INTELLIGENT SOLVING MACHINES: PRINCIPLES OF CONSTRUCTION AND PERSPECTIVES ÅÑ

Koval V.N.[1]
V.M. Glushkov Institute of Cybernetics. National Academy of Sciences of Ukraine.

Abstract

The paper considers the hardware support provided for the distributed data and knowledge bases, and the same support is aimed at the specific class of the knowledge-oriented architectures, i.e. at the intelligent solving machines, which possess the developed high-level internal language and make it possible to efficiently realize the graph structures and parallel computations.

I. The pre-conditions of work

1. During many years, the specialists of V.M. Glushkov Institute of Cybernetics (National Academy of Sciences of Ukraine) were engaged in creation of the microprogram-based computers with the developed internal languages, with the flexible architectures, and oriented to the specified classes of the problems being solved. They are the MIR-series and UKRAINA computers; the shared intelligent terminal, used for the ELBRUS supercomputer, etc. [1].

The notion of machine intelligence, applied to the above-mentioned computers, is formulated and extended in [2]. This notion is characterized by:

- the ability to perceive high- and superhigh-level languages;
- the ability to work not only with data, but also with knowledge;
- the interactive cooperation of a user with a computer;
- the specific arrangement of a computation process.

2. We have the certain experience as for statement and solution of the problems present in the weakly-formalized domains that are directly associated with the AI sphere. They are classification and generation of notions, scheduling of actions, searching for regularities (when dynamical situation are recognized and some other recognition problems are resolved), probable reasoning as for computerization of deductive and inductive inferences, etc. [3], [4[, [5], [6].

3. The advent of some foreign developments used to solve AI problems, implementation and criticism of the Japanese project, the fifth-generation computers, as well as design and creation (during the last years) of some special-purpose architectures oriented either to the separately-taken models of knowledge representation (in particular, of semantics of big sizes), or to the support provided for a number of the declarative languages like PROLOG or LISP [7].

4. There was the intensive development of the architectures used for parallel computations: SMP (Symmetric Multiprocessor System), MPP (Massively Parallel Processing), or SPP (Scalable Parallel Processing) [8].

[1] V.M. Glushkov Institute of Cybernetics. National Academy of Sciences of Ukraine. 40, Prospect Akademika Glushkova, 252222, Kiev, Ukraine.
Tel. 266-70-85, fax 266-45-49,
e-mail: vkov @ valnat. kiev. ua

5. And, finally, there is the trend concerned with the hardware support provided not only for the special-purpose technologies, but also for the existing commonly accessible technologies used for knowledge processing (when different problems are resolved). This was said by Microsoft not so long ago [9]. This circumstance is to promote the universal computers of broad application that are oriented to a mass user.

All these moments taken together, and the demands has forced us to try to develop the distributed parallel architecture that realizes high-level languages, used to solve the traditional problems along with the AI ones and aimed at mass utilization [10], [11].

As for the pre-conditions, it is necessary to make one more remark. Our work is of the conceptual character to a large extent. We hope that the decisions made by us will turn out to be efficient. But we want to emphasize here that two patents were got by us for the main architectural decisions made within our design: one of them is Russian that is devoted to the architecture and structure of the computer system; and the second patent is Ukrainian, and it is devoted to the means of conflict-free switching [12], [13].

Ii. The new information technology used to solve the ai problems.

Now, some words about the features of the information technology concerned with solution of the AI problems. The contents of Table 1 qualitatively compares the technologies used to resolve the traditional problems and the AI ones. The AI-based problem solution technology, called the new information technology (NIT), is described in [14].

Table 1

Characteristics	Traditional	NIT
Application domains	Strongly formalized	Weakly and strongly formalized
Method of solution	Single running	Cyclic application of trial-and-error method
Stages of problem solution process computerization stages	Program execution	Construction of problem models; program synthesis and execution
Order of program construction and	Stages are strictly divided	Stages may alternate during problem execution
Human-computer operation style	Construction of complete problem model; interactive interrelation is used only for program debugging	Strong interactive interrelation; models are corrected during problem construction and execution

The main method used to solve problems according to NIT is the trial-and-error method applied in a cyclic manner (this is seen, because different solution construction strategies are present). The not completely determined situations are often met in this case, therefore, the program construction and execution steps are alternating by turns and are not divided within NIT, while they were strictly divided under the traditional solution technology.

This circumstance, in its turn, provides the necessity not only for the program execution steps, but also for the steps of solution synthesis: it is necessary to construct a problem solution

schedule, i.e. to achieve some known and clearly specified goal on the basis of some chosen strategy.

On the one hand, to support the NIT and, on the other hand, to realize the existing technologies of solution of the traditional problems, some new architectural decisions were needed, and we tried to implement just them within the framework of the Intelligent Solving Machines (ISMs), i.e. of the universal computers with the integrated architecture, which unite the features possessed both by the special-purpose architectures, used to solve the AI problems, and by the traditional architectures.

Iii. The main principles of isms construction

The principal feature of the ISM-class computers is that their architecture embodies four important properties:

- the hardware support of procedural and declarative high- and superhigh-level languages provided on the basis of application of an internal high-level language;
- the hardware support provided for the operations with the distributed data and knowledge bases, represented as the graphs, and with the other complicated data structures (CDSs);
- the combination of the centralized and decentralized control, and the 2D interpretation is the basis here;
- and the distributed processing of the information based on the memory-processor medium (i.e. the active memory is meant here).

The generalized graph concerned with ISM creation and realization problem formation is shown in Fig. 1. The Figure consists of two parts (i.e. of the upper and lower ones) and shows the process of transition from the statement of the ISM creation problem within the NIT, HLL, CDS

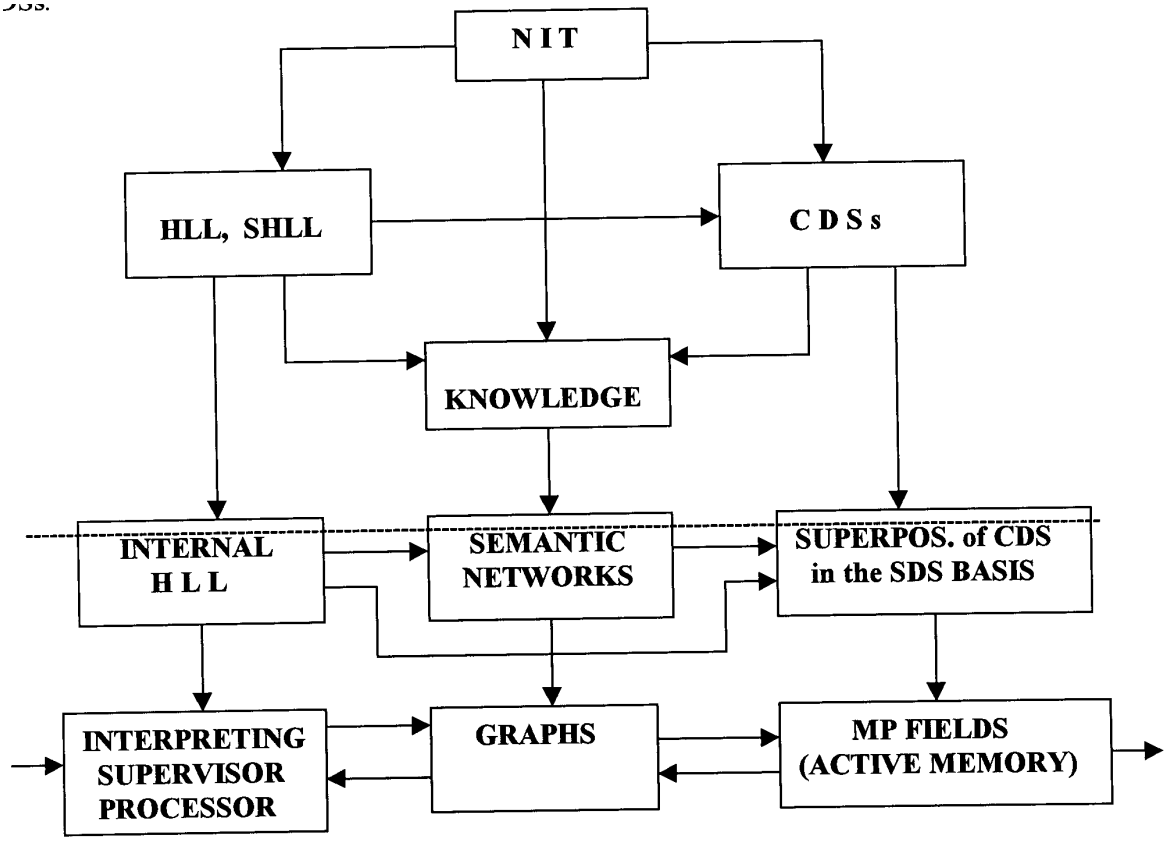

Fig.1

and Knowledge framework to its embodiment within the framework of an Internal HLL, Semantic Networks and Simple Data Structures (SDSs). It can be seen that the HLL is projected into the internal HLL, the knowledge is represented as the semantic network and the CDSs are given as the superposition of the SDSs.

The lower part is already the realization of the Internal HLL by means of the Controlling Interpreting Processor; of the Semantic Networks by means of the graphs; and of the SDS processing by means of the microprocessor (MP) medium (the active memory). The interconnections between the objects that reflect the information processing running in the ISMs will be clarified in the forthcoming discussion.

A peculiar place in the ISM-class computer belongs to its internal language. It is the C++-like procedural HLL enhanced by a number of the graph-oriented operations and by the parallelization and computation process control means. It is called C+Graph. It is supposed that it possible for the programs to be sufficiently easily compiled and converted into this internal language from a number of the traditional HLLs and SHLLs, like C++, JAVA, PASCAL, FORTRAN, BASIC, APL-360, PROLOG, and others.

Therefore, the traditional programs are realized for computational problems on the ISM-class computer in a natural way.

The knowledge and CDSs are represented in the ISM as the oriented graphs. For each graph, there exists the distributed representation in the form of the connecting list (the adjacency matrix, etc.), and, in addition, the graph may be represented as the object (without the internal structure). The set of the graphs-objects may be the semantic network which can be operated without the distributed representation.

The graphs play one of the principal roles. On the one hand, this graph is the CDS, i.e. the data type characterized by its objects and by the operations aimed at them, and, on the other hand, it is the program which is executed, i.e. this is the type of control. The graphs can help to sufficiently easily represent the complicated dynamic structures of the data and knowledge as the trees or semantic networks which may vary in time and grow down, in breadth, and so on.

To process the graph structures in the C+Graph language, the specific means are used, i.e. the "graph"-type variables and “graph”-type class, and this is the set of the operations and of the mechanisms of their interrelation used when the operations are aimed at the "graph"-type variables. As the examples of these functions, there are graph construction, graph search, choice of a value from a graph, etc.

In addition to the graph-based means, the internal language is also enhanced with the means of parallelization and control of a computation process like "fork", "join", "any", parallel "do", parallel "if", etc., and this is done in order to arrange the parallel computations.

Since it is impossible for the C+Graph-like language to be hardware-realized on the basis of the MPs with the RISC-architecture, then the two-level language must be built, while each level is realized on the basis of their components (Fig. 2).

The upper level is the algorithmic language like C+Graph, which contains the graph and parallel operators and the means used to operate with them. This level fixes the machine purpose as a whole and the centralized control that corresponds to the same level. The lower level is the usual MP command language. This lower level fixes the direct information processing and the decentralized control corresponding to the same level.

Therefore, the presence of two internal-language levels provides the combination of the centralized and decentralized controls in the ISM-class computer.

The centralized control is performed by the internal language interpreter according to the program execution graph. This graph is the dynamic object. First of all, it is constructed by the compiler under transformation of a program from an external language into an internal language,

and parallel branches are labeled. It is continuously transformed further on, depending on the problem solution way. The program execution graph is the CDS for the interpreter. Each vertex of this graph is brought in correspondence with some section of the program in the internal language, and some separate sequence of the vertices is brought in line with the parallel branch. Naturally, the internal language interpreter can simultaneously process the set of the parallel branches or of the separate independent problems. Some field of the processing virtual processors may be brought in

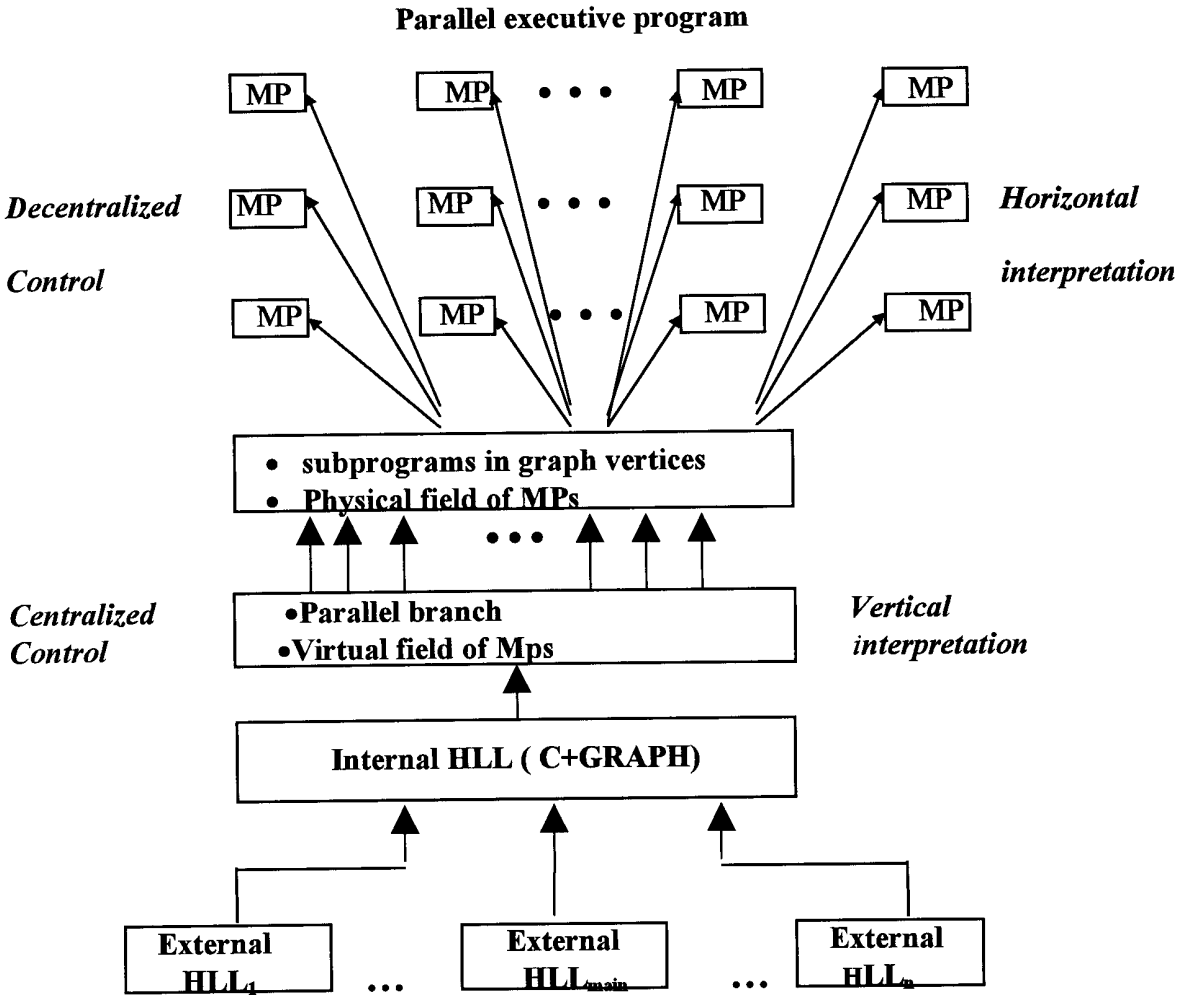

Fig. 2. 2 D INTERPRETATION

correspondence with this set, and, in their turn, they are mapped onto the field of the physical processors. When the set of the virtual processors becomes larger than the one of the physical processors, the latter form the lines as for execution of different program parts. The interpreter saves the whole set of the parallel branches and transfers the control to the operation system in order to realize the serial and parallel processing in the MP field.

The source program may contain the description of the semantic network. If the semantic networks are processed, the interpreter functions in much the same way, as this is done when the program branches undergo processing. It scrolls the semantic network and modifies it, i.e. constructs the nodes and the interrelations between them by means of the graph processing functions. The semantic network is also built and processed dynamically during the program execution process.

Therefore, the internal language interpreter replaces the operational system at the stage of processing of the semantic networks and source programs. i.e. the functions of the interpreter are the same as the loading functions of the operational system. A the same time, the interpreter

initializes the accesses to the operational system in order to call its standard functions for distribution of the problem branches between the MPs and for their arrangement into a line, the memory is distributed, and so on.

The decentralized control is the control of the execution of program sections performed by the separately taken MPs (on the basis of which the parallel ISM architecture is built) within the program execution graph vertices. During the decentralized control, there is the transition from the upper level of the internal language to the language of the MP commands, and these commands are executed. Each MP--field microprocessor saves in its memory the so-called kernel of the internal language interpret, in which the table is fixed that contains the correspondences between the internal language operators and their subprograms in the MP command language. Thus, the internal language interpreter is, so to say, distributed in the space of the whole machine. Since there may be many simultaneously operating MPs, this circumstance provides the possibility for the parallel interpretation of the internal HLL operators.

Thus, the two-step interpretation is implemented in the ISM-class computers. In the first case, it takes place at the level of the internal C+Graph language interpreter and, second, it is at the level of the kernels of the interpreter of the same language that are located in the executive Mps. Hence, there appeared the notions of 2D interpretation, seen when the parallel architecture is controlled, and of the two-level internal language. It is just this circumstance that makes it possible to implement an HLL on a parallel architecture. It can be easily understood from Fig. 2.

IV. ISMs EQUIPMENT STRUCTURE

The ISM equipment is made up in such a way, that the internal language and the mechanisms of operation with the graphs are hardware-supported, and the centralized and decentralized controls are efficiently performed. We, as well as the other specialists, use the cluster principle of arrangement of the system, but with one specifically singled-out Supervisory processor and with the means of switching between the clusters.

The cluster is the SMP-type symmetric parallel architecture. The cluster totality has the features both of the MPP- and SPP-architectures.Each cluster may include 2-8 different-type MP modules (RISC, SISC and the digital signal processors). The cluster has the 8-Gbytes memory subsystem, the interrupt processing subsystem, the subsystem of the external memory of 50 Gbytes, and the I/O subsystem used for the communications established with the control processor, the other clusters and the user terminals on the basis of the PCI and EISA buses. To improve the traffic, several buses are introduced.

The Supervisory processor is aimed at the centralized control and synchronization of the array containing the MP modules. It realizes the C+Graph HLL as the internal language with help of the interpreter of this language. This processor contains the control structures of the HLL interpreter: the tables, the stacks, the queues, the messages and the charts that reflect the queries of users and their programs.

The Supervisory processor communicates with each MP module and with all these modules by the intercluster channel switch and by the short-message one. The short-message switch is the conflictless circuit which transfers the interrupt codes. The intercluster channel switch is the fast table arbitration system without delays that has the specific arbitration mechanism.

When the system is initialized, the MP module is loaded with the kernel of the internal language interpreter, the frequently used operational system subprograms, the mathematical functions and the library HLL functions (for instance, they are the functions of memory distribution, queue processing , sorting, etc.).

Tabl.2

Characteristic	Cluster Level	System Level
Scalability	1 : 4	1 : 8
Clusters in system		2 - 8
MPs in Supervisor processor		1 - 2
Number of Mps	2 - 8	2 - 64
Types of applied Mps	RISC, SISÑ,	DSP
Types of MP models	Power-PC, Pentium-Pro,	TCM320
Common memory	up to 4 Gbytes	16Gbytes (4 clusters)
Disk memory	up to 48 Gbytes	200 Gbytes (4 clusters)
Number of service monitors		64

Intercluster communications:		
by interrupts	Short-message switch	
by data transfer	Intercluster channel switch	
Peak performance:		
digital operations /sec	up to (60x64)SPEC	_rate_95
floating-point operations /sec	int_base	TPP (MFLOPS)
	up to (120x64) LINPACK	
logic instructions /sec	up to 3 MLIPS	
Software for C+ GRAPH	Coprocessor, compiler,	converters, parallel interpreter
Programming languages	C+Graph, C++, PASCAL,	ALGOL, PROLOG
Support of operational systems	SCO UNIX Ware 2, Micro	soft Windows NT4

Table 2 shows the expected characteristics of the ISM-class computers. It is seen that these expected ISMs characteristics are in agreement with the contemporary viewpoints concerned with high-performance distributed architectures. The features inherent in the SMP-, SPP- and MPP-architectures, as well as their software, are organically combined here, and this circumstance makes it possible to wait for the efficient implementation of the traditional problems. Since the ISMs have the developed internal C++-like language, that possesses the powerful means used to process the graphs of a high dimension, the possibility to resolve the non-traditional AI problems is essentially greater, while these AI problems are based on knowledge processing, inductive and deductive constructions, etc.

It should be noted in conclusion, that the direction of the investigations, concerned with ISM and creation is rather promising. It is expected, that the nearest future will bring some of the obtained and completed model-based experiments associated both with the architecture of and the mathematical support provided for the concrete ISM versions, and, maybe this circumstance will promote the further direction of this work.

1. Âû÷èñëèòåëüíûå ìàøèíû ñ ðàçâèòîîì ñèñòåìàìè èíòåðïðåòàöèè / Â.Ì.Ãëóøêîâ, À.À.Áàðàáàíîâ, Ë.À.Êàëèíè÷åíêî è äð. - Êèåâ: Íàóê. äóìêà, 1970. - 258ñ.
2. Ðàáèíîâè÷ Ç.Ë. Î êîíöåïöèè ìàøèíîãî èíòåëëåêòà è åå ðàçâèòèè // Êèáåðíåòèêà è ñèñòåìíûé àíàëèç. - 1993. - ¹ 3. - Ñ.69-78
3. Ãëàäêîé Â.Ï. Ïëàíèðîâàíèå ðåøåíèé. - Êèåâ: Íàóê. äóìêà, 1987. - 168ñ.
4. Ãëàäêîé Â.Ï. Ïðîöåññû ôîðìèðîâàíèÿ íîâûõ çíàíèé. - Ñîôèÿ: Ïåäàãîã, 1994. - 189ñ.
5. Â.Í.Êîâàëü Ìåòîäû îáðàáîòêè èíôîðìàöèè äëÿ ñèñòåì îáúåêòíî-îðèåíòèðîâàííûõ ïðîãðàìì. Ïðåïð. Èí-òà êèáåðíåòèêè èì.Â.Ì.Ãëóøêîâà, - Êèåâ. - 1989. - ¹ 89-55. - 41ñ.

6. Êîâàëü Â.Í. Ýëåìåíòû ìàòåìàòè÷åñêîãî àíàëèçà ìíîãîìåðíûõ ïðîöåññîâ. Ïðåïðèíò Èíñòèòóòà êèáåðíåòèêè èì.Â.Ì.Ãëóøêîâà, - Êèåâ. - 1989. - ¹ 89-38. - 42ñ.
7. Áóëàâåíêî Î. Í. Ïàðàëëåëüíûå àðõèòåêòóðû â âû÷èñëèòåëüíûõ ñèñòåìàõ. Îáçîð // Óïð. ñèñòåìû è ìàøèíû. - 1993. - ¹3. - Ñ.37-51
8. Êîìïüþòåðû, î êîòîðûõ ïèñàëè (ïî ìàòåðèàëàì “Byte") Êîìïüþòåðíîå îáîçðåíèå ¹29 (53). - Êèåâ: àâãóñò 1966ã. Ñ.11-20.
9. Ì îçãîâîé òðåñò Áèëà Ãåéòñà. Computer Word N 4, Êèåâ: Ñ.23-28
10. Èíòåëëåêòóàëüíûå ðåøàþùèå ìàøèíû: ïðîáëåìû ñîçäàíèÿ è îñíîâíûå ïðèíöèïû / À.Â.Ïàëàãèí, Â. Í.Êîâàëü, Ç.Ë.Ðàáèíîâè÷ è äð. // Óïð. ñèñòåìû è ìàøèíû. - 1993. ¹3. - Ñ.37-51.
11. Êîâàëü Â.Í., Ïàëàãèí À.Â., Ðàáèíîâè÷ Ç.Ë. Âîïðîñû ìåòîäîëîãèè è ôîðìàëèçàöèè ïîñòàíîâîê è ðåøåíèÿ ïðîáëåì //Êèáåðíåòèêà è ñèſòåìíûé àíàëèç.-1995. ¹3. - Ñ.138-143.
12. Âû÷èñëèòåëüíàÿ ñèſòåìà. Ïàòåíò ÐÔ ¹2042193. Êë ä-06F15/16 Áþë. 23 îò 20.08.95 / Áóëàâåíêî Î. Í., Êîâàëü Â.Í., Ïàëàãèí À.Â. è äð.
13. Îá÷èſþâàëüíà ſèſòåìà. Ïàòåíò Óêðà¿íè ¹ 19875. Êë ä-06F15/16 Áþë. 6 îò 25.12.1997ð. Áóëàâåíêî Î.Ì., Êîâàëü Â.Ì., Ïàëàã³í Î.Â. òà ³í.
14. Ïſñïåëîâ Ä.À. Ìîäåëèðîâàíèå ðàſſóæäåíèé. - Ì.: Ðàäèî è ſâÿçü, 1`989. - 184ſ.