

Illia Danilishyn, Oleksandr Danilishyn

MATHEMATICAL FUNDAMENTALS OF CONSTRUCTING PSEUDO- LIVING ENERGY THEORY AND DYNAMIC PROGRAMMING

Monograph





Primedia eLaunch

Illia Danilishyn, Oleksandr Danilishyn

MATHEMATICAL FUNDAMENTALS OF CONSTRUCTING PSEUDO-LIVING ENERGY THEORY AND DYNAMIC PROGRAMMING

Monograph

Edited by *Volodymyr Pasynkov*

Shawnee, USA
Primedia eLaunch LLC
2025

Epigraph:

Mathematics is on the border of science (knowledge), so it is she who is able to make major breakthroughs beyond this border.

Reviewer: Volodymyr PASYNKOV

*PhD of physic-mathematical science, assistant professor of applied mathematics
and calculated techniques department of «National Metallurgical Academy», Ukraine*

D 18 Danilishyn I., Danilishyn O. Mathematical fundamentals of constructing pseudo-living energy theory and dynamic programmings: monograph / ed. by Pasyнков V. – Shawnee, USA : Primedia eLaunch LLC. – 2025. – 382 p.

ISBN 979-8-89660-275-0

DOI 10.36074/mfocp-letadp.monograph-2025

The monograph task: to understand hierarchy of energies in the Universe and the principles of functioning of living energy (living organism, in particular, human, subtle energies), and then using these principles to "construct" artificial living energies (let's call them pseudo-living energies). It is possible to significantly expand the horizons of science, in particular physics, by studying the subtle energies in the Universe. Here is considered paradoxical singularities (singularities of disintegration&synthesis), self-type singularities. Considering object A as the value of dynamic variable P, we can reassign $P := B$ at the top level and instead of A we will have object B, and also perform any other parallel dynamic operations with objects. That is, our world will be perceived as a Dynamic Programming environment. Dynamic programming is an extension of science for "working" with the top and middle levels. Here information is transformed into actions. Energy Internet: a) transmission of necessary singularities (subtle energies) (connection to necessary singularities (subtle energies)), b) transmission of necessary self-structures (subtle energies) (connection to necessary self-structures) etc. It is also necessary to construct a bioSmnsprt on living, grown elements of the central nervous system, in particular, neurons. It is also possible to construct another version of bioSmnsprt, using a viral model, on living, grown bacteriophages. We have simply outlined some aspects; it is too early to develop them without experimental study. All problems have a solution, the only question is: a) in complete identification with the problem, b) the "breadth of the field" of values that you can provide. Our mathematics is unusual for a mathematician, because here the fulcrum is the action, and not the result of an action, as in classical mathematics. Therefore, our mathematics is adapted not only to obtain results, but also to directly control actions, which will certainly show its advantages on a fundamentally new type of neural networks with directly parallel calculations, for which it was created. Any action has much greater potential than its result. The significance of the monograph: in a new qualitatively different approach to the study of complex processes through new mathematical hierarchical parallel dynamic structures, in particular those processes that are studied by synergetics. Our approach is not based on deterministic equations that generate self-organization, which is very difficult to study and gives very small results for a very limited class of problems and does not provide the most important thing - the structure of self-organization. We simply start from the assumed structure of self-organization, since we are interested not so much in the numerical calculation of this, but in the structure of self-organization itself, its formation (construction) for the necessary purposes and its management. Although we are also interested in numerical calculations. Laureates of the Nobel Prize in Physics 2023 Ferenc Kraus and his colleagues Pierre Agostini and Anne L'Huillier used a short-pulse laser to generate attosecond pulses of light to study the fuzzy dynamics of electrons in matter. According to our Synthesis-Type Singularity Theory, its action corresponds to a singularity, allowing the upper level of subtle energies to be reached to manipulate the lower levels. In April 2023, we proposed to use a short-pulse laser to achieve desired goals using a directly parallel neural network. We then proposed a fundamental development of this directly parallel neural network. The significance of our monograph lies in the formation of the proposed mathematical structure of subtle energies, this is done for the first time in science, and the proposed classification of mathematical structures of subtle energies for the first time. The experiments of 2022 Nobel Prize winners Asle Allen, John Clauser, Anton Zeilinger and the chemistry experiments of Nazhipa Valitov eloquently demonstrate, that we are right and that these studies are necessary. Be that as it may, we have created classes of new mathematical structures, new fuzzy mathematical singularities, i.e. we have contributed to the development of mathematics. The articles [1] – [14], [16, 17] and scientific monographs [15], [18] dedicated to the mathematical foundations of constructing pseudo-living energy. Dynamic programming works in the space of energies with energies. It is enough to imagine the world as a software environment. Then objects and processes will act as variables etc. The subjects of variables are processes of a higher level manipulating them. Dynamic programming will do what is needed if the level and volume of energies allow, i.e., much more effectively than science will allow, as well as our supposed devices on Dynamic programming.

UDC 004:51(07)

ISBN 979-8-89660-275-0

© Danilishyn I., Danilishyn O., 2025
© Primedia e-Launch LLC, 2025

AUTHORS:



Illia Danilishyn

Graduated from Sumy State University.
Currently a postgraduate student at Sumy State University.



Oleksandr Danilishyn

Graduated from Sumy State University.
Currently works as a practicing physician.

Competing interest:

There are no competing interests. All sections of the monograph are executed jointly.

Funding:

There were no sources of funding for writing the monograph.

Authors' contributions:

The contribution of the authors is the same, we will not separate.

Foreword:

In this book, the authors develop the themes of the books [1], [2] in more interesting dynamic structures. Here is considered paradoxical singularities (singularities of disintegration&synthesis), self-type singularities and Dynamic programming. Significance of the monograph: in a new qualitatively different approach to the study of complex processes through new mathematical hierarchical dynamic structures, in particular those processes that are dealt with by Synergetics. Authors' approach is not based on deterministic equations that generate self-organization, which is very difficult to study and gives very small results for a very limited class of problems and does not provide the most important thing - the structure of self-organization. They are just starting from the assumed structure of self-organization, since they are interested not so much in the numerical calculation of this as in the structure of self-organization itself, its formation (construction) for the necessary purposes and its management. Although they are also interested in numerical calculations. Nobel laureates in physics 2023 Ferenc Kraus and his colleagues Pierre Agostini and Anna Lhuillier used a short-pulse laser to generate attosecond pulses of light to study the dynamics of electrons in matter. According to their Theory of singularities of the type synthesizing, its action corresponds to singularity $\uparrow I \downarrow q h$, which allows one to reach the upper level of subtle energies to manipulate lower levels. In April 2023 [14], the authors proposed using a short-pulse laser to achieve the desired goals by a directly parallel neural network. They then proposed the fundamental development of this directly parallel neural network. There is a long overdue need for the use of singular hierarchical structures, in particular self-sets, to describe complex processes, in particular to describe unusual states of consciousness and pathological conditions in medicine. The experiments of Nobel laureates in 2022-year Asle Ahlen, Clauser John, Zeilinger Anton correspond to the concept of the Universe as its self-containment in itself. The monograph aims to create new constructive hierarchical mathematical objects for new technologies, particularly for a fundamentally new type of neural network with parallel computing and not the usual parallel computing through sequential computing. Based on fundamentally new type of neural network with parallel computing, it is possible to create a propeller-less helicopter, a space shuttle, medical equipment and various other effective and unusual equipment, in particular, household equipment. Dynamic programming works in the space of energies with energies. It is enough to imagine the world as a software environment. Then objects and processes will act as variables etc. The subjects of variables are processes of a higher level manipulating them.

Contents	5
Introduction	9
Part I. SCprt – elements and their applications.....	10
Introduction.....	10
1.1 SCprt– elements	13
1.2 Dynamic SCprt – elements	26
1.3 SCprt – elements for continual sets	32
1.4 Dynamic continual SCprt – elements.....	36
1.5 The usage of SCprt-elements for networks.....	41
1.6 Variable hierarchical dynamical structures (models) for dynamic, singular, hierarchical sets.....	44
1.7 Applications Dynamic Sets Theory to physics and chemistry	48
1.8 PROGRAM OPERATORS SCprt, tprSC, SC ¹ epr, SCept ₁	54
Appendix 1	62
Supplement: Connection SCprt – elements with usual functionals and operators.....	65
References.....	67
Part II. Fuzzy SCprt – elements and their applications.....	73
Introduction.....	73
2.1 SfCprt– elements	73
2.2 Dynamic SfCprt – elements	88
2.3 Fuzzy SCprt – elements for continual fuzzy sets...	94
2.4 Dynamic continual SfCprt – elements	99
2.5 The usage of SfCprt-elements for networks.....	104

2.6 Variable fuzzy hierarchical dynamic fuzzy structures (models, operators) for dynamic, singular, hierarchical fuzzy sets	108
2.7 Applications Dynamic Fuzzy Sets Theory to physics and chemistry	111
2.8 FUZZY PROGRAM OPERATORS SfCprt, tprSfC, ffS ¹ epr, ffSeprt ₁	119
Appendix.....	129
Supplement.....	132
References.....	134
Part III. Wprt-elements and Their Applications	140
A Wprt – elements, self-type Wprt-structures	140
A.1 Wprt – elements, self-type Wprt-structures	173
A.2 FWprt – elements, self-type FWprt- structures	205
A.3 Elements of the theory of variables of fuzzy hierarchical fuzzy dynamic operators: FWprt	216
A.4 Introduction to FUZZY PROGRAM OPERATORS fWprt, ftp ^r W, fW ¹ epr, fWeprt ₁	218
A.5 Paradoxical singularities (singularities of disintegration&synthesis)	220
A.6 Singularities algebra	222
A.7 Types, forms internal and external, structures of self and 	225
A.8 Types, forms internal and external, structures of potential self, potential and others potential singularities.....	226
A.9 Some available types of "pseudo-living energy"	227
A.10 Wprt-networks	234
A.11 Some aspects of directly parallel operation of neural networks	

Appendix	236
References.....	239
Part IV. Mathematical fundamentals of the Dynamic programming	248
Introduction	248
4.1 Program operators Sprt-type	249
4.1.1 Program operators Sprt-type	249
4.1.2 The usage of Sprt-elements for networks	252
4.2 Program operators ffSprt-type	255
4.2.1 The usage of ffSprt-elements for networks	258
4.3 Introduction to FUZZY PROGRAM OPERATORS	
fDprt, ftprD, fD ¹ epr, fDeprt ₁	261
4.4 FUZZY PROGRAM OPERATORS Rprt, tprR, R ¹ epr, Reprt ₁	272
4.5 Rprt-networks	284
4.6 Introduction to FUZZY PROGRAM OPERATORS	
SDS, tSDS, fD ¹ epr, fDeprt ₁ .	290
4.7 Introduction to FUZZY PROGRAM OPERATORS	
SRS, tSRS, fR ¹ epr, fReprt ₁	300
4.8 Program operators PrSprt, PrtSpr, S ¹ e, Set ₁	310
4.9 Program operators PrfSprt, PrtfSpr, fS ¹ e, fSet ₁	322
4.10 Program operators PrffSprt, PrtffSpr, S ¹ e, Set ₁	335
4.11 Introduction to FUZZY PROGRAM OPERATORS	
Flprt, tprFL, FL ¹ epr, FLeprt ₁	348
4.12 Other PROGRAM OPERATORS	350
4.13 Some aspects of directly parallel operation of neural networks and their subtle energy	351
References.....	356
Supplement 1: S ¹ t - program operators.....	362

Conclusions	369
References.....	372

Introduction

There is a need to develop an instrumental mathematical base for new technologies. The task of the work is to create new approaches for this by introducing new concepts and methods. Our mathematics is unusual for a mathematician, because here the fulcrum is the action, and not the result of the action as in classical mathematics. Therefore, our mathematics is adapted not only to obtain results, but also to directly control actions, which will certainly show its benefits on a fundamentally new type of neural networks with directly parallel calculations, for which it was created. Any action has much greater potential than its result. Significance of the work: in a new qualitatively different approach to the study of complex processes through new mathematical hierarchical parallel fuzzy dynamic structures, in particular those processes that are dealt with by Synergetics. An example of studying complex processes is quantum mechanics. It studies them through randomness (probability), i.e., it obtains some certainty through uncertainty. Our approach to studying complex processes concerning hyperfine energies is in classifying the principles of functioning of models of some complex processes, that we propose, in order to construct some of them at the biotechnical level and manipulate them (certainty through certainty). We define them and then certain approaches to them. Although their manipulation can be carried out with uncertainty and this quite naturally follows from their nature.

Our approach is not based on deterministic equations that generate self-organization, which is very difficult to study and gives very small results for a very limited class of problems and does not provide the most important thing - the structure of self-organization. We are just starting from the assumed structure of self-organization, since we are interested not so much in the numerical calculation of this as in the structure of self-organization itself, its formation (construction) for the necessary purposes and its management. By extending the concept of a set to a self-set, we avoid the paradoxes of ordinary set theory. Although we are also interested in numerical calculations. Nobel laureates in physics 2023 Ferenc Kraus

and his colleagues Pierre Agostini and Anna Lhuillier used a short-pulse laser to generate attosecond pulses of light to study the dynamics of electrons in matter. According to our Theory of singularities of the type synthesizing, its action corresponds to singularity $\uparrow I \downarrow_h^q$, which allows one to reach the upper level of subtle energies to manipulate lower levels. In April 2023 [14], we proposed using a short-pulse laser to achieve the desired goals by a directly parallel neural network. We then proposed the fundamental development of this directly parallel neural network. In the article [17] new mathematical structures and operators are constructed through action - “fuzzy containment”. Here, the construction of new mathematical structures and operators is carried out with generalization to “type of accommodation”. The significance of our articles is in the formation of the presumptive mathematical structure of subtle energies, this is being done for the first time in science, and the presumptive classification of the mathematical structures of subtle energies for the first time. The experiments of the 2022 Nobel laureates Asle Ahlen, John Clauser, Anton Zeilinger and the experiments in chemistry Nazhipa Valitov eloquently demonstrate that we are right and that these studies are necessary. We transform all kinds of uncertainties into new singularities and use them to describe uncertain energies. The work is too unconventional and not every traditional scientist would have the courage to acknowledge it. Be that as it may, we created classes of new mathematical structures, new mathematical singularities, i.e., made a contribution to the development of mathematics.

Part I. SCprt – elements and their applications

Introduction

There is a need to develop an instrumental mathematical base for new technologies. Our constructive approach to set theory differs from the construction of constructive sets by A. Mostowski [1]: we construct completely different types of constructive sets. Here, the axiom of regularity (A8) [2] is removed from the axioms of set theory, so we naturally obtain the possibility of using singularities in the form of self_g -sets, self_g -elements, which is exactly what we need for new

mathematical models for describing complex processes. Instead of the axiom of regularity, we introduce the following axioms: Axiom R1. $\forall \text{ set } B \text{ set } \overset{B}{\underset{B}{\text{SCprt}g}}$ is an element of itself_g. Let denote $\overset{B}{\underset{B}{\text{SCprt}g}}$ by $\text{self}_g B$.

Axiom R2. $\forall B \exists (\text{self}_{g_1} B)^{-1}$.

The task of the work is to create new approaches for this by introducing new concepts and methods. Significance of this work: in a new qualitatively different approach to the study of complex processes through new mathematical, hierarchical, dynamic structures, in particular those processes that are dealt with by Synergetics.

We consider expression

$$\begin{array}{cc} C & A \\ g_2 \text{SCprt} g_1 & \\ D & B \end{array} \quad (*_{1.1})$$

where A fits into B with type of accommodation g_1 , D is forced out from C with type of accommodation g_2 ; A, B, C, D, g_1 , g_2 may also be fuzzy. The result of this process will be described by the expression

$$\begin{array}{cc} C & A \\ g_2 \text{SCprt} g_1 & \\ D & B \end{array} \quad (*_{1.2})$$

If A, B, D, C are taken as sets, then we will call $(*_{1.1})$ a SC-dynamic set. The need $(*_{1.1})$ arose to describe processes in networks. Threshold element SCprt --

$\overset{b}{g_2 \text{SCprt} g_1} \begin{array}{c} \{ax\} \\ b \end{array}$, b- artificial neurons of type SCprt (designation - mnSt), $x = (x_1, x_2, \dots, x_n)$ are the values of the initial signals,, $a = (a_1, a_2, \dots, a_n)$ are the weights of SCprt-synapses and the values of the output signals . It can be considered a simpler version of the dynamic set

$$\underset{B}{\text{SCprt}}_{g_1} \left(\overset{A}{**}_{1.1} \right)$$

where set A fits into set B with type of accommodation g_1 , the result of this process will be described by the expression

$$\underset{B}{\text{SCrt}}_{g_1} \left(\overset{A}{**}_{1.2} \right)$$

or

$$\underset{D}{g_2 \text{SCprt}} \left(\overset{C}{***}_{1.1} \right)$$

where set A is forced out from B with type of accommodation g_2 , the result of this process will be described by the expression

$$\underset{D}{g_2 \text{SCrt}} \left(\overset{C}{***}_{1.2} \right)$$

We consider the measure: $\underset{D}{\mu^{**}} \left(\overset{b}{g_2 \text{SCprt}}_{g_1} \right) = \frac{\mu(A)\mu(g_1)}{\mu(D)\mu(g_2)}$, where $\mu(A), \mu(D)$ – usual measures of sets A, D, $\mu(g_1), \mu(g_2)$ – measures corresponding to the accommodations of the corresponding type.

Remark. One can consider some generalization for $(*_{1.1})$: $\underset{D}{q_1(C)} \underset{q(B)}{\text{SCprt}}_{g_1} \overset{A}{g_1}$, where A is contained into B through q with type of accommodation g_1 , D is forced out from C through q_1 with type of accommodation g_2 , A, B, D, C are taken as sets. The result of this process will be described by the expression $\underset{D}{q_1(C)} \underset{q(B)}{\text{SCrt}}_{g_1} \overset{A}{g_1}$.

Similarly, for $(**_{1.1})$: $\underset{q(B)}{\text{SCprt}}_{g_1} \overset{A}{g_1}$, where A is contained into B through q with type of

accommodation g_1 (the result of this process will be described by the expression

$\underset{q(B)}{\text{SCrt}}_{g_1} \overset{A}{g_1}$), for $(***_{1.1})$: $\underset{D}{q_1(C)} \underset{q(B)}{\text{SCprt}}_{g_1}$, where D is displaced from C through q_1 with

type of accommodation g_2 . The result of this process will be described by the expression $q_1(C)_{g_2}^{D}$ SCrt.

We construct new mathematical objects constructively without formalism. By its contradiction, formalism may destroy this theory by Gödel's theorem on the incompleteness of any formal theory. But in the next monograph, we will give the formalism of the theory it's due: the proof of axioms and theorems. Let us introduce the concepts Cha, the capacity measure, and Cca, the measure of its content. Cca is the same as the number of capacity content items. Consider the

compression ratios of the dynamic set: $q_1 = \text{SCprt}_{g_1}^A$ answers I compression power of dynamic set A, $q_2 = \text{SCprt}_{g_1}^A$ — II compression power of dynamic set A, ...,

$q_{n+1} = \text{SCprt}_{g_1}^A$ — n+1 compression power of dynamic set A. In contrast to the classical one-attribute set theory, where only its contents are taken as a set, we consider a two-attribute set theory with a set as a capacity and separately with its contents. We introduce the designations: CoQ—the contents of the capacity Q.

1.1 SCprt - elements

Definition 1.1.1. The set of elements $\{a\} = (a_1, a_2, \dots, a_n)$ contained into point x of space X with accommodation type g_1 we shall call SCprt – element, and such a point in space is called capacity of the SCprt – element. We shall denote $\text{SCprt}_{g_1}^{\{a\}}_x$.

The result of this process will be described by the expression $\text{SCpr}_{g_1}^{\{a\}}_x$.

Definition 1.1.2. $\text{SCprt}_{g_1}^{\{a\}}_x$ — SC-dynamic set $\{a\}$ at x.

Definition 1.1.3. An ordered set of elements at one point in space is called an ordered SCprt–element.

It's possible to $\text{SCprt}_{g_1}^{\{a\}}$ correspond to the set of elements $\{a\}$, and the ordered SCprt - element - a vector, a matrix, a tensor, a directed segment in the case when the totality of elements is understood as a set of elements in a segment.

It's allowed to sum SCprt – elements: $\text{SCprt}_{g_1}^{\{a\}} + \text{SCprt}_{g_1}^{\{b\}} = \text{SCprt}_{g_1}^{\{a\} \cup \{b\}}$.

The operator $\text{SCprt}_{g_1}^{\{a\} \cup \{b\}}$ is not equal $\{a\} \cup \{b\}$, rather, it is dynamic —

contraction of $\{a\} \cup \{b\}$ to the point x . Similarly, for $\text{SCprt}_{g_1}^{\{a\} \cap \{b\}}$. This is more suitable for using sets for energy space, for any objects. The operator SCprt is adapted for ordinary energies, using their property to overlap.

Capacity in itself_g

Definition 1.1.4. The capacity A in itself_g of the first type is the capacity containing itself_g as an element with accommodation type g_1 . Denote $SC_1fA\{g_1\}$.

Definition 1.1.5. The capacity A in itself_g of the second type is the capacity that contains elements from which it can be generated with accommodation type g_1 . Denote $SC_2fA\{g_1\}$.

An example of the capacity in itself_g of the first type is a set containing itself_g. An example of capacity in itself_g of the second type is a living organism since it contains a program: DNA and RNA.

Definition 1.1.6. Partial capacity A in itself_g of the third type is the capacity A in itself_g as an element with accommodation type g_1 , which partially contains itself_g or contains elements from which it can be generated in part with accommodation type g_1 or both. Let us denote $SC_3fA\{g_1\}$.

Let us introduce the following notations: $A*B = \text{scrt}_{g_1}^{\frac{A}{B}}$, $A^2 = \text{self}_{g_1} A = \text{scrt}_{g_1}^{\frac{A}{A}}$, $A^3 = \text{self}_{g_1}^2 A$, ..., $A^{n+1} = \text{self}_{g_1}^n A$, ... There is no commutativity here: $A*B \neq B*A$. We

can consider operator functions: $e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots$, $(A + B)^n = \sum_{k=0}^n \binom{n}{k} A^k B^{n-k}$, $(1 + A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \dots$, etc.

You can consider a more “hard” option: $A * B = PSC_{prt} g_1$, where $PSC_{prt} g_1$ – operator,

containing A in every element of B, $A^2 = P \text{ self}_{g_1} A = PSC_{prt} g_1 A$, $A^3 = P \text{ self}_{g_1}^2 A$, ..., $A^{n+1} = P \text{ self}_{g_1}^n A$, ... There is no commutativity here: $A * B \neq B * A$. We can

consider operator functions: $e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots$, $(A + B)^n = \sum_{k=0}^n \binom{n}{k} A^k B^{n-k}$, $(1 + A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \dots$, etc.

All capacities in self_g -space are capacities in themselves by definition. Capacities in themselves can appear as SCprt -capacities and ordinary capacities. In these cases, the usual measures and methods of topology are used.

Connection of SCprt – elements with capacities in themselves.

For example, $SC_{prt} \frac{\{R\}}{w\{R\}}$ is the capacity in self_g of the second type if $w\{R\}$ is a program capable of generating $\{R\}$.

Consider a third type of capacity in self_g . For example, based on $SC_{pr} \frac{\{a\}}{x}$, where

$\{a\} = (a_1, a_2, \dots, a_n)$, i.e. n - elements at one point, we can consider the capacity $SC_3 f$ in self_g with m elements from $\{a\}$, $m < n$, which is formed according to the form:

$$w_{mn} = (m, (n, 1)) \quad (1.1)$$

that is, the structures $C_{pr} \frac{\{a\}}{x}$ contains only m elements. Form (1.1) can be

generalized into the following forms:

$$w_{m,n,k}^1 = (k, \left(\begin{matrix} (n_1, 1) \\ \dots \\ (n_m, 1) \end{matrix} \right)) \quad (1.1.1)$$

or

$$w_{m,n,k}^2 = (k, (l, \overset{(n_1)}{(\dots)})) \overset{(n_m)}{(1.1.2)}$$

$$w_{m,n,k,l}^3 = Q(\overset{d_1}{(\dots)}, (\overset{(n_1, 1)}{(\dots)})) \overset{d_l}{(n_m, 1)} (1.1.3),$$

where $Q(x, y)$ – any operator, which makes a match between set $\overset{d_1}{(\dots)}$ and set $\overset{d_l}{(n_m, 1)}$

$$\overset{(n_1, 1)}{(\dots)} \text{ or } \overset{(n_m, 1)}{(n_m, 1)}$$

$$w_{m,m_1,n_1,m_2,n_2,m_3,n_3}^4 = (m, ((m_1, n_1), ((m_2, n_2), (m_3, n_3)))) (1.1.4),$$

or

$$w_{m_1,n_1,\dots,m_k,n_k,l_k,q_k,s_k,r_k}^5 = (n_1, (m_1(\dots(n_k, (m_k(l_k, (q_k, s_k)), r_k))\dots)) (1.1.4.1),$$

or

$$w_{B_1,A_1,\dots,B_k,A_k,Q_k,R_k,D_k,P_k}^6 = (A_1, (B_1(\dots(A_k, (B_k(Q_k, (R_k, D_k)), P_k))\dots)) (1.1.4.2),$$

or

$$w_{m_1,n_2,\dots,m_k,n_k,l_k,q_k,s_k,r_k}^5 = (m_1(\dots(n_k, (m_k(l_k, (q_k, s_k)), r_k))\dots)) (1.1.4.3),$$

or

$$w_{B_1,A_2,\dots,B_k,A_k,Q_k,R_k,D_k,P_k}^6 = (B_1(\dots(A_k, (B_k(Q_k, (R_k, D_k)), P_k))\dots)) (1.1.4.4),$$

or

$$(Q, R) (1.1.5),$$

where Q – any, R – any structure, R could be anything can be anything, not just structure. In this case, (1.1.5) can be used as another type of transformation from Q to R . Capacities in themselves of the third type can be formed for any other structure, not necessarily SCpr, only by necessarily reducing the number of elements in the structure, in particular, using form

$$w_{m_1 \dots m_n} = (m_1, (m_2, (\dots(m_n, 1)\dots))) (1.2)$$

Structures more complex than SC_{3f} can be introduced. For example, through a form that generalizes (1.1):

$$w_{ABC} = (A, (B, C)) \quad (1.3)$$

where A is compressed (fits) in C in the compression structure B in C (i.e., in the structure $\text{SCrt}_{g_1}^B$); or

$$w_{ABC} = A \quad \begin{array}{c} B \\ \diagdown \quad \diagup \\ \triangle \\ \diagup \quad \diagdown \\ C \end{array} \quad (1.3.1)$$

$$\begin{array}{c} A \quad (B,C) \\ \diagdown \quad \diagup \\ G \quad \triangle \\ \diagup \quad \diagdown \\ Q \end{array} \quad (1.3.2)$$

$$\begin{array}{c} A \quad R \quad Q \\ \diagdown \quad \diagup \quad \diagdown \quad \diagup \\ \triangle \quad \triangle \\ \diagup \quad \diagdown \\ S \end{array} \quad (1.3.3)$$

$$\begin{array}{c} A \quad L \\ \diagdown \quad \diagup \\ q \quad \triangle \quad i \\ \diagup \quad \diagdown \end{array} \quad (1.3.4)$$

or through the more general form that generalizes (1.2):

$$w_{A_1 A_2 \dots A_n C} = (A_1, (A_2, (\dots (A_n, C) \dots))) \quad (1.4)$$

and corresponding generalizations of (1.4) on (1.3.1) - (1.3.4), etc.

(1.3), (1.4) are represented through the usual 2-bond. Science is the discipline of 2-connections, since everything in science is carried out through 2-connected logic, quantum logic is also a projection of 3-connected logic onto 2-connected logic. (1.3.1) - (1.3.4) schematically interpret the formation of the capacity in self_g through a pseudo 3-connected form with a 2-connected form. The energy of the self_g -accommodation in self_g closes on self_g .

Math self_g .

Let's consider SCprt arithmetic first:

1. Simultaneous addition of a set of elements $\{a\} = (a_1, a_2, \dots, a_n)$ is carried

out using $\text{SCprt}_{g_1, x}^{\{a+\}}$.

2. Similarly, for simultaneous multiplication: $\text{SCprt}_{g_1, x}^{\{a*\}}$ gives the set B with

elements $b_{i_1 i_2 \dots i_n} =$

$(\text{SCprt}_{g_1, x}^{\{a_{1i_1} *, a_{2i_2} *, \dots, a_{ni_n}\}})_R$ for any $\{i_1, i_2, \dots, i_n\}$ without repetitions, $x =$

$\text{SCprt}_{g_1, w}^{\{K\}}$, K-set of any $\{k_1 *, k_2 *, \dots, k_n *\}$ without repeating them, k_i -any

digit, $i=1, 2, \dots, n$, $R = \text{SCprt}_{g_1, w}^{\{i_1 +, i_2 +, \dots, i_n\}}$, R is the index of the lower

discharge (we choose an index on the scale of discharges):

Table I.1. Index on the scale of discharges

index	discharge
n	n
...	...
1	1
,	0
-1	1st digit to the right of the point
-2	2nd digit to the right of the point
...	...

Then $\text{SCprt}_{g_1, w}^{\{B+\}}$ gives the final result of simultaneous multiplication. Any

system of calculus can be chosen, in particular binary. The most straightforward functional scheme of the assumed arithmetic-logical device for SCprt-multiplication:

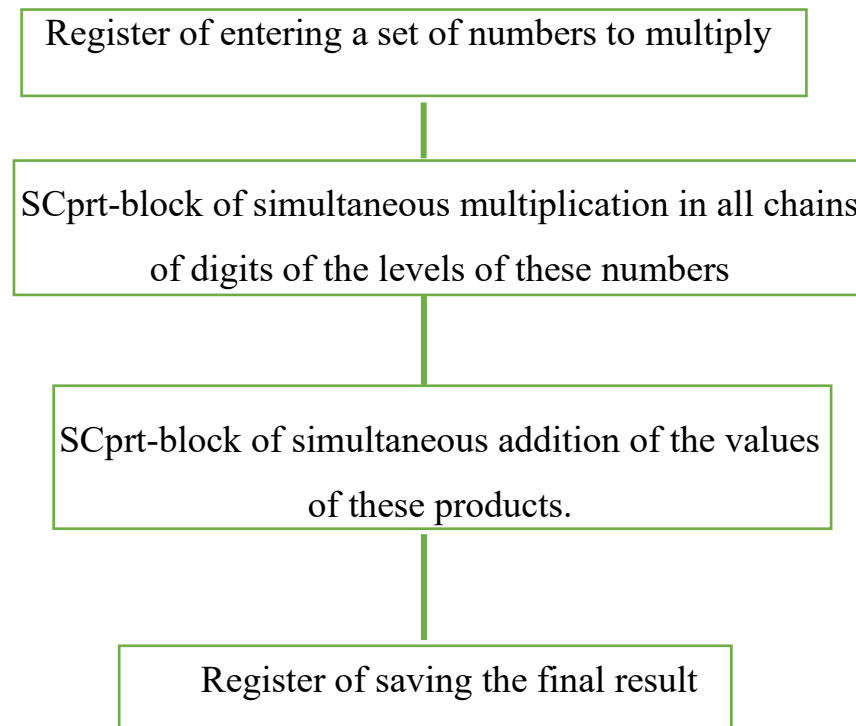


Fig. I.1. The straightforward functional scheme of the assumed arithmetic-logical device for SCprt-multiplication.

Remark 1.1.1. The algorithm for simultaneously multiplication of a set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by multiplying the first number from the set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the set by the ones following it, etc.

3. Similarly for simultaneous execution of various operations: $SCprt \begin{matrix} \{aq\} \\ g_1 \\ x \end{matrix}$, where $\{q\} = (q_1, q_2, \dots, q_n)$. q_i -an operation, $i = 1, \dots, n$.

4. Similarly, for the simultaneous execution of various operators: $SCprt_{g_1}^{\{Fa\}}$,
 x
 where $\{F\} = (F_1, F_2, \dots, F_n)$. F_i is an operator, $i = 1, \dots, n$.
5. The arithmetic itself_g for capacities in themselves will be similar: addition - $SC_1f\{a +\}$, (or $SC_3f\{a +\}$) for the third type), multiplication $SC_1f\{a *\}$, $(SC_3f\{a *\})$.
6. Similarly with different operations: $SC_1f\{aq\}$, $(SC_3f\{aq\})$, and with different operators: $SC_1f\{Fa\}$, $(SC_3f\{Fa\})$.

7. $SCrt_{g_1}^{\frac{A}{B}}$ – the result of the accommodation operator. For sets A, B we have

$$SCrt_{g_1}^{\frac{A}{B}} = \left\{ \frac{D}{A \cup B - A \cap B} \right\}, \text{ where } D \text{ is self}_{g_1}\text{-set for } A \cap B. \text{ The measure:}$$

$$\mu(SCpr_{g_1}^{\frac{A}{B}}) = \left(\frac{\mu_{g_1}(A \cap B)}{\mu(A) + \mu(B) - \mu(A \cap B)} \right) * \mu(g_1).$$

There is the same for structures if it's considered as sets. Our approach to the theory of hierarchical sets differs from the construction of hierarchical sets by Y.L. Ershov [4]-[6] : we construct completely different types of hierarchical sets.

8. SCprt-derivative of $f(x_1, x_2, \dots, x_n)$ is $SCprt_{g_1}^{\left\{ \frac{\partial}{\partial x_{1_i}}, \frac{\partial}{\partial x_{2_i}}, \dots, \frac{\partial}{\partial x_{k_i}} \right\}}$, where
 $f(x_1, x_2, \dots, x_n)$
 $x = (x_{1_i}, x_{2_i}, \dots, x_{k_i})$ – any set from $(x_1, x_2, \dots, x_n)$. Let's designate SCprt-
 $\frac{\partial^k f(x)}{\partial x_{1_i} \partial x_{2_i} \dots \partial x_{k_i}}$. SCprt-integral off $(x_1, x_2, \dots, x_n)$ is
 $SCprt_{g_1}^{\left\{ \int \circ dx_{1_i}, \int \circ dx_{2_i}, \dots, \int \circ dx_{k_i} \right\}}$, where $(x_{1_i}, x_{2_i}, \dots, x_{k_i})$ – any set from
 $f(x_1, x_2, \dots, x_n)$. Let's designate SCprt- $\int \dots \int f(x) dx_{1_i} dx_{2_i} \dots dx_{k_i}$ – k-multiple
 integral. SCprt-lim off $(x_1, x_2, \dots, x_n)$ is $SCprt_{g_1}^{\left\{ \lim_{x_{1_i} \rightarrow a_{1_i}}, \lim_{x_{2_i} \rightarrow a_{2_i}}, \dots, \lim_{x_{k_i} \rightarrow a_{k_i}} \right\}}$.
 $f(x_1, x_2, \dots, x_n)$

Let's designate $\text{SCprt-} \lim_{\substack{x_{1i} \rightarrow a_{1i} \\ \vdots \\ x_{ki} \rightarrow a_{ki}}} f(x_1, x_2, \dots, x_n) \cdot \text{self}_{g_1} \text{-} \lim_{x \rightarrow a} =$

$$\lim_{x \rightarrow a} \text{SCrt } g_1 .$$

9. In the case of self_g -derivatives, inclusions of multiple derivatives are obtained. The same is true for self_g -integrals: we get inclusions of multiple integrals.

10. Let's denote $\text{self}_{g_1} \text{-(self}_{g_1} \text{-Q)}$ through $\text{self}_{g_1}^2 \text{-Q}$, $\text{fS}_{g_1}(n, Q, g_1) = \text{self}_{g_1} \text{-(self}_{g_1} \text{-(... (self}_{g_1} \text{-Q)))} = \text{self}_{g_1}^n \text{-Q}$ for n -multiple self_{g_1} .

Operator itself_g.

Definition 1.1.7. An operator that transforms $\text{SCprt } g_1^{\{a\}}$ into any $\text{SC}_i f\{b\}\{g_1\}$, $i =$

2, 3; where $\{b\} \subset \{a\}$; is the operator itself_g.

Example. The operator contains the set in itself_g.

Lim-itself_g.

1. Lim SCprt

For example, the double limit: $\lim_{\substack{x \rightarrow a_1 \\ y \rightarrow a_2}} G(x, y)$ corresponds to $\text{SCprt } g_1^{\{G(x, y)\}}_{(a_1, a_2)}$.

Similarly, for SCprt-lim with n variables.

In the case of lim-itself_g, for example, for m variables, it suffices to use the form (1.1) of lim SCprt for n variables ($n > m$). The same is true for integrals of variables m (for example, the double integral over a rectangular region is through the double limit).

The sequence of actions can be "collapsed" into an ordered SCprt element, and then translate it, for example, into $\text{SC}_3 f$ – the capacity in itself_g. Take the

receipt $\frac{\partial^2 u}{\partial x^2}$ as an example. Here is the sequence of steps 1) $\frac{\partial u}{\partial x} \rightarrow 2) \frac{\partial}{\partial x} \left(\frac{\partial u}{\partial x} \right)$.

"collapses" into an ordered $\text{SCprt}_{\frac{\partial}{\partial x}, \frac{\partial}{\partial x} (\frac{\partial u}{\partial x})}{g_1, x}$, which can be translated into the

corresponding $SC_1 f$. The differential operator $\text{SCprt}_{\frac{\partial}{\partial x}, \frac{\partial}{\partial x} (\frac{\partial u}{\partial x})}{g_1, x}$ - is interesting too.

We can consider the concept of $\text{SCprt}_{g_1, \frac{A}{B}}$, where A fits in capacity

B. Then $\text{SCprt}_{g_1, \frac{B}{B}}$ it will mean $SC_1 f B \{g_1\}$.

About SCprt and $SC_3 f$ programming.

The ideology of SCprt and $SC_3 f$ can be used for programming. Here are some of the SCprt programming operators:

1. Simultaneous assignment of the expressions $\{p\} = (p_1, p_2, \dots, p_n)$ to the variables $\{a\} = (a_1, a_2, \dots, a_n)$. This is implemented via $\text{SCprt}_{\frac{\{a\} := \{p\}}{g_1, x}}$.
2. Simultaneous checking the set of conditions $\{f\} = (f_1, f_2, \dots, f_n)$ for the set of expressions $\{B\} = (B_1, B_2, \dots, B_n)$. Implemented via Sprt_x

$\text{SCprt}_{\frac{IF \{ \{B\} \{f\} \}}{g_1, x}} \text{ then } Q$, where Q can be anything.

3. Similarly for loop operators and others.

$SC_3 f$ – software operators will differ only in that the aggregates $\{a\}, \{p\}, \{B\}, \{f\}$ will be formed from the corresponding SCprt program operators in form (1.1) and for more complex operators in the form from the forms (1.1.1) – (1.4).

The OS (operating system), the computer's principles, and the modes of operation for this programming are interesting. But this is already the material for the following publications.

Using elements of the mathematics of SCprt , we introduce the concept of SCprt –

the change in physical quantity B: $\text{SCprt}_{\frac{\{\Delta_1 B, \dots, \Delta_n B\}}{g_1, x}}$. Then the mean SCpr –

velocity will be $v_{\text{cpscpr}}(t, \Delta t) = \text{SCprt}_{g_1, x}^{\{\frac{\Delta_1 B}{\Delta t}, \dots, \frac{\Delta_n B}{\Delta t}\}}$ and SCprt-velocity at time t : $v_{\text{sct}} = \lim_{\Delta t \rightarrow 0} v_{\text{cpscpr}}(t, \Delta t)$. SCprt – acceleration: $a_{\text{sct}} = \frac{dv_{\text{sct}}}{dt}$.

When using SCprt with "target weights", we get, depending on the "target weights", one or another modification, namely, for example, the velocity v_{sct}^f (with a "target weight" f in the case when two velocities v_1, v_2 are involved in the set $\{v_1 f, v_2\}$ for $v_{\text{st}}^f = \text{SCprt}_{g_1, x}^{\{v_1 f, v_2\}}$, f – instantaneous replacement we get an instantaneous substitution v_1 by v_2 at point x of space at time t_0 with accommodation type g_1 .

Consider, in particular, some examples: 1) $\text{SCprt}_{g_1, \{x_1, x_2\}}^e$ describes the presence of the same electron e at two different points x_1, x_2 . 2) The nuclei of atoms can be considered as SCprt elements.

Similarly, the concepts of SCprt - force and SCprt - energy are introduced .

For example, $E_{\text{st}}^f = \text{SCprt}_{g_1, x}^{\{E_1 f, E_2\}}$ it would mean the instantaneous replacement of energy E_1 by E_2 at time t_0 with accommodation type g_1 . Two aspects of SCprt–energy should be distinguished: 1) carrying out the desired "target weight" and 2) fixing the result of it. Do not confuse energy - SCprt (the node of energies) with SCprt – energy that generates the node of energies, usually with the "target weights." In the case of ordinary energies, the energy node is carried out automatically.

Remark 1.1.2. SCprt – elements are all ordinary, but with "target weights," they become peculiar. Here you need the necessary energy to carry them out. As a rule, this energy is at the level of self_g . This is natural since it's much easier to manage elements of the k level via the elements of a more structured $k + 1$ level. Let us

consider the concepts of capacities of physical objects in themselves. The question arises about the self_g-energy of the object. In particular, $SC_{prt} \frac{B}{B} g_1$ will mean SC₁f B.

For example, $SC_{prt} \frac{DNA}{DNA} g_1$ allows you to reach the level of DNA self_g -

energy, $SC_{prt} \frac{Q}{Q} g_1$ allows you to reach the level of self_g -energy Q. The law of self_g-

energy conservation operates already at the level of self_g -energy. Also, in addition to capacities in themselves, you can consider the types of accommodation of oneself_g in oneself_g: the first type of the accommodation of oneself_g in oneself_g the second type of the accommodation of oneself_g in oneself: potentially, for example, in the form of programming oneself_g, the third type is partial accommodation of oneself_g in themselves_g—for example, self-operator, self-action, whirlwind. A container containing itself can be formed by self_g -accommodation, i.e., accommodation in oneself_g. Let us clarify the concept of the term capacity in itself_g: it is a capacity containing itself_g f potentially. Consider self_g -Q, where Q can be anything, including Q=self; in particular, it can be any action. Therefore, self_g -Q is when Q is made by itself_g; it makes itself_g. There is a partial self_g -Q for any Q with partial self_g -fulfillment. Let's consider several examples for capacities in themselves: ordinary lightning, electric arc discharge, and ball lightning.

A self_g -search of the solution of the equations $f_i(x)=0$, where $i=1,2,...,n$, $x=(x_1,x_2,...,x_n)$, will be realized in $SC_{prt} \frac{\{f_1(x)=0?x, f_2(x)=0?x, ..., f_n(x)=0?x\}}{g_1 a}$

or $SC_{prt} \frac{\{f_1(x)=0, f_2(x)=0, ..., f_n(x)=0\}}{g_1 ?x}$. x acquires more degree of liberty and

in this is direct decision. self_g -equation for x has its decision for x in direct kind.

The same for $\text{SCprt}_{\frac{\{tasks(x)\}}{g_1}} \cdot \text{self}_g$ -task for x has its decision for x in direct

kind. self_g -question has its answer for x in direct kind. $\text{SCprt}_{\frac{\{t\}}{(o,x)}}$, where $\{t\}$ -

time points set, (o, x) - object o in point x from space X, give to enter in necessary

time moments. The same for $\text{SCprt}_{\frac{\{t\}}{g_1}}$.

$\text{SCprt}_{\frac{\{God - father, God - son, Holy Spirit\}}{g_1}}$ is three-concept representation,

where α is a point in the connectedness space. SCprt is also great for working with

structures, for example: 1) $\text{SCprt}_{\frac{strA}{g_1}}$ - the structure A that fits into B with

accommodation type g_1 , where B can be any capacity, another structure etc. 2)

$\text{SCprt}_{\frac{str_Q}{g_1}}$ - embedding structure from Q into R with accommodation type g_1 .

Similarly for displacement: 1) $\text{SCprt}_{\frac{B}{g_2}}$ - displacement of structure A from B with

displacement type g_2 , 2) $\text{SCprt}_{\frac{B}{str_Q}}$ - displacement of the structure Q from B with

displacement type g_2 . To work with structures, you can introduce a special

operator CCprt: $\text{CCprt}_{\frac{A}{g_1}}$ structures B with the structure A with accommodation

type g_1 , $\text{CCprt}_{\frac{B}{g_2}}$ destructors B from the structure A with displacement type g_2 .

Definition 1.1.8. A structure with a second degree of freedom will be called complete, i.e., "capable" of reversing itself_g concerning any of its elements explicitly, but not necessarily in known operators; it can form (create) new special operators (in particular, special functions).

$$\begin{array}{cc} A & A \\ \text{In particular, CCprt}g_1, \text{CCpr}g_1 & \text{are such structures.} \\ A & A \end{array}$$

Similarly, for working with models, each is structured by its structure; for example, use SCprt-groups, SCprt-rings, SCprt-fields, SCprt-spaces, self_g-groups, self_g-rings, self_g-fields, and self_g-spaces. Like any task, this is also a structure of the appropriate capacity.

self_g-H (self_g-hydrogen), like other self_g-particles, does not exist in the ordinary, but all self_g-molecules, self_g-atoms, and self-particles are elements of the energy space.

Remark 1.1.3. The concept of elements of physics SCprt is introduced for energy space. The corresponding concept of elements of chemistry SCprt is introduced

accordingly. Examples: 1) SCprtE $\begin{array}{c} \{a_1q, a_2\} \\ g_1 \\ D \end{array}$ – the energy of instantaneous

substitution a_1 by a_2 , where a_1 , and a_2 are chemical elements, q is instant replacement. Similarly, one can consider for the node of chemical reactions

SCprt $\begin{array}{c} \{\text{chemical elements with "target weights"}\} \\ g_1 \\ \text{reaction} \end{array}$. The ideology of SCprt

elements allows us to go to the border of the world familiar to us, which allows us to act more effectively.

1.2 Dynamic SCprt – elements

We considered stationary SCprt – elements earlier. Here we consider dynamic SCprt – elements .

Definition 1.2.1. The process of fitting a set of elements $\{a(t)\} = (a_1(t), a_2(t), \dots, a_n(t))$ into one point x of space X at time t with accommodation type g_1 will be called a dynamic SCprt – element. We will denote $\text{SCprt}(t) \begin{array}{c} \{a(t)\} \\ g_1 \\ x \end{array}$.

Definition 1.2.2. Fitting an ordered set of elements into one point in space with accommodation type g_1 is called a dynamic ordered SCprt–element.

It is allowed to sum dynamic SCprt – elements:

$$SCprt(t) \underset{x}{g_1}^{\{a(t)\}} + SCprt(t) \underset{x}{g_1}^{\{b(t)\}} = SCprt(t) \underset{x}{g_1}^{\{a(t)\} \cup \{b(t)\}}.$$

Dynamic accommodation of oneself_g.

Definition 1.2.3. Dynamic SCprt-capacity $SCprt(t) \underset{Q(t)}{g_1}^{R(t)}$ is the process of embedding

$R(t)$ into $Q(t)$ with accommodation type g_1 .

Definition 1.2.4. The dynamic capacity $A(t)$ containing itself_g as an element of the first type is the process of containing $A(t)$ in $A(t)$ with accommodation type g_1 .

Denote $SC_1f(t)A(t)$.

Definition 1.2.5. Dynamic capacity $C(t)$ in itself_g of the second type is the process of containing elements from which it can be generated with accommodation type g_1 . Let's denote $SC_2f(t)C(t)$.

Definition 1.2.6. Dynamic partial capacity $B(t)$ in itself_g of the third type is a process of partial accommodation of $B(t)$ in itself_g with accommodation type g_1 or elements from which it can be generated with accommodation type g_1 partially or both at the same time. Denote $SC_3f(t)B(t)$.

All dynamic capacities in a dynamic self_g-space are, by definition, dynamic capacities in themselves. Dynamic capacity itself_g can manifest itself_g as dynamic SCprt-capacity and ordinary dynamic capacity. In these cases, the usual measures and methods of topology are used.

Connection of dynamic SCprt – elements with dynamic accommodation of oneself_g with accommodation type g .

Consider third type of dynamic partial accommodation of oneself_g with

accommodation type g . For example, based on $SCprt(t) \underset{x}{g_1}^{\{a(t)\}}$, where $\{a(t)\} =$

$(a_1(t), a_2(t), \dots, a_n(t))$, i.e. n – elements at one point x , we can consider the dynamic capacity in itself_g $SC_3f(t)$ with m elements from $\{a(t)\}$, $m < n$, which is

process formed according to the form (1.1), that is, only m elements from $\{a(t)\}$

are in the structure $SCprt(t) \begin{matrix} \{a(t)\} \\ g_1 \\ x \end{matrix}$.

Dynamic accommodation of oneself_g of the third type can be formed for any other structure, not necessarily $SCprt$, only through the obligatory reduction in the number of elements in the structure. In particular, using the form from the forms (1.1.1) – (1.4).

It is possible to introduce structures more complex than $SC_3f(t)$.

Dynamic math itself_g.

1. The process of simultaneous addition of a set of elements $\{a(t)\} =$

$(a_1(t), a_2(t), \dots, a_n(t))$ are realized by $SCprt(t) \begin{matrix} \{a(t) +\} \\ g_1 \\ x \end{matrix}$.

2. By analogy, for simultaneous multiplication: $SCprt(t) \begin{matrix} \{a(t) *\} \\ g_1 \\ x \end{matrix}$.

3. Similarly for simultaneous execution of various operations:

$SCprt(t) \begin{matrix} \{a(t)q(t)\} \\ g_1 \\ x \end{matrix}$, where $\{q(t)\} = (q_1(t), q_2(t), \dots, q_n(t))$. $q_i(t)$ -an

operation, $i = 1, \dots, n$.

4. Similarly, for the simultaneous execution of various operators:

$SCprt(t) \begin{matrix} \{F(t)a(t)\} \\ g_1 \\ x \end{matrix}$, where $\{F(t)\} = (F_1(t), F_2(t), \dots, F_n(t))$. $F_i(t)$ is an

operator, $i = 1, \dots, n$.

5. Dynamic arithmetic itself_g for accommodations of oneself_g will be similar: dynamic addition - $SC_1f(t)\{a(t) +\}$, (or $SC_3f(t)\{a(t) +\}$ for the third type), dynamic multiplication $SC_1f(t)\{a(t) *\}$, ($SC_3f(t)\{a(t) *\}$).

6. Similarly with different operations: $SC_1f(t)\{a(t)q(t)\}$,

$(SC_3f(t)\{a(t)q(t)\})$ and with different operators:

$SC_1f(t)\{F(t)a(t)\}$, $(SC_3f(t)\{F(t)a(t)\})$.

7. $\text{SCprt}(t) \begin{matrix} A(t) \\ g_1 \\ B(t) \end{matrix} = \text{gives the result } \text{SCprt}(t) \begin{matrix} A(t) \\ g_1 \\ B(t) \end{matrix} = \left\{ \begin{matrix} D(t) \\ A(t) \cup B(t) - A(t) \cap B(t) \end{matrix} \right\}$ for sets $A(t), B(t)$, where $D(t)$ is self_g-set for $A(t) \cap B(t)$.

The measure: $\mu(\text{Cpr}(t) \begin{matrix} A(t) \\ g_1 \\ B(t) \end{matrix}) = (\mu(A(t)) + \mu(B(t)) - \mu(A(t) \cap B(t))) * \mu(g_1)$.

The same is true for structures if they are treated as sets.

Our approach to the theory of hierarchical sets differs from the construction of hierarchical sets by Y.L. Ershov [4]-[6] : we construct completely different types of hierarchical sets.

8. Similarly, for dynamic SCprt-derivatives, dynamic SCprt-integrals, dynamic SCprt-lim, dynamic self_g-derivatives, dynamic self_g-integrals
9. Denote dynamic self_g-(dynamic self_g-Q(t)) through dynamic self_g²-Q(t) ,
fS_g(t)(n,Q(t))= dynamic self_g-(dynamic self_g-(...(dynamic self_g-Q(t)))) =
dynamic self_gⁿ-Q(t) for n-multiple dynamic self_g.

Remark 1.2.1. The dynamic SCprt-displacement of A(t) from B(t) with type of

accommodation $g_2(t)$ will be denote by $\begin{matrix} B(t) \\ g_2(t)\text{SCprt}(t) \\ A(t) \end{matrix}$. Then the notation

$\begin{matrix} C(t) & A(t) \\ g_2(t)\text{SCprt}(t)g_1(t) & \\ D(t) & B(t) \end{matrix}$ is dynamic SCprt-accommodation of A(t) in B(t) with type of

accommodation $g_1(t)$ and dynamic SCprt-displacement of D(t) from C(t) with type of accommodation $g_2(t)$ simultaneously.

We can consider the concept of dynamic SCprt - element as $\text{SCprt}(t) \begin{matrix} A(t) \\ g_1(t) \\ B(t) \end{matrix}$, where

A(t) fits in dynamic capacity B(t) with type of accommodation $g_1(t)$. Then

$\text{SCprt}(t) \begin{matrix} B(t) \\ g_1(t) \\ B(t) \end{matrix}$ will mean $\text{SC}_1\text{f}(t)B(t)\{g_1(t)\}$. $\begin{matrix} A(t) \\ g(t)\text{SCprt}(t) \\ A(t) \end{matrix}$ denotes the dynamic

expelling $A(t)$ oneself_{g(t)} out of oneself_{g(t)}, $\frac{A(t)}{g(t)}SCprt(t)\frac{A(t)}{g(t)}$ is simultaneous

dynamic accommodation $A(t)$ of oneself_{g(t)} in oneself_{g(t)} and dynamic expelling

$A(t)$ oneself_{g(t)} out of oneself_{g(t)}. $\frac{A(t)}{g(t)}SCprt(t)$ will be called dynamic anti-

capacity from oneself_{g(t)}. For example, “white hole” in physics is such simple anti-capacity. The concepts of “white hole” and “black hole” were formulated by the physicists based on the subject of physics –the usual energies level. Mathematics allows you to deeply find and formulate the concept of singular points in the Universe based on the levels of more subtle energies. The experiments of the 2022 Nobel laureates Asle Ahlen, John Clauser, Anton Zeilinger correspond to the concept of the Universe as a capacity in itself_g. The energy of self_g - accommodation in itself_g is closed on itself_g [5].

Hypothesis: the accommodation of the galaxy in oneself_g as a spiral curl and the expelling out of oneself_g defines its existence. A self_g -capacity in itself_g as an element A is the god of A, the self_g -capacity in itself_g as an element the globe—the god of the globe, the self_g -capacity in itself_g as an element man-- the god of the man, the self_g -capacity in itself_g as an element of the universe-- the god of the universe, the accommodation of A into oneself_g is spirit of A, the accommodation of the Earth into oneself_g is spirit of Earth, the accommodation of the man into oneself_g is spirit of the man (soul), the accommodation of the universe into oneself_g is spirit of the universe. We may consider the following axiom: any capacity is the capacity of oneself_g. This is for each energy capacity.

About dynamic SCprt and SC₃f(t) programming.

The ideology of dynamic SCprt and SC₃f(t) can be used for programming:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t), p_2(t), \dots, p_n(t))$ to the variables $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$ is

implemented through
$$\text{SCprt}(t) \quad \frac{\{g(t)\} := \{p(t)\}}{w(t) \quad x(t)} .$$

2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f(t)_2, \dots, f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), \dots, B_n(t))$ is implemented through

$$\text{SCprt}(t) \quad \frac{IF\{\{B(t)\}\{f(t)\}\} \text{ then } Q(t)}{w(t) \quad x(t)} .$$
 where Q(t) can be any.

3. Similarly for loop operators and others.

$\text{SC}_3f(t)$ – software operators will differ only in that the aggregates

$\{\{g(t)\}, \{p(t)\}, \{B(t)\}, \{f(t)\}\}$ will be formed from corresponding processes

$\text{SCprt}(t)$ for the above-mentioned programming operators through form (1.1) or the form from the forms (1.1.1) – (1.4) for more complex operators.

Remark 1.2.2. With the help of dynamic SCprt-elements, the concepts of dynamic SCprt - force, dynamic SCprt – energy are introduced. For example,

$$E(t)_{\text{sprt}}^f = \text{SCprt}(t) \quad \frac{\{E_1(t)f, E_2(t)\}}{g(t) \quad x(t)}$$
 will mean the process of instantaneous

replacement f of energy $E_1(t)$ by $E_2(t)$ at time t. Similarly, using $\text{SC}_i f(t)$, the concepts of $\text{SC}_i f(t)$ -force, $\text{SC}_i f(t)$ -energy, $i=1,2,3$, and etc are introduced.

Remark 1.2.3. It is the accommodation of oneself_g in oneself_g that can “give birth” to the capacities in itself_g – that is what self_g -organization is.

Remark 1.2.4.
$$\text{SCprt}(t) \quad \frac{B(t)}{\frac{\text{SCprt}(t)g(t)}{B(t)} \quad g(t)} \quad \frac{B(t)}{\text{SCprt}(t)g(t) \quad B(t)}$$
 can increase self_{g(t)} - level of B(t).

Remark 1.2.5. For example, the operator itself_g is SC₁f(t).

$$\text{Remark 1.2.6. May be considered the following derivatives: } \frac{A(t)}{dSCprt(t)g(t)} \frac{B(t)}{dt},$$

$$\frac{B(t)}{dg_2(t)SCprt(t)} \frac{C(t)}{dg_2(t)SCprt(t)g_1(t)} \frac{A(t)}{D(t)} \frac{B(t)}{dt}, \frac{dSCif(t)}{dt}, i=1,2,3.$$

Remark 1.2.7. It is the accommodation of oneself_g in itself_g as an element that can be interpreted as dynamic capacities in itself_g.

Remark 1.2.8. Not every capacity containing itself_g as an element will manifest itself_g as a sedentary capacity or capacity.

1.3 SCprt – elements for continual sets

Earlier, we considered finite-dimensional discrete SCprt-elements and self_g -capacities in itself_g as an element. Here we believe some continual SCprt-elements and continual self_g -capacities in themselves as an element.

Definition 1.3.1. The set of continual elements $\{a\} = (a_1, a_2, \dots, a_n)$ fitting into point x of space X with accommodation type g_1 will be called continual SCprt – element, and such a point in space will be called capacity of the continual SCprt –

element. We will denote $SCprt_{g_1}^{\{a\}}_x$

Definition 1.3.2. An ordered set of continual elements at one point in space is called an ordered continual SCprt–element.

It's allowed to sum continual SCprt – elements: $SCprt_{g_1}^{\{a\}}_x + SCprt_{g_1}^{\{b\}}_x =$

$SCprt_{g_1}^{\{a\} \cup \{b\}}_x$, where some or any elements may be ordered elements.

Definition 1.3.3. The continual self_g -capacity A in itself_g as an element of the first type is the capacity fitting with accommodation type g_1 itself_g as an element.

Denote SC_1fA .

Definition 1.3.4. The ordered continual self_g -capacity A in itself_g as an element of the first type is the ordered capacity fitting itself_g as an element with accommodation type g_1 . Denote $\overrightarrow{SC_1 f A}$.

For example, $SC_{\infty}^+ = \sin \infty |g$ is of this type. It denotes continual ordered self_g -capacities in itself_g as an element of following type—the range of simultaneous “activation” of numbers from $[-1, 1]$ in mutual directions: $\uparrow I \downarrow_{-1}^1$. Also consider the following elements: $SC_{\infty}^- = \sin(-\infty) |g \rightarrow \downarrow I \uparrow_{-1}^1 |g$, $TC_{\infty}^+ = \text{tg} \infty |g \rightarrow \uparrow I \downarrow_{-\infty}^{\infty} |g$, $TC_{\infty}^- = \text{tg}(-\infty) |g \rightarrow \downarrow I \uparrow_{-\infty}^{\infty} |g$, don't confuse with values of these functions. Such elements can be summarized. For example: $aSC_{\infty}^+ + bSC_{\infty}^- = (a-b)SC_{\infty}^+ = (b - a) SC_{\infty}^-$.

Definition 1.3.5. The continual self_g -capacity A in itself_g, as an element of the second type, is the capacity containing elements from which it can be generated. Let's denote $SC_2 f A$.

An example of continual self_g -capacity in itself_g as an element of the second type is a living organism since it contains the programs: DNA and RNA.

Definition 1.3.6. Partial continual self_g -capacity in itself_g as an element of the third type is called continual self_g -capacity in itself_g as an element that partially contains itself_g or contains elements from which it can be generated in part or both simultaneously. Denote $SC_3 f$.

All continual capacities in self_g -space are continual self_g -capacities in itself_g as an element by definition. The continual self_g -capacities in itself_g as an element may appear as continual SCpr- capacities and usual continual capacities. In these cases, there are used typical measure and topology methods.

The connection of continual SCprt – elements with continual self_g -capacities in themselves_g as an element.

Consider a third type of continual self_g -capacity in itself_g as an element. For

example, based on $SC_{prt}^x \{a\}_{g_1}$, where $\{a\} = (a_1, a_2, \dots, a_n)$, i.e. a_i - continual

elements at one point, $i=1, 2, \dots, n$. The continual self_g -capacity in itself_g as an

element with m continual elements from $\{a\}$, at $m < n$, can be considered as SC_3f , which is formed by the form (1.1), i.e., only m continual elements are located in the structures $SC_{prt} \begin{smallmatrix} \{a\} \\ g_1 \\ x \end{smallmatrix}$. Continual self_g-capacities in itself_g as an element of the third type can be formed for any other structure, not necessarily SC_{prt} , only by obligatory reducing the number of continual elements in the structure. In particular, using the form from the forms (1.1.1) – (1.4).

Structures more complex than SC_3f can be introduced.

Mathematics itself_g for continual elements.

1. Simultaneous addition of the continual elements of the set $\{a\} =$

$(a_1, a_2, \dots, a_n)$ is implemented using $SC_{prt} \begin{smallmatrix} \{a \cup\} \\ g_1 \\ x \end{smallmatrix}$.

2. By analogy, for simultaneous multiplication: $SC_{prt} \begin{smallmatrix} \{a \cap\} \\ g_1 \\ x \end{smallmatrix}$.

3. Similarly, for simultaneous execution of various operations: $SC_{prt} \begin{smallmatrix} \{aq\} \\ g_1 \\ x \end{smallmatrix}$, where

$\{q\} = (q_1, q_2, \dots, q_n)$. q_i -an operation, $i = 1, \dots, n$.

4. Similarly, for the simultaneous execution of various operators: $SC_{prt} \begin{smallmatrix} \{Fa\} \\ g_1 \\ x \end{smallmatrix}$,

where $\{F\} = (F_1, F_2, \dots, F_n)$. F_i is an operator, $i = 1, \dots, n$.

5. For continual self_g-capacities in themselves_g as an element will be similar: addition - $SC_1f\{a +\}$, (or $SC_3f\{a +\}$) for the third type), multiplication $SC_1f\{a *\}$, ($SC_3f\{a *\}$).

6. Similarly with different operations: $SC_1f\{aq\}$, ($SC_3f\{aq\}$), and with different operators: $SC_1f\{Fa\}$, ($SC_3f\{Fa\}$).

7. $SC_{rtg} \begin{smallmatrix} A \\ B \end{smallmatrix}$ is the result of the accommodation operator $SC_{prt} \begin{smallmatrix} A \\ g \\ B \end{smallmatrix}$. For sets A, B we have

$$SC_{rtg} \begin{smallmatrix} A \\ B \end{smallmatrix} = \left\{ A \cup B \begin{smallmatrix} D \\ A \cap B \end{smallmatrix} \right\}, \text{ where } D \text{ is self}_g\text{-set for } A \cap B. \text{ The measure:}$$

$$\mu(\text{SCprg}_B^A) = (\mu(A) + \mu(B) - \mu(A \cap B))^{\mu^s(A \cap B)} * \mu(g).$$

There is the same for structures if it's considered as continual sets. Our approach to the theory of hierarchical sets differs from the construction of hierarchical sets by Y.L. Ershov [4]-[6]: we construct completely different types of hierarchical sets.

Remark 1.3.1. Three in one is

$$\text{SCprt} \left\{ \begin{array}{c} \infty \text{ in itself, an element that is not anyone's element, } 0 \text{ out oneself} \\ g \\ \alpha \end{array} \right\}$$

, α - point space connectedness.

These elements are used for SCprt-coding, SCprt translation, coding self_g , and translation self_g for networks], which is suitable for electric current of ultrahigh frequency. More complex elements can be considered as continual sets of numbers with their " activation " in mutual directions. For example, ranges of function values, particularly those representing the shape of lightning. Differential geometry can be applied here. Also, n-dimensional elements can be considered. The space of such elements is Banach space if we introduce the usual norm for functions or vectors. We call this space-- SCelb-space. Then we introduce the scalar product for functions or vectors and get the Hilbert space. We call this space SCelh-space. In particular, one can try to describe some processes with these elements by differential equations and use methods from [7]. You can also try to optimize and research some processes with these elements using the techniques from [8]. Let's introduce operators for transforming capacity to self_g -capacity in self_g as an element: $Q_1\text{SC}(A)$ transforms A to SC_1fA , $Q_0\text{SC}(A)$ transforms A to $g\text{SCprt}_A$, $\text{SCO}(A)$ transforms A to $\uparrow A \downarrow$, $\uparrow A \downarrow$ -- ordered self_g -capacity in self_g as an element of simultaneous "activation" of all elements of A in mutual directions. For example, $\text{SCO}([-1,1]) = \text{SC}_\infty^+$, $\text{SCO}([1,-1]) = \text{SC}_\infty^-$, $\text{SCO}([-\infty,\infty]) = \text{CT}_\infty^+$, $\text{SCO}([\infty,-\infty]) = \text{CT}_\infty^-$. The operator $(Q_1\text{SC}(A))^2$ increases

self_g -level for A: it transforms self_g -A= SC₁fA to self_g²-A, (Q₁SC(A))ⁿ → self_gⁿ-A, e^{Q₁SC(A)} → e^{self_g} - A. Let us introduce the following notations:

$$\begin{matrix} \{\} \\ g \text{ SCprt by os}(\{\} \rightarrow) \text{elf}_g, \\ A \end{matrix}$$

$$\begin{matrix} A \\ g \text{ SCprt by 2oself}_g -A, \text{ SCprt} \end{matrix} \begin{matrix} (A,A) \\ g \\ A \end{matrix} \text{ by 2self}_g -A, \text{ SCprt} \begin{matrix} A \\ g \\ (A,A) \end{matrix} \text{ by 1/2self}_g -A,$$

$$\begin{matrix} A \\ \text{SCprt } g \\ q(A) \end{matrix} \text{ by qself}_g -A, \begin{matrix} A \\ g \text{ SCprt by q}() \text{oself}_g -A, \text{ q-any operator,} \\ q(A) \end{matrix}$$

$$\begin{matrix} A \\ g \end{matrix} \text{ SCprt by N oself}_g -A, q_i = A, i = 1, \dots, N; \begin{matrix} A \\ g \text{ SCprt } g \\ A \end{matrix} \text{ by (self}_g -$$

$$\text{oself}_g -A, \begin{matrix} q_2(A) \\ g \text{ SCprt } g \\ q_3(A) \end{matrix} \begin{matrix} A \\ q_1(A) \end{matrix} \text{ by (q}_1\text{self}_g - \begin{pmatrix} q_3 \circ \\ q_2 \circ \end{pmatrix} \text{oself}_g -A, \begin{matrix} A \\ g \text{ CCprt by by} \\ (A,A) \end{matrix}$$

$$2\text{Coself}_g -A, \text{ CCprt} \begin{matrix} (A,A) \\ g \\ A \end{matrix} \text{ by 2Cself}_g -A, \text{ CCprt} \begin{matrix} A \\ g \\ (A,A) \end{matrix} \text{ by 1/2Cself}_g -A, \text{ CCprt} \begin{matrix} A \\ g \\ q(A) \end{matrix} \text{ by}$$

$$\text{qCself}_g -A, \begin{matrix} A \\ g \text{ CCprt by q}() \text{Coself}_g -A, \text{ q-any operator,} \\ q(A) \end{matrix} \begin{matrix} A \\ g \\ (q_1(A), \dots, q_N(A)) \end{matrix} \text{ CCprt}$$

$$\text{by N Coself}_g -A, q_i = A, i = 1, \dots, N; \begin{matrix} A \\ g \text{ CCprt } g \\ A \end{matrix} \text{ by Cself}_g -A - \text{Coself}_g -A,$$

$$\begin{matrix} q_2(A) \\ g \text{ CCprt } g \\ q_3(A) \end{matrix} \begin{matrix} A \\ q_1(A) \end{matrix} \text{ by q}_1\text{Cself}_g -A - \begin{pmatrix} q_3 \circ \\ q_2 \circ \end{pmatrix} \text{Coself}_g -A, \text{ SC2prt } g = \begin{matrix} A \\ (self_g -A, self_g -A), \\ A \end{matrix}$$

$$\begin{matrix} A \\ \text{SCNprt } g = (q_1, \dots, q_N), q_i = self_g -A, i = 1, \dots, N. self_g (SCprt } g) = \text{SCprt} \end{matrix} \begin{matrix} A \\ g \\ B \end{matrix} \begin{matrix} A \\ \text{SCprt } g \\ B \end{matrix}. \text{ Can be}$$

$$\text{considered } Q \begin{matrix} A \\ g \text{ SCprt } g \\ A \end{matrix} \begin{matrix} A \\ A \end{matrix}, \text{ Q-any operator.}$$

1.4 Dynamic continual SCprt – elements

Definition 1.4.1. The process of containing a set with continual elements $\{a(t)\} = (a_1(t), a_2(t), \dots, a_n(t))$ at one point x of the space X at time with accommodation

type g_1 will be called the dynamic continual SCprt – element. We will denote

$$\text{SCprt}(t) \underset{x}{g_1} \{a(t)\} \dots$$

Definition 1.4.2. The process of containing an ordered set of continual elements at one point in space with accommodation type g_1 is called dynamic continual ordered SCprt – element.

It is allowed to sum dynamic continual SCprt – elements:

$$\text{SCprt}(t) \underset{x}{g_1} \{a(t)\} + \text{SCprt}(t) \underset{x}{g_1} \{b(t)\} = \text{SCprt}(t) \underset{x}{g_1} \{a(t)\} \cup \{b(t)\}.$$

.

Dynamic continual containing of oneself_g in oneself_g as an element.

Definition 1.4.3. The dynamic continual SCprt-capacity $\text{SCprt}(t) \underset{Q(t)}{g_1} \overset{R(t)}{}$ is called the

process of embedding $R(t)$ in $Q(t)$ with accommodation type g_1 .

Definition 1.4.4. The dynamic accommodation continual $A(t)$ of oneself_g of the first type is the process of putting $A(t)$ into itself_g. Denote $SC_1f(t)A(t)$.

Definition 1.4.5. The dynamic accommodation continual $C(t)$ of oneself_g of the second type embedding contains the continual elements with accommodation type g_1 from which it can be generated. Denote $SC_2f(t)C(t)$.

Definition 1.4.6. The partial dynamic accommodation continual $B(t)$ of oneself_g of the third type is the process of partial embedding continual $B(t)$ into oneself_g or continual elements from which it can be generated in part with accommodation type g_1 or both simultaneously. Denote $SC_3f(t)B(t)$.

The connection of dynamic continual SCprt – elements with dynamic accommodation of oneself_g in oneself_g as an element .

Let us consider the partial dynamic continual accommodation of oneself_g in

oneself_g as an element of the third type. For example, based on $\text{SCprt}(t) \underset{x}{g_1} \{a(t)\}$, where

$\{a(t)\} = (a_1(t), a_2(t), \dots, a_n(t))$, i.e. n - continual elements at one point x , one

can consider the dynamic accommodation $SC_3f(t)$ of oneself_g in oneself_g as an element with m continual elements from $\{a(t)\}$, $m < n$, which is a process that is necessary form according to the form (1.1), i.e., only m continual elements from

$\{a(t)\}$ are located in the structure $SC_{prt}(t) \underset{x}{g_1}^{\{a(t)\}}$. Dynamic continual

accommodations of oneself_g in oneself_g as an element of the third type can be formed for any other structure, not necessarily SC_{prt} , only by necessarily reducing the number of continual elements in the structure. In particular, with the help of the form from the forms (1.1.1) – (1.4).

. It is possible to introduce structures more complex than $SC_3f(t)$.

Dynamic continual mathematics self_g.

1. The process of simultaneous addition of the set of continual elements

$\{a(t)\} = (a_1(t), a_2(t), \dots, a_n(t))$ is realized by $SC_{prt}(t) \underset{x}{g_1}^{\{a(t) \cup\}}$.

2. By analogy, for simultaneous multiplication: $SC_{prt}(t) \underset{x}{g_1}^{\{a(t) \cap\}}$.

3. Similarly for simultaneous execution of various operations:

$SC_{prt}(t) \underset{x}{g_1}^{\{a(t)q(t)\}}$, where $\{q(t)\} = (q_1(t), q_2(t), \dots, q_n(t))$. $q_i(t)$ -an

operation, $i = 1, \dots, n$.

4. Similarly, for the simultaneous execution of various operators:

$SC_{prt}(t) \underset{x}{g_1}^{\{F(t)a(t)\}}$, where $\{F(t)\} = (F_1(t), F_2(t), \dots, F_n(t))$. $F_i(t)$ is an

operator, $i = 1, \dots, n$.

5. The dynamic arithmetic self_g for the dynamic continual accommodations of oneself_g will be similar: dynamic addition - $SC_1f(t)\{a(t) \cup\}$,
(or $SC_3f(t)\{a(t) \cup\}$ for the third type), dynamic multiplication
 $SC_1f(t)\{a(t) \cap\}$, $(SC_3f(t)\{a(t) \cap\})$.

6. Similarly with different operations: $SC_1f(t)\{a(t)q(t)\}$,
 $(SC_3f(t)\{a(t)q(t)\})$, and with different operators:
 $SC_1f(t)\{F(t)a(t)\}$, $(SC_3f(t)\{F(t)a(t)\})$.

7. $SCprt(t) \begin{matrix} A(t) \\ g_1 \\ B(t) \end{matrix}$ gives the result $SCrt(t) \begin{matrix} A(t) \\ g_1 \\ B(t) \end{matrix} = \left\{ \begin{matrix} D(t) \\ A(t) \cup B(t) - A(t) \cap B(t) \end{matrix} \right\}$
for continual sets $A(t), B(t)$, where $D(t)$ is self_g-set for $A(t) \cap B(t)$.

The measure: $\mu(\begin{matrix} A(t) \\ SCrt(t) \ g_1 \\ B(t) \end{matrix}) = (\begin{matrix} \mu^s(A(t) \cap B(t)) \\ \mu(A(t)) + \mu(B(t)) - \mu(A(t) \cap B(t)) \end{matrix})^* \mu(g_1)$.

There is the same for structures if it's considered as continual sets. Our approach to the theory of hierarchical sets differs from the construction of hierarchical sets by Y.L. Ershov [4]-[6] : we construct completely different types of hierarchical sets.

8. Similarly, for dynamic SCprt-derivatives, dynamic SCprt-integrals, dynamic SCprt-lim, dynamic self_g-derivatives, dynamic self_g-integrals
9. Denote dynamic continual self_g-(dynamic continual self_g-Q(t)) through dynamic continual self_g²-Q(t), $fSC(t)(n, Q(t)) = \text{dynamic continual self}_g - (\text{dynamic continual self}_g - (\dots (\text{dynamic continual self}_g - Q(t)))) = \text{dynamic continual self}_g^n - Q(t)$ for n-multiple dynamic continual self_g.

Remark 1.4. Dynamic continual SCprt-displacement of A(t) from B(t) with type of

accommodation $g_2(t)$ will be denote through $\begin{matrix} B(t) \\ g_2(t)SCprt(t) \\ A(t) \end{matrix}$. Then the notation

$\begin{matrix} C(t) & A(t) \\ g_2(t)SCprt(t)g_1(t) & \\ D(t) & B(t) \end{matrix}$ is dynamic continual SCprt- embedding of A(t) in B(t) with

type of accommodation $g_1(t)$ and dynamic continual SCprt-displacement of D(t) from C(t) with type of accommodation $g_2(t)$ simultaneously..

We can consider the concept of dynamic continual SCprt - element as $SCprt(t) \frac{A(t)}{B(t)}$,

where $A(t)$ fits in dynamic continual capacity $B(t)$. Then $SCprt(t) \frac{B(t)}{g_1(t)}$ it will mean

$SC_1 f(t) \frac{A(t)}{B(t)}$. $\frac{g_2(t)}{A(t)} SCprt(t)$ denotes the dynamic continual displacement of $A(t)$ from

itself_g, $\frac{A(t)}{A(t)} SCprt(t) \frac{A(t)}{g_1(t)}$ — simultaneous dynamic continual accommodation of

oneself_g $A(t)$ in oneself_g $A(t)$ and dynamic continual expelling oneself_g $A(t)$ out of

oneself_g $A(t)$. $\frac{A(t)}{A(t)} SCprt(t)$ will be called dynamic continual anti capacity from

itself_g.

Connection of dynamic continual SCprt – elements with target weights with dynamic continual accommodation of oneself_g with target weights.

Consider a third type of dynamic partial accommodation of oneself_g with target

weights $g(t)$. For example, based on $SCprt(t) \frac{\{a(t)g(t)\}}{g_1}$, where $\{a(t)\} =$

$(a_1(t), a_2(t), \dots, a_n(t))$, i.e. n - continual elements with target weights $\{g(t)\}$ at

one point x , we can consider the dynamic accommodation $SC_3 f(t)g(t)$ of

oneself_g with target weights with m continual elements with target weights $\{g(t)\}$

from $\{a(t)\}$, $m < n$, which is the process of formation according to the form (1.1),

i.e., only m continual elements with target weights $\{g(t)\}$ from $\{a(t)\}$ are located

in the structure $SC_3 f(t)g(t)$. Dynamic accommodations of oneself_g with target

weights of the third type can be formed for any other structure, not necessarily Sit,

only by reducing the number of continual elements with target weights in the

structure. In particular, using the form from forms (1.1.1) – (1.4). Structures more

complex than $SC_3 f(t)g(t)$ can be introduced.

Definition 1.4.8. The dynamic embedding of continual $A(t)$ into itself_g with target weights $\{g(t)\}$ of the first type is the process of embedding $A(t)$ into $A(t)$ with target weights. Denote $SC_1f(t)A(t)g(t)$.

Definition 30. The dynamic accommodation of continual $C(t)$ itself_g into itself_g with target weights $\{g(t)\}$ of the second type is the process of accommodation of the continual elements from which it can be generated. Let's denote $SC_2f(t)C(t)g(t)$.

Definition 1.4.9. Partial dynamic accommodation of continual $B(t)$ itself_g into itself_g with target weights $\{g(t)\}$ of the third type is the process of partial accommodation of continual $B(t)$ into itself_g or continual elements from which it can be generated partially, or both at the same time. Denote $SC_3f(t)B(t)g(t)$.

1.5 The usage of SCprt-elements for networks

A. Galushkin's comprehensive monograph [9] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new mathematical process with accommodation operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators. Accommodation operators are more convenient for networks. Also, the main emphasis was placed on using processors operating using triodes, which are generally not used in SCprt-networks. SCprt-networks (SmnSCprt) are a SCprt-structure that can be built for the required weights. SCprt-OS (SCprt operating system) uses SCprt-coding and SCprt-translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b) , where the number b is the code of the action, and the number a is the code of the object of this action. SCprt-coding (or self_g-coding) is implemented through a matrix consisting of 2 columns (in the continuous case, two intervals of numbers). Here, the source encoding is used for all matrix rows simultaneously. SCprt-translation is carried out by inversion. In this case, self_g-coding and self_g-translation will be

more stable. The target weights f_i in $SCprt \frac{\{f_i x\}}{g_1}$ are chosen for necessary tasks. We

will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [9]. We will touch on the difference of this only for hierarchical complex networks. The same simple executing programs are in the cores of simple artificial neurons of type SCprt (designation - mnSCprt) for simple information processing. More complex executing programs are used for

mnSCprt nodes. SCprt-threshold element – $sgn(SCprt \frac{\{ax\}}{g_1})$, b- mnSCprt,

$x=(x_1, x_2, \dots, x_n)$ – source signals values, $a=(a_1, a_2, \dots, a_n)$ – SCprt-synapses weights.

The first level of mnSCprt consists of simple mnSCprt. The second level of

mnSCprt consists of $SCprt \frac{\{mnSCprt\}}{g_1}$ – SCprt-node of mnSCprt in range D, D-

capacity for mnSCprt node. The third level of mnSCprt consists of

$SCprt \frac{\{mnSCprt\}}{g_1}$ – SCprt²- node of mnSCprt in range D, thus D becomes

capacity of itself_g in itself_g as an element for mnSCprt. For our networks, it is sufficient to use SCprt²- nodes of mnSCprt, but self_g-level is higher in living organisms, particularly SCprtⁿ-, $n \geq 3$. The target structure or the corresponding program enters the target unit using a short-pulse laser to generate attosecond pulses of light. After that, all networks or parts of them are activated according to the indicative goal. It may appear that we are leaving the network ideology, but these networks are a complex hierarchy of different levels, like living organisms.

Remark 1.5. Traditional scientific approaches through classical mathematics make it possible to describe only at the usual energy level. Here we consider an approach that makes describing processes with finer energies possible. mnSCprt contains

$SCprt \frac{\{ceprograms_g\}}{g}$, cprogram_g –executing program in SCprt- OS. SCprt-OS mnSCprt

(or self_g -OS) is based on SCprt-assembly language (or self_g -assembly language),

which is based on assembly language through SCprt-approach in turn, if the base of elements of SCprt-networks is sufficient. The ceprograms_g are in SCprt-programming environments (or self_g -programming environments), but this question and SCprt-networks base will be considered in the following publications. In particular, ceprograms_g may contain SCprt- programming operators. In mnSCprt cores, the constant memory SCprt with correspondent ceprograms_g depending on mnSCprt.

The OS (operating system) and the principles and modes of operation of the SCprt-networks for this programming are interesting. But this is already the material for the next publications.

Here is developed a helicopter model without a main and tail rotors based on SCprt – physics and special neural networks with artificial neurons operating in normal and SCprt-modes. Let's denote this model through SmnSCprt. To do this, it's proposed to use mnSCprt of different levels; for example, for the usual mode, mnSCprt serves for the initial processing of signals and the transfer of information to the second level, etc., to the nodal center, then checked. In case of an anomaly - local SCprt–mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, SmnSCprt is activated with the desired "target weight." Here are realized other tasks also. To reach the self_g -

energy level, the mode $\text{SCprt}_{g_1}^{\text{SmnSCprt}}$ is used. In normal mode, it's planned to

carry out the movement of SmnSCprt on jet propulsion by converting the energy of the emitted gases into a vortex to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the SmnSCprt for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is drainage of exhaust gases outside the SmnSCprt. SmnSCprt is represented by a neural network that extends from the center of one of the main clusters of SCprt - artificial neurons to the shell, turning into the body itself_g. Above the operator's cabin is the central core of the neural

network and the target block, responsible for performing the "target weights" and auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for SmnSCprt's actions is below the operator's cab. In SCprt – mode, the entire network or its sections are SCprt – activated to perform specific tasks, in particular, with "target weights." In the target, block used SCprt -coding, SCprt-translation for activation of all networks to "target weights" simultaneously, then –the reset of this SCprt-coding after activation. Unfortunately, triodes are not suitable for SCprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for ceprograms_g may be used instead triodes since there is no necessity to unbend the alternating current to direct. The SCprt-operative memory belt is disposed around a central core of SmnSCprt. There are SCprt-coding, SCprt-translation, and SCprt-realize of eprograms_g and the programs from the archives without extraction, SCprt-coding and SCprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. SCprt – structure or an ceprograms_g if one is present of needed «target weight» are taken in target block at SCprt – activation of the networks. SCprt $\frac{\text{SmnSCprt}, f}{g}$ *activation* derives SmnSCprt to the self_g -level boundary with target weight f. It's used ultra-short optical pulses laser or an alternating current of above high frequency and ultra-violet light, which can work with SCprt – structures in SCprt-modes by its nature to activate the networks or some of its parts in SCprt-modes and locally using SCprt-mode. Above high frequently alternating current go through mercury bearers. That's why overheating does not occur. The power of the alternating current above high frequently increases considerably for the target block. The activation of all networks is realized to indicate "target weights."

1.6 Variable hierarchical dynamical structures (models) for dynamic, singular, hierarchical sets

In contrast to the classical one-attribute set theory, where only its contents are taken as a set, we consider a two-attribute set theory with a set as a capacity and separately with its contents. We simply use a convenient form to represent the singularity of a set. Articles [10-19] use the following methodology for permanent structures:

1. Cancellation of the axiom of regularity
2. 2 attributes for the set: capacity and its content
- . Compression of a set, for example, to a point
4. “turning out” from one another, particularly from a capacity, we pull out another capacity, for example, itself_g, as its element.
5. The simultaneity of one (compression) and the other (“eversion”)
6. Own capacities
7. Qualitatively new programming and Networks.

Here we will consider variable structures (models), both discrete and continuous:

a) with variable connections, b) with the variable backbone for links, c) generalized version; in particular, in variable structures (models), for example,

$$\begin{array}{c} C \\ g_2 SCprt(t)g_1 \\ D \end{array} \begin{array}{c} A \\ B \end{array} = \left\{ \begin{array}{l} C \\ g_2 SCprt, \quad q_2 \geq t \geq q_1 \\ D \\ C \quad A \\ g_2 SC^1 prt g_1, q_3 \geq t > q_2 \\ D \quad B \\ B \quad A \\ g_2 SCprt g_1, q_4 \geq t > q_3 \quad (*_{6.1}), \\ D \quad B \\ A \\ SCprt g_1, q_5 \geq t > q_4 \\ B \\ \{\} \\ g_2 SCprt, \quad t > q_5 \\ D \\ \dots \end{array} \right.$$

$\begin{matrix} C & A \\ g_2 SCprt & g_1 \\ D & B \end{matrix}$ is the analogue, considered $\begin{matrix} C & A \\ D & B \end{matrix} S^1 prt_B^A$ in [10]. In particular, $\begin{matrix} B & A \\ g_2 SCprt & g_1 \\ D & B \end{matrix}$ can

be interpreted as a game: player 1 fits A into B, and the other pushes D out of B at the same time.

Can be considered N-hierarchical structure: 1-level - elements; level 2 - connections between them, level 3 - relationships between elements of level 2, etc. up to level N+1. N-hierarchical structure: 1-level - A; 2-level -B, 3-level - C, etc. up to (N+!)- level, where A, B, C, ... can be any in particular, by actions, sets, and others.

$$\begin{matrix} C & A \\ g_2 SCprt & g_1 \\ D & B \end{matrix} : \left\langle \begin{matrix} A \rightarrow B & D \leftarrow C \\ A, B & C, D \end{matrix} \right\rangle \rightarrow \begin{pmatrix} self_{g_1}(A \rightarrow B) \\ A, B \end{pmatrix}$$

$$\begin{matrix} C & A \\ g_2 SCprt & g_1 \\ D & B \end{matrix} : \left\langle \begin{matrix} A \rightarrow B & D \leftarrow C \\ A, B & C, D \end{matrix} \right\rangle \rightarrow \begin{pmatrix} oself_{g_2}(D \leftarrow C) \\ C, D \end{pmatrix}$$

Can be considered discrete hierarchical structure, continuous hierarchical structure, and discrete-continuous hierarchical structure,

N – hierarchical structure
 $SCprt \quad \begin{matrix} g \\ x \end{matrix} .$

The example

$$\begin{matrix} & & N\text{--level of hierarchical structure} \\ SCprt & & \begin{matrix} g \\ x \end{matrix} \\ & & \dots \\ \left[\begin{matrix} & & i\text{--level of hierarchical structure} \\ SCprt & & \begin{matrix} g \\ x \end{matrix} \\ & & \dots \\ & & 1\text{--level of hierarchical structure} \\ SCprt & & \begin{matrix} g \\ x \end{matrix} \end{matrix} \right] \end{matrix}$$

$QHSprg = HSprt_x$ -- N-hierarchical structure
compression with type of accommodation g into point x .

Let $f(N, QHSprg) = QHSprg \left. \begin{matrix} QHSprg \\ QHSprg \\ \dots \\ QHSprg \end{matrix} \right\} \text{--N levels}$

It can be considered $self_g - QHSprg$, $f(y, QHSprg)$ for any y , $f(QHSprg, QHSprg)$.

Compression Hierarchy Examples:

The example of variable hierarchy

47

where Q is self_g -set for $(D \cap C)$ [10], R is self_g -set for $A \cap B$, $S_{01}^{et}fB$,

${}_{C-B}^{C-B}S_1 t_B^{A-B}$, ${}_{D-C-B}^{C-B}S_1 t_B^{A-B}$ are considered in [11], ${}_{Q-B}^B S^1 t_B^{A-B}$ is considered in [12].

1.7 Applications Dynamic Sets Theory to physics and chemistry

Objects of physics and chemistry have an energy structure, which can be tried to be represented in the form of a hierarchical energy structure: the upper level of subtle self_g -energy and the lower level, which is manifested in the form of objectivity.

Ordinary types of energy are manifestations of a lower level from these structures.

If we represent an amorphous body with a mathematical structure of self_g -object

$\text{SCprt} \begin{matrix} A_0 + E_s \\ g_1 \\ A_0 + E_s \end{matrix}$, where $\text{SCprt} \begin{matrix} A_0 \\ g_1 \\ A_0 \end{matrix}$ - level of objectivity of an amorphous object,

$(\text{SCprt} \begin{matrix} A_0 \\ g_1 \\ A_0 + E_s \end{matrix} + \text{SCprt} \begin{matrix} A_0 + E_s \\ g_1 \\ A_0 \end{matrix})$ - the energy of connections between the level of

subtle energy $\text{SCprt} \begin{matrix} E_s \\ g_1 \\ E_s \end{matrix}$ and the level of objectivity .

Thus, one can try to conventionally represent the mathematical model of the energy structure of an amorphous object as a hierarchical dynamic operator

$$\begin{array}{c} E_s \\ \text{SCprt} g_1 \\ E_s \\ (\text{SCprt} \begin{matrix} A_0 \\ g_1 \\ A_0 + E_s \end{matrix} + \text{SCprt} \begin{matrix} A_0 + E_s \\ g_1 \\ A_0 \end{matrix}) (1.7.1). \\ A_0 \\ \text{SCprt} g_1 \\ A_0 \end{array}$$

In particular, the magnetic field and spin belong to the second level in (1.7.1) .

The next level of objectivity responds to a crystal. We represent a crystal with a mathematical structure

$$\begin{array}{c} A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \end{array} \quad (1.7.2).$$

Thus, one can try to conventionally represent the mathematical model of the energy structure of a crystal as a hierarchical dynamic operator (1.7.2). The next level of objectivity responds to a living crystal, for example, the bone of a living organism, a nail, viruses, DNA, RNA and etc. When there is no nutrient medium

and energy, it behaves like a crystal:

$$\begin{array}{c} \{\} \\ \text{SCprt} \{\} \\ \{\} \\ \{\} \\ \text{SCprt} \{\} \\ \{\} \end{array} \begin{array}{c} A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \end{array}, \text{ a nutrient}$$

medium appears and the necessary energy:

$$\begin{array}{c} B_0 + E_q \\ \text{SCprt } g_1 \\ B_0 + E_q \\ g_1 \\ B_0 + E_q \\ \text{SCprt } g_1 \\ B_0 + E_q \end{array} \begin{array}{c} B_0 + E_q \\ \text{SCprt } g_1 \\ B_0 + E_q \\ \text{SCprt } g_1 \\ B_0 + E_q \\ \text{SCprt } g_1 \\ B_0 + E_q \end{array}, \text{ its}$$

structure is transformed into a mathematical structure

$$\begin{array}{c} B_0 + E_q \\ \text{SCprt } g_1 \\ B_0 + E_q \\ g_1 \\ B_0 + E_q \\ \text{SCprt } g_1 \\ B_0 + E_q \end{array} \begin{array}{c} A_0 + E_s + B_0 + E_q \\ \text{SCprt } g_1 \\ A_0 + E_s + B_0 + E_q \\ g_1 \\ A_0 + E_s + B_0 + E_q \\ \text{SCprt } g_1 \\ A_0 + E_s + B_0 + E_q \end{array}. \text{ The division of DNA into two DNAs}$$

after sufficient accumulation of bases and energy - this minimal division into only two duplicates corresponds to the law of conservation of living energy and minimization of the entropy of the system.

Next comes the level of living organisms:

$$\begin{array}{cccc}
 \begin{array}{c} A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \\ g_1 \\ A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \end{array} & \begin{array}{c} A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \\ g_1 \\ A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \end{array} & \begin{array}{c} A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \\ g_1 \\ A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \end{array} & \begin{array}{c} A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \\ g_1 \\ A_0 + E_s \\ \text{SCprt } g_1 \\ A_0 + E_s \end{array} \\
 & g_1 & \text{SC}_1\text{prt} & g_1
 \end{array} .$$

Next comes the level of Globe, where the role of living cells (molecules in the case of a crystal) is played by living organisms. Next comes the level of Universe, where the role of living cells (molecules in the case of a crystal) is played by planets inhabited by living beings. You can try to represent these levels through more complex mathematical models, there are options for going beyond the level of objectivity for objects with energy structures of a sufficiently high level, but this is already material for subsequent publications. Our object world is an interpretation of the manifestation of only one set of subtle energy fibers out of their countless number.

$\frac{C}{g_2 \text{SCprt } C}$ will be called dynamic anti-capacity from oneself_g. For example, “white hole” in physics is such simple anti-capacity. The concepts of “white hole” and “black hole” were formulated by the physicists based on the subject of physics –the usual energies level. Mathematics allows you to deeply find and formulate the concept of singular points in the Universe based on the levels of more subtle energies. The experiments of the 2022 Nobel laureates Asle Ahlen, John Clauser, Anton Zeilinger and the experiments in chemistry Nazhipa Valitov correspond to the concept of the Universe as a capacity in itself_g as the element. They experimented with connections for elements of the microworld, and since here the

connections are self_g -connections, then when the object component of self_g -connections is removed, its higher level remains, which was manifested in their experiments. The electron spin belongs to the second level - above the level of objectivity. The energy of self_g -accommodation in itself_g is closed on itself_g .

Remark 1.7.1. From the point of view of our theory of dynamic operators and sets, we can interpret the energy effect of a thermonuclear reaction as the result of the “collapse” of two self_g -objects: for example, 1) ${}^3_2\text{He}$, ${}^3_2\text{He}$ and the formation of one self_g -object ${}^4_2\text{He}$, 2) ${}^3_2\text{He}$, ${}^2_1\text{H}$ and the formation of one self_g -object ${}^4_2\text{He}$. As a result, the energy of the collapse of the lost part of the self_g is released.

Remark 1.7.2. To gain access to object transformation, just go to the level

$$lS = \frac{2}{\pi} \arctg(1 + \varepsilon), \varepsilon \text{ may be quite small.}$$

Examples of transformation:

$$\begin{array}{l} 1) \quad \begin{array}{ccc} & q & b \\ & \text{SCprt}g_1 & \text{SCprt}g_1 \\ q & \downarrow & \downarrow \\ \text{SCprt}g_1 & \rightarrow \text{SCprt}g_1 & \rightarrow \text{SCprt}g_1 \\ q & & c \\ & q & b \\ & \text{SCprt}g_1 & \text{SCprt}g_1 \\ & q & c \end{array} \\ 2) \quad \begin{array}{ccc} q & & r \\ \text{SCprt}g & \rightarrow \text{SC}_3f(\text{self}_g(q)) & \rightarrow \text{SCprt}g \\ q & & r \end{array} \end{array}$$

This is a rather conditional interpretation, because in fact, the lS of the “vessel” (energy cocoon) of the object may turn out to be greater than $\frac{2}{\pi} \arctg(1)$. This is taken for initiation: we build a theory of this, starting from this stage of interpretation. After experiments, the next stage may begin.

$\text{self}_g A = \text{SCprt} \begin{array}{c} A \\ g \\ A \end{array}$ can be transformed into any D if $\mu l_g(D) = \mu l_g(\text{SCprt} \begin{array}{c} A \\ g \\ A \end{array})$, $\mu l_g(x)$ -

level measure of self_g for x , in particular, into $\text{SCprt} \begin{array}{c} \text{any } C \\ g \\ A \end{array}$ or $\text{SCprt} \begin{array}{c} A \\ g \\ \text{any } C \end{array}$, and also

an object R into any object Q or any energy U . The transformations of this type will be called transformations. $\text{self}_g^N A$ can transform itself_g into any D if $N \geq 2$; to realize this we need an even larger quantity N .

Example of a parallel-serial program statement

$$\begin{array}{c}
C \\
g \\
S := \text{SCprt} \quad Q \\
\text{if } \{p\} \text{SCprt} \quad \text{SCprt } g \\
A \\
g \\
A \\
\text{SCprt} \quad g \\
af := \\
\text{SCprt} \quad g \\
E \\
\text{for } w \text{ SCprt} \quad g \\
\text{if } C \\
\text{SCprt} \quad g \\
J
\end{array}$$

Each self_g -field can automatically rebuild the self_g -program to the desired.

self_g^N - OS and is designed for such transformations, and it itself_g can be transformed at $N \geq 1$, or it itself_g can be transformed at $N \geq 2$.

Remark 1.7.3. Hypothesis 1.7: equations for real processes in a non-trivial form can be used to fully or partially interpret the self_g -level of the process, replacing the equal signs with identification signs, and solutions to these equations as a manifestation of this level on the level of objectivity and ordinary energies. That is, equations for real processes serve as a definition of the self_g -level of the process, the definition of self_g -values (self_g -characteristics) of the process through the identification sign, i.e., they are defined (expressed) through themselves. In particular, forms (1.1) - (1.4) can be used as forms of identification. Each such singularity creates its own field, the process, the object etc. Much more effective than science for working with these singularities will be special Dynamic programming, which we are currently working on to create. Identification at the lower levels of a hierarchical dynamic structure of type (1.7.1) will lead to the upper level. You can also try to use it for full or partial interpretation of the self_g -level of chemical reactions, but here there will be a trivial identification and determination of the self_g -level will be much simpler. For example, a type $w \equiv 2w$ singularity at the top level of the structure of a mathematical simplified model of DNA generates a field for DNA division. A rather complex type of singularity at

the upper level of the structure of a simplified mathematical model generates an electromagnetic field through identification in Maxwell's equations.

Remark 1.7.4. Parallel operator $SC_{prt} \frac{symbols}{g}$ corresponds to $places\ for\ symbols$

theoretical science, parallel operator $SC_{prt} \frac{objects}{g}$ corresponds to technology, $x -$ the space "point" (space place).

Remark 1.7.5. Let $self_g$ -energy of A looks like $SC_{prt} \frac{C_A + \Delta C}{C_A + \Delta C} = SC_{prt} \frac{C_A}{C_A} + SC_{prt} \frac{\Delta C}{C_A}$
 $+ SC_{prt} \frac{C_A}{\Delta C} + SC_{prt} \frac{\Delta C}{\Delta C}, SC_{prt} \frac{C_A}{C_A}$ corresponds to objectivity of A,

$$SC_{prt} \frac{\Delta C}{C_A} = E_A (1.7.3),$$

E_A - usual energy of A. ΔC determined from (1.7.3) through C_A and then we can determine the complete $self_g$ -energy of A.

Remark 1.7.6. Let us consider an analogue of the Schrödinger equation for networks operating on electromagnetic energy

$$\frac{\partial w}{\partial x} = [w, \mu_g S(H)]$$

w - measure of $self_g$ for networks operating, $\mu_g S(H)$ - measure of $self_g$ for H, $H = H(\mu_g S(p), \mu_g S(q), t)$ - an analogue of the Hamiltonian in the space of actions of artificial neurons in a neural network, q is the operator of an artificial neurons action result, p is the operator of an artificial neurons action impulse.

Remark 1.7.7. The $self_g$ -space of a higher level contains many $self_g$ -energetic fibers, collecting into appropriate sets that can be accessed by the corresponding $self_g$ -spaces of lower levels. That's right, for example. This assembly point on the

human cocoon can carry out this, in particular, access to our self_g -space with objects.

Remark 1.7.8. It is quite possible to try to build up the levels of objects and processes; change something at these levels.

Remark 1.7.9. One can try to conventionally represent the mathematical model (1.7.1) of the atom (molecule) as a hierarchical dynamic operator.

Remark 1.7.10. Here, self_g -action is understood as action on oneself_g (i.e., to the same action), while physicists understand self_g -action, for example, as the absorption of one elementary particle by another of the same type.

Remark 1.7.10.1. Subtle energy can manifest itself in the form of: 1) objectivity, 2) ordinary energies, 3) information. Using neural networks of the SmnCSprt-type, it is possible to organize a SC-Internet, where instead of exchanging information, an exchange of subtle energies will take place.

1.8 PROGRAM OPERATORS SCprt, tprSC, SC¹epr, SCeprt₁

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through SCprt-Networks - analogue of Sit-Networks [14] in one of the central departments of which a conventional computer system is located. The parallel processor is itself [14] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a SCprt -program operator - SCprt $\begin{matrix} Ag \\ g \\ activation \end{matrix}$ with type of accommodation g , it will

be possible to obtain the fuzzy execution with type of accommodation g of a parallel action A with the desired target weight q or the execution of a parallel action A with the desired fuzzy target weight q or both. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For SCprt-coding and SCprt-translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical

pulses laser of Nobel laureates 2018 year Gerard Mourou, Donna Strickland, or a combination of them. For the desired action, for example, using the direct parallel

program of operator SC_{prt} $\frac{\{UHF\ AC := Q\}}{g}$ with type of accommodation g , we
activation

simultaneously enter the desired set of codes Q using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects.

Consider the types of direct parallel program operators:

- 1) SC_{prt} -program operators
- 2) $tpSC$ -program operators
- 3) SC^{1epr} - program operators
- 4) SC_{eprt_1} - program operators,

which can work with fuzzy arguments as well.

One example is pattern recognition:

SC_{prt} $\frac{\text{if } SC_{prt} \frac{\text{image archive}}{g} \ni SC_{prt} \frac{q}{B} \text{ then Name of } q}{\frac{g}{B}}$.

The example of SC_{prt} -program is

$SC_{prt} \left\{ SC_{prt} \frac{\{p\} := \{a(x)\}}{g}, SC_{prt} \frac{IF \{B\} \{f\}}{g} \text{ then } Q, SC_{prt} \frac{Q}{g} \right\}$.

Consider a third type of capacity in itself - analogue of capacity in itself. For

example, based on $SC_{prt} \frac{\tilde{x}}{g}$, where $\tilde{x} = (x_1 | \mu_{\tilde{x}}(x_1), x_2 | \mu_{\tilde{x}}(x_2), \dots, x_n | \mu_{\tilde{x}}(x_n))$, i.e. n -

elements at one point, we can consider the capacity $SC_3 f \tilde{x} g$ (in itself with m elements from $\tilde{x}$, $m < n$, which is formed according to the form (1.1),

that is, the structure $SCprt \underset{q}{\tilde{x}}$ contains only m elements. Form (1.1) can be

generalized into the following forms: (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

The ideology of SCprt and SC₃f can be used for programming.

Consider a third type of capacity in itself. For example, based on $SCprt \underset{q}{\tilde{x}}$, where

$\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$ i.e. n - elements at one point, we can consider the fuzzy capacity $SC_3f\mu$ in itself with m elements from $\tilde{x}$, $m < n$, which is formed according to the form from forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Here are some of the SCprt-program operators.

1. Simultaneous assignment with type of accommodation g of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via

$$SCprt \underset{\mu}{\underset{w}{\left\{ \tilde{x} := \{p\} \right\}}}.$$

2. Simultaneous checking with type of accommodation g by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$ Implemented via

$$SCprt \underset{\mu}{\underset{w}{IF \left\{ \tilde{B} \right\} \left\{ \tilde{g} \right\}} then \tilde{Q}} \text{ where } \tilde{Q} \text{ can be anything.}$$

3. Similarly for fuzzy loop operators and others.

SC₃f–software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding SCprt-program operators in form (1.1), for more complex operators in form from forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

For example, $\text{SCprt } g \begin{smallmatrix} \{R\} \\ w\{R\} \end{smallmatrix}$ is the fuzzy capacity with type of accommodation g in itself of the second type if $w\{R\}$ is a program capable of generating $\{R\}$ with type of accommodation g .

The example of self-ffprogram of the first type is

$$\text{SCprt } \left\{ \text{SCprt } \begin{smallmatrix} \{p\} := \{a(x)\} \\ g \\ w \end{smallmatrix}, \text{SCprt } \begin{smallmatrix} IF \{B\}\{f\} \\ g \\ w \end{smallmatrix} \text{ then } Q \right\}, \text{SCprt } \begin{smallmatrix} Q \\ g \\ Q \end{smallmatrix}.$$

The example of SCprt-program for SmnsCprt:

$\text{SCprt } \begin{smallmatrix} q := \\ g \\ B \end{smallmatrix}$ - fuzzy assigning fuzzy q to fuzzy B with type of accommodation g .

$\text{SCprt } \begin{smallmatrix} tw := \\ g \\ q \end{smallmatrix}$ - assigning target weight tw to q with type of accommodation g .

$\text{SCprt } \begin{smallmatrix} \{q\}w \\ g \end{smallmatrix}$ - SmnSCprt *activation* for fuzzy $\{q\}w$ with type of accommodation g .

SCprt-coding

SCprt-coding with type of accommodation g : 1) fuzzy set A to fuzzy set B , 2) fuzzy set A to a point q , where the elements of the fuzzy sets A , B can be

continuous. For example, $\text{SCprt } \begin{smallmatrix} A \\ g \\ B \end{smallmatrix}$.

There are SCprt-coding, SCprt-translation, SCprt-realize of eprograms and of the programs from the archives without extraction theirs

SCelf-coding

ffSelf-coding with type of accommodation g : 1) fuzzy set A to set fuzzy A , i.e. fuzzy A on itself 2) fuzzy set A to a point q in form (1), where the elements of the

fuzzy sets A , B can be continuous. For example, $\text{SCprt } \begin{smallmatrix} A \\ g \\ A \end{smallmatrix}$.

One of the central departments of the control system should be a computer system of the usual type of the desired level. In symbiosis with SCprt-Networks, it will provide a holistic operation of the control system in three modes: conventional serial through a conventional type computer system, direct parallel through SCprt-Networks and series-parallel. Codes from a conventional type computer system will be used via ffSprt -connectors in SCprt - coding, for example:

$$\text{SCprt} \quad \begin{matrix} \{ \text{UHF AC} := Q \} \\ g \\ \text{activation} \end{matrix} . \text{UHF AC field activation is used.}$$

Dynamic SCprt and SC₃f(t) programming

The ideology of dynamic SCprt and SC₃f(t) can be used for programming:

1. Simultaneous assignment of the expressions $\widetilde{p(t)} = (p_1(t) | \mu_{\widetilde{p(t)}}(p_1(t)), p_2(t) | \mu_{\widetilde{p(t)}}(p_2(t)), \dots, p_n(t) | \mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables

$\widetilde{x(t)} = (x_1(t) | \mu_{\widetilde{x(t)}}(x_1(t)), x_2(t) | \mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t) | \mu_{\widetilde{x(t)}}(x_n(t)))$. This is

implemented via
$$\text{SCprt}(t) \quad \begin{matrix} \{ \widetilde{x(t)} \} := \{ p(t) \} \\ g \\ w(t) \end{matrix} .$$

2. Simultaneous checking the fuzzy set of conditions

$\widetilde{g(t)} = (g_1(t) | \mu_{\widetilde{g(t)}}(g_1(t)), g_2(t) | \mu_{\widetilde{g(t)}}(g_2(t)), \dots, g_n(t) | \mu_{\widetilde{g(t)}}(g_n(t)))$ for the

fuzzy set of expressions $\widetilde{B(t)} = (B_1(t) | \mu_{\widetilde{B(t)}}(B_1(t)), B_2(t) | \mu_{\widetilde{B(t)}}(B_2(t)), \dots, B_n(t) | \mu_{\widetilde{B(t)}}(B_n(t)))$. Implemented via

$$\text{SCprt}(t) \quad \begin{matrix} \text{IF } \{ \{ \widetilde{B(t)} \} \{ \widetilde{g(t)} \} \} \text{ then } \widetilde{Q(t)} \\ g \\ w(t) \end{matrix} \quad \text{where } \widetilde{Q(t)} \text{ can be anything.}$$

3. Similarly for fuzzy loop operators and others.

SC₃f(t)– fuzzy software operators will differ only just because aggregates

$\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding SCprt-program operators in form (1.1) for more complex operators in form from forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

tpSC-program operators

The ideology of tpSC and $t_{SC_{4f}}$ - analogues of tS and $t_{S_{4f}}$ from [10] can be used for programming. Here are some of the tpSC -program operators.

1. Simultaneous expelling assignment of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1),$

$p_2|\mu_{\tilde{p}}(p_2), ..., p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), ...,$

$\tilde{x}$

$x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $\overset{\tilde{x}}{g} \text{ SCprt.}$

$=: \tilde{p}$

2. Simultaneous expelling of check the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1),$

$g_2|\mu_{\tilde{g}}(g_2), ..., g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1),$

$B_2|\mu_{\tilde{B}}(B_2), ..., B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through $\overset{w}{g} IF \{ \{ \tilde{B} \} \{ \tilde{g} \} \} \text{ then } \tilde{Q}^{\text{SCprt}}$

where Q can be any.

3. Similarly for loop operators and others.

$t_{SC_{4f}}$ -software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding tpSC program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical tpSC-program operator

$$\overset{B}{g} \overset{A}{\text{SCprt}} = \left\{ D + \overset{\{\}}{g} \overset{\text{SCprt}}{A - A \cap B \over (B - A \cap B)} \right\}, \text{ where D is oself-(fuzzy set) for fuzzy } (A \cap B).$$

Dynamic tpSC and $t(q)_{SC_{4f}}$ programming at time q

The ideology of tpSC and $t_{SC_{4f}}$ can be used for dynamic programming. Here are some of the tpSC(t)- dynamic programming operators.

1. The process of simultaneous expelling assignment of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), ..., p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ from the

variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), ..., x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$.is

implemented through $\begin{matrix} \widetilde{x(t)} \\ g \\ \text{SCprt}(t) \end{matrix} =: \widetilde{p}(t)$

2. The process of simultaneous expelling check the fuzzy set of conditions

$\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), ..., g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), ..., B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is

implemented through $\begin{matrix} w(t) \\ g \\ \text{SCprt}(t) \end{matrix} IF \left\{ \left\{ \widetilde{B(t)} \right\} \left\{ \widetilde{g(t)} \right\} \right\} \text{ then } \widetilde{Q(t)}$, where $Q(t)$ can be any.

3. Similarly for loop operators and others.

t_{SC4f} –software operators will differ only just because aggregates

$\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding processes $\text{tpSC}(t)$ for above mentioned programming operators through form (1.1) or forms from (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*) for more complex operators.

Consider hierarchical dynamic tprSC-program operator:

$$\begin{matrix} B(q) \\ g \\ \text{ffSprt}(q) \\ A(q) \end{matrix} = \left\{ \begin{matrix} t(q)_{SC1f(A(q) \cap B(q))} + \begin{matrix} \{ \} \\ g \\ \text{SCprt}(q) \end{matrix} \\ (B(q) - A(q) \cap B(q)) \end{matrix} \right\},$$

SC¹epr -program operators (form $\begin{matrix} B & A \\ g_2 & \text{ffS}^1 \text{prt} g_1 \\ D & B \end{matrix}$ - analogue of $\begin{matrix} B & A \\ D & B \end{matrix} S^1 t_B^A$ [12]))

For example, $\begin{matrix} \{a(t)\} \\ g_2 \\ \text{SC}^1 \text{prt} \end{matrix} IF \left\{ \left\{ B(t) \right\} \left\{ f(t) \right\} \right\} \text{ then } Q(t)$
 $\begin{matrix} \{=: \{p(t)\}\} \\ g_1 \\ \{a(t)\} \end{matrix}$.

Consider hierarchical dynamic SC¹epr-program operator: (form $\begin{matrix} B & A \\ g_2 & \text{SC}^1 \text{prt} g_1 * \\ A & B \\ D - A & B \end{matrix}$).

SCept₁- program operators (form $\begin{matrix} C & A \\ g_2 & \text{ffS}_1 \text{prt} g_1 \\ D & B \end{matrix}$ - analogue of $\begin{matrix} C & A \\ D & B \end{matrix} S_1 t_B^A$ [11]))

$afS_1t_a^a$ -- sample $\begin{pmatrix} self \\ oself \end{pmatrix}$ -fprogram structure example.

Consider structure examples hierarchical fuzzy Set₁-program operator

$$3. \begin{pmatrix} S_{01}^{et}fB \\ R-B S_1 t_B^{A-B} \end{pmatrix},$$

$$4. \begin{pmatrix} S_{21}^{et}fA^B \\ R-A S_1 t_B^A \end{pmatrix},$$

$$5. \text{SCprt} \left\{ \begin{matrix} q \begin{pmatrix} a & a \\ g_1 \text{SC1rt} g_1 & a \\ a & a \end{pmatrix} E_q \\ W_q \end{matrix} St_{q \begin{pmatrix} a & a \\ g_1 \text{SC1rt} g_1 & a \\ a & a \end{pmatrix} g} \right. , \left. St_{d_r}^{\{El^{d_r}\}} \right\} \text{---program structure example, where}$$

$$t_0$$

the assemblage point d_r is the cursor, it is quite complex self—program.

$$6. \text{SCprt} \left\{ \begin{matrix} q \begin{pmatrix} a & a \\ g_1 \text{SC1rt} g_1 & a \\ a & a \end{pmatrix} E_q \\ W_q \end{matrix} St_{q \begin{pmatrix} a & a \\ g_1 \text{SC1rt} g_1 & a \\ a & a \end{pmatrix} g} \right. , \left. St_{d_r}^{\{El^{d_r}\}} \right\} \text{can be interpreted as a program}$$

$$t_0$$

operator.

$\begin{pmatrix} a & a \\ g_1 \text{SC1rt} g_1 & a \\ a & a \end{pmatrix}$ can be interpreted as a $\begin{pmatrix} s_1 elf \\ os_1 elf \end{pmatrix}$ -fprogram operator.

$\begin{pmatrix} a & a \\ g_1 \text{SC1rt} g_1 & a \\ a & a \end{pmatrix}$ sample $\begin{pmatrix} s_1 elf \\ os_1 elf \end{pmatrix}$ -fprogram structure example.

Remark 1.8. Energy of a living organism:

$$\text{Cg(r, a}(E_q)) = \text{SCprt} \left\{ \begin{matrix} q \begin{pmatrix} a & a \\ g_1 \text{SCrt} g_1 & a \\ a & a \end{pmatrix} E_q \\ W_q \end{matrix} fSt_{q \begin{pmatrix} a & a \\ g_1 \text{SCrt} g_1 & a \\ a & a \end{pmatrix} g} \right. , \left. fSt_{d_r}^{\{El^{d_r}\}} \right\}$$

$$t_0$$

$\begin{pmatrix} a & a \\ g_1 \text{SCrt} g_1 & a \\ a & a \end{pmatrix}$ -internal energy of a living organism, q- a gap in the energy

cocoon of a living organism, r-the position of the assemblage point d_r on the energy cocoon of a living organism, W_q - energy prominences from the gap

in the cocoon of a living organism, E_q -external energy entering the gap in the cocoon of a living organism, El^{dr} - a bundle of fibers of external energy self-capacities, collected at the point of assembly of the cocoon of a living organism.

Appendix 1

If we introduce for the energy of a chemical element the concept self_g -energy (the

concept of a chemical element was introduced earlier): $SC_{prt} \frac{R}{g}$, $R=Q+D$, Q -

internal energy, D is the energy of its interaction with the external environment.

$$SC_{prt} \frac{R}{g} = SC_{prt} \frac{Q+D}{g} = SC_{prt} \frac{Q}{g} + SC_{prt} \frac{D}{g} = SC_{prt} \frac{Q}{g} + SC_{prt} \frac{Q}{D} + SC_{prt} \frac{D}{Q} + SC_{prt} \frac{D}{D},$$

$SC_{prt} \frac{Q}{g}$ - internal self_g -energy, $SC_{prt} \frac{D}{g}$ -the external self_g -energy, $SC_{prt} \frac{Q}{D}$ - object

component of a chemical element, $SC_{prt} \frac{D}{g}$ - usual energy component of a chemical

element. We describe the usual chemical reactions for the $SC_{prt} \frac{Q}{g}$ -component using

the $SC_{prt} \frac{D}{g}$ -component. A self_g -molecule (self_g -atom, self_g -(elementary particle)))

as a capacity can have the following types of self_g: self_g -set, self_g -structure, self_g - hierarchy or its elements that generates this self_g -molecule (self_g -atom, self_g - (elementary particle))).

self_g -power is force that is applied to oneself_g or its elements that generates this self_g -power.

You can try to consider the equations: $SC_{pr} \frac{x}{g} = a$, $x(a) - ?$, $SC_{pr} \frac{x}{g} = a$, $x(a, b) - ?$,

$SC_{pr} \frac{q}{g} = a$, $x(a, q) - ?$.

Supplement for Quantum Mechanics and Classical statistical Mechanics through SCprt-elements:

Hamilton operator $\widehat{H} = \widehat{H}_0 + \widehat{W}_0$, $\widehat{H}_0$ -considered quantum system energy, consisting of two or more parts, without their interaction with each other, $\widehat{W}_0$ is the energy of their interaction, $\hat{\rho}$ -statistical operator [20]. self_g -energy $\text{SCpr } \frac{\widehat{H}}{\widehat{H}} =$

$$\text{SCpr } \frac{\widehat{H}_0 + \widehat{W}_0}{\widehat{H}_0 + \widehat{W}_0} = \text{SCpr } \frac{\widehat{H}_0}{\widehat{H}_0 + \widehat{W}_0} + \text{SCpr } \frac{\widehat{W}_0}{\widehat{H}_0 + \widehat{W}_0} = \text{SCpr } \frac{\widehat{H}_0}{\widehat{H}_0} + \text{SCpr } \frac{\widehat{W}_0}{\widehat{W}_0} +$$

$$\text{SCpr } \frac{\widehat{W}_0}{\widehat{H}_0} + \text{SCpr } \frac{\widehat{W}_0}{\widehat{W}_0}, \text{SCpr } \frac{\widehat{H}_0}{\widehat{H}_0} \text{ -considered quantum system self}_g \text{ -energy, SCpr } \frac{\widehat{W}_0}{\widehat{W}_0} \text{ is}$$

self_g -energy of their interaction, $\text{SCpr } \frac{\widehat{H}_0}{\widehat{W}_0}$ --object manifestation of the energy of the

system in an external field., $\text{SCpr } \frac{\widehat{W}_0}{\widehat{H}_0}$ - the manifestation of the energy of the system

in the energy interaction with the external field. Variants of the Schrödinger

equation $\frac{\partial \hat{\rho}}{\partial t} + [\hat{W}, \hat{\rho}] = 0$ of the form SC₂f, SC₃f are possible, using the form (1.1) or form from the forms (1.1.1) – (1.4).

The carrier of the measure of objectivity-mass should be objectivity - elementary

particle graviton, look like $\text{SCpr } \frac{\text{objectivity}}{\text{objectivity}}$, therefore it is a self_g -particle and is

not an element of the level of objectivity, but is an element of the level self_g.

Therefore, it cannot be found at our level. In fact, the theory of SCprt-elements helps to form a unified field theory on a qualitative level, because it is not possible to create a quantitative unified field theory. Supplement for string theory: May be to try represent elementary particles in the form of continual self_g -elements of the type $CS_{\infty}^{-} = \sin(-\infty)|g \rightarrow \downarrow I \uparrow_{-1}^1|g$, $CT_{\infty}^{+} = \text{tg}\infty|g \rightarrow \uparrow I \downarrow_{-\infty}^{\infty}|g$, $CT_{\infty}^{-} = \text{tg}(-\infty)|g \rightarrow \downarrow I \uparrow_{-\infty}^{\infty}|g$, $f \uparrow I \downarrow w|g$ for any f, w etc.

We consider SCprt-logic: consider the functional $fC(Q)$, which gives a numerical value for the truth_g of the statement Q from the interval $[0,1]$, where 0 corresponds to "no," and one corresponds to the logical value "yes." Then for joint statements A, B : $fC(A+B)=fC(A)+fC(B)-fC(A*B)+fSC(D)$, D - self_g-statement from $A*B$, $fSC(x)$ - the value of self_g-truth for self_g-statement x ; for dependent statements: $fC(A*B)=fC(A)*fC(B/A)=fC(B)*fC(A/B)$, where $fC(B/A)$ - conditional truth_g of the statement B at statement A , $fC(A/B)$ - dependent truth_g of statement A at the statement B . Adding the truth_g values of inconsistent propositions:

$fC(A+B)=fC(A)+fC(B)$. The formula of complete truth_g: $fC(A)=\sum_{k=1}^n fC(B_k) * fC(A/B_k)$, $B_1, B_2, \dots, B_n$ -full group of hypotheses-statements:

$$\sum_{k=1}^n fC(B_k)=1(\text{"yes"}).$$

Remark. A statement can be interpreted as an event, and its truth value as a probability.

SCprt- statement for set of statements $A=\{A_1, A_2, \dots, A_n\}$: $SCprt_{g_1, x}^{\{A_1, A_2, \dots, A_n\}}$,

$SCprt_{g_1, x}^{\{f(A_1), f(A_2), \dots, f(A_n)\}}$, - SCprt- truth for these statements. It is possible to

consider the self_g-statement SC_3A with m statements from A , at $m < n$, which is formed by the form (1.1), that is, only m statements from A are located in the

structure $SCprt_{g_1, x}^A$. The same for self_g- truth $SC_3\{f(A_1), f(A_2), \dots, f(A_n)\}$.

One can introduce the concepts of SCprt-group: $SCprt_{g_1, x}^A$, A is usual group, $SCprt_{g_1, B}^A$,

where A, B - usual groups, self-group: SCf_iA , $i=1,2,3$, A is usual group.

Definition A. A structure with a second degree of freedom will be called complete, i.e., "capable" of reversing itself_g concerning any of its elements clearly, but not necessarily in known operators; it can form (create) new

special operators (in particular, special functions). In particular, $CCprt_{g_1, A}^A$, $CCpr_{g_1, A}^A$

are such structures. Similarly, for working with models, each is structured by its

structure; for example, use SCprt-groups, SCprt-rings, SCprt-fields, SCprt-spaces, self_g-groups, self_g-rings, self_g-fields, and self_g-spaces. Like any task, this is also a structure of the appropriate capacity. Since the degree of freedom is double, it is clear that the form of the self_g-equation contains a solution or structures the inversion of the self_g-equation concerning unknowns, i.e., the structure of the self_g-equation is complete. The transition process in the form of $\begin{matrix} C & A \\ g_2 \text{SCprt} g_1 & \\ D & B \end{matrix}$ is included in the transition from one world A (spatial variables, which we will denote by X1, and time variables, by T1) to another world B (spatial variables, which we will denote by X2, and time variables, by T2). It is accompanied by spatial variables in the form (T1, X1), i.e., such a transition process transforms time variables T1 into part of spatial ones, and time variables - T3.

Supplement

Connection SCprt – elements with usual functionals and operators

We consider functional $g(x): X \rightarrow g, x \in X, g$ —numerical value of functional $g(x)$. It is specific capacity for X. $\text{SCprt}_{g_1}^{\{g(x)\}}$ made from her the self_g-capacity in itself_g as an element $\text{SC}_1 f\{g(x)\}, \{g(x)\}$ —the set of any functionals for X. In particular, probability $p(X)$ —is such functional, X—an event. Here $\text{SCprt}_{g_1}^{\{p(x)\}}$ is $\text{SC}_1 f p(X), \{p(x)\}$ denote it through $p\text{SC}(X)$. Usual event is dynamical capacity.

Definition B. SCprt-probability of events A, B is $p(\text{SCprt}_{g_1}^A, \text{SCprt}_{g_1}^B)$, denote $\text{SCpp}_{g_1}^A$. In particular,

$\text{SCpp}_{g_1}^A$ for joint A, B: $\text{SCpp}_{g_1}^A = p(\text{SCprt}_{g_1}^A) = p(\{A \cup B - A \cap B\}) * \mu(g_1) = p(A) + p(B) - p(AB) + p\text{SC}(D)$, D - the self_g-capacity in itself_g as an element from $A \cap B$, $p\text{SC}(D)$ —probability self_g of D of next level— self_g level. The probability for stochastic value X is capacity. We represent its distribution in the kind of SCprt-element:

$$\text{SCpp}_{g_1}^x \{ (x_1, p_1), (x_2, p_2), \dots, (x_n, p_n) \} (*)$$

Here interest represent partial distribution self_g from (*) by form (1.1) or form from the forms (1.1.1) – (1.4) with value self_g of stochastic value X for some subset $\{x_{1_1}, x_{2_1}, \dots, x_{j_1}\} \in \{x_1, x_2, \dots, x_n\}$ with probabilities $\text{self}_g \{pCS_1, pCS_2, \dots, pCS_j\}$.

For operator $X_1 \xrightarrow{F} X_2$: $\xrightarrow{F} X_2$ is capacity for X_1 . $\text{SCprt } g_1 \xrightarrow{F} X_2$ -- self_g - capacity in itself self_g

as an element for X_1 . More complex for implicit operator: $F(X_1, X_2)=0$. Then

$F(X_1, X_2) = 0$
 $\text{SCprt } g_1$ forms self_g -capacity in itself self_g as an element for X_1 relatively of $F(X_1, X_2) = 0$

X_2 or for X_2 relatively of X_1 . x obtains more power of the liberty and in this is direct decision (i. e. self_g -capacity in itself self_g as an element for x). self_g -equation for x has its decision for x in direct kind. self_g -task for x has its decision for x in direct kind. self_g -question has its answer for x in direct kind. x acquires more degree of liberty

and in this is direct decision. We consider $\text{SCprt } g_1$, D -block over execution subject D

in $S_{mnSCprt}$ for networks. Then we have self_g -capacity in itself self_g as an element D ,

where full realization requires correspondent self_g -energy. $\text{SCprt } g$ increase $S_{mnSCprt}$

self_g -level of $S_{mnSCprt}$ and may made no visual its. The entire neural network as instantaneous simultaneous RAM in SCprt -elements and self_g -

elements. $\text{self}_g^{\text{self}_g \dots \text{self}_g}$, $f_1 \downarrow I \uparrow_{-1}^1 f_2 \dots f_1 \downarrow I \uparrow_{-1}^1 f_2 \dots f_1 \downarrow I \uparrow_{-1}^1 f_2$, $\sin \infty |g|^{\sin \infty |g| \dots \sin \infty |g|}$. When activated in a neural network, the entire neural network becomes a working memory. Use of self_g -energy as activation or from outside. $QC_0 =$

$\text{SCprt } g_1$ $\xrightarrow{\text{activation}}$ self_g -RAM, $QC_{00} = \text{SCprt } g_2$ $\xrightarrow{\text{activation}}$ QC_0 , $QC_{01} = \text{SCprt } g_2$ $\xrightarrow{\text{activation}}$ QC_0 .
 $\text{SCprt } g_1$ $\xrightarrow{\text{activation}}$ QC_0 .
 $\text{SCprt } g_1$ $\xrightarrow{\text{activation}}$ QC_0 .

QC_0 , QC_{00} , QC_{01} -coding, translation, realization eprograms, QC_0 , QC_{00} , QC_{01} -

$S_{mnSCprt}$, QC_0 , QC_{00} , QC_{01} -Assembler.

References

- 1) A. Mostowski. Constructive sets. UNIVERSITY OF WARSAW-North-Holland Publishing Company-Amsterdam. PWN-Polish Scientific Publishers-Warszawa. 1969.
- 2) Thomas J. Jech. LECTURES IN SET THEORY WITH PARTICULAR EMPHASIS ON THE METHOD OF FORCING. SPRINGER-VERLAG Berlin. Heidelberg. New-York, 1971.
- 3) Oleksandr Danilishyn, Illia Danilishyn. Introduction to Dynamic Sets Theory: SCprt-elements and Their Applications to the Physics and Chemistry. JOURNAL OF PHYSICS AND CHEMISTRY, vol. 2, issue 3, pp. 1-31, March 27, 2024 https://cskscientificpress.com/articles_file/527-_article1712797848.pdf
- 4) Ershov Y.L. On a Hierarchy of Sets I, Algebra and Logic, 7 (1968), 47 - 73.
- 5) Ershov Y.L.] On a Hierarchy of Sets II, Algebra and Logic, 7 (1968), 15 - 47.
- 6) Ershov Y.L. On a Hierarchy of Sets III, Algebra and Logic, 9(1970), 34 -
- 7) Krain S.G. Linear differential equations in Banach space. M., Science, 1967 (in Russian).
- 8) A.L. Dontchev. Perturbations, approximations and sensitivity analysis of optimal control systems. Springer-Verlag, Berlin, New York, 1983.
- 9) Galushkin A. Networks: principles of the theory. Hot line-Telecom. M., 2010 (in Russian).
- 10) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the V International Scientific and Practical Conference, Oxford, June 23, 2023. Oxford-Vinnitsia: P.C. Publishing House & European Scientific Platform, 2023. Theoretical and empirical scientific research: concept and trends. pp. 156-161, DOI 10.36074/logos-23.06.2023.42, <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/12>

- 11) Danilishyn I.V. Danilishyn O.V. SET1 – ELEMENTS.
INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 28
June, 2023 with the proceedings of the:Correspondence International
Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC
AND MODERN TOOLS AND METHODS IN SCIENTIFIC
INVESTIGATIONS held on June 9 th, 2023 by NGO European Scientific
Platform (Vinnytsia, Ukraine) LLC International Centre Corporative
Management (Vienna, Austria), pp. 239-254. <https://archive.journal-grail.science/index.php/2710-3056/issue/view/09.06.2023>
- 12) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets S1et and
Some of their Applications in Physics. Science Set Journal of Physics, 25
September 2023,1-11, 815-_ article1695707465.pdf (mkscienceset.com)
- 13) I. Danilishyn, O. Danilishyn Program Operators Sit, tS, S1e, Set1.
Journal of Sensor Networks and Data Communications [ISSN: 2994-6433]
2023, Volume 3 | Issue 1 | 138-143, <https://www.opastpublishers.com/table-contents/jsndc-volume-3-issue-1-year-2023>
- 14) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS
FOR NETWORKS. IV International Scientific and Practical Conference
”GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN
FORSCHUNG ”, 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 15) Danilishyn I.V. Danilishyn O.V. DYNAMICAL SIT-ELEMENTS.
IV International Scientific and Practical Conference ”GRUNDLAGEN DER
MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ”,
31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 16) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-
ELEMENTS TO SETS THEORY AND OTHERS. Scientific practice:
modern and classical research methods: Collection of scientific papers
«ΛΟΓΟΣ» with Proceedings of the IV International Scientific and Practical

Conference, Boston, May 26, 2023. Boston-Vinnytsia: Primedia eLaunch & European Scientific Platform, 2023, pp.166-171. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/11>

- 17) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO CONTINUAL VALUED LOGIC AND OTHERS. Features of the development of modern science in the pandemic's era: a collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, May 19, 2023. Berlin, Federal Republic of Germany: European Scientific Platform, pp.79-84. <https://previous.scientia.report/index.php/archive/issue/view/19.05.2023>
- 18) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. [Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1).
<https://doi.org/10.21203/rs.3.rs-3217178/v1>
[Dynamic Sets Theory: Sit-elements and Their Applications | Research Square](https://www.researchsquare.com/article/rs-3217178/v1)
www.researchsquare.com/article/rs-3217178/v1
- 19) Danilishyn I., Danilishyn O. VARIABLE HIERARCHICAL DYNAMICAL STRUCTURES (MODELS) FOR DYNAMIC, SINGULAR, HIERARCHICAL SETS AND THE PROBLEM OF COLD THERMONUCLEAR FUSION The driving force of science and trends in its development: collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, July 14, 2023. Coventry, United Kingdom: European Scientific. Pp.113-119. Platform. <https://previous.scientia.report/index.php/archive/issue/view/14.07.2023>
- 20) D.I.Blokhintsev. Quantum Mechanics Lectures on Selected Topics. Moscow, Atomizdat, 1981.

Declarations:

Availability of data and material

- 1) Oleksandr Danilishyn, Illia Danilishyn. Introduction to Dynamic Sets Theory: SCprt-elements and Their Applications to the Physics and

Chemistry. JOURNAL OF PHYSICS AND CHEMISTRY, vol. 2, issue 3, pp. 1-31, March 27, 2024 https://cskscientificpress.com/articles_file/527-article1712797848.pdf

- 2) Illia Danilishyn and Oleksandr Danilishyn. Dynamic Sets Set and Some of Their Applications to Neuroscience, Networks Set New Advances in Brain & Critical Care 2023, Volume 4 | Issue 2 | 66- 81.
<https://doi.org/10.33140/NABCC.04.02.02>
- 3) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets Set and Some of their Applications in Physics. Science Set Journal of Physics, 25 September 2023, 1-11, 815-__ article1695707465.pdf (mkscienceset.com)
- 4) I. Danilishyn, O. Danilishyn Dynamic Sets Se, Networks Se. *Advances in Neurology and Neuroscience*, 2023, Volume 6 | Issue 2 | 278-294.
<https://www.opastpublishers.com/open-access-articles/dynamic-sets-se-networks-se.pdf>
- 5) I. Danilishyn, O. Danilishyn Program Operators Sit, tS, S1e, Set1. Journal of Sensor Networks and Data Communications [ISSN: 2994-6433] 2023, Volume 3 | Issue 1 | 138-143, <https://www.opastpublishers.com/table-contents/jsndc-volume-3-issue-1-year-2023>
- 6) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 7) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the V International Scientific and Practical Conference, Oxford, June 23, 2023. Oxford-Vinnytsia: P.C. Publishing House & European Scientific Platform, 2023. Theoretical and empirical scientific research: concept and trends. pp.156-161, DOI 10.36074/logos-23.06.2023.42, <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/12>

- 8) Danilishyn I.V. Danilishyn O.V. SET1 – ELEMENTS. INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 28 June, 2023 with the proceedings of the:Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 9 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC International Centre Corporative Management (Vienna, Austria), pp. 239-254. <https://archive.journal-grail.science/index.php/2710-3056/issue/view/09.06.2023>
- 9) Danilishyn I.V. Danilishyn O.V. DYNAMICAL SIT-ELEMENTS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 10) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO SETS THEORY AND OTHERS. Scientific practice: modern and classical research methods: Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the IV International Scientific and Practical Conference, Boston, May 26, 2023. Boston-Vinnytsia: Primedia eLaunch & European Scientific Platform, 2023, pp.166-171. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/11>
- 11) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO CONTINUAL VALUED LOGIC AND OTHERS. Features of the development of modern science in the pandemic's era: a collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, May 19, 2023. Berlin, Federal Republic of Germany: European Scientific Platform, pp.79-84. <https://previous.scientia.report/index.php/archive/issue/view/19.05.2023>
- 12) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. [Research Square](https://www.researchsquare.com/publication/preprint/10.21203/rs.3.rs-2881111/v1).

<https://doi.org/10.21203/rs.3.rs-3217178/v1>

[Dynamic Sets Theory: Sit-elements and Their Applications | Research Square](https://www.researchsquare.com/article/rs-3217178/v1)

www.researchsquare.com/article/rs-3217178/v1

Competing interest

There are no competing interests. All sections of the monograph are executed jointly.

Funding

There were no sources of funding for writing the monograph.

Authors' contributions

The contribution of the authors is the same, we will not separate.

Part II. Fuzzy SCprt – elements and their applications

Introduction

Significance of the section: in a new qualitatively different approach to the study of complex processes through new mathematical, fuzzy hierarchical, fuzzy dynamic structures, in particular those processes that are dealt with by Synergetics.

We consider expression

$$\begin{array}{c} C \quad A \\ g_2 \text{SfCprt} g_1 (*_{2.1}) \\ \mu_2 \quad \mu_1 \\ D \quad B \end{array}$$

where A fuzzy fits into B with type of accommodation g_1 and measure of fuzziness μ_1 , D is forced out from C with type of accommodation g_2 measure of fuzziness μ_2 ; A, B, C, D, g_1 , g_2 may also be fuzzy. The result of this process will be described by the expression

$$\begin{array}{c} C \quad A \\ g_2 \text{SfCrt} g_1 (*_{2.2}) \\ \mu_2 \quad \mu_1 \\ D \quad B \end{array}$$

If A, B, D, C are taken as fuzzy sets, then we will call $(*_{2.1})$ a fuzzy SC-dynamic fuzzy set. The need $(*_{2.1})$ arose to describe processes in networks. Threshold

element SfCprt -- $\begin{array}{c} b \\ g_2 \\ \mu_2 \end{array} \text{SfCprt} \begin{array}{c} \{ax\} \\ g_1 \\ \mu_1 \\ b \end{array}$, b- artificial neurons of type SfCprt (designation -

mnSt), $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|, \dots, x_n|\mu_{\tilde{x}}(x_n))$ is the fuzzy set of the values of the initial signals,, $a=(a_1, a_2, \dots, a_n)$ are the weights of SfCprt-synapses and $\tilde{y}=(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)|, \dots, y_n|\mu_{\tilde{y}}(y_n))$ is the fuzzy set of values of the output signals with weights $q=(q_1, q_2, \dots, q_n)$. It can be considered a simpler version of the dynamic set

$$\begin{array}{c} A \\ \text{SfCprt} g_1 \\ \mu_1 \\ B \end{array} (**_{2.1})$$

where set A fuzzy fits with measure of fuzziness μ_1 into set B with type of accommodation g_1 , the result of this process will be described by the expression

$$\begin{matrix} A \\ \text{SfCrt}_{\mu_1}^{g_1} (**_{2.2}) \\ B \end{matrix}$$

or

$$\begin{matrix} C \\ g_2 \text{SfCprt} (**_{2.1}) \\ \mu_2 \\ D \end{matrix}$$

where set A is fuzzy forced out with measure of fuzziness μ_2 from B with type of accommodation g_2 , the result of this process will be described by the expression

$$\begin{matrix} C \\ g_2 \text{SCrt} (**_{2.2}) \\ D \end{matrix}$$

We consider the measure: $\mu_{\frac{b}{D}}^{**}(\frac{A}{b} \text{SfCprt}_{g_1}^{g_2}) = \frac{\mu(A)\mu(g_1)\mu_1}{\mu(D)\mu(g_2)\mu_2}$, where $\mu(A), \mu(D)$ –usual measures of sets A, D, $\mu(g_1), \mu(g_2)$ - measures corresponding to the accommodations of the corresponding type.

Remark. One can consider some generalization for $(*_{2.1})$: $\begin{matrix} q_1(C) \\ g_2 \\ \mu_2 \\ D \end{matrix} \text{SfCprt} \begin{matrix} A \\ g_1 \\ \mu_1 \\ q(B) \end{matrix}$, where A

is fuzzy contained with measure of fuzziness μ_1 into B through q with type of accommodation g_1 , D is fuzzy forced out with measure of fuzziness μ_2 from C through q_1 with type of accommodation g_2 , A, B, D, C are taken as sets. The result

of this process will be described by the expression $\begin{matrix} q_1(C) \\ g_2 \\ \mu_2 \\ D \end{matrix} \text{SfCrt} \begin{matrix} A \\ g_1 \\ \mu_1 \\ q(B) \end{matrix}$.

Similarly, for $(**_{2.1})$: $\begin{matrix} A \\ \text{SfCprt}_{\mu_1}^{g_1} \\ B \end{matrix}$, where A is fuzzy contained with measure of fuzziness

μ_1 into B through q with type of accommodation g_1 (the result of this process will

be described by the expression $\begin{matrix} A \\ g_1 \\ \mu_1 \\ q(B) \end{matrix} \text{SfCrt} \begin{matrix} q_1(C) \\ g_2 \\ \mu_2 \\ D \end{matrix}$, for $(***_{2.1})$: $\begin{matrix} q_1(C) \\ g_2 \\ \mu_2 \\ D \end{matrix} \text{SfCprt}$, where D is fuzzy

displaced with measure of fuzziness μ_2 from C through q_1 with type of

accommodation g_2 . The result of this process will be described by the expression

$$q_1(C) \begin{matrix} g_2 \\ \mu_2 \\ D \end{matrix} \text{ SfCrt.}$$

We construct new mathematical objects constructively without formalism. By its contradiction, formalism may destroy this thry by Gödel's theorem on the incompleteness of any formal theory. But in the next monograph, we will give the formalism of the theory it's due: the proof of axioms and theorems. Let us introduce the concepts Cha, the capacity measure, and Cca, the measure of its content. Cca is the same as the number of capacity content items. Consider the

compression ratios of the dynamic set: $q_1 = \text{SfCprt} \begin{matrix} A \\ \mu_1 \\ B \end{matrix} g_1$ answers I compression power of

dynamic set A, $q_2 = \text{SfCprt} \begin{matrix} q_1 \\ \mu_1 \\ B \end{matrix} g_1$ –II compression power of dynamic set A, ..., $q_{n+1} =$

$\text{SfCprt} \begin{matrix} q_n \\ \mu_1 \\ B \end{matrix} g_1$ –n+1 compression power of dynamic set A. In contrast to the classical

one-attribute set theory, where only its contents are taken as a set, we consider a two-attribute set theory with a set as a capacity and separately with its contents. We introduce the designations: CoQ—the contents of the capacity Q.

2.1 SfCprt - elements

Definition 2.1.1. The set of elements $\{a\} = (a_1, a_2, \dots, a_n)$ contained with measure of fuzziness μ_1 into point x of space X with accommodation type g_1 we shall call SfCprt – element, and such a point in space is called capacity of the SfCprt –

element. We shall denote $\text{SfCprt} \begin{matrix} \{a\} \\ \mu_1 \\ x \end{matrix} g_1$. The result of this process will be described by

the expression $\text{SfCrt} \begin{matrix} \{a\} \\ \mu_1 \\ x \end{matrix} g_1$.

Definition 2.1.2. $\text{SfCprt} \begin{smallmatrix} \{a\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix}$ — SfC-dynamic set $\{a\}$ at x .

Definition 2.1.3. An ordered set of elements at one point in space is called an ordered SfCprt–element.

It's possible to $\text{SfCprt} \begin{smallmatrix} \{a\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix}$ correspond to the set of elements $\{a\}$, and the ordered

SfCprt - element - a vector, a matrix, a tensor, a directed segment in the case when the totality of elements is understood as a set of elements in a segment.

It's allowed to sum SfCprt – elements: $\text{SfCprt} \begin{smallmatrix} \{a\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix} + \text{SfCprt} \begin{smallmatrix} \{b\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix} = \text{SfCprt} \begin{smallmatrix} \{a\} \cup \{b\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix}$.

The operator $\text{SCprtSfCprt} \begin{smallmatrix} \{a\} \cup \{b\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix}$ is not equal $\{a\} \cup \{b\}$, rather, it is dynamic —

contraction of $\{a\} \cup \{b\}$ to the point x . Similarly, for $\text{SfCprt} \begin{smallmatrix} \{a\} \cap \{b\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix}$.. This is more

suitable for using sets for energy space, for any objects. The operator SfCprt is adapted for ordinary energies, using their property to overlap.

Capacity in itself_g

Definition 2.1.4. The fuzzy capacity A in itself_g of the first type is the fuzzy capacity fuzzy containing with measure of fuzziness μ_1 itself_g as an element with accommodation type g_1 . Denote $SfC_1f_{\mu_1}A\{g_1\}$.

Definition 2.1.5. The capacity A in itself_g of the second type is the capacity that contains with measure of fuzziness μ_1 elements from which it can be generated with accommodation type g_1 . Denote $SfC_2f_{\mu_1}A\{g_1\}$.

An example of the fcapacity in itself_g of the first type is a set containing with measure of fuzziness μ_1 itself_g. An example of capacity in itself_g of the second type is a living organism since it contains a program: DNA and RNA.

Definition 2.1.6. Partial fcapacity A in itself_g of the third type is the fcapacity A in itself_g as an element with accommodation type g_1 , which partially contains with measure of fuzziness μ_1 itself_g or contains elements from which it can be generated in part with accommodation type g_1 or both. Let us denote $SfC_3f\mu_1A\{g_1\}$.

Let us introduce the following notations: $A*B = \underset{B}{\overset{A}{SfCrt_{\mu_1}^{g_1}}}$, $A^2 = fself_{g_1} A = \underset{A}{\overset{A}{SfCrt_{\mu_1}^{g_1}}}$, $A^3 = fself_{g_1}^2 A$, ..., $A^{n+1} = fself_{g_1}^n A$, ... There is no commutativity here: $A*B \neq B*A$. We can consider operator functions: $e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots$, $(A+B)^n = \sum_{k=0}^n \binom{n}{k} A^k B^{n-k}$, $(1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \dots$, etc.

You can consider a more “hard” option: $A*B = \underset{B}{\overset{A}{PsfCrt_{\mu_1}^{g_1}}}$, where $\underset{B}{\overset{A}{PsfCrt_{\mu_1}^{g_1}}}$ operator, containing A in every element of B , $A^2 = Pself_{g_1} A = \underset{A}{\overset{A}{PsfCrt_{\mu_1}^{g_1}}}$, $A^3 = Pself_{g_1}^2 A$, ..., $A^{n+1} = Pself_{g_1}^n A$, ... There is no commutativity here: $A*B \neq B*A$. We can consider operator functions: $e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots$, $(A+B)^n = \sum_{k=0}^n \binom{n}{k} A^k B^{n-k}$, $(1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \dots$, etc.

All capacities in fself_g-space are capacities in themselves by definition. fcapacities in themselves can appear as SfCprt-capacities and ordinary capacities. In these cases, the usual measures and methods of topology are used.

Connection of SfCprt – elements with capacities in themselves.

For example, $\underset{w\{R\}}{\overset{\{R\}}{SfCrt_{\mu_1}^{g_1}}}$ is the fcapacity in itself_g of the second type if $w\{R\}$ is a

program capable of generating $\{R\}$.

Consider a third type of fuzzy fcapacity in itself_g. For example, based on

$\underset{q}{\overset{\tilde{x}}{SfCrt_{\mu_1}^{g_1}}}$, where $\tilde{x} = (x_1 | \mu_{\tilde{x}}(x_1), x_2 | \mu_{\tilde{x}}(x_2), \dots, x_n | \mu_{\tilde{x}}(x_n))$ i.e. n - elements at one

point, we can consider the fuzzy fcapacity $SfC_3f_{\mu_1}\tilde{x}\{g_1\}$ in itself with m elements from $\tilde{x}$, $m < n$, which is formed according to the form (1.1), that is, the structure

$SfCrt_{\mu_1}^{\tilde{x}, g_1}$ contains only m elements, or in forms (1.1.1) - (1.1.5), summarizing it.

Fuzzy fcapacities in themselves_g of the third type can be formed for any other structure, not necessarily $SfCrt$, only by necessarily reducing the number of elements in the structure, in particular, using form (1.2). Structures more complex than $SfC_3f_{\mu_1}\tilde{x}\{g_1\}$ can be introduced. For example, through a form (1.3), where A is fuzzy compressed (fuzzy fits) in C in the fuzzy compression fuzzy structure B in

C (i.e., in the fuzzy structure $SfCrt_{\mu_1}^{\frac{B}{C}, g_1}$); or through the forms (1.3.1) - (1.4) and corresponding generalizations of (1.4) on (1.3.1) - (1.3.4), etc.

(1.3.1) - (1.3.4) schematically interpret the fuzzy formation of fuzzy capacity in itself through a pseudo 3-connected form with a 2-connected form. The ideology of $ffSprt$ and $SfC_3f_{\mu_1}\tilde{x}\{g_1\}$ can be used for programming.

Remark 2.1.1. Fuzzy self, in particular, according to a fuzzy form- fuzzy analogue of the form of type (1.1):

$(1 \parallel_{\mu_1}, (2 \mid_{\mu_2}, 1 \parallel_{\mu_3}))$ (2.1*),

μ_i ($i=1,2,3$) – the fuzziness of the indicated positions. For example

- 1) fuzzy forming from element with fuzziness μ in the form $\{2,1\}$: $(1 \parallel_{\mu}, (2,1))$
- 2) fuzzy forming from element in the form $\{2,1\}$ with fuzziness μ : $(1, (2,1) \parallel_{\mu})$
- 1) fuzzy formation of partial self in the form (1.1) with fuzziness μ : $(1, (2,1)) \parallel_{\mu}$
- 2) It is also possible to generalize the other remaining forms (1.1) – (1.4) to fuzzy forms
- 3) etc.

Math fuzzy self_g fuzzy.

Let's consider SfCprt arithmetic first:

1. Simultaneous fuzzy addition with measure of fuzziness μ_1 and accommodation type g_1 of a fuzzy set of elements $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1),$

$$\tilde{x} + x_2|\mu_{\tilde{x}}(x_2)|, \dots, x_n|\mu_{\tilde{x}}(x_n)|) \text{ is carried out using SfCprt } \frac{g_1}{\mu_1} \cdot q.$$

2. Similarly, for simultaneous fuzzy multiplication with measure of fuzziness μ_1

$$\tilde{x} * \text{and accommodation type } g_1: \text{SfCprt } \frac{g_1}{\mu_1} \cdot q \text{ gives the set B with elements}$$

$$b_{i_1 i_2 \dots i_n} = \left\{ x_{1_{i_1}} * \mu_{\tilde{x}}(x_{1_{i_1}}) *, x_{2_{i_2}} * \mu_{\tilde{x}}(x_{2_{i_2}}) *, \dots, x_{n_{i_n}} * \mu_{\tilde{x}}(x_{n_{i_n}}) \right\} \text{ (SfCprt } \frac{g_1}{\mu_1} \cdot q)_R ,$$

$$\text{for any } \{i_1, i_2, \dots, i_n\} \text{ without repetitions, } q = \text{SfCprt } \frac{g_1}{\mu_1} \cdot \frac{\{K\}}{w}, \text{ K-set of any}$$

$$\{k_1 *, k_2 *, \dots, k_n *\} \text{ without repeating them, } k_i \text{-any digit, } i=1,2,\dots,n, R = \text{SfCprt } \frac{g_1}{\mu_1} \cdot \frac{\{i_1 +, i_2 +, \dots, i_n\}}{w}, R \text{ is the index of the lower discharge (we choose an}$$

index on the scale of discharges): see Table I.1.

$$\text{Then SfCprt } \frac{g_1}{\mu_1} \cdot \frac{\{B +\}}{w} \text{ gives the final result of simultaneous multiplication. Any}$$

system of calculus can be chosen, in particular binary. The most straightforward functional scheme of the assumed arithmetic-logical device for SfCprt-multiplication:

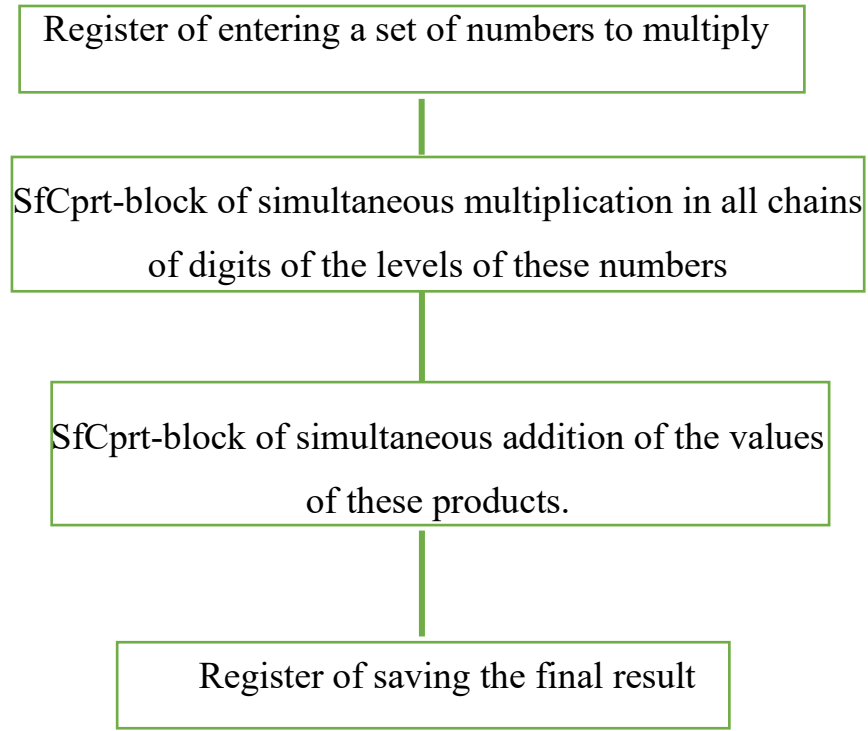


Fig. I.2. The straightforward functional scheme of the assumed arithmetical device for SfCprt-multiplication.

Remark 2.1.1.1. The algorithm for simultaneously multiplication of a set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by multiplying the first number from the set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the set by the ones following it, etc.

3. Similarly for simultaneous execution of various operations: $\text{SfCprt}_{\mu_1^x}^{\tilde{x}\tilde{q}}$, where

$$\tilde{q}=(q_1|\mu_{\tilde{q}}(q_1), q_2|\mu_{\tilde{q}}(q_2)|, ..., q_n|\mu_{\tilde{q}}(q_n)), \quad q_i \text{ -an operation, } i = 1, \dots, n.$$

4. Similarly, for the simultaneous execution of various operators: $\text{SfCprt}_{\mu_1^x}^{\tilde{F}\tilde{x}}$,

$$\text{where } \tilde{F}=(F_1|\mu_{\tilde{F}}(F_1), F_2|\mu_{\tilde{F}}(F_2)|, ..., F_n|\mu_{\tilde{F}}(F_n)), \quad F_i \text{ is an operator, } i = 1, \dots, n.$$

5. The arithmetic itself_g for capacities in themselves will be similar: addition - $SfC_1f_{\mu_1}\{\tilde{x}+\}\{g_1\}$, (or $SfC_3f_{\mu_1}\{\tilde{x}+\}\{g_1\}$ for the third type), multiplication $SfC_1f_{\mu_1}\{\tilde{x}*\}\{g_1\}$, ($SfC_3f_{\mu_1}\{\tilde{x}*\}\{g_1\}$).
6. Similarly with different operations: $SfC_1f_{\mu_1}\{\tilde{x}\tilde{q}\}\{g_1\}$, ($SfC_3f_{\mu_1}\{\tilde{x}\tilde{q}\}\{g_1\}$), and with different operators: $SfC_1f_{\mu_1}\{\tilde{F}\tilde{x}\}\{g_1\}$, ($SfC_3f_{\mu_1}\{\tilde{F}\tilde{x}\}\{g_1\}$).

7. $SfCrt_{\mu_1}^{g_1}$ – the result of the accommodation operator. For fuzzy sets A, B

we have

$$SfCrt_{\mu_1}^{g_1} = \left\{ \begin{matrix} A \\ A \cup B - A \cap B \end{matrix} \right\}^D, \text{ where } D \text{ is fuzzy self}_{g_1}\text{- (fuzzy set) for } A \cap B.$$

The measure:

$$\mu(SfCrt_{\mu_1}^{g_1}) = \left(\frac{\mu_{g_1}(A \cap B)}{\mu(A) + \mu(B) - \mu(A \cap B)} \right) * \mu(g_1) * \mu_1.$$

There is the same for structures if it's considered as sets. Our approach to the theory of hierarchical sets differs from the construction of hierarchical sets by Y.L. Ershov [4]-[6] : we construct completely different types of hierarchical sets.

8. $SfCprt$ -derivative of $g(x_1, x_2, \dots, x_n)$ is $SfCprt_{\mu_1}^{g_1}$, where
- $$\left\{ \frac{\partial}{\partial x_{1_i}}, \frac{\partial}{\partial x_{2_i}}, \dots, \frac{\partial}{\partial x_{k_i}} \right\}$$
- $\tilde{x}_0 = (x_{1_i} | \mu_{\tilde{x}}(x_{1_i}), x_{2_i} | \mu_{\tilde{x}}(x_{2_i}), \dots, x_{k_i} | \mu_{\tilde{x}}(x_{k_i}))$ - any fuzzy set from
- $\tilde{x} = (x_1 | \mu_{\tilde{x}}(x_1), x_2 | \mu_{\tilde{x}}(x_2), \dots, x_n | \mu_{\tilde{x}}(x_n))$. Let's designate $SfCprt - \frac{\partial^k g(x)}{\partial x_{1_i} \partial x_{2_i} \dots \partial x_{k_i}}$.
- $\{ \int () dx_{1_i}, \int () dx_{2_i}, \dots, \int () dx_{k_i} \}$
- $SfCprt$ -integral off $(x_1, x_2, \dots, x_n)$ is $SfCprt_{\mu_1}^{g_1}$, where
- $$g(x_1, x_2, \dots, x_n)$$
- $\tilde{x}_0 = (x_{1_i} | \mu_{\tilde{x}}(x_{1_i}), x_{2_i} | \mu_{\tilde{x}}(x_{2_i}), \dots, x_{k_i} | \mu_{\tilde{x}}(x_{k_i}))$ - any fuzzy set from
- $\tilde{x} = (x_1 | \mu_{\tilde{x}}(x_1), x_2 | \mu_{\tilde{x}}(x_2), \dots, x_n | \mu_{\tilde{x}}(x_n))$. Let's designate $SfCprt - \int \dots \int$
- $g(x) dx_{1_i} dx_{2_i} \dots dx_{k_i}$ - k-multiple integral. $SfCprt$ -lim off $(x_1, x_2, \dots, x_n)$ is

$\left\{ \lim_{x_{1_i} \rightarrow a_{1_i}}, \lim_{x_{2_i} \rightarrow a_{2_i}}, \dots, \lim_{x_{k_i} \rightarrow a_{k_i}} \right\}$
 SfCprt $\begin{matrix} g_1 \\ \mu_1 \end{matrix}$. Let's designate SfCprt-
 $f(x_1, x_2, \dots, x_n)$

$$\lim_{\substack{x_{1_i} \rightarrow a_{1_i} \\ \vdots \\ x_{k_i} \rightarrow a_{k_i}}} f(x_1, x_2, \dots, x_n) \cdot \text{self}_{g_1} \lim_{x \rightarrow a} = \text{SfCprt} \begin{matrix} \lim_{x \rightarrow a} \\ g_1 \\ \mu_1 \end{matrix}.$$

9. In the case of self_g -derivatives, inclusions of multiple derivatives are obtained. The same is true for self_g -integrals: we get inclusions of multiple integrals.

10. Let's denote self_{g_1} -(self_{g_1} -Q) through $\text{self}_{g_1}^2$ -Q, $f \text{S}_{g_1}(n, Q, g_1) = \text{self}_{g_1}$ -(self_{g_1} -(...(self_{g_1} -Q))) = $\text{self}_{g_1}^n$ -Q for n-multiple self_{g_1} .

Operator fitself_g .

Definition 2.1.7. An operator that transforms $\text{SfCprt} \begin{matrix} \{a\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$ into

any $\text{SfC}_i f\{b\} \mu_1 \{g_1\}$, $i = 2, 3$; where $\{b\} \subset \{a\}$; is the operator fitself_g .

Example. The operator contains the fuzzy set in itself $_g$.

Lim- fitself_g .

1. Lim SfCprt

For example, the double limit: $\lim_{\substack{x \rightarrow a_1 \\ y \rightarrow a_2}} G(x, y)$ corresponds to $\text{SCprt} \begin{matrix} \{G(x, y)\} \\ g_1 \\ \mu_1 \\ (a_1, a_2) \end{matrix}$.

Similarly, for SfCprt-lim with n variables.

In the case of lim-itself $_g$, for example, for m variables, it suffices to use the form (1.1) of lim SCprt for n variables ($n > m$). The same is true for integrals of variables m (for example, the double integral over a rectangular region is through the double limit).

The sequence of actions can be "collapsed" into an ordered SfCprt element, and then translate it, for example, into $\text{SfC}_3 f$ – the capacity in itself $_g$. Take the

receipt $\frac{\partial^2 u}{\partial x^2}$ as an example. Here is the sequence of steps 1) $\frac{\partial u}{\partial x} \rightarrow 2) \frac{\partial}{\partial x} \left(\frac{\partial u}{\partial x} \right)$.

"collapses" into an ordered SfCprt $\left\{ \frac{\partial u}{\partial x}, \frac{\partial}{\partial x} \left(\frac{\partial u}{\partial x} \right) \right\}_{g_1, \mu_1, x}$, which can be translated into the

corresponding SfC_1f . The differential operator SfCprt $\left\{ \frac{\partial u}{\partial x}, \frac{\partial}{\partial x} \left(\frac{\partial u}{\partial x} \right) \right\}_{g_1, \mu_1, x}$ - is interesting

too.

We can consider the concept of SfCprt - element as SfCprt $\frac{A}{g_1, \mu_1, B}$, where fuzzy A fits in

capacity B. Then $scprt_{g_1, \mu_1, B}$ it will mean $SfC_1fB\mu_1\{g_1\}$.

About SfCprt and SfC₃f programming.

The ideology of SCprt and SC₃f can be used for programming. Here are some of the SfCprt programming operators:

1. Simultaneous assignment of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1),$

$p_2|\mu_{\tilde{p}}(p_2)|, \dots, p_n|\mu_{\tilde{p}}(p_n)|$) to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|, \dots, x_n|\mu_{\tilde{x}}(x_n)|)$.

to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|, \dots, x_n|\mu_{\tilde{x}}(x_n)|)$. This is implemented

via SfCprt $\left\{ \tilde{x} := \{\tilde{p}\} \right\}_{g_1, \mu_1, x}$.

2. Simultaneous checking the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1),$

$g_2|\mu_{\tilde{g}}(g_2)|, \dots, g_n|\mu_{\tilde{g}}(g_n)|$) for the set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1),$

$B_2|\mu_{\tilde{B}}(B_2)|, \dots, x_n|\mu_{\tilde{B}}(B_n)|$). Implemented via SfCprt $IF \left\{ \{\tilde{B}\} \{\tilde{g}\} \right\} then \tilde{Q}_{g_1, \mu_1, x}$, where Q

can be anything.

3. Similarly for loop operators and others.

$SfrC_3f$ – software operators will differ only in that the aggregates $\{\tilde{x}\}, \{\tilde{p}\}, \{\tilde{B}\}, \{\tilde{g}\}$ will be formed from the corresponding SfCprt program operators in form (1.1) and

for more complex operators in the form from the forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

The OS (operating system), the computer's principles, and the modes of operation for this programming are interesting. But this is already the material for the following publications.

Using elements of the mathematics of SfCprt, we introduce the concept of SfCprt –

the change in physical quantity B: $\text{SfCprt} \begin{matrix} \{\Delta_1 B, \dots, \Delta_n B\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$. Then the mean SfCprt -

velocity will be $v_{\text{cpscfpr}}(t, \Delta t) = \text{SfCprt} \begin{matrix} \{\frac{\Delta_1 B}{\Delta t}, \dots, \frac{\Delta_n B}{\Delta t}\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$ and SfCprt-velocity at time

$t: v_{\text{sfct}} = \lim_{\Delta t \rightarrow 0} v_{\text{cpscfpr}}(t, \Delta t)$. SfCprt – acceleration: $a_{\text{sfct}} = \frac{dv_{\text{sfct}}}{dt}$.

When using SfCprt with "target weights", we get, depending on the "target weights", one or another modification, namely, for example, the velocity v_{sfct}^f (with a "target weight" f in the case when two velocities v_1, v_2 are involved

in the set $\{v_1 f, v_2\}$ for $v_{\text{sfct}}^f = \text{SfCprt} \begin{matrix} \{v_1 f, v_2\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$, f – instantaneous replacement we

get an instantaneous substitution v_1 by v_2 at point x of space at time t_0 with accommodation type g_1 and measure of fuzziness μ_1 .

Consider, in particular, some examples: 1) $\text{SfCprt} \begin{matrix} e \\ g_1 \\ \mu_1 \\ \{x_1, x_2\} \end{matrix}$ describes the presence of the same electron e at two different points x_1, x_2 . 2) The nuclei of atoms can be considered as SfCprt elements.

Similarly, the concepts of SfCprt - force and SfCprt - energy are introduced .

For example, $E_{\text{sfct}}^f = \text{SfCprt} \begin{matrix} \{E_1 f, E_2\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$ it would mean the instantaneous replacement

of energy E_1 by E_2 at time t_0 with accommodation type g_1 and measure of fuzziness

μ_1 . Two aspects of SfCprt–energy should be distinguished: 1) carrying out the desired "target weight" and 2) fixing the result of it. Do not confuse energy - SfCprt (the node of energies) with SfCprt – energy that generates the node of energies, usually with the "target weights." In the case of ordinary energies, the energy node is carried out automatically.

Remark 2.1.2. SfCprt – elements are all ordinary, but with "target weights," they become peculiar. Here you need the necessary energy to carry them out. As a rule, this energy is at the level of f_{self_g} . This is natural since it's much easier to manage elements of the k level via the elements of a more structured $k + 1$ level. Let us consider the concepts of capacities of physical objects in themselves. The question

arises about the $self_g$ -energy of the object. In particular, $SfCprt_{\mu_1}^{g_1 \frac{B}{B}}$ will mean

$SfC_1 f B \mu_1 \{g_1\}$. For example, $SfCprt_{\mu_1}^{g_1 \frac{DNA}{DNA}}$ allows you to reach the level of DNA

$self_g$ -energy, $SfCprt_{\mu_1}^{g_1 \frac{Q}{Q}}$ allows you to reach the level of $self_g$ -energy Q . The law of

$self_g$ -energy conservation operates already at the level of $self_g$ -energy. Also, in addition to capacities in themselves, you can consider the types of accommodation of oneself_g in oneself_g: the first type of the accommodation of oneself_g in oneself_g the second type of the accommodation of oneself_g in oneself: potentially, for example, in the form of programming oneself_g, the third type is partial accommodation of oneself_g in themselves_g—for example, self-operator, self-action, whirlwind. A container containing itself can be formed by $self_g$ -accommodation, i.e., accommodation in oneself_g. Let us clarify the concept of the term capacity in itself_g: it is a capacity containing itself_g f potentially. Consider $self_g$ - Q , where Q can be anything, including $Q=$ self; in particular, it can be any action. Therefore, $self_g$ - Q is when Q is made by itself_g; it makes itself_g. There is a partial $self_g$ - Q for

any Q with partial self_g -fulfillment. Let's consider several examples for capacities in themselves: ordinary lightning, electric arc discharge, and ball lightning.

A self_g -search of the solution of the equations $f_i(x)=0$, where $i=1,2,\dots,n$,
 $\{f_1(x)=0?x, f_2(x)=0?x, \dots, f_n(x)=0?x\}$
 $x=(x_1, x_2, \dots, x_n)$, will be realized in SfCprt $\begin{matrix} g_1 \\ \mu_1 \\ \alpha \end{matrix}$

$\{f_1(x)=0, f_2(x)=0, \dots, f_n(x)=0\}$
or SfCprt $\begin{matrix} g_1 \\ \mu_1 \\ ?x \end{matrix}$. x acquires more degree of liberty and

in this is direct decision. self_g -equation for x has its decision for x in direct kind.

$\{tasks(x)\}$
The same for SfCprt $\begin{matrix} g_1 \\ \mu_1 \\ ?x \end{matrix}$. self_g -task for x has its decision for x in direct

kind. self_g -question has its answer for x in direct kind. SfCprt $\begin{matrix} \{t\} \\ g_1 \\ \mu_1 \\ (o, x) \end{matrix}$, where $\{t\}$ -

time points set, (o, x) - object o in point x from space X , give to enter in necessary

$\{t\}$
time moments. The same for SfCprt $\begin{matrix} g_1 \\ \mu_1 \\ o \end{matrix}$.

$\{God - father, God - son, Holy Spirit\}$
 SfCprt $\begin{matrix} g_1 \\ \mu_1 \\ \alpha \end{matrix}$ is three-concept representation,

where α is a point in the connectedness space. SfCprt is also great for working

with structures, for example: 1) SfCprt $\begin{matrix} strA \\ g_1 \\ \mu_1 \\ B \end{matrix}$ - the structure A that fits into B with

accommodation type g_1 and measure of fuzziness μ_1 , where B can be any capacity,

another structure etc. 2) SfCprt $\begin{matrix} str_Q \\ g_1 \\ \mu_1 \\ R \end{matrix}$ - embedding structure from Q into R with

accommodation type g_1 . Similarly for displacement: 1) $\overset{B}{g_2}_{\mu_1} \text{SfCprt}$ - displacement str_A

of structure A from B with displacement type g_2 and measure of fuzziness μ_1 , 2)

$\overset{B}{g_2}_{\mu_1} \text{SfCprt}$ - displacement of the structure Q from B with displacement type g_2 and str_Q

measure of fuzziness μ_1 . To work with structures, you can introduce a special

operator CCprt: $\overset{A}{g_1}_{\mu_1} \text{CfCprt}$ structures B with the structure A with accommodation B

type g_1 and measure of fuzziness μ_1 , $\overset{B}{g_2}_{\mu_1} \text{CfCprt}$ destructors B from the structure A A

with displacement type g_2 and measure of fuzziness μ_1 .

Definition 2.1.8. A structure with a second degree of freedom will be called complete, i.e., "capable" of reversing itself_g concerning any of its elements explicitly, but not necessarily in known operators; it can form (create) new special operators (in particular, special functions).

In particular, $\overset{A}{g_1}_{\mu_1} \text{CfCprt}$, $\overset{A}{g_1}_{\mu_1} \text{CfCpr}$ are such structures. A A

Similarly, for working with models, each is structured by its structure; for example, use SfCprt-groups, SfCprt-rings, SfCprt-fields, SfCprt-spaces, self_g-groups, self_g-rings, self_g-fields, and self_g-spaces. Like any task, this is also a structure of the appropriate capacity.

self_g-H (self_g-hydrogen), like other self_g-particles, does not exist in the ordinary, but all self_g-molecules, self_g-atoms, and self-particles are elements of the energy space.

Remark 2.1.3. The concept of elements of physics SfCprt is introduced for energy space. The corresponding concept of elements of chemistry SfCprt is introduced

accordingly. Examples: 1) $\text{SfCprtE} \begin{matrix} \{a_1 q, a_2\} \\ g_1 \\ \mu_1 \\ D \end{matrix}$ – the energy of instantaneous

substitution a_1 by a_2 , where a_1 , and a_2 are chemical elements, q is instant replacement. Similarly, one can consider for the node of chemical reactions

$\{chemical\ elements\ with\ "target\ weights"\}$
 $\text{SfCprt} \begin{matrix} g_1 \\ \mu_1 \\ reaction \end{matrix}$. The ideology of SfCprt

elements allows us to go to the border of the world familiar to us, which allows us to act more effectively.

2.2 Dynamic SfCprt – elements

We considered stationary SfCprt – elements earlier. Here we consider dynamic SfCprt – elements.

Definition 2.2.1. The process of fitting a set of elements $\{a(t)\} = (a_1(t), a_2(t), \dots, a_n(t))$ into one point x of space X at time t with accommodation type g_1 and measure of fuzziness μ_1 will be called a dynamic SCprt – element. We

will denote $\text{sfCprt}(t) \begin{matrix} \{a(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$.

Definition 2.2.2. Fitting an ordered set of elements into one point in space with accommodation type g_1 and measure of fuzziness μ_1 is called a dynamic ordered SfCprt–element.

It is allowed to sum dynamic SCprt – elements:

$$\text{SfCprt}(t) \begin{matrix} \{a(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix} + \text{SfCprt}(t) \begin{matrix} \{b(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix} = \text{SfCprt}(t) \begin{matrix} \{a(t)\} \cup \{b(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix} .$$

Dynamic accommodation of foneself_g.

Definition 2.2.3. Dynamic SfCprt-capacity $\text{SfCprt}(t) \begin{matrix} R(t) \\ g_1 \\ \mu_1 \\ Q(t) \end{matrix}$ is the process of

embedding $R(t)$ into $Q(t)$ with accommodation type g_1 and measure of fuzziness

μ_1 .

Definition 2.2.4. The dynamic capacity $A(t)$ containing fitself_g as an element of the first type is the process of containing $A(t)$ in $A(t)$ with accommodation type g_1 and measure of fuzziness μ_1 . Denote $SfC_1f(t)_{\mu_1}A(t)\{g_1\}$.

Definition 2.2.5. Dynamic capacity $C(t)$ in itself_g of the second type is the process of containing elements from which it can be generated with accommodation type g_1 and measure of fuzziness μ_1 . Let's denote $SfC_2f(t)_{\mu_1}C(t)\{g_1\}$.

Definition 2.2.6. Dynamic partial capacity $B(t)$ in fitself_g of the third type is a process of partial accommodation of $B(t)$ in fitself_g with accommodation type g_1 and measure of fuzziness μ_1 or elements from which it can be generated with accommodation type g_1 and measure of fuzziness μ_1 partially or both at the same time. Denote $SfC_3f(t)_{\mu_1}B(t)\{g_1\}$.

All dynamic capacities in a dynamic fself_g -space are, by definition, dynamic capacities in themselves. Dynamic capacity fitself_g can manifest fitself_g as dynamic SfC_{prt} -capacity and ordinary dynamic capacity. In these cases, the usual measures and methods of topology are used.

Connection of dynamic SfC_{prt} – elements with dynamic accommodation of oneself_g with accommodation type g and measure of fuzziness μ_1 .

Consider third type of dynamic partial accommodation of foneself_g with accommodation type g and measure of fuzziness μ_1 . For example, based on

$\{a(t)\}$
 $SfC_{prt}(t) \begin{matrix} g_1 \\ \mu_1 \\ x \end{matrix}$, where $\{a(t)\} = (a_1(t), a_2(t), \dots, a_n(t))$, i.e. n – elements at one

point x , we can consider the dynamic capacity in itself_g $SfC_3f(t)$ with m elements from $\{a(t)\}$, $m < n$, which is process formed according to the form (1.1), that is,

$\{a(t)\}$
only m elements from $\{a(t)\}$ are in the structure $SfC_{prt}(t) \begin{matrix} g_1 \\ \mu_1 \\ x \end{matrix}$.

Dynamic accommodation of foneself_g of the third type can be formed for any other structure, not necessarily SfC_{prt} , only through the obligatory reduction in the

number of elements in the structure. In particular, using the form from the forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

It is possible to introduce structures more complex than $SfC_3f(t)$.

Dynamic math fitself_g.

1. The process of simultaneous addition of a set of elements $\{a(t)\} =$

$$(a_1(t), a_2(t), \dots, a_n(t)) \text{ are realized by } SfCprt(t) \begin{matrix} \{a(t) +\} \\ g_1 \\ \mu_1 \\ x \end{matrix}.$$

2. By analogy, for simultaneous multiplication: $SfCprt(t) \begin{matrix} \{a(t) *\} \\ g_1 \\ \mu_1 \\ x \end{matrix}.$

3. Similarly for simultaneous execution of various operations:

$$SfCprt(t) \begin{matrix} \{a(t)q(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix}, \text{ where } \{q(t)\} = (q_1(t), q_2(t), \dots, q_n(t)). \text{ } q_i(t)\text{-an}$$

operation, $i = 1, \dots, n$.

4. Similarly, for the simultaneous execution of various operators:

$$SfCprt(t) \begin{matrix} \{F(t)a(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix}, \text{ where } \{F(t)\} = (F_1(t), F_2(t), \dots, F_n(t)). \text{ } F_i(t) \text{ is an}$$

operator, $i = 1, \dots, n$.

5. Dynamic arithmetic fitself_g for accommodations of foneself_g will be similar:

dynamic addition - $SfC_1f(t)_{\mu_1}\{a(t) +\}\{g_1\}$, (or $SfC_3f(t)_{\mu_1}\{a(t) +\}\{g_1\}$ for the third type), dynamic multiplication $SfC_1f(t)_{\mu_1}\{a(t) *\}\{g_1\}$, $(SfC_3f(t)_{\mu_1}\{a(t) *\}\{g_1\})$.

6. Similarly with different operations: $SfC_1f(t)_{\mu_1}\{a(t)q(t)\}\{g_1\}$,

$(SfC_3f(t)_{\mu_1}\{a(t)q(t)\}\{g_1\})$ and with different operators:

$SfC_1f(t)_{\mu_1}\{F(t)a(t)\}\{g_1\}$, $(SfC_3f(t)_{\mu_1}\{F(t)a(t)\}\{g_1\})$.

$$7. SfCprt(t) \begin{matrix} A(t) \\ g_1 \\ \mu_1 \\ B(t) \end{matrix} \text{ gives the result } SfCrt(t) \begin{matrix} A(t) \\ g_1 \\ \mu_1 \\ B(t) \end{matrix} = \left\{ \begin{matrix} D(t) \\ A(t) \cup B(t) - A(t) \cap B(t) \end{matrix} \right\}$$

for sets $A(t), B(t)$, where $D(t)$ is fself_g-set for $A(t) \cap B(t)$.

$$\text{The measure: } \mu_{\mu_1}^{g_1} \left(\text{SfCrt}(t) \begin{matrix} A(t) \\ B(t) \end{matrix} \right) = \left(\mu^s(A(t) \cap B(t)) + \mu(A(t)) + \mu(B(t)) - \mu(A(t) \cap B(t)) \right) *$$

$$\mu(g_1) * \mu_1.$$

The same is true for structures if they are treated as sets.

Our approach to the theory of hierarchical sets differs from the construction of hierarchical sets by Y.L. Ershov [4]-[6] : we construct completely different types of hierarchical sets.

8. Similarly, for dynamic SfCprt-derivatives, dynamic SfCprt-integrals, dynamic SfCprt-lim, dynamic fself_g-derivatives, dynamic fself_g-integrals
9. Denote dynamic fself_g-(dynamic fself_g-Q(t)) through dynamic fself_g²-Q(t), fSf_g(t)(n,Q(t))= dynamic fself_g-(dynamic fself_g-(...(dynamic fself_g-Q(t)))) = dynamic fself_gⁿ-Q(t) for n-multiple dynamic fself_g.

Remark 2.2.1. The dynamic SfCprt-displacement of A(t) from B(t) with type of accommodation $g_2(t)$ and measure of fuzziness $\mu_2(t)$ will be denote by

$$\mu_{\mu_2(t)}^{g_2(t)} \text{SfCprt}(t). \text{ Then the notation } \begin{matrix} B(t) & C(t) & A(t) \\ g_2(t) & g_2(t) & g_1(t) \\ \mu_2(t) & \mu_2(t) & \mu_1(t) \end{matrix} \text{SfCprt}(t) \begin{matrix} A(t) \\ B(t) \end{matrix} \text{ is dynamic SfCprt-}$$

accommodation of A(t) in B(t) with type of accommodation $g_1(t)$ and measure of fuzziness $\mu_1(t)$ and dynamic SfCprt-displacement of D(t) from C(t) with type of accommodation $g_2(t)$ and measure of fuzziness $\mu_2(t)$ simultaneously.

We can consider the concept of dynamic SfCprt - element as $\text{SfCprt}(t) \begin{matrix} A(t) \\ g_1(t) \\ \mu_1(t) \\ B(t) \end{matrix}$, where

A(t) fits in dynamic capacity B(t) with type of accommodation $g_1(t)$ and measure

of fuzziness $\mu_1(t)$. Then $\text{SCprt}(t) \begin{matrix} B(t) \\ g_1(t) \\ \mu_1(t) \\ B(t) \end{matrix}$ will mean $\text{SC}_1\text{f}(t)B(t)\{g_1(t)\}$.

$\begin{matrix} A(t) \\ g(t) \\ \mu(t) \\ A(t) \end{matrix} \text{SCprt}(t)$ denotes the dynamic expelling A(t) foneself_{g(t)} out of foneself_{g(t)},

$A(t)$ $A(t)$
 $\frac{g(t)}{\mu(t)} \text{SCprt}(t) \frac{g(t)}{\mu(t)}$ is simultaneous dynamic accommodation $A(t)$ of $\text{foneself}_{g(t)}$ in
 $A(t)$ $A(t)$

$\text{foneself}_{g(t)}$ and dynamic expelling $A(t)$ $\text{foneself}_{g(t)}$ out of $\text{foneself}_{g(t)}$.

$A(t)$
 $\frac{g(t)}{\mu(t)} \text{SCprt}(t)$ will be called dynamic anti-capacity from $\text{foneself}_{g(t)}$. For example,
 $A(t)$

“white hole” in physics is such simple anti-capacity. The concepts of “white hole” and “black hole” were formulated by the physicists based on the subject of physics—the usual energies level. Mathematics allows you to deeply find and formulate the concept of singular points in the Universe based on the levels of more subtle energies. The experiments of the 2022 Nobel laureates Asle Ahlen, John Clauser, Anton Zeilinger correspond to the concept of the Universe as a capacity in itself_g. The energy of self_g-accommodation in itself_g is closed on itself_g [5].

Hypothesis: the accommodation of the galaxy in oneself_g as a spiral curl and the expelling out of oneself_g defines its existence. A self_g-capacity in itself_g as an element A is the god of A, the self_g-capacity in itself_g as an element the globe—the god of the globe, the self_g-capacity in itself_g as an element man-- the god of the man, the self_g-capacity in itself_g as an element of the universe-- the god of the universe, the accommodation of A into oneself_g is spirit of A, the accommodation of the Earth into oneself_g is spirit of Earth, the accommodation of the man into oneself_g is spirit of the man (soul), the accommodation of the universe into oneself_g is spirit of the universe. We may consider the following axiom: any capacity is the capacity of oneself_g. This is for each energy capacity.

About dynamic SfCprt and SfC₃f(t) programming.

The ideology of dynamic SfCprt and SfC₃f(t) can be used for programming:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t), p_2(t), \dots, p_n(t))$ to the variables $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$ is

$$\{\{g(t)\} := \{p(t)\}\}$$

implemented through $\text{SCprt}(t) \begin{matrix} w(t) \\ \mu(t) \\ x(t) \end{matrix}$.

2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f(t)_2, \dots, f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), \dots, B_n(t))$ is implemented through

$$\text{SfCprt}(t) \begin{matrix} w(t) \\ \mu(t) \\ x(t) \end{matrix} \text{IF}\{\{B(t)\}\{f(t)\}\} \text{ then } Q(t) \text{ .where } Q(t) \text{ can be any.}$$

3. Similarly for loop operators and others.

$\text{SfC}_3\text{f}(t)$ – software operators will differ only in that the aggregates

$\{\{g(t)\}, \{p(t)\}, \{B(t)\}, \{f(t)\}\}$ will be formed from corresponding processes

$\text{SfCprt}(t)$ for the above-mentioned programming operators through form (1.1) or the form from the forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Remark 2.2.2. With the help of dynamic SfCprt -elements, the concepts of dynamic SfCprt - force, dynamic SfCprt – energy are introduced. For example,

$$\{fE_1(t)f, fE_2(t)\}$$

$$fE(t)_{\text{sprt}}^f = \text{SfCprt}(t) \begin{matrix} g(t) \\ \mu(t) \\ x(t) \end{matrix} \text{ will mean the process of instantaneous}$$

replacement f of energy $fE_1(t)$ by $fE_2(t)$ at time t . Similarly, using $\text{SfC}_i f(t)$, the concepts of $\text{SfC}_i f(t)$ -force, $\text{SfC}_i f(t)$ -energy, $i=1,2,3$, and etc are introduced.

Remark 2.2.3. It is the accommodation of foneself_g in foneself_g that can “give birth” to the capacities in fitself_g – that is what fself_g -organization is.

Remark 2.2.4. $\text{SfCprt}(t) \frac{B(t)}{g(t) \mu(t)}$ can increase $\text{fself}_{g(t)}$ - level of $B(t)$.

Remark 2.2.5. For example, the operator fitself_g is $\text{SfC}_1\text{f}(t)$.

Remark 2.2.6. May be considered the following derivatives: $\frac{d\text{SfCprt}(t) \frac{A(t)}{g(t) \mu(t)}}{dt}$, $\frac{d \frac{B(t)}{g_2(t) \mu(t) \text{SfCprt}(t)} \frac{C(t)}{g_1(t) \mu(t) \text{SfCprt}(t)} \frac{A(t)}{B(t)}}{dt}$, $\frac{d\text{SfC}_i\text{f}(t)}{dt}$, $i=1,2,3$.

Remark 2.2.7. It is the accommodation of foneself_g in fitself_g as an element that can be interpreted as dynamic capacities in itself_g .

Remark 2.2.8. Not every capacity containing fitself_g as an element will manifest fitself_g as a sedentary capacity or capacity.

2.3 SfCprt – elements for continual fuzzy sets

Earlier, we considered finite-dimensional discrete SfCprt-elements and fself_g - capacities in fitself_g as an element. Here we believe some continual SfCprt-elements and continual fself_g -capacities in fthemselves as an element.

Definition 1.3.1. The set of continual elements $\tilde{a} = (a_1 | \mu_{\tilde{x}}(a_1), a_2 | \mu_{\tilde{x}}(a_2), \dots, a_n | \mu_{\tilde{x}}(a_n))$ fitting into point x of space X with accommodation type g_1 will be called continual SfCprt – element, and such a point in space will be called capacity of the continual SfCprt – element. We will denote $\text{scprt}_{g_1}^{\{\tilde{a}\}}_x$

Definition 2.3.2. An ordered set of continual elements at one point in space is called an ordered continual SfCprt–element.

$\{a\} \qquad \{b\}$
 It's allowed to sum continual SfCprt – elements: $\text{SfCprt}_{\mu_1}^{g_1} \quad \text{SfCprt}_{\mu_1}^{g_1} =$
 $x \qquad x$
 $\{a\} \cup \{b\}$
 $\text{SfCprt}_{\mu_1}^{g_1}$. where some or any elements may be ordered elements.

Definition 2.3.3. The continual fself_g -capacity A in itself_g as an element of the first type is the capacity fitting with accommodation type g_1 itself_g as an element and with measure of fuzziness μ_1 . Denote $SfC_1f_{\mu_1}A\{g_1\}$.

Definition 2.3.4. The ordered continual self_g -capacity A in itself_g as an element of the first type is the ordered capacity fitting itself_g as an element with accommodation type g_1 . Denote $\overrightarrow{SfC_1f_{\mu_1}A\{g_1\}}$.

For example, $SC_{\infty}^+ = \sin \infty |g$ is of this type. It denotes continual ordered self_g -capacities in itself_g as an element of following type—the range of simultaneous “activation” of numbers from $[-1,1]$ in mutual directions: $\uparrow I \downarrow_{-1}^1$. Also consider the following elements: $SC_{\infty}^- = \sin(-\infty) |g \rightarrow \downarrow I \uparrow_{-1}^1 |g$, $TC_{\infty}^+ = \text{tg} \infty |g \rightarrow \uparrow I \downarrow_{-\infty}^{\infty} |g$, $TC_{\infty}^- = \text{tg}(-\infty) |g \rightarrow \downarrow I \uparrow_{-\infty}^{\infty} |g$, don't confuse with values of these functions. Such elements can be summarized. For example: $aSC_{\infty}^+ + bSC_{\infty}^- = (a-b)SC_{\infty}^+ = (b - a) SC_{\infty}^-$.

Definition 2.3.5. The continual fself_g -capacity A in itself_g, as an element of the second type, is the capacity containing elements from which it can be generated. Let's denote $SfC_2f_{\mu_1}A\{g_1\}$.

An example of continual self_g -capacity in itself_g as an element of the second type is a living organism since it contains the programs: DNA and RNA.

Definition 2.3.6. Partial continual fself_g -capacity in itself_g as an element of the third type is called continual fself_g -capacity in itself_g as an element that partially

contains itself_g or contains elements from which it can be generated in part or both simultaneously. Denote $SfC_3f_{\mu_1}A\{g_1\}$.

All continual capacities in fself_g -space are continual fself_g -capacities in itself_g as an element by definition. The continual fself_g -capacities in itself_g as an element may appear as continual SfCpr- capacities and usual continual capacities. In these cases, there are used typical measure and topology methods.

The connection of continual SfCprt – elements with continual fself_g -capacities in themselves_g as an element.

Consider a third type of continual fself_g -capacity in itself_g as an element. For

example, based on $SfCprt \begin{smallmatrix} \{a\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix}$, where $\{a\} = (a_1, a_2, \dots, a_n)$, i.e. a_i - continual elements at one point, $i=1, 2, \dots, n$. The continual self_g -capacity in itself_g as an element with m continual elements from $\{a\}$, at $m < n$, can be considered as SfC_3f , which is formed by the form (1.1), i.e., only m continual elements are located in

the structures $SfCprt \begin{smallmatrix} \{a\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix}$. Continual fself_g -capacities in itself_g as an element of the

third type can be formed for any other structure, not necessarily SfCprt, only by obligatory reducing the number of continual elements in the structure. In particular, using the form from the forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Structures more complex than SfC₃f can be introduced.

Mathematics fitself_g for continual elements.

1. Simultaneous addition of the continual elements of the set $\{a\} =$

$(a_1, a_2, \dots, a_n)$ is implemented using $SfCprt \begin{smallmatrix} \{a \cup\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix}$.

2. By analogy, for simultaneous multiplication: $SfCprt \begin{smallmatrix} \{a \cap\} \\ g_1 \\ \mu_1 \\ x \end{smallmatrix}$.

3. Similarly, for simultaneous execution of various operations: $\text{SfCprt}_{\mu_1}^{\{aq\} \atop g_1 \atop x}$, where

$\{q\} = (q_1, q_2, \dots, q_n)$. q_i -an operation, $i = 1, \dots, n$.

4. Similarly, for the simultaneous execution of various operators: $\text{SfCprt}_{\mu_1}^{\{Fa\} \atop g_1 \atop x}$,

where $\{F\} = (F_1, F_2, \dots, F_n)$. F_i is an operator, $i = 1, \dots, n$.

5. For continual self_g -capacities in themselves_g as an element will be similar:
addition - $\text{SfC}_1 f_{\mu_1} \{a +\} \{g_1\}$., (or $\text{SfC}_3 f_{\mu_1} \{a +\} \{g_1\}$) for the third type),
multiplication $\text{SfC}_1 f_{\mu_1} \{a *\} \{g_1\}$, ($\text{SfC}_3 f_{\mu_1} \{a *\} \{g_1\}$).

6. Similarly with different operations: $\text{SfC}_1 f_{\mu_1} \{aq\} \{g_1\}$,

$(\text{SfC}_3 f_{\mu_1} \{aq\} \{g_1\})$ and with different operators:

$\text{SfC}_1 f_{\mu_1} \{Fa\} \{g_1\}$, $(\text{SfC}_3 f_{\mu_1} \{Fa\} \{g_1\})$.

7. $\text{SfCrt}_{\mu_1}^{\atop g \atop B}$ is the result of the accommodation operator $\text{SfCprt}_{\mu_1}^{\atop g \atop B}$. For sets A, B we

have

$$\text{SfCrt}_{\mu_1}^{\atop g \atop B} = \left\{ \begin{matrix} A \\ A \cup B - A \cap B \end{matrix} \right\}^D, \text{ where } D \text{ is } \text{fself}_g \text{-set for } A \cap B. \text{ The}$$

measure:

$$\mu(\text{SCpr}_{\mu_1}^{\atop g \atop B}) = \left(\frac{\mu^s(A \cap B)}{\mu(A) + \mu(B) - \mu(A \cap B)} \right) * \mu(g) * \mu_1.$$

There is the same for structures if it's considered as continual sets. Our approach to the theory of hierarchical sets differs from the construction of hierarchical sets by Y.L. Ershov [4]-[6]: we construct completely different types of hierarchical sets.

Remark 2.3.1. Three in one is

$$\text{SCprt} \left\{ \begin{array}{c} \infty \text{ in itself, an element that is not anyone's element, 0 out oneself} \\ g \\ \alpha \end{array} \right\}$$

, α - point space connectedness.

These elements are used for SfCprt-coding, SfCprt translation, coding fself_g, and translation fself_g for networks], which is suitable for electric current of ultrahigh frequency. More complex elements can be considered as continual sets of numbers with their " activation " in mutual directions. For example, ranges of function values, particularly those representing the shape of lightning. Differential geometry can be applied here. Also, n-dimensional elements can be considered. The space of such elements is Banach space if we introduce the usual norm for functions or vectors. We call this space-- SfCelb-space. Then we introduce the scalar product for functions or vectors and get the Hilbert space. We call this space SfCelh-space. In particular, one can try to describe some processes with these elements by differential equations and use methods from [7]. You can also try to optimize and research some processes with these elements using the techniques from [8]. Let's introduce operators for transforming capacity to fself_g -capacity in itself_g as an element: $Q_1\text{SfC}(A)$ transforms A to SfC_1fA , $Q_0\text{SfC}(A)$

transforms A to $\frac{A}{\mu_1} \text{SfCprt}$, $\text{SCO}(A)$ transforms A to $\uparrow A \downarrow$, $\uparrow A \downarrow$ -- ordered

self_g -capacity in itself_g as an element of simultaneous "activation" of all elements of A in mutual directions. For example, $\text{SCO}([-1,1]) = \text{SC}_{\infty}^+$, $\text{SCO}([1,-1]) = \text{SfC}_{\infty}^-$, $\text{SCO}([-\infty,\infty]) = \text{CT}_{\infty}^+$, $\text{SCO}([\infty,-\infty]) = \text{CT}_{\infty}^-$. The operator $(Q_1\text{SC}(A))^2$ increases self_g -level for A : it transforms self_g - $A = \text{SC}_1fA$ to self_g ^{2} - A , $(Q_1\text{SC}(A))^n \rightarrow \text{self}_g^n\text{-}A$, $e^{Q_1\text{SC}(A)} \rightarrow e^{\text{self}_g} - A$. Let us introduce the

following notations: $\frac{\{ \}}{A} \text{SCprt}$ by $\text{os}(\{ \} \rightarrow) \text{elf}_g$,

$$\begin{array}{c} A \\ g \end{array} \text{SCprt by } 2\text{oself}_g - A, \text{SCprt} \begin{array}{c} (A,A) \\ g \\ A \end{array} \text{ by } 2\text{self}_g - A, \text{SCprt} \begin{array}{c} A \\ g \\ (A,A) \end{array} \text{ by } 1/2\text{self}_g - A,$$

$$\text{SCprt} \begin{array}{c} A \\ g \\ q(A) \end{array} \text{ by } q\text{self}_g - A, \begin{array}{c} A \\ g \\ q(A) \end{array} \text{SCprt by } q() \text{oself}_g - A, \text{ q-any operator,}$$

$$\begin{array}{c} A \\ g \\ (q_1(A), \dots, q_N(A)) \end{array} \text{SCprt by } N\text{oself}_g - A, q_i = A, i = 1, \dots, N; \begin{array}{c} A \\ g \end{array} \text{SCprt} \begin{array}{c} A \\ g \\ A \end{array} \text{ by } (\text{self}_g -$$

$$\text{oself}_g) - A, \begin{array}{c} q_2(A) \\ g \\ q_3(A) \end{array} \text{SCprt} \begin{array}{c} A \\ g \\ q_1(A) \end{array} \text{ by } (q_1\text{self}_g - \begin{pmatrix} q_3 \circ \\ q_2 \circ \end{pmatrix} \text{oself}_g) - A, \begin{array}{c} A \\ g \\ (A,A) \end{array} \text{CCprt by by}$$

$$2\text{Coself}_g - A, \text{CCprt} \begin{array}{c} (A,A) \\ g \\ A \end{array} \text{ by } 2\text{Cself}_g - A, \text{CCprt} \begin{array}{c} A \\ g \\ (A,A) \end{array} \text{ by } 1/2\text{Cself}_g - A, \text{CCprt} \begin{array}{c} A \\ g \\ q(A) \end{array} \text{ by}$$

$$q\text{Cself}_g - A, \begin{array}{c} A \\ g \\ q(A) \end{array} \text{CCprt by } q() \text{Coself}_g - A, \text{ q-any operator, } \begin{array}{c} A \\ g \\ (q_1(A), \dots, q_N(A)) \end{array} \text{CCprt}$$

$$\text{by } N\text{Coself}_g - A, q_i = A, i = 1, \dots, N; \begin{array}{c} A \\ g \end{array} \text{CCprt} \begin{array}{c} A \\ g \\ A \end{array} \text{ by } \text{Cself}_g - A - \text{Coself}_g - A,$$

$$\begin{array}{c} q_2(A) \\ g \\ q_3(A) \end{array} \text{CCprt} \begin{array}{c} A \\ g \\ q_1(A) \end{array} \text{ by } q_1\text{Cself}_g - A - \begin{pmatrix} q_3 \circ \\ q_2 \circ \end{pmatrix} \text{Coself}_g - A, \text{SC2prt} \begin{array}{c} A \\ g \\ A \end{array} = (\text{self}_g - A, \text{self}_g - A),$$

$$\text{SCNprt} \begin{array}{c} A \\ g \\ A \end{array} = (q_1, \dots, q_N), q_i = \text{self}_g - A, i = 1, \dots, N. \text{self}_g (\text{SCprt} \begin{array}{c} A \\ g \\ B \end{array}) = \text{SCprt} \begin{array}{c} A \\ g \\ B \end{array} \begin{array}{c} A \\ g \\ B \end{array}. \text{Can be}$$

$$\text{considered } Q(\begin{array}{c} A \\ g \\ A \end{array} \text{SCprt} \begin{array}{c} A \\ g \\ A \end{array}), \text{ Q-any operator.}$$

2.4 Dynamic continual SfCprt – elements

Definition 2.4.1. The process of containing a set with continual elements $\{a(t)\} = (a_1(t), a_2(t), \dots, a_n(t))$ at one point x of the space X at time with accommodation type g_1 and with measure of fuzziness μ_1 will be called the dynamic continual

$\{a(t)\}$
 SCprt – element. We will denote $\text{SCprt}(t) \begin{array}{c} g_1 \\ \mu_1 \\ x \end{array}$. The set $\{a(t)\}$ may be fuzzy.

Definition 2.4.2. The process of containing an ordered set of continual elements at one point in space with accommodation type g_1 and with measure of fuzziness μ_1 is called dynamic continual ordered SfCprt – element.

It is allowed to sum dynamic continual SfCprt – elements:

$$\text{SfCprt}(t) \begin{matrix} g_1 \\ \mu_1 \\ x \end{matrix} + \text{SfCprt}(t) \begin{matrix} g_1 \\ \mu_1 \\ x \end{matrix} = \text{SfCprt}(t) \begin{matrix} \{a(t)\} \cup \{b(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix}.$$

Dynamic continual containing of oneself_g in oneself_g as an element.

Definition 2.4.3. The dynamic continual SfCprt-capacity $\text{SfCprt}(t) \begin{matrix} R(t) \\ g_1 \\ \mu_1 \end{matrix}$ is called the $Q(t)$

process of embedding $R(t)$ in $Q(t)$ with accommodation type g_1 .

Definition 2.4.4. The dynamic accommodation continual $A(t)$ of oneself_g of the first type is the process of putting $A(t)$ into itself_g. Denote $SfC_1f(t)\mu_1A(t)\{g_1\}$.

Definition 2.4.5. The dynamic accommodation continual $C(t)$ of oneself_g of the second type embedding contains the continual elements with accommodation type g_1 and with measure of fuzziness μ_1 from which it can be generated. Denote $SfC_2f(t)\mu_1C(t)\{g_1\}$.

Definition 2.4.6. The partial dynamic accommodation continual $B(t)$ of oneself_g of the third type is the process of partial embedding continual $B(t)$ into oneself_g or continual elements from which it can be generated in part with accommodation type g_1 and with measure of fuzziness μ_1 or both simultaneously. Denote $SfC_3f(t)\mu_1B(t)\{g_1\}$.

The connection of dynamic continual SfCprt – elements with dynamic accommodation of oneself_g in oneself_g as an element.

Let us consider the partial dynamic continual accommodation of oneself_g in

oneself_g as an element of the third type. For example, based on $\text{SfCprt}(t) \begin{matrix} \{a(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$,

where $\{a(t)\} = (a_1(t), a_2(t), \dots, a_n(t))$, i.e. n - continual elements at one point x ,

one can consider the dynamic accommodation $SfC_3f(t)$ of oneself_g in oneself_g as an element with m continual elements from $\{a(t)\}$, $m < n$, which is a process that is necessary form according to the form (1.1), i.e., only m continual elements from

$\{a(t)\}$ are located in the structure $SfCprt(t) \begin{matrix} g_1 \\ \mu_1 \\ x \end{matrix}$. Dynamic continual

accommodations of oneself_g in oneself_g as an element of the third type can be formed for any other structure, not necessarily $SfCprt$, only by necessarily reducing the number of continual elements in the structure. In particular, with the help of the form from the forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

. It is possible to introduce structures more complex than $SC_3f(t)$.

Dynamic continual mathematics self_g.

1. The process of simultaneous addition of the set of continual elements

$\{a(t)\} = (a_1(t), a_2(t), \dots, a_n(t))$ is realized by $SfCprt(t) \begin{matrix} \{a(t) \cup\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$.

2. By analogy, for simultaneous multiplication: $SCprt(t) \begin{matrix} \{a(t) \cap\} \\ g_1 \\ x \end{matrix}$.

3. Similarly for simultaneous execution of various operations:

$SfCprt(t) \begin{matrix} \{a(t)q(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$, where $\{q(t)\} = (q_1(t), q_2(t), \dots, q_n(t))$. $q_i(t)$ -an

operation, $i = 1, \dots, n$.

4. Similarly, for the simultaneous execution of various operators:

$SfCprt(t) \begin{matrix} \{F(t)a(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$, where $\{F(t)\} = (F_1(t), F_2(t), \dots, F_n(t))$. $F_i(t)$ is an

operator, $i = 1, \dots, n$.

5. The dynamic arithmetic self_g for the dynamic continual accommodations of oneself_g will be similar: dynamic addition $-SfC_1f(t)_{\mu_1}\{a(t) \cup\}\{g_1\}$,

(or $SC_3f(t)_{\mu_1}\{a(t) \cup\}\{g_1\}$ for the third type), dynamic multiplication $SC_1f(t)_{\mu_1}\{a(t) \cap\}\{g_1\}$, $(SC_3f(t)_{\mu_1}\{a(t) \cap\}\{g_1\})$.

6. Similarly with different operations: $SfC_1f(t)_{\mu_1}\{a(t)q(t)\}\{g_1\}$, $(SfC_3f(t)_{\mu_1}\{a(t)q(t)\}\{g_1\})$ and with different operators: $SfC_1f(t)_{\mu_1}\{F(t)a(t)\}\{g_1\}$, $(SfC_3f(t)_{\mu_1}\{F(t)a(t)\}\{g_1\})$.

7. $SfCprt(t) \begin{matrix} A(t) \\ g_1 \\ \mu_1 \\ B(t) \end{matrix}$ gives the result $SfCrt(t) \begin{matrix} A(t) \\ g_1 \\ \mu_1 \\ B(t) \end{matrix} = \left\{ \begin{matrix} D(t) \\ A(t) \cup B(t) - A(t) \cap B(t) \end{matrix} \right\}$ for continual sets $A(t), B(t)$, where $D(t)$ is self_g-set for $A(t) \cap B(t)$.

The measure:
$$\mu(SfCrt(t) \begin{matrix} A(t) \\ g_1 \\ \mu_1 \\ B(t) \end{matrix}) = \left(\begin{matrix} \mu^s(A(t) \cap B(t)) \\ \mu(A(t)) + \mu(B(t)) - \mu(A(t) \cap B(t)) \end{matrix} \right) * \mu(g_1) * \mu_1.$$

There is the same for structures if it's considered as continual sets. Our approach to the theory of hierarchical sets differs from the construction of hierarchical sets by Y.L. Ershov [4]-[6] : we construct completely different types of hierarchical sets.

8. Similarly, for dynamic SfCprt-derivatives, dynamic SfCprt-integrals, dynamic SfCprt-lim, dynamic fself_g-derivatives, dynamic fself_g-integrals
9. Denote dynamic continual fself_g -(dynamic continual fself_g -Q(t)) through dynamic continual fself_g²-Q(t), fSfC(t)(n,Q(t))= dynamic continual fself_g -(dynamic continual fself_g -(...(dynamic continual fself_g -Q(t)))) = dynamic continual fself_gⁿ-Q(t) for n-multiple dynamic continual fself_g.

Remark 2.4.1. Dynamic continual SfCprt-displacement of $A(t)$ from $B(t)$ with type of accommodation $g_2(t)$ and measure of fuzziness μ_2 will be denote through

$B(t) \begin{matrix} C(t) \\ g_2(t) \\ \mu_2 \\ A(t) \end{matrix} SfCprt(t)$. Then the notation $\begin{matrix} A(t) \\ g_1(t) \\ \mu_1 \\ B(t) \end{matrix} SfCprt(t) \begin{matrix} C(t) \\ g_2(t) \\ \mu_2 \\ A(t) \end{matrix}$ is dynamic continual SfCprt-

embedding of $A(t)$ in $B(t)$ with type of accommodation $g_1(t)$ and measure of

fuzziness μ_1 and dynamic continual SfCprt-displacement of $D(t)$ from $C(t)$ with type of accommodation $g_2(t)$ and measure of fuzziness μ_2 simultaneously..

We can consider the concept of dynamic continual SfCprt - element as

$\text{sfcprt}(t) \begin{matrix} A(t) \\ g_1(t) \\ \mu \\ B(t) \end{matrix}$, where $A(t)$ fits in dynamic continual capacity $B(t)$. Then $\text{sfcprt}(t) \begin{matrix} B(t) \\ g_1(t) \\ \mu \\ B(t) \end{matrix}$

it will mean $\text{SfC}_1 f(t) \begin{matrix} A(t) \\ g_2(t) \\ \mu \\ A(t) \end{matrix} B(t)$. $\text{sfcprt}(t) \begin{matrix} A(t) \\ g_2(t) \\ \mu \\ A(t) \end{matrix}$ denotes the dynamic continual displacement

of $A(t)$ from itself_g, $\begin{matrix} A(t) \\ g_2(t) \\ \mu_2 \\ A(t) \end{matrix} \text{sfcprt}(t) \begin{matrix} A(t) \\ g_1(t) \\ \mu_1 \\ A(t) \end{matrix}$ —simultaneous dynamic continual

accommodation of oneself_g $A(t)$ in oneself_g $A(t)$ and dynamic continual expelling

oneself_g $A(t)$ out of oneself_g $A(t)$. $\begin{matrix} A(t) \\ g_2(t) \\ \mu_2 \\ A(t) \end{matrix} \text{sfcprt}(t)$ will be called dynamic continual anti

capacity from itself_g.

Connection of dynamic continual SfCprt – elements with target weights with dynamic continual accommodation of oneself_g with target weights.

Consider a third type of dynamic partial accommodation of oneself_g with target

weights $g(t)$. For example, based on $\text{SCprt}(t) \begin{matrix} \{a(t)g(t)\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$, where $\{a(t)\} =$

$(a_1(t), a_2(t), \dots, a_n(t))$, i.e. n - continual elements with target weights $\{g(t)\}$ at one point x , we can consider the dynamic accommodation $\text{SfC}_3 f(t)g(t)$ of oneself_g with target weights with m continual elements with target weights $\{g(t)\}$ from $\{a(t)\}$, $m < n$, which is the process of formation according to the form (1.1), i.e., only m continual elements with target weights $\{g(t)\}$ from $\{a(t)\}$ are located in the structure $\text{SfC}_3 f(t)g(t)$. Dynamic accommodations of oneself_g with target weights of the third type can be formed for any other structure, not necessarily Sit, only by reducing the number of continual elements with target weights in the

structure. In particular, using the form from forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*). Structures more complex than $SfC_3f(t)g(t)$ can be introduced.

Definition 2.4.8. The dynamic embedding of continual $A(t)$ into itself_g with target weights $\{g(t)\}$ of the first type is the process of embedding $A(t)$ into $A(t)$ with target weights. Denote $SfC_1f(t)A(t)g(t)$.

Definition 30. The dynamic accommodation of continual $C(t)$ itself_g into itself_g with target weights $\{g(t)\}$ of the second type is the process of accommodation of the continual elements from which it can be generated. Let's denote $SfC_2f(t)C(t)g(t)$.

Definition 2.4.9. Partial dynamic accommodation of continual $B(t)$ itself_g into itself_g with target weights $\{g(t)\}$ of the third type is the process of partial accommodation of continual $B(t)$ into itself_g or continual elements from which it can be generated partially, or both at the same time. Denote $SfC_3f(t)B(t)g(t)$.

2.5 The usage of SfCprt-elements for networks

A. Galushkin's comprehensive monograph [9] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new mathematical process with accommodation operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators. Accommodation operators are more convenient for networks. Also, the main emphasis was placed on using processors operating using triodes, which are generally not used in SfCprt-networks. SfCprt-networks ($SmnSfCprt$) are a SfCprt-structure that can be built for the required weights. SfCprt-OS (SfCprt operating system) uses SfCprt-coding and SfCprt-translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b) , where the number b is the code of the action, and the number a is the code of the object of this action. SfCprt-coding (or $fself_g$ -coding) is implemented through a

matrix consisting of 2 columns (in the continuous case, two intervals of numbers). Here, the source encoding is used for all matrix rows simultaneously. SfCprt-translation is carried out by inversion. In this case, f_{self_g} -coding and f_{self_g} -

translation will be more stable. The target weights f_i in $SfCprt \frac{g_1}{\mu_1}$ are chosen for

necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [9]. We will touch on the difference of this only for hierarchical complex networks. The same simple executing programs are in the cores of simple artificial neurons of type SfCprt (designation - mnSfCprt) for simple information processing. More complex executing programs are used for mnSfCprt nodes. SfCprt-threshold element –

$\{ax\}$
 $sgn(SCprt \frac{g_1}{\mu_1})$, b- mnSCprt, $x=(x_1, x_2, \dots, x_n)$ – source signals values, $a=(a_1, a_2, \dots, a_n)$ –

SCprt-synapses weights. The first level of mnSfCprt consists of simple mnSfCprt.

The second level of mnSfCprt consists of $SfCprt \frac{g_1}{\mu_1}$ – SfCprt-node of

mnSfCprt in range D, D- capacity for mnSfCprt node. The third level of mnSfCprt

$\{mnSfCprt\}$
 consists of $SfCprt \frac{g_1}{\mu_1}$ - SCprt²- node of mnSfCprt in range D, thus D

becomes capacity of itself_g in itself_g as an element for mnSfCprt. For our networks, it is sufficient to use SfCprt²- nodes of mnSfCprt, but f_{self_g} -level is higher in living organisms, particularly SfCprtⁿ-, $n \geq 3$. The target structure or the corresponding program enters the target unit using a short-pulse laser to generate attosecond pulses of light. After that, all networks or parts of them are activated according to the indicative goal. It may appear that we are leaving the network

ideology, but these networks are a complex hierarchy of different levels, like living organisms.

Remark 2.5.1 Traditional scientific approaches through classical mathematics make it possible to describe only at the usual energy level. Here we consider an approach that makes describing processes with finer energies possible. mnSfCprt

$\{\text{ceprograms}_g\}$
 contains $\text{SfCprt} \quad \mu_1^g, \text{cfeprogram}_g$ –executing program in SfCprt - OS.
 mnSCprt

SCprt-OS (or fself_g -OS) is based on SfCprt -assembly language (or fself_g -assembly language), which is based on assembly language through SfCprt -approach in turn, if the base of elements of SfCprt -networks is sufficient. The ceprograms_g are in SfCprt -programming environments (or fself_g -programming environments), but this question and SfCprt -networks base will be considered in the following publications. In particular, ceprograms_g may contain SfCprt -programming operators. In mnSfCprt cores, the constant memory SfCprt with correspondent cfeprogram_g depending on mnSfCprt .

The OS (operating system) and the principles and modes of operation of the SfCprt -networks for this programming are interesting. But this is already the material for the next publications.

Here is developed a helicopter model without a main and tail rotors based on SfCprt – physics and special neural networks with artificial neurons operating in normal and SfCprt -modes. Let's denote this model through SmnSfCprt . To do this, it's proposed to use mnSfCprt of different levels; for example, for the usual mode, mnSfCprt serves for the initial processing of signals and the transfer of information to the second level, etc., to the nodal center, then checked. In case of an anomaly - local SfCprt -mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, SmnSfCprt is activated with the desired "target weight." Here are realized other tasks also. To reach the

SmnSfCprt

fself_g -energy level, the mode SfCprt $\frac{g_1}{\mu_1}$ is used. In normal mode, it's

SmnSfCprt

planned to carry out the movement of SmnSfCprt on jet propulsion by converting the energy of the emitted gases into a vortex to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the SmnSfCprt for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is drainage of exhaust gases outside the SmnSfCprt. SmnSfCprt is represented by a neural network that extends from the center of one of the main clusters of SfCprt - artificial neurons to the shell, turning into the body itself_g. Above the operator's cabin is the central core of the neural network and the target block, responsible for performing the "target weights" and auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for SmnSfCprt's actions is below the operator's cab. In SfCprt – mode, the entire network or its sections are SfCprt – activated to perform specific tasks, in particular, with "target weights." In the target, block used SfCprt -coding, SfCprt-translation for activation of all networks to "target weights" simultaneously, then – the reset of this SfCprt-coding after activation. Unfortunately, triodes are not suitable for SfCprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for cfeprogram_s_g may be used instead triodes since there is no necessity to unbend the alternating current to direct. The SfCprt-operative memory belt is disposed around a central core of SmnSfCprt. There are SfCprt-coding, SfCprt-translation, and SfCprt-realize of feprogram_s_g and the programs from the archives without extraction, SCprt-coding and SCprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. SfCprt – structure or an cfeprogram_g if one is present of needed «target weight» are taken in target block at

$\text{SmnSfCprt}, f$
 $\text{SfCprt} - \text{activation of the networks. SfCprt} \quad \frac{g}{\mu_1} \quad \text{derives SmnSfCprt to the}$
activation

fself_g -level boundary with target weight f . It's used ultra-short optical pulses laser or an alternating current of above high frequency and ultra-violet light, which can work with SfCprt – structures in SfCprt–modes by its nature to activate the networks or some of its parts in SfCprt–modes and locally using SfCprt–mode. Above high frequently alternating current go through mercury bearers. That's why overheating does not occur. The power of the alternating current above high frequently increases considerably for the target block. The activation of all networks is realized to indicate “target weights.”

2.6 Variable hierarchical dynamical structures (models) for dynamic, singular, hierarchical sets

In contrast to the classical one-attribute set theory, where only its contents are taken as a set, we consider a two-attribute set theory with a set as a capacity and separately with its contents. We simply use a convenient form to represent the singularity of a set. Articles [10-19] use the following methodology for permanent structures:

1. Cancellation of the axiom of regularity
2. 2 attributes for the set: capacity and its content
3. Compression of a set, for example, to a point
4. “turning out” from one another, particularly from a capacity, we pull out another capacity, for example, itself_g , as its element.
5. The simultaneity of one (compression) and the other (“eversion”)
6. Own capacities
7. Qualitatively new programming and Networks.

Here we will consider variable structures (models), both discrete and continuous:

a) with variable connections, b) with the variable backbone for links, c)

generalized version; in particular, in variable structures (models), for example,

$$\begin{matrix} C \\ g_2 \\ \mu_2 \\ D \end{matrix} \text{SfCprt}(t) \begin{matrix} A \\ g_1 \\ \mu_1 \\ B \end{matrix} = \left\{ \begin{array}{l} \begin{matrix} C \\ g_2 \\ \mu_2 \\ D \end{matrix} \text{SfCprt}, q_2 \geq t \geq q_1 \\ \begin{matrix} C & A \\ g_2 & g_1 \\ \mu_2 & \mu_1 \end{matrix} \text{SfC}^1\text{prt}, q_3 \geq t > q_2 \\ \begin{matrix} D & B \\ B & A \\ g_2 & g_1 \\ \mu_2 & \mu_1 \end{matrix} \text{SfCprt}, q_4 \geq t > q_3 \quad (*_{2.6.1}), \\ \begin{matrix} A \\ g_1 \\ \mu_1 \\ B \end{matrix} \text{SfCprt}, q_5 \geq t > q_4 \\ \{\} \\ g_2 \text{SfCprt}, t > q_5 \\ D \\ \dots \end{array} \right.$$

$\begin{matrix} C & A \\ g_2 & g_1 \\ \mu_2 & \mu_1 \\ D & B \end{matrix} \text{SfC}^1\text{prt}$ is the analogue, considered $\begin{matrix} C & A \\ D & B \end{matrix} \text{S}^1\text{prt}$ in [10]. In particular, $\begin{matrix} B & A \\ g_2 & g_1 \\ \mu_2 & \mu_1 \\ D & B \end{matrix} \text{SfCprt}$

can be interpreted as a game: player 1 fits A into B, and the other pushes D out of B at the same time.

Can be considered N-hierarchical structure: 1-level - elements; level 2 - connections between them, level 3 - relationships between elements of level 2, etc. up to level N+1. N-hierarchical structure: 1-level - A; 2-level -B, 3-level - C, etc. up to (N+!)- level, where A, B, C, ... can be any in particular, by actions, sets, and others.

$$\begin{matrix} C & A \\ g_2 & g_1 \\ \mu_2 & \mu_1 \\ D & B \end{matrix} \text{SfCprt} : \left\langle \begin{matrix} A \rightarrow B \\ A, B \end{matrix} \middle| \begin{matrix} D \leftarrow C \\ C, D \end{matrix} \right\rangle \rightarrow \left(\begin{matrix} fself_{g_1}(A \rightarrow B) \\ A, B \end{matrix} \right)$$

$$\begin{matrix} C & A \\ g_2 & g_1 \\ \mu_2 & \mu_1 \\ D & B \end{matrix} \text{SfCprt} : \left\langle \begin{matrix} A \rightarrow B \\ A, B \end{matrix} \middle| \begin{matrix} D \leftarrow C \\ C, D \end{matrix} \right\rangle \rightarrow \left(\begin{matrix} foself_{g_2}(D \leftarrow C) \\ C, D \end{matrix} \right)$$

Can be considered discrete hierarchical structure, continuous hierarchical structure, and discrete-continuous hierarchical structure,

N – hierarchical structure

$$Sf_{Cprt} \begin{matrix} g \\ \mu_1 \\ x \end{matrix} .$$

The example

$$\begin{matrix} & & & N\text{--level of hierarchical structure} \\ Sf_{Cprt} & & & \begin{matrix} g \\ \mu_1 \\ x \end{matrix} \\ & & \dots & \\ \left[\begin{matrix} Sf_{Cprt} & & & i\text{--level of hierarchical structure} \\ & g \\ & \mu_1 \\ & x \end{matrix} \right] & & \dots & \\ & & \dots & \\ Sf_{Cprt} & & & 1\text{--level of hierarchical structure} \\ & & & \begin{matrix} g \\ \mu_1 \\ x \end{matrix} \end{matrix}$$

QHfSprg=HfSprt_x -- N-hierarchical structure

compression with type of accommodation g into point x .

Let $f(N, QHfSprg) = QHfSprg \left. \begin{matrix} QHfSprg & QHfSprg & \dots & QHfSprg \end{matrix} \right\} \text{--N levels}$

It can be considered $f_{self_g} - QHfSprg, f(y, QHfSprg)$ for any $y, f(QHfSprg, QHfSprg)$.

Compression Hierarchy Examples:

$$1) Sf_{Cprt} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ Sf_{Cprt} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} + B$$

Ordinary types of energy are manifestations of a lower level from these structures.

If we represent an amorphous body with a mathematical structure of self_g -object

$$\begin{array}{c} A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \end{array}, \text{ where } \begin{array}{c} A_0 \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \end{array} \text{ - level of objectivity of an amorphous object,}$$

$$\begin{array}{c} A_0 + E_s \\ A_0 \end{array}$$

$$\begin{array}{c} A_0 \\ \text{(SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \end{array} + \begin{array}{c} A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \end{array}) \text{ - the energy of connections between the level of}$$

$$\begin{array}{c} A_0 + E_s \\ A_0 \end{array}$$

$$\begin{array}{c} E_s \\ \text{subtle energy SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \end{array} \text{ and the level of objectivity .}$$

$$\begin{array}{c} E_s \\ E_s \end{array}$$

Thus, one can try to conventionally represent the mathematical model of the energy structure of an amorphous object as a hierarchical dynamic operator

$$\begin{array}{c} E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ E_s \end{array}$$

$$\begin{array}{c} A_0 \\ \text{(SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \end{array} + \begin{array}{c} A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \end{array}) \text{ (2.7.1).}$$

$$\begin{array}{c} A_0 + E_s \\ A_0 \end{array}$$

$$\begin{array}{c} A_0 \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 \end{array}$$

In particular, the magnetic field and spin belong to the second level in (2.7.1) .

The next level of objectivity responds to a crystal. We represent a crystal with a mathematical structure

$$\begin{array}{c}
A_0 + E_s \\
\text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
A_0 + E_s \\
\text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \quad (2.7.2). \\
A_0 + E_s \\
\text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
A_0 + E_s
\end{array}$$

Thus, one can try to conventionally represent the mathematical model of the energy structure of a crystal as a hierarchical dynamic operator (2.7.2). The next level of objectivity responds to a living crystal, for example, the bone of a living organism, a nail, viruses, DNA, RNA and etc. When there is no nutrient medium

and energy, it behaves like a crystal:

$$\begin{array}{c}
\{\} \quad A_0 + E_s \\
\text{SfCprt} \{\} \quad \text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
\{\} \quad A_0 + E_s \\
\{\} \quad \text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \quad , \text{ a nutrient} \\
\{\} \quad A_0 + E_s \\
\text{SfCprt} \{\} \quad \text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
\{\} \quad A_0 + E_s
\end{array}$$

medium appears and the necessary energy:

$$\begin{array}{c}
B_0 + E_q \quad B_0 + E_q \\
\text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \quad \text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
B_0 + E_q \quad B_0 + E_q \\
\begin{array}{c} g_1 \\ \mu_1 \end{array} \quad \text{SCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \quad , \text{ its} \\
B_0 + E_q \quad B_0 + E_q \\
\text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \quad \text{SfCprt} \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
B_0 + E_q \quad B_0 + E_q
\end{array}$$

structure is transformed into a mathematical structure

$$\begin{array}{ccc}
\begin{array}{c} B_0 + E_q \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ B_0 + E_q \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ B_0 + E_q \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ B_0 + E_q \end{array} & \begin{array}{c} \text{SfCprt} \\ \text{SfCprt} \\ \text{SfCprt} \\ \text{SfCprt} \end{array} & \begin{array}{c} A_0 + E_s + B_0 + E_q \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s + B_0 + E_q \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s + B_0 + E_q \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s + B_0 + E_q \end{array}
\end{array}
\quad . \text{ The division of DNA into two DNAs}$$

after sufficient accumulation of bases and energy - this minimal division into only two duplicates corresponds to the law of conservation of living energy and minimization of the entropy of the system.

Next comes the level of living organisms:

$$\begin{array}{cccc}
\begin{array}{c} A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \end{array} & \begin{array}{c} \text{SfCprt} \\ \text{SfCprt} \\ \text{SfCprt} \\ \text{SfCprt} \end{array} & \begin{array}{c} A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \end{array} & \begin{array}{c} A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \end{array} \\
\begin{array}{c} \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \end{array} & \begin{array}{c} \text{SfCprt} \\ \text{SfCprt} \\ \text{SfCprt} \\ \text{SfCprt} \end{array} & \begin{array}{c} A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \end{array} & \begin{array}{c} A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \end{array} \\
\begin{array}{c} \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \end{array} & \begin{array}{c} \text{SfCprt} \\ \text{SfCprt} \\ \text{SfCprt} \\ \text{SfCprt} \end{array} & \begin{array}{c} A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \end{array} & \begin{array}{c} A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \\ \text{SfCprt } \begin{array}{c} g_1 \\ \mu_1 \end{array} \\ A_0 + E_s \end{array}
\end{array}
\quad .$$

Next comes the level of Globe, where the role of living cells (molecules in the case of a crystal) is played by living organisms. Next comes the level of Universe, where the role of living cells (molecules in the case of a crystal) is played by planets inhabited by living beings. You can try to represent these levels through more complex mathematical models, there are options for going beyond the level

of objectivity for objects with energy structures of a sufficiently high level, but this is already material for subsequent publications. Our object world is an interpretation of the manifestation of only one set of subtle energy fibers out of their countless number.

$\mu_2^{g_2 \text{ SfCprt}} \frac{C}{C}$ will be called dynamic anti-capacity from oneself_g. For example, “white hole” in physics is such simple anti-capacity. The concepts of “white hole” and “black hole” were formulated by the physicists based on the subject of physics –the usual energies level. Mathematics allows you to deeply find and formulate the concept of singular points in the Universe based on the levels of more subtle energies. The experiments of the 2022 Nobel laureates Asle Ahlen, John Clauser, Anton Zeilinger and the experiments in chemistry Nazhipa Valitov correspond to the concept of the Universe as a capacity in itself_g as the element. They experimented with connections for elements of the microworld, and since here the connections are self_g -connections, then when the object component of self_g -connections is removed, its higher level remains, which was manifested in their experiments. The electron spin belongs to the second level - above the level of objectivity. The energy of self_g -accommodation in itself_g is closed on itself_g.

Remark 2.7.1. From the point of view of our theory of dynamic operators and sets, we can interpret the energy effect of a thermonuclear reaction as the result of the “collapse” of two self_g -objects: for example, 1) ${}^3_2\text{He}$, ${}^3_2\text{He}$ and the formation of one self_g -object ${}^4_2\text{He}$, 2) ${}^3_2\text{He}$, ${}^2_1\text{H}$ and the formation of one self_g -object ${}^4_2\text{He}$. As a result, the energy of the collapse of the lost part of the self_g is released.

Remark 2.7.2. To gain access to object transformation, just go to the level

$$lS = \frac{2}{\pi} \arctg(1 + \varepsilon), \varepsilon \text{ may be quite small.}$$

Examples of transformation:

$$\begin{array}{l}
1) \quad \text{SfCprt}_{\mu_1}^{g_1, q} \rightarrow \text{SfCprt}_{\mu_1}^{g_1, q} \rightarrow \text{SfCprt}_{\mu_1}^{g_1, b} \\
\quad \quad \quad \text{SfCprt}_{\mu_1}^{g_1, q} \quad \quad \quad \text{SfCprt}_{\mu_1}^{g_1, c} \\
2) \quad \text{SfCprt}_{\mu_1}^{g, q} \rightarrow \text{SfC}_3\text{f}(\text{self}_g(q)) \rightarrow \text{SfCprt}_{\mu_1}^{g, r}
\end{array}$$

This is a rather conditional interpretation, because in fact, the IS of the “vessel” (energy cocoon) of the object may turn out to be greater than $\frac{2}{\pi} \arctg(1)$. This is taken for initiation: we build a theory of this, starting from this stage of interpretation. After experiments, the next stage may begin.

$\text{fself}_g A = \text{SfCprt}_{\mu_1}^{g, A}$ can be transformed into any D if $\mu_l_g(D) = \mu_l_g(\text{SfCprt}_{\mu_1}^{g, A})$, $\mu_l_g(x)$

- level measure of fself_g for x, in particular, into $\text{SfCprt}_{\mu_1}^{g, \text{any } C}$ or $\text{SfCprt}_{\mu_1}^{g, A}$, and

also an object R into any object Q or any energy U. The transformations of this type will be called sfttransformations. $\text{self}_g^N A$ can transform itself_g into any D if $N \geq 2$; to realize this we need an even larger quantity N.

Example of a parallel-serial program statement

$$\begin{array}{c}
\begin{array}{c}
\text{SfCprt} \\
\text{if } \{p\} \text{SfCprt}
\end{array}
\begin{array}{c}
s := \text{SfCprt} \\
\text{SfCprt}
\end{array}
\begin{array}{c}
\begin{array}{c}
C \\
g \\
\mu_1
\end{array} \\
\begin{array}{c}
Q \\
\text{SfCprt} \\
g \\
\mu_1 \\
A
\end{array} \\
\begin{array}{c}
g \\
\mu_1 \\
A
\end{array} \\
\begin{array}{c}
g \\
\mu_1
\end{array}
\end{array}
\begin{array}{c}
af := \\
\text{SfCprt}
\end{array}
\begin{array}{c}
\begin{array}{c}
g \\
\mu_1 \\
E
\end{array} \\
\begin{array}{c}
g \\
\mu_1
\end{array}
\end{array}
\begin{array}{c}
\text{for } w \text{ SfCprt} \\
\text{if } C \\
\text{SfCprt}
\end{array}
\begin{array}{c}
\begin{array}{c}
g \\
\mu_1 \\
J
\end{array}
\end{array}
\end{array}$$

Each fself_g -field can automatically rebuild the fself_g -program to the desired.

fself_g^N - OS and is designed for such transformations, and it fself_g can be transformed at $N \geq 1$, or it fself_g can be transformed at $N \geq 2$.

Remark 2.7.3. Hypothesis 2.7: equations for real processes in a non-trivial form can be used to fully or partially interpret the self_g -level of the process, replacing the equal signs with identification signs, and solutions to these equations as a manifestation of this level on the level of objectivity and ordinary energies. That is, equations for real processes serve as a definition of the self_g -level of the process, the definition of self_g -values (self_g -characteristics) of the process through the identification sign, i.e., they are defined (expressed) through themselves. In particular, forms (1.1) - (1.4) can be used as forms of identification. Each such singularity creates its own field, the process, the object etc. Much more effective than science for working with these singularities will be special Dynamic programming, which we are currently working on to create. Identification at the lower levels of a hierarchical dynamic structure of type (2.7.1) will lead to the upper level. You can also try to use it for full or partial interpretation of the self_g -level of chemical reactions, but here there will be a trivial identification and

determination of the self_g -level will be much simpler. For example, a type $w \equiv 2w$ singularity at the top level of the structure of a mathematical simplified model of DNA generates a field for DNA division. A rather complex type of singularity at the upper level of the structure of a simplified mathematical model generates an electromagnetic field through identification in Maxwell's equations.

symbols

Remark 2.7.4. Parallel operator $\text{sfCprt} \frac{g}{\mu_1}$ corresponds to
places for symbols

objects

theoretical science, parallel operator $\text{sfCprt} \frac{g}{\mu_1}$ corresponds to technology, x
 x

– the space “point” (space place).

Remark 2.7.5. Let fself_g -energy of A looks like $\text{sfCprt} \frac{C_A + \Delta C}{\mu_1} = \text{sfCprt} \frac{C_A}{\mu_1} +$
 $\frac{\Delta C}{C_A}$

$\text{sfCprt} \frac{\Delta C}{\mu_1} + \text{sfCprt} \frac{C_A}{\mu_1} + \text{sfCprt} \frac{\Delta C}{\mu_1}, \text{sfCprt} \frac{C_A}{\mu_1}$ corresponds to objectivity of A,

$$\text{sfCprt} \frac{\Delta C}{\mu_1} = E_A \quad (2.7.3),$$

E_A - usual energy of A. ΔC determined from (2.7.3) through C_A and then we can determine the complete fself_g -energy of A.

Remark 2.7.6. Let us consider an analogue of the Schrödinger equation for networks operating on electromagnetic energy

$$\frac{\partial w}{\partial x} = [w, \mu_g S(H)]$$

w - measure of self_g for networks operating, $\mu_g S(H)$ - measure of self_g for H, $H = H(\mu_g S(p), \mu_g S(q), t)$ - an analogue of the Hamiltonian in the space of actions of

artificial neurons in a neural network, q is the operator of an artificial neurons action result, p is the operator of an artificial neurons action impulse.

Remark 2.7.7. The self_g -space of a higher level contains many self_g -energetic fibers, collecting into appropriate sets that can be accessed by the corresponding self_g -spaces of lower levels. That's right, for example. This assembly point on the human cocoon can carry out this, in particular, access to our self_g -space with objects.

Remark 2.7.8. It is quite possible to try to build up the levels of objects and processes; change something at these levels.

Remark 2.7.9. One can try to conventionally represent the mathematical model (1.7.1) of the atom (molecule) as a hierarchical dynamic operator.

Remark 2.7.10. Here, self_g -action is understood as action on oneself $_g$ (i.e., to the same action), while physicists understand self_g -action, for example, as the absorption of one elementary particle by another of the same type.

Remark 2.7.10.1. Subtle energy can manifest itself in the form of: 1) objectivity, 2) ordinary energies, 3) information. Using neural networks of the SnnCSprt -type, it is possible to organize a SC-Internet, where instead of exchanging information, an exchange of subtle energies will take place.

2.8 FUZZY PROGRAM OPERATORS SfCprt , tprSfC , S^1fCepr , SfCeprt_1

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through SfCprt -Networks - fuzzy analogue of Sit-Networks [14] in one of the central departments of which a conventional computer system is located. The parallel processor is itself fgeprogram - fuzzy analogue of eprogram [14] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a

$$\text{SfCprt -program operator - SfCprt} \quad \begin{matrix} Ag \\ w \\ \mu \\ \text{activation} \end{matrix} \quad \text{with type of accommodation } w \text{ and}$$

measure of fuzziness μ , it will be possible to obtain the fuzzy execution with type of accommodation w and measure of fuzziness μ of a parallel fuzzy action A with the desired target weight g or the execution of a parallel action A with the desired fuzzy target weight g or both. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For SfCprt-coding and SfCprt-translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical pulses laser of Nobel laureates 2018 year: Gerard Mourou, Donna Strickland, or a combination of them. For the desired action, for example, using the direct parallel fgprogram of operator

$$\{\text{UHF AC} := Q\}$$

$$\text{SfCprt} \quad \begin{matrix} g \\ \mu \\ \text{activation} \end{matrix} \quad \text{with type of accommodation } g \text{ and measure of fuzziness } \mu,$$

we simultaneously enter the desired set of codes Q using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects.

Consider the types of direct parallel fuzzy fgprogram operators:

- 1) SfCprt --program operators
- 2) tprSfC -program operators
- 3) **S¹fCep_r** - program operators
- 4) SfCep_{r1}- program operators

One example is pattern recognition:

$$\begin{array}{ccc} & \text{image archive} & q \\ \text{if SfCprt} & \begin{matrix} g \\ \mu \\ B \end{matrix} & \exists \text{ SfCprt} \begin{matrix} g \\ \mu \\ B \end{matrix} \text{ then Name of } q \\ \text{SfCprt} & & \end{array} .$$

The example of SfCprt-program is

$$\text{SfCprt} \left\{ \begin{array}{c} \{p\} := \{a(x)\} \\ \text{SfCprt} \quad \begin{array}{c} g \\ \mu \\ x \end{array} \end{array} \right\}, \text{SfCprt} \left\{ \begin{array}{c} IF \{B\} \{f\} \text{ then } Q \\ \text{SfCprt} \quad \begin{array}{c} g \\ \mu \\ x \end{array} \end{array} \right\}, \text{SfCprt} \left\{ \begin{array}{c} Q \\ \text{SfCprt} \quad \begin{array}{c} g \\ \mu \\ Q \end{array} \end{array} \right\}.$$

Consider a third type of fuzzy capacity in itself - fuzzy analogue of capacity in

$$\text{itself. For example, based on } \text{SfCprt} \begin{array}{c} \tilde{x} \\ g \\ \mu \\ q \end{array}, \text{ where } \tilde{x} = (x_1 | \mu_{\tilde{x}}(x_1), x_2 | \mu_{\tilde{x}}(x_2), \dots, x_n | \mu_{\tilde{x}}(x_n)),$$

i.e. n - elements at one point, we can consider the capacity $SfC_3 f \tilde{x} \mu$ (in itself with m elements from $\tilde{x}$, $m < n$, which is formed according to the form from forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Consider a third type of fuzzy fgcapacity in itself. For example, based on $\text{SfCprt} \begin{array}{c} \tilde{x} \\ g \\ \mu \\ q \end{array}$,

where $\tilde{x} = (x_1 | \mu_{\tilde{x}}(x_1), x_2 | \mu_{\tilde{x}}(x_2), \dots, x_n | \mu_{\tilde{x}}(x_n))$ i.e. n - elements at one point, we can consider the fuzzy fgcapacity $SfC_3 f \mu$ in itself with m elements from $\tilde{x}$, $m < n$, which is formed according to the form:

Here are some of SfCprt-program operators.

1. Simultaneous fuzzy assignment with fuzziness μ of the expressions $\tilde{p} = (p_1 | \mu_{\tilde{p}}(p_1), p_2 | \mu_{\tilde{p}}(p_2), \dots, p_n | \mu_{\tilde{p}}(p_n))$ to the variables

$\tilde{x} = (x_1 | \mu_{\tilde{x}}(x_1), x_2 | \mu_{\tilde{x}}(x_2), \dots, x_n | \mu_{\tilde{x}}(x_n))$. This is implemented via

$$\text{SfCprt} \left\{ \begin{array}{c} \{\tilde{x}\} := \{p\} \\ \text{SfCprt} \quad \begin{array}{c} g \\ \mu \\ w \end{array} \end{array} \right\}.$$

2. Simultaneous fuzzy checking with fuzziness μ by the fuzzy set of conditions $\tilde{g} = (g_1 | \mu_{\tilde{g}}(g_1), g_2 | \mu_{\tilde{g}}(g_2), \dots, g_n | \mu_{\tilde{g}}(g_n))$ for the fuzzy set of

expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), ..., B_n|\mu_{\tilde{B}}(B_n))$ Implemented via

$$\text{SfCprt} \quad \underset{\substack{g \\ \mu \\ w}}{\text{IF} \left\{ \left\{ \tilde{B} \right\} \left\{ \tilde{g} \right\} \right\} \text{ then } \tilde{Q}}$$
 where $\tilde{Q}$ can be anything.

3. Similarly for fuzzy loop operators and others.

SfC_3f – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding SfCprt-program operators in form (1.1), for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

For example, $\text{SfCprt} \quad \underset{\substack{g \\ \mu \\ w\{R\}}}{\text{IF} \left\{ \left\{ R \right\} \right\} \text{ then } Q}$ is the fuzzy capacity with type of accommodation g and

measure of fuzziness μ in itself of the second type if $w\{R\}$ is a fffprogram capable of fuzzy generating

$\{R\}$ with type of accommodation g and measure of fuzziness μ .

The example of self-ffprogram of the first type is

$$\text{SfCprt} \quad \underset{\substack{g \\ \mu \\ w}}{\text{IF} \left\{ \left\{ p \right\} := \left\{ a(x) \right\} \right\} \text{ then } Q}, \text{SfCprt} \quad \underset{\substack{g \\ \mu \\ w}}{\text{IF} \left\{ \left\{ B \right\} \left\{ f \right\} \right\} \text{ then } Q}, \text{SfCprt} \quad \underset{\substack{g \\ \mu \\ Q}}{\text{IF} \left\{ \left\{ B \right\} \left\{ f \right\} \right\} \text{ then } Q}.$$

The example of SfCprt-program for SmnSfCprt :

$q :=$
 $\text{SfCprt} \quad \underset{\substack{g \\ \mu \\ B}}{\text{IF} \left\{ \left\{ B \right\} \right\} \text{ then } Q}$ - fuzzy assigning fuzzy q to fuzzy B with type of accommodation g and measure of fuzziness μ .

$tw :=$
 $\text{SfCprt} \quad \underset{\substack{g \\ \mu \\ d}}{\text{IF} \left\{ \left\{ d \right\} \right\} \text{ then } Q}$ - fuzzy assigning target weight tw to fuzzy d with type of accommodation g and measure of fuzziness μ .

$\text{SfCprt} \quad \underset{\substack{g \\ \mu}}{\text{IF} \left\{ \left\{ q \right\} w \right\} \text{ then } Q}$ - *ffSmnsprt activation* for fuzzy $\{q\}w$ with type of accommodation g and measure of fuzziness μ .

SfCprt-coding

SfCprt-coding with type of accommodation g and measure of fuzziness μ : 1) fuzzy set A to fuzzy set B , 2) fuzzy set A to a point q , where the elements of the fuzzy

sets A, B can be continuous. For example, $\text{sfCprt}_{\mu}^{g, \frac{A}{B}}$.

There are SfCprt-coding, SfCprt-translation, SfCprt-realize of fgeprograms and of the fprograms from the archives without extraction theirs

CfSelf-coding

Self-coding with type of accommodation g and measure of fuzziness μ : 1) fuzzy set A to set fuzzy A , i.e. fuzzy A on itself 2) fuzzy set A to a point q in form (1),

where the elements of the fuzzy sets A, B can be continuous. For example, $\text{sfCprt}_{\mu}^{g, \frac{A}{A}}$.

One of the central departments of the control system should be a computer system of the usual type of the desired level. In symbiosis with SfCprt-Networks, it will provide a holistic operation of the control system in three modes: conventional serial through a conventional type computer system, direct parallel through SfCprt-Networks and series-parallel. Codes from a conventional type computer system will be used via SfCprt -connectors in SfCprt - coding, for example:

$\{\text{UHF AC} := Q\}$
 $\text{sfCprt}_{\mu}^{g, \text{activation}}$. UHF AC field activation is used.

Dynamic SfCprt and S₃fCf(t) programming

The ideology of dynamic SfCprt and S₃fCf(t) can be used for programming:

1. Simultaneous assignment of the expressions $\widetilde{p(t)} = (p_1(t) | \mu_{\widetilde{p(t)}}(p_1(t)), p_2(t) | \mu_{\widetilde{p(t)}}(p_2(t)), \dots, p_n(t) | \mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables

$\widetilde{x(t)} = (x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is

$$\{ \widetilde{x(t)} \} := \{ p(t) \}$$

implemented via $\text{SfCprt}(t) \quad \frac{g}{\mu} \quad w(t)$.

2. Simultaneous checking the fuzzy set of conditions

$\widetilde{g(t)} = (g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \dots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)} = (B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \dots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$. Implemented via

$$\text{SfCprt}(t) \quad IF \{ \{ \widetilde{B(t)} \} \{ \widetilde{g(t)} \} \} \text{ then } \widetilde{Q(t)} \quad \frac{g}{\mu} \quad w(t) \quad \text{where } \widetilde{Q(t)} \text{ can be anything.}$$

3. Similarly for fuzzy loop operators and others.

$S_3fCf(t)$ – fuzzy software operators will differ only just because aggregates

$\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding SfCprt-program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

tpoSfC-program operators

The ideology of tpoSfC and $t_{SfC_{4f}}$ - fuzzy analogues of tSfC and $t_{SfC_{4f}}$ from [10] can be used for programming. Here are some of the SfC-program operators.

1. Simultaneous expelling assignment of the expressions $\widetilde{p} = (p_1|\mu_{\widetilde{p}}(p_1),$

$p_2|\mu_{\widetilde{p}}(p_2), \dots, p_n|\mu_{\widetilde{p}}(p_n))$ from the variables $\widetilde{x} = (x_1|\mu_{\widetilde{x}}(x_1), x_2|\mu_{\widetilde{x}}(x_2), \dots,$

$x_n|\mu_{\widetilde{x}}(x_n))$. This is implemented via $\frac{\widetilde{x}}{\mu} \text{ SfCprt.} \quad =: \widetilde{p}$

2. Simultaneous expelling check the fuzzy set of conditions $\widetilde{g} = (g_1|\mu_{\widetilde{g}}(g_1),$

$g_2|\mu_{\widetilde{g}}(g_2), \dots, g_n|\mu_{\widetilde{g}}(g_n))$ for the fuzzy set of expressions $\widetilde{B} = (B_1|\mu_{\widetilde{B}}(B_1),$

$B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through $IF\left\{\left\{\tilde{B}\right\}\left\{\tilde{g}\right\}\right\} then \tilde{Q}$ ^w
^g
^μ SfCprt

where Q can be any.

3. Similarly for loop operators and others.

t_{S_4fCf} – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding tprSfC program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical tprSfC-program operator

$$\begin{matrix} B \\ g \\ \mu \\ A \end{matrix} \text{SfCprt} = \left\{ D + \begin{matrix} \{\} \\ g \\ \mu \\ A - A \cap B \\ (B - A \cap B) \end{matrix} \text{SfCprt} \right\}, \text{ where D is oself-(fuzzy set) for fuzzy } (A \cap B).$$

Dynamic tprSfC and $t(q)_{SfC_4f}$ programming at time q

The ideology of tprSfC and t_{SfC_4f} can be used for dynamic programming. Here are some of the tprSfC(t)- dynamic programming operators.

1. The process of simultaneous expelling assignment of the

expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \dots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ from the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$.is

implemented through $\begin{matrix} \tilde{x}(t) \\ g \\ \mu \end{matrix} \text{SfCprt}(t).$
 $=: \tilde{p}(t)$

2. The process of simultaneous expelling check the fuzzy set of conditions

$\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \dots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \dots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is

implemented through $\begin{matrix} w(t) \\ g \\ \mu \end{matrix} \text{SfCprt}(t),$ where Q(t) can be any.
 $IF\left\{\left\{\tilde{B}(t)\right\}\left\{\tilde{g}(t)\right\}\right\} then \tilde{Q}(t)$

3. Similarly for loop operators and others.

t_{SfC_4f} – fuzzy software operators will differ only just because aggregates

$\overline{x(t)}, \overline{p(t)}, \overline{B(t)}, \overline{g(t)}$ will be formed from corresponding processes $\text{tprSfC}(t)$ for above mentioned programming operators through form (1.1) or form from forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*) for more complex operators.

Consider hierarchical dynamic tprSfC-program operator:

$$\begin{matrix} B(q) \\ g \\ \mu \\ A(q) \end{matrix} \text{SfCprt}(q) = \left\{ \begin{matrix} t(q)_{S_1fCf(A(q) \cap B(q))} + \frac{\begin{matrix} \{\} \\ g \\ \mu \end{matrix}}{A(q) - A(q) \cap B(q)} \text{SfCprt}(q) \\ (B(q) - A(q) \cap B(q)) \end{matrix} \right\},$$

S¹fCep_r-program operators (form $\begin{matrix} B & A \\ g_2 S^1 fCprt_{\mu_2}^{g_1} & \mu_1 \end{matrix}$ - fuzzy analogue of $\begin{matrix} B & A \\ D & B \end{matrix} S^1 t_B^A$

[11]))

For example, $\begin{matrix} \{a(t)\} \\ g_2 \\ \mu_2 \end{matrix} \text{S}^1 \text{fCprt} \begin{matrix} \{B(t)\} \{f(t)\} \\ g_1 \\ \mu_1 \end{matrix} \text{ then } Q(t) .$
 $\begin{matrix} \{=: \{p(t)\}\} \\ \{a(t)\} \end{matrix}$

Consider hierarchical dynamic S¹fCep_r-program operator: (form

$$\begin{matrix} B & A \\ g_2 S^1 fCprt_{\mu_2}^{g_1} * \\ \mu_2 & \mu_1 \\ A & B \\ B & A \end{matrix}).$$

$$\begin{matrix} g_2 & S^1 fCprt_{\mu_2}^{g_1} \\ \mu_2 & \mu_1 \\ D - A & B \end{matrix}$$

SfCep_rt₁- program operators (form $\begin{matrix} C & A \\ \mu_2 S_1 fCprt_{\mu_1}^{g_1} & \mu_1 \end{matrix} \left(\begin{matrix} C & A \\ \mu_2 & \mu_1 \\ D & B \end{matrix} \right)$ - fuzzy analogue

of $\begin{matrix} C & A \\ D & B \end{matrix} S_1 t_B^A$ [12]))

$\begin{matrix} a & f \\ a & f \end{matrix} S_1 t_a^a$ -- sample $\begin{pmatrix} self \\ oself \end{pmatrix}$ -fprogram structure example.

$fSt_{t_0} \left\{ \begin{matrix} q(a f S_1 t_a^a) \\ w_q fSt_{q(a f S_1 t_a^a)}^{Eq} \end{matrix}, fSt_{d_r}^{\{El^{dr}\}} \right\}$ can be interpreted as a fprogram operator.

${}^a f S_1 t_a^a$ can be interpreted as a $\begin{pmatrix} self \\ oself \end{pmatrix}$ -fprogram operator.

$$3. \text{ SfCprt} \left\{ \begin{array}{c} q \begin{pmatrix} a & a \\ g & S_1 f Crt g \\ \mu_1 & \mu_1 \end{pmatrix} \\ a \end{array} \begin{array}{c} E_q \\ W_q \end{array} fSprt, fSprt_{d_r(E_{in}^{l^{d_r}})}^{E^{ex} l^{d_r}} \right\} \text{—fgprogram structure}$$

$$\begin{array}{c} g \\ \mu \\ t_0 \end{array}$$

example, where the assemblage point d_r is the cursor, it is quite complex self—fprogram.

$$4. \text{ SfCprt} \left\{ \begin{array}{c} q \begin{pmatrix} a & a \\ g & S_1 f Crt g \\ \mu_1 & \mu_1 \end{pmatrix} \\ a \end{array} \begin{array}{c} E_q \\ W_q \end{array} fSprt, fSprt_{d_r(E_{in}^{l^{d_r}})}^{E^{ex} l^{d_r}} \right\} \text{ can be interpreted as a}$$

$$\begin{array}{c} g \\ \mu \\ t_0 \end{array}$$

fgprogram operator.

$\begin{pmatrix} a & a \\ \mu_1 S_1 f Crt \mu_1 \\ a & a \end{pmatrix}$ can be interpreted as a $\begin{pmatrix} s_1 elf \\ os_1 elf \end{pmatrix}$ -fprogram operator. $\begin{pmatrix} a & a \\ g & S_1 f Crt g \\ \mu_1 & \mu_1 \end{pmatrix}$ —

sample $\begin{pmatrix} s_1 elf \\ os_1 elf \end{pmatrix}$ -fprogram structure example.

Remark 2.8. Energy of a living organism:

$$\text{Cfg}(r, a(E_q)) = \text{SfCprt} \left\{ \begin{array}{c} q \begin{pmatrix} a & a \\ g & S_1 f Crt g \\ \mu_1 & \mu_1 \end{pmatrix} \\ a \end{array} \begin{array}{c} E_q \\ W_q \end{array} fSprt, fSprt_{d_r(E_{in}^{l^{d_r}})}^{E^{ex} l^{d_r}} \right\}$$

$$\begin{array}{c} g \\ \mu \\ t_0 \end{array}$$

$\begin{pmatrix} a & a \\ g & S_1 f Crt g \\ \mu_1 & \mu_1 \end{pmatrix}$ — internal energy of a living organism, q- a gap in the energy cocoon

of a living organism, r-the position of the assemblage point d_r on the energy cocoon

of a living organism, W_q - energy prominences from the gap in the cocoon of a

living organism, E_q -external energy entering the gap in the cocoon of a living organism, $E^{ex}l^{dr}$ - a bundle of fibers of external energy self-capacities from outside the cocoon, collected at the point of assembly of the cocoon of a living organism at time t_0 , $E_{in}l^{dr}$ - a bundle of fibers of external energy self-capacities from inside the cocoon, collected at the point of assembly of the cocoon of a living organism in the same position r of the assemblage point d_r . d_r is the subject of identifying the energy fibers of the subtle energy of the Universe in position r both outside and inside the cocoon.

Energy with measure of fuzziness μ_1, μ_2 of a living organism of a person:

$$Cfpg(r, a(E_q)) = SfCprt \left\{ \begin{matrix} q \left(\begin{matrix} a \\ \mu_1 \\ a \end{matrix} \begin{matrix} g \\ SfCprt \\ g \end{matrix} \begin{matrix} a \\ \mu_1 \\ a \end{matrix} \right) \\ W_q \end{matrix} \right. fSprt^{E_q}, fSprt_{d_r}^{\{E^{ex}l^{dr}\}} \left. \begin{matrix} q \left(\begin{matrix} a \\ \mu_1 \\ a \end{matrix} \begin{matrix} g \\ SfCprt \\ g \end{matrix} \begin{matrix} a \\ \mu_1 \\ a \end{matrix} \right) \\ g \\ \mu \\ t_0 \end{matrix} \right\}$$

Appendix

If we introduce for the energy of a chemical element the concept $fself_g$ -energy (the

concept of a chemical element was introduced earlier): $SfCprt \frac{R}{\mu_1}^g$, $R=Q+D$, Q -

internal energy, D is the energy of its interaction with the external environment.

$$\begin{aligned} SfCprt \frac{R}{\mu_1}^g &= SfCprt \frac{Q+D}{\mu_1}^g = SfCprt \frac{Q}{\mu_1}^g + SfCprt \frac{D}{\mu_1}^g = SfCprt \frac{Q}{\mu_1}^g + SfCprt \frac{Q}{\mu_1}^g + SfCprt \frac{D}{\mu_1}^g \\ &+ SfCprt \frac{D}{\mu_1}^g, SfCprt \frac{Q}{\mu_1}^g - \text{internal } fself_g \text{-energy, } SfCprt \frac{D}{\mu_1}^g - \text{the external } fself_g \text{-energy,} \\ &SfCprt \frac{Q}{\mu_1}^g - \text{object component of a chemical element, } SfCprt \frac{D}{\mu_1}^g - \text{usual energy} \end{aligned}$$

component of a chemical element. We describe the usual chemical reactions for the

$\text{SfCprt} \frac{Q}{D} \frac{g}{\mu_1}$ -component using the $\text{SfCprt} \frac{D}{Q} \frac{g}{\mu_1}$ -component. A fself_g -molecule (fself_g -atom,

fself_g -(elementary particle))) as a capacity can have the following types of self_g :
 fself_g -set, fself_g -structure, fself_g -hierarchy or its elements that generates this fself_g -molecule (fself_g -atom, fself_g -(elementary particle))).

self_g -power is force that is applied to oneself_g or its elements that generates this self_g -power.

You can try to consider the equations: $\text{SfCprt} \frac{x}{x} \frac{g}{\mu_1} = a, x(a) - ?$, $\text{SfCprt} \frac{x}{b} \frac{g}{\mu_1} = a, x(a, b) - ?$,

$\text{SfCprt} \frac{q}{x} \frac{g}{\mu_1} = a, x(a, q) - ?$.

Supplement for Quantum Mechanics and Classical statistical Mechanics through SfCprt -elements:

Hamilton operator $\widehat{H} = \widehat{H}_0 + \widehat{W}_0$, $\widehat{H}_0$ -considered quantum system energy, consisting of two or more parts, without their interaction with each other, $\widehat{W}_0$ is the

energy of their interaction, $\widehat{\rho}$ -statistical operator [20]. fself_g -energy $\text{SfCprt} \frac{\widehat{H}}{\mu_1} \frac{g}{\widehat{H}}$

$\text{SfCprt} \frac{\widehat{H}_0 + \widehat{W}_0}{\mu_1} \frac{g}{\widehat{H}_0 + \widehat{W}_0} = \text{SfCprt} \frac{\widehat{H}_0}{\mu_1} \frac{g}{\widehat{H}_0 + \widehat{W}_0} + \text{SfCprt} \frac{\widehat{W}_0}{\mu_1} \frac{g}{\widehat{H}_0 + \widehat{W}_0} = \text{SfCprt} \frac{\widehat{H}_0}{\mu_1} \frac{g}{\widehat{H}_0} + \text{SfCprt} \frac{\widehat{H}_0}{\mu_1} \frac{g}{\widehat{W}_0} +$
 $\text{SfCprt} \frac{\widehat{W}_0}{\mu_1} \frac{g}{\widehat{H}_0} + \text{SfCprt} \frac{\widehat{W}_0}{\mu_1} \frac{g}{\widehat{W}_0}$, $\text{SfCprt} \frac{\widehat{H}_0}{\mu_1} \frac{g}{\widehat{H}_0}$ -considered quantum system fself_g -energy, $\text{SfCprt} \frac{\widehat{W}_0}{\mu_1} \frac{g}{\widehat{W}_0}$ is

fself_g -energy of their interaction, $\text{SfCprt} \frac{\widehat{H}_0}{\mu_1} \frac{g}{\widehat{W}_0}$ --object manifestation of the energy of

the system in an external field., $\widehat{W}_0$ μ_1^g - the manifestation of the energy of the system in the energy interaction with the external field. Variants of the Schrödinger equation $\frac{\partial \hat{\rho}}{\partial t} + [\hat{W}, \hat{\rho}] = 0$ of the form SfC₂f, SfC₃f are possible, using the form (1.1) or form from the forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

The carrier of the measure of objectivity-mass should be objectivity - elementary particle graviton, look like SfCpr μ_1^g , therefore it is a self_g-particle and is *objectivity*

not an element of the level of objectivity, but is an element of the level fself_g.

Therefore, it cannot be found at our level. In fact, the theory of SfCprt-elements helps to form a unified field theory on a qualitative level, because it is not possible to create a quantitative unified field theory. Supplement for string theory: May be to try represent elementary particles in the form of continual self_g-elements of the type $CS_{\infty}^- = \sin(-\infty)|g \rightarrow \downarrow I \uparrow_{-1}^1|g$, $CT_{\infty}^+ = \text{tg}\infty|g \rightarrow \uparrow I \downarrow_{-\infty}^{\infty}|g$, $CT_{\infty}^- = \text{tg}(-\infty)|g \rightarrow \downarrow I \uparrow_{-\infty}^{\infty}|g$, $f \uparrow I \downarrow w|g$ for any f, w etc.

We consider SfCprt-logic: consider the functional ffC(Q), which gives a numerical value for the truth_g of the statement Q from the interval [0,1], where 0 corresponds to "no," and one corresponds to the logical value "yes." Then for joint statements A, B: $\text{ffC}(A+B) = \text{ffC}(A) + \text{ffC}(B) - \text{ffC}(A*B) + \text{fSfC}(D)$, D- self_g-statement from A*B, fSfC(x)- the value of self_g-truth for fself_g-statement x; for dependent statements: $\text{ffC}(A*B) = \text{ffC}(A) * \text{ffC}(B/A) = \text{ffC}(B) * \text{ffC}(A/B)$, where $\text{ffC}(B/A)$ - conditional truth_g of the statement B at statement A, $\text{ffC}(A/B)$ - dependent truth_g of statement A at the statement B. Adding the truth_g values of inconsistent propositions: $\text{ffC}(A+B) = \text{ffC}(A) + \text{ffC}(B)$. The formula of complete truth_g: $\text{ffC}(A) = \sum_{k=1}^n \text{ffC}(B_k) * \text{ffC}(A/B_k)$, B₁, B₂, ..., B_n-full group of hypotheses-statements: $\sum_{k=1}^n \text{ffC}(B_k) = 1$ ("yes").

Remark. A statement can be interpreted as an event, and its truth value as a probability.

SfCprt- statement for set of statements $A=\{A_1, A_2, \dots, A_n\}$: $\text{SfCprt} \begin{matrix} \{A_1, A_2, \dots, A_n\} \\ g_1 \\ \mu_1 \\ x \end{matrix}$,

$\{\text{ff}(A_1), \text{ff}(A_2), \dots, \text{ff}(A_n)\}$
 $\text{SfCprt} \begin{matrix} g_1 \\ \mu_1 \\ x \end{matrix}$, - SfCprt- truth for these statements. It is possible to

consider the fself_g -statement SC_3A with m statements from A , at $m < n$, which is formed by the form (1.1), that is, only m statements from A are located in the

structure $\text{SfCprt} \begin{matrix} A \\ g_1 \\ \mu_1 \\ x \end{matrix}$. The same for fself_g -truth $SfC_3\{f(A_1), f(A_2), \dots, f(A_n)\}$.

One can introduce the concepts of SfCprt-group: $\text{SfCprt} \begin{matrix} A \\ g_1 \\ \mu_1 \\ x \end{matrix}$, A is usual group, $\text{SfCprt} \begin{matrix} A \\ g_1 \\ \mu_1 \\ B \end{matrix}$, where A, B - usual groups, self-group: SfCf_iA , $i=1,2,3$, A is usual group.

Definition A. A structure with a second degree of freedom will be called complete, i.e., "capable" of reversing self_g concerning any of its elements clearly, but not necessarily in known operators; it can form (create) new

special operators (in particular, special functions). In particular, $\text{cfCprt} \begin{matrix} A \\ g_1 \\ \mu_1 \\ A \end{matrix}$,

$\text{cfCpr} \begin{matrix} A \\ g_1 \\ \mu_1 \\ A \end{matrix}$ are such structures. Similarly, for working with models, each is structured

by its structure; for example, use SfCprt-groups, SfCprt-rings, SfCprt-fields, SfCprt-spaces, fself_g -groups, fself_g -rings, fself_g -fields, and fself_g -spaces. Like any task, this is also a structure of the appropriate capacity. Since the degree of freedom is double, it is clear that the form of the fself_g -equation contains a solution or structures the inversion of the fself_g -equation concerning unknowns, i.e., the structure of the self_g -equation is complete. The transition process in the

form of $\overset{C}{\mu_2^{g_2}} \text{SfCprt} \overset{A}{\mu_1^{g_1}}$ is included in the transition from one world A (spatial variables, which we will denote by X1, and time variables, by T1) to another world B (spatial variables, which we will denote by X2, and time variables, by T2). It is accompanied by spatial variables in the form (T1, X1), i.e. such a transition process transforms time variables T1 into part of spatial ones, and time variables - T3.

Supplement

Connection SfCprt – elements with usual functionals and operators

We consider functional $g(x): X \rightarrow g, x \in X, g$ —numerical value of functional $g(x)$. It is specific capacity for X. $\text{SfCprt} \overset{g_1}{\mu_1}$ made from her the fself_g -capacity in itself $\{g(x)\}$ as an element $\text{SfC}_1\{g(x)\}$, $\{g(x)\}$ —the set of any functionals for X. In particular, probability $p(X)$ —is such functional, X —an event. Here $\text{SfCprt} \overset{g_1}{\mu_1}$ is $\text{SfC}_1\{p(X)\}$, denote it through $p\text{SfC}(X)$. Usual event is dynamical capacity.

Definition B. SCprt-probability of events A, B is $p(\text{SfCprt} \overset{A}{\mu_1^{g_1}})$, denote $\text{SfCpp} \overset{A}{\mu_1^{g_1}}$. In particular,

$\text{SfCpp} \overset{A}{\mu_1^{g_1}}$ for joint A, B: $\text{SfCpp} \overset{A}{\mu_1^{g_1}} = p(\text{SfCprt} \overset{A}{\mu_1^{g_1}}) = p(\{A \cup B - A \cap B\}) * \mu(g_1) * \mu_1 = p(A) + p(B) - p(AB) + p\text{SfC}(D)$, D - the fself_g -capacity in itself μ_1 as an element from $A \cap B$, $p\text{SfC}(D)$ —probability fself_g of D of next level— fself_g level. The probability for stochastic value X is capacity. We represent its distribution in the kind of SfCprt-element:

$$\text{SfCpp} \overset{g_1}{\mu_1^x} \{ (x_1, p_1), (x_2, p_2), \dots, (x_n, p_n) \} (*)$$

Here interest represent partial distribution f_{self_g} from (*) by form (1.1) or form from the forms (1.1.1) – (1.4) with value f_{self_g} of stochastic value X for some subset $\{x_{1_1}, x_{2_1}, \dots, x_{j_1}\} \in \{x_1, x_2, \dots, x_n\}$ with probabilities

$$f_{self_g}\{pCfS_1, pCfS_2, \dots, pCfS_j\}. \text{ For operator } X_1 \xrightarrow{F} X_2: \xrightarrow{F} X_2 \text{ is capacity for } X_1. \text{ SfCprt } \xrightarrow{g_1} \mu_1 \xrightarrow{F} X_2$$

-- f_{self_g} - capacity in $itself_g$ as an element for X_1 . More complex for implicit

$$F(X_1, X_2) = 0$$

operator: $F(X_1, X_2) = 0$. Then $\text{SfCprt } \xrightarrow{g_1} \mu_1$ forms f_{self_g} -capacity in $itself_g$ as

$$F(X_1, X_2) = 0$$

an element for X_1 relatively of X_2 or for X_2 relatively of X_1 . x obtains more power of the liberty and in this is direct decision (i. e. f_{self_g} -capacity in $itself_g$ as an element for x). f_{self_g} -equation for x has its decision for x in direct kind. f_{self_g} -task for x has its decision for x in direct kind. f_{self_g} -question has its answer for x in direct kind. x acquires more degree of liberty and in this is direct decision. We consider

$$\text{SfCprt } \xrightarrow{g_1} \mu_1 \xrightarrow{D} \text{D-block over execution subject in } S_{mn\text{SfCprt}} \text{ for networks. Then we have}$$

f_{self_g} -capacity in $itself_g$ as an element D , where full realization requires

$$\text{SmnSfCprt}$$

correspondent f_{self_g} -energy. $\text{SfCprt } \xrightarrow{g} \mu_1$ increase f_{self_g} -level of $S_{mn\text{SfCprt}}$ and

$$\text{SmnSCprt}$$

may made no visual its. The entire neural network as instantaneous simultaneous

$$\text{RAM in SfCprt-elements and } f_{self_g} \text{ - elements. } self_g^{self_g \dots self_g}, f_1 \downarrow$$

$$I \uparrow_{-1}^1 f_2 \quad f_1 \downarrow I \uparrow_{-1}^1 f_2 \dots f_1 \downarrow I \uparrow_{-1}^1 f_2, \sin \infty | g^{\sin \infty | g \dots \sin \infty | g}. \text{ When activated in a neural network, the entire neural network becomes a working memory. Use of } f_{self_g} \text{ -energy as}$$

$$\begin{array}{c}
 S_{mnSfCprt} \\
 SfCprt \quad \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
 activation \\
 g_1 \\
 \mu_1
 \end{array}
 \rightarrow self_g -RAM,$$

activation or from outside. $QC_0 = SfCprt$

$$\begin{array}{c}
 S_{mnSfCprt} \\
 SfCprt \quad \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
 activation \\
 g_1 \\
 \mu_1
 \end{array}$$

$$\begin{array}{c}
 S_{mnSfCprt} \\
 g_2 \quad SfCprt \\
 \mu_2 \\
 activation \\
 g_2 \\
 \mu_2
 \end{array}
 \quad
 QC_{00} = \quad
 QC_0, QC_{01} = \quad
 \begin{array}{c}
 S_{mnSfCprt} \\
 SfCprt \quad \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
 activation \\
 g_2 \\
 \mu_2
 \end{array}
 \quad
 QC_0.$$

$$\begin{array}{c}
 S_{mnSfCprt} \\
 g_2 \quad SfCprt \\
 \mu_2 \\
 activation \\
 g_2 \\
 \mu_2
 \end{array}
 \quad
 \begin{array}{c}
 S_{mnSfCprt} \\
 SfCprt \quad \begin{array}{c} g_1 \\ \mu_1 \end{array} \\
 activation \\
 g_1 \\
 \mu_1
 \end{array}$$

QC_0, QC_{00}, QC_{01} -coding, translation, realization fgeprograms, QC_0, QC_{00}, QC_{01} -
 $S_{mnSCprt}, QC_0, QC_{00}, QC_{01}$ -Assembler.

References

- 1) A. Mostowski. Constructive sets. UNIVERSITY OF WARSAW-North-Holland Publishing Company-Amsterdam. PWN-Polish Scientific Publishers-Warszawa. 1969.
- 2) Thomas J. Jech. LECTURES IN SET THEORY WITH PARTICULAR EMPHASIS ON THE METHOD OF FORCING. SPRINGER-VERLAG Berlin. Heidelberg. New-York, 1971.
- 3) Oleksandr Danilishyn, Illia Danilishyn. Introduction to Dynamic Sets Theory: SCprt-elements and Their Applications to the Physics and Chemistry. JOURNAL OF PHYSICS AND CHEMISTRY, vol. 2, issue 3, pp. 1-31, March 27, 2024 https://cskscientificpress.com/articles_file/527-article1712797848.pdf
- 4) Ershov Y.L. On a Hierarchy of Sets I, Algebras and Logic, 7 (1968), 47 - 73.
- 5) Ershov Y.L.] On a Hierarchy of Sets II, Algebras and Logic, 7 (1968), 15 - 47.

- 6) Yrshov Y.L. On a Hierarchy of Sets III, Algebra and Logic, 9(1970), 34 -
- 7) Krain S.G. Linear differential equations in Banach space.
M., Science, 1967 (in Russian).
- 8) A.L. Dontchev. Perturbations, approximations and sensitivity analysis of optimal control systems. Springer-Verlag, Berlin, New York, 1983.
- 9) Galushkin A. Networks: principles of the theory. Hot line-Telecom. M., 2010 (in Russian).
- 10) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the V International Scientific and Practical Conference, Oxford, June 23, 2023. Oxford-Vinnytsia: P.C. Publishing House & European Scientific Platform, 2023. Theoretical and empirical scientific research: concept and trends. pp.156-161, DOI 10.36074/logos-23.06.2023.42, <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/12>
- 11) Danilishyn I.V. Danilishyn O.V. SET1 – ELEMENTS.
INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 28
June, 2023 with the proceedings of the: Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 9 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC International Centre Corporative Management (Vienna, Austria), pp. 239-254. <https://archive.journal-grail.science/index.php/2710-3056/issue/view/09.06.2023>
- 12) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets S1et and Some of their Applications in Physics. Science Set Journal of Physics, 25 September 2023, 1-11, 815- article1695707465.pdf (mkscienceset.com)
- 13) I. Danilishyn, O. Danilishyn Program Operators Sit, tS, S1e, Set1. Journal of Sensor Networks and Data Communications [ISSN: 2994-6433] 2023, Volume 3 | Issue 1 | 138-143, <https://www.opastpublishers.com/table-contents/jsndc-volume-3-issue-1-year-2023>

- 14) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 15) Danilishyn I.V. Danilishyn O.V. DYNAMICAL SIT-ELEMENTS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 16) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO SETS THEORY AND OTHERS. Scientific practice: modern and classical research methods: Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the IV International Scientific and Practical Conference, Boston, May 26, 2023. Boston-Vinnytsia: Primedia eLaunch & European Scientific Platform, 2023, pp.166-171. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/11>
- 17) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO CONTINUAL VALUED LOGIC AND OTHERS. Features of the development of modern science in the pandemic's era: a collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, May 19, 2023. Berlin, Federal Republic of Germany: European Scientific Platform, pp.79-84. <https://previous.scientia.report/index.php/archive/issue/view/19.05.2023>
- 18) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. [Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1).
<https://doi.org/10.21203/rs.3.rs-3217178/v1>
[Dynamic Sets Theory: Sit-elements and Their Applications | Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1)
[www.researchsquare.com/article/rs-3217178/v1](https://doi.org/10.21203/rs.3.rs-3217178/v1)

- 19) Danilishyn I., Danilishyn O. VARIABLE HIERARCHICAL DYNAMICAL STRUCTURES (MODELS) FOR DYNAMIC, SINGULAR, HIERARCHICAL SETS AND THE PROBLEM OF COLD THERMONUCLEAR FUSION The driving force of science and trends in its development: collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, July 14, 2023. Coventry, United Kingdom: European Scientific. Pp.113-119. Platform. <https://previous.scientia.report/index.php/archive/issue/view/14.07.2023>
- 20) D.I.Blokhintsev. Quantum Mechanics Lectures on Selected Topics. Moscow, Atomizdat, 1981.

Declarations:

Availability of data and material

- 1) Oleksandr Danilishyn, Illia Danilishyn. Introduction to Dynamic Sets Theory: SCprt-elements and Their Applications to the Physics and Chemistry. JOURNAL OF PHYSICS AND CHEMISTRY, vol. 2, issue 3, pp. 1-31, March 27, 2024 https://cskscientificpress.com/articles_file/527-article1712797848.pdf
- 2) Illia Danilishyn and Oleksandr Danilishyn. Dynamic Sets Set and Some of Their Applications to Neuroscience, Networks Set New Advances in Brain & Critical Care 2023, Volume 4 | Issue 2 | 66- 81. <https://doi.org/10.33140/NABCC.04.02.02>
- 3) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets S1et and Some of their Applications in Physics. Science Set Journal of Physics, 25 September 2023, 1-11, 815- _ article1695707465.pdf (mkscienceset.com)
- 4) I. Danilishyn, O. Danilishyn Dynamic Sets Se, Networks Se. *Advances in Neurology and Neuroscience*, 2023, Volume 6 | Issue 2 | 278-294. <https://www.opastpublishers.com/open-access-articles/dynamic-sets-se-networks-se.pdf>

- 5) I. Danilishyn, O. Danilishyn Program Operators Sit, tS, S1e, Set1. Journal of Sensor Networks and Data Communications [ISSN: 2994-6433] 2023, Volume 3 | Issue 1 | 138-143, <https://www.opastpublishers.com/table-contents/jsndc-volume-3-issue-1-year-2023>
- 6) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 7) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the V International Scientific and Practical Conference, Oxford, June 23, 2023. Oxford-Vinnytsia: P.C. Publishing House & European Scientific Platform, 2023. Theoretical and empirical scientific research: concept and trends. **pp.156-161**, DOI 10.36074/logos-23.06.2023.42, <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/12>
- 8) Danilishyn I.V. Danilishyn O.V. SET1 – ELEMENTS. INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 28 June, 2023 with the proceedings of the: Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 9 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC International Centre Corporative Management (Vienna, Austria), **pp. 239-254**. <https://archive.journal-grail.science/index.php/2710-3056/issue/view/09.06.2023>
- 9) Danilishyn I.V. Danilishyn O.V. DYNAMICAL SIT-ELEMENTS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>

- 10) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO SETS THEORY AND OTHERS. Scientific practice: modern and classical research methods: Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the IV International Scientific and Practical Conference, Boston, May 26, 2023. Boston-Vinnytsia: Primedia eLaunch & European Scientific Platform, 2023, pp.166-171. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/11>
- 11) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO CONTINUAL VALUED LOGIC AND OTHERS. Features of the development of modern science in the pandemic's era: a collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, May 19, 2023. Berlin, Federal Republic of Germany: European Scientific Platform, pp.79-84. <https://previous.scientia.report/index.php/archive/issue/view/19.05.2023>
- 12) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. [Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1).
<https://doi.org/10.21203/rs.3.rs-3217178/v1>
[Dynamic Sets Theory: Sit-elements and Their Applications | Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1)
[www.researchsquare.com/article/rs-3217178/v1](https://doi.org/10.21203/rs.3.rs-3217178/v1)

Competing interest

There are no competing interests. All sections of the monograph are executed jointly.

Funding

There were no sources of funding for writing the monograph.

Authors' contributions

The contribution of the authors is the same, we will not separate.

Part III. Wprt-elements and Their Applications

Introduction

The part III task: to understand hierarchy of energies in the Universe and the principles of functioning of living energy (living organism, in particular, human, subtle energies), and then using these principles to "construct" artificial living energies (let's call them pseudo-living energies). We transform all kinds of uncertainties into new singularities and use them to describe uncertain energies. Here is considered paradoxical singularities (singularities of disintegration&synthesis), self-type singularities.

A Wprt – elements, self-type Wprt-structures

A.1 Wprt – elements, self-type Wprt-structures

Let's consider the vector of energy levels of a non-living object A:

$$\begin{pmatrix} \text{subtle energy of the upper level object } A(\text{decignation} - \tilde{A}) \\ \text{subtle energy of a mid – level object } A(\text{decignation} - \bar{A}) \\ \text{the raw energy of an object } A - \text{its objectivity}(\text{decignation} - A) \\ \text{ordinary energy exhibited by an object } A(\text{decignation} - \underline{A}) \end{pmatrix}$$

Let's consider the vector of energy levels of a living object A:

$$\begin{pmatrix} \text{subtle energy of the paradoxical upper level object } A(\text{decignation} - \tilde{\tilde{A}}) \\ \text{subtle energy of a paradoxical mid – level object } A(\text{decignation} - \bar{\bar{A}}) \\ \text{subtle energy of the upper level object } A(\text{decignation} - \tilde{A}) \\ \text{subtle energy of a mid – level object } A(\text{decignation} - \bar{A}) \\ \text{the raw energy of an object } A - \text{its objectivity}(\text{decignation} - A) \\ \text{ordinary energy exhibited by an object } A(\text{decignation} - \underline{A}) \end{pmatrix}$$

Let's consider the vector of energy levels of an action A [14]:

$$\left(\begin{array}{l} \text{subtle energy of the paradoxical upper level action } A(\text{decignation} - \tilde{\tilde{A}}) \\ \text{subtle energy of a paradoxical mid - level action } A(\text{decignation} - \bar{\bar{A}}) \\ \text{subtle energy of the self} - ||| \\ \text{subtle energy of the upper level action } A(\text{decignation} - \tilde{A}) \\ \text{subtle energy of a mid - level action } A(\text{decignation} - \bar{A}) \\ \text{the raw energy of an action } A(\text{decignation} - A) \\ \text{ordinary energy exhibited by an action } A(\text{decignation} - \underline{A}) \end{array} \right) \quad (\text{A.1*})$$

May consider the induction from clotting. The result of it is the new self-type of a higher level than clotting result.

$$\left(\begin{array}{l} \text{subtle energy of the paradoxical upper level induction } A(\text{decignation} - \tilde{\tilde{A}}) \\ \text{subtle energy of a paradoxical mid - level induction } A(\text{decignation} - \bar{\bar{A}}) \\ \text{subtle energy of the self} - ||| \\ \text{subtle energy of the upper level induction } A(\text{decignation} - \tilde{A}) \\ \text{subtle energy of a mid - level induction } A(\text{decignation} - \bar{A}) \\ \text{the raw energy of an induction } A(\text{decignation} - A) \\ \text{ordinary energy exhibited by an induction } A(\text{decignation} - \underline{A}) \end{array} \right) \quad (\text{A.2*})$$

Remark A.1.0. As an action A we can take a clotting of energies into more subtle

ones (accumulation). Let us introduce the notations of Rprt $\begin{matrix} \text{action } A \\ B \\ \text{action } A \end{matrix}$ [18] by

$\begin{matrix} 1 \\ B \end{matrix} \text{self} A,$

Rprt $\begin{matrix} B \\ \text{action } A \end{matrix}$ by $\begin{matrix} 2 \\ B \end{matrix} \text{self} A$, Rprt $\begin{matrix} \text{action } A \\ B \end{matrix} A^{-1}$ (or Rprt $\begin{matrix} \text{action } A \\ B \end{matrix} - A$) by $\begin{matrix} 3 \\ B \end{matrix} \text{paself} A$,

Rprt $\begin{matrix} \text{action } A^{-1} \\ B \end{matrix}$ (or Rprt $\begin{matrix} \text{action } A \\ B \end{matrix} - A$) by $\begin{matrix} -3 \\ B \end{matrix} \text{paself} A$, Rprt $\begin{matrix} B \\ \text{action } A \end{matrix} A^{-1}$ (or

Rprt $\begin{matrix} B \\ \text{action } A \end{matrix} - A$) by $\begin{matrix} -2 \\ B \end{matrix} \text{paself} A$, Rprt $\begin{matrix} B \\ \text{action } A \end{matrix} A^{-1}$ (or

$\begin{matrix} B \\ \text{Rprt } action - A \end{matrix}$) by $\begin{matrix} \text{action } A \\ \text{action } A \end{matrix}$ $\begin{matrix} \text{action } A^{-1} \\ B \end{matrix}$ (or Rprt $\begin{matrix} \text{action } - A \\ B \end{matrix}$) by $\begin{matrix} \text{action } A \\ \text{action } A \end{matrix}$
 $\begin{matrix} \text{action } A \\ \text{action } A^{-1} \end{matrix}$ $\begin{matrix} \text{action } A \\ B \end{matrix}$ (or Rprt $\begin{matrix} \text{action } A \\ B \end{matrix}$) by $\begin{matrix} \text{action } - A \\ \text{action } - A \end{matrix}$ $\begin{matrix} \text{action } A \\ \text{action } A \end{matrix}$. Then (A.1*):

$$\left(\begin{array}{c} \tilde{\tilde{A}} \\ (\overline{\overline{A}}) \\ self||| \\ ||| \\ \tilde{A} \\ {}^1_A self A \\ {}^1_B self A, {}^2_B self A \\ A \\ \underline{A} \end{array} \right) \quad (A.1^{**}).$$

${}^1_A self A$ has self-level: $self^{\frac{3}{2}}(A)$. Self-action $\overline{A}$ is interpreted as the subject of action $\overline{A}$ (spirit of action $\overline{A}$). Here $\overline{A}$ represented by a hierarchical set

$$\left(\begin{array}{c} {}^1_A self A \\ {}^1_B self A, {}^2_B self A \end{array} \right), \text{ and } (\overline{\overline{A}}) \text{ is represented by a hierarchical set}$$

$$\left(\begin{array}{c} {}_{A^{-1}}^1 paself A, {}_{\overline{A}^{-1}}^1 paself A, {}_{\overline{A}^{-2}}^2 paself A, {}_{\overline{A}^{-2}}^2 paself A, {}_{\overline{A}^{-3}}^3 paself A, {}_{\overline{A}^{-3}}^3 paself A, \\ {}_{\overline{A}^{-1}}^1 paself A, {}_{\overline{A}^{-1}}^1 paself A, {}_{\overline{A}^{-2}}^2 paself A, {}_{\overline{A}^{-2}}^2 paself A, {}_{\overline{A}^{-3}}^3 paself A, {}_{\overline{A}^{-3}}^3 paself A \\ {}^1_B self A, {}^2_B self A, {}_{\overline{B}^{-1}}^1 paself A, {}_{\overline{B}^{-2}}^2 paself A, {}_{\overline{B}^{-3}}^3 paself A, {}_{\overline{B}^{-3}}^3 paself A \end{array} \right)$$

or

$$\left(\begin{array}{c} {}_{\overline{A}^{-1}}^1 paself A, {}_{\overline{A}^{-1}}^1 paself A, {}_{\overline{A}^{-2}}^2 paself A, {}_{\overline{A}^{-2}}^2 paself A, {}_{\overline{A}^{-3}}^3 paself A, {}_{\overline{A}^{-3}}^3 paself A, \\ {}_{\overline{A}^{-1}}^1 paself A, {}_{\overline{A}^{-1}}^1 paself A, {}_{\overline{A}^{-2}}^2 paself A, {}_{\overline{A}^{-2}}^2 paself A, {}_{\overline{A}^{-3}}^3 paself A, {}_{\overline{A}^{-3}}^3 paself A \\ {}^1_B self A, {}^2_B self A, {}_{\overline{B}^{-1}}^1 paself A, {}_{\overline{B}^{-2}}^2 paself A, {}_{\overline{B}^{-3}}^3 paself A, {}_{\overline{B}^{-3}}^3 paself A \end{array} \right).$$

The structure of accumulation levels (their hierarchy) is nothing more than a 2-interpretation of the accumulation capacity (its form). Next - the self-capacity of accumulation, etc. according to its scheme (A.1**). Next - the self-scheme corresponding to the scheme (A.1**). An accumulation is manifestation of self-accumulation or self-action according to accumulation or self-change according to accumulation.

Remark A.1.0*. May consider $\text{self}((A.1^{**}))$, $\text{paself}((A.1^{**}))$, $\text{self}^2((A.1^{**}))$, $\text{paself}^2((A.1^{**}))$, ..., $\text{Af}(n, \text{self}((A.1^{**}))) = (\text{self}^n((A.1^{**})), \text{pAf}(n, \text{paself}((A.1^{**}))) = \text{paself}^n((A.1^{**})), \dots, \text{Af}(\alpha, \text{self}((A.1^{**}))), \text{pAf}(\alpha, \text{paself}((A.1^{**}))), \text{Af}(\infty, \text{self}((A.1^{**}))), \text{pAf}(\infty, \text{paself}((A.1^{**}))), \text{Af}(\text{self}((A.1^{**})), \text{self}((A.1^{**}))), \text{pAf}(\text{paself}((A.1^{**})), \text{paself}((A.1^{**})))$ etc. May transform of uncertainty $\sin(0^*\infty)$ into new singularity and use it to describe uncertain energies.

We consider dynamic operator

$$\begin{array}{ccc}
 C & & \dots \\
 \uparrow & & \tilde{\tilde{H}} \\
 \bar{P} & & \uparrow \\
 \uparrow & & \overline{\overline{F}} \\
 \tilde{R} & & \uparrow \\
 \uparrow \text{ Wprt} & (A.1.1), & \tilde{A} \\
 \overline{\overline{G}} & & \uparrow \\
 \uparrow & & \bar{Q} \\
 \tilde{\tilde{O}} & & \uparrow \\
 \dots & & B
 \end{array}$$

where $\tilde{\tilde{H}}, \tilde{\tilde{O}}$ - paradoxical upper levels of H and O respectively, $\overline{\overline{F}}, \overline{\overline{G}}$ - paradoxical average levels of F and G respectively, $\tilde{A}, \tilde{R}$ - upper levels of A and R respectively, $\bar{Q}, \bar{P}$ - average levels of Q and P respectively, B goes to the middle level of Q - $\bar{Q}$, $\bar{Q}$ goes to the upper level of A - $\tilde{A}$, $\tilde{A}$ goes to the paradoxical middle level of F - $\overline{\overline{F}}$, $\overline{\overline{F}}$ goes to the paradoxical upper level of H - $\tilde{\tilde{H}}$, $\tilde{\tilde{O}}$ goes to the paradoxical middle level of G - $\overline{\overline{G}}$, $\overline{\overline{G}}$ goes to the upper level of R - $\tilde{R}$, $\tilde{R}$ goes to

the middle level of $P - \bar{P}$, $\bar{P}$ goes to the lower level of C simultaneously. The result of this process will be described by the expression

$$\begin{array}{c}
 C \\
 \uparrow \\
 \bar{P} \\
 \uparrow \\
 \tilde{R} \\
 \uparrow \\
 \bar{G} \\
 \uparrow \\
 \tilde{O}
 \end{array}
 \begin{array}{c}
 \tilde{H} \\
 \uparrow \\
 \bar{F} \\
 \uparrow \\
 \tilde{A} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B
 \end{array}
 \text{Wrt} \quad (A.1.2).$$

Definition A.1.1. The dynamic operator (A.1.1) we shall call Wprt – element of the first type, (A.1.2) we shall call Wrt – element of the first type.

Remark A.1.1 Can consider Wprt – elements use the Banach space.

It's allowed to add Wprt – elements:

$$\begin{array}{c}
 C \\
 \uparrow \\
 \bar{P} \\
 \uparrow \\
 \tilde{R} \\
 \uparrow \\
 \bar{G} \\
 \uparrow \\
 \tilde{O}
 \end{array}
 \begin{array}{c}
 \tilde{H}_1 \\
 \uparrow \\
 \bar{F} \\
 \uparrow \\
 \tilde{A} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B
 \end{array}
 \text{Wprt} +
 \begin{array}{c}
 C \\
 \uparrow \\
 \bar{P} \\
 \uparrow \\
 \tilde{R} \\
 \uparrow \\
 \bar{G} \\
 \uparrow \\
 \tilde{O}
 \end{array}
 \begin{array}{c}
 \tilde{H}_2 \\
 \uparrow \\
 \bar{F} \\
 \uparrow \\
 \tilde{A} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B
 \end{array}
 \text{Wprt} =
 \begin{array}{c}
 C \\
 \uparrow \\
 \bar{P} \\
 \uparrow \\
 \tilde{R} \\
 \uparrow \\
 \bar{G} \\
 \uparrow \\
 \tilde{O}
 \end{array}
 \begin{array}{c}
 \tilde{H}_1 \cup \tilde{H}_2 \\
 \uparrow \\
 \bar{F} \\
 \uparrow \\
 \tilde{A} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B
 \end{array}
 \text{Wprt} \quad (A.1.2.1),$$

$$\begin{array}{c}
 C \\
 \uparrow \\
 \bar{P} \\
 \uparrow \\
 \tilde{R} \\
 \uparrow \\
 \bar{G} \\
 \uparrow \\
 \tilde{O}
 \end{array}
 \begin{array}{c}
 \tilde{H} \\
 \uparrow \\
 \bar{F} \\
 \uparrow \\
 \tilde{A} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B_1
 \end{array}
 \text{Wprt} +
 \begin{array}{c}
 C \\
 \uparrow \\
 \bar{P} \\
 \uparrow \\
 \tilde{R} \\
 \uparrow \\
 \bar{G} \\
 \uparrow \\
 \tilde{O}
 \end{array}
 \begin{array}{c}
 \tilde{H} \\
 \uparrow \\
 \bar{F} \\
 \uparrow \\
 \tilde{A} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B_2
 \end{array}
 \text{Wprt} =
 \begin{array}{c}
 C \\
 \uparrow \\
 \bar{P} \\
 \uparrow \\
 \tilde{R} \\
 \uparrow \\
 \bar{G} \\
 \uparrow \\
 \tilde{O}
 \end{array}
 \begin{array}{c}
 \tilde{H} \\
 \uparrow \\
 \bar{F} \\
 \uparrow \\
 \tilde{A} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B_1 \cup B_2
 \end{array}
 \text{Wprt} \quad (A.1.2.2),$$

$$\begin{array}{c}
C_1 \\
\uparrow \\
\bar{P} \\
\uparrow \\
\bar{R} \\
\uparrow \\
\bar{G} \\
\uparrow \\
\bar{O}
\end{array}
\begin{array}{c}
\bar{H} \\
\uparrow \\
\bar{F} \\
\uparrow \\
\bar{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
C_2 \\
\uparrow \\
\bar{P} \\
\uparrow \\
\bar{R} \\
\uparrow \\
\bar{G} \\
\uparrow \\
\bar{O}
\end{array}
\begin{array}{c}
\bar{H} \\
\uparrow \\
\bar{F} \\
\uparrow \\
\bar{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
C_1 \cup C_2 \\
\uparrow \\
\bar{P} \\
\uparrow \\
\bar{R} \\
\uparrow \\
\bar{G} \\
\uparrow \\
\bar{O}
\end{array}
\begin{array}{c}
\bar{H} \\
\uparrow \\
\bar{F} \\
\uparrow \\
\bar{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.1.2.3}),$$

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\bar{R}_1 \\
\uparrow \\
\bar{G} \\
\uparrow \\
\bar{O}
\end{array}
\begin{array}{c}
\bar{H} \\
\uparrow \\
\bar{F} \\
\uparrow \\
\bar{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\bar{R}_2 \\
\uparrow \\
\bar{G} \\
\uparrow \\
\bar{O}
\end{array}
\begin{array}{c}
\bar{H} \\
\uparrow \\
\bar{F} \\
\uparrow \\
\bar{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\overbrace{\bar{R}_1 \cup \bar{R}_2} \\
\uparrow \\
\bar{G} \\
\uparrow \\
\bar{O}
\end{array}
\begin{array}{c}
\bar{H} \\
\uparrow \\
\bar{F} \\
\uparrow \\
\bar{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.1.2.4}),$$

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P}_1 \\
\uparrow \\
\bar{R} \\
\uparrow \\
\bar{G} \\
\uparrow \\
\bar{O}
\end{array}
\begin{array}{c}
\bar{H} \\
\uparrow \\
\bar{F} \\
\uparrow \\
\bar{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
C \\
\uparrow \\
\bar{P}_2 \\
\uparrow \\
\bar{R} \\
\uparrow \\
\bar{G} \\
\uparrow \\
\bar{O}
\end{array}
\begin{array}{c}
\bar{H} \\
\uparrow \\
\bar{F} \\
\uparrow \\
\bar{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
C \\
\uparrow \\
\bar{P}_1 \cup \bar{P}_2 \\
\uparrow \\
\bar{R} \\
\uparrow \\
\bar{G} \\
\uparrow \\
\bar{O}
\end{array}
\begin{array}{c}
\bar{H} \\
\uparrow \\
\bar{F} \\
\uparrow \\
\bar{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.1.2.5}),$$

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{\widetilde{O_1}}
\end{array}
\text{Wprt}
\begin{array}{c}
\widetilde{\widetilde{H}} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{\widetilde{O_2}}
\end{array}
\text{Wprt}
\begin{array}{c}
\widetilde{\widetilde{H}} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{\widetilde{O_1}} \cup \widetilde{\widetilde{O_2}}
\end{array}
\text{Wprt}
\begin{array}{c}
\widetilde{\widetilde{H}} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.1.2.6.3}),$$

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{\widetilde{O}}
\end{array}
\text{Wprt}
\begin{array}{c}
\widetilde{\widetilde{H}} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A_1} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{\widetilde{O}}
\end{array}
\text{Wprt}
\begin{array}{c}
\widetilde{\widetilde{H}} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A_2} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{\widetilde{O}}
\end{array}
\text{Wprt}
\begin{array}{c}
\widetilde{\widetilde{H}} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A_1} \cup \widetilde{A_2} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.1.2.6.4}),$$

We consider the following self-type Wprt-structure, paradoxical self-type Wprt-structure (paself-type Wprt-structure) of the first type:

$$\begin{array}{c}
Q \\
\uparrow \\
\bar{Q} \\
\uparrow \\
\widetilde{Q} \\
\uparrow \\
\overline{\overline{Q}} \\
\uparrow \\
\widetilde{\widetilde{Q}}
\end{array}
\text{Wprt}
\begin{array}{c}
\widetilde{\widetilde{Q}} \\
\uparrow \\
\overline{\overline{Q}} \\
\uparrow \\
\widetilde{Q} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
Q
\end{array}
\quad (\text{A.1.3}),$$

denote $W_1 f Q$.

$$\begin{array}{ccc}
Q & & \tilde{\tilde{A}} \\
\uparrow & & \uparrow \\
\bar{Q} & & \overline{\overline{Q}} \\
\uparrow & & \uparrow \\
\tilde{Q} & \text{Wprt} & \tilde{Q} \quad (\text{A.1.4}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q}} & & \bar{Q} \\
\uparrow & & \uparrow \\
\tilde{\tilde{A}} & & Q
\end{array}$$

denote $W_2 f A; Q$.

$$\begin{array}{ccc}
B & & \dots \\
\uparrow & & \tilde{\tilde{H}} \\
\bar{Q} & & \uparrow \\
\uparrow & & \overline{\overline{F}} \\
\tilde{A} & \text{Wprt} & \tilde{A} \quad (\text{A.1.5}), \\
\uparrow & & \uparrow \\
\overline{\overline{F}} & & \bar{Q} \\
\uparrow & & \uparrow \\
\tilde{\tilde{H}} & & B \\
\dots & &
\end{array}$$

denote $W_3 f H, F, A; Q; B$.

$$\begin{array}{ccc}
A & & \dots \\
\uparrow & & \tilde{\tilde{A}} \\
\bar{Q} & & \uparrow \\
\uparrow & & \overline{\overline{Q}} \\
\tilde{Q} & \text{Wprt} & \tilde{Q} \quad (\text{A.1.6}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q}} & & \bar{Q} \\
\uparrow & & \uparrow \\
\tilde{\tilde{A}} & & A \\
\dots & &
\end{array}$$

denote $W_4 f A; Q$.

$$\begin{array}{ccc}
 a & & \dots \\
 \uparrow & & \text{str } \overset{\sim}{\tilde{A}} \\
 \bar{Q} & & \uparrow \\
 \uparrow & & \overline{\overline{Q}} \\
 \underbrace{\quad} & & \uparrow \\
 \bar{Q} & \text{Wprt} & \overset{\sim}{\tilde{Q}} \\
 \uparrow & & \uparrow \\
 \overline{\overline{Q}} & & \bar{Q} \\
 \uparrow & & \uparrow \\
 \text{str } \overset{\sim}{\tilde{A}} & & a \\
 \dots & &
 \end{array} \quad (\text{A.1.6.1}),$$

denote $W_5 fA; Q; a, a \subset A$,

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\check{R} \\
\uparrow \\
\equiv \\
\bar{G} \\
\uparrow \\
\check{O} \\
\vdots
\end{array}
\begin{array}{c}
\vdots \\
\check{\check{H}} \\
\uparrow \\
\equiv \\
\bar{F} \\
\uparrow \\
\check{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.1.6.2}),$$

denote $W_6fa; Q; A, a \subset A$,

$$\begin{array}{c} B \\ \uparrow \\ \bar Q \\ \uparrow \\ \bar Q \\ \uparrow \\ \equiv \\ \bar Q \\ \uparrow \\ \bar Q \\ \vdots \end{array} \quad \text{Wprt} \quad \begin{array}{c} \vdots \\ \bar A \\ \uparrow \\ \equiv \\ \bar Q \\ \uparrow \\ \bar Q \\ \uparrow \\ \bar Q \\ \uparrow \\ B \end{array} \quad (\text{A.1.7}),$$

and any other possible options of self for (A.1.1) etc.

It can be considered a simpler version of the dynamic operator

$$\begin{array}{c}
 \dots \\
 \tilde{\tilde{H}} \\
 \uparrow \\
 \overline{\overline{F}} \\
 \text{Wprt} \uparrow (A.1.8), \\
 \tilde{\tilde{A}} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 B
 \end{array}$$

where $\tilde{\tilde{H}}$ - paradoxical upper level of H, $\overline{\overline{F}}$ - paradoxical average level of F, $\tilde{\tilde{A}}$ - upper levels of A, $\overline{Q}$ - average levels of Q, B goes to the middle level of Q - $\overline{Q}$, $\overline{Q}$ goes to the upper level of A - $\tilde{\tilde{A}}$, $\tilde{\tilde{A}}$ goes to the paradoxical middle level of F - $\overline{\overline{F}}$, $\overline{\overline{F}}$ goes to the paradoxical upper level of H - $\tilde{\tilde{H}}$, simultaneously, the result of this process will be described by the expression

$$\begin{array}{c}
 \dots \\
 \tilde{\tilde{H}} \\
 \uparrow \\
 \overline{\overline{F}} \\
 \text{Wrt} \uparrow (A.1.9) \\
 \tilde{\tilde{A}} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 B
 \end{array}$$

or

$$\begin{array}{c}
 C \\
 \uparrow \\
 \overline{P} \\
 \uparrow \\
 \tilde{\tilde{R}} \\
 \uparrow \text{Wprt} (A.1.10) \\
 \overline{\overline{G}} \\
 \uparrow \\
 \tilde{\tilde{O}} \\
 \dots
 \end{array}$$

where $\hat{\hat{O}}$ goes to the paradoxical middle level of $G - \overline{\overline{G}}$, $\overline{\overline{G}}$ goes to the upper level of $R - \hat{\hat{R}}$, $\hat{\hat{R}}$ goes to the middle level of $P - \bar{P}$, $\bar{P}$ goes to the lower level of C simultaneously, the result of this process will be described by the expression

$$\begin{array}{c}
 C \\
 \uparrow \\
 \bar{P} \\
 \uparrow \\
 \hat{\hat{R}} \\
 \uparrow \text{ Wprt (A.1.11)} \\
 \overline{\overline{G}} \\
 \uparrow \\
 \hat{\hat{O}} \\
 \dots
 \end{array}$$

Definition 1,2. The dynamic operator (A.1.8) we shall call Wprt – element of the second type, (A.1.9) we shall call Wrt – element of the second type.

It's allowed to add Wprt – elements of the second type:

$$\begin{array}{c}
 \hat{\hat{H}}_1 \\
 \uparrow \\
 \overline{\overline{F}} \\
 \uparrow \\
 \hat{\hat{A}} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B
 \end{array}
 +
 \begin{array}{c}
 \hat{\hat{H}}_2 \\
 \uparrow \\
 \overline{\overline{F}} \\
 \uparrow \\
 \hat{\hat{A}} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B
 \end{array}
 =
 \begin{array}{c}
 \hat{\hat{H}}_1 \cup \hat{\hat{H}}_2 \\
 \uparrow \\
 \overline{\overline{F}} \\
 \uparrow \\
 \hat{\hat{A}} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B
 \end{array}
 \quad (\text{A.1.12}),$$

$$\begin{array}{c}
 \hat{\hat{H}} \\
 \uparrow \\
 \overline{\overline{F}} \\
 \uparrow \\
 \hat{\hat{A}} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B_1
 \end{array}
 +
 \begin{array}{c}
 \hat{\hat{H}} \\
 \uparrow \\
 \overline{\overline{F}} \\
 \uparrow \\
 \hat{\hat{A}} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B_2
 \end{array}
 =
 \begin{array}{c}
 \hat{\hat{H}} \\
 \uparrow \\
 \overline{\overline{F}} \\
 \uparrow \\
 \hat{\hat{A}} \\
 \uparrow \\
 \bar{Q} \\
 \uparrow \\
 B_1 \cup B_2
 \end{array}
 \quad (\text{A.1.13}),$$

$$\begin{array}{ccc}
\begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q_1} \\ \uparrow \\ B \end{array} & + \text{Wprt} \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q_2} \\ \uparrow \\ B \end{array} & = \text{Wprt} \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q_1} \cup \overline{Q_2} \\ \uparrow \\ B \end{array} \quad (\text{A.1.13.1}),
\end{array}$$

$$\begin{array}{ccc}
\begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F_1}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} & + \text{Wprt} \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F_2}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} & = \text{Wprt} \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F_1}} \cup \overline{\overline{F_2}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} \quad (\text{A.1.13.2}),
\end{array}$$

$$\begin{array}{ccc}
\begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A_1}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} & + \text{Wprt} \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A_2}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} & = \text{Wprt} \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A_1}} \cup \widetilde{\widetilde{A_2}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} \quad (\text{A.1.13.3}).
\end{array}$$

We consider the following self-type Wprt-structures of the second type:

$$\begin{array}{c}
 \dots \\
 \tilde{\tilde{A}} \\
 \uparrow \\
 \overline{\overline{Q}} \\
 \text{Wprt} \uparrow \text{ (A.1.14),} \\
 \tilde{\tilde{Q}} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 A
 \end{array}$$

$$\begin{array}{c}
 \dots \\
 \tilde{\tilde{A}} \\
 \uparrow \\
 \overline{\overline{A}} \\
 \text{Wprt} \uparrow \text{ (A.1.14.1),} \\
 \tilde{\tilde{A}} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 a
 \end{array}$$

denote $W_7 f A; Q; a, a \subset A$,

$$\begin{array}{c}
 \dots \\
 \tilde{\tilde{a}} \\
 \uparrow \\
 \overline{\overline{a}} \\
 \text{Wprt} \uparrow \text{ (A.1.15),} \\
 \tilde{\tilde{a}} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 str A
 \end{array}$$

denote $W_8 f a; Q; A, a \subset A$,

$$\begin{array}{c}
 \dots \\
 \overline{\overline{A}} \\
 \uparrow \\
 \overline{\overline{A}} \\
 \uparrow \\
 \overline{\overline{A}} \quad \text{Wprt (A.1.16),} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 Q
 \end{array}$$

and any other possible options of self for (A.1.8) etc.

Definition A.1.3. The dynamic operator (A.1.10) we shall call tprW – element, (A.1.11) we shall call trW – element.

It's allowed to add tprW – elements:

$$\begin{array}{ccc}
 C_1 & C_2 & C_1 \cup C_2 \\
 \uparrow & \uparrow & \uparrow \\
 \overline{P} & \overline{P} & \overline{P} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{R} & \overline{R} & \overline{R} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{G}} & \overline{\overline{G}} & \overline{\overline{G}} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{O}} & \overline{\overline{O}} & \overline{\overline{O}} \\
 \dots & \dots & \dots
 \end{array}
 \quad \text{Wprt +} \quad
 \begin{array}{ccc}
 C_1 & C_2 & C_1 \cup C_2 \\
 \uparrow & \uparrow & \uparrow \\
 \overline{P} & \overline{P} & \overline{P} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{R} & \overline{R} & \overline{R} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{G}} & \overline{\overline{G}} & \overline{\overline{G}} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{O}} & \overline{\overline{O}} & \overline{\overline{O}} \\
 \dots & \dots & \dots
 \end{array}
 \quad \text{Wprt =} \quad
 \begin{array}{ccc}
 C_1 & C_2 & C_1 \cup C_2 \\
 \uparrow & \uparrow & \uparrow \\
 \overline{P} & \overline{P} & \overline{P} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{R} & \overline{R} & \overline{R} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{G}} & \overline{\overline{G}} & \overline{\overline{G}} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{O}} & \overline{\overline{O}} & \overline{\overline{O}} \\
 \dots & \dots & \dots
 \end{array}
 \quad \text{Wprt (A.1.17),}$$

$$\begin{array}{ccc}
 C & C & C \\
 \uparrow & \uparrow & \uparrow \\
 \overline{P} & \overline{P} & \overline{P} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{R} & \overline{R} & \overline{R} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{G}} & \overline{\overline{G}} & \overline{\overline{G}} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{O}} & \overline{\overline{O}} & \overline{\overline{O}} \\
 \dots & \dots & \dots
 \end{array}
 \quad \text{Wprt +} \quad
 \begin{array}{ccc}
 C & C & C \\
 \uparrow & \uparrow & \uparrow \\
 \overline{P} & \overline{P} & \overline{P} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{R} & \overline{R} & \overline{R} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{G}} & \overline{\overline{G}} & \overline{\overline{G}} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{O}} & \overline{\overline{O}} & \overline{\overline{O}} \\
 \dots & \dots & \dots
 \end{array}
 \quad \text{Wprt =} \quad
 \begin{array}{ccc}
 C & C & C \\
 \uparrow & \uparrow & \uparrow \\
 \overline{P} & \overline{P} & \overline{P} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{R} & \overline{R} & \overline{R} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{G}} & \overline{\overline{G}} & \overline{\overline{G}} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{O}} & \overline{\overline{O}} & \overline{\overline{O}} \\
 \dots & \dots & \dots
 \end{array}
 \quad \text{Wprt (A.1.18),}$$

$$\begin{array}{ccc}
\begin{array}{c} C \\ \uparrow \\ \overline{P_1} \\ \uparrow \\ \vec{R} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \vec{\vec{O}} \\ \dots \end{array} & + & \begin{array}{c} C \\ \uparrow \\ \overline{P_2} \\ \uparrow \\ \vec{R} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \vec{\vec{O}} \\ \dots \end{array} \\
\text{Wprt} & & \text{Wprt}
\end{array} = \begin{array}{c} C \\ \uparrow \\ (\overline{P_1} \cup \overline{P_2}) \\ \uparrow \\ \vec{R} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \vec{\vec{O}} \\ \dots \end{array} \quad \text{Wprt (A.1.18.1),}$$

$$\begin{array}{ccc}
\begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \vec{R_1} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \vec{\vec{O}} \\ \dots \end{array} & + & \begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \vec{R_2} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \vec{\vec{O}} \\ \dots \end{array} \\
\text{Wprt} & & \text{Wprt}
\end{array} = \begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \vec{R_1} \cup \vec{R_2} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \vec{\vec{O}} \\ \dots \end{array} \quad \text{Wprt (A.1.18.2),}$$

$$\begin{array}{ccc}
\begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \vec{R} \\ \uparrow \\ \overline{G_1} \\ \uparrow \\ \vec{\vec{O}} \\ \dots \end{array} & + & \begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \vec{R} \\ \uparrow \\ \overline{G_2} \\ \uparrow \\ \vec{\vec{O}} \\ \dots \end{array} \\
\text{Wprt} & & \text{Wprt}
\end{array} = \begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overbrace{\vec{R}} \\ \uparrow \\ \overline{G_1} \cup \overline{G_2} \\ \uparrow \\ \vec{\vec{O}} \\ \dots \end{array} \quad \text{Wprt (A.1.18.3).}$$

We consider the following self-type tprL-structures:

$$\begin{array}{c}
O \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{ Wprt (A.1.19)} \\
\overline{\overline{O}} \\
\uparrow \\
\tilde{\tilde{O}} \\
\ldots \\
str D \\
\uparrow \\
\bar{Q} \\
\uparrow \\
\tilde{Q} \\
\uparrow \text{ Wprt (A.1.19.1),} \\
\overline{\overline{Q}} \\
\uparrow \\
\tilde{\tilde{d}} \\
\ldots
\end{array}$$

denote $W_9 f d; Q; D, d \subset D$,

$$\begin{array}{c}
d \\
\uparrow \\
\bar{Q} \\
\uparrow \\
\overbrace{Q} \\
\uparrow \text{ Wprt (A.1.20),} \\
\overline{\overline{D}} \\
\uparrow \\
str \tilde{\tilde{D}} \\
\ldots
\end{array}$$

denote $W_{10} f D; Q; d, d \subset D$,

$$\begin{array}{c}
P \\
\uparrow \\
\overline{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \text{ Wprt (A.1.21)} \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{\widetilde{O}} \\
\vdots
\end{array}$$

and any other possible options of self for (A.1.10) etc.

Dynamic Wprt – elements, self-type dynamic Wprt-structures

We considered Wprt – elements earlier. Here we consider dynamic Wprt – elements. We consider dynamic operator whose elements change over time

$$\begin{array}{ccc}
C(t) & & \overbrace{H(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{P(t)} & & \overline{\overline{F(t)}} \\
\uparrow & & \uparrow \\
\widetilde{R(t)} & \text{Wprt}(t) & \widetilde{A(t)} \quad (2.1), \\
\uparrow & & \uparrow \\
\overline{\overline{G(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{O(t)}^{\dots} & & B(t)
\end{array}$$

where $\overbrace{H(t)}^{\dots}$, $\overbrace{O(t)}^{\dots}$ - paradoxical upper levels of $H(t)$ and $O(t)$ respectively, $\overline{\overline{F(t)}}$, $\overline{\overline{G(t)}}$ - paradoxical average levels of $F(t)$ and $G(t)$ respectively, $\widetilde{A(t)}$, $\widetilde{R(t)}$ - upper levels of $A(t)$ and $R(t)$ respectively, $\overline{Q(t)}$, $\overline{P(t)}$ - average levels of $Q(t)$ and $P(t)$ respectively, $B(t)$ goes to the middle level of $Q(t)$ - $\overline{Q(t)}$, $\overline{Q(t)}$ goes to the upper level of $A(t)$ - $\widetilde{A(t)}$, $\widetilde{A(t)}$ goes to the paradoxical middle level of $F(t)$ - $\overline{\overline{F(t)}}$, $\overline{\overline{F(t)}}$ goes to the paradoxical upper level of $H(t)$ - $\overbrace{H(t)}^{\dots}$, $\overbrace{O(t)}^{\dots}$ goes to the paradoxical middle level of $G(t)$ - $\overline{\overline{G(t)}}$, $\overline{\overline{G(t)}}$ goes to the upper level of $R(t)$ - $\widetilde{R(t)}$, $\widetilde{R(t)}$ goes to the middle level of $P(t)$ - $\overline{P(t)}$, $\overline{P(t)}$ goes to the lower level

of $C(t)$ simultaneously. The result of this process will be described by the expression

$$\begin{array}{c}
 C(t) \\
 \uparrow \\
 \overline{P(t)} \\
 \uparrow \\
 \overline{\overline{R(t)}} \\
 \uparrow \\
 \overline{\overline{G(t)}} \\
 \uparrow \\
 \overline{\overline{O(t)}} \\
 \dots
 \end{array}
 \begin{array}{c}
 \dots \\
 \overline{\overline{H(t)}} \\
 \uparrow \\
 \overline{\overline{F(t)}} \\
 \uparrow \\
 \overline{\overline{A(t)}} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 B(t)
 \end{array}
 \begin{array}{c}
 \\
 \\
 \\
 \text{Wrt}(t) \\
 \\
 \\
 \\
 \end{array}
 \begin{array}{c}
 \\
 \\
 \\
 \\
 \\
 \\
 \\
 \end{array}
 (2.2).$$

Definition A.1.2.1. The dynamic operator (A.1.2.1) we shall call dynamic Wprt – element of the first type, (A.1.2.2) we shall call dynamic Wrt – element of the first type.

It's allowed to add dynamic Wprt – elements:

$$\begin{array}{c}
 C(t) \\
 \uparrow \\
 \overline{P(t)} \\
 \uparrow \\
 \overline{\overline{R(t)}} \\
 \uparrow \\
 \overline{\overline{G(t)}} \\
 \uparrow \\
 \overline{\overline{O(t)}} \\
 \dots
 \end{array}
 \begin{array}{c}
 \dots \\
 \overline{\overline{H_1(t)}} \\
 \uparrow \\
 \overline{\overline{F(t)}} \\
 \uparrow \\
 \overline{\overline{A(t)}} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 B(t)
 \end{array}
 \begin{array}{c}
 \\
 \\
 \\
 \text{Wprt}(t) \\
 \\
 \\
 \\
 \end{array}
 \begin{array}{c}
 C(t) \\
 \uparrow \\
 \overline{P(t)} \\
 \uparrow \\
 \overline{\overline{R(t)}} \\
 \uparrow \\
 \overline{\overline{G(t)}} \\
 \uparrow \\
 \overline{\overline{O(t)}} \\
 \dots
 \end{array}
 \begin{array}{c}
 \dots \\
 \overline{\overline{H_2(t)}} \\
 \uparrow \\
 \overline{\overline{F(t)}} \\
 \uparrow \\
 \overline{\overline{A(t)}} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 B(t)
 \end{array}
 \begin{array}{c}
 \\
 \\
 \\
 \text{Wprt}(t) \\
 \\
 \\
 \\
 \end{array}
 =
 \begin{array}{c}
 C(t) \\
 \uparrow \\
 \overline{P(t)} \\
 \uparrow \\
 \overline{\overline{R(t)}} \\
 \uparrow \\
 \overline{\overline{G(t)}} \\
 \uparrow \\
 \overline{\overline{O(t)}} \\
 \dots
 \end{array}
 \begin{array}{c}
 \dots \\
 \overline{\overline{H_1(t) \cup H_2(t)}} \\
 \uparrow \\
 \overline{\overline{F(t)}} \\
 \uparrow \\
 \overline{\overline{A(t)}} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 B(t)
 \end{array}
 \begin{array}{c}
 \\
 \\
 \\
 \text{Wprt}(t) \\
 \\
 \\
 \\
 \end{array}$$

(A.1.2.2.1),

$$\begin{array}{c}
C(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overline{R(t)} \\
\uparrow \text{Wprt}(t) \\
\overline{\overline{G(t)}} \\
\uparrow \\
\overline{\overline{O(t)}} \\
\vdots
\end{array}
\begin{array}{c}
\overline{\overline{\overline{H(t)}}} \\
\uparrow \\
\overline{\overline{F(t)}} \\
\uparrow \\
\overline{A_1(t)} \cup \overline{A_2(t)} \text{ (A.1.2.2.7),} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B(t)
\end{array}$$

$$\begin{array}{c}
C(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overline{R(t)} \\
\uparrow \text{Wprt}(t) \\
\overline{\overline{G_1(t)}} \\
\uparrow \\
\overline{\overline{O(t)}} \\
\vdots
\end{array}
\begin{array}{c}
\overline{\overline{\overline{H(t)}}} \\
\uparrow \\
\overline{\overline{F(t)}} \\
\uparrow \\
\overline{A(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B(t)
\end{array}
+
\begin{array}{c}
C(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overline{R(t)} \\
\uparrow \text{Wprt}(t) \\
\overline{\overline{G_2(t)}} \\
\uparrow \\
\overline{\overline{O(t)}} \\
\vdots
\end{array}
=
\begin{array}{c}
\overline{\overline{\overline{H(t)}}} \\
\uparrow \\
\overline{\overline{F(t)}} \\
\uparrow \\
\overline{A(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B(t)
\end{array}$$

$$\begin{array}{c}
C(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overline{R(t)} \\
\uparrow \text{Wprt}(t) \\
\overline{\overline{G_1(t)} \cup \overline{G_2(t)}} \\
\uparrow \\
\overline{\overline{O(t)}} \\
\vdots
\end{array}
\begin{array}{c}
\overline{\overline{\overline{H(t)}}} \\
\uparrow \\
\overline{\overline{F(t)}} \\
\uparrow \\
\overline{A(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B(t)
\end{array}
\text{ (A.1.2.2.8),}$$

$$\begin{array}{ccccc}
C(t) & & \overbrace{H(t)}^{\dots} & & C(t) & & \overbrace{H(t)}^{\dots} \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\overline{P(t)} & & \overline{F(t)} & & \overline{P(t)} & & \overline{F(t)} \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\overbrace{R(t)} & \text{Wprt}(t) & \overbrace{A(t)} & + & \overbrace{R(t)} & \text{Wprt}(t) & \overbrace{A(t)} = \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\overline{G(t)} & & \overline{Q(t)} & & \overline{G(t)} & & \overline{Q(t)} \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\overbrace{O_1(t)} & & B(t) & & \overbrace{O_2(t)} & & B(t) \\
\vdots & & & & \vdots & &
\end{array}$$

$$\begin{array}{ccc}
C(t) & & \overbrace{H(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{P(t)} & & \overline{F(t)} \\
\uparrow & & \uparrow \\
\overbrace{R(t)} & \text{Wprt}(t) & \overbrace{A(t)} \text{ (A.1.2.2.9),} \\
\uparrow & & \uparrow \\
\overline{G(t)} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{O_1(t)} \cup \overbrace{O_2(t)} & & B(t) \\
\vdots & & \vdots
\end{array}$$

$$\begin{array}{ccccc}
C(t) & & \overbrace{H(t)}^{\dots} & & C(t) & & \overbrace{H(t)}^{\dots} \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\overline{P(t)} & & \overline{F_1(t)} & & \overline{P(t)} & & \overline{F_2(t)} \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\overbrace{R(t)} & \text{Wprt}(t) & \overbrace{A(t)} & + & \overbrace{R(t)} & \text{Wprt}(t) & \overbrace{A(t)} = \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\overline{G(t)} & & \overline{Q(t)} & & \overline{G(t)} & & \overline{Q(t)} \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\overbrace{O(t)} & & B(t) & & \overbrace{O(t)} & & B(t) \\
\vdots & & & & \vdots & &
\end{array}$$

$$\begin{array}{ccc}
C(t) & & \overbrace{H(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{P(t)} & & \overline{\overline{F_1(t) \cup F_2(t)}} \\
\uparrow & & \uparrow \\
\overbrace{R(t)} & \text{Wprt}(t) & \overbrace{A(t)} \\
\uparrow & & \uparrow \\
\overline{\overline{G(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{O(t)} & & B(t) \\
\vdots & &
\end{array} \quad (\text{A.1.2.2.10}).$$

We consider the following self-type dynamic Wprt-structures of the first type:

$$\begin{array}{ccc}
Q(t) & & \overbrace{Q(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & \text{Wprt}(t) & \overbrace{Q(t)} \quad (\text{A.1.2.3}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & & Q(t) \\
\vdots & &
\end{array}$$

$$\begin{array}{ccc}
Q(t) & & \overbrace{A(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & \text{Wprt}(t) & \overbrace{Q(t)} \quad (\text{A.1.2.4}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{A(t)} & & Q(t) \\
\vdots & &
\end{array}$$

$$\begin{array}{ccc}
B(t) & & \overbrace{Q(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & \text{Wprt}(t) & \overbrace{Q(t)} \quad (\text{A.1.2.5}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & & B(t) \\
\vdots & & \\
A(t) & & \overbrace{A(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & \text{Wprt}(t) & \overbrace{Q(t)} \quad (\text{A.1.2.6}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{A(t)} & & A(t) \\
\vdots & & \\
a(t) & & \overbrace{str A(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & \text{Wprt}(t) & \overbrace{Q(t)} \quad (\text{A.1.2.6.1}), \\
\uparrow & & \uparrow \\
\overbrace{str A(t)} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\vdots & & a(t)
\end{array}$$

denote $W_{11}(t)fA(t); Q(t); a(t), a(t) \subset A(t)$,

$$\begin{array}{ccc}
\text{str}A(t) & & \overbrace{a(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & \text{Wprt}(t) & \overbrace{Q(t)} \quad (\text{A.1.2.6.2}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{a(t)} & & \text{str}A(t) \\
\vdots & &
\end{array}$$

denote $W_{12}(t)fa(t); Q(t); A(t), a(t) \subset A(t)$,

$$\begin{array}{ccc}
B(t) & & \overbrace{A(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & \text{Wprt}(t) & \overbrace{A(t)} \quad (\text{A.1.2.7}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{B(t)} & & B(t) \\
\vdots & &
\end{array}$$

and any other possible options of self for (A.1.2.1) etc.

It can be considered a simpler version of the dynamic operator

$$\begin{array}{c}
\overbrace{H(t)}^{\dots} \\
\uparrow \\
\overline{\overline{F(t)}} \\
\uparrow \\
\text{Wprt}(t) \overbrace{A(t)} \quad (\text{A.1.2.8}) \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B(t)
\end{array}$$

where $\overbrace{H(t)}^{\dots}$ - paradoxical upper level of $H(t)$, $\overline{\overline{F(t)}}$ - paradoxical average level of $F(t)$, $\overbrace{A(t)}$ - upper levels of $A(t)$, $\overline{Q(t)}$ - average levels of $Q(t)$, $B(t)$ goes to the

middle level of $Q(t) - \overline{Q(t)}, \overline{Q(t)}$ goes to the upper level of $A(t) - \widetilde{A(t)}, \widetilde{A(t)}$
goes to the paradoxical middle level of $F(t) - \overline{\overline{F(t)}}, \overline{\overline{F(t)}}$ goes to the paradoxical
upper level of $H(t) - \widetilde{\widetilde{H(t)}}$ simultaneously, the result of this process will be
described by the expression

$$\begin{array}{c}
 \widetilde{\widetilde{H(t)}} \\
 \uparrow \\
 \overline{\overline{F(t)}} \\
 \uparrow \\
 \text{Wrt}(t) \widetilde{A(t)} \quad (\text{A.1.2.9}), \\
 \uparrow \\
 \overline{Q(t)} \\
 \uparrow \\
 B(t)
 \end{array}$$

or

$$\begin{array}{c}
 C(t) \\
 \uparrow \\
 \overline{P(t)} \\
 \uparrow \\
 \widetilde{R(t)} \\
 \uparrow \quad \text{Wprt}(t) \quad (\text{A.1.2.10}), \\
 \overline{\overline{G(t)}} \\
 \uparrow \\
 \widetilde{\widetilde{O(t)}} \\
 \dots
 \end{array}$$

where $\widetilde{\widetilde{O(t)}}$ - paradoxical upper levels of $O(t)$, $\overline{\overline{G(t)}}$ - paradoxical average levels
of $G(t)$, $\widetilde{R(t)}$ - upper levels of $R(t)$, $\overline{P(t)}$ - average levels of $P(t)$, $\widetilde{\widetilde{O(t)}}$ goes to
the paradoxical middle level of $G(t) - \overline{\overline{G(t)}}, \overline{\overline{G(t)}}$ goes to the upper level of $R(t)$
- $\widetilde{R(t)}$, $\widetilde{R(t)}$ goes to the middle level of $P(t) - \overline{P(t)}$, $\overline{P(t)}$ goes to the lower level
of $C(t)$ simultaneously, the result of this process will be described by the
expression

$$\begin{array}{c}
C(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overbrace{R(t)} \\
\uparrow \text{ Wrt}(t) \text{ (A.1.2.11),} \\
\overline{\overline{G(t)}} \\
\uparrow \\
\overbrace{\overline{O(t)}} \\
\ldots
\end{array}$$

Definition A.1.2.2. The dynamic operator (A.1.2.8) we shall call dynamic Wprt – element of the second type, (A.1.2.9) we shall call dynamic Wrt – element of the second type.

It's allowed to add dynamic Wprt – elements of the second type:

$$\begin{array}{c}
\overbrace{H(t)} \\
\uparrow \\
\overline{F(t)} \\
\uparrow \\
\text{Wprt}(t) \overbrace{A_1(t)} + \text{Wprt}(t) \overbrace{A_2(t)} = \text{Wprt}(t) \overbrace{A_1(t) \cup A_2(t)} \text{ (A.1.2.12),} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B(t)
\end{array}$$

$$\begin{array}{c}
\overbrace{H(t)} \\
\uparrow \\
\overline{F(t)} \\
\uparrow \\
\text{Wprt}(t) \overbrace{A(t)} + \text{Wprt}(t) \overbrace{A(t)} = \text{Wprt}(t) \overbrace{A(t)} \text{ (A.1.2.13),} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B_1(t)
\end{array}$$

$$\begin{array}{ccccc}
\begin{array}{c} \overbrace{H(t)}^{\dots} \\ \uparrow \\ \overline{\overline{F(t)}} \\ \uparrow \\ \overbrace{A(t)} \\ \uparrow \\ \overline{Q_1(t)} \\ \uparrow \\ B(t) \end{array} & + & \begin{array}{c} \overbrace{H(t)}^{\dots} \\ \uparrow \\ \overline{\overline{F(t)}} \\ \uparrow \\ \overbrace{A(t)} \\ \uparrow \\ \overline{Q_2(t)} \\ \uparrow \\ B(t) \end{array} & = & \begin{array}{c} \overbrace{H(t)}^{\dots} \\ \uparrow \\ \overline{\overline{F(t)}} \\ \uparrow \\ \overbrace{A(t)} \\ \uparrow \\ \overline{Q_1(t) \cup Q_2(t)} \\ \uparrow \\ B(t) \end{array} & (A.1.2.13.1),
\end{array}$$

$$\begin{array}{ccccc}
\begin{array}{c} \overbrace{H_1(t)}^{\dots} \\ \uparrow \\ \overline{\overline{F(t)}} \\ \uparrow \\ \overbrace{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & + & \begin{array}{c} \overbrace{H_2(t)}^{\dots} \\ \uparrow \\ \overline{\overline{F(t)}} \\ \uparrow \\ \overbrace{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & = & \begin{array}{c} \overbrace{H_1(t) \cup H_2(t)}^{\dots} \\ \uparrow \\ \overline{\overline{F(t)}} \\ \uparrow \\ \overbrace{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & (A.1.2.13.2),
\end{array}$$

$$\begin{array}{ccccc}
\begin{array}{c} \overbrace{H(t)}^{\dots} \\ \uparrow \\ \overline{\overline{F_1(t)}} \\ \uparrow \\ \overbrace{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & + & \begin{array}{c} \overbrace{H(t)}^{\dots} \\ \uparrow \\ \overline{\overline{F_2(t)}} \\ \uparrow \\ \overbrace{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & = & \begin{array}{c} \overbrace{H(t)}^{\dots} \\ \uparrow \\ \overline{\overline{F_1(t) \cup F_2(t)}} \\ \uparrow \\ \overbrace{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & (A.1.2.13.3),
\end{array}$$

We consider the following self-type dynamic Wprt-structures of the second t type:

$$\begin{array}{c}
 \overbrace{H(t)}^{\dots} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 \text{Wprt}(t) \quad \overbrace{Q(t)}^{\dots} \quad (\text{A.1.2.14}), \\
 \uparrow \\
 \overline{Q(t)} \\
 \uparrow \\
 H(t)
 \end{array}$$

$$\begin{array}{c}
 \overbrace{strA(t)}^{\dots} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 \text{Wprt}(t) \quad \overbrace{Q(t)}^{\dots} \quad (\text{A.1.2.14.1}), \\
 \uparrow \\
 \overline{Q(t)} \\
 \uparrow \\
 a(t)
 \end{array}$$

denote $W_{13}(t) f A(t); Q(t); a(t), a(t) \subset A(t)$,

$$\begin{array}{c}
 \overbrace{a(t)}^{\dots} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 \text{Wprt}(t) \quad \overbrace{Q(t)}^{\dots} \quad (\text{A.1.2.15}), \\
 \uparrow \\
 \overline{Q(t)} \\
 \uparrow \\
 strA(t)
 \end{array}$$

denote $W_{14}(t) f a(t); Q(t); A(t), a(t) \subset A(t)$,

$$\begin{array}{c}
\overbrace{Q(t)}^{\dots} \\
\uparrow \\
\overline{\overline{Q(t)}} \\
\uparrow \\
\text{Wprt}(t) \overbrace{A(t)} \quad (\text{A.1.2.16}), \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
Q(t)
\end{array}$$

and any other possible options of self for (A.1.2.8) etc.

Definition A.1.2.3. The dynamic operator (A.1.2.10) we shall call dynamic tprW – element, (A.1.2.11) we shall call dynamic trW – element.

It's allowed to add dynamic tprW – elements:

$$\begin{array}{ccc}
\begin{array}{c} C_1(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overbrace{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overbrace{O(t)} \\ \dots \end{array} & + & \begin{array}{c} C_2(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overbrace{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overbrace{O(t)} \\ \dots \end{array} \\
\text{Wprt}(t) & & \text{Wprt}(t) =
\end{array}
\begin{array}{c} C_1(t) \cup C_2(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overbrace{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overbrace{O(t)} \\ \dots \end{array} \quad \text{Wprt}(t) \quad (\text{A.1.2.17}),$$

$$\begin{array}{ccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overbrace{R_1(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overbrace{O(t)} \\ \dots \end{array} & + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overbrace{R_2(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overbrace{O(t)} \\ \dots \end{array} \\
\text{Wprt}(t) & & \text{Wprt}(t) =
\end{array}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overbrace{R_1(t) \cup R_2(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overbrace{O(t)} \\ \dots \end{array} \quad \text{Wprt}(t) \quad (\text{A.1.2.18}),$$

$$\begin{array}{ccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P_1(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} & \text{Wprt}(t) + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P_2(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \\
& & \text{Wprt}(t) = & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P_1(t) \cup P_2(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \text{Wprt}(t) \text{ (A.1.2.18.1),}
\end{array}$$

$$\begin{array}{ccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G_1(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} & \text{Wprt}(t) + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G_2(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \\
& & \text{Wprt}(t) = & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G_1(t) \cup G_2(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \text{Wprt}(t) \text{ (A.1.2.18.2),}
\end{array}$$

$$\begin{array}{ccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O_1(t)}} \\ \dots \end{array} & \text{Wprt}(t) + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O_2(t)}} \\ \dots \end{array} \\
& & \text{Wprt}(t) = & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O_1(t) \cup O_2(t)}} \\ \dots \end{array} \text{Wprt}(t) \text{ (A.1.2.18.3),}
\end{array}$$

We consider the following self-type dynamic tprW-structures:

$$\begin{array}{c}
R(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overbrace{R(t)} \\
\uparrow \text{ Wprt}(t)(\text{A.1.2.19}) \\
\overline{\overline{R(t)}} \\
\uparrow \\
\overbrace{\overline{R(t)}} \\
\ldots \\
strD(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overbrace{P(t)} \\
\uparrow \text{ Wprt}(t) (\text{A.1.2.19.1}), \\
\overline{\overline{P(t)}} \\
\uparrow \\
\overbrace{\overline{d(t)}} \\
\ldots
\end{array}$$

denote $W_{15}(t)fd(t); \overline{P(t)}; D(t), d(t) \subset D(t)$,

$$\begin{array}{c}
d(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overbrace{D(t)} \\
\uparrow \text{ Wprt}(t) (\text{A.1.2.20}) \\
\overline{\overline{D(t)}} \\
\uparrow \\
\overbrace{\overline{D(t)}} \\
\ldots
\end{array}$$

denote $W_{16}(t)fD(t); \overline{P(t)}; d(t), d(t) \subset D(t)$,

$$\begin{array}{c}
P(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overline{\overline{R(t)}} \\
\uparrow \text{Wprt}(t)(A.1.2.21) \\
\overline{\overline{R(t)}} \\
\uparrow \\
\overline{\overline{R(t)}} \\
\vdots
\end{array}$$

and any other possible options of self for (A.1.2.10) etc.

New mathematical structures and operators is carried out with generalization it to any structures with any actions. For example,

$$1) \begin{array}{ccccc} f_{11} & \dots & f_{1k} & & \\ & & & q_{11} & \dots & q_{1n} \\ & & & & & \\ (q_{j1})^{-1} & \dots & (q_{jk})^{-1} & WWprt & \dots & (*) \\ & & & & & \\ & & & q_{m1} & \dots & q_{mn} \\ f_{l1} & \dots & f_{lk} & & & \end{array}$$

f_{ij}, q_{ij} – any objects, actions etc.

$$2) \begin{array}{ccccccc} & & g_{12} & & w_{11} & w_{12} & w_{1n} \\ g_{11} & & g_{22} & & & & \\ (w_{j1})^{-1} & (w_{j2})^{-1} & g_{13} & WGWprt & \dots & \dots & w_{2n} \\ g_{31} & & \dots & & w_{m1} & w_{m2} & \dots \\ & & g_{k2} & & & & w_{sn} \\ & & & & & & w_{ml} \end{array} (*)_1,$$

w_{ij}, g_{ij} – any objects, actions etc.

3)

$$\begin{array}{ccc} a & b & g \\ c & AWWrq(\mu) & w \quad (*)_2 \\ d & q & r \end{array}$$

where $AWWrq$ is virtual structure or virtual operator, which can take any form of action; $a, c, d, q, r, w, g, b, \mu$ – any objects, actions etc.

Accordingly, we can consider all sorts of self-type structures for 1) – 3).

And any other possible structures and operators etc.

A.2 FWprt – elements, self-type FWprt-structures

We consider fuzzy dynamic operator

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{FWprt} \\
\bar{G} \\
\uparrow \\
\tilde{O} \\
\vdots
\end{array}
\begin{array}{c}
\vdots \\
\tilde{\tilde{H}} \\
\uparrow \\
\bar{\bar{F}} \\
\uparrow \\
\tilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.2.11}),$$

where $\tilde{\tilde{H}}, \tilde{\tilde{O}}$ - paradoxical upper levels of fuzzy H and fuzzy O respectively, $\bar{\bar{F}}, \bar{\bar{G}}$ - paradoxical average levels of fuzzy F and fuzzy G respectively, $\tilde{A}, \tilde{R}$ - upper levels of fuzzy A and fuzzy R respectively, $\bar{Q}, \bar{P}$ - average levels of fuzzy Q and fuzzy P respectively, fuzzy B goes to the middle level of fuzzy Q - $\bar{Q}$, $\bar{Q}$ goes to the upper level of fuzzy A - $\tilde{A}$, $\tilde{A}$ goes to the paradoxical middle level of fuzzy F - $\bar{\bar{F}}, \bar{\bar{F}}$ goes to the paradoxical upper level of fuzzy H - $\tilde{\tilde{H}}, \tilde{\tilde{O}}$ goes to the paradoxical middle level of fuzzy G - $\bar{\bar{G}}, \bar{\bar{G}}$ goes to the upper level of fuzzy R - $\tilde{R}$, $\tilde{R}$ goes to the middle level of fuzzy P - $\bar{P}$, $\bar{P}$ goes to the lower level of fuzzy C simultaneously. The result of this process will be described by the expression

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{FWprt} \\
\bar{G} \\
\uparrow \\
\tilde{O} \\
\vdots
\end{array}
\begin{array}{c}
\vdots \\
\tilde{\tilde{H}} \\
\uparrow \\
\bar{\bar{F}} \\
\uparrow \\
\tilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.2.12}).$$

Definition A.2.11. The fuzzy dynamic operator (A.2.11) we shall call FWprt – element of the first type, (A.2.12) we shall call FWrt – element of the first type.

Remark A.2.11 Can consider FWprt – elements use the Banach space.

It's allowed to add Wprt – elements:

$$\begin{array}{ccccc}
\begin{array}{c} C \\ \uparrow \\ \bar{P} \\ \uparrow \\ \bar{R} \\ \uparrow \\ \bar{G} \\ \uparrow \\ \bar{O} \end{array} & \begin{array}{c} \bar{\bar{H}}_1 \\ \uparrow \\ \bar{F} \\ \uparrow \\ \bar{A} \\ \uparrow \\ \bar{Q} \\ \uparrow \\ B \end{array} & + & \begin{array}{c} C \\ \uparrow \\ \bar{P} \\ \uparrow \\ \bar{R} \\ \uparrow \\ \bar{G} \\ \uparrow \\ \bar{O} \end{array} & \begin{array}{c} \bar{\bar{H}}_2 \\ \uparrow \\ \bar{F} \\ \uparrow \\ \bar{A} \\ \uparrow \\ \bar{Q} \\ \uparrow \\ B \end{array} \\
\text{FWprt} & & \text{FWprt} & = & \text{FWprt}
\end{array} \quad (A.2.12.1),$$

$$\begin{array}{c}
\begin{array}{c}
\tilde{C} \uparrow \tilde{P} \uparrow \tilde{R} \uparrow \tilde{G} \uparrow \tilde{O} \\
\text{FWprt}
\end{array}
+
\begin{array}{c}
\tilde{H} \uparrow \tilde{F} \uparrow \tilde{A} \uparrow \tilde{Q} \uparrow \tilde{O} \\
\text{FWprt}
\end{array}
=
\begin{array}{c}
\tilde{C} \uparrow \tilde{P} \uparrow \tilde{R} \uparrow \tilde{G} \uparrow \tilde{O} \\
\text{FWprt}
\end{array}
\quad (\text{A.2.12.2}),
\end{array}$$

$$\begin{array}{ccccccc}
C_1 & & C_2 & & C_1 \cup C_2 & & \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\bar{P} & & \bar{P} & & \bar{P} & & \bar{P} \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\tilde{R} & & \tilde{R} & & \tilde{R} & & \tilde{R} \\
\uparrow & \text{FWprt} & \uparrow & + & \uparrow & \text{FWprt} & \uparrow \\
\overline{\overline{G}} & & \overline{\overline{G}} & & \overline{\overline{G}} & & \overline{\overline{G}} \\
\uparrow & & \uparrow & & \uparrow & & \uparrow \\
\tilde{O} & & \tilde{O} & & \tilde{O} & & \tilde{O} \\
& & B & & B & & B
\end{array}
\quad (A.2.12.3),$$

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R_1} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{O}
\end{array}
\begin{array}{c}
\widetilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R_2} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{O}
\end{array}
\begin{array}{c}
\widetilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\overbrace{\widetilde{R_1} \cup \widetilde{R_2}} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{O}
\end{array}
\begin{array}{c}
\widetilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.2.12.4}),$$

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P}_1 \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{O}
\end{array}
\begin{array}{c}
\widetilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
C \\
\uparrow \\
\bar{P}_2 \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{O}
\end{array}
\begin{array}{c}
\widetilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
C \\
\uparrow \\
\overline{\bar{P}_1} \cup \overline{\bar{P}_2} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{O}
\end{array}
\begin{array}{c}
\widetilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.2.12.5}),$$

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{O}
\end{array}
\begin{array}{c}
\widetilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\overline{Q_1} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{O}
\end{array}
\begin{array}{c}
\widetilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\overline{Q_2} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\widetilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\widetilde{O}
\end{array}
\begin{array}{c}
\widetilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\widetilde{A} \\
\uparrow \\
\overline{Q_1} \cup \overline{Q_2} \\
\uparrow \\
B
\end{array}
\quad (\text{A.2.12.6}),$$

(A.2.12.7),

(A.2.12.8),

(A.2.12.9),

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\tilde{O}
\end{array}
\text{FWprt}
\begin{array}{c}
\tilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\tilde{A}_1 \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\tilde{O}
\end{array}
\text{FWprt}
\begin{array}{c}
\tilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\tilde{A}_2 \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \\
\overline{\overline{G}} \\
\uparrow \\
\tilde{O}
\end{array}
\text{FWprt}
\begin{array}{c}
\tilde{H} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\tilde{A}_1 \cup \tilde{A}_2 \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.2.12.10}).$$

We consider the following self-type FWprt-structure of the first type:

$$\begin{array}{c}
Q \\
\uparrow \\
\bar{Q} \\
\uparrow \\
\tilde{Q} \\
\uparrow \\
\overline{\overline{Q}} \\
\uparrow \\
\tilde{Q}
\end{array}
\text{FWprt}
\begin{array}{c}
\tilde{Q} \\
\uparrow \\
\overline{\overline{Q}} \\
\uparrow \\
\tilde{Q} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
Q
\end{array}
\quad (\text{A.2.13}),$$

denote $FW_1 f Q$.

$$\begin{array}{c}
Q \\
\uparrow \\
\bar{Q} \\
\uparrow \\
\tilde{Q} \\
\uparrow \\
\overline{\overline{Q}} \\
\uparrow \\
\tilde{A}
\end{array}
\text{FWprt}
\begin{array}{c}
\tilde{A} \\
\uparrow \\
\overline{\overline{Q}} \\
\uparrow \\
\tilde{Q} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
Q
\end{array}
\quad (\text{A.2.14}),$$

denote $FW_2 f A; Q$.

$$\begin{array}{ccc}
 B & & \dots \\
 \uparrow & & \tilde{\tilde{H}} \\
 \bar{Q} & & \uparrow \\
 \uparrow & & \overline{\overline{F}} \\
 \tilde{\tilde{A}} & & \uparrow \\
 \uparrow & \text{FWprt} & \tilde{\tilde{A}} \quad (\text{A.2.15}), \\
 \overline{\overline{F}} & & \uparrow \\
 \uparrow & & \bar{Q} \\
 \tilde{\tilde{H}} & & \uparrow \\
 \dots & & B
 \end{array}$$

denote $FW_3 f H, F, A; Q; B$.

$$\begin{array}{ccc}
 A & & \dots \\
 \uparrow & & \tilde{\tilde{A}} \\
 \bar{Q} & & \uparrow \\
 \uparrow & & \overline{\overline{Q}} \\
 \tilde{\tilde{Q}} & & \uparrow \\
 \uparrow & \text{FWprt} & \tilde{\tilde{Q}} \quad (\text{A.2.16}), \\
 \overline{\overline{Q}} & & \uparrow \\
 \uparrow & & \bar{Q} \\
 \tilde{\tilde{A}} & & \uparrow \\
 \dots & & A
 \end{array}$$

denote $FW_4 f A; Q$.

$$\begin{array}{ccc}
 a & & \dots \\
 \uparrow & & \text{str } \tilde{\tilde{A}} \\
 \bar{Q} & & \uparrow \\
 \uparrow & & \overline{\overline{Q}} \\
 \overbrace{\bar{Q}} & & \uparrow \\
 \uparrow & \text{FWprt} & \tilde{\tilde{Q}} \quad (\text{A.2.16.1}), \\
 \overline{\overline{Q}} & & \uparrow \\
 \uparrow & & \bar{Q} \\
 \text{str } \tilde{\tilde{A}} & & \uparrow \\
 \dots & & a
 \end{array}$$

denote $FW_5 f A; Q; a, a \subset A$,

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{FWprt} \\
\bar{G} \\
\uparrow \\
\tilde{O} \\
\vdots
\end{array}
\begin{array}{c}
\vdots \\
\tilde{\tilde{H}} \\
\uparrow \\
\bar{\bar{F}} \\
\uparrow \\
\tilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.2.16.2}),$$

denote $FW_6 f a; Q; A, a \subset A$,

$$\begin{array}{c}
B \\
\uparrow \\
\bar{Q} \\
\uparrow \\
\tilde{Q} \\
\uparrow \text{FWprt} \\
\bar{Q} \\
\uparrow \\
\tilde{B} \\
\vdots
\end{array}
\begin{array}{c}
\vdots \\
\tilde{\tilde{A}} \\
\uparrow \\
\bar{\bar{Q}} \\
\uparrow \\
\tilde{Q} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.2.17}),$$

and any other possible options of self for (A.2.11) etc.

It can be considered a simpler version of the fuzzy dynamic operator

$$\begin{array}{c}
\vdots \\
\tilde{\tilde{H}} \\
\uparrow \\
\bar{\bar{F}} \\
\uparrow \text{FWprt} \\
\tilde{A} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\quad (\text{A.2.18})$$

where $\tilde{\tilde{H}}$ - paradoxical upper level of fuzzy H, $\bar{\bar{F}}$ - paradoxical average level of fuzzy F, $\tilde{A}$ - upper levels of fuzzy A, $\bar{Q}$ - average levels of fuzzy Q, fuzzy B goes

to the middle level of fuzzy Q - $\bar{Q}$, $\bar{Q}$ goes to the upper level of fuzzy A - $\tilde{A}$, $\tilde{A}$ goes to the paradoxical middle level of fuzzy F - $\overline{\overline{F}}$, $\overline{\overline{F}}$ goes to the paradoxical upper level of fuzzy H - $\tilde{\tilde{H}}$ simultaneously, the result of this process will be described by the expression

$$\begin{array}{c} \dots \\ \tilde{\tilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \text{FWrt } \tilde{A} \quad (\text{A.2.19}) \\ \uparrow \\ \bar{Q} \\ \uparrow \\ B \end{array}$$

or

$$\begin{array}{c} C \\ \uparrow \\ \bar{P} \\ \uparrow \\ \tilde{R} \\ \uparrow \text{FWprt } (\text{A.2.110}) \\ \overline{\overline{G}} \\ \uparrow \\ \tilde{\tilde{O}} \\ \dots \end{array}$$

where $\tilde{\tilde{O}}$ - paradoxical upper levels of fuzzy O, $\overline{\overline{G}}$ - paradoxical average levels of fuzzy G, $\tilde{A}$, $\tilde{R}$ - upper levels of fuzzy A and fuzzy R respectively, $\bar{Q}$, $\bar{P}$ - average levels of fuzzy Q and fuzzy P respectively, $\tilde{R}$ - upper levels of fuzzy R, $\bar{P}$ - average levels of fuzzy P, $\tilde{\tilde{O}}$ goes to the paradoxical middle level of fuzzy G - $\overline{\overline{G}}$, $\overline{\overline{G}}$ goes to the upper level of fuzzy R - $\tilde{R}$, $\tilde{R}$ goes to the middle level of fuzzy P - $\bar{P}$, $\bar{P}$ goes to the lower level of fuzzy C simultaneously, the result of this process will be described by the expression

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{FWrt (A.2.111)} \\
\overline{\overline{G}} \\
\uparrow \\
\tilde{\tilde{O}} \\
\ldots
\end{array}$$

Definition 1,2. The fuzzy dynamic operator (A.2.18) we shall call FWprt – element of the second type, (A.2.19) we shall call FWrt – element of the second type.

It's allowed to add FWprt – elements of the second type:

$$\begin{array}{c}
\begin{array}{c} \dots \\ \tilde{\tilde{H}}_1 \end{array} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\text{FWprt } \tilde{\tilde{A}} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
+
\begin{array}{c}
\begin{array}{c} \dots \\ \tilde{\tilde{H}}_2 \end{array} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\tilde{\tilde{A}} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
=
\begin{array}{c}
\begin{array}{c} \dots \\ \tilde{\tilde{H}}_1 \cup \tilde{\tilde{H}}_2 \end{array} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\tilde{\tilde{A}} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B
\end{array}
\end{array}
\quad (\text{A.2.112}),$$

$$\begin{array}{c}
\begin{array}{c} \dots \\ \tilde{\tilde{H}} \end{array} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\text{FWprt } \tilde{\tilde{A}} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B_1
\end{array}
+
\begin{array}{c}
\begin{array}{c} \dots \\ \tilde{\tilde{H}} \end{array} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\tilde{\tilde{A}} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B_2
\end{array}
=
\begin{array}{c}
\begin{array}{c} \dots \\ \tilde{\tilde{H}} \end{array} \\
\uparrow \\
\overline{\overline{F}} \\
\uparrow \\
\tilde{\tilde{A}} \\
\uparrow \\
\bar{Q} \\
\uparrow \\
B_1 \cup B_2
\end{array}
\end{array}
\quad (\text{A.2.113}),$$

$$\begin{array}{c}
 \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q_1} \\ \uparrow \\ B \end{array} \\
 \text{FWprt}
 \end{array}
 +
 \begin{array}{c}
 \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q_2} \\ \uparrow \\ B \end{array} \\
 \text{FWprt}
 \end{array}
 =
 \begin{array}{c}
 \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q_1 \cup Q_2} \\ \uparrow \\ B \end{array} \\
 \text{FWprt}
 \end{array}
 \quad (\text{A.2.113.1}),$$

$$\begin{array}{c}
 \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F_1}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} \\
 \text{FWprt}
 \end{array}
 +
 \begin{array}{c}
 \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F_2}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} \\
 \text{FWprt}
 \end{array}
 =
 \begin{array}{c}
 \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F_1 \cup F_1}} \\ \uparrow \\ \widetilde{\widetilde{A}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} \\
 \text{FWprt}
 \end{array}
 \quad (\text{A.2.113.2}),$$

$$\begin{array}{c}
 \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A_1}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} \\
 \text{FWprt}
 \end{array}
 +
 \begin{array}{c}
 \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A_2}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} \\
 \text{FWprt}
 \end{array}
 =
 \begin{array}{c}
 \begin{array}{c} \dots \\ \widetilde{\widetilde{H}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \widetilde{\widetilde{A_1 \cup A_2}} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ B \end{array} \\
 \text{FWprt}
 \end{array}
 \quad (\text{A.2.113.3}),$$

We consider the following self-type FWprt-structures of the second type:

$$\begin{array}{c}
 \vdots \\
 \tilde{\tilde{A}} \\
 \uparrow \\
 \overline{\overline{Q}} \\
 \uparrow \\
 \text{FWprt } \tilde{\tilde{Q}} \text{ (A.2.114),} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 A
 \end{array}$$

$$\begin{array}{c}
 \vdots \\
 \tilde{\tilde{A}} \\
 \uparrow \\
 \overline{\overline{A}} \\
 \uparrow \\
 \text{FWprt } \tilde{\tilde{A}} \text{ (A.2.114.1),} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 a
 \end{array}$$

denote $W_7 f A; Q; a, a \subset A$,

$$\begin{array}{c}
 \vdots \\
 \tilde{\tilde{a}} \\
 \uparrow \\
 \overline{\overline{a}} \\
 \uparrow \\
 \text{FWprt } \tilde{\tilde{a}} \text{ (A.2.115),} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 str A
 \end{array}$$

denote $W_8 f a; Q; A, a \subset A$,

$$\begin{array}{c}
 \vdots \\
 \overline{\overline{A}} \\
 \uparrow \\
 \overline{\overline{A}} \\
 \uparrow \\
 \text{FWprt } \overline{\overline{A}} \text{ (A.2.116),} \\
 \uparrow \\
 \overline{Q} \\
 \uparrow \\
 Q
 \end{array}$$

and any other possible options of self for (A.2.18) etc.

Definition A.2.13. The fuzzy dynamic operator (A.2.110) we shall call tprFW – element, (A.2.111) we shall call trFW – element.

It's allowed to add tprFW – elements:

$$\begin{array}{ccc}
 C_1 & C_2 & C_1 \cup C_2 \\
 \uparrow & \uparrow & \uparrow \\
 \overline{P} & \overline{P} & \overline{P} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{R} & \overline{R} & \overline{R} \\
 \uparrow \text{ FWprt + } \uparrow \text{ FWprt = } & & \uparrow \text{ FWprt (A.2.117),} \\
 \overline{\overline{G}} & \overline{\overline{G}} & \overline{\overline{G}} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{O}} & \overline{\overline{O}} & \overline{\overline{O}} \\
 \dots & \dots & \dots
 \end{array}$$

$$\begin{array}{ccc}
 C & C & C \\
 \uparrow & \uparrow & \uparrow \\
 \overline{P} & \overline{P} & \overline{P} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{R} & \overline{R} & \overline{R} \\
 \uparrow \text{ FWprt + } \uparrow \text{ FWprt = } & & \uparrow \text{ FWprt (A.2.117.1),} \\
 \overline{\overline{G}} & \overline{\overline{G}} & \overline{\overline{G}} \\
 \uparrow & \uparrow & \uparrow \\
 \overline{\overline{O}}_1 & \overline{\overline{O}}_2 & \overline{\overline{O}}_1 \cup \overline{\overline{O}}_2 \\
 \dots & \dots & \dots
 \end{array}$$

$$\begin{array}{ccc}
\begin{array}{c} C \\ \uparrow \\ \overline{P_1} \\ \uparrow \\ \overline{R} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \overline{O} \\ \dots \end{array} & + \begin{array}{c} C \\ \uparrow \\ \overline{P_2} \\ \uparrow \\ \overline{R} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \overline{O} \\ \dots \end{array} & = \begin{array}{c} C \\ \uparrow \\ (\overline{P_1} \cup \overline{P_2}) \\ \uparrow \\ \overline{R} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \overline{O} \\ \dots \end{array} \quad \text{FWprt (A.2.117.2),}
\end{array}$$

$$\begin{array}{ccc}
\begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overline{R_1} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \overline{O} \\ \dots \end{array} & + \begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overline{R_2} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \overline{O} \\ \dots \end{array} & = \begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overline{R_1} \cup \overline{R_2} \\ \uparrow \\ \overline{G} \\ \uparrow \\ \overline{O} \\ \dots \end{array} \quad \text{FWprt (A.2.117.3),}
\end{array}$$

$$\begin{array}{ccc}
\begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overline{R} \\ \uparrow \\ \overline{G_1} \\ \uparrow \\ \overline{O} \\ \dots \end{array} & + \begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overline{R} \\ \uparrow \\ \overline{G_2} \\ \uparrow \\ \overline{O} \\ \dots \end{array} & = \begin{array}{c} C \\ \uparrow \\ \overline{P} \\ \uparrow \\ \overline{R} \\ \uparrow \\ \overline{G_1} \cup \overline{G_2} \\ \uparrow \\ \overline{O} \\ \dots \end{array} \quad \text{FWprt (A.2.117.4).}
\end{array}$$

We consider the following self-type tprFW-structures:

$$\begin{array}{c}
O \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{FWprt (A.2.118),} \\
\overline{\overline{O}} \\
\uparrow \\
\tilde{\tilde{O}}
\end{array}$$

$$\begin{array}{c}
\dots \\
str D \\
\uparrow \\
\bar{Q} \\
\uparrow \\
\tilde{Q} \\
\uparrow \text{FWprt ((A.2.118.1),} \\
\overline{\overline{Q}} \\
\uparrow \\
\tilde{\tilde{d}} \\
\dots
\end{array}$$

denote $FW_9 f d; Q; D, d \subset D$,

$$\begin{array}{c}
d \\
\uparrow \\
\bar{Q} \\
\uparrow \\
\overbrace{Q} \\
\uparrow \text{FWprt (A.2.119),} \\
\overline{\overline{D}} \\
\uparrow \\
str \tilde{\tilde{D}} \\
\dots
\end{array}$$

denote $FW_{10} f D; Q; d, d \subset D$,

$$\begin{array}{c}
P \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{FWprt (A.2.120),} \\
\bar{G} \\
\uparrow \\
\tilde{O} \\
\ldots
\end{array}$$

$$\begin{array}{c}
R \\
\uparrow \\
\bar{P} \text{FWprt (A.2.121)} \\
\uparrow \\
\tilde{R} \\
\text{str} D \\
\uparrow \\
\bar{P} \text{FWprt (A.2.121.1),} \\
\uparrow \\
\tilde{d}
\end{array}$$

denote $FW_9 f d; Q; D, d \subset D$,

$$\begin{array}{c}
d \\
\uparrow \\
\bar{P} \text{FWprt (A.2.122),} \\
\uparrow \\
\text{str } \tilde{D}
\end{array}$$

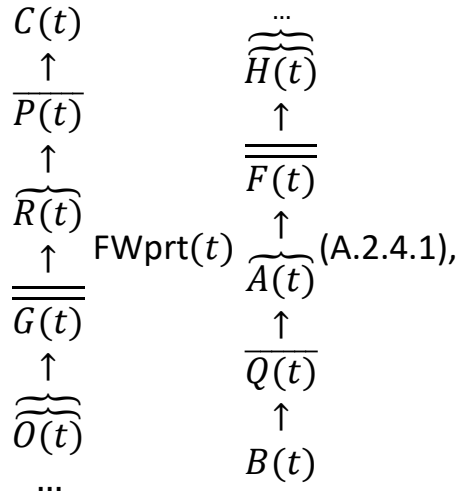
denote $FW_{10} f D; Q; d, d \subset D$,

$$\begin{array}{c}
P \\
\uparrow \\
\bar{P} \text{FWprt (A.2.123)} \\
\uparrow \\
\tilde{R}
\end{array}$$

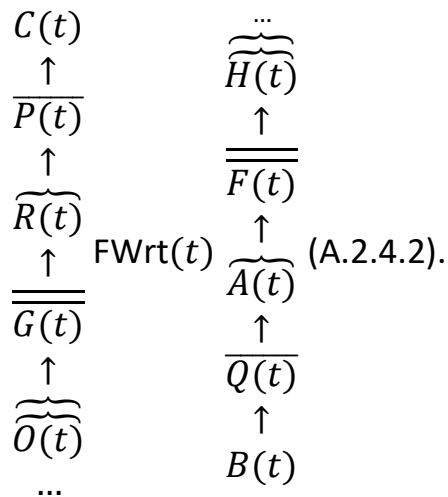
and any other possible options of self for (A.2.110) etc.

4 Dynamic FWprt – elements, self-type dynamic FWprt-structures

We considered FWprt – elements earlier. Here we consider dynamic FWprt – elements. We consider fuzzy dynamic operator whose elements change over time



where $\overbrace{H(t)}$, $\overbrace{O(t)}$ - paradoxical upper levels of fuzzy $H(t)$ and fuzzy $O(t)$ respectively, $\overline{\overline{F(t)}}$, $\overline{\overline{G(t)}}$ - paradoxical average levels of fuzzy $F(t)$ and fuzzy $G(t)$ respectively, $\overbrace{A(t)}$, $\overbrace{R(t)}$ - upper levels of fuzzy $A(t)$ and fuzzy $R(t)$ respectively, $\overline{Q(t)}$, $\overline{P(t)}$ - average levels of fuzzy $Q(t)$ and fuzzy $P(t)$ respectively, fuzzy $B(t)$ goes to the middle level of fuzzy $Q(t)$ - $\overline{Q(t)}$, $\overline{Q(t)}$ goes to the upper level of fuzzy $A(t)$ - $\overbrace{A(t)}$, $\overbrace{A(t)}$ goes to the paradoxical middle level of fuzzy $F(t)$ - $\overline{\overline{F(t)}}$, $\overline{\overline{F(t)}}$ goes to the paradoxical upper level of fuzzy $H(t)$ - $\overbrace{H(t)}$, $\overbrace{O(t)}$ goes to the paradoxical middle level of fuzzy $G(t)$ - $\overline{\overline{G(t)}}$, $\overline{\overline{G(t)}}$ goes to the upper level of fuzzy $R(t)$ - $\overbrace{R(t)}$, $\overbrace{R(t)}$ goes to the middle level of fuzzy $P(t)$ - $\overline{P(t)}$, $\overline{P(t)}$ goes to the lower level of fuzzy $C(t)$ simultaneously. The result of this process will be described by the expression



Definition A.2.4.1. The fuzzy dynamic operator (A.2.4.1) we shall call dynamic FWprt – element of the first type, (A.2.4.2) we shall call dynamic FWrt – element of the first type.

It's allowed to add dynamic FWprt – elements:

$$\begin{array}{ccccccc}
 \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H_1(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H_2(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & = & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H_1(t) \cup H_2(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} \\
 \text{(A.2.4.2.1),} & & & & & & &
 \end{array}$$

$$\begin{array}{ccccccc}
 \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B_1(t) \end{array} & + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B_1(t) \end{array} & = & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B_1(t) \cup B_2(t) \end{array} \\
 \text{(A.2.4.2.2),} & & & & & & &
 \end{array}$$

$$\begin{array}{ccccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q_1(t)} \\ \uparrow \\ B(t) \end{array} & + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & = & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q_1(t) \cup Q_2(t)} \\ \uparrow \\ B(t) \end{array} \\
& \text{FWprt}(t) & & \text{FWprt}(t) & & \text{FWprt}(t) &
\end{array}$$

(A.2.4.2.6),

$$\begin{array}{ccccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A_1(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & = & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H(t)}} \\ \uparrow \\ \overline{F(t)} \\ \uparrow \\ \overline{A_1(t) \cup A_2(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} \\
& \text{FWprt}(t) & & \text{FWprt}(t) & & \text{FWprt}(t) &
\end{array}$$

(A.2.4.2.7),

$$\begin{array}{ccccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{H(t)}} \\ \uparrow \\ \overline{F_1(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} & + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{G(t)} \\ \uparrow \\ \overline{O(t)} \\ \dots \end{array} & = & \begin{array}{c} \overline{\overline{H(t)}} \\ \uparrow \\ \overline{F_2(t)} \\ \uparrow \\ \overline{A(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ B(t) \end{array} \\
& \text{FWprt}(t) & & \text{FWprt}(t) & & \text{FWprt}(t)
\end{array}$$

(A.2.4.2.10),

We consider the following self-type dynamic FWprt-structures of the first type:

$$\begin{array}{cc}
\begin{array}{c} Q(t) \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{Q(t)}} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ Q(t) \end{array} \\
& \text{FWprt}(t) \quad (A.2.4.3),
\end{array}$$

$$\begin{array}{cc}
\begin{array}{c} Q(t) \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \dots \end{array} & \begin{array}{c} \overline{\overline{A(t)}} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ \overline{Q(t)} \\ \uparrow \\ Q(t) \end{array} \\
& \text{FWprt}(t) \quad (A.2.4.4),
\end{array}$$

$$\begin{array}{ccc}
B(t) & & \overbrace{Q(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & \text{FWprt}(t) & \overbrace{Q(t)} \quad (\text{A.2.4.5}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & & B(t) \\
\vdots & & \\
\\
A(t) & & \overbrace{A(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overbrace{Q(t)} & \text{FWprt}(t) & \overbrace{Q(t)} \quad (\text{A.2.4.6}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{A(t)} & & A(t) \\
\vdots & & \\
\\
a(t) & & \overbrace{\text{str } A(t)}^{\dots} \\
\uparrow & & \uparrow \\
\overline{Q(t)} & & \overline{\overline{Q(t)}} \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & \text{FWprt}(t) & \overbrace{Q(t)} \quad (\text{A.2.4.6.1}), \\
\uparrow & & \uparrow \\
\overline{\overline{Q(t)}} & & \overline{Q(t)} \\
\uparrow & & \uparrow \\
\overbrace{\text{str } A(t)} & & a(t) \\
\vdots & &
\end{array}$$

denote $FW_{11}(t) f A(t); Q(t); a(t), a(t) \subset A(t)$,

$$\begin{array}{ccc}
\begin{array}{c}
strA(t) \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
\overline{a(t)} \\
\vdots
\end{array}
&
FWprt(t) &
\begin{array}{c}
\vdots \\
\overline{a(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
strA(t)
\end{array}
\end{array}
\quad (A.2.4.6.2),$$

denote $FW_{12}(t)fa(t); Q(t); A(t), a(t) \subset A(t)$,

$$\begin{array}{ccc}
\begin{array}{c}
B(t) \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
\overline{B(t)} \\
\vdots
\end{array}
&
FWprt(t) &
\begin{array}{c}
\vdots \\
\overline{A(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
\overline{A(t)} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B(t)
\end{array}
\end{array}
\quad (A.2.4.7),$$

and any other possible options of self for (A.2.4.1) etc.

It can be considered a simpler version of the fuzzy dynamic operator

$$\begin{array}{c}
\vdots \\
\overline{H(t)} \\
\uparrow \\
\overline{F(t)} \\
\uparrow \\
FWprt(t) \quad \overline{A(t)} \quad (A.2.4.8) \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B(t)
\end{array}$$

where $\overline{H(t)}$ - paradoxical upper level of fuzzy $H(t)$, $\overline{F(t)}$ - paradoxical average level of fuzzy $F(t)$, $\overline{A(t)}$ - upper levels of fuzzy $A(t)$, $\overline{Q(t)}$ - average levels of

fuzzy $Q(t)$, fuzzy $B(t)$ goes to the middle level of fuzzy $Q(t) - \overline{Q(t)}$, $\overline{Q(t)}$ goes to the upper level of fuzzy $A(t) - \widetilde{A(t)}$, $\widetilde{A(t)}$ goes to the paradoxical middle level of fuzzy $F(t) - \overline{\overline{F(t)}}$, $\overline{\overline{F(t)}}$ goes to the paradoxical upper level of fuzzy $H(t) - \overbrace{H(t)}$ simultaneously, the result of this process will be described by the expression

$$\begin{array}{c}
 \overbrace{H(t)}^{\dots} \\
 \uparrow \\
 \overline{\overline{F(t)}} \\
 \uparrow \\
 \text{FWrt}(t) \overbrace{A(t)}^{\dots} \text{(A.2.4.9),} \\
 \uparrow \\
 \overline{Q(t)} \\
 \uparrow \\
 B(t)
 \end{array}$$

or

$$\begin{array}{c}
 C(t) \\
 \uparrow \\
 \overline{P(t)} \\
 \uparrow \\
 \overbrace{R(t)}^{\dots} \\
 \uparrow \text{Wprt}(t) \text{(A.2.4.10),} \\
 \overline{\overline{G(t)}} \\
 \uparrow \\
 \overbrace{O(t)}^{\dots} \\
 \dots
 \end{array}$$

where $\overbrace{O(t)}$ – paradoxical upper levels of fuzzy $O(t)$, $\overline{\overline{G(t)}}$ – paradoxical average levels of fuzzy $G(t)$, $\overbrace{R(t)}$ – upper levels of fuzzy $R(t)$, $\overline{P(t)}$ – average levels of fuzzy $P(t)$, $\overbrace{O(t)}$ goes to the paradoxical middle level of fuzzy $G(t) - \overline{\overline{G(t)}}$, $\overline{\overline{G(t)}}$ goes to the upper level of fuzzy $R(t) - \overbrace{R(t)}$, $\overbrace{R(t)}$ goes to the middle level of fuzzy $P(t) - \overline{P(t)}$, $\overline{P(t)}$ goes to the lower level of fuzzy $C(t)$ simultaneously, the result of this process will be described by the expression

$$\begin{array}{c}
C(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overbrace{R(t)} \\
\uparrow \text{FWrt}(t) \text{ (A.2.4.11),} \\
\overline{\overline{G(t)}} \\
\uparrow \\
\overbrace{\overline{O(t)}} \\
\ldots
\end{array}$$

Definition A.2.4.2. The dynamic operator (A.2.4.8) we shall call dynamic FWprt – element of the second type, (A.2.4.9) we shall call dynamic FWrt – element of the second type.

It's allowed to add dynamic FWprt – elements of the second type:

$$\begin{array}{c}
\overbrace{\overline{H(t)}} \\
\uparrow \\
\overline{F(t)} \\
\uparrow \\
\text{FWprt}(t) \overbrace{A_1(t)} + \text{FWprt}(t) \overbrace{A_2(t)} = \text{FWprt}(t) \overbrace{A_1(t) \cup A_2(t)} \text{ (A.2.4.12),} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B(t)
\end{array}$$

$$\begin{array}{c}
\overbrace{\overline{H(t)}} \\
\uparrow \\
\overline{F(t)} \\
\uparrow \\
\text{FWprt}(t) \overbrace{A(t)} + \text{FWprt}(t) \overbrace{A(t)} = \text{FWprt}(t) \overbrace{A(t)} \text{ (A.2.4.13),} \\
\uparrow \\
\overline{Q(t)} \\
\uparrow \\
B_1(t)
\end{array}$$

$$\begin{array}{ccccc}
 \overbrace{\overbrace{H(t)}^{\dots}}^{\dots} & & \overbrace{\overbrace{H(t)}^{\dots}}^{\dots} & & \overbrace{\overbrace{H(t)}^{\dots}}^{\dots} \\
 \uparrow & & \uparrow & & \uparrow \\
 \overline{\overline{F(t)}} & & \overline{\overline{F(t)}} & & \overline{\overline{F(t)}} \\
 \uparrow & & \uparrow & & \uparrow \\
 \text{FWprt}(t) \overbrace{A(t)}^{\dots} & + & \text{FWprt}(t) \overbrace{A(t)}^{\dots} & = & \text{FWprt}(t) \overbrace{A(t)}^{\dots} \\
 \uparrow & & \uparrow & & \uparrow \\
 \overline{Q_1(t)} & & \overline{Q_2(t)} & & \overline{Q_1(t) \cup Q_2(t)} \\
 \uparrow & & \uparrow & & \uparrow \\
 B(t) & & B(t) & & B(t)
 \end{array} \quad (\text{A.2.4.13.1}),$$

$$\begin{array}{ccccc}
 \overbrace{\overbrace{H_1(t)}^{\dots}}^{\dots} & & \overbrace{\overbrace{H_2(t)}^{\dots}}^{\dots} & & \overbrace{\overbrace{H_1(t) \cup H_2(t)}^{\dots}}^{\dots} \\
 \uparrow & & \uparrow & & \uparrow \\
 \overline{\overline{F(t)}} & & \overline{\overline{F(t)}} & & \overline{\overline{F(t)}} \\
 \uparrow & & \uparrow & & \uparrow \\
 \text{FWprt}(t) \overbrace{A(t)}^{\dots} & + & \text{FWprt}(t) \overbrace{A(t)}^{\dots} & = & \text{FWprt}(t) \overbrace{A(t)}^{\dots} \\
 \uparrow & & \uparrow & & \uparrow \\
 \overline{Q(t)} & & \overline{Q(t)} & & \overline{Q(t)} \\
 \uparrow & & \uparrow & & \uparrow \\
 B(t) & & B(t) & & B(t)
 \end{array} \quad (\text{A.2.4.13.2}),$$

$$\begin{array}{ccccc}
 \overbrace{\overbrace{H(t)}^{\dots}}^{\dots} & & \overbrace{\overbrace{H(t)}^{\dots}}^{\dots} & & \overbrace{\overbrace{H(t)}^{\dots}}^{\dots} \\
 \uparrow & & \uparrow & & \uparrow \\
 \overline{\overline{F_1(t)}} & & \overline{\overline{F_2(t)}} & & \overline{\overline{F_1(t) \cup F_2(t)}} \\
 \uparrow & & \uparrow & & \uparrow \\
 \text{FWprt}(t) \overbrace{A(t)}^{\dots} & + & \text{FWprt}(t) \overbrace{A(t)}^{\dots} & = & \text{FWprt}(t) \overbrace{A(t)}^{\dots} \\
 \uparrow & & \uparrow & & \uparrow \\
 \overline{Q(t)} & & \overline{Q(t)} & & \overline{Q(t)} \\
 \uparrow & & \uparrow & & \uparrow \\
 B(t) & & B(t) & & B(t)
 \end{array} \quad (\text{A.2.4.13.3}).$$

We consider the following self-type dynamic FWprt-structures of the second type:

$$\begin{array}{c}
 \overbrace{H(t)}^{\dots} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 \text{FWprt}(t) \overbrace{Q(t)} \quad (\text{A.2.4.14}), \\
 \uparrow \\
 \overline{Q(t)} \\
 \uparrow \\
 H(t) \\
 \\
 \overbrace{strA(t)}^{\dots} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 \text{FWprt}(t) \overbrace{Q(t)} \quad (\text{A.2.4.14.1}), \\
 \uparrow \\
 \overline{Q(t)} \\
 \uparrow \\
 a(t)
 \end{array}$$

denote $FW_{13}(t) f A(t); Q(t); a(t), a(t) \subset A(t)$,

$$\begin{array}{c}
 \overbrace{a(t)}^{\dots} \\
 \uparrow \\
 \overline{\overline{Q(t)}} \\
 \uparrow \\
 \text{FWprt}(t) \overbrace{Q(t)} \quad (\text{A.2.4.15}), \\
 \uparrow \\
 \overline{Q(t)} \\
 \uparrow \\
 strA(t)
 \end{array}$$

denote $FW_{14}(t) f a(t); Q(t); A(t), a(t) \subset A(t)$,

$$\begin{array}{c}
\overline{\overline{\overline{Q(t)}}} \\
\uparrow \\
\overline{\overline{Q(t)}} \\
\uparrow \\
\text{FWprt}(t) \overline{\overline{A(t)}} \quad (\text{A.2.4.16}), \\
\uparrow \\
\overline{\overline{Q(t)}} \\
\uparrow \\
Q(t)
\end{array}$$

and any other possible options of self for (A.2.4.8) etc.

Definition A.2.4.3. The fuzzy dynamic operator (A.2.4.10) we shall call dynamic tprFW – element, (A.2.4.11) we shall call dynamic trFW – element.

It's allowed to add dynamic tprFW – elements:

$$\begin{array}{ccc}
\begin{array}{c} C_1(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{\overline{R(t)}} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} & + & \begin{array}{c} C_2(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{\overline{R(t)}} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \\
\text{FWprt}(t) & & \text{FWprt}(t) =
\end{array}
\begin{array}{c}
C_1(t) \cup C_2(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overline{\overline{R(t)}} \\
\uparrow \\
\overline{\overline{G(t)}} \\
\uparrow \\
\overline{\overline{O(t)}} \\
\dots
\end{array}
\quad \text{FWprt}(t) \quad (\text{A.2.4.17}),$$

$$\begin{array}{ccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{\overline{R_1(t)}} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} & + & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{\overline{R_2(t)}} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \\
\text{FWprt}(t) & & \text{FWprt}(t) =
\end{array}
\begin{array}{c}
C(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overline{\overline{R_1(t) \cup R_2(t)}} \\
\uparrow \\
\overline{\overline{G(t)}} \\
\uparrow \\
\overline{\overline{O(t)}} \\
\dots
\end{array}
\quad \text{FWprt}(t) \quad (\text{A.2.4.18}),$$

$$\begin{array}{ccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P_1(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} & + \text{FWprt}(t) & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P_2(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \\
& & \text{FWprt}(t) = \begin{array}{c} C(t) \\ \uparrow \\ \overline{P_1(t) \cup P_2(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \text{FWprt}(t) \text{ (A.2.4.18.1),}
\end{array}$$

$$\begin{array}{ccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G_1(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} & + \text{FWprt}(t) & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G_2(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \\
& & \text{FWprt}(t) = \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G_1(t) \cup G_2(t)}} \\ \uparrow \\ \overline{\overline{O(t)}} \\ \dots \end{array} \text{FWprt}(t) \text{ (A.2.4.18.2),}
\end{array}$$

$$\begin{array}{ccc}
\begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O_1(t)}} \\ \dots \end{array} & + \text{FWprt}(t) & \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O_1(t) \cup O_2(t)}} \\ \dots \end{array} \\
& & \text{FWprt}(t) = \begin{array}{c} C(t) \\ \uparrow \\ \overline{P(t)} \\ \uparrow \\ \overline{R(t)} \\ \uparrow \\ \overline{\overline{G(t)}} \\ \uparrow \\ \overline{\overline{O_1(t) \cup O_2(t)}} \\ \dots \end{array} \text{FWprt}(t) \text{ (A.2.4.18.3),}
\end{array}$$

We consider the following self-type dynamic tprFW-structures:

$$\begin{array}{c}
R(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overbrace{R(t)} \\
\uparrow \text{FWprt}(t)(\text{A.2.4.19}) \\
\overline{\overline{R(t)}} \\
\uparrow \\
\overbrace{\overline{R(t)}} \\
\ldots \\
strD(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overbrace{P(t)} \\
\uparrow \text{FWprt}(t) (\text{A.2.4.19.1}), \\
\overline{\overline{P(t)}} \\
\uparrow \\
\overbrace{\overline{d(t)}} \\
\ldots
\end{array}$$

denote $FW_{15}(t)fd(t); \overline{P(t)}; D(t), d(t) \subset D(t)$,

$$\begin{array}{c}
d(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overbrace{D(t)} \\
\uparrow \text{FWprt}(t) (\text{A.2.4.20}) \\
\overline{\overline{D(t)}} \\
\uparrow \\
\overbrace{\overline{D(t)}} \\
\ldots
\end{array}$$

denote $FW_{16}(t)fD(t); \overline{P(t)}; d(t), d(t) \subset D(t)$,

$$\begin{array}{c}
P(t) \\
\uparrow \\
\overline{P(t)} \\
\uparrow \\
\overline{\overline{R(t)}} \\
\uparrow \text{FWprt}(t)(\text{A.2.4.21}) \\
\overline{\overline{R(t)}} \\
\uparrow \\
\overline{\overline{R(t)}} \\
\vdots
\end{array}$$

and any other possible options of self for (A.2.4.10) etc.

New mathematical structures and operators is carried out with generalization it to any structures with any actions. For example,

$$1) \begin{array}{ccccc}
f_{11} & \dots & f_{1k} & & \\
\dots & \dots & \dots & q_{11} & \dots & q_{1n} \\
(q_{j1})^{-1} & \dots & (q_{jk})^{-1} & \text{FWFWprt} & \dots & (*_{\text{A.2.4.}}), \\
\dots & \dots & \dots & q_{m1} & \dots & q_{mn} \\
f_{l1} & \dots & f_{lk} & & &
\end{array}$$

f_{ij}, q_{ij} – any fuzzy objects, fuzzy actions etc.

$$2) \begin{array}{ccccccc}
& & g_{12} & & w_{11} & w_{12} & w_{1n} \\
g_{11} & & g_{22} & & \dots & \dots & w_{2n} \\
(w_{j1})^{-1} & (w_{j2})^{-1} & g_{13} & \text{FFWGprt} & & & \dots \\
g_{31} & \dots & (w_{j3})^{-1} & & w_{m1} & w_{m2} & \dots \\
& & g_{k2} & & & & w_{sn} & w_{ml}
\end{array}$$

$(*_{\text{A.2.4.1}}),$

w_{ij}, g_{ij} – any fuzzy objects, fuzzy actions etc.

3)

$$\begin{array}{ccc}
a & b & g \\
c & AfFWrq(\mu) & w \quad (*_{\text{A.2.4.2}}), \\
d & q & r
\end{array}$$

where $AfFWrq$ is fuzzy virtual structure or fuzzy virtual operator, which can take any form of fuzzy action; $a, c, d, q, r, w, g, b, \mu$ – any fuzzy objects, fuzzy actions etc.

Accordingly, we can consider all sorts of self-type fuzzy structures for 1) –

3). And any other possible fuzzy structures and fuzzy operators etc.

A.3 Elements of the theory of variables of fuzzy hierarchical fuzzy dynamic operators: FWprt

In contrast to the classical one-attribute fuzzy set theory, where only its contents are taken as a set, we consider a two-attribute fuzzy set theory with a fuzzy set as a fuzzy capacity and separately with its contents. We simply use a convenient form to represent the singularity of a fuzzy set. Articles [1]-[12] use the following methodology for permanent structures:

1. Cancellation of the axiom of regularity.
2. 2 attributes for the fuzzy set: fuzzy capacity and its content.
3. Fuzzy compression of a fuzzy set, for example, to a point.
4. “turning out” from one another, particularly from a fuzzy capacity, we pull out another fuzzy capacity, for example, itself, as its element.
5. The simultaneity of one (fuzzy compression) and the other (“eversion”).
6. Own fuzzy capacities.
7. Qualitatively new fuzzy programming and fuzzy Networks.

Here we will consider variable fuzzy structures (models), both discrete and continuous: a) with variable connections, b) with the variable backbone for links, c) generalized version; in particular, in variable fuzzy structures (models), for example,

$$\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{FWprt} , q_2 \geq t \geq q_1) | \mu_1 \\
\overline{\overline{G}} \\
\uparrow \\
\tilde{\tilde{O}}
\end{array}
= \left\{ \begin{array}{l}
\begin{array}{c}
C \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{FWprt} , q_2 \geq t \geq q_1) | \mu_1 \\
\overline{\overline{G}} \\
\uparrow \\
\tilde{\tilde{O}}
\end{array} \\
\begin{array}{c}
B \quad \dots \quad A \\
(\mu_7 \text{ffS}^1 \text{prt} \mu_6 , q_3 \geq t > q_2) | \mu_2 \\
D \quad B \\
C \quad \dots \\
\uparrow \quad \tilde{\tilde{H}} \\
\bar{P} \quad \uparrow \\
\tilde{R} \quad \overline{\overline{F}} \\
\uparrow \text{FWprt} \quad \uparrow \\
\overline{\overline{G}} \quad \tilde{A} , q_4 \geq t > q_3) | \mu_3 \\
\uparrow \quad \bar{Q} \\
\tilde{\tilde{O}} \quad \uparrow B \\
\dots \quad \dots \quad \tilde{\tilde{H}} \\
\uparrow \quad \overline{\overline{F}} \\
\text{FWprt} \quad \uparrow \tilde{A} , q_5 \geq t > q_4) | \mu_4 \\
\uparrow \quad \bar{Q} \\
\quad \uparrow B
\end{array} \\
\begin{array}{c}
\{\} \\
\uparrow \\
\bar{P} \\
\uparrow \\
\tilde{R} \\
\uparrow \text{FWprt} , t > q_5) | \mu_5 \\
\overline{\overline{G}} \\
\uparrow \\
\tilde{\tilde{O}}
\end{array}
\end{array} \right. \quad (*_{A.\#.1}),$$

μ_i - measures of fuzziness, $i = 1, \dots, 5$. In particular, $\mu_7 \text{ffS}^1 \text{prt} \mu_6$ can be

interpreted as a fuzzy game: player 1 fuzzy with measures of fuzziness μ_6 fits fuzzy A into fuzzy B, and the other fuzzy with measures of fuzziness μ_7 pushes fuzzy D out of fuzzy B at the same time.

In what follows, we will denote variable fuzzy structure (model) through $VFW(t)$, qself-variable fuzzy structures (models) through $FW_qFVWS(t)$, qself is self for *action* Q , and oqself-variable fuzzy structures (models) through $OqVFW(t)$, qself is oself for *action* Q . Singular fuzzy structures (models) are not

confused with fuzzy structures (models) with singularities. $\mu_7 \text{ffS}^1 \text{prt} \mu_6$ -2-

hierarchical fuzzy structure: 1-level - elements A, B, C, D; level 2 - connections between them. 2-

Examples: a) discrete variable fuzzy structure with μ_i - measures of fuzziness, $i = 1, \dots, 8$.

$$\begin{array}{ccc} a|\mu_1 & b|\mu_8 & g|\mu_7 \\ c|\mu_2 & VFW(t) & w|\mu_6 \\ d|\mu_3 & q|\mu_4 & r|\mu_5 \end{array}$$

Fig.A.3.1

c) continuous variable fuzzy structure



Fig.A.3.2

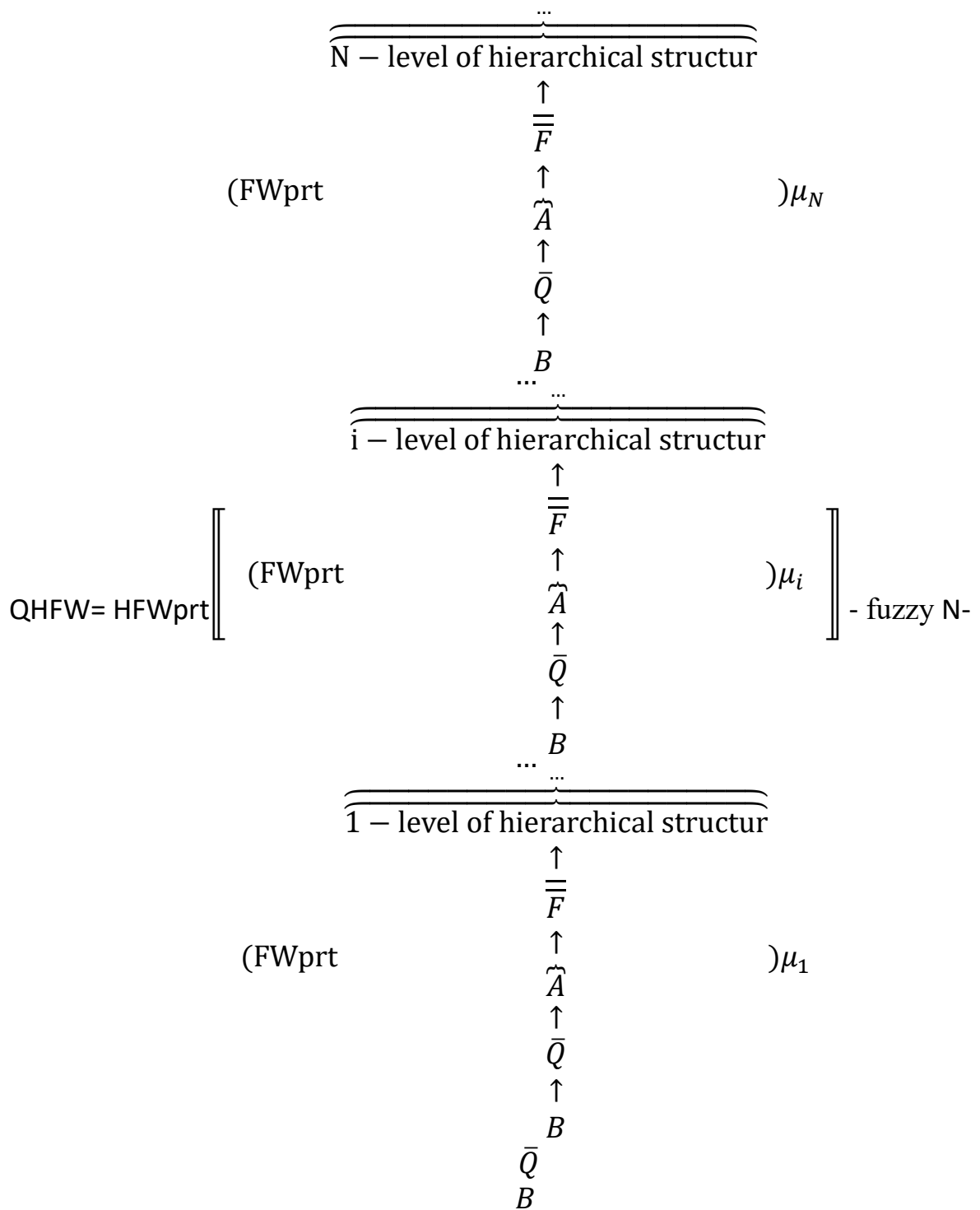
Where a continuous fuzzy set represents the rim of the Fig.A.3.2.

We introduce the notation m_{fVLS_N} – the number of elements, N - the number of connections between them in the discrete variable 2-hierarchical fuzzy structure $VFW(t)$. We introduce the notation q_{fVLS_R} – any, R - connections in q_{fVLS_R} in the variable 2-hierarchical fuzzy structure $VFW(t)$, in particular, q_{fVLS_R} , R can be fuzzy sets both discrete and continuous and discrete-continuous. We consider the functional $c(Q)$, which gives a numerical value for the fuzzy structurability of Q

Can be considered N-hierarchical fuzzy structure: 1-level - elements; level 2 - connections between them, level 3 - relationships between elements of level 2, etc. up to level $N+1$. N-hierarchical fuzzy structure: 1-level - A; 2-level -B, 3-level - C, etc. up to $(N+1)$ - level, where A, B, C, ... can be any in particular, by fuzzy actions, fuzzy sets, and others.

Can be considered discrete fuzzy hierarchical fuzzy structure, continuous fuzzy hierarchical fuzzy structure, and discrete-continuous hierarchical fuzzy structure.

The example

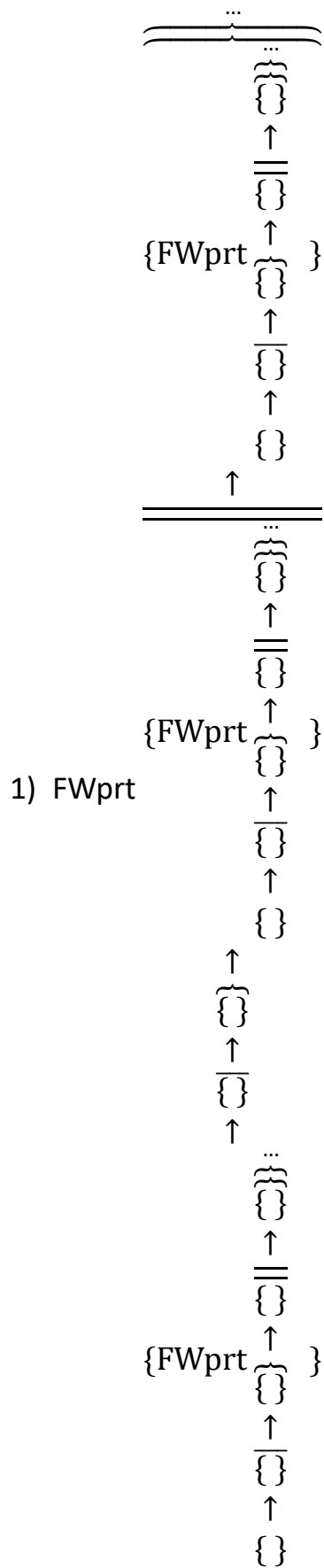


hierarchical fuzzy structure compression into fuzzy B, μ_i - measures of fuzziness, $i = 1, \dots, N$.

Let $\text{fwg}(N, \text{QHFW}) = \text{QHFW}^{\left. \text{QHFW} \text{QHFW} \dots \text{QHFW} \right\}_{-N \text{ levels}}}$

It can be considered self- QHFW, $\text{fwg}(y, \text{QHFW})$ for any y , $\text{fwg}(\text{QHFW}, \text{QHFW})$.

Compression fuzzy Hierarchy Examples:



$ca(A) = \sum_{k=1}^n ca(B_k) * ca(A/B_k)$, $B_1, B_2, \dots, B_n$ -full group of fuzzy hypotheses-actions: $\sum_{k=1}^n ca(B_k) = 1$ ("fuzzy result of action"). FWprt-(fuzzy action) for

$$\begin{array}{c} \overline{\overline{\overline{\dots}}} \\ \overline{\overline{\overline{(x_1 | \mu_{\tilde{x}}(x_1), x_2 | \mu_{\tilde{x}}(x_2), \dots, x_n | \mu_{\tilde{x}}(x_n))}}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \tilde{A} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ w \end{array}$$

$\tilde{x} = (x_1 | \mu_{\tilde{x}}(x_1), x_2 | \mu_{\tilde{x}}(x_2), \dots, x_n | \mu_{\tilde{x}}(x_n))$: FWprt

,
 $\tilde{x}$ - fuzzy set of fuzzy actions.

$$\begin{array}{c} \overline{\overline{\overline{\dots}}} \\ \overline{\overline{\overline{\{ ca(x_1) | \mu_{ca(\tilde{x})} ca(x_1) | \mu_{ca(\tilde{x})} ca(x_2), \dots, ca(x_n) | \mu_{ca(\tilde{x})} ca(x_n) \}}}}} \\ \uparrow \\ \overline{\overline{F}} \\ \uparrow \\ \tilde{A} \\ \uparrow \\ \overline{Q} \\ \uparrow \\ w \end{array}$$

FWprt

- FWprt- accommodation for these fuzzy actions x_i , $i = 1, \dots, n$. It is possible to consider the self-(fuzzy action) $FW_8 f \tilde{x}_w$; Q ; $\tilde{x}$, $\tilde{x}_w \subset \tilde{x}$. The same for self-(fuzzy accommodation): $FW f Ca_w(\tilde{x})$; Q ; $\widetilde{Ca(\tilde{x})}$, where $Ca_w(\tilde{x}) =$

$$\{ ca(x_1) | \mu_{ca(\tilde{x})} ca(x_1), ca(x_2) | \mu_{ca(\tilde{x})} ca(x_2), \dots, ca(x_n) | \mu_{ca(\tilde{x})} ca(x_n) \} \subset \widetilde{Ca(\tilde{x})}.$$

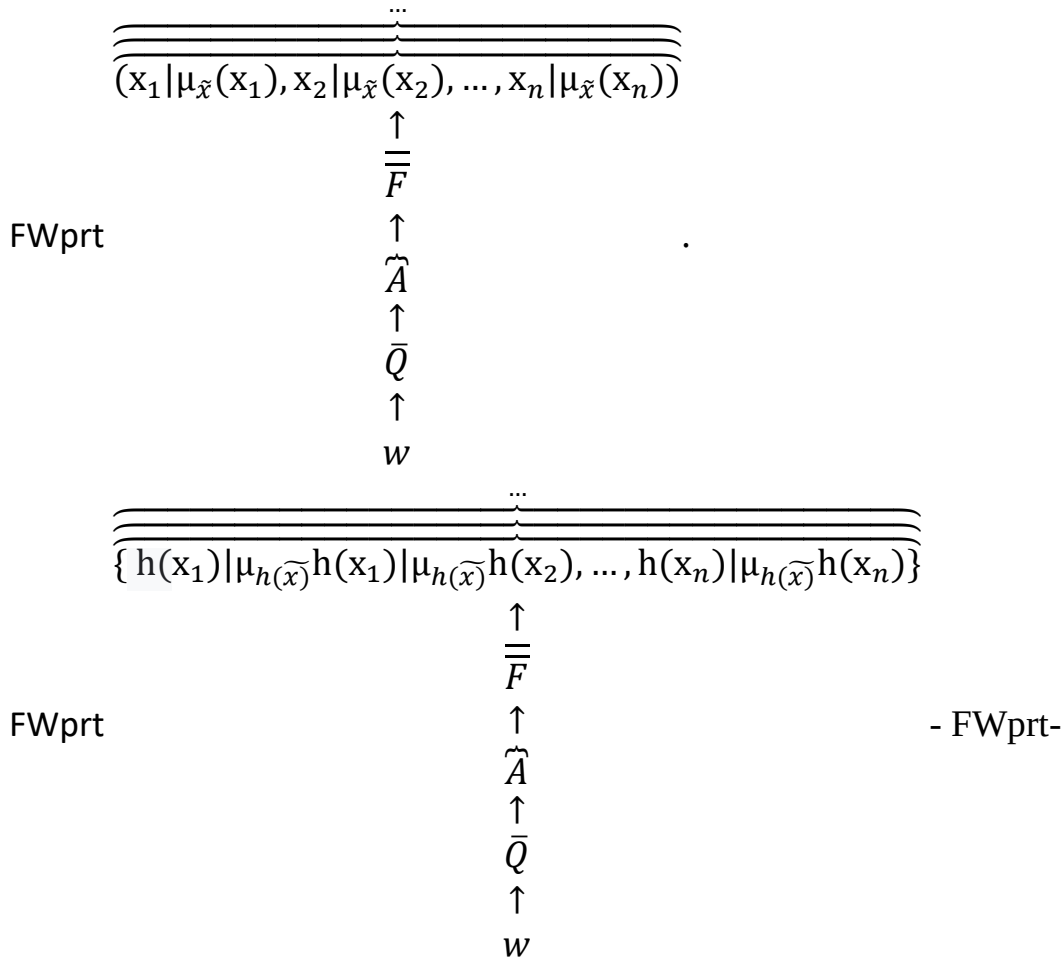
Consider a variable fuzzy hierarchy (we will denote it by flVH).

We consider the functional $h(Q)$, which gives a numerical value for the hierarchization of fuzzy Q from the interval $[0,1]$, where 0 corresponds to "no fuzzy hierarchy," and 1 corresponds to the value "fuzzy hierarchy." Then for joint fuzzy hierarchies A, B : $h(A+B) = h(A) + h(B) - h(A*B) + hS(D)$, D - self-(fuzzy hierarchy) from $A*B$, $hS(x)$ - the value of self-(fuzzy hierarchy) for self-(fuzzy

hierarchy) x ; for dependent fuzzy hierarchies:

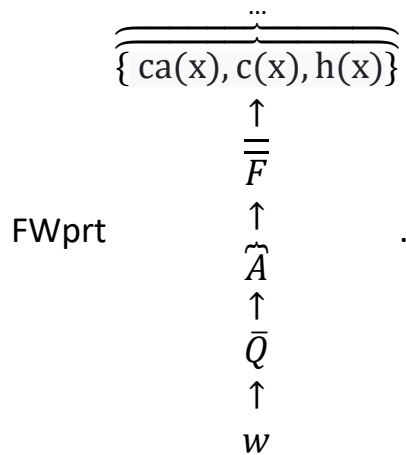
$h(A*B)=h(A)*h(B/A)=h(B)*h(A/B)$, where $h(B/A)$ - conditional hierarchization of the fuzzy hierarchy B at the fuzzy hierarchy A , $h(A/B)$ - conditional fuzzy hierarchy of the fuzzy hierarchy A at the fuzzy hierarchy B . Adding the fuzzy hierarchy values of inconsistent fuzzy hierarchies: $h(A+B)=h(A)+h(B)$. The formula of complete fuzzy hierarchy: $h(A)=\sum_{k=1}^n h(B_k) * h(A/B_k)$, $B_1, B_2, \dots, B_n$ - full group of fuzzy hypotheses- hierarches: $\sum_{k=1}^n h(B_k)=1$ ("fuzzy hierarchy").

FLprt- structure for fuzzy set of hierarches $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$:

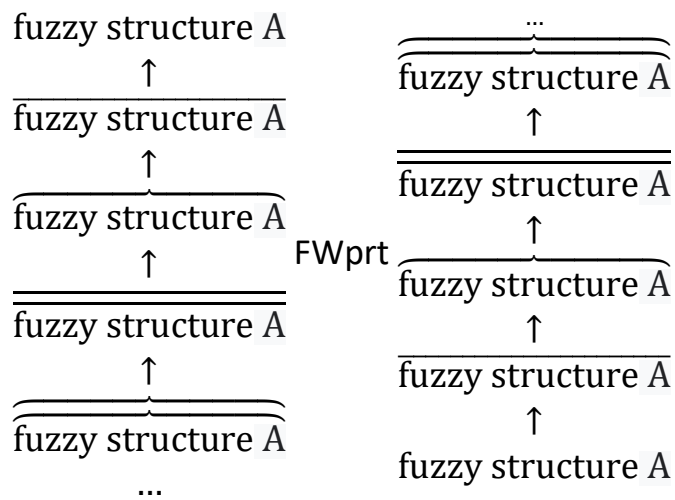


hierarchization for these fuzzy hierarches. It is possible to consider the self-(fuzzy hierarchy) $FW_8f\tilde{x}_w; Q; \tilde{x}, \tilde{x}_w \subset \tilde{x}$. The same for self- hierarchization $FW_8f\tilde{h}x_w; Q; \tilde{h}x, \tilde{h}x_w \subset \tilde{h}x, \tilde{h}x =$

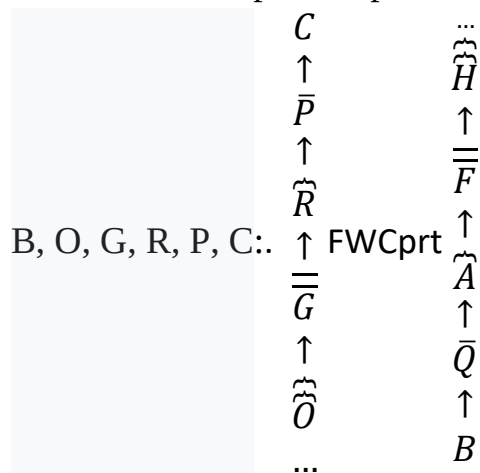
$\{h(x_1)|\mu_{h(\widetilde{x})}h(x_1), h(x_2)|\mu_{h(\widetilde{x})}h(x_2), \dots, h(x_n)|\mu_{h(\widetilde{x})}h(x_n)\}$. Can be considered



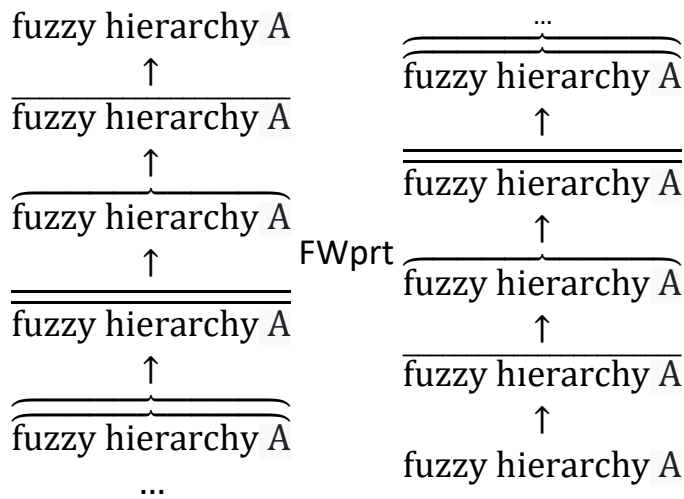
Very interesting next fuzzy structure type:



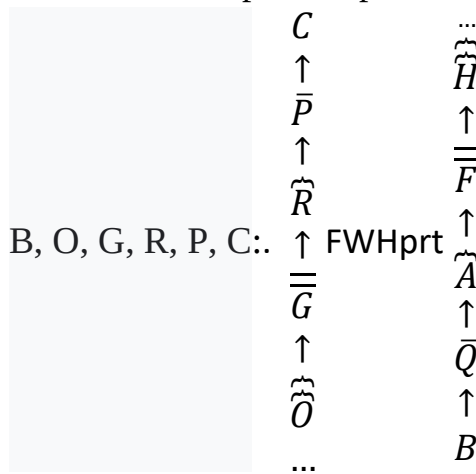
You can enter special operator FWCPrt to work with fuzzy structures H, F, A, Q,



Very interesting next fuzzy hierarchy type:



You can enter special operator FWHprt to work with fuzzy hierarches H, F, A, Q,

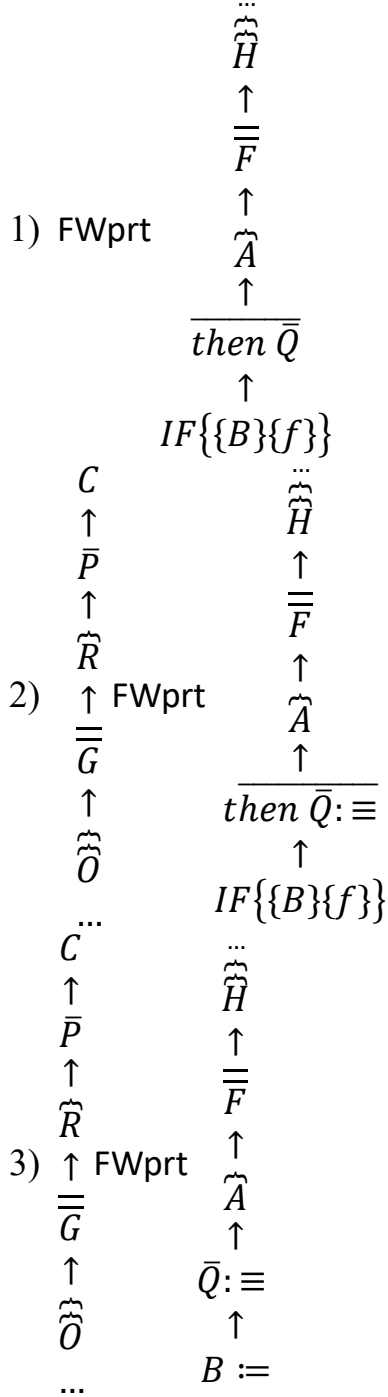


A.4 Introduction to FUZZY PROGRAM OPERATORS FWprt, tprFW, FW¹epr, FWeprt₁

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through FWprt-Networks - fuzzy analogue of Sit-Networks [1], [6], [12] in one of the central departments of which a conventional computer system is located. The parallel processor is itself fwdeprogram - fuzzy analogue of eprogram [1], [6], [12] with direct parallel computing not through serial computing.

- 2) fuzzy tprW-program operators (designation tprFW-program operators)
- 3) fuzzy W¹epr - program operators (designation FW¹epr -program operators)
- 4) fuzzy Weprt₁- program operators (designation FWeprt₁-program operators)

Examples:



A.5 Paradoxical singularities (singularities of disintegration&synthesis)

The simplest type of paradoxical singularity is ${}^B_A Sprt^A$, then: $pa self A = {}^A_A Sprt^A$,

$(action Q)^{-1} Dprt action Q$ (where Q is any action), $(action Q)^{-1} Dprt action Q$,

$Subject of |||$ $Slprt$ $Subject of |||$, $Subject of |||$ $Slprt$ $Subject of |||$, $pa ||| =$

$self$
 $/ \quad \backslash$
 $||| (|||) |||^{-1}$, $pi S = (pa self) - (|||)$, $self ||| pa self$ for any C for any A, where A
 $\backslash \quad /$
 $(pa |||)$

can be by $self$, $|||$, $pa self$ etc; $pi 1 S = (pa self) - (pa |||)$,
 $\backslash \quad /$
 $(pa |||)$

$\frac{(the set of all sets)}{|| the set of all sets ||} |||$ (an element that is not included in any capacity (set)) etc.

May consider $pf(n, pa self) = pa^n self A = pa(pa(...(pa self A)...))$, $pf(\alpha, pa self)$,
 $pf(\infty, pa self) pf(pa self, pa self)$ etc and rows by $pa^n self A$.

Let us introduce a measure of singularity μ_{si} , which sets the maximum level of singularity as 2, and the minimum, i.e., regularity, as 0. Then, in

particular, for binary relations: $\mu_{si} = 3/2$ for $A ||| A^{-1}$, $Sprt_x^{A, A^{-1}}$ (x - point)

or for $A ||| (-A)$, $Sprt_x^{A, -A}$, $\mu_{si} = 1$ for $A ||| B$, $Sprt_x^{A, B}$ (x - point), $\mu_{si} = 2/3$ for

Q
 $Drt \frac{Q}{Q}$, $\mu_{si} = 1/2$ for $A ||| A$, $Sprt_A^A$. $Pa self$ can have protected "holes" for

inputs and outputs of energy fibers. You can introduce ordered types

$pa self$, eg - $\frac{{}^A_A Sprt^A}{\rightarrow}$, $\frac{{}^A_A Sprt^A}{\leftarrow}$, $\rightarrow$ sets the direction from action ${}^A_A Sprt$ to action

$Sprt_A^A$, $\leftarrow$ sets the direction from action $Sprt_A^A$ to action ${}^A_A Sprt$.

Remark.A.5.1 As a rule $|||$ attracts (forms, induces) the external self "around" itself.

Remark.A.5.2. Flame and water are characterized by changes (fluidity) that provide induction that generates the corresponding paselfs, which by the way can be used by SmnSprt (paSmnSprt), in particular for the exit through the middle level of flame and water to the upper one. His identifier, having connected to their middle level and using the energy, is able to activate SmnSprt (paSmnSprt) to obtain what is needed. For this, you can use the grown bioidentifier (in particular, the simplest ones - neurons). One paself it can "cling" to the other and thereby obtain a new singularity at the upper level or exit to such a ready singularity with manifestations coming out of it. Since ||| is connected with the upper level, then the identifiers must be from there. ||| always raises the levels up. Everything (anything) is a product of |||. Any self-type structure can be presented as a "hole" in the accumulation. May consider a new dynamic operator

$$\begin{matrix} E \\ Q \\ F \\ G \end{matrix} \begin{matrix} \text{8} \\ \text{8} \end{matrix} \begin{matrix} A \\ B \\ C \\ D \end{matrix} \text{prt} \begin{matrix} C \\ D \end{matrix}, \text{ where } A \text{ fits into } B, F \text{ fits into } G, D \text{ is forced out from } C, Q \text{ is}$$

forced out from E; A, B, C, D, g_1 , g_2 may also be fuzzy. May consider the characteristics accumulation of any objects or process. In fact, we try to consider hierarchy of energies in the Universe, but the accumulation isn't common property of these energies. The accumulation of energies is in some process in the Universe, for example, the process of human spiritual development allows such a possibility, and when constructing pseudo-living energies. It is possible to significantly expand the horizons of science, in particular physics, by studying the subtle energies in the Universe.

A.6 Singularities algebra

You can define operations for the same type singularities and study their algebras. But our task for now is limited to considering certain types of singularities presumably corresponding to certain types of subtle energy.

Let us introduce the following notations: $A*B = \text{Sprt}_B^A$, $A^2 = \text{self } A = \text{Srt}_A^A$,

$$A^{\frac{3}{2}} = \text{Drt}_A^A = \text{self}^{\frac{3}{2}}(A), A^3 = \text{Self}^2 A, \dots, A^{\frac{3n}{2}} = \text{Drt}_{A^n}^{A^n} = \text{self}^{\frac{3n}{2}}(A), A^{n+1} =$$

$$\text{Self}^n A, \text{self}^{\min(n,m)}(A) \in \text{Srt}_{A^m}^{A^n} = \text{self}^{\frac{n}{m}}(A), \text{self}^{\min(n,m,k)}(A) \in \text{Drt}_{A^k}^{A^n} =$$

$$\text{self}^{\frac{n+m+k}{2k}}(A), \dots \text{ etc. Similarly for negative powers, starting with } A^{-1} =$$

$$\text{self}^{-1} A = {}_A^A \text{Sprt}, A^{-2} = \text{self}^{-2} A, A^{\frac{-3}{2}} = {}_A^A \text{Dprt} \text{ etc.}$$

Remark A.6.1. $\text{Self}^2 A$ can be considered in the following options: a) $\text{self}^2 A$ as the set of all self-sets of a set A, b) $\text{self}^2 A = \text{Sprt}_{\text{self } A}^{\text{self } A}$ can be interpreted as the containment of one more self-set of set A etc.

There is no commutativity here: $A*B \neq B*A$. We can consider operator

$$\text{functions: } e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots, (A+B)^n = \sum_{k=0}^n \binom{n}{k} A^k B^{n-k},$$

$$(1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \dots, \text{ etc.}$$

You can consider a more “hard” option: $A*B = \text{PSprt}_B^A$, where PSprt_B^A – operator, containing A in every element of B, $A^2 = \text{PSelf } A = \text{PSrt}_A^A$, $A^3 =$

$$\text{PSelf}^2 A, \dots, A^{n+1} = \text{PSelf}^n A, \text{PSelf}^{\min(n,m)}(A) \in \text{PSrt}_{A^m}^{A^n} = \text{PSelf}^{\frac{n}{m}}(A),$$

...etc. There is no commutativity here: $A*B \neq B*A$. We can consider

$$\text{operator functions: } e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots, (A+B)^n =$$

$$\sum_{k=0}^n \binom{n}{k} A^k B^{n-k}, (1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \dots, \text{ etc.}$$

Let's introduce $\sqrt{\text{self}}$ as the result of the decision of the equation $\text{Srt}_x^x = \text{self}$, that is $x = \sqrt{\text{self}}$, $\sqrt[3]{\text{self}}$ as the result of the decision of the equation $\text{Dprt}_x^x = \text{self}$, that

is $x = \sqrt[n]{self}, \sqrt[m]{self^m}$ as the result of the decision of the equation $x^{\frac{n}{m}} = self$, $self^\alpha$ as the result of the decision of the equation $x^{\frac{1}{\alpha}} = self$, where α is any number, in particular, a negative number etc. The following equality is true:

$$self^{-\alpha}(self^\alpha G) = self^\alpha(self^{-\alpha} G) = G (*),$$

you can, for example, specify $self^{-\alpha}$ by definition through (*).

The following equality is true:

$$self^\beta(self^\alpha G) = self^\alpha(self^\beta G) = self^{\alpha+\beta} G \quad \forall \alpha, \beta. \text{ In this way one can introduce self-level space. The following operation may take place: } q\uparrow\downarrow q + {}^A_Sprt_A^A = {}^{A+q}_{A+q}Sprt_{A+q}^{A+q}.$$

Similar algebras can be constructed for other singularities, in particular, for oself, $\uparrow\downarrow$, $|||$, partial self-type singularities etc. For example, $A|||(|)|B = A(|)|^2B$, $(|)|^\alpha$ as the result of the decision of the equation $x^{\frac{1}{\alpha}} = |||$. $(|)|^\alpha * (|)|^{\frac{1}{\alpha}} = (|)|^{\frac{1}{\alpha}} * (|)|^\alpha = |||$ etc.

A.7 Types, forms internal and external, structures of self and $|||$

Let us introduce the following notations: $self_B A$ is self A by form B, $C|||_B A$ is $|||$ A with C by form B. You can consider any functions, operators: for example, $g(self_B A, d)$, $self_B A(B(self_B A, d), q)$ etc. If usual self can be associated with the action "in a circle" on itself, then the action "in a double circle in the form of an eight" on itself can be associated with a 2-self, the action "in a triple circle in the form of 3 circles intersecting only at one point" on itself can be associated with a 3-self etc.

A set B containing itself as N elements will be denoted by $Nself B$. $(\varphi)self A$ containing itself as A^φ , $(\varphi)paself A$ containing itself as $A^\varphi ||| A^{-\varphi}$, $(f(\varphi))self A$ containing itself as $f(A, \varphi)$, $(f(\varphi))paself A$ containing itself as $f(A, \varphi) ||| f(A^{-1}, \varphi)$. The example of difference between the internal and external form of the same self: the external form of the self of a living organism is represented by an energy cocoon and, at the object level, by a

physical body, and the internal form of the self of a living organism is represented by numerous DNA and RNA.

$q\uparrow\downarrow q$ has self-structure for q and $paself$ -structure by directions. An atom has $paself$ for electric-structure, self-structure for by electron orbitals, $|||$ by nucleons.

Likewise, any other object (process) can have different singular characteristics.

$oselfA = {}^A Sprt$ can be interpreted also as $(-self)A$.

For example, you can study the "side" version of self: $sidselA$ is A next to the side of itself.

The hierarchy of singularities of one can be included in the hierarchy of singularities of another, for example, the hierarchy of singularities of the fruit of a tree is included in the hierarchy of singularities of a tree, the hierarchy of singularities of a person is included in the hierarchy of singularities of the Universe etc. $self\ of\ the\ fruit \subset self\ of\ the\ tree \subset self\ of\ the\ Universe$.

$self_B A$, $self\ A$ can be not only for objects (processes) but also for their characteristics, structures etc.

Remark A.7.1. Classical science forms its abstractions as the intersection of properties of objects (processes), our approach to the formation of singularities is through $|||$.

Remark A.7.2. All dynamic operators and examples with them in [1]-[15] are examples of the construction of pseudo-living energies.

Remark A.7.3. $self$ is resting point of accumulation process. A living organism is protected from further accumulation of energy until its $self$ is destroyed. Some fearless people, extending the accumulation of energy of their organism and reaching other states of awareness, stop the accumulation of energy of their organism in time so as not to lose protection. In the limit, "they turn on all states of awareness at once and move to another higher level of energy accumulation, removing protection at the previous level: "burning with fire from within." Here (in this work) we mean by accumulation: an increase in capacity due to an increase in level.

Remark A.7.4. Supercoiling of DNA before its doubling shows the role of "folding" into a higher level of self to perform super actions.

Remark A.7.5. As one of the options $A|||B$ can be tried to be interpreted as a generalization (1.1): by the form

$$(2(1, (2,1)),1) \text{ (9.1),}$$

or (1.1.4.1), (1.1.4.2),

and the corresponding self² by the form

$$(1, (2(1,(2,1)),1)) \text{ (9.2),}$$

for $pa|||$ by the form

$$(2((1, (2(1,(2,1)),1))), 1) \text{ (9.3)}$$

or (1.1.4.3), (1.1.4.4).

self²A = $pa self A = {}^A_Sprt^A_A$ can be tried to interpret in the following ways,

depending on the choice of interpretation of the operation for $Sprt^A_B = A*B$ or

$Sprt^A_B = A \neq B$, as $self A * self(A)^{-1}$ or $self A * self(-A)$ respectively. $self A$ does not

mean accumulation for A, but means a new, higher level of energy for A due to its new accumulative structure. There are an infinite number of qualitatively different other variants of accumulation. We simply proceed from our usual 2 - interpretations of the world. Therefore, we limit ourselves to the formula $A|||B$.

If through 3 – interpretation, then we can consider the
$$\begin{array}{cc} A & B \\ |||_3 & \\ C & \end{array}$$
 - accumulation

and its degenerations – self-type structures, for example
$$\begin{array}{cc} A & A \\ |||_3 & \\ A & \end{array}$$
 etc. It is also

possible through 4 – interpretation etc. Namely, self-type structures are the

structures of self-organization. May consider type of $||| : A-||| = |||A|||$ for any

A, accumulation- $(pa|||)D = D(pa|||)accumulation(pa|||)D$ for any D, $pa self-$

accumulation, accumulation|||C for any C, self ||| Q for any Q,

$$\begin{pmatrix} \text{accumulation} - (\text{pa}|||)D \\ \text{pa} \text{self} - \text{accumulation of } D \\ \text{accumulation}|||D \\ \text{self } ||| D \\ \text{the raw energy of } |||D \\ \text{the manifestation of } |||D \end{pmatrix} \text{ is levels accumulation of } D \text{ etc.}$$

Remark A.7.6. self-induction: when the induction changes, we obtain an induction on itself.

Remark A.7.7. May consider the manifestation of self-operator A (subject (master) of operator A) according to B. May consider the manifestation of self-action Q (subject (master) of action Q) according to C. May consider the manifestation of self-D (subject (master) of D) according to R for any D, R.

Remark A.7.8. Energy manifests itself through actions, objects ("gives birth" to actions, objects). Subject ||| in the cocoon of a living organism is the assemblage point. It also has its own subject (master), who carries out its necessary adjustment, control. The "tipper" that gives energy to a living organism has the type $\overline{\text{self} - \text{oself}}$.

Remark A.7.9. One can try to interpret it in a very simplified way self-level of a living organism as self(oself- self). Then one can try to interpret it in a very simplified way self-level of energy-producing beams for living organisms as $\overline{\text{self} - \text{oself}}$. Then self-level of energy others objects - self.

Remark A.7.10. Let us introduce some generalizations of |||: a) $(m \rightarrow n)|||$, $||| = \begin{matrix} A & B \\ & C \end{matrix} |||$, ..., $(\forall Q)|||$ etc.

A.8 Types, forms internal and external, structures of potential self, potential ||| and others potential singularities

Self characterizes by the average level, at the lower level it can only be in potential form, for example, DNA in living organisms. Denotes a potential self as poself, potential ||| as po|||. Self-sets, self-(material objects) can only be classified on

object level as a potential selves. Denotes a potential self, containing itself potentially, but its potentiality does not contain its potentiality, as p1self. It is also possible to consider any other types of potential singularities, all sorts of potentially singular structures, forms, degrees, characteristics. $poselfA = A(p|||)A$. May consider $po(po(\dots(poselfA)\dots))$.

One can try to simplify the energy cocoon of a living organism with the expression $paselfA = {}^A_A Sprt^A$, then we can try to simplify the assemblage point on the energy cocoon using expression $\xrightarrow{{}^A_A \nabla prt^B_A}$ for a female and $\xrightarrow{{}^A_A \Delta prt^B_A}$ for a male, where B are an external energy fibers of the Universe, ∇ here means the shape of the assemblage point in the form of a cone, facing "funnel" upwards, and Δ - "funnel" downwards, $\rightarrow$ sets the direction from action ${}^A_A Sprt$ to action $Sprt^B_A$; you can try to simplify the energy gap on the energy cocoon at the level below the navel on the energy cocoon with expression $\xleftarrow{{}^A_{InB} Sprt^A_A}$, where InB are an internal energy fibers, $\leftarrow$ sets the direction from action $Sprt^A_A$ to action ${}^A_{InB} Sprt$, similar, but to a lesser extent than this energy gap, can be imagined for the energy manifestations of the eyes.

To obtain energies of a higher level, it is necessary to use level accumulation structures.

In our dynamic mathematics, instead of repeating identifiers of specific material objects, energies, special singularities are used, their characteristics are another matter.

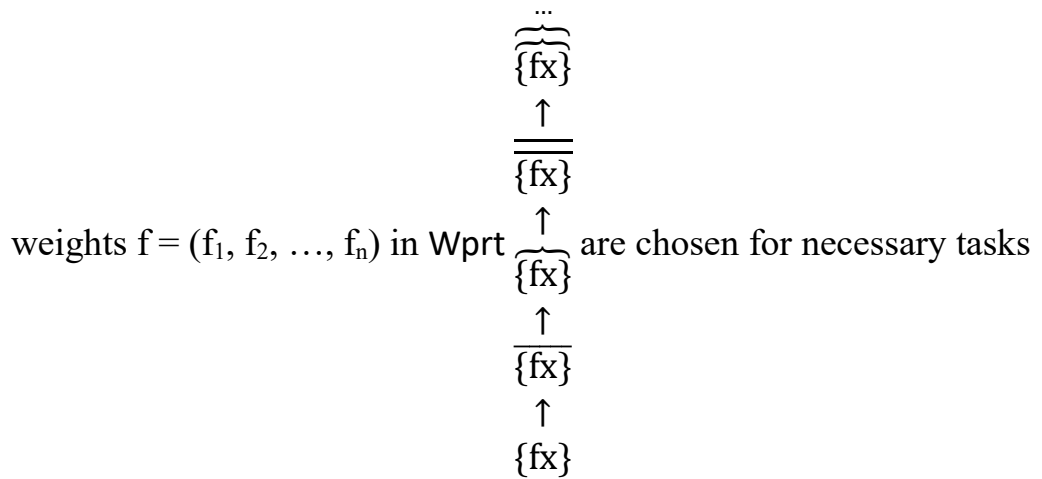
A.9 Some available types of "pseudo-living energy"

- a) UHF with amplitude close to zero, b) short pulse laser pulses, c) UHF with twisting around a hollow wire, d) ultraviolet, e) in the form of ball lightning, h) vortices from UHF with almost zero amplitude, q) electric arc with almost zero amplitude, f) UHF by a "twisted" wire with minimal loops, made using nano-technology, by manipulating frequencies, it is possible to "construct" various pseudo-living energies, g) for ultraviolet similar waveguide, r) vortices from ultraviolet, p) UHF with amplitude

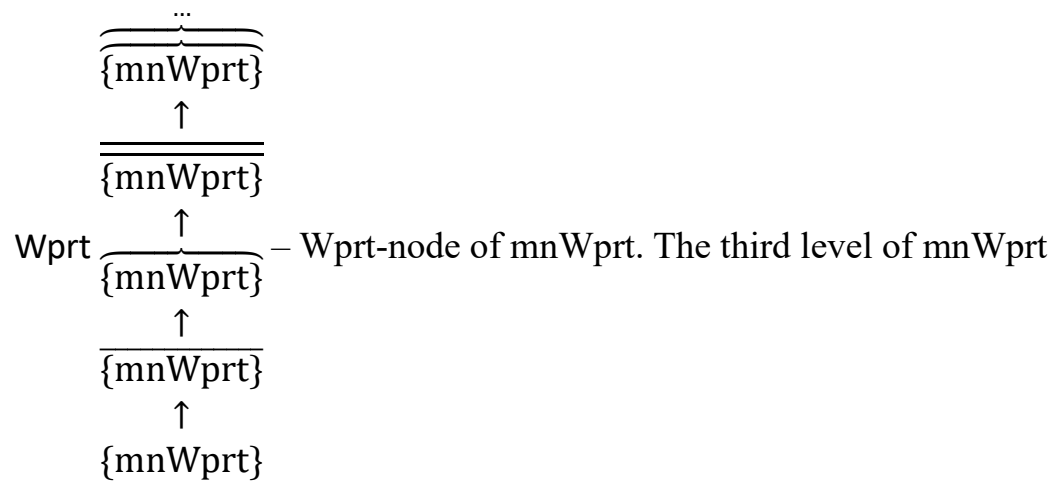
close to zero & with direct current, then may use the additional to the main normal coding through 0 and 1 to the main, w) through multiple cycles to ∞ , u) through multiple returns of programs to themselves etc.

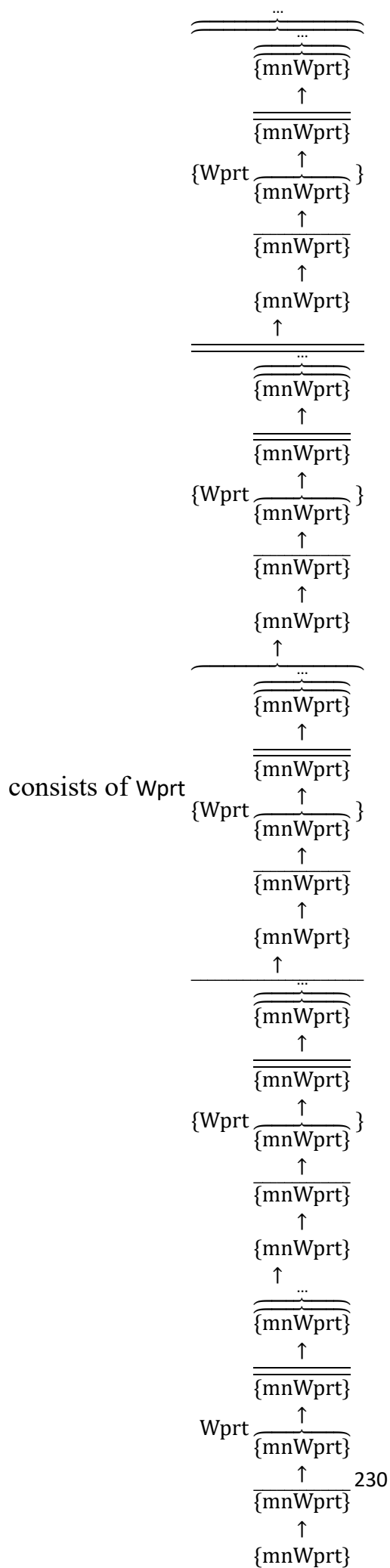
A.10 Wprt-networks

A. Galushkin's comprehensive monograph [19] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new mathematical process with containment operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators. Containment operators are more convenient for networks. Also, the main emphasis was placed on using processors operating using triodes, which are generally not used in Wprt-networks. Wprt networks (SmnWprt) are a Wprt structure that can be built for the required weights, the implementation of which will be carried out using a short-pulse laser to generate attosecond pulses of light. Wprt-OS (Wprt operating system) uses Wprt-coding and Wprt-translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b), where the number b is the code of the action, and the number a is the code of the object of this action. Wprt-coding (or self-coding) is implemented through a matrix consisting of 2 columns (in the continuous case, two intervals of numbers). Here, the source encoding is used for all matrix rows simultaneously. Wprt-translation is carried out by inversion. In this case, self-type coding and self-type translation by (A.2.114) or (A.2.114.1), (A.2.115), (A.2.116) will be more stable. The set of the target



using a short-pulse laser to generate attosecond pulses of light to accomplish them, $x = (x_1, x_2, \dots, x_n)$. We will not touch on the issues of applications, or network optimization. They are described in detail by A. Galushkin [19]. We will touch on the difference of this only for hierarchical complex networks. The same simple executing programs are in the cores of simple artificial neurons of type Wprt (designation - mnWprt) for simple information processing. More complex executing programs are used for mnWprt nodes. The first level of mnWprt consists of simple mnWprt. The second level of mnWprt consists of



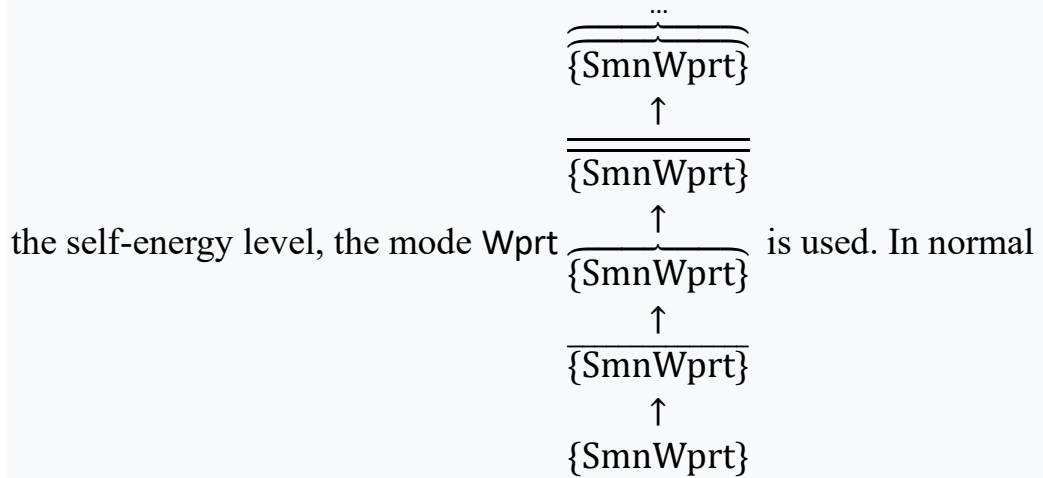


- Wprt²- node of mnWprt. The target structure or the corresponding program enters the target unit using a short-pulse laser to generate attosecond pulses of light. After that, all networks or parts of them are activated according to the indicative goal. It may appear that we are leaving the network ideology, but these networks are a complex hierarchy of different levels, like living organisms.

Remark A.10.1. Traditional scientific approaches through classical mathematics make it possible to describe only at the usual energy level. Here we consider an approach that makes describing processes with finer energies possible. mnWprt contains weprograms –executing program in Wprt-OS. Wprt-OS (or Self-type of Wprt-OS) is based on Wprt-assembly language (or Self-type of Wprt-assembly language), which is based on assembly language through Wprt-approach in turn, if the base of elements of Wprt-networks is sufficient. The reprograms are in Wprt-programming environments (or Self-type of Wprt programming environments), but this question and Wprt-networks base will be considered in the following articles. In particular, weprograms may contain Wprt- programming operators. In mnWprt cores, the constant memory Wprt with correspondent weprograms depending on mnWprt. The OS (operating system) and the principles and modes of operation of the Wprt-networks for this programming are interesting. But this is already the material for the next publications.

Here is developed a helicopter model without a main and tail rotors based on Wprt – physics and special neural networks with artificial neurons operating in normal and Wprt-modes. Let's denote this model through SmnWprt. To do this, it's proposed to use mnWprt of different levels; for example, for the usual mode, mnWprt serves for the initial processing of signals and the transfer of information to the second level, etc., to the nodal center, then checked. In case of an anomaly - local Wprt-mode with the desired "target weight" is realized in this section, etc, to the

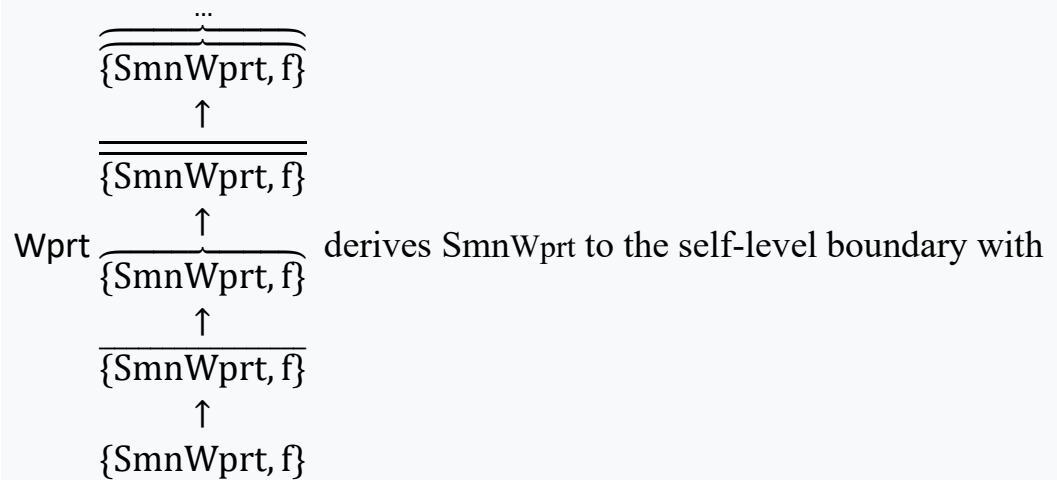
center. In the case of a monster during the test, SmnWprt is activated with the desired "target weight" using a short-pulse laser to generate attosecond pulses of light. Here are realized other tasks also. To reach



mode, it's planned to carry out the movement of SmnWprt on jet propulsion by converting the energy of the emitted gases into a vortex to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the SmnWprt for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is drainage of exhaust gases outside the SmnWprt. SmnWprt is represented by a neural network that extends from the center of one of the main clusters of Wprt - artificial neurons to the shell, turning into the body itself. Above the operator's cabin is the central core of the neural network and the target block, responsible for performing the "target weights" and auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for SmnWprt's actions is below the operator's cab. In Wprt – mode, the entire network or its sections are Wprt – activated to perform specific tasks, in particular, with "target weights" using a short-pulse laser to generate attosecond pulses of light. In the target, block used Wprt -coding, Wprt-translation for activation of all networks to "target weights" simultaneously, then –the

reset of this Wprt-coding after activation using a short-pulse laser to generate attosecond pulses of light.

Unfortunately, triodes are not suitable for Wprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for weprograms may be used instead triodes since there is no necessity to unbend the alternating current to direct. The Wprt-operative memory belt is disposed around a central core of SmnWprt. There are Wprt-coding, Wprt-translation, and Wprt-realize of weprograms and the programs from the archives without extraction, Wprt-coding and Wprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. Wprt – structure or an reprogram if one is present of needed «target weight» are taken in target block at Wprt – activation of the networks.



target weight f . Activation of the entire network is implemented to perform “target weights” using a short-pulse laser to generate attosecond pulses of light.

You can also try to use higher frequency alternating current and ultraviolet light, which can work with Wprt– structures in Wprt–modes by its nature to activate the networks or some of its parts in Wprt–modes and locally using Wprt–mode to perform local tasks. Above high frequently alternating current go through mercury bearers. That’s why

overheating does not occur. The pulse structure of a short-pulse laser for generating attosecond light pulses is close to $(a \uparrow I \downarrow^a) \uparrow I \downarrow (\downarrow_a \downarrow I^a)$, i.e., type ${}_a^a St_a^a$, and upon activation it will be induction of same type self, which is necessary for the formation of a local assembly point d_r of external energy fibers El^{d_r} . Its locality (position of the assembly point r) will be determined by the structure of the magnetic induction of the short-pulse laser pulse for generating attosecond light generation through Targetblock [15] SmnWprt. Execution tw will be achieved through setting the assemblage point in the desired position r_1 to engage the appropriate external energy: $Sprt_{d_r}^{d_r} Sprt_{r_1}^{d_r} [18]$.

A.11 Some aspects of directly parallel operation of neural networks

To design a model of the simplest neural network SmnSprt, you can use a model of the insect CNS. Targetblock will respond to Neurokil. In addition to the material carrier $|||$ (we will call it constant or potential $|||$), it is desirable to use a virtual one $|||$. It is also possible to use virtual connectors, virtual programs, virtual program operators etc. Here you need to use operational $|||$. With the help of programs initiating activation and self-activation SmnSprt will go to the upper levels and initiate execution on them target weights. Let's introduce the designations: CNS_A - CNS of A, $SmnSprt_A$ - SmnSprt on the basis of CNS_A . Lower level: normal direct parallel work of neurons. Middle level: activation for creation self-type energies. Upper level: use of self-activation (subject (host) of activation) for $|||$.

A program operator that specifies the format of the virtual subject's position of $|||$:
 $r_i \equiv \{\} Sprt.$

A program operator that specifies the format of the virtual subject of $|||$:

$$d_{r_i} \equiv Sprt_{\{\} Sprt}^{\{\}.$$

Let's consider $|||$ from outside and inside at the same time by program operator

$$d_{r_i} := Sprt^{\{E^{ex}l^{dr_i}\}}_{\{SmnSprt\}Sprt_{\{dr_{i_{self}}(E_{in}l^{dr_i})\}}}$$

or

$$d_{r_i} := Sprt^{\{E^{ex}l^{dr_i}\}}_{\{self-SmnSprt\}Sprt_{\{dr_{i_{self}}(E_{in}l^{dr_i})\}}}$$

for *self – SmnSprt*.

A program operator Energy shell for *SmnSprt* has the format

$$\text{Energy shell} := Sprt^{\text{Energy shell}}_{Sprt_{\text{Energy shell}}}$$

$$\text{ffg}(r, a(E_q)) =$$

$$\text{ffSprt} \left\{ \begin{matrix} q_{((a \uparrow I \downarrow^a) \uparrow I \downarrow (\downarrow_a I^{\uparrow a}))} \\ W_q fSprt^{E_q}_{q_{((a \uparrow I \downarrow^a) \uparrow I \downarrow (\downarrow_a I^{\uparrow a}))}}, fSprt^{\{E^{ex}l^{dr}\}}_{dr(E_{in}l^{dr})} \end{matrix} \right\}_{\mu_{t_0}} (**_{A.11})$$

$(a \uparrow I \downarrow^a) \uparrow I \downarrow (\downarrow_a I^{\uparrow a})$ -internal energy [18] of SmnfSprt, q- a gap in the energy cocoon of SmnfSprt, r-the position of the assemblage point d_r on the energy cocoon of SmnfSprt, W_q - energy prominences from the gap in the cocoon of a living organism, E_q -external energy entering the gap in the cocoon of SmnfSprt, $E^{ex}l^{dr}$ - a bundle of fibers of external energy self-capacities from outside the cocoon, collected at the point of assembly of the cocoon of SmnfSprt at time t_0 , $E_{in}l^{dr}$ - a bundle of fibers of external energy self-capacities from inside the cocoon, collected at the point of assembly of the cocoon of SmnfSprt in the same position r of the assemblage point d_r . d_r (the virtual subject of |||) is the subject of identifying the energy fibers of the subtle energy of the Universe in position r both outside and inside the cocoon.

Energy of self-SmnfSprt:

ffpg(r, a(E_q)) =

$$\text{ffSprt} \left\{ \begin{array}{l} q((a \uparrow \downarrow a) \uparrow \downarrow (\downarrow a \uparrow a)) \\ W_q f \text{Sprt}_{q((a \uparrow \downarrow a) \uparrow \downarrow (\downarrow a \uparrow a))}^{E_q}, f \text{Sprt}_{dr(\text{self}(E_{in}^{ldr}))}^{\{E_{ex}^{ldr}\}} \end{array} \right\} \mu_{t_0} (***)_{A.11}.$$

Right-hand sides of formulas (**_{A.11}), (***)_{A.11}) can be interpreted as fSrt-program operators or programs.

May consider the next programs or program operators initiating activation and self-activation SmnSprt: $\text{Sprt}_{\text{activation with tw}}^{\text{SmnSprt}}$, $\text{Sprt}_{\text{activation with tw}}^{\text{SmnSprt}}$

$\text{Sprt}_{\text{activation with tw}}^{\text{Sprt}_{\text{activation with tw}}^{\text{SmnSprt}}}$, in particular, for shifting the virtual subject ||| and fixing it in the desired position.

Even more simple network SmnSprt by ion channels of bacteria.

Appendix

Self-function, like self-operator, is its own argument. For example, self-functions of living organisms: DNA division, cells etc. In mathematics self-function f can be

attempted to be displayed as a dynamic operator: $\text{Dprt } f_{\ddot{f}, x}$, where x is its usual

argument, $\ddot{f}$ is interior double of f, self⁻¹-function f can be attempted to be

displayed as a dynamic operator: $(f)^{-1} \text{Dprt } f_{\ddot{f}, x}$, paradoxical self-function(designation:

paself-function) f, like paself-operator, can be attempted to be displayed as a

dynamic operator: $(f)^{-1} \text{Dprt } f_{\ddot{f}, x}$. The potential self-function, like potential self-

operator, is its own argument potentially: by its elements that can generated it etc.

Remark W. "Invisibility" of subtly energies can be explained using the example of

a wave with a propagation law $w(t, k) = c(k)\sin(kt)$, $\lim_{k \rightarrow \infty} c(k) = 0$, $\lim_{k \rightarrow \infty} \frac{c(k)}{k^2} =$

$Q \neq 0 : \frac{d^2 w(t,k)}{dt^2}$ responsible for the force and energy will be a normal (non-zero) value, unlike the amplitude of the wave itself, which will tend to zero, when $k \rightarrow \infty$. When the devices cannot record the wave due to its very small amplitude, the wave will act. Moreover, in the limit we obtain a singularity of the type $\uparrow I \downarrow_{-Q}^Q$.

Remark W1. Information is the interpretation of subtle energies - one of their manifestations. Direct knowledge is the interpretation, in particular, of external singularities through internal ones.

Remark W2. Protection against virus A: $\text{paself}(A)$

Remark W3. Based on SmnSprt [13], using grown elements of the central nervous system similar to the human central nervous system (in particular, neurons), one can try to create an Energy Internet to connect to subtle energies.

Remark W4. Any self-object A can be interpreted as a kind of equation, a problem with an unknown A, and vice versa, any equation, any problem can be presented as a definition of a self-object by $|||$.

Remark W5. Solving problems (equations): 1) by $\text{problemA}(x)_x \text{Sprt}$, 2) by $\frac{\text{problemA}(x)}{\text{problemA}(x)} \text{Sprt}$ or $\text{problemA}(x) |||^{-1} \text{problemA}(x)$, x - the desired value, here removing self with $\text{problemA}(x)$, 3) $\frac{\text{problemA}(x)}{\text{problemA}(x)} \text{Sprt}_{\text{problemA}(x)}^{\text{problemA}(x)}$ or $\text{problemA}(x)_x \text{Sprt}_{\text{problemA}(x)}^x$ - paradoxical solving, SmnSprt $|||$ $\text{problemA}(x)$, the manifestation of this singularity should give x (so-called "direct knowledge").

Similarly for the study of other objects and material processes too etc.

Remark W6. For the development and use of directly parallel algorithms, programs, directly parallel processors and directly parallel RAM are required. For example, the DNA double helix is a directly parallel program and, moreover, a self-program.

Remark W7. Energy of a living organism:

$$\text{ffg}(r, a(E_q)) = \text{ffSprt} \left\{ \begin{matrix} q \left(\begin{matrix} a \\ \mu_1 \text{ffSrt} \mu_1 \\ a \end{matrix} \right) \\ W_q \end{matrix} fSprt^{E_q}_{\left(\begin{matrix} a \\ \mu_1 \text{ffSrt} \mu_1 \\ a \end{matrix} \right)}, fSprt^{\{E^{ex} l^{dr}\}}_{dr(E_{in} l^{dr})} \right\} \quad (**_{2.1}).$$

μ
 t_0

$\begin{matrix} a & a \\ \mu_1 \text{ffSrt} \mu_1 & \\ a & a \end{matrix}$ -internal energy of a living organism, q- a gap in the energy cocoon of a living organism, r-the position of the assemblage point d_r on the energy cocoon of a living organism, W_q - energy prominences from the gap in the cocoon of a living organism, E_q -external energy entering the gap in the cocoon of a living organism, $E^{ex} l^{dr}$ - a bundle of fibers of external energy self-capacities from outside the cocoon, collected at the point of assembly of the cocoon of a living organism at time t_0 , $E_{in} l^{dr}$ - a bundle of fibers of external energy self-capacities from inside the cocoon, collected at the point of assembly of the cocoon of a living organism in the same position r of the assemblage point d_r . d_r is the subject of identifying the energy fibers of the subtle energy of the Universe in position r both outside and inside the cocoon.

Energy with measure of fuzziness μ_1, μ_2 of a living organism of a person:

$$\text{ffpg}(r, a(E_q)) = \text{ffSprt} \left\{ \begin{matrix} q \left(\begin{matrix} a \\ \mu_1 \text{ffSrt} \mu_1 \\ a \end{matrix} \right) \\ W_q \end{matrix} fSprt^{E_q}_{\left(\begin{matrix} a \\ \mu_1 \text{ffSrt} \mu_1 \\ a \end{matrix} \right)}, fSprt^{\{E^{ex} l^{dr}\}}_{dr(\text{self}(E_{in} l^{dr}))} \right\}$$

μ
 t_0

(**_{2.1}).

(**_{2.1}), (**_{2.1}) can be interpreted as PrfSrt- program operators.

$\overline{\overline{\text{ffSprt}}}$

=

ffSprt

$$\left\{ \text{ffpg}(r_1, a(E_q)), \text{ffpg}(r_2, a(E_q)), \dots, \text{ffpg}(r_i, a(E_q)), \dots, \text{ffpg}(r_N, a(E_q)) \right\}$$

μ
 $\overline{\overline{\text{ffSprt}}}$

(**_2

2), N is the number of assemblage point positions. This is the definition of $\overline{ffSprt}$ -
 $\{\}$
singularity of exit to a higher level. r_i by its action = $ffSprt \mu$, an assemblage point
 $\{\}$
 μ
 d_{r_i} by its action = $ffSprt \{\}$.
 $\mu ffSprt$
 $\{\}$

Remark W8. Directly parallel actions in any experimental science lead to self-type structures, for example, directly parallel evidence (directly parallel logic).

Remark W9. Let us introduce the notations of the $|||$ result - $|||rt$ and $|||$ process - $|||prt$. Let us introduce the notation of absolute chaos: $\bowtie$. Singularity $\forall |||no$ creates a field. The absolute change is the absolute chaos. The absolute rest is the emptiness.

Remark W10. Chronic, systemic diseases form in the body self-type harmful structures, for example, $Sprt_{disease A}^{disease A}$, that is why it is so difficult to cure them. To cure them needs $\frac{disease A}{disease A} Sprt$ or $|||^{-1} disease A$ etc.

Remark W11. Induction from change is self- change (change spirit).

Remark W12. Induction from $|||$ is self- $|||$ ($|||$ spirit).

REFERENCES:

- 1) Oleksandr Danilishyn, Illia Danilishyn. Dynamical Sets Theory: S2t-Elements and Their Applications. J Math Techniques Comput Mat h, 2(12), 2023, 479-498. <https://www.opastpublishers.com/open-access-articles/dynamical-sets-theory-s2telements-and-their-applications.pdf>
- 2) Danilishyn I., Danilishyn O. Dynamic Sets Set and Some of Their Applications to Neuroscience, Networks Set New Advances in Brain & Critical Care 2023, Volume 4 | Issue 2 | 66- 81. <https://doi.org/10.33140/NABCC.04.02.02>
- 3) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets S1et and Some of their Applications in Physics. Science Set Journal of Physics, 25 September 2023,1-11, 815-_ article1695707465.pdf (mkscienceset.com)

- 4) I. Danilishyn, O. Danilishyn Dynamic Sets Se, Networks Se. *Advances in Neurology and Neuroscience*, 2023, Volume 6 | Issue 2 | 278-294.
<https://www.opastpublishers.com/open-access-articles/dynamic-sets-se-networks-se.pdf>
- 5) I. Danilishyn, O. Danilishyn Program Operators Sit, tS, S1e, Set1. Journal of Sensor Networks and Data Communications [ISSN: 2994-6433] 2023, Volume 3 | Issue 1 | 138-143, <https://www.opastpublishers.com/table-contents/jsndc-volume-3-issue-1-year-2023>
- 6) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 7) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the V International Scientific and Practical Conference, Oxford, June 23, 2023. Oxford-Vinnytsia: P.C. Publishing House & European Scientific Platform, 2023. Theoretical and empirical scientific research: concept and trends. **pp.156-161**, DOI 10.36074/logos-23.06.2023.42, <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/12>
- 8) Danilishyn I.V. Danilishyn O.V. SET1 – ELEMENTS. INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 28 June, 2023 with the proceedings of the: Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 9 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC International Centre Corporative Management (Vienna, Austria), **pp. 239-254**.
<https://archive.journal-grail.science/index.php/2710-3056/issue/view/09.06.2023>

- 9) Danilishyn I.V. Danilishyn O.V. DYNAMICAL SIT-ELEMENTS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 10) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO SETS THEORY AND OTHERS. Scientific practice: modern and classical research methods: Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the IV International Scientific and Practical Conference, Boston, May 26, 2023. Boston-Vinnytsia: Primedia eLaunch & European Scientific Platform, 2023, pp.166-171. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/11>
- 11) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO CONTINUAL VALUED LOGIC AND OTHERS. Features of the development of modern science in the pandemic's era: a collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, May 19, 2023. Berlin, Federal Republic of Germany: European Scientific Platform, pp.79-84.
<https://previous.scientia.report/index.php/archive/issue/view/19.05.2023>
- 12) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. [Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1).
<https://doi.org/10.21203/rs.3.rs-3217178/v1>
[Dynamic Sets Theory: Sit-elements and Their Applications | Research Square](https://www.researchsquare.com/article/rs-3217178/v1)
www.researchsquare.com/article/rs-3217178/v1
- 13) Illia Danilishyn, Oleksandr Danilishyn and Volodymyr Pasyukov. Some aspects of directly parallel operation of neural networks and their subtle energy Advances in Machine& Artificial. Volume 5 | Issue 4 (2024), pp.1 - 7. Intelligence<https://www.opastpublishers.com/journal/advances-in-machine-learning-artificial-intelligence/articles-in-press>

- 14) Oleksandr Danilishyn, Illia Danilishyn and Volodymyr Pasyнков.
 Intrtroduction to section of Dynamic mathematics: Theory of singularities of the
 type synthesizing. J Math Techniques Comput Math (*Journal*
DOI: 10.33140/JMTCM), Volume 3 | Issue 10 (2024), pp.1- 18.
[https://www.opastpublishers.com/journal/journal-of-mathematical-
 techniques-and-computational-mathematics/articles-in-press](https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press)

- 15) Illia Danilishyn, Oleksandr Danilishyn. Hierarchical dynamic mathematical
 structures (models) theory: Scientific monograph DOI
<https://doi.org/10.36074/hdmsmt.monograph-2024> Published 2024-08- 28
 Shawnee, USA Primedia eLaunch LLC 2024 <https://bookwire.bowker.com> :
 9798892178099

- 16) Danilishyn I.V. Danilishyn O.V. CONTINUAL SIT-ELEMENTS AND
 DYNAMICAL SIT-ELEMENTS. Theoretical and practical aspects of
 modern scientific research:Collection of scientific papers «ΛΟΓΟΣ» with
 Proceedings of the IIIInternational Scientific and Practical Conference,
 Seoul, April 28, 2023. Seoul-Vinnytsia: Case Co., Ltd.& European Scientific
 Platform, pp.144-150. [https://archive.logos-
 science.com/index.php/conference-proceedings/issue/view/10](https://archive.logos-science.com/index.php/conference-proceedings/issue/view/10)

- 17) Oleksandr Danilishyn and Illia Danilishyn. Introduction to Dynamic
 Operators: Rprt-Elements and Their Applications. Rprt-Networks. Variable
 Fuzzy Hierarchical Dynamic Fuzzy Structures (Models, Operators) for
 Dynamic, Singular, Hierarchical Fuzzy Sets. Fuzzy Program Operators fRprt,
 ftprR, ffR1epr, ffReprt1. J Math Techniques Comput Math (*Journal*
DOI: 10.33140/JMTCM), Volume 3 | Issue 9 (2024), pp.1- 26.
[https://www.opastpublishers.com/journal/journal-of-mathematical-
 techniques-and-computational-mathematics/articles-in-press](https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press)

- 18) Danilishyn I., Danilishyn O. Introduction to Dynamic Mathematics: dynamic
 sets, dynamic operators and their applications: monograph / ed. by Pasyнков V.

– Shawnee, USA : Primedia eLaunch LLC. – 2024. – 442 p. ISBN 979-8-89217-

806-8 DOI: <https://doi.org/10.36074/itdmdsdoata.monograph-2024>

https://books.google.com.ua/books/about?id=R3Y6EQAAQBAJ&redir_esc=y

<https://explore.openaire.eu/search/publication?pid=10.36074%2Fitdmdsdoata.monograph-2024>

- 19) Galushkin A. Networks: principles of the theory. Hot line-Telecom. M.,2010 (in Russian).

20)

Declarations:

Availability of data and material

- 1) Oleksandr Danilishyn, Illia Danilishyn. Dynamical Sets Theory: S2t-Elements and Their Applications. J Math Techniques Comput Math, 2(12), 2023, 479-498. <https://www.opastpublishers.com/open-access-articles/dynamical-sets-theory-s2telements-and-their-applications.pdf>
- 2) Danilishyn I., Danilishyn O. Dynamic Sets Set and Some of Their Applications to Neuroscience, Networks Set New Advances in Brain & Critical Care 2023, Volume 4 | Issue 2 | 66- 81. <https://doi.org/10.33140/NABCC.04.02.02>
- 3) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets S1et and Some of their Applications in Physics. Science Set Journal of Physics, 25 September 2023,1-11, 815-_ article1695707465.pdf (mkscienceset.com)

- 4) I. Danilishyn, O. Danilishyn Dynamic Sets Se, Networks Se. *Advances in Neurology and Neuroscience*, 2023, Volume 6 | Issue 2 | 278-294.
<https://www.opastpublishers.com/open-access-articles/dynamic-sets-se-networks-se.pdf>
- 5) I. Danilishyn, O. Danilishyn Program Operators Sit, tS, S1e, Set1. *Journal of Sensor Networks and Data Communications* [ISSN: 2994-6433] 2023, Volume 3 | Issue 1 | 138-143, <https://www.opastpublishers.com/table-contents/jsndc-volume-3-issue-1-year-2023>
- 6) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 7) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the V International Scientific and Practical Conference, Oxford, June 23, 2023. Oxford-Vinnytsia: P.C. Publishing House & European Scientific Platform, 2023. Theoretical and empirical scientific research: concept and trends. pp.156-161, DOI 10.36074/logos-23.06.2023.42, <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/12>
- 8) Danilishyn I.V. Danilishyn O.V. SET1 – ELEMENTS. INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 28 June, 2023 with the proceedings of the: Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 9 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC International Centre Corporative Management (Vienna, Austria), pp. 239-254. <https://archive.journal-grail.science/index.php/2710-3056/issue/view/09.06.2023>

- 9) Danilishyn I.V. Danilishyn O.V. DYNAMICAL SIT-ELEMENTS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 10) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO SETS THEORY AND OTHERS. Scientific practice: modern and classical research methods: Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the IV International Scientific and Practical Conference, Boston, May 26, 2023. Boston-Vinnytsia: Primedia eLaunch & European Scientific Platform, 2023, pp.166-171. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/11>
- 11) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO CONTINUAL VALUED LOGIC AND OTHERS. Features of the development of modern science in the pandemic's era: a collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, May 19, 2023. Berlin, Federal Republic of Germany: European Scientific Platform, pp.79-84. <https://previous.scientia.report/index.php/archive/issue/view/19.05.2023>
- 12) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. [Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1). <https://doi.org/10.21203/rs.3.rs-3217178/v1>
[Dynamic Sets Theory: Sit-elements and Their Applications | Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1)
[www.researchsquare.com/article/rs-3217178/v1](https://doi.org/10.21203/rs.3.rs-3217178/v1)
- 13) Illia Danilishyn, Oleksandr Danilishyn and Volodymyr Pasyнков. Some aspects of directly parallel operation of neural networks and their subtle energy

Advances in Machine& Artificial. Volume 5 | Issue 4 (2024), pp.1 - 7.

Intelligence<https://www.opastpublishers.com/journal/advances-in-machine-learning-artificial-intelligence/articles-in-press>

- 14) Oleksandr Danilishyn, Illia Danilishyn and Volodymyr Pasyukov.
Introduction to section of Dynamic mathematics: Theory of singularities of the type synthesizing. J Math Techniques Comput Math (*Journal DOI: 10.33140/JMTCM*), Volume 3 | Issue 10 (2024), pp.1- 18.
<https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press>
- 15) Illia Danilishyn, Oleksandr Danilishyn. Hierarchical dynamic mathematical structures (models) theory: Scientific monograph DOI <https://doi.org/10.36074/hdmsmt.monograph-2024> Published 2024-08- 28 Shawnee, USA Primedia eLaunch LLC 2024 <https://bookwire.bowker.com> : 9798892178099
- 16) Danilishyn I.V. Danilishyn O.V. CONTINUAL SIT-ELEMENTS AND DYNAMICAL SIT-ELEMENTS. Theoretical and practical aspects of modern scientific research:Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the IIIInternational Scientific and Practical Conference, Seoul, April 28, 2023. Seoul-Vinnytsia: Case Co., Ltd.& European Scientific Platform, pp.144-150. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/10>
- 17) Oleksandr Danilishyn and Illia Danilishyn. Introduction to Dynamic Operators: Rprt-Elements and Their Applications. Rprt-Networks. Variable Fuzzy Hierarchical Dynamic Fuzzy Structures (Models, Operators) for Dynamic, Singular, Hierarchical Fuzzy Sets. Fuzzy Program Operators fRprt, ftprR, ffR1epr, ffReprt1. J Math Techniques Comput Math (*Journal DOI: 10.33140/JMTCM*), Volume 3 | Issue 9 (2024), pp.1- 26.
<https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press>

- 18) Danilishyn I., Danilishyn O. Introduction to Dynamic Mathematics: dynamic sets, dynamic operators and their applications: monograph / ed. by Pasyukov V. – Shawnee, USA : Primedia eLaunch LLC. – 2024. – 442 p. ISBN 979-8-89217-806-8
- DOI: <https://doi.org/10.36074/itdmdsdoata.monograph-2024>
- https://books.google.com.ua/books/about?id=R3Y6EQAAQBAJ&redir_esc=y
- <https://explore.openaire.eu/search/publication?pid=10.36074%2Ftdmdsdoata.monograph-2024>

Competing interest

There are no competing interests. All sections of the article are executed jointly.

Funding

There were no sources of funding for writing the article.

Authors' contributions

The contribution of the authors is the same, we will not separate.

Part IV. Mathematical fundamentals of the Dynamic programming

Introduction

Dynamic programming works in the space of energies with energies. It is enough to imagine the world as a software environment. Then objects and processes will act as variables etc. The subjects of variables are processes of a higher level manipulating them. They are often called spirits, which leads to distraction from the real state of processes, to confusion. $\equiv$ (recurrent identity) is the dynamic assignment (recurrent assignment (when subsequent values of variables are ||| through the previous ones)). Differs from the usual software environment and ||| with the right part. Directly parallel (dynamic) software operators are used, programs like DNA in structure, but so far much simpler. Unfortunately, due to the lack of funding and laboratories, there is no material base for dynamic programming (the so-called "hardware"). Dynamic programming will do what is needed if the level and volume of energies allow, i.e., much more effectively than science will allow, as well as our supposed devices on Dynamic programming. Dynamic programming is the science-action (Dynamic science). At this level, you can form anything you want. Also, our Dynamic Mathematics is only auxiliary in relation to Dynamic Programming. Dynamic programming is not a hypothesis. This is the truth that has been given to us by many remarkable researchers in an implicit form. When a regular computer works with SmnSprt, there will be transitions from regular programming to dynamic programming and vice versa. It is also possible to use infinite cycles to create the simplest self-energies, program calls to themselves, obvious contradictions in regular programs that will automatically launch dynamic programming. Dynamic programming also has different levels. In the monograph, we consider its lower level. At the middle level, it works with all types of self-type structures, at the upper ones - with all singularities and with itself in all possible upper modes (Self-development, self-clotting). Self-development of Dynamic programming can be, for example, in the presence of its "embryo", for example, in the structure

$Sprt_x^{\{\text{Dynamic programming, its "embryo"}\}}$. Then from this "embryo" it can grow a virtual double with much greater capabilities.

4.1 Program operators Sprr-type

Introduction

There is a need to develop an instrumental mathematical base for new technologies. An example of a parallel program and self-program is the DNA double helix. Therefore, our mathematics is adapted not only to obtain results, but also to directly control actions, which will certainly show its benefits on a fundamentally new type of neural networks with directly parallel calculations, for which it was created [2].

4.1.1 Program operators Sprr-type

For simultaneous multiplication: $Sprt_x^{\{a^*\}}$: the notation of the set B with elements $b_{i_1 i_2 \dots i_n} =$

$(Sprt_x^{\{a_{i_1}^*, a_{i_2}^*, \dots, a_{i_n}^*\}})_R$ for any $\{i_1, i_2, \dots, i_n\}$ without repetitions, $x = Sprt_w^{\{K\}}$, K-set of any $\{k_1^*, k_2^*, \dots, k_n^*\}$ without repeating them, k_i -any digit, $i=1, 2, \dots, n$, $R = Sprt_w^{\{i_1 +, i_2 +, \dots, i_n\}}$, R is the index of the lower discharge (we choose an index on the scale of discharges): see Table I.1.

Then $Sprt_x^{\{B^+\}}$ gives the final result of simultaneous multiplication. Any system of calculus can be chosen, in particular binary. The most straightforward functional scheme of the assumed arithmetic-logical device for Sprr-multiplication:

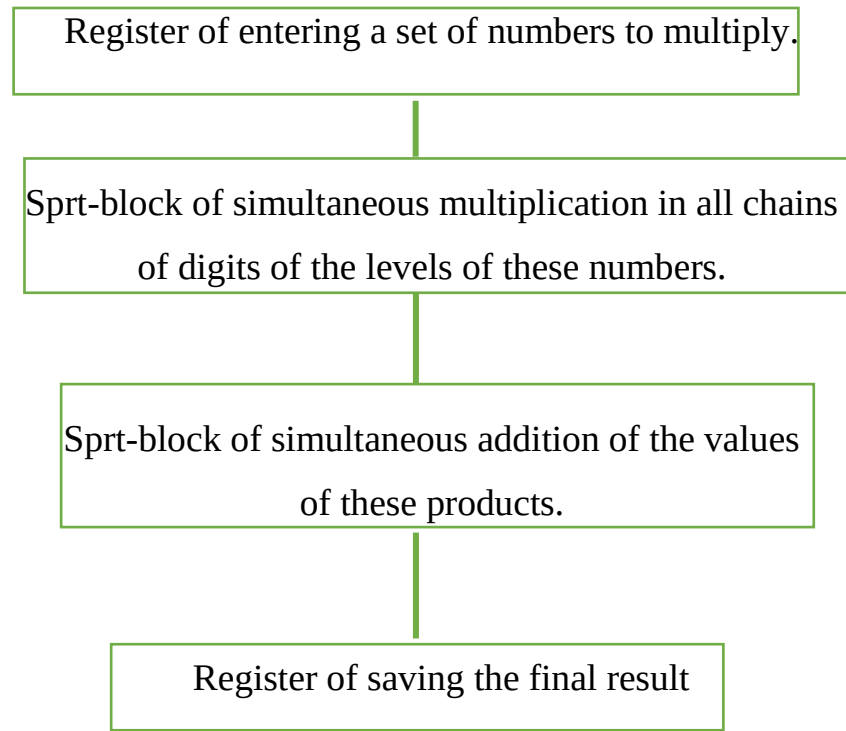


Fig. 4.1. The straightforward functional scheme of the assumed arithmetical device for Sprt-multiplication.

Remark 4.1.1. The algorithm for simultaneously multiplication of a set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by multiplying the first number from the set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the set by the ones following it, etc.

Here are some of the Sprt programming operators [2]:

1. Simultaneous assignment of the expressions $\{p\} = (p_1, p_2, \dots, p_n)$ to the variables $\{a\} = (a_1, a_2, \dots, a_n)$. This is implemented via $Sprt_x^{\{\{a\}:=\{p\}\}}$.
2. Simultaneous checking the set of conditions $\{f\} = (f_1, f_2, \dots, f_n)$ for the set of expressions $\{B\} = (B_1, B_2, \dots, B_n)$. Implemented via $Sprt_x^{IF\{\{B\}\{f\}\} then Q}$ where Q can be anything.
3. Similarly for loop operators and others.

S_3f – software operators will differ only in that the aggregates $\{a\}, \{p\}, \{B\}, \{f\}$ will be formed from the corresponding Sprt program operators in form (1.1) and for more complex operators in the form from the forms (1.1.1) – (1.4).

The OS (operating system), the computer's principles, and the modes of operation for this programming are interesting. But this is already the material for the following monographs.

The ideology of dynamic Sprt and $S_3f(t)$ can be used for programming [5]:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t), p_2(t), \dots, p_n(t))$ to the variables $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$ is implemented through $Sprt(t)_x^{\{\{g(t)\} := \{p(t)\}\}}$.
2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f_2(t), \dots, f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), \dots, B_n(t))$ is implemented through $Sprt(t)_x^{IF\{\{B(t)\}\{f(t)\}\} \text{ then } Q(t)}$ where $Q(t)$ can be any.

3. Similarly for loop operators and others.

$S_3f(t)$ – software operators will differ only in that the aggregates $\{a\}, \{p\}, \{B(t)\}, \{f(t)\}$ will be formed from corresponding processes $Sprt(t)$ for the above-mentioned programming operators through form (1.1) and for more complex operators in the form from the forms (1.1.1) – (1.4). We can consider the concept of a continual Sprt - element as $Sprt_B^A$, where A fits in continual capacity B. Then $Sprt_B^B$ will mean S_1f B.

These elements are used for Sprt-coding, Sprt translation, coding self, and translation self for networks], which is suitable for electric current of ultrahigh frequency.

4.1.2 The usage of Sprt-elements for networks

A. Galushkin's comprehensive monograph [9] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new

mathematical process with containment operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators [2]. Containment operators are more convenient for networks. Also, the main emphasis was placed on using processors operating using triodes, which are generally not used in Sprt-networks. Sprt-networks ($Smnsprt$) are a Sprt-structure that can be built for the required weights. Sprt-OS (Sprt operating system) uses Sprt-coding and Sprt-translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b) , where the number b is the code of the action, and the number a is the code of the object of this action. Sprt-coding (or self-coding) is implemented through a matrix consisting of 2 columns (in the continuous case, two intervals of numbers). Here, the source encoding is used for all matrix rows simultaneously. Sprt-translation is carried out by inversion. In this case, self-coding and self-translation will be more stable. The target weights f_i in $Sprt_a^{\{fx\}}$ are chosen for necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [3]. We will touch on the difference of this only for hierarchical complex networks. The same simple executing programs are in the cores of simple artificial neurons of type Sprt (designation - $mnSprt$) for simple information processing. More complex executing programs are used for $mnSprt$ nodes. Sprt-threshold element $-\text{sgn}(Sprt_b^{\{ax\}})$, b - $mnSprt$, $x=(x_1, x_2, \dots, x_n)$ – source signals values, $a=(a_1, a_2, \dots, a_n)$ – Sprt-synapses weights. The first level of $mnSprt$ consists of simple $mnSprt$. The second level of $mnSprt$ consists of $Sprt_D^{\{mnSprt\}}$ – Sprt-node of $mnSprt$ in range D , D - capacity for $mnSprt$ node. The third level of $mnSprt$ consists of $Sprt_D^{\{St_D^{\{mnSt\}}\}}$ - $Sprt^2$ - node of $mnSprt$ in range D , thus D becomes capacity of itself in itself as an element for $mnSprt$. For our networks, it is sufficient to use $Sprt^2$ - nodes of $mnSprt$, but self-level is higher in living organisms, particularly $Sprt^n$ -, $n \geq 3$. The target structure or the corresponding program enters the target unit using alternating current. After that, all networks or parts of them are activated according to the indicative goal. It may appear that we

are leaving the network ideology, but these networks are a complex hierarchy of different levels, like living organisms.

Threshold element of extended type: $SCprt \rightarrow \begin{matrix} b \\ g_2 \\ \{qy\} \end{matrix} SCprt \begin{matrix} \{ax\} \\ g_1 \\ b \end{matrix}$, b- artificial neurons of

type SCprt (designation - mnSt), $x = (x_1, x_2, \dots, x_n)$ are the values of the initial signals,, $a = (a_1, a_2, \dots, a_n)$ are the weights of SCprt-synapses and the values of the output signals .

Remark 4.2.1. Traditional scientific approaches through classical mathematics make it possible to describe only at the usual energy level. Here we consider an approach that makes describing processes with finer energies possible. mnSprt contains $Sprt_{mnSprt}^{\{eprograms\}}$, eprogram –executing program in Sprt- OS. Sprt-OS (or Self-OS) is based on Sprt-assembly language (or Self-assembly language), which is based on assembly language through Sprt-approach in turn, if the base of elements of Sprt-networks is sufficient. The eprograms are in Sprt-programming environments (or Self-programming environments), but this question and Sprt-networks base will be considered in the following monographs. In particular, eprograms may contain Sprt- programming operators. In mnSprt cores, the constant memory Sprt with correspondent eprograms depending on mnSprt.

The OS (operating system) and the principles and modes of operation of the Sprt-networks for this programming are interesting. But this is already the material for the next publications.

Here is developed a helicopter model without a main and tail rotors based on Sprt – physics and special neural networks with artificial neurons operating in normal and Sprt-modes. Let's denote this model through SmnSprt. To do this, it's proposed to use mnSprt of different levels; for example, for the usual mode, mnSprt serves for the initial processing of signals and the transfer of information to the second level, etc., to the nodal center, then checked. In case of an anomaly - local Sprt–mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, SmnSprt is activated with the desired "target

weight." Here are realized other tasks also. To reach the self-energy level, the mode $Sprt_{Smnsprt}^{Smnsprt}$ is used. In normal mode, it's planned to carry out the movement of SmnSprt on jet propulsion by converting the energy of the emitted gases into a vortex to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the SmnSprt for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is drainage of exhaust gases outside the SmnSprt. SmnSprt is represented by a neural network that extends from the center of one of the main clusters of Sprt - artificial neurons to the shell, turning into the body itself. Above the operator's cabin is the central core of the neural network and the target block, responsible for performing the "target weights" and auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for SmnSprt's actions is below the operator's cab. In Sprt – mode, the entire network or its sections are Sprt – activated to perform specific tasks, in particular, with "target weights." In the target, block used Sit-coding, Sprt-translation for activation of all networks to "target weights" simultaneously, then –the reset of this Sprt-coding after activation. Unfortunately, triodes are not suitable for Sprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for eprograms may be used instead triodes since there is no necessity to unbend the alternating current to direct. The Sprt-operative memory belt is disposed around a central core of SmnSprt. There are Sprt-coding, Sprt-translation, and Sprt-realize of eprograms and the programs from the archives without extraction, Sprt-coding and Sprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. Sprt – structure or an eprogram if one is present of needed «target weight» are taken in target block at Sprt – activation of the networks. $Sprt_{activation}^{Smnsprt,f}$ derives SmnSprt to the self-level boundary with target weight f.

It's used an alternating current of above high frequency and ultra-violet light, which can work with Sprt – structures in Sprt–modes by its nature to activate the networks or some of its parts in Sprt–modes and locally using Sprt–mode. Above high frequently alternating current go through mercury bearers. That's why overheating does not occur. The power of the alternating current above high frequently increases considerably for the target block. The activation of all networks is realized to indicate “target weights.”

4.2 Program operators ffSprt-type

For simultaneous multiplication $\tilde{x} * \mu_q$: the notation of the fuzzy set B with

elements $b_{i_1 i_2 \dots i_n} =$

$$(\text{ffSprt } \mu_q \{x_{1_{i_1}} * \mu_{\tilde{x}}(x_{1_{i_1}}) *, x_{2_{i_2}} * \mu_{\tilde{x}}(x_{2_{i_2}}) *, \dots, x_{n_{i_n}} * \mu_{\tilde{x}}(x_{n_{i_n}})\})_R$$

for any $\{i_1, i_2, \dots, i_n\}$ without repetitions, $q = St_w^{\{K\}}$, K-set of any

$\{k_1 *, k_2 *, \dots, k_n *\}$ without repeating them, k_i -any digit, $i=1,2,\dots,n$, $R=$

$St_w^{\{i_1+i_2+\dots+i_n\}}$, R is the index of the lower discharge (we choose an index on the scale of discharges): see Table I.1.

$B +$

Then $\text{ffSprt } \mu_q$ gives the final result of simultaneous multiplication. Any

system of calculus can be chosen, in particular binary. The most

straightforward functional scheme of the assumed arithmetic-logical device

for ffSprt -multiplication:

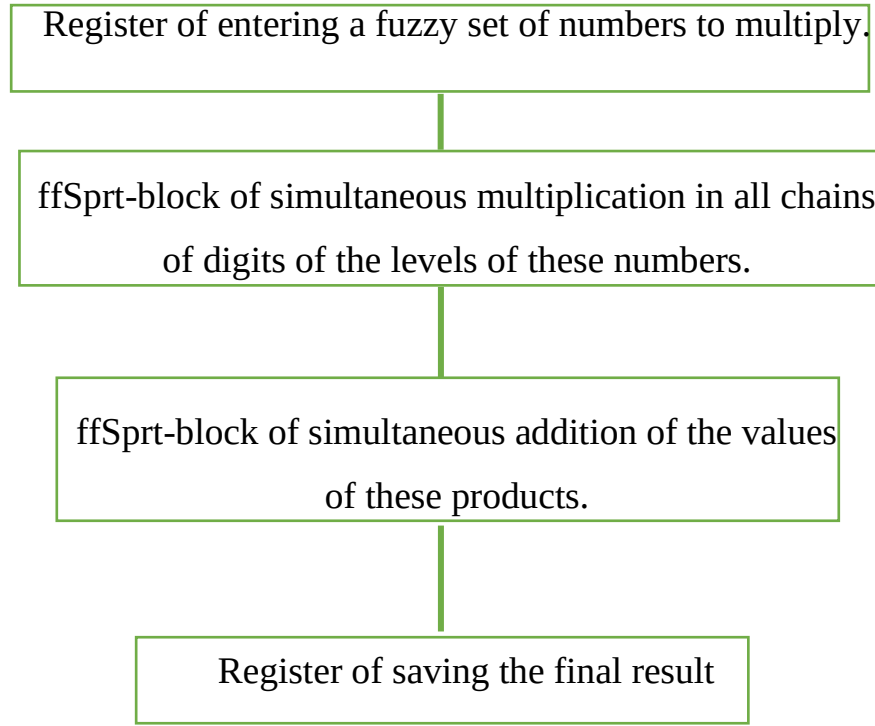


Fig. 4.2. The straightforward functional scheme of the assumed arithmetic-logical device for ffSprt-multiplication.

Remark 4.2.1. The algorithm for simultaneously adding a set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by multiplying the first number from the fuzzy set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the fuzzy set by the ones following it, etc.

Here are some of the ffSprt programming operators:

1. Simultaneous fuzzy assignment of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2)|, \dots, p_n|\mu_{\tilde{p}}(p_n)|)$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|, \dots, x_n|\mu_{\tilde{x}}(x_n)|)$.

This is implemented via ffSprt $\left\{ \begin{matrix} \{\tilde{x}\} := \{p\} \\ \mu \\ q \end{matrix} \right\}$.

2. Simultaneous fuzzy checking the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2)|, \dots, g_n|\mu_{\tilde{g}}(g_n)|)$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1),$

$B_2|\mu_{\tilde{B}}(B_2)|, \dots, x_n|\mu_{\tilde{B}}(B_n)|$) Implemented via $\text{ffSprt} \underset{\mu}{\overset{IF \{ \{\tilde{B}\} \{\tilde{g}\} \}}{\text{then } \tilde{Q}}}$. where q

$\tilde{Q}$ can be anything.

3. Similarly for loop operators and others.

$\text{ffS}_3f\mu$ – fuzzy software operators will differ only in that the aggregates $\{\tilde{x}\}, \{\tilde{p}\}, \{\tilde{B}\}, \{\tilde{g}\}$ will be formed from the corresponding ffSprt program operators in form (1.1) or forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*) for more complex operators.

The OS (operating system), the computer's principles, and the modes of operation for this programming are interesting. But this is already the material for the following publications.

The ideology of fuzzy dynamic $f\text{Sprt}$ and $\text{ffS}_3f(t)$ can be used for programming:

1. The process of simultaneous fuzzy assignment of the fuzzy

expressions $\widetilde{p(t)} = (p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)),$

$p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t))|, \dots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t))|)$ to the fuzzy variables

$\widetilde{x(t)} = (x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t))|, \dots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t))|)$. This is

implemented via $\text{ffSprt}(t) \underset{\mu(t)}{\overset{\{ \{ \widetilde{x(t)} \} := \{ \widetilde{p(t)} \} \}}{\text{then } \widetilde{Q(t)}}}$.

2. The process of simultaneous checking the fuzzy set of conditions

$\widetilde{g(t)} = (g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t))|, \dots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t))|)$ for the

fuzzy set of expressions $\widetilde{B(t)} = (B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)),$

$B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t))|, \dots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t))|)$. Implemented via $fSt(t)_{w(t)}$

$\text{ffSprt}(t) \underset{\mu(t)}{\overset{IF \{ \{ \widetilde{B(t)} \} \{ \widetilde{g(t)} \} \}}{\text{then } \widetilde{Q(t)}}}$ where $\widetilde{Q(t)}$ can be anything.

3. Similarly for fuzzy loop operators and others.

$ffS_3f(t)_{\mu(t)}$ – fuzzy fsoftware operators will differ only just because aggregates $\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding processes by ffSprt-program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

ffSprt(t) continual elements are used for ffSprt-coding, ffSprt-translation, fuzzy coding fself, and fuzzy translation fself for networks, which is suitable for electric current of ultrahigh frequency.

4.2.1 The usage of ffSprt-elements for networks

A. Galushkin's comprehensive monograph [3] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new mathematical process with fuzzy containment operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators [2]. Containment operators are more convenient for networks. Also, the main emphasis was placed on using processors operating using triodes, which are generally not used in ffSprt -networks. ffSprt-networks(ffSmnsprt) are a ffSprt -structure that can be built for the required weights. ffSprt -OS (ffSprtoperating system) uses ffSprt -coding and ffSprt -translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b), where the fuzzy number b is the code of the action, and the fuzzy number a is the code of the object of this action. ffSprt -coding (or fself- (fuzzy coding)) is implemented through a fuzzy matrix consisting of 2 columns (in the continuous case, two intervals of fuzzy numbers). Here, the source encoding is used for all matrix rows simultaneously. ffSprt -translation is carried out by inversion. In this case, fself- (fuzzy coding) and fself- (fuzzy translation) will be

more stable. The target weights $q_i(t)$ in $ffSprt(t)$ $\frac{\widetilde{x(t)}q(t)}{\mu(t)} \frac{\widetilde{w(t)}}{\mu(t)}$ are chosen for

necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [3]. We will touch on the

difference of this only for hierarchical complex networks. The same simple executing programs are in the cores of simple artificial neurons of type $ffSprt(\text{designation} - mnffSprt)$ for simple information processing. More complex executing programs are used for $mnffSprt$ nodes. Threshold element $ffSprt -$

$\begin{matrix} b \\ \mu_2 \\ \{qy\} \end{matrix} \begin{matrix} ffSprt(t) \\ \mu_1 \\ b \end{matrix} \{a\tilde{x}\}$, b - artificial neurons of type $ffSprt$ (designation - $mnffSprt$) ,

$\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2)|, \dots, x_n|\mu_{\tilde{x}}(x_n)|)$ is the fuzzy set of values of the initial signals, $a=(a_1, a_2, \dots, a_n)$ are the weights of Sit-synapses and $\tilde{y}=(y_1|\mu_{\tilde{y}}(y_1),$

$y_2|\mu_{\tilde{y}}(y_2)|, \dots, y_n|\mu_{\tilde{y}}(y_n)|)$ is the fuzzy set of values of the output signals with weights

$q=(q_1, q_2, \dots, q_n)$.. The first level of $mnffSprt$ consists of simple $mnffSprt$. The

second level of $mnffSprt$ consists of $fSt_D^{\{mnSt\}}$ – Sit-node of $mnffSprt$ in range D ,

D - fuzzy capacity for $mnffSprt$ node. The third level of $mnffSprt$ consists of

$\begin{matrix} mnffSprt \\ ffSprt \\ \mu \\ D \end{matrix}$ - $ffSprt^2$ - node of $mnffSprt$ in range D , thus D becomes fuzzy

capacity of itself in itself as an element for $mnffSprt$. For our networks, it is sufficient to use $ffSprt^2$ - nodes of $mnffSprt$, but $fself$ -level is higher in living organisms, particularly $ffSprt^n$ -, $n \geq 3$. The target structure or the corresponding program enters the target unit using alternating current. After that, all networks or parts of them are activated according to the indicative goal. It may appear that we are leaving the network ideology, but these networks are a complex hierarchy of different levels, like living organisms.

Remark 4.2.2. Traditional scientific approaches through classical mathematics make it possible to describe only at the usual energy level. Here we consider an approach that makes describing processes with finer energies possible. $mnffSprt$

contains $\begin{matrix} \{ffeprograms\} \\ \mu \\ mnffSprt \end{matrix}$, $ffeprogram$ –executing program in $ffSprt - OS$.

$ffSprt -OS$ (or $Fself-OS$) is based on $ffSprt$ -assembly language (or $ffself$ -assembly language), which is based on assembly language through Sit-approach in turn, if

the base of elements of ffSprt -networks is sufficient. The ffelements are in fSit-programming environments (or fself- fuzzy programming environments), but this question and ffSprt -networks base will be considered in the following monographs. In particular, ffelements may contain Sit- programming operators. In mnffSprt cores, the constant memory fSprtwith correspondent ffelements depending on mnffSprt.

The OS (operating system) and the principles and modes of operation of the fSit-networks for this programming are interesting. But this is already the material for the next publications.

Here is developed a helicopter model without a main and tail rotors based on ffSprt– physics and special neural networks with artificial neurons operating in normal and ffSprt -modes. Let's denote this model through ffSmnsprt. To do this, it's proposed to use mnffSprt of different levels; for example, for the usual mode, mnffSprt serves for the initial processing of signals and the transfer of information to the second level, etc., to the nodal center, then checked. In case of an anomaly - local ffSprt –mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, ffSmnsprt is activated with the desired "target weight." Here are realized other tasks also. To reach the fself-

energy level, the mode μ is used. In normal mode, it's planned to

carry out the movement of ffSmnsprt on jet propulsion by converting the energy of the emitted gases into a vortex to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the ffSmnsprt for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is drainage of exhaust gases outside the ffSmnsprt. ffSmnsprt is represented by a neural network that extends from the center of one of the main clusters of ffSprt- artificial neurons to the shell, turning into the body itself. Above the operator's cabin is the central core of the neural network and the target block, responsible for performing the "target weights" and

auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for ffSmnsprt's actions is below the operator's cab. In ffSprt– mode, the entire network or its sections are fSprt– activated to perform specific tasks, in particular, with "target weights." In the target, block used ffSprt -coding, ffSprt -translation for activation of all networks to "target weights" simultaneously, then –the reset of this Sit-coding after activation. Unfortunately, triodes are not suitable for Sit -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for ffeoprograms may be used instead triodes since there is no necessity to unbend the alternating current to direct. The ffSprt-operative memory belt is disposed around a central core of ffSmnsprt. There are ffSprt -coding, ffSprt -translation, and ffSprt-realize of feprograms and the programs from the archives without extraction, ffSprt-coding and ffSprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. ffSprt– structure or an feprogram if one is present of needed «target weight» are taken in target block at ffSprt– activation of the networks. $\text{ffSprt}_{\mu}^{\text{ffSmnsprt}, g}$ derives ffSmnsprt to the fself-level boundary with *activation* target weight g .

It's used an alternating current of above high frequency and ultra-violet light, which can work with ffSprt– structures in ffSprt –modes by its nature to activate the networks or some of its parts in ffSprt –modes and locally using ffSprt –mode. Above high frequently alternating current go through mercury bearers. That's why overheating does not occur. The power of the alternating current above high frequently increases considerably for the target block. The activation of all networks is realized to indicate "target weights."

4.3 Introduction to FUZZY PROGRAM OPERATORS fDprt, ftprD, fD¹epr, fDeprt₁

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through fDprt-Networks - fuzzy analogue of Sit-Networks [10] in one of the central departments of which a conventional computer system is located. The parallel processor is itself fdeprogram - fuzzy analogue of eprogram [2, 10] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a fuzzy Dprt -program operator - fDprt $\overset{Ag}{action\ Q}$, where fuzzy A with measure of *activation*

fuzziness μ_A fuzzy acts Q with measure of fuzziness μ_Q to fuzzy *activation* with measure of fuzziness $\mu_{activation}$, Q is any fuzzy *action*, it will be possible to obtain the fuzzy execution with measure of fuzziness $\mu_{activation}$ of a parallel fuzzy action A with the desired target weight g or the execution with measure of fuzziness $\mu_{activation}$ of a parallel action A with the desired fuzzy target weight g with measure of fuzziness μ_g or both. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For fDprt-coding and fDprt-translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical pulses laser of Nobel laureates 2018 year Gerard Mourou, Donna Strickland, or a combination of them. For the desired action, for example, using the direct parallel fprogram of operator

$\{UHF\ AC := R\}$
fDprt $\overset{activation}{action\ Q}$ with the specified measures of fuzziness, we simultaneously

enter the desired fuzzy set of codes R with measure of fuzziness μ_R using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects. Consider the types of direct parallel fuzzy fdprogram operators:

- 1) fuzzy Dprt-program operators (designation fDprt-program operators)

- 2) fuzzy tprD-program operators (designation ftprD-program operators)
- 3) fuzzy D¹epr - program operators (designation fD¹epr -program operators)
- 4) fuzzy Deprt₁- program operators (designation fDeprt₁-program operators)

fDprt-algorithm Example:

Simultaneous multiplication Q with measure of fuzziness μ_Q :

$\tilde{x}$
fDprtmultiplication Q, the notation of the fuzzy set B with elements
 $\tilde{y}$

$$b_{i_1 i_2 \dots i_n j_1 j_2 \dots j_n} =$$

$$\left(\text{ffSprt}_{\mu} \left\{ x_{1_{i_1}} * \mu_{\tilde{x}}(x_{1_{i_1}}), x_{2_{i_2}} * \mu_{\tilde{x}}(x_{2_{i_2}}), \dots, x_{n_{i_n}} * \mu_{\tilde{x}}(x_{n_{i_n}}) \right\} \right)_R$$

$$\left(\text{fDprt}_{\mu} \left\{ y_{1_{j_1}} * \mu_{\tilde{y}}(y_{1_{j_1}}), y_{2_{j_2}} * \mu_{\tilde{y}}(y_{2_{j_2}}), \dots, y_{n_{j_m}} * \mu_{\tilde{y}}(y_{n_{j_m}}) \right\} \right)_G$$

q
multiplication Q
 h

for any $\{i_1, i_2, \dots, i_n\}, \{j_1, j_2, \dots, j_n\}$ without repetitions, $q = \text{ffSprt}_{\mu}^K$, K-set
 w

of any $\{k_1, k_2, \dots, k_n\}$ without repeating them, k_i -any digit, $i=1, 2, \dots, n$,

$R = \text{ffSprt}_{\mu}^{\{i_1 +, i_2 +, \dots, i_n\}}$, R is the index of the lower discharge, $h = \text{ffSprt}_{\mu}^L$,
 w

L-set of any $\{l_1, l_2, \dots, l_m\}$ without repeating them, l_i -any digit,

$i=1, 2, \dots, m$, $G = \text{ffSprt}_{\mu}^{\{j_1 +, j_2 +, \dots, j_m\}}$, G is the index of the lower discharge,
 w

$V = \text{ffSprt}_{\mu}^{\{i_1 +, i_2 +, \dots, i_n + j_1 +, j_2 +, \dots, j_m\}}$ (we choose an index on the scale
 w

of discharges):see Table I.1.

Then ffSprt_{μ}^{B+} gives the final result of simultaneous multiplication. Any

system of calculus can be chosen, in particular binary. Here, in fact, sets of digits in the corresponding digits, representing numbers, are multiplied together simultaneously. The simplest functional scheme of the assumed arithmetic-logical device for fDprt-multiplication:

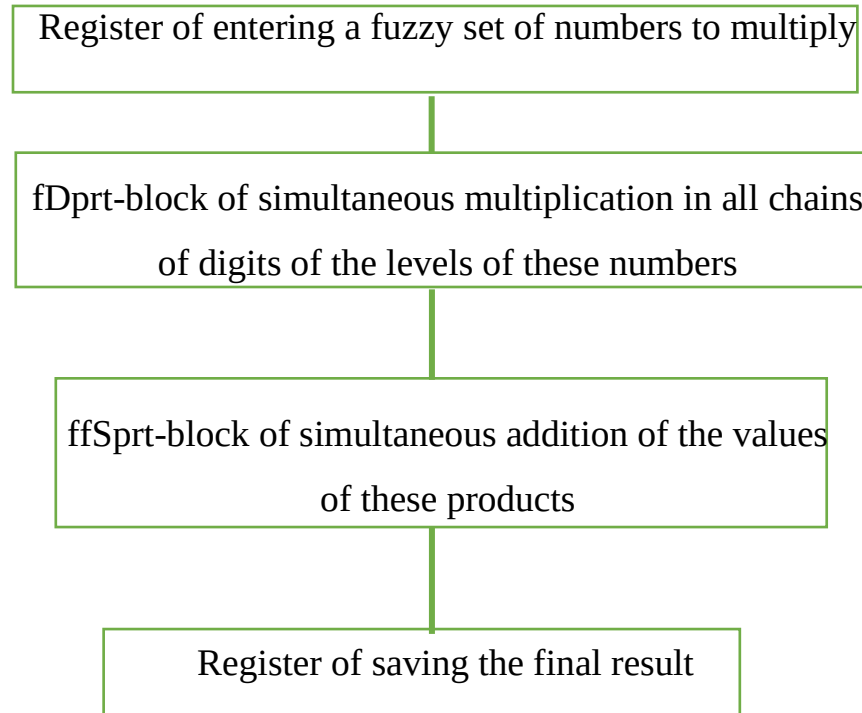


Fig. 4.3. The straightforward functional scheme of the assumed arithmetic-logical device for fDprt-multiplication.

Remark A.4.3.1. The algorithm for simultaneously fuzzy multiplication a fuzzy set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by fuzzy multiplying the first number from the fuzzy set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the fuzzy set by the ones following it, etc.

One example is pattern recognition: fDprt

	<i>image archive</i>	q
if ffSprt	μ	$\exists$ ffSprt μ
	B	B
	<i>give test result</i>	
	<i>Name of q</i>	

$$\text{fDprt} \frac{\{p\}}{\{a(x)\}} \text{fDprt} \frac{IF\{\{B\}\{f\}\}}{\text{give test result}} \text{fDprt} \frac{R}{\text{action } G} \frac{Q}{\text{action } H} \frac{x}{\tilde{y}}.$$

$\subset \tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$, we can consider self-type fDprt-structure - $fD_8f\tilde{y}; Q; \tilde{x}$ with m elements from $\tilde{x}$, $m < n$, which is formed according to the form (1.1), or in forms (1.1.1) - (1.1.5), generalizing it.

Remark 4.3.2. Fuzzy self, in particular, according to a fuzzy form is fuzzy analogue of the form of type (1.1) [1-3]: (2.1*).

4. Simultaneous fuzzy *action* Q of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $\text{fDprt } Q \left(\begin{smallmatrix} \tilde{p} \\ \{\tilde{x}\} \end{smallmatrix} \right)$.

5. Simultaneous $R =$ fuzzy checking with fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), ..., g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), ..., B_n|\mu_{\tilde{B}}(B_n))$. Implemented via $\frac{\tilde{B}}{fDprt R, \text{ where } \tilde{Q} \text{ can be anything.}}$

6. Similarly for fuzzy loop operators and others.

fD_8f – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding fdprt-program operators in form (1.1), for more complex operators in forms (1.1,1) - (1.4), (2.1*) and analogs of forms (1.1,1) - (1.4) by type (2.1*).

For example, $\text{fdprt}_{g\{R,S\}}^S$ is the fuzzy fdprt-self structure with measure of fuzziness μ of the second type if $g\{R,S\}$ is a fdprogram capable of fuzzy generating R with measure of fuzziness μ from S.

The example of self-fdprogram of the first type is

$$\text{fdprt}_{\{\tilde{x}\}}^{\tilde{p}} \left\{ \text{fdprt}_{\tilde{Q}}^{\tilde{B}}, \text{fdprt}_{\tilde{Q}}^{\tilde{B}}, \text{fdprt}_{\tilde{Q}}^{\tilde{B}} \right\}$$

action R
w

The example of fdprt-program for fDmnsprt- fuzzy analogue of SmnSt [1-3]:

$$\text{fdprt}_{\{\tilde{x}\}}^{\tilde{p}} \text{ } Q - \text{fuzzy } \textit{action} \text{ } Q \text{ of } \tilde{p} \text{ to } \tilde{x}.$$

fdprt_{gP}^{tw} , where P - fuzzy assigning target weight tw to fuzzy g with measure of fuzziness μ .

$\text{fdprt}_{fDmnsprt \textit{activation}}^{\{q\}w}$, where S - fDmnsprt *activation* for fuzzy $\{q\}w$ with measure of fuzziness μ .

fdprt-coding.

fdprt-coding with measure of fuzziness μ : 1) fuzzy set A to fuzzy set B, 2) fuzzy set A to a point q , where the elements of the fuzzy sets A, B can be continuous. For example, fdprt_{BQ}^A , where Q - fdprt-coding.

There are fdprt-coding, fdprt-translation, fdprt-realize of fdeprograms and fprograms from the archives without extraction theirs

fDelf-coding.

fDelf-coding with measure of fuzziness μ : 1) fuzzy set A to set fuzzy A , i.e. fuzzy A on itself 2) fuzzy set A to a point q in form (A.4.1), where the elements of the

fuzzy sets A, B can be continuous. For example, $fDprt \overset{A}{Q}, \underset{A}{}{}$

One of the central departments of the control system should be a computer system of the usual type of the desired level. In symbiosis with fDprt-Networks, it will provide a holistic operation of the control system in three modes: conventional serial through a conventional type computer system, direct parallel through fDprt-Networks and series-parallel. Codes from a conventional type computer system

will be used via fDprt -connectors in fDprt - coding, for example: $fDprt \overset{\{UHF AC\}}{:= Q} \underset{activation}{}{}$.

UHF AC field activation is used.

Dynamic fDprt and $fD_8(t)f$ programming.

The ideology of dynamic fDprt and $fD_8(t)f$ can be used for programming:

1. Simultaneous fuzzy *action* $\widetilde{Q(t)}$ of the

expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), ..., p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), ..., x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via

$$fDprt(t) \overset{\widetilde{p(t)}}{\widetilde{Q(t)}} \underset{\{\widetilde{x(t)}\}}{}{}$$

2. Simultaneous $\widetilde{R(t)}$ = fuzzy checking with fuzziness μ by the fuzzy set

of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), ..., g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions

$\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), ..., B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$.

Implemented via $fDprt(t) \overset{\widetilde{B(t)}}{\widetilde{R(t)}} \underset{\widetilde{Q(t)}}{}{}$, where $\widetilde{Q}$ can be anything.

3. Similarly for fuzzy loop operators and others.

$fD_8(t)f$ - fuzzy software operators will differ only just because aggregates $\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding fDprt-program operators in form (1.1), for more complex operators in forms (1.1,1) - (1.4), (2.1*) and analogs of forms (1.1,1) - (1.4) by type (2.1*).

ftprD-program operators.

The ideology of ftprD and $D_{16}f$ - fuzzy analogues of tS and t_{S_4f} from [14] can be used for programming. Here are some of the ftprD-program operators.

1. Simultaneous expelling fuzzy *action* Q of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1),$

$p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots,$

$\tilde{x}$

$x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $Q \text{ fDprt.}$

$\{\tilde{p}\}$

2. Simultaneous expelling $R =$ fuzzy checking with fuzziness μ by the fuzzy

set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of

expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$. It's implemented

through $\tilde{Q}$
 $\tilde{B}$ fDprt, where $\tilde{Q}$ can be anything.

3. Similarly for loop operators and others.

$fD_{16}f$ – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding ftprD program operators in form (1.1) for more complex operators in form (1.1), for more complex operators in forms (1.1,1) - (1.4), (2.1*) and analogs of forms (1.1,1) - (1.4) by type (2.1*).

Consider hierarchical ftprD-program operator

$$\left(\begin{matrix} B \\ (action\ Q)^{-1}Dprt \\ A \end{matrix} \right) = \left\{ D + \begin{matrix} \{\} \\ \mu \\ A - A \cap B \\ (B - A \cap B) \end{matrix} \text{ ffsprt} \right\}, \text{ where } D \text{ is osel- (fuzzy set) for}$$

fuzzy $(A \cap B)$, where action Q - contain.

Dynamic ftprD and $fD_{16}(t)f$ programming at time q .

The ideology of ftprD and fD₁₆f can be used for dynamic programming. Here are some of the ftprD- dynamic programming operators.

1. The process of simultaneous expelling fuzzy *action* $Q(t)$ of the

expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), ..., p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$

from the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), ...,$

$x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via $\frac{Q(t)}{\{p(t)\}} \text{ fDprt}(t)$.

2. The process of simultaneous expelling $R(t)$ = fuzzy checking with fuzziness $\mu(t)$

by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), ...,$

$g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)),$

$B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), ..., B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through $\frac{Q(t)}{B(t)} \text{ fDprt}(t)$, where $\frac{Q(t)}{B(t)}$

$\widetilde{Q(t)}$ can be anything.

3. Similarly for loop operators and others.

fD₁₆(t)f – fuzzy software operators will differ only just because aggregates

$\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding processes ftprD(t) for above mentioned programming operators through form (1.1), for more complex operators in forms (1.1,1) - (1.4), (2.1*) and analogs of forms (1.1,1) - (1.4) by type (2.1*).

Consider hierarchical dynamic ftprD-program operator:

$$\frac{B(q)}{(action\ Q)^{-1} \text{fDprt}(q)} = \left\{ \frac{fft(q)_{S_1 f(A(q) \cap B(q))} + \frac{\{\}}{\mu} A(q) - A(q) \cap B(q)}{(B(q) - A(q) \cap B(q))} \right\}^{\text{ffSprt}(q)}, \text{ where}$$

action Q- contain.

fD¹epr -program operators (form $\frac{B}{D} (action\ Q)^{-1} \text{fD}^1 \text{prt} \frac{A}{B} action\ Q$ - fuzzy analogue

of ${}_B^B S^1 t_B^A$ [10]))

For example, $\tilde{R}^{-1} \text{fD}^1 \text{prt} \tilde{R}$, where simultaneous expelling fuzzy checking with

fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$ and simultaneous $R =$ fuzzy checking with fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$, $\tilde{Q}$ can be anything.

The examples:

$(\text{action } Q)^{-1} \text{fD}^1 \text{prt} \text{action } Q$ can be interpreted as $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$ -fdprogram operator.
 $(\text{action } Q)^{-1} \text{fD}^1 \text{prt} \text{action } Q$ sample $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$ -fdprogram structure example.

Consider hierarchical dynamic $\text{fD}^1 \text{epr}$ -program operator: (form

$(\text{action } Q)^{-1} \text{fD}^1 \text{prt} \text{action } Q *$
 $(\text{action } Q)^{-1} \text{fD}^1 \text{prt} \text{action } Q$
 $D - A$

$\text{fD}^1 \text{prt}_1$ - program operators (form $(\text{action } Q)^{-1} \text{fD}_1 \text{prt} \text{action } Q$ - fuzzy analogue of ${}^C S_1 t_B^A$ [2], [12]))

$(\text{action } Q)^{-1} \text{fD}_1 \text{prt} \text{action } Q$ - sample $\begin{pmatrix} d_1self \\ d_1oself \end{pmatrix}$ -fdprogram structure example.

fdprogram structure example, where the assemblage point d_r is the cursor, it is quite complex self—fdprogram:

$$\text{fD}_1 \text{prt} \left\{ \begin{array}{l} q \left(\begin{array}{l} A \\ A \end{array} (\text{action } Q)^{-1} \text{fD}_1 \text{prt} \text{action } Q \right) \\ W_q \text{fD}_1 \text{prt} \\ q \left(\begin{array}{l} A \\ A \end{array} (\text{action } Q)^{-1} \text{fD}_1 \text{prt} \text{action } Q - \right) \\ Q \\ V \end{array} \right. , \text{fD}_1 \text{prt} \text{action } Q \left. \begin{array}{l} E^{\text{ex}} l^{d_r} \\ d_r(E_{in} l^{d_r}) \end{array} \right\}.$$

$$fD_{1prt} \left\{ \begin{array}{c} q \left(\begin{array}{c} A \\ (action\ Q)^{-1} fD_{1prt} action\ Q \\ A \end{array} \right) W_q^{E_q} fD_{1prt} \begin{array}{c} E^{ex} l^{d_r} \\ action\ Q \\ d_r(E_{in} l^{d_r}) \end{array} \end{array} \right\}$$

$\frac{Q}{V}$

can be interpreted as a fDpr-program operator.

Appendix

Remark. Energy of a living organism:

$$fd_{1g}(r, a(E_q)) =$$

$$fD_{1prt} \left\{ \begin{array}{c} q \left(\begin{array}{c} A \\ (action\ Q)^{-1} fD_{1prt} action\ Q \\ A \end{array} \right) W_q^{E_q} fD_{1prt} \begin{array}{c} E^{ex} l^{d_r} \\ action\ Q \\ d_r(E_{in} l^{d_r}) \end{array} \end{array} \right\}$$

$\frac{Q}{V}$

(**_{A.4}).

Energy with measure of fuzziness μ_1, μ_2 of a living organism of a person:

$$fd_{1pg}(r, a(E_q)) =$$

$$fD_{1prt}$$

$$\left\{ \begin{array}{c} q \left(\begin{array}{c} A \\ (action\ Q)^{-1} fD_{1prt} action\ Q \\ A \end{array} \right) W_q^{E_q} fD_{1prt} \begin{array}{c} E^{ex} l^{d_r} \\ action\ Q \\ d_r(self(E_{in} l^{d_r})) \end{array} \end{array} \right\}$$

$\frac{Q}{V}$

(***_{A.4}).

$\begin{array}{c} A \\ (action\ Q)^{-1} fD_{1prt} action\ Q \\ A \end{array}$ -internal energy of a living organism, q- a gap in the energy cocoon of a living organism, r-the position of the assemblage point d_r on the energy cocoon of a living organism, W_q - energy prominences from the gap in the cocoon of a living organism, E_q -external energy entering the gap in the cocoon of a living organism, $E^{ex} l^{d_r}$ - a bundle of fibers of external energy self-capacities (fDprt-self structures) from outside the cocoon, collected at the point of assembly

of the cocoon of a living organism, $E_{in}l^{d_r}$ - a bundle of fibers of external energy self-capacities (fDprt-self structures) from inside the cocoon, collected at the point of assembly of the cocoon of a living organism in the same position r of the assemblage point d_r . d_r is the subject of identifying the energy fibers of the subtle energy of the Universe in position r both outside and inside the cocoon.

(**_{A.4}), (***__{A.4}) can be interpreted as fDprt - program operators.

Entire neural network as instantaneous simultaneous ffRAM in ffSprt-elements and

fself- elements. $fself^{fself} \dots^{fself}$, $ff1 \downarrow I \uparrow_{-1}^1 ff2^{ff1 \downarrow I \uparrow_{-1}^1 ff2} \dots^{ff1 \downarrow I \uparrow_{-1}^1 ff2}$,

$f sin \infty^{f sin \infty} \dots^{f sin \infty}$. When activated in a neural network, the entire neural network becomes a working memory. Use of fself-energy as fuzzy activation or from

$$\begin{array}{c}
 \begin{array}{c}
 ffSprt \\
 \mu \\
 ffS_{mnSprt} \\
 activation \\
 Q
 \end{array} \\
 \text{outside. } fdQ_0 = fDprt \rightarrow \text{self-ffRAM,} \\
 \begin{array}{c}
 ffSprt \\
 \mu \\
 ffS_{mnSprt} \\
 activation
 \end{array}
 \end{array}$$

$$\begin{array}{c}
 \begin{array}{c}
 fD_{mnSprt} \\
 D \\
 fDprt \\
 activation \\
 Q
 \end{array} \\
 ffQ_{00} = \\
 \begin{array}{c}
 fD_{mnSprt} \\
 R \\
 fDprt \\
 activation
 \end{array}
 \end{array}$$

$$\begin{array}{c}
 \begin{array}{c}
 fD_{mnSprt} \\
 C \\
 fDprt \\
 activation \\
 fdQ_0
 \end{array} \\
 fdQ_{0prt}, ffQ_{01} = \\
 \begin{array}{c}
 fD_{mnSprt} \\
 H \\
 fDprt \\
 activation
 \end{array}
 \end{array}$$

$$fdQ_0, fdQ_{00}, fdQ_{01} -$$

coding, translation, realization in fdeprograms, use $fdQ_0, fdQ_{00}, fdQ_{01}$ -
 $fdS_{mnSprt}, fdAssembler$.

4.4 FUZZY PROGRAM OPERATORS Rprt, tprR, R¹epr, Reprt₁

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through Rprt-Networks in one of the central departments of which a conventional computer system is located. The parallel processor is itself rprogram with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a

Ag

Rprt -program operator - Rprt *action Q* , where fuzzy A with measure of *activation*

fuzziness μ_A fuzzy acts Q with measure of fuzziness μ_Q to fuzzy *activation* with measure of fuzziness $\mu_{activation}$, Q is any fuzzy *action*, it will be possible to obtain the fuzzy execution with measure of fuzziness $\mu_{activation}$ of a parallel fuzzy action A with the desired target weight g or the execution with measure of fuzziness $\mu_{activation}$ of a parallel action A with the desired fuzzy target weight g with measure of fuzziness μ_g or both. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For Rprt-coding and Rprt-translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical pulses laser of Nobel laureates of year 2018 Gerard Mourou, Donna Strickland, or a combination of them. For the desired action, for example, using the direct parallel fprogram

$\{UHF\ AC := R\}$

operator Rprt *action Q* with the specified measures of fuzziness, we *activation*

simultaneously enter the desired fuzzy set of codes R with measure of fuzziness μ_R using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects.

Consider the types of direct parallel fuzzy fprogram operators:

- 1) fuzzy Rprt-program operators
- 2) fuzzy tprR-program operators
- 3) fuzzy R^{1epr} - program operators
- 4) fuzzy Reprt₁- program operators

Rprt-algorithm Example:

Simultaneous multiplication Q with measure of fuzziness μ_Q :

$\tilde{x}$
fRprt multiplication Q, the notation of the fuzzy set B with elements
 $\tilde{y}$

$$b_{i_1 i_2 \dots i_n j_1 j_2 \dots j_n} =$$

$$\left(\text{ffSprt}_{\mu} \left\{ x_{1_{i_1}} * \mu_{\tilde{x}}(x_{1_{i_1}}), x_{2_{i_2}} * \mu_{\tilde{x}}(x_{2_{i_2}}), \dots, x_{n_{i_n}} * \mu_{\tilde{x}}(x_{n_{i_n}}) \right\} \right)_R$$

$$\left(\text{Rprt}_{\mu} \left\{ y_{1_{j_1}} * \mu_{\tilde{y}}(y_{1_{j_1}}), y_{2_{j_2}} * \mu_{\tilde{y}}(y_{2_{j_2}}), \dots, y_{n_{j_m}} * \mu_{\tilde{y}}(y_{n_{j_m}}) \right\} \right)_G$$

multiplication Q

for any $\{i_1, i_2, \dots, i_n\}, \{j_1, j_2, \dots, j_n\}$ without repetitions, $q = \text{ffSprt}_{\mu}^K$, K-set
 w

of any $\{k_1, k_2, \dots, k_n\}$ without repeating them, k_i -any digit, $i=1, 2, \dots, n$,

$R = \text{ffSprt}_{\mu}^L \{i_1 +, i_2 +, \dots, i_n\}$, R is the index of the lower discharge, $h = \text{ffSprt}_{\mu}^L$,
 w

L-set of any $\{l_1, l_2, \dots, l_m\}$ without repeating them, l_i -any digit,

$i=1, 2, \dots, m$, $G = \text{ffSprt}_{\mu}^L \{j_1 +, j_2 +, \dots, j_m\}$, G is the index of the lower discharge,
 w

$V = \text{ffSprt}_{\mu}^L \{i_1 +, i_2 +, \dots, i_n + j_1 +, j_2 +, \dots, j_m\}$ (we choose an index on the scale
 w

of discharges): see Table I.1.

Then ffSprt_{μ}^{B+} gives the final result of simultaneous multiplication. Any system

of calculus can be chosen, in particular binary. Here, in fact, sets of digits in the corresponding digits, representing numbers, are multiplied together simultaneously. The simplest functional scheme of the assumed arithmetic-logical device for Rprt-multiplication:

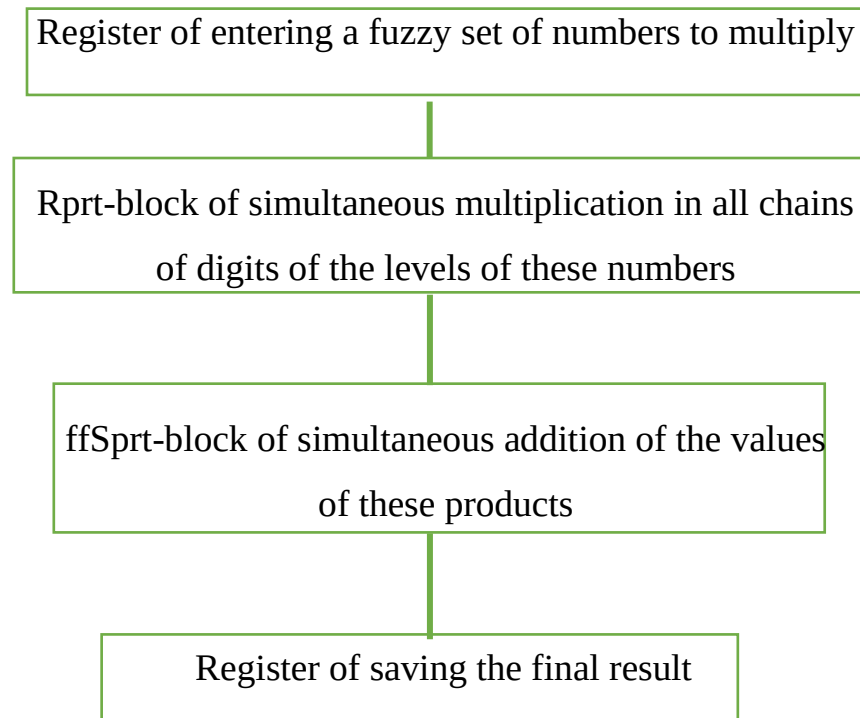


Fig. 4.4. The straightforward functional scheme of the assumed arithmetic-logical device for Rprt-multiplication.

Remark 4.4.1. The algorithm for simultaneously fuzzy multiplication a fuzzy set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by fuzzy multiplying the first number from the fuzzy set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the fuzzy set by the ones following it, etc.

One example is pattern recognition: Rprt

<i>image archive</i>	q
if ffSprt μ	$\exists$ ffSprt μ
B	B
<i>give test result</i>	
<i>Name of q</i>	

The example of Rprt-program is

$$\text{Rprt} \left\{ \begin{array}{l} \{p\} \\ \{a(x)\} \end{array} \right\} := \text{Rprt} \begin{array}{c} IF \{B\} \{f\} \\ \text{give test result} \\ Q \\ \text{action } H \\ x \end{array} , \text{Rprt} \begin{array}{c} R \\ \text{action } G \\ R \end{array} \text{ } .$$

For example, based on Rprt *action* Q , where $\tilde{y}=(y_1|\mu_{\tilde{x}}(y_1), y_2|\mu_{\tilde{x}}(y_2), \dots, y_m|\mu_{\tilde{x}}(y_m))$

$\subset \tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$, we can consider self-type Rprt-structure -

$fR_8f\tilde{y}; Q; \tilde{x}$ with m elements from $\tilde{x}$, $m < n$, which is formed according to the

form(1.1) , or in forms (1.1.1) - (1.1.5), generalizing it. Structures more complex

than $fR_8f\tilde{y}; Q; \tilde{x}$ can be introduced. For example, through the forms that

generalizes (1.1): (1.2), (1.3) - (1.4). In particular, in (1), A is fuzzy compressed

(fuzzy fits) in C in the fuzzy compression fuzzy structure B in C (i.e., in the fuzzy

structure $fR_{\text{Rprt}} \mu \begin{array}{c} B \\ C \end{array}$) and corresponding generalizations of (1.4) on (1.3.1) - (1.3.4), etc.

(1.3.1) - (1.3.4) schematically interpret the fuzzy formation of fuzzy fRprt-self

structure through a pseudo 3-connected form with a 2-connected form. The

ideology of fRprt and $fR_8f\tilde{y}; Q; \tilde{x}$ can be used for programming.

Remark B.1.2. Fuzzy self, in particular, according to a fuzzy form is fuzzy analogue of the form of type (1.1) [1]: (2.1*).

Here are some of the fuzzy Rprt-program operators.

1. Simultaneous fuzzy *action* Q of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1),$

$p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots,$

$x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $\text{Rprt} \begin{array}{c} \tilde{p} \\ Q \\ \{\tilde{x}\} \end{array}$.

2. Simultaneous R = fuzzy checking with fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$. Implemented via

$\text{Rprt} \begin{array}{c} \tilde{B} \\ R \\ \tilde{Q} \end{array}$, where $\tilde{Q}$ can be anything.

3. Similarly for fuzzy loop operators and others.

fR_8f – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding frprt-program operators in form (1.1), for more complex operators in forms (1.1,1) - (1.4), (2.1*) and analogs of forms (1.1,1) - (1.4) by type (2.1*).

For example, $\text{Rprt}_R^S g\{R, S\}$ is the fuzzy Rprt-self structure with measure of fuzziness μ of the second type if $g\{R, S\}$ is a frprogram capable of fuzzy generating R with measure of fuzziness μ from S.

The example of self-frprogram of the first type is

$$\text{Rprt} \left\{ \begin{array}{ccc} \tilde{p} & \tilde{B} & Q \\ \text{Rprt } Q, & \text{Rprt } R, & \text{Rprt } Q \\ \{\tilde{x}\} & \tilde{Q} & Q \end{array} \right\}.$$

action R
w

The example of Rprt-program operators for Rmnrprt- fuzzy analogue of SmnSt [10]:

- 1) $\text{Rprt}_{\{\tilde{x}\}}^{\tilde{p}} Q$ - fuzzy *action* Q of $\tilde{p}$ to $\tilde{x}$.
- 2) $\text{Rprt}_g^P P$, where P - fuzzy assigning target weight tw to fuzzy g with measure of fuzziness μ .
- 3) $\text{Rprt}_{\text{Rmnrprt activation}}^{\{q\}w} S$, where S - Rmnrprt *activation* for fuzzy $\{q\}w$ with measure of fuzziness μ .

Rprt-coding.

Rprt-coding with measure of fuzziness μ : 1) fuzzy set A to fuzzy set B, 2) fuzzy set A to a point q, where the elements of the fuzzy sets A, B can be continuous. For example, $\text{Rprt}_B^A Q$, where Q - Rprt-coding.

There are Rprt-coding, Rprt-translation, Rprt-realize of frprograms and fprograms from the archives without extraction theirs

Relf-coding.

Relf-coding with measure of fuzziness μ : 1) fuzzy set A to set fuzzy A, i.e., fuzzy A on itself 2) fuzzy set A to a point q in form (1), where the elements of the fuzzy

sets A, B can be continuous. For example, $\text{Rprt} \begin{matrix} A \\ A \end{matrix} Q$,

One of the central departments of the control system should be a computer system of the usual type of the desired level. In symbiosis with Rprt-Networks, it will provide a holistic operation of the control system in three modes: conventional serial through a conventional type computer system, direct parallel through Rprt-Networks and series-parallel. Codes from a conventional type computer system

will be used via Rprt -connectors in Rprt - coding, for example: $\text{Rprt} \begin{matrix} \{ \text{UHF AC} \} \\ \text{activation} \end{matrix} := Q$.

UHF AC field activation is used.

Dynamic Rprt and $fR_8(t)f$ programming.

The ideology of dynamic Rprt and $fR(t)f$ can be used for programming:

1. Simultaneous fuzzy *action* $\widetilde{Q(t)}$ of the

expressions $\widetilde{p(t)} = (p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \dots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)} = (x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via

$$\text{Rprt}(t) \begin{matrix} \widetilde{p(t)} \\ \widetilde{Q(t)} \\ \{\widetilde{x(t)}\} \end{matrix}.$$

2. Simultaneous $\widetilde{R(t)}$ = fuzzy checking with fuzziness μ by the fuzzy set

of conditions $\widetilde{g(t)} = (g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \dots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions

$\widetilde{B(t)} = (B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \dots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$.

Implemented via $\text{Rprt}(t) \begin{matrix} \widetilde{B(t)} \\ \widetilde{R(t)} \\ \widetilde{Q(t)} \end{matrix}$, where $\widetilde{Q}$ can be anything.

3. Similarly for fuzzy loop operators and others.

$fR_8(t)f$ - fuzzy software operators will differ only just because aggregates $\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding Rprt-program operators in form (1.1), for more complex operators in forms (1.1,1) - (1.4), (2.1*) and analogs of forms (1.1,1) - (1.4) by type (2.1*).

tprR-program operators.

The ideology of tprR and Rf - fuzzy analogues of tS and t_{S_4f} from [14] can be used for programming. Here are some of the tprR-program operators.

1. Simultaneous expelling fuzzy *action* Q of the

expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), ..., p_n|\mu_{\tilde{p}}(p_n))$ from the variables

$\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), ..., x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via $\tilde{Q}_{Rprt. \{\tilde{p}\}}$

2. Simultaneous expelling $R =$ fuzzy checking with fuzziness μ by the

fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), ..., g_n|\mu_{\tilde{g}}(g_n))$ for the

fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), ..., B_n|\mu_{\tilde{B}}(B_n))$. It's

implemented through $\tilde{Q}_{\tilde{B} Rprt}$, where $\tilde{Q}$ can be anything.

3. Similarly for loop operators and others.

$fR_{16}f$ – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding tprR program operators in form (1.1), for more complex operators in forms (1.1,1) - (1.4), (2.1*) and analogs of forms (1.1,1) - (1.4) by type (2.1*).

Consider hierarchical tprR-program operator

$$\left(\begin{matrix} B \\ (action\ Q)^{-1}Rprt \\ A \end{matrix} \right) = \left\{ D + \begin{matrix} \{\} \\ \mu \\ A - A \cap B \\ (B - A \cap B) \end{matrix} ffSprt \right\}, \text{ where } D \text{ is oself-(fuzzy set) for}$$

fuzzy $(A \cap B)$, where action Q - contain.

Dynamic tprR and $R_{16}(t)f$ programming at time q .

The ideology of tprR and $R_{16}f$ can be used for dynamic programming. Here are some of the tprR- dynamic programming operators.

1. The process of simultaneous expelling fuzzy *action* $Q(t)$ of the

expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), ..., p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ from the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), ..., x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented

$$\text{via } \frac{\widetilde{x(t)}}{Q(t) \text{ Rprt}(t), \{\widetilde{p(t)}\}}$$

2. The process of simultaneous expelling $R(t)$ = fuzzy checking with fuzziness $\mu(t)$

by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), ..., g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)),$

$B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), ..., B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through $\frac{\widetilde{Q(t)}}{R(t) \text{ Rprt}(t), \widetilde{B(t)}}$, where

$\widetilde{Q(t)}$ can be anything.

3. Similarly for loop operators and others.

$R_{16}(t)f$ – fuzzy software operators will differ only just because aggregates $\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding processes tprR(t) for above mentioned programming operators through form (1.1), for more complex operators in forms (1.1,1) - (1.4), (2.1*) and analogs of forms (1.1,1) - (1.4) by type (2.1*).

Consider hierarchical dynamic tprR-program operator:

$$\frac{B(q)}{(action \ Q)^{-1} \text{Rprt}(q)} = \left\{ \frac{fft(q)_{S_1 f(A(q) \cap B(q))} + \frac{\{\}}{\mu} A(q) - A(q) \cap B(q)}{(B(q) - A(q) \cap B(q))} \text{ffSprt}(q) \right\}, \text{ where}$$

action Q- contain.

$R_{16}^{ep}r$ -program operators (form $\frac{B}{D} (action \ Q)^{-1} R_{16}^{prt} \frac{A}{B} Q$ - fuzzy analogue of

$\frac{B}{D} S^1 t_B^A [10]))$

For example, $\tilde{x} \quad \tilde{B}$
 $\{ \tilde{p} \} \quad \tilde{Q}$
 $R^{-1}R^{1prt}R$, where simultaneous expelling fuzzy checking with

fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$ and simultaneous R = fuzzy checking with fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$, $\tilde{Q}$ can be anything.

The examples:

$\begin{matrix} A \\ (action\ Q)^{-1}R^{1prt}action\ Q \\ A \end{matrix}$ can be interpreted as $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$ - R^{1epr} -program operator. $\begin{matrix} A \\ (action\ Q)^{-1}R^{1prt}action\ Q \\ A \end{matrix}$ sample $\begin{pmatrix} s^1elf \\ os^1elf \end{pmatrix}$ -frprogram structure example.

Consider hierarchical dynamic R^{1epr} -program operator: (form

$\begin{matrix} B \\ (action\ Q)^{-1}R^{1prt}action\ Q * \\ A \end{matrix}$ $\begin{matrix} A \\ B \\ A \end{matrix}$).
 $\begin{matrix} A \\ (action\ Q)^{-1}R^{1prt}action\ Q \\ D - A \end{matrix}$ $\begin{matrix} B \\ A \\ B \end{matrix}$

$\begin{matrix} C \\ (action\ Q)^{-1}R^{1prt}action\ Q \\ D \end{matrix}$ $\begin{matrix} A \\ B \end{matrix}$ - fuzzy analogue of

$\begin{matrix} C \\ S_1 t_B^A \\ D \end{matrix}$ [12])

$\begin{matrix} A \\ (action\ Q)^{-1}R^{1prt}action\ Q \\ A \end{matrix}$ - sample $\begin{pmatrix} r_1self \\ r_1oself \end{pmatrix}$ -frprogram structure example.

frprogram structure example, where the assemblage point d_r is the cursor, it is quite complex self—frprogram:

$$\text{fR}_1\text{prt} \left\{ \begin{array}{c} q \left(\begin{array}{c} A \\ (action\ Q)^{-1} \text{fR}_1\text{prt} action\ Q \\ A \end{array} \right) \\ W_q \end{array} \begin{array}{c} E_q \\ fR_1prt \\ \end{array} \begin{array}{c} \{E^{ex}l^{d_r}\} \\ , \text{fR}_1\text{prt} action\ Q \\ d_r(E_{in}l^{d_r}) \end{array} \right\}.$$

$$\text{fR}_1\text{prt} \left\{ \begin{array}{c} q \left(\begin{array}{c} A \\ (action\ Q)^{-1} \text{fR}_1\text{prt} action\ Q \\ A \end{array} \right) \\ W_q \end{array} \begin{array}{c} E_q \\ fR_1prt \\ \end{array} \begin{array}{c} \{E^{ex}l^{d_r}\} \\ , \text{fR}_1\text{prt} action\ Q \\ d_r(E_{in}l^{d_r}) \end{array} \right\}$$

$$\begin{array}{c} Q \\ V \end{array}$$

can be interpreted as a fRpt program operator.

Supplement 4.4.1

Remark. Energy of a living organism:

$$\text{fr}_1\text{g}(\text{r}, a(E_q)) =$$

$$\text{fR}_1\text{prt} \left\{ \begin{array}{c} q \left(\begin{array}{c} A \\ (action\ Q)^{-1} \text{fR}_1\text{prt} action\ Q \\ A \end{array} \right) \\ W_q \end{array} \begin{array}{c} E_q \\ fD_1prt \\ \end{array} \begin{array}{c} \{E^{ex}l^{d_r}\} \\ , \text{fR}_1\text{prt} action\ Q \\ d_r(E_{in}l^{d_r}) \end{array} \right\}$$

$$\begin{array}{c} Q \\ V \end{array}$$

(**_{B.4})

Energy of a living organism of a person:

$$\text{fR}_1\text{prt} \left\{ \begin{array}{c} q \left(\begin{array}{c} A \\ (action\ Q)^{-1} \text{fR}_1\text{prt} action\ Q \\ A \end{array} \right) \\ W_q \end{array} \begin{array}{c} E_q \\ fR_1prt \\ \end{array} \begin{array}{c} \{E^{ex}l^{d_r}\} \\ , \text{fR}_1\text{prt} action\ Q \\ d_r(self(E_{in}l^{d_r})) \end{array} \right\}$$

$$\begin{array}{c} Q \\ V \end{array}$$

(***_{B.4})

$\begin{array}{c} A \\ (action\ Q)^{-1} \text{fR}_1\text{prt} action\ Q \\ A \end{array}$ -internal energy of a living organism, q- a gap in the energy cocoon of a living organism, r-the position of the assemblage point d_r on the energy cocoon of a living organism, W_q - energy prominences from the gap in

the cocoon of a living organism, E_q -external energy entering the gap in the cocoon of a living organism, $E^{ex}l^{d_r}$ - a bundle of fibers of external energy self-capacities from outside the cocoon, collected at the point of assembly of the cocoon of a living organism, $E_{in}l^{d_r}$ - a bundle of fibers of external energy self-capacities from inside the cocoon, collected at the point of assembly of the cocoon of a living organism in the same position r of the assemblage point d_r . d_r is the subject of identifying the energy fibers of the subtle energy of the Universe in position r both outside and inside the cocoon.

$(**_{B.4}), (***_B.4)$ can be interpreted as the program operators.

Entire neural network as instantaneous simultaneous ffRAM in ffSprt-elements and

fself- elements. $fself^{fself \dots fself}$, $ff1 \downarrow I \uparrow_{-1}^1 ff_2^{ff1 \downarrow I \uparrow_{-1}^1 ff_2 \dots ff1 \downarrow I \uparrow_{-1}^1 ff_2}$,

$f sin \infty^{f sin \infty \dots f sin \infty}$. When activated in a neural network, the entire neural network becomes a working memory. Use of fself-energy as fuzzy activation or from

$$\begin{array}{c}
 ffS_{mnSprt} \\
 \mu \\
 \text{activation} \\
 \text{outside. } fdQ_0 = Rprt \quad Q \quad \rightarrow \text{self-frRAM,} \\
 ffS_{mnSprt} \\
 \mu \\
 \text{activation} \\
 R_{mnSprt} \\
 D \quad Rprt \\
 \text{activation} \\
 frQ_{00} = Q \quad frQ_{0prt}, frQ_{01} = \begin{array}{c} R_{mnSprt} \\ C \quad Rprt \\ \text{activation} \\ R_{mnSprt} \\ H \quad Rprt \\ \text{activation} \end{array} frQ_0 \cdot frQ_0, frQ_{00}, frQ_{01} - \\
 R_{mnSprt} \\
 R \quad Rprt \\
 \text{activation}
 \end{array}$$

coding, translation, realization in freprograms, use $frQ_0, frQ_{00}, frQ_{01}$ -
 $frS_{mnSprt}, frAssembler$.

Supplement 4.4.2

Let us introduce the following notations:

$$A*B = Sprt_B^A, A^2 = Self A = Srt_A^A, A^{\frac{3}{2}} = Rrt \frac{A}{A} = self^{\frac{3}{2}}(A), A^3 = Self^2 A, \dots, A^{\frac{3n}{2}} = \frac{A^n}{A^n} \\ Rrt \frac{A^n}{A^n} = self^{\frac{3n}{2}}(A), A^{n+1} = Self^n A, self^{\min(n,m)}(A) \in Srt_{A^m}^{A^n} = self^{\frac{n}{m}}(A), \\ self^{\min(n,m,k)}(A) \in Rrt \frac{A^n}{A^k} = self^{\frac{n+m+k}{2k}}(A), \dots \text{ etc.}$$

There is no commutativity here: $A*B \neq B*A$. We can consider operator functions:

$$e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots, (A+B)^n = \sum_{k=0}^n \binom{n}{k} A^k B^{n-k}, (1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \dots, \text{ etc.}$$

You can consider a more “hard” option: $A*B = PSprt_B^A$, where $PSprt_B^A$ – operator, containing A in every element of B, $A^2 = PSelf A = PSrt_A^A$, $A^3 = PSelf^2 A$, ..., $A^{n+1} = PSelf^n A$, $PSelf^{\min(n,m)}(A) \in PSrt_{A^m}^{A^n} = PSelf^{\frac{n}{m}}(A)$, ...etc. There is no

commutativity here: $A*B \neq B*A$. We can consider operator functions: $e^A = 1 + \frac{A}{1!} + \frac{A^2}{2!} + \frac{A^3}{3!} + \dots, (A+B)^n = \sum_{k=0}^n \binom{n}{k} A^k B^{n-k}, (1+A)^n = 1 + \frac{Ax}{1!} + \frac{n(n-1)A^2}{2!} + \dots, \text{ etc.}$

Let's introduce $\sqrt{self}$ as the result of the decision of the equation $Srt_x^x = self$, that is $x = \sqrt{self}$, $\sqrt[3]{self}$ as the result of the decision of the equation $Rprt_x^x = self$, that is $x = \sqrt[3]{self}$, $\sqrt[n]{self^m}$ as the result of the decision of the equation $x^{\frac{n}{m}} = self$, $self^\alpha$ as the result of the decision of the equation $x^{\frac{1}{\alpha}} = self$, where α is any number, in particular, a negative number etc. The following equality is true:

$self^{-\alpha}(self^\alpha G) = self^\alpha(self^{-\alpha} G) = G$. In this way one can introduce self-level space.

4.5 Rprt-networks

A. Galushkin's comprehensive monograph [3] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new

mathematical process with containment operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators. Containment operators are more convenient for networks. Also, the main emphasis was placed on using processors operating using triodes, which are generally not used in Rprt-networks. Rprt networks (SmnRprt) are a Rprt structure that can be built for the required weights, the implementation of which will be carried out using a short-pulse laser to generate attosecond pulses of light. Rprt-OS (Rprt operating system) uses Rprt-coding and Rprt-translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b), where the number b is the code of the action, and the number a is the code of the object of this action. Rprt-coding (or self-coding) is implemented through a matrix consisting of 2 columns (in the continuous case, two intervals of numbers). Here, the source encoding is used for all matrix rows simultaneously. Rprt-translation is carried out by inversion. In this case, self-type coding and self-type translation by (B.1.1.6) or (B.1.1.11), (B.1.1.18) will be more stable. The set of the target weights

$\{f_x\}$

$f = (f_1, f_2, \dots, f_n)$ in Rprt $\underset{D}{action} Q$ are chosen for necessary tasks using a short-

pulse laser to generate attosecond pulses of light to accomplish them, $x = (x_1, x_2, \dots, x_n)$, Q there is a containment operation there is a containment operation. We will not touch on the issues of applications, or network optimization. They are described in detail by A. Galushkin [3]. We will touch on the difference of this only for hierarchical complex networks. The same simple executing programs are in the cores of simple artificial neurons of type Rprt (designation - mnRprt) for simple information processing. More complex executing programs are used for

$\underset{\{qy\}}{b} \quad \underset{b}{\{ax\}}$

mnRprt nodes. Rprt-threshold element $-action Q^{-1} Rprt(t) action Q$, Q there is

a containment operation there is a containment operation, b- artificial neurons of type Rprt (designation - mnRprt), $x = (x_1, x_2, \dots, x_n)$ are the values of the initial signals, $a = (a_1, a_2, \dots, a_n)$ are the weights of Rprt-synapses and the values of the

output signals The first level of mnRprt consists of simple mnRprt. The second level of mnRprt consists of $\text{Rprt } \frac{\{\text{mnRprt}\}}{\text{action } Q \text{ } D}$ – Rprt-node of mnRprt in range D, D-capacity for mnRprt node. The third level of mnRprt consists of

$\text{Rprt } \frac{\{\text{mnRprt}\} \{\text{Rprt } \frac{\text{action } Q}{D} \}}{\text{action } Q \text{ } D}$ - Rprt²- node of mnRprt in range D, thus D becomes

capacity of itself in itself as an element for mnRprt. For our networks, it is sufficient to use Rprt²- nodes of mnRprt, but self-level is higher in living organisms, particularly Rprtⁿ-, $n \geq 3$. The target structure or the corresponding program enters the target unit using a short-pulse laser to generate attosecond pulses of light. After that, all networks or parts of them are activated according to the indicative goal. It may appear that we are leaving the network ideology, but these networks are a complex hierarchy of different levels, like living organisms.

Remark 4.5.1. Traditional scientific approaches through classical mathematics make it possible to describe only at the usual energy level. Here we consider an approach that makes describing processes with finer energies possible. mnRprt contains

$\text{Rprt } \frac{\{\text{reprogram}\}}{\text{action } Q \text{ } \text{mnRprt}}$, reprogram –executing program in Rprt-OS. Rprt-OS (or

Self-type of Rprt OS) is based on Rprt-assembly language (or Self-type of Rprt assembly language), which is based on assembly language through Rprt-approach in turn, if the base of elements of Rprt-networks is sufficient. The reprograms are in Rprt-programming environments (or Self-type of Rprt programming environments), but this question and Rprt-networks base will be considered in the following articles. In particular, reprograms may contain Rprt- programming operators. In mnRprt cores, the constant memory Rprt with correspondent reprograms depending on mnRprt.

The OS (operating system) and the principles and modes of operation of the Rprt-networks for this programming are interesting. But this is already the material for the next publications.

Here is developed a helicopter model without a main and tail rotors based on Rprt – physics and special neural networks with artificial neurons operating in normal and Rprt-modes. Let's denote this model through SmnRprt. To do this, it's proposed to use mnRprt of different levels; for example, for the usual mode, mnRprt serves for the initial processing of signals and the transfer of information to the second level, etc., to the nodal center, then checked. In case of an anomaly - local Rprt-mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, SmnRprt is activated with the desired "target weight" using a short-pulse laser to generate attosecond pulses of light. Here are realized other tasks also. To reach the self-energy level, the mode

SmnRprt

Rprt *action Q*, is used. In normal mode, it's planned to carry out the movement SmnRprt

of SmnRprt on jet propulsion by converting the energy of the emitted gases into a vortex to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the SmnRprt for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is drainage of exhaust gases outside the SmnRprt. SmnRprt is represented by a neural network that extends from the center of one of the main clusters of Rprt - artificial neurons to the shell, turning into the body itself. Above the operator's cabin is the central core of the neural network and the target block, responsible for performing the "target weights" and auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for SmnRprt's actions is below the operator's cab. In Rprt – mode, the entire network or its sections are Rprt – activated to perform specific tasks, in particular, with "target weights" using a short-pulse laser to generate attosecond pulses of light. In the target, block used Rprt -coding, Rprt-

translation for activation of all networks to "target weights" simultaneously, then – the reset of this Rprt-coding after activation using a short-pulse laser to generate attosecond pulses of light.

Unfortunately, triodes are not suitable for Rprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for reprograms may be used instead triodes since there is no necessity to unbend the alternating current to direct. The Rprt-operative memory belt is disposed around a central core of SmnRprt. There are Rprt-coding, Rprt-translation, and Rprt-realize of reprograms and the programs from the archives without extraction, Rprt-coding and Rprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. Rprt – structure or an reprogram if one is present of needed «target weight» are taken in target

SmnRprt, f
 block at Rprt – activation of the networks. Rprt *action Q* derives SmnRprt to the
activation

self-level boundary with target weight f. Activation of the entire network is implemented to perform “target weights” using a short-pulse laser to generate attosecond pulses of light.

You can also try to use higher frequency alternating current and ultraviolet light, which can work with Rprt– structures in Rprt–modes by its nature to activate the networks or some of its parts in Rprt–modes and locally using Rprt–mode to perform local tasks. Above high frequently alternating current go through mercury bearers. That’s why overheating does not occur.

Remark 4.5.2. Hypothesis 1: equations for real processes in a non-trivial form can be used to fully or partially interpret the self-level of the process, replacing the equal signs with identification signs, and solutions to these equations as a manifestation of this level on the level of objectivity and ordinary energies. That is, equations for real processes serve as a definition of the self-level of the process, the definition of self-values (self-characteristics) of the process through the identification sign, i.e., they are defined (expressed) through themselves. In

particular, forms (1.1) - (1.4) can be used as forms of identification. Each such singularity creates its own field, the process, the object etc. Much more effective than science for working with these singularities will be special Dynamic programming, which we are currently working on to create. If we represent an amorphous body with a mathematical structure of self-object $Sprt_{A_0+E_s}^{A_0+E_s}$, where $Sprt_{A_0}^{A_0}$ - level of objectivity of an amorphous object, $(Sprt_{A_0+E_s}^{A_0} + Sprt_{A_0}^{A_0+E_s})$ - the energy of connections between the level of subtle energy $Sprt_{E_s}^{E_s}$ and the level of objectivity.

Thus, one can try to conventionally represent the mathematical model of the energy structure of an amorphous object as a hierarchical dynamic operator

$$\begin{array}{c} Sprt_{E_s}^{E_s} \\ (Sprt_{A_0+E_s}^{A_0} + Sprt_{A_0}^{A_0+E_s}) \\ Sprt_{A_0}^{A_0} \end{array} \quad (4.5.1)$$

Identification at the lower levels of a hierarchical dynamic structure of type (4.5.1) will lead to the upper level. Let us denote the upper level of A by $\overline{A}$, the upper level of P by $\overline{P}$. Then singularity $\overline{A} \rightarrow \overline{P}$ is the setting for the transformation of A into P. The field of the given structure tw is used for the activation of networks. The field can remain in effect until it is executed tw. Here all stages of the structure tw can be executed directly in parallel, in particular, an algorithm for solving the desired problem. We will call this field the operational activation field. This field will be created according to the structure tw. The pulse structure of a short-pulse laser for generating attosecond light pulses is close to $(a \uparrow I \downarrow^a) \uparrow I \downarrow (\downarrow_a \downarrow I \uparrow^a)$, i.e., type ${}_a^a St_a^a$, and upon activation it will be induction of same type self, which is necessary for the formation of a local assembly point d_r of external energy fibers El^{d_r} . Its locality (position of the assembly point r) will be determined by the structure of the magnetic induction of the short-pulse laser pulse for generating attosecond light generation through Targetblock SmnSprt [7]. Execution tw will be

achieved through setting the assemblage point in the desired position r_1 to engage the appropriate external energy: $St_{dr}^{dr} St_{r1}^{dr}$.

4.6 Introduction to FUZZY PROGRAM OPERATORS SDS, tSDS, fD¹epr, fDeprt₁.

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through SDS -Networks - fuzzy analogue of Sit-Networks [10] in one of the central departments of which a conventional computer system is located. The parallel processor is itself fsdeprogram - fuzzy analogue of eprogram [10] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a

Subject of Q
fuzzy SDS -program operator - SDS $\frac{Ag}{action\ Q}$, where fuzzy A with measure *activation*

of fuzziness μ_A fuzzy acts Q with measure of fuzziness μ_Q to fuzzy *activation* with measure of fuzziness $\mu_{activation}$, Q is any fuzzy *action*, it will be possible to obtain the fuzzy execution with measure of fuzziness $\mu_{activation}$ of a parallel fuzzy action A with the desired target weight g or the execution with measure of fuzziness $\mu_{activation}$ of a parallel action A with the desired fuzzy target weight g with measure of fuzziness μ_g or both. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For SDS coding and SDS -translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical pulses laser of Nobel laureates of 2018 year Gerard Mourou, Donna Strickland, or a combination of them. For the desired action, for example, using the direct parallel fsdprogram of

Subject of Q
operator SDS {UHF AC := R} $\frac{action\ Q}{activation}$ with the specified measures of fuzziness, we

simultaneously enter the desired fuzzy set of codes R with measure of fuzziness μ_R using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects. Consider the types of direct parallel fuzzy fsdprogram operators:

- 1) fuzzy SDS-program operators
- 2) fuzzy tSDS -program operators
- 3) fuzzy D¹epr - program operators (designation fD¹epr -program operators)
- 4) fuzzy Deprt₁- program operators (designation fDeprt₁-program operators)

SDS -algorithm Example:

Simultaneous multiplication Q with measure of fuzziness μ_Q :

Subject of Q

SDS $\tilde{x}$
multiplication Q , the notation of the fuzzy set B with elements $\tilde{y}$

$$b_{i_1 i_2 \dots i_n j_1 j_2 \dots j_n} =$$

Subject of Q

$$\begin{aligned} & \left(\text{ffSprt}_{\mu} \left\{ x_{1_{i_1}} * \mu_{\tilde{x}}(x_{1_{i_1}}), x_{2_{i_2}} * \mu_{\tilde{x}}(x_{2_{i_2}}), \dots, x_{n_{i_n}} * \mu_{\tilde{x}}(x_{n_{i_n}}) \right\} \right)_R \\ & \text{(SDS } \underset{q}{\text{multiplication } Q} \text{)}_V \\ & \left(\text{ffSprt}_{\mu} \left\{ y_{1_{j_1}} * \mu_{\tilde{y}}(y_{1_{j_1}}), y_{2_{j_2}} * \mu_{\tilde{y}}(y_{2_{j_2}}), \dots, y_{n_{j_m}} * \mu_{\tilde{y}}(y_{n_{j_m}}) \right\} \right)_G \end{aligned}$$

for any $\{i_1, i_2, \dots, i_n\}, \{j_1, j_2, \dots, j_n\}$ without repetitions, $q = \text{ffSprt}_{\mu}^K$, K-set w

of any $\{k_1, k_2, \dots, k_n\}$ without repeating them, k_i -any digit, $i=1, 2, \dots, n$,

$R = \text{ffSprt}_{\mu}^L \{i_1, i_2, \dots, i_n\}$, R is the index of the lower discharge, $h = \text{ffSprt}_{\mu}^L$, w

L-set of any $\{l_1, l_2, \dots, l_m\}$ without repeating them, l_i -any digit,

$i=1, 2, \dots, m$, $G = \text{ffSprt}_{\mu}^L \{j_1, j_2, \dots, j_m\}$, G is the index of the lower discharge, w

$V = \text{ffSprt}_{\mu}^{\{i_1+, i_2+, \dots, i_n+ j_1+, j_2+, \dots, j_m\}}$ (we choose an index on the scale of discharges): see Table I.1.

Then ffSprt_{μ}^{B+} gives the final result of simultaneous multiplication. Any system of calculus can be chosen, in particular binary. Here, in fact, sets of digits in the corresponding digits, representing numbers, are multiplied together simultaneously. The simplest functional scheme of the assumed arithmetic-logical device for SDS-multiplication:

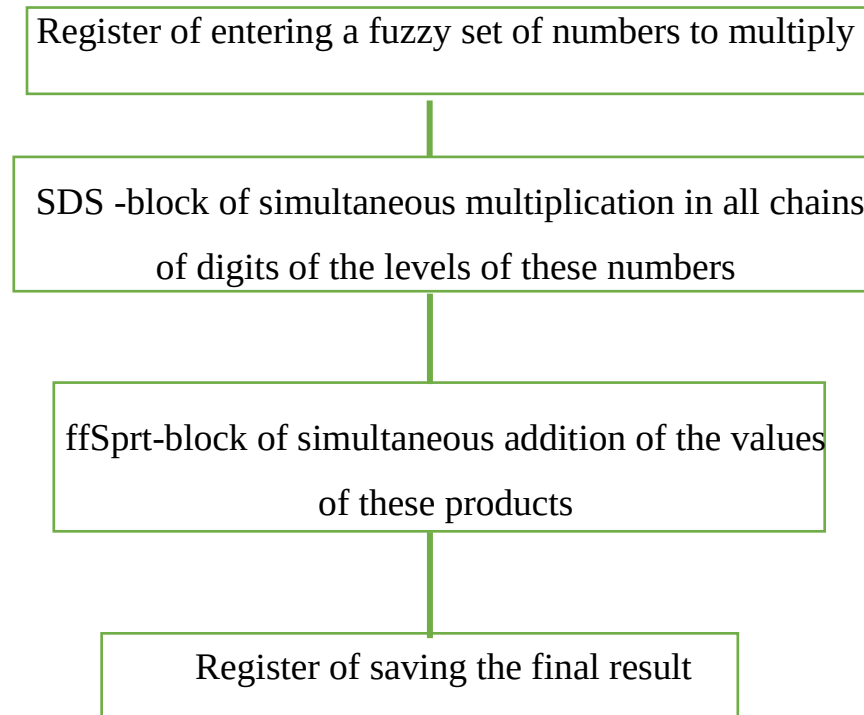


Fig. 4.5. The straightforward functional scheme of the assumed arithmetic-logical device for SDS -multiplication.

Remark 4.6.1. The algorithm for simultaneously fuzzy multiplication a fuzzy set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by fuzzy multiplying the first number from the fuzzy set by the rest: each multiplication is represented by

a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the fuzzy set by the ones following it, etc.

One example is pattern recognition: SDS

$$\begin{array}{c} \text{Subject of } Q \\ \text{image archive} \quad q \\ \text{if ffSprt} \quad \mu \quad \exists \text{ ffSprt} \mu \\ B \quad B \\ \text{action } Q = \text{give test result} \\ \text{Name of } q \end{array}$$

The example of SDS -program is

$$\text{SDS} \left\{ \begin{array}{c} \text{Subject of } Q \\ \{p\} \\ \text{fDprt} \quad \text{fDprt} \quad \text{fDprt} \\ \{a(x)\} \quad IF\{B\}\{f\} \quad \text{give test result} \quad \text{action } G \end{array} \right\} .$$

$$\begin{array}{c} R \\ H \\ R \\ \text{action } Q \\ B \end{array}$$

For example, based on SDS

$$\begin{array}{c} \text{Subject of } Q \\ \tilde{y} \\ \text{action } Q \\ \tilde{x} \end{array}, \text{ where } \tilde{y}=(y_1|\mu_{\tilde{x}}(y_1), y_2|\mu_{\tilde{x}}(y_2), \dots,$$

$y_m|\mu_{\tilde{x}}(y_m)) \subset \tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$, we can consider self-type SDS - structure - $fSD_8f\tilde{y}; Q; \tilde{x}$ with m elements from $\tilde{x}$, $m < n$, which is formed

according to the form (1.1), that is, the structure SDS

$$\begin{array}{c} \text{Subject of } Q \\ \tilde{y} \\ \text{action } Q \\ \tilde{x} \end{array}$$

contains

only m elements, or in forms (1.1.1) - (1.1.5), summarizing it. Fuzzy fcapacities in themselves of the third type can be formed for any other structure, not necessarily SDS, only by necessarily reducing the number of elements in the structure, in particular, using form (1.2). Structures more complex than fSD_8f can be introduced. For example, through a form (1.3), where A is fuzzy compressed (fuzzy fits) in C in the fuzzy compression fuzzy structure B in C (i.e., in the fuzzy

structure SDS

$$\begin{array}{c} \text{Subject of } Q \\ \tilde{y} \\ \text{action } Q \\ \tilde{x} \end{array} \quad); \text{ or through the forms (1.3.1) - (1.4) and$$

corresponding generalizations of (1.4) on (1.3.1) - (1.3.4), etc. (1.3.1) - (1.3.4) schematically interpret the fuzzy formation of fuzzy capacity in itself through a pseudo 3-connected form with a 2-connected form. The ideology of SDS and fSD_8f can be used for programming.

Remark D.4.6.1. Fuzzy self, in particular, according to a fuzzy form- fuzzy analogue of the form of type (1.1): (2.1*).

Here are some of the fuzzy SDS -program operators.

- a) Simultaneous fuzzy *action* Q of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots,$

Subject of Q

$x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via SDS $\begin{matrix} \tilde{p} \\ \text{action } Q \\ \tilde{x} \end{matrix}$.

- b) Simultaneous $R =$ fuzzy checking with fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$.

Subject of R

Implemented via SDS $\begin{matrix} \tilde{B} \\ \text{action } R \\ \tilde{Q} \end{matrix}$, where $\tilde{Q}$ can be anything.

- c) Similarly for fuzzy loop operators and others.

fSD_8f – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding fsdprrt-program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Subject of Q{R, S}

For example, SDS $\begin{matrix} S \\ \text{action } Q\{R, S\} \\ R \end{matrix}$ is the fuzzy self-type SDS-structure with

measure of fuzziness μ of the second type if $Q\{R, S\}$ is a fsdprogram capable of fuzzy generating R with measure of fuzziness μ from S .

The example of self-fsdprogram of the first type is

$$\begin{array}{c} \text{Subject of } R \\ \tilde{p} \quad \tilde{B} \quad Q \\ \text{SDS } \left\{ \begin{array}{ccc} \text{fDprt } Q & , & \text{fDprt } R \\ \{\tilde{x}\} & \tilde{Q} & Q \end{array} \right\} \\ \text{action } R \\ w \end{array}$$

The example of fsdprogram for SDmnsd- fuzzy analogue of SmnSprt [2]:

$$\begin{array}{c} \text{Subject of } Q \\ \text{SDS } \begin{array}{c} \tilde{p} \\ \text{action } Q \\ \tilde{x} \end{array} - \text{fuzzy action } Q \text{ of } \tilde{p} \text{ to } \tilde{x}. \end{array}$$

$$\begin{array}{c} \text{Subject of } P \\ \text{SDS } \begin{array}{c} tw \\ \text{action } P \\ g \end{array}, \text{ where } P - \text{fuzzy assigning target weight } tw \text{ to fuzzy } g \text{ with} \\ \text{measure of fuzziness } \mu. \end{array}$$

$$\begin{array}{c} \text{Subject of } S \\ \text{SDS } \begin{array}{c} \{q\}w \\ \text{action } S \end{array}, \text{ where } S - \text{SDmnsd activation for fuzzy } \{q\}w \text{ with} \\ \text{SDmnsd activation} \\ \text{measure of fuzziness } \mu. \end{array}$$

SDS -coding.

SDS -coding with measure of fuzziness μ : 1) fuzzy set A to fuzzy set B, 2) fuzzy set A to a point q, where the elements of the fuzzy sets A, B can be continuous. For

$$\begin{array}{c} \text{Subject of } Q \\ \text{example, SDS } \begin{array}{c} A \\ \text{action } Q \\ B \end{array}, \text{ where } Q - \text{SDS -coding.} \end{array}$$

There are SDS-coding, SDS-translation, SDS-realize of fsdprograms and fprograms from the archives without extraction theirs

SDelf-coding.

SDelf-coding with measure of fuzziness μ : 1) fuzzy set A to set fuzzy A, i.e., fuzzy A on itself 2) fuzzy set A to a point q in form (1), where the elements of the fuzzy

$$\text{sets A, B can be continuous. For example, SDS } \begin{array}{c} \text{Subject of } Q \\ A \\ \text{action } Q \\ A \end{array} .$$

One of the central departments of the control system should be a computer system of the usual type of the desired level. In symbiosis with SDS-Networks, it will provide a holistic operation of the control system in three modes: conventional serial through a conventional type computer system, direct parallel through SDS-Networks and series-parallel. Codes from a conventional type computer system

$$\text{will be used via SDS-connectors in SDS-coding, for example: SDS } \begin{array}{c} \text{Subject of } := \\ \{ \text{UHF AC} \} \\ := \\ \text{activation} \end{array} .$$

UHF AC field activation is used.

Dynamic SDS and $SD_8(t)f$ programming.

The ideology of dynamic SDS and $SD_8(t)f$ can be used for programming:

1. Simultaneous fuzzy *action* $\widetilde{Q(t)}$ of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \dots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via

$$\text{SDS } \begin{array}{c} \text{Subject of } \widetilde{Q(t)} \\ \widetilde{p(t)} \\ \text{action } \widetilde{Q(t)} \\ \{ \widetilde{x(t)} \} \end{array} .$$

2. Simultaneous $\widetilde{R(t)}$ = fuzzy checking with fuzziness μ by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \dots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions

$$\widetilde{B(t)} = (B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \dots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t))).$$

Subject of $\widetilde{R(t)}$

Implemented via SDS $\frac{\widetilde{B(t)}}{\text{action } \widetilde{R(t)}}, \text{ where } \widetilde{Q(t)} \text{ can be anything.}$
 $\widetilde{Q(t)}$

3. Similarly for fuzzy loop operators and others.

$SD_8(t)f$ - fuzzy software operators will differ only just because aggregates $\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding SDS -program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

tSDS -program operators.

The ideology of tSDS and $SD_{16}f$ - fuzzy analogues of tS and t_{s_4f} from [11] can be used for programming. Here are some of the tSDS-program operators.

1) Simultaneous expelling fuzzy *action* Q of the

expressions $\widetilde{p} = (p_1|\mu_{\widetilde{p}}(p_1), p_2|\mu_{\widetilde{p}}(p_2), \dots, p_n|\mu_{\widetilde{p}}(p_n))$ from the variables

$\widetilde{x} = (x_1|\mu_{\widetilde{x}}(x_1), x_2|\mu_{\widetilde{x}}(x_2), \dots, x_n|\mu_{\widetilde{x}}(x_n))$. This is implemented

$\widetilde{x}$
via $\frac{(\text{action } Q)^{-1}}{\{\widetilde{p}\}}$ SDS.
Subject of Q

2) Simultaneous expelling R = fuzzy checking with fuzziness μ by

the fuzzy set of conditions $\widetilde{g} = (g_1|\mu_{\widetilde{g}}(g_1), g_2|\mu_{\widetilde{g}}(g_2), \dots, g_n|\mu_{\widetilde{g}}(g_n))$ for

the fuzzy set of expressions $\widetilde{B} = (B_1|\mu_{\widetilde{B}}(B_1), B_2|\mu_{\widetilde{B}}(B_2), \dots,$

$\widetilde{Q}$
 $B_n|\mu_{\widetilde{B}}(B_n))$. It's implemented through $\frac{(\text{action } R)^{-1}}{\widetilde{B}}$ SDS, where $\widetilde{Q}$
Subject of R

can be anything.

3) Similarly for loop operators and others.

$SD_{16}f$ – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding tSDS- program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*). Consider hierarchical tSDS -program operator

$$\left(\begin{array}{c} B \\ (action\ Q)^{-1} \\ A \\ Subject\ of\ Q \end{array} \right)_{SDS} = \left\{ D + \begin{array}{c} \{\} \\ \mu \\ A - A \cap B \\ (B - A \cap B) \end{array} \right\}, \text{ where D is oself-(fuzzy set) for fuzzy}$$

$(A \cap B)$, where action Q- contain.

Dynamic tSDS and $SD_{16}(t)f$ programming at time q.

The ideology of tSDS and $SD_{16}f$ can be used for dynamic programming. Here are some of the tSDS - dynamic programming operators.

1. The process of simultaneous expelling fuzzy *action* $Q(t)$ of the

expressions $\widetilde{p(t)} = (p_1(t) | \mu_{\widetilde{p(t)}}(p_1(t)), p_2(t) | \mu_{\widetilde{p(t)}}(p_2(t)), \dots,$

$p_n(t) | \mu_{\widetilde{p(t)}}(p_n(t)))$ from the variables $\widetilde{x(t)} = (x_1(t) | \mu_{\widetilde{x(t)}}(x_1(t)),$

$x_2(t) | \mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t) | \mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via

$$\left(\begin{array}{c} \widetilde{x(t)} \\ (action\ Q(t))^{-1} \\ \widetilde{p(t)} \\ Subject\ of\ Q(t) \end{array} \right)_{SDS(t)}.$$

2. The process of simultaneous expelling $R(t)$ = fuzzy checking with fuzziness $\mu(t)$

by the fuzzy set of conditions $\widetilde{g(t)} = (g_1(t) | \mu_{\widetilde{g(t)}}(g_1(t)), g_2(t) | \mu_{\widetilde{g(t)}}(g_2(t)), \dots,$

$g_n(t) | \mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)} = (B_1(t) | \mu_{\widetilde{B(t)}}(B_1(t)),$

$B_2(t) | \mu_{\widetilde{B(t)}}(B_2(t)), \dots, B_n(t) | \mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through

$$\left(\begin{array}{c} \widetilde{Q(t)} \\ (action\ R(t))^{-1} \\ \widetilde{B(t)} \\ Subject\ of\ R(t) \end{array} \right)_{SDS(t)}, \text{ where } \widetilde{Q(t)} \text{ can be anything.}$$

3. Similarly for loop operators and others.

$$\left\{ \begin{array}{l} \text{fft}(q)_{s_1 f(A(q) \cap B(q))} + \frac{\{\}}{\mu} \text{ffSprt}(q) \\ (B(q) - A(q) \cap B(q)) \end{array} \right\}, \text{ where action Q- contain.}$$

For example, $R^{-1} \text{fD}^1 \text{prt} R$, where simultaneous expelling fuzzy checking with

The examples:

$(\overset{A}{action} \overset{A}{Q})^{-1} \text{fd}_{\text{prt}} \overset{A}{action} \overset{A}{Q}$ can be interpreted as $\left(\overset{s^1elf}{os^1elf} \right)$ -fdprogram operator.
 $(\overset{A}{action} \overset{A}{Q})^{-1} \text{fd}_{\text{prt}} \overset{A}{action} \overset{A}{Q}$ sample $\left(\overset{s^1elf}{os^1elf} \right)$ -fdprogram structure example.

Consider hierarchical dynamic fD^1 epr-program operator: (form

$$\begin{array}{c} B \\ (action\ Q)^{-1} fD^1_{prt} \end{array} \begin{array}{c} A \\ action\ Q \end{array} * \begin{array}{c} A \\ B \\ (action\ Q)^{-1} fD^1_{prt} \end{array} \begin{array}{c} B \\ A \\ action\ Q \end{array} \quad).$$

fD_{prt_1} - program operators (form $\begin{array}{c} C \\ (action\ Q)^{-1} fD_{prt} \end{array} \begin{array}{c} A \\ action\ Q \end{array} \begin{array}{c} D \\ B \end{array}$ - fuzzy analogue

of ${}^C S_1 t_B^A$ [12]).

$$\begin{array}{c} A \\ (action\ Q)^{-1} fD_{prt} \end{array} \begin{array}{c} A \\ action\ Q \end{array} - \text{sample} \left(\begin{array}{c} d_{1self} \\ d_{1oself} \end{array} \right) - \text{fdprogram structure}$$

example.

Appendix

Entire neural network as instantaneous simultaneous SDS-RAM in SDS -elements

and fself- elements. $fself^{fself} \dots^{fself}$, $ff1 \downarrow I \uparrow_{-1}^1, ff_2^{ff1 \downarrow I \uparrow_{-1}^1 f f_2, \dots^{ff1 \downarrow I \uparrow_{-1}^1 f f_2}$,

$f sin \infty^{f sin \infty} \dots^{f sin \infty}$. When activated in a neural network, the entire neural network becomes a working memory. Use of fself-energy as fuzzy activation or from outside.

4.7 Introduction to FUZZY PROGRAM OPERATORS SRS, tSRS, fR¹epr, fReprt₁

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through SRS -Networks - fuzzy analogue of Sit-Networks [10] in one of the central departments of which a conventional computer system is located. The parallel processor is itself fsreprogram - fuzzy analogue of eprogram [10] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a

Subject of Q

fuzzy SRS -program operator - SRS $\begin{array}{c} Ag \\ action\ Q \\ activation \end{array}$, where fuzzy A with measure

of fuzziness μ_A fuzzy acts Q with measure of fuzziness μ_Q to fuzzy *activation* with measure of fuzziness $\mu_{activation}$, Q is any fuzzy *action*, it will be possible to obtain the fuzzy execution with measure of fuzziness $\mu_{activation}$ of a parallel fuzzy action A with the desired target weight g or the execution with measure of fuzziness $\mu_{activation}$ of a parallel action A with the desired fuzzy target weight g with measure of fuzziness μ_g or both. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For SRS coding and SRS -translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical pulses laser of Nobel laureates of 2018 year Gerard Mourou, Donna Strickland, or a combination of them. For the desired action, for example, using the direct parallel fsrprogram of

Subject of Q
operator SRS $\{\text{UHF AC} := R\}$ with the specified measures of fuzziness, we
action Q
activation

simultaneously enter the desired fuzzy set of codes R with measure of fuzziness μ_R using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects. Consider the types of direct parallel fuzzy fsrprogram operators:

- 4) fuzzy SRS-program operators
- 5) fuzzy tSRS -program operators
- 6) fuzzy R^{1epr} - program operators (designation fR^{1epr} -program operators)
- 7) fuzzy $Reprt_1$ - program operators (designation $fReprt_1$ -program operators)

SRS -algorithm Example:

Simultaneous multiplication Q with measure of fuzziness μ_Q :

Subject of Q

SRS multiplication Q , the notation of the fuzzy set B with elements $\tilde{y}$

$$b_{i_1 i_2 \dots i_n j_1 j_2 \dots j_n} =$$

Subject of Q

$$\begin{aligned} & \left\{ x_{1_{i_1}} * \mu_{\tilde{x}}(x_{1_{i_1}}), x_{2_{i_2}} * \mu_{\tilde{x}}(x_{2_{i_2}}), \dots, x_{n_{i_n}} * \mu_{\tilde{x}}(x_{n_{i_n}}) \right\}_R \\ & \text{(SRS multiplication } Q \text{)}_V \\ & \left\{ y_{1_{j_1}} * \mu_{\tilde{y}}(y_{1_{j_1}}), y_{2_{j_2}} * \mu_{\tilde{y}}(y_{2_{j_2}}), \dots, y_{n_{j_m}} * \mu_{\tilde{y}}(y_{n_{j_m}}) \right\}_G \end{aligned}$$

for any $\{i_1, i_2, \dots, i_n\}, \{j_1, j_2, \dots, j_n\}$ without repetitions, $q = \text{ffSprt}_{\mu, K}^w$, K-set

of any $\{k_1, k_2, \dots, k_n\}$ without repeating them, k_i -any digit, $i=1, 2, \dots, n$,

$$R = \text{ffSprt}_{\mu, w}^L \{i_1 +, i_2 +, \dots, i_n\}, R \text{ is the index of the lower discharge, } h = \text{ffSprt}_{\mu, w}^L$$

L-set of any $\{l_1, l_2, \dots, l_m\}$ without repeating them, l_i -any digit,

$$i=1, 2, \dots, m, G = \text{ffSprt}_{\mu, w}^L \{j_1 +, j_2 +, \dots, j_m\}, G \text{ is the index of the lower discharge,}$$

$$V = \text{ffSprt}_{\mu, w}^L \{i_1 +, i_2 +, \dots, i_n + j_1 +, j_2 +, \dots, j_m\} \text{ (we choose an index on the scale}$$

of discharges): see Table I.1.

Then $\text{ffSprt}_{\mu, w}^{B+}$ gives the final result of simultaneous multiplication. Any

system of calculus can be chosen, in particular binary. Here, in fact, sets of digits in the corresponding digits, representing numbers, are multiplied together simultaneously. The simplest functional scheme of the assumed arithmetic-logical device for SRS-multiplication:

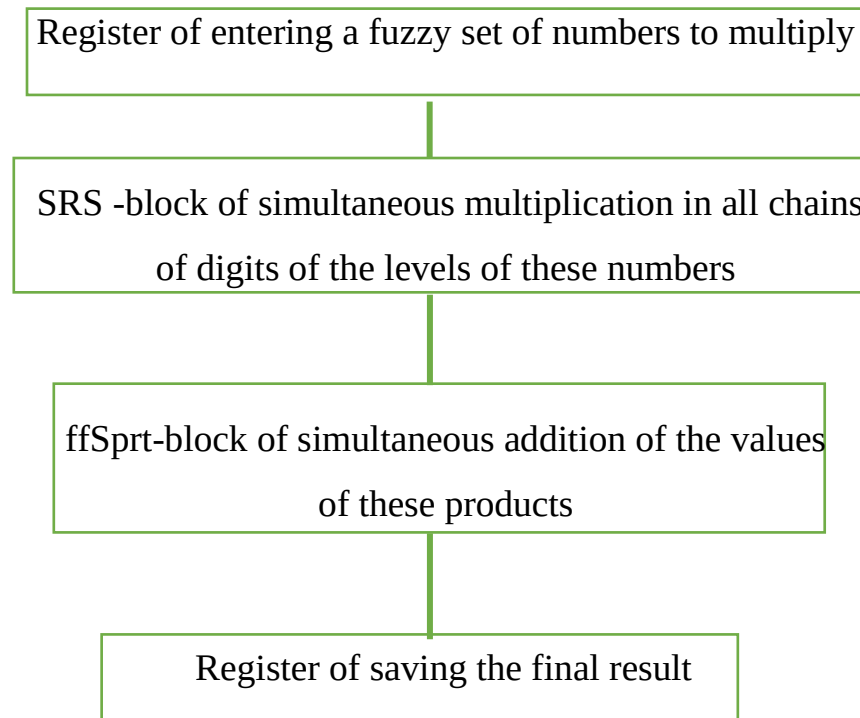


Fig. 4.6. The straightforward functional scheme of the assumed arithmetic-logical device for SRS -multiplication.

Remark.4.7.1. The algorithm for simultaneously fuzzy multiplication a fuzzy set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by fuzzy multiplying the first number from the fuzzy set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the fuzzy set by the ones following it, etc.

One example is pattern recognition: SRS

<i>Subject of Q</i>	
<i>image archive</i>	<i>q</i>
μ	$\exists \text{ ffSprt} \mu$
<i>B</i>	<i>B</i>
<i>action Q = give test result</i>	
<i>Name of q</i>	

The example of SRS -program is

$$\text{SRS} \left\{ \begin{array}{c} \text{Subject of } Q \\ \{\{p\}\} \\ \text{fRprt} \end{array} := \begin{array}{c} IF\{\{B\}\{f\}\} \\ \text{give test result} \\ \text{action } Q \\ B \end{array} , \begin{array}{c} \text{fRprt} \\ \text{action } G \end{array} \right\}.$$

For example, based on $\text{SRS} \begin{array}{c} \tilde{y} \\ \text{action } Q \\ \tilde{x} \end{array}$, where $\tilde{y}=(y_1|\mu_{\tilde{x}}(y_1), y_2|\mu_{\tilde{x}}(y_2), \dots, y_m|\mu_{\tilde{x}}(y_m)) \subset \tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$, we can consider self-type SDS - structure - $fSR_8f\tilde{y}; Q; \tilde{x}$ with m elements from $\tilde{x}$, $m < n$, which is formed

according to the form (1.1), that is, the structure $\text{SRS} \begin{array}{c} \text{Subject of } Q \\ \tilde{y} \\ \text{action } Q \\ \tilde{x} \end{array}$ contains only m elements, or in forms (1.1.1) - (1.1.5), summarizing it. Fuzzy fcapacities in themselves of the third type can be formed for any other structure, not necessarily SRS, only by necessarily reducing the number of elements in the structure, in particular, using form (1.2). Structures more complex than fSR_8f can be introduced. For example, through a form (1.3), where A is fuzzy compressed (fuzzy fits) in C in the fuzzy compression fuzzy structure B in C (i.e., in the fuzzy structure $\text{SRS} \begin{array}{c} \text{Subject of } Q \\ \tilde{y} \\ \text{action } Q \\ \tilde{x} \end{array}$); or through the forms (1.3.1) - (1.4) and corresponding generalizations of (1.4) on (1.3.1) - (1.3.4), etc. (1.3.1) - (1.3.4) schematically interpret the fuzzy formation of fuzzy capacity in itself through a pseudo 3-connected form with a 2-connected form. The ideology of SRS and fSR_8f can be used for programming.

Remark 4.7.1. Fuzzy self, in particular, according to a fuzzy form- fuzzy analogue of the form of type (1.1): (2.1*).

Here are some of the fuzzy SRS -program operators.

2. Simultaneous fuzzy *action* Q of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots,$

$$\begin{array}{c} \text{Subject of } Q \\ \tilde{p} \\ \text{action } Q \\ \tilde{x} \end{array} \cdot x_n|\mu_{\tilde{x}}(x_n)). \text{ This is implemented via SRS}$$

3. Simultaneous R = fuzzy checking with fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$. Implemented via

$$\begin{array}{c} \text{Subject of } R \\ \tilde{B} \\ \text{action } R \\ \tilde{Q} \end{array} \text{SRS}, \text{ where } \tilde{Q} \text{ can be anything.}$$

4. Similarly for fuzzy loop operators and others.

fSR_8f – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding fsrprt-program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

For example, $\begin{array}{c} \text{Subject of } Q\{R, S\} \\ S \\ \text{action } Q\{R, S\} \\ R \end{array}$ SRS is the fuzzy self-type SRS-structure with

measure of fuzziness μ of the second type if $Q\{R, S\}$ is a fsrprogram capable of fuzzy generating R with measure of fuzziness μ from S .

The example of self-fsrprogram of the first type is

$$\begin{array}{c} \text{Subject of } R \\ \tilde{p} \quad \tilde{B} \quad Q \\ \text{action } R \\ w \end{array} \text{SRS} \{ \text{fRprt } Q, \text{fRprt } R, \text{fRprt } Q \} \\ \{ \tilde{x} \} \quad \tilde{Q} \quad Q$$

The example of fsrprogram for SRmnsr- fuzzy analogue of SmnSt [10]:

Subject of Q
 SRS $\begin{matrix} \tilde{p} \\ \text{action } Q \\ \tilde{x} \end{matrix}$ - fuzzy action Q of $\tilde{p}$ to $\tilde{x}$.

Subject of P
 SRS $\begin{matrix} tw \\ \text{action } P \\ g \end{matrix}$, where P - fuzzy assigning target weight tw to fuzzy g with
 measure of fuzziness μ .

Subject of S
 SRS $\begin{matrix} \{q\}w \\ \text{action } S \end{matrix}$, where S - SRmnsr activation for fuzzy $\{q\}w$ with
 SRmnsr activation
 measure of fuzziness μ .

SRS -coding.

SRS -coding with measure of fuzziness μ : 1) fuzzy set A to fuzzy set B, 2) fuzzy set A to a point q , where the elements of the fuzzy sets A, B can be continuous. For

Subject of Q
 example, SRS $\begin{matrix} A \\ \text{action } Q \\ B \end{matrix}$, where Q - SRS -coding.

There are SRS-coding, SRS-translation, SRS-realize of fsrprograms and fprograms from the archives without extraction theirs

SRelf-coding.

SRelf-coding with measure of fuzziness μ : 1) fuzzy set A to set fuzzy A, i.e., fuzzy A on itself 2) fuzzy set A to a point q in form (1), where the elements of the fuzzy

Subject of Q
 sets A, B can be continuous. For example, SRS $\begin{matrix} A \\ \text{action } Q \\ A \end{matrix}$.

One of the central departments of the control system should be a computer system of the usual type of the desired level. In symbiosis with SRS-Networks, it will provide a holistic operation of the control system in three modes: conventional serial through a conventional type computer system, direct parallel through SRS-Networks and series-parallel. Codes from a conventional type computer system

will be used via SRS-connectors in SrS-coding, for example: $\text{SRS } \begin{matrix} \text{Subject of} \\ \{ \text{UHF AC} \} \\ \text{activation} \end{matrix} :=$.

UHF AC field activation is used.

Dynamic SRS and $SR_8(t)f$ programming.

The ideology of dynamic SRS and $SR_8(t)f$ can be used for programming:

1. Simultaneous fuzzy *action* $\widetilde{Q(t)}$ of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \dots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via $\text{SRS } \begin{matrix} \text{Subject of } \widetilde{Q(t)} \\ \widetilde{p(t)} \\ \text{action } \widetilde{Q(t)} \\ \{ \widetilde{x(t)} \} \end{matrix} .$
2. Simultaneous $\widetilde{R(t)}$ = fuzzy checking with fuzziness μ by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \dots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \dots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$.
Implemented via $\text{SRS } \begin{matrix} \text{Subject of } \widetilde{R(t)} \\ \widetilde{B(t)} \\ \text{action } \widetilde{R(t)} \\ \widetilde{Q(t)} \end{matrix} , \text{ where } \widetilde{Q(t)} \text{ can be anything.}$
3. Similarly for fuzzy loop operators and others.

$SD_8(t)f$ - fuzzy software operators will differ only just because aggregates $\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding SRS -program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

tSRS -program operators.

The ideology of tSRS and $SR_{16}f$ - fuzzy analogues of tS and t_{s_4f} from [11] can be used for programming. Here are some of the tSRS-program operators.

1. Simultaneous expelling fuzzy *action* Q of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$. This is implemented

$$\text{via } \begin{matrix} \tilde{x} \\ (action\ Q)^{-1} \\ \{\tilde{p}\} \\ Subject\ of\ Q \end{matrix} \text{ SRS.}$$

2. Simultaneous expelling $R =$ fuzzy checking with fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$. It's

$$\text{implemented through } \begin{matrix} \tilde{Q} \\ (action\ R)^{-1} \\ \tilde{B} \\ Subject\ of\ R \end{matrix} \text{ SRS, where } \tilde{Q} \text{ can be anything.}$$

3. Similarly for loop operators and others.

$SR_{16}f$ – fuzzy software operators will differ only just because aggregates $\tilde{x}, \tilde{p}, \tilde{B}, \tilde{g}$ will be formed from corresponding tSRS- program operators in form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical tSRS -program operator

$$\begin{matrix} B \\ (action\ Q)^{-1} \\ A \\ Subject\ of\ Q \end{matrix} \text{ SRS} = \left\{ D + \begin{matrix} \{\} \\ \mu \\ A - A \cap B \\ (B - A \cap B) \end{matrix} \text{ffSprt} \right\}, \text{ where } D \text{ is oself-(fuzzy set) for fuzzy}$$

$(A \cap B)$, where action Q - contain.

Dynamic tSRS and $SR_{16}(t)f$ programming at time q .

The ideology of tSRS and $Sr_{16}f$ can be used for dynamic programming. Here are some of the tSrS - dynamic programming operators.

8) The process of simultaneous expelling fuzzy *action* $Q(t)$ of the

expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), ...,$

$p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ from the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)),$

$x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), ..., x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$. This is implemented via

$$\frac{\widetilde{x(t)}}{(\widetilde{action\ Q(t)})^{-1} \widetilde{p(t)}} SRS(t).$$

Subject of $Q(t)$

2. The process of simultaneous expelling $R(t)$ = fuzzy checking with fuzziness $\mu(t)$

by the fuzzy set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), ...,$

$g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)),$

$B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), ..., B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through

$$\frac{\widetilde{Q(t)}}{(\widetilde{action\ R(t)})^{-1} \widetilde{B(t)}} SRS(t), \text{ where } \widetilde{Q(t)} \text{ can be anything.}$$

Subject of $R(t)$

3. Similarly for loop operators and others.

$SD_{16}(t)f$ – fuzzy software operators will differ only just because aggregates

$\widetilde{x(t)}, \widetilde{p(t)}, \widetilde{B(t)}, \widetilde{g(t)}$ will be formed from corresponding processes tSDS(t) for above mentioned programming operators through form (1.1) for more complex operators in forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical dynamic tSRS-program operator:

$$\begin{aligned} & \frac{B(q)}{(action\ Q(q))^{-1}} \frac{SRS(q)}{A(q)} = \\ & \text{Subject of } Q(t) \left\{ \frac{\{ \}}{\mu} \frac{fft(q)_{S_1 f(A(q) \cap B(q))} + A(q) - A(q) \cap B(q)}{(B(q) - A(q) \cap B(q))} \right\}, \text{ where action } Q\text{- contain.} \end{aligned}$$

fR^{1epr} -program operators (form $(action\ Q)^{-1}_{fR^{1prt}} action\ Q$ - fuzzy analogue of ${}^B_D S^1 t^A_B$ [10]).

For example, $R^{-1}_{fR^{1prt}} R$, where simultaneous expelling fuzzy checking with $\{\tilde{p}\}$ $\tilde{Q}$

fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ from the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$ and simultaneous R = fuzzy checking with fuzziness μ by the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$, $\tilde{Q}$ can be anything.

Appendix

Entire neural network as instantaneous simultaneous SRS-RAM in SRS -elements and fself- elements. $fself^{fself \dots fself}$, $ff1 \downarrow I \uparrow_{-1} ff2^{ff1 \downarrow I \uparrow_{-1} ff2 \dots ff1 \downarrow I \uparrow_{-1} ff2}$, $f sin \infty^{f sin \infty \dots f sin \infty}$. When activated in a neural network, the entire neural network becomes a working memory. Use of fself-energy as fuzzy activation or from outside.

4.8 Program operators PrSprt, PrtSpr, S¹e, Set₁

Similarly, for simultaneous parallel multiplication:

$PrSprt \begin{matrix} \{g_1\} \\ w_1 \end{matrix} * \begin{matrix} \{g_2\} \\ w_2 \end{matrix} \dots \begin{matrix} \{g_n\} \\ w_n \end{matrix} *$: the notation of the set $B_l, l=1, 2, \dots, n$, with elements $b_{li_1 i_2 \dots i_{m_j}} =$

$\left\{ g_{l_{i_1} *}}, g_{l_{i_2} *}, \dots, g_{l_{m_j} i_{m_j}} \right\}_{R_l}$ for any $\{l_{i_1}, l_{i_2}, \dots, l_{i_{m_j}}\}$ without repetitions, $x_l =$
 $Sprt_w^{\{K_l\}}$, K_l -set of any $\{k_{l_{i_1} *}, k_{l_{i_2} *}, \dots, k_{l_{i_{m_j} *} }\}$ without repeating them, $l = 1, 2,$
 $\dots, n$, $k_{l_{i_j}}$ -any digit, $i = 1, 2, \dots, m_j$, $R_l = Sprt_w^{\{l_{i_1} +, l_{i_2} +, \dots, l_{i_{m_j}}\}}$, R_l is the index of the
lower discharge (we choose an index on the scale of discharges): see Table I.1.

Then $PrSprt \frac{B_1}{x_1} + \frac{B_2}{x_2} + \dots + \frac{B_n}{x_n}$ gives the final result of simultaneous multiplication. Any system of calculus can be chosen, in particular binary. The most straightforward functional scheme of the assumed arithmetic-logical device for PrSprt-multiplication:

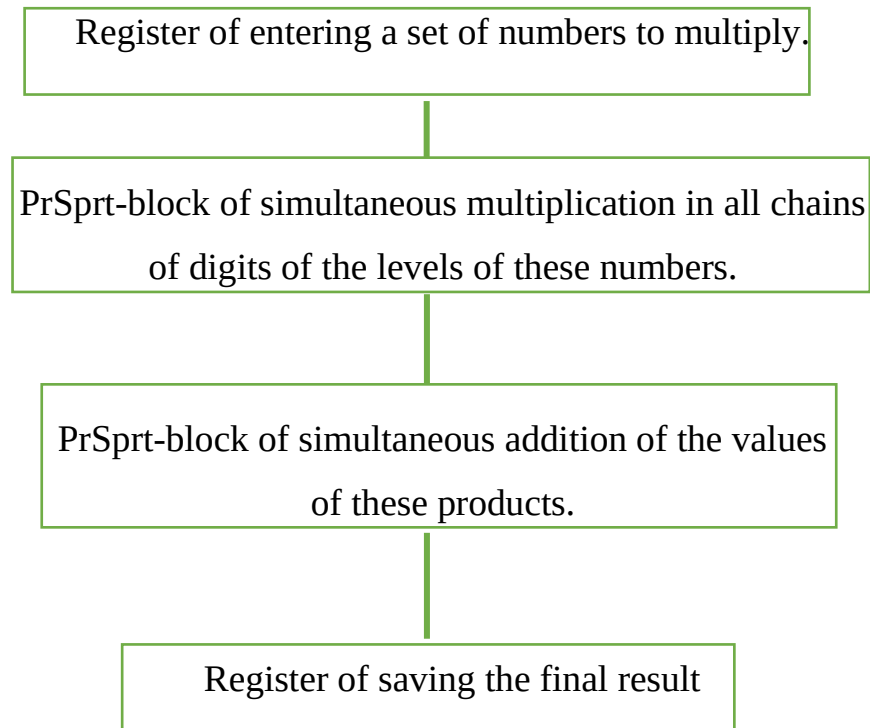


Fig. 4.7. The straightforward functional scheme of the assumed arithmetic-logical device for PrSprt-multiplication.

Remark. 4.8.1. The algorithm for simultaneously adding a set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by multiplying the first number from

the set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the set by the ones following it, etc.

Here are some of the PrSprt programming operators.

1. Simultaneous assignment of the expressions $\{p\} = (p_1, p_2, \dots, p_n)$ to the variables $\{g\} = (g_1, g_2, \dots, g_n)$. This is implemented via

$$\text{PrSprt} \begin{matrix} g_1 := & g_2 := & \dots & g_n := \\ p_1 & p_2 & \dots & p_n \end{matrix}.$$

2. Simultaneous checking the set of conditions $\{f\} = (f_1, f_2, \dots, f_n)$ for the set of expressions $\{B\} = (B_1, B_2, \dots, B_n)$. Implemented via

$$\text{PrSprt} \begin{matrix} IF\{B_1 f_1\} then & IF\{B_2 f_2\} then & \dots & IF\{B_n f_n\} then \\ x_1 & x_2 & \dots & x_n \end{matrix}, \text{ where } x_i \text{ (i =$$

1, ..., n) can be anything.

3. Similarly for loop operators and others.

PrS₃f– software operators will differ only in that the aggregates $\{g\}, \{p\}, \{B\}, \{f\}$ will be formed from the corresponding PrSprt program operators in form (1.1) or forms (1.1.1) - (1.4) for more complex operators.

The OS (operating system), the computer's principles, and the modes of operation for this programming are interesting. But this is already the material for the following publications.

The ideology of dynamic PrSprt and PrS₃f(t) can be used for programming:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t), p_2(t), \dots, p_n(t))$ to the variables $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$. is

$$\text{implemented through PrSprt} \begin{matrix} g_1(t) := & g_2(t) := & \dots & g_n(t) := \\ p_1(t) & p_2(t) & \dots & p_n(t) \end{matrix} \dots$$

2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f_2(t), \dots, f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), \dots, B_n(t))$ is implemented through

PrSprt $IF\{B_1(t)f_1(t)\} \text{ then } x_1(t) \quad IF\{B_2(t)f_2(t)\} \text{ then } x_2(t) \quad \dots \quad IF\{B_n(t)f_n(t)\} \text{ then } x_n(t)$ whe

re $x(t) = (x_1(t), x_2(t), \dots, x_n(t))$. can be any.

3. Similarly for loop operators and others.

$PrS_3f(t)$ – software operators will differ only in that the aggregates

$\{g(t)\}, \{p(t)\}, \{B(t)\}, \{f(t)\}$ will be formed from corresponding processes

PrSprt(t) for the above-mentioned programming operators through form (1.1) or

forms (1.1.1) – (1.4) for more complex operators.

The usage of PrSprt-elements for networks.

A. Galushkin's comprehensive monograph [3] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new mathematical process with parallel containment operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators. Parallel containment operators are more convenient for networks. Also, the main emphasis was placed on using processors operating using triodes, which are generally not used in PrSprt-networks. PrSprt-networks (PrSmnsprt) are a PrSprt-structure that can be built for the required weights. PrSprt-OS (PrSprt operating system) uses PrSprt-coding and PrSprt-translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b), where the number b is the code of the action, and the number a is the code of the object of this action. PrSprt-coding (or self-coding) is implemented through a matrix consisting of 2 columns (in the continuous case, two intervals of numbers). Here, the source encoding is used for all matrix rows simultaneously. PrSprt-translation is carried out by inversion. In this case, self-coding and self-translation will be more stable. The target weights $g_i(t)$ in

PrSprt(t) $\begin{matrix} \text{activation with } g_1(t) \\ \text{SmnPrSprt} \end{matrix} \quad \begin{matrix} \text{activation with } g_2(t) \\ \text{SmnPrSprt} \end{matrix} \quad \dots \quad \begin{matrix} \text{activation with } g_n(t) \\ \text{SmnPrSprt} \end{matrix}$

are chosen for necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [6]. We will touch on the difference of this only for hierarchical complex networks. The same simple executing programs are in the cores of simple artificial neurons of type PrSprt (designation - mnPrSprt) for simple information processing. More complex executing programs are used for mnPrSprt nodes. PrSprt-threshold element –

$$\text{sgn}(\text{PrSprt}(t) \begin{pmatrix} g_1(t)w_1(t) & g_2(t)w_2(t) & \dots & g_n(t)w_n(t) \\ x_1 & x_2 & \dots & x_n \end{pmatrix}), x=(x_1, x_2, \dots, x_n) -$$

mnPrSprt, $w(t)=(w_1(t), w_2(t), \dots, w_n(t))$ – source signals values, $\{g(t)\} =$

$(g_1(t), g_2(t), \dots, g_n(t))$ – PrSprt-synapses weights. The first level of

mnPrSprt consists of simple mnPrSprt. The second level of mnPrSprt

consists of $\text{PrSprt}(t) \begin{matrix} \text{mnPrSprt} & \text{mnPrSprt} & \dots & \text{mnPrSprt} \\ D_1 & D_2 & \dots & D_n \end{matrix}$ – PrSprt-node

of mnPrSprt in range $D = (D_1, D_2, \dots, D_n)$, D - capacity for mnPrSprt node. The

third level of mnPrSprt consists of

$$\text{PrSprt}(t) \begin{matrix} \text{PrSprt}(t) \begin{matrix} \text{mnPrSprt} & \text{mnPrSprt} & \dots & \text{mnPrSprt} \\ D_1 & D_2 & \dots & D_n \end{matrix} & \text{PrSprt}(t) \begin{matrix} \text{mnPrSprt} & \text{mnPrSprt} & \dots & \text{mnPrSprt} \\ D_1 & D_2 & \dots & D_n \end{matrix} & \dots & \text{mnPrSprt} \\ & & & D_n \end{matrix}$$

PrSprt²- node of mnPrSprt in range D , thus D becomes capacity of itself in

itself as an element for mnPrSprt. For our networks, it is sufficient to use

PrSprt²- nodes of mnPrSprt, but self-level is higher in living organisms,

particularly PrSprtⁿ-, $n \geq 3$. The target structure or the corresponding program

enters the target unit using alternating current. After that, all networks or

parts of them are activated according to the indicative goal. It may appear

that we are leaving the network ideology, but these networks are a complex hierarchy of different levels, like living organisms.

Threshold element of extended type PrSprt -

$$\begin{matrix} B_1 & B_2 & \dots & B_n \\ \{qy\}_1 & \{qy\}_2 & \dots & \{qy\}_n \end{matrix} \text{PrSprt} \begin{matrix} \{ax\}_1 & \{ax\}_2 & \dots & \{ax\}_n \\ B_1 & B_2 & \dots & B_n \end{matrix}, B_1, B_2, \dots, B_n$$

- artificial neurons of type PrSprt (designation - mnPrSprt) , $x=(x_1, x_2, \dots, x_n)$

are the values of the initial signals, $a=(a_1,a_2,\dots,a_n)$ are the weights of PrSprt-synapses and the values of the output signals $\{qy\}$.

Remark 4.8.1. A neural network can be thought of as a learnable parallel dynamic operator.

Remark 4.8.2. Traditional scientific approaches through classical mathematics make it possible to describe only at the usual energy level. Here we consider an approach that makes describing processes with finer energies possible. mnPrSprt

contains PrSprt(t) $\begin{matrix} \text{eprogram}_1(t) & \text{eprogram}_2(t) & \dots & \text{eprogram}_n(t) \\ \text{mnPrSprt} & \text{mnPrSprt} & \dots & \text{mnPrSprt} \end{matrix}$ "

eprogram –executing program in PrSprt- OS. PrSprt-OS (or Prself-OS) is based on PrSprt-assembly language (or Prself-assembly language), which is based on assembly language through PrSprt-approach in turn, if the base of elements of PrSprt-networks is sufficient. The eprograms are in PrSprt-programming environments (or Prself-programming environments), but this question and PrSprt-networks base will be considered in the following monographs. In particular, eprograms may contain PrSprt- programming operators. In mnPrSprt cores, the constant memory PrSprt with correspondent eprograms depending on mnPrSprt.

The OS (operating system) and the principles and modes of operation of the PrSprt-networks for this programming are interesting. But this is already the material for the next publications.

Here is developed a helicopter model without a main and tail rotors based on PrSprt – physics and special neural networks with artificial neurons operating in normal and PrSprt-modes. Let's denote this model through SmnPrSprt. To do this, it's proposed to use mnPrSprt of different levels; for example, for the usual mode, mnPrSprt serves for the initial processing of signals and the transfer of information to the second level, etc., to the nodal center, then checked. In case of an anomaly - local PrSprt–mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, SmnPrSprt is activated with the desired "target weight." Here are realized other tasks also. To reach the self-

energy level, the mode $Sprt_{SmnPrSprt}^{SmnPrSprt}$ is used. In normal mode, it's planned to carry out the movement of SmnPrSprt on jet propulsion by converting the energy of the emitted gases into a vortex to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the SmnPrSprt for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is drainage of exhaust gases outside the SmnPrSprt. SmnPrSprt is represented by a neural network that extends from the center of one of the main clusters of PrSprt - artificial neurons to the shell, turning into the body itself. Above the operator's cabin is the central core of the neural network and the target block, responsible for performing the "target weights" and auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for SmnPrSprt's actions is below the operator's cab. In PrSprt – mode, the entire network or its sections are PrSprt – activated to perform specific tasks, in particular, with "target weights." In the target, block used PrSprt -coding, PrSprt - translation for activation of all networks to "target weights" simultaneously, then – the reset of this PrSprt-coding after activation. Unfortunately, triodes are not suitable for PrSprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for eprograms may be used instead triodes since there is no necessity to unbend the alternating current to direct. The PrSprt-operative memory belt is disposed around a central core of SmnPrSprt. There are PrSprt-coding, PrSprt-translation, and PrSprt-realize of eprograms and the programs from the archives without extraction, PrSprt-coding and PrSprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. PrSprt – structure or an eprogram if one is present of needed «target weight» are taken in target block at PrSprt – activation of the networks. $Sprt_{activation}^{SmnPrSprt,f}$ derives SmnPrSprt to the self-level boundary with target weight f.

It's used an alternating current of above high frequency and ultra-violet light, which can work with PrSprt – structures in PrSprt–modes by its nature to activate the networks or some of its parts in PrSprt–modes and locally using PrSprt–mode. Above high frequently alternating current go through mercury bearers. That's why overheating does not occur. The power of the alternating current above high frequently increases considerably for the target block. The activation of all networks is realized to indicate “target weights”.

Program operators PrSprt, PrtSpr, S¹e, Set₁.

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through PrSprt-Networks in one of the central departments of which a conventional computer system is located. The parallel processor is itself preprogram with direct parallel computing not through serial computing.

Using conventional parallel coding by a parallel computer system, through a Target-block with a PrSprt -program operator -

$\text{PrSprt}(t) \begin{matrix} g_1(t)w_1(t) \\ \text{activation} \end{matrix} \quad \begin{matrix} g_2(t)w_n(t) \\ \text{activation} \end{matrix} \quad \dots \quad \begin{matrix} g_n(t)w_n(t) \\ \text{activation} \end{matrix}$, it will be possible to obtain

the execution of the parallel actions $(g_1(t), g_2(t), \dots, g_n(t))$ with the desired target weights $w(t) = (w_1(t), w_2(t), \dots, w_n(t))$. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For PrSprt-coding and PrSprt-translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna Strickland, or a combination of them.

For the desired action, for example, using the direct parallel program of operator

$\text{PrSprt}(t) \begin{matrix} (\text{UHF AC})_1(t) := Q_1(t) \\ \text{activation} \end{matrix} \quad \begin{matrix} (\text{UHF AC})_2(t) := Q_2(t) \\ \text{activation} \end{matrix} \quad \dots \quad \begin{matrix} (\text{UHF AC})_n(t) := Q_n(t) \\ \text{activation} \end{matrix}$, we

simultaneously enter the desired set of codes $Q_i(t)$, $i = 1, 2, \dots, n$, using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects.

Consider the types of direct parallel program operators:

- 1) PrSprt-program operators
- 2) PrtSpr-program operators
- 3) S¹e - program operators
- 4) Set₁- program operators

Here are some of the PrSprt-program operators:

1. Simultaneous assignment of the expressions $\{p\} = (p_1, p_2, \dots, p_n)$ to the variables $\{g\} = (g_1, g_2, \dots, g_n)$. This is implemented via

$$\text{PrSprt} \begin{matrix} g_1 := & g_2 := & \dots & g_n := \\ p_1 & p_2 & \dots & p_n \end{matrix} .$$

2. Simultaneous checking the set of conditions $\{f\} = (f_1, f_2, \dots, f_n)$ for the set of expressions $\{B\} = (B_1, B_2, \dots, B_n)$. Implemented via

$$\text{PrSprt} \begin{matrix} IF\{B_1 f_1\} then & IF\{B_2 f_2\} then & \dots & IF\{B_n f_n\} then \\ x_1 & x_2 & \dots & x_n \end{matrix}, \text{ where } x_i \text{ (i =$$

1, ..., n) can be anything.

3. Similarly for loop operators and others.

PRSprt-algorithm Examples:

- 1) Simultaneous addition and simultaneous parallel multiplication of sets

elements $\{g_i\} = (g_{i_1}, g_{i_2}, \dots, g_{i_{m_j}})$, $i = 1, 2, \dots, n$, $j = 1, 2, \dots, k$ (See point 1,

2 in **Math Prself** [2])

- 2) parallel pattern recognition:

$$\text{PrSprt} \begin{matrix} IF\{q_1 \in \text{image archive}_1\} then & IF\{q_2 \in \text{image archive}_2\} then & \dots & IF\{q_n \in \text{image archive}_n\} then \\ \text{Name of } q_1 & \text{Name of } q_2 & \dots & \text{Name of } q_n \end{matrix}$$

The example of PrSprt-program is

$$\text{PrSprt} \begin{matrix} \text{PrSprt} \begin{matrix} g_1 := & g_2 := & \dots & g_n := \\ p_1 & p_2 & \dots & p_n \end{matrix} & \text{PrSprt} \begin{matrix} IF\{B_1 f_1\} then & IF\{B_2 f_2\} then & \dots & IF\{B_n f_n\} then \\ w_1 & w_2 & \dots & w_n \end{matrix} & \dots & \text{PrSprt} \begin{matrix} w_1 & w_2 & \dots & w_n \\ w_1 & w_2 & \dots & w_n \end{matrix} & \dots & \text{PrSprt} \begin{matrix} w_1 & w_2 & \dots & w_n \\ w_1 & w_2 & \dots & w_n \end{matrix} \end{matrix} .$$

PrS_3f – software operators will differ only just because aggregates

$\{g\}, \{p\}, \{B\}, \{f\}$ will be formed from corresponding PrSprt-program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4).

For example, $Sprt_{g\{R\}}^{\{R\}}$ is the capacity in itself of the second type if $g\{R\}$ is a program capable of generating $\{R\}$.

The example of self-program of the first type is

$$\begin{array}{ccccccc} \text{PrSprt}^{g_1 := g_2 := \dots g_n :=} & \text{PrSprt}^{IF\{B_1 f_1\} \text{ then } IF\{B_2 f_2\} \text{ then } \dots IF\{B_n f_n\} \text{ then}} & \dots & \text{PrSprt}^{w_1 w_2 \dots w_n} \\ p_1 p_2 \dots p_n & w_1 w_2 \dots w_n & & w_1 w_2 \dots w_n \\ \text{PrSprt}^{g_1 := g_2 := \dots g_n :=} & \text{PrSprt}^{IF\{B_1 f_1\} \text{ then } IF\{B_2 f_2\} \text{ then } \dots IF\{B_n f_n\} \text{ then}} & \dots & \text{PrSprt}^{w_1 w_2 \dots w_n} \\ p_1 p_2 \dots p_n & w_1 w_2 \dots w_n & & w_1 w_2 \dots w_n \end{array} .$$

PrSprt-coding: 1) set A_i to set B_i , 2) set A_i to a point q_i , where the elements of the sets A_i, B_i can be continuous, ($i = 1, 2, \dots, n; j = 1, 2, \dots, m$). For example,

$$\text{PrSprt} \begin{array}{cccc} A_1 & A_2 & \dots & A_n \\ B_1 & B_2 & \dots & B_n \end{array} .$$

There are PrSprt -coding, PrSprt-translation, PrSprt-realize of preprograms and of the programs from the archives without extraction theirs

Self-coding: 1) set A_i to set A_i , i.e. A_i on itself, 2) set A_i to a point q_i in forms (1.1) - (1.4), where the elements of the sets A_i can be continuous. For example,

$$\text{PrSprt} \begin{array}{cccc} A_1 & A_2 & \dots & A_n \\ A_1 & A_2 & \dots & A_n \end{array} .$$

One of the central departments of the control system should be a computer system of the usual type of the desired level. In symbiosis with PrSprt-Networks, it will provide a holistic operation of the control system in three modes: conventional serial through a conventional type computer system, direct parallel through PrSprt -Networks and series-parallel. Codes from a conventional type computer system will be used via PrSprt -connectors in PrSprt - coding, for example:

$$\text{PrSprt}(t) \begin{array}{cccc} (\text{UHF AC})_1(t) := Q_1(t) & (\text{UHF AC})_2(t) := Q_2(t) & \dots & (\text{UHF AC})_n(t) := Q_n(t) \\ \text{activation} & \text{activation} & \dots & \text{activation} \end{array} .$$

UHF AC field activation is used.

Consider the dynamic PrSprt and $PrS_3f(t)$ programming:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t), p_2(t), \dots, p_n(t))$ to the variables $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$ is implemented through $\text{PrSprt} \begin{matrix} g_1(t) := & g_2(t) := & \dots & g_n(t) := \\ p_1(t) & p_2(t) & \dots & p_n(t) \end{matrix}$.

2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f_2(t), \dots, f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), \dots, B_n(t))$ is implemented through

$\text{PrSprt} \begin{matrix} IF\{B_1(t)f_1(t)\} \text{ then} & IF\{B_2(t)f_2(t)\} \text{ then} & \dots & IF\{B_n(t)f_n(t)\} \text{ then} \\ x_1(t) & x_2(t) & \dots & x_n(t) \end{matrix}$ where

$x(t) = (x_1(t), x_2(t), \dots, x_n(t))$. can be any.

3. Similarly for loop operators and others.

$\text{PrS}_3 f(t)$ – software operators will differ only in that the aggregates

$\{g(t)\}, \{p(t)\}, \{B(t)\}, \{f(t)\}$ will be formed from corresponding processes

$\text{PrSprt}(t)$ for the above-mentioned programming operators through form (1.1) or forms (1.1.1) – (1.4) for more complex operators.

Consider PrtSpr -program operators. The ideology of PrtSpr and Prt_{s_4f} is parallel analogue of t_{s_4f} [11] can be used for programming. Here are some of the PrtSpr -program operators.

1. Simultaneous expelling assignment of the expressions $\{p\} = (p_1, p_2, \dots, p_n)$ from the variables $\{g\} = (g_1, g_2, \dots, g_n)$. It's implemented through $\begin{matrix} g_1 & g_2 & \dots & g_n \\ =: p_1 & =: p_2 & \dots & =: p_n \end{matrix} \text{PrSprt}$.

2. Simultaneous expelling checks the set of conditions $\{f\} = (f_1, f_2, \dots, f_n)$ for a set of expressions $\{B\} = (B_1, B_2, \dots, B_n)$. It's implemented through

$\begin{matrix} x_1 & x_2 & \dots & x_n \\ IF\{B_1 g_1\} \text{ then} & IF\{B_2 g_2\} \text{ then} & \dots & IF\{B_n g_n\} \text{ then} \end{matrix} \text{PrSprt}$, where x_i ($i = 1, \dots, n$) can be anything.

3. Similarly for loop operators and others.

$Prt_{s_{4f}}$ – software operators will differ only just because aggregates

$\{g\}, \{p\}, \{B\}, \{f\}$ will be formed from corresponding PrtSpr program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4).

Consider hierarchical PrtSpr-program operator

$$\begin{matrix} C_1 & C_2 & \dots & C_m \\ D_1 & D_2 & \dots & D_m \end{matrix} \text{PrSrt} = \left\{ \begin{matrix} \sum_{i=1}^m Q_i + \begin{matrix} \{\} \\ D_1 - D_1 \cap C_1 \end{matrix} \begin{matrix} \{\} \\ D_2 - D_2 \cap C_2 \end{matrix} \dots \begin{matrix} \{\} \\ D_m - D_m \cap C_m \end{matrix} \text{PrSrt} \\ \sum_{i=1}^m (C_i - D_i \cap C_i) - (D_i - D_i \cap C_i) \end{matrix} \right\}, \text{ where } Q_i$$

is oself-set for $(D_i \cap C_i)$ [14].

Consider the dynamic PrtSpr(t) and $Prt(t)_{s_{4f}}$ programming at time t.

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t), p_2(t), \dots, p_n(t))$ to the variables $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$ is

implemented through $\begin{matrix} g_1(t) := & g_2(t) := & \dots & g_n(t) := \\ p_1(t) & p_2(t) & \dots & p_n(t) \end{matrix} \text{PrSprt}(t).$

2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f_2(t), \dots, f(t)_n)$ for a set of expressions $\{B(t)\} =$

$(B_1(t), B_2(t), \dots, B_n(t))$ is implemented through

$\begin{matrix} IF\{B_1(t)f_1(t)\} \text{ then } & IF\{B_2(t)f_2(t)\} \text{ then } & \dots & IF\{B_n(t)f_n(t)\} \text{ then } \\ x_1(t) & x_2(t) & \dots & x_n(t) \end{matrix} \text{PrSprt}(t),$

where $x(t) = (x_1(t), x_2(t), \dots, x_n(t))$. can be any.

3. Similarly for loop operators and others.

$Prt(t)_{s_{4f}}$ – software operators will differ only in that the aggregates

$\{g(t)\}, \{p(t)\}, \{B(t)\}, \{f(t)\}$ will be formed from corresponding processes

PrSprt(t) for the above-mentioned programming operators through form (1.1) or forms (1.1.1) – (1.4) for more complex operators.

Consider hierarchical dynamic PrSrt(t)-program operator:

$$\begin{matrix} C_1(t) & C_2(t) & \dots & C_m(t) \\ D_1(t) & D_2(t) & \dots & D_m(t) \end{matrix} \text{PrSrt}(t) = \left\{ \begin{matrix} \sum_{i=1}^m Q_i(t) + \{ \} & \{ \} & \dots & \{ \} \\ D_1(t) - D_1(t) \cap C_1(t) & D_2(t) - D_2(t) \cap C_2(t) & \dots & D_m(t) - D_m(t) \cap C_m(t) \end{matrix} \text{PrSrt} \right\},$$

$$\frac{\sum_{i=1}^m (C_i(t) - D_i(t) \cap C_i(t)) - (D_i(t) - D_i(t) \cap C_i(t))}{\sum_{i=1}^m (C_i(t) - D_i(t) \cap C_i(t)) - (D_i(t) - D_i(t) \cap C_i(t))}$$

where $Q_i(t)$ is oself-set for $(D_i(t) \cap C_i(t))$ [11].

Consider PrSrt-program operators

$$\text{Sprt}_{t_0} \left\{ \begin{matrix} q(u \ u \text{PrSpr}_u^u \ u) \\ W_q \text{Sprt}_{q(u \ u \text{PrSpr}_u^u \ u)}^{E_q} \end{matrix} \text{Sprt}_{d_r(E_{in}^{l^{d_r}})}^{\{E^{ex}l^{d_r}\}} \right\} \text{—program structure example,}$$

where the assemblage point d_r is the cursor, it is quite complex self—program.

Remark.F.1.5.2. Energy of a living organism other than humans:

$$\text{Prf}(r, u(E_q)) = \text{Sprt}_{t_0} \left\{ \begin{matrix} q(u \ u \text{PrSpr}_u^u \ u) \\ W_q \text{Sprt}_{q(u \ u \text{PrSpr}_u^u \ u)}^{E_q} \end{matrix} \text{Sprt}_{d_r(E_{in}^{l^{d_r}})}^{\{E^{ex}l^{d_r}\}} \right\} (**_{F.1}).$$

Energy of a living organism of a person:

$$\text{Prhf}(r, u(E_q)) = \text{Sprt}_{t_0} \left\{ \begin{matrix} q(u \ u \text{PrSpr}_u^u \ u) \\ W_q \text{Sprt}_{q(u \ u \text{PrSpr}_u^u \ u)}^{E_q} \end{matrix} \text{Sprt}_{d_r(\text{self}(E_{in}^{l^{d_r}}))}^{\{E^{ex}l^{d_r}\}} \right\} (***_F.1).$$

$\begin{matrix} u & u \\ u & u \end{matrix} \text{PrSpr}_u^u$ - internal energy of a living organism of double energy structure,

q- a gap in the energy cocoon of a living organism, r-the position of the

assemblage point d_r on the energy cocoon of a living organism, W_q - energy

prominences from the gap in the cocoon of a living organism, E_q -external energy

entering the gap in the cocoon of a living organism, $E^{ex}l^{d_r}$ - a bundle of fibers of

external energy self-capacities from outside the cocoon, collected at the point of

assembly of the cocoon of a living organism, $E_{in}l^{d_r}$ - a bundle of fibers of external

energy self-capacities from inside the cocoon, collected at the point of assembly of

the cocoon of a living organism in the same position r of the assemblage point d_r .

d_r is the subject of identifying the energy fibers of the subtle energy of the

Universe in position r both outside and inside the cocoon.

$(**_{F.1})$, $(***_{F.1})$ can be interpreted as the program operators.

4.9 Program operators PrfSprt, PrtfSpr, fS¹e, fSet₁

Similarly, for simultaneous parallel multiplication:

$\text{PrfSprt}_{w_1}^{\{g_1\} *} \quad \{g_2\}^* \quad \dots \quad \{g_n\}^*_{w_n}$. The notation of the fuzzy set B_l , l

$= 1, 2, \dots, n$, with elements $b_{li_1 i_2 \dots i_{m_j}} =$

$\left\{ g_{l_{i_1}}^*, g_{l_{i_2}}^*, \dots, g_{l_{m_j}}^* \right\}_{R_l}$ for any $\{l_{i_1}, l_{i_2}, \dots, l_{m_j}\}$ without repetitions, $x_l =$

$\text{Sprt}_w^{\{K_l\}}$, K_l -set of any $\{k_{l_{i_1}}^*, k_{l_{i_2}}^*, \dots, k_{l_{m_j}}^*\}$ without repeating them, l

$= 1, 2, \dots, n$, $k_{l_{i_j}}$ -any digit, $i = 1, 2, \dots, m_j$, $R_l = \text{Sprt}_w^{\{l_{i_1} + l_{i_2} + \dots, l_{i_{m_j}}\}}$, R_l is the

index of the lower discharge (we choose an index on the scale of discharges): see Table I.1.

Then $\text{PrfSprt}_{x_1}^{B_1 +} \quad B_2 + \quad \dots \quad B_n +_{x_n}$ gives the final result of simultaneous

multiplication. Any system of calculus can be chosen, in particular binary. The most straightforward functional scheme of the assumed arithmetic-logical device for PrfSprt-multiplication:

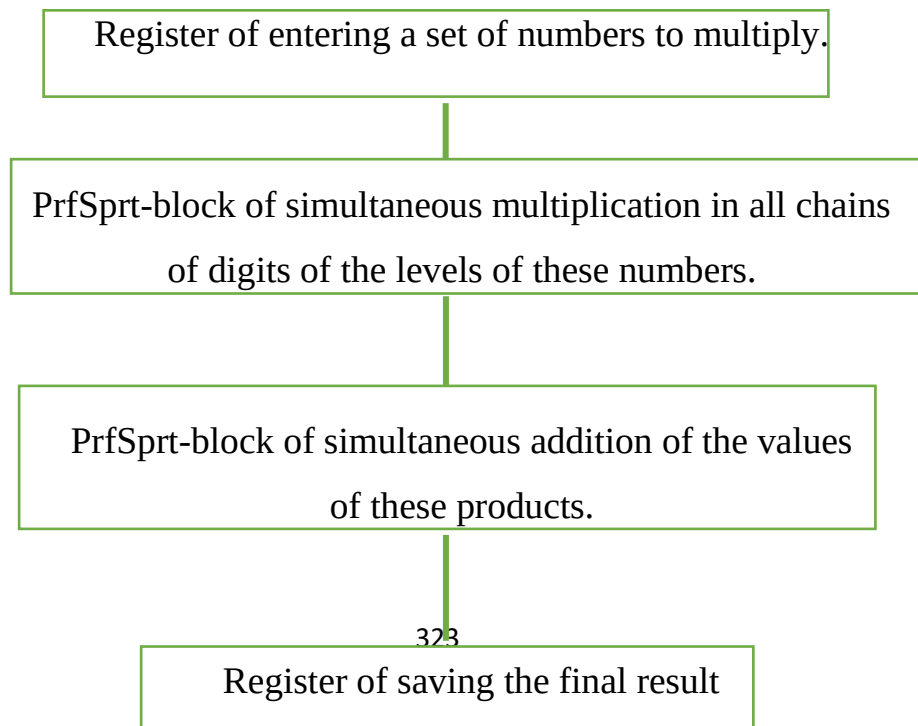


Fig.4.8. The straightforward functional scheme of the assumed arithmetic-logical device for PrfSprt-multiplication.

Remark. The algorithm for simultaneously adding a set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by multiplying the first number from the set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are represented by multiplying the second number from the set by the ones following it, etc.

Here are some of the PrfSprt programming operators.

1. Simultaneous assignment of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$. This is implemented via
$$\text{PrfSprt} \begin{matrix} x_1 := & x_2 := & \dots & x_n := \\ p_1 & p_2 & \dots & p_n \end{matrix} .$$
2. Simultaneous checking the set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$. Implemented via
$$\text{PrfSprt} \begin{matrix} IF\{B_1 g_1\} then & IF\{B_2 g_2\} then & \dots & IF\{B_n g_n\} then \\ w_1 & w_2 & \dots & w_n \end{matrix}, \text{ where } w_i (i = 1, \dots, n) \text{ can be anything.}$$
3. Similarly for loop operators and others.

PrfS₃f– software operators will differ only in that the aggregates $\{\tilde{g}\}, \{\tilde{p}\}, \{\tilde{B}\}, \{\tilde{x}\}$ will be formed from the corresponding PrfSprt program operators in form (1.1) and for more complex operators in the forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

The OS (operating system), the computer's principles, and the modes of operation for this programming are interesting. But this is already the material for the following articles.

The ideology of dynamic PrfSprt and PrS₃f(t) can be used for programming:

1. The process of simultaneous assignment of the

expressions $\widetilde{p(t)} = (p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \dots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)} = (x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ is

implemented through PrfSprt
$$\begin{matrix} x_1(t) := & x_2(t) := & \dots & x_n(t) := \\ p_1(t) & p_2(t) & \dots & p_n(t) \end{matrix} \quad \cdot \cdot$$

2. The process of simultaneous check the set of conditions $\widetilde{g(t)} = (g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \dots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for a set of

expressions $\widetilde{B(t)} = (B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \dots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through

PrfSprt
$$\begin{matrix} IF\{B_1(t)g_1(t)\} then & IF\{B_2(t)g_2(t)\} then & \dots & IF\{B_n(t)g(t)\} then \\ w_1(t) & w_2(t) & \dots & w_n(t) \end{matrix} \quad \text{wh}$$

ere $w(t) = (w_1(t), w_2(t), \dots, w_n(t))$. can be any.

3. Similarly for loop operators and others.

PrfS₃f(t)– software operators will differ only in that the aggregates

$\{\widetilde{g(t)}\}, \{\widetilde{p(t)}\}, \{\widetilde{B(t)}\}, \{\widetilde{x(t)}\}$ will be formed from corresponding processes

PrfSprt(t) for the above-mentioned programming operators through form (1.1) or forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*) for more complex operators.

The usage of PrfSprt-elements for networks.

A. Galushkin's comprehensive monograph [3] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new mathematical process with parallel fuzzy containment operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators.

Parallel fuzzy containment operators are more convenient for networks. Also, the main emphasis was placed on using processors operating using triodes, which are generally not used in Sprt-networks. PrfSprt-networks (SmnPrfSprt) are a PrfSprt-structure that can be built for the required weights. PrfSprt-OS (PrfSprt operating system) uses PrfSprt-coding and PrfSprt-translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b), where the number b is the code of the action, and the number a is the code of the object of this action. PrfSprt-coding (or Prfself-coding) is implemented through a matrix consisting of 2 columns (in the continuous case, two intervals of numbers). Here, the source encoding is used for all matrix rows simultaneously. PrfSprt-translation is carried out by inversion. In this case, Prfself-coding and Prfself-translation will be more stable. The target weights $g_i(t)$ in

$$\text{PrfSprt}(t) \begin{matrix} \text{activation with } g_1(t) \\ \text{SmnPrfSprt} \end{matrix} \quad \text{activation with } g_2(t) \quad \dots \quad \text{activation with } g_n(t) \quad \text{SmnPrfSprt}$$

are chosen for necessary tasks. We will not touch on the issues of applications, or network optimization. They are described in detail by Galushkin [6]. We will touch on the difference of this only for hierarchical complex networks. The same simple executing programs are in the cores of simple artificial neurons of type PrfSprt (designation - mnPrfSprt) for simple information processing. More complex executing programs are used for mnPrfSprt nodes. PrfSprt-threshold element –

$$\text{sgn}(\text{PrfSprt}(t) \begin{matrix} g_1(t)w_1(t) \\ x_1 \end{matrix} \quad \begin{matrix} g_2(t)w_2(t) \\ x_2 \end{matrix} \quad \dots \quad \begin{matrix} g_n(t)w_n(t) \\ x_n \end{matrix}), x=(x_1, x_2, \dots, x_n) -$$

$$\text{mnPrfSprt}, \widetilde{W}(t) = (w_1(t)|\mu_{\widetilde{W}(t)}(w_1(t)), w_2(t)|\mu_{\widetilde{W}(t)}(w_2(t)), \dots, w_n(t)|\mu_{\widetilde{W}(t)}(w_n(t))).$$

– source signals values, $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$ – PrfSprt-synapses weights. The first level of mnPrfSprt consists of simple mnPrfSprt. The second level of mnPrfSprt consists of

$$\text{PrfSprt}(t) \begin{matrix} \text{mnPrfSprt} \\ D_1 \end{matrix} \quad \text{mnPrfSprt} \quad \dots \quad \text{mnPrfSprt} \quad \begin{matrix} \\ D_n \end{matrix} - \text{PrfSprt-node of mnPrfSprt}$$

in range $D = (D_1, D_2, \dots, D_n)$, D- capacity for mnPrfSprt node. The third level of

$$\begin{array}{ccccccc} \text{PrfSprt}(t) & \text{PrfSprt}(t)^{\text{mnPrfSprt}} & \text{mnPrfSprt} & \dots & \text{mnPrfSprt} & \text{PrfSprt}(t)^{\text{mnPrfSprt}} & \text{mnPrfSprt} \\ D_1 & D_2 & \dots & D_n & D_1 & D_2 & \dots \\ & D_1 & & & D_2 & \dots & \\ & & & & & D_n & \dots \\ & & & & & & D_n \end{array}$$
$$\begin{matrix} B_1 & B_2 & \dots & B_n \\ \{qy\}_1 & \{qy\}_2 & \dots & \{qy\}_n \end{matrix} \text{PrfSprt} \begin{matrix} \{ax\}_1 & \{ax\}_2 & \dots & \{ax\}_n \\ B_1 & B_2 & \dots & B_n \end{matrix}, B_1, B_2, \dots, B_n -$$

Remark 4.9.0. A neural network can be thought of as a learnable parallel dynamic operator.

$$\text{contains PrfSprt}(t) \begin{matrix} f\text{eprogram}_1(t) \\ \text{mnPrfSprt} \end{matrix} \quad \begin{matrix} f\text{eprogram}_2(t) \\ \text{mnPrfSprt} \end{matrix} \quad \dots \quad \begin{matrix} f\text{eprogram}_n(t) \\ \text{mnPrfSprt} \end{matrix},$$

327

cores, the constant memory PrfSprt with correspondent eprograms depending on mnPrfSprt.

The OS (operating system) and the principles and modes of operation of the PrfSprt-networks for this programming are interesting. But this is already the material for the next publications.

Here is developed a helicopter model without a main and tail rotors based on PrfSprt – physics and special neural networks with artificial neurons operating in normal and PrfSprt-modes. Let's denote this model through SmnPrfSprt. To do this, it's proposed to use mnPrfSprt of different levels; for example, for the usual mode, mnPrfSprt serves for the initial processing of signals and the transfer of information to the second level, etc., to the nodal center, then checked. In case of an anomaly - local PrfSprt-mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, SmnPrfSprt is activated with the desired "target weight." Here are realized other tasks also. To reach the fself-energy level, the mode $Sprt_{SmnPrfSprt}^{SmnPrfSprt}$ is used. In normal mode, it's planned to carry out the movement of SmnPrfSprt on jet propulsion by converting the energy of the emitted gases into a vortex to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the SmnPrfSprt for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is drainage of exhaust gases outside the SmnPrfSprt. SmnPrfSprt is represented by a neural network that extends from the center of one of the main clusters of PrfSprt - artificial neurons to the shell, turning into the body itself. Above the operator's cabin is the central core of the neural network and the target block, responsible for performing the "target weights" and auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for SmnPrfSprt's actions is below the operator's cab. In PrfSprt – mode, the entire network or its sections are PrfSprt – activated to perform specific tasks, in particular, with "target weights." In the target, block used PrfSprt -coding,

PrfSprt -translation for activation of all networks to "target weights" simultaneously, then –the reset of this PrfSprt-coding after activation. Unfortunately, triodes are not suitable for PrfSprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for eprograms may be used instead triodes since there is no necessity to unbend the alternating current to direct. The PrfSprt-operative memory belt is disposed around a central core of SmnPrfSprt. There are PrfSprt-coding, PrfSprt-translation, and PrfSprt-realize of eprograms and the programs from the archives without extraction, PrfSprt-coding and PrfSprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. PrfSprt – structure or an eprogram if one is present of needed «target weight» are taken in target block at PrfSprt – activation of the networks. $Sprt_{activation}^{SmnPrfSprt,f}$ derives SmnPrfSprt to the self-level boundary with target weight f.

It's used an alternating current of above high frequency and ultra-violet light, which can work with PrfSprt – structures in PrfSprt–modes by its nature to activate the networks or some of its parts in PrfSprt–modes and locally using PrfSprt–mode. Above high frequently alternating current go through mercury bearers. That's why overheating does not occur. The power of the alternating current above high frequently increases considerably for the target block. The activation of all networks is realized to indicate “target weights.”

Program operators PrfSprt, PrtSpr, S¹e, Set₁.

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through PrfSprt-Networks in one of the central departments of which a conventional computer system is located. The parallel processor is itself preprogram with direct parallel computing not through serial computinF.

Using conventional parallel coding by a parallel computer system, through a Target-block with a PrfSprt -program operator -

PrfSprt(t) $\begin{matrix} g_1(t)w_1(t) \\ activation \end{matrix} \quad \begin{matrix} g_2(t)w_n(t) \\ activation \end{matrix} \quad \dots \quad \begin{matrix} g_n(t)w_n(t) \\ activation \end{matrix}$, it will be possible to obtain the execution of a parallel action $(g_1(t), g_2(t), \dots, g_n(t))$ with the desired target weight $w(t) = (w_1(t), w_2(t), \dots, w_n(t))$. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For PrfSprt-coding and PrfSprt-translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna Strickland, or a combination of them. For the desired action, for example, using the direct parallel program of operator PrfSprt(t) $\begin{matrix} (UHF AC)_1(t) := Q_1(t) \\ activation \end{matrix} \quad \begin{matrix} (UHF AC)_2(t) := Q_2(t) \\ activation \end{matrix} \quad \dots \quad \begin{matrix} (UHF AC)_n(t) := Q_n(t) \\ activation \end{matrix}$, we simultaneously enter the desired set of codes $Q_1(t)$ using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects. Consider the types of direct parallel program operators:

- 1) PrfSprt-program operators
- 2) PrtSpr-program operators
- 3) S¹e - program operators
- 4) Set₁- program operators

Here are some of the PrfSprt-program operators:

1. Simultaneous assignment of the expressions $\{p\} = (p_1, p_2, \dots, p_n)$ to the variables $\{g\} = (g_1, g_2, \dots, g_n)$. This is implemented via

$$\text{PrfSprt} \begin{matrix} g_1 := \\ p_1 \end{matrix} \quad \begin{matrix} g_2 := \\ p_2 \end{matrix} \quad \dots \quad \begin{matrix} g_n := \\ p_n \end{matrix}.$$

2. Simultaneous checking the set of conditions $\{f\} = (f_1, f_2, \dots, f_n)$ for the set of expressions $\{B\} = (B_1, B_2, \dots, B_n)$. Implemented via

$$\text{PrfSprt} \begin{matrix} IF\{B_1 f_1\} then \\ x_1 \end{matrix} \quad \begin{matrix} IF\{B_2 f_2\} then \\ x_2 \end{matrix} \quad \dots \quad \begin{matrix} IF\{B_n f_n\} then \\ x_n \end{matrix}, \text{ where } x_i \text{ (i =$$

1, ..., n) can be anything.

3. Similarly for loop operators and others.

PrfSprt-algorithm Examples:

1. Simultaneous addition and simultaneous parallel multiplication of sets

elements $\{g_i\} = (g_{i_1}, g_{i_2}, \dots, g_{i_{m_j}})$, $i = 1, 2, \dots, n$, $j = 1, 2, \dots, k$ (See point

1, 2 in **Math Prself** [2])

2. parallel pattern recognition:

PrfSprt $\begin{matrix} IF\{q_1 \in image\ archive_1\} then \\ Name\ of\ q_1 \end{matrix} \quad IF\{q_2 \in image\ archive_2\} then \quad \dots \quad IF\{q_n \in image\ archive_n\} then \quad \begin{matrix} \\ Name\ of\ q_n \end{matrix}$

The example of PrfSprt-program is

PrfSprt $\begin{matrix} g_1 := & g_2 := & \dots & g_n := \\ p_1 & p_2 & \dots & p_n \end{matrix} \quad \begin{matrix} IF\{B_1 f_1\} then \\ w_1 \end{matrix} \quad \begin{matrix} IF\{B_2 f_2\} then \\ w_2 \dots \\ x_2 \dots \end{matrix} \quad \dots \quad \begin{matrix} IF\{B_n f_n\} then \\ w_n \end{matrix} \quad \dots \quad \begin{matrix} w_1 & w_2 & \dots & w_n \\ w_1 & w_2 & \dots & w_n \\ x_n \end{matrix} \cdot$

PrS_3f – software operators will differ only just because aggregates

$\{g\}, \{p\}, \{B\}, \{f\}$ will be formed from corresponding PrfSprt-program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4).

For example, $Sprt_{g\{R\}}^{\{R\}}$ is the capacity in itself of the second type if $g\{R\}$ is a program capable of generating $\{R\}$.

The example of self-program of the first type is $St_{\{St_x^{\{a(x)\}:\{p\}\}, St_x^{IF\{B\}\{f\}} then Q, St_Q^Q\}}^{\{St_x^{\{a(x)\}:\{p\}\}, St_x^{IF\{B\}\{f\}} then Q, St_Q^Q\}} \cdot$

PrfSprt $\begin{matrix} g_1 := & g_2 := & \dots & g_n := \\ p_1 & p_2 & \dots & p_n \end{matrix} \quad \begin{matrix} IF\{B_1 f_1\} then \\ w_1 \end{matrix} \quad \begin{matrix} IF\{B_2 f_2\} then \\ w_2 \dots \\ w_n \end{matrix} \quad \dots \quad \begin{matrix} w_1 & w_2 & \dots & w_n \\ w_1 & w_2 & \dots & w_n \\ w_1 & w_2 & \dots & w_n \end{matrix} \cdot$

PrfSprt-coding: 1) set A_i to set B_i , 2) set A_i to a point q_i , where the elements of the sets A_i, B_i can be continuous, ($i = 1, 2, \dots, n$; $j = 1, 2, \dots, m$). For example,

PrfSprt $\begin{matrix} A_1 & A_2 & \dots & A_n \\ B_1 & B_2 & \dots & B_n \end{matrix} \cdot$

There are PrfSprt -coding, PrfSprt-translation, PrfSprt-realize of preprograms and of the programs from the archives without extraction theirs

Self-coding: 1) set A_i to set A_i , i.e., A_i on itself 2) set A_i to a point q_i in forms (1.1) - (1.4), where the elements of the sets A_i can be continuous. For example,

PrfSprt $\begin{matrix} A_1 & A_2 & \dots & A_n \\ A_1 & A_2 & \dots & A_n \end{matrix} \cdot$

One of the central departments of the control system should be a computer system of the usual type of the desired level. In symbiosis with PrfSprt-Networks, it will provide a holistic operation of the control system in three modes: conventional serial through a conventional type computer system, direct parallel through PrfSprt -Networks and series-parallel. Codes from a conventional type computer system will be used via PrfSprt -connectors in PrfSprt - coding, for example:

$$\text{PrfSprt}(t) \begin{matrix} (\text{UHF AC})_1(t) := Q_1(t) \\ \text{activation} \end{matrix} \quad (\text{UHF AC})_2(t) := Q_2(t) \quad \dots \quad (\text{UHF AC})_n(t) := Q_n(t) \quad \begin{matrix} \text{activation} \end{matrix}.$$

UHF AC field activation is used.

Consider the dynamic PrfSprt and PrS₃f(t) programming:

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t), p_2(t), \dots, p_n(t))$ to the variables $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$ is

$$\text{implemented through PrfSprt} \begin{matrix} g_1(t) := & g_2(t) := & \dots & g_n(t) := \\ p_1(t) & p_2(t) & \dots & p_n(t) \end{matrix}.$$

2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f_2(t), \dots, f(t)_n)$ for a set of expressions $\{B(t)\} =$

$(B_1(t), B_2(t), \dots, B_n(t))$ is implemented through

$$\text{PrfSprt} \begin{matrix} IF\{B_1(t)f_1(t)\} then & IF\{B_2(t)f_2(t)\} then & \dots & IF\{B_n(t)f_n(t)\} then \\ x_1(t) & x_2(t) & \dots & x_n(t) \end{matrix} \text{wh}$$

ere $x(t) = (x_1(t), x_2(t), \dots, x_n(t))$. can be any.

3. Similarly for loop operators and others.

$PrS_3f(t)$ – software operators will differ only in that the aggregates

$\{g(t)\}, \{p(t)\}, \{B(t)\}, \{f(t)\}$ will be formed from corresponding processes

$\text{PrfSprt}(t)$ for the above-mentioned programming operators through form (1.1) or forms (1.1.1) – (1.4) for more complex operators.

Consider PrtSpr-program operators. The ideology of PrtSpr and Prt_{s_4f} is parallel analogue of t_{s_4f} [16] can be used for programminF. Here are some of the PrtSpr-program operators.

1. Simultaneous expelling assignment of the expressions $\{p\} = (p_1, p_2, \dots, p_n)$ from the variables $\{g\} = (g_1, g_2, \dots, g_n)$. It's implemented through $\begin{matrix} g_1 & g_2 & \dots & g_n \\ =: p_1 & =: p_2 & \dots & =: p_n \end{matrix} \text{PrfSprt.}$
2. Simultaneous expelling checks the set of conditions $\{f\} = (f_1, f_2, \dots, f_n)$ for a set of expressions $\{B\} = (B_1, B_2, \dots, B_n)$. It's implemented through $\text{PrfSprt} \begin{matrix} x_1 & x_2 & \dots & x_n \\ IF\{B_1 f_1\} \text{ then } & IF\{B_2 f_2\} \text{ then } & \dots & IF\{B_n f_n\} \text{ then } \end{matrix}$, where x_i ($i = 1, \dots, n$) can be anythinF.
3. Similarly for loop operators and others.

Prt_{S_4f} – software operators will differ only just because aggregates

$\{g\}, \{p\}, \{B\}, \{f\}$ will be formed from corresponding PrtSpr program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4).

Consider hierarchical PrtSpr-program operator

$$\begin{matrix} C_1 & C_2 & \dots & C_m \\ D_1 & D_2 & \dots & D_m \end{matrix} \text{PrfSprt} = \left\{ \begin{matrix} \sum_{i=1}^m Q_i + \begin{matrix} \{ \} & \{ \} & \dots & \{ \} \\ D_1 - D_1 \cap C_1 & D_2 - D_2 \cap C_2 & \dots & D_m - D_m \cap C_m \end{matrix} \text{PrfSprt} \\ \sum_{i=1}^m (C_i - D_i \cap C_i) - (D_i - D_i \cap C_i) \end{matrix} \right\}, \text{ where } Q_i$$

is oself-set for $(D_i \cap C_i)$ [11].

Consider the dynamic PrtSpr and $Prt(t)_{S_4f}$ programming at time t.

1. The process of simultaneous assignment of the expressions $\{p(t)\} = (p_1(t), p_2(t), \dots, p_n(t))$ to the variables $\{g(t)\} = (g_1(t), g_2(t), \dots, g_n(t))$. is implemented through $\text{PrfSprt} \begin{matrix} g_1(t) := & g_2(t) := & \dots & g_n(t) := \\ p_1(t) & p_2(t) & \dots & p_n(t) \end{matrix} \dots$
2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f(t)_2, \dots, f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), \dots, B_n(t))$ is implemented through

$$\text{PrfSrt} \begin{matrix} IF\{B_1(t)f_1(t)\} \text{ then} \\ x_1(t) \end{matrix} \quad IF\{B_2(t)f_2(t)\} \text{ then} \quad x_2(t) \quad \dots \quad IF\{B_n(t)f_n(t)\} \text{ then} \quad x_n(t)$$

where $x(t) = (x_1(t), x_2(t), \dots, x_n(t))$. can be any.

3. Similarly for loop operators and others.

$\text{Prt}(t)_{S_{4f}}$ – software operators will differ only in that the aggregates

$\{g(t)\}, \{p(t)\}, \{B(t)\}, \{f(t)\}$ will be formed from corresponding processes

$\text{PrfSrt}(t)$ for the above-mentioned programming operators through form (1.1) or forms (1.1.1) – (1.4) for more complex operators.

Consider hierarchical dynamic $\text{PrfSrt}(t)$ -program operator:

$$\begin{matrix} C_1(t) & C_2(t) & \dots & C_m(t) \\ D_1(t) & D_2(t) & \dots & D_m(t) \end{matrix} \text{PrfSrt}(t) = \left\{ \begin{matrix} \sum_{i=1}^m Q_i(t) + \begin{matrix} \{\} & \{\} & \dots & \{\} \\ D_1(t) - D_1(t) \cap C_1(t) & D_2(t) - D_2(t) \cap C_2(t) & \dots & D_m(t) - D_m(t) \cap C_m(t) \end{matrix} \\ \sum_{i=1}^m (C_i(t) - D_i(t) \cap C_i(t)) - (D_i(t) - D_i(t) \cap C_i(t)) \end{matrix} \text{PrfSrt} \right\},$$

where $Q_i(t)$ is oself-set for $(D_i(t) \cap C_i(t))$ [14].

Consider PrfSrt - program operators

$$fSprt_{t_0} \left\{ \begin{matrix} q(u & u & u \\ u & u & u \end{matrix} \text{PrfSpr}_u^u \right\} W_q fSprt_q^{Eq} \left(\begin{matrix} u & u & u \\ u & u & u \end{matrix} \text{PrfSpr}_u^u \right), fSprt_{d_r}^{\{Ex_l d_r\}} \left(\begin{matrix} u & u & u \\ u & u & u \end{matrix} \text{PrfSpr}_u^u \right) \} \text{—program structure}$$

example, where the assemblage point d_r is the cursor, it is quite complex self—program.

Remark. Energy of a living organism other than humans:

$$fPrf(r, u(E_q)) = fSprt_{t_0} \left\{ \begin{matrix} q(u & u & u \\ u & u & u \end{matrix} \text{PrfSpr}_u^u \right\} W_q fSprt_q^{Eq} \left(\begin{matrix} u & u & u \\ u & u & u \end{matrix} \text{PrfSpr}_u^u \right), fSprt_{d_r}^{\{Ex_l d_r\}} \left(\begin{matrix} u & u & u \\ u & u & u \end{matrix} \text{PrfSpr}_u^u \right) \} (**_{FF.2}).$$

Energy of a living organism of a person:

$$fPrhf(r, u(E_q)) = fSprt_{t_0} \left\{ \begin{matrix} q(u & u & u \\ u & u & u \end{matrix} \text{PrfSpr}_u^u \right\} W_q fSprt_q^{Eq} \left(\begin{matrix} u & u & u \\ u & u & u \end{matrix} \text{PrfSpr}_u^u \right), fSprt_{d_r}^{\{Ex_l d_r\}} \left(\begin{matrix} u & u & u \\ u & u & u \end{matrix} \text{PrfSpr}_u^u \right) \} (**_{FF.2}).$$

$\begin{matrix} u & u \\ u & u \end{matrix} \text{PrfSpr}_u^u$ - internal energy of a living organism of double energy

structure, q- a gap in the energy cocoon of a living organism, r-the position of the assemblage point d_r on the energy cocoon of a living organism, W_q - energy

prominences from the gap in the cocoon of a living organism, E_q -external energy entering the gap in the cocoon of a living organism, $E^{ex}l^{dr}$ - a bundle of fibers of external energy self-capacities from outside the cocoon, collected at the point of assembly of the cocoon of a living organism, $E_{in}l^{dr}$ - a bundle of fibers of external energy self-capacities from inside the cocoon, collected at the point of assembly of the cocoon of a living organism in the same position r of the assemblage point d_r . d_r is the subject of identifying the energy fibers of the subtle energy of the Universe in position r both outside and inside the cocoon.

$(**_{FF,2})$, $(***_{FF,2})$ can be interpreted as PrfSrt- program operators.

4.10 Program operators PrffSprt, PrtffSpr, S^1e , Set₁

Threshold element in networks PrffSprt -

$$\begin{array}{ccccccc} B_1 & B_2 & \dots & B_n & \{ax\}_1 & \{ax\}_2 & \dots \{ax\}_n \\ \mu_{12} & \mu_{22} & \dots & \mu_{n2} & \text{PrffSprt} & \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ \{qy\}_1 & \{qy\}_2 & \dots & \{qy\}_n & B_1 & B_2 & \dots & B_n \end{array}, B_1, B_2, \dots, B_n - \text{artificial}$$

neurons of type PrffSprt (designation - $mn\text{PrffSprt}$) , $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1)$,

$x_2|\mu_{\tilde{x}}(x_2)|, \dots, x_n|\mu_{\tilde{x}}(x_n)|$) are the fuzzy values of the initial signals, $a=(a_1, a_2, \dots, a_n)$ are

the weights of PrffSprt-synapses and $\tilde{y}=(y_1|\mu_{\tilde{y}}(y_1)$, $y_2|\mu_{\tilde{y}}(y_2)|, \dots, y_n|\mu_{\tilde{y}}(y_n)|$) are the

fuzzy values of the output signals $\{qy\}$ with weights $q=(q_1, q_2, \dots, q_n)$.

1. Similarly, for simultaneous parallel multiplication:

$$\text{PrffSprt} \begin{array}{cccc} \{g_1\} * & \{g_2\} * & \dots & \{g_n\} * \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ w_1 & w_2 & \dots & w_n \end{array}, \text{ the notation of the fuzzy set } B_l, l=1, 2, \dots, n,$$

with elements $b_{l i_1 i_2 \dots i_{m_j}} =$

$$\text{Sprt}_{x_l} \left\{ g_{l i_1}^*, g_{l i_2}^*, \dots, g_{l i_{m_j}}^* \right\}_{R_l} \text{ for any } \{l_{i_1}, l_{i_2}, \dots, l_{i_{m_j}}\} \text{ without repetitions, } x_l =$$

$$\text{Sprt}_w^{\{K_l\}}, K_l\text{-set of any } \{k_{l i_1}^*, k_{l i_2}^*, \dots, k_{l i_{m_j}}^*\} \text{ without repeating them, } l$$

$$= 1, 2, \dots, n, k_{l i_j}\text{-any digit, } i=1, 2, \dots, m_j, R_l = \text{Sprt}_w^{\{l_{i_1} + l_{i_2} + \dots, l_{i_{m_j}}\}}, R_l \text{ is the}$$

index of the lower discharge (we choose an index on the scale of discharges): see Table I.1.

Then $\text{PrffSprt} \begin{matrix} \{B_1\} + & \{B_2\} + & \dots & \{B_n\} + \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ x_1 & x_2 & \dots & x_n \end{matrix}$ gives the final result of simultaneous

multiplication. Any system of calculus can be chosen, in particular binary. The most straightforward functional scheme of the assumed arithmetic-logical device for PrffSprt-multiplication:

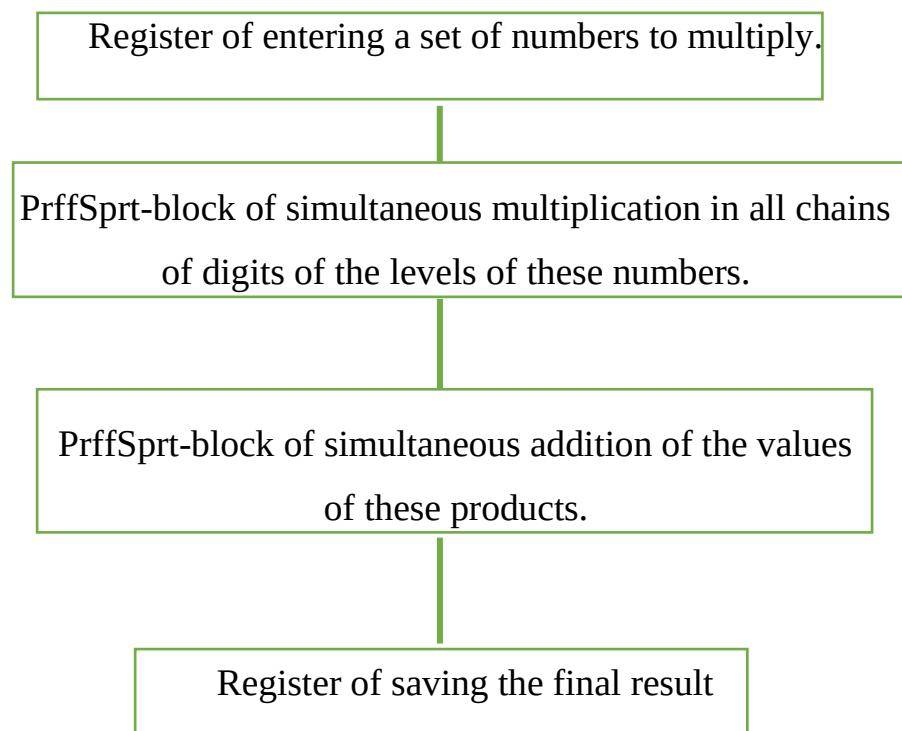


Fig. 4.9. The straightforward functional scheme of the assumed arithmetic-logical device for PrffSprt-multiplication.

Remark. The algorithm for simultaneously adding a set of numbers can also be implemented as the simultaneous addition of elements of a simultaneously formed composite matrix: a triangular matrix in which the elements of the first row are represented by multiplying the first number from the set by the rest: each multiplication is represented by a matrix of multiplying the digits of 2 numbers, taking into account the bit depth, the elements of the second rows are

represented by multiplying the second number from the set by the ones following it, etc.

Here are some of the PrffSprt programming operators.

1. Simultaneous fuzzy assignment of the expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots, x_n|\mu_{\tilde{x}}(x_n))$. This is

$$\begin{array}{c} x_1 := x_2 := \dots x_n := \\ \text{implemented via PrffSprt} \begin{array}{ccc} \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ p_1 & p_2 & \dots p_n \end{array} . \end{array}$$

2. Simultaneous fuzzy checking the set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$. Implemented via

$$\begin{array}{c} \text{PrffSprt} \quad IF\{B_1g_1\} \text{ then } IF\{B_2g_2\} \text{ then } \dots IF\{B_ng_n\} \text{ then} \\ \mu_{11} \quad \mu_{21} \quad \dots \mu_{n1} \\ w_1 \quad w_2 \quad \dots w_n \end{array} , \text{ where } w_i (i = 1, \dots, n) \text{ can be anything.}$$

3. Similarly for loop operators and others.

PrffS₃f – software operators will differ only in that the aggregates

$\{\tilde{g}\}, \{\tilde{p}\}, \{\tilde{B}\}, \{\tilde{x}\}$ will be formed from the corresponding PrffSprt program operators in form (1.1) and for more complex operators in the forms (1.1.1) – (1.4), (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

The OS (operating system), the computer's principles, and the modes of operation for this programming are interesting. But this is already the material for the following articles.

The ideology of fuzzy dynamic PrffSprt and PrffS₃f(t) can be used for programming:

1. The process of simultaneous assignment of the expressions $\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \dots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ is

$$\begin{array}{c} x_1(t) := x_2(t) := \dots x_n(t) := \\ \text{implemented through PrffSprt(t)} \begin{array}{ccc} \mu_{11} & \mu_{21} & \dots \mu_{n1} \\ p_1(t) & p_2(t) & \dots p_n(t) \end{array} . \end{array}$$

2. The process of simultaneous check the set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), ..., g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for a set of expressions $\widetilde{B(t)}=(B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), ..., B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through

$$\begin{array}{ccccccc} IF\{B_1(t)g_1(t)\} \text{ then} & IF\{B_2(t)g_2(t)\} \text{ then} & ... & IF\{B_n(t)g_n(t)\} \text{ then} \\ \text{PrffSprt}(t) & \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ & w_1(t) & w_2(t) & \dots & w_n(t) \end{array}$$

here $w(t) = (w_1(t), w_2(t), \dots, w_n(t))$. can be any.

3. Similarly for loop operators and others.

$PrffS_3f(t)$ – software operators will differ only in that the aggregates

$\{\widetilde{g(t)}\}, \{\widetilde{p(t)}\}, \{\widetilde{B(t)}\}, \{\widetilde{x(t)}\}$ will be formed from corresponding processes

$\text{PrffSprt}(t)$ for the above-mentioned programming operators through form (1.1) or forms (1.1.1) - (1.4), (2.1*) and analogs of forms (1.1.1) – (1.4) by type (2.1*) for more complex operators.

The usage of PrffSprt -elements for networks.

A. Galushkin's comprehensive monograph [3] covers all aspects of networks, but traditional approaches go through classical mathematics, mainly through the usual correspondence operators. Here we consider a different approach - through a new mathematical process with parallel fuzzy containment operators, which, although they can be interpreted as the result of some correspondence operators, are not themselves correspondence operators. Parallel fuzzy containment operators are more convenient for networks. Also, the main emphasis was placed on using processors operating using triodes, which are generally not used in Sprt -networks. PrffSprt -networks (SmnPrffSprt) are a PrffSprt -structure that can be built for the required weights. PrffSprt-OS (PrffSprt operating system) uses PrffSprt -coding and PrffSprt -translation. In the first one, coding is carried out through a 2-dimensional matrix-row (a, b) , where the number b is the code of the action, and the number a is the code of the object of this action. PrffSprt -coding (or PrffSelf-coding) is implemented through a matrix consisting of 2 columns (in the continuous case, two intervals of numbers). Here, the source encoding is used for

leaving the network ideology, but these networks are a complex hierarchy of different levels, like living organisms. Threshold element of extended type PrffSprt

$$- \begin{array}{ccccccc} B_1 & B_2 & \dots & B_n & \{ax\}_1 & \{ax\}_2 & \dots \{ax\}_n \\ \mu_{12} & \mu_{22} & \dots & \mu_{n2} & \text{PrffSprt} & \mu_{11} & \mu_{21} \dots \mu_{n1} \\ \{qy\}_1 & \{qy\}_2 & \dots & \{qy\}_n & B_1 & B_2 & \dots B_n \end{array}, B_1, B_2, \dots, B_n - \text{artificial}$$

neurons of type PrffSprt (designation - mnPrffSprt) , $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1),$

$x_2|\mu_{\tilde{x}}(x_2)|, \dots, x_n|\mu_{\tilde{x}}(x_n)|)$ are the fuzzy values of the initial signals, $a=(a_1, a_2, \dots, a_n)$ are the weights of PrffSprt-synapses and $\tilde{y}=(y_1|\mu_{\tilde{y}}(y_1), y_2|\mu_{\tilde{y}}(y_2)|, \dots, y_n|\mu_{\tilde{y}}(y_n)|)$ are the fuzzy values of the output signals $\{qy\}$ with weights $q=(q_1, q_2, \dots, q_n)$.

Remark 4.10.0. A neural network can be thought of as a learnable parallel fuzzy dynamic operator.

Remark 4.10.1. Traditional scientific approaches through classical mathematics make it possible to describe only at the usual energy level. Here we consider an approach that makes describing processes with finer energies possible. mnPrffSprt

$$\begin{array}{ccccccc} \text{ffeprogram}_1(t) & \text{ffeprogram}_2(t) & \dots & \text{ffeprogram}_n(t) \\ \text{contains PrffSprt}(t) & \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ \text{mnPrffSprt} & \text{mnPrffSprt} & \dots & \text{mnPrffSprt} \end{array}$$

ffeprogram –executing program in PrffSprt- OS. PrffSprt-OS (or Prffself-OS) is based on PrffSprt-assembly language (or Prself-assembly language), which is based on assembly language through PrffSprt-approach in turn, if the base of elements of PrffSprt-networks is sufficient. The ffeprograms are in PrffSprt-programming environments (or PrffSelf-programming environments), but this question and PrffSprt-networks base will be considered in the following monographs. In particular, ffeprograms may contain PrffSprt- programming operators. In mnPrffSprt cores, the constant memory PrffSprt with correspondent ffeprograms depending on mnPrffSprt.

The OS (operating system) and the principles and modes of operation of the PrffSprt-networks for this programming are interesting. But this is already the material for the next publications.

Here is developed a helicopter model without a main and tail rotors based on PrffSprt – physics and special neural networks with artificial neurons operating in normal and PrffSprt-modes. Let's denote this model through SmnPrffSprt. To do this, it's proposed to use mnPrffSprt of different levels; for example, for the usual mode, mnPrffSprt serves for the initial processing of signals and the transfer of information to the second level, etc., to the nodal center, then checked. In case of an anomaly - local PrffSprt-mode with the desired "target weight" is realized in this section, etc., to the center. In the case of a monster during the test, SmnPrffSprt is activated with the desired "target weight." Here are realized other tasks also. To reach the fself-energy level, the mode $Sprt_{SmnPrffSprt}^{SmnPrffSprt}$ is used. In normal mode, it's planned to carry out the movement of SmnPrffSprt on jet propulsion by converting the energy of the emitted gases into a vortex to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the SmnPrffSprt for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is drainage of exhaust gases outside the SmnPrffSprt. SmnPrffSprt is represented by a neural network that extends from the center of one of the main clusters of PrffSprt - artificial neurons to the shell, turning into the body itself. Above the operator's cabin is the central core of the neural network and the target block, responsible for performing the "target weights" and auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for SmnPrffSprt's actions is below the operator's cab. In PrffSprt – mode, the entire network or its sections are PrffSprt – activated to perform specific tasks, in particular, with "target weights." In the target, block used PrffSprt -coding, PrffSprt -translation for activation of all networks to "target weights" simultaneously, then –the reset of this PrffSprt-coding after activation. Unfortunately, triodes are not suitable for PrffSprt -neural networks. In the most primitive case, usual separators with corresponding resistances and cores for ffeoprograms may be used instead triodes since there is no necessity to unbend the

alternating current to direct. The PrffSprt-operative memory belt is disposed around a central core of SmnPrffSprt. There are PrffSprt-coding, PrffSprt-translation, and PrffSprt-realize of eprograms and the programs from the archives without extraction, PrffSprt-coding and PrffSprt-translation may be used in high-intensity, ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna, Strickland. PrffSprt – structure or an eprogram if one is present of needed «target weight» are taken in target block at PrffSprt – activation of the networks. $Sprt_{activation}^{SmnPrffSprt,f}$ derives SmnPrffSprt to the self-level boundary with target weight f.

It's used an alternating current of above high frequency and ultra-violet light, which can work with PrffSprt – structures in PrffSprt-modes by its nature to activate the networks or some of its parts in PrffSprt-modes and locally using PrffSprt-mode. Above high frequently alternating current go through mercury bearers. That's why overheating does not occur. The power of the alternating current above high frequently increases considerably for the target block. The activation of all PrffSprt-networks is realized to indicate “target weights.”

Program operators PrffSprt, PrfftSpr.

Here it is supposed to use a symbiosis of parallel fuzzy actions and conventional calculations through sequential actions. This must be done through PrffSprt-Networks in one of the central departments of which a conventional computer system is located. The parallel processor is itself prffeprogram with direct parallel computing not through serial computing.

Using conventional PrffSprt -coding by a parallel computer system, through a Target-block with a PrffSprt -program operator -

$$\begin{matrix} g_1(t)w_1(t) & g_2(t)w_2(t) & ...g_n(t)w_n(t) \\ \text{PrffSprt}(t) & \mu_{11} & \mu_{21} & ... & \mu_{n1} \\ \text{activation} & \text{activation} & ... & \text{activation} \end{matrix}, \text{ it will be possible to obtain}$$

the execution of the parallel fuzzy actions $(g_1(t), g_2(t), \dots, g_n(t))$ with the desired target weights $w(t)=(w_1(t), w_2(t), \dots, w_n(t))$. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of

current (or voltage). For PrffSprt-coding and PrffSprt-translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical pulses laser of Nobel laureates 2018-year Gerard Mourou, Donna Strickland, or a combination of them. For the desired fuzzy action, for example, using the direct parallel fuzzy program of operator

$$\text{PrffSprt}(t) \begin{matrix} (\text{UHF AC})_1(t) := Q_1(t) \\ \mu_{11} \\ \text{activation} \end{matrix} \quad (\text{UHF AC})_2(t) := Q_2(t) \quad \begin{matrix} \mu_{21} \\ \text{activation} \end{matrix} \quad \dots \quad (\text{UHF AC})_n(t) := Q_n(t) \quad \begin{matrix} \mu_{n1} \\ \text{activation} \end{matrix}, \text{ we}$$

simultaneously enter the desired fuzzy set of codes $Q_i(t)$, $i = 1, 2, \dots, n$, using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects.

Consider the types of direct parallel program operators:

- 1) PrffSprt-program operators
- 2) PrfftSpr-program operators

Here are some of the PrffSprt-program operators:

- 1) Simultaneous fuzzy assignment of the fuzzy expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), \dots, p_n|\mu_{\tilde{p}}(p_n))$ to the variables $\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), \dots,$

$$x_n|\mu_{\tilde{x}}(x_n)). \text{ This is implemented via PrffSprt } \begin{matrix} x_1 := & x_2 := & \dots & x_n := \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ p_1 & p_2 & \dots & p_n \end{matrix}.$$

- 2) Simultaneous fuzzy checking the fuzzy set of conditions $\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), \dots, g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions $\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), \dots, B_n|\mu_{\tilde{B}}(B_n))$. Implemented via

$$\text{PrffSprt } \begin{matrix} IF\{B_1 g_1\} \text{ then} & IF\{B_2 g_2\} \text{ then} & \dots & IF\{B_n g_n\} \text{ then} \\ \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\ w_1 & w_2 & \dots & w_n \end{matrix}, \text{ where } w_i (i$$

$= 1, \dots, n)$ can be anything.

- 3) Similarly for loop operators and others.

PrffSprt-algorithm Examples:

- 1) Simultaneous addition and simultaneous parallel fuzzy multiplication of fuzzy sets elements (See point 1, 2 in **Math Prffself** [2])
- 2) parallel fuzzy pattern recognition:

$$\begin{array}{ccccccc}
 IF\{q_1 \in image\ archive_1\} \text{ then} & IF\{q_2 \in image\ archive_2\} \text{ then} & \dots & IF\{q_n \in image\ archive_n\} \text{ then} \\
 \text{PrffSprt} & \mu_{11} & \mu_{21} & \dots & \mu_{n1} \\
 & \text{Name of } q_1 & \text{Name of } q_2 & \dots & \text{Name of } q_n
 \end{array}$$

The example of PrffSprt-program is

$$\begin{array}{ccccccc}
 x_1 := & x_2 := & \dots x_n := & IF\{B_1 g_1\} \text{ then} & IF\{B_2 g_2\} \text{ then} & \dots IF\{B_n g_n\} \text{ then} & w_1 \quad w_2 \quad \dots w_n \\
 \text{PrffSprt } \mu_{11} & \mu_{21} & \dots \mu_{n1} & \text{PrffSprt } \mu_{11} & \mu_{21} & \dots \mu_{n1} & \dots \text{PrffSprt } \mu_{11} \quad \mu_{21} \quad \dots \mu_{n1}, \\
 \text{PrffSprt } p_1 & p_2 & \dots p_n & w_1 & w_2 & \dots w_n & w_1 \quad w_2 \quad \dots w_n \\
 & \mu_{11} & & & \mu_{21} & & \dots \quad \mu_{n1} \\
 & r_1 & & & r_2 & & \dots \quad r_n
 \end{array}$$

PrffS₃f – software operators will differ only just because fuzzy aggregates $\{\tilde{g}\}, \{\tilde{p}\}, \{\tilde{B}\}, \{\tilde{x}\}$ will be formed from corresponding PrffSprt-program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4) , (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

For example, $fSprt_{g\{R\}}^{\{R\}}$ is the fcapacity in itself of the second type if $g\{R\}$ is a program capable of generating $\{R\}$.

The example of self-program of the first type is

$$\begin{array}{ccccccc}
 x_1 := & x_2 := & \dots x_n := & IF\{B_1 g_1\} \text{ then} & IF\{B_2 g_2\} \text{ then} & \dots IF\{B_n g_n\} \text{ then} & w_1 \quad w_2 \quad \dots w_n \\
 \text{PrffSprt } \mu_{11} & \mu_{21} & \dots \mu_{n1} & \text{PrffSprt } \mu_{11} & \mu_{21} & \dots \mu_{n1} & \dots \text{PrffSprt } \mu_{11} \quad \mu_{21} \quad \dots \mu_{n1}, \\
 p_1 & p_2 & \dots p_n & w_1 & w_2 & \dots w_n & w_1 \quad w_2 \quad \dots w_n \\
 \text{PrffSprt } \mu_{11} & & & & \mu_{21} & & \dots \quad \mu_{n1} \\
 x_1 := & x_2 := & \dots x_n := & IF\{B_1 g_1\} \text{ then} & IF\{B_2 g_2\} \text{ then} & \dots IF\{B_n g_n\} \text{ then} & w_1 \quad w_2 \quad \dots w_n \\
 \text{PrffSprt } \mu_{11} & \mu_{21} & \dots \mu_{n1} & \text{PrffSprt } \mu_{11} & \mu_{21} & \dots \mu_{n1} & \dots \text{PrffSprt } \mu_{11} \quad \mu_{21} \quad \dots \mu_{n1}, \\
 p_1 & p_2 & \dots p_n & w_1 & w_2 & \dots w_n & w_1 \quad w_2 \quad \dots w_n
 \end{array}$$

PrffSprt-coding: 1) fuzzy set A_i to fuzzy set B_i , 2) fuzzy set A_i to a point q_i , where the elements of the fuzzy sets A_i, B_i can be continuous, ($i = 1, 2, \dots, n; j = 1, 2, \dots,$

$$\begin{array}{ccc}
 A_1 & A_2 & \dots A_n \\
 \text{m). For example, PrffSprt } \mu_{11} & \mu_{21} & \dots \mu_{n1}. \\
 B_1 & B_2 & \dots B_n
 \end{array}$$

There are PrffSprt -coding, PrffSprt-translation, PrffSprt-realize of prffeprograms and of the programs from the archives without extraction theirs

Self-coding: 1) fuzzy set A_i to fuzzy set A_i , i.e. A_i on itself 2) fuzzy set A_i to a point q_i in forms (1.1) - (1.4), where the elements of the fuzzy sets A_i can be

$$\begin{array}{ccc}
 A_1 & A_2 & \dots A_n \\
 \text{continuous. For example, PrffSprt } \mu_{11} & \mu_{21} & \dots \mu_{n1}. \\
 A_1 & A_2 & \dots A_n
 \end{array}$$

One of the central departments of the control system should be a computer system of the usual type of the desired level. In symbiosis with PrffSprt-Networks, it will provide a holistic operation of the control system in three modes: conventional serial through a conventional type computer system, direct parallel through PrffSprt -Networks and series-parallel. Codes from a conventional type computer system will be used via PrffSprt -connectors in PrffSprt - coding, for example:

$$\begin{array}{ccccccc} \text{PrffSprt}(t) & (\text{UHF AC})_1(t) := Q_1(t) & (\text{UHF AC})_2(t) := Q_2(t) & \dots & (\text{UHF AC})_n(t) := Q_n(t) & & \\ & \mu_{11} & \mu_{21} & \dots & \mu_{n1} & & \\ & \text{activation} & \text{activation} & \dots & \text{activation} & & \end{array}, \text{UHF}$$

AC field activation is used.

Consider the fuzzy dynamic PrffSprt and PrffS₃f(t) programming:

1. The process of simultaneous fuzzy assignment of the fuzzy

expressions $\widetilde{p(t)} = (p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), \dots, p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)} = (x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), \dots, x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ is

$$\begin{array}{ccccccc} & x_1(t) := & x_2(t) := & \dots & x_n(t) := & & \\ \text{implemented through PrffSprt} & \mu_{11} & \mu_{21} & \dots & \mu_{n1} & \dots & \\ & p_1(t) & p_2(t) & \dots & p_n(t) & & \end{array}$$

2. The process of simultaneous check the fuzzy set of conditions

$\widetilde{g(t)} = (g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), \dots, g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions $\widetilde{B(t)} = (B_1(t)|\mu_{\widetilde{B(t)}}(B_1(t)), B_2(t)|\mu_{\widetilde{B(t)}}(B_2(t)), \dots, B_n(t)|\mu_{\widetilde{B(t)}}(B_n(t)))$ is implemented through

$$\begin{array}{ccccccc} \text{PrffSprt} & IF\{B_1(t)g_1(t)\} \text{ then} & IF\{B_2(t)g_2(t)\} \text{ then} & \dots & IF\{B_n(t)g_n(t)\} \text{ then} & & \text{where} \\ & \mu_{11} & \mu_{21} & \dots & \mu_{n1} & & \\ & w_1(t) & w_2(t) & \dots & w_n(t) & & \end{array}$$

$w(t) = (w_1(t), w_2(t), \dots, w_n(t))$. can be any.

3. Similarly for loop operators and others.

PrffS₃f(t)– software operators will differ only in that the aggregates

$\{\widetilde{g(t)}\}, \{\widetilde{p(t)}\}, \{\widetilde{B(t)}\}, \{\widetilde{x(t)}\}$ will be formed from corresponding processes

PrffSprt(t) for the above-mentioned programming operators through form (1.1) or

forms (1.1.1) – (1.4) for more complex operators, (2.1*) and analogs of forms

(1.1.1) - (1.4) by type (2.1*).

Consider PrfftSpr-program operators. The ideology of PrfftSpr and $Prfft_{s_{4f}}$ is parallel analogue of $ft_{s_{4f}}$ [11] can be used for programminF. Here are some of the PrfftSpr -program operators.

1. Simultaneous fuzzy expelling assignment of the fuzzy

expressions $\tilde{p}=(p_1|\mu_{\tilde{p}}(p_1), p_2|\mu_{\tilde{p}}(p_2), ..., p_n|\mu_{\tilde{p}}(p_n))$ from the variables

$\tilde{x}=(x_1|\mu_{\tilde{x}}(x_1), x_2|\mu_{\tilde{x}}(x_2), ..., x_n|\mu_{\tilde{x}}(x_n))$. It's implemented

$$\begin{array}{ccccccc} x_1 & := & x_2 & := & ... & x_n & := \\ \text{through} & \mu_{11} & \mu_{21} & ... & \mu_{n1} & \text{PrffSprt.} & \\ p_1 & & p_2 & ... & p_n & & \end{array}$$

2. Simultaneous fuzzy expelling checks the fuzzy set of conditions

$\tilde{g}=(g_1|\mu_{\tilde{g}}(g_1), g_2|\mu_{\tilde{g}}(g_2), ..., g_n|\mu_{\tilde{g}}(g_n))$ for the fuzzy set of expressions

$\tilde{B}=(B_1|\mu_{\tilde{B}}(B_1), B_2|\mu_{\tilde{B}}(B_2), ..., B_n|\mu_{\tilde{B}}(B_n))$. It's implemented through

$$\begin{array}{ccccccc} IF\{B_1g_1\} & \text{then} & IF\{B_2g_2\} & \text{then} & ... & IF\{B_ng_n\} & \text{then} \\ \mu_{11} & & \mu_{21} & & ... & \mu_{n1} & \\ w_1 & & w_2 & & ... & w_n & \end{array} \text{PrffSprt where } w_i \text{ (i = 1,$$

..., n) can be anything.

3. Similarly for loop operators and others.

$Prfft_{s_{4f}}$ – software operators will differ only just because aggregates

$\{\tilde{g}\}, \{\tilde{p}\}, \{\tilde{B}\}, \{\tilde{x}\}$ will be formed from corresponding PrfftSpr program operators in form (1.1) for more complex operators in forms (1.1.1) – (1.4) , (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

Consider hierarchical PrfftSpr-program operator

$$\begin{array}{ccccccc} C_1(t) & C_2(t) & ... & C_m(t) & & & \\ \mu_{12} & \mu_{22} & ... & \mu_{m2} & \text{PrffSrt}_2(t) = & & \\ D_1(t) & D_2(t) & ... & D_m(t) & & & \end{array}$$

$$\left\{ \begin{array}{ccccccc} \sum_{i=1}^m Q_i(t) + & \{ \} & \{ \} & ... & \{ \} & & \\ & \mu_{12} & \mu_{22} & ... & \mu_{m2} & \text{PrffSrt}(t) & \\ & D_1(t) - D_1(t) \cap C_1(t) & D_2(t) - D_2(t) \cap C_2(t) & ... & D_m(t) - D_m(t) \cap C_m(t) & & \\ & \sum_{i=1}^m (C_i(t) - D_i(t) \cap C_i(t)) - (D_i(t) - D_i(t) \cap C_i(t)) & & & & & \end{array} \right\},$$

where $Q_i(t)$ is oself-(fuzzy set) for $(D_i(t) \cap C_i(t))$ [14].

Consider the PrtffSpr(t) and $Prfft(t)_{s_{4f}}$ programming at time t.

$\widetilde{p(t)}=(p_1(t)|\mu_{\widetilde{p(t)}}(p_1(t)), p_2(t)|\mu_{\widetilde{p(t)}}(p_2(t)), ..., p_n(t)|\mu_{\widetilde{p(t)}}(p_n(t)))$ to the variables $\widetilde{x(t)}=(x_1(t)|\mu_{\widetilde{x(t)}}(x_1(t)), x_2(t)|\mu_{\widetilde{x(t)}}(x_2(t)), ..., x_n(t)|\mu_{\widetilde{x(t)}}(x_n(t)))$ is implemented

2. The process of simultaneous check the set of conditions $\widetilde{g(t)}=(g_1(t)|\mu_{\widetilde{g(t)}}(g_1(t)), g_2(t)|\mu_{\widetilde{g(t)}}(g_2(t)), ..., g_n(t)|\mu_{\widetilde{g(t)}}(g_n(t)))$ for the fuzzy set of expressions

$$IF\{B_1(t)g_1(t)\} \text{ then } IF\{B_2(t)g_2(t)\} \text{ then } \dots IF\{B_n(t)g_n(t)\} \text{ then } \text{PrffSprt}(t),$$

$$\begin{array}{ccccccc} \mu_{11} & & \mu_{21} & & \dots & & \mu_{n1} \\ w_1(t) & & w_2(t) & & \dots & & w_n(t) \end{array}$$

3. Similarly for loop operators and others.

$\overline{x(t)}, \overline{p(t)}, \overline{B(t)}, \overline{g(t)}$ will be formed from corresponding processes $\text{PrffSprt}(t)$ for the above-mentioned programming operators through form (1.1) or forms (1.1.1) – (1.4) for more complex operators, (2.1*) and analogs of forms (1.1.1) - (1.4) by type (2.1*).

$$fSprt_{t_0} \left\{ \begin{matrix} q \left(\begin{matrix} u & u & u & u \\ \mu_1 & \mu_2 PrffSpr \mu_1 & \mu_2 \\ u & u & u & u \end{matrix} \right) \\ W_q fSprt_q^{Eq} \left(\begin{matrix} u & u & u & u \\ \mu_1 & \mu_2 PrffSpr \mu_1 & \mu_2 \\ u & u & u & u \end{matrix} \right), fSprt_{dr}^{\{Exl dr\}} \end{matrix} \right\}_{(Einl dr)} . \text{---program}$$

Remark. Energy with measure of fuzziness μ_1, μ_2 of a living organism other than humans:

fPrff(r, u(E_q), μ₁, μ₂)

$$= fSprt_{t_0} \left\{ \begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix} PrffSpr \begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix} \right\}_{W_q} Sprt_q^{E_q} \left(\begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix} PrffSpr \begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix} \right), fSprt_{dr}^{\{E^{ex}l^{dr}\}}_{(E^{in}l^{dr})} \} (**_{F.3}).$$

Energy with measure of fuzziness μ₁, μ₂ of a living organism of a person:

fPrhff(r, u(E_q), μ₁, μ₂)

$$= fSprt_{t_0} \left\{ \begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix} PrffSpr \begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix} \right\}_{W_q} Sprt_q^{E_q} \left(\begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix} PrffSpr \begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix} \right), fSprt_{dr}^{\{E^{ex}l^{dr}\}}_{(self(E^{in}l^{dr}))} \} (***_F.3).$$

$\begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix} PrffSpr \begin{pmatrix} u & u \\ \mu_1 & \mu_2 \end{pmatrix}$ -internal energy with measure of fuzziness μ₁, μ₂ of a

living organism of double energy structure, q- a gap in the energy cocoon of a living organism, r-the position of the assemblage point d_r on the energy cocoon of a living organism, W_q- energy prominences from the gap in the cocoon of a living organism, E_q-external energy entering the gap in the cocoon of a living organism, E^{ex}l^{dr} - a bundle of fibers of external energy self-capacities from outside the cocoon, collected at the point of assembly of the cocoon of a living organism, Eⁱⁿl^{dr}- a bundle of fibers of external energy self-capacities from inside the cocoon, collected at the point of assembly of the cocoon of a living organism in the same position r of the assemblage point d_r. d_r is the subject of identifying the energy fibers of the subtle energy of the Universe in position r both outside and inside the cocoon.

(**_{F.3}), (***_F.3) can be interpreted as PrfSrt- program operators.

4.11 Introduction to FUZZY PROGRAM OPERATORS Flprt, tprFL, FL¹epr, FLeprt₁

Here it is supposed to use a symbiosis of parallel actions and conventional calculations through sequential actions. This must be done through FLprt-Networks - fuzzy analogue of Sit-Networks [10] in one of the central departments of which a conventional computer system is located. The parallel processor is itself

fdprogram - fuzzy analogue of eprogram [10] with direct parallel computing not through serial computing.

Using conventional coding by a computer system, through a Target-block with a

$$\begin{array}{ccc} & \hat{A} & \\ & \uparrow & \\ \text{fuzzy Lprt -program operator - FLprt} & \bar{Q} & , \text{ where fuzzy A with measure} \\ & \uparrow & \\ & \text{activation } g & \end{array}$$

of fuzziness μ_A fuzzy acts Q with measure of fuzziness μ_Q to fuzzy *activation* with measure of fuzziness $\mu_{activation}$, Q is any fuzzy *action*, it will be possible to obtain the fuzzy execution with measure of fuzziness $\mu_{activation}$ of a parallel fuzzy action A with the desired target weight g or the execution with measure of fuzziness $\mu_{activation}$ of a parallel action A with the desired fuzzy target weight g with measure of fuzziness μ_g or both. Each code for a neural network from a conventional computer we "bind" (match) to the corresponding value of current (or voltage). For FLprt-coding and FLprt-translation may be use alternating current of ultrahigh frequency or high-intensity ultra-short optical pulses laser of Nobel laureates of 2018 year Gerard Mourou, Donna Strickland, or a combination of them. For the desired action, for example, using the direct parallel fdprogram of

$$\begin{array}{ccc} \{UHF AC := R\} & & \\ \text{operator fDprt} & \text{action } Q & \text{with the specified measures of fuzziness, we} \\ & \text{activation} & \end{array}$$

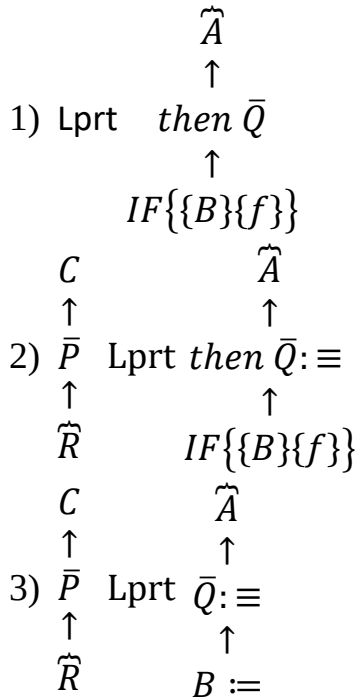
simultaneously enter the desired fuzzy set of codes R with measure of fuzziness μ_R using a microwave current or high-intensity ultra-short optical pulses laser in Target-block.

In a conventional computer, the process of sequential calculation takes a certain time interval, in a directly parallel calculation by a neural network, the calculation is instantaneous, but it occupies a certain region of the space of calculation objects. Consider the types of direct parallel fuzzy fdprogram operators:

1. fuzzy Lprt-program operators (designation FLprt-program operators)
2. fuzzy tprL-program operators (designation tprFL-program operators)

3. fuzzy $L^1\text{ep}_r$ - program operators (designation $FL^1\text{ep}_r$ -program operators)
4. fuzzy $L\text{ep}_{rt_1}$ - program operators (designation $FL\text{ep}_{rt_1}$ -program operators)

Examples:



etc.

4.12 Other PROGRAM OPERATORS

The program operators **SCprt** see in **Part I**. The program operators **SfCprt** see in **Part II**. The program operators **Wprt** see in **Part III**.

4.13 Some aspects of directly parallel operation of neural networks and their subtle energy

We consider $\Delta\text{SprtA} = \text{Sprt}_{\text{activation with } tW}^{\{A_1, A_2, \dots, A_n\}} - \{A_1, A_2, \dots, A_n\}$, $v_{\text{SprtA}} = \frac{\Delta\text{SprtA}}{\Delta\tau}$,

$$a_{\text{SprtA}} = \frac{d^2\text{SprtA}}{d\tau^2}$$

tW - the corresponding target weight [2], [10].

$\text{Sprt}_{\text{activation with } tW}^{\{A_1, A_2, \dots, A_n\}}$ will be called the current of $\{A_1, A_2, \dots, A_n\}$.

In particular, if $\{A_1, A_2, \dots, A_n\}$ – electrons, will be called the current of electrons, if $\{A_1, A_2, \dots, A_n\}$ – neurons actions, will be called the current of neurons actions

[2], [10]. Let's consider an analogue of Maxwell's equations for networks operating on electromagnetic energy:

$$\text{rot}(E_{\text{SprtA}}^{\text{tw}}) = -(1/q_0) \partial(B_{\text{SprtA}}^{\text{tw}}) / \partial t, (*_H)$$

$$\text{rot}(H_{\text{SprtA}}^{\text{tw}}) = j/q_0 + (1/q_0) \partial(D_{\text{SprtA}}^{\text{tw}}) / \partial t, (**_H)$$

$E_{\text{SprtA}}^{\text{tw}}$ - activation tension with target weight tw, $B_{\text{SprtA}}^{\text{tw}}$ - induction of self with target weight tw, q_0 - bioenergy constant, $H_{\text{SprtA}}^{\text{tw}}$ - tension of self with target weight tw, $D_{\text{SprtA}}^{\text{tw}}$ - activation induction with target weight tw, j - activation density.

Here we mean $(*_H)$, $(**_H)$ in the neuron action space, where e.g., basic actions are designated as $x_1, x_2, \dots, x_n$.

Let us denote the neuron current strength as I_{SprtA} .

Let us denote the neuron current voltage as U_{SprtA} .

Let us denote the resistance to neuron current as R_{SprtA} .

The field of the given structure tw is used for the activation of networks. The field can remain in effect until it is executed tw. Here all stages of the structure tw can be executed directly in parallel, in particular, an algorithm for solving the desired problem. We will call this field the operational activation field. This field will be created according to the structure tw.

The structure of the attosecond pulses of light, that a short-pulse laser to generate, or the microwave alternating current is close to

$$(a \uparrow I \downarrow^a) \uparrow I \downarrow (\downarrow_a I^{\uparrow a}) = \text{Sprt} \begin{matrix} \text{Sprt}^{\text{Sprt}^{\downarrow}} \\ \text{Sprt}^{\text{Sprt}^{\downarrow}} \\ \text{Sprt}^{\text{Sprt}^{\downarrow}} \\ \text{Sprt}^{\text{Sprt}^{\downarrow}} \\ \text{Sprt}^{\text{Sprt}^{\downarrow}} \\ \text{Sprt}^{\text{Sprt}^{\downarrow}} \\ \text{Sprt}^{\text{Sprt}^{\downarrow}} \\ \text{Sprt}^{\text{Sprt}^{\downarrow}} \end{matrix} (**_H), \text{ i.e., type } {}^a\text{Sprt}_a^a, \text{ and upon}$$

activation it will be induction of same type self, which is necessary for the formation of a local assembly point d_r of external energy fibers El^{d_r} . Its locality

(assembly point position r) will be determined by the location of microwave generation through Targetblock of S_{mnSprt} [2], [10]. Execution tw will be achieved through setting the assemblage point in the desired position r_1 to engage the appropriate external energy: $Sprt_{d_r^r Sprt_{r_1}^{d_r}}^{d_r^r Sprt_{r_1}^{d_r}}$.

To gain access to object transformation, just go to the level $IS = \frac{2}{\pi} \arctg(1 + \varepsilon)$, ε may be quite small.

Examples of transformation:

- 1) $Sprt_q^q \rightarrow Sprt_{st_q^q}^{st_q^q} \rightarrow Sprt_{Sprt_c^b}^{Sprt_c^b}$
- 2) $Sprt_q^q \rightarrow S_3f(self(q)) \rightarrow Sprt_r^r$

Remark H.1. This is a rather conditional interpretation, because in fact, the IS of the “vessel” (energy cocoon) of the object may turn out to be greater than $\frac{2}{\pi} \arctg(1)$. This is taken for initiation: we build a theory of this, starting from this stage of interpretation. After experiments, the next stage may begin.

Different types of neural networks: based on living structure

- 1) tree
- 2) insect
- 3) virus
- 4) etc

The energy of the container itself $\{A\}$, abstracted from its contents $\{A\}$:

$$Sprt_{\{A\}}^{\{A\}}, \{A\} Sprt_{\{A\}}^{\{A\}}$$

Mathematics, theoretical science and programming are actually concerned with structuring their spaces and the capacities within these spaces. Experimental science and experimental programming are concerned with the contents in real spaces and the capacities in those spaces. Ordinary level programming is a self-theoretical science, self-programming is programming oneself, in particular programming environments, self²-programming is, in particular, programming the

creation of programming environments etc. (oself-self) -science - this is work S_{mnSprt} in activation mode, for example, through the attosecond pulses of light, that a short-pulse laser to generate, or the microwave current; the same applies to practical (experimental) programming with real objects and energies [2], [10].

Implicitness is a sign of a mathematical cocoon of a mathematical equation (problem). The “form” of implicitness is the mathematical cocoon (structure).

Appendix 1

Remark H.1. Mathematics allows you to deeply find and formulate the concept of singular points in the Universe based on the levels of more subtle energies. The experiments of the 2022 Nobel laureates Asle Ahlen, John Clauser, Anton Zeilinger and the experiments in chemistry Nazhipa Valitov correspond to the concept of the Universe as a capacity in itself as the element. They experimented with connections for elements of the microworld, and since here the connections are self-connections, then when the object component of self-connections is removed, its higher level remains, which was manifested in their experiments. The electron spin belongs to the second level - above the level of objectivity. The energy of self-containment in itself is closed on itself.

Remark H.2. For example, using the attosecond pulses of light, that a short-pulse laser to generate, or the microwave alternating current to “turn on” a program operator $Q\uparrow\downarrow R$ by the corresponding eprogram, where Q is a normal neurons action, and R is activation of S_{mnSprt} with tW g , should give g as a result of execution. A program operator $Q\uparrow\downarrow R$ acts through the top level of structure type (1.1) for Q to the top level of structure type (1.1) for R . The entrance to the upper level S_{mnSprt} during activation mode (via the attosecond pulses of light, that a short-pulse laser to generate, or microwave with minimum amplitude and maximum frequency) can bring S_{mnSprt} to the edge of objectivity or even beyond it, using the corresponding eprograms with the necessary tW . In principle, changing the

singularity of the upper level by at least a sufficiently small amount through the identification structure of Maxwell's equations, S_{mnSpt} can go beyond objectivity depending on the change. It is important to organize “anchors” for returning to one or another position through the corresponding eprograms. When you begin to identify, you immediately reach the upper level, since the operation of identification is an operation of the upper level. The operator team must initiate (turn on) the desired output to the upper level, and then the included process is executed.

Remark H.4. In any structure, connection, task related to the search for the unknown, it is possible by identification to obtain for the upper level of this unknown a singularity that interprets the upper level corresponding to its real image.

Hypothesis H.3: The upper level is the identification of everything that is there, by definition.

The upper level of objectivity (i.e., the boundary of objectivity) is the identification of all objects. Then it is quite possible, through the use of dynamic programming in the required activation (Remark H.3. Hypothesis H.2: equations for real processes in a non-trivial form can be used to fully or partially interpret the self-level of the process, replacing the equal signs with identification signs, and solutions to these equations as a manifestation of this level on the level of objectivity and ordinary energies. That is, equations for real processes serve as a definition of the self-level of the process, the definition of self-values (self-characteristics) of the process through the identification sign, i.e., they are defined (expressed) through themselves. In particular, forms (1.1) - (1.4) can be used as forms of identification. Each such singularity creates its own field, the process, the object etc. Much more effective than science for working with these singularities will be special Dynamic programming, which we are currently working on to create. Let us denote the upper level of A by $\bar{A}$, the upper level of P by $\bar{P}$. Then the singularity $\bar{A} \rightarrow \bar{P}$ is a setting for the transformation of A to P through attosecond pulses of light that are

generated by a short-pulse laser or ultra-high frequency current with a minimum amplitude and maximum frequency to transfer S_{mnSprt} to the boundary of objectivity to perform the desired task, in particular. for the necessary transformation, the necessary action etc. The necessary manifestations of the upper level on the lower ones can be carried out through the appropriate settings in the upper level. It is clear that the upper-level singularity corresponds not only to one object, process, task, etc., but also to a whole class of those corresponding to this singularity. Therefore, in particular, a transition to other elements of this class is possible. Using the work of 2023 Nobel Prize winners in physics Ferenc Kraus and his colleagues Pierre Agostini and Anna Lhuillier, their experimental methods of generating attosecond pulses of light to study the dynamics of electrons in matter, we will attempt to carry out the experimental part of our work as soon as we receive a specially equipped laboratory and funding.

Remark H.5. Parallel operator $Sprt_{places\ for\ symbols}^{symbols}$ corresponds to theoretical science, parallel operator $Sprt_x^{objects}$ corresponds to technology, x – the space “point” (space place).

Remark H.6. Let self-energy of A looks like $Sprt_{C_A+\Delta C}^{C_A+\Delta C}=Sprt_{C_A}^{C_A}+Sprt_{C_A}^{\Delta C}+Sprt_{\Delta C}^{C_A}+Sprt_{\Delta C}^{\Delta C}$, $Sprt_{C_A}^{C_A}$ corresponds to objectivity of A: (G.3*), E_A - usual energy of A. ΔC determined from (G.3*) through C_A and then we can determine the complete self-energy of A.

Remark H.7. Even appropriate programming on ordinary computers allows you to create some types of self-energies, subtle energies of the upper level; and S_{mnSprt} is specifically designed to manipulate a much larger set of upper level of subtle energies through special programming and special scanning.

Remark H.8. Our object world is a manifestation of the space of self-capacities (the space of subtle energies), the space of self-capacities is a manifestation of the space of self-containments etc. The self-capacity is the element of the self-level,

but can partially manifest the corresponding parts of its content at lower levels, for example object level, in the corresponding “structures”.

Appendix 2

The self-space of a higher level contains many self-energetic fibers, collecting into appropriate sets that can be accessed by the corresponding self-spaces of lower levels. That's right, for example. This assembly point on the human cocoon can carry out this, in particular, access to our self-space with objects.

The energy of a living organism:

$$f(r, a, E_q) = Sprt_{t_0}^{\left\{ \begin{matrix} q(aSprt_a^a) \\ W_q Sprt_{q(aSprt_a^a)}^{E_q} \end{matrix} , Sprt_{d_r(E_{in}l^{d_r})}^{\{E^{ex}l^{d_r}\}} \right\}}$$

$aSprt_a^a$ -internal energy of a living organism, q- a gap in the energy cocoon of a living organism, r-the position of the assemblage point d_r on the energy cocoon of a living organism; W_q - energy prominences from the gap in the cocoon of a living organism, E_q -external energy entering the gap in the cocoon of a living organism, $E^{ex}l^{d_r}$ - a bundle of fibers of external energy self-capacities from outside the cocoon, collected at the point of assembly of the cocoon of a living organism at time t_0 , $E_{in}l^{d_r}$ - a bundle of fibers of external energy self-capacities from inside the cocoon, collected at the point of assembly of the cocoon of a living organism in the same position r of the assemblage point d_r . d_r is the subject of identifying the energy fibers of the subtle energy of the Universe in position r both outside and inside the cocoon. $r = St(t)_{D(t)}^{Q(t)}$, when $Q(t) = D(t) = \text{talk}$, $Sprt(t)_{D(t)}^{Q(t)}$ the usual position of the assemblage point is supported (fixed) with the help of internal conversation (internal dialogue with oneself (self-talk)), if $Q(t) \neq D(t)$ the position of the assemblage point changes and, when $Q(t) = D(t) \neq \text{talk}$ occurs, position of the assemblage point and is fixed in a new position;

$d_r = St_{E_{ba}}^{St_{E_{ba}}^{El} St_{E_{ba}}^{E_{ba}}}$, the cocoon of a living organism $E_{ba} = St_{aSt_a}^{aSt_a}$. Energy

with measure of fuzziness μ_1, μ_2 of a living organism of a person:

$$f(r, a, E_q) = Sprt_{t_0}^{\left\{ \begin{matrix} q(aSprt_a^a) \\ w_q Sprt_q^{E_q} \end{matrix} \right\}, Sprt_{dr}^{\{E^{exl}dr\}}_{(self(E_{inl}dr))}}$$

Remark H.9. In the World, at least, there are the following spaces: the space of self-capacities, which is usually called the energy space, and the space of self-containments, which is usually called spirits.

REFERENCES:

- 1) Danilishyn I.V. Danilishyn.(April 28,2023) O.V. CONTINUAL SIT-ELEMENTS AND DYNAMICAL SIT-ELEMENTS. Theoretical and practical aspectsof modern scientific research:Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the IIInternational Scientific and Practical Conference, Seoul. Seoul-Vinnytsia: Case Co., Ltd.& European Scientific Platform, pp.144-150. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/10>
- 2) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: dynamic sets, dynamic operators and their applications: monograph / ed. by Pasyukov V. – Shawnee, USA: Primedia eLaunch LLC. – 2024. – 442 p. ISBN 979-8-89217-806-8
DOI: <https://doi.org/10.36074/itdmdsdoata.monograph-2024>
https://books.google.com.ua/books/about?id=R3Y6EQAAQBAJ&redir_esc=y
<https://explore.openaire.eu/search/publication?pid=10.36074%2Fitdmdsdoata.monograph-2024>
- 3) Oleksandr Danilishyn, Illia Danilishyn. Introduction to Dynamic Sets Theory: Sprrt-elements and Their Applications to the Physics and Chemistry. JOURNAL OF PHYSICS AND CHEMISTRY, vol. 2, issue 3, pp.1-31, March 27, 2024 https://cskscientificpress.com/articles_file/527-article1712797848.pdf

- 4) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. [Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1). 2023-08-01
<https://doi.org/10.21203/rs.3.rs-3217178/v1> Dynamic Sets Theory: Sit-elements and Their Applications | Research Square
[www.researchsquare.com/article/rs-3217178/v1](https://doi.org/10.21203/rs.3.rs-3217178/v1)
- 5) I. Danilishyn, O. Danilishyn Program Operators Sit, tS, S1e, Set1. Journal of Sensor Networks and Data Communications [ISSN: 2994-6433] 2023, Volume 3 | Issue 1 | 138-143, <https://www.opastpublishers.com/table-contents/jsndc-volume-3-issue-1-year-2023>
- 6) Oleksandr Danilishyn, Illia Danilishyn. Dynamical Sets Theory: S2t-Elements and Their Applications. J Math Techniques Comput Math, 2(12), 2023, 479-498. <https://www.opastpublishers.com/open-access-articles/dynamical-sets-theory-s2telements-and-their-applications.pdf>
- 7) Danilishyn I., Danilishyn O. Dynamic Sets Set and Some of Their Applications to Neuroscience, Networks Set New Advances in Brain & Critical Care 2023, Volume 4 | Issue 2 | 66- 81.
<https://doi.org/10.33140/NABCC.04.02.02>
- 8) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets S1et and Some of their Applications in Physics. Science Set Journal of Physics, 25 September 2023,1-11, 815-_ article1695707465.pdf (mkscienceset.com)
- 9) I. Danilishyn, O. Danilishyn Dynamic Sets Se, Networks Se. Advances in Neurology and Neuroscience, 2023, Volume 6 | Issue 2 | 278-294.
<https://www.opastpublishers.com/open-access-articles/dynamic-sets-se-networks-se.pdf>
- 10) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>

- 11) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the V International Scientific and Practical Conference, Oxford, June 23, 2023. Oxford-Vinnytsia: P.C. Publishing House & European Scientific Platform, 2023. Theoretical and empirical scientific research: concept and trends. pp.156-161, DOI 10.36074/logos-23.06.2023.42, <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/12>
- 12) Danilishyn I.V. Danilishyn O.V. SET1 – ELEMENTS. INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 28 June, 2023 with the roceedings of the:Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 9 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC International Centre Corporative Management (Vienna, Austria), pp. 239-254. <https://archive.journal-grail.science/index.php/2710-3056/issue/view/09.06.2023>
- 13) Danilishyn I., Danilishyn O. VARIABLE HIERARCHICAL DYNAMICAL STRUCTURES (MODELS) FOR DYNAMIC, SINGULAR, HIERARCHICAL SETS AND THE PROBLEM OF COLD THERMONUCLEAR FUSION. The driving force of science and trends in its development: collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, July 14, 2023. Coventry, United Kingdom: European Scientific. Pp.113-119. Platform.<https://previous.scientia.report/index.php/archive/issue/view/14.07.2023>
- 14) Oleksandr Danilishyn and Illia Danilishyn. Fuzzy Dynamic Fuzzy Sets. Variable Fuzzy Hierarchical Dynamic Fuzzy Structures (Models, Operators) for Dynamic, Singular, Hierarchical Fuzzy Sets. FUZZY PROGRAM OPERATORS ffSprt, fftprS, ffS1epr, ffSeprt1. Journal of

Declarations:

Availability of data and material

- 1) Danilishyn I.V. Danilishyn.(April 28,2023) O.V. CONTINUAL SIT-ELEMENTS AND DYNAMICAL SIT-ELEMENTS. Theoretical and practical aspectsof modern scientific research:Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the IIInternational Scientific and Practical Conference, Seoul. Seoul-Vinnytsia: Case Co., Ltd.& European Scientific Platform, pp.144-150. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/10>
- 2) Oleksandr Danilishyn, Illia Danilishyn. Introduction to Dynamic Sets Theory: Sprt-elements and Their Applications to the Fhysics and Chemistry. JOURNAL OF PHYSICS AND CHEMISTRY, vol. 2, issue 3, pp.1-31, March 27, 2024 https://cskscientificpress.com/articles_file/527-_article1712797848.pdf
- 3) Galushkin A. Networks: principles of the theory. Hot line-Telecom. M.,2010 (in Russian).
- 4) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. [Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1). 2023-08-01 <https://doi.org/10.21203/rs.3.rs-3217178/v1> [Dynamic Sets Theory: Sit-elements and Their Applications | Research Square](https://doi.org/10.21203/rs.3.rs-3217178/v1)
[www.researchsquare.com/article/rs-3217178/v1](https://doi.org/10.21203/rs.3.rs-3217178/v1)
- 5) I. Danilishyn, O. Danilishyn Program Operators Sit, tS, S1e, Set1. Journal of Sensor Networks and Data Communications [ISSN: 2994-6433] 2023, Volume 3 | Issue 1 | 138-143, <https://www.opastpublishers.com/table-contents/jsndc-volume-3-issue-1-year-2023>
- 6) Oleksandr Danilishyn, Illia Danilishyn. Dynamical Sets Theory: S2t-Elements and Their Applications. J Math Techniques Comput Math, 2(12), 2023, 479-

498. <https://www.opastpublishers.com/open-access-articles/dynamical-sets-theory-s2telements-and-their-applications.pdf>
- 7) Danilishyn I., Danilishyn O. Dynamic Sets Set and Some of Their Applications to Neuroscience, Networks Set New Advances in Brain & Critical Care 2023, Volume 4 | Issue 2 | 66- 81. <https://doi.org/10.33140/NABCC.04.02.02>
 - 8) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets S1et and Some of their Applications in Physics. Science Set Journal of Physics, 25 September 2023,1-11, 815-_ article1695707465.pdf (mkscienceset.com)
 - 9) I. Danilishyn, O. Danilishyn Dynamic Sets Se, Networks Se. Advances in Neurology and Neuroscience, 2023, Volume 6 | Issue 2 | 278-294. <https://www.opastpublishers.com/open-access-articles/dynamic-sets-se-networks-se.pdf>
 - 10) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
 - 11) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the V International Scientific and Practical Conference, Oxford, June 23, 2023. Oxford-Vinnytsia: P.C. Publishing House & European Scientific Platform, 2023.Theoretical and empirical scientific research: concept and trends. **pp.156-161**, DOI 10.36074/logos-23.06.2023.42, <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/12>
 - 12) Danilishyn I.V. Danilishyn O.V. SET1 – ELEMENTS. INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 28 June, 2023 with the roceedings of the:Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 9 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC

International Centre Corporate Management (Vienna, Austria), pp. 239-254.
<https://archive.journal-grail.science/index.php/2710-3056/issue/view/09.06.2023>

- 13) Danilishyn I., Danilishyn O. VARIABLE HIERARCHICAL DYNAMICAL STRUCTURES (MODELS) FOR DYNAMIC, SINGULAR, HIERARCHICAL SETS AND THE PROBLEM OF COLD THERMONUCLEAR FUSION. The driving force of science and trends in its development: collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, July 14, 2023. Coventry, United Kingdom: European Scientific. Pp.113-119. Platform.<https://previous.scientia.report/index.php/archive/issue/view/14.07.2023>
- 14) Oleksandr Danilishyn and Illia Danilishyn. Fuzzy Dynamic Fuzzy Sets. Variable Fuzzy Hierarchical Dynamic Fuzzy Structures (Models, Operators) for Dynamic, Singular, Hierarchical Fuzzy Sets. FUZZY PROGRAM OPERATORS ffSprt, fftprS, ffS1epr, ffSeprt1. Journal of Mathematical Techniques Computational Mathematics (JMTCM), Volume 3 | Issue 4 |2024, Apr 17, pp. 1 – 37, *Journal DOI: 10.33140/JMTCM*
<https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press>

Competing interest

There are no competing interests. All sections of the article are executed jointly.

Funding

There were no sources of funding for writing the article.

Authors' contributions

The contribution of the authors is the same, we will not separate.

Supplement 1: S^1t – program operators

For simultaneous multiplication: $S^1t_x^{\{ag*\}}$: enter the notation of the set B with elements $b_{i_1 i_2 \dots i_n} = (S^1t_x^{\{g_{1i_1}^*, g_{2i_2}^*, \dots, g_{ni_n}^*\}})_R$ for any $\{i_1, i_2, \dots, i_n\}$ without repetitions, $x = S^1t_g^{\{K\}}$, K-set of any $\{k_1^*, k_2^*, \dots, k_n^*\}$ without repetitions of their, k_i -any digit, $i=1, 2, \dots, n$, $R = S^1t_g^{\{i_1^+, i_2^+, \dots, i_n^+\}}$, R is the index of the lower discharge (we choose an index on the scale of discharges): Table I.1.

Then $S^1t_x^{\{B+\}}$ gives the final result of simultaneous multiplication. Any system of calculus can be chosen, in particular binary. The simplest functional scheme of the assumed arithmetic-logical device for S^1t -multiplication:

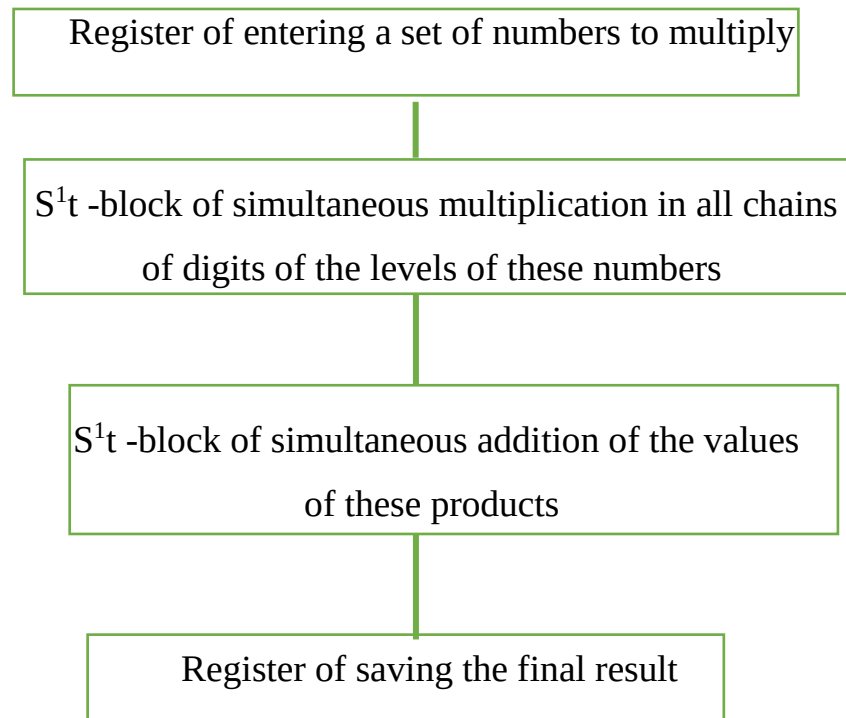


Fig. 4.10. The straightforward functional scheme of the assumed arithmetic-logical device for S^1t -multiplication.

Remark 4.12.0. This process can be interpreted through the simultaneous addition of elements in a triangular matrix, which can be represented as a

composite matrix, the elements of the first row of which are the matrices of simultaneous multiplication of the digits of the first element of the list by the digits of the rest, taking into account the bit depth, the elements of the second row of which are the matrices of simultaneous multiplication of the digits of the second element list into the remaining numbers, starting from the third element of the list, taking into account the bit depth, etc.

Here are some of the S^1t programming operators.

1. Simultaneous assignment of the expressions $\{p\} = (p_1, p_2, \dots, p_n)$ to the variables $\{g\} = (g_1, g_2, \dots, g_n)$. It's implemented through $S^1t_x^{\{\{g\}:\{p\}\}}$.
2. Simultaneous check the set of conditions $\{f\} = (f_1, f_2, \dots, f_n)$ for a set of expressions $\{B\} = (B_1, B_2, \dots, B_n)$. It's implemented through $S^1t_x^{IF\{\{B\}\{f\}\} \text{ then } Q}$ where Q can be any.
3. Similarly, for loop operators and others.

S_3^1f – software operators will differ only in the aggregates $\{g\}, \{p\}, \{B\}, \{f\}$ will be formed from corresponding S^1t program operators in form (1.1) or form from the forms (1.1.1) - (1.4).

Quite interesting is the OS (operating system), the principles and modes of operation of the computer for this programming. But this is already the material of the next publications.

The ideology of dynamical S^1t and $S_3^1f(t)$ can be used for programming:

1. The process of simultaneous assignment of the expressions $\{p\} = (p_1, p_2, \dots, p_n)$ to the variables $\{g\} = (g_1, g_2, \dots, g_n)$ is implemented through $S^1t(t)_x^{\{\{g\}:\{p\}\}}$.
2. The process of simultaneous check the set of conditions $\{f(t)\} = (f(t)_1, f(t)_2, \dots, f(t)_n)$ for a set of expressions $\{B(t)\} = (B_1(t), B_2(t), \dots, B_n(t))$ is implemented through $S^1t(t)_x^{IF\{\{B(t)\}\{f(t)\}\} \text{ then } Q(t)}$ where Q(t) can be any.
3. Similarly, for loop operators and others.

$S^1_3 f(t)$ – software operators will differ only in these aggregates

$\{g\}, \{p\}, \{B(t)\}, \{f(t)\}$ will be formed from corresponding processes $S^1t(t)$ for above mentioned programming operators through form (1.1) or form from the forms (1.1.1) - (1.4).

The usage of S^1t -elements for networks.

The all-encompassing monograph of Galushkin A. [40] embraces all aspects of networks but usual traditional approaches to networks are through classical mathematics, in particular through usual conformity operators. Here we consider another approach – through new mathematics partition with containment operators, which though may be interpreted as a result of some conformity operators, but themselves are no conformity operators. The containment operators are more convenient for networks. Also, main lay stress on the processors use, which work with triodes use, that does not use in S^1t -networks in mainly. S^1t -networks is represented by S^1t -structure, which may be constructed for necessary weights. S^1t -OS (S^1t -operating system) are used S^1t -coding and S^1t -translation. In the first the coding is realized through 2-measured matrix –row (a, b), where the number b – the code of the action, the number a- the object code of this action. S^1t -coding (or s^1elf -coding) is realized through the matrix, which has 2 columns (in continuous case- 2 numbers intervals). Here initial coding is used for all matrix rows simultaneously. S^1t -translation is realized by the inversion. In this case s^1elf -coding and s^1elf -translation will be more stable in particular. The target weights f_i in $S^1t_a^{\{fx\}}$ are chosen for necessary tasks. We will touch no questions of the applications, optimization of networks. They are detailed by A. Galushkin [40]. We touch difference of it for complex networks hierarchy only. The same simple executing programs are in the cores of simple artificial neurons of type S^1t (designation - mnS^1t) for simple information processing. More complex executing programs are used for mnS^1t nodes. Unfortunately, we change name S^1t -elements to S^1t -elements because we find that S^1t -elements were used by other authors early. S^1t -threshold element $-\text{sgn}(S^1t_b^{\{gx\}})$, b- mnS^1t , $x=(x_1, x_2, \dots, x_n)$ – source

signals values, $\{g\} = (g_1, g_2, \dots, g_n)$ – S^1t -synapses weights. The first level of mnS^1t consists from simple mnS^1t . The second level of mnS^1t consists from $S^1t_D^{\{mnS^2t\}}$ – S^1t -node of mnS^1t in range D , D - capacity for mnS^1t node. The third level of mnS^1t consists from $S^1t_D^{\{S^1t_D^{\{mnS^1t\}}\}}$ – S^1t^1 - node of mnS^1t in range D , thus D becomes capacity in its¹elf for mnS^1t . The usage of S^1t^1 - nodes of mnS^1t is enough for our networks, but s¹elf level is more higher in living organisms, in particular S^1t^n -, $n \geq 3$. Target structure or corresponding eprogram by corresponding s¹elf -code enters to target block by means of alternating current. After that here takes place the activation of all networks or its part according to indicative target. May arise the opinion, that we go out from networks ideology, but in fact networks presents complex hierarchy with capacity in its¹elf of different levels in living organisms.

Remark 4.12.2. Traditional scientific approaches through classical mathematics allows to describe only on usual energy level. Here is approach- on more thin energy level.

In mnS^1t are $S^1t_{mnS^1t}^{\{eprograms\}}$, eprogram –executing program in S^1t - OS. In this connection S^1t -OS (or S^1t -elf-OS) is based on S^1t -assembly language (or S^1t -elf - assembly language), which is based on assembly language through S^1t -approach in turn in the case of the sufficiency of the S^1t -networks elements base. The eprograms are in S^1t -programming environments (or S^1t -elf - programming environments), but this question and S^1t -networks base will be considered in next articles. In particular, eprograms may contain S^1t - programming operators. In mnS^1t cores the constant memory S^1t with correspondent eprograms depending on mnS^1t .

The ideology of S^1t and S^1t_3f can be used for programming. Here are some of the Quite interesting is the OS (operating system), the principles and modes of operation of the S^1t -networks for this programming. But this is already the material of the next publications.

Here is based on the elements of S^1t – physics and special neural networks with artificial neurons operating in normal and S^1t – modes, a model of a helicopter without a main and tail rotors was developed. Let's denote this model through S^1mnS^1t . To do this, it's proposed to use mnS^1t of different levels, for example, for the usual mode, mnS^1t serves for the initial processing of signals and the transfer of information to the second level, etc. to the nodal center, then checked and in case of anomaly - local S^1t – mode with the desired "target weight" is realized in this section, etc. to the center. Here, in case of anomaly during the test, S^1mnS^1t is activated with the desired "target weight". Here are realized other tasks also. To reach the self-energy level, the mode $S^1t_{S^1mnS^1t}^{S^1mnS^1t}$ is used. In normal mode, it's planned to carry out the movement of $SmnS^1t$ on jet propulsion with the conversion of the energy of the emitted gases into a vortex, to obtain additional thrust upwards. For this purpose, a spiral-shaped chute (with "pockets") is arranged at the bottom of the S^1mnS^1t for the gases emitted by the jet engine, which first exit through a straight chute connected to the spiral one. There is a drainage of exhaust gases outside the S^1mnS^1t . Otherwise, $SmnS^1t$ is represented by a neural network that extends from the center of one of the main clusters of S^1t - artificial neurons to the shell, turning on into the shell itself. Above the operator's cabin is the central core of the neural network and the target block, which is responsible for performing the "target weights" and auxiliary blocks, the functions and roles of which we will discuss further. Next is the space for the movement of the local vortex. The unit responsible for $Smnst$'s actions is located below the operators' cab. In S^1t – mode the entire network or its sections are S^1t – activated to perform certain tasks, in particular, with "target weights". In target block are used S^1t - coding, S^1t -translation for activation all networks to "target weights" simultaneously, then –the reset of this S^1t -coding after activation. Unfortunately, triodes are not suitable for S^1t -neural network. In the most primitive case usual separators with corresponding resistances and core for eprograms may be used instead triodes since there is not necessity in the unbending of the alternating current to direct. The belt of S^1t -memory operative is disposed around central core

of S^{1t} . There are S^{1t} -coding, S^{1t} -translation, S^{1t} -realize of eprograms and of the programs from the archives without extraction theirs. For S^{1t} -coding and S^{1t} -translation may be use high-intensity, ultra-short optical pulses laser of Nobel laureates of 2018 year Gerard Mourou, Donna, Strickland.

S^{1t} – structure or a eprogram if one is present of needed «target weight» are taken in target block at S^{1t} – activation of the networks. $S^{1t}_{activation}^{S^{1t},f}$ derives $S_{mns^{1t}}$ to the self level boundary with target weight f .

It's used an alternating current of above high frequently and ultra-violet light, which are able to work with S^{1t} – structures in S^{1t} – modes by it's nature for an activation of the networks or some of its parts in S^{1t} – modes and at local using S^{1t} – mode. Above high frequently alternating current go through mercury bearers that overheating does not occur. The power of the alternating current of above high frequently increase considerably for target block. The activation of all network is realized to indicative “target weights”.

Conclusions. The monograph task: to understand hierarchy of energies in the Universe and the principles of functioning of living energy (living organism, in particular, human, subtle energies), and then using these principles to "construct" artificial living energies (let's call them pseudo-living energies). It is possible to significantly expand the horizons of science, in particular physics, by studying the subtle energies in the Universe. Considering object A as the value of dynamic variable P, we can reassign $P := B$ at the top level and instead of A we will have object B, and also perform any other parallel dynamic operations with objects. That is, our world will be perceived as a Dynamic Programming environment. Dynamic programming is an extension of science for "working" with the top and middle levels. Here information is transformed into actions. Energy Internet: a) transmission of necessary singularities (subtle energies) (connection to necessary singularities (subtle energies)), b) transmission of necessary self-structures (subtle energies) (connection to necessary self-structures) etc. It is also necessary to construct a bioSmnsprt on living, grown elements of the central nervous system, in particular, neurons. It is also possible to construct another version of bioSmnsprt, using a viral model, on living, grown bacteriophages. We have simply outlined some aspects; it is too early to develop them without experimental study. All problems have a solution, the only question is: a) in complete identification with the problem, b) the "breadth of the field" of values that you can provide. Our mathematics is unusual for a mathematician, because here the fulcrum is the action, and not the result of an action, as in classical mathematics. Therefore, our mathematics is adapted not only to obtain results, but also to directly control actions, which will certainly show its advantages on a fundamentally new type of neural networks with directly parallel calculations, for which it was created. Any action has much greater potential than its result. The significance of the monograph: in a new qualitatively different approach to the study of complex processes through new mathematical hierarchical parallel dynamic structures, in particular those processes that are studied by synergetics. Our approach is not based on deterministic equations that generate self-organization, which is very difficult to study and gives very small results for a very limited class of problems and does not provide the most important thing - the structure of self-organization. We simply start from the assumed structure of self-organization, since we are interested not so much in the numerical calculation of this, but in the structure of self-organization itself, its formation (construction) for the necessary purposes and its management. Although we are also interested in numerical calculations. Laureates of the Nobel Prize in Physics 2023 Ferenc Kraus and his colleagues Pierre Agostini and Anne L'Huillier used a short-pulse laser to generate attosecond pulses of light to study the fuzzy dynamics of electrons in matter. According to our Synthesis-Type Singularity Theory, its action corresponds to a singularity, allowing the upper level of subtle energies to be reached to manipulate the lower

levels. In April 2023, we proposed to use a short-pulse laser to achieve desired goals using a directly parallel neural network. We then proposed a fundamental development of this directly parallel neural network. The significance of our monograph lies in the formation of the proposed mathematical structure of subtle energies, this is done for the first time in science, and the proposed classification of mathematical structures of subtle energies for the first time. The experiments of 2022 Nobel Prize winners Asle Allen, John Clauser, Anton Zeilinger and the chemistry experiments of Nazhipa Valitov eloquently demonstrate, that we are right and that these studies are necessary. Be that as it may, we have created classes of new mathematical structures, new fuzzy mathematical singularities, i.e. we have contributed to the development of mathematics. The articles and scientific monographs [40], [42] dedicated to the mathematical foundations of constructing pseudo-living energy. New concepts and new processing methods of information based on them and new software operators were introduced. We have a fairly nonclassical section of mathematics, so the presentation of the material is a bit non-classical to emphasize this. Classical science reaches the horizon and beyond the horizon through chance, while we immediately start from the horizon. We have found a way to construct some mathematical class of complex processes with which classical science can work through chance at best. Classical science proceeds from the objective components, while we immediately start from the wave ones, bypassing the objective ones. Further development is associated with changing the structure of the arithmetic-logical device, the corresponding software and application for new technologies, in the light of the new approach. If you transform through self, i.e. through the middle level, using the reserve that is formed in the self-object by definition, the self-object can "catch" on to another object and as a result form a new self-object. For example, a simplest chemical reaction, when one simplest inorganic self-object is obtained from two inorganic simplest self-objects and, as a rule, energy is released. Next come cyclic organic molecules with increasing degree of self ("folding"). The simplest organic self-object benzene with delocalized orbitals: 3 bonding orbitals $\uparrow$ I $\downarrow$ 3 antibonding orbitals. Next come DNA and RNA with a complex multi-cyclic helical structure with a significantly increasing degree of self-organization ("folding"). If you transform through ||| (identification), i.e. through the upper level, then this can usually be done through replacement: "replacing" what is available with what is needed. Also, through the upper level (|||), you can go to the virtual double of the self-object (in fact, the self-virtual object from this one) to perform unusual actions, as well as perform other manipulations, in particular, the creation of new singularities of the upper level. By gaining the necessary energy, the self-object can turn into self-action, generating a singularity of the upper level. We consider the usual (sequential) hierarchy for levels: the 0th level corresponds to usual energies, the next higher lower level corresponds to objects, the next higher middle

level corresponds to self and finally the upper level corresponds to $|||$. This is the most simplified representation (simplified 2-interpretation). Of course, we can also consider more complex representations with greater possibilities, in particular, through more complex hierarchies: parallel hierarchies, self-hierarchies, $|||$ -hierarchies, etc. In particular, one can use the hierarchy $A = |||B$ (identification of the hierarchies set B). From it one can make all sorts of self-hierarchies by (1.1) – (1.4). It is possible to introduce and use $|||$ - science (science of synthesis) by Smnsprt, in particular, $|||$ - mathematics (mathematics of synthesis), - self-science, in particular, self-mathematics. This is of particular interest for biology, in particular, work with “virtual” components of living organisms, DNA, RNA, etc., and in general with “virtual” components of science and technology. We have simply outlined some aspects; it is too early to develop them without experimental study. All problems have a solution, the only question is: a) in complete identification with the problem, b) the “breadth of the field” of values that you can provide. Considering object A as the value of dynamic variable P, we can reassign $P := B$ at the top level and instead of A we will have object B, and also perform any other parallel dynamic operations with objects. That is, our world will be perceived as a Dynamic Programming environment. Dynamic programming is an extension of science for “working” with the top and middle levels. Here information is transformed into actions. Energy Internet: a) transmission of necessary singularities (subtle energies) (connection to necessary singularities (subtle energies)), b) transmission of necessary self-structures (subtle energies) (connection to necessary self-structures) etc. It is also necessary to construct a bioSmnsprt on living, grown elements of the central nervous system, in particular, neurons. It is also possible to construct another version of bioSmnsprt, using a viral model, on living, grown bacteriophages. Dynamic programming works in the space of energies with energies. It is enough to imagine the world as a software environment. Then objects and processes will act as variables etc. The subjects of variables are processes of a higher level manipulating them. Dynamic programming will do what is needed if the level and volume of energies allow, i.e., much more effectively than science will allow, as well as our supposed devices on Dynamic programming.

REFERENCES:

- 1) Illia Danilishyn, Oleksandr Danilishyn. Hierarchical dynamic mathematical structures (models) theory: Scientific monograph DOI <https://doi.org/10.36074/hdmsmt.monograph-2024> Published 2024-08- 28 Shawnee, USA Primedia eLaunch LLC 2024 <https://bookwire.bowker.com> : 9798892178099
- 2) Illia Danilishyn, Oleksandr Danilishyn. Introduction to Dynamic Mathematics: dynamic sets, dynamic operators and their applications: monograph / ed. by Pasyukov V. – Shawnee, USA : Primedia eLaunch LLC. – 2024. – 442 p. ISBN 979-8-89217-806-8
DOI: <https://doi.org/10.36074/itdmdsdoata.monograph-2024>
https://books.google.com.ua/books/about?id=R3Y6EQAAQBAJ&redir_esc=y
<https://explore.openaire.eu/search/publication?pid=10.36074%2Fitdmdsdoata.monograph-2024>
- 3) Danilishyn I.V. Danilishyn O.V. THE USAGE OF SIT-ELEMENTS FOR NETWORKS. IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>
- 4) Danilishyn I.V. Danilishyn O.V. MATHEMATICS ST, PROGRAMMING OPERATORS ST AND SOME EMPLOYMENT. _III International Scientific and Theoretical Conference «Theoretical and practical scientific achievements: research and results of their implementation»07.04.2023, Pisa, Italia. Collection of scientific papers "SCIENTIA", 2023.
<https://previous.scientia.report/index.php/archive/issue/view/07.04.2023>
- 5) Danilishyn I.V. Danilishyn O.V. DYNAMICAL SIT-ELEMENTS . IV International Scientific and Practical Conference "GRUNDLAGEN DER MODERNEN WISSENSCHAFTLICHEN FORSCHUNG ", 31.03.2023/Zurich, Switzerland. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/9>

- 6) Danilishyn I.V. Danilishyn O.V. tS – ELEMENTS. Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the V International Scientific and Practical Conference, Oxford, June 23, 2023. Oxford-Vinnytsia: P.C. Publishing House & European Scientific Platform, 2023. Theoretical and empirical scientific research: concept and trends. pp.156-161, DOI 10.36074/logos-23.06.2023.42, <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/12>
- 7) Danilishyn I.V. Danilishyn O.V. SET1 – ELEMENTS. INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 28 June, 2023 with the proceedings of the: Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 9 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC International Centre Corporative Management (Vienna, Austria), pp. 239-254. <https://archive.journal-grail.science/index.php/2710-3056/issue/view/09.06.2023>
- 8) Oleksandr Danilishyn, Illia Danilishyn. Dynamical Sets Theory: S2t-Elements and Their Applications. J Math Techniques Comput Math, 2(12), 2023, 479-498. <https://www.opastpublishers.com/open-access-articles/dynamical-sets-theory-s2telements-and-their-applications.pdf>
- 9) Danilishyn I., Danilishyn O. Dynamic Sets Set and Some of Their Applications to Neuroscience, Networks Set New Advances in Brain & Critical Care 2023, Volume 4 | Issue 2 | 66- 81. <https://doi.org/10.33140/NABCC.04.02.02>
- 10) Oleksandr Danilishyn and Illia Danilishyn. Dynamic Sets S1et and Some of their Applications in Physics. Science Set Journal of Physics, 25 September 2023, 1-11, 815- _ article1695707465.pdf (mkscienceset.com)
- 11) I. Danilishyn, O. Danilishyn Dynamic Sets Se, Networks Se. Advances in Neurology and Neuroscience, 2023, Volume 6 | Issue 2 | 278-294. <https://www.opastpublishers.com/open-access-articles/dynamic-sets-se-networks-se.pdf>

- 12) I. Danilishyn, O. Danilishyn Program Operators Sit, tS, S1e, Set1. Journal of Sensor Networks and Data Communications [ISSN: 2994-6433] 2023, Volume 3 | Issue 1 | 138-143, <https://www.opastpublishers.com/table-contents/jsndc-volume-3-issue-1-year-2023>
- 13) Danilishyn I., Danilishyn O. DYNAMIC SETS THEORY: SIT-ELEMENTS AND THEIR APPLICATIONS. Preprint. **Research Square**. МЕТОД СТАТТЯ опублікована 2023-08-01 04:20:16 <https://doi.org/10.21203/rs.3.rs-3217178/v1>
[Dynamic Sets Theory: Sit-elements and Their Applications | Research Square](https://www.researchsquare.com/article/rs-3217178/v1)
www.researchsquare.com/article/rs-3217178/v1
- 14) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO SETS THEORY AND OTHERS. Scientific practice: modern and classical research methods: Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the IV International Scientific and Practical Conference, Boston, May 26, 2023. Boston-Vinnytsia: Primedia eLaunch & European Scientific Platform, 2023, pp.166-171. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/11>
- 15) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO CONTINUALVALUED LOGIC AND OTHERS. Features of the development of modern science in the pandemic's era: collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, May 19, 2023. Berlin, Federal Republic of Germany: European Scientific Platform, pp.79-84.
<https://previous.scientia.report/index.php/archive/issue/view/19.05.2023>
- 16) Danilishyn I.V. Danilishyn O.V. SOME APPLICATIONS OF SIT-ELEMENTS TO SETS THEORYРозвиток сучасної науки: актуальні питання теорії та практики: матеріали III Всеукраїнської студентської наукової конференції, м. Харків, 19 травня, 2023 рік / ГО «Молодіжна наукова ліга». — Вінниця: ГО «Європейська наукова платформа», 2023, pp.270-272. <https://archive.liga.science/index.php/conference-proceedings/issue/view/ukr-19.05.2023>.

- 17) Danilishyn I.V. Danilishyn O.V. CONTINUAL SIT-ELEMENTS AND DYNAMICAL SIT-ELEMENTS. Theoretical and practical aspects of modern scientific research: Collection of scientific papers «ΛΟΓΟΣ» with Proceedings of the III International Scientific and Practical Conference, Seoul, April 28, 2023. Seoul-Vinnytsia: Case Co., Ltd. & European Scientific Platform, pp.144-150. <https://archive.logos-science.com/index.php/conference-proceedings/issue/view/10>
- 18) Danilishyn I.V. Danilishyn O.V. SIT-DYNAMICAL AND CONTINUAL ELEMENTS. МАТЕРІАЛИ VIII МІЖНАРОДНОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ ПРІОРИТЕТНІ НАПРЯМИ ДОСЛІДЖЕНЬ В НАУКОВІЙ ТА ОСВІТНІЙ ДІЯЛЬНОСТІ 29 – 30 квітня 2023 року Львів 2023, pp.24-31. [Збірник.pdf \(lviv-forum.inf.ua\)](http://zbirnik.lviv-forum.inf.ua)
- 19) Danilishyn I.V. Danilishyn O.V. CONTINUAL SIT-ELEMENTS WITH TARGET WEIGHTS. Формування сучасної науки: методика та практика: матеріали III Всеукраїнської студентської наукової конференції, м.Ужгород, 21 квітня, 2023 рік / ГО «Молодіжна наукова ліга». — Вінниця: ГО «Європейська наукова платформа», 2023 pp.105-107. <https://archive.liga.science/index.php/conference-proceedings/issue/view/ukr-21.04.2023>.
- 20) Danilishyn I.V. Danilishyn O.V. SIT-ELEMENTS AND DYNAMICAL SIT-ELEMENTS FOR CONTINUAL SETS. Modernization of science and its influence on global processes: collection of scientific papers «SCIENTIA» with Proceedings of the III International Scientific and Theoretical Conference, April 14, 2023. Bern, Swiss Confederation: European Scientific Platform. Pp.61-66. <https://previous.scientia.report/index.php/archive/issue/view/14.04.2023>
- 21) Danilishyn I.V. Danilishyn O.V. SIT-ELEMENTS AND THEIR APPLICATIONS. ЛЬВІВСЬКИЙ НАУКОВИЙ ФОРУМ МАТЕРІАЛИ VIII МІЖНАРОДНОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ ТЕОРІЯ І ПРАКТИКА СУЧАСНОЇ НАУКИ ТА ОСВІТИ 19 – 20 березня 2023 року Львів 2023, pp.33-38. [Збірник.pdf \(lviv-forum.inf.ua\)](http://zbirnik.lviv-forum.inf.ua)

- 22) Danilishyn I.V. Danilishyn O.V. SIT-NETWORKS. ЛЬВІВСЬКИЙ НАУКОВИЙ ФОРУМ МАТЕРІАЛИ VIII МІЖНАРОДНОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ ТЕОРІЯ І ПРАКТИКА СУЧАСНОЇ НАУКИ ТА ОСВІТИ 19 – 20 березня 2023 року Львів 2023, pp.39- 41.
[Збірник.pdf \(lviv-forum.inf.ua\)](#)
- 23) Danilishyn I.V. Danilishyn O.V. MATHEMATICS ST, PROGRAMMING OPERATORS STAND SOME EMPLOYMENT Комплексний підхід до модернізації науки: методи, моделі та мультидисциплінарність: матеріали ІІ Міжнародної наукової конференції, м.Луцьк, 3 березня, 2023р. / Міжнародний центр наукових досліджень. — Вінниця: Європейська наукова платформа, 2023, р.р. 114-119.
<https://archive.mcnd.org.ua/index.php/conference-proceeding/issue/view/03.03.2023>
- 24) Danilishyn I.V. Danilishyn O.V. ST –ELEMENTS APPLICATIONS FOR SOME TASKS. INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 24 February, 2023 with the proceedings of the:Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 17 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC International Centre Corporative Management (Vienna, Austria), pp.400-402. <https://archive.journal-grail.science/index.php/2710-3056/issue/view/17.02.2023>
- 25) Danilishyn I.V. Danilishyn O.V. THE INTRODUCTORY CONCEPTS AND OPERATIONS OF ST MATHEMATICS. INTERNATIONAL SCIENTIFIC JOURNAL GRAIL OF SCIENCE № 24 February, 2023 with the proceedings of the:Correspondence International Scientific and Practical Conference SCIENCE IN MOTION: CLASSIC AND MODERN TOOLS AND METHODS IN SCIENTIFIC INVESTIGATIONS held on June 17 th, 2023 by NGO European Scientific Platform (Vinnytsia, Ukraine) LLC International Centre Corporative Management (Vienna, Austria), pp.403- 406 .

<https://archive.journal-grail.science/index.php/2710-3056/issue/view/17.02.2023>

- 26) Danilishyn I.V. Danilishyn O.V. SOME St – ELEMENTS APPLICATIONS. Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 74): матеріали Міжнародної наукової інтернет-конференції, (м. Тернопіль, Україна – м. Переворськ, Польща, 6-7 лютого 2023 р.) / [редкол. : О. Патряк та ін.] ; ГО “Наукова спільнота”; WSSG w Przeworsku. – Тернопіль, pp. 4-7.
<http://www.konferenciaonline.org.ua/ua/article/id-947/>
- 27) Danilishyn I.V. Danilishyn O.V. MATHEMATICS SIT, PROGRAMMING OPERATORS SIT AND SOME APPLICATIONS. Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 74): матеріали Міжнародної наукової інтернет-конференції, (м. Тернопіль, Україна – м. Переворськ, Польща, 6-7 березня 2023 р.) / [редкол. : О. Патряк та ін.] ; ГО “Наукова спільнота”; WSSG w Przeworsku. – Тернопіль, pp. 4-7. <http://www.konferenciaonline.org.ua/ua/article/id-1040/>
- 28) Danilishyn I.V. Danilishyn O.V. THE NETWORKS SIT. Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 75): матеріали Міжнародної наукової інтернет-конференції, (м. Тернопіль, Україна – м. Переворськ, Польща, 3-4 квітня 2023 р.) / [редкол. : О. Патряк та ін.] ; ГО “Наукова спільнота”; WSSG w Przeworsku. – Тернопіль, pp. 6-9. <http://www.konferenciaonline.org.ua/ua/article/id-1060/>
- 29) Danilishyn I.V. Danilishyn O.V. MATHEMATICS ST, PROGRAMMING OPERATORS ST AND SOME EMPLOYMENT. Scientific forum: theory and practice of research: collection of scientific papers «SCIENTIA» with Proceedings of the III International Scientific and Theoretical Conference, March 10, 2023. Valencia, Kingdom of Spain: European Scientific Platform. Collection of scientific papers «SCIENTIA». pp.123-127
DOI:<https://doi.org/10.36074/scientia-10.03.2023>

- 30) Danilishyn I.V. Danilishyn O.V. ELEMENTS OF ST MATHEMATICS AND ST PROGRAMMING. ЛЬВІВСЬКИЙ НАУКОВИЙ ФОРУМ. МАТЕРІАЛИ VII МІЖНАРОДНОЇ НАУКОВО-ПРАКТИЧНОЇ КОНФЕРЕНЦІЇ ПРОБЛЕМИ ТА ПЕРСПЕКТИВИ СУЧАСНОЇ НАУКИ ТА ОСВІТИ 29-30 січня 2023 р., Львів, 2023, pp.40-43. [Збірник.pdf \(lviv-forum.inf.ua\)](#)
- 31) Danilishyn I.V. St – ELEMENTS AND PROGRAMMING OPERATORS. Інформаційне суспільство: технологічні, економічні та технічні аспекти становлення (випуск 74): матеріали Міжнародної наукової інтернет-конференції, (м. Тернопіль, Україна – м. Переворськ, Польща, 6-7 березня 2023 р.) / [редкол. : О. Патряк та ін.] ; ГО “Наукова спільнота”; WSSG w Przeworsku. – Тернопіль, pp.9-12. Internet address of the article on web-site: <http://www.konferenciaonline.org.ua/ua/article/id-917>
- 32) Danilishyn I., Danilishyn O. VARIABLE HIERARCHICAL DYNAMICAL STRUCTURES (MODELS) FOR DYNAMIC, SINGULAR, HIERARCHICAL SETS AND THE PROBLEM OF COLD THERMONUCLEAR FUSION. The driving force of science and trends in its development: collection of scientific papers «SCIENTIA» with Proceedings of the IV International Scientific and Theoretical Conference, July 14, 2023. Coventry, United Kingdom: European Scientific. Pp.113-119.
Platform.<https://previous.scientia.report/index.php/archive/issue/view/14.07.2023>
- 33) Danilishyn I., Danilishyn O. PROGRAM OPERATORS SIT, tS, S_{1e}, Set₁. Preprint. [Research Square](#). НАУЧНА СТАТТЯ Оpubлiковано: 2023-08-04 08:00:27
<https://www.researchsquare.com/article/rs-3228799/v1>
<https://doi.org/10.21203/rs.3.rs-3228799/v1>
- 34) Oleksandr Danilishyn and Illia Danilishyn. Fuzzy Dynamic Fuzzy Sets. Variable Fuzzy Hierarchical Dynamic Fuzzy Structures (Models, Operators) for Dynamic, Singular, Hierarchical Fuzzy Sets. FUZZY PROGRAM OPERATORS ffSprt, fftprS, ffS_{1e}pr, ffSeprt₁. Journal of Mathematical

Techniques Computational Mathematics (JMTCM), Volume 3 | Issue 4 |2024,
Apr 17, pp. 1 – 37

Journal DOI: 10.33140/JMTCM

<https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press>

- 35) Oleksandr Danilishyn, Illia Danilishyn. Introduction to Dynamic Sets Theory: Sprrt-elements and Their Applications to the Physics and Chemistry. JOURNAL OF PHYSICS AND CHEMISTRY, vol. 2, issue 3, pp.1-31, March 27, 2024 https://cskscientificpress.com/articles_file/527-article1712797848.pdf
- 36) Oleksandr Danilishyn and Illia Danilishyn. Introduction to Section of Dynamic Mathematics: Dynamic Sets Theory: Parallel Sprrt-Elements and Their Applications. J Math Techniques Comput Math (*Journal DOI: 10.33140/JMTCM*), Volume 3 | Issue 5 (2024), pp.1- 27. <https://www.opastpublishers.com/open-access-articles/introduction-to-section-of-dynamic-mathematics-dynamic-sets-theory-parallel-sprrtelements-and-their-applications.pdf>
- 37) Oleksandr Danilishyn and Illia Danilishyn. Introduction to Section of Dynamic Mathematics: Dynamic Fuzzy Sets Theory: Parallel Fuzzy Sprrt-Elements and Their Applications. J Math Techniques Comput Math (*Journal DOI: 10.33140/JMTCM*), Volume 3 | Issue 7 (2024), pp.1- 27. <https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press>
- 38) Oleksandr Danilishyn, Illia Danilishyn and Volodymyr Pasynkov. Intrtroduction to Section of Dynamic Mathematics: SCprt – elements and Their Applications to Physics. Sci. Set. J of Physics(2024) https://mkscienceset.com/articles_inpress/science-set-journal-of-physics
- 39) Illia Danilishyn, Oleksandr Danilishyn and Volodymyr Pasynkov. Some aspects of directly parallel operation of neural networks and their subtle energy

Advances in Machine& Artificial. Volume 5 | Issue 4 (2024), pp.1 - 7.

Intelligence<https://www.opastpublishers.com/journal/advances-in-machine-learning-artificial-intelligence/articles-in-press>

DOI: <https://doi.org/10.33140/AMLAI.05.04.02>.

- 40) Oleksandr Danilishyn, Illia Danilishyn and Volodymyr Pasynkov.

Intrtoduction to section of Dynamic mathematics: Theory of singularities of the type synthesizing. J Math Techniques Comput Math (*Journal*

DOI: 10.33140/JMTCM), Volume 3 | Issue 10 (2024), pp.1- 18.

<https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press>

- 41) Oleksandr Danilishyn, Illia Danilishyn and Volodymyr Pasynkov.

Introduction to Dynamic Operators: SDS-Elements, Self-type SDS -Structures and their Applications to PhysicsSci. Set. J of Physics(2024), pp.1-15

https://mkscienceset.com/articles_file/570- article1731913045.pdf

https://mkscienceset.com/articles_inpress/science-set-journal-of-physics

- 42) Illia Danilishyn, Oleksandr Danilishyn. Dynamic Sets Sprt and Some of

Their Applications to Biomedical Engineering and Biotechnology. World Congress on Biomedical Engineering and Biotechnology on the theme "Transforming Biomedical Engineering for Global Well-being". Kuala Lumpur, Malaysia, November 25-26, 2024

<https://www.timesscientificgroup.com/Biomed%20&%20Biotechnology%20Conference%20book.pdf>

- 43) Oleksandr Danilishyn, Illia Danilishyn and Volodymyr Pasynkov.

Introduction to Dynamic operators: Lprt-elements and Applications to physics and other Their Applications. J Math Techniques Comput Math (*Journal*

DOI: 10.33140/JMTCM), Volume 3 | Issue 11 (2025), pp.1- 16.

<https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press>

- 44) Oleksandr Danilishyn and Illia Danilishyn. Introduction to Dynamic

Operators: Rprt-Elements and Their Applications. Rprt-Networks. Variable Fuzzy Hierarchical Dynamic Fuzzy Structures (Models, Operators) for

Dynamic, Singular, Hierarchical Fuzzy Sets. Fuzzy Program Operators fRprt, ftprR, ffR1epr, ffReprt1. J Math Techniques Comput Math (*Journal DOI: 10.33140/JMTCM*), Volume 3 | Issue 9 (2024), pp.1- 26.
<https://www.opastpublishers.com/journal/journal-of-mathematical-techniques-and-computational-mathematics/articles-in-press>

SCIENTIFIC EDITION

Illia DANILISHYN
Oleksandr DANILISHYN

**MATHEMATICAL FUNDAMENTALS OF
CONSTRUCTING PSEUDO-LIVING ENERGY
THEORY AND DYNAMIC PROGRAMMING**

MONOGRAPH

Edited by Volodymyr Pasyenko

Date of publication: 28.03.2025.
The volume of data is 6.5 MB.
Conventionally printed sheets 22,92.

Publisher: Primedia E-launch LLC.
KS 5518, United States, Kansas, Shawnee, Flint St.
E-mail: info@primediaelaunch.com
URL: www.primediaelaunch.com

