American Journal of Electrical and Computer Engineering
2017; 1(1): 1-8
http://www.sciencepublishinggroup.com/j/ajece
doi: 10.11648/j.ajece.20170101.11

Construction and Continuous Processing of Programs with Determinate-Connected Modules

Evgeniy Bryndin

Research Centre "NATURAL INFORMATIC", National Supercomputer Technological Platform, Novosibirsk, Russia

Email address:
bryndin15@yandex.ru

To cite this article:
Evgeniy Bryndin. Construction and Continuous Processing of Programs with Determinate-Connected Modules. *American Journal of Electrical and Computer Engineering.* Vol. 1, No. 1, 2017, pp. 1-8. doi: 10.11648/j.ajece.20170101.11

Received: January 29, 2017; **Accepted**: February 28, 2017; **Published**: March 15, 2017

Abstract: Continuous processing of large volumes of information on virtual computer memory was considered as technically not realized. This problem is not technical, but algorithmic. In article the solution of this problem is considered: creation of programs with the determinate-connected modules; structure of the supercomputer for continuous processing of programs with the determinate-connected modules; management of the continuous automated processing of programs with the determinate-connected modules on virtual memory without delay of obtaining results because of exchanges of information between random access and external memory. Also block structure of segments of random access memory with anticipatory switching of blocks of random access memory with processors for optimum processing of programs with the determinate-connected modules is considered. The formalism of operator schemes with natural interpretation is developed for reduction of programs with nondeterministic communications of modules to programs with the determined tactics and the strategy of behavior. This formalism is used for the proof of resolvability of a problem of creation of programs with the determinate-connected modules. In the offered theory of operator schemes the problem of creation of programs with determinate appeals to modules is solvable. Continuous processing of programs with the determinate-connected modules on virtual memory of the universal supercomputer with anticipatory management of memory reduces waiting time of result exponential when processing on virtual memory of the target computer in comparison with the existing supercomputers with casual management of memory.

Keywords: Programs with the Determinate-Connected Modules, Anticipatory Switching of Memory with Processors, Anticipatory Replacement of Modules of the Program

1. Introduction

Time of processing of big programs on modern computers is reduced by application of special methods of their processing with effective use of computer facilities and creation of high-speed element base [1-3, 5-6].

Efficiency of processing of big programs depends on the strategy of movement of their modules on virtual memory. The existing strategy of combination of processing of modules of programs and movement them on virtual memory has unproductive cost of exchange of modules. The number of exchanges between external and operational devices of memory is depending on the communications between modules subjected to exchange. The aspiration to reduce unproductive expenses was led to the solution of a problem of the minimum replacement which consists in search of such splitting the program into modules of given size at which the number of replacements comes down to a minimum.

The problem of elimination of unproductive expenses at exchanges is solved in systolic computers for one program.

Processing of various big programs without loss of productivity of processing because of exchanges it is possible for a message if proactively to replace the used modules in random access memory with the next modules on processing which are on external memory. For this purpose it was required to solve a problem of creation of programs with it is determined - the connected modules and to develop structure of the computers with anticipatory replacement.

2. Programs with the Determinate-Connected Modules

Let the modular and connected program P where P = {PIj}, j = 1,…n, PIj – either the operational module, or the module

of data of the program, or the input-output module is set.

PIj = { UIj, ODj, LDj, Pj, SPPj, SPIj }, where UIj – the operating information, ODj – the general data, LDj – local data, Pj – the program of processing, SPPj – communication on search of the module of the following in execution, SPIj – algorithmic communication of modules on execution transfer.

Definition 1. PIj … PIk connected through programs of search of SPP1 … by SPPj … SPPk we will call the sequence of PI1 modules modular execution of PPI of the program P if the sequence of programs of processing of P1 … Pj … determines by Pk result of the program P for her data, where l = 1 … m.

Definition 2. The structure of communications on search of modules of the program P is determined if according to her data it is possible to execute the sequences {(SPP1 … SPPj … SPPk)} programs of search of modules for all modular executions {PPIj} of the program P.

Definition 3. The modular and connected program P with a certain size of all modules is called the program with the determinate-connected modules if the structure of communications on search of her modules is determined.

2.1. Operator Schemes for Creation of Programs with the Determinate-Connected Modules

The theory of operator schemes with natural interpretation is developed for transformation of any program to the program with the determinate-connected modules [4, 7-8].

Definition 4. Focused columns G (X, U) in which operators of the program P are presented by tops of X = { to O1, O2,…., On } the column, and information communications – by data are set by function G: X® Y, and function U: X \Y - Y - Y - – communications on execution between operators are set, we will call the operator scheme of the program P.

Definition 5. The Oj operator alternatively copes on execution if he has several entrances on execution.

Definition 6. The Oj operator alternatively operates execution if he has several exits on execution.

Definition 7. The Oj operator is called polysemantic if he possesses various semantics of execution.

Definition 8. The Oj operator is called polyinformation if he has several information communications on transfer of result as operands (arguments) to other operators.

2.2. Types of Operator Schemes

Definition 9. The operator scheme G (X, U) is called linear on execution if all operators in the scheme unambiguously cope on execution and unambiguously operate execution.

Definition 10. The operator scheme G (X, U), consisting of independent linear operator sub circuits, we will call the operator scheme, linearly figurative on execution.

Definition 11. The operator sub circuit of G (Xi) having one operator of an entrance through communications on execution, alternatively the managing director of two or more operators of a sub circuit, and one operator of an exit from a sub circuit through communications on execution who is alternatively operated by two and more operators of a sub circuit and operators of an entrance and exit various is called an operator sub circuit, hammock-image on execution.

Definition 12. An operator sub circuit of G(X0 Ì X), U) which has X0 operators = {Oj} form one closed chain (Oj1, Oj2,…, OjN) through communications on execution of U: X0®X0, is called not linked operator sub circuit with return on execution.

Definition 13. Operator sub circuits of G(X1 Ì X), U)

and G(X2 Ì X), G, U) with return on execution are called linked if = Æ the general operators have sub circuits of X1 Ç X2.

Definition 14. Not linearly figurative operator scheme G (X, U) without gamakoobrazny sub circuits on execution and without sub circuits with return on execution we will call linearly cross operator scheme on execution.

Definition 15. Execution of Oi1, Oi2,…, OiN of the operator scheme G1(X1, G1, U1) and execution of Oj1, Oj2,…, OjN of the operator scheme G2(X2, G2, U2) are called equivalent if for identical entrance data result in identical results.

Definition 16. The operator schemes G1(X1, G1, U1) and G2(X2, G2, U2) are called equivalent if sets of their executions equivalent.

Definition 17. The P1 and P2 programs having equivalent operator schemes are called equivalent.

Definition 18. Transformation of the operator scheme of the P1 program to the operator scheme of the equivalent P2 program, is called transformation of operator schemes on algorithmic basis.

Definition 19. Transformation of the operator scheme G (X, U), at which of G(X1, G sub circuit, U) the operator scheme G (X, U) turn out the polysemantic polyinformation operator with preservation of all information communications and communications on transfer of execution with the operator scheme G (X, U), we will call synthesis.

Definition 20. We will call transformation of an operator sub circuit to several independent sub circuits with preservation of all information communications and communications on transfer of management with the operator scheme splitting.

Definition 21. The sub circuit of the operator scheme processing group of entrance data irrespective of an additional sub circuit is called autonomous according to entrance data.

Definition 22. Transformation of the operator scheme G (X, U), at which of G(X1, G sub circuit, U) the operator scheme G (X, U) turn out the polysemantic polyinformation operator with preservation of all information communications and communications on transfer of execution with the operator scheme G (X, U), we will call synthesis.

Definition 23. We will call transformation of an operator sub circuit to several independent sub circuits with preservation of all information communications and communications on transfer of management with the operator scheme splitting

Definition 24. The sub circuit of the operator scheme

processing group of entrance data irrespective of an additional sub circuit is called autonomous according to entrance data.

Theorem 1. In the operator scheme G (X, U) can allocate all hammock-image operator sub circuits on execution.

Theorem 2. In the operator scheme G (X, U) can allocate everything the linked and not linked operator sub circuits with return.

Theorem 3. In linearly cross operator scheme G (X, U) can allocate all linear sub circuits.

3. Examples of Programs with the Determinate-Connected Modules

Let's consider several programs with the determinate-connected modules.

Sorting of data. Let there are data a1,…, am. These data need to be reassorted in not decreasing sequence. Let's place consistently basic data in k modules. Program of sorting by search of comparisons of basic data following:

i: = 0; j: = 0; t1: = 1; t2: = 1;

cycle so far t1 = k cycle so far j ≠c cycle so far t2 ≠ k

cycle so far i ≠ c to establish ≤ ($a_{t1,i}$, $a_{t2,i+1}$); i:= i+1 end

to keep (t_2); t_2:= t_2 + 1; i:= 0 end j:= j + 1; t_2:= t_1;

i: = j the end to keep (t1); t1: = t1 + 1; t2: = t1; i: = 0; j: = 0;

i: = j the end to keep (t1); t: = t + 1; t; = t; i: = 0; j: + 0 end

t1, t2 – counters of modules.

The operator to establish ≤ ($a_{t1,i}$, $a_{t2,i+1}$) places values of the specified couple according relation ≤. To the address $a_{t1,i}$ smaller value, and to the address $a_{t2,i+1}$– bigger value is placed.

The operator to keep (t_1) refuses from the module with number t_1 with his preservation.

The structure of the program with the determinate-connected modules defining use of the operational module and modules of data is represented in figure 1.

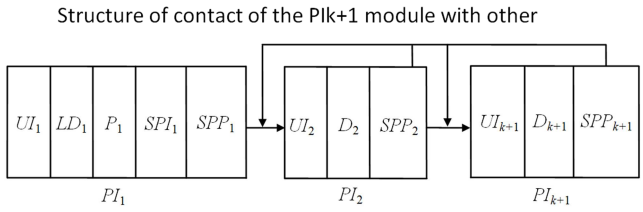

Figure 1. *Structure of the program of sorting.*

Conclusion of modules of data is appointed in the operational module. The program of sorting is placed in the first module, in other modules data are placed. Modules of the coherent program modular canonically are connected according to numbers, except the module with number k + 1. Structure of communication of the k + 1 module with other modules with returns. It is defined by the SPPk+1 program: if t1 = 1 that PI2 if t1 = 2 that PI3 … if t1 = (k – 2) that PIk–1.

Numbers of communication of the module and the address of the program of communication of the module k + 1 are stored in the operating information.

For four hours of processing replacement of modules on random access memory takes with processors of the program of sorting with casual appeals to modules on the modern computer 4 hours. The user receives results in 8 hours. At continuous processing of programs with the determinate-connected modules on the COMPUTER with anticipatory management of memory, the user will receive result in 4 hours.

Processing of databanks. Let in a databank there are questionnaires of five billion people in which the following data are reported: weight, growth, age, nationality, floor, social status, specialty, position, address, etc. It is necessary to choose all who answer on the biographical particulars to a standard from five billion people. Biographical particulars are consistently placed in k modules. In each module q of questionnaires is placed. Program of the choice of questionnaires against a standard following:

i: = 0; t1: = POD j: = 0; t2: = 1; n: = 0; t3: = k + 1;

cycle so far $t \neq k$ M: so far of i ≠ q to compare a cycle (a standard, the questionnaire (t2, i)) if yes to write down that ($t_{1,j}$, $t_{2,i}$); j: = j + 1

if j > q; that to keep (t1, t3); t3: = t3 + 1; n: = n +1

if n = 2 that t1: = t1 - 2 differently t1: = t1 + 1; j: = 0 on M

otherwise i: = i + 1 end to release (t2); t2: = t2 + 1 end all

t1, t2, t3 - counters of modules.

The subprogramme to compare (a standard, the questionnaire (t2, i) compares the questionnaire of i from the t2 module to a standard.

The subprogramme to write down (t1, j, t) writes down the questionnaire of i from the t1 module in the t2 module in the place of j.

The structure of the program with the determinate-connected modules determining use of the first operational module, k of modules of questionnaires and from k+2 by k+p of modules of standards is represented in figure 2.

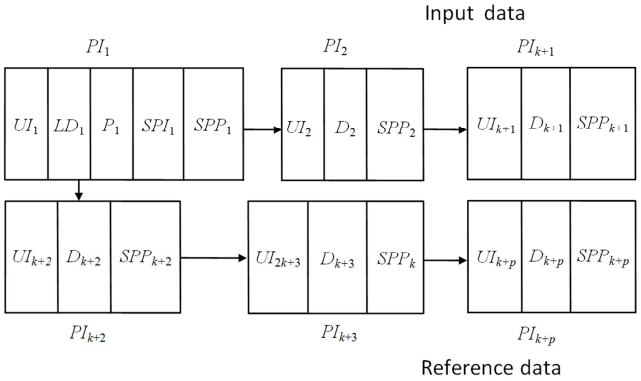

Figure 2. *Structure of the program of processing of databanks.*

The program of processing is stored in the first module. Further there is a sequence of modules with questionnaires. Initial questionnaires are stored in PI2,…PIk+1 modules. Reference questionnaires are stored in PIk+2,…PIk+p modules. Modules are connected according to numbers. Numbers of modules are stored in the operating information. In the resident module of the general data the current initial

questionnaires coinciding with a standard collect.

Program of syntactic analysis of expressions. It is necessary to carry out syntactic control of correctness of record of the expression which is in k modules of introduction data if the concept of expression is defined by the following syntactic rules:

EXPRESSION:: = TERM I TERM + EXPRESSION
TERM:: = MULTIPLIER I MULTIPLIER * TERM
MULTIPLIER:: = NAME I NUMBER I VSKOBKAKH
NAME LETTER I <name> <letter> I <name> <figure>
NUMBER:: = FIGURE I <figure> NUMBER
VSKOBKAKH:: = (EXPRESSION)

The program of control of syntactic correctness of expression is formed from subprogrammes: VSKOBKAH, CHISLO, IMYA, MNOZHITEL, TERM, VYRAZHENIE.

These subprogrammes define correctness of expression of his corresponding part. Syntactic correct concepts are allocated from the text consistently. Subprogrammes yield result yes, if the text consists of syntactic correct concepts. The analysis is kept until then when the counter of modules meets t condition = k + 1. In the analysis the module of the general data is used.

Texts for one pass easily give in to parse. The structure of the program with the determinate-connected modules defining use of the first operational module and other text modules it is represented in figure 3.

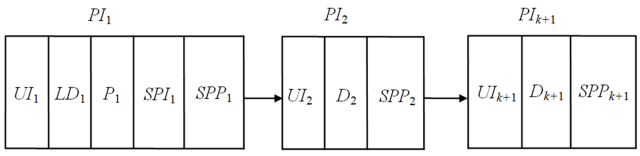

Figure 3. *Structure of the program of syntactic analysis of expressions.*

The program of the analysis settles down in the first module. Further modules with the entrance text settle down. Modules in the sequence are connected linearly according to numbers. Numbers of modules are stored in the operating information.

4. Continuous Processing on Virtual Memory of Programs with Determinate-Connected Modules

Let's consider computing processes with anticipatory management of virtual memory. Computing processes with anticipatory management of virtual memory process programs with the determinate-connected modules [9-12].

The anticipatory method of management of memory uses:

- the determined, dynamic method of distribution of resources of random access memory;
- anticipatory acts of the analysis of connectivity of modules of the program;
- the anticipatory inquiry anticipating a method of parallel replacement of the processed program modules;
- stream anticipatory movement of values of the general data between modules on operational segments of memory.

Programs with the determinate-connected modules allow the anticipating analyses of chains of communications, search of modules, movement of modules and values of the general data of depth of m on the current data where m – number of segments of random access memory. The anticipatory method of management allows to process continuously the program with the determinate-connected modules on virtual memory. Modules of the program are processed consistently, processing of each module parallel.

Criterion of continuous processing. Computing processes with an anticipatory method of management of memory continuously process the program with the determinate-connected modules if time of processing of each sequence from m of coherent modules is more than time of the analysis of communications, movement of the general data and parallel replacement of m of modules with 2 m segments of random access memory.

The method of replacement of modules of programs with the determinate-connected modules on random access memory anticipating their processing allows to provide a processing continuity on virtual memory by criterion of a continuity, unlike all used methods of replacement of modules of programs with casual appeals to modules. Computing processes with an anticipatory method of management of memory is a new direction in informatics of the automated processing of big programs with the determinate-connected modules.

5. The Supercomputer with Anticipatory Management of Virtual Memory

Supercomputers with anticipatory management of memory realize parallel and asynchronous interaction of acts of use of the computers resources under control of programs with the determinate-connected modules (figure 4). The supercomputer contains new devices: the processor of the analysis of communications between program modules (7), counters of use of segments of random access memory modules (14), the processor of movement of modules on virtual memory (13), the processor of movement of the general data of these modules (4). The processor of the analysis carries out the anticipatory analysis of communications of modules of programs with the determinate-connected modules. The processor of the analysis realizes process of calculation of number of the external module of movement of random access memory according to the program of communication of the module, process of correction of value of the counter of use of segments of random access memory program modules, process of start of the processor of movement of modules of the program from external memory on segments of random access memory. The processor of movement of modules realizes processes of movement of modules between devices of external and random access memory, providing existence

of necessary modules on random access memory according to the application of the processor of the analysis of communications.

The processor of movement of modules realizes: process of switching of segments of random access memory with store segments, processes of transfer and control of information of modules of the program, processes of interruptions on information transfer. The processor of movement of the general data realizes movement of the general data between modules. The general variables have the sequences of addresses of movement of their current values. On the sequence of addresses of movement streams of values of the general data with their delivery to the place of use in modules on operational segments will be organized. The appeal to modules happens according to their numbers. For modules of external memory of value of the general variables are transferred in the resident of the general data. At replacement of the module containing the general data and their values are transferred from the resident. Activization of computers devices is carried out by the processor of management according to the compiled program of the user with the determinate-connected modules.

Figure 4. *The supercomputer with anticipatory management of virtual memory.*

Continuous computing process begins with loading of initial modules of the program from external memory on segments of random access memory from movement of values of the general data and the analysis of communications between modules. When loading modules values on counters of use of operational segments are established by modules. For modules of data, input, a conclusion value of the counter is equal to unit. When loading the module on a segment of random access memory number of an operational segment is entered in the operating information of the module previous on search. Then the processor of management starts processing processors on processing of the first module. After an exit unit is subtracted from each operational module from value of the corresponding counter of use of memory and its value is checked for zero. For modules of data value of the counter is nullified from the operational module at the command of "nullify the counter.

When value of the counter becomes equal to zero and the index on the external module is defined, then the processor of movement of modules for replacement of the used module on random access memory by the external module according to the index is initiated. After replacement the index on the external module is nullified. If in the module on random access memory there was no record, then it isn't transferred to external memory. The processor of movement of the general data on the address sequences transfers the collected values of the general data from the resident of the general data to the replaced module. The processor of the analysis corrects the counter of use of random access memory the replaced module and determines by the program of communication of the replaced module the index on the external module.

The processor of management (8) will organize processing, movement of the general data, the analysis of communications and replacement of modules. He combines operation of devices on one module for different cycles of appeals to an operational segment.

The quantity of operational segments (1, 2) for continuous processing of the program decides on the determinate-connected modules in the course of her broadcasting or compilation.

Segments of random access memory unite in blocks. Switching of processors with operational segments block. Blocks are switched with processors consistently, according to the sequence of processing of the modules located on them. It allows to minimize switching of processors with random access memory, switching consistently proactively dynamically processors from the block to the block of operational segments.

At technical realization continuous presence of the current modules at random access memory is reached by the necessary number of blocks of segments of random access memory and by balance of speed of the multichannel parallel movement of modules anticipating processing from external memory (17) in quick with an information processing speed from random access memory.

The computer with anticipatory management of memory provides continuous process of processing of the program with the determinate-connected modules anticipatory parallel movement of the general these modules and anticipatory parallel replacement of modules on random access memory through multichannel exchange (3, 9, 16).

5.1. Stream of Values of the General Data

Values of the general data, ready to the subsequent processing, move on the program modules which are in random access memory. For each value of the general this d the sequence of the modules using him, places of their use in these modules and the relative moments of use of values d in modules are defined. On a set of modules of use of d the additional set of modules via which values of this d move is formed.

Values of the general data move on modules, being on segments of random access memory, dynamically forming a data flow.

The general data of the modules which are not on random access memory move to resident ROD modules. Are stored in the resident module of the general data of value together with indexes of movement. The values moved to one module settle down in a row. At the beginning of the sequence their quantity is specified. After record of new values his index of an empty seat (record) moves to the module of the general data if the counter of the module of the general data doesn't exceed admissible number of values.

Values are supplied with recalculation signs. If the sign accepts a condition of stability, then value moves to all used modules.

Values are located in the module of the general data as their moving to the modules arriving from external memory on random access memory. In the module of the general data of value can be supplied with several indexes. After moving of all values to the program module in him the sign is established "is moved" which specifies that the module is ready to processing.

Let there is k of modules of the sequence of execution and π segments of random access memory. Let the first module have variables. For each variable we will write out numbers of the subsequent modules in which it is used. For the second module we will write out all variables which aren't in the first module. For each variable we will write out numbers of the subsequent modules in which it is used. For the subsequent modules we will similarly write out the sequences of use of variables which aren't specified in the previous modules.

For each variable we will define external modules. Let's write out the sequences of numbers of the external using modules. Variables will be stored in the resident module of the general data it agrees of sequential numbering of the external modules using variables.

5.2. Replacement of Modules of the Program on Random Access Memory

For storage of modules of programs external memory is used. Let each module be placed on a separate segment. And let the sizes of a segment and the module are defined by the maximum portion of exchange of information between devices of external memory and random access memory.

The segment of random access memory is local address space for the module. A scope of commands of the module, except commands of the processor of movement of data, is the address space of an operational segment on which there is the corresponding module. Compliance between the segments of random access memory and modules which are on operational segments is established through associative registers of the equipment of switching.

If two segments of random access memory are used, then on one segment the processed module is stored, on another the module prepares for processing. If three operational segments are used, then on one segment there is a replacement of the processed module with the module according to the external reference, on another the processed module is stored, on the third the module prepares for processing.

Replacement of modules on segments of random access memory be defined by a condition of counters of use of segments modules of the program P and the index of the external module of the equipment of control of movement of modules.

Division of random access memory into independent segments becomes for anticipatory pumping of modules from external memory on quick. Several modules can move between devices of external and random access memory at the same time.

Hardware realization of replacement of modules on random access memory brings practically time of management of this process to naught in comparison with time of their replacement and processing.

The realization of the polysemantic polyinformation operators (PPO) of the program with the determinate-connected modules is enabled either on cotton velvet structures, or on network GRID-systems with the reconstructed schemes. For execution of several in a row PPO cotton velvet structure or network GRID-systems with the reconstructed schemes for continuous processing of programs are required two. The sequence of execution of

PPO on network changeable structures is defined after broadcasting per the sequence of execution is determined - the connected modules of the program.

The body of PPO is stored on external memory separately from the program. The procedure of loading of a body of PPO on network structures is started by the mechanism of anticipatory pumping.

6. Competitive Advantages

Continuous processing of big programs with casual appeals to modules on the existing supercomputers is realized by increase in random access memory, creation of grid-systems, and also by creations of the supercomputer with anticipatory management of memory.

(1) Use of the supercomputer with big random access memory for continuous processing of programs with the casual address to modules is inefficient because of decline in production because of switching complication "processors memories". Such approach is economically unprofitable because of essential increase in cost of random access memory.

(2) The distributed grid-systems consisting of a kind of parallel computers with standard processors, the data storage devices, power supply units, etc. connected to network (local or global) by means of usual protocols, also can't provide continuous processing of big programs with casual appeals to modules. Thus, we receive practically the same computing capacities for processing of big programs, as on the modern supercomputers working under control of programs with casual appeals to their modules. As well as in the first case, processing of big programs inefficient and expensive.

(3) Continuous processing of programs with the determinate-connected modules on virtual memory of the universal supercomputer with anticipatory management of memory reduces waiting time of result exponential when processing on virtual memory of the target computer in comparison with the existing supercomputers with casual management of memory. For the big programs which are executed for several days, and also for processing of a large number of data on virtual memory in real time it is the most effective and economically optimum method. Time of obtaining result is reduced due to anticipatory replacement of modules of the program on virtual memory, anticipatory movement of data on modules of the program, the multiprocessor parallel data processing ready for calculations and minimization of switching of processors with random access memory by consistently dynamic anticipatory switching of processors from the block by the block of operational segments.

Efficiency of the organization of computing process is checked on the interpreter of the target computer. Anticipatory operations of computing process of the interpreter of the target computer were performed by extracodes of the tool computer.

7. The Strategy of Realization of the Supercomputer with Anticipatory Management of Memory

The supercomputer with anticipatory management of memory can realize modification of a domestic supercomputer Lomonosov. It is expedient to carry out it with developers of the company of the T-platform. The supercomputer Lomonosov with a productivity of 1 peta-FLOPS having an order the 280th terabyte of random access memory and an order the 20th petabyte of disk space needs to be investigated previously on effective modification in the target supercomputer with anticipatory management of memory and multichannel exchange of virtual memory.

The project can be realized modification of a supercomputer of "Tyankhe 2". It is expedient to carry out it with the Chinese developers through the state holding "Roselektronika" in which there are research, design, production and marketing divisions. After transformation of a target supercomputer into market holding can create the international joint-stock company with preservation of infrastructure for the development gained an innovative market product and realization of demand for him.

The project can be realized creation of the consortium within a national supercomputer technological platform uniting fundamental research activity, educational activity for training, applied research and development and corporations of mass production of the target supercomputer with anticipatory management of memory and multichannel exchange of virtual memory.

References

[1] E. Khant, D. Tomas. Programmist-pragmatik. M: Izdatel'skii dom: Lori. 2004. 250 s.

[2] Kharol'd Abel'son, Dzheral'd Dzhei Sassman. Struktura i Interpretatsiya Komp'yuternykh Programm. Izdatel'stvo: Dobrosvet. 2004. 596 s.

[3] D. Knut «Iskusstvo programmirovaniya». T. 1-4. Massachusetts: Addison-Wesley. 2001-2008.

[4] Bryndin E. G. Teoreticheskie aspekty nepreryvnoi obrabotki na virtual'noi pamyati. Informatsionnye tekhnologii. – M., 2009. – № 9. S. 33–39.

[5] Stiv Makkonnell. Sovershennyi kod. Izdatel'stvo: Russkaya redaktsiya. 2010. 896 s.

[6] Dzhordzh Riz. Cloud Application Architectures. BKhV-Peterburg. 2011. 288 s.

[7] Bryndin E. G. Teoreticheskie osnovy imitatsii myshleniya i nepreryvnoi obrabotki na virtual'noi pamyati. – Tomsk: Izd-vo TPU, 2011. – 235 s.

[8] Evgenii Bryndin. Osnovy imitatsii myshleniya i nepreryvnoi obrabotki programm. – Germany: LAP LAMBERT Academic Publishing, 2012. – 197 s.

[9] Bryndin E. G. Upravlenie nepreryvnoi obrabotkoi programm na virtual'noi pamyati. /SUPERKOMP"YuTERY, № 3. 2014. S. 50-52.

[10] E. G. Bryndin. Tekhnologicheskie platformy shestogo uklada. //Reputatsiologiya № 3. –M. - 2014 – S. 56-64.

[11] Bryndin E. G. Nepreryvnaya obrabotka programm supercomputers uprezhdayushchim upravleniem pamyat'yu. Vseros. Nauch.-prakt. konf. «Mnogoyadernye protsessory, parallel'noe programmirovanie, PLIS, sistemy obrabotki signalov (MPPOS – 2015)». Barnaul: BGU. 2015. S. 20-27.

[12] Bryndin E. G. Reshenie problemy nepreryvnoi obrabotki programm na virtual'noi pamyati. "Vestnik PNIPU: Elektrotekhnika, Informatsionnye tekhnologii, Sistemy upravleniya", № 13. Perm': PNIPU. 2015. S. 91-107.