malshakov V.D., Nesterenko L.V., Sofronov I.D.

This report presents one of the methods for solving two-dimensional problems in computational physics using irregular meshes. The methods of this type appeared within Lagrangian approach as a base for implementing "halt-free" programs for flow calculations involving strong deformations, jets and vortex motion. A method called MEDUZA uses an irregular polygonal mesh obtained as a matching a triangular one built for establishing neibourhood relations between points of discrete set replacing the computation region. The major property of MEDUZA is the variation of point neibourhood when computations proceed, so each mesh cell is loaded with data on neibouring points in addition to gasdynamic parameters.

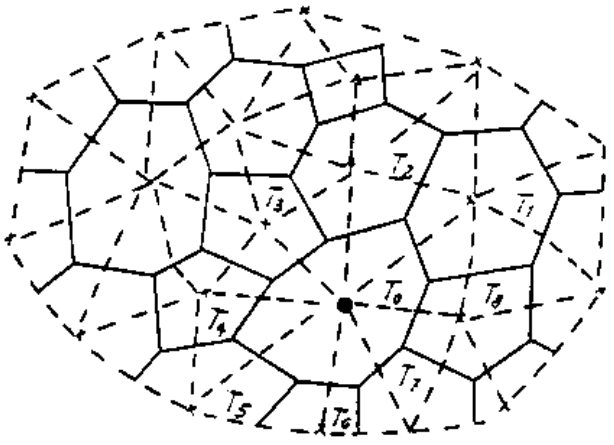
MEDUZA: A METHOD FOR NUMERICAL GASDYNAMIC CALCULATIONS ON IRREGULAR MESHES.

Volkov S.G., Malshakov V.D., Nesterenko L.V., Sofronov I.D.
VNIIEF, 37 Mir Avenue, Arzamas-16, Nizhny Novgorod Region, 607200.

In the late 60-s numerical methods for solving two-dimensional problems in computational physics using irregular meshes appeared within Lagrangian approach to serve a base for implementing "halt-free" programs for flow calculations involving strong deformations, jets, and vortex motion. Many methods and programs rely upon this approach. For more details on related papers refer to [1,2]. This report presents one of the methods of this class.

MAJOR MEDUZA FEATURES

MEDUZA MESH. Meduza [4] is designed for time-dependent two-dimensional gasdynamic calculations on an irregular polygonal mesh using an explicit scheme. The computation region is replaced by a discrete point set where each point is determined by gasdynamic parameters and data on neighbouring point states. All of the gasdynamic parameters are defined by the mesh cells, that is we have a scheme with *nondistant* mesh functions, as opposed to CREST-type methods [5]. *Neighbourhood relations* are established at early times using some principles. For example, Delaunay triangulation is performed over a given point set. A polygonal mesh *matching* a triangular one may be used for MEDUZA (Fig.1). The properties of such mesh are reported in multiple papers, for example, in /6/. The major property of MEDUZA is *the variation of point neigbourhood* when computations proceed.



Delaunay triangulation is shown by dashed lines. Solid lines correspond to the irregular MEDUZA mesh.

Figure 1

MESH EQUATIONS. The equations are approximated with intergation/interpolation technique commonly used in explicit Lagrangian methods. Without detailed description of how to obtain a difference version of gasdynamic equations written in Lagrangian form and noting that integral subscripts correspond to computation points, semi - integral subscripts represent cell vertices, next timestep values are marked with un upper zero while the current value is marked with a lower zero, one has:
for cell volume

$$V_0^0 = \sum_{i=1}^{k} S_i \frac{Y_0 + Y_{i+\frac{1}{2}} + Y_{i-\frac{1}{2}}}{3} ;$$

for specific density

$$\sigma_0^0 = \frac{V_0^0}{M_0} ;$$

the pressure is obtained from the relations

$$E_0^0 - E_0 + \frac{p_0^0 + p_0 + 2q}{2} (\sigma_0^0 - \sigma_0) = 0 ,$$

$$E_0^\circ = E\,(p_0^\circ\ ,\sigma_0^\circ)\ ,$$

$$E_0 = E\,(p_0\ ,\sigma_0)\ ,$$

velocities are given by Newton equations

$$U_0^\circ = U_0 + \frac{1}{6}\frac{\tau}{M_0}\sum_{i=1}^{k}(Y_{i-\frac{1}{2}} - Y_{i+\frac{1}{2}})\Big[(2Y_{i-\frac{1}{2}} + Y_{i+\frac{1}{2}})P_{i-\frac{1}{2}}^\circ +$$

$$+ (Y_{i-\frac{1}{2}} + 2Y_{i+\frac{1}{2}})P_{i+\frac{1}{2}}^\circ\Big]\ ,$$

$$V_0^\circ = V_0 + \frac{1}{6}\frac{\tau}{M_0}\sum_{i=1}^{k}(X_{i+\frac{1}{2}} - X_{i-\frac{1}{2}})\Big[(2Y_{i-\frac{1}{2}} + Y_{i+\frac{1}{2}})P_{i-\frac{1}{2}}^\circ +$$

$$+ (Y_{i-\frac{1}{2}} + 2Y_{i+\frac{1}{2}})P_{i+\frac{1}{2}}^\circ\Big] + \sum_{i=1}^{k}\frac{1}{3}S_i(p_0^\circ + p_{i+\frac{1}{2}}^\circ - p_{i-\frac{1}{2}}^\circ)\ ;$$

and the point coordinates for the next timestep are

$$X_0^\circ = X_0 + \tau\,U_0^\circ\ , \qquad Y_0^\circ = Y_0 + \tau\,V_0^\circ\ ,$$

where k is the number of neigbors for the point to be computed, S_i is the area of a triangle with its vertices being located at the basic point and on two consecutive cell vertices $T_{i-\frac{1}{2}}$, $T_{i+\frac{1}{2}}$; X, Y, U, V – point coordinates and velocities; E, P, σ, q, M – internal energy, pressure, specific density, mathematical viscosity, and point mass, respectively; τ – timestep.

The computations are performed on a uniform mesh without explicitly selecting physical region boundaries. The boundaries are treated according to *mixed cell* concepts including several materials separated by an interface which may be placed anywhere within the cell. For mixed point model, difficulties occur when solving an energy equation which is replaced by $2n+1$ system, where n is the mixture component number, for a multicomponent cell.

$$dE_i(p,\sigma_i) + d\sigma_i(p + q) = 0$$

$$E_i = E_i(p,\sigma_i) \qquad i=1, 2, 3, \ldots n$$

$$\sum_{i=1}^{n} M_i \frac{\sigma_i}{M} = \sigma_{av}$$

where σ_{av} – average specific point density; M– total point mass; M_i, σ_i, E_i – specific density and internal energy for i component, respectively.

This approach is commonly used both for particle-in-cell-like methods and other Euler-Lagrange techniques [5,7].

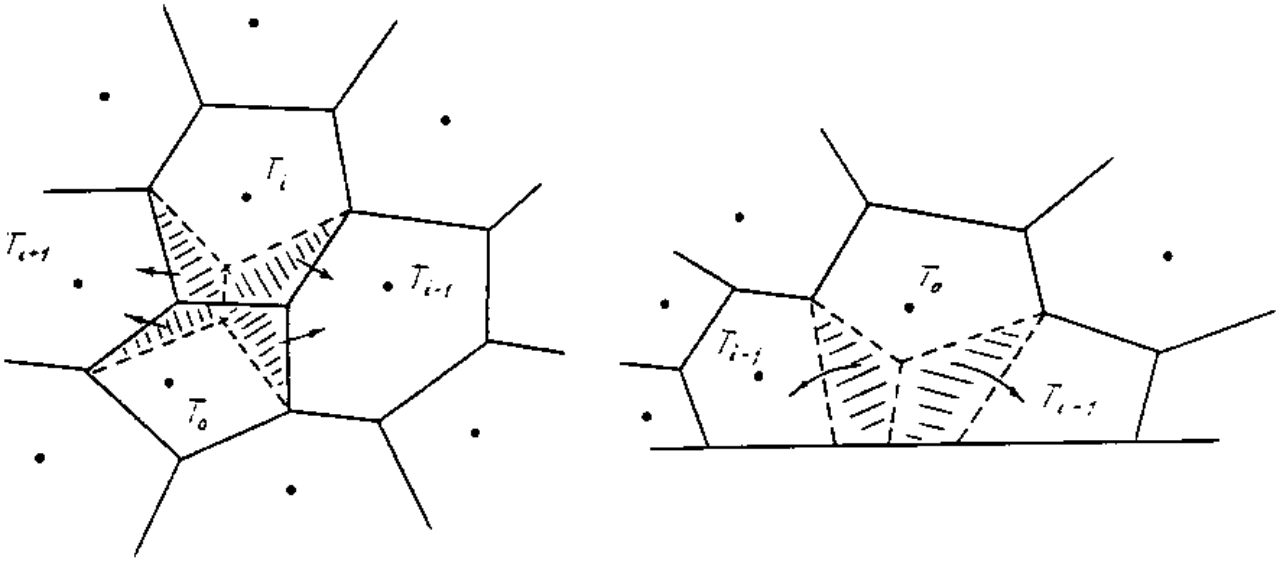
Clearly, the above system is explicitly solved only for the simplest medium component models. In a general case, iteration techniques are used to solve it. Formulas for one of the potential iterative processes are given in /5,6/.

MESH OPERATIONS. The most remarkable feature of the mesh described is represented by the operation modes which exploit irregular mesh invariance to local changes in internal mesh structure which allows the mesh operations, such as removing or introducing a new vertex, replacing point neighbors etc. Note, that such features are specific only to the class of meshes described. Each of the above operations has its *own origines*; some of them are *scheme-defined* (nondistant values), others arise from an attempt to perform computer simulations of physical processes poorly described by a pure Lagrangian approach (*strong medium deformations, jets* etc.), or may be due to the desire to improve numerical *technology level* (*pouring* points where increase in accuracy is needed or removing them from where, for example, the shock wave passed. No place is available in this paper to describe these origines in more details. Here some mesh operations only will be illustrated (Fig. 2,3,4).

Figure 2 contains two of several possible cases of point location when the neighbourhood changes. The points shown on Fig. 2a are all internal and T_3 are consecutive neigbors of

T_{i-1}, T_i, T_{i+1}. Each timestep verifies whether *the neighbourhood criterium* is valid and if it is no longer valid within the period starting from the current time, the neighbourhood changes. If the pair T_0, T_i breaks the neighbourhood relationships then T_{i-1}, T_{i+1} become neighbors. The figure shows elementary volumes (masses) participating in reinterpolation and the mesh changes. The arrows corespond to the transfer direction of mass and whatever it carries. Figure 2b illustrates a similar situation for the case where the points are near - boundary with respect to the symmetry axis.

CHANGING NEIGHBORS IN MEDUZA



Shaded volume elements participate in mesh function interpolations.

a) internal point case b) "rigid" wall case

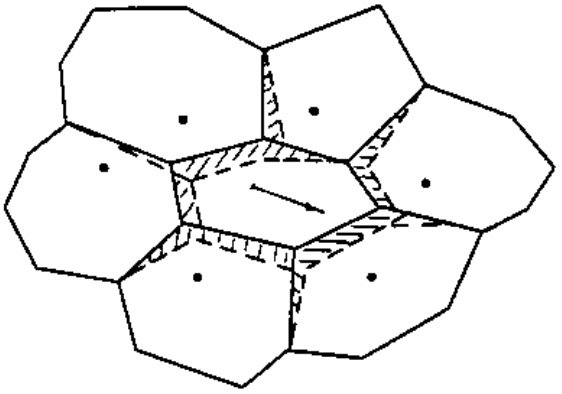
Figure 2

Figure 3 presents the point spacing which consists of moving the point in a given direction at some distance. The point neighbor tables do not change when spacing is performed while each mesh function should be reconsidered as in the case of neighbor changes.

Figure 4 contains a case of removing a point which restricts

the timestep. When a point is removed the neighbor tables change for all neighbors of this point.

MOVING APART IN MEDUZA



The dashed line shows the cell boundary locations after the point moved along the given direction.

Figure 3

REMOVING A POINT IN MEDUZA

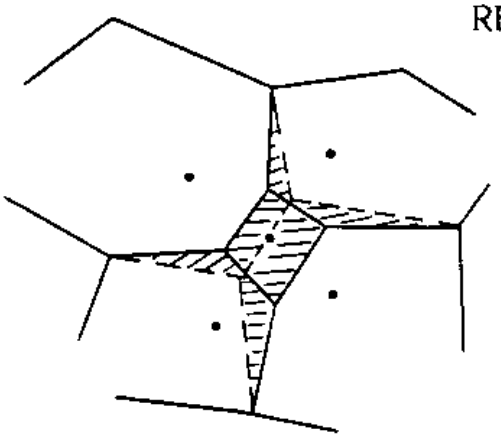


Figure 4

We shall say the following as an additional comment on figures. Actually, "halt-free" nature and high technology level of MEDUZA are just provided by the ability to manipulate the mesh without changing its conceptual structure.

MESH FUNCTION REINTERPOLATION. For any mesh change, the mesh functions are reconsidered using mass , total energy, and momentum conservation laws that are written for the vicinity of a changing point.

The mesh changes may result in *mixed points*. If a "mixed

point" participates in reconsideration then the following rule is valid: if the points being handled contain the same material component composition, then reconsideration is partially performed regarding the concentrations. If one point is "pure" and another is a "mixed" one, then values are transferred so that the "mixing zone" should remain as small as possible (which would minimize the computation diffusion). The mixed point first transfer to the pure point the material contained by the latter and if it is insufficient, the remainder is compensated by another material.

This approach appeared to be a *poor feature* of the method during the long-time program usage. However careful testing (other program computations and comparison with experiments) has dispersed our doubts.

To illustrate the MEDUZA possibilities some computational results will be given, but first we shall make some remarks on implementation.

MEDUZA IMPLEMENTATIONS

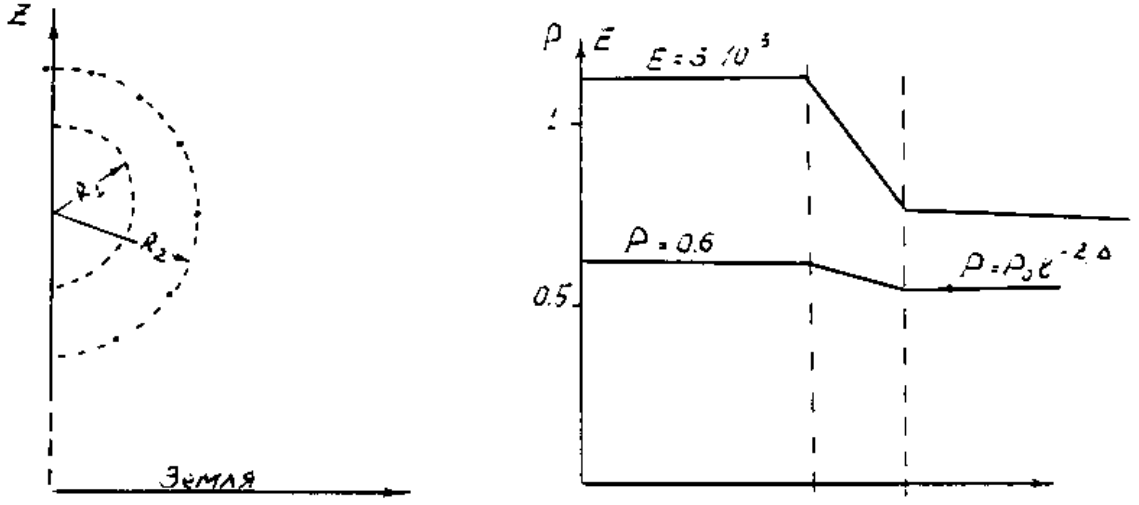
MEDUZA has been always implemented "marginally" on such computers as BESM-6, ES, Elbrus-2 or IBM PCs. It is clear intuitively, that using directly a Fortran-like language which does not support the capabilities for handling complex aggregate data, adequate to the methods and simulated medium concepts, is actually impossible. Maximum language dynamics required (inserting and removing points, an arbitrary number of mixed point components, arbitrary difference template and others are to be kept in mind) make things even worse. Even though a language includes appropriate concepts (sets, template and list pointers, memory allocation and deallocation etc.) the MEDUZA still remains relatively complex. One of the ways to decrease the software complexity is to create a multilevel structure: the lowest level carries modules depending on computer architecture and resources (memory control system, input/output etc.), which are followed by

data management routines (creation of difference template buffer, neighbor table, connected point set buffer, neighbor table for several generations) and finally multiple computational modules which were organized in view of formula considerations: calculations of pressure, cell viscosity, mesh quality verification, mesh changes etc. Such programming discipline allowed to minimize the difficulties involved in program portability and make multiple modifications. This is worth to be included in a detailed report that is still waiting for its time to appear.

COMPUTATIONAL EXAMPLES

Problem 1. Light gas bubble rise-up within a heavy one and circular vortex formation [8].

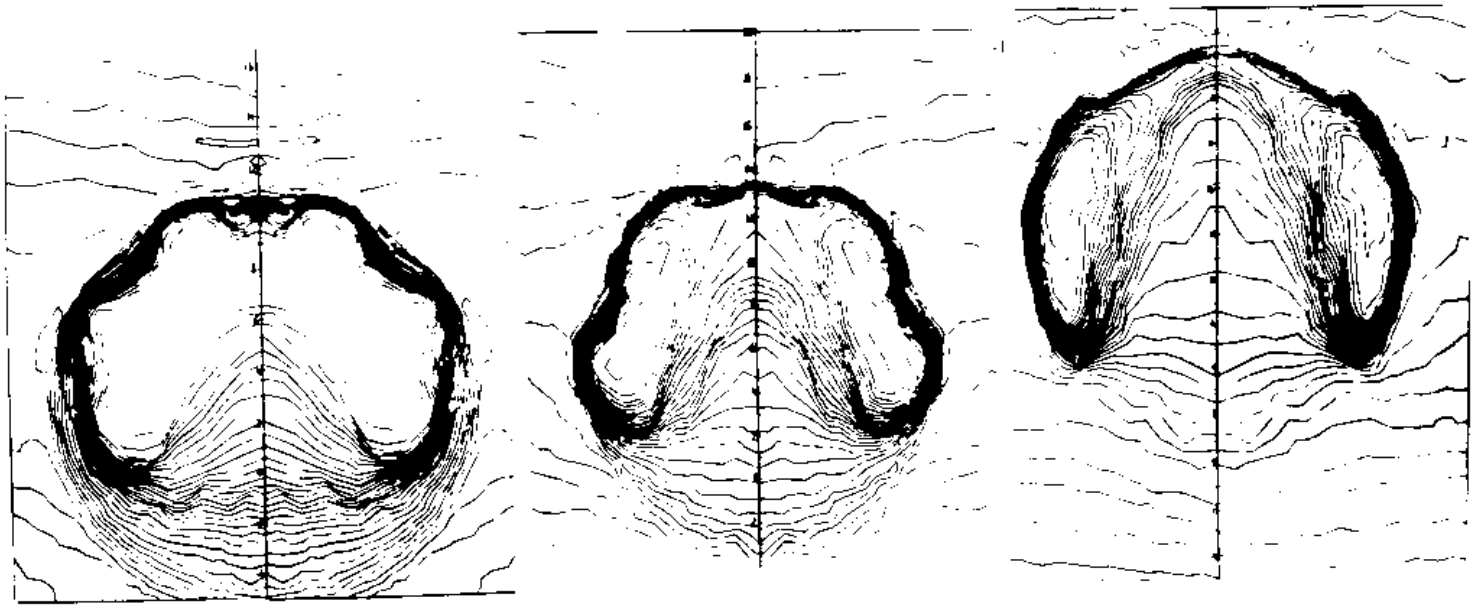
PROBLEM FORMULATION



R_1 is a sphere radius, R_2 is a transient layer sphere radius, ρ is a pressure, E is a specific energy, Δ is air nonuniformity parameter, $P(z)=P_0 e^{-z/\Delta}$. The equation of state is that of ideal gas in each region, $\gamma=1,4$

Figure 5

Cloud history, velocity field and the mesh at vortex formation time



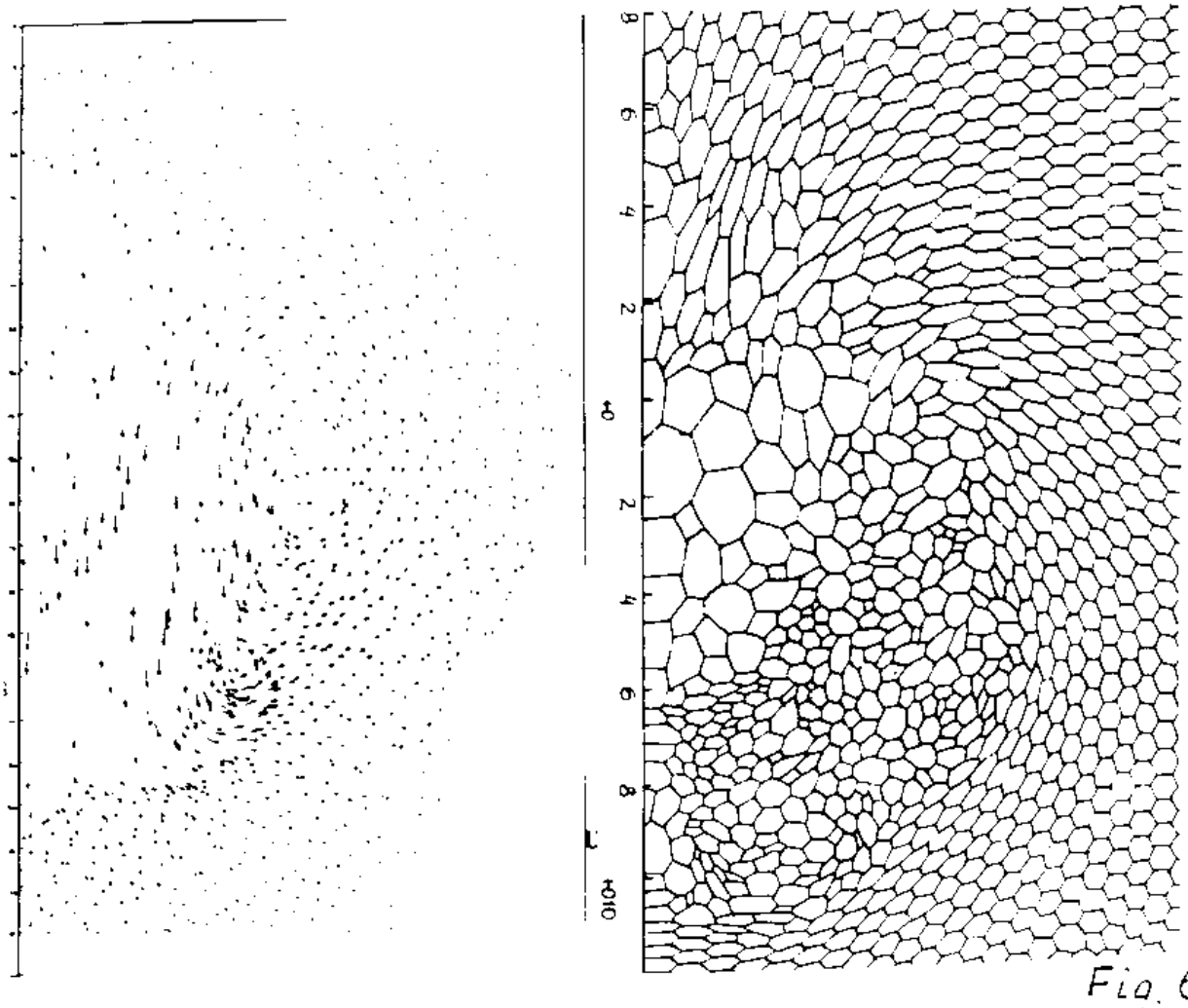


Fig. 6

Problem 2. Numerical calculation of a pinch effect [10].

PROBLEM FORMULATION

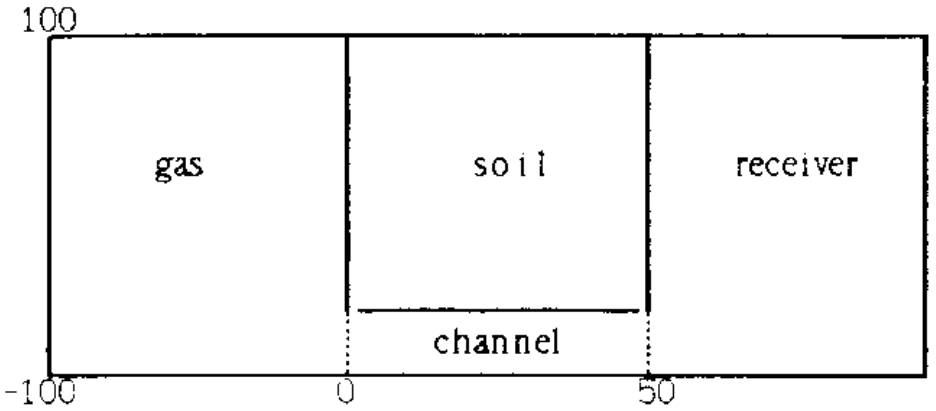
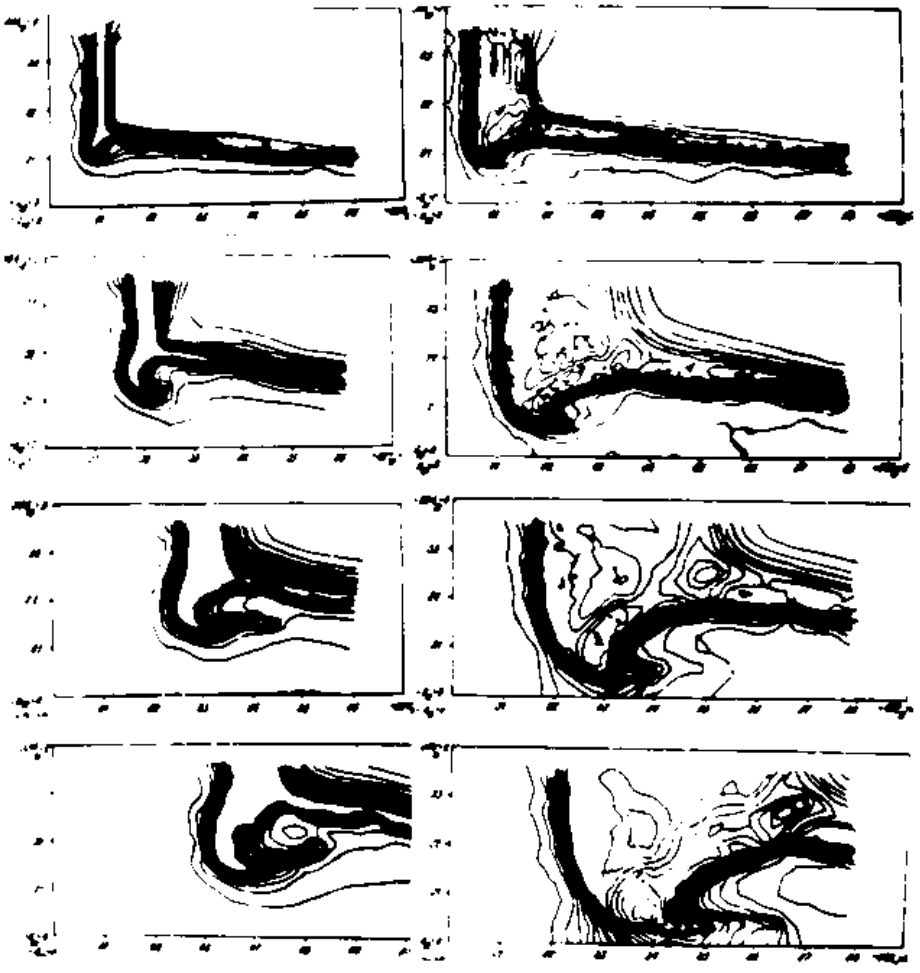


Figure 7

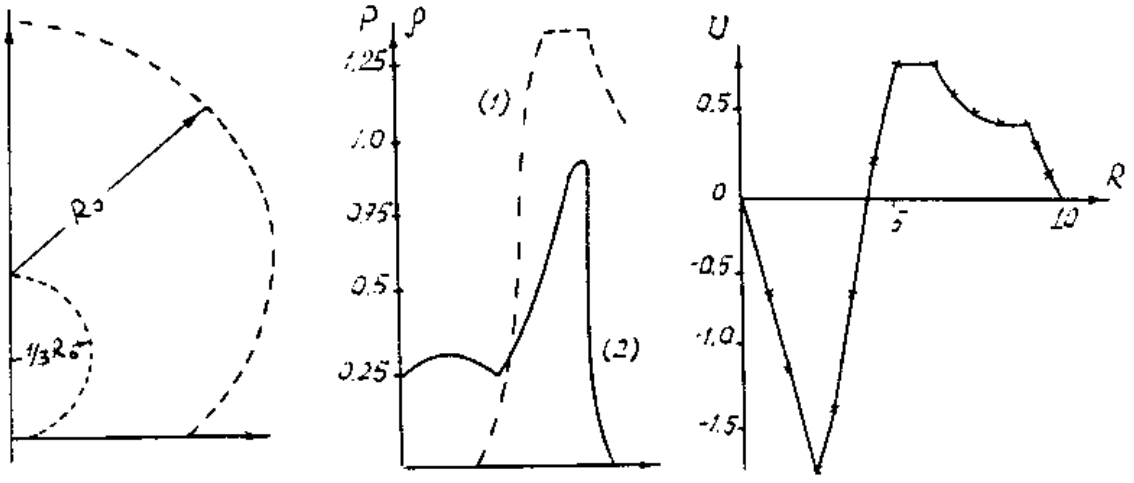


At the left γ=1,4. No pinch effect is observed. At the right γ=5. The channel is closed.

Figure 8

Problem 3. Explosion in a water-filled cavity [11].

PROBLEM FORMULATION



R_0 is the radius of spherical portion of the cavity, $2/3R_0$ is a distance from the center to the cavity bottom, $1/3R_0$ is a distance from the burst point to the bottom. (1) - density profile, (2) - pressure profile, (3) - velocity profile.

Figure 9

Velocity field for various times

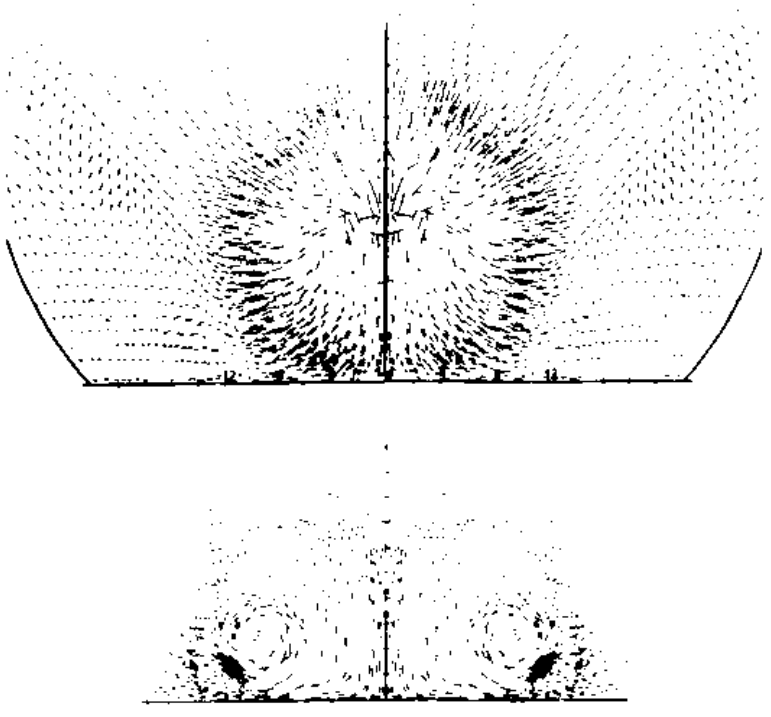


Figure 10

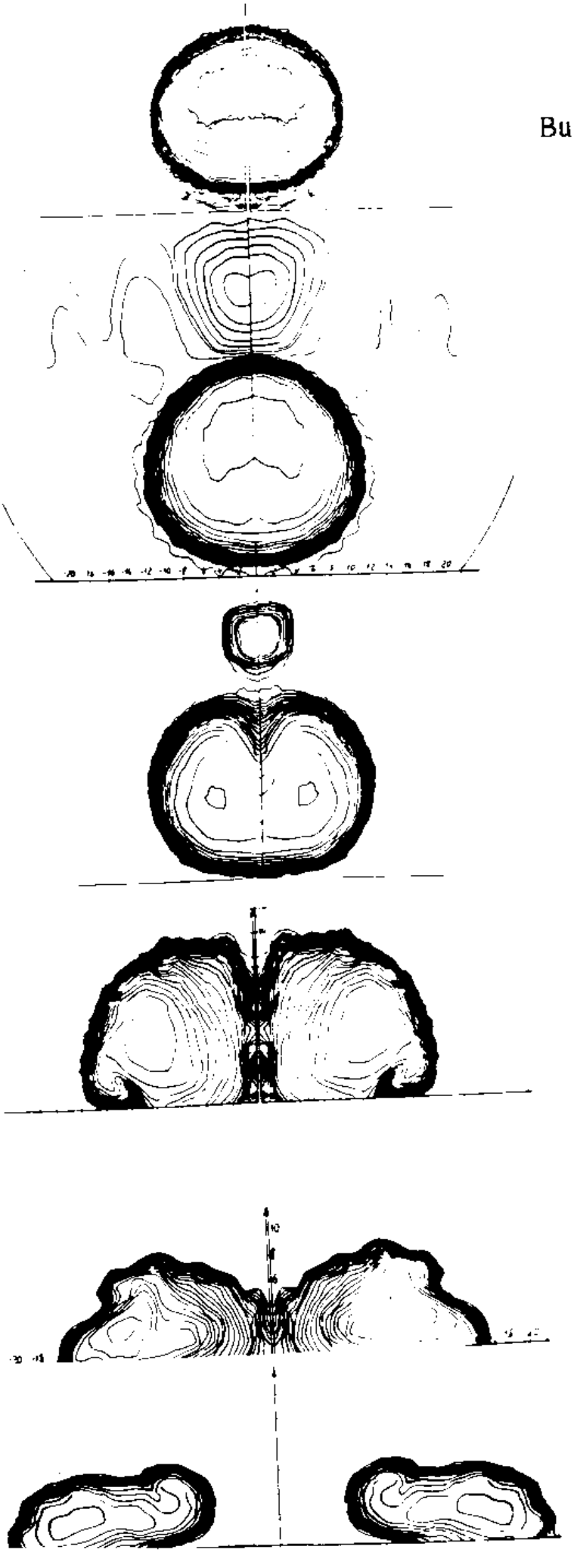


Figure 11

REFERENCES

1. Hockney R.M., Eastwood J.W. Computer Simulation Using Particles. Mc.Graw-Hill Inc. 1981
2. Belotserkovsky O.M. Numerical simulation in mechanics of solids, Moscow, Nauka, 1984.
3. Finite - Difference Techniques for Vectorized Fluid Dinamics Calculations. Edited by David L. Book. Springer Series in Computational Physics. 1981
4. Glagoleva Yu. P. et al., MEDUZA code for 2-D time-depended gasdinamic calculatins. Fundamentals. In: Numerical methods in mechanics of solids, Novosibirsk, 1972, vol. 3, no. 7
5. Bakhrakh S. M. et al. Gas Dinamic flow calculations with the concentrcion method, Proceedings of Ac. of Sci., 1981 vol 257, no. 3 p. 566 - 570
6. Franco P. Preparata, Michael Ian Shamous. Computational Geometry: An introduction. Springer - Verlag. 1985
7. Methods in Computational Physics Advances in Research and Applications. Vol.3. Fundamental Methods in Hydrodinamics. Ac.Press. 1964.
8. Sofronov I.D., Rasskazova V.V., Nesterenko L.V., The Use of Nonregular Nets for Solving Two - Dimensional Nonstationary Problems in Gas Dynamics /Edited by N.N. Janenko and Ju.I. Shokin. - Moscow: MIR, 1984. -p.p. 82-121.
9. Glagoleva Yu. P. et al. Circular vortex formation for a light gas rise-up in a heary one, Numerical methods in mechanic of solids, Novosibirsk, 1974, vol. 5, no. 1
10. Glagoleva Yu. P. et al., Numerical solution for wall channel closure by pressing gas, Numerical methods in mechanics of solids, Novosibirsk, 1976, vol. 7, no. 4
11. Adamsky V. B. et al, 2-D calculations for an explosion in a water-filled cavity having the form of truncated sphere, VANT, ser.: Mat. mod. fiz. proc., 1989, no. 3, p. 3 - 6
12. Oran E.S., Boris J.P. Numerical Simulationt of Reactive Flow. Elsevier. New-York-Amsterdam-London. 1987

SESSION E

High-Performance Computing

Computer Simulation at the VNIIEF

Sofronov I.D.

The history of the Computational Division of VNIIEF is briefly presented. Methods and programs for one-dimensional problems are summarized. The evolution of the multidimentional calculations in VNIIEF is considered and existing regular and irregular schemes are reviewed. Programs for multidimensional calculations on Elbrus-2 are presented.

The evolution and the state of computer network are examined.

Computer Simulation at the VNIIEF

I.I. Sofronov

The Computational Division of VNIIEF was created as separate structure on May 1, 1952. Of course, mathematicians had come to the Institute far earlier, but mathematical staff was then a part of physical divisions.

Currently, the Computational Division is one of the greatest mathematical institutes engaged in method developments for a wide spectrum of calculations in computational physics, their program implementation for various computer generations and, finally, solving many applications which result from designing a variety of instruments and devices with different types of nuclear and thermonuclear weapons being the most important among them. Our mathematicians were not allowed to publish their main papers in open journals because of the work specificity, and for the same reason we could not use the experience of our colleagues abroad in solving similar problems. The above said suggests that the great majority of calculational methods and computer programs are original products.

It should be noted that from the very beginning our mathematicians had to deal with applications where the amount of computations exceeded considerably the hardware capabilities. Therefore one had not only to invent a method for an application but an extremely cost-efficient method requiring minimum resources and hardware: we had not on, to write a computer program, we needed the most efficient program both with respect to the number of arithmetic operations and to the main and external memory size. Yet, the Institute was equipped with the most powerful and latest domestic computers.

A great mathematical team working forty years under such conditions created a large number of good program complexes which are our "gold fund". Designed methods and programs together with computer operation technology compensated substantially for our lagging behind the west in the computer field. Our nuclear weapons makers have always had mathematical models that are slightly inferior to those in the west and sometimes even overperformed them which played an important role in achieving nuclear parity.

I don't want to say here that our mathematicians are more gifted than those from U.S. Since more efficient methods were needed, we had to use more researchres, to choose the calculation strategy more carefully, to make the computations more cost-efficient. Eventually all this resulted in that our mathematicians spent more time and

efforts to develop and continuously improve the methods to perform the computations. In other words, final results turn out to require more human labor in our institute than abroad.

It is worth to say that computations play much more important role in our field as compared to other industries where laboratory modeling and full-scale tests are of great help when developing new designs.

Laboratory modeling allows to see whether the designer's decisions are correct before a prototype is manufactured with relatively low time and resource cost. Prototyping and later tests permit to study all operational details of units and devices for the new structure and to make necessary modifications. This is not the case in the atomic industry.

It is impossible to reproduce physical conditions occuring when nuclear devices are operated in laboratory. Full - scale tests are expensive and require many people and institutions. Full - scale experiments are severely restricted in yield, number, environmental requirements etc. The most important is that full - scale field experiments do not allow to see the behavior of many units and devices when the product is operated. One can measure only a small number of integral values which then are used to reproduce mathematically the final results.

In other words computations represent one of the principle ways to obtain data on new device performance and sometimes it is the only way. For this reason, the government officials paved a great attention to developing the mathematical basis of the weapons institutes. In addition to good technical equipment, many gifted mathematicians were involved into the atomic industry activity. Atomic industry school saw the most famous mathematicians both in the USSR and in USA.

This list includes M.Keldysh, A.Tikhonov, M.Bogolyubov, I.Yanenko, V.Vladimirov, S.Nikolsky and many others.

As was mentioned above, the computation - intensity of our applications was much greater than the hardware capabilities. For this reason, we had to adopt the computation strategy where a single problem is run simultaneously on several computers.

In the mid 1960-s VNIIEF created BESM-4 four-computer complex. Hundreds of 2-D time-dependent gasdynamic calculations were performed on this complex. After the experience in BESM-4 complex operation had been acquired, we were encouraged to design a BESM-6 computer complex.

In the 1970-s the major computational capability was represented by the BESM-6 multicomputer complex that was successfully used till the 1990-s. Its maximum configuration

consisted of nine machines though a single problem was actually run on 3 or 4 computers. Connecting a great number of computers did not reduce substantially the computation time. The experience acquired allowed to perform efficiently parallel computations on the Elbrus 1-2k multiprocessor since the 1980-s. This complex is still operated demonstrating a reliable performance. Its maximum configuration contains five processors and is used by several programs for 2-D and 3-D calculations. The startup of two Elbrus-2 multiprocessors has been our recent achievement in this field. Our experience allowed to create several program complexes that use efficiently the maximum Elbrus-2 configuration containing ten processors. Finally the multiuser computer network (MCN) should be mentioned which is being developed and improved over twenty years.

The kernel consists of several homogeneous computer complexes: Elbrus1-2k, Elbrus-2, ES-1066 and others. MCN is controlled by the ES-1066 two-computer complex which must input and output data on every task entered into the MCN, perform file functions, maintain local and global magnetic banks etc. Control computers are not directly involved into computations which are assigned to the remaining nodes. Computer networking gave us some advantages as compared to using the same machines independently. These advantages include, for example, the following:

- resources of multiple machines can be available for large computations;
- increased service life of the computer system;
- considerable main and external memory savings;
- smaller number of I/O devices are used with their types reduced;
- the personnel is considerably reduced;
- new computers can be easily introduced etc.

Essential components of the MCN are represented by intercomputer switch (IS) and data communication network (DCN). The IS provides a fast communication between different homogeneous computer complexes. It is implemented based on serial channel-channel adapters modified by our specialists. The IS includes both electric wires and high-performance fiber links. The data communication network is used to connect a great number of terminals to any machine from the MCN kernel. The DCN is implemented on SM-121 minicomputers connected via assigned and switched links. The DCN connects both closely located and remote (up to tens kilometers) terminals.

The network supports user wave search and emulates

ES-7920 standard.

The MCN has both local (for each homogeneous computer complex) disk files and a global disk file on control computers. The global disk file is accessed by each computer. The computers exchange data through these disks. The second level of magnetic file is represented by magnetic tapes. Less frequently used data is pushed to magnetic tapes for long-term storage. The Division has a rich software. The software available is oriented to various problems in computational physics.

1. The most massive calculations are represented by so called one-dimensional problems where the equations and solutions depend on a single time or space variable. The equations to be solved have the form:

$$A\frac{\partial}{\partial t}V + B\frac{\partial}{\partial x} + CV = D \qquad (1)$$

where A,B,C,D and V are coefficient matrices and free-term and unknown vectors of dimension P. The equation (1) is correctly complemented with initial and boundary conditions. Equations of the form 1) describe adiabatic gas dynamics, gas dynamics including heat conduction, strength forces, neutron transport calculated using kinetic and diffusion approximation etc.

The initial and boundary-value problem for Eq.(1) is typically formulated as follows: a fraction of space between X0 and X1 is filled with gas layers which differ in physical properties and are characterized by initial density, pressure, temperature etc. Eqs (1) are complemented with various equations describing medium properties. We should primarily mention here equations of state, paths, neutron constants etc.

One-dimensional calculations use a variety of numerical methods. gasdynamic calculations are more frequently done using explicit schemes. The heat conduction equation for high-temperature gas dynamics is solved with implicit schemes. The kinetic equation often uses running schemes. For all one-dimensional calculations, an integrated applications package called "Odnomerny Complex" (one-dimensional complex) is created which includes some tens of modules implementing various methods for a wide spectrum of problems.

The most massive calculations are represented by the following classes:

1. Adiabatic gasdynamics, "Crest", Godunov and implicit schemes for some classes of problems are used.

2. High-temperature gas dynamics where heat conduction should be included, particularly, with a nonlinear

conductivity.

3. Gas dynamics with strength forces and detonation included.

4. Multicomponent and multiphase gas dynamics.

5. Multitemperature gas dynamics.

6. Gas dynamics with magnetic field and propagation of various radiation types included.

7. Gas dynamics with heat conduction, neutron propagation, chemical and neutron-nuclear reaction kinetics included.

All computation modules use the same service routines which facilitates new method and program developments. Integrated systems of equations of state, paths, neutron constants and others are created within the Odnomerny Complex. Data reported in domestic and foreign literature along with experimental results from our Institute were used to create the systems of equations of state, paths and neutron constants. FORTRAN serves the basic programming language. Service blocks are written in Assembler to achieve the peak efficiency of the Complex. It seems possible to abandon Assembler as the computer perfomance will increase. However using the low-level language was reasonable in view of low processor performance and main memory capacity though it affects the program portability.

The Odnomerny Complex runs successfully on various computer types for many years: the programs it contains are used for several tens of hunders of problems. This means that the efforts to write programs carefully and to achieve the peak efficiency are reasonable. Usually the problems to be solved contain several tens of computation regions, hundreds (up one thousand) of points and require hundreds or thousands of timesteps. The computation time ranges from tens of minutes to tens of hours for ES-1066.

Since 1978 the papers on "Odnomerny Complex" by various authors can be found in the review entitled 'Problems of Atomic Science and Engineering. Series: Methods and Programs for numerical calculations in Computational Physics".

2. The greatest fraction of computation time is taken by two-dimensional calculations that is by those where the system of equations and the solution depend on two space coordinates and one time coordinate. These systems have the form:

$$A\frac{\partial U}{\partial t} + B\frac{\partial U}{\partial x} + C\frac{\partial U}{\partial y} + DU = \mathcal{E} \qquad (2)$$

Here the coefficients and unknowns have the same meaning as in Eq.(1). We assume Eq.(2) to be correctly complemented with boundary and initial conditions along with all necessary equations of state, paths and constants.

Lagrangian and Langrangian/Eulerian coordinates are the most popular, though Eulerian and arbitrary mobile coordinates are also used.

The greatest part of two-dimensional methods use the so called regular grid where data is ordered in the form of a two-dimensional array. However for many years irregular grids are successfully operated which have no fixed algorithm for neighbor selection.

For neighbor selection an original algorithm considering the point location at a given time and in a given space is used at each point of the irregular grid. Irregular approach has some advantages in a number of cases and in successfully used for a variety of problems.

In practice, the irregular approach components are encountered in many regular grids in the form of interfaces between two different regular grids, or separate points, or extended template on highly nonorthogonal grids.

Practical two-dimensional calculations are performed with several program complexes containing multiple special programs such as initial data computation, computational modules, grid operation routines, computation control routines, result handling routines and many others. We did not succeed in creation of an integrated service system for all methods as we did in one-dimensional case. Currently several program complexes are run, each implementing multiple methods. The lack of an integrated service system complicates data transfer between programs. However an agreement on standard exists "de facto" which determines problem information type to be transferred to another program complex.

BESM-6 was our basic machine for a long time. Of course, this is not intended for two-dimensional calculations, however higher performance computers were not available in the former Soviet Union till late 1970s. Manual programming allowed to write highly efficient programs fully exploiting computer capabilites. No serious two-dimensional problem can be completely stored in the main memory of BESM-6. Therefore we had to split a problem into several subtasks and to compute solution values at each timestep in parallel. Our programmers had to develop the desired splitting theory and to implement the splitting technique in multiple programs. The experience acquired with BESM-6 multicomputers allowed a painless transition to the Elbrus 1-2/ multiprocessor and was further used when developing methods and programs for Elbrus-2.

The first two-dimensional calculations were performed in 1950-s after D code has been written in 1956 to implement an algorithm for two-dimensional time-dependent gasdynamic calculations using Lagrangian variables and an explicit scheme (the well-known Neumann-Richtmeyer method generalized to two-dimensional case). This procedure is still successfully used.

The greatest number of complex two-dimensional calculations were performed since late 60-s to yearly 70-s on BESM-6 using the Sigma code. Sigma implements several computational methods. Particularly, gasdynamic calculations in Langrangian variables are performed using a method similar to D.

The heat conduction equation is solved with iterative methods using directional splitting. This complex was later enriched with many codes: in particular, it was run together with energy release which in turn is a relatively large program complex. The latter includes codes for kinetic neutron equations, chemical and nuclear reaction kinetics etc.

Hundrers of problems were yearly solved using TIGR-F, B-71, DMK MEDUZA programs and others. The first one is a modified version of TIGR program written by I. Yanenko (VNIIEF), the second is that of B-71 from Keldysh Institute, two remaining programs implement irregular methods. The capabilities of TIGR-F are similar to those of SIGMA, it also runs together with energy release codes. DMK which implements irregular methods for gasdynamic calculations with heat conduction included can also run in parallel with energy release codes.

3. Three-dimensional method development started many years ago. However first 3-D calculations were actually performed on ES-1066 and Elbrus-2 in the 1980-s. The development of an operational code takes many years. It is not only because an algorithm should be designed and implemented as a code. This work is of importance, still it is only the first stage in creating a practical code. The program should be further provided with rich subroutine library describing material properties and parameters must be specified for these subroutines.

Then the program must be used to perform relatively illustrative experimental computations which will demonstrate its special features and capabilities to potential customers. The program must be adjusted using the results of real product field tests and this only will ensure the customer's credibility and the code will be used to design new products. A practical program must have a number of technological features: for example, one could not expect the new program to be used solely by the authors. Practical program is run

by technicians and laboratory assistants which are sometimes helped by engineers. The authors are involved in highly important and complicated computations only.

Computers having an order of magnitude higher performance naturally require new methods and programs. A higher performance computer encourages one to think less about reducing the number of arithmetic operations but more about halt-free computation. less attention could be attributed to main memory savings and more importance should be given to the desired accuracy etc. For this reason we have completely modified our programs many times.

Till late 1980-s the computations were performed using a number of 3-D programs which allowed to calculate the electromagnetic radiation transfer (including light propagation) in complex three-dimensional geometries accounting detonation and strength forces: neutron motion. kinetics and energy release along with gasdynamic flows.

The new program generation was originally designed for three-dimensional calculations. however their efficient use for two-dimensional calculations was also considered. The goals seem to be achieved. Many thousands of two-dimensional calculations have been performed using new 3-D programs in recent 5 or 7 years. One can conclude that the decision to write integrated programming complexes for two- and three-dimensional calculations was reasonable. Moreover. recently several cases occurred where good two-dimensional programs and methods were modified by the authors so that they could be used both for two- and three-dimensional calculations. This is because two-dimensional calculations are still needed though good three-dimensional methods and programs are available.

This results primarily from the fact that the performance of domestic computers including the Elbrus-2 ten-processor complex is not high enough for large three-dimensional calculations. Two-dimensional problems run well on this system. It is also suited for 2.5-dimensional calculations where three-dimensional processes are computed in two-dimensional geometry or three-dimensional fixed geometry is used to compute a three-dimensional process etc. Large three-dimensional computations containing hundreds of thousands of points and accounting the impact of a variety of physical processes require hundreds or thousands of hours of CPU time. Mathematicians must do heroic efforts to perform such calculations. When we have GFLOP computers our mathematicians life may become easier.

However it may not be the case since mathematical models should be updated. a more detailed description of physical processes occurring in products will be needed and the computation accuracy will be increased. All this will be done at the expense of computer resources and

mathematicians' time. Mathematical models obviously need to be improved: for example, the detonation process is poorly computed, mixing models are not satisfactory, neutron computations include a low number of energy groups etc. Finally, the number of computation points is too small. Two-dimensional problems only contain the convergence calculations while the results obtained from two- and three-dimensional calculations depend on the mathematician, his (or her) experience and skill.

We understand quite well why computers able to perfom billions of arithmetic operations per second or more, terabytes of main memory and global networks are needed.

The majority of the above mentioned developments are reported in the review "Problems of Atomic Science and Engineering".

It should be noted that the complete list of three-dimensional codes is not exhausted. Some interesting and important projects are now under development, however they do not reach the production level. Our goal is to create at least two independent methods for the same three-dimensional problem. This objective was achieved in one- and two-dimensional calculations and this allowed to avoid many difficulties.

Correspondence or discrepancy between the results for the same problem run with different methods either allow to expect the result to be close to the true one or show the uncertainty level of the computation.