Journal of Fundamental and Applied Sciences

ISSN 1112-9867

Available online at http://www.jfas.info

Research Article

Special Issue

# SEMANTIC-NUMERICAL APPROACH TO THE AUTOMATIC PARALLEL FRAGMENT MODELS SYNTHESIS FOR COMPUTER SYSTEMS WITH DISTRIBUTED MEMORY

Gennady A. Polyakov, Nikolay I. Korsunov, Konstantin V. Lysykh*, Vladimir V. Lysykh, Alexey G. Lipnitskii

Belgorod State University, 308015, Pobedy 85, Belgorod, Russia,

Published online: 15 February 2017

**ABSTRACT**

The new semantic – numerical approach of the traditional C, C++ – programs separation into Fragments Set (fragmentation) and possibility of automatic synthesis of fragmental models of time parameterized (timing) parallel processes is suggested. The approach supports architecture and configuration of the computer systems with true distributed memory (supercomputers/Clusters) as well as the users demands and limitations (time of task solution or resources limitation). The practical realization of this approach provides the possibility for increasing of the effectiveness of famous traditional parallel Software of existing systems with distributed memory and can be considered as the new tools class of parallel software creating of high-effective perspective Adaptive Intellectual–Selforganising Technologies & Systems (AISTS) automatic technologies design.

**Keywords.** automatic fragmentation; adaptive; distributed memory, timeparametrized parallel model/process, multiprocessing computers, architecture, semantic-numerical specification (SNS), C++ - programs, verification, visualization.

## I. INTRODUCTION

The aim of the article is describing of the new semantic - numerical approach to the solving of a problem of automatic synthesis of the fragmentary models of timeparametrized parallel processes for the multiprocessing Systems of different types with distributed memory

Author Correspondence, e-mail: lysykh_k@bsu.edu.ru

doi: http://dx.doi.org/10.4314/jfas.v9i1s.782

(supercomputers/CLUSTER) - «Automatic Semantic-Numerical Soft Timing Fragmentation (ASNS TF). Into the basis of approach to automatic fragmentation and synthesis of parallel models the representation of Soft&Hard objects of computer systems and their operations as objects of the new discrete mathematics apparatus – Algebra structures of Space-Timing Semantic-Numerical Specification (SNS) and automatic execution of relevant operations of SNS structures is set [1].

The basis of scientific and technical progress nowadays is supply of highly efficient information technologies, IT. The essential factor of IT- effectiveness is the quality and terms of production of Soft&Hard provided with the help of automatic system.

The analysis of the problem included the appraisal of approaches and solving, which are used in standards of Message Passing Interface (MPI) and Parallel Virtual Machine (PVM) for the computers with true shared memory, and in standard Open MP for the computers with distributed memory; in extension High Performance Fortran, HPF – for Fortran, High Performance C, HPC – for C, High Performance HPC++ – for C++, Fortran – DVM; in program systems mpC; and in program systems on the base of languages of real time ADA, Occam.

Nowadays it is admitted that one of the main problems of IT and computing Systems is the creating of methods and ways of automatic synthesis of parallel Software for the technologies and systems in different applying spheres and multiprocessing computers of different types. The one of such problems is automatic synthesis of parallel fragmented/fragment programs for the systems with distributed memory [1,2,3,4,5].

## II. METHODOLOGY

The basis of ASNS TF approach are:

1. The new common (in comparison with general parallel programs) class of programs is TimeParametrized (timing) MultiParallel programs, TPMP (fig.1,2), first introduced by the authors [6].

2. The new discrete algebra of the Structures of Space and Time Semantic-Numerical Specification (SNS), first substantiated by the authors [1,6].

3. The new created theory and methods of formal synthesis, verification, appraisal of the effectiveness and visualization of timing multiparallel models of processes in the existing multiprocessing systems and in the Adaptive Selforganising Computing Systems), first substantiated by the authors [1].

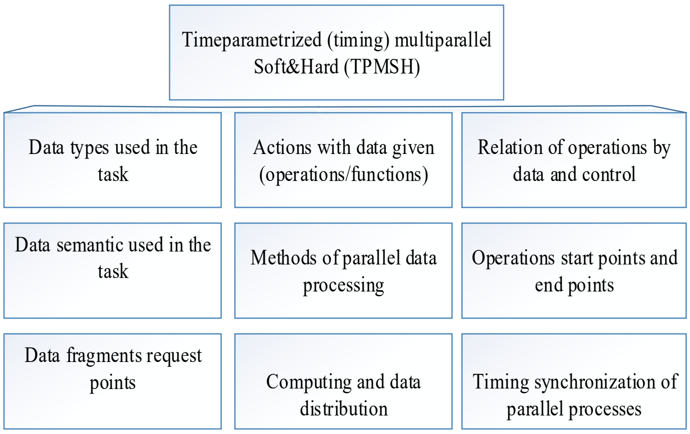


Figure 1. Compound of information categories, supported by TPMPS-programs

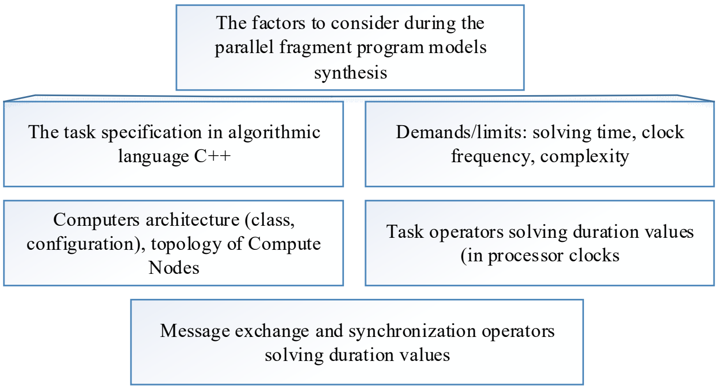


Figure 2. Factors of models formal projecting of fragment timeparametrized parallel programs

## III. Results and Discussion

Practical realization of ASNS_TF approach includes fulfilment of the following steps:

1. Synthesis of structures of semantic-numerical specification (SNS) of the operators set and relations set.

2. Synthesis of the Constructive C-Graph (CCG) of static visualization of SNS and C++-program.

3. Verification of Semantic-Numerical and Graphic Specification.

4. Synthesis of timing structures (SNS) and Timing Parallel Graph-schemes (TPGS) of graphic visualization of parallel task models (for the Soft&Hard configurations and demands/limits given).

5. Verification (and correction) of semantic-numerical specification (SNS) of timing C++ programs models.

6. Forming of SNS F fragments FR set of SNS of timing C++ programs models.

7. Forming of subsets FR for every fragment FR for message exchange operators and timing synchronization operators.

Because of the limited volume of the article the main steps of ASNS_TF approach are illustrated with the help of the sample using MPP system.

*A. Synthesis of BF and CF structures of semantic-numerical specification (SNS) of the task (operators set and relations set).*

The content of the first step is automatic text translation of the initial C-program in SNS structures of BF and CF in D1 task.

```
#include <stdio.h>
void main(void)
{
        int a,b,c,d;
        int k,z,p,s;
        scanf("%d %d %d %d",&a,&b);
   scanf("%d %d %d %d",&c,&d);
    if(a == b)
 {
         k = a % c;
         z = d * b;
                  printf("%4d\n",k);
                  printf("%4d\n",z);
 }
else
 {
         p = c * d;
         s = b / a ;
        printf("%4d\n",p);
        printf("%4d\n",s);
 }
}
```

Figure 3. C-program of D1 task

The process includes:

- forming of Reverse Polish notation (RPN) of the initial C++ -program;
- synthesis for RPN structures of semantic-numerical specification.

The results are illustrated in figure 4 and figure 5. The program support of this step is confirmed by the authors’ certificate [8, 9, 10].

| N | MET | TYP | NSJ | SJD | BJ | NWJ | WJD | MP1 | MP2 | VH | VIH | RES |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 58 | -1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | a_in |
| 1 | 0 | 58 | -1 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | b_in |
| 2 | 0 | 58 | -1 | 0 | 0 | 2 | 1 | 0 | 0 | 0 | 1 | c_in |
| 3 | 0 | 58 | -1 | 0 | 0 | 3 | 1 | 0 | 0 | 0 | 1 | d_in |
| 4 | 0 | 47 | -1 | 0 | 0 | 4 | 1 | 0 | 0 | 0 | 2 | a |
| 5 | 0 | 47 | -1 | 0 | 0 | 5 | 1 | 0 | 0 | 0 | 2 | b |
| 6 | 0 | 47 | -1 | 0 | 0 | 6 | 1 | 0 | 0 | 0 | 2 | c |
| 7 | 0 | 47 | -1 | 0 | 0 | 7 | 1 | 0 | 0 | 0 | 2 | d |
| 8 | 0 | 47 | -1 | 0 | 0 | 8 | 1 | 0 | 0 | 0 | 2 | k |
| 9 | 0 | 47 | -1 | 0 | 0 | 9 | 1 | 0 | 0 | 0 | 2 | z |
| 10 | 0 | 47 | -1 | 0 | 0 | 10 | 1 | 0 | 0 | 0 | 2 | p |
| 11 | 0 | 47 | -1 | 0 | 0 | 11 | 1 | 0 | 0 | 0 | 2 | s |
| 12 | 0 | 12 | 0 | 2 | 0 | 12 | 3 | 0 | 0 | 2 | 1 | = |
| 13 | 0 | 12 | 2 | 2 | 0 | 15 | 3 | 0 | 0 | 2 | 1 | = |
| 14 | 0 | 12 | 4 | 2 | 0 | 18 | 2 | 0 | 0 | 2 | 1 | = |
| 15 | 0 | 12 | 6 | 2 | 0 | 20 | 2 | 0 | 0 | 2 | 1 | = |
| 16 | 0 | 23 | 8 | 2 | 0 | 22 | 1 | 0 | 0 | 2 | 1 | == |
| 17 | 0 | 51 | 10 | 1 | 0 | 23 | 4 | 1 | 2 | 1 | 2 | upl |
| 18 | 1 | 5 | 11 | 3 | 1 | 27 | 1 | 0 | 0 | 3 | 1 | % |
| 19 | 0 | 12 | 14 | 2 | 1 | 28 | 2 | 0 | 0 | 2 | 2 | = |
| 20 | 0 | 3 | 16 | 3 | 1 | 30 | 1 | 0 | 0 | 3 | 1 | * |
| 21 | 0 | 12 | 19 | 2 | 1 | 31 | 2 | 0 | 0 | 2 | 2 | = |
| 22 | 0 | 50 | 21 | 2 | 1 | 33 | 1 | 3 | 0 | 2 | 1 | bp |
| 23 | 2 | 3 | 23 | 3 | 2 | 34 | 1 | 0 | 0 | 3 | 1 | * |
| 24 | 0 | 12 | 26 | 2 | 2 | 35 | 2 | 0 | 0 | 2 | 2 | = |
| 25 | 0 | 4 | 28 | 3 | 2 | 37 | 1 | 0 | 0 | 3 | 1 | / |
| 26 | 0 | 12 | 31 | 2 | 2 | 38 | 2 | 0 | 0 | 2 | 2 | = |
| 27 | 0 | 50 | 33 | 2 | 2 | 40 | 1 | 3 | 0 | 2 | 1 | bp |
| 28 | 3 | 54 | 35 | 2 | 3 | 41 | 1 | 0 | 0 | 2 | 1 | l.o |
| 29 | 0 | 49 | 37 | 1 | 3 | -1 | 0 | 0 | 0 | 1 | 0 | stop |
| 30 | 0 | 48 | 38 | 1 | 3 | -1 | 0 | 0 | 0 | 1 | 0 | k_out |
| 31 | 0 | 48 | 39 | 1 | 3 | -1 | 0 | 0 | 0 | 1 | 0 | z_out |
| 32 | 0 | 48 | 40 | 1 | 3 | -1 | 0 | 0 | 0 | 1 | 0 | p_out |
| 33 | 0 | 48 | 41 | 1 | 3 | -1 | 0 | 0 | 0 | 1 | 0 | s_out |

Figure 4. Structure of BF SNS of operators set of C-program D1 task

| N | JSD | SPJD | SNWIH | SNWHO | TSS | JWD | WPJD | WNWHO | WNWIH |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 0 | -1 | 12 | 0 | 0 |
| 1 | -1 | 4 | 1 | 1 | 2 | -1 | 13 | 0 | 0 |
| 2 | 3 | 1 | 0 | 0 | 0 | -1 | 14 | 0 | 0 |
| 3 | -1 | 5 | 1 | 1 | 2 | -1 | 15 | 0 | 0 |
| 4 | 5 | 2 | 0 | 0 | 0 | -1 | 12 | 1 | 1 |
| 5 | -1 | 6 | 1 | 1 | 2 | -1 | 13 | 1 | 1 |
| 6 | 7 | 3 | 0 | 0 | 0 | -1 | 14 | 1 | 1 |
| 7 | -1 | 7 | 1 | 1 | 2 | -1 | 15 | 1 | 1 |
| 8 | 9 | 12 | 0 | 0 | 0 | -1 | 19 | 1 | 1 |
| 9 | -1 | 13 | 0 | 1 | 0 | -1 | 21 | 1 | 1 |
| 10 | -1 | 16 | 0 | 0 | 0 | -1 | 24 | 1 | 1 |
| 11 | 12 | 12 | 0 | 0 | 0 | -1 | 26 | 1 | 1 |
| 12 | 13 | 14 | 0 | 1 | 0 | 13 | 16 | 0 | 0 |
| 13 | -1 | 17 | 0 | 2 | 1 | 14 | 18 | 0 | 0 |
| 14 | 15 | 8 | 1 | 1 | 2 | -1 | 25 | 1 | 0 |
| 15 | -1 | 18 | 0 | 0 | 0 | 16 | 16 | 1 | 0 |
| 16 | 17 | 15 | 0 | 0 | 0 | 17 | 20 | 1 | 0 |
| 17 | 18 | 13 | 0 | 1 | 0 | -1 | 25 | 0 | 0 |
| 18 | -1 | 17 | 0 | 2 | 1 | 19 | 18 | 1 | 0 |
| 19 | 20 | 9 | 1 | 1 | 2 | -1 | 23 | 0 | 0 |
| 20 | -1 | 20 | 0 | 0 | 0 | 21 | 20 | 0 | 0 |
| 21 | 22 | 21 | 1 | 0 | 1 | -1 | 23 | 1 | 0 |
| 22 | -1 | 19 | 1 | 1 | 1 | -1 | 17 | 0 | 0 |
| 23 | 24 | 14 | 0 | 0 | 0 | 24 | 18 | 2 | 0 |
| 24 | 25 | 15 | 0 | 1 | 0 | 25 | 20 | 2 | 0 |
| 25 | -1 | 17 | 1 | 2 | 1 | 26 | 23 | 2 | 1 |
| 26 | 27 | 10 | 1 | 1 | 2 | -1 | 25 | 2 | 1 |
| 27 | -1 | 23 | 0 | 0 | 0 | -1 | 19 | 0 | 0 |
| 28 | 29 | 13 | 0 | 0 | 0 | 29 | 22 | 1 | 1 |
| 29 | 30 | 12 | 0 | 1 | 0 | -1 | 30 | 0 | 0 |
| 30 | -1 | 17 | 1 | 2 | 1 | -1 | 21 | 0 | 0 |
| 31 | 32 | 11 | 1 | 1 | 2 | 32 | 22 | 0 | 1 |
| 32 | -1 | 25 | 0 | 0 | 0 | -1 | 31 | 0 | 0 |
| 33 | 34 | 26 | 1 | 0 | 1 | -1 | 28 | 0 | 0 |
| 34 | -1 | 24 | 1 | 1 | 1 | -1 | 24 | 0 | 0 |
| 35 | 36 | 27 | 0 | 1 | 1 | 36 | 27 | 1 | 1 |
| 36 | -1 | 22 | 0 | 0 | 1 | -1 | 32 | 0 | 0 |
| 37 | -1 | 28 | 0 | 0 | 1 | -1 | 26 | 0 | 0 |
| 38 | -1 | 19 | 0 | 0 | 0 | 39 | 27 | 0 | 1 |
| 39 | -1 | 21 | 0 | 0 | 0 | -1 | 33 | 0 | 0 |
| 40 | -1 | 24 | 0 | 0 | 0 | -1 | 28 | 1 | 0 |
| 41 | -1 | 26 | 0 | 0 | 0 | -1 | 29 | 0 | 0 |

Figure 5. Structure of CF SNS *relations set* in C-program

### *B. SNS-structures and C-program task visualization*

During the second step SNS-structures and C-program task visualization is executed (in the form of the Constructive C-Graph (CCG)). As it is illustrated in fig.6, CCG provides users with graphic visualization of static SNS-structures and with text of initial C- program. The program support of this step is confirmed by the authors' certificate [8].

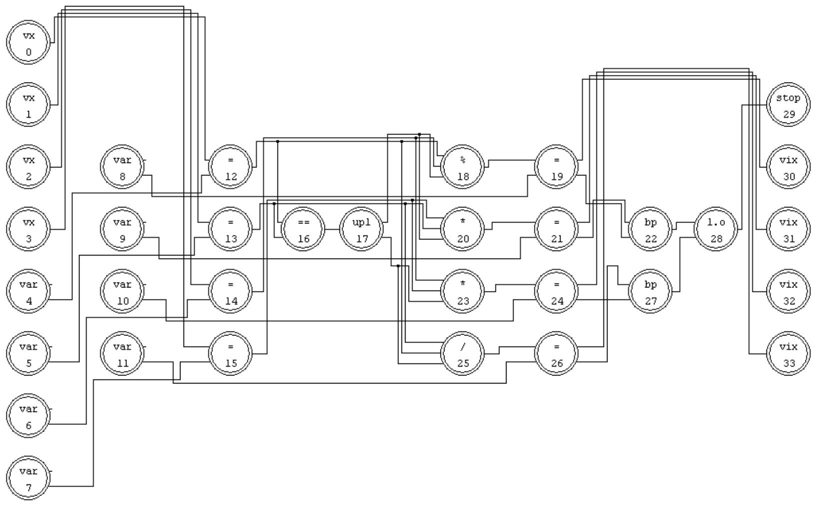


Figure 6. Constructive C-Graph (CCG) of C-program in D1 task

### C. *Verification of obtained SNS and CCG for C-program task*

During the third step the verification of obtained SNS and CCG for C-program task is executed. The process of the «compiling verification» includes:

- «back jump»: automatic synthesis for the obtained SNS of the derivative text (*D_text*) of initial C-program (*D_C*);
- verification of texts *D_text* and *D_C match*.

The program support of this step is confirmed by the authors' certificate [10].

For further consideration we use more common C++-program D2 (fig.7), the Constructive Graph is illustrated in the fig. 8. The results of verification are illustrated in fig.9.

```
#include <stdio. h>
void main(void)
{  int a, b, c, r, k, l, m, p, s, t;
    scanf("%d %d %d\n",&a, &b, &c);
    k = a * b;   l = b % a;
    if(k <  a - c)
       { m = (k % 2) * 2; r = l * 2;  p = k + l;
       }
   else
       { p = 2 * l; r = l - k; m = p + l; |
       }
         s = p - r;   t = (m * 2) / a;
         printf("%4d %4d\n", s, t);
         s = p - r;   t = (m * 2) / a;
         printf("%4d %4d\n", s, t);
}
```

Figure 7. C-program of D2 task

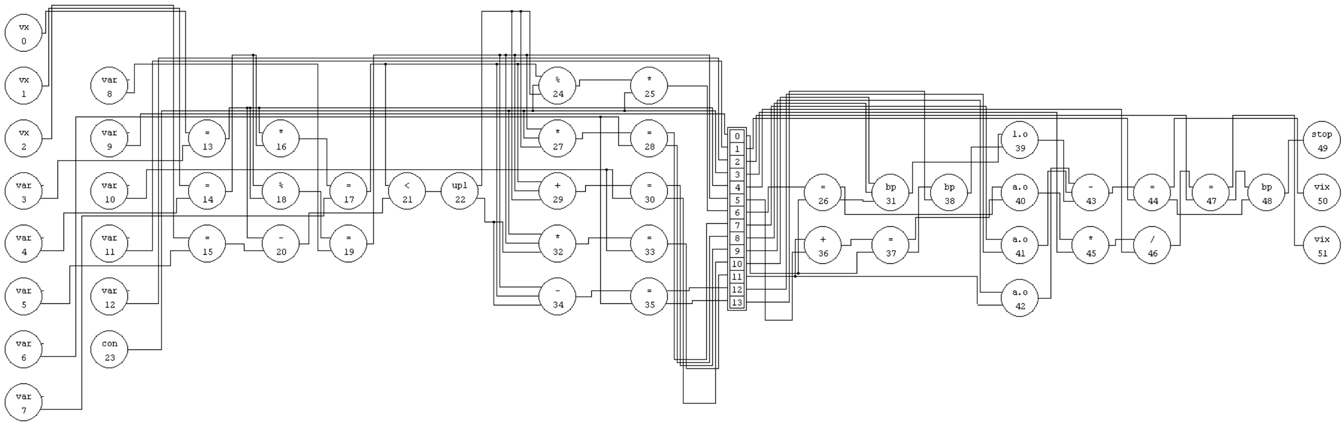

Figure 8. Constructive C-Graph (CCG) of C-program in D2 task

```
VERIFICATION TEST OF FILES:
the maximum number of elements:  0 - 51
the maximum number of relations:     0 - 78

CONFORMITY TEST OF NUMBER OF INITIAL AND EXTERNAL
RELATIONS:      OK
TEST OF NUMBER OF RELATION ON INITIAL ELEMENTS:         OK
TEST OF NUMBER OF RELATION ON OUTPUT ELEMENTS:          OK
CONFORMITY TEST OF OUTPUTS ON INITIAL ELEMENTS:         OK
CONFORMITY TEST OF OUTPUTS ON OUTPUT ELEMENTS:          OK
CONFORMITY TEST OF NUMBER OF INPUT ELEMENTS AND NUMBER OF
IT'S INITIAL:   OK
```

Figure 9. Verification result of SNS and CCG structures – «OK»

### *D. Synthesis of timing parallel models of initial C,C++-program*

The fourth step executes the synthesis of timing parallel models of initial C,C++-program for the computers configuration data and demands/limits given.

Fig.11 illustrates the timeparametrized model of C - program D_2 task, synthesized for the quantity of NM=2 processors (operators/processors operations with fixed point *t0_fix* and floating point *t0_pla* execution duration values are given in fig.10).

| oper | to_fix | to_pla |
|---|---|---|
| + | 1.00 | 2.00 |
| - | 1.00 | 2.00 |
| * | 10.00 | 2.00 |
| *,= | 2.00 | 1.00 |
| vh_d | 1.00 | 1.00 |
| stop | 1.00 | 1.00 |
| bp | 1.00 | 2.00 |
| upl | 1.00 | 2.00 |
| a.o | 1.00 | 2.00 |
| l.o | 1.00 | 2.00 |
| % | 35.00 | 11.00 |
| < | 1.00 | 2.00 |
| / | 35.00 | 11.00 |
| vih | 1.00 | 1.00 |
| send | 4.00 | 4.00 |
| rece. | 4.00 | 4.00 |

Figure 10. Operators/processors operations execution duration

The program support of this step is confirmed by the authors' certificates [9].

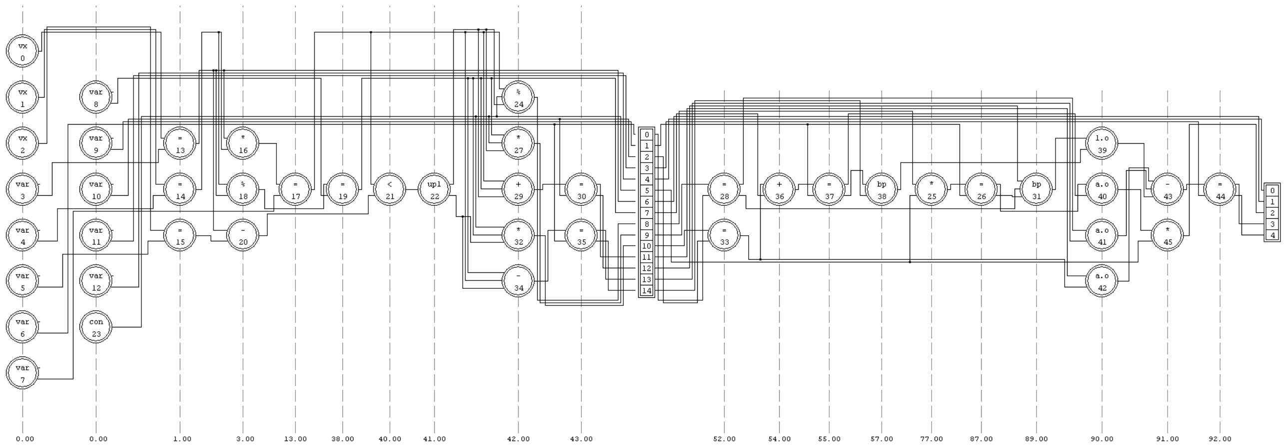

Figure 11. Timing Parallel Graph-Scheme (TPGS) of C-program model

### E. *Verification of semantic-numerical specification (SNS) of timing C++ programs models*

The content of the fifth step is verification of semantic-numerical specification (SNS) of timing C++ programs models. The results of verification are: evaluation of operators set correctness, their relations, timing diagram of C-program parallel model execution and describing of mistakes explanation.

The program support of this step is confirmed by the authors' certificate [10].

### F. *Synthesis of fragment SNS_FR*

The sixth step executes the synthesis of fragment SNS_FR structure of semantic-numerical specification (SNS) of C++ program timing models.

The task solving is based on forming of "optimal covers" of operators set of model every stage.

The execution of this step is supported by software product.

Figure 12 represents C-program D_2 task fragmentation results for the quantity of fragments *kfr* = 2.

| N | TYP | NSJ | SJD | BJ | NWJ | WJD | FRAG | SLO | NT | NY | RES |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 58 | -1 | 0 | 0 | 0 | 1 | 1 | 0 | 0.00 | 0 | a_in |
| 1 | 58 | -1 | 0 | 0 | 1 | 1 | 1 | 0 | 0.00 | 0 | b_in |
| 2 | 58 | -1 | 0 | 0 | 2 | 1 | 1 | 0 | 0.00 | 0 | c_in |
| 3 | 47 | -1 | 0 | 0 | 3 | 1 | 1 | 0 | 0.00 | 0 | a |
| 4 | 47 | -1 | 0 | 0 | 4 | 1 | 1 | 0 | 0.00 | 0 | b |
| 5 | 47 | -1 | 0 | 0 | 5 | 1 | 1 | 0 | 0.00 | 0 | c |
| 6 | 47 | -1 | 0 | 0 | 6 | 2 | 1 | 0 | 0.00 | 0 | r |
| 7 | 47 | -1 | 0 | 0 | 8 | 1 | 1 | 0 | 0.00 | 0 | k |
| 8 | 47 | -1 | 0 | 0 | 9 | 1 | 1 | 0 | 0.00 | 0 | l |
| 9 | 47 | -1 | 0 | 0 | 10 | 2 | 2 | 0 | 0.00 | 0 | m |
| 10 | 47 | -1 | 0 | 0 | 12 | 2 | 1 | 0 | 0.00 | 0 | p |
| 11 | 47 | -1 | 0 | 0 | 14 | 1 | 1 | 0 | 0.00 | 0 | s |
| 12 | 47 | -1 | 0 | 0 | 15 | 1 | 2 | 0 | 0.00 | 0 | t |
| 13 | 12 | 0 | 2 | 0 | 16 | 4 | 1 | 0 | 1.00 | 1 | = |
| 14 | 12 | 2 | 2 | 0 | 20 | 2 | 1 | 0 | 1.00 | 1 | = |
| 15 | 12 | 4 | 2 | 0 | 22 | 1 | 1 | 0 | 1.00 | 1 | = |
| 16 | 3 | 6 | 2 | 0 | 23 | 1 | 1 | 0 | 2.00 | 2 | * |
| 17 | 12 | 8 | 2 | 0 | 24 | 4 | 1 | 1 | 3.00 | 3 | = |
| 18 | 5 | 10 | 2 | 0 | 28 | 1 | 1 | 0 | 2.00 | 2 | % |
| 19 | 12 | 12 | 2 | 0 | 29 | 5 | 1 | 1 | 3.00 | 3 | = |
| 20 | 2 | 14 | 2 | 0 | 34 | 1 | 1 | 0 | 2.00 | 2 | - |
| 21 | 25 | 16 | 2 | 0 | 35 | 1 | 1 | 0 | 4.00 | 4 | < |
| 22 | 51 | 18 | 2 | 0 | 36 | 5 | 1 | 0 | 5.00 | 5 | upl |
| 23 | 57 | -1 | 0 | 1 | 41 | 5 | 1 | 0 | 0.00 | 0 | C2_ |
| 24 | 5 | 20 | 3 | 1 | 46 | 1 | 2 | 0 | 6.00 | 6 | % |
| 25 | 3 | 23 | 2 | 1 | 47 | 1 | 2 | 0 | 8.00 | 8 | * |
| 26 | 12 | 25 | 2 | 1 | 48 | 2 | 2 | 1 | 9.00 | 9 | = |
| 27 | 3 | 27 | 3 | 1 | 50 | 1 | 1 | 0 | 6.00 | 6 | * |
| 28 | 12 | 30 | 2 | 1 | 51 | 2 | 1 | 0 | 7.00 | 7 | = |
| 29 | 1 | 32 | 3 | 1 | 53 | 1 | 1 | 0 | 6.00 | 6 | + |
| 30 | 12 | 35 | 2 | 1 | 54 | 2 | 1 | 0 | 7.00 | 7 | = |
| 31 | 50 | 37 | 4 | 1 | 56 | 1 | 1 | 0 | 10.00 | 10 | bp |
| 32 | 3 | 41 | 3 | 2 | 57 | 1 | 1 | 0 | 6.00 | 6 | * |
| 33 | 12 | 44 | 2 | 2 | 58 | 2 | 1 | 1 | 7.00 | 7 | = |
| 34 | 2 | 46 | 3 | 2 | 60 | 1 | 1 | 0 | 6.00 | 6 | - |
| 35 | 12 | 49 | 2 | 2 | 61 | 2 | 1 | 0 | 7.00 | 7 | = |
| 36 | 1 | 51 | 2 | 2 | 63 | 1 | 2 | 0 | 8.00 | 8 | + |
| 37 | 12 | 53 | 2 | 2 | 64 | 2 | 2 | 1 | 9.00 | 9 | = |
| 38 | 50 | 55 | 2 | 2 | 66 | 1 | 1 | 0 | 10.00 | 10 | bp |
| 39 | 54 | 57 | 2 | 3 | 67 | 1 | 1 | 0 | 11.00 | 11 | l.o |
| 40 | 53 | 59 | 2 | 3 | 68 | 1 | 2 | 0 | 11.00 | 11 | a.o |
| 41 | 53 | 61 | 2 | 3 | 69 | 1 | 1 | 0 | 11.00 | 11 | a.o |
| 42 | 53 | 63 | 2 | 3 | 70 | 1 | 1 | 0 | 11.00 | 11 | a.o |
| 43 | 2 | 65 | 3 | 3 | 71 | 1 | 1 | 0 | 12.00 | 12 | - |
| 44 | 12 | 68 | 2 | 3 | 72 | 2 | 1 | 0 | 13.00 | 13 | = |
| 45 | 3 | 70 | 2 | 3 | 74 | 1 | 2 | 0 | 12.00 | 12 | * |
| 46 | 4 | 72 | 2 | 3 | 75 | 1 | 2 | 0 | 14.00 | 14 | / |
| 47 | 12 | 74 | 2 | 3 | 76 | 2 | 2 | 1 | 15.00 | 15 | = |
| 48 | 50 | 76 | 5 | 3 | 78 | 1 | 1 | 0 | 17.00 | 17 | bp |
| 49 | 49 | 81 | 1 | 4 | -1 | 0 | 1 | 0 | 18.00 | 18 | stop |
| 50 | 48 | 82 | 1 | 3 | 79 | 1 | 1 | 0 | 14.00 | 14 | s_out |
| 51 | 48 | 83 | 1 | 3 | 80 | 1 | 2 | 0 | 16.00 | 16 | t_out |
| 52 | 47 | -1 | 0 | 0 | 81 | 1 | 1 | 0 | 0.00 | 0 | a1 |
| 53 | 12 | 84 | 2 | 0 | 82 | 2 | 1 | 0 | 3.00 | 3 | = |
| 54 | 47 | -1 | 0 | 1 | 84 | 1 | 2 | 0 | 0.00 | 0 | k1 |
| 55 | 12 | 86 | 2 | 1 | 85 | 2 | 2 | 1 | 7.00 | 7 | = |
| 56 | 47 | -1 | 0 | 3 | 87 | 1 | 2 | 0 | 0.00 | 0 | m1 |
| 57 | 12 | 88 | 2 | 3 | 88 | 2 | 2 | 1 | 13.00 | 13 | = |

Figure 12. C-program D_2 task fragmentation results

## G. *Forming of C++ programs timing model*

The seventh step executes the forming for fragments of semantic-numerical specification (SNS) of C++ programs timing model (fig.11) of "messages exchange" and "timing synchronization" operators subsets. The execution of this step is supported by the program product, i.e. by the operations library "Library SNS".

The initial data of seventh step are BF, CF structures of fragment semantic-numerical specification SNS_FR (fig.12) of C++ program timing models.

The results of insertion of data exchange operator and synchronization operator for C++-program D_2 are represented in fig.13.

| NN | priz | oper | A1 | A2 | NT |
|---|---|---|---|---|---|
| 0 | 0 | scanf | 0 | -1 | 1.00 |
| 1 | 0 | scanf | 1 | -1 | 1.00 |
| 2 | 0 | scanf | 2 | -1 | 3.00 |
| 3 | 0 | * | 0 | 1 | 5.00 |
| 4 | 0 | = | 3 | -1 | 15.00 |
| 5 | | send(1,3,2) | | | 17.00 |
| 6 | 0 | - | 0 | 2 | 21.00 |
| 7 | 0 | = | 4 | -1 | 22.00 |
| 8 | | send(1,4,2) | | | 24.00 |
| 9 | 0 | % | 1 | 0 | 28.00 |
| 10 | 0 | = | 5 | -1 | 63.00 |
| 11 | | send(1,5,2) | | | 65.00 |
| 12 | 2 | < | 3 | 4 | 80.00 |
| 13 | 1 | upl | 14 | 20 | 81.00 |
| 14 | 0 | * | 5 | 6 | 82.00 |
| 15 | 0 | = | 7 | -1 | 92.00 |
| 16 | 0 | + | 3 | 5 | 82.00 |
| 17 | 0 | = | 8 | -1 | 83.00 |
| 18 | | wait (37.00); | | | 94.00 |
| 19 | 1 | bp | 27 | -1 | 131.00 |
| 20 | 0 | * | 6 | 5 | 82.00 |
| 21 | 0 | = | 8 | -1 | 92.00 |
| 22 | | send(1,8,2) | | | 94.00 |
| 23 | 0 | - | 5 | 3 | 82.00 |
| 24 | 0 | = | 7 | -1 | 83.00 |
| 25 | | wait (14.00); | | | 98.00 |
| 26 | 1 | bp | 27 | -1 | 112.00 |
| 27 | 0 | - | 8 | 7 | 132.00 |
| 28 | 0 | = | 9 | -1 | 133.00 |
| 29 | 0 | print | 9 | -1 | 135.00 |
| 30 | | wait (46.00); | | | 136.00 |
| 31 | 1 | bp | 32 | -1 | 182.00 |
| 32 | 1 | stop | -1 | -1 | 183.00 |

| NN | priz | oper | A1 | A2 | NT |
|---|---|---|---|---|---|
| 0 | 0 | scanf | 0 | -1 | 1.00 |
| 1 | 0 | scanf | 1 | -1 | 1.00 |
| 2 | 0 | scanf | 2 | -1 | 3.00 |
| 3 | | wait (16.00); | | | 5.00 |
| 4 | | receive(1,3,2) | | | 21.00 |
| 5 | 0 | = | 3 | -1 | 30.00 |
| 6 | | receive(1,4,2) | | | 28.00 |
| 7 | 0 | = | 4 | -1 | 37.00 |
| 8 | | wait (30.00); | | | 39.00 |
| 9 | | receive(1,5,2) | | | 69.00 |
| 10 | 0 | = | 5 | -1 | 78.00 |
| 11 | 2 | < | 3 | 4 | 80.00 |
| 12 | 1 | upl | 13 | 18 | 81.00 |
| 13 | 0 | % | 3 | 6 | 82.00 |
| 14 | 0 | = | 7 | -1 | 117.00 |
| 15 | 0 | * | 7 | 6 | 119.00 |
| 16 | 0 | = | 8 | -1 | 129.00 |
| 17 | 1 | bp | 24 | -1 | 131.00 |
| 18 | | delay (16.00); | | | 82.00 |
| 19 | | receive(1,9,2) | | | 98.00 |
| 20 | 0 | = | 9 | -1 | 107.00 |
| 21 | 0 | + | 9 | 5 | 109.00 |
| 22 | 0 | = | 8 | -1 | 110.00 |
| 23 | 1 | bp | 24 | -1 | 112.00 |
| 24 | 0 | * | 8 | 6 | 132.00 |
| 25 | 0 | = | 10 | -1 | 142.00 |
| 26 | 0 | / | 10 | 0 | 144.00 |
| 27 | 0 | = | 11 | -1 | 179.00 |
| 28 | 0 | print | 11 | -1 | 181.00 |
| 29 | 1 | bp | 30 | -1 | 182.00 |
| 30 | 1 | stop | -1 | -1 | 183.00 |

Figure 13. First and second fragment operators

## CONCLUSION

Nowadays the basis of scientific and technical progress is the usage of high-effective information technologies, providing the automatic synthesis of high-quality parallel hardware and software in the shortest possible time. One of such problems is automatic synthesis of parallel fragmented/fragment programs for systems with distributed memory. The existing automation systems of design and programing are not capable to solve this problem.

The discrete mathematic algebra apparatus of the Structures of Space and Time semantic-numerical specification elaborated by the authors provided the basis of the creation of formal synthesis and analysis of timeparametrized multiparallel software and hardware of the perspective technologies and systems theory.

The practical realization of «Automatic Semantic-Numerical Soft_Timing Fragmentation ASNS_TF» approach provides the possibility for increasing of the effectiveness of famous traditional parallel Software of existing systems with distributed memory and can be considered as the new tools class of parallel software creating of high-effective perspective Adaptive Intellectual–Selforganising Technologies&Systems (AISTS) automatic technologies design.

## SUMMARY

The main advantages of this approach are:

- the usage of C, C++ language as the base of designing;
- the support of synthesis of a new kind of programs – timeparametrized (timing) parallel programs, which are not supported by existing systems of parallel programming;
- multiparallelism;
- automatic verification;
- the support of solving tasks of high dimension and reducing of time for solving these tasks;
- automatic visualization of all the objects of designing and the effectiveness index evaluation;
- cooperative automatic designing of Soft&Hard processes.

## REFERENCES

[1] Polyakov G.A., 2012. Synthesis and analysis of parallel processes in the Adaptive Timeparametrized Computing Systems /G.A. Polyakov, S.I. Shmatkov, E.G. Tolstoluzhskaya, D.A. Tolstoluzhskiy: monography. – V.N. Karazin Kharkiv National University, Kharkiv, p. 672

[2] P. Coussy, A. Morawiec, High-Level Synthesis from algoritmh to digital circuit, Springer Netherlands, 2008.

[3] Polykov G.A., 2002. The adaptive self-organizing systems with multiparallel data processing - the strategy of development of digital computer technology in the XXI century, Applied electronics, 1: 57-69.

[4] Polyakov, G.A., Lysykh, V.V., Lysykh, K.V., 2015. A formal method of functional SNS – synthesis of problem-oriented parallel-pipelined devices // Fundamental science and technology - promising developments VI: Proceedings of the Conference. North Charleston, 3-4.08.2015, —North Charleston, SC, USA:CreateSpace, 176-182

[5] Developing Parallel Programs — A Discussion of Popular Models (Oracle White Paper September 2010) http://www.oracle.com/us/products/tools/oss-parallel-programs-170709.pdf

[6] Polyakov, G.A., Lysykh, V.V., Lysykh, K.V., Tolstolujskaya E.G, Tolstolujsky D.A, 2016. Semantic-numerical approach to automatic synthesis of timeparamertized parallel programs, Proceedings of IFOST-2016. 2016 11th International Forum on Strategic Technology (IFOST), June 1-3, 2016 Novosibirsk, Russia, Part 1, 379-385

[7] Polykov, G.A., 2003. The hard-and-soft Automatic Design of Self-Organizing Adaptive Systems. Electronics and Computer Science, 3:149-155

[8] Polyakov G.A., Program "Generator of graphs of C-programs" The certificate №2004610165 from 12.01.2004; Rightholder Polyakov G.A.

[9] Polyakov G.A., Program "Generator of parallel models of C-programs" The certificate №2004610800 from 31.03.2004; Rightholder Polyakov G.A.

[10] Polyakov G.A., Program "Verificator of static-dynamic objects of designing of parallel digital devices" The certificate №2004612411 from 26.10.2004; Rightholder Polyakov G.A.

**How to cite this article**:
Gennady A. Polyakov, Nikolay I. Korsunov, Konstantin V. Lysykh*, Vladimir V. Lysykh, Alexey G. Lipnitskii. Semantic-numerical approach to the automatic parallel fragment models synthesis for computer systems with distributed memory. J. Fundam. Appl. Sci., 2017, *9(1S), 1289-1300.*